

Introduction to Android™ Development

Philippe Leefsma
Senior Developer Consultant
Autodesk Developer Network

Class summary

Learn how to start programming for Android-based devices.

This class presents how to set up an Android development environment and create from scratch a simple application with basic UI elements. We will then deploy the App on a device and show how you can debug it.

We will also illustrate how to consume web services, present a couple of more advanced APIs and take a look at some popular frameworks among Android developers.

Key learning objectives

At the end of this class, you will be able to:

- Understand the basic components of the Android SDK
- Start developing on Android devices
- Create Apps with simple UI
- Consume web services from Android
- Get a taste of some more advanced API's and Fmx

About the Presenter

Philippe Leefsma



*Autodesk Developer Technical Services
EMEA (Neuchatel, Switzerland)*

Philippe has a master's degree in Computer Sciences. He carried his studies in Paris at I.S.E.P and in USA, at Colorado School of Mines.

He joined Autodesk 7 years ago where he works as developer consultant for Autodesk Developer Network. He supports various products APIs such as AutoCAD® and Autodesk® Inventor®. Since couple of years he's also investigating various technologies around Cloud & Mobile area and share his knowledge through a devblog:

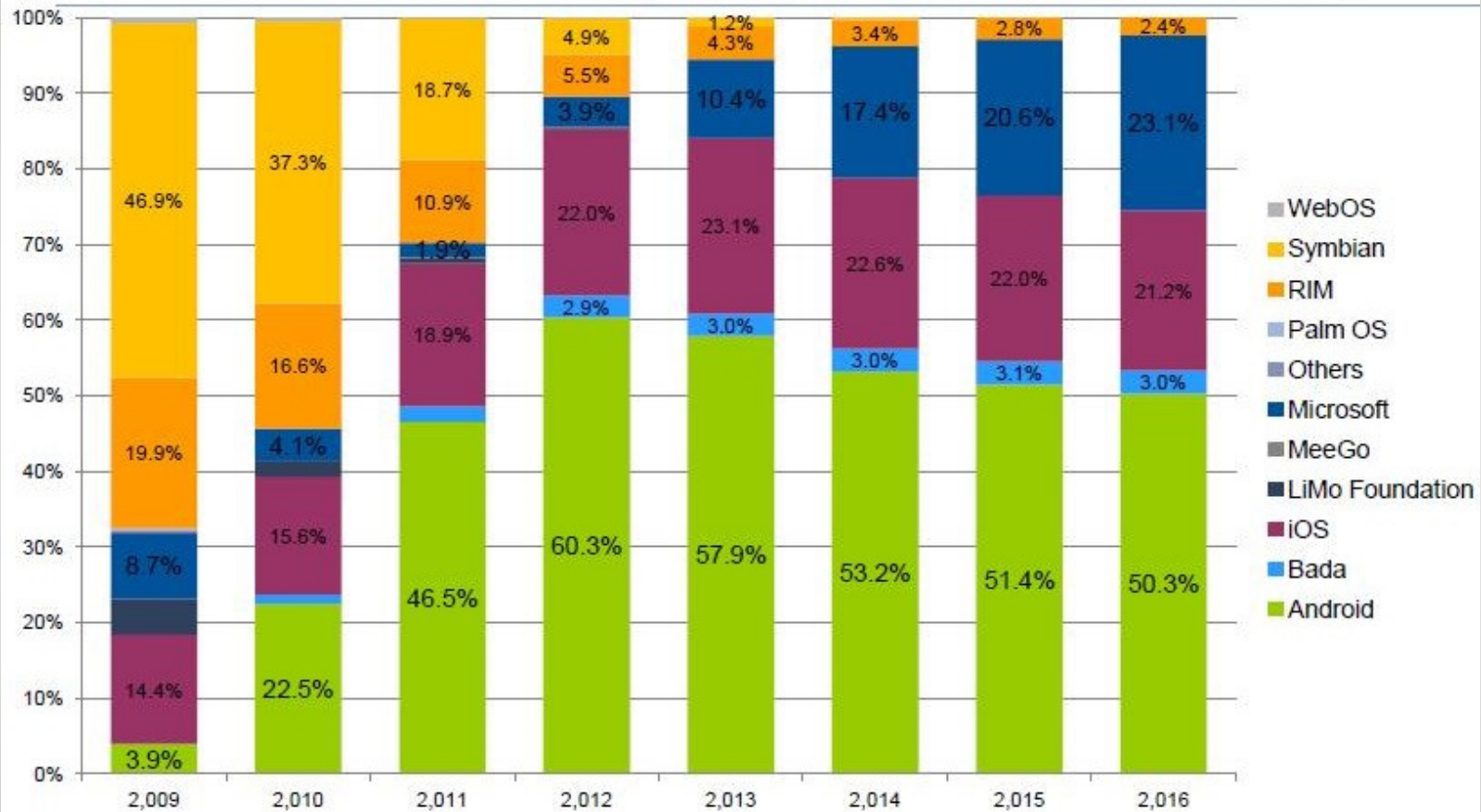
http://adndevblog.typepad.com/cloud_and_mobile

