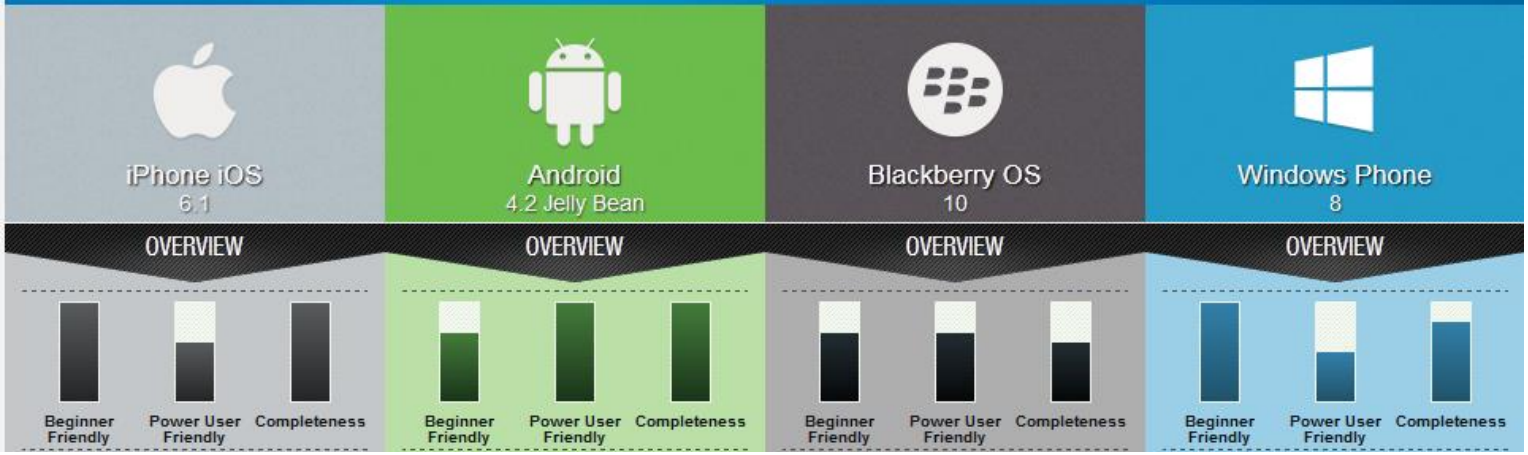
# Programming mobile devices

Introduction to Android

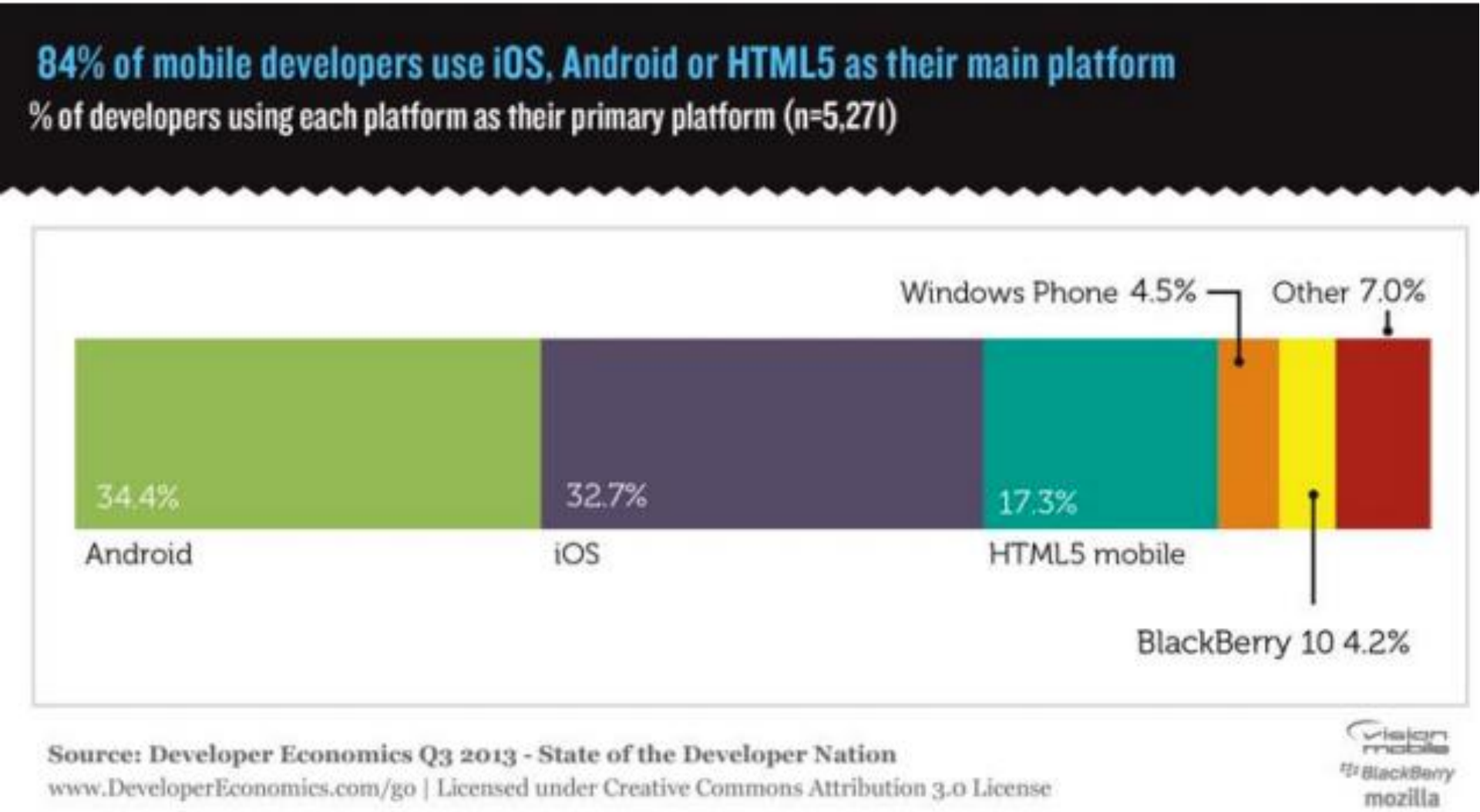
# War of platforms




http://myphonedeals.co.uk/blog/33-the-smartphone-os-complete-comparison-chart

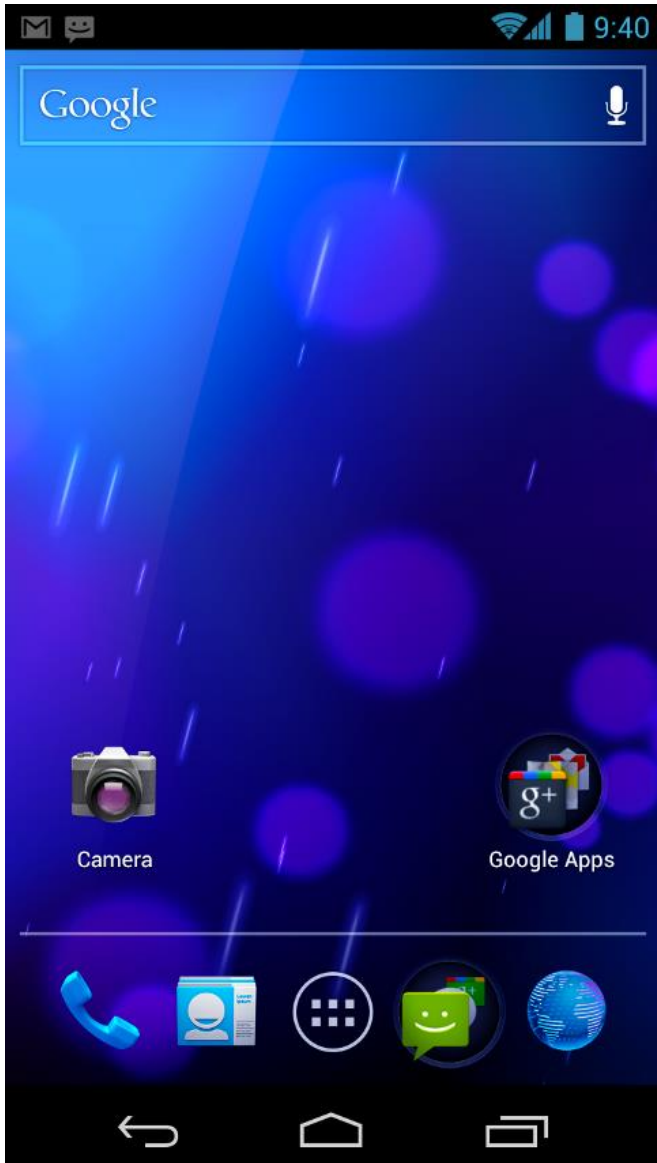

www.techradar.com/news/phone-and-communications/mobile-phones/ios7-vs-android-jelly-bean-vs-windows-phone-8-vs-bb10-1159893

# Market segmentation



# Android - what is it?

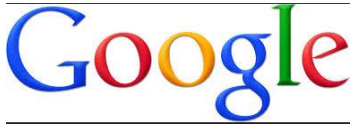
- An operating system based on the Linux kernel and philosophy, adapted for mobile devices.
- Developed by the "Open Handset Alliance" (including Google).
- An Open Source project.



# A short history of Android


**2003**

- Creation of **Android Inc.**
- Andy Rubin, Rich Miner, Nick Sears, Chris White




**2005**

- Android Inc. taken over by Google
- Google wanted to create „*a flexible and upgradable system*"

**2007**

- The world has seen the iPhone
- Creation of Open Handset Alliance
- First Android presentation

**2008**

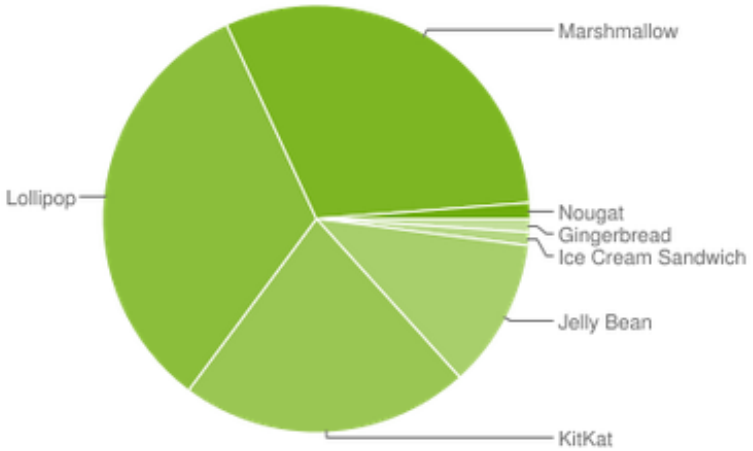
- First Android device – HTC Dream

What was Android for?

