# Getting Started with Android

Before we can run a simple Hello World App we need to install the programming environment. We will run Hello World on the Android Emulator (which runs very slowly) and on the phone (which is super-fast). The best resource for developing Android apps is [Android Developers](http://developer.android.com/index.html). We will use developers as a source for material for parts of this course.

## Get the Android Studio

Android Studio is the official Integrated Development Environment (IDE) for Android development. It includes everything you need to begin developing Android apps.

After installing Android Studio, you have:

* **IntelliJ IDEA + Android Studio plugin** (IntelliJ IDEA is a Java integrated development environment (IDE) for developing computer software)
* **Android SDK Tools** (Android SDK Tools is a component for the Android SDK. It includes the complete set of development and debugging tools for Android. It is included with Android Studio. The Android SDK provides you the API (application programming interface) libraries and developer tools necessary to build, test, and debug apps for Android.)
* **A version of the Android platform** (The Android platform is a platform for mobile devices that uses a modified Linux kernel. The Android Platform was introduced by the Open Handset in November of 2007.)
* **Android Platform-tools**
* **A version of the Android system image for the emulator**

Click on [downloaded Android Studio](https://developer.android.com/studio/index.html) to get started.

It will automatically detect your operating system and gives the correct Android Studio download link.

## Install Android Studio

Open <https://developer.android.com/studio/install.htm> (assuming Windows 64 bit is the target) (You should have already [downloaded Android Studio](https://developer.android.com/studio/index.html).) Follow the instructions in this link.

To install Android Studio on Windows, proceed as follows:

1. Launch the .exe file you downloaded.
2. Follow the setup wizard to install Android Studio and any necessary SDK tools.

There is a video in this link that shows each step of the recommended setup procedure.

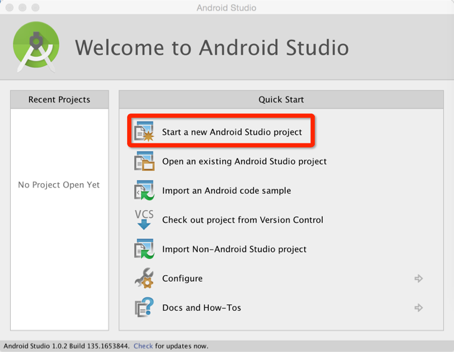
## What is in the Android SDK?

There are several packages associated with the Android SDK. If you look in the directory ~/Android/sdk you will see a number of sub directories associated with these packages. For more information of what's in these directories checkout [exploring the SDK](http://developer.android.com/tools/help/index.html)

