

FALL 2018 - NARGES NOROUZI

MOBILE APPLICATIONS

LECTURE 2

Resource: <https://developers.google.com/training/android/>

CONTENT

- Android is an ecosystem
- Android platform architecture
- Android Versions
- App fundamentals
- Basic app development workflow with Android Studio

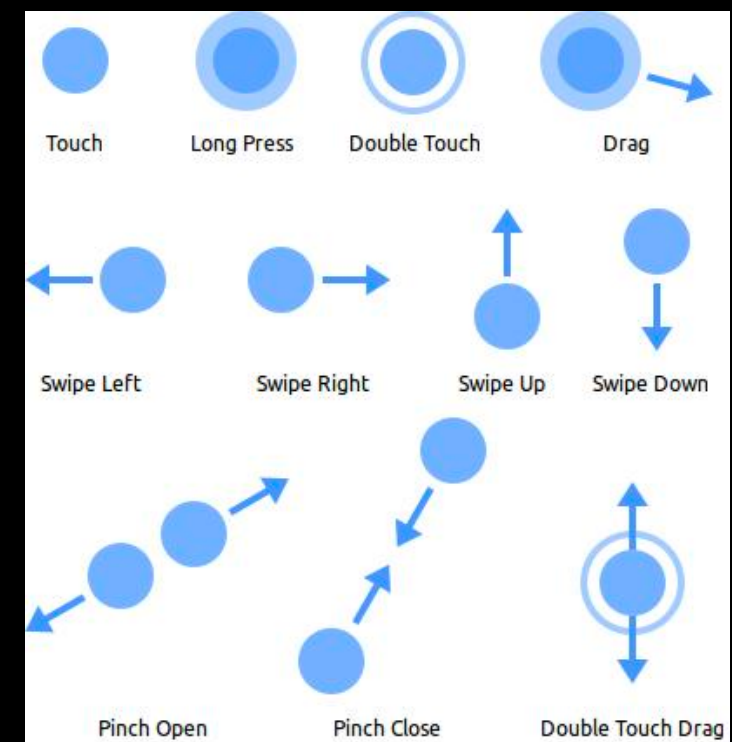
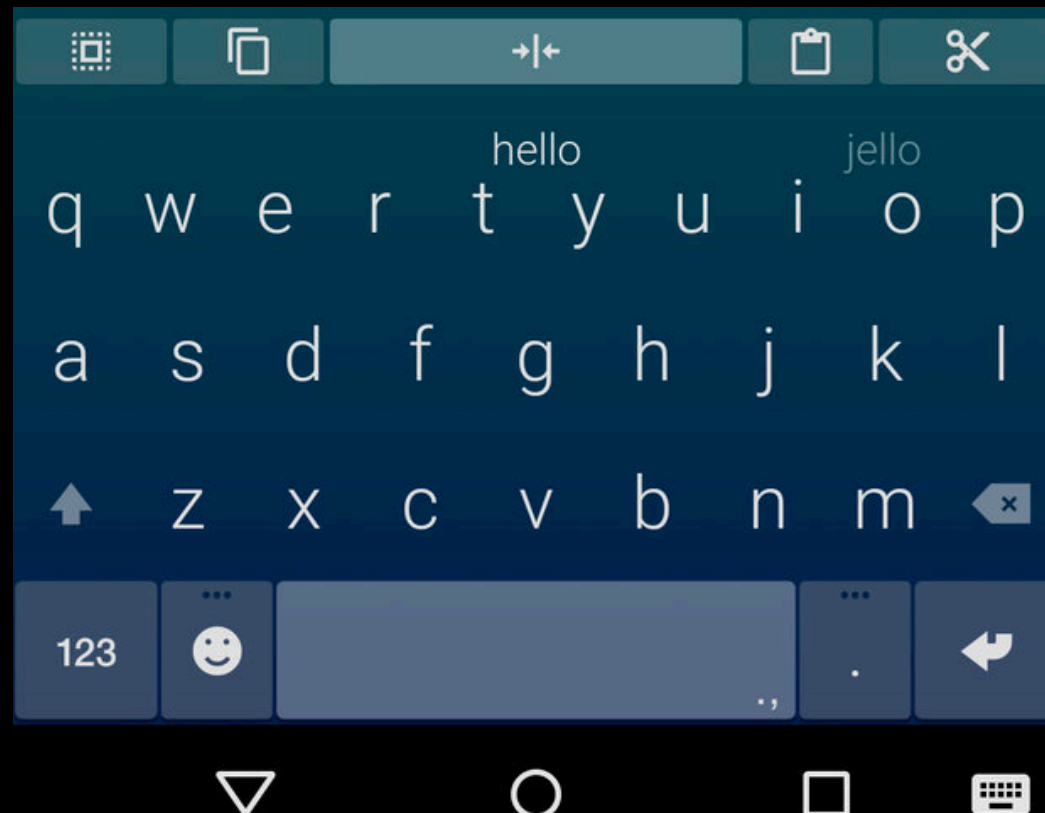
ANDROID ECOSYSTEM

WHAT IS ANDROID?

- Mobile operating system based on Linux kernel
- User Interface for touch screens
- Used on over 80% of all smartphones
- Powers devices such as watches, TVs, and cars
- Over 2 Million Android apps in Google Play store
- Highly customizable for devices / by vendors
- Open source

ANDROID USER INTERACTION

- Touch gestures: swiping, tapping, pinching
- Virtual keyboard for characters, numbers, and emoji
- Support for Bluetooth, USB controllers and peripherals

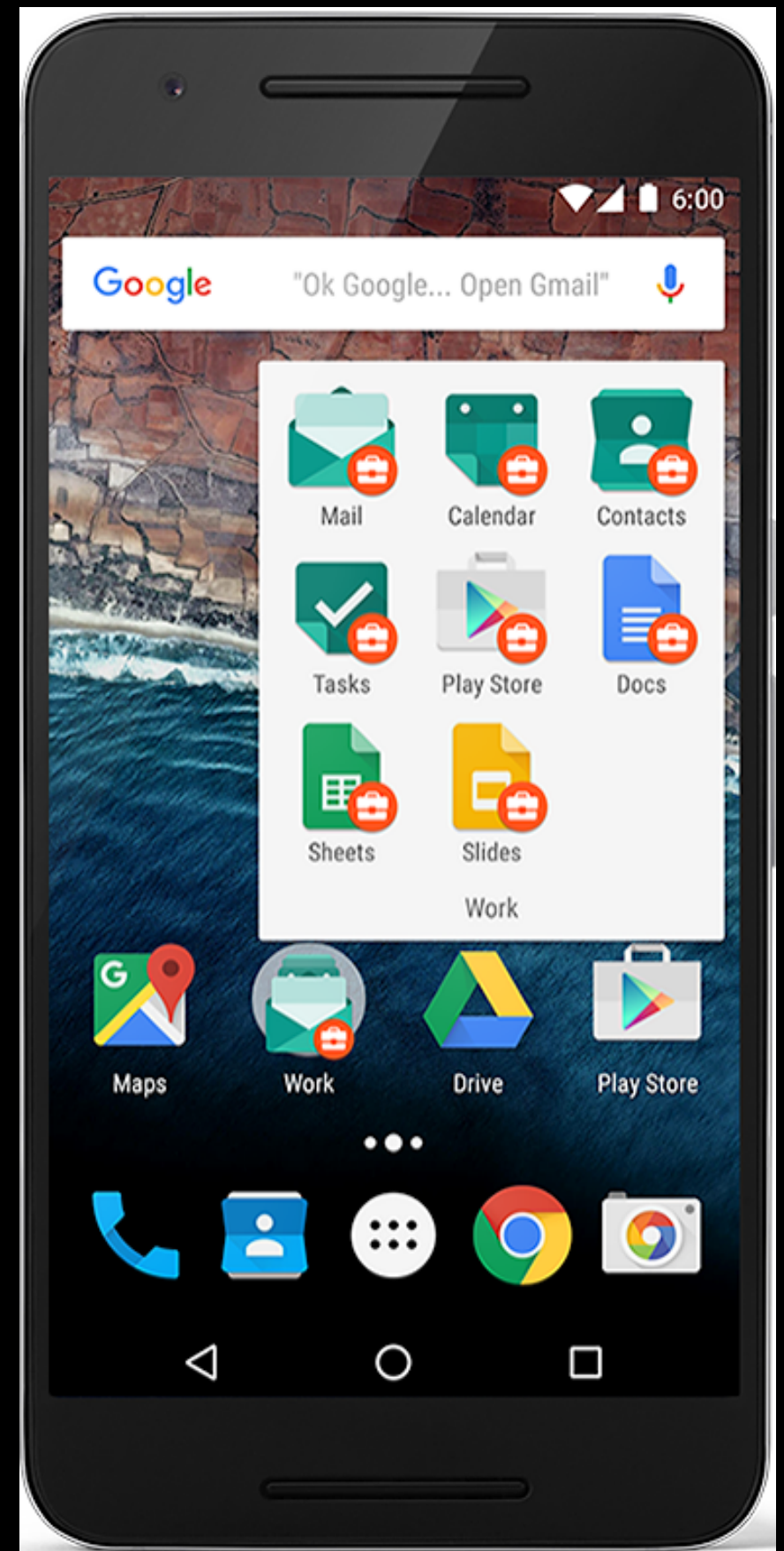


ANDROID AND SENSORS

- Sensors can discover user action and respond
 - Device contents rotate as needed
 - Walking adjusts position on map
 - Tilting steers a virtual car or controls a physical toy
 - Moving too fast disables game interactions

ANDROID HOME SCREEN

- Launcher icons for apps
- Self-updating widgets for live content
- Can be multiple pages
- Folders to organize apps
- "OK Google"

