

**Supplementary Lab Notes**  
**on the**  
**Android SDK and the Eclipse Integrated Development**  
**Environment**

**Iain D. Lambie**

**January 2015**

# Supplementary Lab Notes

## Contents

1	Using Android SDK and AVD Manager .....	3
1.1	Creating an Android Virtual Device (AVD) .....	3
1.2	Installed Packages .....	4
1.3	Starting the Android SDK and AVD Manager from the Command Line.....	4
2	Starting the Emulator.....	4
3	Using the Eclipse Integrated Development Environment (IDE).....	6
3.1	Perspectives .....	6
3.2	Views.....	6
3.3	Android Views.....	7
3.4	Importing Existing Projects .....	7
3.5	Creating a Project.....	7
4	Accessing SQLite using the Android Debugger (adb).....	8
5	Using Telnet to access and set emulator settings.....	10
5.1	Using Telnet to Set Latitude and Longitude for use in testing GPS applications.....	10

# 1 Using Android SDK and AVD Manager

When you are developing any software application you generally make use of an Integrated Development Environment (IDE). The Eclipse IDE is a popular development tool because it can be customised for a specific development platform via a Plug-In. This is the case for Android development. In your lab sessions you will be using the Eclipse environment with the Android Plug-In. All your code will be written in the Java programming language, but you will make extensive use of classes in various Android API's which are available via the Android SDK.

If you have not already started Eclipse do so now. If you are working in a University lab you will find the version of Eclipse in the SEB-VM-Win7x64.2014v1 VMWare image. The Eclipse icon should be on the desktop of the virtual machine, double click on eclipse.exe. When the Application starts you should get a splash screen indicating that Eclipse is starting up. Accept the default location for the projects that you are going to create. The default location should be a folder in the users folder of the c: drive.

## 1.1 Creating an Android Virtual Device (AVD)

Your first task is to create an Android Virtual machine which you can use to deploy your applications to. It is generally a good idea to deploy your application to a simulator first to check that the application runs correctly.

From Eclipse Select Window | Android Virtual Device Manager.

Expand the window if necessary until you can see the New Button on the right hand side of the screen. Click on the New Button. A dialogue box should appear. This dialogue box allows you to create a customised Android Virtual machine that you can use to deploy your applications to in order to test the application against the application's specification. There are a number of entries in this dialogue box that you need to fill in. Enter the values below.

Name: labavd1

Device: Nexus S (4.0", 480 x 800 hdpi)

Target : Android 4.2.2

Skin: No skin

SD Card 4 Gigabytes

Accept the other default settings for the moment and then click Ok.

You should now have created an Android Virtual Device that you can use to deploy your applications too. Now select labavd1 and click Start | Launch. You should now have a new Android virtual device running

In the next section you will learn how to start the Android AVD from the command line. This can be useful in some situations.

If this approach is problematic, navigate to the android sdk directory of the Android installation and run the SDK manager directly, but do not install any new features!

## 1.2 Installed Packages

From the Android SDK and AVD Manager you can also view the packages that have been installed as part of the Android SDK. This is useful because you may be developing your application for a specific Android version.

## 1.3 Starting the Android SDK and AVD Manager from the Command Line

The Android SDK and Device Manager can be started from the command line. The application is called android and is located in the tools folder of the Android installation.

## 2 Starting the Emulator

You can start the Android Emulator directly from the command line. In order to do this you need to start a DOS session. In the labs the environment variable PATH should be set to include the Android tools directory. If the PATH variable does not include the tools directory you will need to navigate to the tools directory of the Android SDK.

Generally this will be

*androidinstallation*\sdk\tools

where *androidinstallation* is the installation location for your Eclipse/Android installation

Once you have navigated to this directory you need to execute specific command to start the emulator. Here is one such command.

```
emulator -avd labavd1
```

Here labavd1 is the name of a specific android virtual machine that was created using the Android SDK and AVD Manager. (This may have been achieved via the Windows Tab from the eclipse IDE.

Add the option `-verbose` if you want to view the start up process.

```
emulator -verbose -avd labavd1
```

You can also use `-scale` as an option in an attempt to fit the emulator on the screen. (You may find that the top of the emulator is just off the screen and that you cannot resize it. If this is the case try the command below and adjust the fractional value accordingly.

```
emulator -scale 0.9 -avd labavd1
```

Once the emulator is running you should get a screen representing the mobile android device and an adjacent keypad. You can use the key pad to enter values into applications running on the phone.

If for any reason the emulator is larger than the screen, that is you cannot access the options in the top right hand corner you can use shortcut keys instead.

ALT+F4 will close the emulator or ALT+ENTER will close the emulator.

