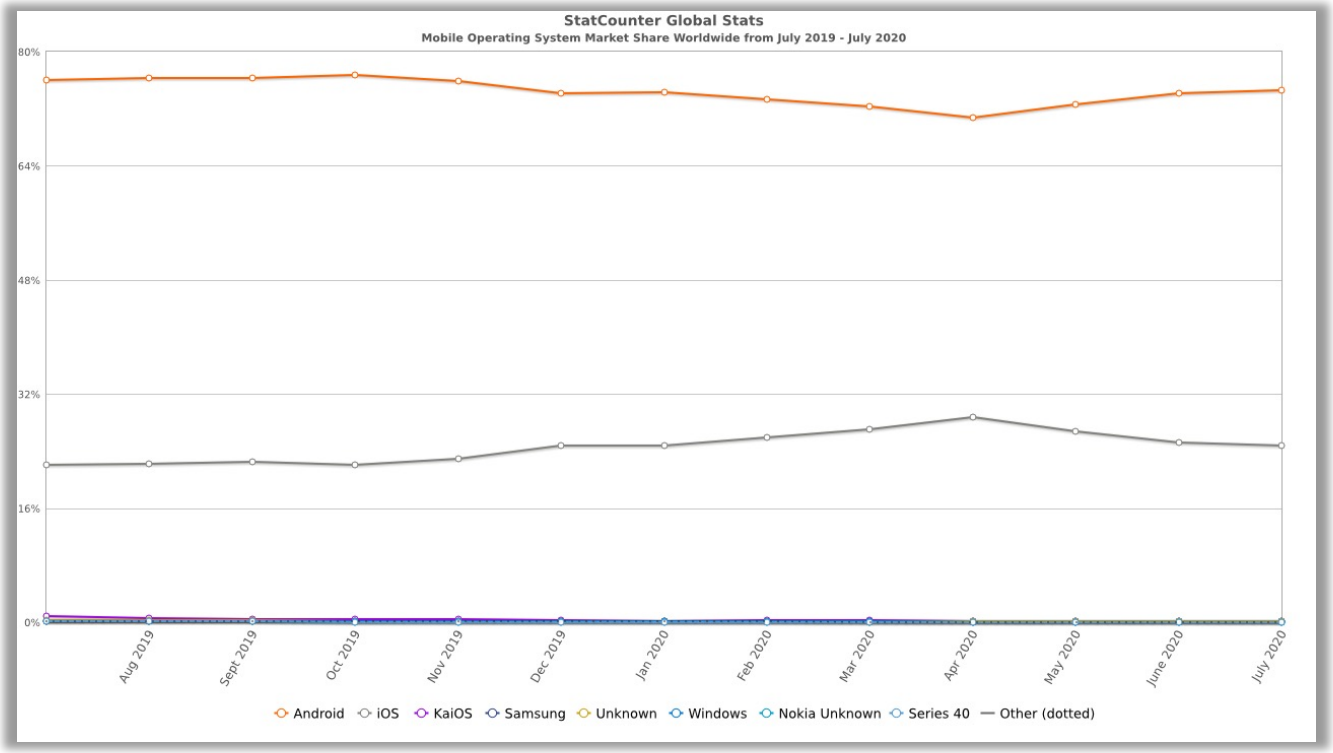
Lecture 4: Introduction to Android Development

Setting up, creating projects, testing apps

Jianjun Chen (Jianjun.Chen@xjtlu.edu.cn)

Smart Phone OS

- Android
- iOS
- Windows
- Blackberry
- Symbian
- WebOS



https://gs.statcounter.com/os-market-share/mobile/worldwide

Android Platform

- Android is a Linux-based operating system designed primarily for touchscreen mobile devices such as smartphones and tablet.
- Android is open source. It can be freely modified and distributed by device manufacturers, wireless carriers and enthusiast developers.
- Android was unveiled in 2007, the first Android phone was sold in October 2008

VPN

- It is recommended to get a VPN for this module.
- Some resources are easier to download with a VPN.

Android Studio and SDK

Download source, sdk versions.

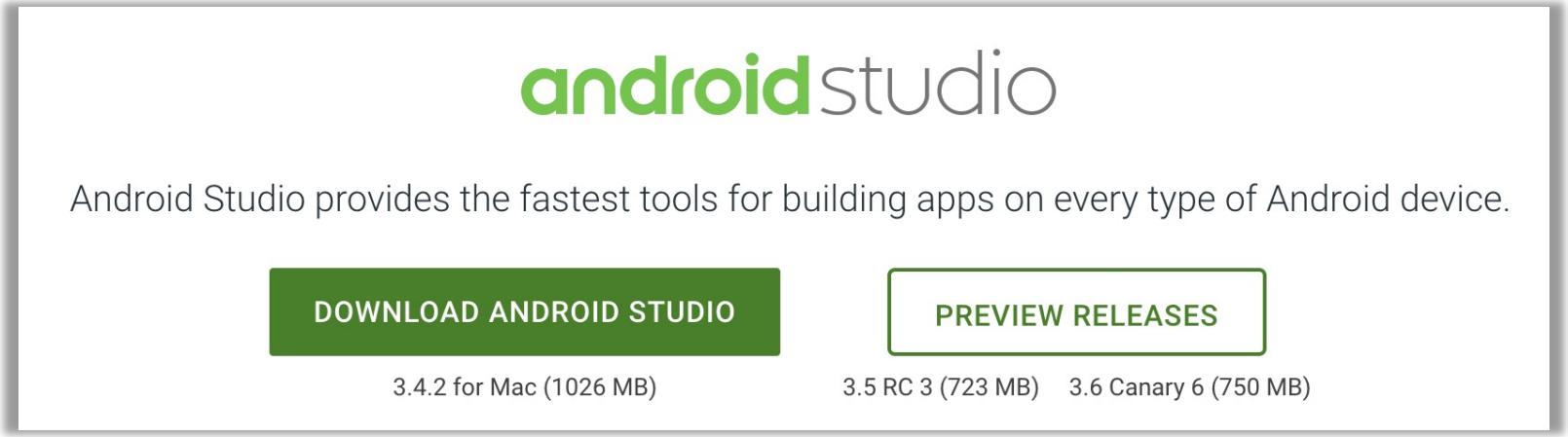
Setting up Things

- We will stick to using Android Studio (AS) throughout this module.
  - But Eclipse + ADT is an alternative option.
  - You must install JDK first.

- What will be covered in this lecture?
  - Android Studio set up.
  - The Android SDK.
  - The Gradle build system for AS.
  - A real or virtual Android device.

Getting Android Studio

- Official website:
  - https://developer.android.com/studio/
- Alternative website:
  - https://developer.android.google.cn/studio/



Android SDK and API level

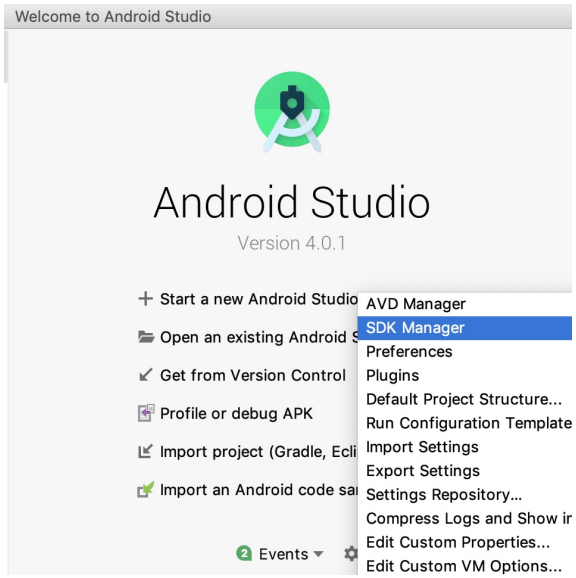
- The API decides how you call functions to access the service from the Android system. API is usually downward compatible.
  - E.g. Android 9 can run Android 4.X apps.
  - An app that requires higher version of Android API will run on less devices.
- SDK: A set of tools that can provide different API libraries and developer tools.