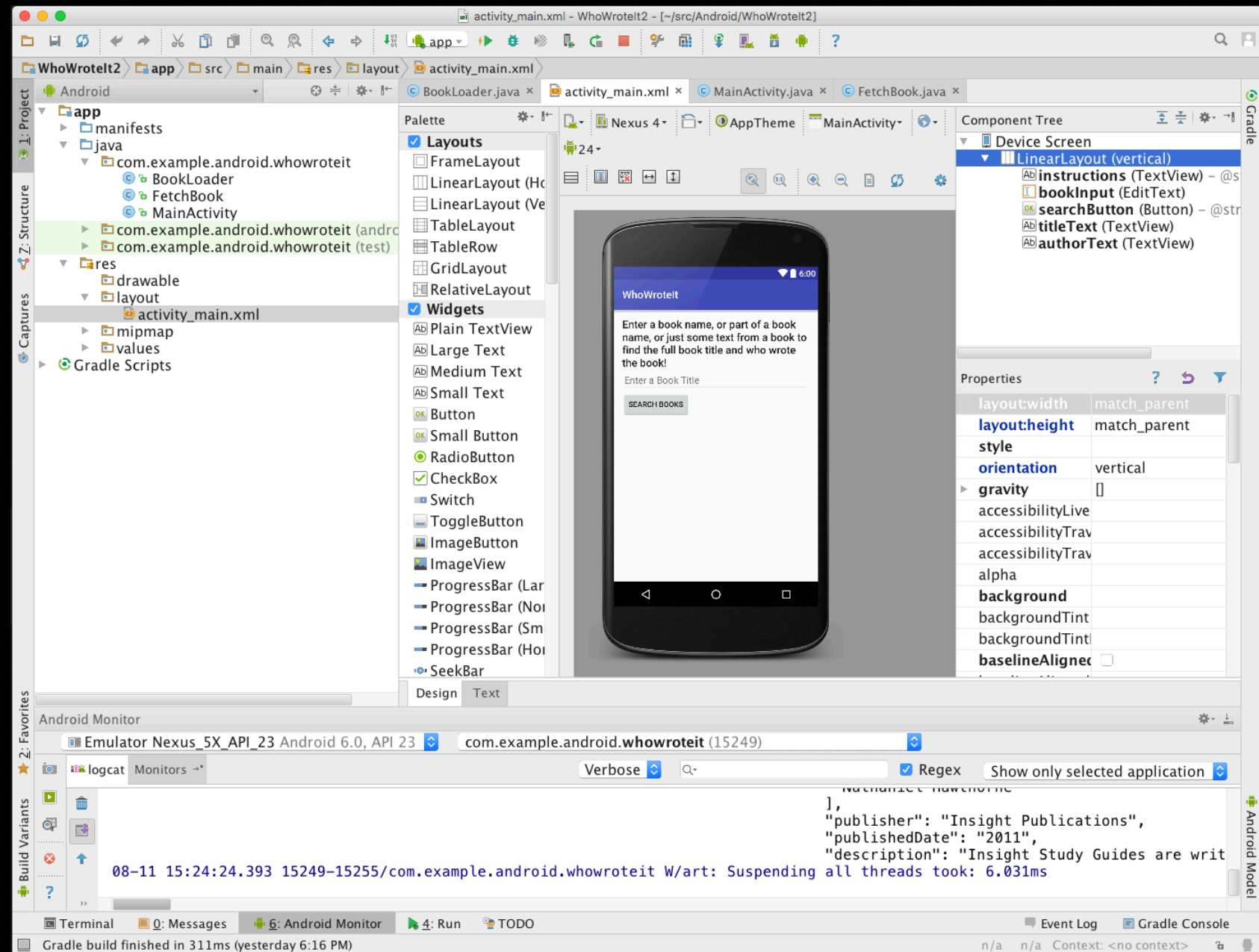


ANDROID SOFTWARE DEVELOPMENT KIT (SDK)

- Development tools (debugger, monitors, editors)
- Libraries (maps, wearables)
- Virtual devices (emulators)
- Documentation (developers.android.com)
- Sample code

ANDROID STUDIO

- Android IDE
- Project structure
- Templates
- Visual Layout Editor
- Testing tools
- Gradle-based build
- Log Console
- Debugger
- Monitors
- Emulators



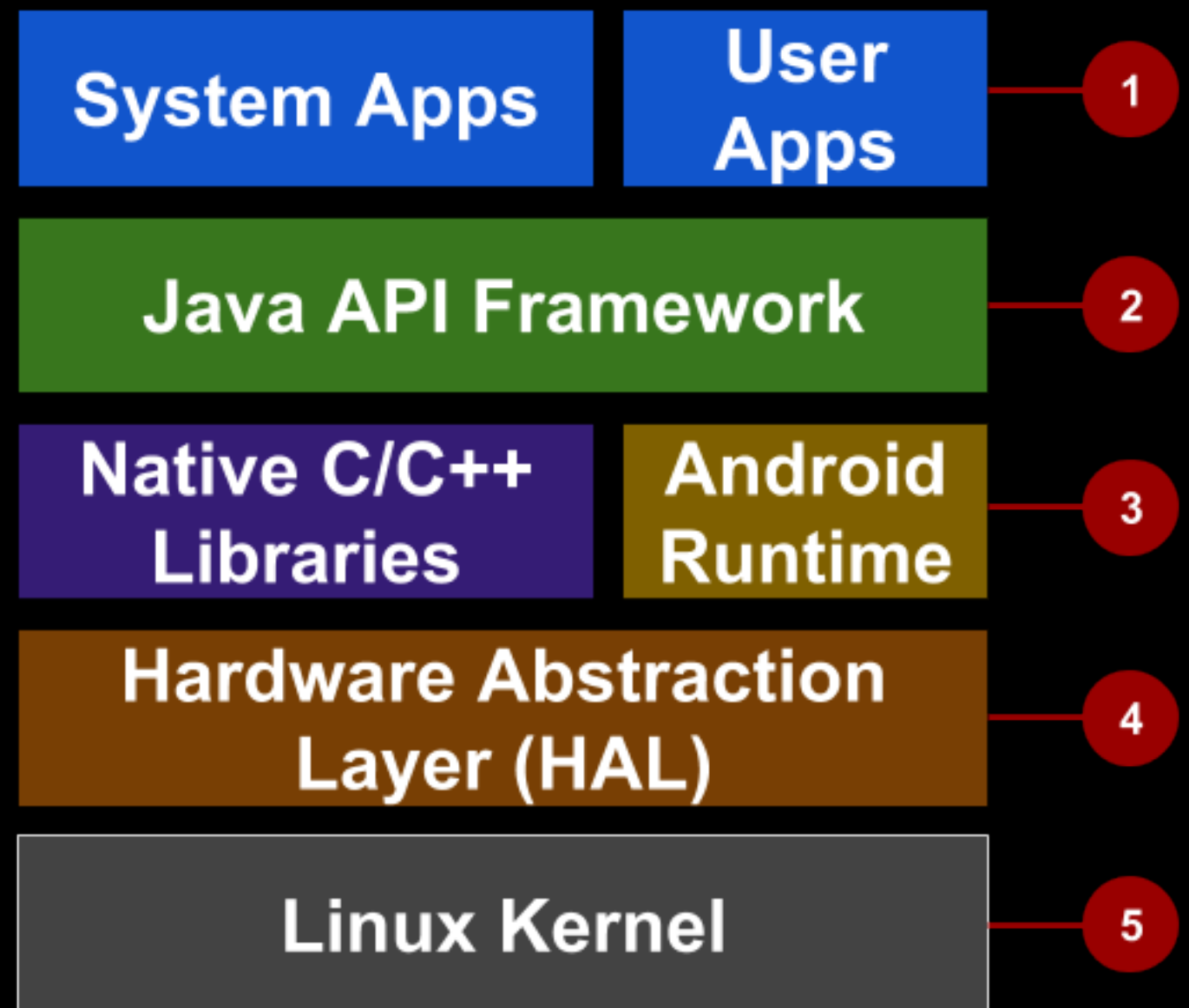
GOOGLE PLAY STORE

- Publish apps through Google Play store:
- Official app store for Android
- Digital distribution service operated by Google

ANDROID PLATFORM ARCHITECTURE

ANDROID STACK

1. System and user apps
2. Android OS API in Java framework
3. Expose native APIs; run apps
4. Expose device hardware capabilities
5. Linux Kernel



SYSTEM AND USER APPS

- System apps have no special status
- Built-in applications
- System apps provide key capabilities to app developers
- Example: Your app can use a system app to deliver a SMS message.

JAVA API FRAMEWORK

- The entire feature-set of the Android OS is available to you through APIs written in the Java language.
- View class hierarchy to create UI screens
- Notification manager
- Activity manager for life cycles and navigation
- Content providers to access data from other apps

ANDROID RUNTIME

- Each app runs in its own process with its own instance of the Android Runtime.

C/C++ LIBRARIES

- Core C/C++ Libraries give access to core native Android system components and services.

