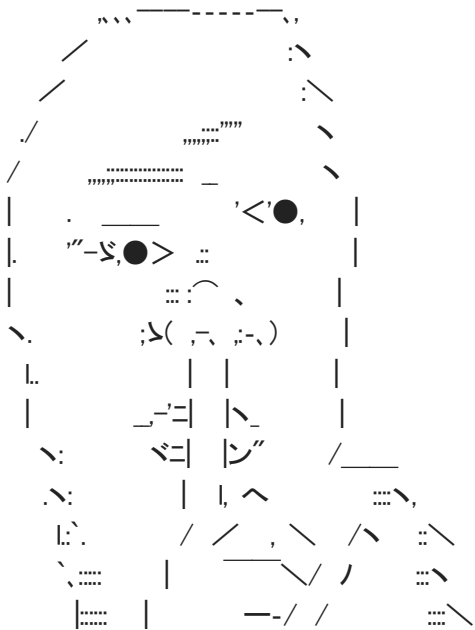


Title : HELLO ANDROID



Author : Ryo Terunuma

Table of Contents

1. PREFACE	1
1.1. INTRO	1
1.2. ENVIRONMENT	1
1.3. SOURCES	3
2. SETUP	5
2.1. SDK	5
2.2. DEVICE & DRIVER	6
2.3. EMULATOR	9
2.4. DEBUGGER	10
2.5. DEVICE ARRANGEMENT	11
3. DEVELOPMENT	12
3.1. CYCLE	12
3.2. MARKET	12
4. APPLICATION	13
4.1. HELLO WORLD	13
4.2. CREATE APPLICATION INFORMATION PROJECT	14
4.3. UPDATE BUILD.XML	14
4.4. CREATE PRIVATE KEY	14
4.5. EDIT SOURCE AND MANIFEST	15
4.6. INSTALL APK AND RUN APP	16
5. ACTIVITY	17
5.1. LIFECYCLE	17
5.2. START ACTIVITY	18
5.3. RESOURCE	18
6. CONTENT PROVIDER & SQLITE	19
6.1. OVERVIEW	19
6.2. EDIT SOURCE	19
6.3. SHOW DATABASE	19
7. WIDGET & SERVICE	20
7.1. OVERVIEW	20
7.2. EDIT SOURCE & RESOURCE	20
7.3. RUN	21
8. UNITTEST	22
8.1. CREATE TEST PROJECT	22
8.2. EDIT TESTCASE	22
8.3. RUN	22
APPENDIX	23
1. APPLICATIONS RUN ON MOTOROLA	23

1. Preface

1.1. Intro

This document focus to simple android development things .
So this document do not include sale , maintenance and others things.

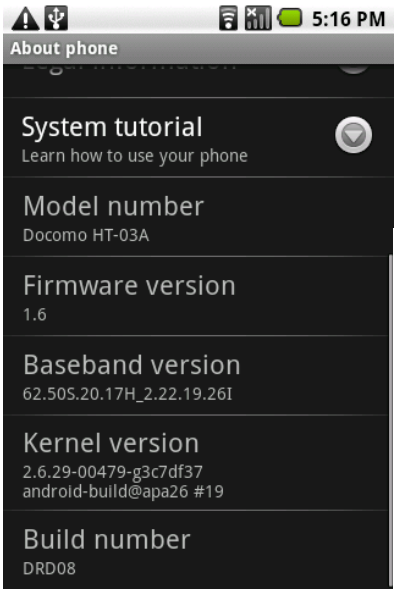
1.2. Environment

This document wrote content of following development & confirmation environment.

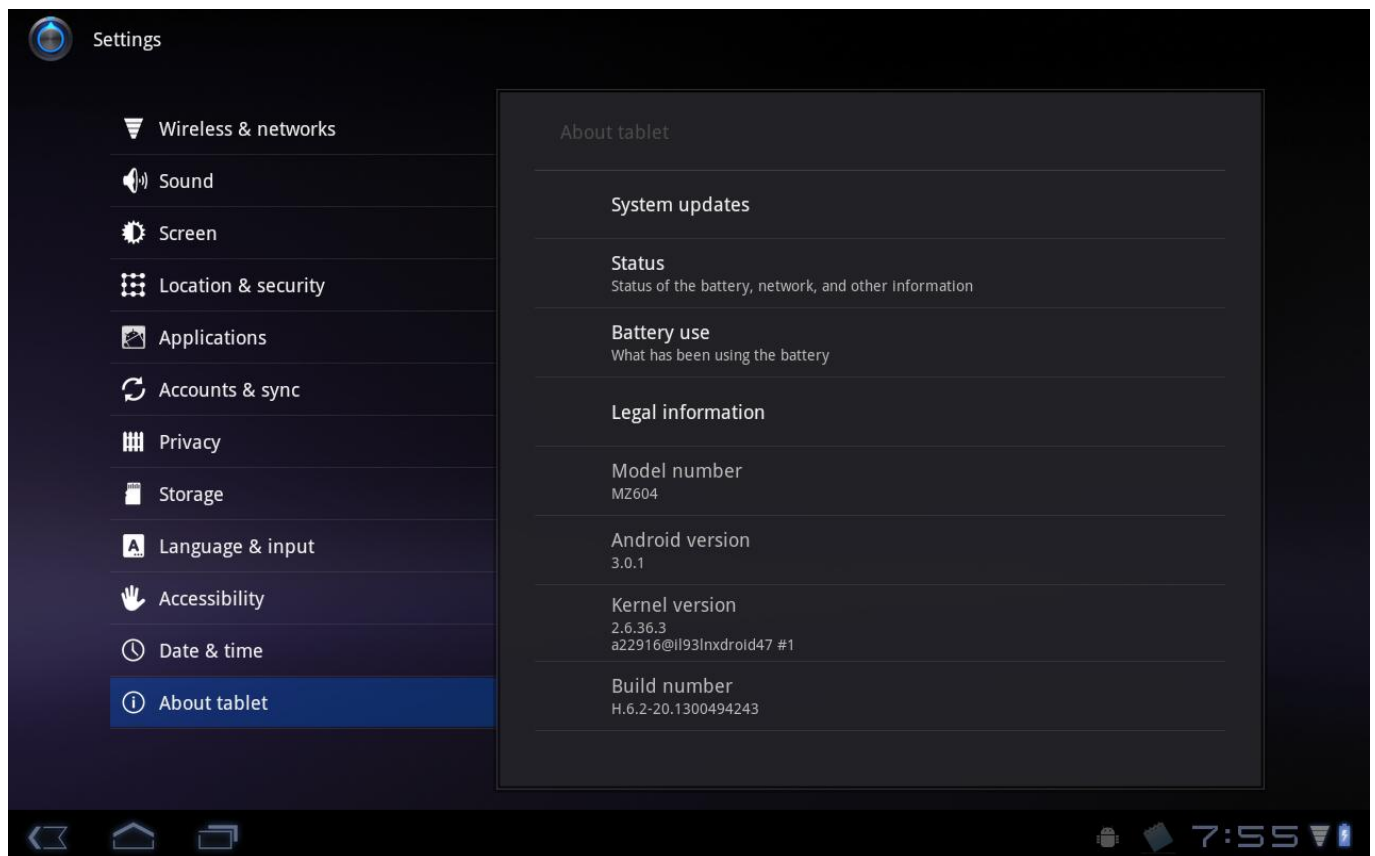
Platform

Platform	Development	Confirmation	JDK
Ubuntu 11.04	✓	✓	1.6.0_24
Windows 7	✗	✓	1.6.0_22

Device



HT 03

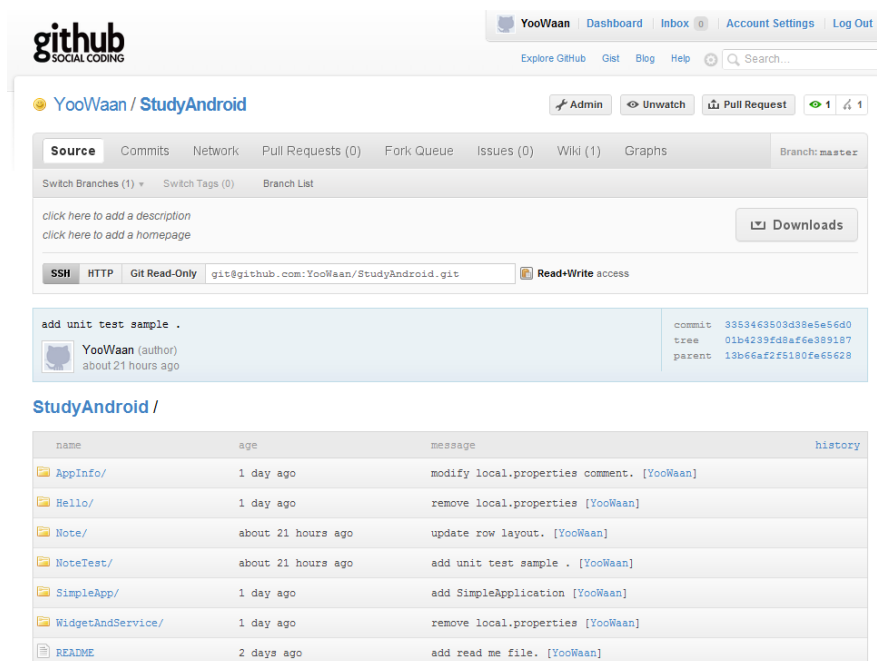


MOTOROLA XOOM

1.3. Sources

Sources is stored on git-hub . If you need source check below .

