



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

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
- ▶ ArrayAdapter
- ▶ 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 \(Mobile OS Market Share\)](#)

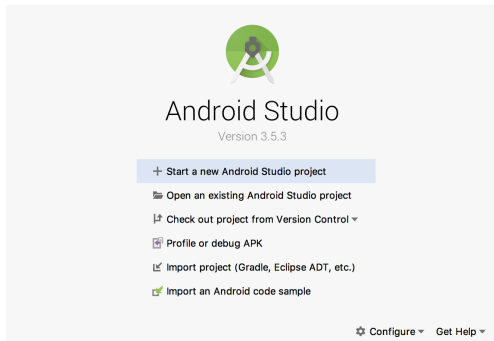
QEMU

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

KOTLIN

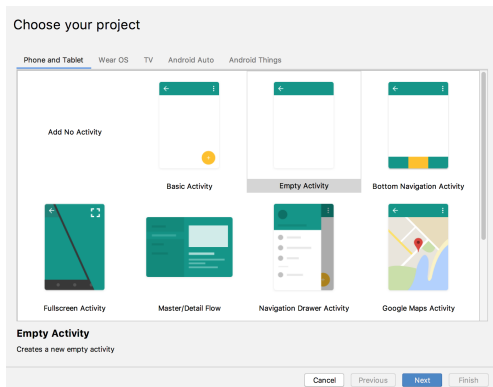
- ▶ Cross-platform
- ▶ Statically typed
- ▶ Type inference
- ▶ Interoperable with Java
- ▶ Preferred programming language for Android application developers
- ▶ One of my favourite programming languages

ANDROID STUDIO: APPLICATION SETUP



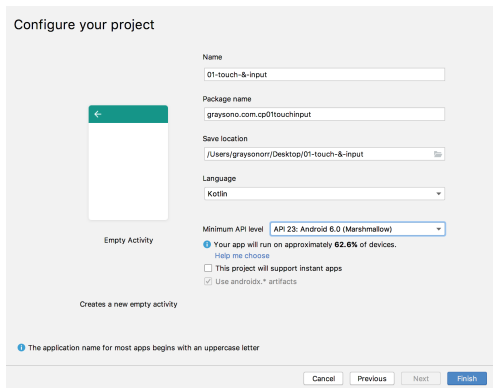
ANDROID STUDIO: APPLICATION SETUP

- ▶ 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



The screenshot shows the 'Configure your project' dialog in Android Studio. On the left, there is a preview of an 'Empty Activity' with a green header bar and a white body. Below the preview, it says 'Creates a new empty activity'. On the right, there are several input fields and a dropdown menu:

- Name:** 01-touch-&-input
- Package name:** graysono.com.cp01touchinput
- Save location:** /Users/graysonrr/Desktop/01-touch-&-input
- Language:** Kotlin (selected from a dropdown)
- Minimum API level:** API 23: Android 6.0 (Marshmallow) (selected from a dropdown)

Below the 'Minimum API level' dropdown, there is a blue information icon followed by the text: 'Your app will run on approximately 62.6% of devices.' Below this, there is a link 'Help me choose'. Further down, there are two checkboxes:

- ☐ This project will support instant apps
- ☒ Use androidx.* artifacts

At the bottom left, there is a blue information icon followed by the text: 'The application name for most apps begins with an uppercase letter'. At the bottom right, there are four buttons: 'Cancel', 'Previous', 'Next', and 'Finish'.