HARDWARE ABSTRACTION LAYER (HAL)

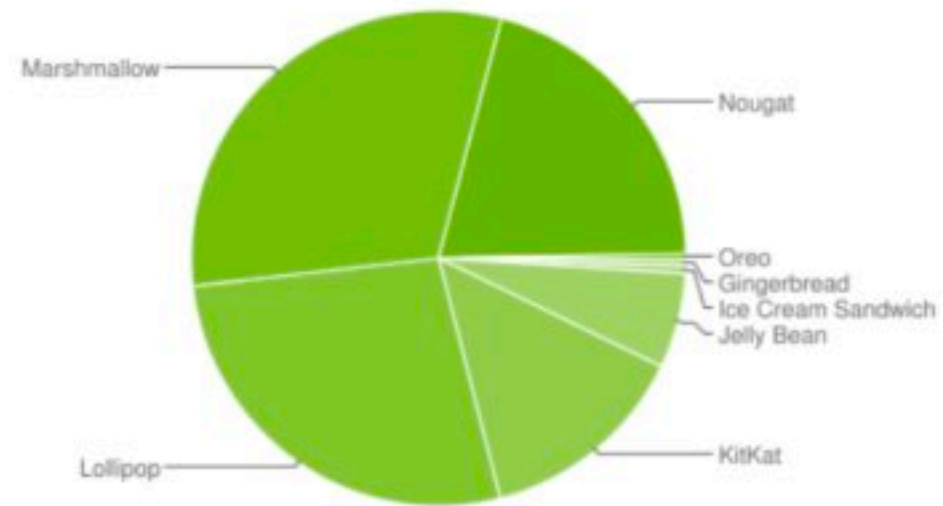
- Standard interfaces that expose device hardware capabilities as libraries
- Examples: Camera, bluetooth module

LINUX KERNEL

- Threading and low-level memory management
- Security features
- Drivers

ANDROID VERSIONS

Version	Codename	API	Distribution
2.3.3 - 2.3.7	Gingerbread	10	0.5%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	0.5%
4.1.x	Jelly Bean	16	2.2%
4.2.x		17	3.1%
4.3		18	0.9%
4.4	KitKat	19	13.8%
5.0	Lollipop	21	6.4%
5.1		22	20.8%
6.0	Marshmallow	23	30.9%
7.0	Nougat	24	17.6%
7.1		25	3.0%
8.0	Oreo	26	0.3%



Data collected during a 7-day period ending on November 9, 2017.

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

APP DEVELOPMENT

WHAT IS AN ANDROID APP?

- One or more interactive screens
- Written using **Java** Programming Language and **XML**
- Uses the Android Software Development Kit (**SDK**)
- Uses Android libraries and Android Application Framework
- Executed by Android Runtime Virtual machine (ART)

APP BUILDING BLOCKS

- **Resources:** layouts, images, strings, colors as XML and media files
- **Components:** activities, services, ..., and helper classes as Java code
- **Manifest:** information about app for the runtime
- **Build configuration:** APK versions in Gradle config files

COMPONENT TYPES

- **Activity** is a single screen with a user interface
- **Service** performs long-running tasks in background
- **Content provider** manages shared set of data
- **Broadcast receiver** responds to system-wide announcements

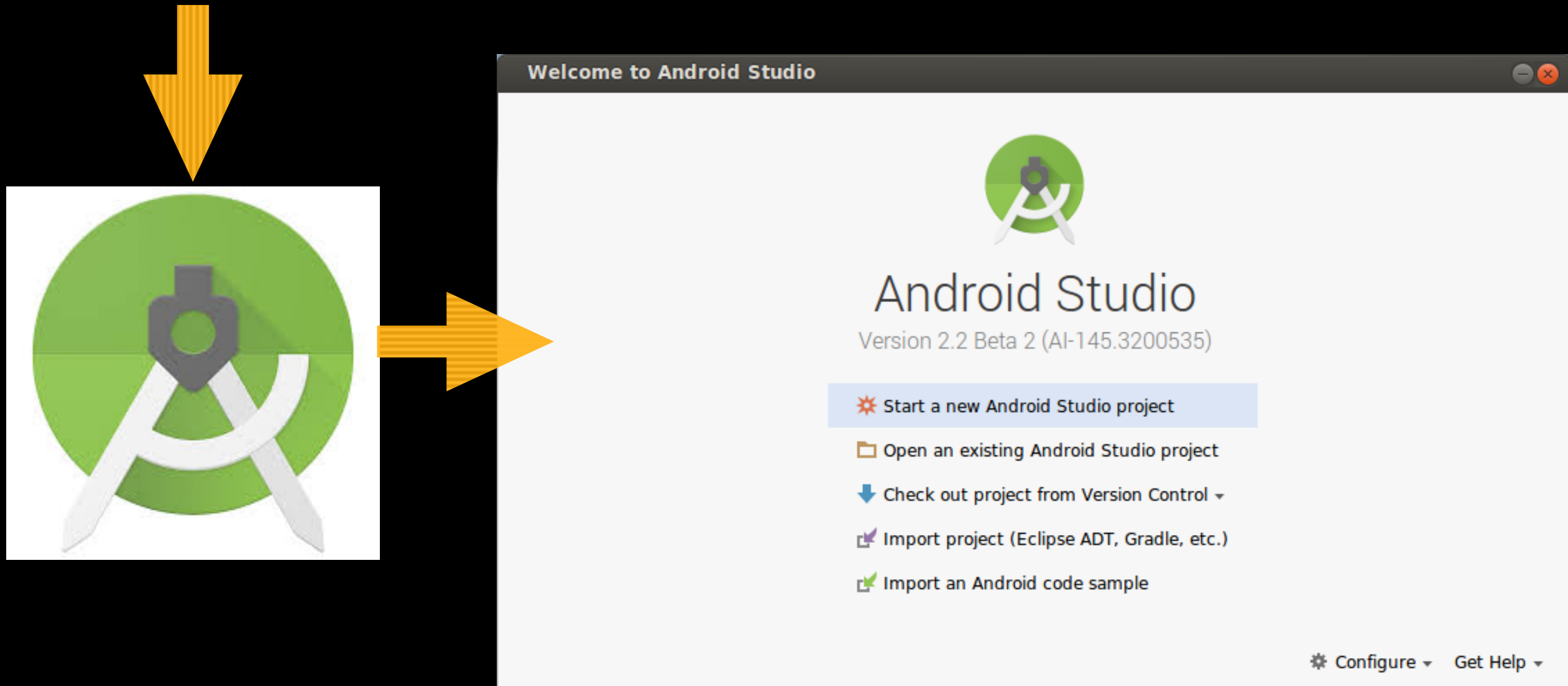
THINK OF ANDROID AS A HOTEL

- Your app is the guest
- The Android System is the hotel manager
- Services are available when you request them (intents)
- In the foreground (activities) such as registration
- In the background (services) such as laundry
- Calls you when a package has arrived (broadcast receiver)
- Access the city's tour companies (content provider)