URL : <https://github.com/YooWaan/StudyAndroid>



Summary of Directory

name	description
------	-------------

AppInfo/	Application Information Sample
----------	--------------------------------

Hello/	Hello World Sample
--------	--------------------

Note/	SQLite ContentProvider Sample
-------	-------------------------------

NoteTest/	Note app UnitTest Sample
-----------	--------------------------

SimpleApp/	Simple Application Sample
------------	---------------------------

WidgetAndService/	Widget and Service Sample
-------------------	---------------------------

Guid.pdf	Guide Document
----------	----------------

local.properties	building local properties
------------------	---------------------------

tool.xml	utility ant tasks xml
----------	-----------------------

About tool.xml

ant Target Overview

- clean : ant -f build.xml clean
- compile : ant -f build.xml releae
- apk-copy : copy from bin/your.apk to dist/specified apk file.
- keystore : generate new keystore
- sign : sign to apk file.

name	description
clean	execute clean of build.xml on android project
compile	execute compile of build.xml on android project
apk-copy	copy apk to distribution dir
keystore	generate private key
sign	sign to apf file
test	execute unittest

For execute ant by tool.xml , Please prepare tool.properties file

tool.properties

name	description
app.apk	App apk filename
unsigned.apk	File name of generated by "ant release" command
sign.alias	private key alias
sign.keystore	keystore file name
sign.storepass	keystore password
dname.xxx	private key for c, st, l, o, ou, cn names
test.package	test package name

2. Setup

This chapter is setup of android development environment.
Each sections are wrote simple operation steps .

2.1. SDK

1.setup system . Install JDK and Others .

Note

Ubuntu x64 need preparing .

URL : <http://developer.android.com/sdk/requirements.html>

Install for ubuntu

Terminal

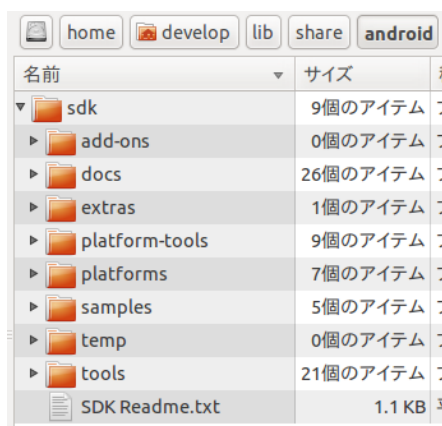
```
$ sudo apt-get install ia-32libs
```

```
$ sudo apt-get install sun-java6-jdk
```

2.download sdk .

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

3.deploy (decompress) sdk archive file .



4.execute android and intall sdk .

URL : <http://developer.android.com/sdk/installing.html>

5.set path to android tools platform-tools

.bashrc

```
export JAVA_HOME=/usr/lib/jvm/java-6-sun
```

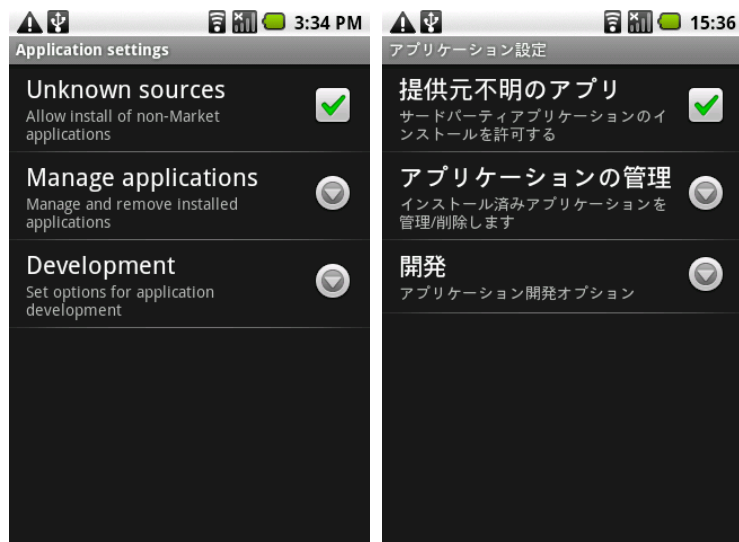
```
export ANDROID_HOME=~/.lib/share/android/sdk
```

```
export PATH=${JAVA_HOME}/bin:${PATH}:${ANDROID_HOME}/tools:${ANDROID_HOME}/platform-tools
```

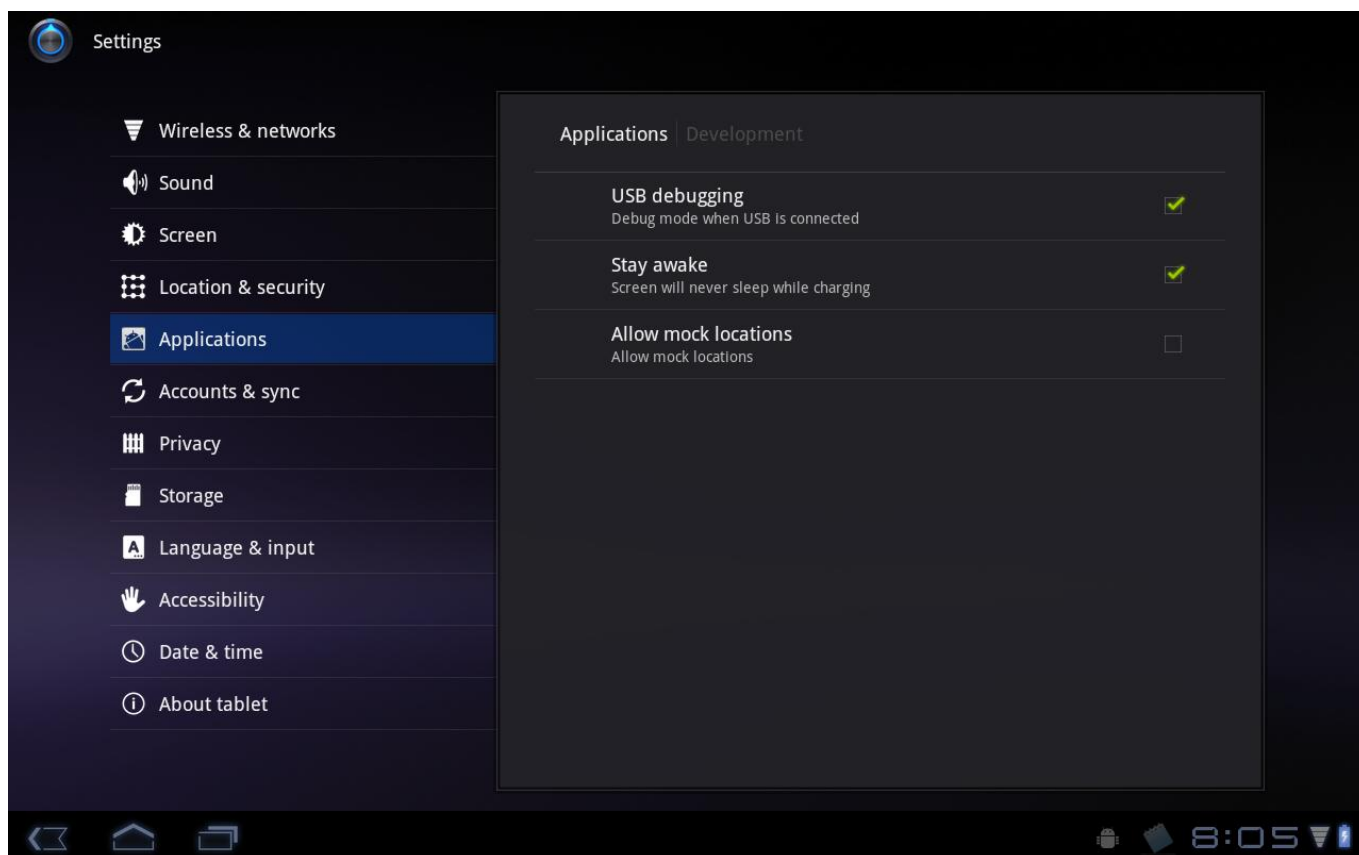
2.2. Device & Driver

1. Setting your device

Check device's "Unknown sources" .



