Mobile Application Development

Produced by

David Drohan (ddrohan@wit.ie)

Department of Computing & Mathematics
Waterford Institute of Technology
http://www.wit.ie

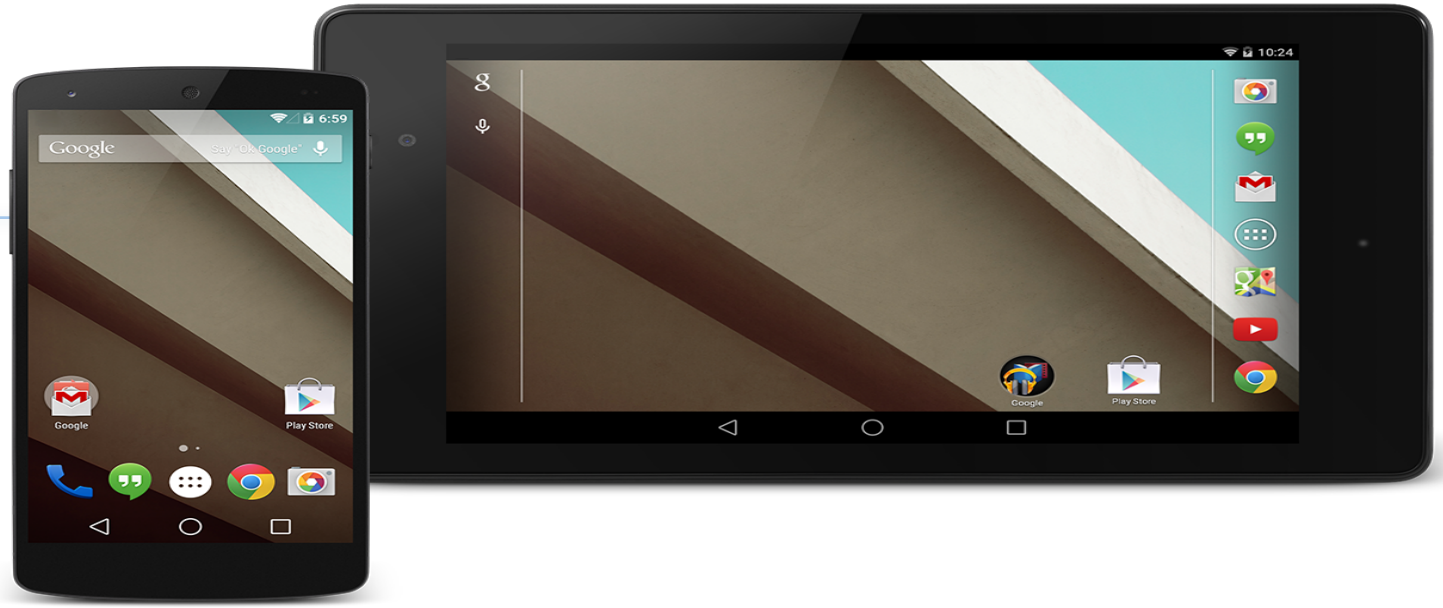
Waterford Institute *of* Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

Introducing Android

Agenda

- Background (and more)
- Quick tour of Android
- Setting up the Environment
- Android Apps vs iPhone Apps
- Books & References

Background (1)



- Android is a comprehensive open source platform designed for mobile devices.
- It is championed by Google and owned by Open Handset Alliance.
- The goal of the alliance is to “accelerate innovation in mobile and offer consumers a richer, less expensive, and better mobile experience.” (November 2007)

Background (2)

- Android, along with IOS, is revolutionising the mobile space.
- Unlike IOS, Android is an open platform that separates the hardware from the software that runs on it.
- This allows for a much larger number of devices to run the same applications and creates a much richer ecosystem for developers and consumers.

