Lesson 2

Android Development Tools = Eclipse + ADT + SDK

Victor Matos
Cleveland State University

2. Development Environment = Eclipse + ADT + SDK

- Android applications are usually created using the **Java** programming language[1]

- Your Java project must import various **Android Libraries** (such as android.jar, maps.jar, etc) to gain the functionality needed to work inside the Android OS.

- Even the simplest of Android apps is composed of several elements such as: user-defined classes, android jars, third-party libraries, XML files defining the UIs or views, multimedia resources, data assets such as disk files, external arrays and strings, databases, and finally a *Manifest* summarizing the 'anatomy' and permissions requested by the app.

- The package(s) holding the raw app components are given to the compiler to obtain a single signed and deployable **Android Package** (an .apk file).

- Like in Java, apk files are the **byte-code** version of the app that finally will be 'executed' by interpretation inside a **Dalvik Virtual Machine** (DVM).

[1] Visit http://xamarin.com/monoforandroid for a commercial iOS and Android IDE that works with C# and Windows .NET

2. Development Environment = Eclipse + ADT + SDK

- Creating, organizing and managing the components of an Android app is better done using a ‘friendly’ workbench.

- The Android developer’s workbench typically includes the following tools:
  1. Eclipse IDE
  2. Android Development Tools (ADT), and
  3. Android System Development Kit (SDK)

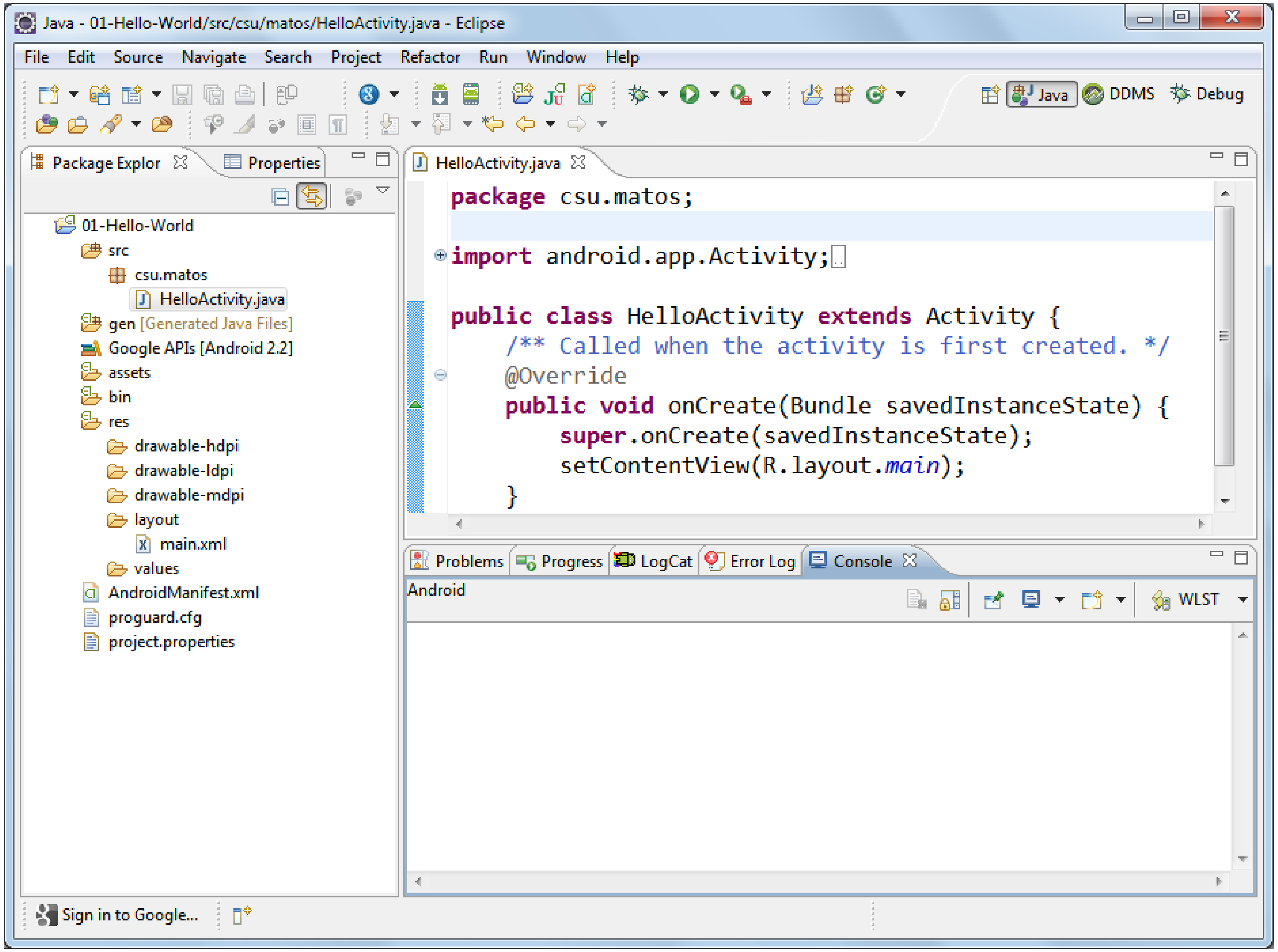
- **Eclipse IDE** allows you to create and debug your Java code, and manage the various resources that normally are used in the making of an Android app.

- The **ADT plugin** extends Eclipse so you can easily reach the tools of the SDK through the use of menus, perspectives and icons seamlessly integrated in the Eclipse’s IDE.

- The **SDK** contains tools needed to transfer, profile, emulate, observe, and debug your applications which could run into any virtual or physical Android device.

2. Development Environment = Eclipse + ADT + SDK

Typical Layout of the Eclipse IDE for Android Development

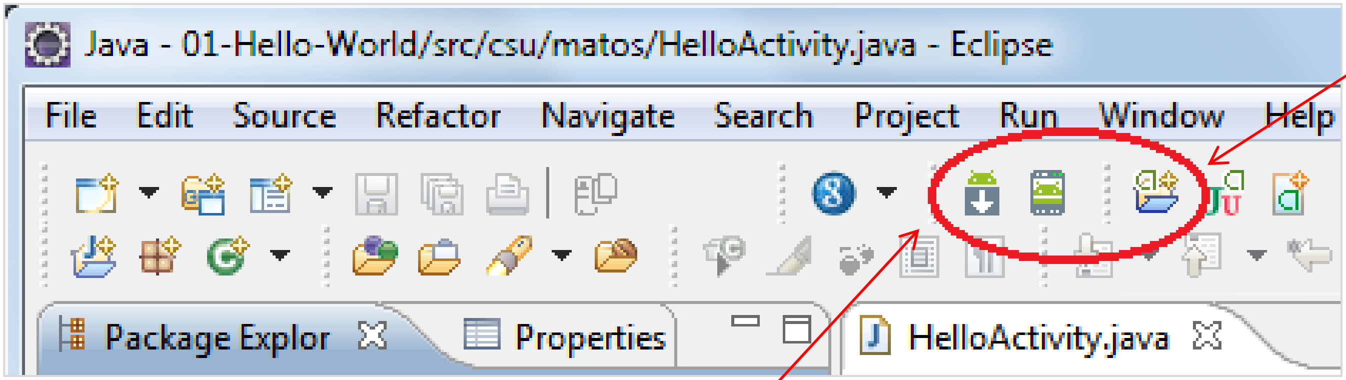


```
package csu.matos;

import android.app.Activity;

public class HelloActivity extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }
```

2. Development Environment = Eclipse + ADT + SDK

Typical Layout of the Eclipse IDE for Android Development
(details…)

These icons are added to Eclipse by the ADT plugin

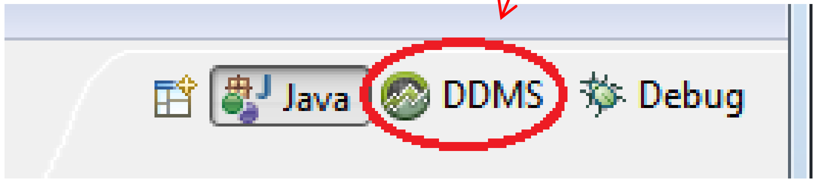


Java - 01-Hello-World/src/csu/matos/HelloActivity.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explor
Properties
HelloActivity.java

Opens Android SDK manager

Opens Android AVD Virtual Device Manager

Wizard creates a new Android Project



Opens DDMS Perspective
Dalvik Debugging Monitoring System

***Note**: The **DDMS** and **Hierarchy View** can be manually added by the user to Eclipse's tool bar*

2. Development Environment = Eclipse + ADT + SDK

SETUP

Prepare your computer – Install SDK: Windows, Mac, Linux

We assume you have already installed the Java JDK and Eclipse IDE in your computer

- Java JDK is available at: http://www.oracle.com/technetwork/java/javase/downloads/index.html
- Eclipse IDE for Java EE Developers is available at: http://www.eclipse.org/downloads/

The next instructions are given to:

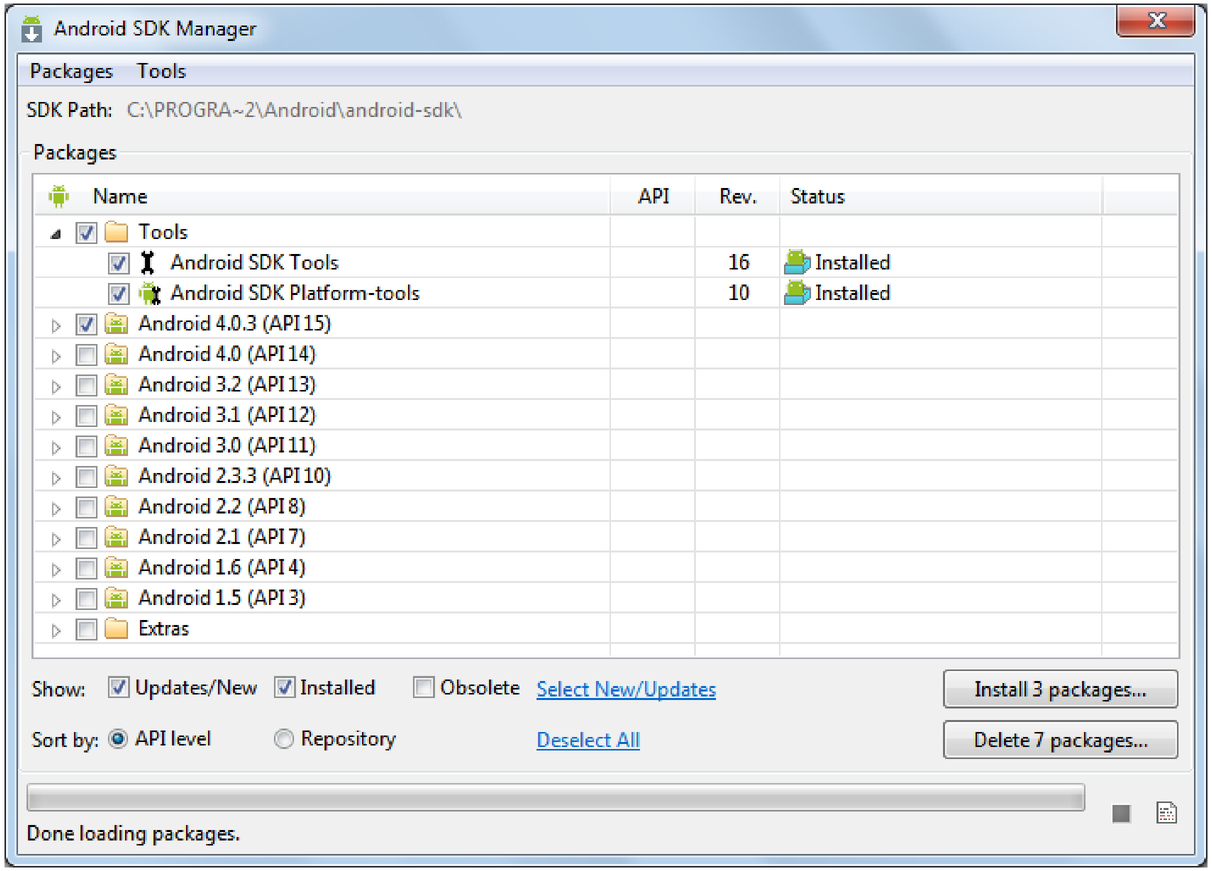
(a) User Wanting to Update their Older Android Workbench,
(b) First Time Users.

2. Development Environment = Eclipse + ADT + SDK

Aside Note:

SDKs are named after a dessert item. Available versions at the time of writing are:

1.5	Cupcake,
1.6	Donut,
2.1	Eclair,
2.2	Froyo,
2.3	Gingerbread [1],
3.x	Honeycomb,
4.x	Ice Cream Sandwich



SDK Path: C:\PROGRA~2\Android\android-sdk\

Packages

Name	API	Rev.	Status
☑ Tools			
☑ Android SDK Tools		16	Installed
☑ Android SDK Platform-tools		10	Installed
☑ Android 4.0.3 (API 15)			
☐ Android 4.0 (API 14)			
☐ Android 3.2 (API 13)			
☐ Android 3.1 (API 12)			
☐ Android 3.0 (API 11)			
☐ Android 2.3.3 (API 10)			
☐ Android 2.2 (API 8)			
☐ Android 2.1 (API 7)			
☐ Android 1.6 (API 4)			
☐ Android 1.5 (API 3)			
☐ Extras			

Show: ☑ Updates/New ☑ Installed ☐ Obsolete Select New/Updates

Sort by: [✓ API level] Repository Deselect All

Done loading packages.

[1] By March 2012 Gingerbread accounted for approximately 66% of the Android market share.
See page: http://www.appbrain.com/stats/top-android-sdk-versions

2. Development Environment = Eclipse + ADT + SDK

SETUP

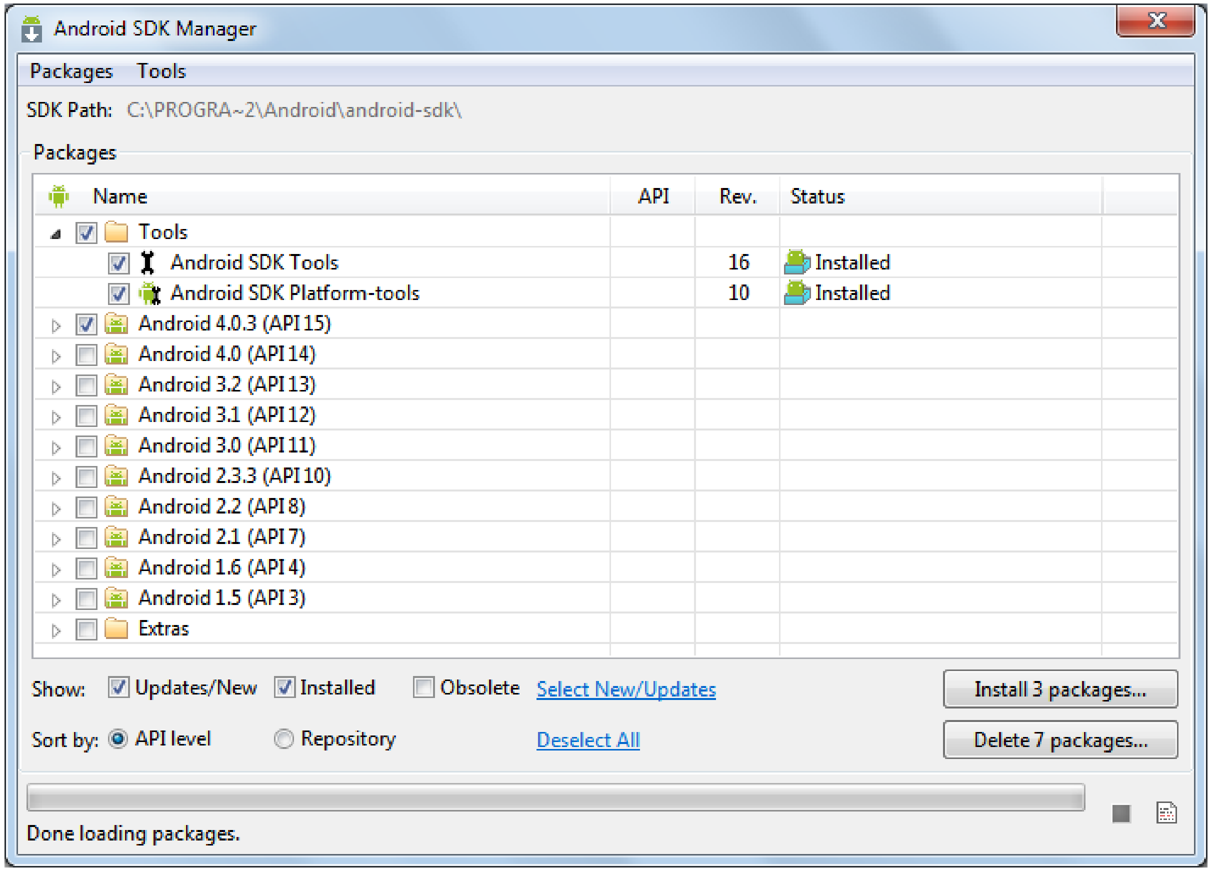
(a) Users Wanting to Update an Older Android Workbench

If you are currently using the Android SDK, you just need to *update* to the latest tools or platform using the already installed *Android SDK and AVD Manager.*

1. Click on the *SDK Manager* icon.

2. You will see a form similar to the one on the right.

3. Select the Packages you want to install and wait until they are setup in your machine.



Android SDK Manager

Packages Tools

SDK Path: C:\PROGRA~2\Android\android-sdk\

Packages

Name	API	Rev.	Status
☑ Tools			
☑ Android SDK Tools		16	Installed
☑ Android SDK Platform-tools		10	Installed
☑ Android 4.0.3 (API 15)			
☐ Android 4.0 (API 14)			
☐ Android 3.2 (API 13)			
☐ Android 3.1 (API 12)			
☐ Android 3.0 (API 11)			
☐ Android 2.3.3 (API 10)			
☐ Android 2.2 (API 8)			
☐ Android 2.1 (API 7)			
☐ Android 1.6 (API 4)			
☐ Android 1.5 (API 3)			
☐ Extras			

Show: ☑ Updates/New ☑ Installed ☐ Obsolete Select New/Updates

Sort by: ● API level ○ Repository Deselect All

Install 3 packages...