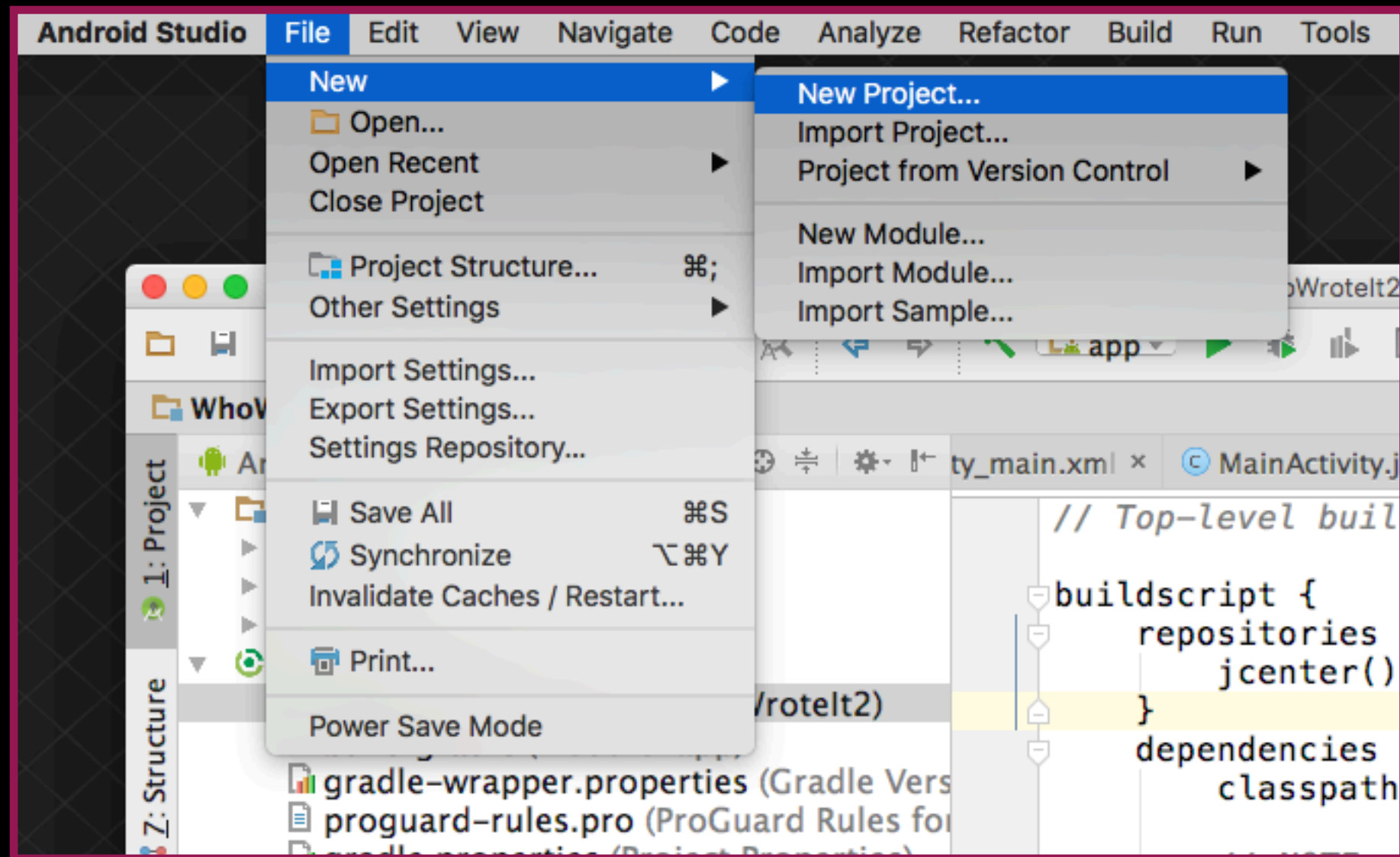
5 MINUTES BREAK

CREATING FIRST ANDROID APP

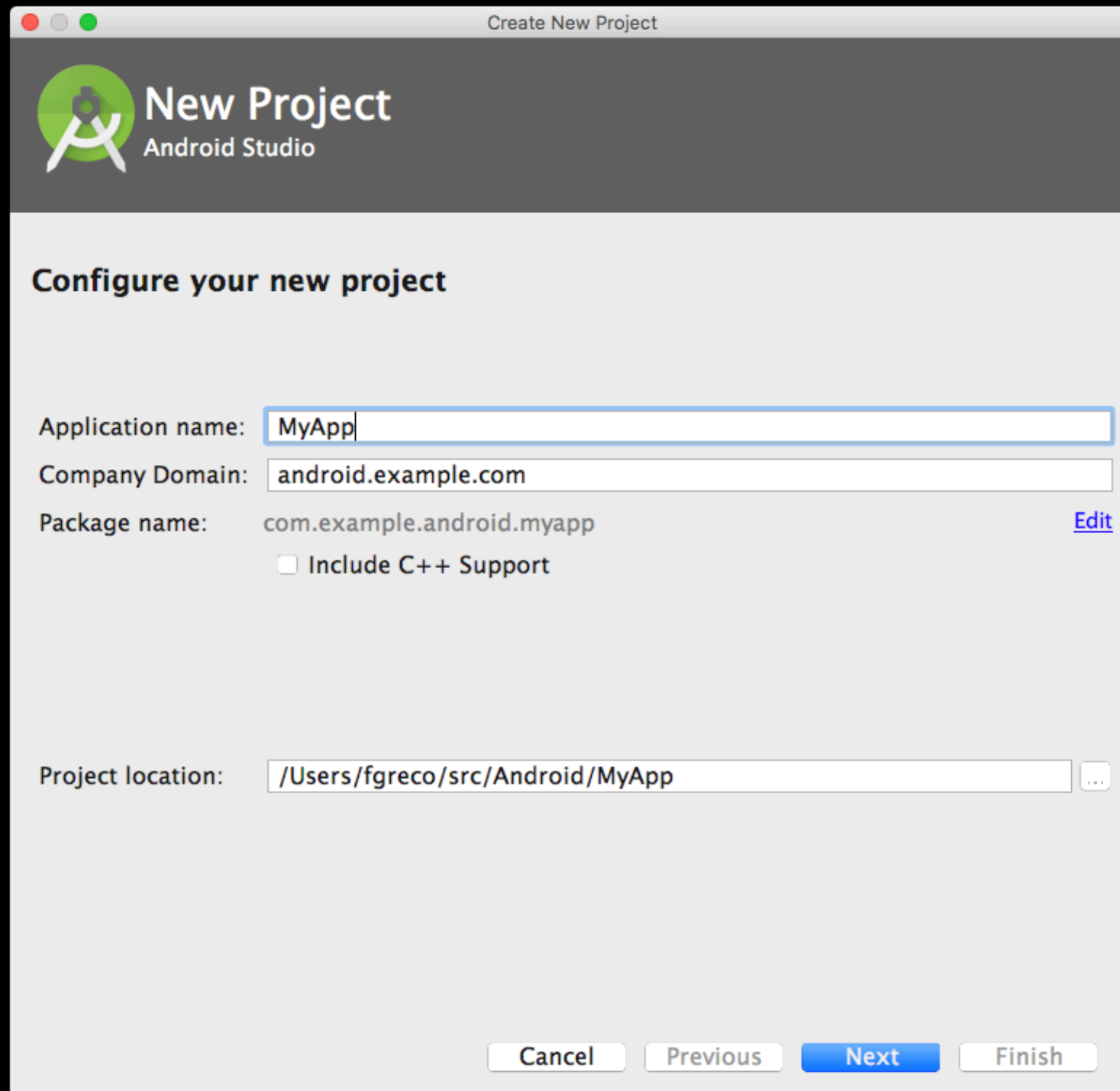
START ANDROID STUDIO




CREATE A PROJECT INSIDE ANDROID STUDIO



NAME YOUR APP



Create New Project

 **New Project**
Android Studio

Configure your new project

Application name:

Company Domain:

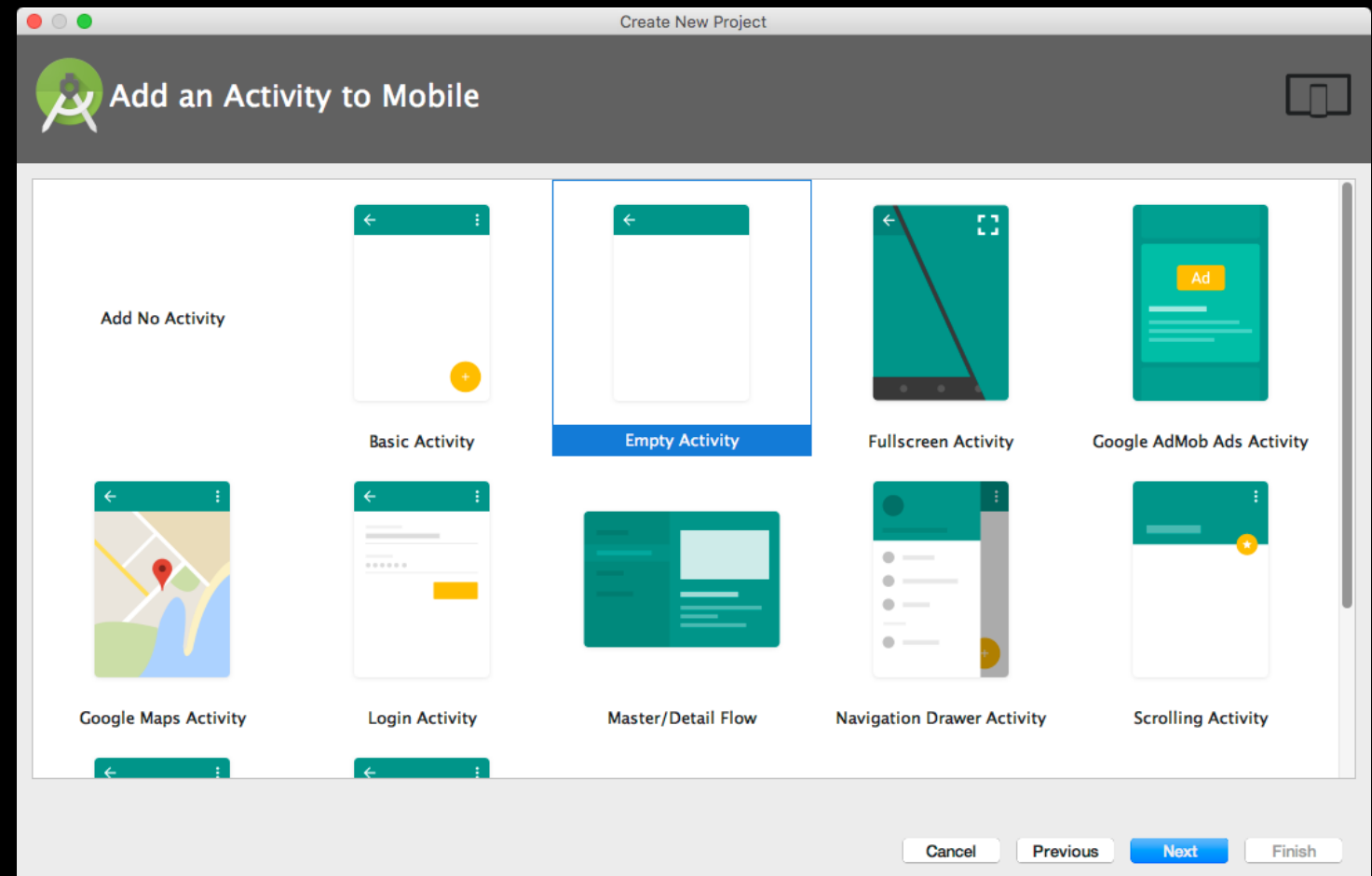
Package name: [Edit](#)

☐ Include C++ Support

Project location:

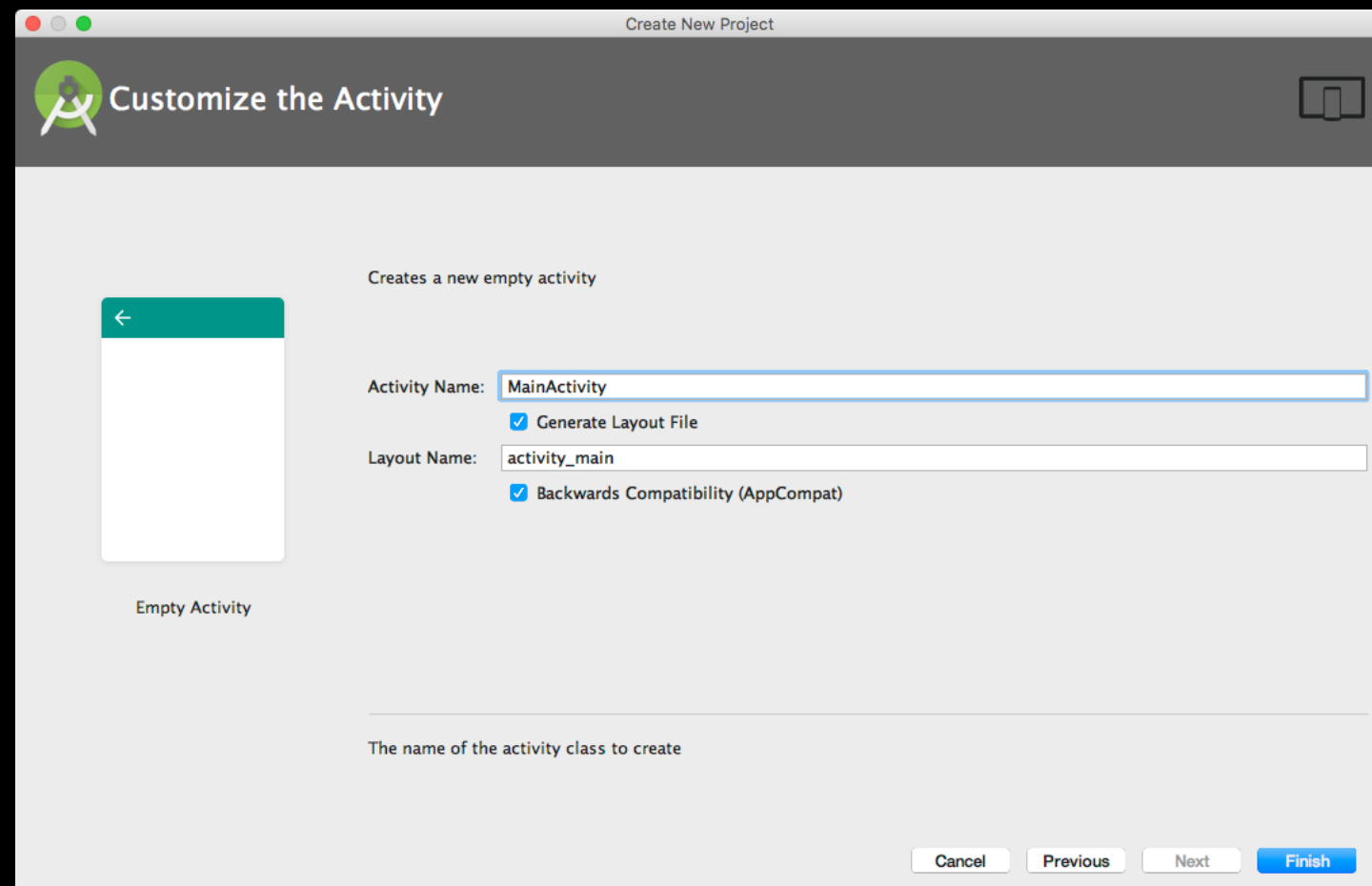
PICK ACTIVITY TEMPLATE

- Choose templates for common activities, such as maps or navigation drawers.
- Pick Empty Activity or Basic Activity for simple and custom activities.

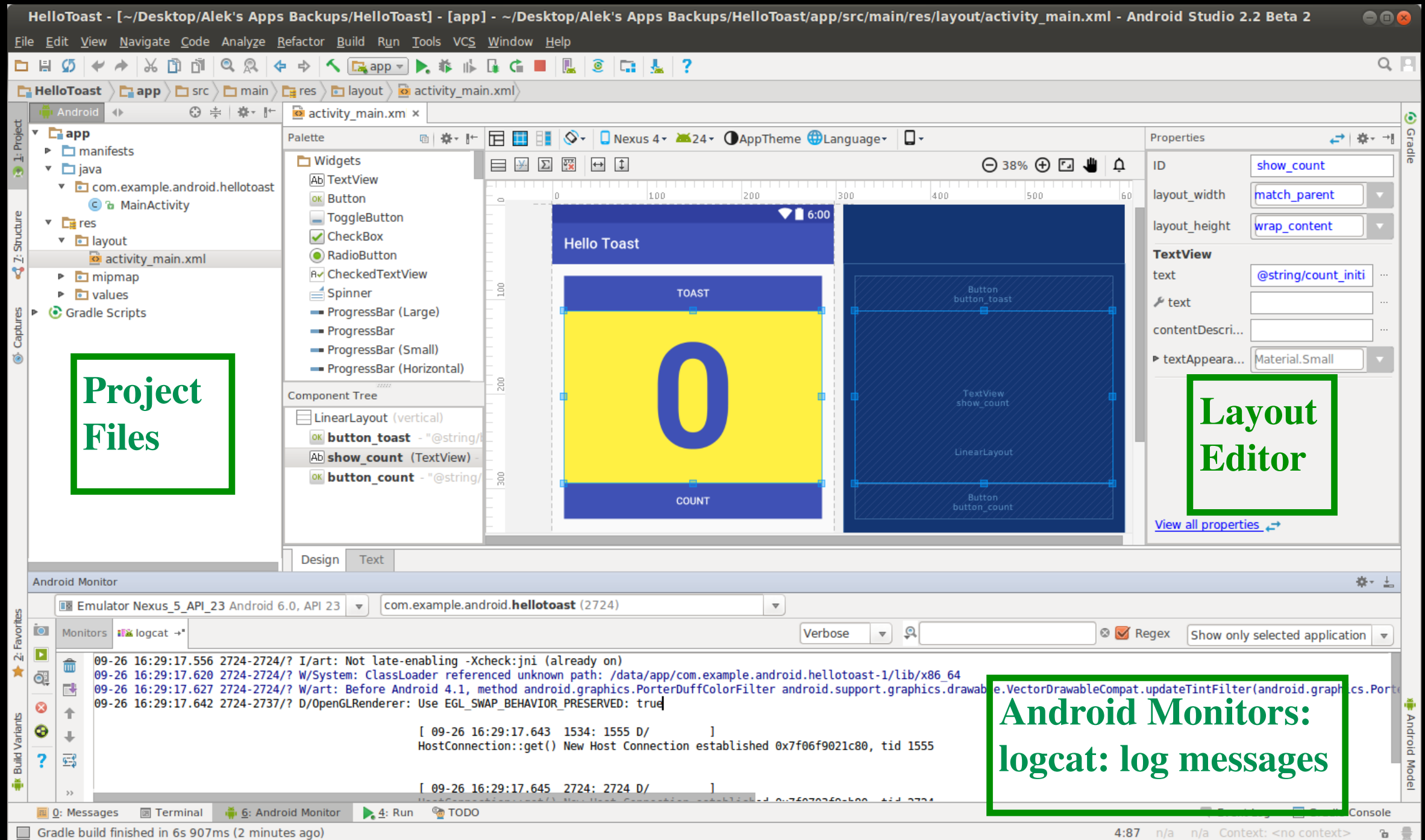


NAME YOUR ACTIVITY

- Good practice to name main activity MainActivity and activity_main layout
- Use AppCompatActivity
- Generating layout file is convenient

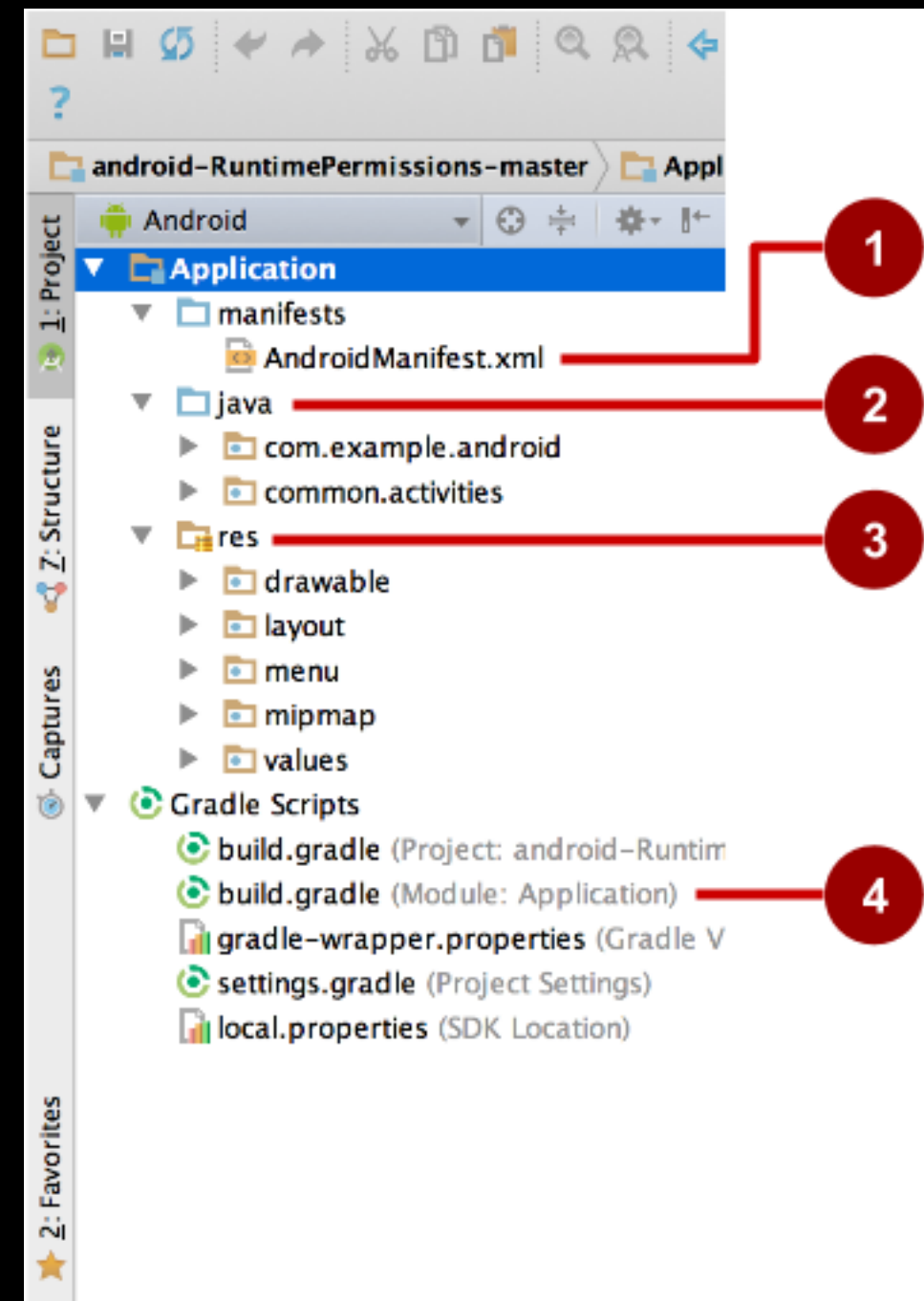


ANDROID STUDIO PANES



PROJECT FOLDERS

- manifests—Android Manifest file - description of app read by the Android runtime
- java—Java source code packages
- res—Resources (XML) - layout, strings, images, dimensions, colors...
- build.gradle—Gradle build files