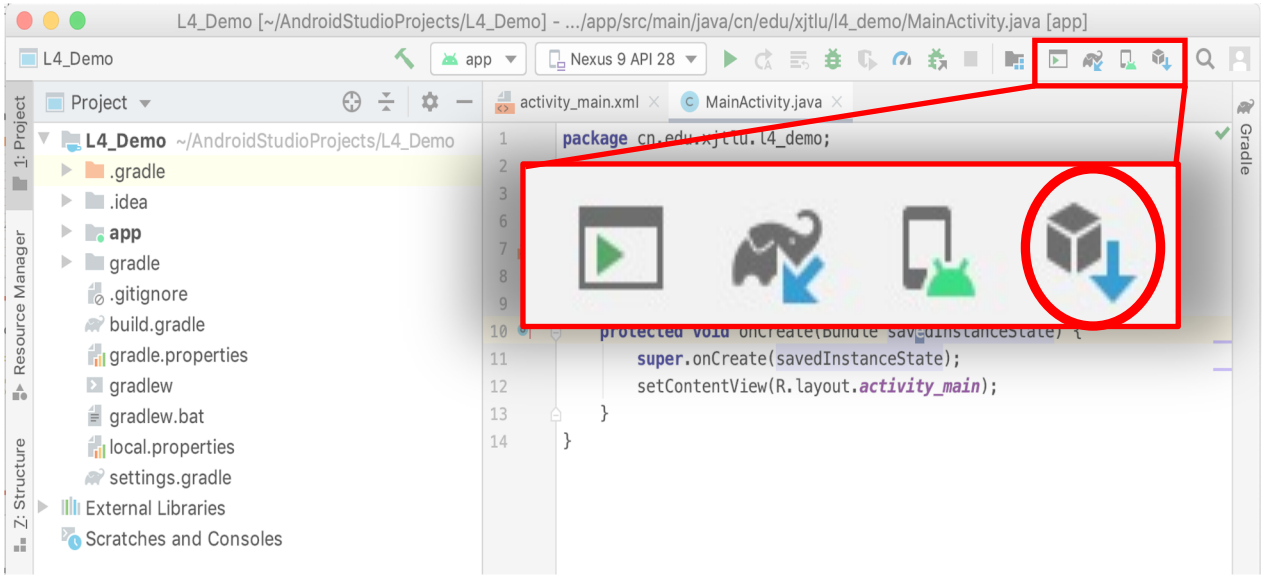
Android Versions and API Level

Codename	Version	API level/NDK release
Android10	10	API level 29
Pie	9	API level 28
Oreo	8.1.0	API level 27
Oreo	8.0.0	API level 26
Nougat	7.1	API level 25
Nougat	7.0	API level 24
Marshmallow	6.0	API level 23
Lollipop	5.1	API level 22
Lollipop	5.0	API level 21
KitKat	4.4 - 4.4.4	API level 19
Jelly Bean	4.3.x	API level 18
Jelly Bean	4.2.x	API level 17

Android SDK and API level

- AS does not come with SDKs pre-installed.
  - Access the SDK manager at the welcome screen
  - Menu->Tools->SDK manager
  - Also available as a button on the toolbar:

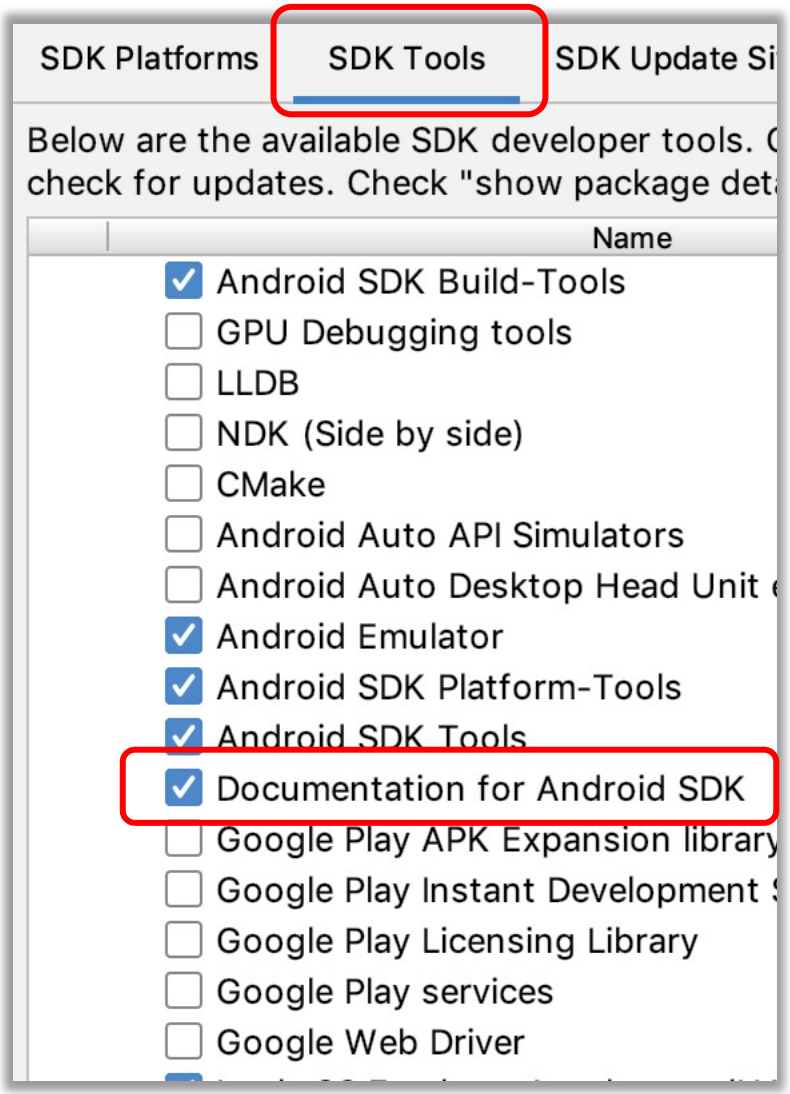




Documentation for SDK

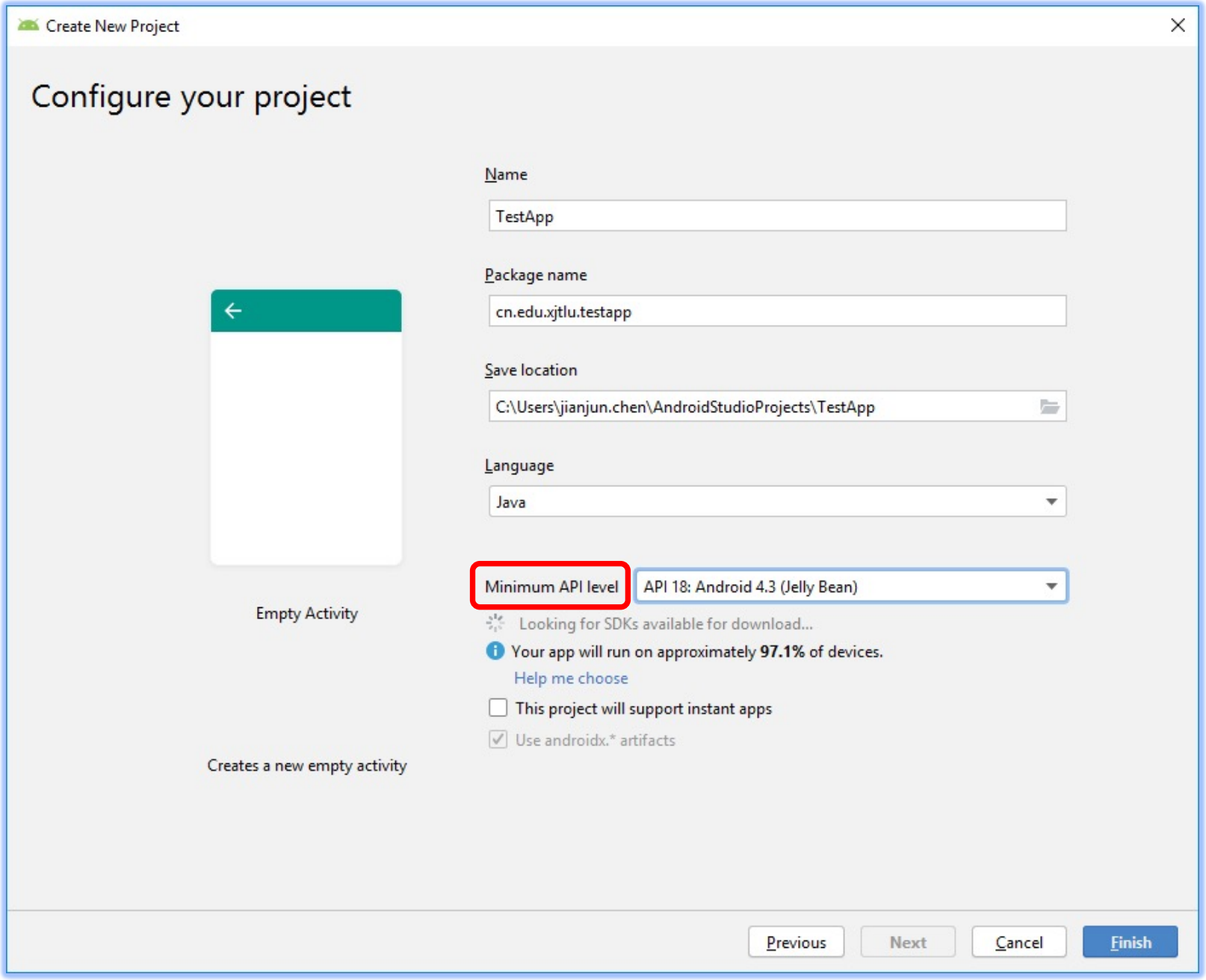
You might need to use VPN here if download fails

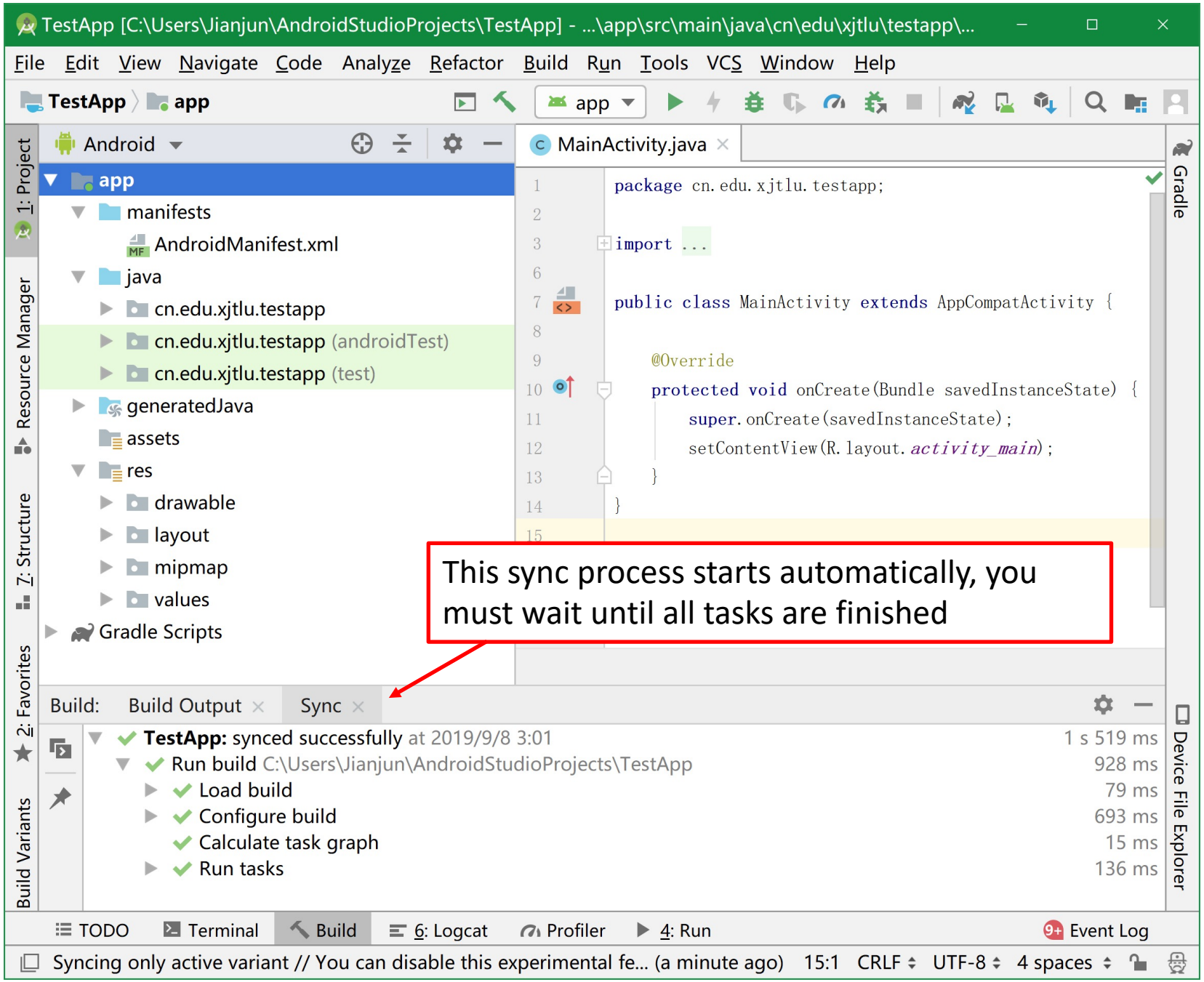
- While the SDK manager is still open, do remember to download the documentation for Android SDK
- This enables you to easily find definitions of functions and variables.
- Once done, you can check the documentation by selecting a function or variable identifier, then:
  - Windows: Ctrl + q
  - MacOS: F1
  - Linux: ??



Creating a Project

- Start your Android Studio
  - Start a new Android Studio project
  - Assign an application name and a project location.
  - Select an API version for your app
  - Select “Empty Activity”
  - Confirm the activity name.