Delete 7 packages...

Done loading packages.

2. Development Environment = Eclipse + ADT + SDK

SETUP

(b) First Time Users (Windows, Mac, Linux)

1. Install the appropriate **SDK starter package** from the page http://developer.android.com/sdk/index.html

2. Install the **ADT Plugin** for Eclipse
   1. Start Eclipse, then select **Help** > **Install New Software....**
   2. Click **Add** button (top-right corner)
   3. In the next dialog-box enter "**ADT Plugin**" for the *Name* and the following URL for the *Location*: **https://dl-ssl.google.com/android/eclipse/**
   4. Click **OK**
   5. Select the checkbox next to Developer Tools and click **Next** > **Next**
   6. Accept the license agreements, then click **Finish**.
   7. After the installation end you need to restart Eclipse.

3. Add **Android platforms** and other components to your SDK (see previous option (a))

2. Development Environment = Eclipse + ADT + SDK

Configuring the ADT Plugin

The next step is to modify your ADT preferences in Eclipse to point to the Android SDK directory:

1. Select **Window** > **Preferences...** to open the Preferences panel (Mac OS X: **Eclipse** > **Preferences**).
1. Select **Android** from the left panel.
2. To set the box *SDK Location* that appears in the main panel, click **Browse...** and locate your downloaded SDK directory (usually c:/Program Files (x86)/Android /android-sdk)
3. Click **Apply**, then **OK**.

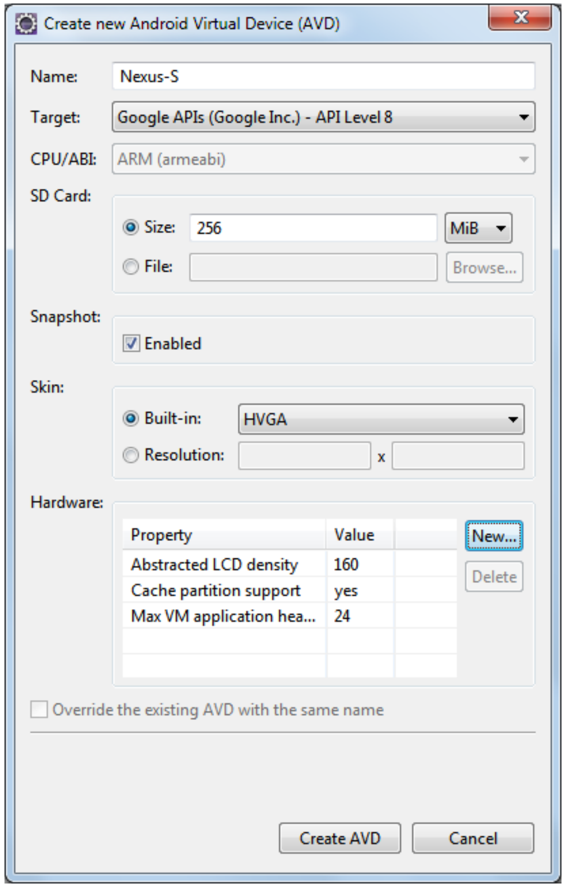
Done!

2. Development Environment = Eclipse + ADT + SDK

Creating an Android Virtual Device (AVD)

You should test your applications on a real phone (or tablet).
However, the SDK allows you to create realistic virtual devices on which your applications could be tested.

1. To create an emulator, click on the AVD Manager
2. Click **New**. The **Create New AVD** dialog appears.
3. Type the name of the AVD, such as "Nexus-S"
4. Choose a target (such as "Google APIs… API Level8").
5. Indicate how much memory the simulator will use.
6. Tick option box "Snapshot" to load faster.
7. Indicate screen size (HVGA is sufficient in general)
8. Optionally specify any additional hardware components (such as SD-card, camer, accelerometer, GPS,…)
9. Click **Create AVD**.


Create new Android Virtual Device (AVD)

Name: Nexus-S
Target: Google APIs (Google Inc.) - API Level 8
CPU/ABI: ARM (armeabi)
SD Card: Size: 256 MiB
File:
Snapshot: [✓] Enabled
Skin: Built-in: HVGA
Resolution: x
Hardware:

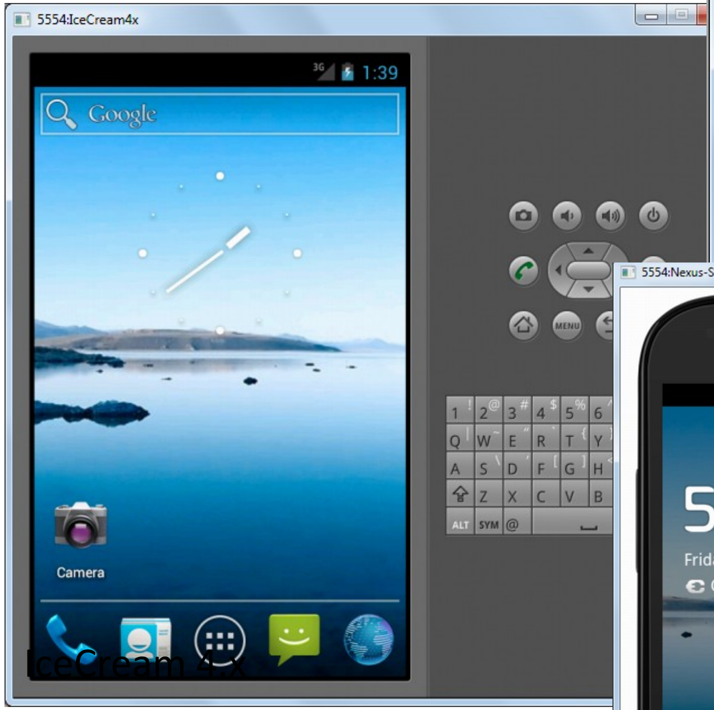
Property	Value
Abstracted LCD density	160
Cache partition support	yes
Max VM application hea...	24

Override the existing AVD with the same name

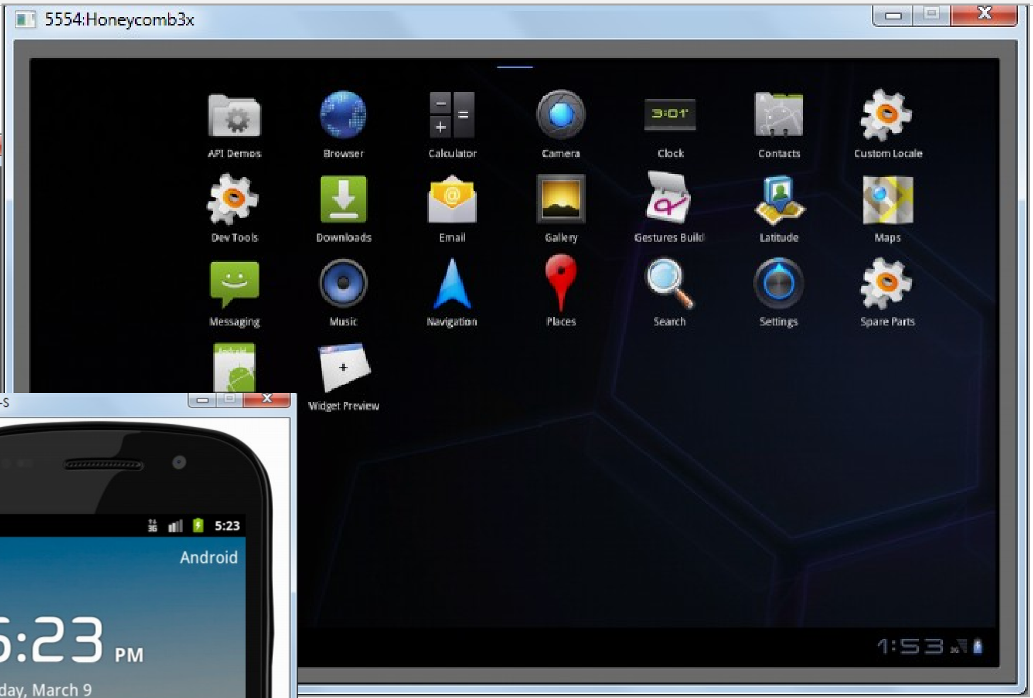
2. Development Environment = Eclipse + ADT + SDK

Creating Android Virtual Devices (AVD)

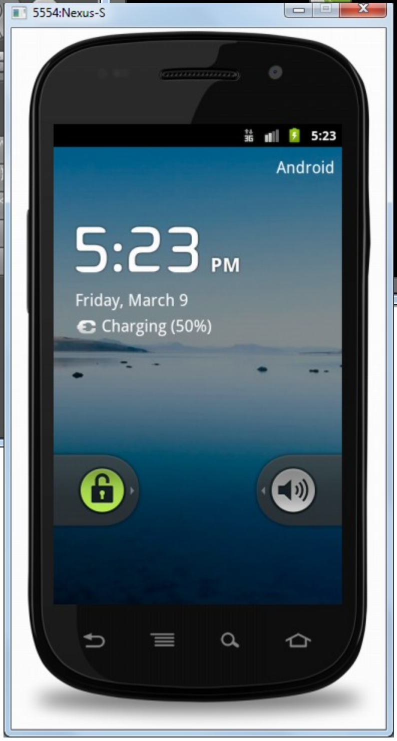
Some examples:



Phone Emulator **IceCream 4.x**



Tablet showing **Honeycomb 3.x**

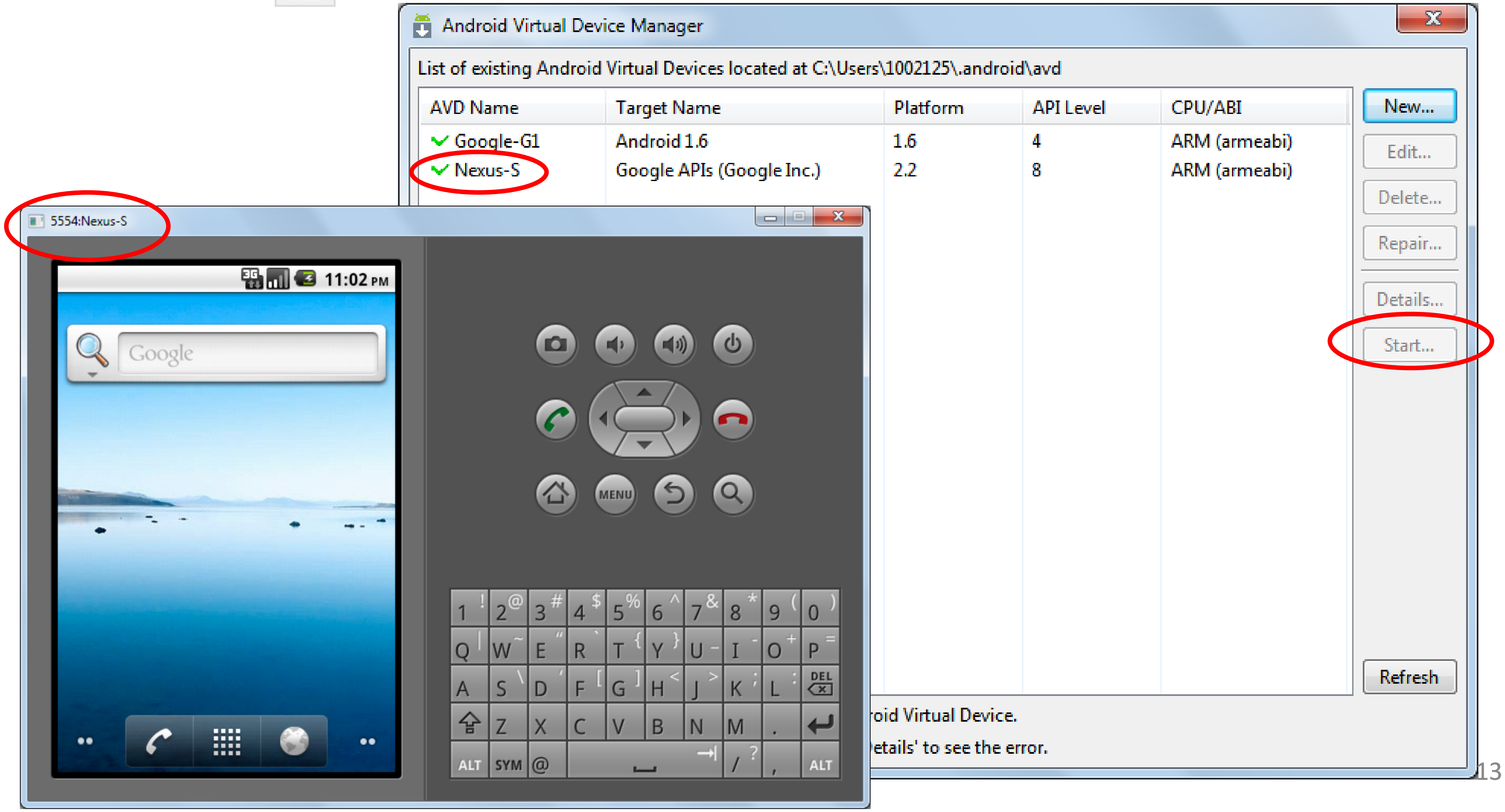


Gingerbread 2.3 running on a custom skin for Nexus-S. See pages:
http://heikobehrens.net/2011/03/15/android-skins/
and
http://zandog.deviantart.com/

2. Development Environment = Eclipse + ADT + SDK

Testing the Emulator

Click on the AVD Manager. Choose an emulator, click **Start.**



Android Virtual Device Manager

List of existing Android Virtual Devices located at C:\Users\1002125\.android\avd

AVD Name	Target Name	Platform	API Level	CPU/ABI
✓ Google-G1	Android 1.6	1.6	4	ARM (armeabi)
✓ Nexus-S	Google APIs (Google Inc.)	2.2	8	ARM (armeabi)

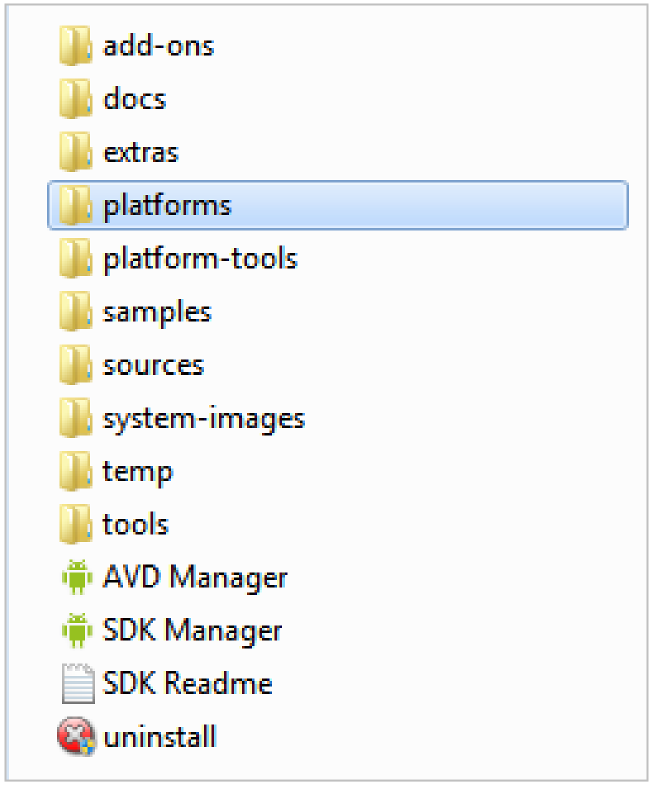
roid Virtual Device.

etails' to see the error.