ANDROID WEAR

PHONES

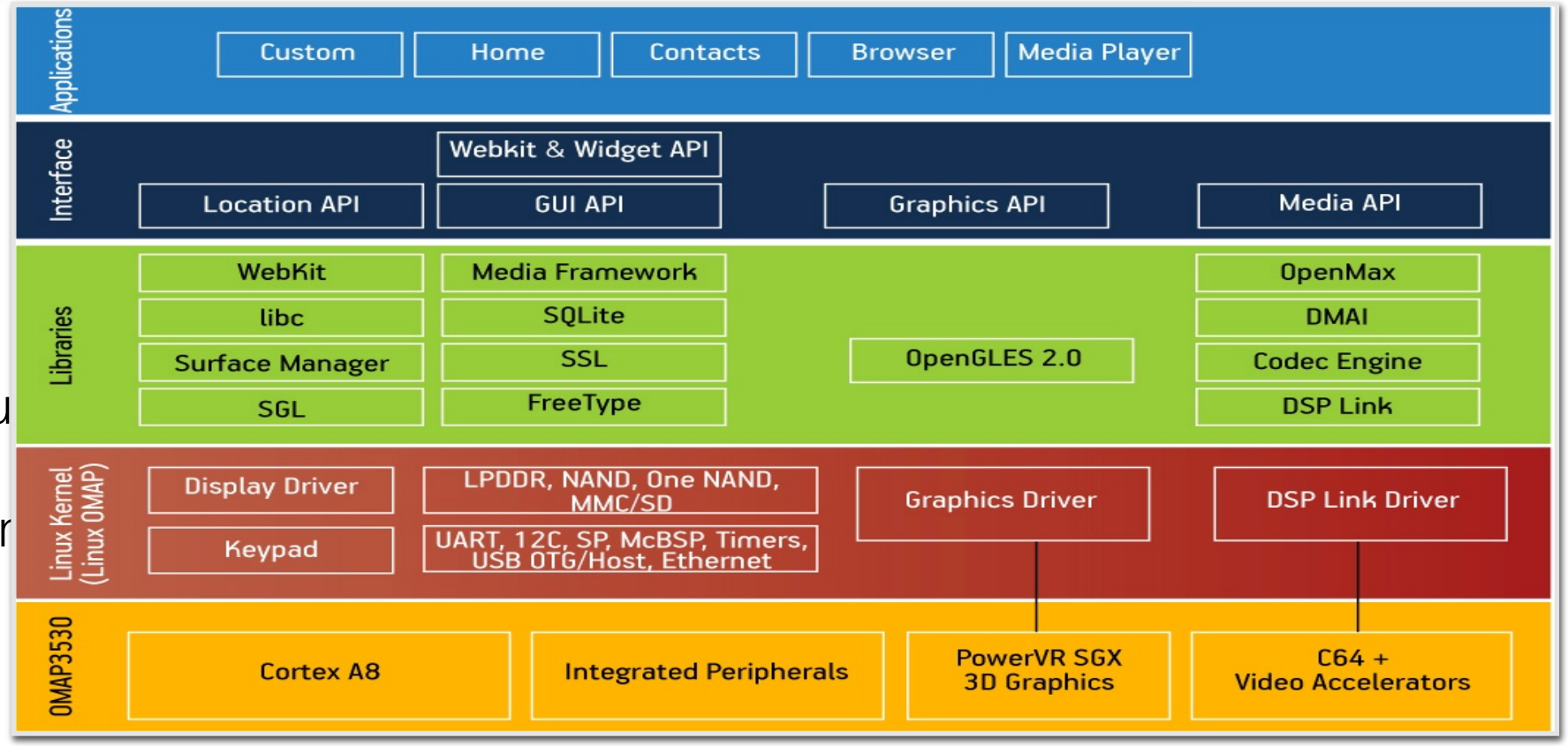
TABLETS

ANDROID TV

ANDROID AUTO

Comprehensive

- Android is a comprehensive platform, which means it is a complete software stack for a mobile device.
- The Android SDK is all you need to start developing for Android; you don't ever need a physical device.

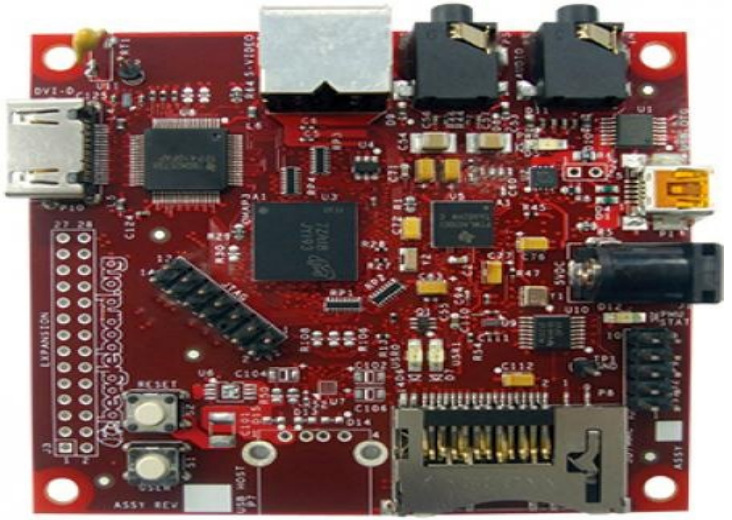


- Users can customize their phone experience substantially.
- Manufacturers can also customise the platform can substantial ways - even generating complete 'forks' of the original project (Amazon).

Open Source

- Android is an open source platform.
- Aside from the Linux kernel itself, Android is licensed under business-friendly licenses (Apache/ MIT/BSD) so that others can freely extend it and use it for variety of purposes.
- Manufacturers can port Android OS to specific hardware. with minimal legal issues.
- Android has many hooks at various levels of the platform, allowing anyone to extend it in unforeseen ways.

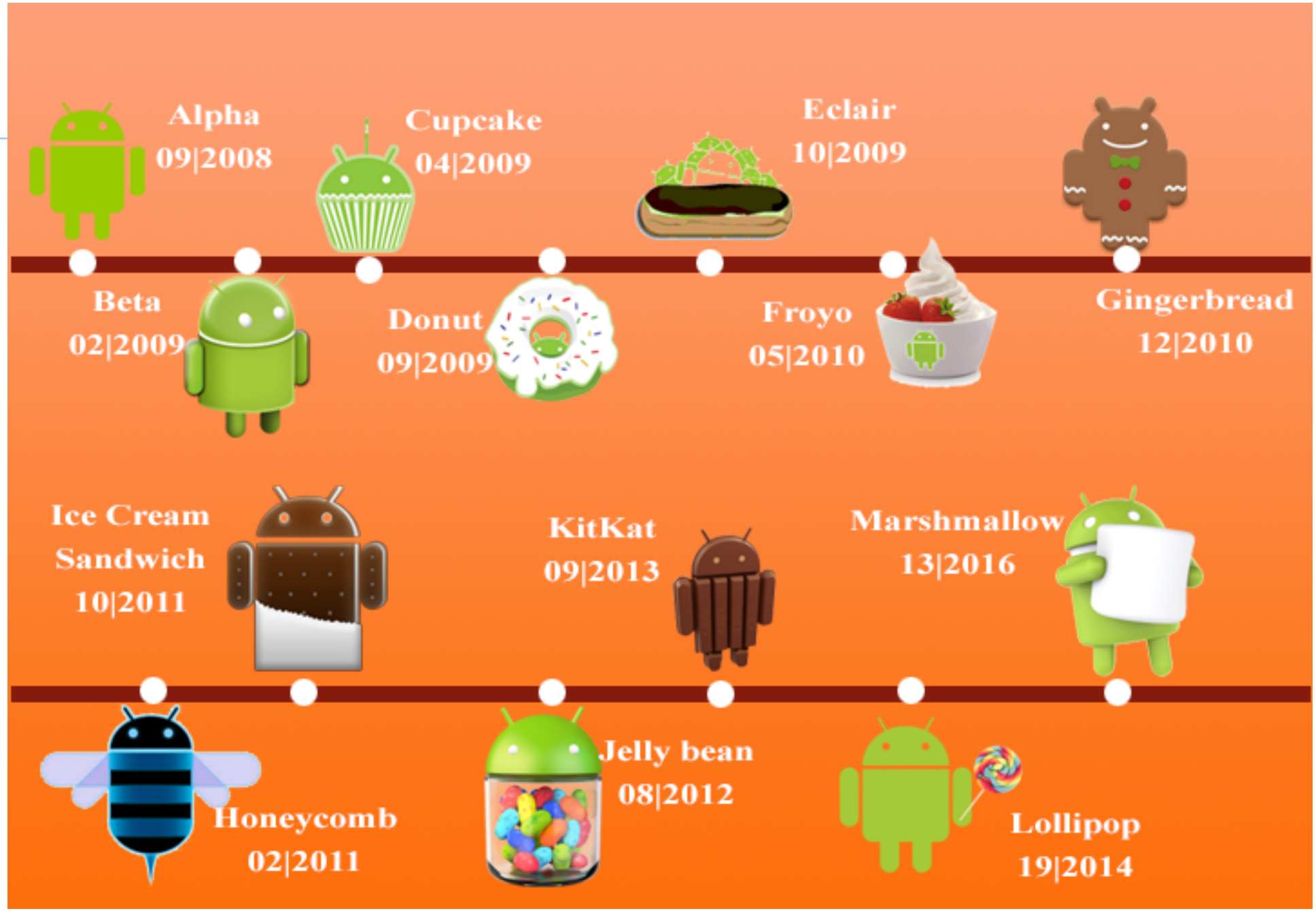
BeagleBoard, a low-power open-source hardware single-board computer produced by Texas Instruments