ANDROID STUDIO: APPLICATION SETUP

- ▶ Android platform version
- ▶ Cumulative distribution

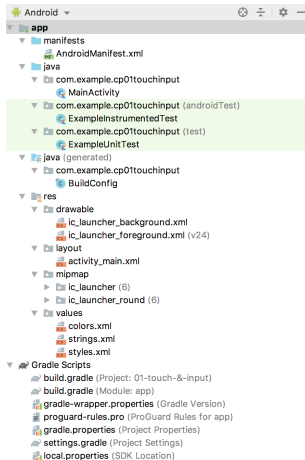
ANDROID PLATFORM VERSION	API LEVEL	CUMULATIVE DISTRIBUTION	Marshmallow	
4.0 Ice Cream Sandwich	15		Security Fingerprint Authentication Confirm Credential System App Linking Adoptable Storage Devices Multimedia 4K Display Mode Support for MIDI Create digital audio capture and playback objects APIs to associate audio and input devices List of all audio devices Updated video processing APIs Flashlight API Reprocessing Camera2 API Updated ImageWriter objects and ImageReader class User Input Voice Interactions Assist API Bluetooth Stylus Support	User Interface Themeable ColorStateLists Wireless & Connectivity Hotspot 2.0 Improved Bluetooth Low Energy Scanning Android for Work Controls for Corporate-Owned, Single-Use devices Silent install and uninstall of apps by Device Owner Silent enterprise certificate access Auto-acceptance of system updates Delegated certificate installation Data usage tracking Runtime permission management Work status notification
4.1 Jelly Bean	16	99.6%		
4.2 Jelly Bean	17	98.1%		
4.3 Jelly Bean	18	95.9%		
4.4 KitKat	19	95.3%		
5.0 Lollipop	21	85.0%		
5.1 Lollipop	22	80.2%		
6.0 Marshmallow	23	62.6%		
7.0 Nougat	24	37.1%		
7.1 Nougat	25	14.2%		
8.0 Oreo	26	6.0%		
8.1 Oreo	27	1.1%		

<https://developer.android.com/about/versions/marshmallow/android-6.0.html>

Cancel OK

ANDROID STUDIO: APPLICATION SETUP

► Directory structure



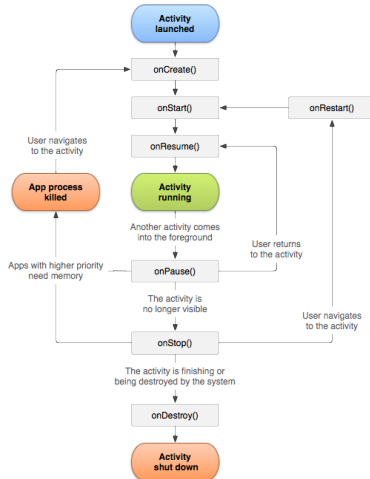
ANDROID STUDIO: MainActivity.KT

- ▶ AppCompatActivity
- ▶ Activity lifecycle - onCreate()

A screenshot of the Android Studio code editor showing the MainActivity.kt file. The code is written in Kotlin and defines the MainActivity class, which inherits from AppCompatActivity. It includes imports for AppCompatActivity and Bundle, and overrides the onCreate method to call super.onCreate and setContentView.

```
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

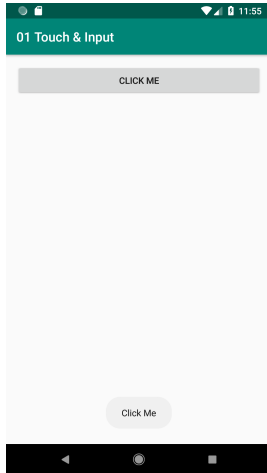
```
MainActivity.kt
1 package com.example.cp01touchinput
2
3 import androidx.appcompat.app.AppCompatActivity
4 import android.os.Bundle
5 import android.view.View
6 import android.widget.Toast
7 import kotlinx.android.synthetic.main.activity_main.*
8
9 class MainActivity : AppCompatActivity() {
10
11     override fun onCreate(savedInstanceState: Bundle?) {
12         super.onCreate(savedInstanceState)
13         setContentView(R.layout.activity_main)
14         btnClick.setOnClickListener(ButtonOnClickListener())
15     }
16
17     inner class ButtonOnClickListener : View.OnClickListener {
18         override fun onClick(view: View?) {
19             Toast.makeText(context: this@MainActivity, text: "Click Me", Toast.LENGTH_LONG).show()
20         }
21     }
22 }
```

ALTERNATIVE TO INNER CLASS

► Simplified