# Android Versions

## Target which Android version?

| Version | Codename | API | Distribution |
|---|---|---|---|
| 2.3.3 - 2.3.7 | Gingerbread | 10 | 1.0% |
| 4.0.3 - 4.0.4 | Ice Cream Sandwich | 15 | 1.0% |
| 4.1.x | Jelly Bean | 16 | 4.0% |
| 4.2.x | | 17 | 5.7% |
| 4.3 | | 18 | 1.6% |
| 4.4 | KitKat | 19 | 21.9% |
| 5.0 | Lollipop | 21 | 9.8% |
| 5.1 | | 22 | 23.1% |
| 6.0 | Marshmallow | 23 | 30.7% |
| 7.0 | Nougat | 24 | 0.9% |
| 7.1 | | 25 | 0.3% |

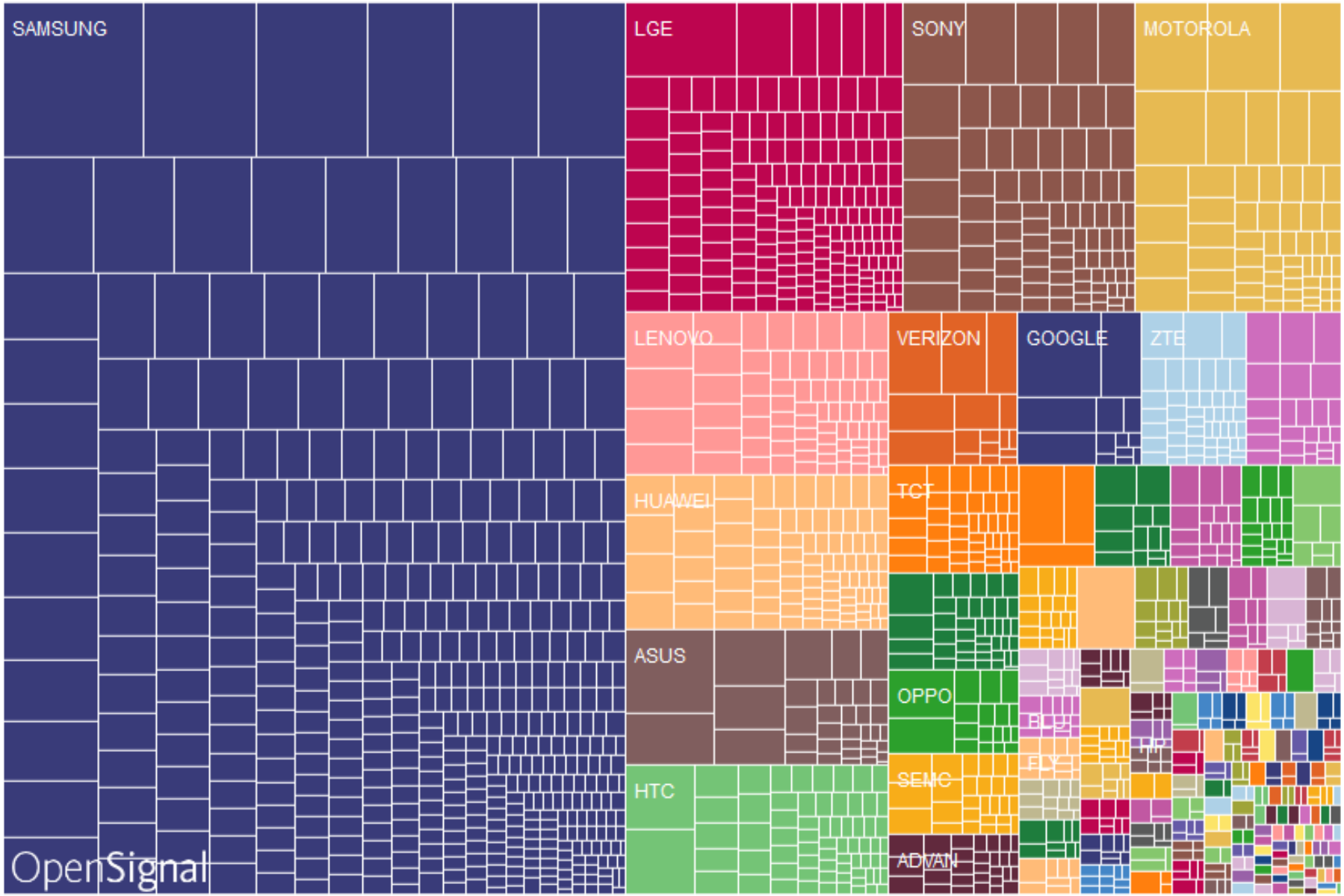



*Data collected during a 7-day period ending on February 6, 2017.*

*Any versions with less than 0.1% distribution are not shown.*

*https://developer.android.com/about/dashboards/index.html*

# Fragmentation - manufacturers

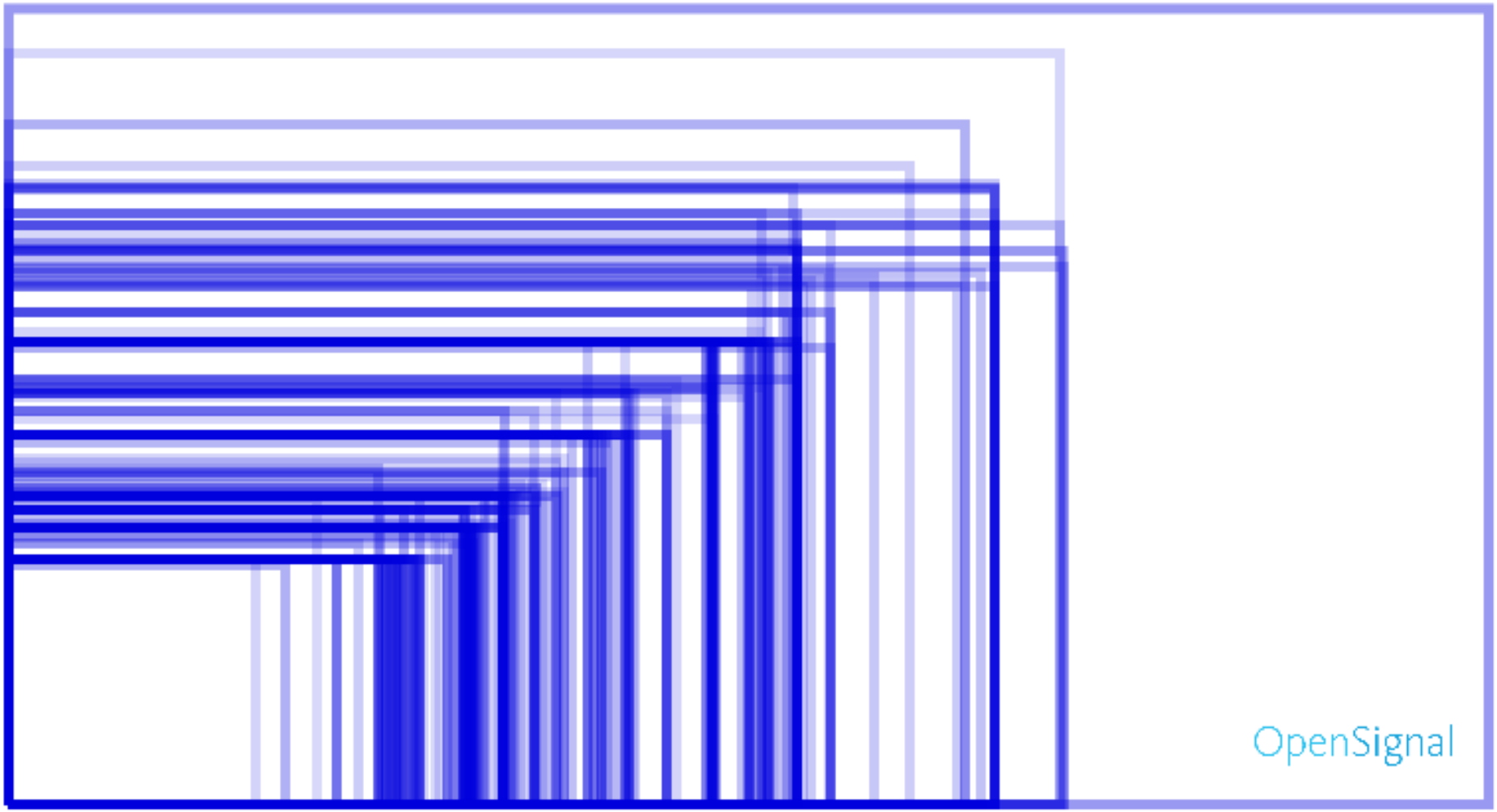



*http://opensignal.com/reports/2015/08/android-fragmentation/*

# Fragmentation - devices


*http://opensignal.com/reports/2015/08/android-fragmentation/*

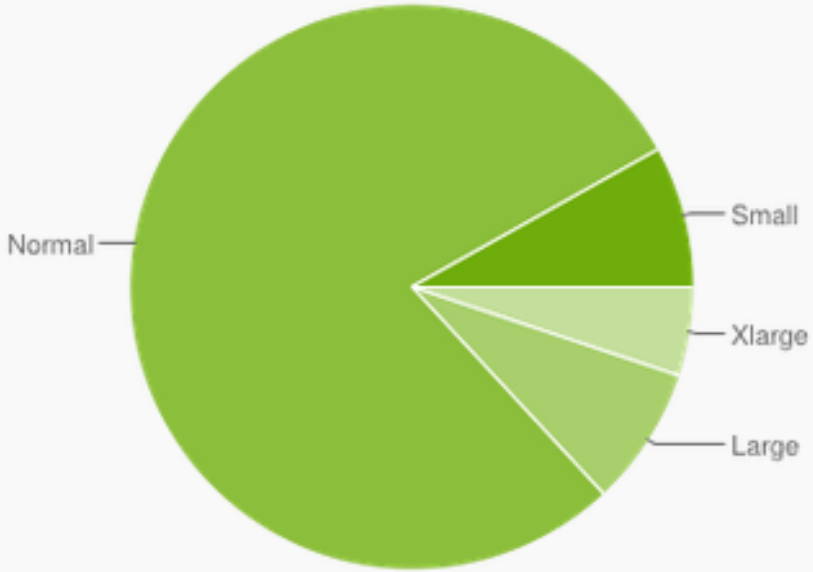
# Fragmentation - screen size

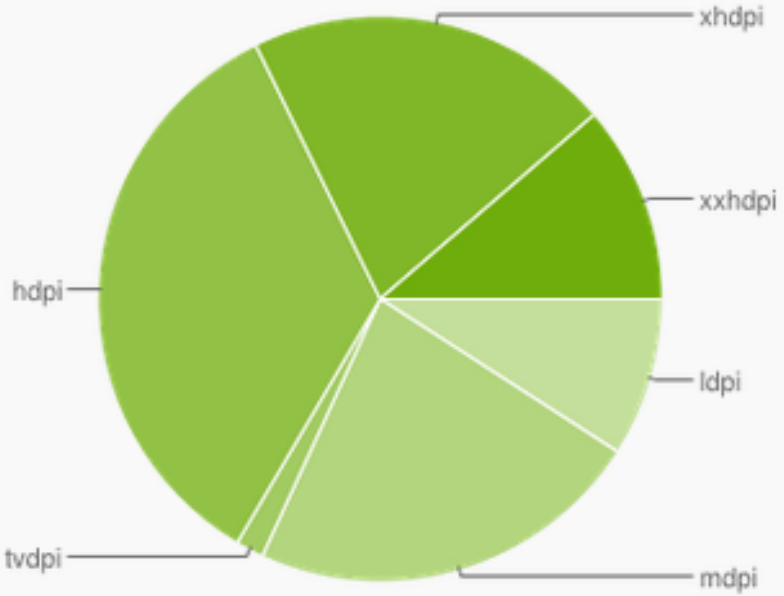



*http://opensignal.com/reports/2015/08/android-fragmentation/*

# Fragmentation - screen size

| | ldpi | mdpi | tvdpi | hdpi | xhdpi | xxhdpi | Total |
|---|---|---|---|---|---|---|---|
| Small | 8.1% | | | | | | 8.1% |
| Normal | 0.1% | 13.9% | | 33.3% | 20.2% | 11.3% | 78.8% |
| Large | 0.8% | 4.4% | 1.6% | 0.6% | 0.6% | | 8.0% |
| Xlarge | 0.1% | 4.5% | | 0.3% | 0.2% | | 5.1% |
| **Total** | 9.1% | 22.8% | 1.6% | 34.2% | 21.0% | 11.3% | |




