# How to Create an Android Application using Eclipse on Windows 7

**Kevin Gleason** 

11/11/11

This application note is design to teach the reader how to setup an Android Development Environment on a Windows 7 Computer using eclipse and how to create an Android application. It first explains all of the software and hardware requirements and explains to the reader how to setup and install each of the required programs. It then explains how to design and code an Android application. Finally it explains how to use the Android Virtual Device to test the application.

# Table of Contents

1 Key Terms	2
2 Objective	2
3 System Requirements	2
4 Installation and Configuration	3
4.1 Download and Configure Eclipse IDE	3
4.2 Download Android SDK	4
4.3 Install ADT plugin for eclipse	5
4.4 Configure the ADT plugin.	5
4.5 Adding SDK Components	6
4.6 Create an AVD	6
5 Create a new Android Application	7
6 Coding the Application	8
6.1 Changing the text	8
6.2 Editing the layout	9
6.3 Editing the the java code	10
7 Running the Application	11
8 Conclusion	12
9 Resources	12

## 1 Key Terms

**SDK** (Software Development Kit) - A set of tools and libraries that allow the user to create an application based on a product.

**IDE** (Integrated Development Environment) – A software application that consists of a source code editor, A compiler, build automation tools and a debugger. It makes programing and running applications easier.

**ADT** (Android Development Tools) – A plugin for eclipse that extends the Eclipse IDE by providing more tools to develop Android Applications

**AVD** (Android Virtual Device) – An Android emulator that allows you to simulate how the application will run on an actual Android device.

**JDK** (Java SE Development Kit) – A popular Java SDK that is used to program Android applications.

# 2 Objective

This document will demonstrate how to create an Android application using the Eclipse IDE. It is design to guide the user through configuring his or her computer for Android development. It also walks the user through creating and running an Android application.

# **3 System Requirements**

Make sure that the computer being used for development meets the software and hardware requirements for the Android SDK and Eclipse.

http://developer.android.com/sdk/requirements.html

#### Supported Operating Systems

- Windows XP (32-bit), Vista (32- or 64-bit), or Windows 7 (32- or 64-bit)
- Mac OS X 10.5.8 or later (x86 only)
- . Linux (tested on Ubuntu Linux, Lucid Lynx)
  - o GNU C Library (glibc) 2.7 or later is required.
  - On Ubuntu Linux, version 8.04 or later is required
  - o 64-bit distributions must be capable of running 32-bit applications. For information about how to add support for 32-bit applications, see the Ubuntu Linux installation notes

#### Supported Development Environments

#### Eclipse IDE

- Eclipse 3.5 (Galileo) or greater
- Note: Eclipse 3.4 (Ganymede) is no longer supported with the latest version of ADT.
- . Eclipse JDT plugin (included in most Eclipse IDE packages)
- If you need to install or update Eclipse, you can download it from <a href="http://www.eclipse.org/downloads/">http://www.eclipse.org/downloads/</a>.
- Several types of Eclipse packages are available for each platform. For developing Android applications, we recommend that you install one of these packages:
- o Eclipse IDE for Java Developers
- o Eclipse Classic (versions 3.5.1 and higher)
- Eclipse IDE for Java EE Developers
- . JDK 5 or JDK 6 (JRE alone is not sufficient)
- Android Development Tools plugin (recommended)
- Not compatible with Gnu Compiler for Java (gcj)

#### Other development environments or IDEs

- JDK 5 or JDK 6 (JRE alone is not sufficient)
- Apache Ant 1.8 or later
- Not compatible with Gnu Compiler for Java (gcj)

Note: If JDK is already installed on your development computer, please take a moment to make sure that it meets the version requirements listed above. In particular, note that some Linux distributions may include JDK 1.4 or Gnu Compiler for Java, both of which are not supported for Android development.

#### Hardware requirements

The Android SDK requires disk storage for all of the components that you choose to install. The table below provides a rough idea of the disk-space requirements to expect, based on the components that you plan to use.

Component type	Approximate size	Comments
SDK Tools	35 MB	Required.
SDK Platform-tools	6 MB	Required.
Android platform (each)	150 MB	At least one platform is required.
SDK Add-on (each)	100 MB	Optional.
USB Driver for Windows	10 MB	Optional. For Windows only.
Samples (per platform)	10M	Optional.
Offline documentation	250 MB	Optional.

Note that the disk-space requirements above are in addition to those of the Eclipse IDE, JDK, or other prerequisite tools that you may need to install on your development computer.