I - The Android Operating System

A bit of history...

- **2003** - Android, Inc. is founded by Andy Rubin, Rich Miner, Nick Sears and Chris White with the goal of developing smarter mobile devices
- **2005** - Google seeing a large growth of Internet use and search on mobile devices, acquires Android Inc.
- **2007** - Apple introduces the iPhone with some ground-breaking ideas:
 - *Multi-touch screen, Open market for applications*
 - *Android is quickly adapted to include these features & other distinctions*
- **2008** - First Android-powered phone (*HTC Dream T-Mobile G1*)
- **2010** - Android becomes world's leading smartphone platform

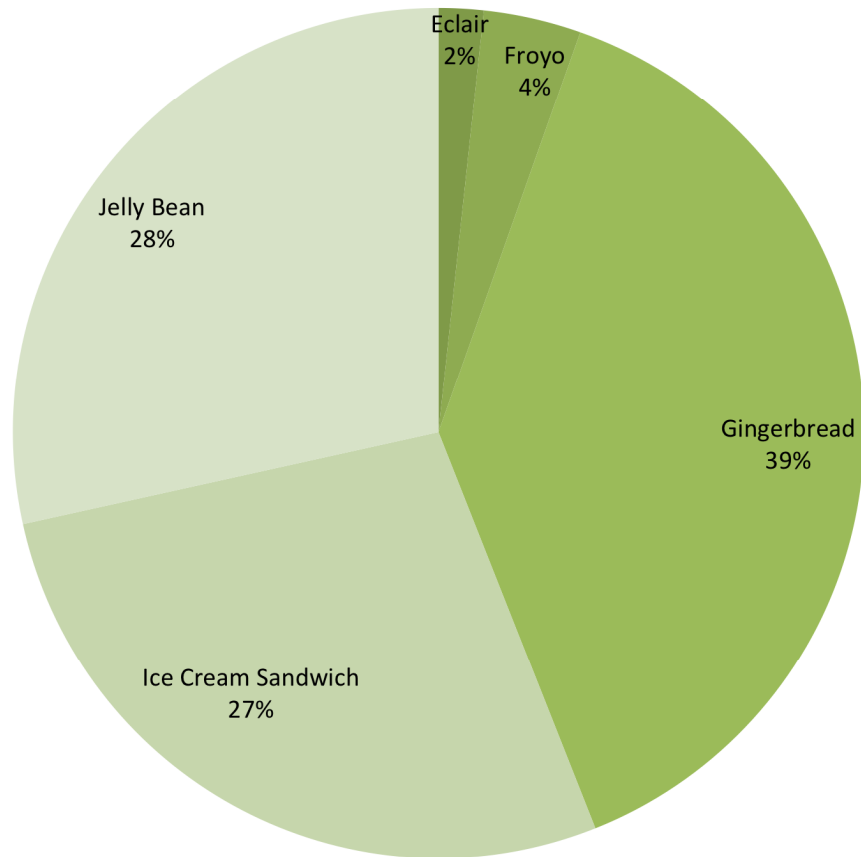
Mobile OS Sales by Market Share (2009-2016)



Source: Gartner

Android Platform Distribution

(April 2013)



| Version | Codename | API |
|---------------|--------------------|-----|
| 1.6 | Donut | 4 |
| 2.1 | Eclair | 7 |
| 2.2 | Froyo | 8 |
| 2.3.3 - 2.3.7 | Gingerbread | 10 |
| 3.2 | Honeycomb | 13 |
| 4.0.3 - 4.0.4 | Ice Cream Sandwich | 15 |
| 4.1.x | Jelly Bean | 16 |
| 4.2.x | Jelly Bean | 17 |

Source: developer.android.com

What Android is and isn't...

