Part 2-b

Android Environment Emulator

Victor Matos
Cleveland State University

Notes are based on:
http://developer.android.com/index.html
http://developer.android.com/guide/developing/tools/emulator.html

Android Emulator

- The Android SDK includes a mobile device emulator -- a virtual mobile device that runs on your computer.

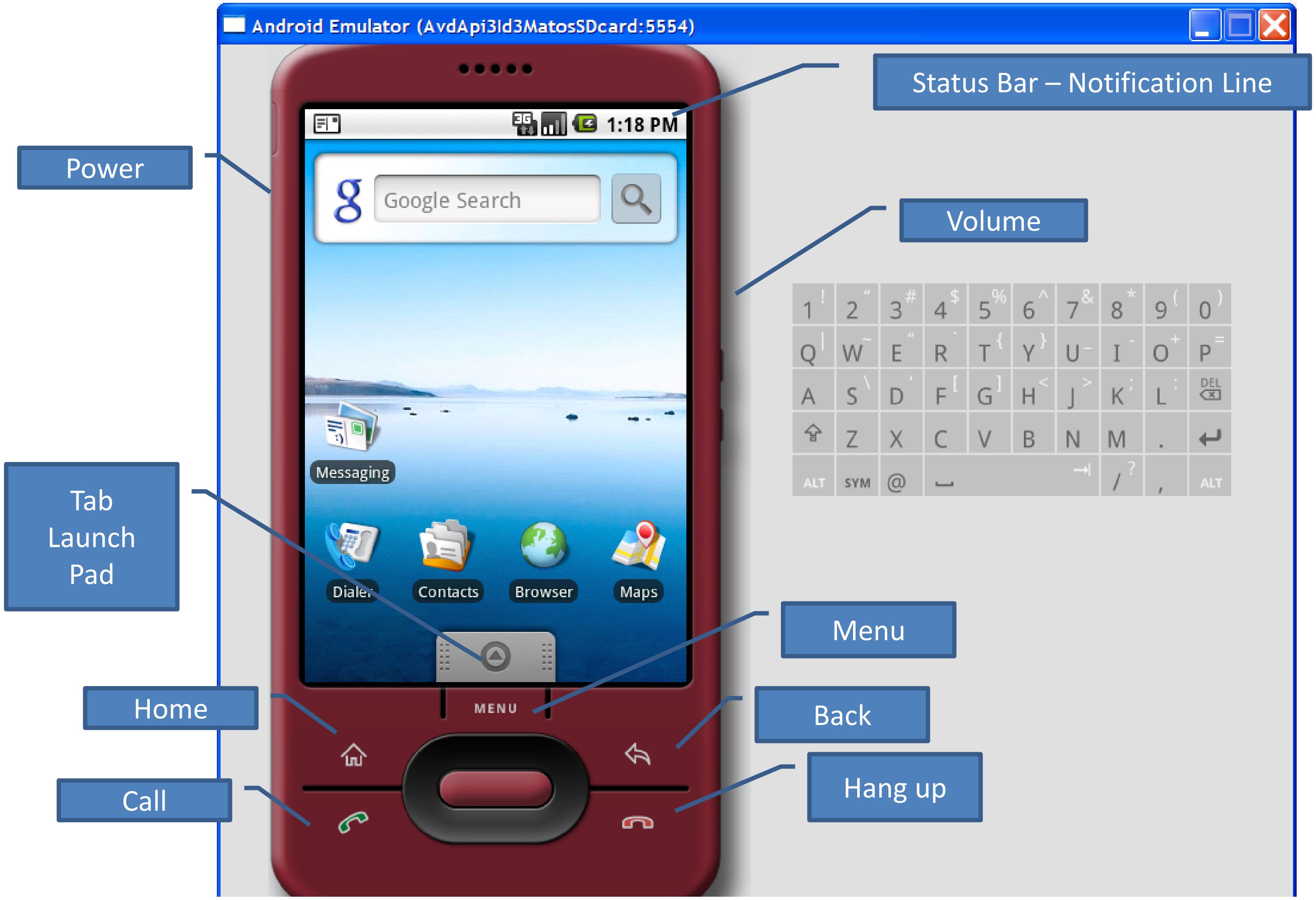
- The emulator lets you prototype, develop, and test Android applications without using a physical device.

- The Android emulator mimics *all* of the hardware and software features of a typical mobile device, except that it can not receive or place actual phone calls.

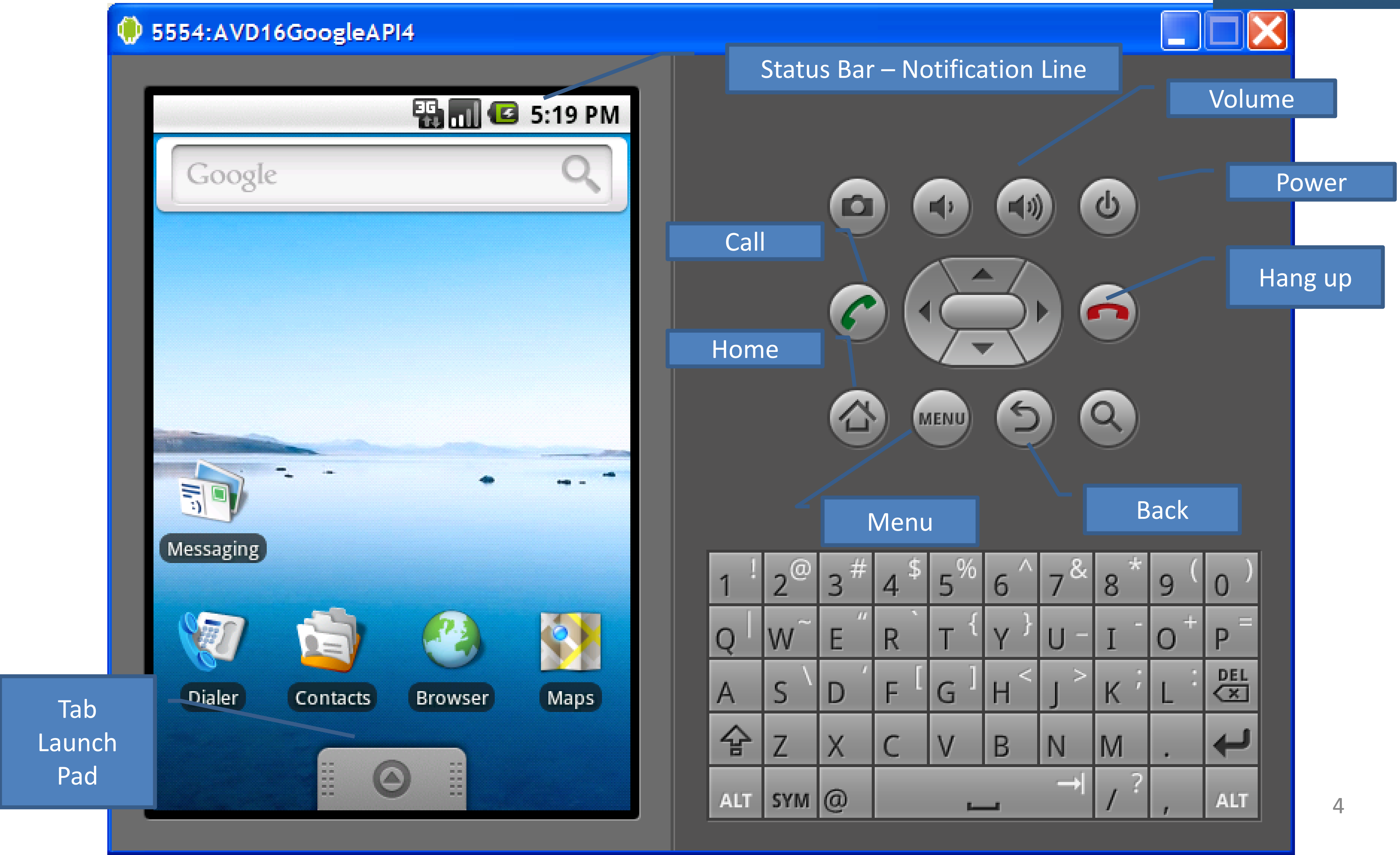
- It provides a variety of navigation and control keys, which you can "press" using your mouse or keyboard to generate events for your application.

- It also provides a screen in which your application is displayed, together with any other Android applications running.

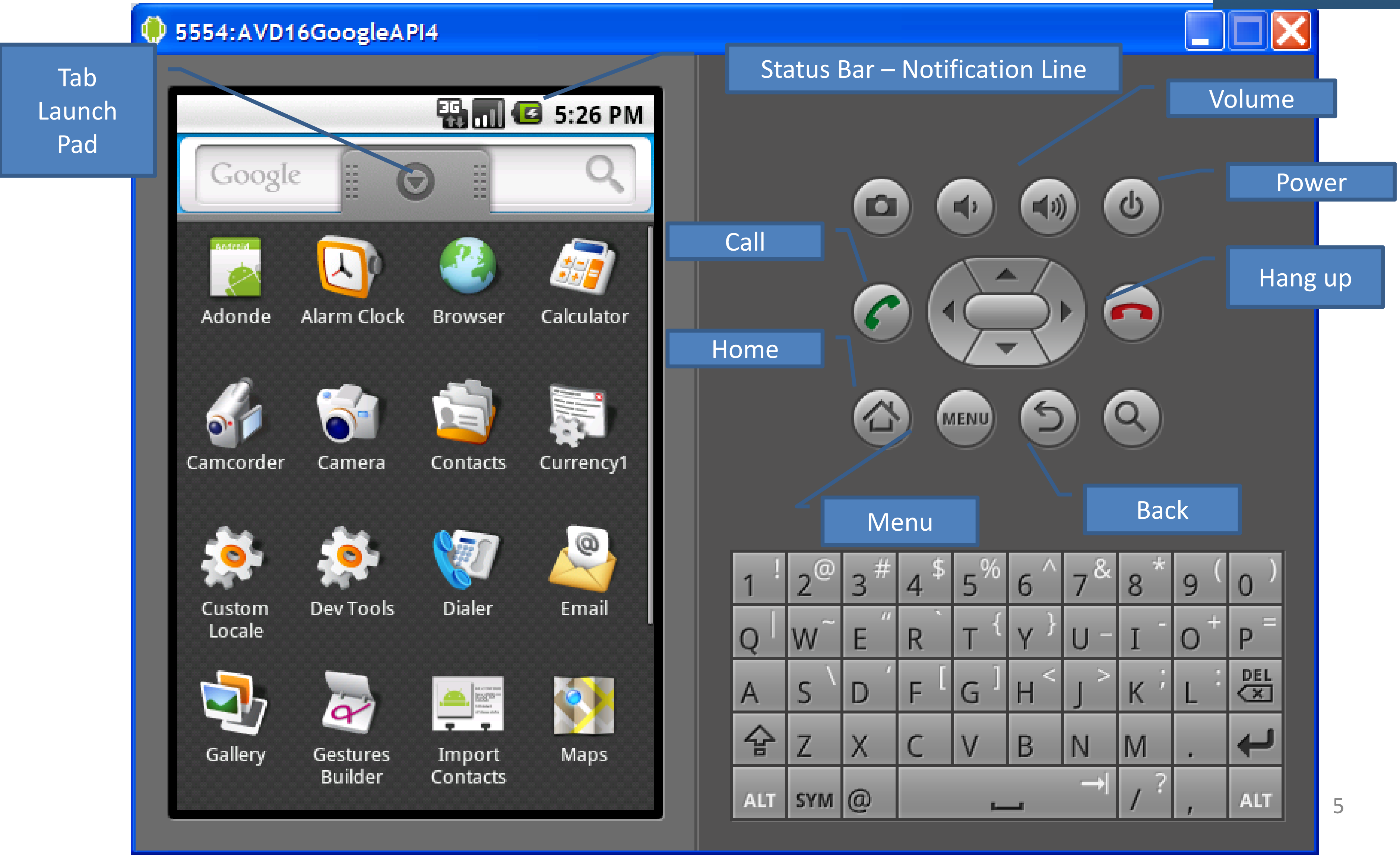
Android Emulator v1.5 Skin



Android Emulator v1.6 Skin



Android Emulator v1.6 Skin



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 keyboard keys

Keypad keys only work when *NumLock* is deactivated.