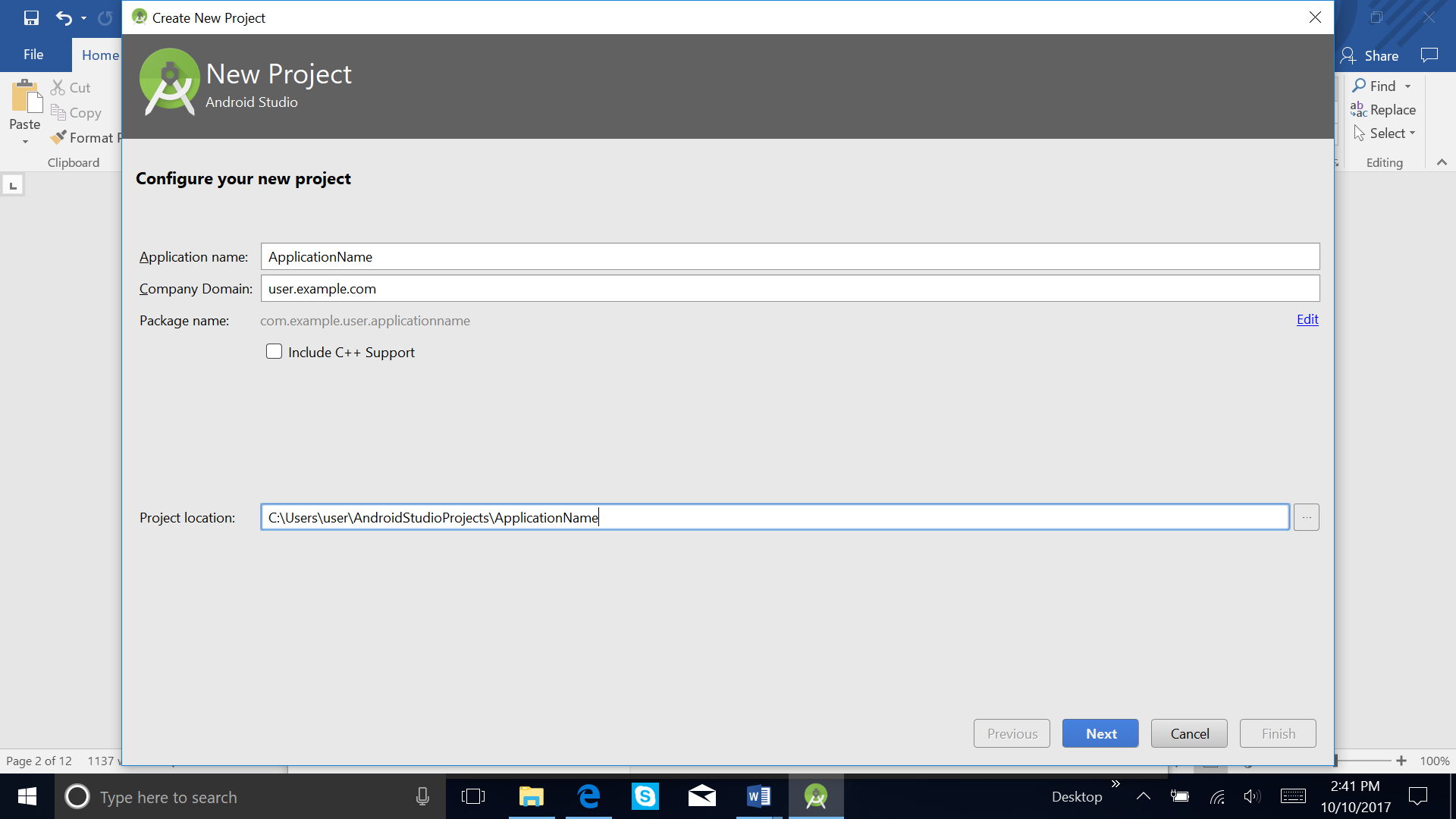
## Creating a New Android Project

To start a new Android app project for HelloWorld follow the wizard input; that is:

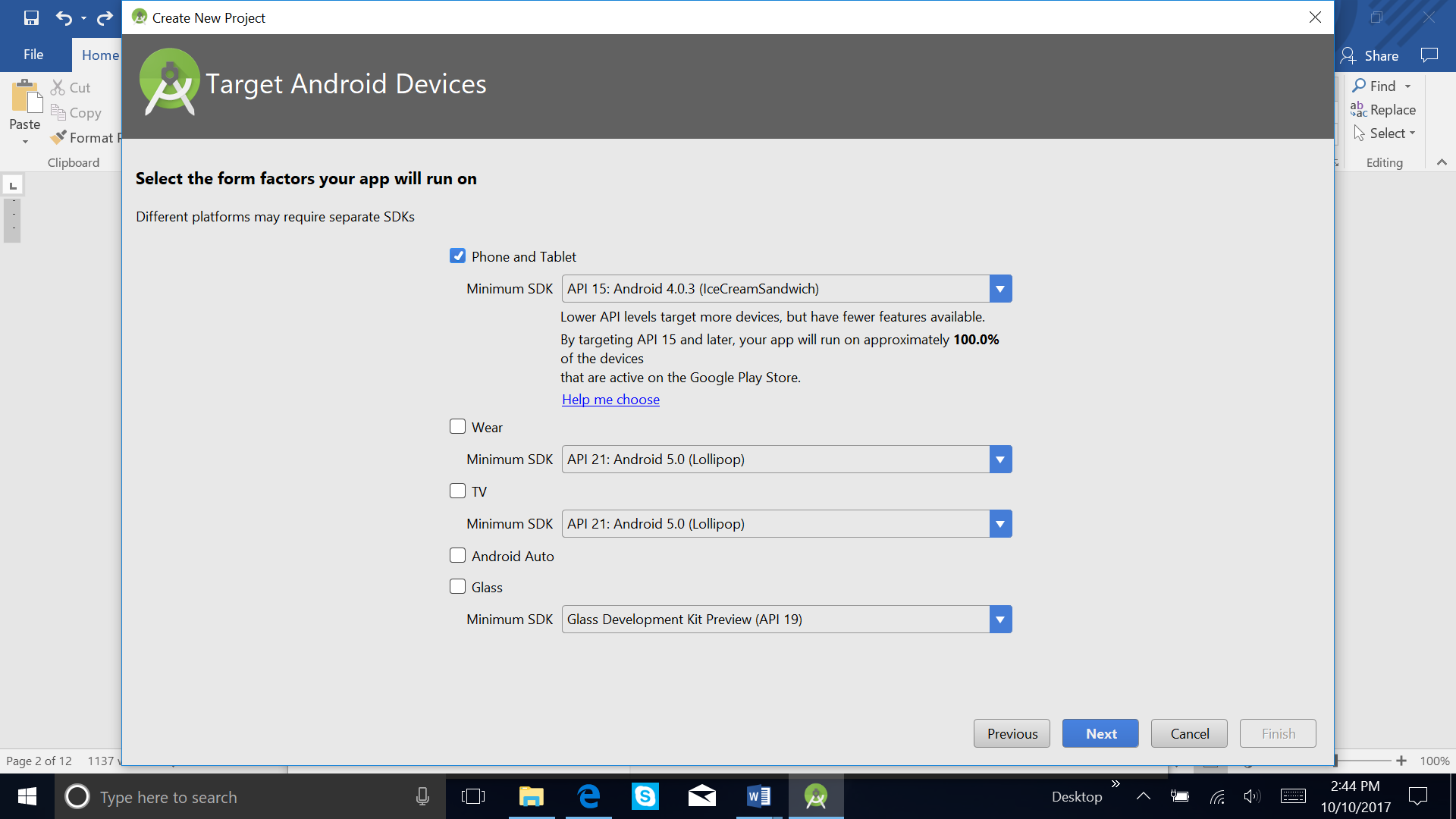
1. Click "Start a new Android Studio project" as shown below.



1. Fill in the application name, package name, and project location as shown below. The package name is the reverse of the company domain. If you want to customize the package name, click edit on the right of Package name. The project location is where your project files reside. You can choose any location you want. Click next when you are done.