Android Setup Tutorial

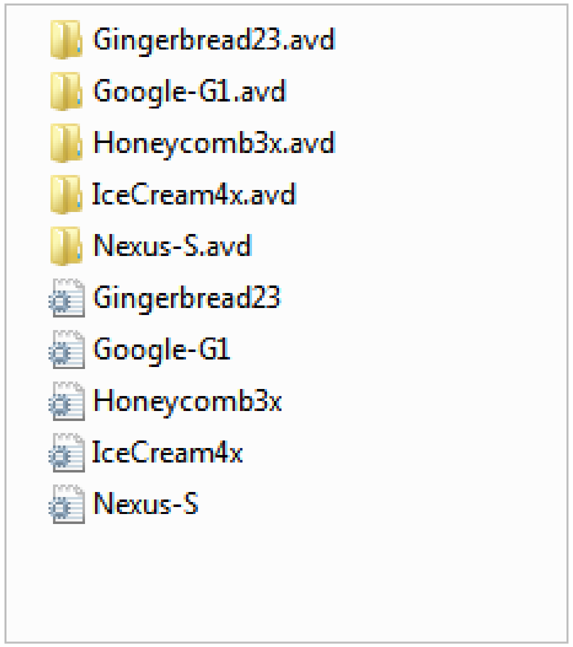
After you complete your setup look for the following two subdirectories in your file system

C:\Program Files (x86)\Android\android-sdk



```
add-ons
docs
extras
platforms
platform-tools
samples
sources
system-images
temp
tools
AVD Manager
SDK Manager
SDK Readme
uninstall
```

C:\Users\1002125\.android\avd



```
Gingerbread23.avd
Google-G1.avd
Honeycomb3x.avd
IceCream4x.avd
Nexus-S.avd
Gingerbread23
Google-G1
Honeycomb3x
IceCream4x
Nexus-S
```

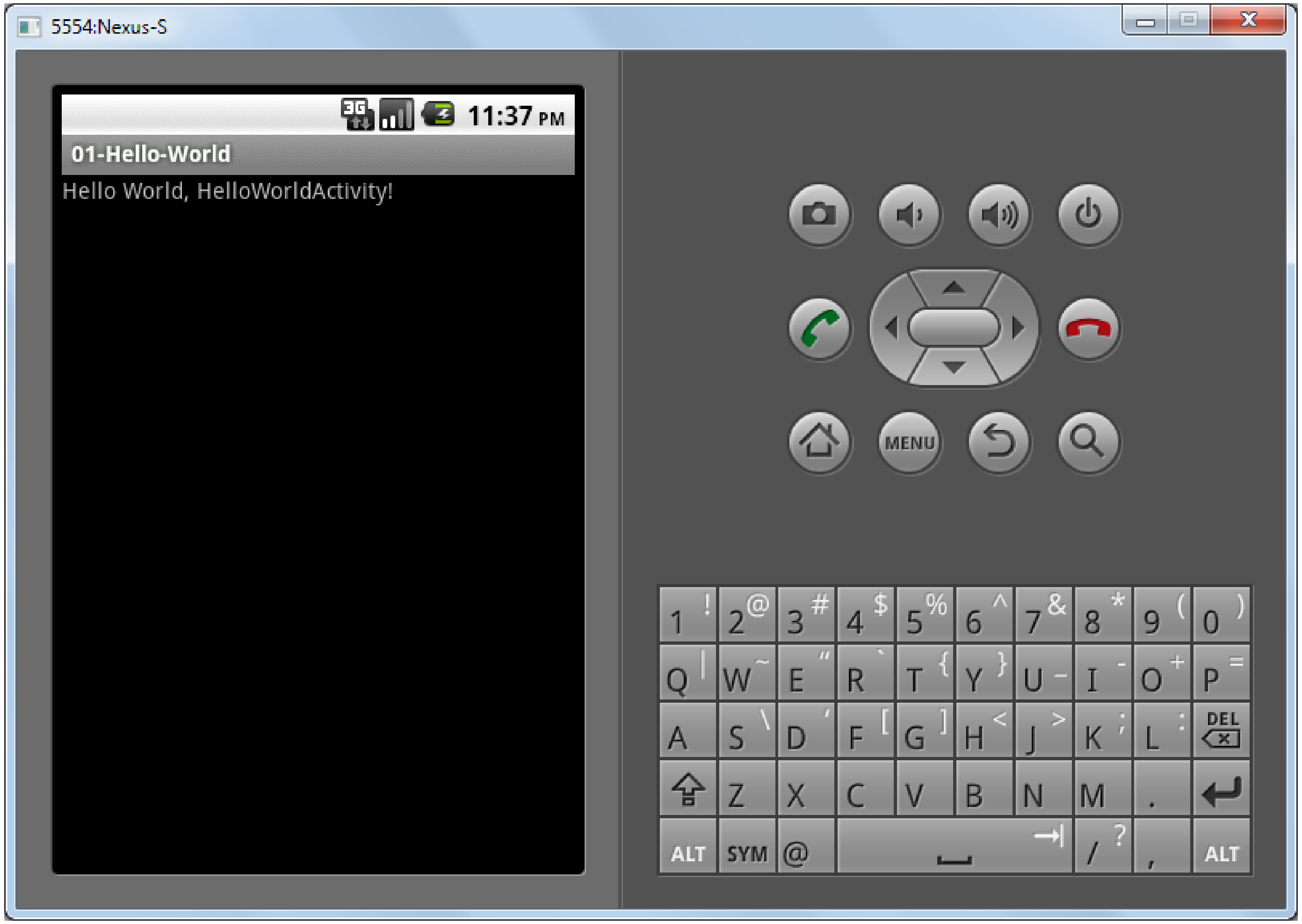
This folder contains your Android SDK, tools, and platforms

This directory holds your Virtual Devices (AVDs)

Testing Setup - Example: Hello World

Appendix. Creating an Android Project (made for SDK2.2 - Froyo)

An unabridged version of “Hello World”



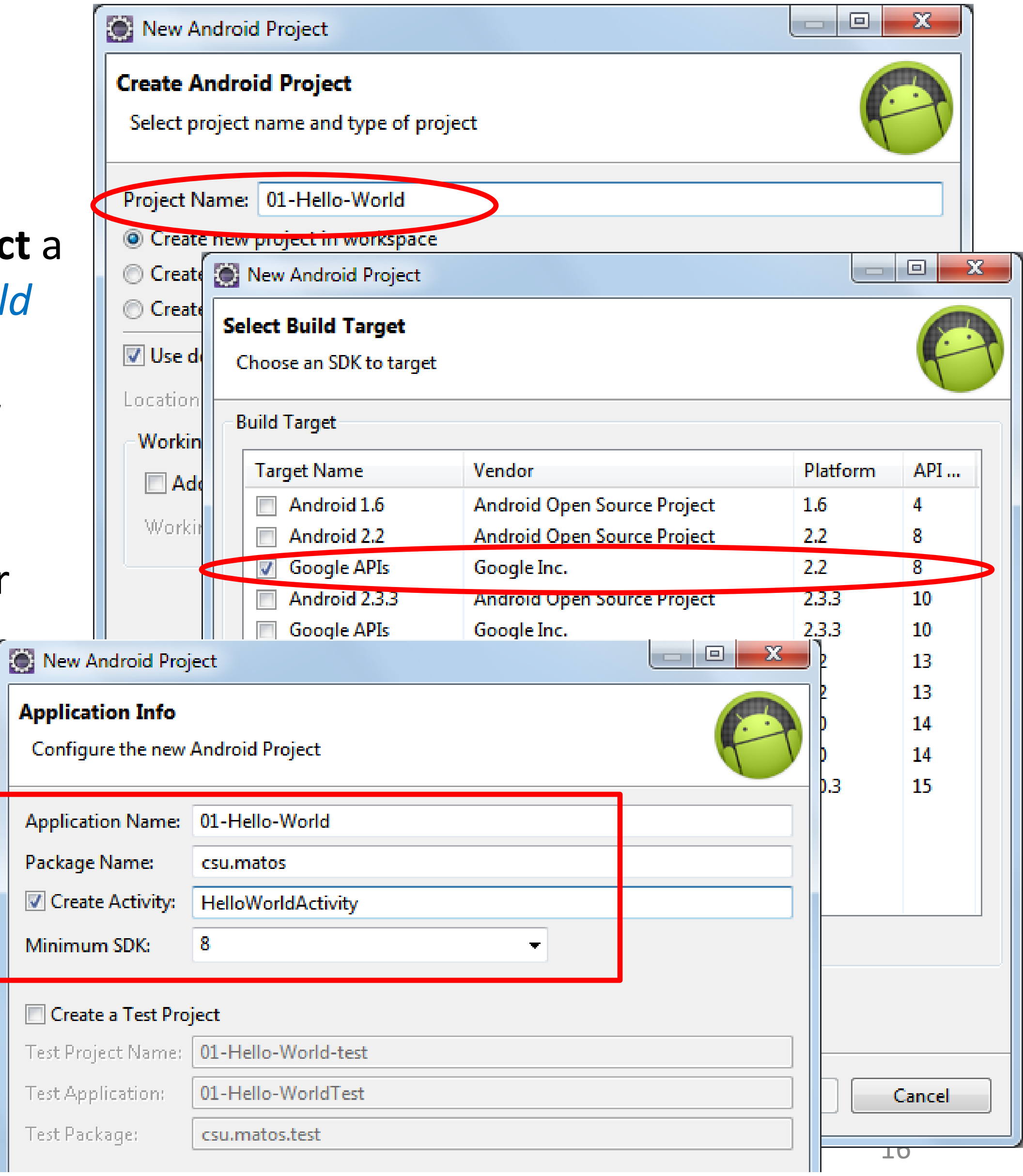
Hello World, HelloWorldActivity!

Testing Setup - Example: Hello World

Creating an Android Project

To create a new project:

1. Start **Eclipse**
2. Select **File** > **New** > **Android Project** a
3. Enter Project name: *01-Hello-World*
4. Click **Next**
5. On Select Build Target choose row
   Google APIs   Google Inc. 2.2   8
6. Click **Next**
7. On the *Application Info* form enter
   Pacakage Name: *csu.matos*
   Check box *Create Activity*
   Activity name: *HelloActivity*.
   Min SDK Version: *8*.
   Click *Finish*.



New Android Project

Create Android Project
Select project name and type of project

Project Name: 01-Hello-World
(●) Create new project in workspace
( ) Create
( ) Create
[✓] Use d
Location
Workin
[ ] Add
Workin

New Android Project

Select Build Target
Choose an SDK to target

Build Target

Target Name	Vendor	Platform	API ...
[ ] Android 1.6	Android Open Source Project	1.6	4
[ ] Android 2.2	Android Open Source Project	2.2	8
[✓] Google APIs	Google Inc.	2.2	8
[ ] Android 2.3.3	Android Open Source Project	2.3.3	10
[ ] Google APIs	Google Inc.	2.3.3	10
		2	13
		2	13
		0	14
		0	14
		0.3	15

Cancel

New Android Project

Application Info
Configure the new Android Project

Application Name: 01-Hello-World
Package Name: csu.matos
[✓] Create Activity: HelloWorldActivity
Minimum SDK: 8

[ ] Create a Test Project
Test Project Name: 01-Hello-World-test
Test Application: 01-Hello-WorldTest
Test Package: csu.matos.test

Testing Setup - Example: Hello World

OBSERVATION: Creating an Android Project using Eclipse

The *New Android Project* Wizard creates the following folders and files in your new project space:

- **src/** Includes your skeleton Activity Java file. All other Java files for your application go here.
- ***<Android Version>/*** (e.g., Android 2.2/) Includes the android.jar file that your application will build against.
- **gen/** This contains the Java files generated by ADT, such as your R.java file
- **assets/** This is empty. You can use it to store raw asset files.
- **res/** This folder holds application resources such as *drawable* files, *layout* files, *string* values, etc.
- **bin/** The bytecode (.apk) version of your app is stored here
- **AndroidManifest.xml** The Android Manifest for your project.
- **default.properties** This file contains project settings, such as the build target.

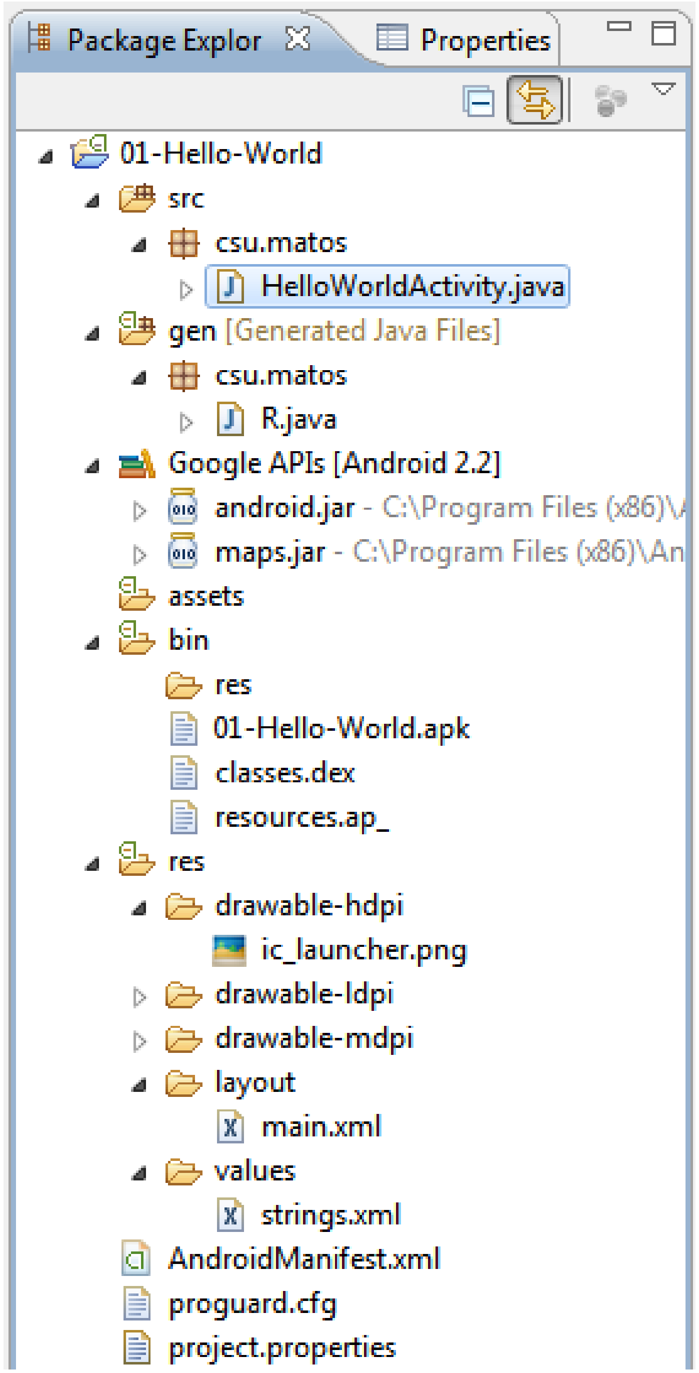
Testing Setup – Example: Hello World

Creating an Android Project

The following folders and files are created for the 01-Hello-World project.

To test the application, position the cursor on the code panel, and then click on the *Run* menu button.