Android Emulator

Features - Emulating First Generation Android Phones

The Android emulator supports many hardware features likely to be found on mobile devices (such as the HTC-G1), including:

1. An ARMv5 CPU and the corresponding memory-management unit (MMU)
2. A 16-bit LCD display (mimicking 360 x 480 pixels)
3. One or more keyboards (a Qwerty-based keyboard and associated Dpad/Phone buttons)
4. A sound chip with output and input capabilities
5. Flash memory partitions (emulated through disk image files on the development machine)
6. A GSM modem, including a simulated SIM Card

Android Emulator

Nexus One (newer Google developer phone)

nexus one™

Size and weight

Height	119 mm
Width	59.8 mm
Depth	11.5 mm
Weight	130 g (with battery) 100 g (without battery)

Display
3.7 inch (diagonal) widescreen WVGA AMOLED touchscreen

800 x 480 pixels

100,000:1 typical contrast ratio

1ms typical response rate

Camera & Flash
5 megapixels

Autofocus from 6 cm to infinity

2X digital zoom

LED flash

User can include location of photos from phone's AGPS receiver

Video captured at 720x480 pixels at 20 frames per second or higher, depending on lighting conditions

Cellular & Wireless
3 UMTS bands (either 900/AWS/2100 MHz or 850/1900/2100 MHz)

HSDPA 7.2 Mbps

HSUPA 2 Mbps

GSM/EDGE (850, 900, 1800, 1900 MHz)

Wi-Fi (802.11b/g)

Bluetooth 2.1 + EDR

A2DP stereo Bluetooth

Power and battery
Removable 1400 mAh battery

Charges at 480 mA from USB, at 980 mA from supplied charger

Talk time	Up to 10 hours on 2G Up to 7 hours on 3G
Standby time	Up to 290 hours on 2G Up to 250 hours on 3G
Internet use	Up to 5 hours on 3G Up to 6.5 hours on Wi-Fi
Video playback	Up to 7 hours
Audio playback	Up to 20 hours

Processor
Qualcomm QSD 8250 1 GHz

Operating system
Android Mobile Technology Platform 2.1 (Eclair)

Capacity
512 MB Flash

512 MB RAM

4 GB Micro SD Card (Expandable to 32 GB)

Location
Assisted global positioning system (AGPS) receiver

Cell tower and Wi-Fi positioning

Digital compass

Accelerometer

Some phones in the market already surpass these specs (Fall 2010)

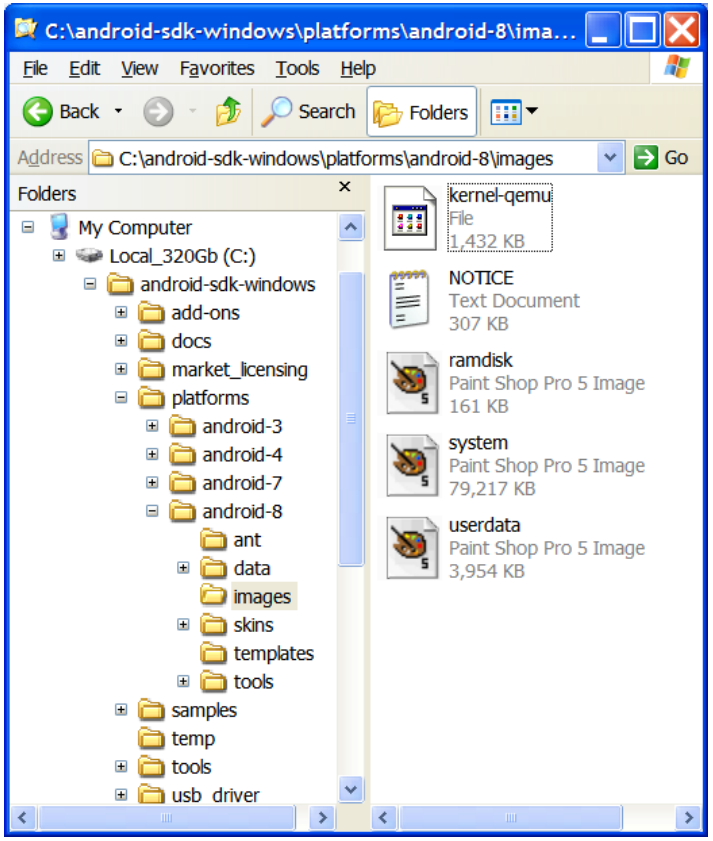
Android Emulator

Working with Emulator Disk Images

The emulator uses mountable disk images (*ANDROID SYSTEM IMAGE*) stored on your development machine to simulate flash (or similar) partitions on an actual device.

For example, it uses **disk images** containing
(1) an emulator-specific kernel,
(2) the Android system,
(3) a ram-disk image, and
(4) writeable images for user data and simulated SD card.

*By default, the Emulator always looks for the disk images in the private storage area of the AVD in use (**c:\android-sdk-windows\platform\ …**)*


C:\android-sdk-windows\platforms\android-8\ima…

Address C:\android-sdk-windows\platforms\android-8\images

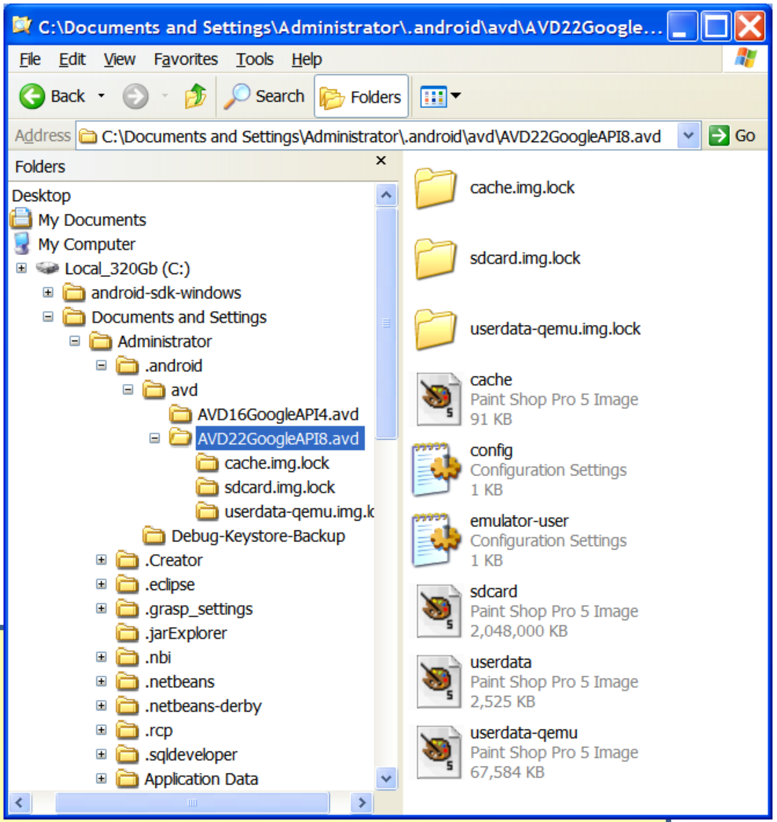
Folders:
- My Computer
  - Local_320Gb (C:)
    - android-sdk-windows
      - add-ons
      - docs
      - market_licensing
      - platforms
        - android-3
        - android-4
        - android-7
        - android-8
          - ant
          - data
          - images
          - skins
          - templates
          - tools
      - samples
      - temp
      - tools
      - usb_driver

kernel-qemu	File	1,432 KB
NOTICE	Text Document	307 KB
ramdisk	Paint Shop Pro 5 Image	161 KB
system	Paint Shop Pro 5 Image	79,217 KB
userdata	Paint Shop Pro 5 Image	3,954 KB

Android Emulator

Working with Emulator Disk Images

If no platform images exist there
when the Emulator is launched,
it creates the images in the
AVD directory based on
default versions stored in the SDK.


C:\Documents and Settings\Administrator\.android\avd\AVD22Google...

File Edit View Favorites Tools Help

Back Search Folders

Address C:\Documents and Settings\Administrator\.android\avd\AVD22GoogleAPI8.avd Go

Folders

```
Desktop
My Documents
My Computer
⊞ Local_320Gb (C:)
  ⊞ android-sdk-windows
  ⊟ Documents and Settings
    ⊟ Administrator
      ⊟ .android
        ⊟ avd
            AVD16GoogleAPI4.avd
          ⊟ AVD22GoogleAPI8.avd
              cache.img.lock
              sdcard.img.lock
              userdata-qemu.img.lo
          Debug-Keystore-Backup
      ⊞ .Creator
      ⊞ .eclipse
      ⊞ .grasp_settings
        .jarExplorer
      ⊞ .nbi
      ⊞ .netbeans
      ⊞ .netbeans-derby
      ⊞ .rcp
      ⊞ .sqldeveloper
      ⊞ Application Data
```

Name	Type	Size
cache.img.lock		
sdcard.img.lock		
userdata-qemu.img.lock		
cache	Paint Shop Pro 5 Image	91 KB
config	Configuration Settings	1 KB
emulator-user	Configuration Settings	1 KB
sdcard	Paint Shop Pro 5 Image	2,048,000 KB
userdata	Paint Shop Pro 5 Image	2,525 KB
userdata-qemu	Paint Shop Pro 5 Image	67,584 KB

Note:
The default storage location for AVDs is in

~/.android/avd on OS X and Linux,
C:\Documents and Settings\<user>\.android\avd\... on Windows XP, and
C:\Users\<user>\.android on Windows Vista.

Android Emulator

Creating an AVD using the android tool

Listing targets

To generate a list of system image targets, use this command:

android list targets

```
C:\Documents and
Settings\Administrator\.android\avd\AVD22GoogleAPI8.a
vd>a
Available Android targets:
id: 1 or "android-3"
    Name: Android 1.5
    Type: Platform
    API level: 3
    Revision: 4
    Skins: HVGA (default), HVGA-L, HVGA-P, QVGA-L,
QVGA-P
id: 2 or "Google Inc.:Google APIs:3"
    Name: Google APIs
    Type: Add-On
    Vendor: Google Inc.
    Revision: 3
    Description: Android + Google APIs
    Based on Android 1.5 (API level 3)
    Libraries:
     * com.google.android.maps (maps.jar)
        API for Google Maps
    Skins: QVGA-P, HVGA-L, HVGA (default), QVGA-L,
HVGA-P
id: 3 or "android-4"
    Name: Android 1.6
    Type: Platform
    API level: 4
    Revision: 3
    Skins: HVGA (default), QVGA, WVGA800, WVGA854
id: 4 or "Google Inc.:Google APIs:4"
    Name: Google APIs
    Type: Add-On
    Vendor: Google Inc.
    Revision: 2
    Description: Android + Google APIs
    Based on Android 1.6 (API level 4)
    Libraries:
     * com.google.android.maps (maps.jar)
        API for Google Maps
    Skins: WVGA854, HVGA (default), WVGA800, QVGA
id: 5 or "android-7"
    Name: Android 2.1-update1
    Type: Platform
    API level: 7
    Revision: 2
    Skins: HVGA (default), QVGA, WQVGA400, WQVGA432,
WVGA800, WVGA854
id: 6 or "Google Inc.:Google APIs:7"
    Name: Google APIs
    Type: Add-On
    Vendor: Google Inc.
    Revision: 1
    Description: Android + Google APIs
    Based on Android 2.1-update1 (API level 7)
    Libraries:
     * com.google.android.maps (maps.jar)
        API for Google Maps
    Skins: WVGA854, WQVGA400, HVGA (default),
WQVGA432, WVGA800, QVGA
id: 7 or "android-8"
    Name: Android 2.2
    Type: Platform
    API level: 8
    Revision: 2
    Skins: HVGA (default), QVGA, WQVGA400, WQVGA432,
WVGA800, WVGA854
id: 8 or "Google Inc.:Google APIs:8"
    Name: Google APIs
    Type: Add-On
    Vendor: Google Inc.
    Revision: 2
    Description: Android + Google APIs
    Based on Android 2.2 (API level 8)
    Libraries:
     * com.google.android.maps (maps.jar)
        API for Google Maps
    Skins: WVGA854, WQVGA400, HVGA (default),
WQVGA432, WVGA800, QVGA
```

Android Emulator

Starting – Stopping the Emulator

To **start** an instance of the emulator from the command line, change to the ***tools/*** folder of the SDK. Enter emulator command like this:

```
emulator -avd <avd_name>
```

This initializes the emulator and loads an AVD configuration .
After a *few* seconds you will see the emulator window appear on your screen.