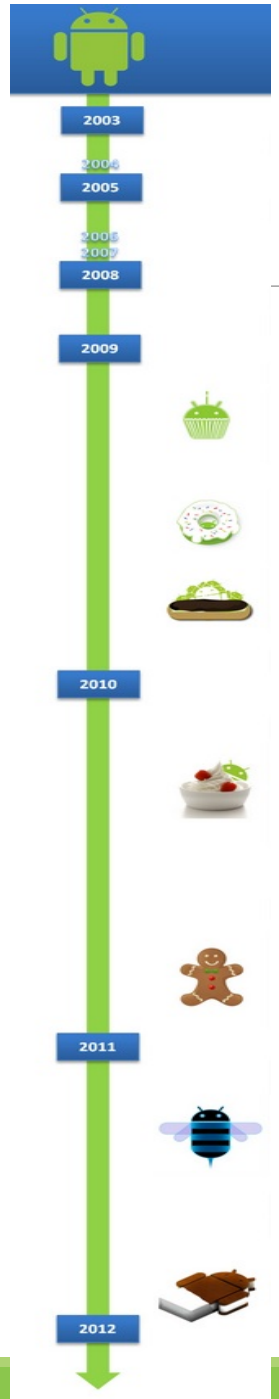
Designed for Mobile Devices

- When designing Android, the team looked at which mobile device constraints were likely not going to change for the foreseeable future:
  - *Battery* power, and *battery* performance is probably not going to get much better anytime soon.
  - In general, small *size* of mobile devices means that they will always be limited in terms of memory and speed.
  - However, a device's screen size, resolution, chipset may vary considerably
- These constraints have been taken into consideration throughout the platform.

Android History



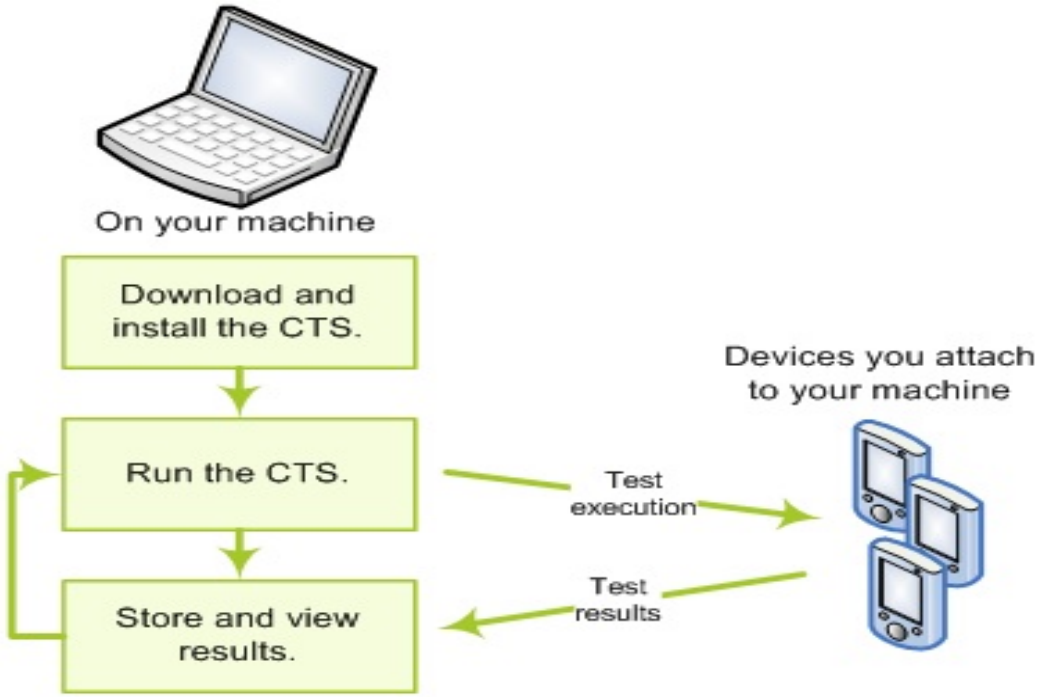
Android History



- In 2005, Google buys Android, Inc.
- In 2007, the Open Handset Alliance is announced. Android is officially open sourced.
- In 2008, the Android SDK 1.0 is released. The G1 phone, manufactured by HTC and sold by the wireless carrier T-Mobile USA, follows shortly afterward.
- 2009 sees a proliferation of Android-based devices. New versions of the operating system are released: Cupcake (1.5), Donut (1.6), and Eclair (2.0 and 2.1). More than 20 devices run Android.
- In 2010, Android is second only to BlackBerry as the best-selling smart phone platform. Froyo (Android 2.2) is released and so are more than 60 devices that run it.
- In 2011, Android is the #1 mobile platform by number of new activations and number of devices sold. The battle for dominating the tablet market is on.
- In 2012, GoogleTV, powered by Android and running on Intel x86 chips, is released. Android is now running on everything from the smallest of screens to the largest of TVs.
- In 2013, Google Glass, a wearable computing platform with an optical head- mounted display powered by Android is released to a select few.
- Beyond phones, tablets, and TVs, Android continues to be the big challenger to Embedded Linux as the platform for developing a number of specialized devices, such as home automation systems, car dashboards and navigation systems, as well as NASA satellites.