Create New Project
Configure your project
Name
TestApp
Package name
cn.edu.xjtlu.testapp
Save location
C:\Users\jianjun.chen\AndroidStudioProjects\TestApp
Language
Java
Minimum API level
API 18: Android 4.3 (Jelly Bean)
Looking for SDKs available for download...
Your app will run on approximately 97.1% of devices.
Help me choose
This project will support instant apps
Use androidx.* artifacts
Empty Activity
Creates a new empty activity
Previous
Next
Cancel
Finish

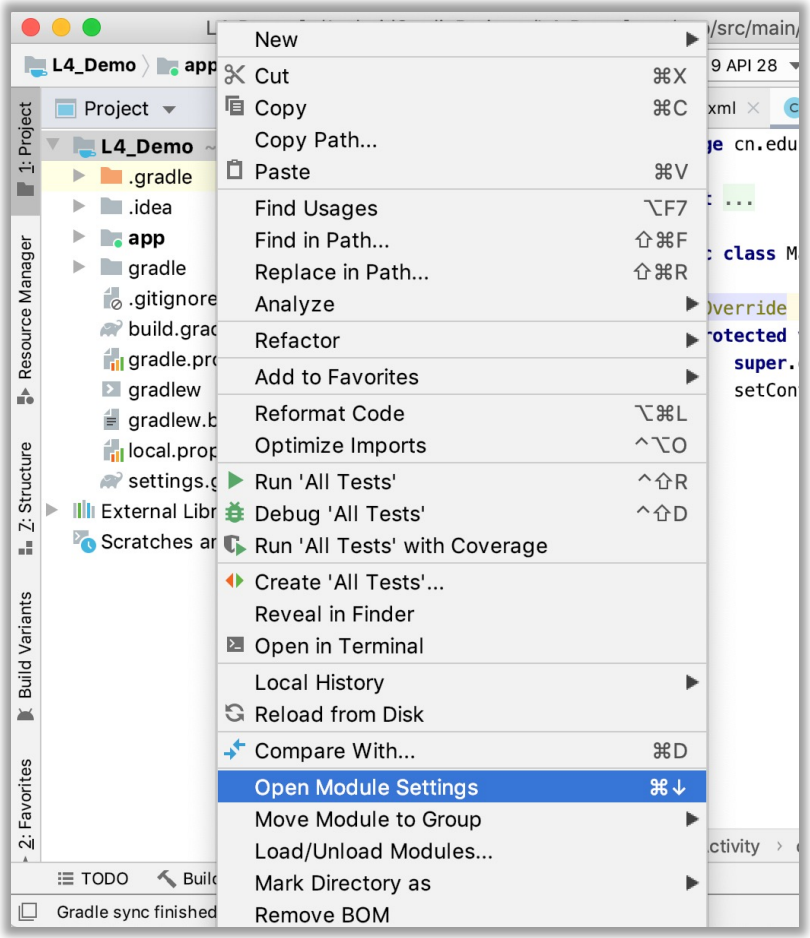

TestApp [C:\Users\Jianjun\AndroidStudioProjects\TestApp] - ...\app\src\main\java\cn\edu\xjtlu\testapp\...
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
TestApp app
app
Android
app
manifests
AndroidManifest.xml
java
cn.edu.xjtlu.testapp
cn.edu.xjtlu.testapp (androidTest)
cn.edu.xjtlu.testapp (test)
generatedJava
assets
res
drawable
layout
mipmap
values
Gradle Scripts
MainActivity.java
package cn.edu.xjtlu.testapp;
import ...
public class MainActivity extends AppCompatActivity {
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
}
}
This sync process starts automatically, you must wait until all tasks are finished
Build: Build Output Sync
TestApp: synced successfully at 2019/9/8 3:01 1 s 519 ms
Run build C:\Users\Jianjun\AndroidStudioProjects\TestApp 928 ms
Load build 79 ms
Configure build 693 ms
Calculate task graph 15 ms
Run tasks 136 ms
TODO Terminal Build 6: Logcat Profiler 4: Run Event Log
Syncing only active variant // You can disable this experimental fe... (a minute ago) 15:1 CRLF UTF-8 4 spaces
1: Project Resource Manager 7: Structure 2: Favorites Build Variants
Gradle Device File Explorer

Gradle Setup

Sync issues and solution without a VPN

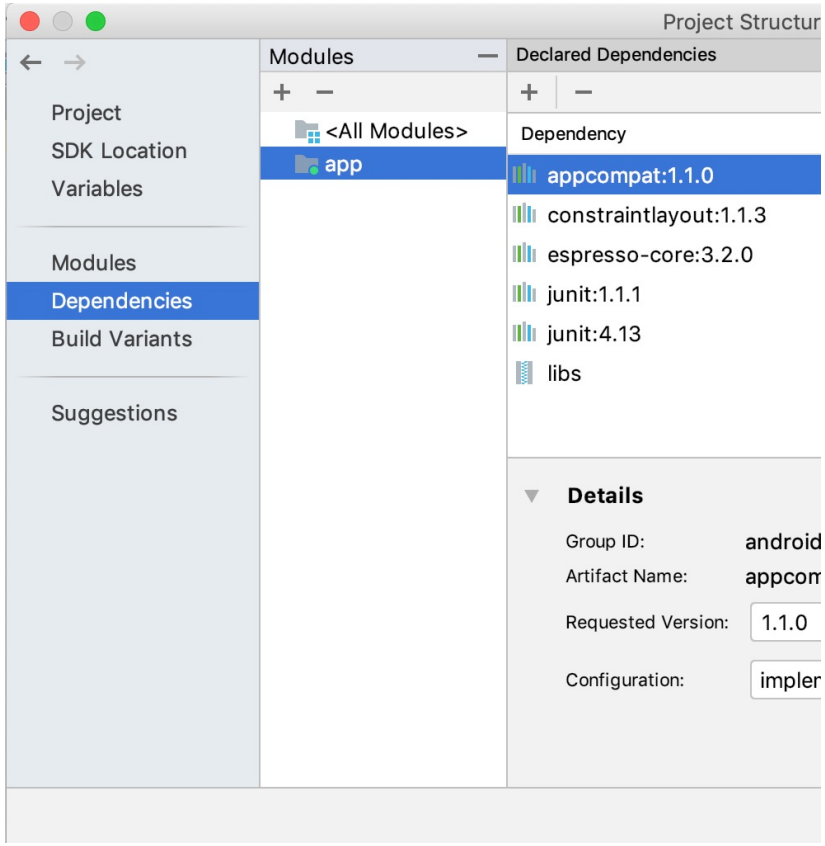
Gradle Sync Issues

- If you right click on the android project, and select “open module settings”, you will see all dependent modules of your project



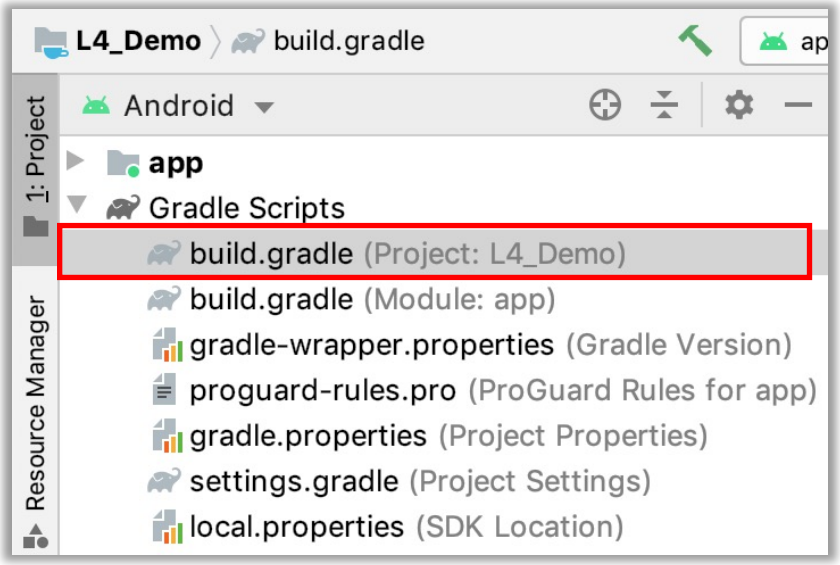
Gradle Sync Issues

- As shown on the right, some modules comes from online repository.
- This is managed by our build system: Gradle.