*Data collected during a 7-day period ending on February 4, 2014.*
*Any screen configurations with less than 0.1% distribution are not shown.*

*https://developer.android.com/about/dashboards/index.html*

# Density Independence

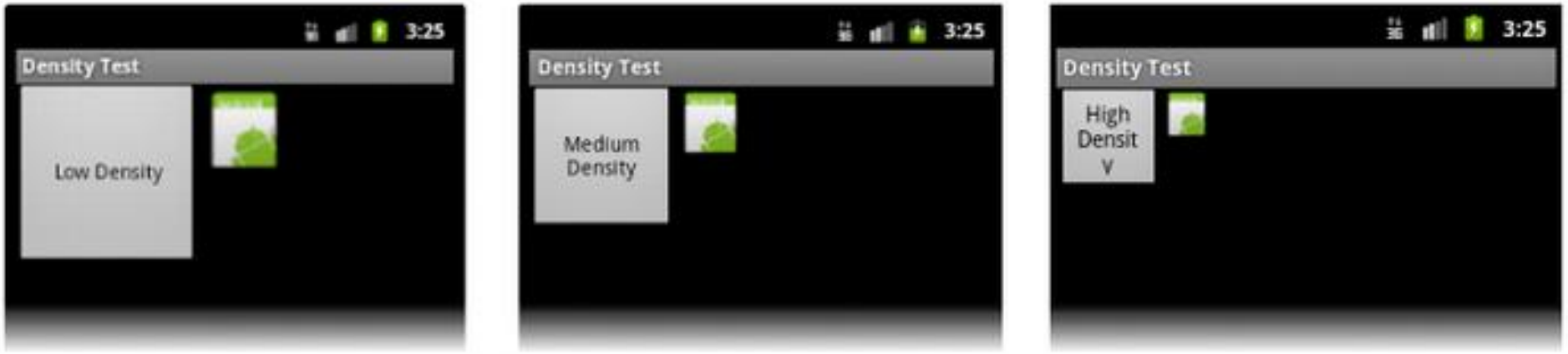
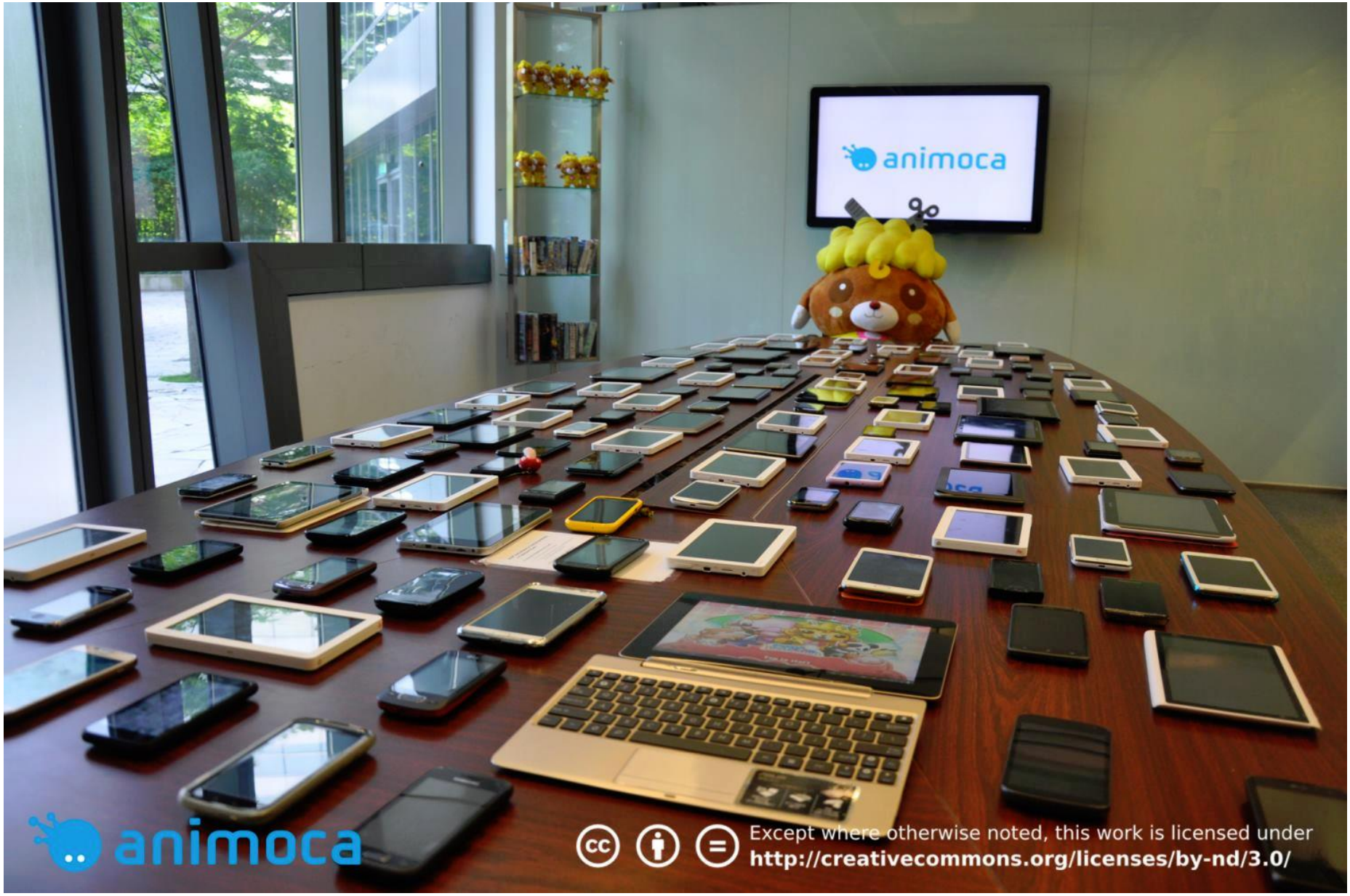


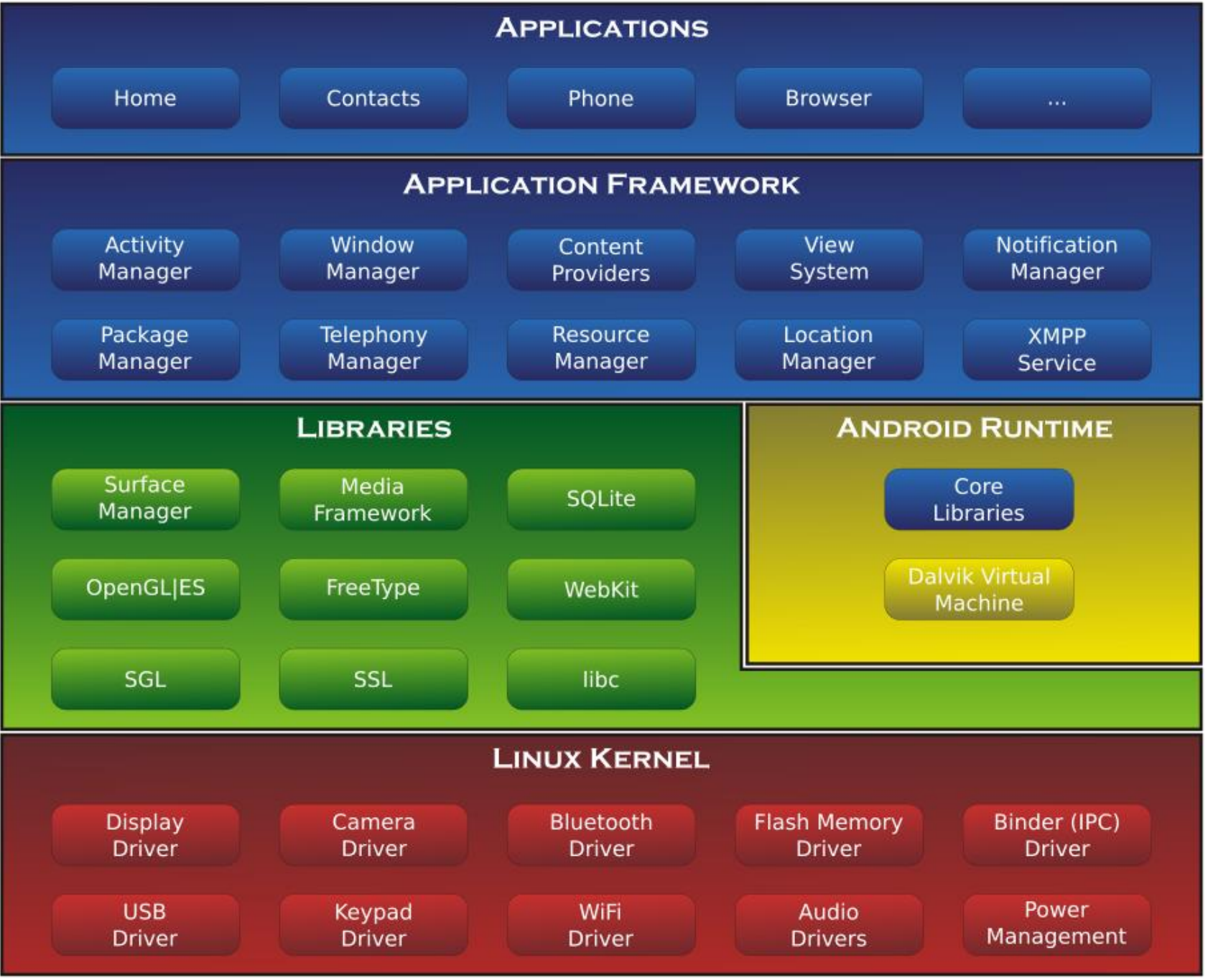

**Figure 2.** Example application without support for different densities, as shown on low, medium, and high density screens.