Check device's "USB debugging"



2. setup usb driver

see <http://developer.android.com/intl/ja/guide/developing/device.html>

Ubuntu: write /etc/udev/rules.d/51-android.rules

```
/etc/udev/rules.d/51-android.
```

```
# HTC
```

```
SUBSYSTEM=="usb", SYSFS{idVendor}="0bb4", MODE="0666"
```

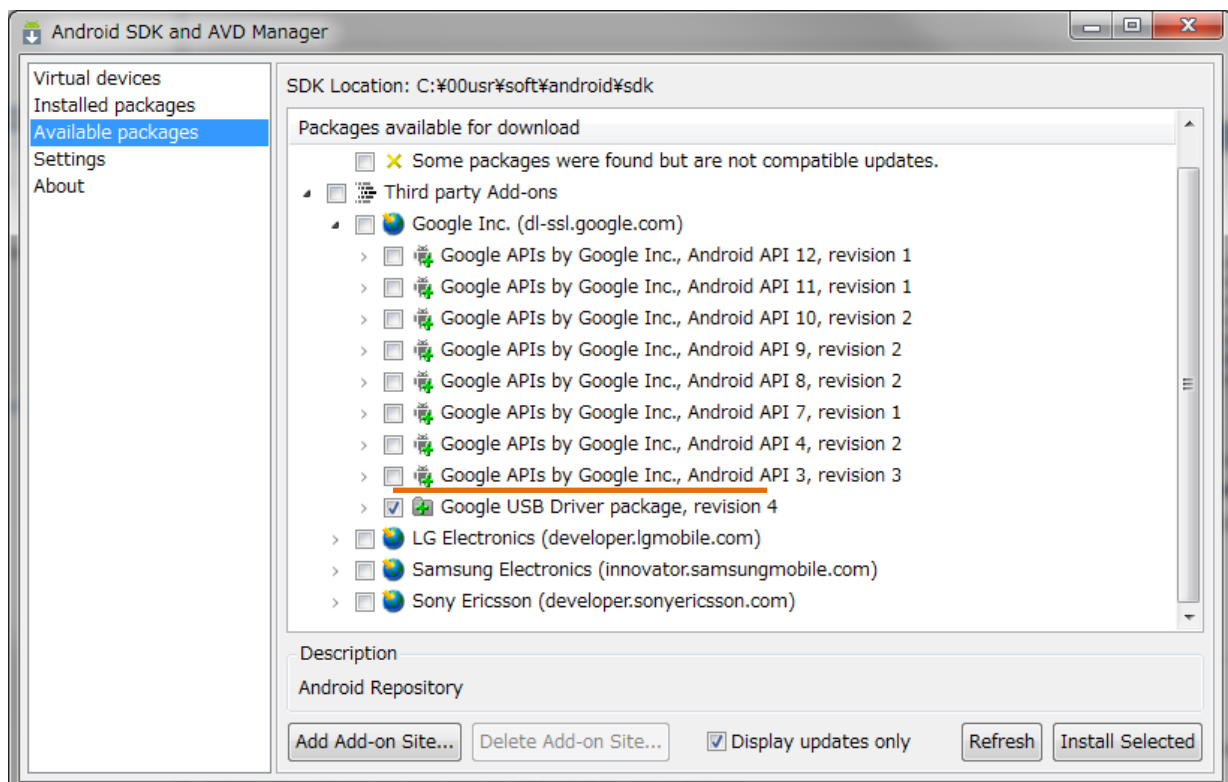
```
SUBSYSTEM=="usb_devide", SYSFS{idVendor}="0bb4", MODE="0666"
```

```
# Motorola
```

```
SUBSYSTEM=="usb", SYSFS{idVendor}="22b8", MODE="0666"
```

```
SUBSYSTEM=="usb_devide", SYSFS{idVendor}="22b8", MODE="0666"
```

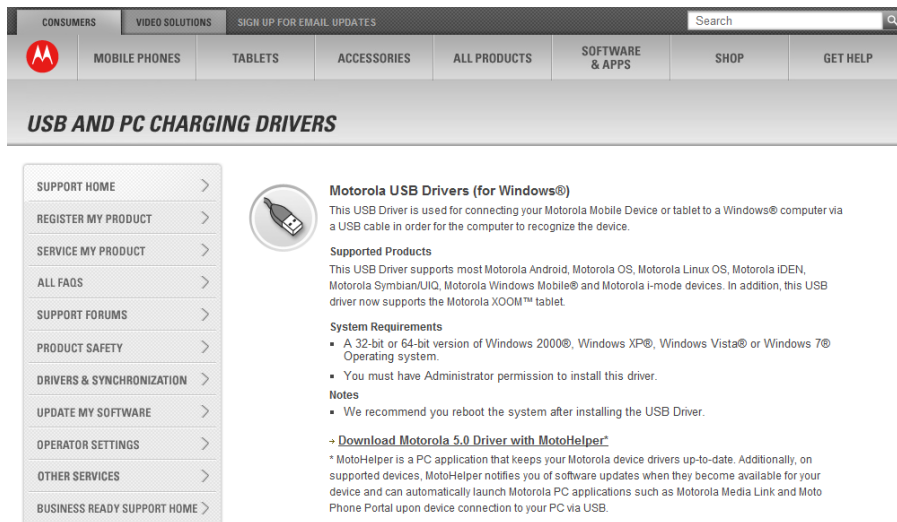
Windows : Install by SDK & AVD Manager



Motorola XOOM Usb Driver for

Download from below URL.

<http://www.motorola.com/Support/US-EN/Support-Homepage/Software and Drivers/USB-and-PC-Charging-Drivers>



The screenshot shows the Motorola Support website interface. At the top, there is a navigation bar with links for CONSUMERS, VIDEO SOLUTIONS, and SIGN UP FOR EMAIL UPDATES. Below this is a search bar and a menu with categories: MOBILE PHONES, TABLETS, ACCESSORIES, ALL PRODUCTS, SOFTWARE & APPS, SHOP, and GET HELP. The main heading is "USB AND PC CHARGING DRIVERS". On the left, there is a sidebar with links: SUPPORT HOME, REGISTER MY PRODUCT, SERVICE MY PRODUCT, ALL FAQs, SUPPORT FORUMS, PRODUCT SAFETY, DRIVERS & SYNCHRONIZATION, UPDATE MY SOFTWARE, OPERATOR SETTINGS, OTHER SERVICES, and BUSINESS READY SUPPORT HOME. The main content area features a USB cable icon and the title "Motorola USB Drivers (for Windows®)". It explains that the USB Driver is used for connecting Motorola Mobile Devices or tablets to a Windows® computer via a USB cable. It lists supported products (Motorola Android, Motorola OS, Motorola Linux OS, Motorola iDEN, Motorola Symbian/UIQ, Motorola Windows Mobile®, and Motorola i-mode devices) and system requirements (32-bit or 64-bit version of Windows 2000®, Windows XP®, Windows Vista®, or Windows 7® Operating system). It also includes notes about Administrator permission and a recommendation to reboot the system after installation. A link to "Download Motorola 5.0 Driver with MotoHelper" is provided, along with a note explaining that MotoHelper is a PC application that keeps drivers up-to-date and notifies of software updates.

Motorola USB Drivers (for Windows®)

This USB Driver is used for connecting your Motorola Mobile Device or tablet to a Windows® computer via a USB cable in order for the computer to recognize the device.

Supported Products

This USB Driver supports most Motorola Android, Motorola OS, Motorola Linux OS, Motorola iDEN, Motorola Symbian/UIQ, Motorola Windows Mobile® and Motorola i-mode devices. In addition, this USB driver now supports the Motorola XOOM™ tablet.

System Requirements

- A 32-bit or 64-bit version of Windows 2000®, Windows XP®, Windows Vista® or Windows 7® Operating system.
- You must have Administrator permission to install this driver.

Notes

- We recommend you reboot the system after installing the USB Driver.