Gradle Sync Issues

- The online repository is defined in the main Gradle script
  - By default, it connects to Google's server.
- If your sync fails because connection to google fails, you should use the method in the next few slides.



Solving Gradle Sync Issues

1. In “buildscript” section, “repositories” subsection, replace the original content with:

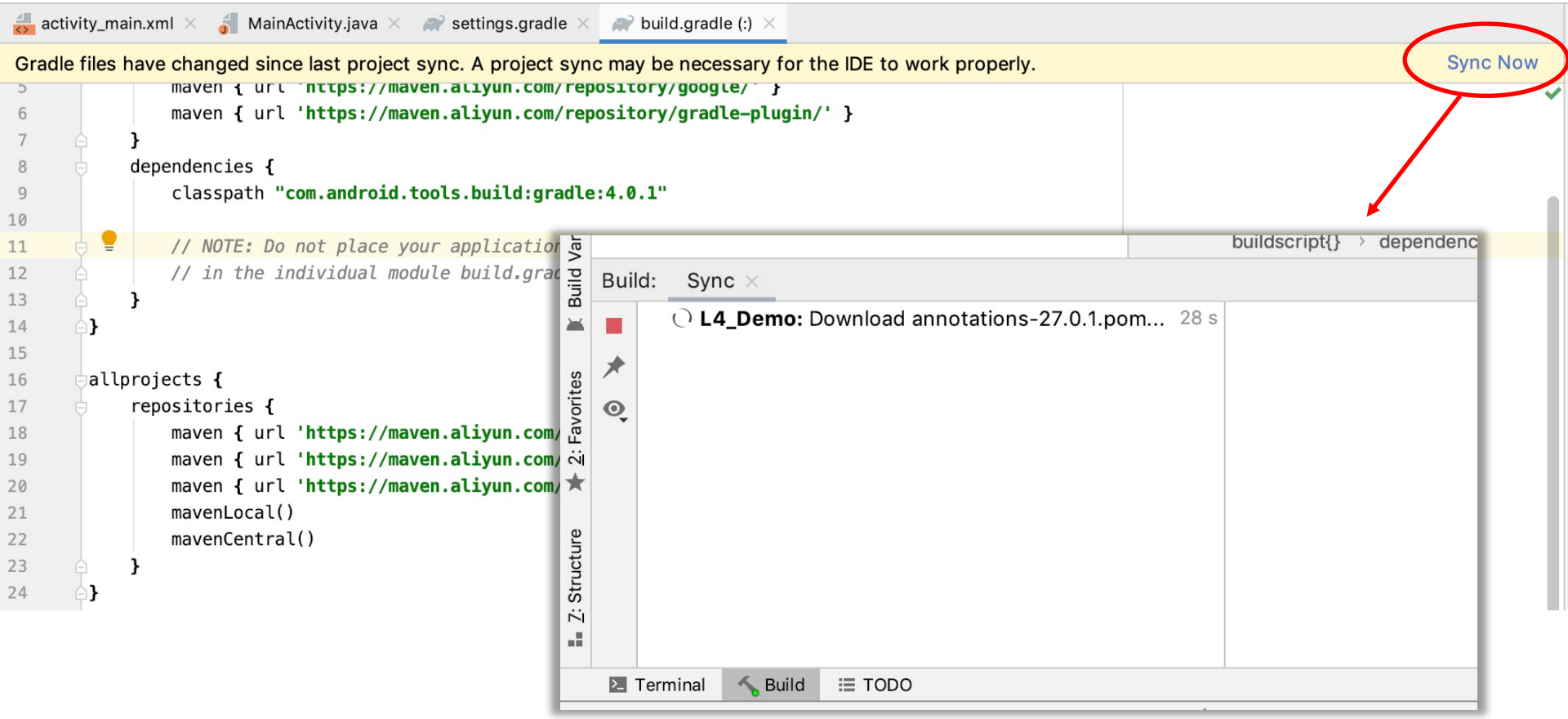
```
maven { url 'https://maven.aliyun.com/repository/public/' }
maven { url 'https://maven.aliyun.com/repository/google/' }
maven { url 'https://maven.aliyun.com/repository/gradle-plugin/' }
```

2. In “allprojects” section, “repositories” subsection, replace the original content with:

```
maven { url 'https://maven.aliyun.com/repository/public/' }
maven { url 'https://maven.aliyun.com/repository/google/' }
maven { url 'https://maven.aliyun.com/repository/gradle-plugin/' }
mavenLocal()
mavenCentral()
```

Gradle Sync Issues

- Once done, save and then resync.