**Figure 3.** Example application with good support for different densities (it's density independent), as shown on low, medium, and high density screens.

*https://developer.android.com/guide/practices/screens_support.html*

# A collection of testing equipment






# Anatomy of Android

http://www.youtube.com/watch?v=QBGfUs9mQYY

# Lowest layer - kernel

- Android kernel = Linux kernel after minor modifications
- Kernel provides:
  - Hardware abstraction layer
  - Memory management
  - Process management
  - Network Management
  - Power management
  - Linux shell: `adb shell`
- Why the Linux kernel?

# Native libraries

- Code written in C or C ++

Examples:

- **Surfaces Manager** – screen management
- **OpenGL|ES** –2D and 3D graphics
- **Media Framework** – codecs (MPEG 4, H.264, MP3, AAC)
- **FreeType** – font rendering
- **SQLite** – data storage
- **Webkit** – web browser's engine

# Dalvik

- Android Runtime – adaptation to mobile device conditions (limited CPU, memory, battery resources)
- Dalvik – custom Java Virtual Machine implementation

Differences between JVM:

- Dalvik uses .dex files
- Different set of libraries than JDK
- More compact and efficient implementation
- Dalvik is a „register-based VM”

Why create a custom VM?

# Art (from 4.4 and up)

To turn on:

- In Android 4.4 ART (Android Runtime) has to be enabled in `Settings -> Developer options`.

Advantages:

- (AOT) Ahead-of-time compilation
- Improved garbage collector mechanism
- Improved debugger

```
java.lang.NullPointerException: Attempt to write to field 'int
android.accessibilityservice.AccessibilityServiceInfo.flags' on a
null object reference
```

```
java.lang.NullPointerException: Attempt to invoke virtual method
'java.lang.String java.lang.Object.toString()' on a null object reference
```

Disadvantages:

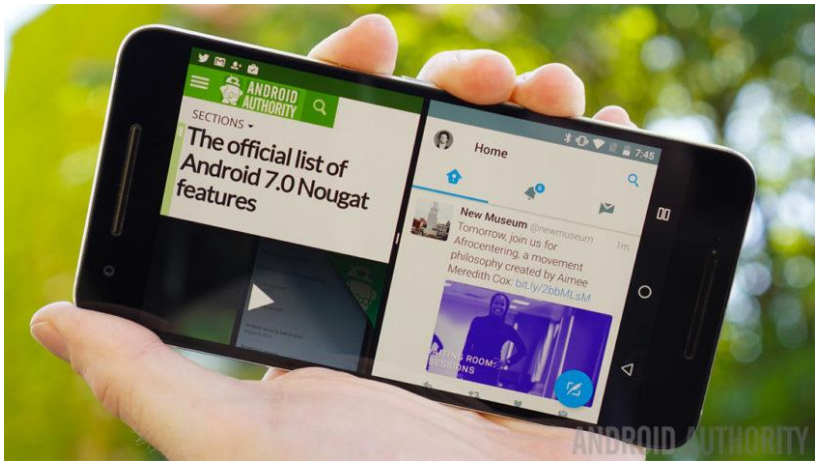
- Some very old apps may not work correctly

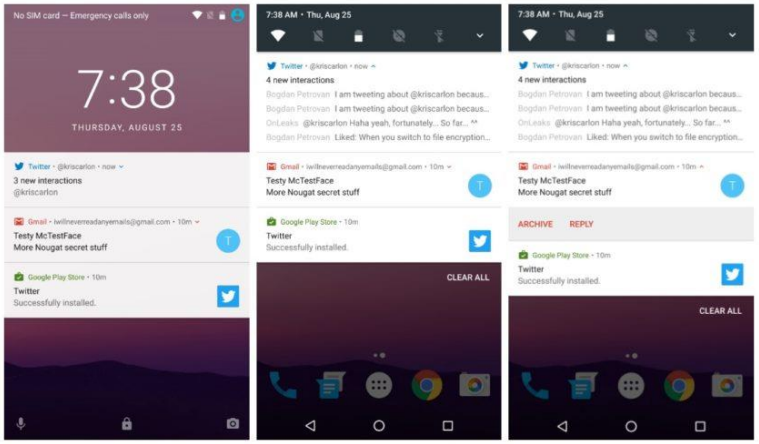
# Marshmallow - Android 6.0

- Changes in application permissions – the system asks for permission (the first time) exactly when it is needed.
- Support for fingerprint readers
- USB Type C support
- Improved voice assistant
- Android Pay payments
- Improvements in the energy saving system

# Nugat – what's new

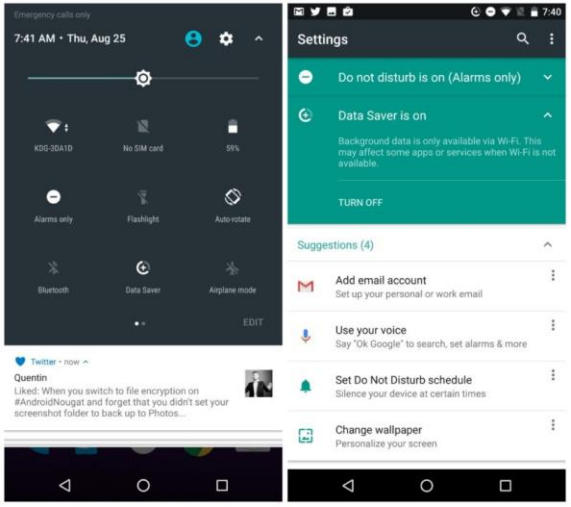
- Split screen mode
- Notification priorities



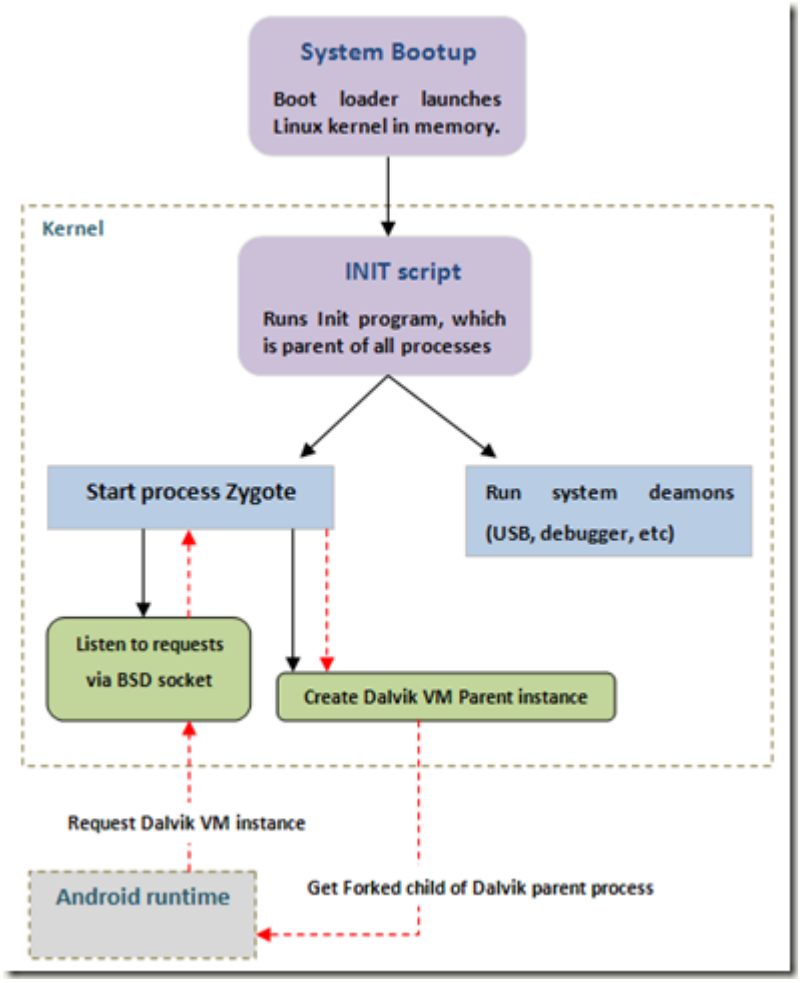

Improved doze mode

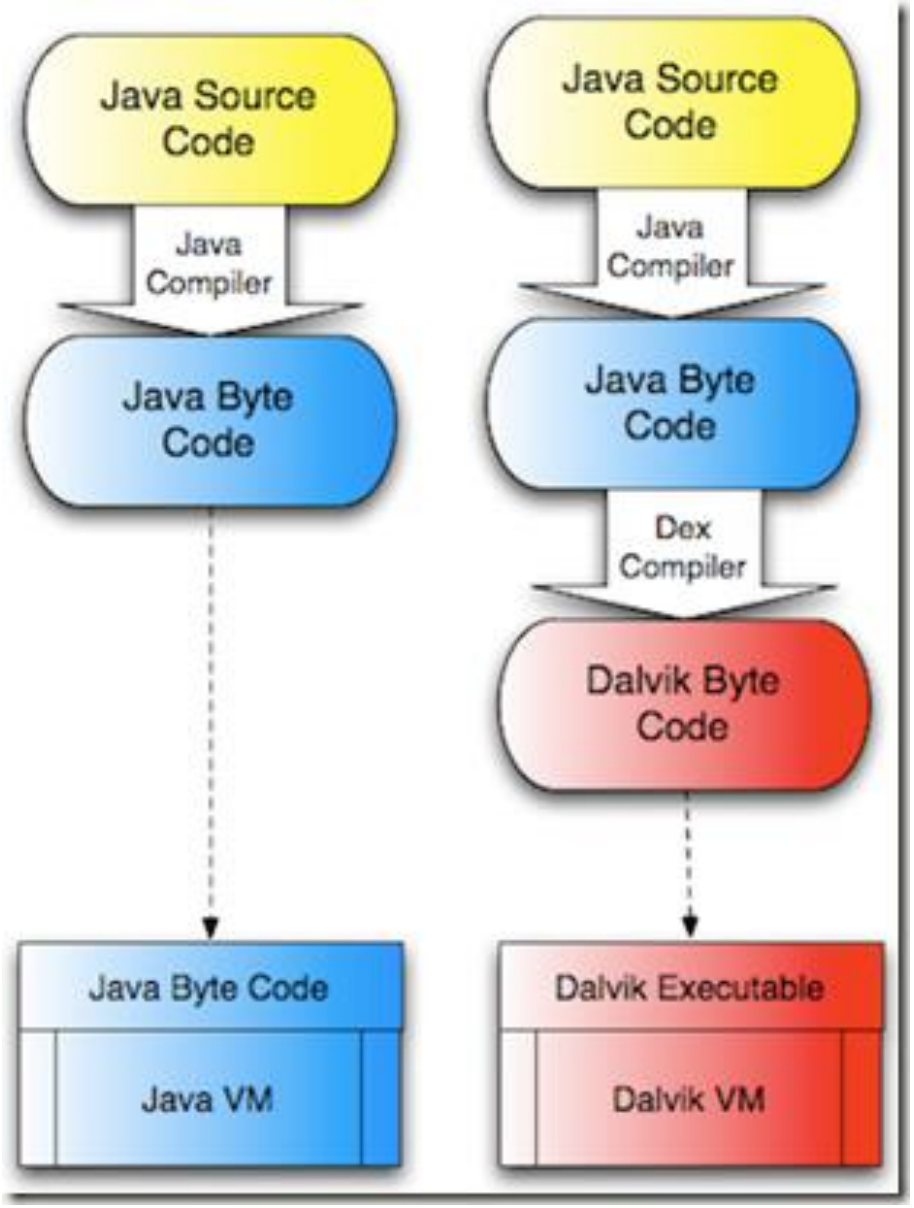
DPI changing possible




- Newer „Do not disturb" mode
- Support for **Vulkan API, Java 8**

# Dalvik VM




Why is the Java code not directly compiled into Dalvik Byte Code?