# **4 Installation and Configuration**

## 4.1 Download and Configure Eclipse IDE

Eclipse is the preferred development environment for Android. You can download eclipse at <a href="http://www.eclipse.org/downloads/">http://www.eclipse.org/downloads/</a>. It is recommended that you install either classic or the Java Version of eclipse.



Home screen of Eclipse for Java Developers

#### 4.2 Download Android SDK

- Go to <a href="http://developer.android.com/sdk/index.html">http://developer.android.com/sdk/index.html</a> and click on the Windows installer.
- 2. This should automatically check to see if you have the proper JDK installed. If you do not you can download the newest JDK here: <a href="http://www.oracle.com/technetwork/java/javase/downloads/index.html">http://www.oracle.com/technetwork/java/javase/downloads/index.html</a>

Platform	Package	Size	MD5 Checksum
Windows	android-sdk_r15-windows.zip	33895447 bytes	cc2aadf7120d12b574981461736a96e9
	installer_r15-windows.exe (Recommended)	33902520 bytes	ee8481cb86a6646a4d963d5142902c5c
Mac OS X (intel)	android-sdk_r15-macosx.zip	30469921 bytes	03d2cdd3565771e8c7a438f1c40cc8a5
Linux (i386)	android-sdk_r15-linux.tgz	26124434 bytes	f529681fd1eda11c6e1e1d44b42c1432

Example of options when downloading the SDK.

### 4.3 Install ADT plugin for eclipse

- 1. Open Eclipse and go to Help->Install new Software.
- 2. Enter https://dl-ssl.google.com/android/eclipse/ in the Work with box.

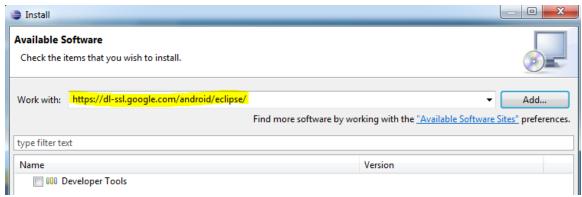


Figure 3.2 Example of options when downloading the SDK.

- 3. Then put a check mark next to the **Developer Tools** and click **Next**.
- 4. On the next screen it will show all of the tools that will be downloaded. Click **Next.**
- 5. Read and agree the license agree to the license agreements then click **Finish**.
- 6. Restart Eclipse.

#### 4.4 Configure the ADT plugin.

After installing the ADT it will need to be configured to point to the SDK directory.

- 1. Click on Window->Preferences->Android.
- 2. Click on **Browse** to find the location of your SDK directory.
- 3. Click Apply, and then OK.

#### 4.5 Adding SDK Components.

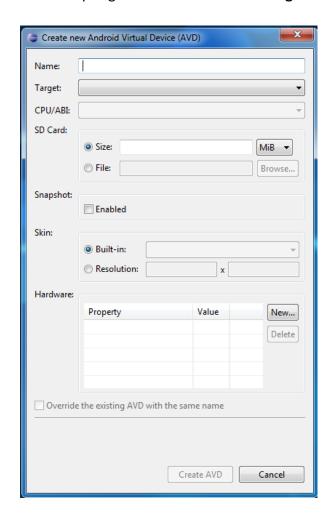
- 1. Click on Window->Android SDK Manager.
- 2. This will allow you to choose the Android platform versions, add-ons tools and other components. Choose the version of Android that you would like you application to work on.

Note: This application note will be using 2.3.3.

Click Install Selected and wait for the components to down. When the download is finished, verify and accept the new components.

#### 4.6 Create an AVD

1. In Eclipse go to Window->AVD Manager->New.



**AVD Creation Screen** 

- 2. Type in the **Name** of the **AVD** and choose a **Target**. The target is the version of the Android SDK that you would like to run on the emulator.
- 3. Click Create AVD.