CTS

- The Compatibility Test Suite (CTS), defines what it means to be an Android-compatible device.
- CTS is a combination of automated tests as well as a document that specifies what an Android device must have, should have, or what features are simply optional.
- The goal of CTS is to ensure that, for a regular consumer, an average app from the market will run on an average Android device if that device claims to be supporting a certain version of Android.



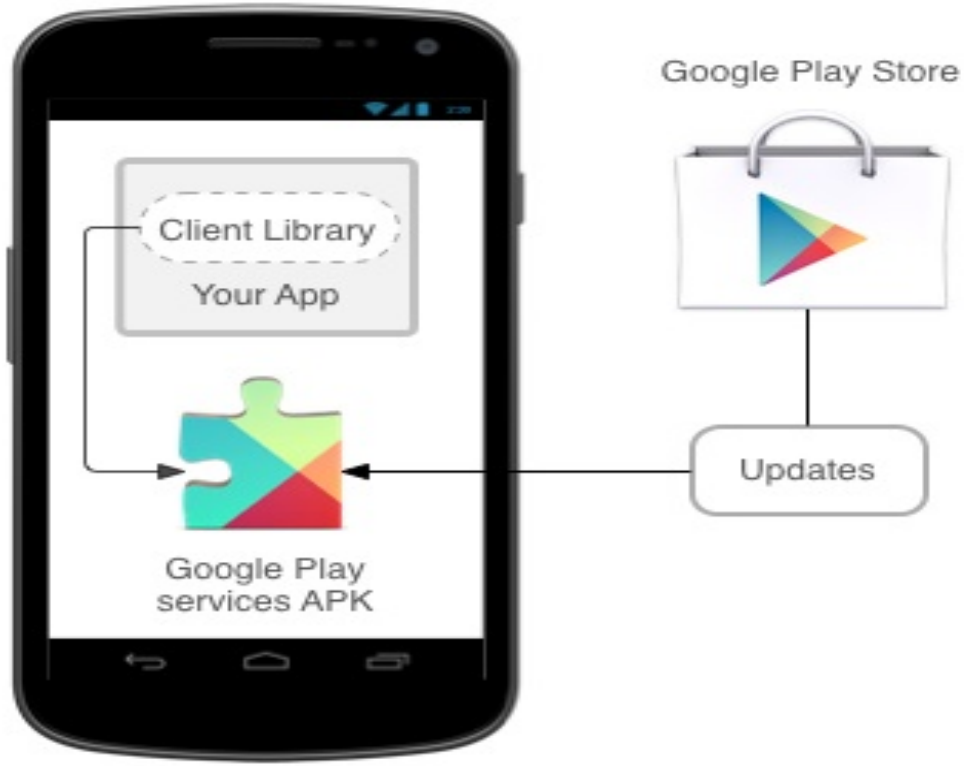
Compatibility

- CTS is being completely avoided by Amazon with the Kindle Fire and phone series of devices, built on top of the Android OS.
- Note that manufacturers by no means have to adhere to CTS.
- Anyone is welcome to download and “remix” Android in any way they see fit.
- Android has been customized for everything from cars to satellites, and from photocopiers to washing machines.

Area	Description
Signature tests	For each Android release, there are XML files describing the signatures of all public APIs contained in the release. The CTS contains a utility to check those API signatures against the APIs available on the device. The results from signature checking are recorded in the test result XML file.
Platform API Tests	Test the platform (core libraries and Android Application Framework) APIs as documented in the SDK Class Index to ensure API correctness, including correct class, attribute and method signatures, correct method behavior, and negative tests to ensure expected behavior for incorrect parameter handling.
Dalvik VM Tests	The tests focus on testing the Dalvik VM.
Platform Data Model	The CTS tests the core platform data model as exposed to application developers through content providers, as documented in the SDK android.provider package: contacts, browser, settings, etc.
Platform Intents	The CTS tests the core platform intents, as documented in the SDK Available Intents.
Platform Permissions	The CTS tests the core platform permissions, as documented in the SDK Available Permissions.
Platform Resources	The CTS tests for correct handling of the core platform resource types, as documented in the SDK Available Resource Types. This includes tests for: simple values, drawables, nine-patch, animations, layouts, styles and themes, and loading alternate resources.

Google Play Services

- The major reason manufacturers would want to ensure Android compatibility is access to Google Play, and its rich set of apps.
- Play services allow apps to take advantage of the latest, Google-powered features such as Maps, Google+, and more, with automatic platform updates distributed as an APK through the Google Play store.
- Makes it faster for phone to receive updates and easier for developers to integrate the some new features into their apps.



Versions – So Far

Cupcake
Android 1.5

Donut
Android 1.6

Eclair
Android 2.0/2.1

Froyo
Android 2.2.x

Gingerbread
Android 2.3.x

Honeycomb
Android 3.x

Ice Cream Sandwich
Android 4.0.x

Jelly Bean
Android 4.1.x

KitKat
Android 4.4.x

Lollipop
Android 5.0

Marshmallow
android 6.0

Nougat
android 7.0

Versions – API Level

- The Android version number itself partly tells the story of the software platform's major and minor releases. What is most important is the API level. Version numbers change all the time, sometimes because the APIs have changed, and other times because of minor bug fixes or performance improvements.
- As an application developer, you will want to make sure you know which API level your application is targeting in order to run. That API level will determine which devices can and cannot run your application.