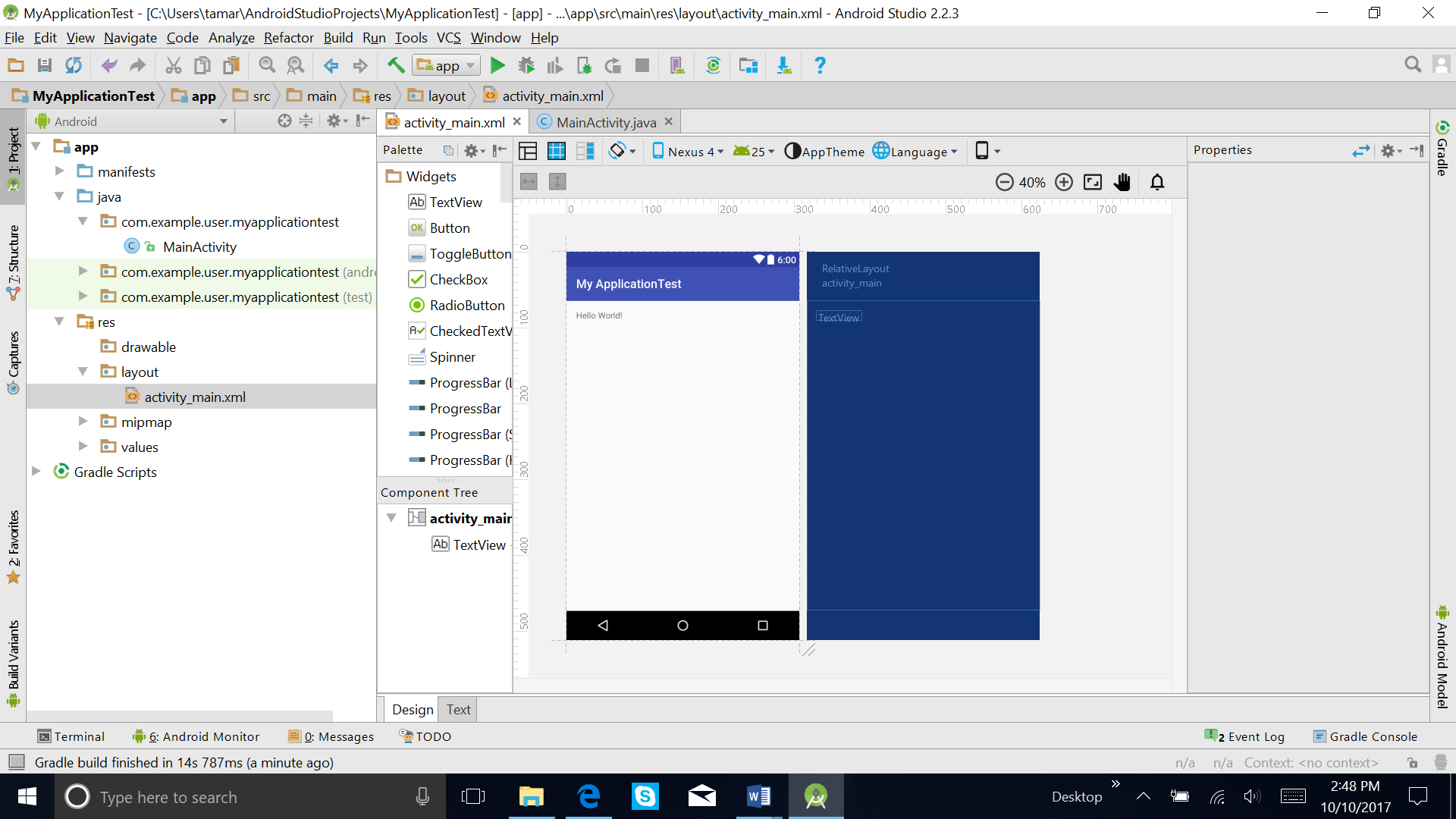


1. For Minimum Required SDK, just accept the default and click next



1. Create activity. Here you select an activity template for your app. Select Empty Activity and click Next.
2. Click finish

Once you have clicked finished you are presented with the project and all its files as shown below.



## MainActivity.java

Open the app/java folder and look at the MainActivity.java code.

**package** . . .;  
  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
  
**public class** MainActivity **extends** AppCompatActivity {  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
 }  
}

## Running HelloWorld App

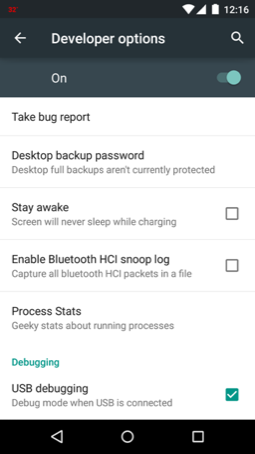
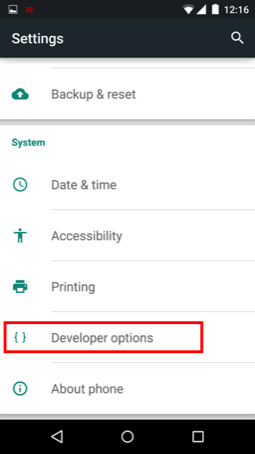
How you run your app depends on two things: whether you have a real Android-powered device and whether you're using Android emulator. This lesson shows you how to install and run your app on a real device and on the Android emulator.