Download Motorola 5.0 Driver with MotoHelper*

* MotoHelper is a PC application that keeps your Motorola device drivers up-to-date. Additionally, on supported devices, MotoHelper notifies you of software updates when they become available for your device and can automatically launch Motorola PC applications such as Motorola Media Link and Moto Phone Portal upon device connection to your PC via USB.

2.3. Emulator

1. create avd (Android Virtual Device).

Terminal

```
$ android create avd -n <avd_name> -t <targetID> -s <skin_name>| <width>-<height>
```

```
develop@hoover:[android]$ android list
Available Android targets:
id: 1 or "android-3"
    Name: Android 1.5
    Type: Platform
    API level: 3
    Revision: 4
    Skins: QVGA-L, HVGA-L, HVGA (default), HVGA-P, QVGA-P
id: 2 or "android-4"
    Name: Android 1.6
    Type: Platform
    API level: 4
    Revision: 3
    Skins: QVGA, HVGA, WVGA800 (default), WVGA854
id: 3 or "android-7"
```

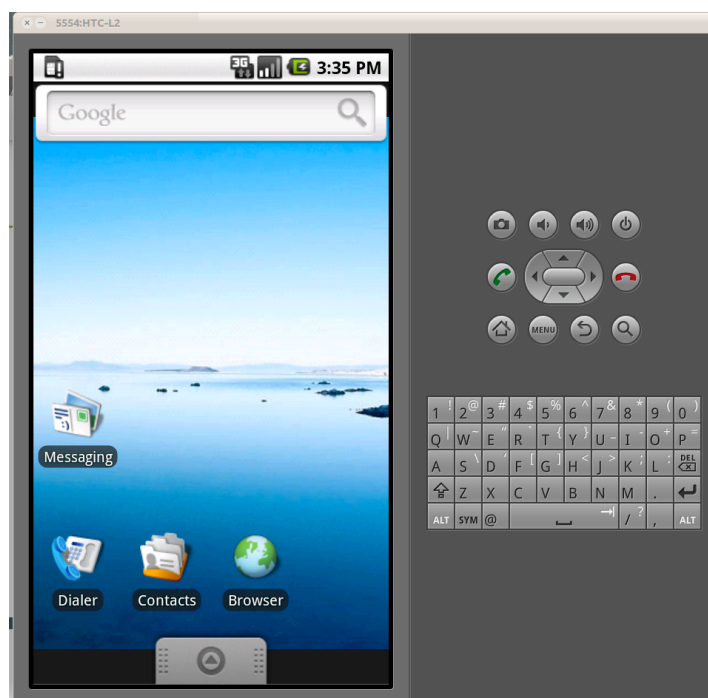
Target ID

Skin Names

2. execute emulator.

Terminal

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



2.4. Debugger

1. execute ddms (Dalvik Debug Monitor Server) .

Terminal

```
$ ddms
```

DDMS Window

The screenshot shows the Dalvik Debug Monitor (DDMS) window. The top section displays a list of processes, including 'HT979LF00032' (com.google.android.settings) which is online and running on API level 1.6. The 'Threads' tab is selected, showing a list of threads with columns for ID, Tid, Status, utime, stime, and Name. The threads listed include 'main', 'HeapWorker', 'Signal Catcher', 'JDWP', and 'Binder Thread #1'. Below the threads list is a 'Log' view showing system messages, such as 'No window to dispatch pointer action 1', 'Starting activity: Intent { act=android.intent.action.MAIN cmp=com.android.settings/.DateTimeSettings }', and 'Auto time state changed'.

How to get Screen Capture

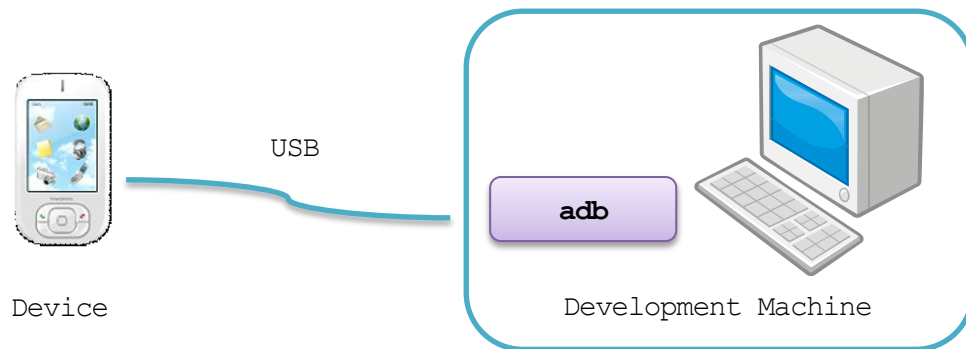
Device > Screen capture > Opened Dialog Save

The screenshot shows the Dalvik Debug Monitor (DDMS) window with the 'Device' menu open. The 'Screen capture...' option is highlighted, which is used to capture the screen of the device being debugged. The menu also includes options like 'File Explorer...', 'Show process status...', 'Dump device state...', 'Dump app state...', 'Dump radio state...', and 'Run logcat...'. The background shows the same process and thread information as the previous screenshot.

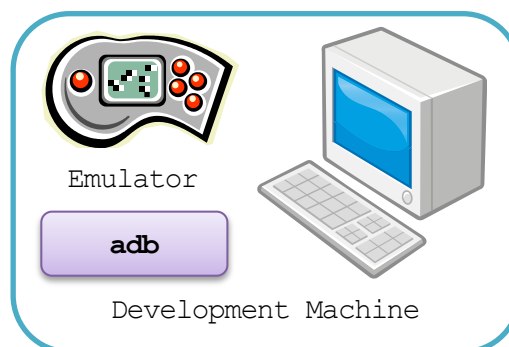
2.5. Device Arrangement

This section is Development Tools arrangement example .
Simple arrangement is connecting one device or one emulator .

Device - Development Machine



Emulator - Development Machine



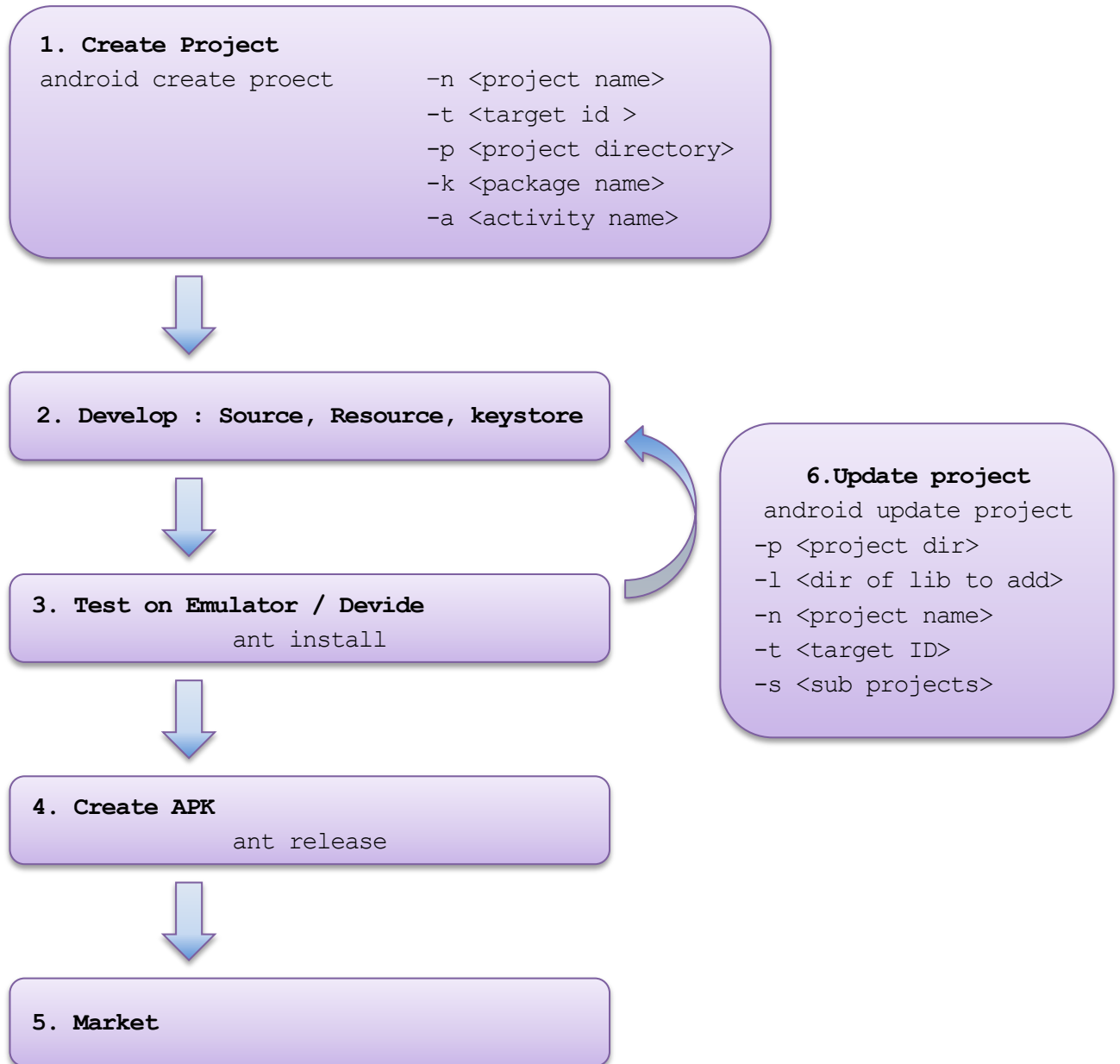
Check Devices

Terminal

```
$ adb devices
List of devices attached
HT979LF00032    device
Emulator-5554   device
```