If you are working in Eclipse, the ADT plugin for Eclipse installs your application and starts the emulator automatically, when you run or debug the application.

To **stop** an emulator instance, just close the emulator's window.

To list all available AVDs enter DOS command

```
android list avd
```

Android Emulator

AVD - Android Virtual Devices

Android Virtual Devices (AVDs) are configurations of emulator options that let you better model an actual device.

Each AVD is made up of:

- **A hardware profile.** You can set options to define the hardware features of the virtual device. For example, you can define whether the device has a camera, whether it uses a physical QWERTY keyboard or a dialing pad, how much memory it has, and so on.
- **A mapping to a system image**. You can define what version of the Android platform will run on the virtual device. You can choose a version of the standard Android platform or the system image packaged with an SDK add-on.
- **Other options.** You can specify the emulator skin you want to use with the AVD, which lets you control the screen dimensions, appearance, and so on. You can also specify the emulated SD card to use with the AVD.
- **A dedicated storage area** on your development machine, in which is stored the device's user data (installed applications, settings, and so on) and emulated **SD** card.

Android Emulator

AVD - Android Virtual Devices

You can create as many AVDs as you need, based on the types of devices you want to model and the Android platforms and external libraries you want to run your application on.

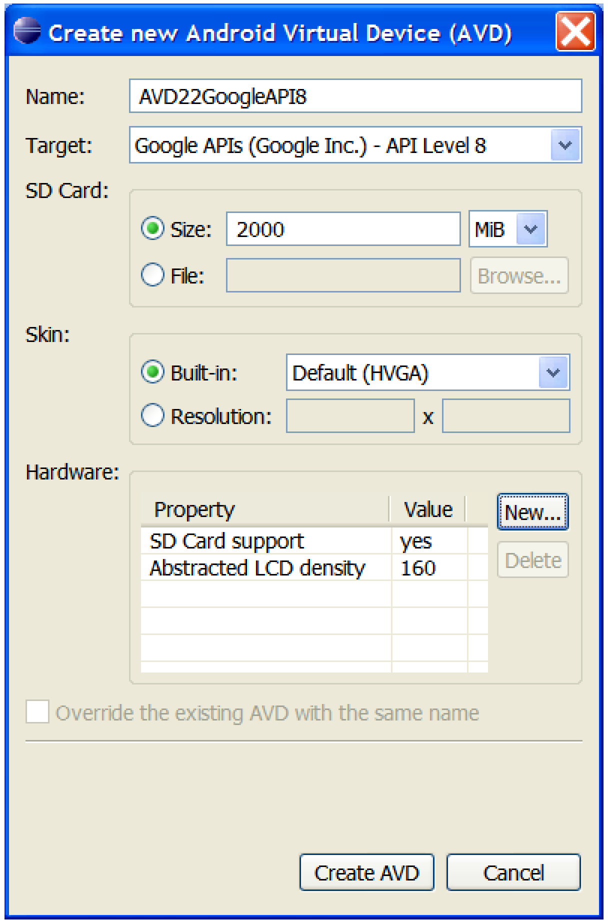
Android Emulator

Creating an AVD using the Eclipse-ADT Tool

From Eclipse, follow the sequence: Main menu (AVD Manager)
> Virtual Devices > New >

Provide a Name,
choose an Android target,
create a new SD card with about 2Gb,
choose a screen type,
add hardware devices…

Click on: Create AVD
(*wait, it takes several minutes to format the new SD card*)

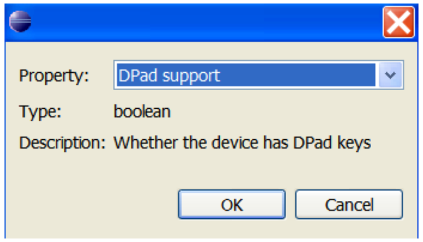

Create new Android Virtual Device (AVD)

Name:	AVD22GoogleAPI8
Target:	Google APIs (Google Inc.) - API Level 8
SD Card:	[✓ Size:] 2000 MiB; File:
Skin:	[✓ Built-in:] Default (HVGA); Resolution: x

Hardware:

Property	Value
SD Card support	yes
Abstracted LCD density	160

New...
Delete
Override the existing AVD with the same name
Create AVD
Cancel



Property:	DPad support
Type:	boolean
Description:	Whether the device has DPad keys

Android Emulator

Creating an AVD using the android tool

When creating an AVD, you simply specify the **-c** option, like this:

```
android create avd -n <avd_name> -t <targetID>  -c <size>[K|M]
```

The **–t** (target) argument sets up a mapping between the AVD and the system image that you want to use whenever the AVD is invoked. Later, when applications use the AVD, they'll be running on the system that you specify in the **-t** argument.

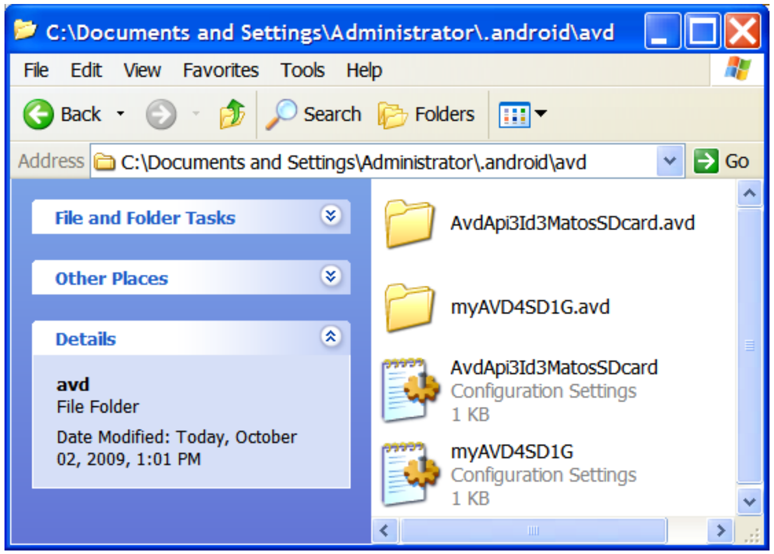
To specify the system image to use, you refer to its *target ID* — an integer — as assigned by the android tool. The target ID is not derived from the system image name, version, or API Level, or other attribute, so you need to have the android tool list the available system images and the target ID of each, as described in the next section. You should do this *before* you run the android create avd command.

Android Emulator

Example: Creating an AVD using the android tool

After listing all targets (see previous image) we have decided to make a profile based on target **id:4** to support SDK1.6 with Google API Mapping libraries. It should also include a 1Gig SD card. We enter the command

```
android create avd -n myAVD4SD1G -t 4  -c 1024M
```

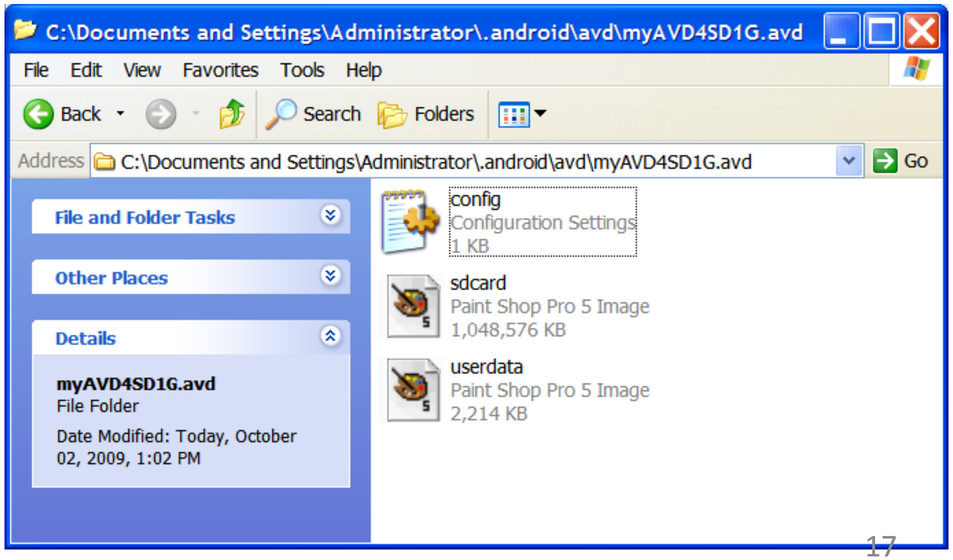


C:\Documents and Settings\Administrator\.android\avd

Address C:\Documents and Settings\Administrator\.android\avd

- AvdApi3Id3MatosSDcard.avd
- myAVD4SD1G.avd
- AvdApi3Id3MatosSDcard — Configuration Settings — 1 KB
- myAVD4SD1G — Configuration Settings — 1 KB

Details: **avd** File Folder. Date Modified: Today, October 02, 2009, 1:01 PM



C:\Documents and Settings\Administrator\.android\avd\myAVD4SD1G.avd

Address C:\Documents and Settings\Administrator\.android\avd\myAVD4SD1G.avd

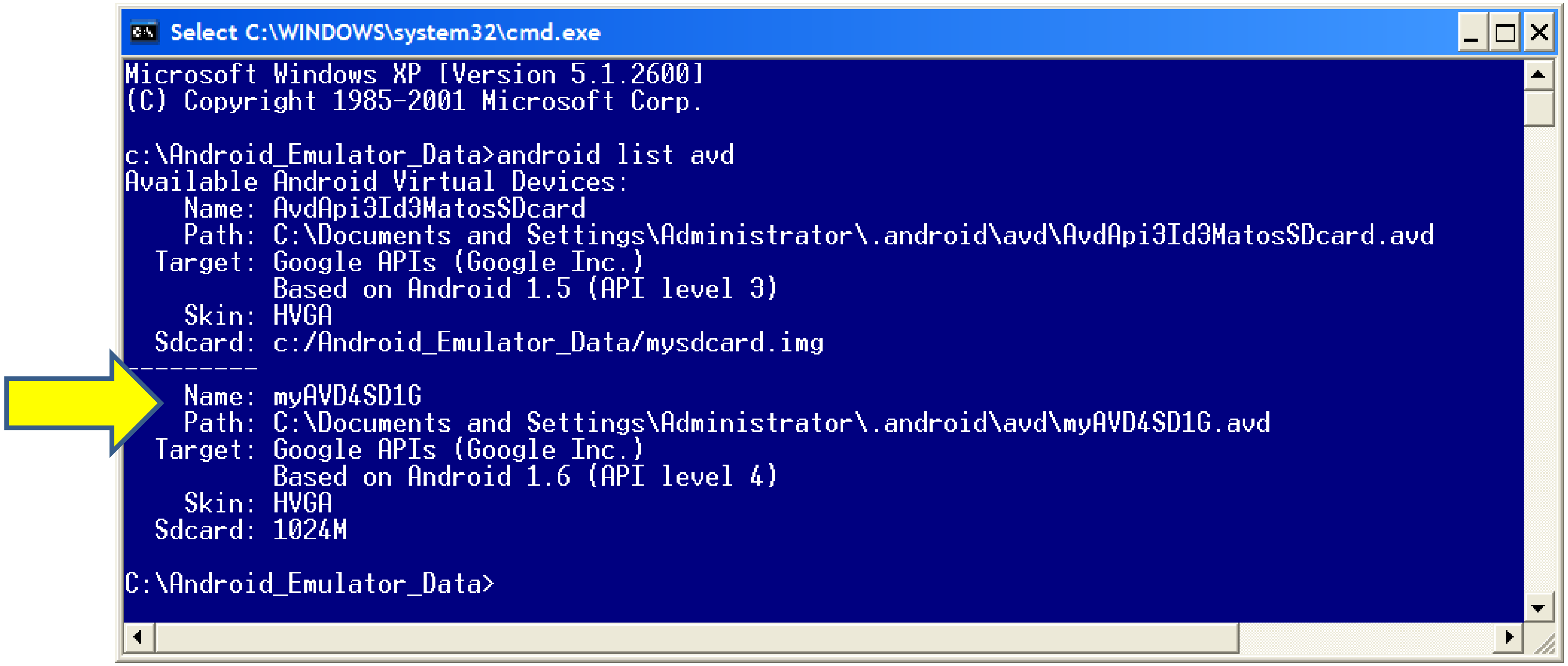
- config — Configuration Settings — 1 KB
- sdcard — Paint Shop Pro 5 Image — 1,048,576 KB
- userdata — Paint Shop Pro 5 Image — 2,214 KB