```
MainActivity.kt x
1  package com.example.cp01touchinput
2
3
4  import androidx.appcompat.app.AppCompatActivity
5  import android.os.Bundle
6  import android.view.View
7  import android.widget.Toast
8  import kotlinx.android.synthetic.main.activity_main.*
9
10 class MainActivity : AppCompatActivity() {
11
12     override fun onCreate(savedInstanceState: Bundle?) {
13         super.onCreate(savedInstanceState)
14         setContentView(R.layout.activity_main)
15         btnClick.setOnClickListener { it: View!
16             Toast.makeText(context: this@MainActivity, text: "Click Me", Toast.LENGTH_LONG).show()
17         })
18     }
19 }
```

EMULATOR - ONCLICKLISTENER



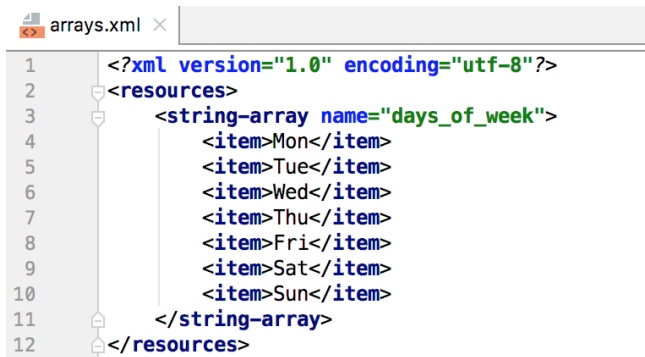
ARRAYADAPTER

► arrayOf - Kotlin

```
MainActivity.kt x
1 package com.example.cp01touchinput
2
3 import androidx.appcompat.app.AppCompatActivity
4 import android.os.Bundle
5 import android.widget.ArrayAdapter
6 import android.widget.Spinner
7 import kotlinx.android.synthetic.main.activity_main.*
8
9 class MainActivity : AppCompatActivity() {
10
11     override fun onCreate(savedInstanceState: Bundle?) {
12         super.onCreate(savedInstanceState)
13         setContentView(R.layout.activity_main)
14         val daysOfWeek = arrayOf("Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun")
15         populateSpinner(spnDaysOfWeek, daysOfWeek)
16     }
17
18     private fun populateSpinner(spinner: Spinner, array: Array<String>) {
19         val layoutID: Int = android.R.layout.simple_spinner_item
20         spinner.adapter = ArrayAdapter<String>(context: this@MainActivity, layoutID, array)
21     }
22 }
```

ALTERNATIVE TO ARRAYOF

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



The screenshot shows an XML editor window titled 'arrays.xml'. The XML code is as follows:

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <resources>
3   <string-array name="days_of_week">
4     <item>Mon</item>
5     <item>Tue</item>
6     <item>Wed</item>
7     <item>Thu</item>
8     <item>Fri</item>
9     <item>Sat</item>
10    <item>Sun</item>
11  </string-array>
12 </resources>
```

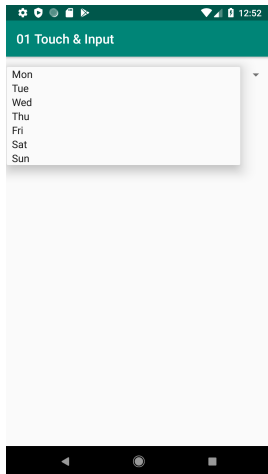
The editor includes a line number margin on the left, a tree view on the left side of the code area, and a search bar at the top right.

ALTERNATIVE TO ARRAYOF

- `resources.getStringArray(id: Int)`

```
MainActivity.kt x
1 package com.example.cp01touchinput
2
3 import androidx.appcompat.app.AppCompatActivity
4 import android.os.Bundle
5 import android.widget.ArrayAdapter
6 import android.widget.Spinner
7 import kotlinx.android.synthetic.main.activity_main.*
8
9 class MainActivity : AppCompatActivity() {
10
11     override fun onCreate(savedInstanceState: Bundle?) {
12         super.onCreate(savedInstanceState)
13         setContentView(R.layout.activity_main)
14         val daysOfWeek = resources.getStringArray(R.array.days_of_week)
15         populateSpinner(spnDaysOfWeek, daysOfWeek)
16     }
17
18     private fun populateSpinner(spinner: Spinner, array: Array<String>) {
19         val layoutID: Int = android.R.layout.simple_spinner_item
20         spinner.adapter = ArrayAdapter(context = this@MainActivity, layoutID, array)
21     }
22 }
```