- An embedded OS that relies on Linux kernel for core services but it is NOT embedded Linux
- Standard Linux utilities (X-windows, GNU C libraries) NOT supported
- Applications use Java framework but some standard libs such as Swing are not supported
- Several libs replaced by Android libs > optimized for resource-constrained embedded environment

Android Images

The Android OS is organized into the following images:

- **Boot image**
Kernel and RAMdisk
- **Bootloader**
Initiates loading of the boot image during startup
- **System image**
Android operating system and apps
- **Data image**
User data saved across power cycles
- **Recovery image**
Files used for rebuilding or updating the system
- **Radio image**
Files of the radio stack

Main Android Application Components

| Functionality | Android Base Class | Example |
|--|---------------------------------|---------------------------------|
| Focused thing user can do | <i>Activity</i> | Edit note, play game |
| Background process | <i>Service</i> | Play music, update weather icon |
| Receive messages | <i>BroadcastReceiver</i> | Trigger alarm upon event |
| Connect data & code in different processes | <i>ContentProvider</i> | Accessing SQLite database |

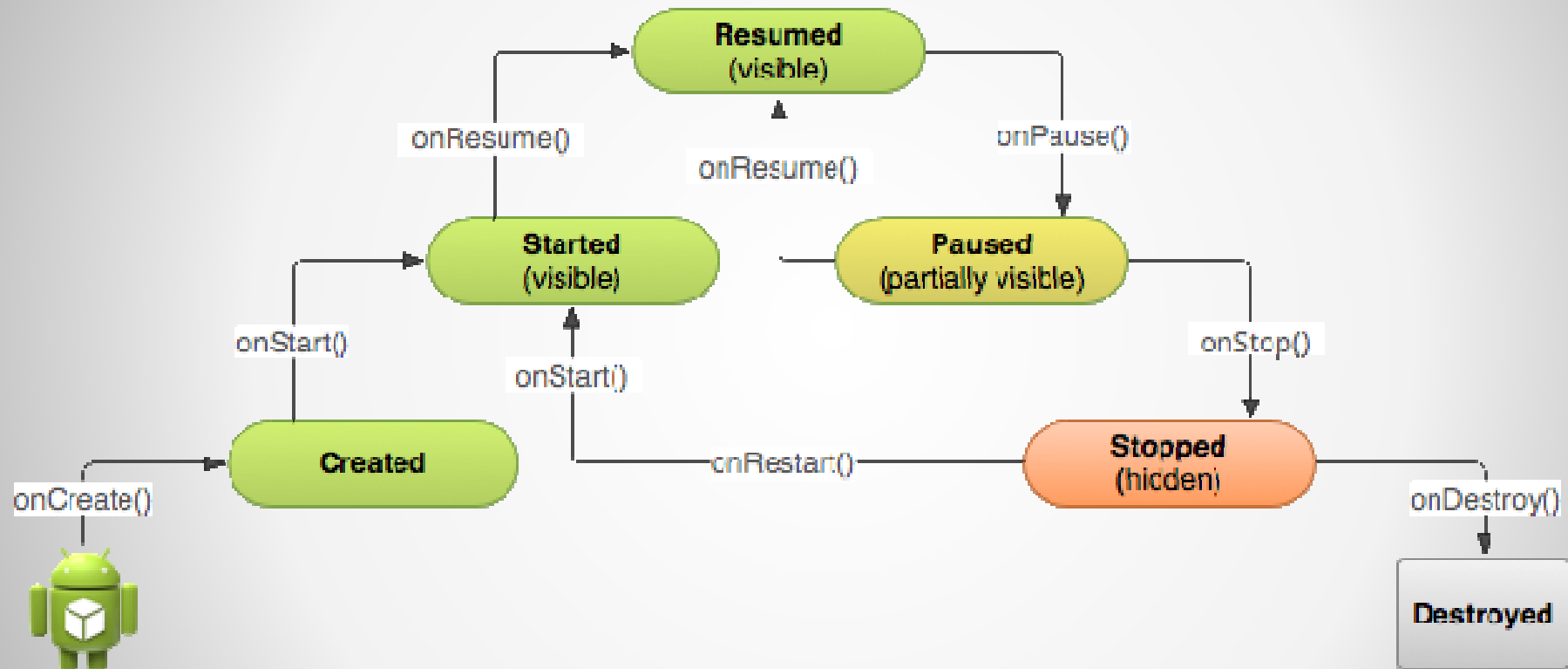
Android Activity

```
public class MyActivity extends Activity
{
    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);

        // Do some stuff..
    }

    public void onStart() ...
    public void onRestart() ...
    public void onResume() ...
    public void onPause() ...
    public void onStop() ...
    public void onDestroy() ...
}
```