In the University labs you should be able to start the emulator from the command line in any DOS window within the VMware Web Apps image. This is because the PATH environment variable is set up to include the tools directory.

On your own laptops or desktop machines at home it is worth adding the location of the tools directory to the PATH environment variable to allow you to start the emulator from any directory at the DOS prompt rather than having to navigate to the tools directory.

To do this you would need to add the location of the tools directory to the PATH environment variable by accessing the Advanced Properties option. You do this by right clicking my computer on Windows XP or in Windows 7 | right clicking the computer name in Windows Explorer | selecting options | selecting advanced systems settings.

Once you have done this set a System variable called ANDROID\_HOME to say

C:\adt-bundle-windows-x86-201330522\eclipse\sdk

(This is the path in the Web Apps virtual machine in the University labs)

Now add an entry to the PATH variable for your system (add this to the end of the entries that are already there and separate the new entry from the others by a semi-colon ;)

```
;%ANDROID_HOME%\tools
```

This allows you to start the emulator from any DOS directory without having to navigate to the android sdk tools directory. You can now start the Android SDK and AVD Manager from any directory as well now.

## 3 Using the Eclipse Integrated Development Environment (IDE)

### 3.1 Perspectives

Each Workbench window contains one or more perspectives. A perspective defines the initial set and layout of views in the Workbench window. Within the window, each perspective shares the same set of editors. Each perspective provides a set of functionality aimed at accomplishing a specific type of task or works with specific types of resources. For example, the Java perspective combines views that you would commonly use while editing Java source files, while the Debug perspective contains the views that you would use while debugging Java programs. As you work in the Workbench, you will probably switch perspectives frequently.

Perspectives control what appears in certain menus and toolbars. They define visible *action sets*, which you can change to customize a perspective. You can save a perspective that you build in this manner, making your own custom perspective that you can open again later.

You can use the [General | Perspectives](#) preference page to open perspectives in the same window or in a new window.

### 3.2 Views

Perspectives offer pre-defined combinations of views and editors. To open a view that is not included in the current perspective, select Window | Show View from the main menu bar.

You can create fast views to provide quick access to views that you use often.

After adding a view to the current perspective, you may wish to save your new layout by clicking Window | Save Perspective As

### 3.3 Android Views

There are a number of views that are specific to Android Application Development. The Android Views should be available in versions of Eclipse that have been installed with the Android Plug in. To see a list of the specific Android Views select Window | Show View | Other. A list of Android views now appears. Scroll through the list to get an idea of the views that are available.

### 3.4 Importing Existing Projects

You can use the Import Wizard to import an existing project into the workspace

From the main menu bar, select File | Import. The Import wizard should now open.

Select General | Existing Project into Workspace and click Next

Choose either Select root directory or Select archive file and click the associated Browse to locate to locate the directory or projects which you would like to import

Click Finish to start the import.

### 3.5 Creating a Project

To create a project:

On the main menu bar, click **File > New Android Application** Project. The New Project wizard opens.

The Android Project Wizard will open and you can make appropriate selections for the project.

In the Project name field, type a name for your new project.

Now select the target for the version of Android that you are developing for, as follows: Min 2.2, Target 4.2, Compile with 4.2 and select None for the Theme.

Select defaults on the next 2 screen by selecting next and then Select Empty Activity

Change the Activity name as you see fit. The layout name will then follow.

Click Finish. The new project is listed in one of the navigation views.

Now Select Project | Clean (This may be needed to get rid of the exclamation mark on the project name in the Package Explorer. The Package Explorer should be on the top left and should show the projects that you have loaded. You can click on the +

next to the project and then next to each folder to see the files that are contained in each folder. You will be adding/modifying to these files/folders in the lab exercise.

If the package explorer is not visible then select Window | Show View | Package Explorer.

Generally in the Introduction to Mobile Programming module when you are using Eclipse you will be creating and modifying Android projects.

#### **4 Accessing SQLite using the Android Debugger (adb)**

SQLite is an extremely effective and efficient way to store data on a mobile device. There are occasions when you will want to access the files forming the database directly. To do this you can make use of the Android Debugger (adb) to do this.

First you need to make sure that the folder containing the debugger is in the path environment variable of the computer that you are using. You will find the adb tool in the platform-tools directory of your Android sdk installation. If you have followed through the previous instructions in section 2 Starting the Emulator you should already have set an environment variable called ANDROID\_HOME. If this is the case you only need to add the following entry to the path command.

```
%ANDROID_HOME%\platform-tools
```

Before you can run the debugger you must ensure that you have an emulator running.

Once you have done this you can start a DOS session and start the Android debugger by typing

```
adb -e shell
```

on the command line.