*http://markfaction.wordpress.com/2012/07/15/stack-based-vs-register-based-virtual-machine-architecture-and-the-dalvik-vm/*

# Application Framework

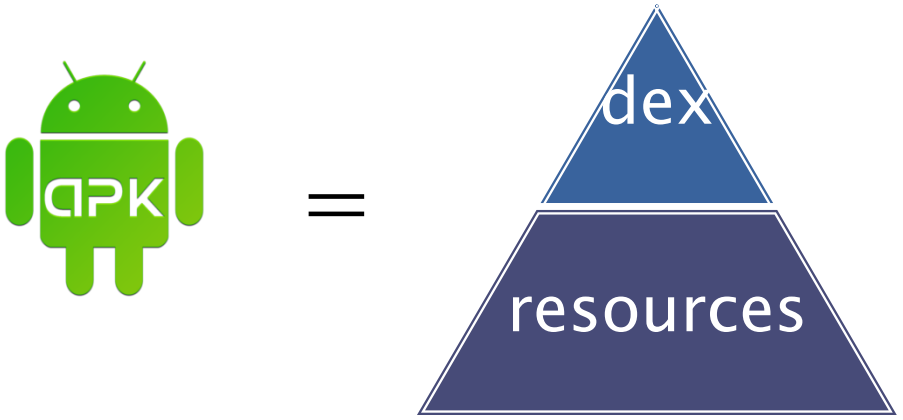
Main components:

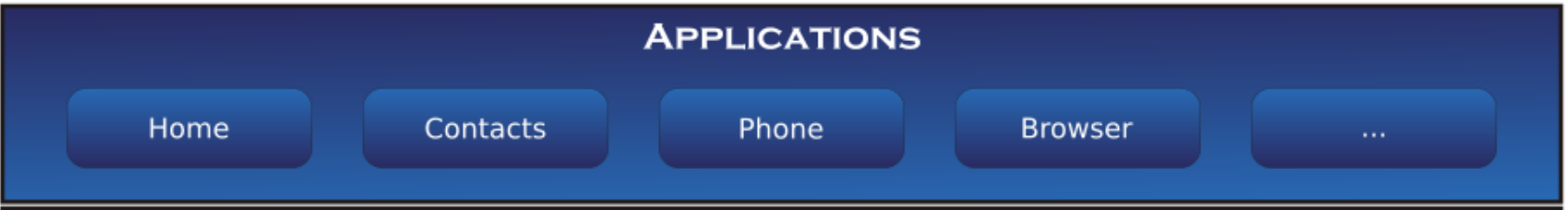
- **Activity Manager** – Life cycle management (and navigation) of the application
- **Package Manager** – Management of installed applications
- **Content Providers** – Manages data sharing between applications
- **View System** – GUI layer management


https://sites.google.com/site/io/inside-the-android-application-framework

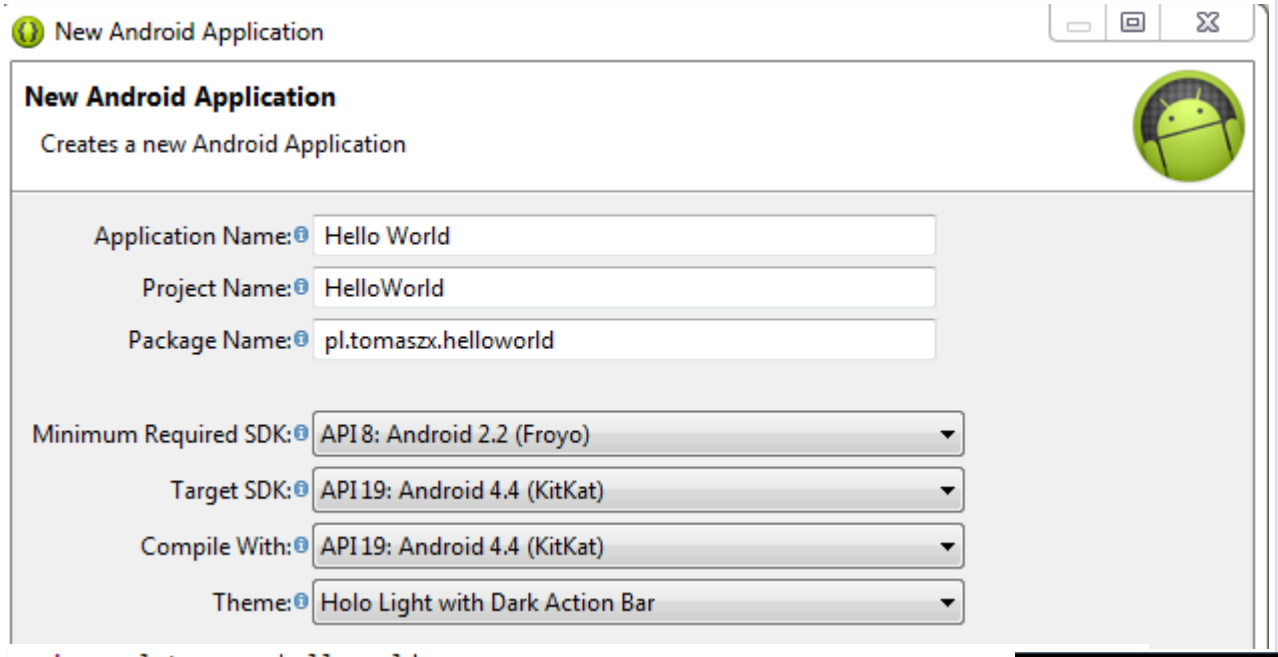
# Apps

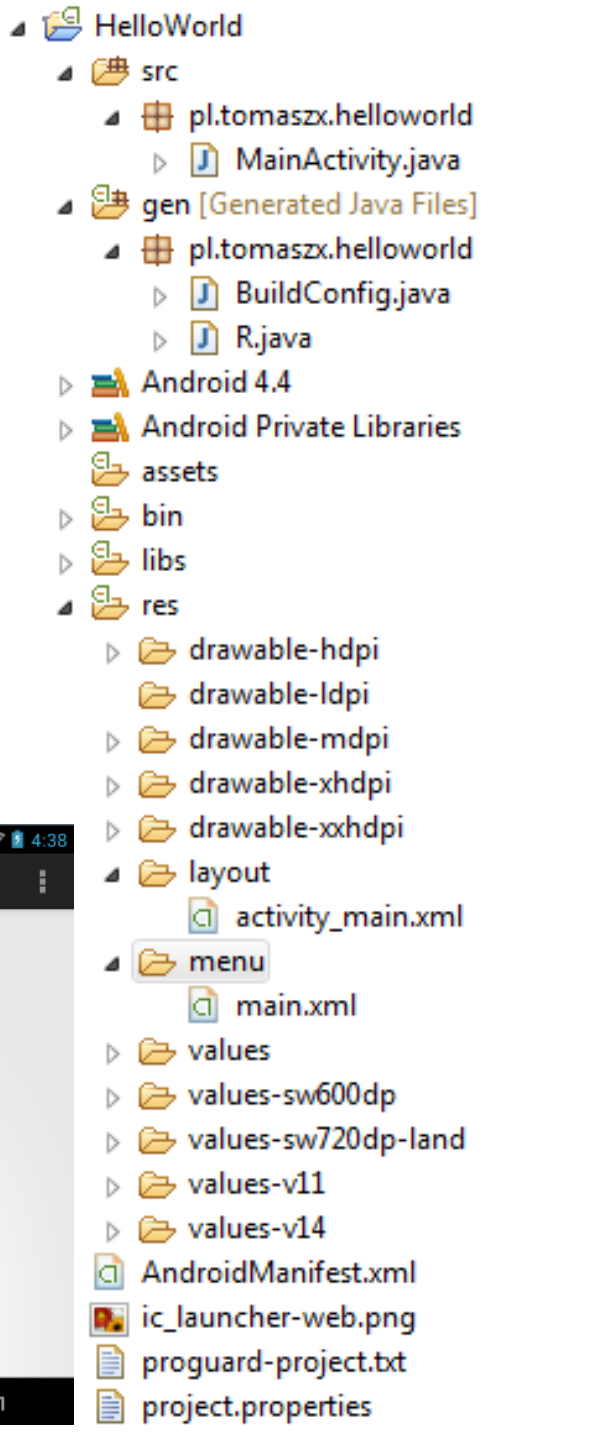
- Dalvik executable file + resources = APK
- Applications must be signed
  - There is a debug key
- Multiple app stores: Google Play, Amazon AppStore, GetJar, AppBrain, F-Droid etc.