## Running HelloWorld on your phone

Let's run our program directly on the phone. The big advantage of using the phone to develop is that it is fast to load and run programs. In contrast the emulator discussed next runs slowly in comparison and is irritating to use for a course -- but it is a good back up. Before we get started plug your phone into your laptop using the USB port, as shown below. Android Studio installs the Hello World app on your connected phone and starts it.



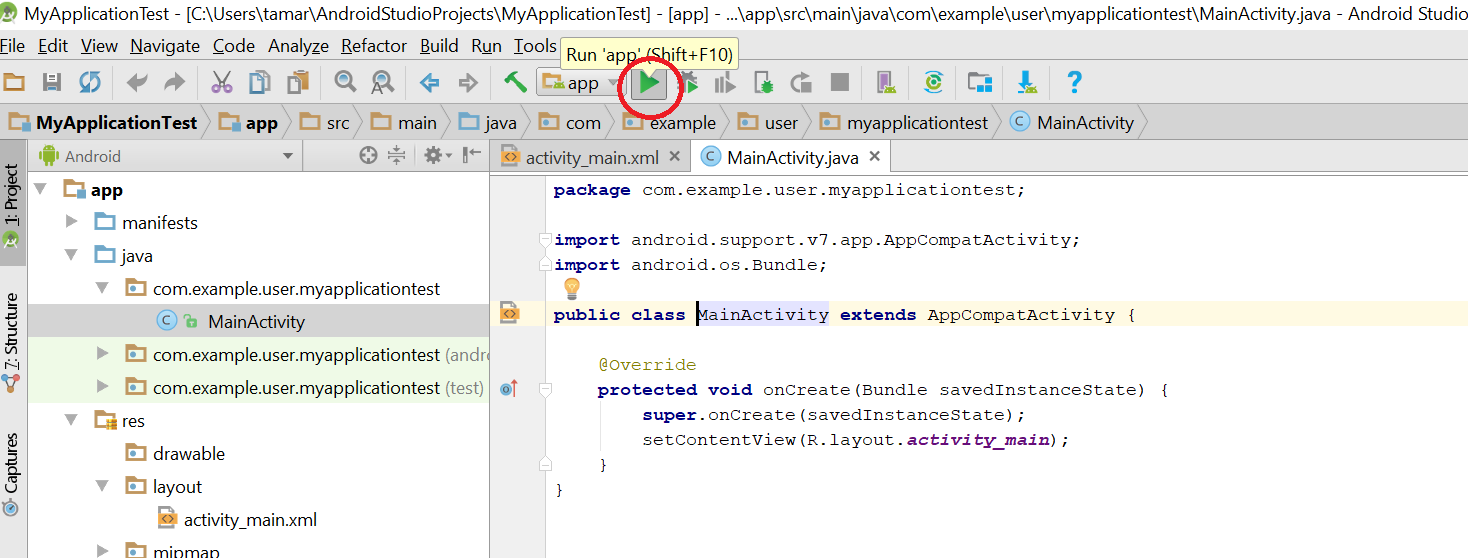
Next, go to settings on your phone and select **Developer option** and then select USB debugging -- this will allow Android Studio to communicate with the phone to program it.