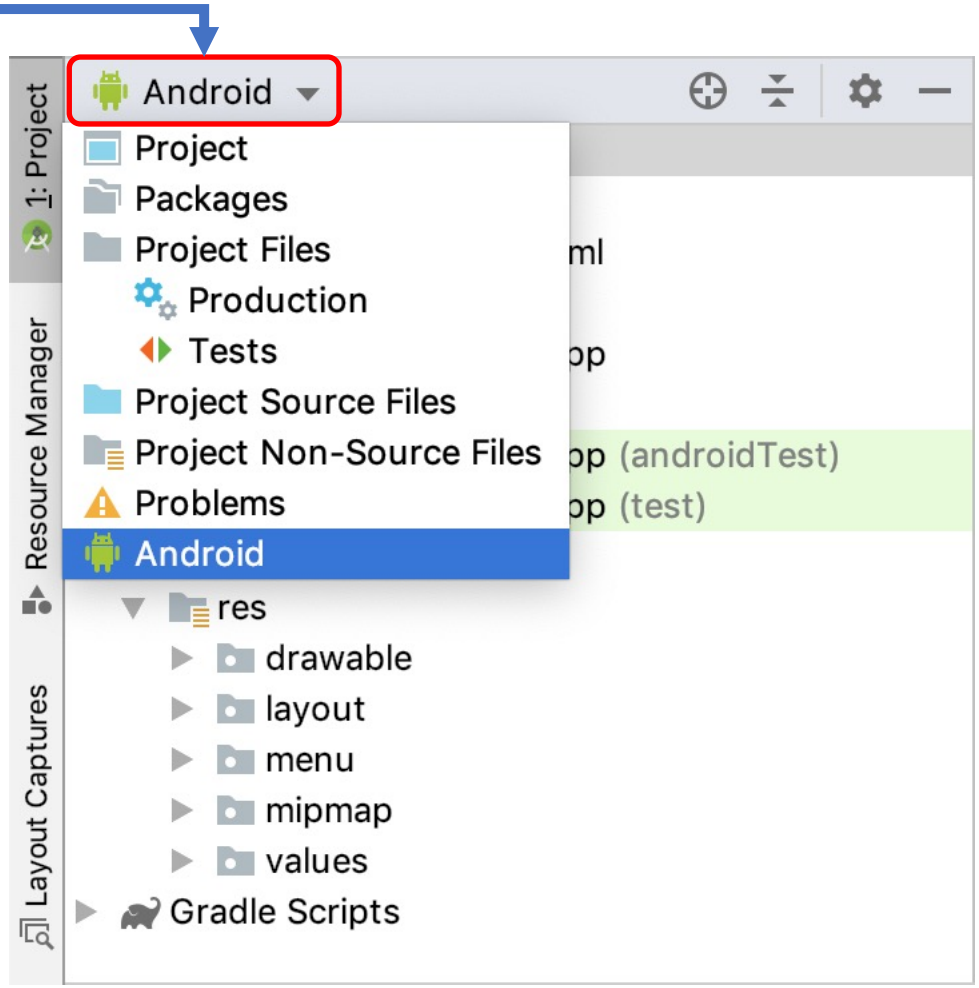


IDE Interfaces and Project File Structure

Project folders, layout files

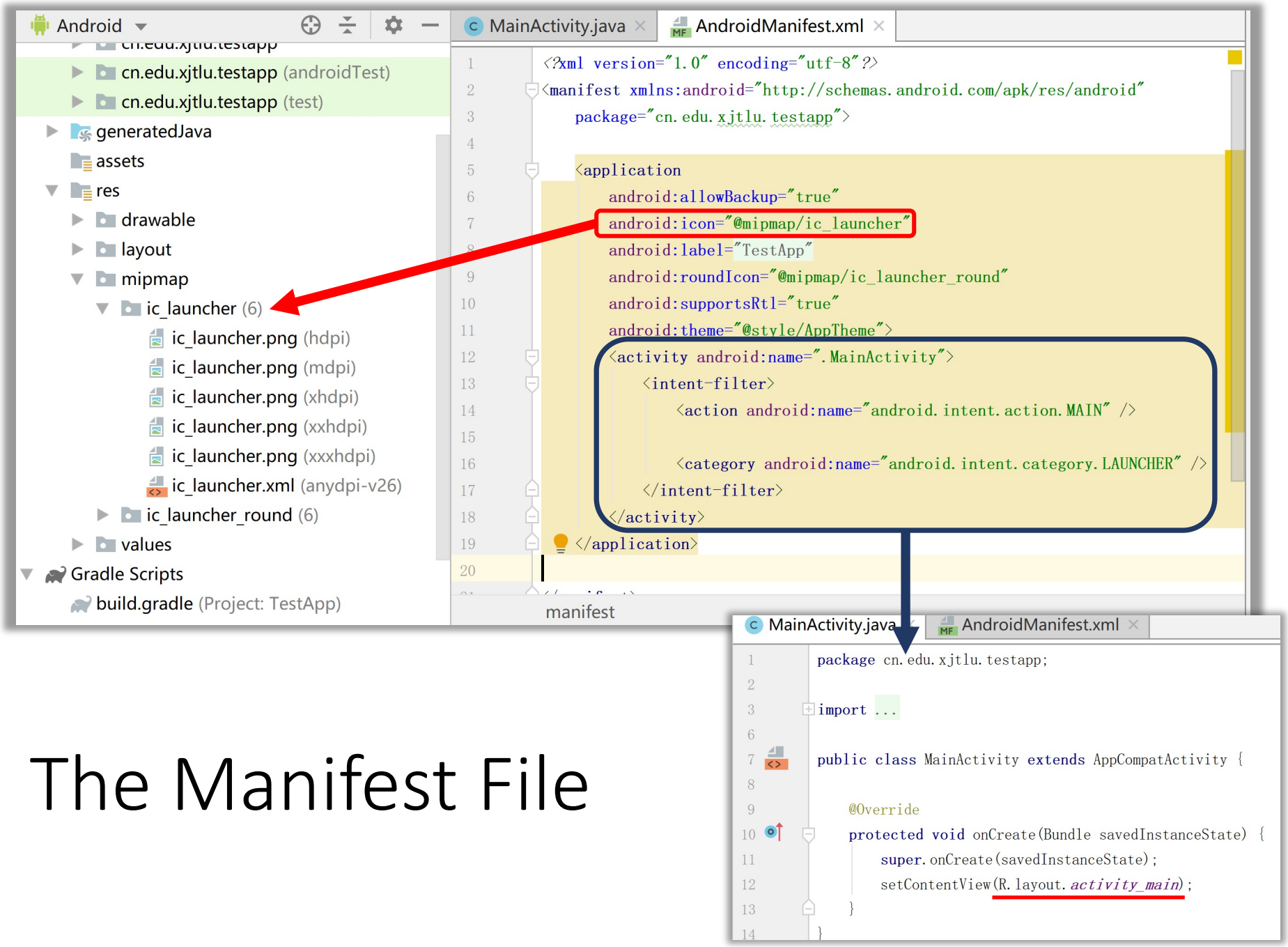
Project Window

- There are several view options:
  - "Android" view is easier to understand.
  - "Project" view matches the folder structure in the file system (as seen in your file explorer)



Project Folders

- **Manifests/AndroidManifest.xml**: Basic information of the app, such as app name, its activities, permissions required etc..
- **Java**: Your source code and test code.
- **res**: Various resources used by your app
  - **mipmap**: App icons.
  - **drawable**: Images (bitmaps, jpegs, pngs) used in your app.
  - **values**: store the values for the resources, such as color, styles, dimensions etc..
  - **layout**: UI layouts.
- **Gradle scripts**: Settings and scripts for the build system, such as the location for SDK.