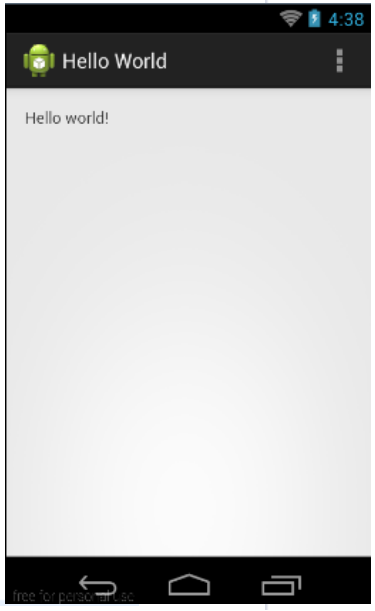



# New Project



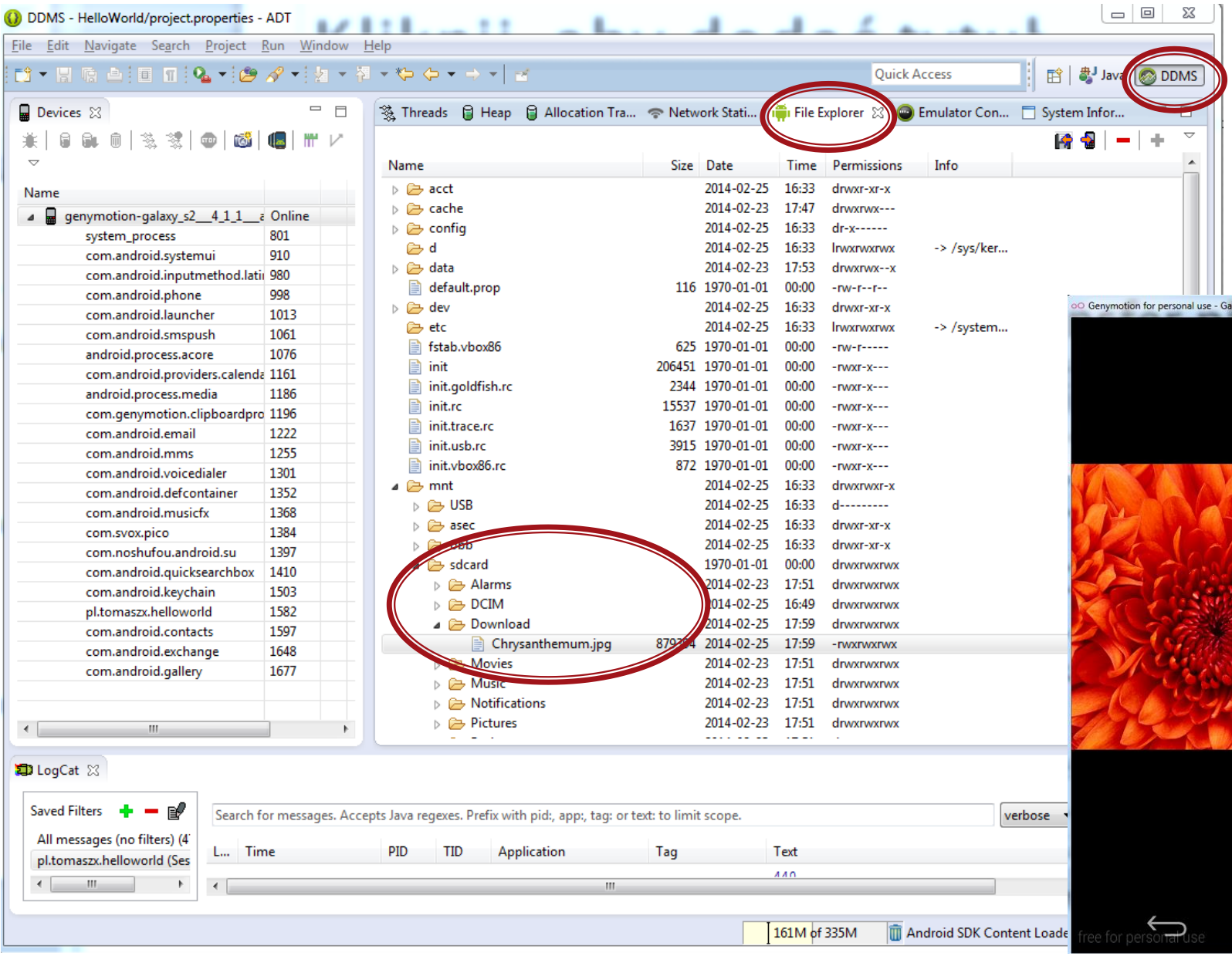

```java
package pl.tomaszx.helloworld;

import android.os.Bundle;

public class MainActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }

}
```



# Emulator keyboard shortcuts

| Emulated Device Key | Keyboard Key |
|---|---|
| Home | HOME |
| Menu (left softkey) | F2 *or* Page-up button |
| Star (right softkey) | Shift-F2 *or* Page Down |
| Back | ESC |
| Call/dial button | F3 |
| Hangup/end call button | F4 |
| Search | F5 |
| Power button | F7 |
| Audio volume up button | KEYPAD_PLUS, Ctrl-F5 |
| Audio volume down button | KEYPAD_MINUS, Ctrl-F6 |
| Camera button | Ctrl-KEYPAD_5, Ctrl-F3 |
| Switch to previous layout orientation (for example, portrait, landscape) | KEYPAD_7, Ctrl-F11 |
| Switch to next layout orientation (for example, portrait, landscape) | KEYPAD_9, Ctrl-F12 |
| Toggle cell networking on/off | F8 |
| Toggle code profiling | F9 (only with `-trace` startup option) |
| Toggle fullscreen mode | Alt-Enter |
| Toggle trackball mode | F6 |
| Enter trackball mode temporarily (while key is pressed) | Delete |
| DPad left/up/right/down | KEYPAD_4/8/6/2 |
| DPad center click | KEYPAD_5 |
| Onion alpha increase/decrease | KEYPAD_MULTIPLY(*) / KEYPAD_DIVIDE(/) |

# File transfer



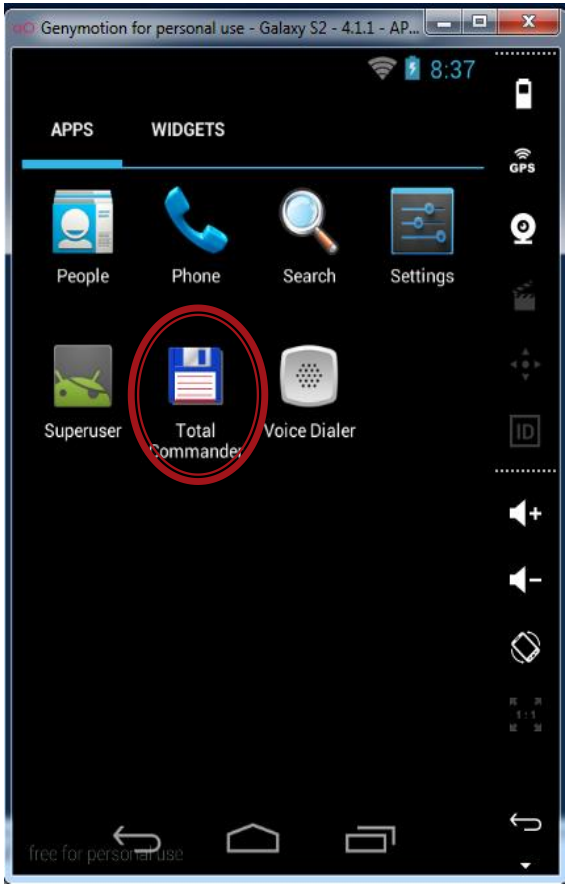
# Access to the system shell

adb shell

```
PS D:\adt-bundle-windows-x86_64-20131030\sdk\platform-tools> .\adb.exe shell
root@android:/ # ls -al
ls -al
drwxr-xr-x root     root              2014-02-25 20:19 acct
drwxrwx--- system   cache             2014-02-23 17:47 cache
dr-x------ root     root              2014-02-25 20:19 config
lrwxrwxrwx root     root              2014-02-25 20:19 d -> /sys/kernel/debug
drwxrwx--x system   system            2014-02-23 17:53 data
-rw-r--r-- root     root          116 1970-01-01 00:00 default.prop
drwxr-xr-x root     root              2014-02-25 20:19 dev
lrwxrwxrwx root     root              2014-02-25 20:19 etc -> /system/etc
-rw-r----- root     root          625 1970-01-01 00:00 fstab.vbox86
-rwxr-x--- root     root       206451 1970-01-01 00:00 init
-rwxr-x--- root     root         2344 1970-01-01 00:00 init.goldfish.rc
-rwxr-x--- root     root        15537 1970-01-01 00:00 init.rc
-rwxr-x--- root     root         1637 1970-01-01 00:00 init.trace.rc
-rwxr-x--- root     root         3915 1970-01-01 00:00 init.usb.rc
-rwxr-x--- root     root          872 1970-01-01 00:00 init.vbox86.rc
drwxrwxr-x root     system            2014-02-25 20:19 mnt
dr-xr-xr-x root     root              2014-02-25 20:19 proc
drwx------ root     root              2012-10-01 14:14 root
drwxr-x--- root     root              1970-01-01 00:00 sbin
lrwxrwxrwx root     root              2014-02-25 20:19 sdcard -> /mnt/sdcard
dr-xr-xr-x root     root              2014-02-25 20:19 sys
drwxr-xr-x root     root              1970-01-01 00:00 system
-rw-r--r-- root     root          272 1970-01-01 00:00 ueventd.goldfish.rc
-rw-r--r-- root     root         3879 1970-01-01 00:00 ueventd.rc
-rw-r--r-- root     root           30 1970-01-01 00:00 ueventd.vbox86.rc
lrwxrwxrwx root     root              2014-02-25 20:19 vendor -> /system/vendor
root@android:/ #
```

# ADB capabilities

```
PS D:\adt-bundle-windows-x86_64-20131030\sdk\platform-tools> .\adb.exe -h
Android Debug Bridge version 1.0.31

 -a                            - directs adb to listen on all interfaces for a connection
 -d                            - directs command to the only connected USB device
                                 returns an error if more than one USB device is present.
 -e                            - directs command to the only running emulator.
                                 returns an error if more than one emulator is running.
 -s <specific device>          - directs command to the device or emulator with the given
                                 serial number or qualifier. Overrides ANDROID_SERIAL
                                 environment variable.
 -p <product name or path>     - simple product name like 'sooner', or
                                 a relative/absolute path to a product
                                 out directory like 'out/target/product/sooner'.
                                 If -p is not specified, the ANDROID_PRODUCT_OUT
                                 environment variable is used, which must
                                 be an absolute path.
 -H                            - Name of adb server host (default: localhost)
 -P                            - Port of adb server (default: 5037)
 devices [-l]                  - list all connected devices
                                 ('-l' will also list device qualifiers)
 connect <host>[:<port>]       - connect to a device via TCP/IP
                                 Port 5555 is used by default if no port number is specified.
 disconnect [<host>[:<port>]]  - disconnect from a TCP/IP device.
                                 Port 5555 is used by default if no port number is specified.
                                 Using this command with no additional arguments
                                 will disconnect from all connected TCP/IP devices.

device commands:
  adb push <local> <remote>    - copy file/dir to device
  adb pull <remote> [<local>]  - copy file/dir from device
  adb sync [ <directory> ]     - copy host->device only if changed
                                 (-l means list but don't copy)
                                 (see 'adb help all')
  adb shell                    - run remote shell interactively
  adb shell <command>          - run remote shell command
  adb emu <command>            - run emulator console command
  adb logcat [ <filter-spec> ] - View device log
```