3. Development

3.1. Cycle



3.2. Market

see Android Market Topics

see <http://developer.android.com/guide/publishing/licensing.html>

see <http://developer.android.com/guide/market/billing/index.html>

4. Application

This chapter is building app Hello World and application's information .
The application's information app build by follow steps .

- 4.2 Create application project .
- 4.3 Update build.xml .
- 4.4 Create private key for sign .
- 4.5 Edit AndroidManifest.xml and source .
- 4.6 Install to device and run app.

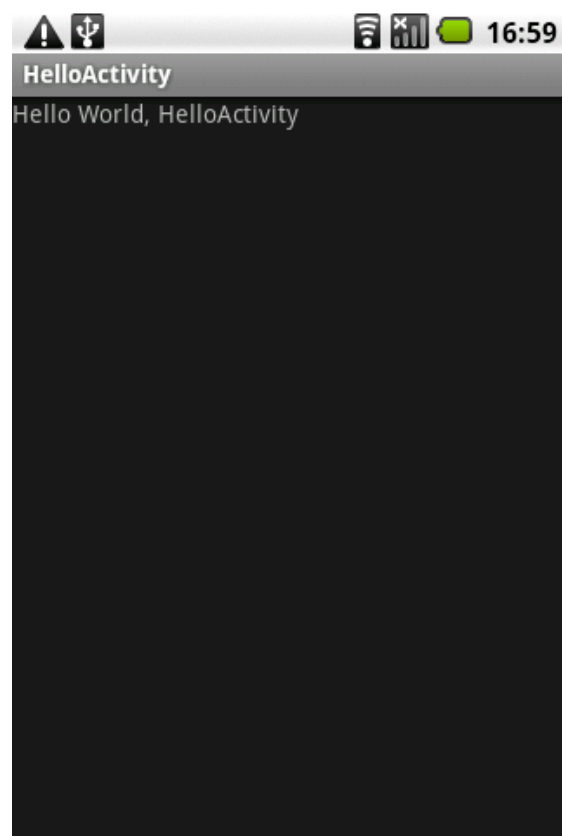
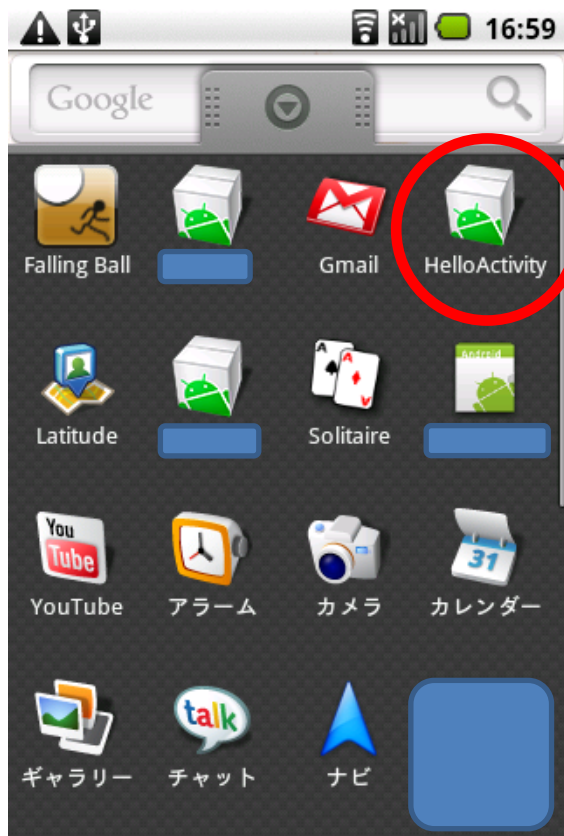
4.1. HelloWorld

This section create Hello World Application and install it to device.

Terminal

```
$ android create project -n Hello -t 2 -p Hello -k app.hello -a HelloActivity  
  
$ ant install
```

Install & Run Application Result



4.2. Create Application Information Project

This section is createing Aplicaton Information app project.

Terminal

```
$ android create project -n AppInfo -t 2 -p AppInfo -k app.AppInfo -a AppInfo
```

4.3. Update build.xml

This section is customising build.xml and adding releasetool.xml for easy development and release .

Update build.xml for easy development .

```
<?xml version="1.0" encoding="UTF-8"?>
<project name="Your Project" default="help">
  <property file="../local.properties" />
  :
  : Original Ant Targets
  :
  <!--Clean and Compile -->
  <target name="compapp" depends="clean,compile" />
  <!--Clean and Install -->
  <target name="app" depends="clean,install" />

</project>
```

This file depend on local environment.
Be careful committing to source repository .

4.4. Create Private Key

Application requires digitally sign . This section is creating private key by releasetool.xml

Terminal

```
$ ant -f ../tool.xml keystore

$ ls
AndroidManifest.xml  apk.sign.keystore  ...
```


4.5. Edit Source and Manifest

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="app.appinfo"
    android:versionCode="1"
    android:versionName="Application&#xa;Information&#xa;1.0.0&#xa;
    Organization.&#xa;All rights reserved.">
    <application android:label="@string/app_name">
```

Version code

Version Name

app.appinfo.AppInfo.java

```
/**
 * @return version Code
 */
public int getVersionCode() {
    try {
        PackageManager manager = getPackageManager();
        PackageInfo packInfo = manager.getPackageInfo(getPackageName(),
            PackageManager.GET_ACTIVITIES);
        return packInfo.versionCode;
    } catch (Exception e) {
        return -1;
    }
}

/**
 * @return version Name
 */
public String getVersionName() {
    try {
        PackageManager manager = getPackageManager();
        PackageInfo packInfo = manager.getPackageInfo(getPackageName(),
            PackageManager.GET_ACTIVITIES);
        return packInfo.versionName;
    } catch (Exception e) {
        return "---";
    }
}
```

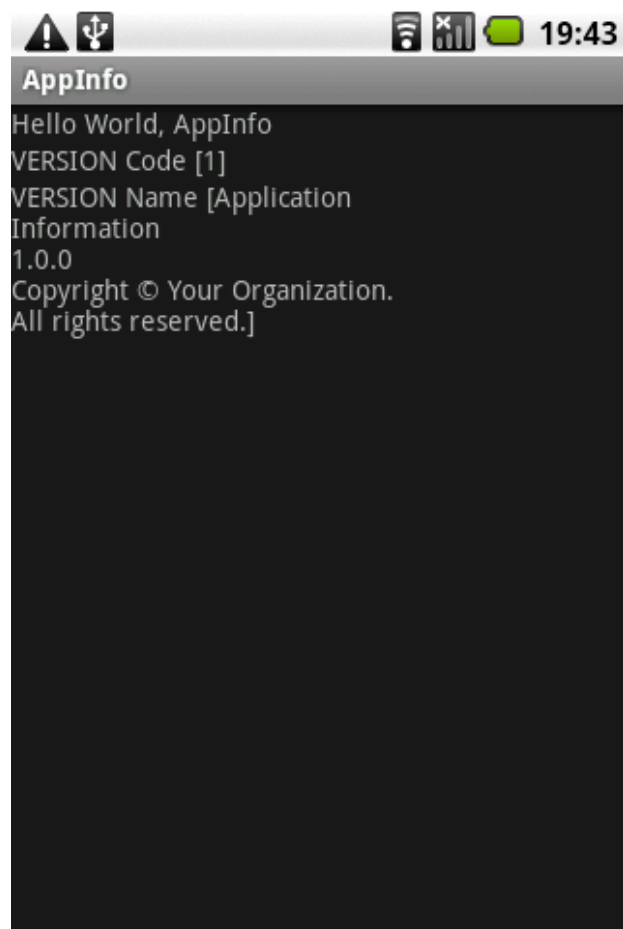
4.6. Install APK and Run app

After create apk file . install application to device .

Terminal