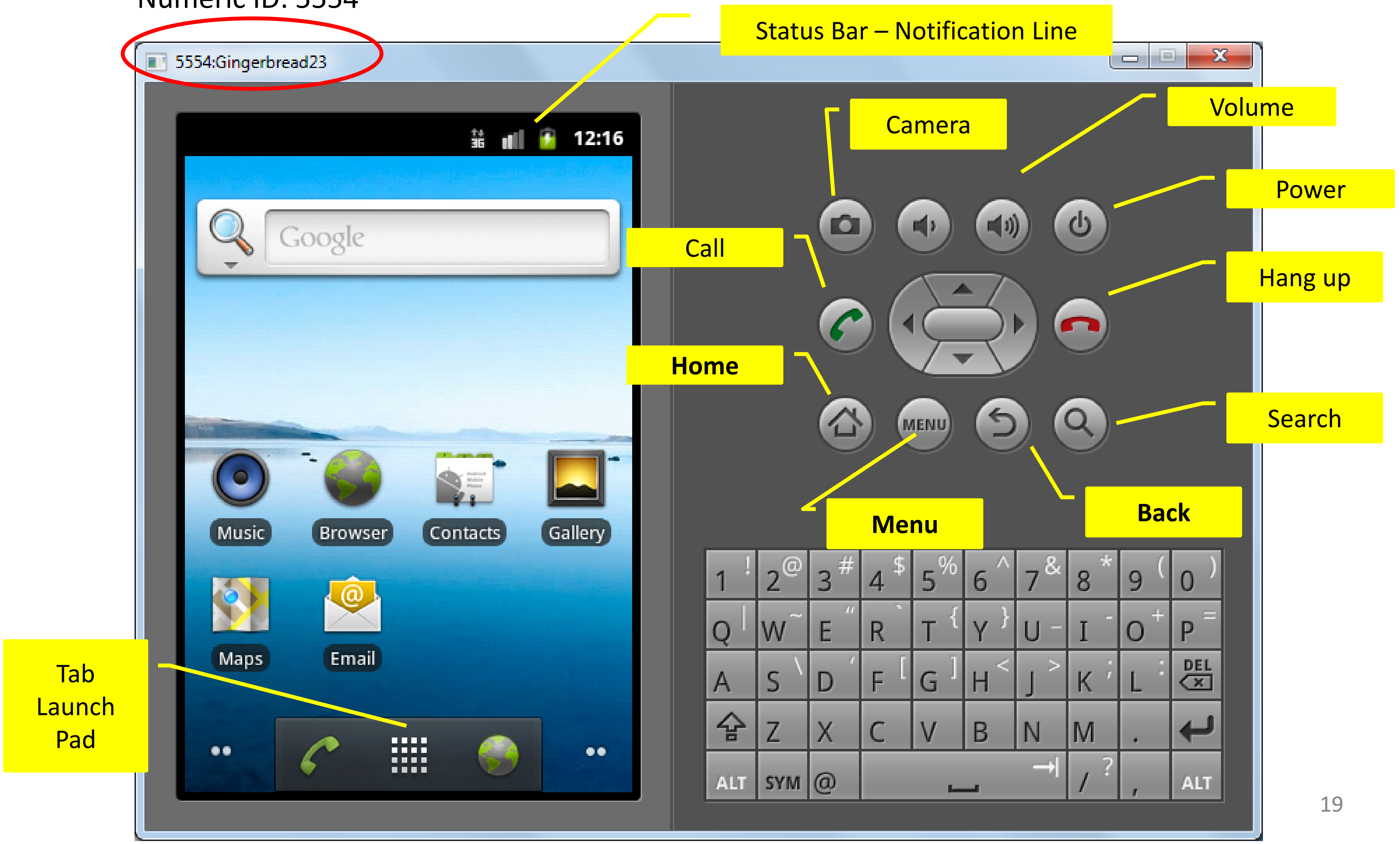
The fragment of code illustrated on page 4 is executed, and its effect on the emulator is shown on page 12.



```
01-Hello-World
  src
    csu.matos
      HelloWorldActivity.java
  gen [Generated Java Files]
    csu.matos
      R.java
  Google APIs [Android 2.2]
    android.jar - C:\Program Files (x86)\
    maps.jar - C:\Program Files (x86)\An
  assets
  bin
    res
    01-Hello-World.apk
    classes.dex
    resources.ap_
  res
    drawable-hdpi
      ic_launcher.png
    drawable-ldpi
    drawable-mdpi
    layout
      main.xml
    values
      strings.xml
  AndroidManifest.xml
  proguard.cfg
  project.properties
```

Android Emulator (v2.3 skin)

Numeric ID: 5554

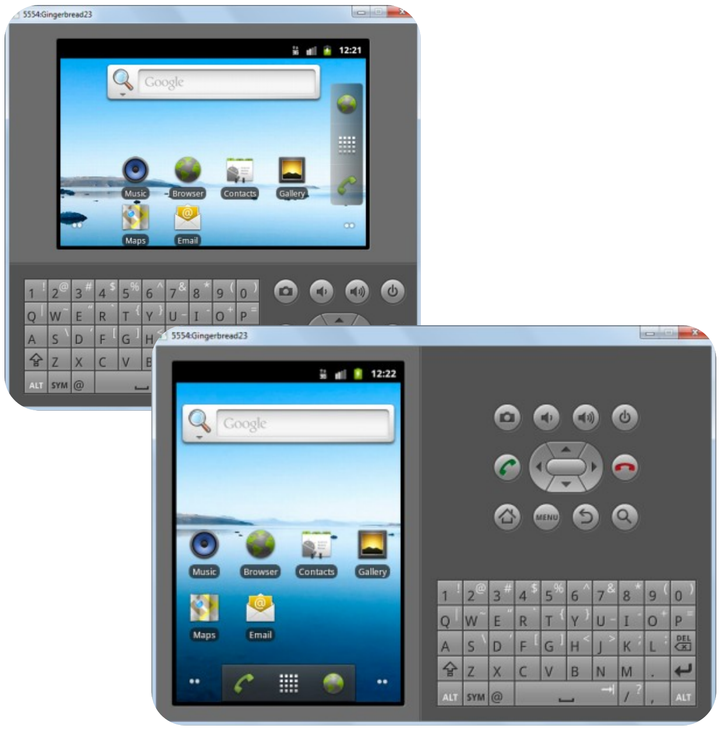


Android Emulator

Keyboard	OS function
Escape	Back button
Home	Home button
F2, PageUp	Menu (Soft-Left) button
Shift-F2, PageDown	Start (Soft-Right) button
F3	Call/Dial button
F4	Hangup / EndCall button
F5	Search button
F7	Power button
Ctrl-F3, Ctrl-KEYPAD_5	Camera button
Ctrl-F5, KEYPAD_PLUS	Volume up button
Ctrl-F6, KEYPAD_MINUS	Volume down button
KEYPAD_5	DPad center
KEYPAD_4	DPad left
KEYPAD_6	DPad right
KEYPAD_8	DPad up
KEYPAD_2	DPad down
F8	toggle cell network on/off
F9	toggle code profiling (when -trace option set)
Alt-ENTER	toggle FullScreen mode
Ctrl-T	toggle trackball mode
Ctrl-F11, KEYPAD_7	switch to previous layout
Ctrl-F12, KEYPAD_9	switch to next layout

Controlling the Android Emulator through (*your computer's*) keyboard keys

Keypad keys only work when *NumLock* is deactivated.



Android Emulator

Working with Emulator Disk Images

- *The Android simulator uses QEMU technology [Website: www.qemu.org]*
- *QEMU is an open source machine emulator which allows the operating system and programs made for one machine (e.g. an ARM CPU) run on a different machine (e.g. your own PC).*

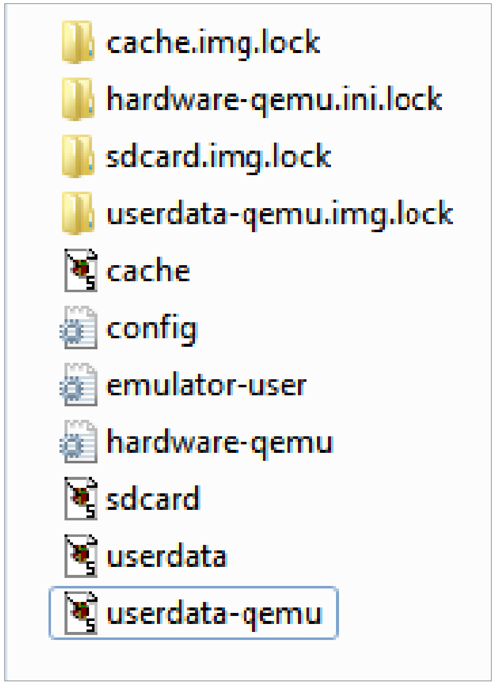
When you create a Virtual Device, the SDK Makes several **disk images** containing among others:

(1) OS kernel,
(2) the Android system,
(3) user data (userdata-qemu.img)
(4) simulated SD card (sdcard.img).

By default, the Emulator searches for the disk images in the private storage area of the AVD in use, for instance the "Gingerbread23" AVD is at:
C:\Users\yourFolder\.android\avd\Gingerbread23

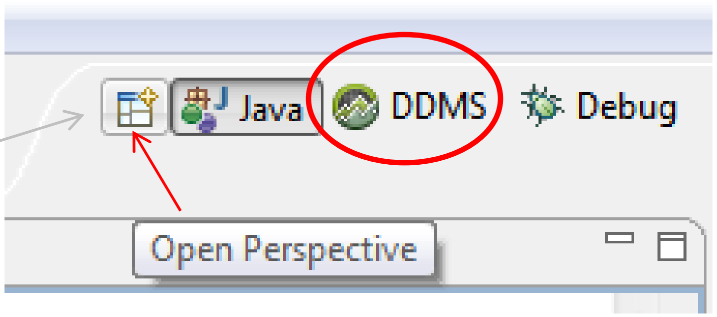
Mac OS users should look into **~/.android/avd**

C:\Users\yourFolder\.android\avd\Gingerbread23

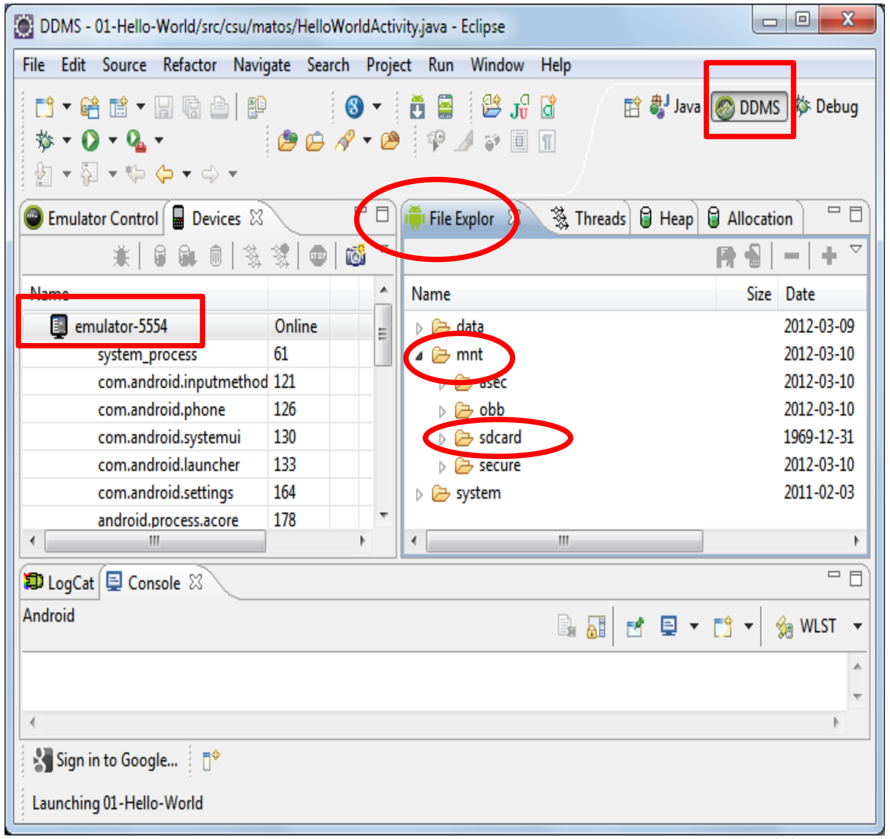

cache.img.lock
hardware-qemu.ini.lock
sdcard.img.lock
userdata-qemu.img.lock
cache
config
emulator-user
hardware-qemu
sdcard
userdata
userdata-qemu

Android Emulator

Moving Data, Music and Picture files to the Emulator's SDcard



1. You need to add the **DDMS** perspective to your Eclipse IDE.
2. Change to the DDMS perspective. Make sure your AVD has started (You will see a layout similar to the following)
3. Click on the **File Explorer** tab.
4. Expand the **mnt** (mounted devices) folder.
5. Expand the **sdcard** folder
6. Open your Window's **Explorer**.
7. Choose a file in your PC. Transfer a copy to the emulator by dragging and dropping it on top of the **sdcard** folder.

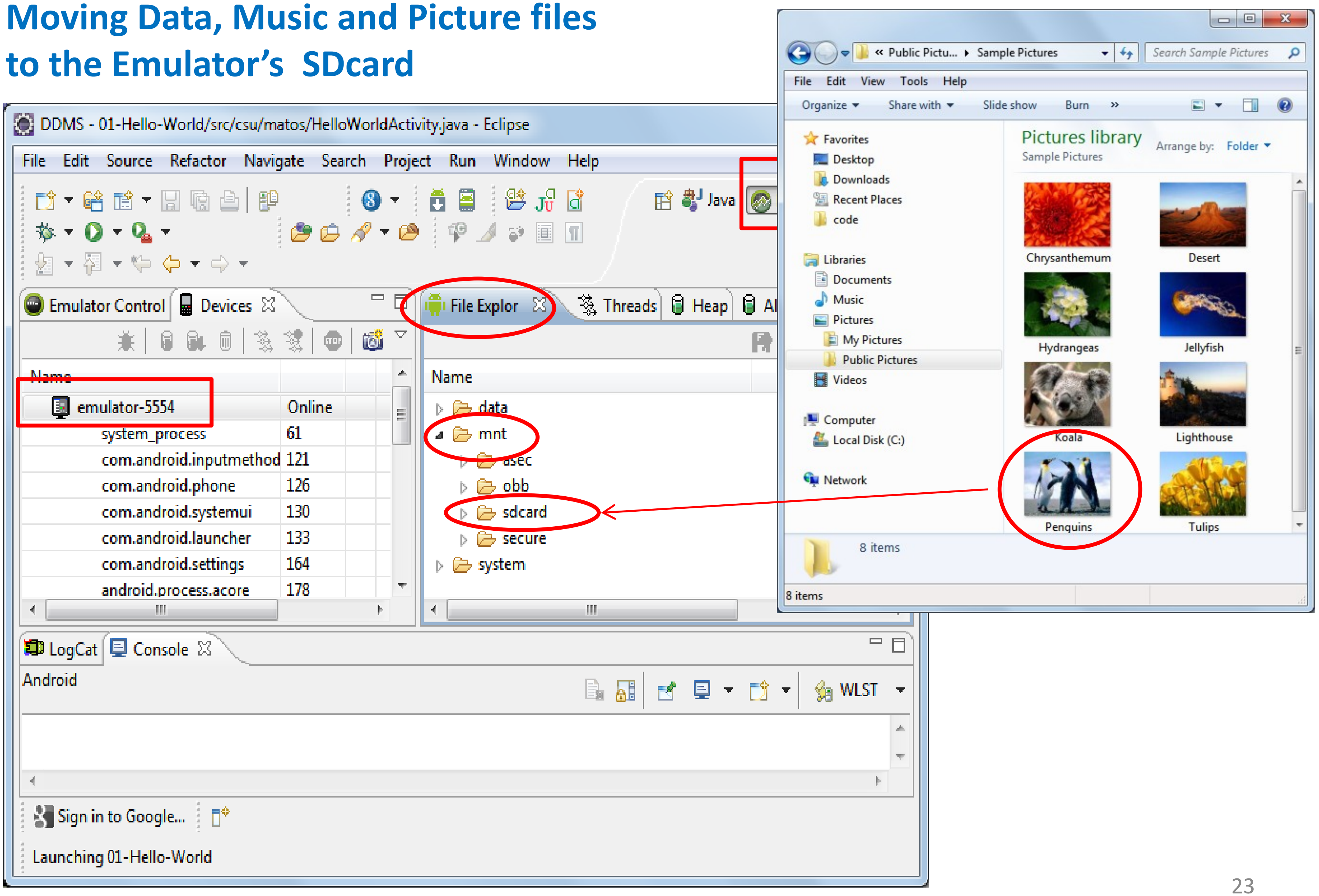

DDMS - 01-Hello-World/src/csu/matos/HelloWorldActivity.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Java DDMS Debug
Emulator Control Devices
File Explor Threads Heap Allocation
LogCat Console
Android
WLST
Sign in to Google...
Launching 01-Hello-World

Name	
emulator-5554	Online
system_process	61
com.android.inputmethod	121
com.android.phone	126
com.android.systemui	130
com.android.launcher	133
com.android.settings	164
android.process.acore	178

Name	Size	Date
data		2012-03-09
mnt		2012-03-10
asec		2012-03-10
obb		2012-03-10
sdcard		1969-12-31
secure		2012-03-10
system		2011-02-03

Android Emulator

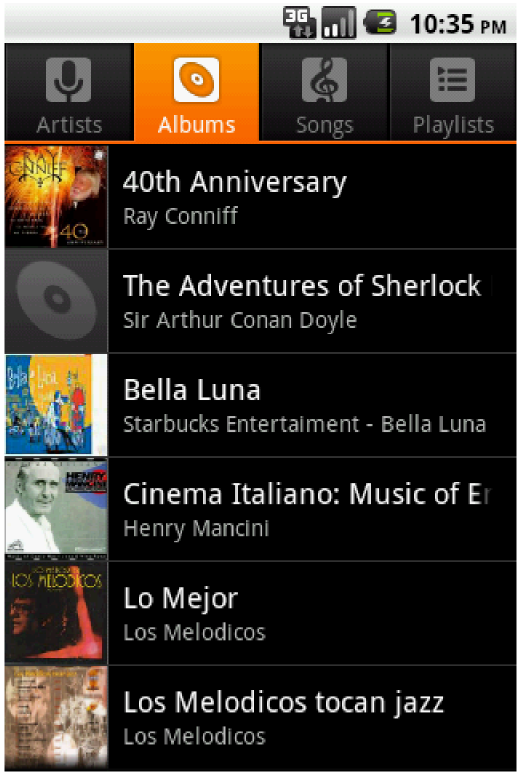
Moving Data, Music and Picture files to the Emulator's SDcard