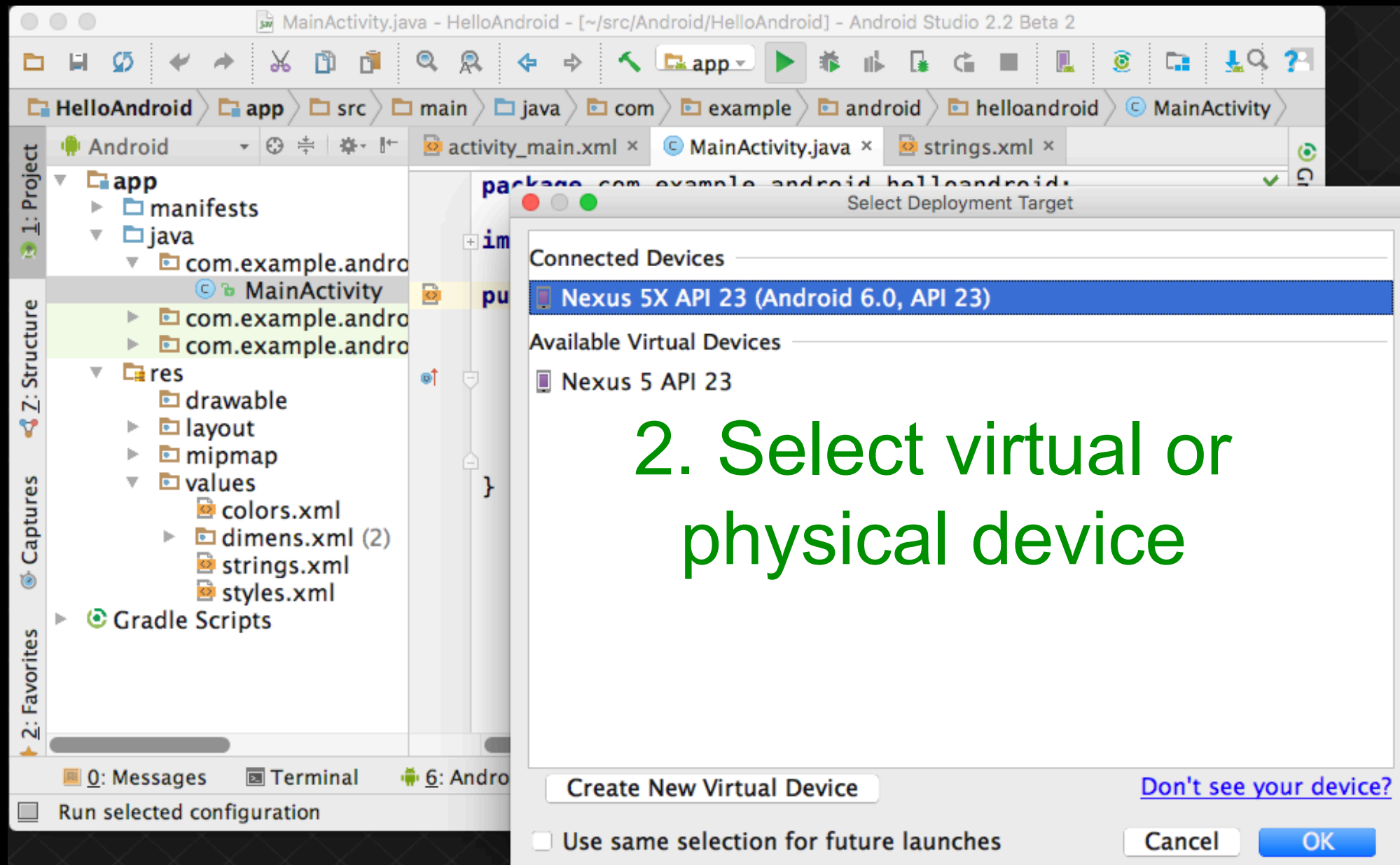


GRADLE BUILD SYSTEM

- Modern build subsystem in Android Studio
- Three build.gradle:
 - project
 - module
 - settings
- Typically not necessary to know low-level Gradle details
- Learn more about gradle at <https://gradle.org/>

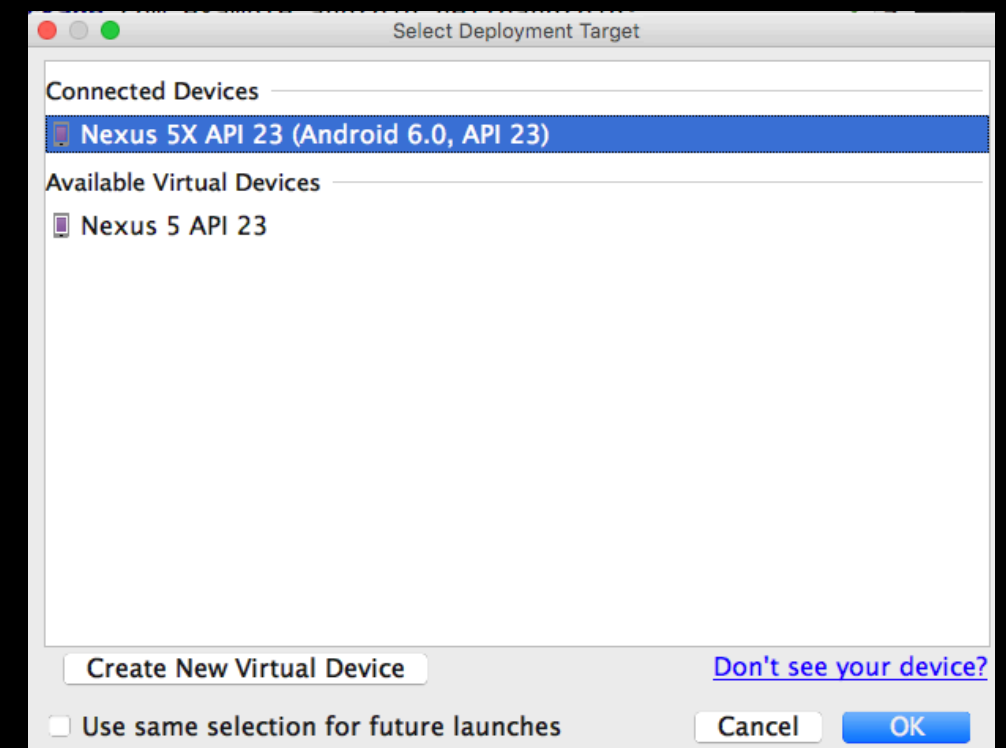
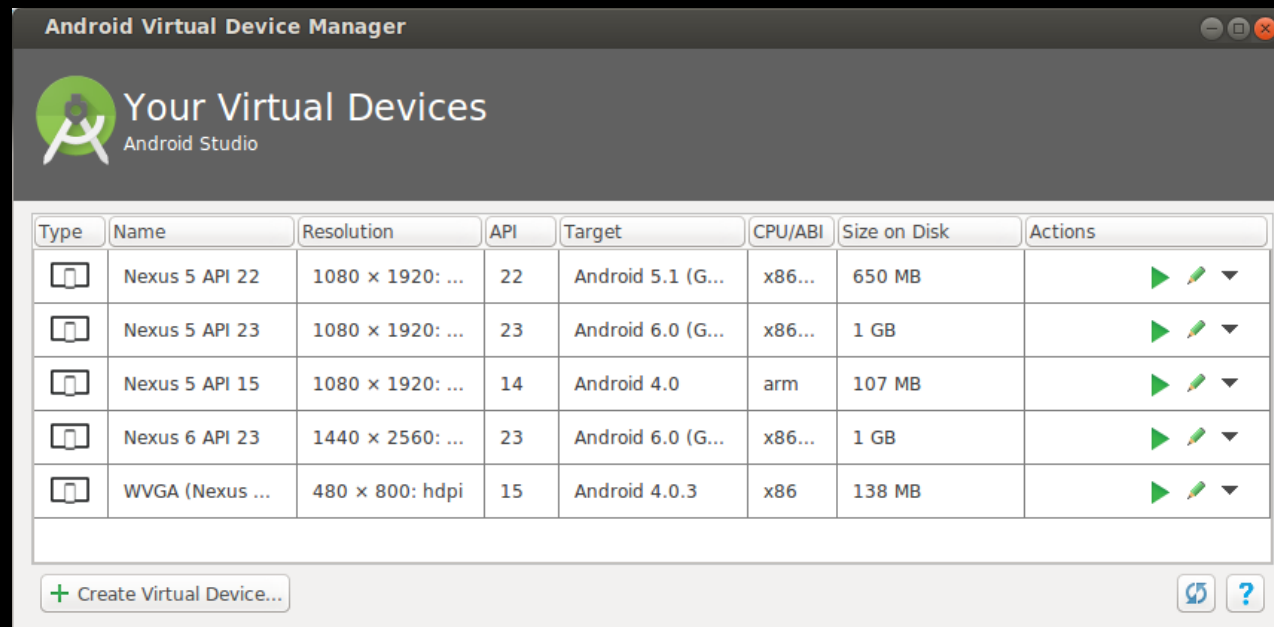
RUN YOUR APP