Android Activity Lifecycle



II - Getting Started with Android Programming

Setting up your Programming Environment

- Install Eclipse

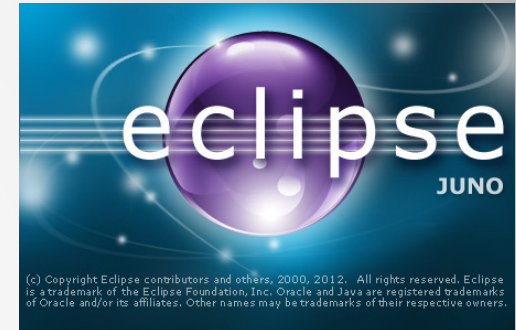
<http://www.eclipse.org/downloads>

- Install Android SDK

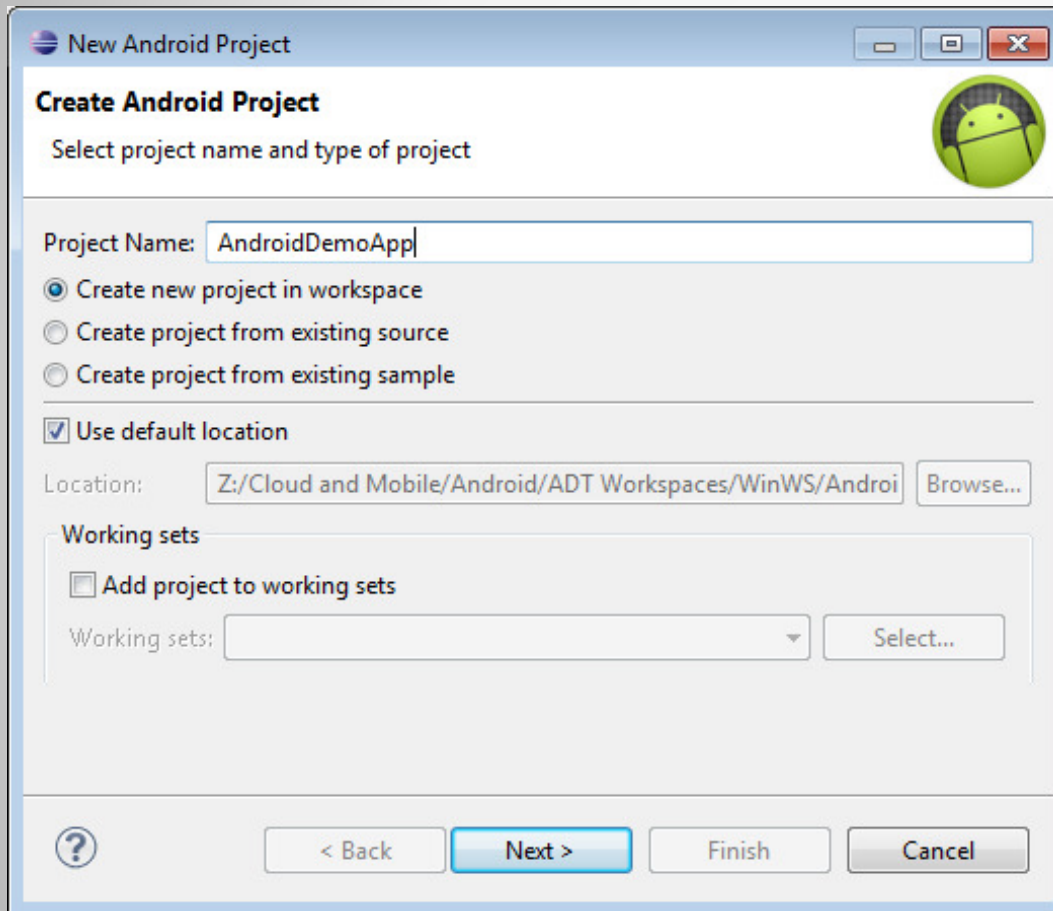
<http://developer.android.com/sdk/installing.html>