```
$ adb install dist/AppInfo.apk
```

Application Result

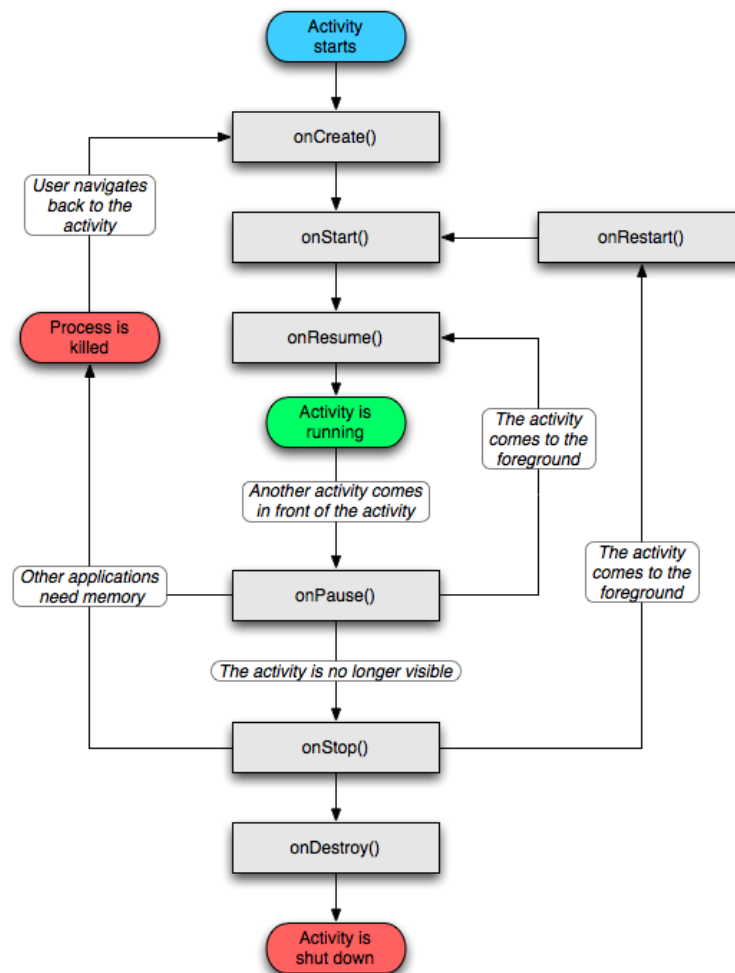


5. Activity

5.1. Lifecycle

Activity state path diagram .

URL : <http://developer.android.com/reference/android/app/Activity.html>



Result Log

Terminal

```
$ adb logcat -d | tail -n 20
I/ActivityManager( 76): Start proc app.simple for activity app.simple/.SimpleApp: pid=3438 uid=10042 gids={1015}
I/ActivityManager( 76): Displayed activity app.simple/.SimpleApp: 1047 ms (total 1047 ms)
I/ActivityManager( 76): Starting activity: Intent { cmp=app.simple/.LifecycleActivity }
I/LifecycleActivity( 3438): onCreate()
I/LifecycleActivity( 3438): onStart()
I/LifecycleActivity( 3438): onResume
I/ActivityManager( 76): Displayed activity app.simple/.LifecycleActivity: 136 ms (total 136 ms)
I/LifecycleActivity( 3438): onPause
I/LifecycleActivity( 3438): onStop
I/LifecycleActivity( 3438): onDestroy
```

5.2. Start Activity

```
SimpleApp/src/app/simple/SimpleApp.java
```

```
Intent intent = new Intent();  
intent.setClass(SimpleApp.this, LifecycleActivity.class);  
startActivity(intent);
```

5.3. Resource

name	description
------	-------------

res/drawable	
res/drawable-xdpi	ldpi, mdpi, hdpi, nodpi, xdpi
res/layout	default UI layout xml
res/layout-land	UI layout xml in landscape
res/values/	default resource
res/values-{lang}	each language resource
strings.xml	resource file
arrays.xml	TypedArray resource file

* drawable

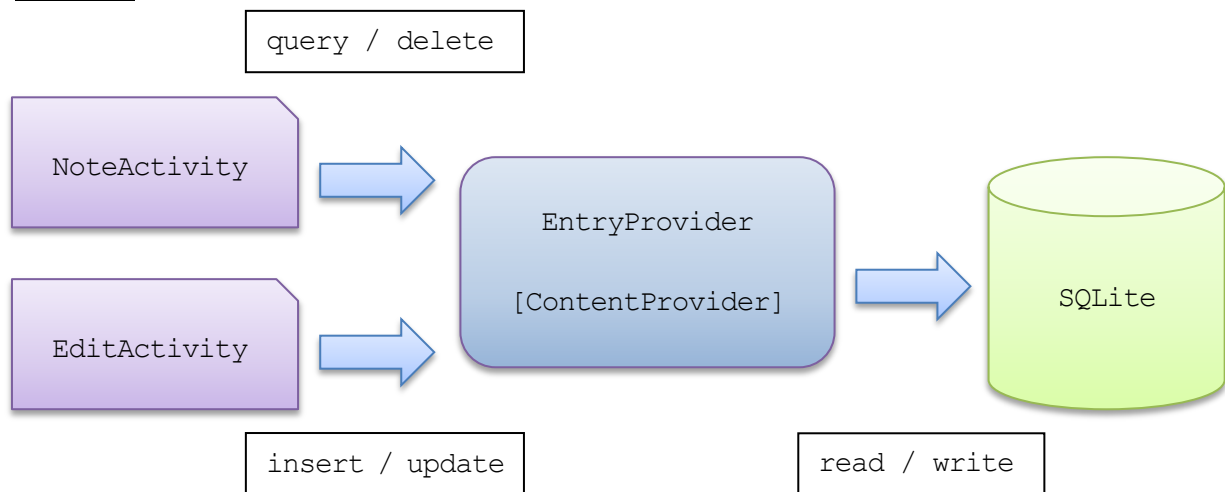
http://developer.android.com/guide/practices/screens_support.html#qualifiers

* Typed Array

<http://developer.android.com/guide/topics/resources/more-resources.html#TypedArray>

6. ContentProvider & SQLite

6.1. Overview



6.2. Edit Source

name	description
------	-------------

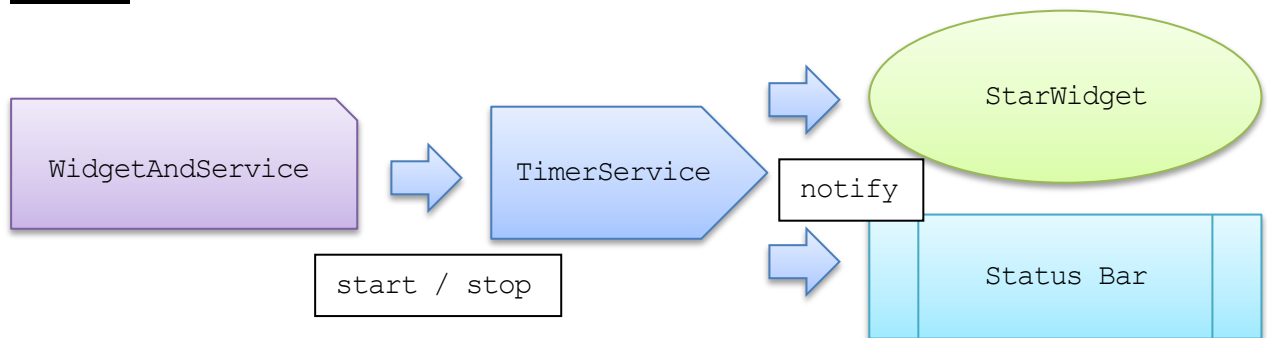
src/app/note	Application source
NoteActivity	List Activity
EditActivity	Edit Entry form Activity
EntryProvider	ContentProvider
res/layout	Application layout
main.xml	List view layout
edit.xml	edit form layout
row.xml	list item layout

6.3. Show Database