1. Run

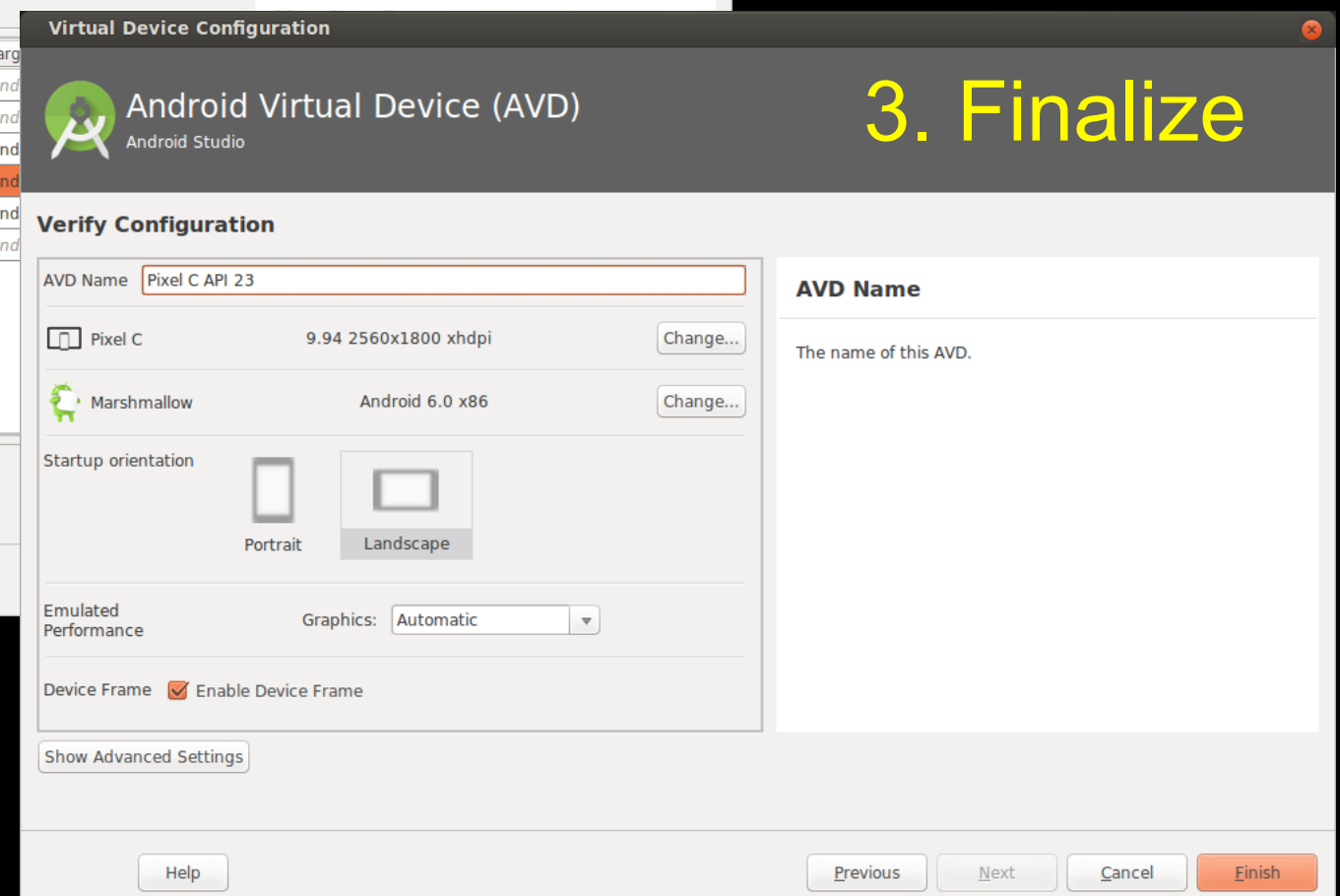
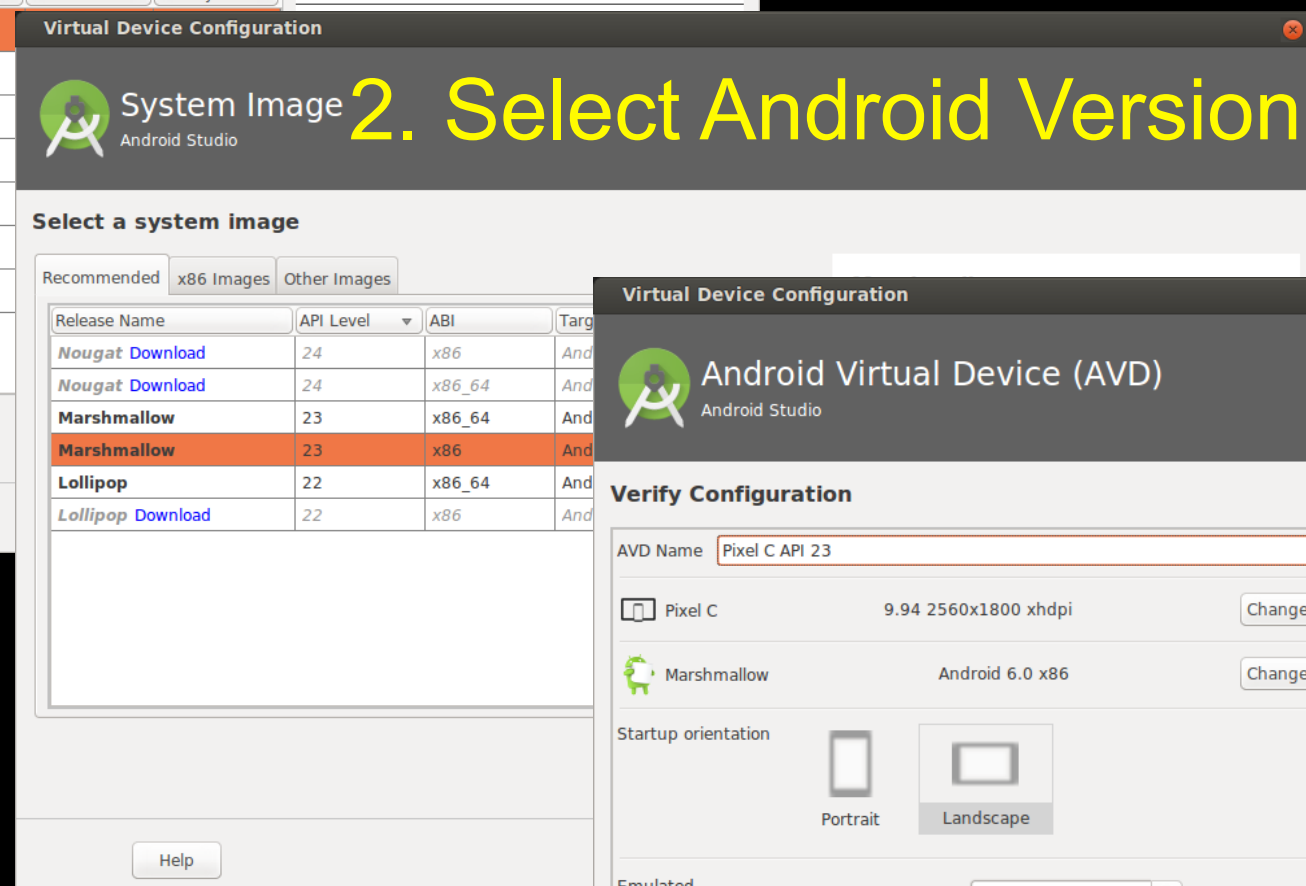
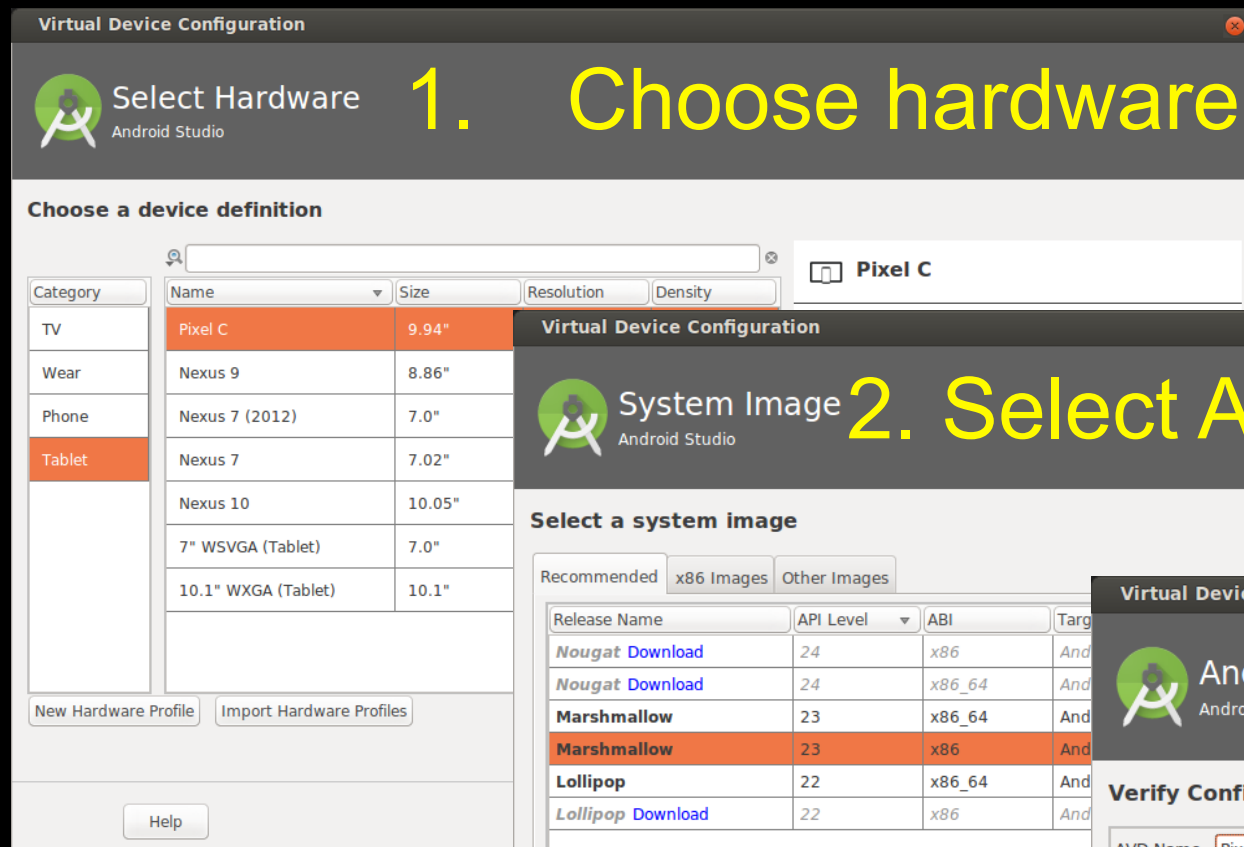