- Get started with Android development

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



My first Android App using Eclipse



The screenshot shows the 'New Android Project' dialog box in the Eclipse IDE. The title bar reads 'New Android Project'. The main heading is 'Create Android Project', followed by the instruction 'Select project name and type of project'. An Android robot icon is visible in the top right corner. The 'Project Name' field contains 'AndroidDemoApp'. There are three radio button options: 'Create new project in workspace' (selected), 'Create project from existing source', and 'Create project from existing sample'. A checked checkbox labeled 'Use default location' is present. The 'Location' field shows the path 'Z:/Cloud and Mobile/Android/ADT Workspaces/WinWS/Androi', with a 'Browse...' button next to it. A section titled 'Working sets' contains a checkbox 'Add project to working sets' and a 'Working sets:' dropdown menu with a 'Select...' button. At the bottom, there is a help icon (?) and four buttons: '< Back', 'Next >' (highlighted in blue), 'Finish', and 'Cancel'.

New Android Project

Create Android Project

Select project name and type of project

Project Name:

☒ Create new project in workspace
☐ Create project from existing source
☐ Create project from existing sample

☒ Use default location

Location:

Working sets

☐ Add project to working sets

Working sets:

Android Project Structure

- AndroidManifest.xml
- src/
- res/
 - drawable-XXXX/
 - layout/
 - values/

Debugging your App

- Android Emulator
 - ARM / x86 GPU-Enabled
 - <http://developer.android.com/tools/devices/emulator.html>
- Connecting device with USB
 - Requires installation of device specific driver for Windows
 - Immediate on Mac OS (settings to be enabled on device)
- Virtualization solution
 - <http://www.android-x86.org>
 - Connect through adb: **Android Debug Bridge**

Android Studio

- New development environment from Google:
 - <http://developer.android.com/sdk/installing/studio.html>



