**Installing Android Studio (version 2.3.0 and above ) on 64-bit Windows**

Launched android-studio-bundle-143.2821654-windows.exe to start the installation process. The installer responded by presenting the Android Studio Setup dialog box shown in Figure 1.



Figure 1. Set up Android Studio

Clicking Next took me to the following dialog box, which gives you the option to decline installing the Android SDK (included with the installer) and an Android Virtual Device (AVD).

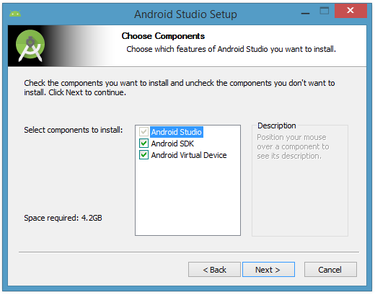


Figure 2. Do you want to install the Android SDK and AVD?

I chose to keep the default settings. After clicking Next, you'll be taken to the license agreement dialog box. Accept the license to continue the installation.

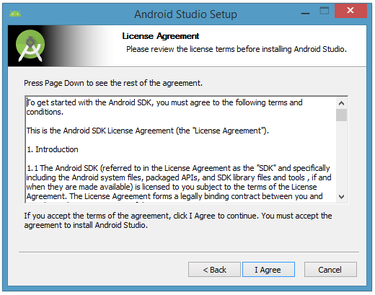


Figure 3. Accept the license agreement to continue installation

The next dialog box invites you to change the installation locations for Android Studio and the Android SDK.

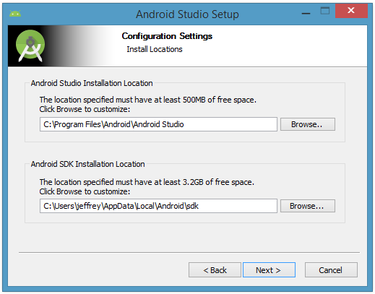


Figure 4. Set the Android Studio and Android SDK installation locations

Change the location or accept the default locations and click Next.

The installer defaults to creating a shortcut for launching this program, or you can choose to decline. I recommend that you create the shortcut, then click the Install button to begin installation.

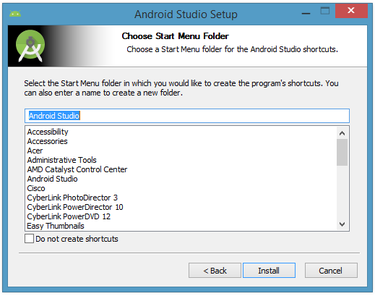


Figure 5. Create a new shortcut for Android Studio

The resulting dialog box shows the progress of installing Android Studio and the Android SDK. Clicking the Show Details button will let you view detailed information about the installation progress.

The dialog box will inform you when installation has finished. When you click Next, you should see the following:

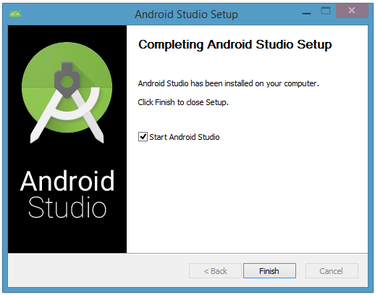


Figure 6. Leave the Start Android Studio check box checked to run this software

To complete your installation, leave the Start Android Studio box checked and click Finish.

**Running Android Studio**

Android Studio presents a splash screen when it starts running:



Figure 7. Android Studio's start screen

On your first run, you'll be asked to respond to several configuration-oriented dialog boxes. The first dialog box focuses on importing settings from any previously installed version of Android Studio.

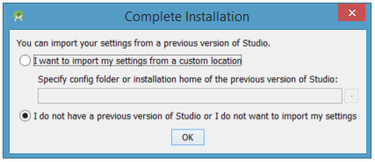


Figure 8. Import settings

If you're like me, and don't have a previously installed version, you can just keep the default setting and click OK. Android Studio will respond with a slightly enhanced version of the splash screen, followed by the Android Studio Setup Wizard dialog box:

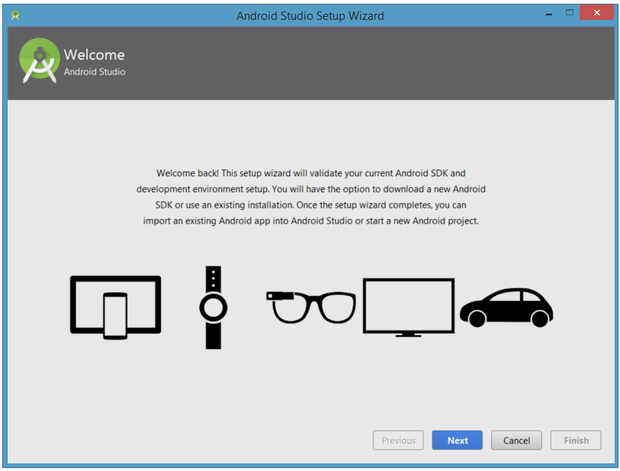


Figure 9. Validate your Android SDK and development environment setup

When you click Next, the setup wizard invites you to select an installation type for your SDK components. For now I recommend you keep the default standard setting.