This will direct commands to the only running emulator. An error is returned if there is more than one emulator running. If the debugger starts successfully you will be presented with a # prompt.

To proceed further with SQLite you need to know the name of the database that you want to access. Let us assume that you have a database called shopping.db and that this database was created from an application that was located in the package



org.me.myandroidstuff. In that case you would enter the following command to start up sqlite on the shopping.db database

```
sqlite3 /data/data/org.me.myandroidstuff/databases/shopping.db
```

at the # command prompt. You will receive some feedback at this point which indicates that SQLite is starting. Once this is complete you are ready to issue some basis SQL commands that are supported by SQLite. So for example if you issued the SQL command:

```
select * from standardItems
```

then you may get something like

```
1|Bread
```

```
2|Milk
```

```
3|Potatoes
```

```
4|Porridge Oats
```

```
5|Vegetables
```

```
6|Rice
```

```
sqlite>
```

If you did not include the name of the database that you wanted to work with when you started sqlite, you will need to change to the location of the database before you actually issue any SQL command. You would achieve this with the cd command and in this case would write:

```
cd /data/data/org.me.myandroidstuff/databases/
```

You can also list what is in a directory by using the ls command. If you have worked with Unix or the Linux operating system, these commands will be familiar to you.

Minimally you will find this approach useful to check that you have in fact created the database and that it has been populated by some values as directed by your Android programme.

To view the structure of the files on the emulator you can switch to the DDMS perspective

## 5 Using Telnet to access and set emulator settings

First of all the Telnet Client needs to be installed. To do this Select Control Panel from the Start menu and then then select Programs following by Turn Windows Features on. Tick the box for telnet Client and wait while the application installs. In future once telnet is installed you can start a DOS session and simply type Telnet on the command line.

### 5.1 Using Telnet to Set Latitude and Longitude for use in testing GPS applications

Once Telnet is running enter the following commands

open localhost 5554 // This opens a command line connection to the

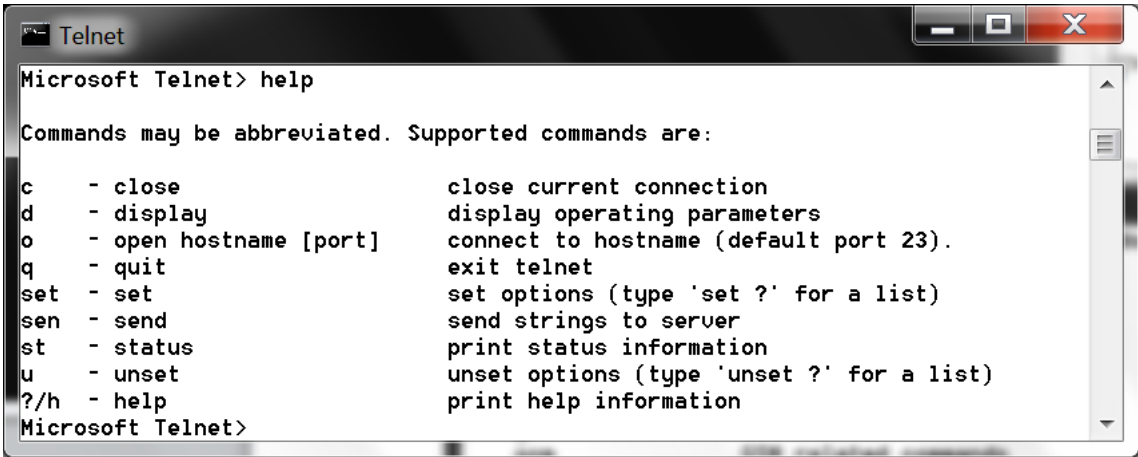
//Android Emulator. You are now connected directly to the

// Android operating system.

geo fix {latitude value} {longitude value} // Enter specific values for latitude

// and longitude.

When you are connected to Telnet you can type help for a list of commands. The screen capture below lists the commands.

A screenshot of a Telnet window titled "Telnet". The window shows the output of the "help" command in a Microsoft Telnet session. The text displayed is: "Microsoft Telnet> help", "Commands may be abbreviated. Supported commands are:", followed by a list of commands and their descriptions. The commands listed are: c (close), d (display), o (open hostname [port]), q (quit), set (set), sen (send), st (status), u (unset), and ?/h (help). Each command is followed by a brief description of its function. The window has standard Windows window controls (minimize, maximize, close) in the top right corner.