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

A screenshot of an IDE showing the AndroidManifest.xml file. The file is opened in a tab titled 'AndroidManifest.xml'. The code is as follows:

```
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="@string/app_name"
9         android:roundIcon="@mipmap/ic_launcher_round"
10        android:supportRtl="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

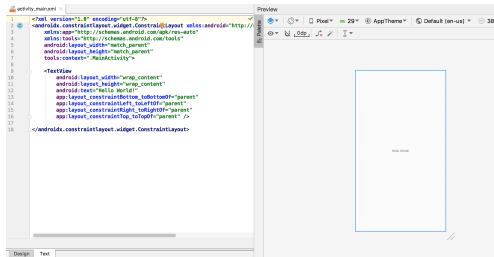
- ▶ General abstraction for something that can be drawn
- ▶ Type of resource retrieved for drawing things to the screen
- ▶ Generic API for dealing with an underlying visual resource
- ▶ Examples of drawables - bitmaps & vectors

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

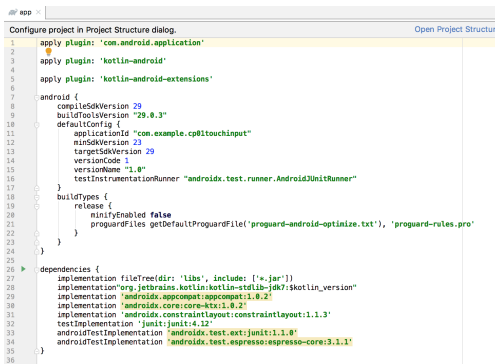
- ▶ XML files that contain simple values such as colors, strings & styles
- ▶ Each child of the `<resources>` element defines a single resource
- ▶ For example, a `<string>` element creates an `R.string` resource

ANDROID STUDIO: RESOURCE TYPES

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

ANDROID STUDIO: GRADLE

► build.gradle



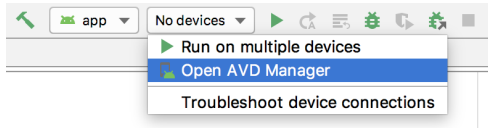
```
1  apply plugin: 'com.android.application'
2
3  apply plugin: 'kotlin-android'
4
5  apply plugin: 'kotlin-android-extensions'
6
7  android {
8      compileSdkVersion 29
9      buildToolsVersion "29.0.3"
10     defaultConfig {
11         applicationId "com.example.cp01touchinput"
12         minSdkVersion 23
13         targetSdkVersion 29
14         versionCode 1
15         versionName "1.0"
16         testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
17     }
18     buildTypes {
19         release {
20             minifyEnabled false
21             proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'
22         }
23     }
24 }
25
26 dependencies {
27     implementation fileTree(dir: 'libs', include: ['*.jar'])
28     implementation 'org.jetbrains.kotlin:kotlin-stdlib-jdk7:$kotlin_version'
29     implementation 'androidx.appcompat:appcompat:1.0.2'
30     implementation 'androidx.core:core-ktx:1.0.2'
31     implementation 'androidx.constraintlayout:constraintlayout:1.1.3'
32     testImplementation 'junit:junit:4.12'
33     androidTestImplementation 'androidx.test.ext:junit:1.1.0'
34     androidTestImplementation 'androidx.test.espresso:espresso-core:3.1.1'
35 }
36
```

