**Laboratory Project 1: Part 1- Android Installation**

In this project we will learn how to install the Android SDK and set up your development environment.

1. **Installation of the Android SDK**

Please follow the below link to do so: [Download the SDK](http://code.google.com/android/download.html). Select the "zip" archive version corresponding to your operating system.

After downloading the SDK, unpack the .zip archive to a suitable location on your machine. By default, the SDK files are unpacked into a directory named android\_sdk\_*<release>\_<platform>* (e.g. android-sdk\_r08-windows). The directory contains the subdirectories tools/, samples/, and others.

Make a note of the name and location of the unpacked SDK directory on your system — you will need to refer to the SDK directory later, when setting up the Android plugin or using SDK tools.

Optionally, you can add the path to the SDK tools/ directory to your path. This would allow you to access the emulator, debugger (adb), SD card image creator and a number of other tools from the commandline. As mentioned above, the tools/ directory is located in the SDK directory.

* On Linux, edit your ~/.bash\_profile or ~/.bashrc file. Look for a line that sets the PATH environment variable and add the full path to the tools/ directory to it. If you don't see a line setting the path, you can add one:

export PATH=${PATH}:*<your\_sdk\_dir>*/tools

[if you used a console editor like vi, type the following after exiting editor: source ~/.bash\_profile (or ~/.bashrc)]

* On a Mac, look in your home directory for .bash\_profile and proceed as for Linux. You can create the .bash\_profile, if you haven't already set one up on your machine.
* On Windows, right click on **My Computer**, and select **Properties**. Under the **Advanced** tab, hit the **Environment Variables** button, and in the dialog that comes up, double-click on **Path** under **System Variables**. Add a ';' character to the current path list if it does not end with one. Append the full path to the tools/ directory to the PATH variable.

Note that, if you update your SDK, you should remember to update your PATH settings to point to the new location, if different.

1. **Installing the Java** [JDK](http://java.sun.com/javase/downloads/index.jsp) **(JRE alone is not sufficient)**
   * Follow the above link and click on the "Download JDK" button.
   * In the opened window select from the drop down menu the operating system of your choice (your Platform – Windows, Linux), check the box next to License Agreement statement and click Continue.
   * Mac OS X is bundled with an older version of JDK. If eclipse report errors when running Android applications, it may be necessary to update the JDK through Apple Update. New releases of JDK may also be obtained through [Apple Developer website](http://connect.apple.com/cgi-bin/WebObjects/MemberSite.woa/wo/5.1.17.2.1.3.3.1.0.1.1.0.3.9.3.3.1) (requires a student developer license).
   * Once the file is downloaded to your machine, start it and follow the onscreen instructions for installing. If installing on Linux, in console type chmod +x *<install file>* before running the file.
2. **Installing supporting environment – Eclipse IDE**

Next we need to download and install Eclipse IDE environment for easier application creation, debugging and emulation.

* Follow the Eclipse IDE link: [Eclipse](http://www.eclipse.org/downloads/) - 3.4 (Ganymede) or 3.5 (Galileo) are the latest supported versions (3.6 currently has compatibility issue with Android SDK). You may need to click on the "Older Versions" link to find these releases. Note that Linux users should not use the Eclipse package in the distribution's package repository, as these may be packaged differently.
* Next, in the window that opened select one of the mirror sites you want to download from or just click on the link to the right of “Download from:”
* Download the zipped version of the Eclipse and unzip it in the desired directory.
* Note that in the downloaded Eclipse distribution, a JDT plugin is included (these are necessary for Eclipse).

Next we need to install the web tools.

1. **Installing Web Tools**

Download the Web Tools following this [link](http://download.eclipse.org/webtools/downloads/). We want to download just the "Web App Developers (wtp)" build from the "Traditional Zip Files for Web Tools Platform" section. [Version 3.2.2](http://www.eclipse.org/downloads/download.php?file=/webtools/downloads/drops/R3.2.2/R-3.2.2-20100915173744/wtp-R-3.2.2-20100915173744.zip) was tested with this guide.

Unzip in a temporary folder and copy everything under the ‘eclipse\’ directory over to the eclipse installation under the ‘eclipse\’ directory. Mac users should be careful not to "Replace" original files, but "Merge" with the existing directories. You may need to individually copy paste the folders.