Details: **myAVD4SD1G.avd** File Folder. Date Modified: Today, October 02, 2009, 1:02 PM

Android Emulator

Example: Creating an AVD using the android tool

Verifying what AVDs are available in the system:


Select C:\WINDOWS\system32\cmd.exe

```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

c:\Android_Emulator_Data>android list avd
Available Android Virtual Devices:
    Name: AvdApi3Id3MatosSDcard
    Path: C:\Documents and Settings\Administrator\.android\avd\AvdApi3Id3MatosSDcard.avd
  Target: Google APIs (Google Inc.)
          Based on Android 1.5 (API level 3)
    Skin: HVGA
  Sdcard: c:/Android_Emulator_Data/mysdcard.img
---------
    Name: myAVD4SD1G
    Path: C:\Documents and Settings\Administrator\.android\avd\myAVD4SD1G.avd
  Target: Google APIs (Google Inc.)
          Based on Android 1.6 (API level 4)
    Skin: HVGA
  Sdcard: 1024M

C:\Android_Emulator_Data>
```

Android Emulator

SD Card Emulation

- You can create a disk image and then load it to the emulator at startup, to simulate the presence of a user's SD card in the device.
- The emulator supports emulated SDHC cards, so you can create an SD card image of any size up to 128 gigabytes.
- You can browse, send files to, and copy/remove files from a simulated SD card either with **adb** or the emulator.

Creating an SD card image using mksdcard

Use the mksdcard tool, included in the SDK, to create a FAT32 disk images.

```
mksdcard <size> <file>
```

For example:

mksdcard 1024M c:/temp/mysdcard.iso

Android Emulator

Android Emulator – How to use the SDCARD device

The general syntax to create an SD card is

```
mksdcard [ -l label ] <size> <file>
```

- The tool *mksdcard* is part of the Android SDK. The SD *label* is optional.
- The device's size is expressed as an integer number followed by either **K** (kilobytes) or **M** (megabytes).

Example: Create a 1GB SDcard device using the following command

```
mksdcard 1024M c:\mysdcard.img
```

Run the emulator with the command

```
emulator -sdcard c:\mysdcard.img
```

or alternatively

```
emulator -avd  myAvdFile
```

Android Emulator

Moving Data, Music and Pictures to the Sdcard

1. Use the program **ddms** to push files into the SDcard (the emulator must be running with the SD card attached to it).

2. Click on: **Device > File Explorer**, this will open a new window and there you will select the SDcard.

3. Now you move data to the sdcard. Your options are

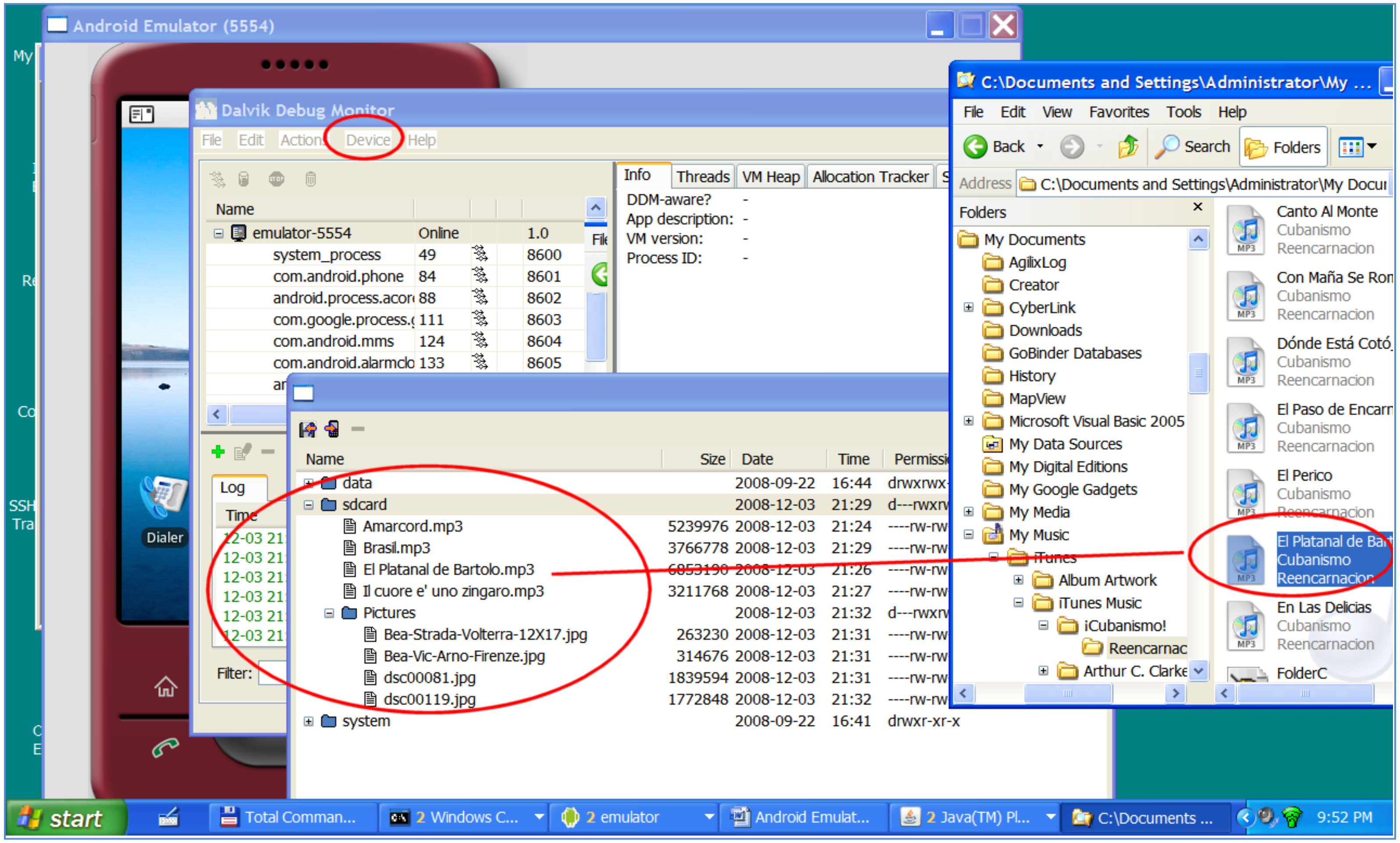
   - Open a Windows **Explore** panel to *drag & drop* files/folders on the card, or

   - Press on the button "*Push File onto Device*"
     (see upper left icons: *push, pull, delete*).

*(**DDMS** stands for Dalvik Debug Monitor Services. The program is located in the /tools folder of the SDK. Also available in Eclipse perspective – Top upper right icons)*

Android Emulator

Moving Data, Music and Pictures to the SDcard


Android Emulator (5554)
Dalvik Debug Monitor
File Edit Actions Device Help
Info Threads VM Heap Allocation Tracker
Dialer
Log
Filter:
C:\Documents and Settings\Administrator\My ...
File Edit View Favorites Tools Help
Back Search Folders
Address C:\Documents and Settings\Administrator\My Docu
start Total Comman... 2 Windows C... 2 emulator Android Emulat... 2 Java(TM) Pl... C:\Documents ... 9:52 PM

Name			
emulator-5554	Online		1.0
system_process	49		8600
com.android.phone	84		8601
android.process.acor	88		8602
com.google.process.(	111		8603
com.android.mms	124		8604
com.android.alarmclo	133		8605

DDM-aware? -
App description: -
VM version: -
Process ID: -

Name	Size	Date	Time	Permissi
data		2008-09-22	16:44	drwxrwx
sdcard		2008-12-03	21:29	d---rwxr
Amarcord.mp3	5239976	2008-12-03	21:24	----rw-rw
Brasil.mp3	3766778	2008-12-03	21:29	----rw-rw
El Platanal de Bartolo.mp3	6853190	2008-12-03	21:26	----rw-rw
Il cuore e' uno zingaro.mp3	3211768	2008-12-03	21:27	----rw-rw
Pictures		2008-12-03	21:32	d---rwxr
Bea-Strada-Volterra-12X17.jpg	263230	2008-12-03	21:31	----rw-rw
Bea-Vic-Arno-Firenze.jpg	314676	2008-12-03	21:31	----rw-rw
dsc00081.jpg	1839594	2008-12-03	21:31	----rw-rw
dsc00119.jpg	1772848	2008-12-03	21:32	----rw-rw
system		2008-09-22	16:41	drwxr-xr-x

Folders

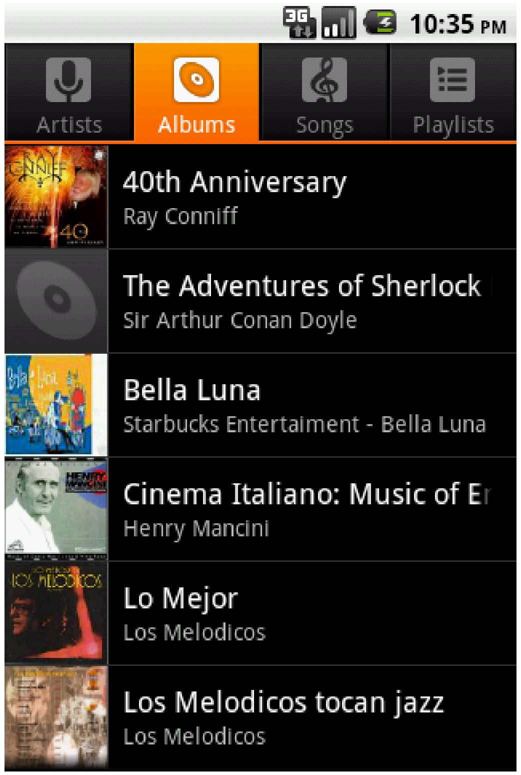
- My Documents
  - AgilixLog
  - Creator
  - CyberLink
  - Downloads
  - GoBinder Databases
  - History
  - MapView
  - Microsoft Visual Basic 2005
  - My Data Sources
  - My Digital Editions
  - My Google Gadgets
  - My Media
  - My Music
    - iTunes
      - Album Artwork
      - iTunes Music
        - iCubanismo!
          - Reencarnac
        - Arthur C. Clarke

Canto Al Monte — Cubanismo — Reencarnacion
Con Maña Se Ron — Cubanismo — Reencarnacion
Dónde Está Cotó — Cubanismo — Reencarnacion
El Paso de Encarr — Cubanismo — Reencarnacion
El Perico — Cubanismo — Reencarnacion
El Platanal de Ba — Cubanismo — Reencarnacion
En Las Delicias — Cubanismo — Reencarnacion
FolderC