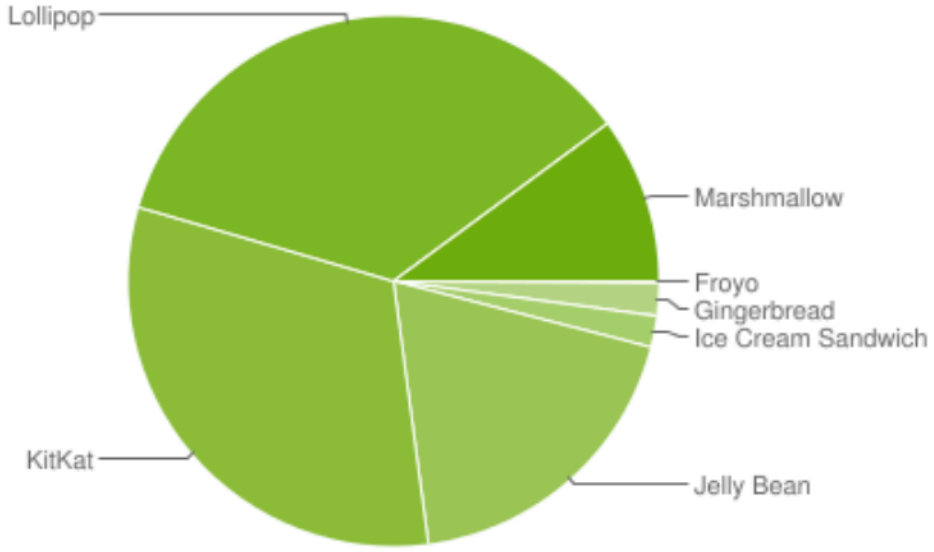
Version	Codename	API	Distribution
2.2	Froyo	8	0.1%
2.3.3 - 2.3.7	Gingerbread	10	2.0%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	1.9%
4.1.x	Jelly Bean	16	6.8%
4.2.x		17	9.4%
4.3		18	2.7%
4.4	KitKat	19	31.6%
5.0	Lollipop	21	15.4%
5.1		22	20.0%
6.0	Marshmallow	23	10.1%

Data collected during a 7-day period ending on June 6, 2016.

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

Versions - Platforms

Version	Codename	API	Distribution
2.2	Froyo	8	0.1%
2.3.3 - 2.3.7	Gingerbread	10	2.0%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	1.9%
4.1.x	Jelly Bean	16	6.8%
4.2.x		17	9.4%
4.3		18	2.7%
4.4	KitKat	19	31.6%
5.0	Lollipop	21	15.4%
5.1		22	20.0%
6.0	Marshmallow	23	10.1%



Target Version

- A developers objective may be to have an application run on as many devices as possible.
  - shoot for the lowest API level possible. Keep in mind the distribution of Android versions on real devices out there.
- You may notice that there are a lot of users of Android 2.3.3+ and 4.1.x. This places the latest and greatest (4.1.x) version as the second largest version currently in the wild.
- This hasn't always been the case because OEMs tended to be very slow in upgrading their OS versions. However, this has changed with Google's strong push to get everyone onto the latest versions.
- Unfortunately, there are still a lot of people who have the 2.3.3 version because they have yet to upgrade their phones to a phone with the hardware capable of handling the newer version.

Latest Version 7.0 – Android Nuguat

- Provides a lot of major improvements and refinements over Marshmallow, along with quite a few nice new usability changes.
- Multi-tasking and Split Screen functionality.
- Bundled Notifications & Quick Reply
- API Level : 24
- ART is still the default runtime

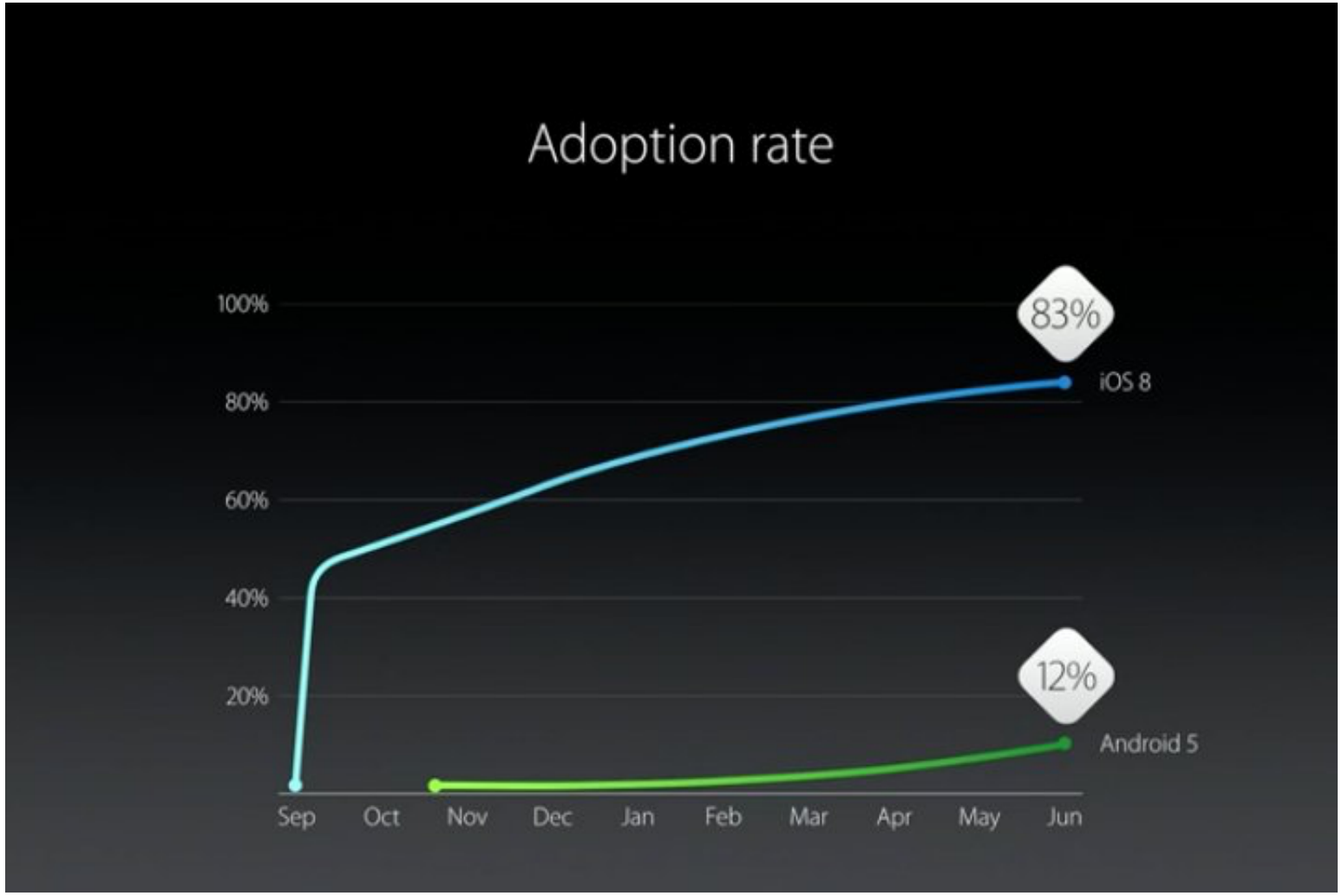
Android 7.0 Nougat

Google Apps

Adoption Rates

- Android adoption traditionally lags behind Apple's because Apple can make its latest iOS available for all users at once because it makes all the hardware and the software.
- Android, on the other hand, is much more fragmented. Because Google lets many different hardware makers use its Android software, it can't control when all phones update to the latest software. So, the newest version of Android is always made available for Google's lineup of Nexus smartphones and tablets, but it can't release software updates to every single Android phones at once.
- Smartphone manufacturers often add their own modifications to Android, which means that the update needs to be approved by each individual carrier first.
- This means that depending on which phone you own, it could take ages to receive the latest software update.
- Here's the brutal chart:

Adoption Rates



http://uk.businessinsider.com/ios-vs-android-adoption-rate-2015-6?r=US&IR=T

Quick Tour of Android (1)

- Most Common Features:
  - User Interface
    - IO widgets (buttons, textboxes, lists etc.)
    - Images
    - 2D/3D drawing
    - ActionBar/ActionMode (recent)
    - ViewPager (recent)
    - Drawer (recent)
  - Database (Sqlite)
  - App Lifecycle
- (More Later on…)

Quick Tour of Android (2)

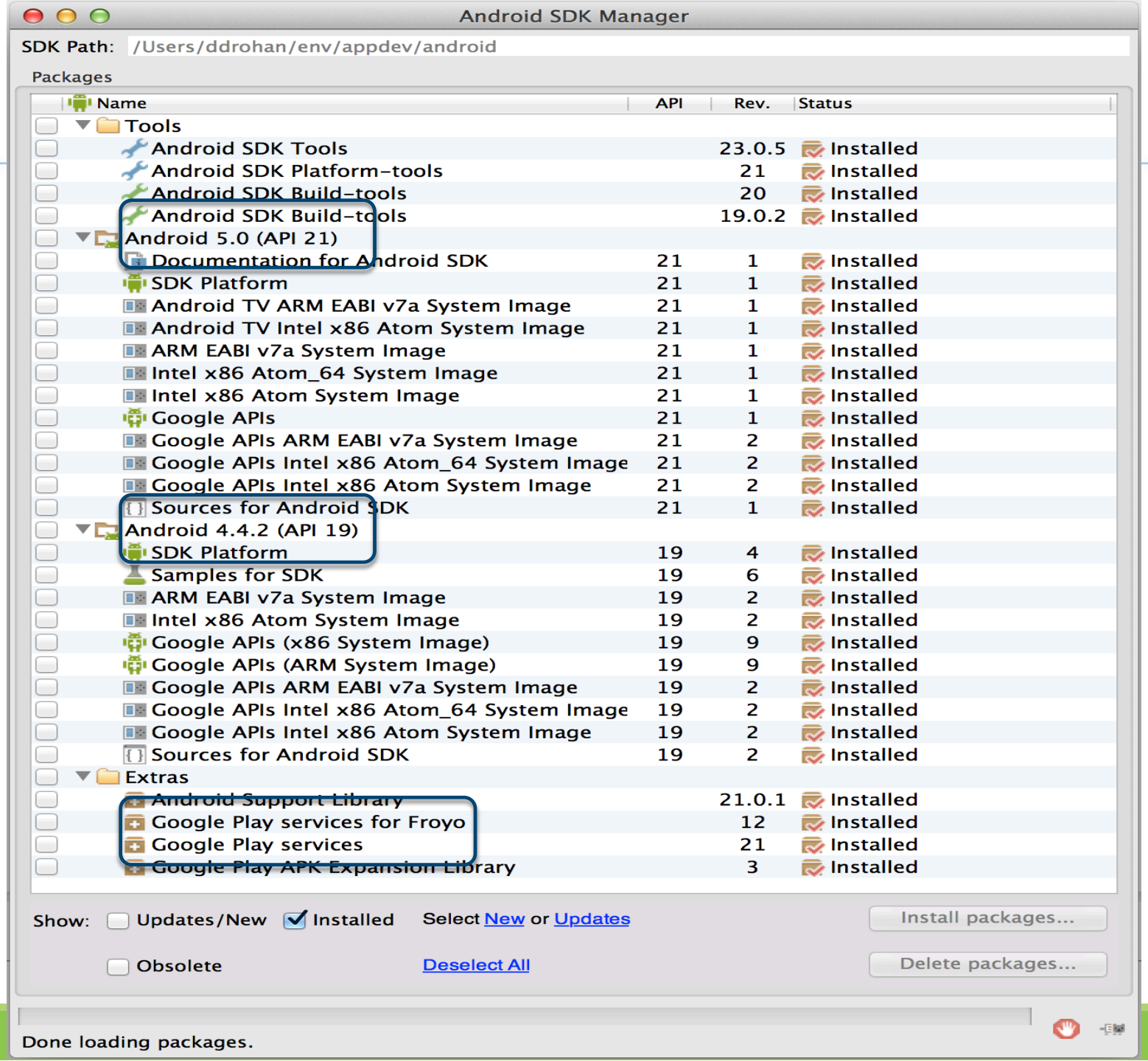
- Less Common (more specific/advanced) Features:
  - Google Maps
  - Hardware APIs
    - GPS/Geo-location, calls, accelerometer, compass, bluetooth, camera
    - Fused Location Provider (latest addition)
  - Multiple processes
    - Managed by DVM (or ART as of Kit Kat 4.4+)
    - Background Services (e.g. AsyncTasks/Volley)
    - Inter-process communications (e.g. Intents)
  - No difference between third-party and native apps (2 lines of code to launch Camera on Device)

Android Applications (re : the installation stuff)

- Android applications get distributed in a .apk file
- APK == “Android Package”
  - It is simply a zip file that has a particular file structure (similar to JAR files that take snapshots of the file system)
  - An APK contains
    - The Android Manifest file (an XML file with lots of metadata)
    - A Resource bundle containing sounds, graphics, etc.
    - The Dalvik classes / ART bytecodes that make up your application

What You need to Get started

- The Android SDK supports several different integrated development environments (IDEs). Here we will focus on Android Studio because it is supported by Google, with the SDK and it's free (***and*** Google aren't supporting Eclipse anymore!)
- No matter which operating system you are using, you will need essentially the same set of tools:
  - IDE
  - Sun's / ORACLE's Java Development Kit (JDK)
  - The Android Software Developer's Kit (SDK)
  - (if you're working with Eclipse) The Android Developer Tool (ADT) plugin, a special Eclipse plug-in
  - Phone driver (for specific device development)
- Android offer an "ADT Bundle" with everything you need http://developer.android.com/sdk/index.html just confirm your SDK with mine (next slide) re solutions…..