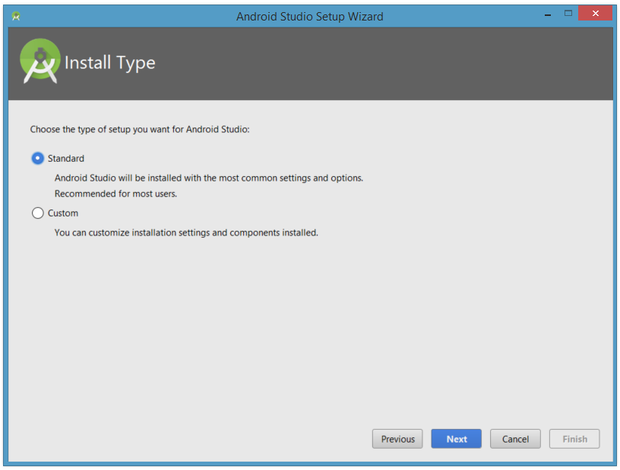


Figure 10. Choose an installation type

Click Next and verify your settings, then click Finish to continue.

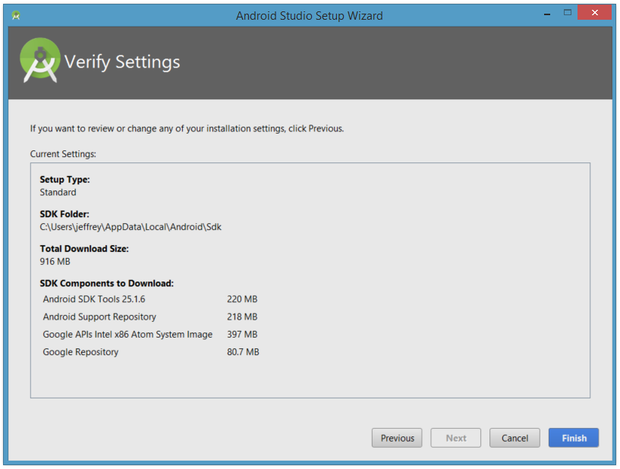


Figure 11. Review settings

The wizard will download and unzip various components. Click Show Details if you want to see more information about the archives being downloaded and their contents.

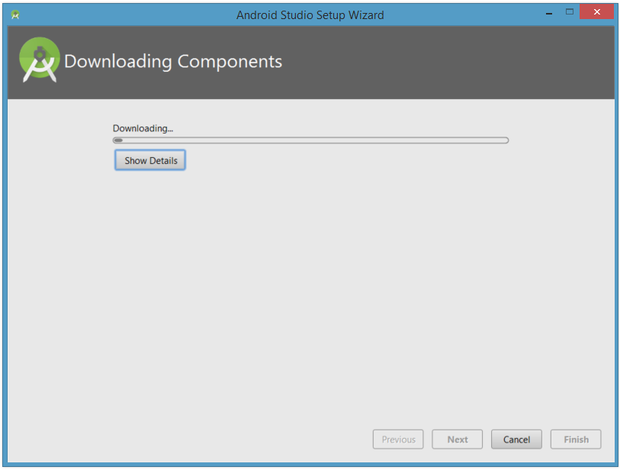


Figure 12. The wizard downloads and unzips Android Studio components

If your computer isn't Intel based, you might get an unpleasant surprise after the components have completely downloaded and unzipped:

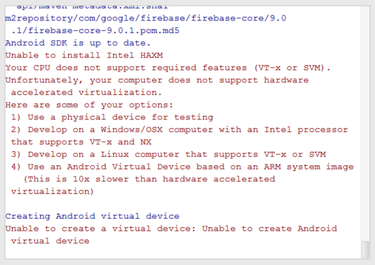


Figure 13. Intel-based hardware acceleration is unavailable

Your options are to either put up with the slow emulator or use an Android device to speed up development. I'll discuss the latter option later in the tutorial.