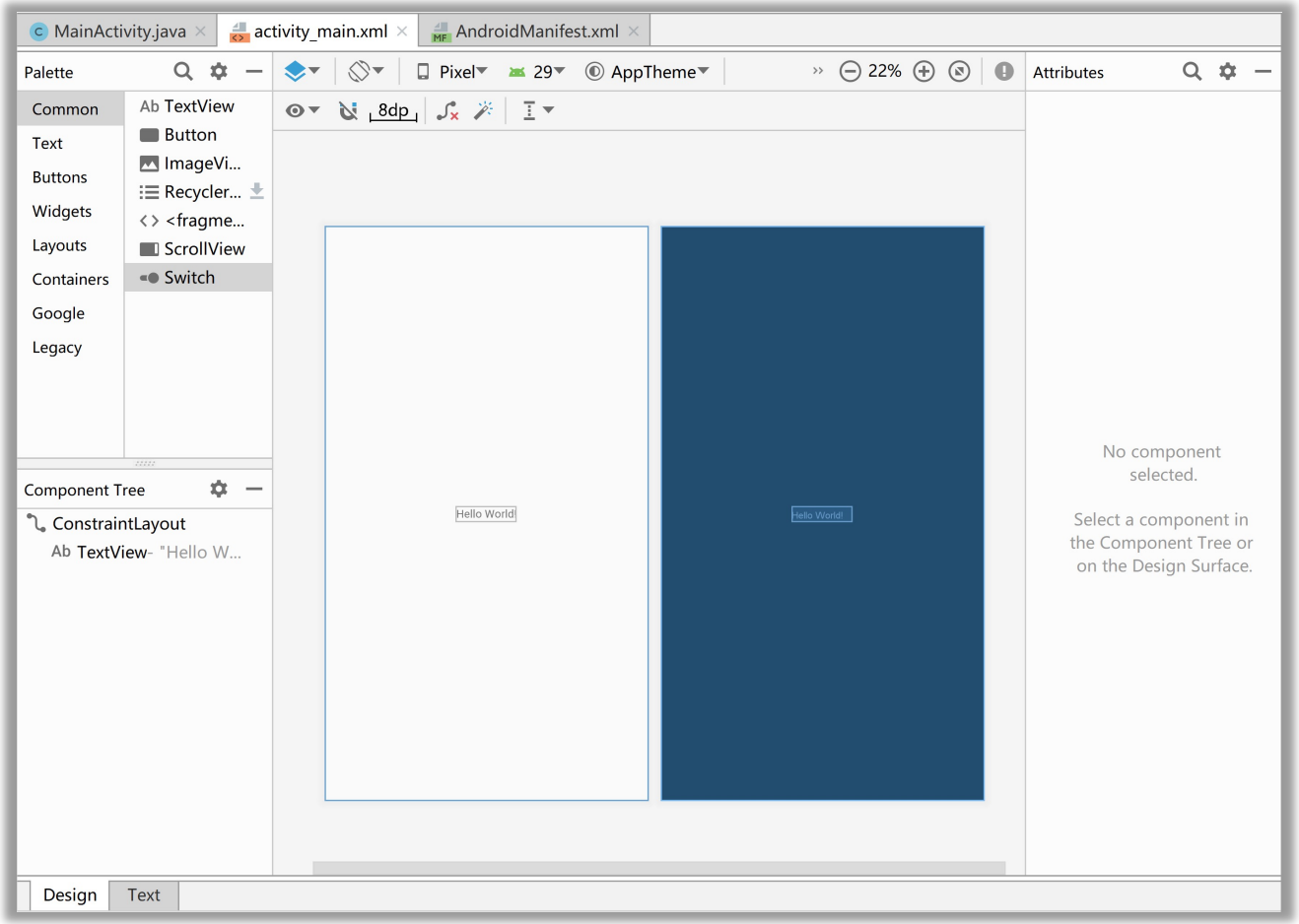


The Manifest File

Layout Files

- Layout files are xml files that describe the widget settings.
  - They can be loaded from your java code.
  - Through a special java file generated automatically by android studio: R.java (will be discussed later)
- There are two different views for layout files
  - Design view
  - Text view