1. **Installing Andorid Development Tools (ADT)**

Since you will be using the Eclipse IDE as your environment for developing Android applications, you can install a custom plugin called Android Development Tools (ADT), which adds integrated support for Android projects and tools. The ADT plugin includes a variety of powerful extensions that make creating, running, and debugging Android applications faster and easier.

To download and install the ADT plugin, follow the steps below for your respective Eclipse version.

**Steps for** **Eclipse 3.4 (Ganymede)**

1. Start Eclipse, then select **Help** > **Software Updates...**.
2. In the dialog that appears, click the **Available Software** tab.
3. Click **Add Site...**
4. Enter this as the Location:

https://dl-ssl.google.com/android/eclipse/

1. Back in the Available Software view, you should see the plugin. Select the checkbox next to *Developer Tools* and click **Install...**
2. On the subsequent Install window, "Android Developer Tools", and "Android Editors" should both be checked. The Android Editors feature is optional, but recommended. If you choose to install it, you need the WST plugin mentioned earlier in this page.  
   Click **Finish**.
3. Restart Eclipse.

**Steps for** **Eclipse 3.5 (Galileo)**

1. Start Eclipse, then select **Help** > **Install New Software**
2. In the dialog that appears, click **Add...**
3. Enter this as the Location:

https://dl-ssl.google.com/android/eclipse/

1. Back in the Available Software view, you should see the plugin. Select the checkbox next to *Developer Tools* and click **Next>**
2. Restart Eclipse.

After restart, **update your Eclipse preferences** to point to the SDK directory:

1. Select **Window** > **Preferences...** to open the Preferences panel. (Mac OS X: **Eclipse** > **Preferences**)
2. Select **Android** from the left panel.
3. For the SDK Location in the main panel, click **Browse...** and locate the SDK directory.
4. Click **Apply**, then **OK**. The Android SDK version should now be listed in the Targets panel when you reopen the preferences window.
5. **Installing the SDK Platform and Virtual Devices**

The last step in setting up the Android SDK is to install the SDK platform, and setup the virtual phone device (emulator).

**Steps for** **Eclipse 3.4/3.5**

1. Select **Window** > **Android SDK and AVD Manager**
2. Click on **Available Packages** on the left in the manager window
3. Expand **Android Repository**, and select **SDK Platform Android *<Version>***
4. Click **Install Selected**
5. Once installed, click on **Virtual Devices** on the left in the manager window
6. Click on **New**
7. Enter a name, and select a **Target** that matches your SDK version. Click **Create AVD.**

Now we are ready for Android development. The guide below describes ways to troubleshoot, update and migrate your Android SDK installation and project.

1. **Troubleshooting ADT Installation**

If you are having trouble downloading the ADT plugin after following the steps listed above, here are some suggestions:

* Try changing the remote update site URL to use http, rather than https.
* If you are behind a firewall (such as a corporate firewall), make sure that you have properly configured your proxy settings in Eclipse. In Eclipse 3.4/3.5, you can configure proxy information from the main Eclipse menu in **Window** (on Mac, **Eclipse**) > **Preferences** > **General** > **Network Connections.**

If you are still unable to use Eclipse to download the ADT plugin, follow these steps to download and install the plugin from your computer:

1. [Download the ADT zip file](http://developer.android.com/sdk/index.html) (do not unpack it).
2. Follow steps 1 and 2 in the default install instructions (above).
3. In Eclipse 3.4, click **Add Site...**(called **Add...** in Eclipse 3.5), then **Archive...**
4. Browse and select the downloaded the zip file.
5. Follow the remaining procedures, above, starting from steps 5.

Note that to update your plugin, you will have to follow these steps again instead of the default update instructions.

Note that the "Android Editors" feature of ADT requires several optional Eclipse components (for example, WST). If you encounter an error when installing ADT, your Eclipse installation might not include those components. For information about how to quickly add the necessary components to your Eclipse installation, see the troubleshooting topic [ADT Installation Error: "requires plug-in org.eclipse.wst.sse.ui"](http://developer.android.com/guide/appendix/faq/troubleshooting.html#installeclipsecomponents).