CREATE A VIRTUAL DEVICE

- Use emulators to test app on different versions of Android and form factors.
- Tools > Android > AVD Manager or:

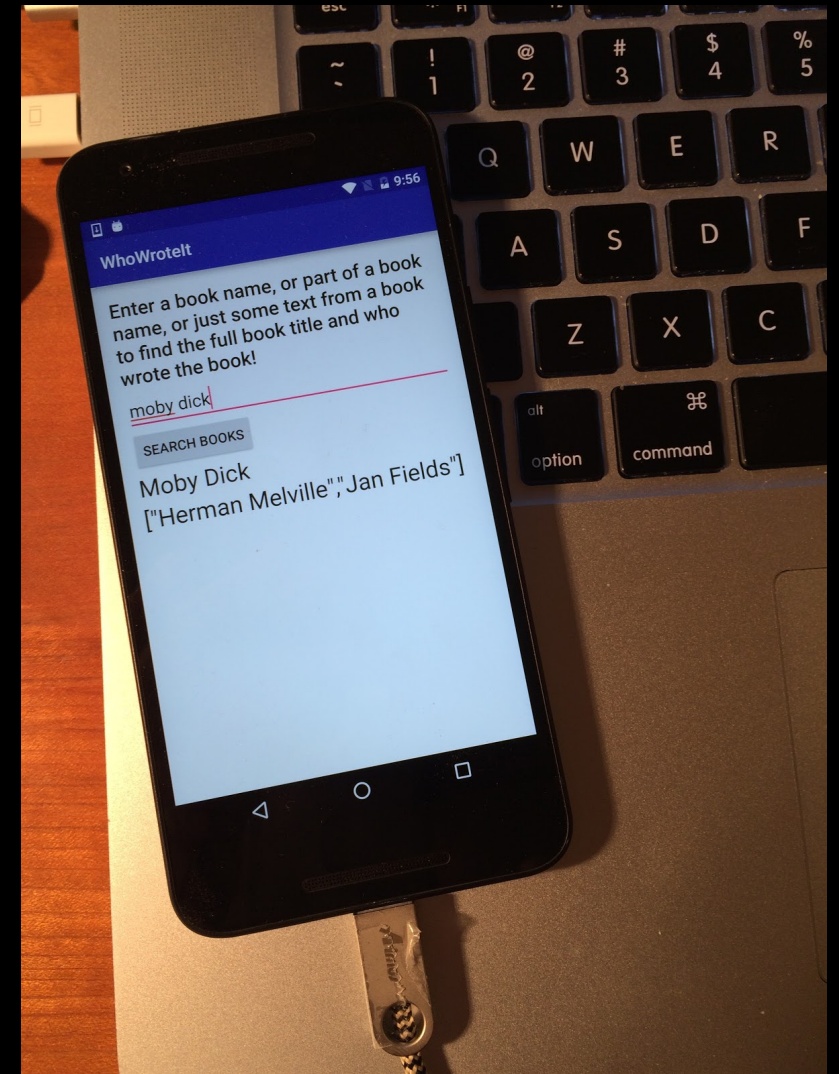


CONFIGURE VIRTUAL DEVICE



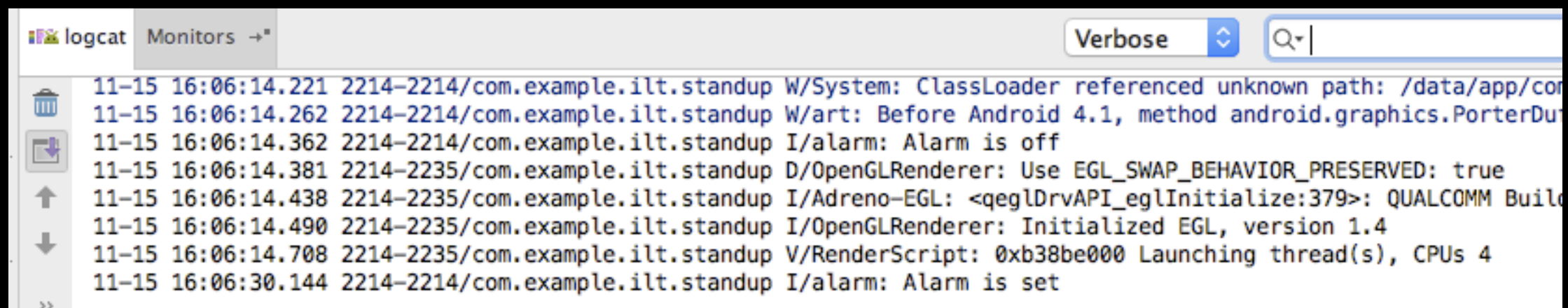
RUN ON A PHYSICAL DEVICE

- Turn on Developer Options:
 - Settings > About phone
 - Tap Build number seven times
- Turn on USB Debugging
 - Settings > Developer Options > USB Debugging
- Connect phone to computer with cable



GET FEEDBACK AS YOUR APP RUNS

- As the app runs, Android Monitor logcat shows information
- You can add logging statements to your app that will show up in logcat.



LOGGING

```
import android.util.Log;

// Use class name as tag

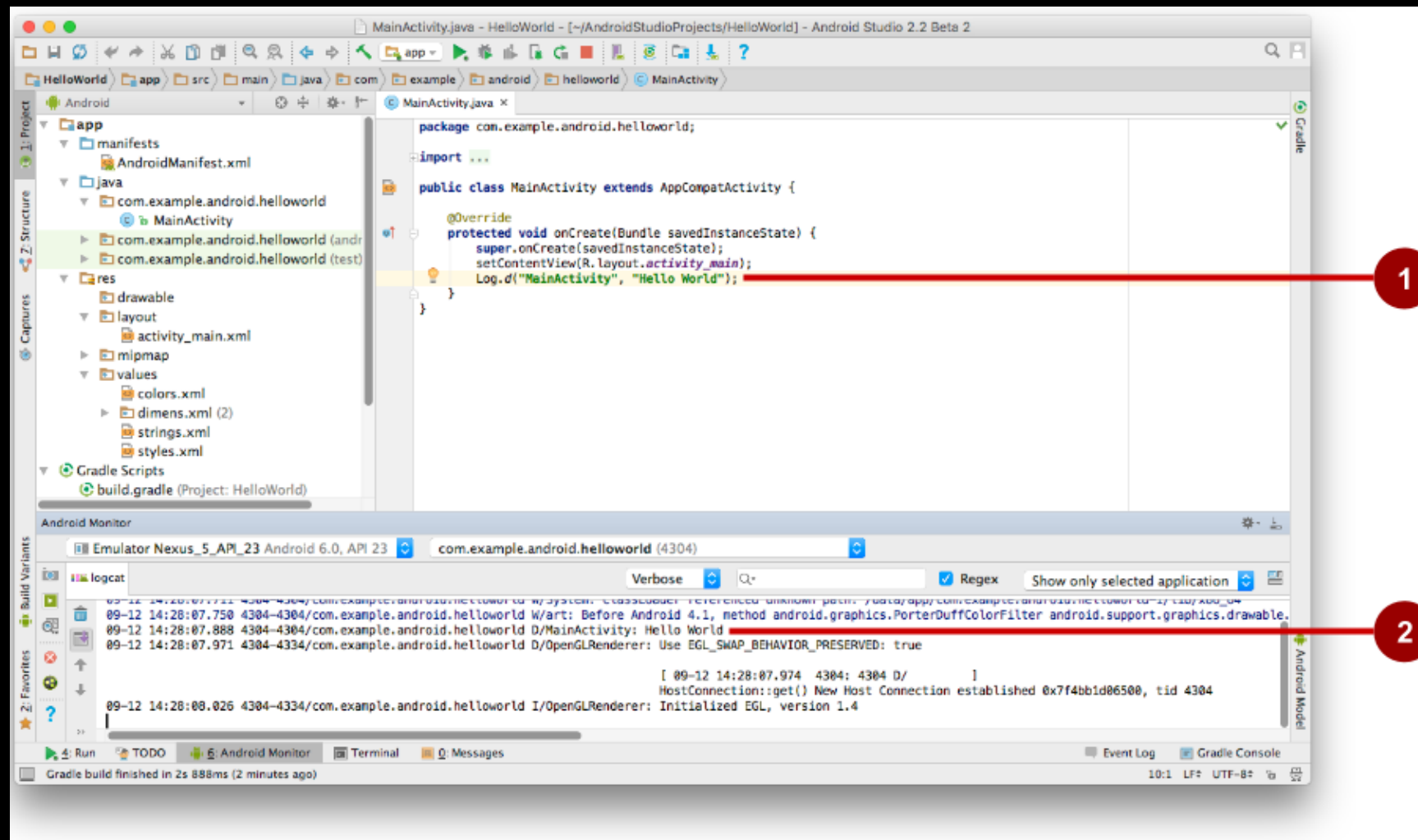
private static final String TAG =
    MainActivity.class.getSimpleName();

// Show message in Android Monitor, logcat pane

// Log.<log-level>(TAG, "Message");

Log.d(TAG, "Creating the URI...");
```


ANDROID MONITOR > LOGCAT PANE



1. Log statements in code.

2. logcat pane shows system and logging messages

DEMO

UPCOMING DEADLINES

- October 9th: Project Approval-in-principle & forming teams
- October 16th: Project proposal
- October 12th: First TSR

THE ULTIMATE
INSPIRATION
IS THE
DEADLINE
NOLAN BUSHNELL

QUESTIONS?