Android Emulator

Moving Data, Music and Pictures to the SDcard

4. Return to the emulator. This time you will see the selected (music) files in the SDcard



10:35 PM

40th Anniversary
Ray Conniff

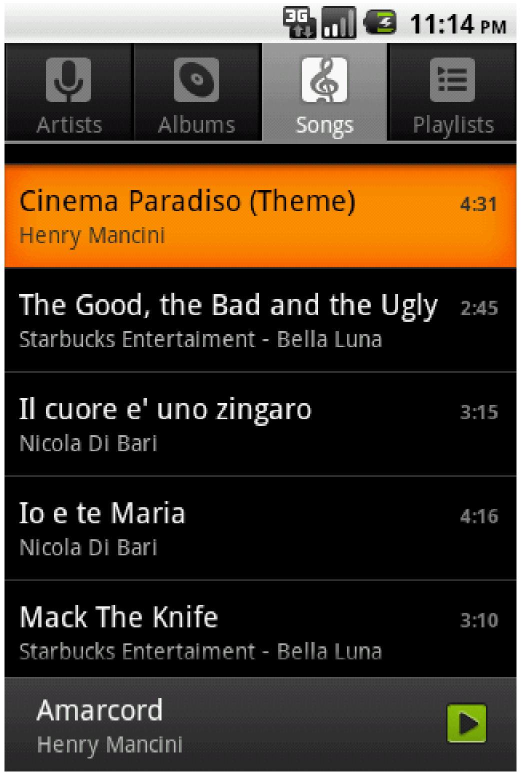
The Adventures of Sherlock
Sir Arthur Conan Doyle

Bella Luna
Starbucks Entertaiment - Bella Luna

Cinema Italiano: Music of E
Henry Mancini

Lo Mejor
Los Melodicos

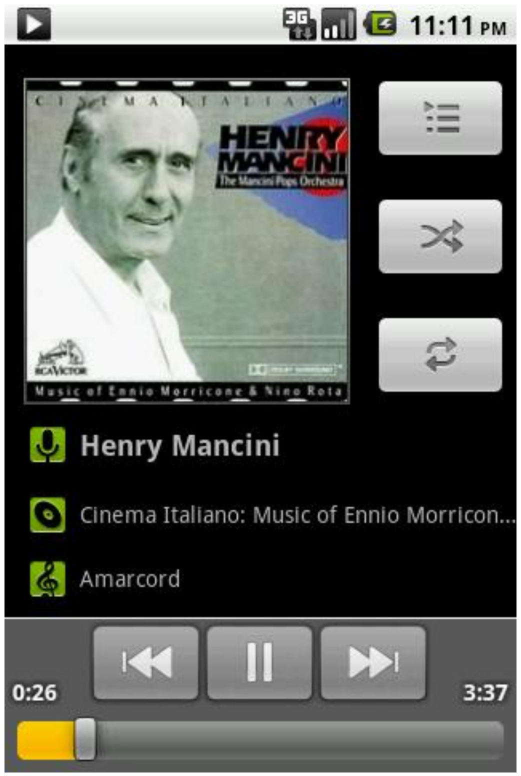
Los Melodicos tocan jazz
Los Melodicos



11:14 PM

Cinema Paradiso (Theme) Henry Mancini	4:31
The Good, the Bad and the Ugly Starbucks Entertaiment - Bella Luna	2:45
Il cuore e' uno zingaro Nicola Di Bari	3:15
Io e te Maria Nicola Di Bari	4:16
Mack The Knife Starbucks Entertaiment - Bella Luna	3:10

Amarcord
Henry Mancini



11:11 PM

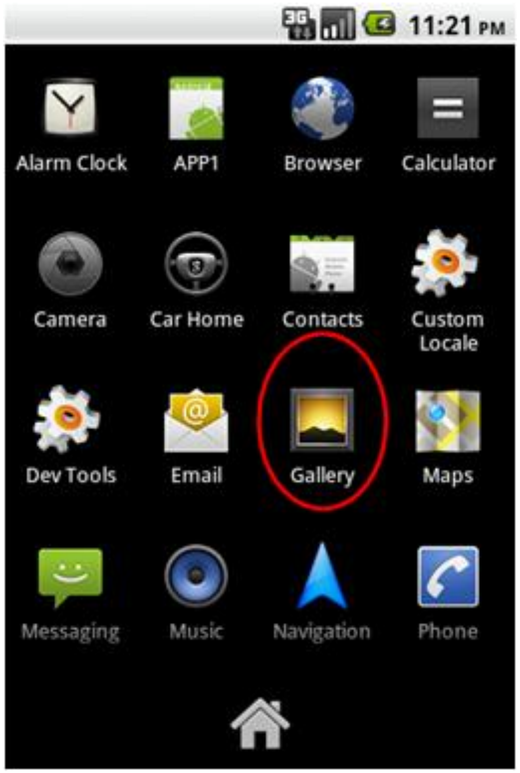
Henry Mancini
Cinema Italiano: Music of Ennio Morricon...
Amarcord

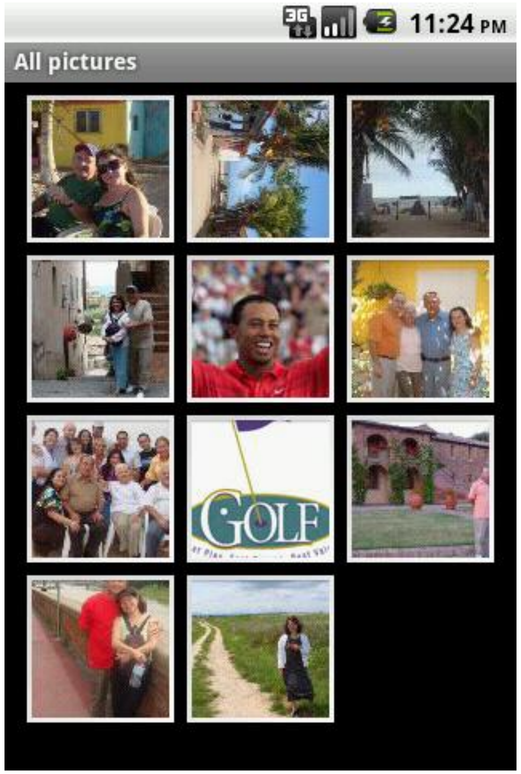
0:26 3:37

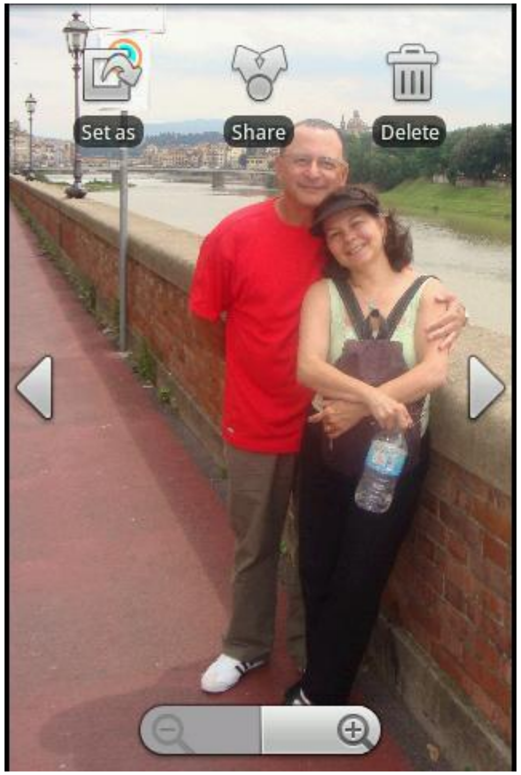
Android Emulator

Moving Data, Music and Pictures to the SDcard

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







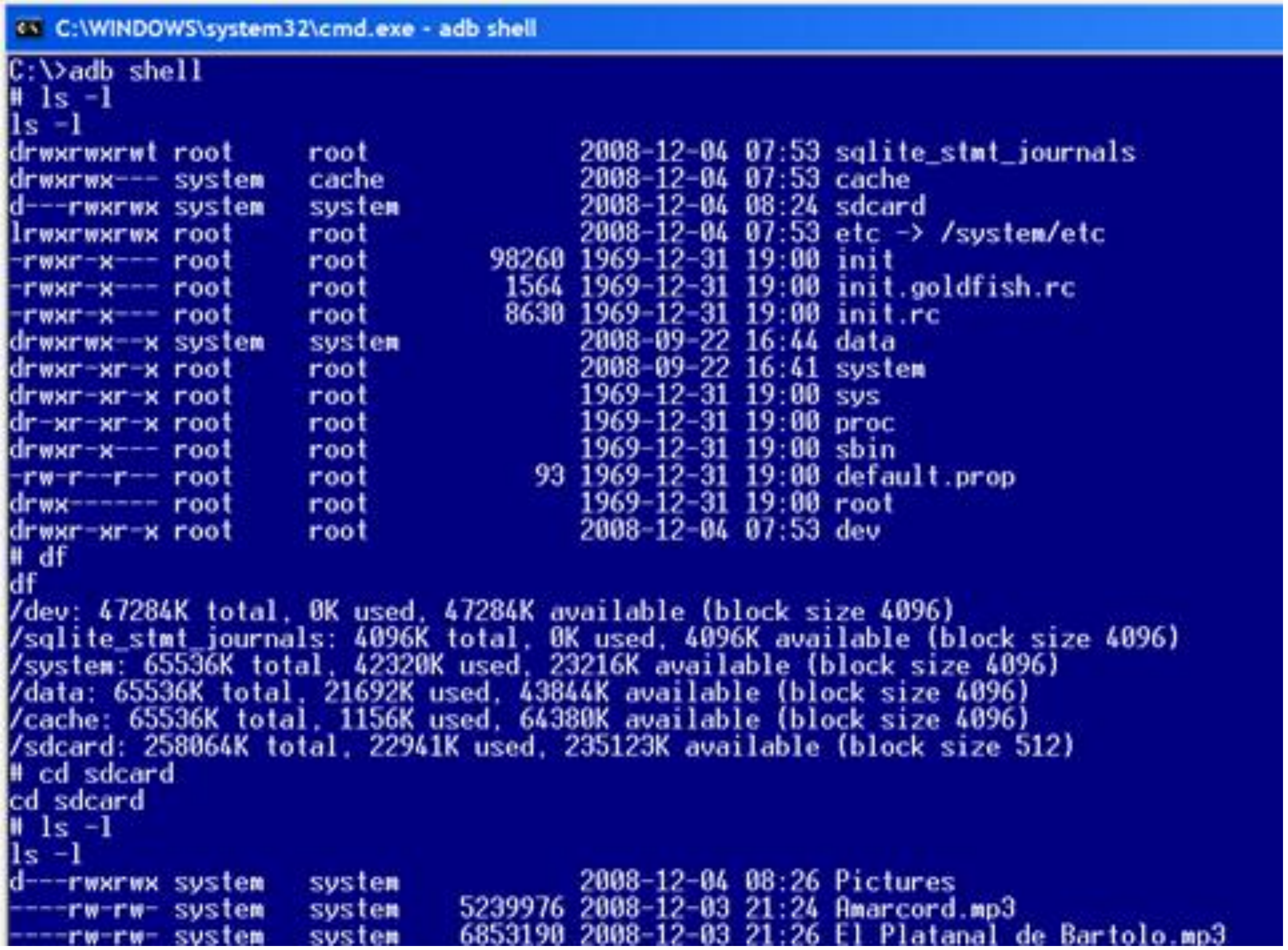
Android Emulator

Android – Login into the OS shell

You can log into the OS Linux version of Android executing in the emulator and issue selected commands.