III - Web Services & Other APIs

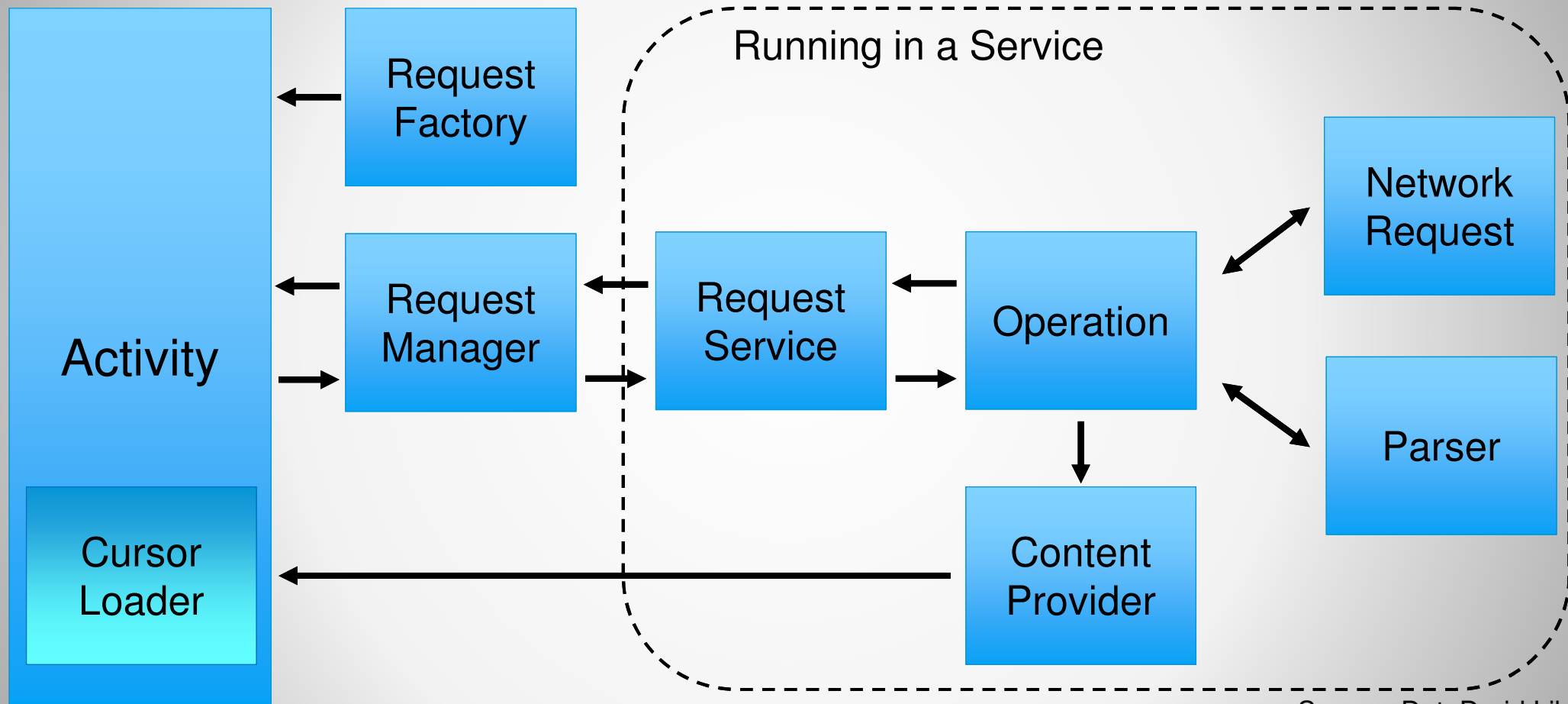
Performing REST requests on Android

- Native support through **org.apache.http** package:

```
DefaultHttpClient httpClient = new DefaultHttpClient();  
HttpGet request = new HttpGet(new URI(url));  
HttpResponse response = httpClient.execute(request);
```

- Web request **MUST** be performed from non-UI Thread:
 - Thread (non-UI), AsyncTask, Service
- Many 3rd party frameworks available:
 - Volley, DataDroid, RoboSpice, RESTProvider, PostmanLib, ...

Networking Request – Example Design Pattern



Source: DataDroid Lib

3D Graphics on Android

- OpenGL ES 1.0/1.1
 - javafx.microedition.khronos.org/engles
Standard implementation of OpenGL ES 1.0/1.1
 - android.opengl
Better performance
- OpenGL ES 2.0
 - android.opengl.GLES20
Available from Android 2.2 (API Level 8)
- OpenGL ES 3.0
 - android.opengl.GLES30
Available from Android 4.3 (API Level 18)