```
develop@hooover:[Note]$ adb shell
# cd /data/data/app.note/databases
# sqlite3 entry.db
SQLite version 3.5.9
Enter ".help" for instructions
sqlite> select * from entries ;
1|aaa|test|1307270141264
sqlite> .quit
# exit
develop@hooover:[Note]$
```

7. Widget & Service

7.1. Overview



7.2. Edit Source & Resource

name	description
src/app/ws	Application source
WidgetAndService	Service controller
TimerService	Notify Service
StarWidget	Widget
res/drawable	Application layout
some icons	Widget Icons
res/layout	
main.xml	Service controller layout
widget.xml	widget layout
res/xml	
star.xml	widget configuration file

7.3. Run



8. UnitTest

8.1. Create Test Project

Terminal

```
$ android create test-project -p <project dir> -m <target project dir> -n <project name>
```

#Example

```
$ android create test-project -p NoteTest -m ../Note -n NoteTest
```

8.2. Edit Testcase

name	description
------	-------------

src/app/note	Test Application source
--------------	-------------------------

NoteActivityTest	Testcase
------------------	----------

8.3. Run

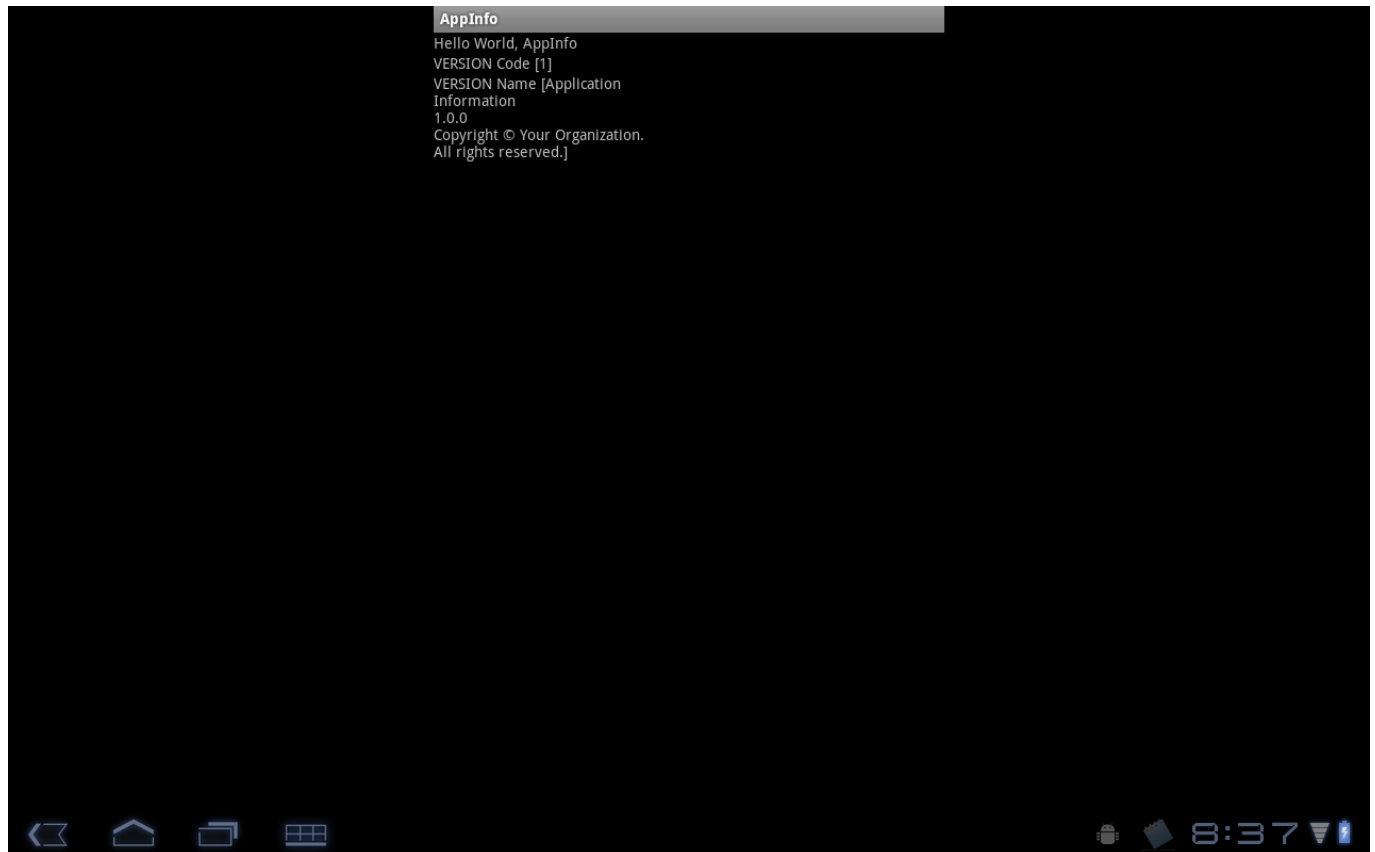
After install test project application, execute adb command.