1. Run the Android emulator
2. Run **adb** application as follows:
   c:> adb shell

(adb is the Android Debug Bridge app. It is Located in the /tools folder of the SDK)


C:\WINDOWS\system32\cmd.exe - adb shell

```
C:\>adb shell
# ls -l
ls -l
drwxrwxrwt root     root              2008-12-04 07:53 sqlite_stmt_journals
drwxrwx--- system   cache             2008-12-04 07:53 cache
d---rwxrwx system   system            2008-12-04 08:24 sdcard
lrwxrwxrwx root     root              2008-12-04 07:53 etc -> /system/etc
-rwxr-x--- root     root        98260 1969-12-31 19:00 init
-rwxr-x--- root     root         1564 1969-12-31 19:00 init.goldfish.rc
-rwxr-x--- root     root         8630 1969-12-31 19:00 init.rc
drwxrwx--x system   system            2008-09-22 16:44 data
drwxr-xr-x root     root              2008-09-22 16:41 system
drwxr-xr-x root     root              1969-12-31 19:00 sys
dr-xr-xr-x root     root              1969-12-31 19:00 proc
drwxr-x--- root     root              1969-12-31 19:00 sbin
-rw-r--r-- root     root           93 1969-12-31 19:00 default.prop
drwx------ root     root              1969-12-31 19:00 root
drwxr-xr-x root     root              2008-12-04 07:53 dev
# df
df
/dev: 47284K total, 0K used, 47284K available (block size 4096)
/sqlite_stmt_journals: 4096K total, 0K used, 4096K available (block size 4096)
/system: 65536K total, 42320K used, 23216K available (block size 4096)
/data: 65536K total, 21692K used, 43844K available (block size 4096)
/cache: 65536K total, 1156K used, 64380K available (block size 4096)
/sdcard: 258064K total, 22941K used, 235123K available (block size 512)
# cd sdcard
cd sdcard
# ls -l
ls -l
d---rwxrwx system   system            2008-12-04 08:26 Pictures
----rw-rw- system   system    5239976 2008-12-03 21:24 Amarcord.mp3
----rw-rw- system   system    6853190 2008-12-03 21:26 El Platanal de Bartolo.mp3
```

Android Emulator

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.

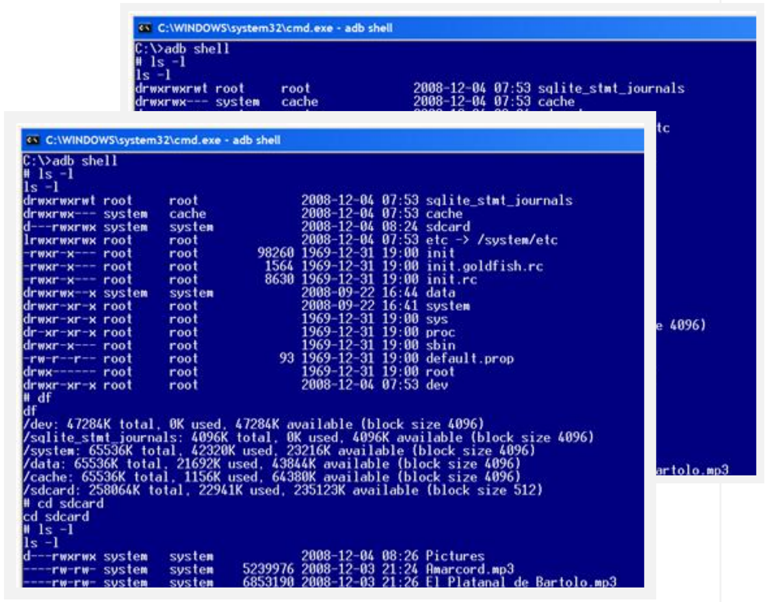
Follow the steps:

1. Get a list of all active emulators

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

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

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



```
C:\WINDOWS\system32\cmd.exe - adb shell
C:\>adb shell
# ls -l
ls -l
drwxrwxrwt root     root              2008-12-04 07:53 sqlite_stmt_journals
drwxrwx--- system   cache             2008-12-04 07:53 cache
d---rwxrwx system   system            2008-12-04 08:24 sdcard
lrwxrwxrwx root     root              2008-12-04 07:53 etc -> /system/etc
-rwxr-x--- root     root        98260 1969-12-31 19:00 init
-rwxr-x--- root     root         1564 1969-12-31 19:00 init.goldfish.rc
-rwxr-x--- root     root         8630 1969-12-31 19:00 init.rc
drwxrwx--x system   system            2008-09-22 16:44 data
drwxr-xr-x root     root              2008-09-22 16:41 system
drwxr-xr-x root     root              1969-12-31 19:00 sys
dr-xr-xr-x root     root              1969-12-31 19:00 proc
drwxr-x--- root     root              1969-12-31 19:00 sbin
-rw-r--r-- root     root           93 1969-12-31 19:00 default.prop
drwx------ root     root              1969-12-31 19:00 root
drwxr-xr-x root     root              2008-12-04 07:53 dev
# df
df
/dev: 47284K total, 0K used, 47284K available (block size 4096)
/sqlite_stmt_journals: 4096K total, 0K used, 4096K available (block size 4096)
/system: 65536K total, 42320K used, 23216K available (block size 4096)
/data: 65536K total, 21692K used, 43844K available (block size 4096)
/cache: 65536K total, 1156K used, 64380K available (block size 4096)
/sdcard: 258064K total, 22941K used, 235123K available (block size 512)
# cd sdcard
cd sdcard
# ls -l
ls -l
d---rwxrwx system   system            2008-12-04 08:26 Pictures
----rw-rw- system   system    5239976 2008-12-03 21:24 Amarcord.mp3
----rw-rw- system   system    6853190 2008-12-03 21:26 El Platanal de Bartolo.mp3
```

(**adb** is the Android Debug Bridge app. It is Located in the /tools folder of the SDK)

Android Emulator

NOTE1: Emulators & Hardware Devices

You may test your applications in either a *software emulator* or a *hardware device*.

All you need to do is connect your phone to the computer via USB cable.

On a command shell type the command: "**adb devices**" you should see something like "**HT845GZ45737 device**" indicating the presence of your hardware device.

Gaining Root Access to Your Hardware device

A developer's phone such as the G1 comes with root access enabled and is fully opened.

Run the terminal application (**adb shell**) and see if you have the **#** prompt; if not try the command **su**. It should give you the root prompt, if you have a *permission denied* error then *you do not have root access*.

Android Emulator

NOTE2: Moving an app from (Rooted) Hardware to Emulator

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

1. Run command shell: > **adb devices** (find out the id of your hardware, say **HT845GZ45737**)

2. Pull the file from the device to your computer's file system. Enter the command **adb -s HT845GZ45737 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**

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

Android Emulator

Android – Login into the OS shell

3. 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
```

Android Emulator

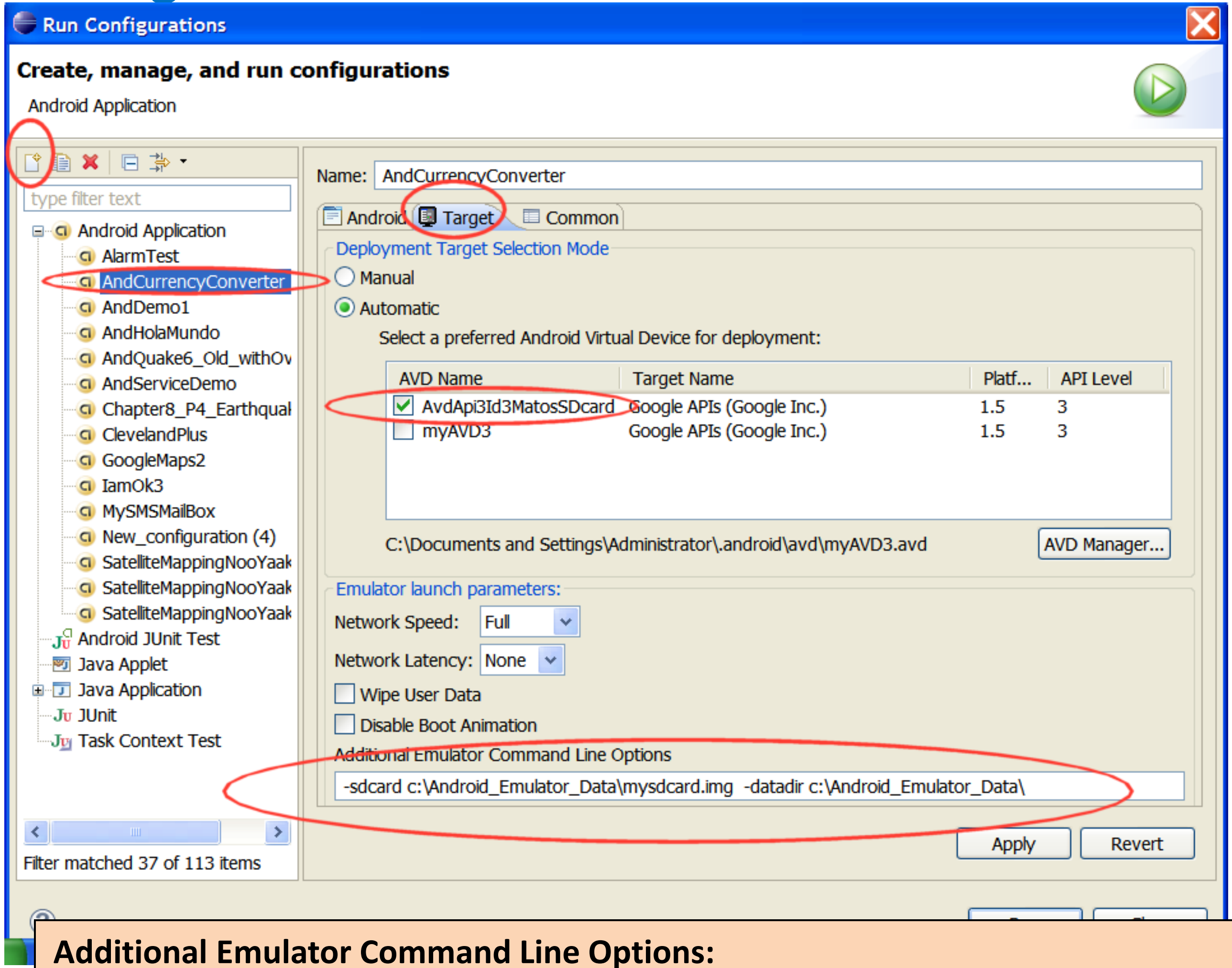
Android – Login into the OS shell

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

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

Android Emulator

Using the Emulator with “inserted” SD card from Eclipse



Run Configurations

Create, manage, and run configurations

Android Application

Name: AndCurrencyConverter

Android | Target | Common

Deployment Target Selection Mode

Manual [✓ Automatic]

Select a preferred Android Virtual Device for deployment:

AVD Name	Target Name	Platf...	API Level
✓ AvdApi3Id3MatosSDcard	Google APIs (Google Inc.)	1.5	3
myAVD3	Google APIs (Google Inc.)	1.5	3

C:\Documents and Settings\Administrator\.android\avd\myAVD3.avd

Emulator launch parameters:

Network Speed: Full

Network Latency: None

Wipe User Data

Disable Boot Animation

Additional Emulator Command Line Options