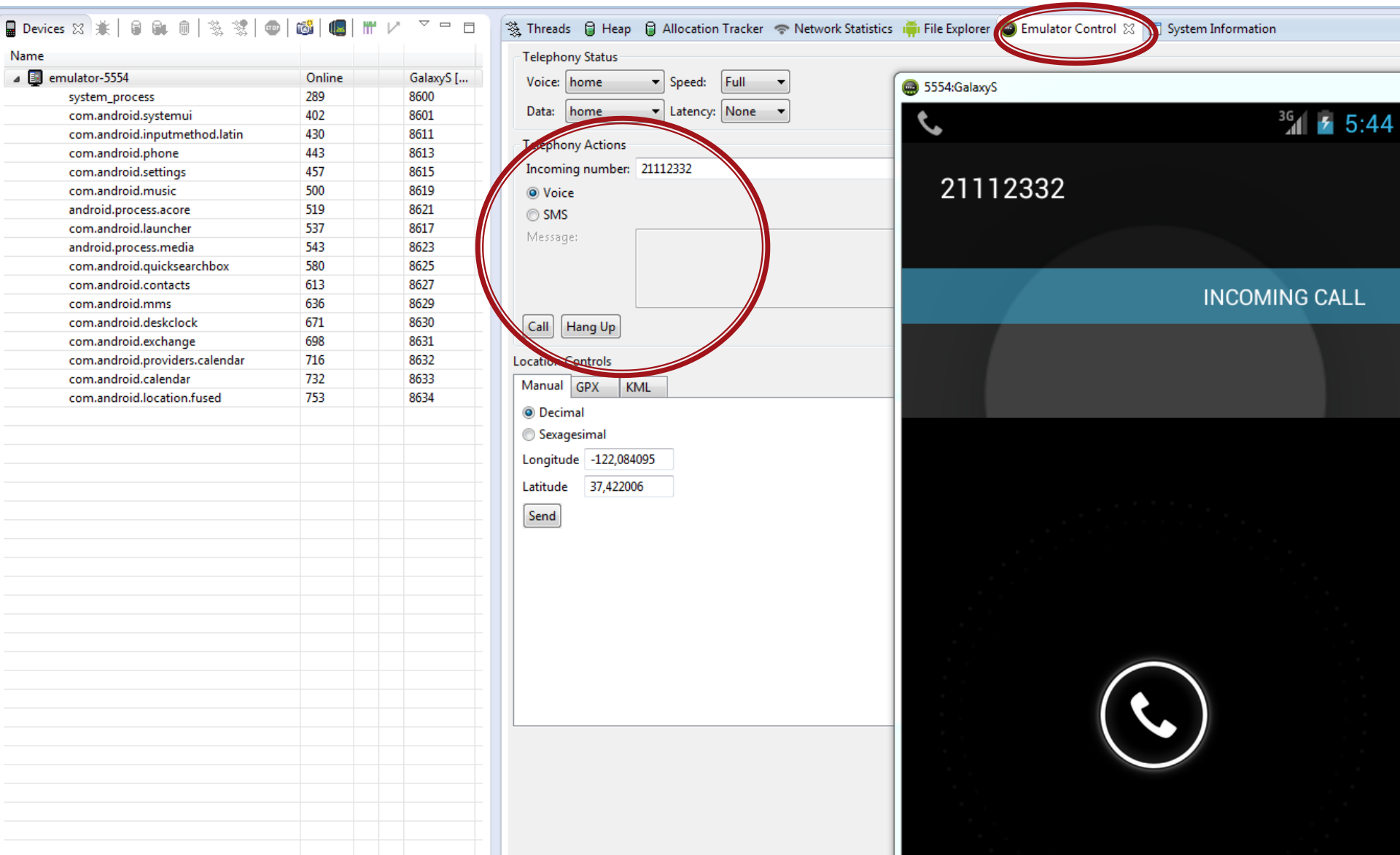

```
PS D:\adt-bundle-windows-x86_64-20131030\sdk\platform-tools> .\adb.exe devices
List of devices attached
192.168.56.101:5555     device

PS D:\adt-bundle-windows-x86_64-20131030\sdk\platform-tools> .\adb.exe pull data/app/ApiDemos.apk c:/ApiDemos.apk
5056 KB/s (3050203 bytes in 0.589s)
PS D:\adt-bundle-windows-x86_64-20131030\sdk\platform-tools>    .\adb.exe uninstall data/app/ApiDemos.apk
Failure
PS D:\adt-bundle-windows-x86_64-20131030\sdk\platform-tools>    .\adb.exe install C:\tcandroid204.apk
5988 KB/s (1202118 bytes in 0.196s)
        pkg: /data/local/tmp/tcandroid204.apk
Success
```

# Sending SMS / Making Calls - Emulator

- `telnet localhost 5554`
- `sms send <number> <text>`
- `gsm call <number>`

```
Android console command help:

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

try 'help <command>' for command-specific help
OK
sms send 12345678 sms testowy
OK
```

# SMS / Connections from GUI



# Components of the application

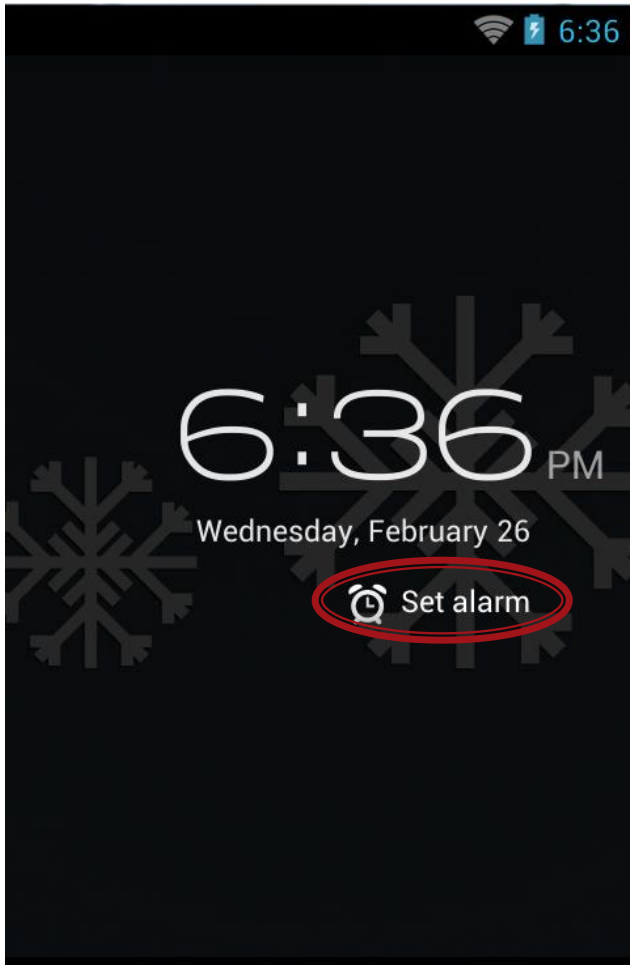
- The Android app consists of one or more components.

- Such an element can be:

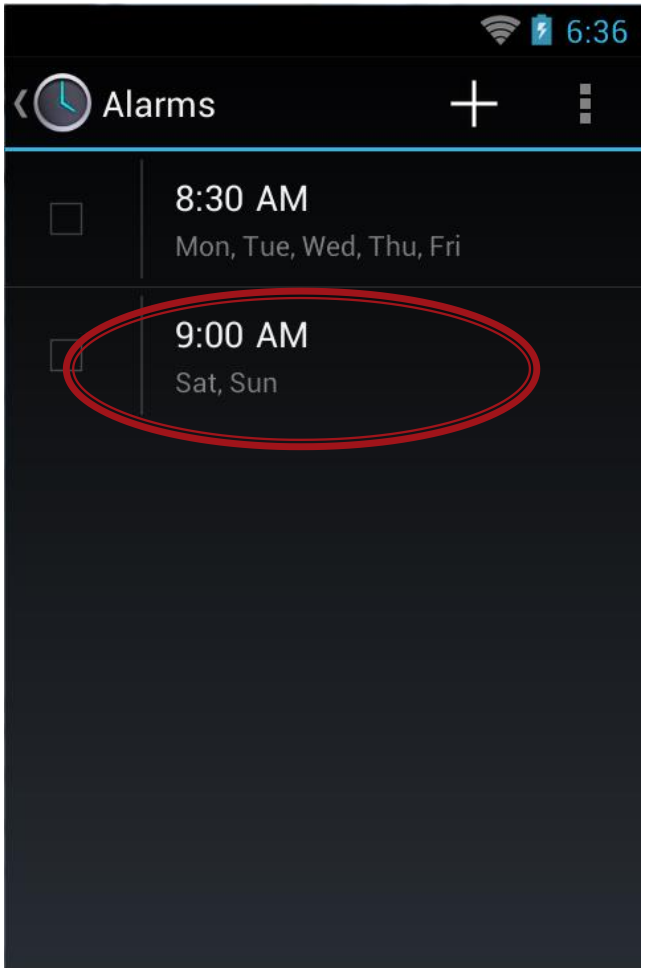
1. **Activity**
2. **Service**
3. **Broadcast receiver**
4. **Content provider**

# Example of three activities

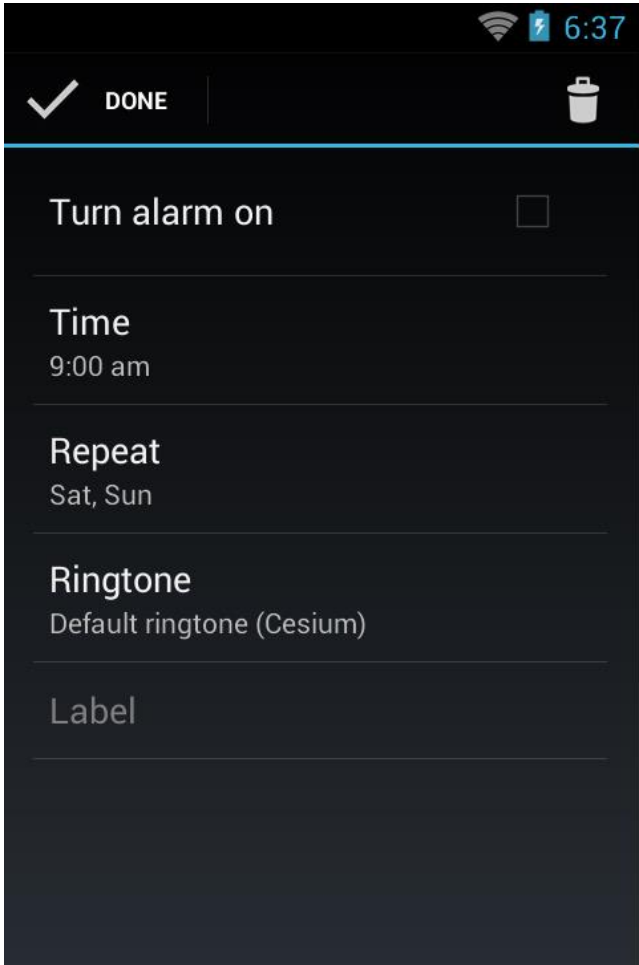
Activity 1




Activity 2




Activity 3



# Service

- A service is a component that runs in the background to perform long-running operations or to perform work for remote processes.
- A service does not provide a user interface.
- For example, a service might play music in the background while the user is in a different app, or it might fetch data over the network without blocking user interaction with an activity.
- Another component, such as an activity, can start the service and let it run or bind to it in order to interact with it. A service is implemented as a subclass of Service

# Broadcast receiver

- A broadcast receiver is a component that responds to system-wide broadcast announcements.
- Many broadcasts originate from the system—for example, a broadcast announcing that the screen has turned off, the battery is low, or a picture was captured.
- Apps can also initiate broadcasts—for example, to let other apps know that some data has been downloaded to the device and is available for them to use.
- Although broadcast receivers don't display a user interface, they may create a status bar notification to alert the user when a broadcast event occurs.