```
develop@hooover:[NoteTest]$ adb shell am instrument -w app.note.tests/android.test.InstrumentationTestRunner
the package of com.example.android.apis. To run the tests use the command:
app.note.NoteActivityTest:.
Test results for InstrumentationTestRunner=.
Time: 1.429
OK (1 test)
```

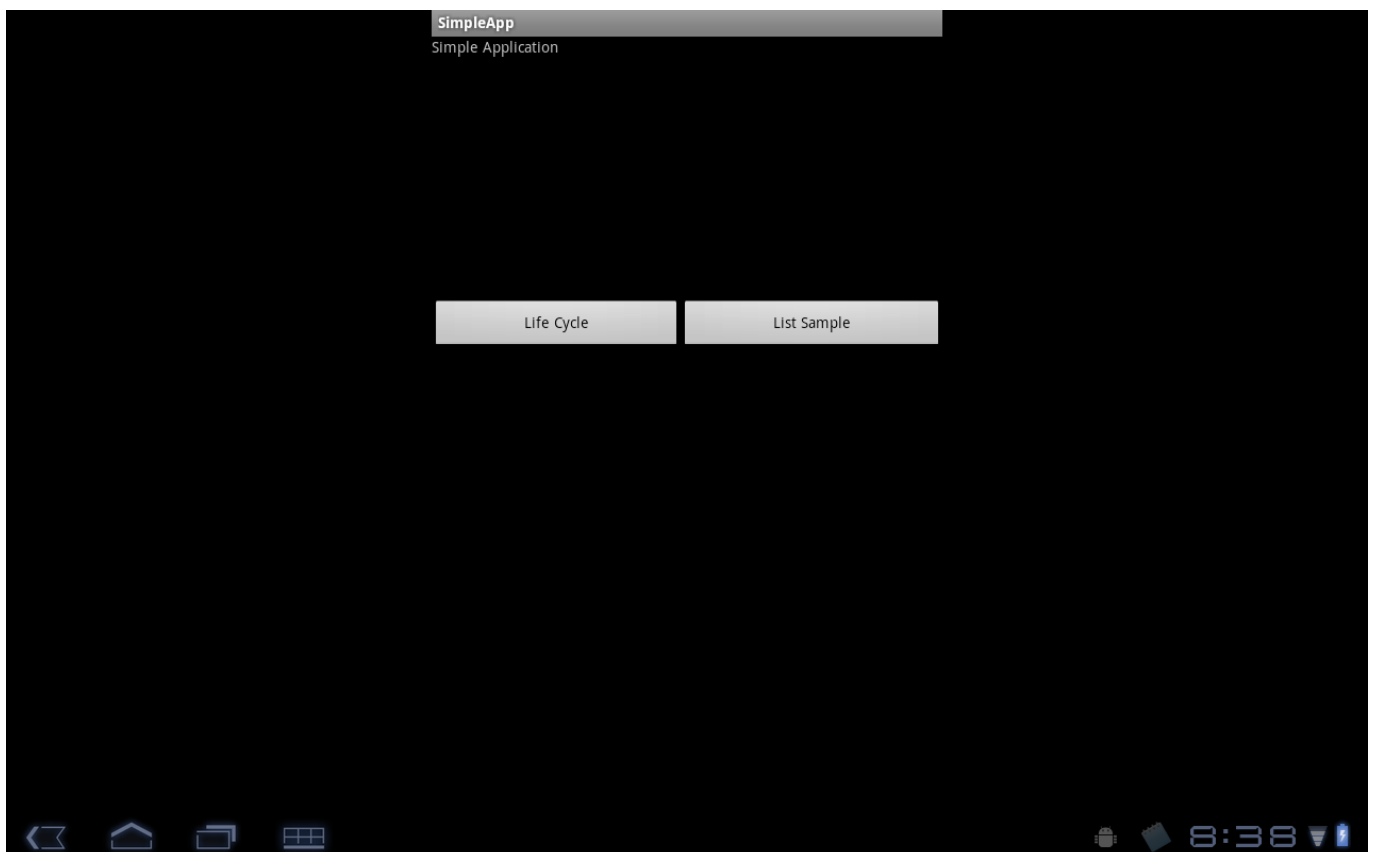
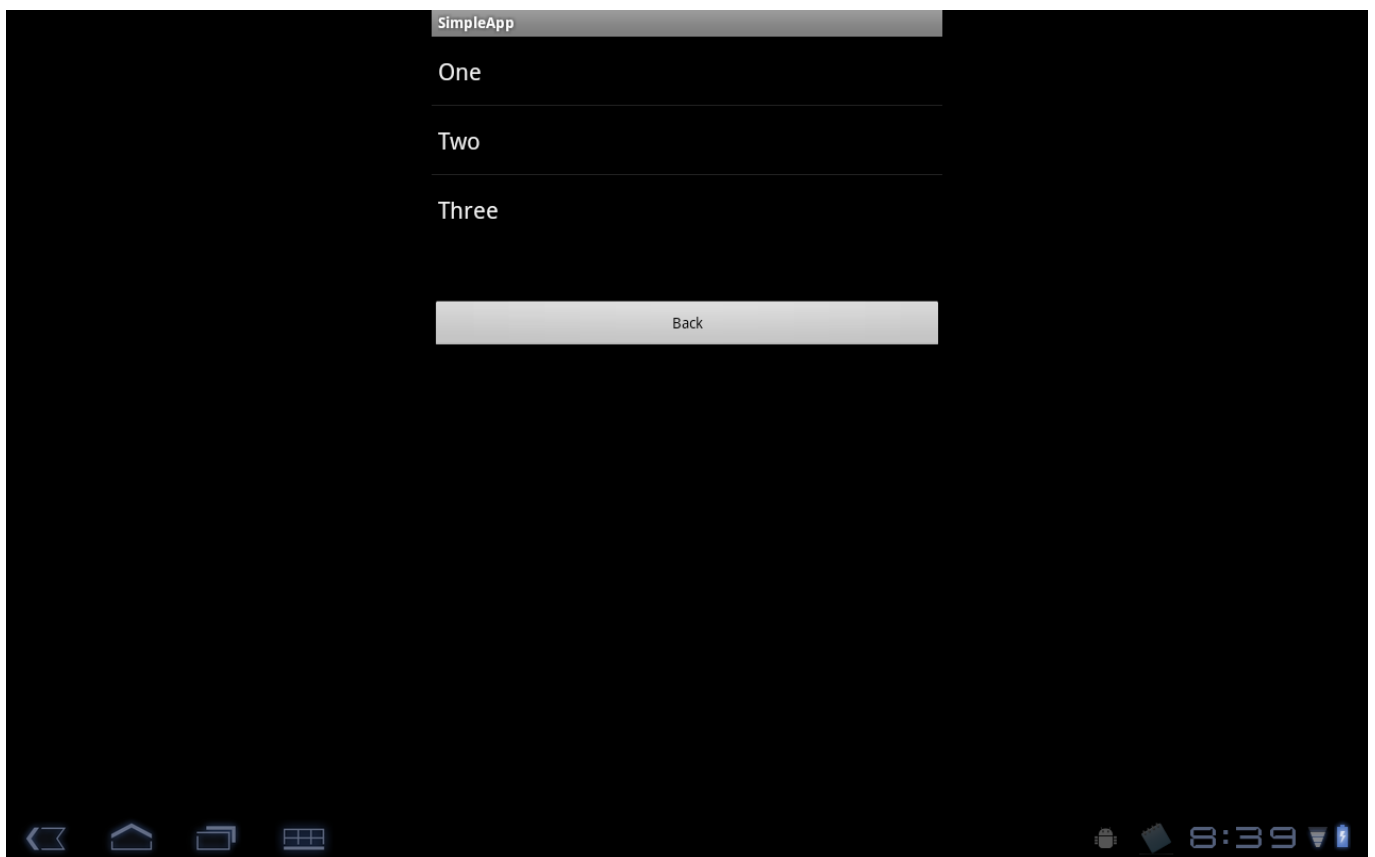

Appendix

1. Applications run on Motorola

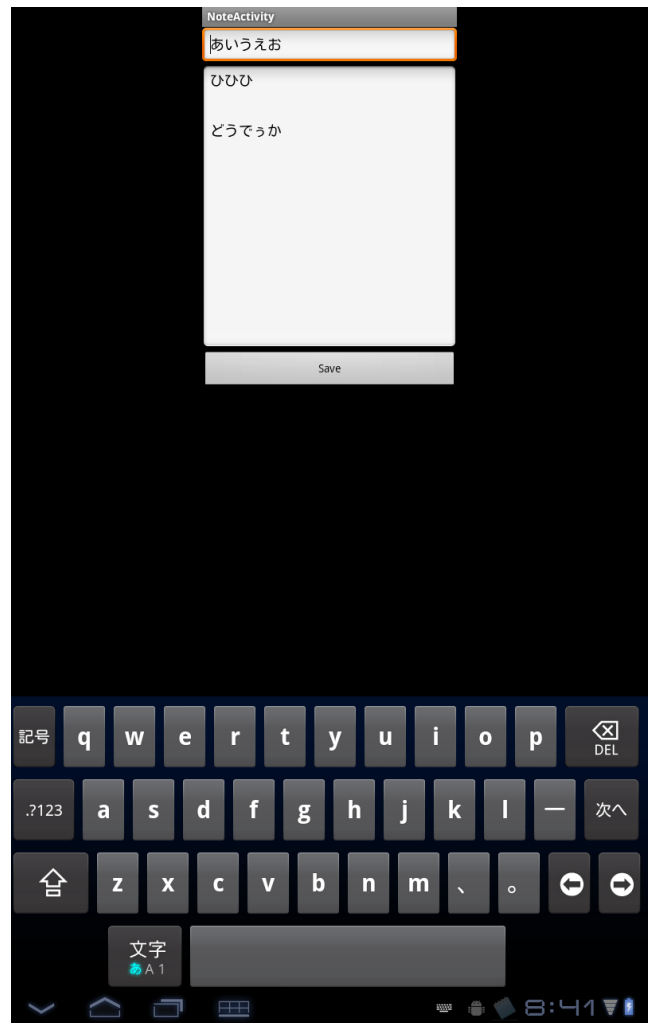
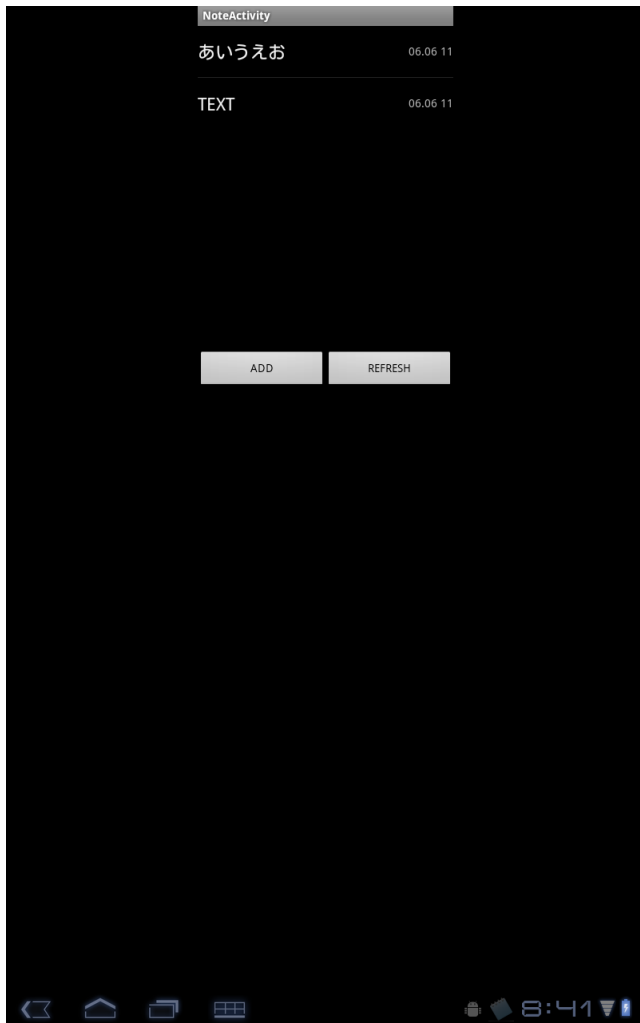
AppInfo



SimpleApp



Note



WidgetAndService