Android Emulator

Moving Data, Music and Pictures to the SDcard

4. Return to the emulator. This time you will see your selected multimedia files in the SDcard. For instance…



40th Anniversary
Ray Conniff

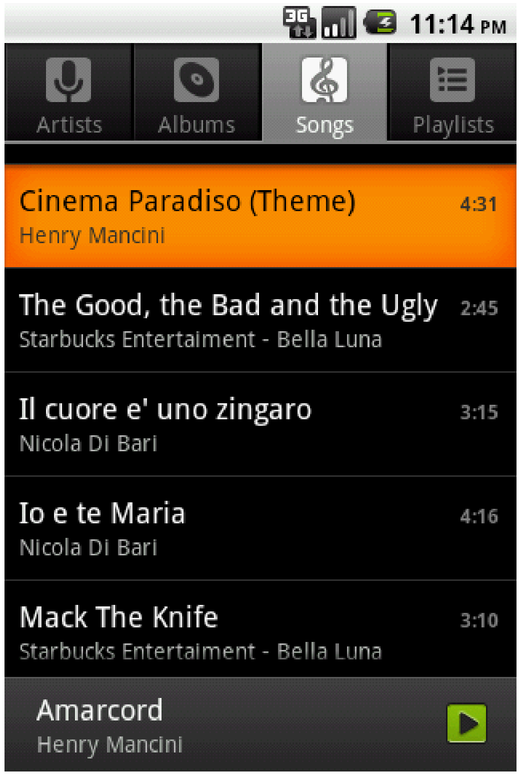
The Adventures of Sherlock
Sir Arthur Conan Doyle

Bella Luna
Starbucks Entertaiment - Bella Luna

Cinema Italiano: Music of E
Henry Mancini

Lo Mejor
Los Melodicos

Los Melodicos tocan jazz
Los Melodicos



Cinema Paradiso (Theme) 4:31
Henry Mancini

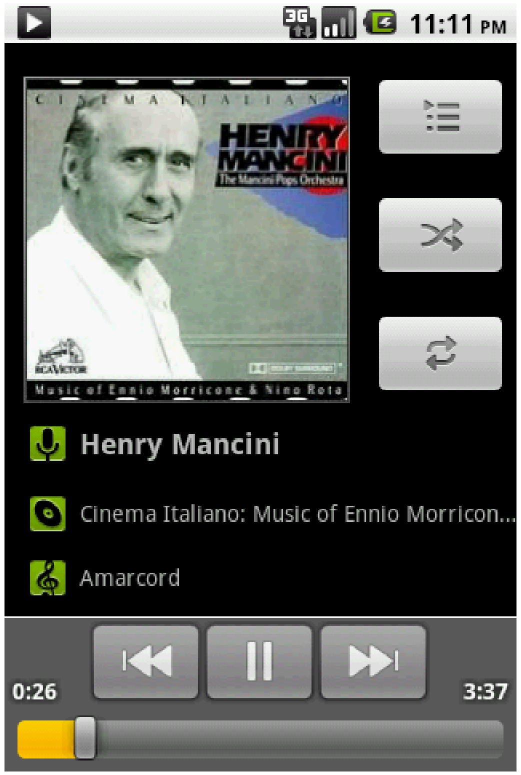
The Good, the Bad and the Ugly 2:45
Starbucks Entertaiment - Bella Luna

Il cuore e' uno zingaro 3:15
Nicola Di Bari

Io e te Maria 4:16
Nicola Di Bari

Mack The Knife 3:10
Starbucks Entertaiment - Bella Luna

Amarcord
Henry Mancini

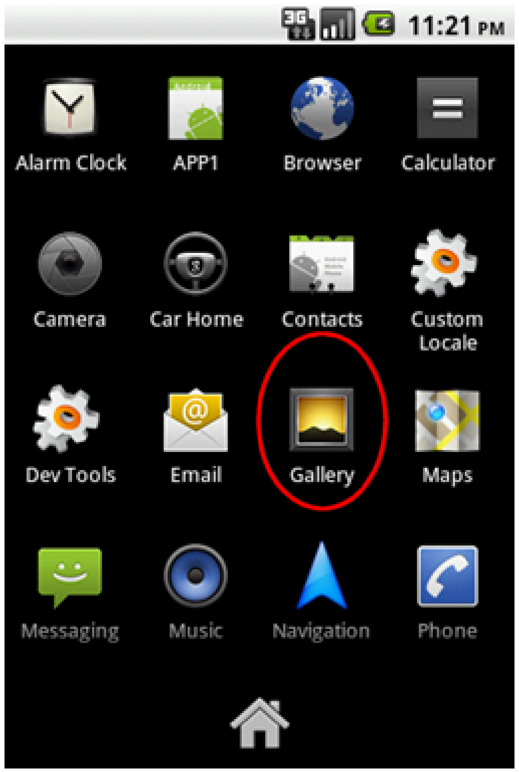


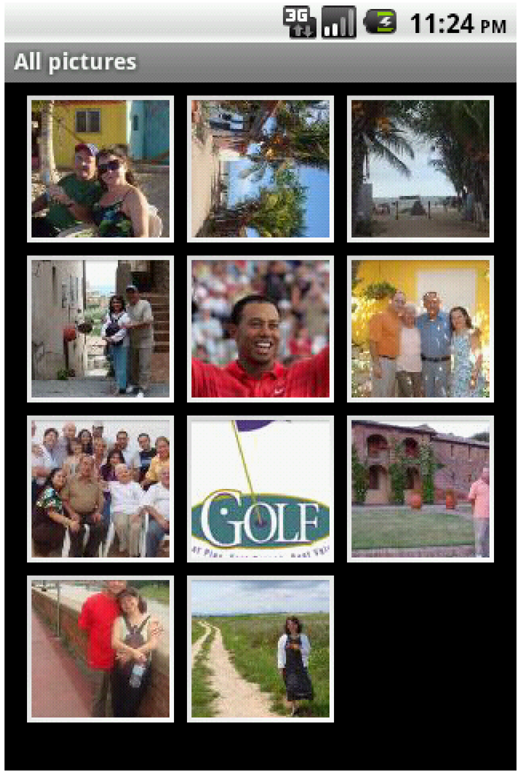
Henry Mancini
Cinema Italiano: Music of Ennio Morricon...
Amarcord

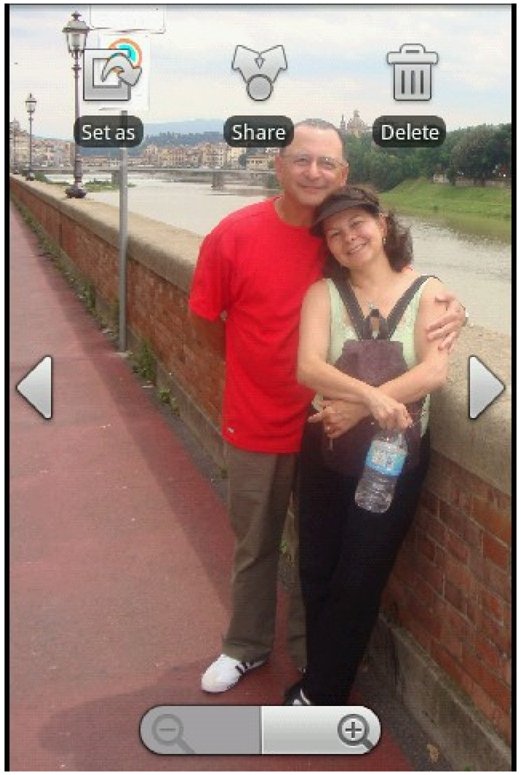
Android Emulator

Moving Data, Music and Pictures to the SDcard

5. Pictures are displayed by clicking the *Application Pad* and invoking the **Gallery** application



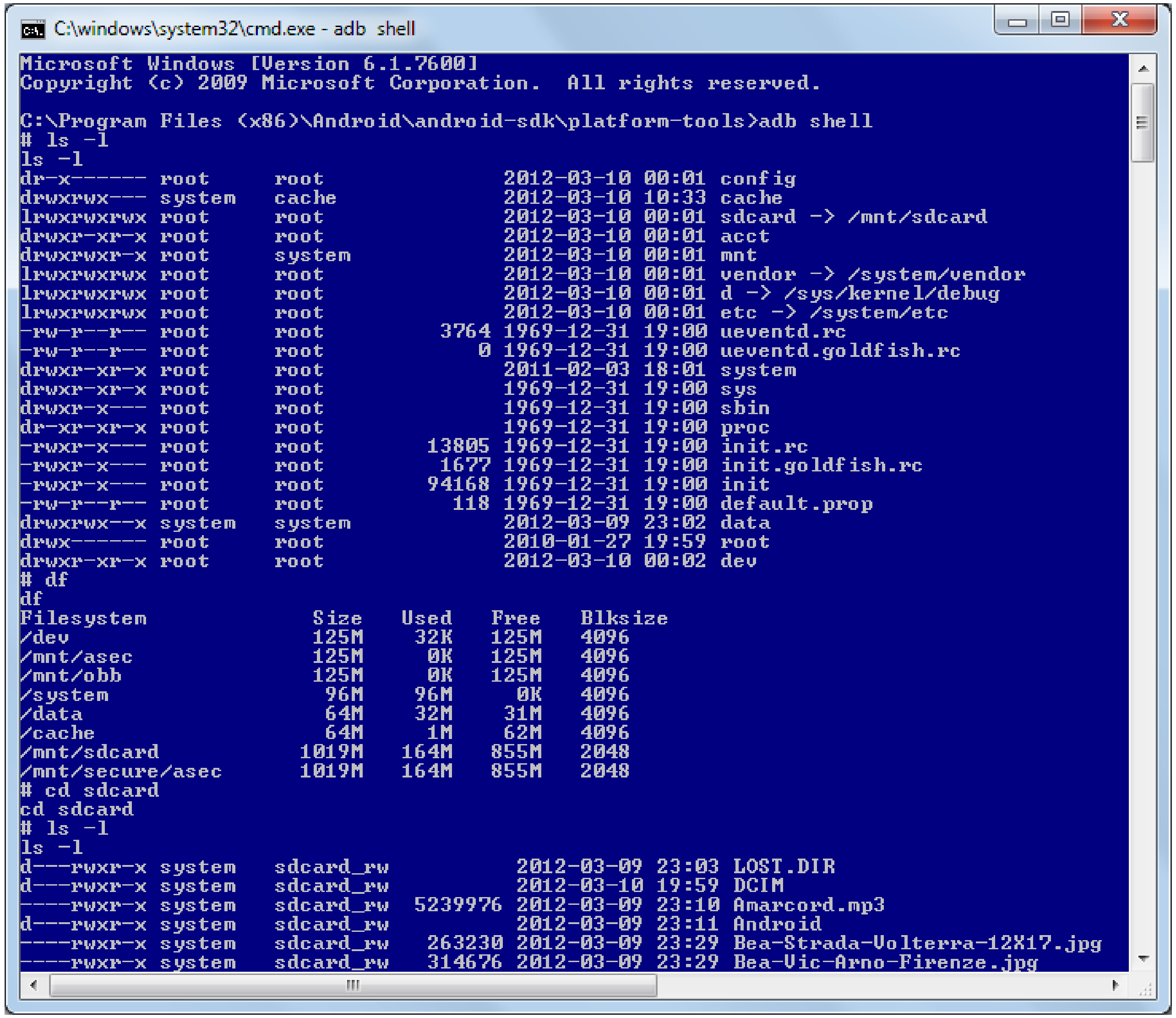




Android Emulator – Looking Under the Hood

Login into the Android OS shell

- Although it is not necessary, a developer may gain access to the innermost parts of the Android OS.
- For a Unix-like experience you can log into the system by executing the emulator and issuing selected shell commands.



```
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Program Files (x86)\Android\android-sdk\platform-tools>adb shell
# ls -l
ls -l
dr-x------ root     root              2012-03-10 00:01 config
drwxrwx--- system   cache             2012-03-10 10:33 cache
lrwxrwxrwx root     root              2012-03-10 00:01 sdcard -> /mnt/sdcard
drwxr-xr-x root     root              2012-03-10 00:01 acct
drwxrwxr-x root     system            2012-03-10 00:01 mnt
lrwxrwxrwx root     root              2012-03-10 00:01 vendor -> /system/vendor
lrwxrwxrwx root     root              2012-03-10 00:01 d -> /sys/kernel/debug
lrwxrwxrwx root     root              2012-03-10 00:01 etc -> /system/etc
-rw-r--r-- root     root         3764 1969-12-31 19:00 ueventd.rc
-rw-r--r-- root     root            0 1969-12-31 19:00 ueventd.goldfish.rc
drwxr-xr-x root     root              2011-02-03 18:01 system
drwxr-xr-x root     root              1969-12-31 19:00 sys
drwxr-x--- root     root              1969-12-31 19:00 sbin
dr-xr-xr-x root     root              1969-12-31 19:00 proc
-rwxr-x--- root     root        13805 1969-12-31 19:00 init.rc
-rwxr-x--- root     root         1677 1969-12-31 19:00 init.goldfish.rc
-rwxr-x--- root     root        94168 1969-12-31 19:00 init
-rw-r--r-- root     root          118 1969-12-31 19:00 default.prop
drwxrwx--x system   system            2012-03-09 23:02 data
drwx------ root     root              2010-01-27 19:59 root
drwxr-xr-x root     root              2012-03-10 00:02 dev
# df
df
Filesystem             Size   Used   Free   Blksize
/dev                   125M    32K   125M   4096
/mnt/asec              125M     0K   125M   4096
/mnt/obb               125M     0K   125M   4096
/system                 96M    96M     0K   4096
/data                   64M    32M    31M   4096
/cache                  64M     1M    62M   4096
/mnt/sdcard           1019M   164M   855M   2048
/mnt/secure/asec      1019M   164M   855M   2048
# cd sdcard
cd sdcard
# ls -l
ls -l
d---rwxr-x system   sdcard_rw          2012-03-09 23:03 LOST.DIR
d---rwxr-x system   sdcard_rw          2012-03-10 19:59 DCIM
----rwxr-x system   sdcard_rw  5239976 2012-03-09 23:10 Amarcord.mp3
d---rwxr-x system   sdcard_rw          2012-03-09 23:11 Android
----rwxr-x system   sdcard_rw   263230 2012-03-09 23:29 Bea-Strada-Volterra-12X17.jpg
----rwxr-x system   sdcard_rw   314676 2012-03-09 23:29 Bea-Vic-Arno-Firenze.jpg
```