# **5 Create a new Android Application**

- 1. In Eclipse go to File->New->Project...
- 2. Select an Android Project from the Android Folder and press Next.
- 3. Fill in the details of your Android application.
  - a. **Project Name**: The project name and folder that Eclipse will store the project files
  - b. **Build Target:** The version of the Android SDK that will be used when you build your program. Select a platform that is equal to or lower than the target chosen for the AVD.
  - c. **Application Name:** This is the name of the application.
  - d. **Package Name**: The namespace that all of the source code will reside under.
  - **e. Create Activity:** The name for that class stub that is generated by the plugin.
- 4. The values that are used in this example are:

a. Project Name: SampleApp

b. Build Target: 2.3.3

c. **Application Name:** SampleApp

d. Package Name: com.sample.example

e. Create Activity: SampleApp

5. Click on Finish.

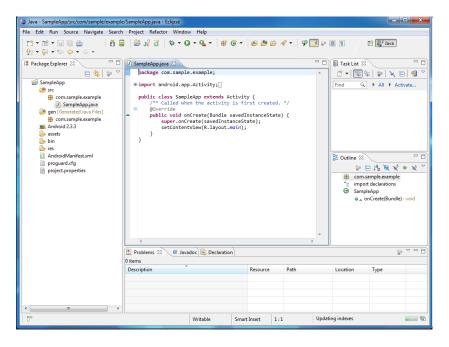


Figure 7.10 Screen after project creation.

# **6 Coding the Application**

# 6.1 Changing the text

- Open strings.xml which is located in res->values->string.xml. This file will hold all of the text that our layout will use.
  - a. Click on the strings.xml at the bottom to bring up the raw xml file.
  - b. Using the current string xml format change the xml to include
    - i. Name: name String: Please type your name.
    - ii. Name: app name String: Hello User
  - c. Your xml file should look like:

Figure 7.10 Completed string.xml

#### 6.2 Editing the layout

- Open main.xml which is located in res->layout->main.xml. This file will hold the layout of the application which includes labels, textboxes, and other UI interfaces.
  - a. Main.xml has a GUI and a raw xml view. The GUI has many different widgets that you can drag and drop onto the screen.
  - b. From the form widgets drag a Large text label and from Text Fields drag the first and last name text box onto the phone screen.
  - c. Then go to the raw xml view and add android:text="@string/name" to the TextView. You can see the final raw xml below. This will take the text that is labeled "name" in the strings.xml and display it for the TextView.

```
<?xml version="1.0" encoding="utf-8"?>
G<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"</pre>
      android:orientation="vertical"
      android:layout_width="fill_parent"
     android:layout height="fill parent" >
      <TextView
         android:id="@+id/textView1"
          android:layout_width="wrap_content"
          android:layout_height="wrap_content"
          android:text="@string/typename"
          android:textAppearance="?android:attr/textAppearanceLarge" />
      <EditText
         android:id="@+id/edittext"
          android:layout_width="fill_parent"
          android:layout height="wrap content"/>
 </LinearLayout>
```

Figure 7.10 Final main.xml

#### 6.3 Editing the the java code

- 1. Open SampleApp.java from the left hand side.
- 2. At the top add the following import statements

```
import android.view.View;
import android.view.View.OnKeyListener;
import android.widget.EditText;
import android.widget.Toast;
import android.view.KeyEvent;
```

- b. This allows you to use the functions located in these files.
- 3. Next we need to add the code to get the information from the text box and display it.

  The following code gets the name from the text box and forms it into a string that says

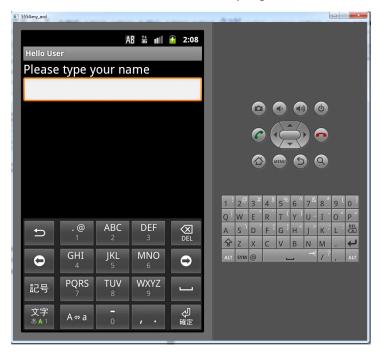
  "Hello [name]". Then it uses the toast class to display the message to the user when

  enter is pressed. Enter the following code at the end of onCreate.

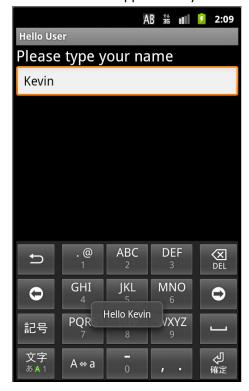
4. Save the files.

# 7 Running the Application

- 1. Click on the green circle with the white arrow.
- 2. Choose the AVD that we created in a previous step.
- 3. The android AVD will load and the program will run.



5. This is what will happen after you enter your Name and press enter.



6.

4.

7. You have successfully completed an android application.

## **8 Conclusion**

Congratulations you have successfully created an Android application. While this application note taught you how to create a basic application you can use the techniques you learned to expand on your application and turn it into something more powerful.

## 9 Resources

http://www.developer.android.com

http://www.eclipse.org/downloads/

http://www.oracle.com/technetwork/java/javase/downloads/index.html