**8. Updating the ADT Plugin**

In some cases, a new ADT plugin may become available for your existing version of the SDK. You can use the steps below to update the ADT plugin from inside Eclipse.

Steps for **Eclipse 3.4/3.5**

1. Select **Help** > **Software Updates...** if using 3.4, or **Help > Check for Updates...** in 3.5
2. Select the **Installed Software** tab.
3. Click **Update...**
4. If an update for ADT is available, select it and click **Finish**.

# 9. Upgrading the SDK

Sometimes a newer version of the Android SDK becomes available. Follow the instructions below to upgrade the SDK and corresponding software.

## 9.1 Install the new SDK

After unpacking the new SDK, you should:

* Wipe your emulator data.

Some data formats have changed since the last SDK release, so any previously saved data in your emulator must be removed. Open a console/terminal and navigate to the /tools directory of your SDK. Launch the emulator with the -wipe-data option.

Windows: emulator -wipe-data  
Mac/Linux: ./emulator -wipe-data

* Update your PATH variable (Only if you have them set from Section 1).

If you had previously setup your PATH variable to point to the SDK tools directory, then you'll need to update it to point to the new SDK. E.g., for a .bashrc or .bash\_profile file: export PATH=$PATH:<your\_new\_sdk\_dir>/tools

## 9.2 Update your ADT Eclipse Plugin

Follow the directions above (Section 8) to update the ADT plugin.

## 10. Set up Application Signing

All applications must now be signed before you can install them on the emulator. Both the ADT plugin and the Ant-based build tools support this requirement by signing compiled .apk files with a debug key. To do so, the build tools use the Keytool utility included in the JDK to create a keystore and a key with a known alias and password. For more information, see [Signing Your Applications](http://code.google.com/android/intro/develop-and-debug.html).

To support signing, you should first make sure that Keytool is available to the SDK build tools. In most cases, you can tell the SDK build tools how to find Keytool by making sure that your JAVA\_HOME environment variable is set and that it references a suitable JDK. Alternatively, you can add the JDK version of Keytool to your PATH variable (Refer to Section 1 for details on how to set environment variables).

If you are developing on a version of Linux that originally came with Gnu Compiler for Java or openJDK, make sure that the system is using the JDK version of Keytool. If keytool is already in your PATH, it might be pointing to a symlink at /usr/bin/keytool. In this case, relink to the currect keytool:

sudo mv keytool keytool.bak

sudo ln -s /*<path to official JDK>/bin/keytool /usr/bin/keytool*

If you use Ant to build your .apk files (rather than ADT for Eclipse), you must regenerate your build.xml file. To do that, follow these steps:

1. In your Android application project directory, locate and delete the current build.xml file.
2. Run activitycreator, directing output to the folder containing your application project.

- exec activitycreator --out <project folder> your.activity.YourActivity

Run in this way, activityCreator will not erase or create new Java files (or manifest files), provided the activity and package already exists. It is important that the package and the activity are real. The tool creates a new build.xml file, as well as a new directory called "libs" in which to place 3rd jar files, which are now automatically handled by the Ant script.

## 11. Migrate your applications

After updating your SDK, you will likely encounter breakages in your code, due to framework and API changes. You'll need to update your code to match changes in the Andriod APIs.

One way to start is to open your project in Eclipse and see where the ADT identifies errors in your application. From there, you can lookup respective changes in the Overview of Changes and API Diff Report, accessed by going to the [SDK homepage](http://developer.android.com/sdk/index.html), and clicking on the desired version on the left.

If you have additional trouble updating your code, visit the [Android Discussion Groups](http://code.google.com/android/groups.html) to seek help from other Android developers.

If you have modified one of the ApiDemos applications and would like to migrate it to the new SDK, note that you will need to uninstall the version of ApiDemos that comes preinstalled in the emulator. For more information, or if you encounter an "reinstallation" error when running or installing ApiDemos, see the troubleshooting topic [I can't install ApiDemos apps in my IDE because of a signing error](http://code.google.com/android/kb/troubleshooting.html#apidemosreinstall) for information about how to solve the problem.