Layout Files: Design View



Layout Files: Text View

MainActivity.java | activity_main.xml | AndroidManifest.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="ht
3      xmlns:app="http://schemas.android.com/apk/res-auto"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent"
7      tools:context=".MainActivity">
8
9      <TextView
10         android:layout_width="wrap_content"
11         android:layout_height="wrap_content"
12         android:text="Hello World!"
13         app:layout_constraintBottom_toBottomOf="parent"
14         app:layout_constraintLeft_toLeftOf="parent"
15         app:layout_constraintRight_toRightOf="parent"
16         app:layout_constraintTop_toTopOf="parent" />
17
18 </androidx.constraintlayout.widget.ConstraintLayout>
```

Running & Testing

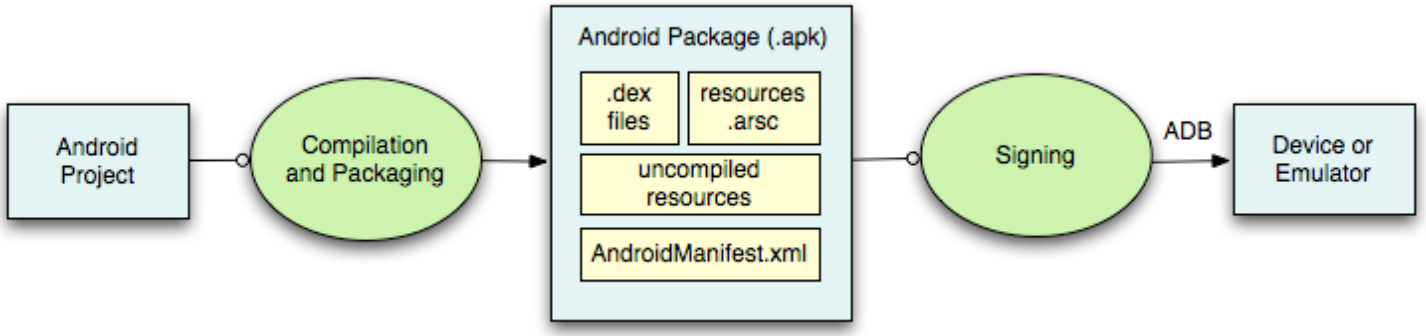
AVD setup

Connecting real devices

Tests

APK Files

- Android projects are compiled and packaged into a .apk file, the container for your application.
- To run an application on an emulator or device, the application must be signed using debug (or release) mode.
  - Signed: A certificate to identify the author of an application and establish trust relationships between applications



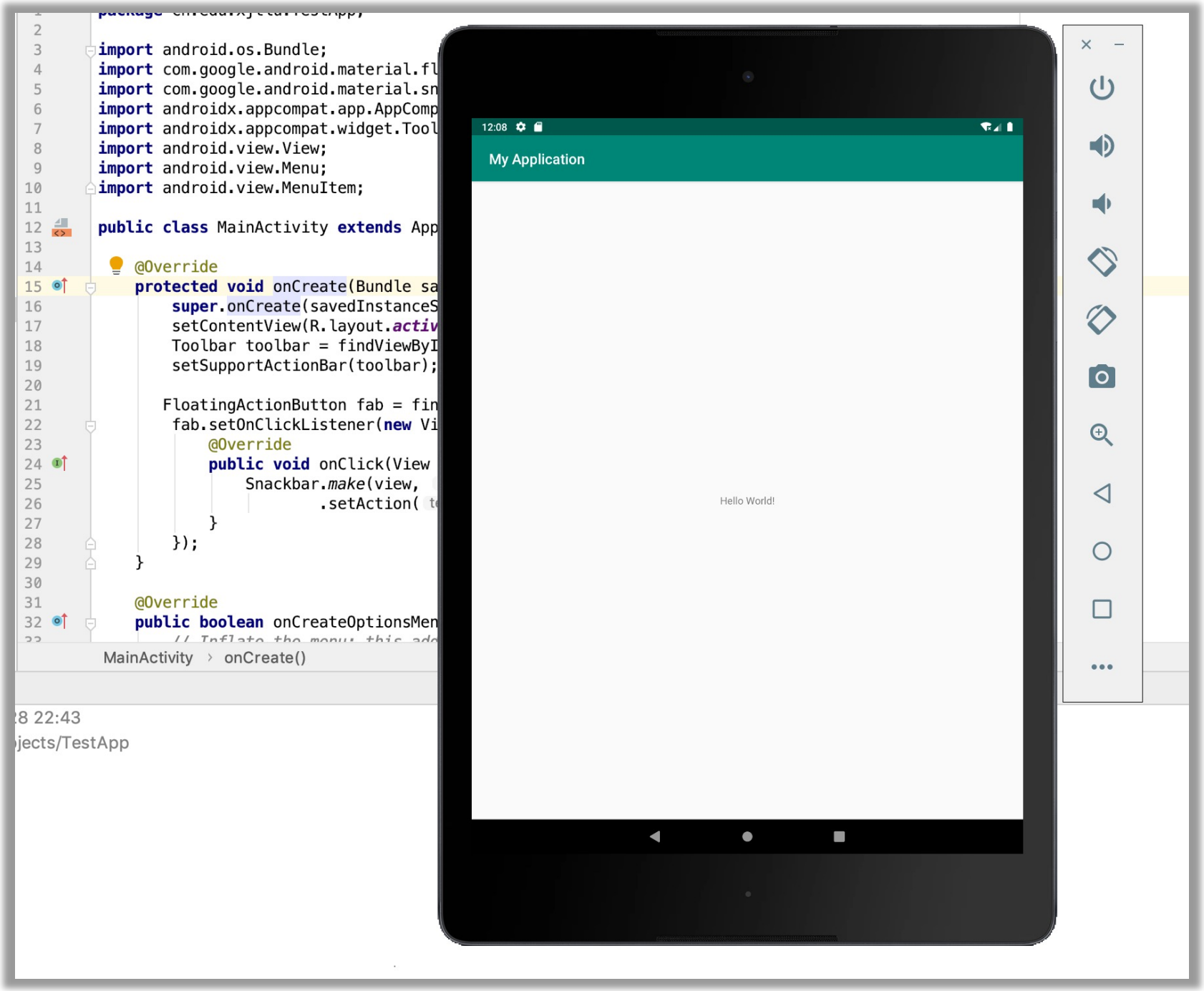
Running Android Apps

- By default, each application
  - is assigned a unique Linux user ID
  - executes in its own Linux process
- Android Apps are not Linux apps
- Android manages process creation & shutdown
  - Starts process when any of the application's code needs to be executed
  - Shuts down when process is no longer needed, and system resources are required by other applications

Android Virtual Device (AVD)

- To test the app you are developing, you'll need an android device, either virtual or real.
  - We have Xiaomi tablets this year. But the number is not enough for everyone. They will be used for the group project.

- To add an AVD:
  - Menu->Tools->AVD Manager
  - Toolbar Button


import android.os.Bundle;
import com.google.android.material.fl
import com.google.android.material.sn
import androidx.appcompat.app.AppComp
import androidx.appcompat.widget.Tool
import android.view.View;
import android.view.Menu;
import android.view.MenuItem;
public class MainActivity extends App
@Override
protected void onCreate(Bundle sa
super.onCreate(savedInstanceS
setContentView(R.layout.activ
Toolbar toolbar = findViewByI
setSupportActionBar(toolbar);
FloatingActionButton fab = fin
fab.setOnClickListener(new Vi
@Override
public void onClick(View
Snackbar.make(view,
.setAction(
});
@Override
public boolean onCreateOptionsMen
MainActivity › onCreate()
8 22:43
jects/TestApp
12:08
My Application
Hello World!

Connecting Real Devices

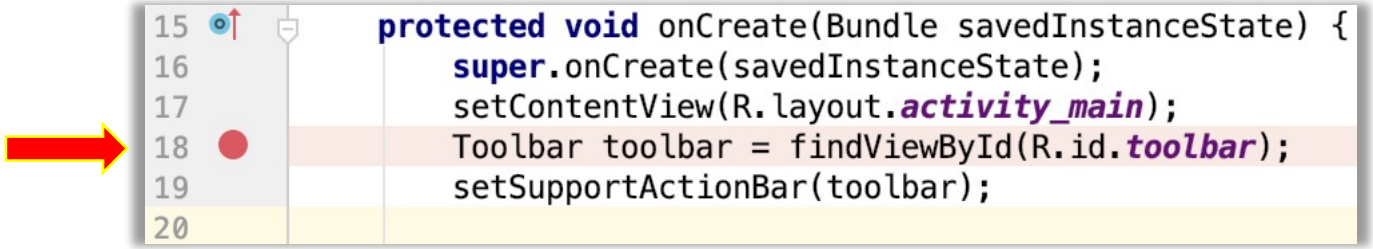
- Connect your device to your development machine
- Enable USB debugging in your device
  - Open the Settings app.
  - Scroll to the bottom and select About phone.
  - Scroll to the bottom and tap Build number 7 times.
  - Return to the previous screen to find Developer options near the bottom.
  - Open Developer options, and then scroll down to find and enable USB debugging.

Connecting Real Devices

- Run the app on your device as follows:
  - In Android Studio, click Run in the toolbar.
  - In the Select Deployment Target window, select your device, and click OK.
  - Android Studio will install the app on your device and starts it (may take a while when starting for the first time!). You should see the app on your device.

Debugging

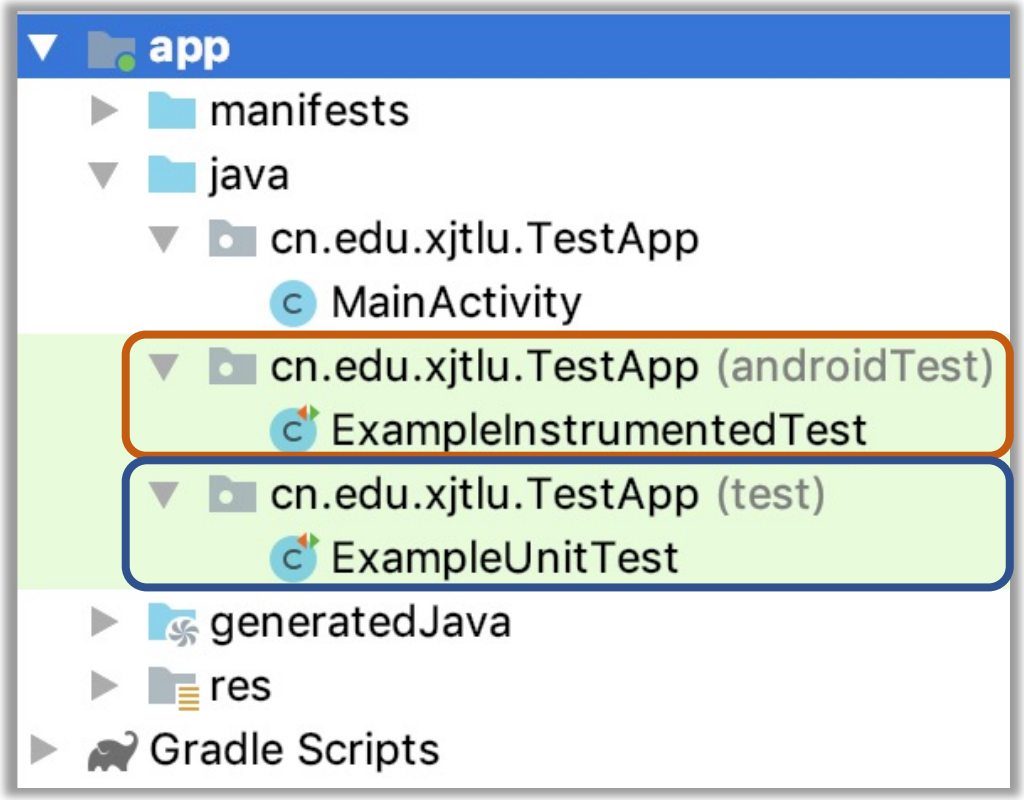
- Adding breakpoints:
  - Single click on the left area of the code editor (the empty area at the right side of code line numbers)
  - Or press Ctrl + F8 at the line you want to add/remove breakpoint.



- Debug mode:
  - Menu -> Run -> Debug ‘App’

Writing Tests

- Test cases are placed along with your main App source in the project view.
- Unit tests run locally in the IDE and can help check the correctness of single components.
- Instrumented unit tests are tests that run on physical devices and emulators



https://developer.android.com/training/testing/unit-testing/local-unit-tests
https://developer.android.com/training/testing/unit-testing/instrumented-unit-tests

Lab Session Practice

- Get yourselves familiar with Android Studio
- Practice the following tasks:
  - Create an application named “MyApp”.
  - Change the icon of your app with a picture you preferred
  - Change the name of your app
  - Change the content your app shows when launched
- Set up Android Studio on desktop if you don’t plan to use your laptop in future labs.
  - Download SDK and AVD.
  - You will have to do this again if you change seat.

Don’t Forget

The official manuals and guides can be found at:

https://developer.android.com/guide

or

https://developer.android.google.cn/guide