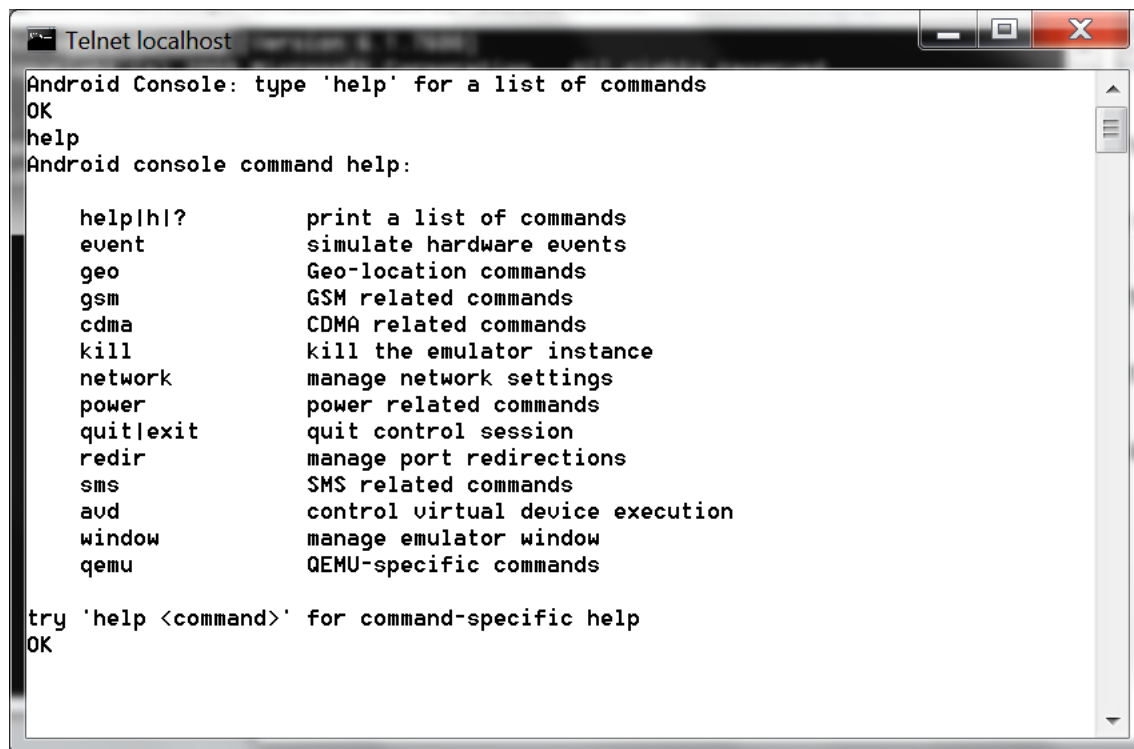
```
Microsoft Telnet> help

Commands may be abbreviated. Supported commands are:

c      - close                close current connection
d      - display              display operating parameters
o      - open hostname [port] connect to hostname (default port 23).
q      - quit                  exit telnet
set    - set                  set options (type 'set ?' for a list)
sen    - send                  send strings to server
st     - status                print status information
u      - unset                 unset options (type 'unset ?' for a list)
?/h    - help                  print help information

Microsoft Telnet>
```

When you are connected to Android you can type help for a list of commands. The screen capture below lists the commands.



```
Telnet localhost
Android Console: type 'help' for a list of commands
OK
help
Android console command help:

  help|hl?      print a list of commands
  event         simulate hardware events
  geo           Geo-location commands
  gsm           GSM related commands
  cdma          CDMA related commands
  kill          kill the emulator instance
  network       manage network settings
  power         power related commands
  quit|exit     quit control session
  redir        manage port redirections
  sms          SMS related commands
  aud          control virtual device execution
  window       manage emulator window
  qemu         QEMU-specific commands

try 'help <command>' for command-specific help
OK
```

## Appendix Getting Started

In order to get started developing applications for Android Devices you will need to download a number the Android Software Development Kit (SDK). You are also advised to download a suitable Integrated Development Environment (IDE). The following link is a good starting point.

<http://developer.android.com/sdk/index.html>

The link is to the Developer Area on the Android web site. If you work your way through the material found at this link you will be able to install and the SDK and the Eclipse IDE.

Development of Application in Android require a knowledge of XML and Java. The XML is used to describe the appearance of the graphical components in an Android Applications. Attributes such as size, colour, font position and type of graphical component can be specified in XML.

If you have completed the year 2 course on Object Oriented Software Development you will have covered the vast majority of the concepts needed in Java. The main area that you will not have covered is the idea of an Event and in particular an event that is generated by user interaction with a graphical interface.

I will not be recommending any books as such but there is a useful Electronic book in the library that you can access to provide some initial/further reading.

All classes will start in week 1. This means *labs*, *tutorials* and *lectures*. Therefore make sure that you turn up at your first scheduled class regardless of the type of class that is shown on your timetable.