# Content provider

- A Content Provider is a wrapper that hides the actual physical data. Users interact with their data through a common object interface.
- A content provider manages a shared set of app data. You can store the data in the file system, an SQLite database, on the web, or any other persistent storage location your app can access. Through the content provider, other apps can query or even modify the data (if the content provider allows it).
- For example, the Android system provides a content provider that manages the user's contact information. As such, any app with the proper permissions can query part of the content provider (such as ContactsContract.Data) to read and write information about a particular person.

# Activity Stack

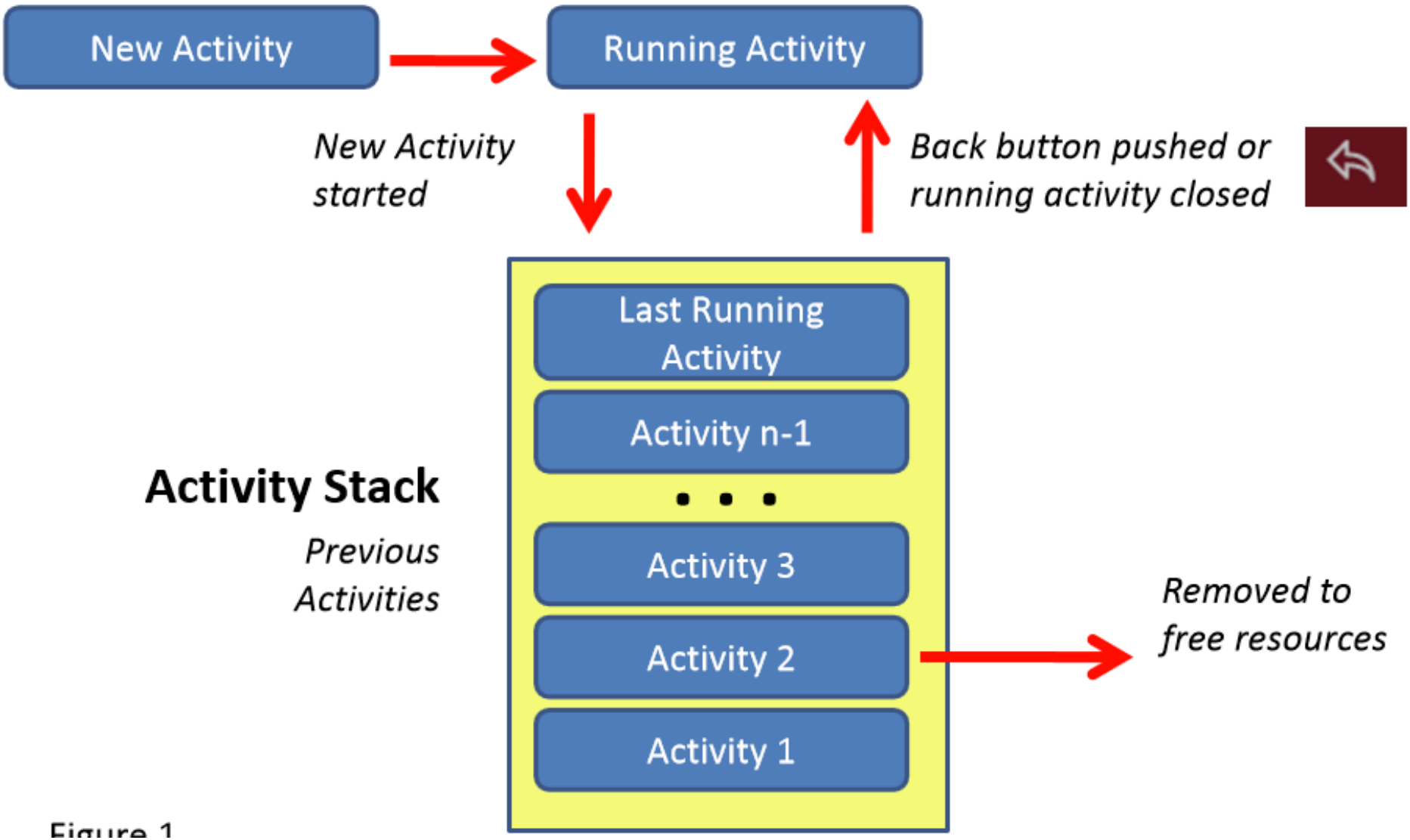
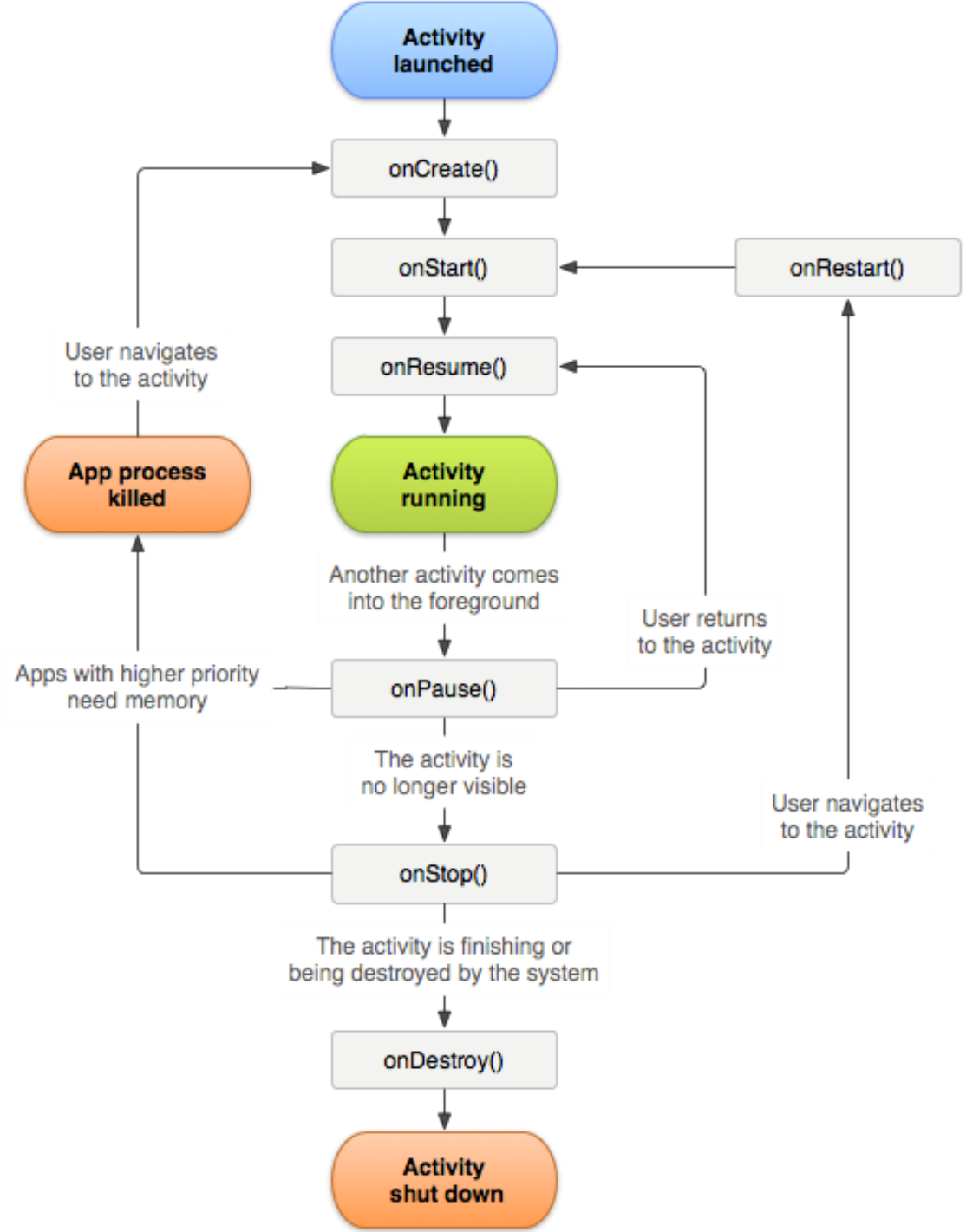



Figure 1

http://wwwsimpleandroid.blogspot.com/2013/05/activity-stack-in-android.html

# Life cycle of an activity



# Callbacks

```
public class ExampleActivity extends Activity {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        // It is created
    }
    @Override
    protected void onStart() {
        super.onStart();
        // It is visible (eg. partially)
    }
    @Override
    protected void onResume() {
        super.onResume();
        // Activity is visible (in foreground)
    }
    @Override
    protected void onPause() {
        super.onPause();
        // Other activity has focus
    }
    @Override
    protected void onStop() {
        super.onStop();
        // Activity is no longer visible
    }
    @Override
    protected void onDestroy() {
        super.onDestroy();
        // The activity will be deleted
    }
}
```

# Life Cycle of an Activity - An Example

```
package pl.tomaszx.helloworld;

import android.os.Bundle;..

public class MainActivity extends Activity {
    private Context context;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        Button btnExit = (Button) findViewById(R.id.button1);

        btnExit.setOnClickListener(new OnClickListener() {
            @Override
            public void onClick(View v) {
                finish();
            }
        });

        context = getApplicationContext();
        Toast.makeText(context, "onCreate", Toast.LENGTH_SHORT).show();
    }
```

# Life Cycle of an Activity - An Example

```
@Override
protected void onDestroy() {
    super.onDestroy();
    Toast.makeText(context, "onDestroy", Toast.LENGTH_SHORT).show();
}

@Override
protected void onPause() {
    super.onPause();
    Toast.makeText(context, "onPause", Toast.LENGTH_SHORT).show();

}

@Override
protected void onRestart() {
    super.onRestart();
    Toast.makeText(context, "onRestart", Toast.LENGTH_SHORT).show();
}

@Override
protected void onResume() {
    super.onResume();
    Toast.makeText(context, "onResume", Toast.LENGTH_SHORT).show();
}
```

# Life Cycle of an Activity - An Example

```
    @Override
    protected void onStart() {
        super.onStart();
        Toast.makeText(context, "onStart", Toast.LENGTH_SHORT).show();
    }

    @Override
    protected void onStop() {
        super.onStop();
        Toast.makeText(context, "onStop", Toast.LENGTH_SHORT).show();
    }
}
```