```
-sdcard c:\Android_Emulator_Data\mysdcard.img  -datadir c:\Android_Emulator_Data\
```

Android Application: AlarmTest, AndCurrencyConverter, AndDemo1, AndHolaMundo, AndQuake6_Old_withOv, AndServiceDemo, Chapter8_P4_Earthqual, ClevelandPlus, GoogleMaps2, IamOk3, MySMSMailBox, New_configuration (4), SatelliteMappingNooYaak, SatelliteMappingNooYaak, SatelliteMappingNooYaak

Android JUnit Test, Java Applet, Java Application, JUnit, Task Context Test

Filter matched 37 of 113 items

From Eclipse’s menu create new launch configuration:

Run >
Run Configurations >
New icon

On the **Target** panel

1. Select existing Android Virtual device (AVD)
2. Enter additional Command Line Options (see caption)
3. Apply > Run

Additional Emulator Command Line Options:
-sdcard c:\Android_Emulator_Data\mysdcard.img -datadir c:\Android_Emulator_Data

Android Emulator

Sending Text Messages to the Emulator

1. Start the emulator.
2. Open a new shell and type :

   c:> adb devices

   so you know the emulator's numeric port id (usually **5554**, **5556**, and so on)

3. Connect to the console using telnet command like:

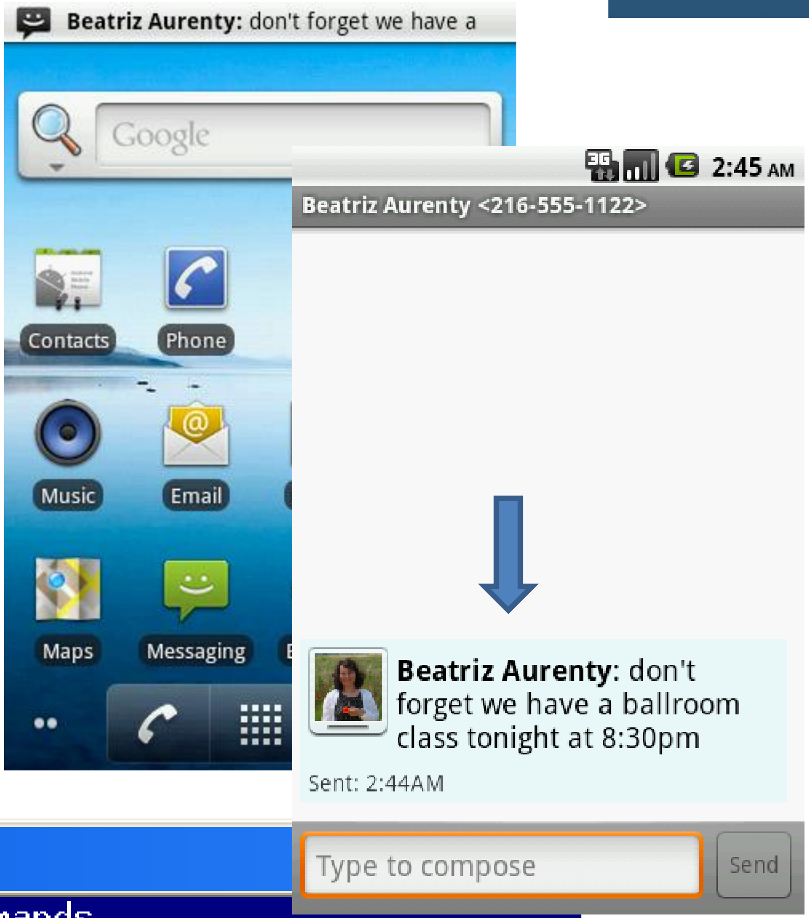
   c:> telnet localhost 5554

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>

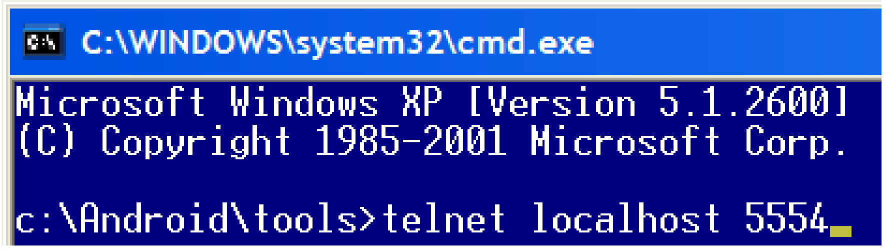
Android Emulator

Example:
Sending Text Messages
to the Emulator


Beatriz Aurenty: don't forget we have a
Google
Contacts
Phone
Music
Email
Maps
Messaging
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

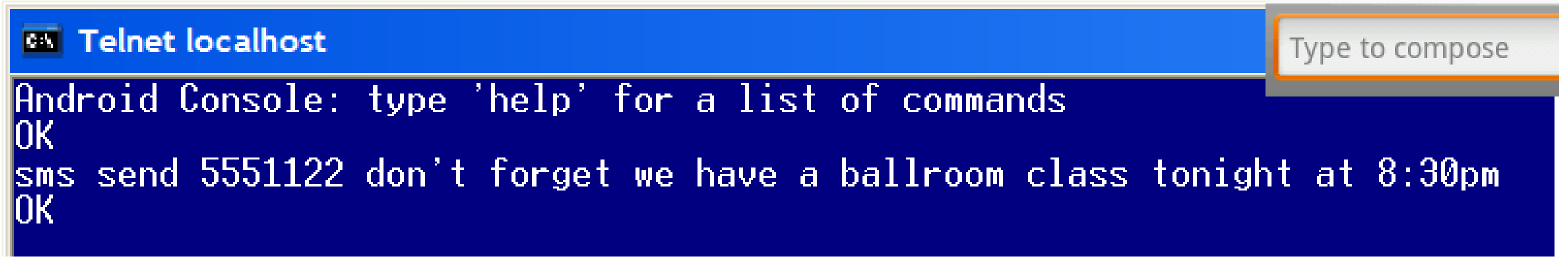
Sent: 2:44AM



C:\WINDOWS\system32\cmd.exe

```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

c:\Android\tools>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

Making a Voice Call to the Emulator

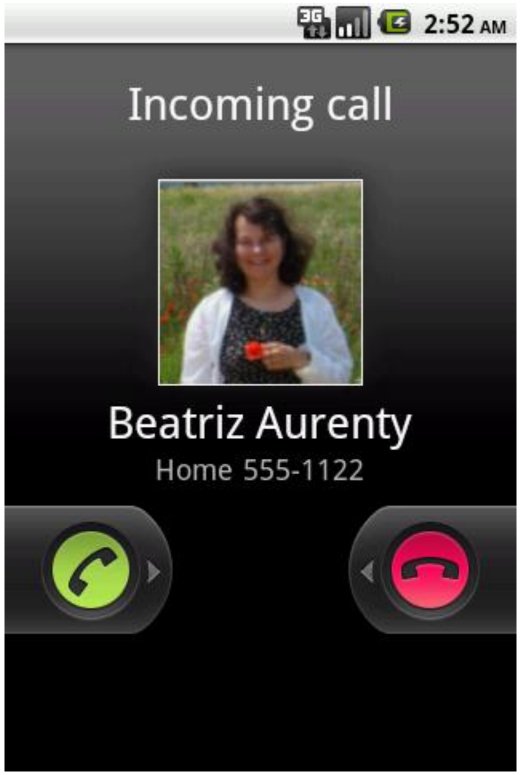
1. Start the emulator.
2. Open a new shell and type :
   adb devices
   to know the emulator's numeric port id (usually **5554**, **5556**, and so on)

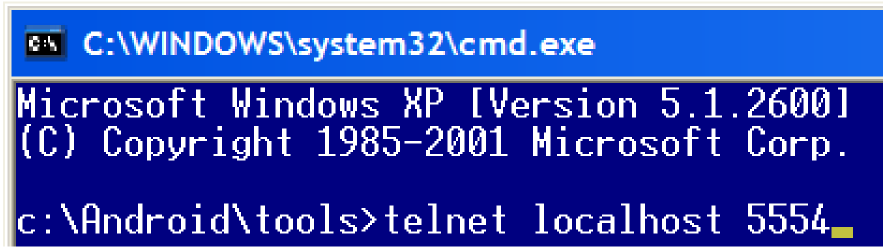
3. Connect to the console using telnet command like:
   telnet localhost 5554 (this is the 'number' to be called)

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

Android Emulator

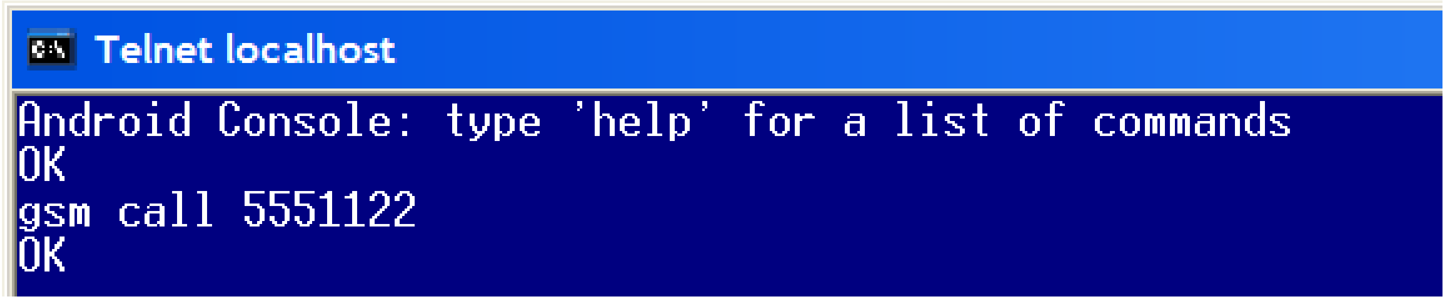
Example: Making a Phone Call to the Emulator