Android Emulator – Looking Under the Hood

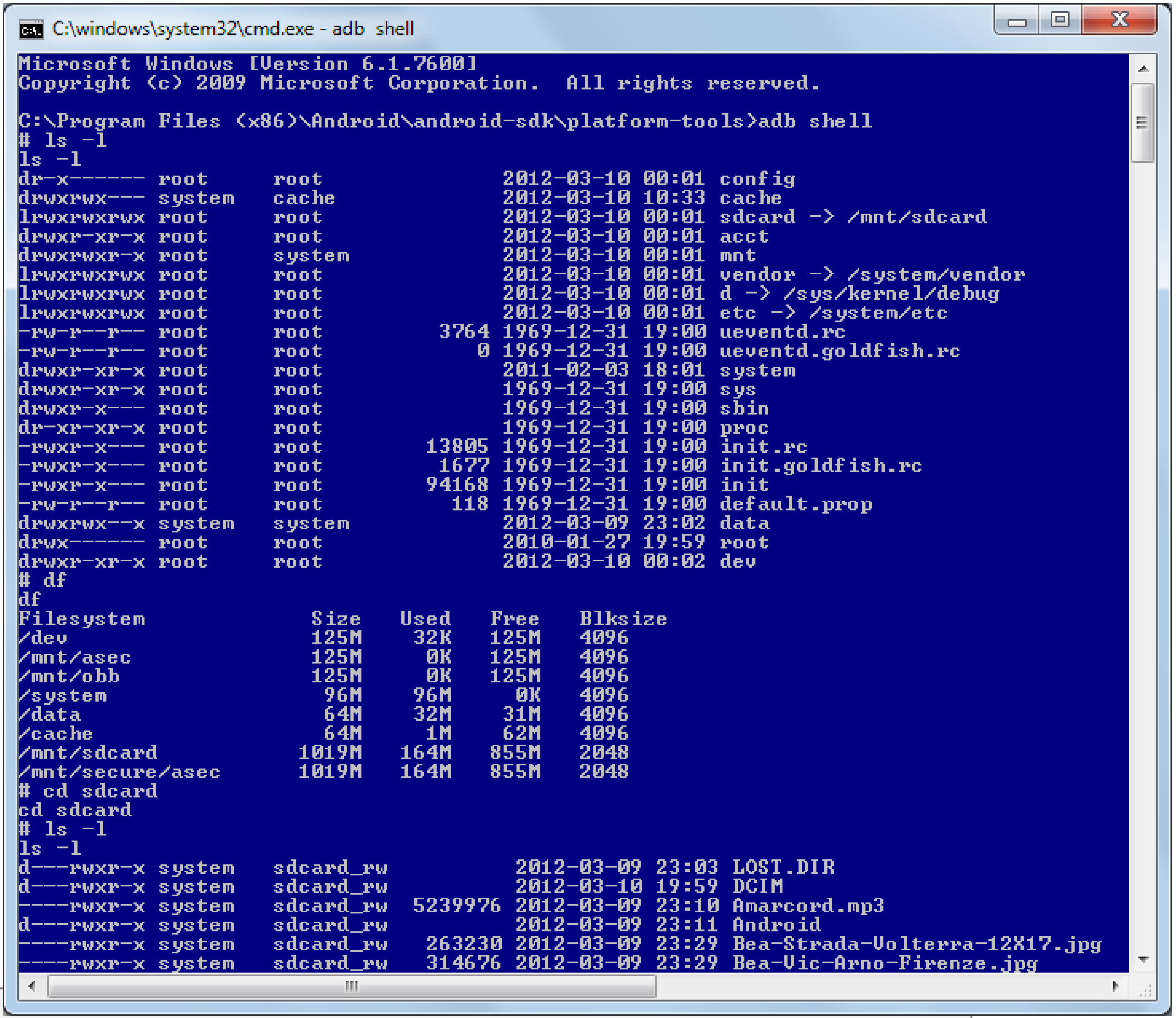
Login into the Android OS shell

STEPS

1. Use the Eclipse **AVD Manager** to start a selected AVD (say Gingerbread23)

2. At the DOS command prompt level run the Android Debug Bridge **(adb)** application

   adb shell



```
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Program Files (x86)\Android\android-sdk\platform-tools>adb shell
# ls -l
ls -l
dr-x------ root     root              2012-03-10 00:01 config
drwxrwx--- system   cache             2012-03-10 10:33 cache
lrwxrwxrwx root     root              2012-03-10 00:01 sdcard -> /mnt/sdcard
drwxr-xr-x root     root              2012-03-10 00:01 acct
drwxrwxr-x root     system            2012-03-10 00:01 mnt
lrwxrwxrwx root     root              2012-03-10 00:01 vendor -> /system/vendor
lrwxrwxrwx root     root              2012-03-10 00:01 d -> /sys/kernel/debug
lrwxrwxrwx root     root              2012-03-10 00:01 etc -> /system/etc
-rw-r--r-- root     root         3764 1969-12-31 19:00 ueventd.rc
-rw-r--r-- root     root            0 1969-12-31 19:00 ueventd.goldfish.rc
drwxr-xr-x root     root              2011-02-03 18:01 system
drwxr-xr-x root     root              1969-12-31 19:00 sys
drwxr-x--- root     root              1969-12-31 19:00 sbin
dr-xr-xr-x root     root              1969-12-31 19:00 proc
-rwxr-x--- root     root        13805 1969-12-31 19:00 init.rc
-rwxr-x--- root     root         1677 1969-12-31 19:00 init.goldfish.rc
-rwxr-x--- root     root        94168 1969-12-31 19:00 init
-rw-r--r-- root     root          118 1969-12-31 19:00 default.prop
drwxrwx--x system   system            2012-03-09 23:02 data
drwx------ root     root              2010-01-27 19:59 root
drwxr-xr-x root     root              2012-03-10 00:02 dev
# df
df
Filesystem             Size   Used   Free   Blksize
/dev                   125M    32K   125M   4096
/mnt/asec              125M     0K   125M   4096
/mnt/obb               125M     0K   125M   4096
/system                 96M    96M     0K   4096
/data                   64M    32M    31M   4096
/cache                  64M     1M    62M   4096
/mnt/sdcard           1019M   164M   855M   2048
/mnt/secure/asec      1019M   164M   855M   2048
# cd sdcard
cd sdcard
# ls -l
ls -l
d---rwxr-x system   sdcard_rw          2012-03-09 23:03 LOST.DIR
d---rwxr-x system   sdcard_rw          2012-03-10 19:59 DCIM
----rwxr-x system   sdcard_rw  5239976 2012-03-09 23:10 Amarcord.mp3
d---rwxr-x system   sdcard_rw          2012-03-09 23:11 Android
----rwxr-x system   sdcard_rw   263230 2012-03-09 23:29 Bea-Strada-Volterra-12X17.jpg
----rwxr-x system   sdcard_rw   314676 2012-03-09 23:29 Bea-Vic-Arno-Firenze.jpg
```

adb is a tool located in the directory:
C:\Your-SDK-Folder\Android\android-sdk\platform-tools\

Android Emulator – Looking Under the Hood

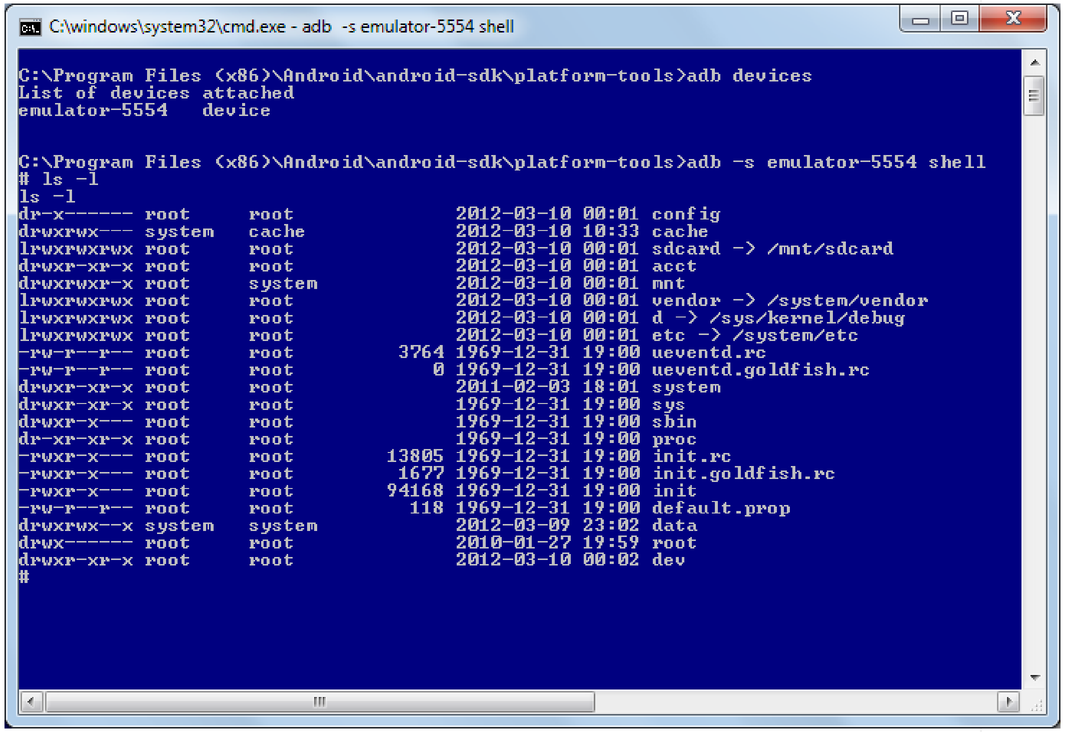
Android – Login into the OS shell

If more than one emulator is running (or your phone is physically connected to the computer using the USB cable) you need to identify the target.

Follow the next steps:

1. Get a list of attached devices

```
adb devices

List of devices attached
emulator-5554   device
emulator-5556   device
HT845GZ45737    device
```


C:\windows\system32\cmd.exe - adb -s emulator-5554 shell

```
C:\Program Files (x86)\Android\android-sdk\platform-tools>adb devices
List of devices attached
emulator-5554   device


C:\Program Files (x86)\Android\android-sdk\platform-tools>adb -s emulator-5554 shell
# ls -l
ls -l
dr-x------ root     root              2012-03-10 00:01 config
drwxrwx--- system   cache             2012-03-10 10:33 cache
lrwxrwxrwx root     root              2012-03-10 00:01 sdcard -> /mnt/sdcard
drwxr-xr-x root     root              2012-03-10 00:01 acct
drwxrwxr-x root     system            2012-03-10 00:01 mnt
lrwxrwxrwx root     root              2012-03-10 00:01 vendor -> /system/vendor
lrwxrwxrwx root     root              2012-03-10 00:01 d -> /sys/kernel/debug
lrwxrwxrwx root     root              2012-03-10 00:01 etc -> /system/etc
-rw-r--r-- root     root         3764 1969-12-31 19:00 ueventd.rc
-rw-r--r-- root     root            0 1969-12-31 19:00 ueventd.goldfish.rc
drwxr-xr-x root     root              2011-02-03 18:01 system
drwxr-xr-x root     root              1969-12-31 19:00 sys
drwxr-x--- root     root              1969-12-31 19:00 sbin
dr-xr-xr-x root     root              1969-12-31 19:00 proc
-rwxr-x--- root     root        13805 1969-12-31 19:00 init.rc
-rwxr-x--- root     root         1677 1969-12-31 19:00 init.goldfish.rc
-rwxr-x--- root     root        94168 1969-12-31 19:00 init
-rw-r--r-- root     root          118 1969-12-31 19:00 default.prop
drwxrwx--x system   system            2012-03-09 23:02 data
drwx------ root     root              2010-01-27 19:59 root
drwxr-xr-x root     root              2012-03-10 00:02 dev
#
```

2. Run the **adb** application as follows:

```
adb  -s emulator-5554  shell
```

Remember, the **adb** tool is located at **C:\Program Files (x86)\Android\android-sdk\platform-tools\**

Android Emulator – Looking Under the Hood

Hacking: Moving an app from a Rooted Phone to the Emulator

If you want to transfer an app that is currently installed in your developer's phone to the emulator, follow the next steps:

1. Run command shell: > **adb devices** (find out your hardware's id, say **HT096P800176**)

2. Pull the file from the device to your computer's file system. Enter the command
   adb -s HT096P800176 pull data/app/theInstalled.apk   c:/theInstalled.apk

3. Disconnect your Android phone

4. Run an instance of the Emulator

5. Now install the app on the emulator using the command
   adb -s emulator-5554 install c:\theInstalledApp.apk
   adb -s emulator-5554 uninstall data/app/theInstalled.apk   ⟵ *to uninstall*

You should see a message indicating the size of the installed package, and finally: *Success*.

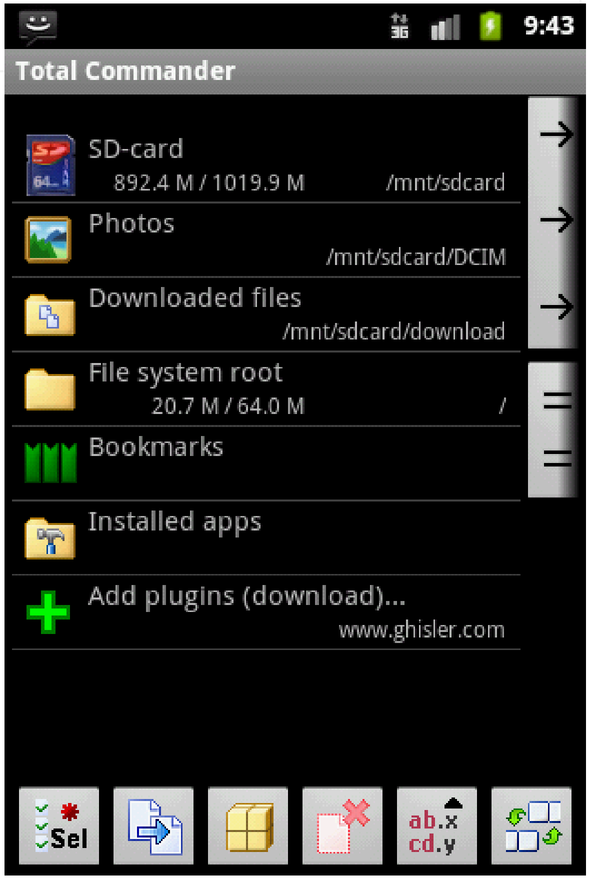
Android Emulator – Looking Under the Hood

More Hacking: Install TotalCommander for Android

TotalCommander is a useful Windows file manager that has been ported to Android. You could use it to administer the folders and files in the system's flash memory and SD card of your emulator or device.

To install the app in your emulator follow the next steps

1. Start the emulator's web browser on the URL: http://www.ghisler.com/android.htm
2. Press **Ctrl + Click** on the "**direct download"** hyperlink to start the app's download.
3. Wait for completion (scroll down Notification line if necessary)
4. Follow setup instructions.



Android Emulator – Looking Under the Hood

Android – Login into the OS shell

Android accepts a number of Linux shell commands including the useful set below

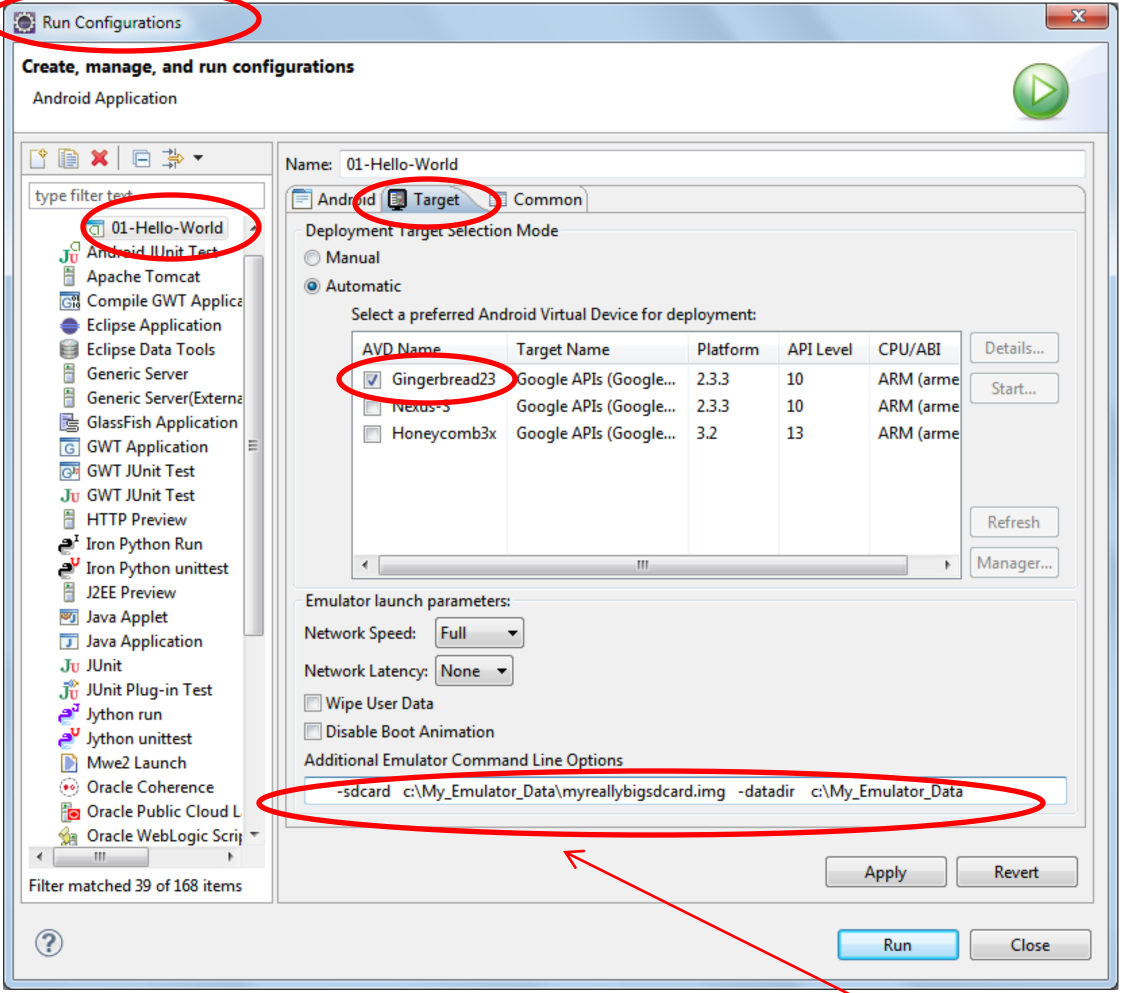
```
ls .................. show directory (alphabetical order)
mkdir ............... make a directory
rmdir ............... remove directory
rm -r ............... to delete folders with files
rm .................. remove files
mv .................. moving and renaming files
cat ................. displaying short files
cd .................. change current directory
pwd ................. find out what directory you are in
df .................. shows available disk space
chmod ............... changes permissions on a file
date ................ display date
exit ................ terminate session
```