Android SDK Setup for this Module

Android Apps vs iPhone Apps

Installing Apps

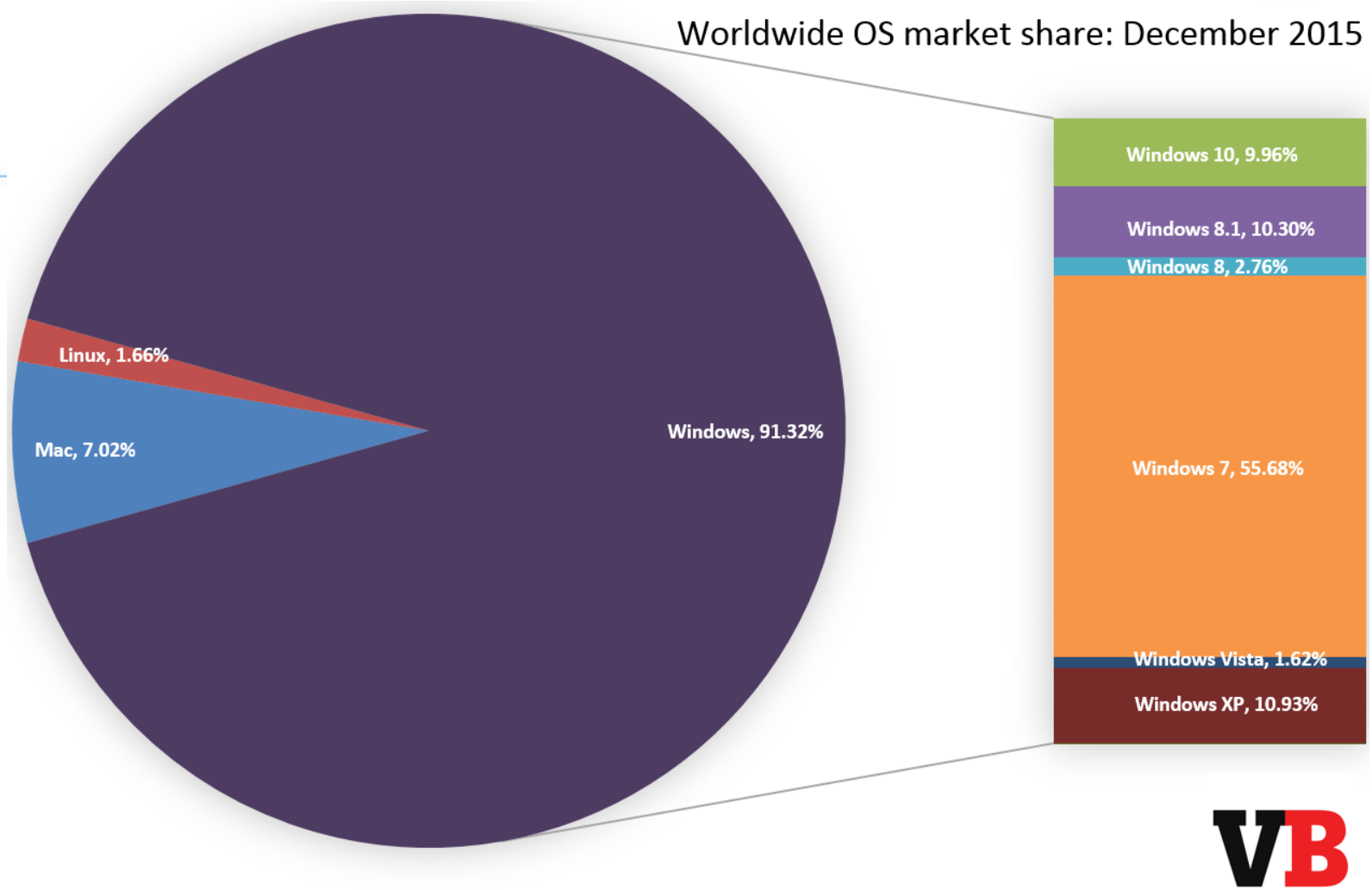
- For Generic (on the market) apps
  - iPhone has larger selection
  - Android catching up
- For In-house-developed corporate apps
  - iPhone apps can only (mostly) be installed via the App Store
    - iPhone requires you to submit app to the Apple App Store and get approval, even for apps from your own company
      - Unless you setup a Provisioning profile or
      - you use something like TestFlight or
      - jailbreak your phone of course…..
  - Android apps can be installed through
    - Google App Store / Google Play
    - Amazon App Store
    - USB connection from PC
    - Email
    - Corporate Web site

Languages for Apps

- iPhone
  - Objective-C
    - Similar to, but not exactly the same as, C++
    - Virtually no corporate presence for Objective-C, other than for mobile apps
    - Swift starting to gain adoption
- Android
  - Java
    - The single most widely used language inside corporations
  - C/C++
    - Can call native apps (with some difficulty) via an approach similar to JNI for desktop Java

OS for DevApps

- iPhone
  - Macs
- Android
  - Anything with Java and Eclipse
    - Macs
    - PCs
    - Linux
    - Solaris