C:\WINDOWS\system32\cmd.exe

```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

c:\Android\tools>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

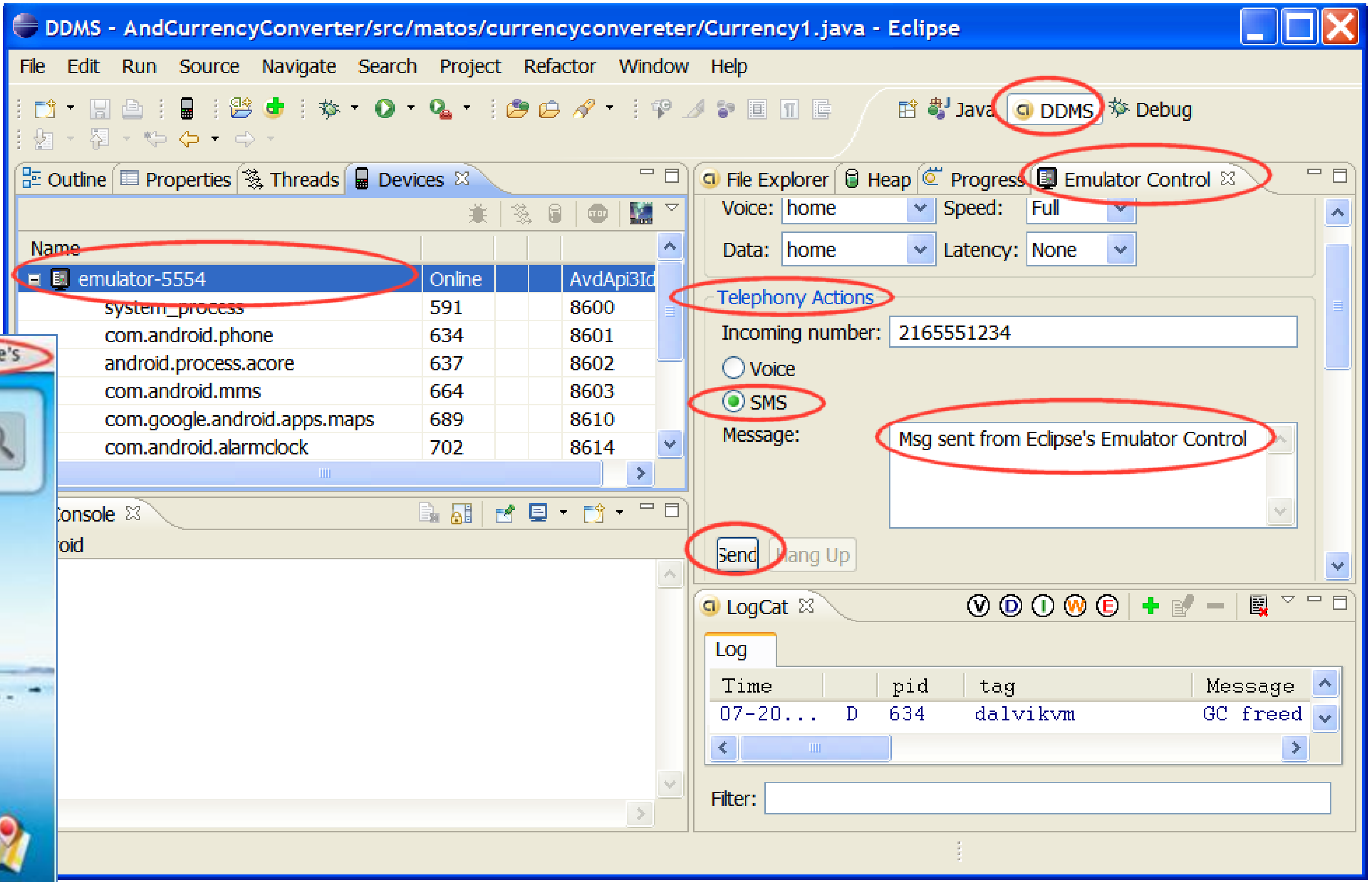
Emulator Control

With these controls, you can simulate special device states and activities. Features include:

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. To use the Location Controls, launch your application in the Android emulator and open DDMS. Click the Emulator Controls tab and scroll down to Location Controls. From here, you can:
   - 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.

Android Emulator

Using Eclipse to test Emulator's Telephony Actions


DDMS - AndCurrencyConverter/src/matos/currencyconvereter/Currency1.java - Eclipse

Name			
emulator-5554	Online		AvdApi3Id
system_process	591		8600
com.android.phone	634		8601
android.process.acore	637		8602
com.android.mms	664		8603
com.google.android.apps.maps	689		8610
com.android.alarmclock	702		8614

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

Telephony Actions

Incoming number: 2165551234

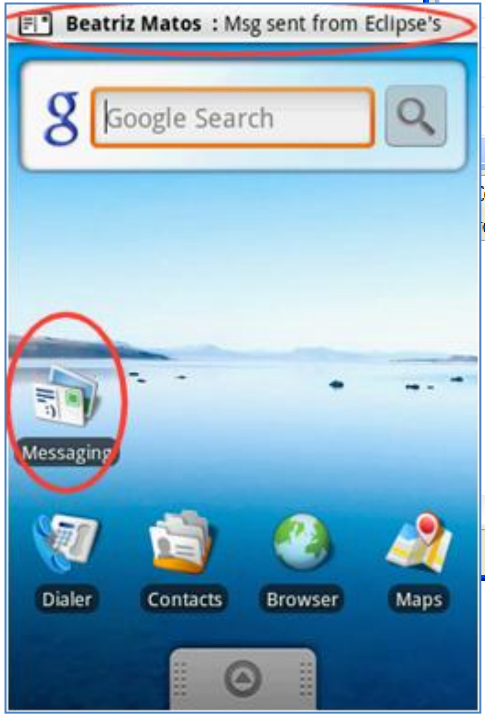
( ) Voice
(●) SMS

Message: Msg sent from Eclipse's Emulator Control

Send Hang Up

Time		pid	tag	Message
07-20...	D	634	dalvikvm	GC freed

Filter:


Beatriz Matos : Msg sent from Eclipse's

Android Emulator

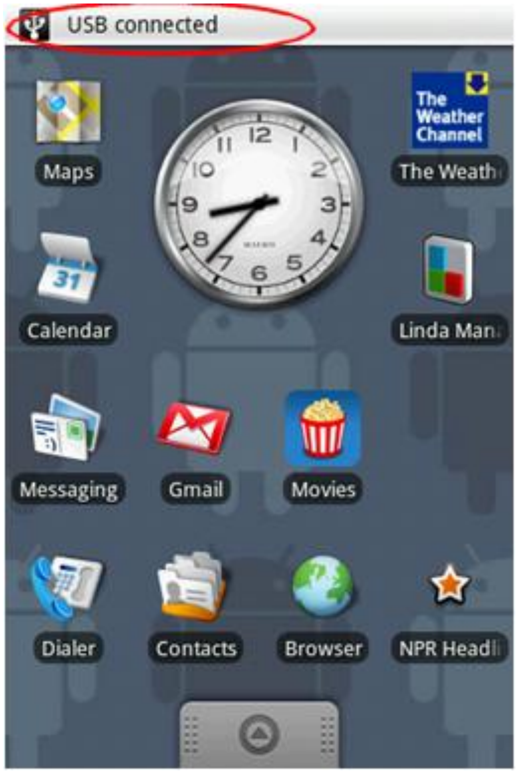
Questions ?

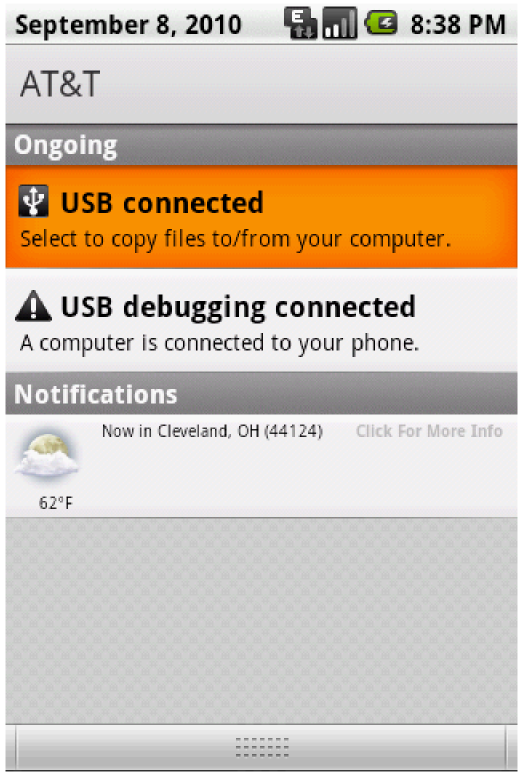
Android Emulator

Appendix 1 – Connecting your Hardware Device to the Computer

1. Use a mini-USB cable to link the device and your computer
2. Expand the Notification bar
3. Mount the device

You could now use the Eclipse-ADT-File Explorer panel to pull/push files to the device.




September 8, 2010 8:38 PM

AT&T

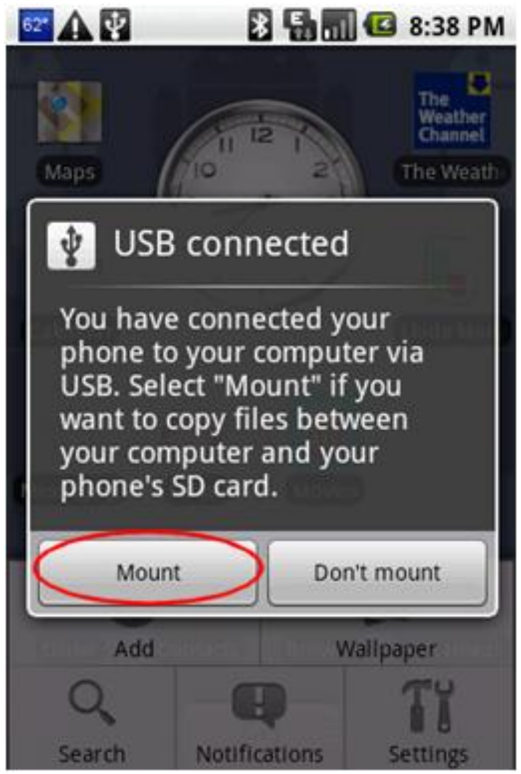
Ongoing

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

USB debugging connected
A computer is connected to your phone.

Notifications

Now in Cleveland, OH (44124) Click For More Info
62°F


8:38 PM

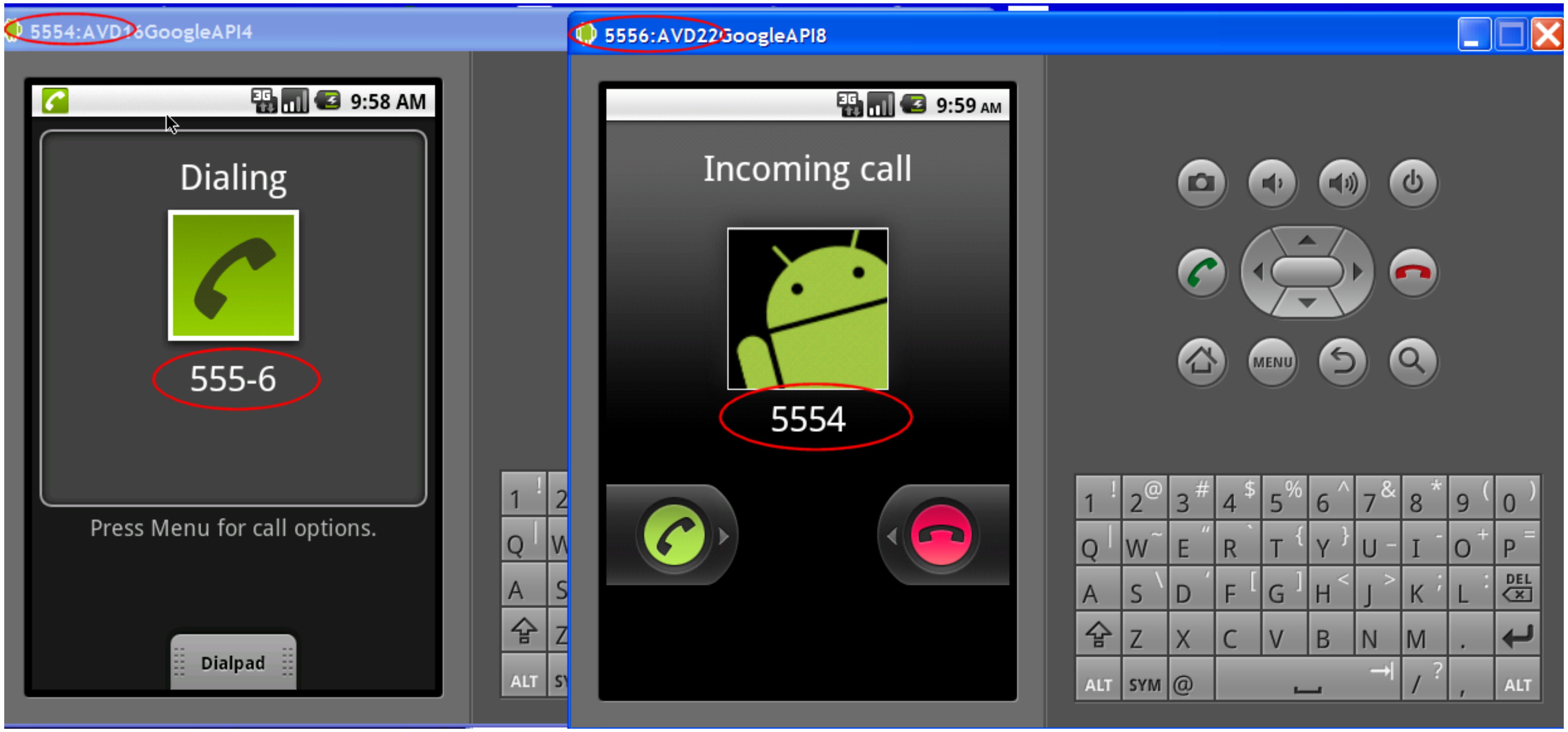
USB connected

You have connected your phone to your computer via USB. Select "Mount" if you want to copy files between your computer and your phone's SD card.

Android Emulator

Appendix 1 – Emulator to Emulator Communication

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)



5554:AVD16GoogleAPI4
9:58 AM
Dialing
555-6
Press Menu for call options.
Dialpad
5556:AVD22GoogleAPI8
9:59 AM
Incoming call
5554
MENU