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

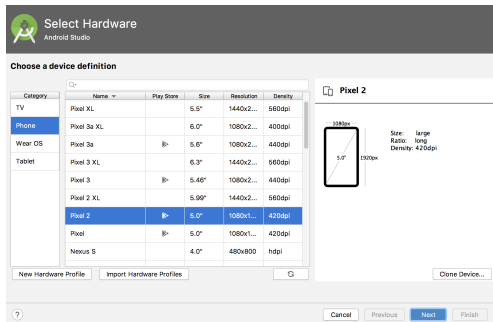
ANDROID STUDIO: EMULATOR

► AVD Manager



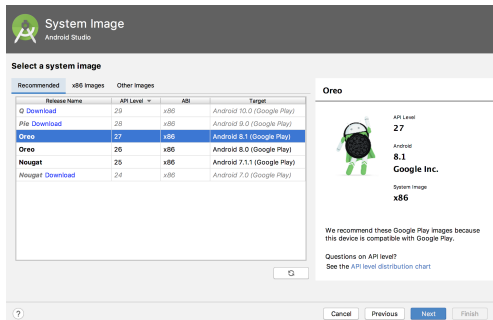
ANDROID STUDIO: EMULATOR

- ▶ Selecting hardware
 - ▶ Category
 - ▶ Name



ANDROID STUDIO: EMULATOR

- ▶ System Image
 - ▶ Release name
 - ▶ API level



The screenshot shows the 'System Image' selection window in Android Studio. The window has a title bar with the Android Studio logo and the text 'System Image' and 'Android Studio'. Below the title bar, there is a section titled 'Select a system image' with three tabs: 'Recommended', 'x86 Images', and 'Other images'. The 'Recommended' tab is active, showing a table of system images. The table has four columns: 'Release Name', 'API Level', 'ABI', and 'Target'. The 'Oreo' row with API level 27 and ABI x86 is highlighted. To the right of the table, there is a section titled 'Oreo' showing the API level 27, the Android logo, and the text '8.1 Google Inc.' and 'System Image x86'. Below this, there is a recommendation message: 'We recommend these Google Play images because this device is compatible with Google Play.' and a link to 'See the API level distribution chart'. At the bottom of the window, there are navigation buttons: 'Cancel', 'Previous', 'Next', and 'Finish'.

Release Name	API Level	ABI	Target
Q Download	29	x86	Android 10.0 (Google Play)
Pie Download	28	x86	Android 9.0 (Google Play)
Oreo	27	x86	Android 8.1 (Google Play)
Oreo	26	x86	Android 8.0 (Google Play)
Nougat	25	x86	Android 7.1.1 (Google Play)
Nougat Download	24	x86	Android 7.0 (Google Play)

Oreo

API Level
27

Android
8.1
Google Inc.

System Image
x86

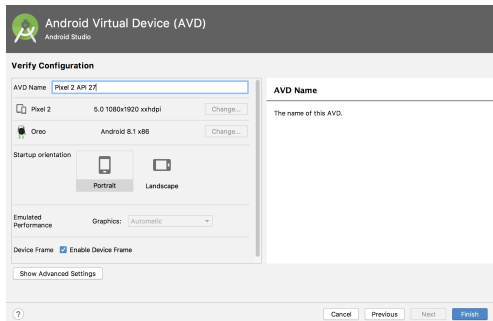
We recommend these Google Play images because this device is compatible with Google Play.

Questions on API level?
See the [API level distribution chart](#)

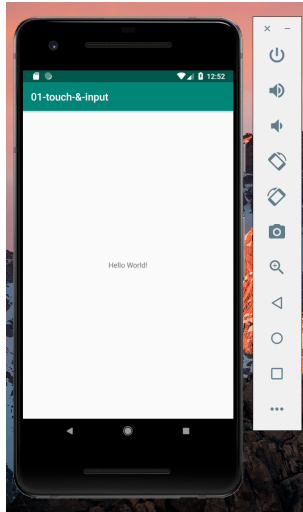
Cancel Previous **Next** Finish

ANDROID STUDIO: EMULATOR

- ▶ Verifying configuration
 - ▶ AVD name
 - ▶ Startup orientation



ANDROID STUDIO: EMULATOR



PRACTICAL

- ▶ Two tasks covering today's lecture
- ▶ Worth 1% of your final mark for the Design and Development of Applications for Mobile Devices course
- ▶ Deadline: Friday, 10 April at 5pm

ASSESSMENT 1 & 2

- ▶ Two assessments 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 four Android 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