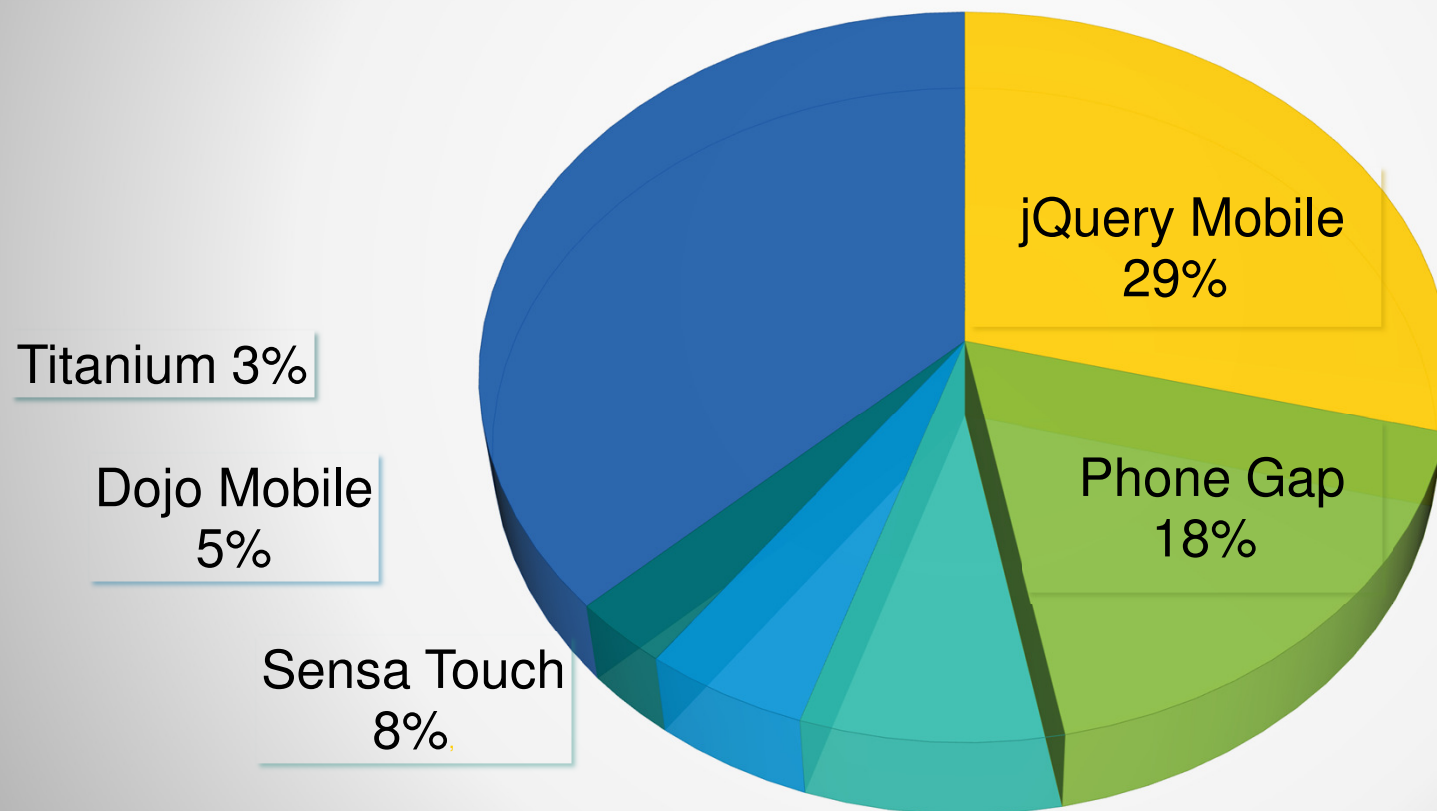
Graphics Engines on Android

- Lib GDX
 - <http://libgdx.badlogicgames.com>
- Rajawali
 - <https://github.com/MasDennis/Rajawali>
- Hoops Visualize
 - <http://developer.techsoft3d.com/hoops/hoops-visualize>
- And many more...
 - <http://software.intel.com/en-us/blogs/2012/03/13/game-engines-for-android>

Writing Android Apps with HTML5 & JavaScript



Popular HTML-based cross-platform frameworks



HTML5/JS vs Native Apps

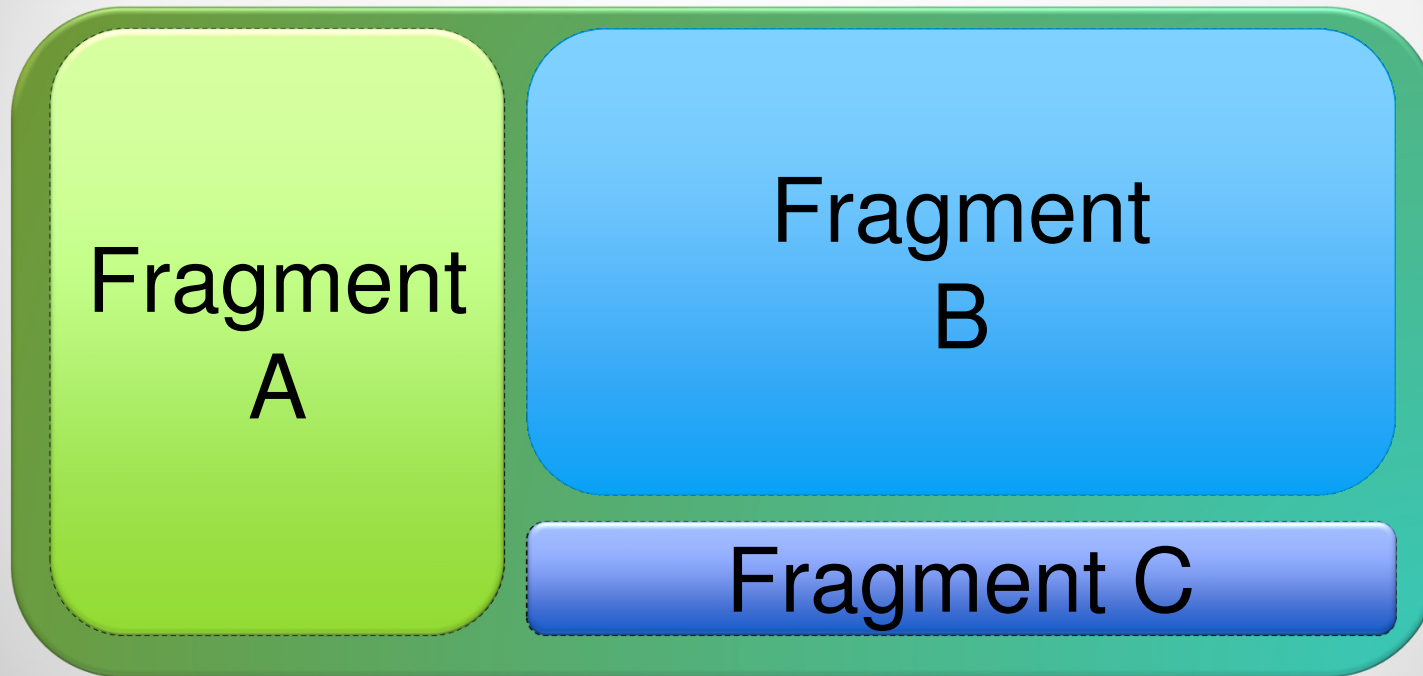
- HTML5/JS is a good choice if your team has web development skills
- If you want to use JS but also need native functionality, then use solutions like PhoneGap
- Implementing cross platform HTML5 solution can be just as expensive as creating multiple native apps
- Lots of available components create confusion about which ones to use
- Big apps are usually written in native language