There is no copy (**cp**) command in Android, but you could use **cat** instead.
For instance:

```
# cat data/app/theInstalledApp.apk > cache/theInstalledApp.apk
```

Android Emulator

Using an alternate SD card & userData Image



Run Configurations

Create, manage, and run configurations

Android Application

type filter text

01-Hello-World
Android JUnit Test
Apache Tomcat
Compile GWT Applica
Eclipse Application
Eclipse Data Tools
Generic Server
Generic Server(Externa
GlassFish Application
GWT Application
GWT JUnit Test
GWT JUnit Test
HTTP Preview
Iron Python Run
Iron Python unittest
J2EE Preview
Java Applet
Java Application
JUnit
JUnit Plug-in Test
Jython run
Jython unittest
Mwe2 Launch
Oracle Coherence
Oracle Public Cloud L
Oracle WebLogic Scrip

Filter matched 39 of 168 items

Name: 01-Hello-World

Android | Target | Common

Deployment Target Selection Mode

( ) Manual
(●) Automatic

Select a preferred Android Virtual Device for deployment:

AVD Name	Target Name	Platform	API Level	CPU/ABI
✓ Gingerbread23	Google APIs (Google...	2.3.3	10	ARM (arme
Nexus-S	Google APIs (Google...	2.3.3	10	ARM (arme
Honeycomb3x	Google APIs (Google...	3.2	13	ARM (arme

Emulator launch parameters:

Network Speed: Full

Network Latency: None

☐ Wipe User Data

☐ Disable Boot Animation

Additional Emulator Command Line Options

-sdcard c:\My_Emulator_Data\myreallybigsdcard.img -datadir c:\My_Emulator_Data

From the Eclipse menu create a new launch configuration:

Run >

Run Configurations >

New icon

On the **Target** panel:

1. Select existing AVD (Gingerbread in this example)
2. Enter additional Command Line Options (see caption below)
3. Click on **Apply** > **Run**

Additional Emulator Command Line Options:
-sdcard c:\My_Emulator_Data\myreallybigsdcard.img **-datadir** c:\My_Emulator_Data

Android Emulator / SMS

Sending Text Messages from your Window's PC to the Emulator

1. Start the emulator.

2. Open a new DOS command shell and type :
   ```
   c:> adb devices
   ```
   this way you get to know the emulator's numeric port id (usually **5554**, **5556**, and so on)

3. Connect to the console using telnet command like:
   ```
   c:> telnet localhost 5554
   ```

4. After receiving the telnet prompt you can send a text message with the command (no quotes needed for the message)
   ```
   sms send <Sender's phone number> <text message>
   ```

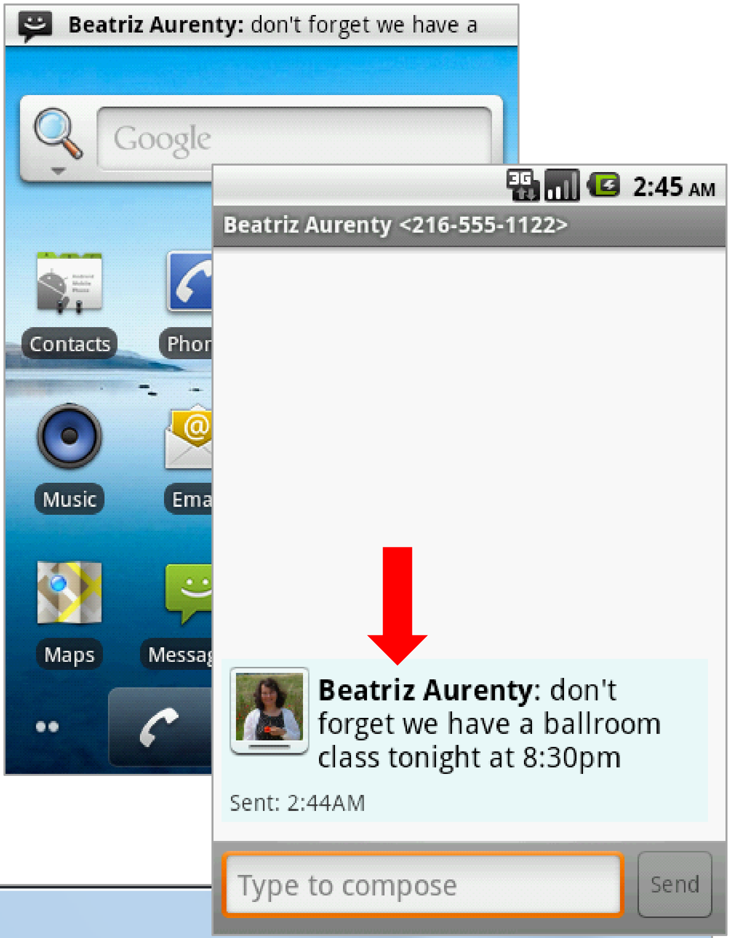
Windows7 – temporarily install Telnet Client by using a command line

1. Open a command prompt window.
2. Click **Start**, type **cmd** in the **Start Search** box, and then press **ENTER**.
3. Type the following command: `pkgmgr /iu:"TelnetClient”`

Android Emulator / SMS

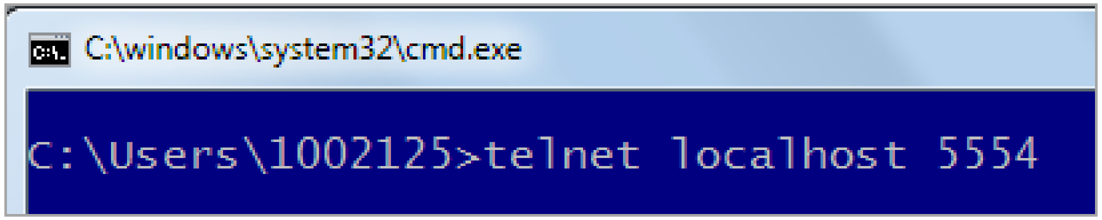
Example:

Sending a text Message (SMS) from your PC to the Emulator

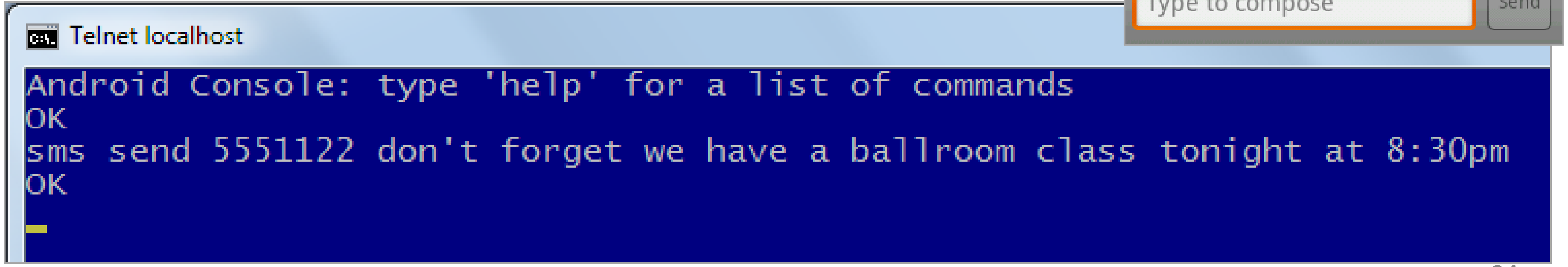

Beatriz Aurenty: don't forget we have a
Google
Contacts
Phon
Music
Ema
Maps
Messag
2:45 AM
Beatriz Aurenty <216-555-1122>
Type to compose
Send

Beatriz Aurenty: don't forget we have a ballroom class tonight at 8:30pm

Sent: 2:44AM


C:\windows\system32\cmd.exe

```
C:\Users\1002125>telnet localhost 5554
```


Telnet localhost

```
Android Console: type 'help' for a list of commands
OK
sms send 5551122 don't forget we have a ballroom class tonight at 8:30pm
OK
_
```

Android Emulator / Voice

Making a Phone Call from your PC to the Emulator

1. Start the emulator.

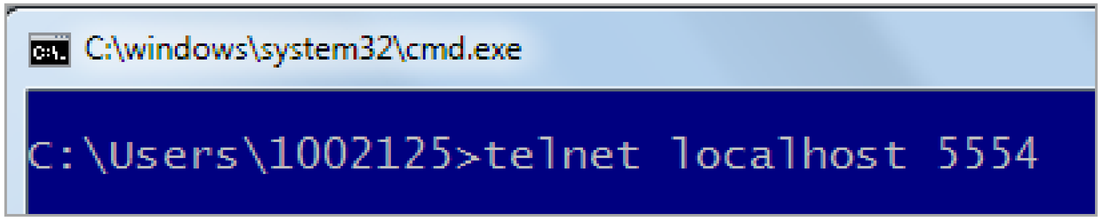
2. Open a new shell and type :
   adb devices
   to know the emulator's numeric port id (usually **5554**, **5556**, and so on)

3. Connect to the console using telnet command like:
   telnet localhost 5554 (this is the 'number' to be called)

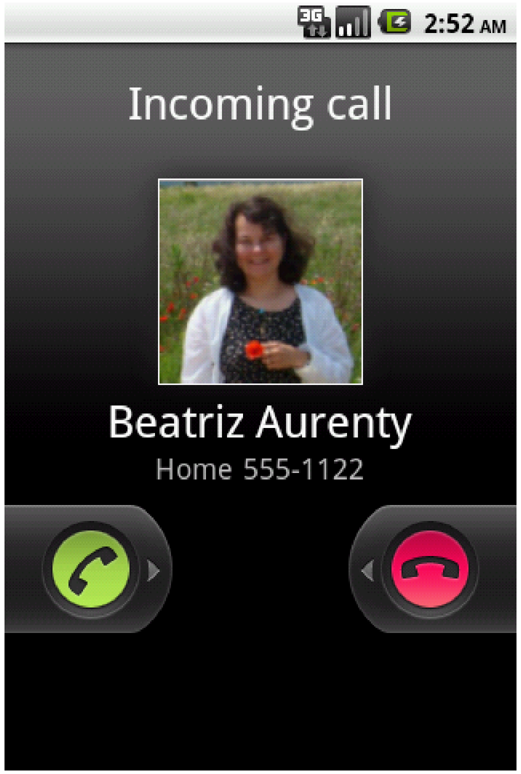
4. After receiving the telnet prompt you can place a call (voice) with the command
   gsm call <caller's phone number>

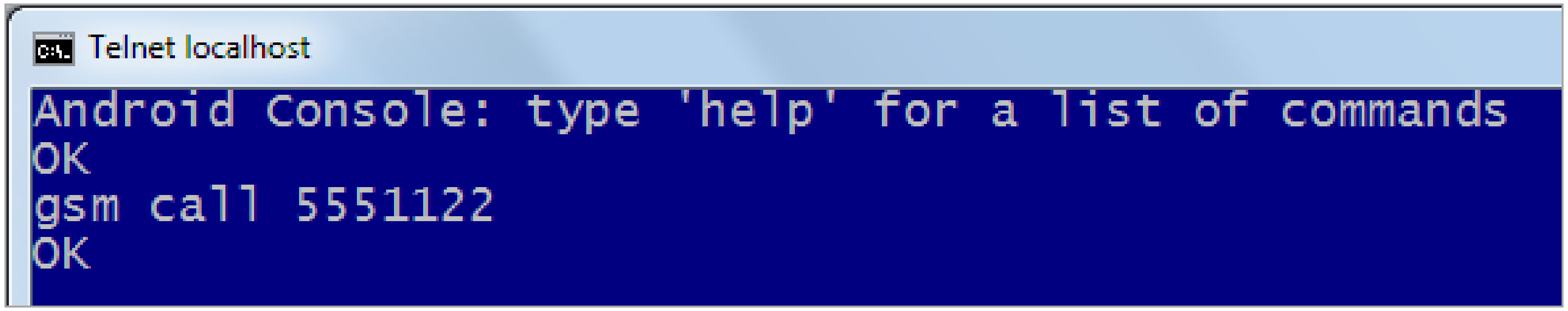
Android Emulator / Voice

Example: Making a Phone Call to the Emulator



```
C:\windows\system32\cmd.exe
C:\Users\1002125>telnet localhost 5554
```





```
Telnet localhost
Android Console: type 'help' for a list of commands
OK
gsm call 5551122
OK
```

Android Emulator

Using Eclipse's DDMS facility

DDMS Emulator Controls

It is *much simpler* to test telephony operations (SMS/Voice) as well as GPS services using the controls included in the Eclipse DDMS perspective

1. **Telephony Status** - change the state of the phone's Voice and Data plans (home, roaming, searching, etc.), and simulate different kinds of network Speed and Latency (GPRS, EDGE, UTMS, etc.).

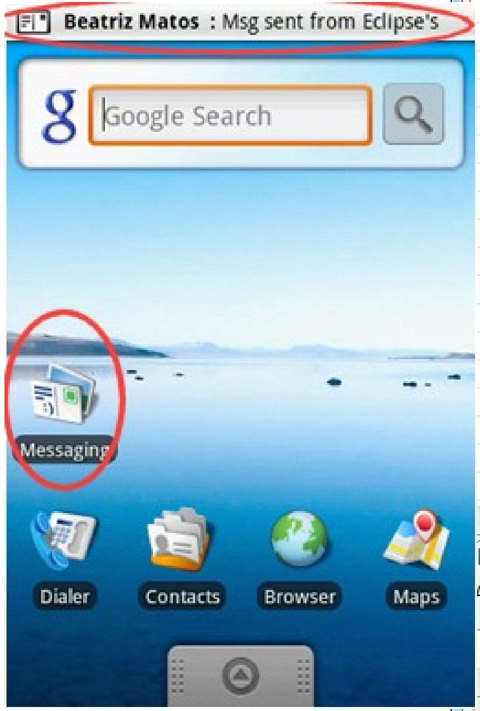
2. **Telephony Actions** - perform simulated phone calls and SMS messages to the emulator.

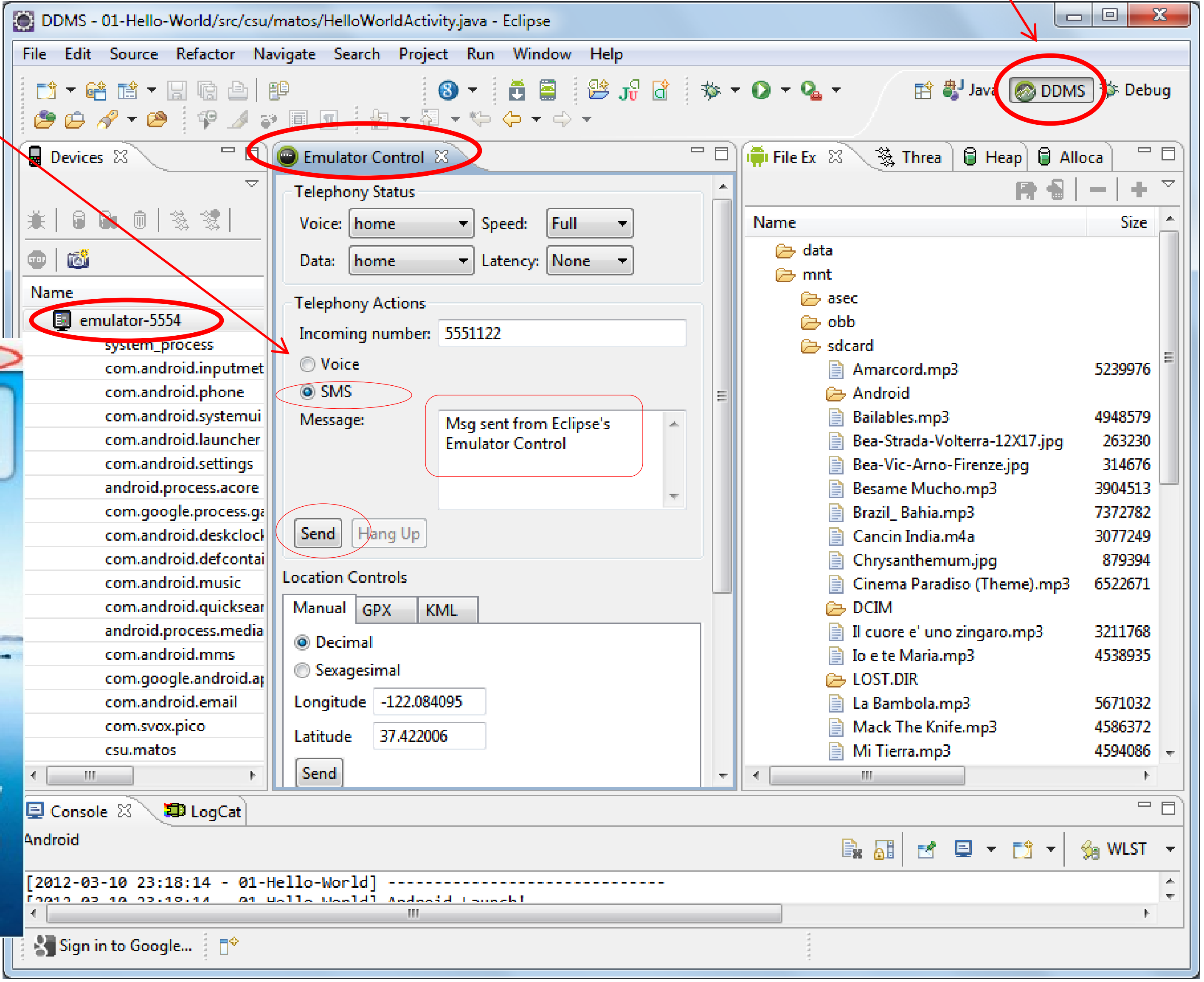
3. **Location Controls** - send mock location data to the emulator so that you can perform location-aware operations like GPS mapping.
   - Manually send individual longitude/latitude coordinates to the device. Click **Manual**, select the coordinate format, fill in the fields and click **Send**.
   - Use a **GPX file** describing a route for playback to the device.
   - Use a **KML** file to place multiple *placemarker points* on a map