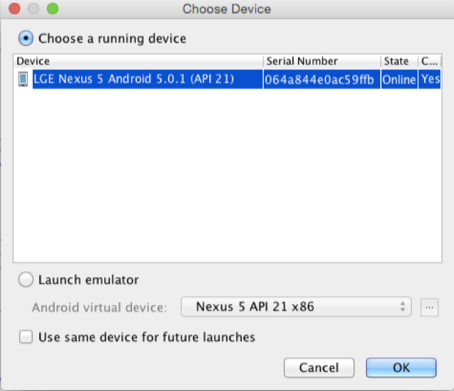
If you do not see developer option (circled on the image above) watch this [video](http://www.androidcentral.com/how-enable-developer-settings-android-42) and it mystery will be revealed.

Basically, the vendors and providers don't want that option accessible without enabling it -- for good reason. So as the video shows if you cannot see the developer options button, you should tap on the About phone button (under settings) and the click the Build number' button 7~8 times and voila -- now you see developer option. Now go in to developer option and select USB debugging.

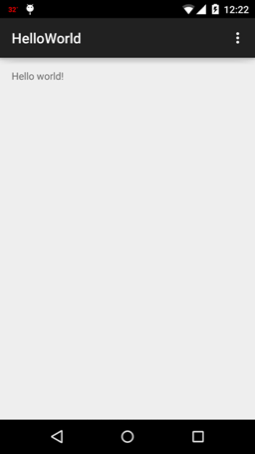
* Click on Run button showed in the toolbar below.



* Select Android Application and click OK.

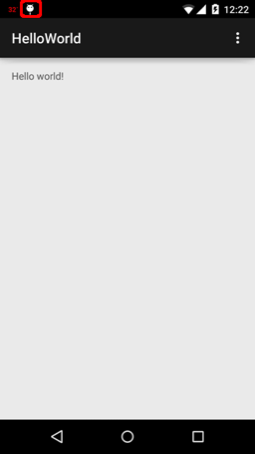


* The app is downloaded, installed and run on the phone -- you'll see the following Hello World on your phone.



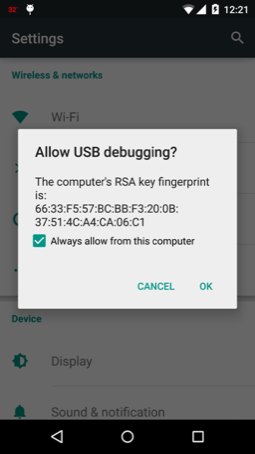
## If you app did not run

If for some reason you do not see the app running on the phone check that you see USB enabled in the status bar as indicated below



If it is not go, make sure you clicked and enabled the USB debugging option, as discussed above.

If it did not work, unplug the USB cable and then plug it in again. See if a dialog shows up on your phone. If it does, check "Always allow from this computer" and click **OK**, as shown below.

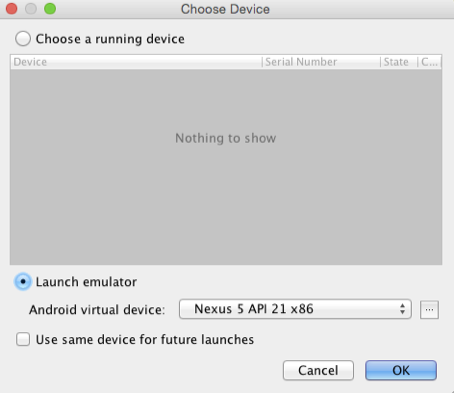


Now after re-launch the app, you should see you HelloWorld on the phone.

## Running HelloWorld on the Android Emulator

Now let's run the program on the emulator -- that's right it emulates the functionality of the phone as best it can. And for developers without a phone it allows them to run code, test and debug code. Code that runs on the emulator runs unchanged on the real device.

To run the app, disconnect your phone first, then click the **run** button in Android Studio. In **Choose Device** dialog, check **Launch emulator** and click **OK** as shown below. A device window will pop up. Android Studio will installs the app (the HelloWorld.apk) on your AVD (**Android** Virtual Device) and starts it.