Android NDK – Native Development Kit

- NDK is a toolkit that allows implementing parts of your app using native-code: C/C++
 - Reuse existing libraries
 - Potentially increases performances
- NDK will **NOT** benefit most apps
 - Native code on Android generally does not result in noticeable performance
 - Increases your app complexity
 - Do not use NDK because you simply prefer to program in C++
- Good candidates for the NDK
 - Self-contained, CPU-intensive operations that don't allocate much memory

Advanced UI features - Fragments

- A *Fragment* allows to insert portion of UI into an Activity



Wrap Up

Wrap up...

- We exposed the basic components of the Android SDK
- Created, deployed and debugged an App on a device
- Exposed basic UI components and workflows
- Explored how to handle networking
- Got a taste of some more advanced APIs

Android Programming considerations

- How does your App affect the battery life ?
 - Graphics, background threads, services, ...
- How much space needs the App on the device?
 - Check size of 3rd party SDKs, fetch data from server vs local, ...
- Is the App tolerant to bad network/mobile connection?
 - How does it behave on 3G/4G vs Wifi, ...
- How does the App look on different devices?
 - Size of the screen, orientation, CPU, memory...

Resources for Android developers

- Android Developer Center
<http://developer.android.com/index.html>
- Google Play - Get Started
<http://developer.android.com/distribute/index.html>
- Android Cookbook
<http://androidcookbook.com/home.seam>
- Blogs
[http://adndevblog.typepad.com/cloud and mobile](http://adndevblog.typepad.com/cloud_and_mobile)
<http://www.xda-developers.com>
<http://android-developers.blogspot.com>
<https://developers.google.com/events/io>
- Forums
<http://stackoverflow.com>
<http://www.codeproject.com/KB/android>

Presentation Links

Those are the links mentioned during the presentation:

- http://adndevblog.typepad.com/cloud_and_mobile/2012/06/android.html
- <http://software.intel.com/en-us/blogs/2012/03/13/game-engines-for-android>
- http://adndevblog.typepad.com/cloud_and_mobile/2012/12/giving-a-try-at-html5-canvas-and-javascript.html
- http://adndevblog.typepad.com/cloud_and_mobile/2013/08/writing-android-apps-with-html5-javascript.html
- <https://www.linux.com/news/embedded-mobile/mobile-linux/612366-15-android-ready-application-development-frameworks->
- <http://design.appmachine.com/home>
- http://adndevblog.typepad.com/cloud_and_mobile/2013/08/android-ndk-passing-complex-data-to-jni.html
- <https://www.elusivestars.com>
- <http://keynotopia.com>
- <http://acra.ch>
- <http://www.appbrain.com/stats/libraries/dev>

Q & A