Android Emulator

Using Eclipse to test Emulator's Telephony Actions

Send text-messages
Make a phone call




DDMS - 01-Hello-World/src/csu/matos/HelloWorldActivity.java - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help
Java DDMS Debug
Devices
Emulator Control
File Ex Threa Heap Alloca

Name
emulator-5554
system_process
com.android.inputmet
com.android.phone
com.android.systemui
com.android.launcher
com.android.settings
android.process.acore
com.google.process.ga
com.android.deskclock
com.android.defcontai
com.android.music
com.android.quicksear
android.process.media
com.android.mms
com.google.android.ap
com.android.email
com.svox.pico
csu.matos

Telephony Status
Voice: home Speed: Full
Data: home Latency: None

Telephony Actions
Incoming number: 5551122
Voice [✓ SMS]
Message: Msg sent from Eclipse's Emulator Control
Send Hang Up

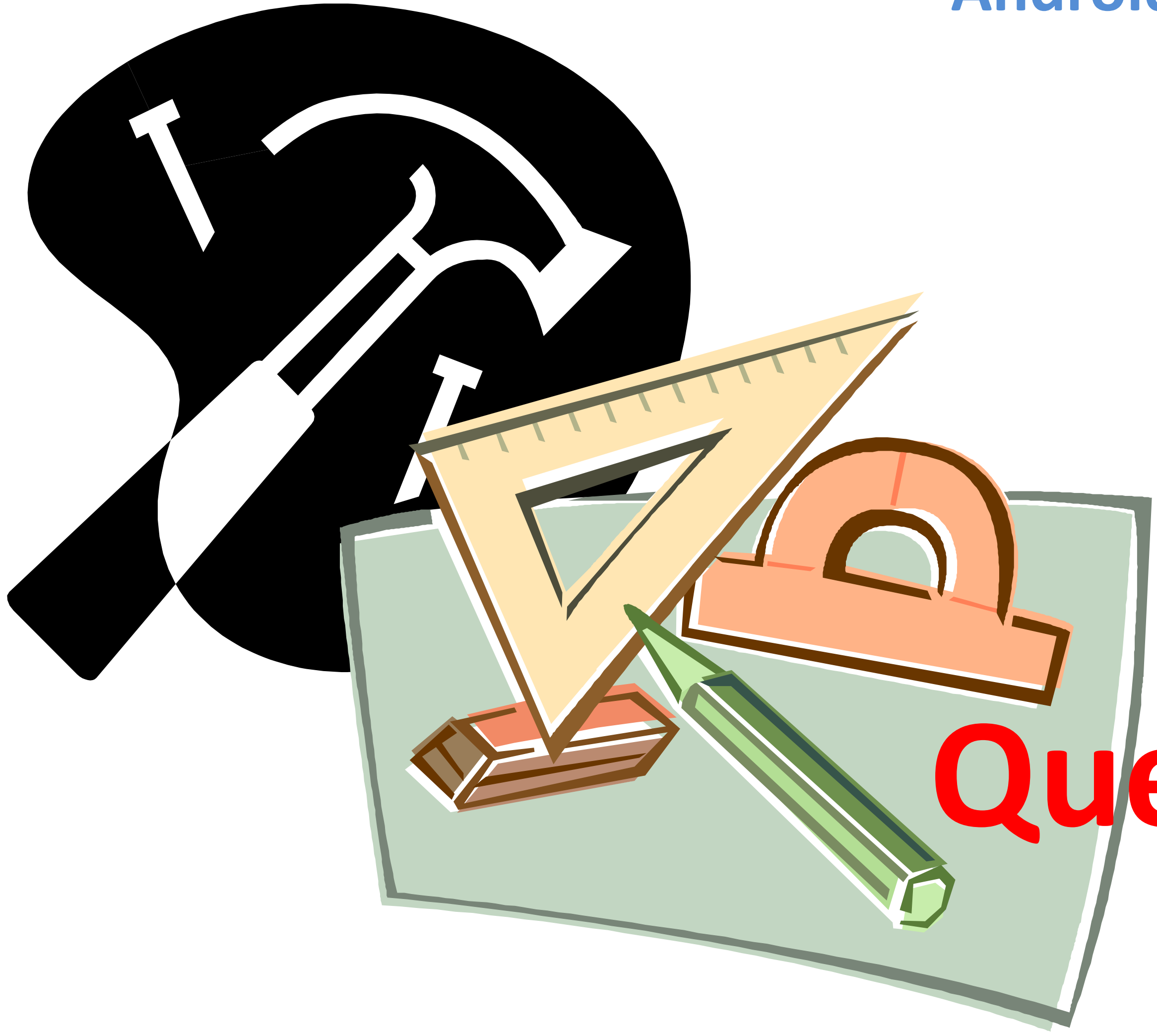
Location Controls
Manual GPX KML
[✓ Decimal] Sexagesimal
Longitude -122.084095
Latitude 37.422006
Send

Name	Size
data	
mnt	
asec	
obb	
sdcard	
Amarcord.mp3	5239976
Android	
Bailables.mp3	4948579
Bea-Strada-Volterra-12X17.jpg	263230
Bea-Vic-Arno-Firenze.jpg	314676
Besame Mucho.mp3	3904513
Brazil_ Bahia.mp3	7372782
Cancin India.m4a	3077249
Chrysanthemum.jpg	879394
Cinema Paradiso (Theme).mp3	6522671
DCIM	
Il cuore e' uno zingaro.mp3	3211768
Io e te Maria.mp3	4538935
LOST.DIR	
La Bambola.mp3	5671032
Mack The Knife.mp3	4586372
Mi Tierra.mp3	4594086

Android

```
[2012-03-10 23:18:14 - 01-Hello-World] ------------------------------
```

Lesson 2: Android Setup & Emulator

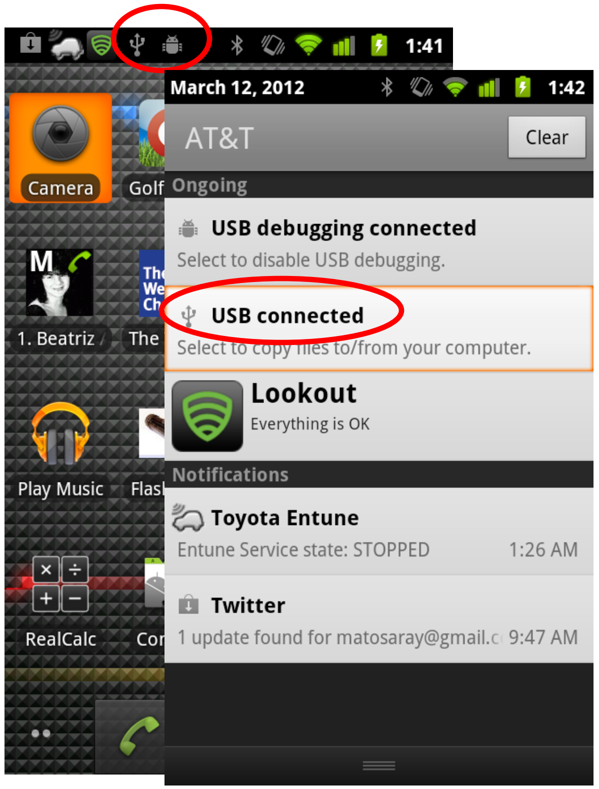


Questions ?

Android Emulator

Appendix 1 – Connecting your Hardware Device to the Computer

1. Make sure the USB driver has been installed in your PC (click SDK Manager > Extras > check box [*Google USB driver package*] to install)
2. Use a mini-USB cable to connect the device to your computer.
3. Expand the Notification bar. Click on [*USB connected*] option.
4. Click on [*Turn on USB storage*] to mount the device.
5. Now you could now use the Eclipse-ADT-File Explorer and your Window's Explorer tool to pull/push/delete/rename files to the device.



March 12, 2012 1:42

AT&T Clear

Ongoing

USB debugging connected
Select to disable USB debugging.

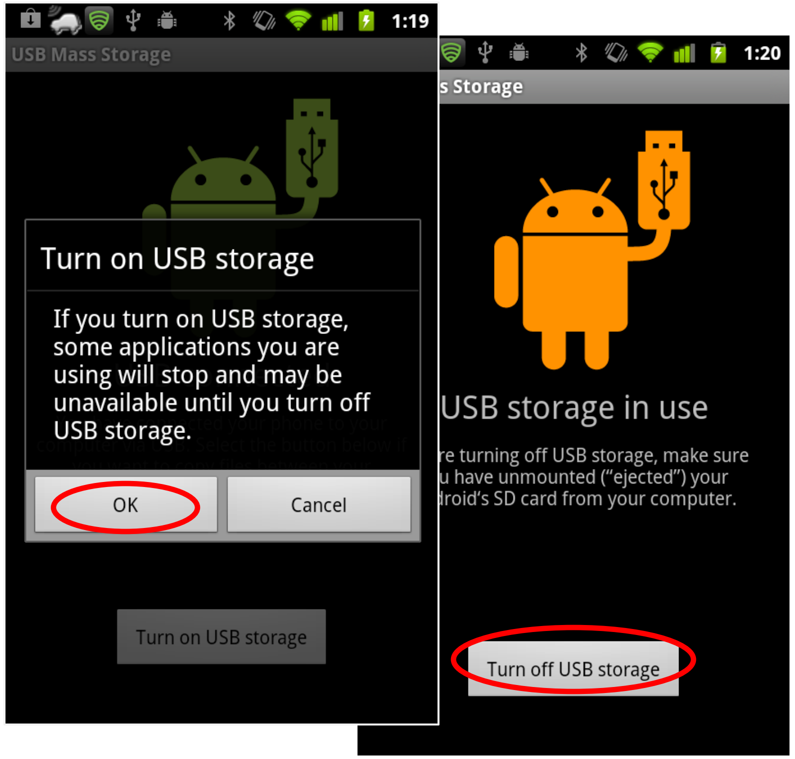
USB connected
Select to copy files to/from your computer.

Lookout
Everything is OK

Notifications

Toyota Entune
Entune Service state: STOPPED 1:26 AM

Twitter
1 update found for matosaray@gmail.c 9:47 AM



Turn on USB storage

If you turn on USB storage, some applications you are using will stop and may be unavailable until you turn off USB storage.

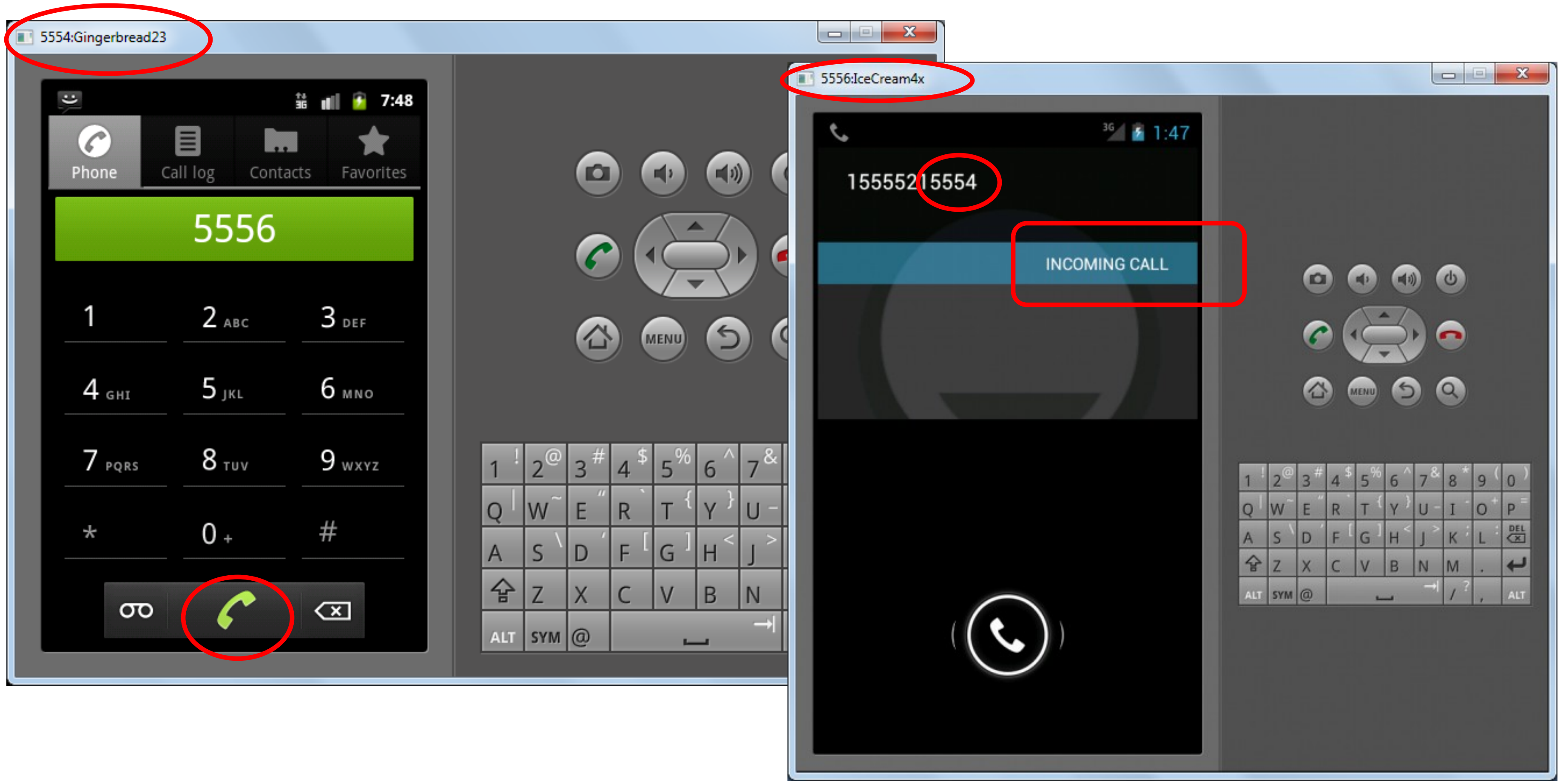
USB storage in use

re turning off USB storage, make sure u have unmounted ("ejected") your droid's SD card from your computer.

Android Emulator

Appendix 2 – Emulator to Emulator Communication (SMS & Voice)

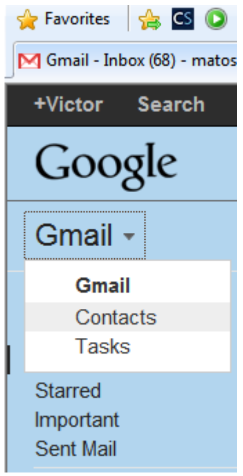
1. Run two instances of the emulator (typical IDs are: 5554, 5556, …)
2. Dial (or send SMS) from one of them (say 5554) to the other (5556)
3. Press the Green/Red call buttons to accept/terminate the call
4. Try sending SMS (use numbers 5554 and 5556)



Android Emulator

Appendix 3.
How to Transfer and Sync Your Google Contacts into the Emulator

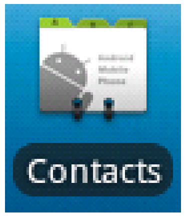
Taken from:
http://stackoverflow.com/questions/1114052/importing-gmail-contacts-on-android-emulator



1. Go to your Gmail account using a web browser, click on **Gmail** > **Contacts** on the left sidebar.

2. Select all the contacts you want on your emulator/phone. Then click on **More > Export** and select **vCard** format. Download the "contacs.vcf" file to your PC.

3. Push the contacs.vcf file from the PC to the emulator's **SD card**.



4. Open the emulator's **Contacts** app hit **Menu** > **Import**.

5. Choose the option *Import from SD card*.