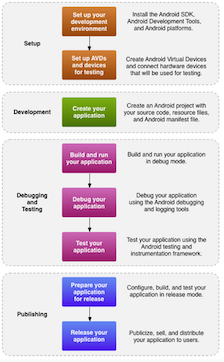
It can take some time to run the emulator. There are faster emulators out there but overall the one that comes with the ADT is just too slow to be useful.

Overall it took 5 minutes to start the emulator and run the app.

# The world of Android

## App development cycle

As coders you are aware of the development cycle of any software but the unique part of Android is the publishing part -- see the development cycle below.



### Some factoids on Androids

* Java: Android applications are written in Java
* filename.apk: The Android SDK tools compile the code—along with any data and resource files (layouts, strings)—into an **Android PacKage** (filename.apk). When you download an app from Google Play or install it yourself (from email, using ADB(Android Debug Bridge)) you are installing the apk file.
* UI design: Android separates the UI design from the program so they can be developed independently. In this manner you could theoretically completely change the UI without changing a line of Java code (not true in reality). The UI is designed using a graphical tool or **Extensible Markup Language - XML**:``XML is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable''
* Publishing is the general process that makes your Android applications available to users. When you publish an Android application you perform two main tasks:
* You prepare the application for release.

During the preparation step you build a release version of your application, which users can download and install on their Android-powered devices.

* You release the application to users.

Usually, you release your application through an application marketplace, such as Google Play. However, you can also release applications by sending them directly to users or by letting users download them from your own website.

## Application

You can specify a custom application class in your Android manifest file.

### What's in an app: application components

When you develop apps, you will be using five key building blocks -- some apps may have all these components and others not -- really depends on what type of app you are planning on building:

* **activities**: An activity is the key execution component. You need at least one activity in an app that deals with the UI but you probably will have multiple activity. This is the entry point to the application -- think of main() in C. An activity might handle one or more screens in terms of UI control.
* **fragments**: Fragments are new to Android but very important in programming the UI. We can think of them as mini-activities. In implementation there are considerable differences; for example, an activity has to be specified in the manifest, while fragments do not because they can only exist in the context of an activity and are coupled to the activity state. While fragments are strongly integrated into the UI you can use fragments without a view or UI components. For now, we will just consider fragments when applied to the UI.
* **services**: Typically, services are long running programs that don't need to interact with the UI. A service you create is implemented as a subclass of **Service**.
* **content providers**: Apps share data. Content providers can help an application manage access to data stored by itself, stored by other apps, and provide a way to share data with other apps. They encapsulate the data, and provide mechanisms for defining data security. A content provider is implemented as a subclass of **ContentProvider**.
* **broadcast receivers**: A *broadcast receiver* (*receiver*) is an Android component which allows you to register for system or application events. All registered receivers for an event are notified by the Android runtime once this event happens. For example, applications can register for the ACTION\_BOOT\_COMPLETED system event which is fired once the Android system has completed the boot process. A broadcast receiver is implemented as a subclass of **BroadcastReceiver** and each broadcast is delivered as an Intent object.

## Android software framework

Each Android app runs in its own security sandbox (In computer security, a **sandbox** is a security mechanism for separating running programs, usually in an effort to mitigate system failures or software vulnerabilities from spreading):

The Android operating system is a multi-user Linux system in which each application is a different user.

By default, the system assigns each application a unique Linux user ID (the ID is used only by the system and is unknown to the application). The system sets permissions for all the files in an application so that only the user ID assigned to that application can access them.

Each process has its own virtual machine (VM), so an application's code runs in isolation from other applications.

By default, every application runs in its own Linux process. Android starts the process when any of the application's components need to be executed, then shuts down the process when it's no longer needed or when the system must recover memory for other applications.

## References:

<https://www.codeproject.com/Articles/798780/Introduction-to-Android>

<https://developer.android.com/index.html>

## <https://developer.android.com/studio/index.html>

<http://developer.android.com/tools/help/index.html>

https://developer.android.com/studio/publish/index.html#publishing-prepare

## Assignment

Download and Install Android Studio in your computer. Make Hello World to run in virtual device (emulator) and in real device.