Finally, click Finish to complete the wizard. You should see the Welcome to Android Studio dialog box:

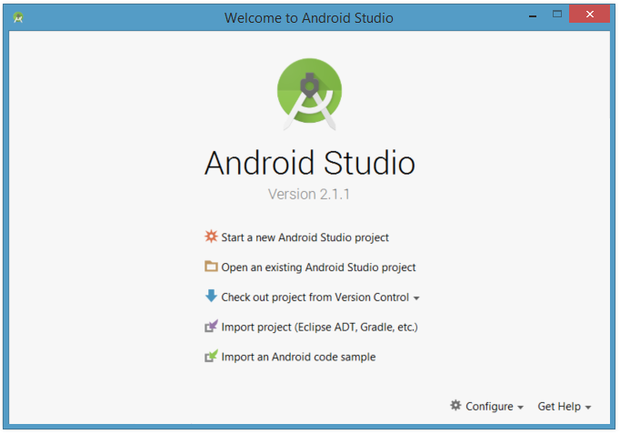


Figure 14. Welcome to Android Studio

You'll use this dialog to start up a new Android Studio project, work with an existing project, and more. You can access it anytime by double-clicking the Android Studio shortcut on your desktop.

**Your first Android Studio mobile app**

The quickest way to get to know Android Studio is to use it to develop an app. We'll start with a variation on the "Hello, World" application: a little mobile app that displays a "Welcome to Android" message.

In the steps that follow, you'll start a new Android Studio project and get to know the project workspace, including the project editor that you'll use to code the app in Part 2.

**Starting a new project**

From our setup so far, you should still have Android Studio running with the Welcome to Android Studio dialog box. From here, click Start a new Android Studio project. Android Studio will respond with the Create New Project dialog box shown in Figure 15.

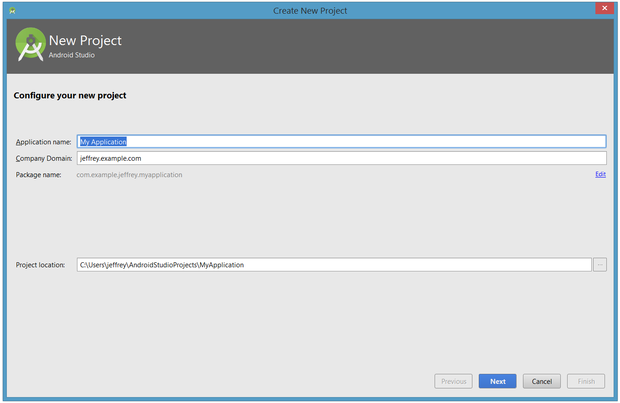


Figure 15. Create a new project

Enter *W2A* (Welcome to Android) as the application name and *javajeff.ca* as the company domain name. You should then see C:\Users\jeffrey\AndroidStudioProjects\W2A as the project location. Click Next to select your target devices.

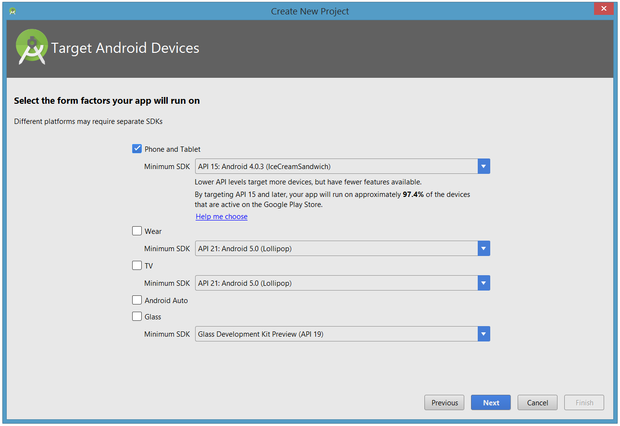


Figure 16. Select your target device categories

Android Studio lets you select *form factors*, or categories of target devices, for every app you create. I would have preferred to keep the default API 15: Android 4.0.3 (IceCreamSandwich) minimum SDK setting (under Phone and Tablet), which is supported by my Amazon Kindle Fire HD tablet. Because Android Studio doesn't currently support this API level (even when you add the 4.0.3 system image via the SDK Manager), I changed this setting to API 14: Android 4.0 (IceCreamSandwich), which is also supported by my tablet.

Click Next, and you will be given the opportunity to choose a template for your app's main activity. For now we'll stick with Empty Activity. Select this template and click Next.

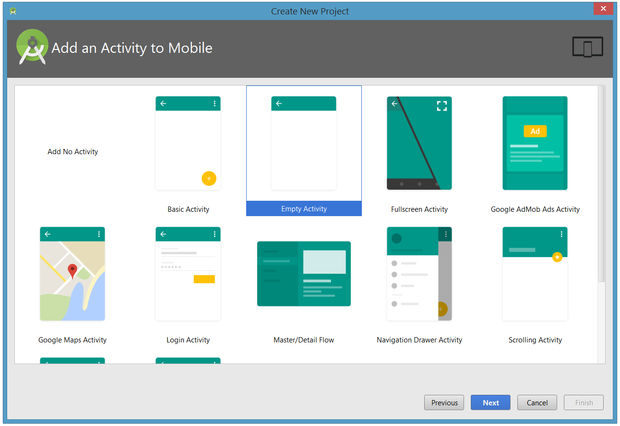


Figure 17. Specify an activity template

Next you'll customize the activity:

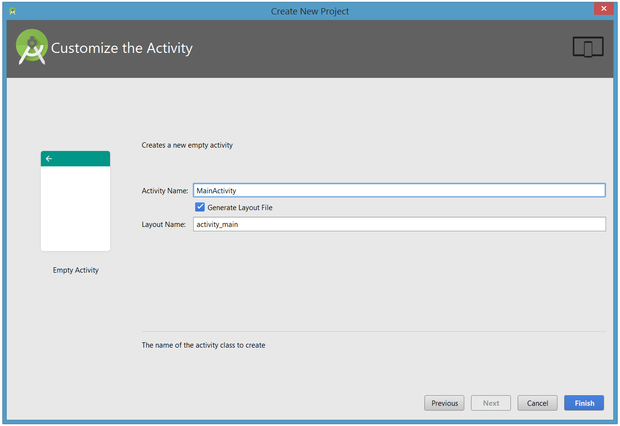


Figure 18. Customize your activity

Enter *W2A* as the activity name and *main* as the layout name, and click Finish to complete this step. Android Studio will respond that it is creating the project, then take you to the project workspace.

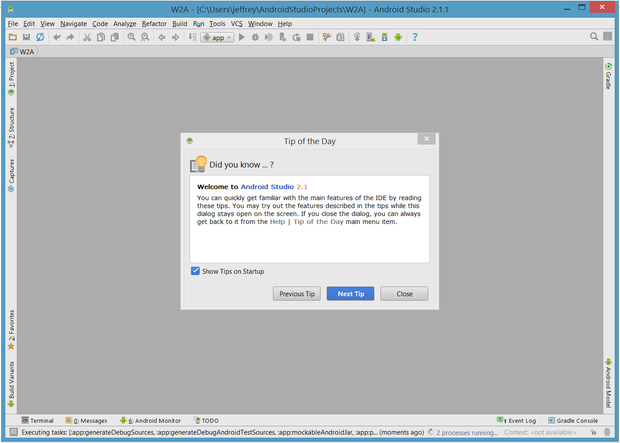


Figure 19. Android Studio workspace

The project workspace is organized around a menu bar, a tool bar, a work area, additional components that lead to more windows (such as a Gradle Console window), and a status bar. Also note the Tip of the Day dialog box, which you can disable if you like.

**Accessing AVD Manager or SDK Manager from menu and tool bar**

To access the traditional AVD Manager or SDK Manager, select Android from the Tools menu followed by AVD Manager or SDK Manager from the resulting pop-up menu (or click their tool bar icons).

**The project and editor windows**

When you enter the project workspace, W2A is identified as the current project, but you won't immediately see the project details. After a few moments, these details will appear in two new windows.

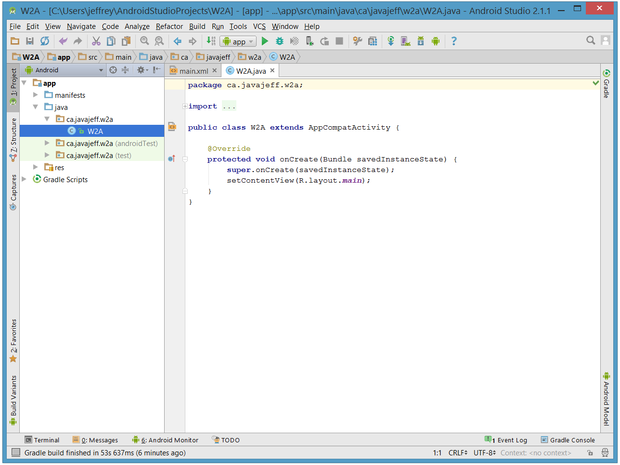


Figure 20. The project and editor windows

The project window is organized into a tree whose main branches are App and Gradle Scripts. The App branch is further organized into manifests, java, and res subbranches:

* manifests stores AndroidManifest.xml, which is an XML file that describes the structure of an Android app. This file also records permission settings (where applicable) and other details about the app.
* java stores an app's Java source files according to a package hierarchy, which is ca.javajeff.w2a in this example.
* res stores an app's resource files, which are organized into drawable, layout, mipmap, and values subbranches:
  + drawable: an initially empty location in which to store an app's artwork
  + layout: a location containing an app's layout files; initially, main.xml(the main activity's layout file) is stored here
  + mipmap: a location containing various ic\_launcher.png files that store launcher screen icons of different resolutions
  + values: a location containing colors.xml, dimens.xml, strings.xml, and styles.xml

The Gradle Scripts branch identifies various .gradle (such as build.gradle) and .properties (such as local.properties) files that are used by the Gradle-based build system.