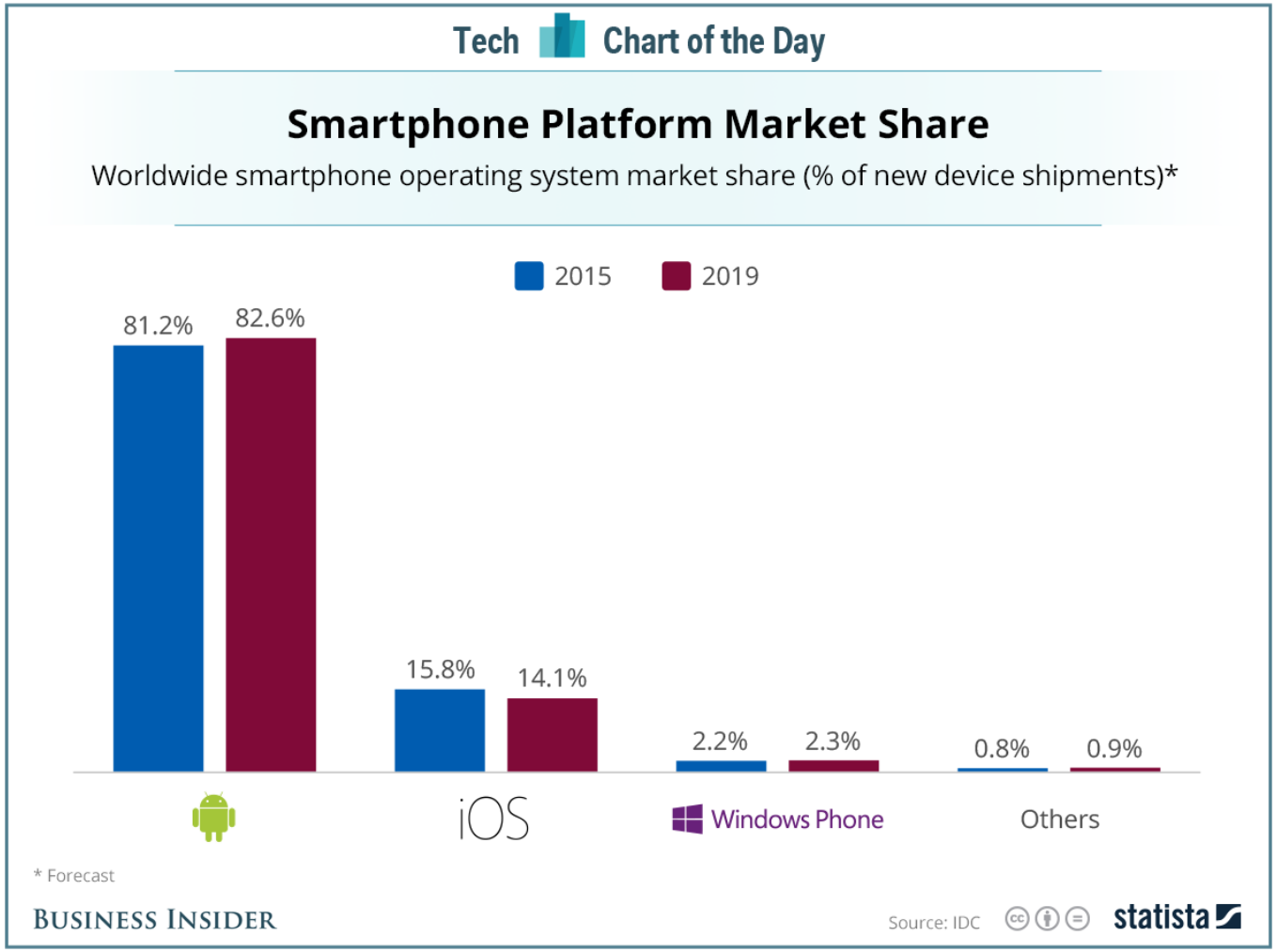


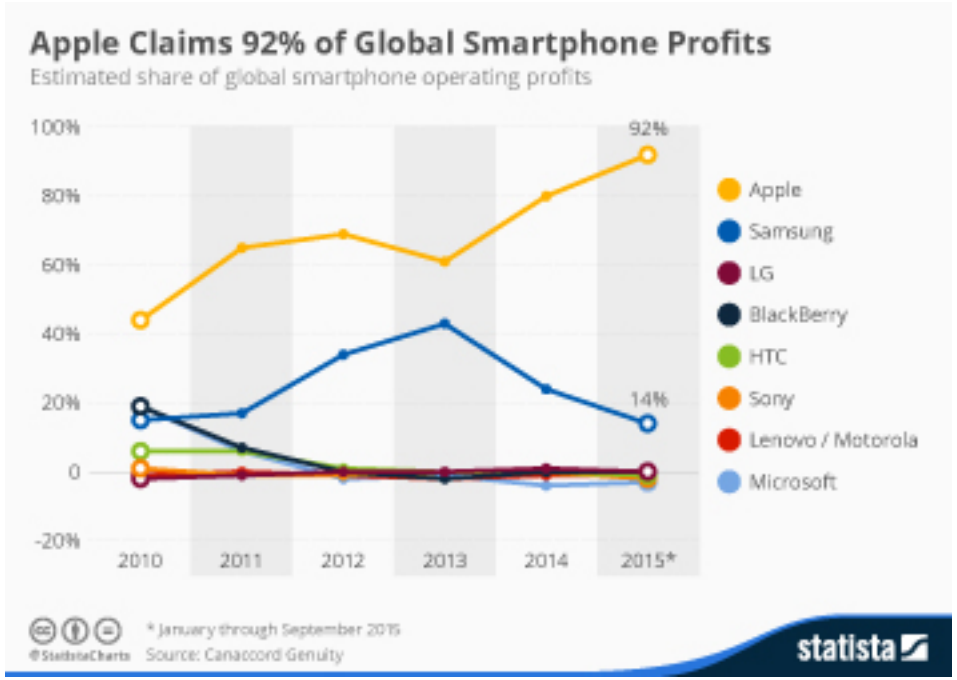
- Issue

http://venturebeat.com/2016/01/01/windows-10-ends-2015-under-10-market-share/

  - Not so much which is cooler and which you personally prefer, but rather which is **already** installed in corporate environments.

Market Share Vs Operating Profits





Other Issues

- Phone features, quality of apps, **customer loyality** and coolness factors
  - Matter of opinion, but iPhone **very** strong here
- Market presence based on sales / app downloads etc.

Bottom Line: Android vs iPhone

- Which to use personally
  - iPhone has bigger app store, and more loyal users
  - Android more open and growing more rapidly
  - Bottom line: no clear winner, personal preferences prevail, but iPhone has the edge (for me!)
- Which to use for in-house apps
  - iPhone apps very hard/restrictive to install, Android very simple
  - iPhone uses mainly Objective-C (Swift on the up), Android uses Java
  - Bottom line: Android is clear winner ☺️ IMHO

References

- Books (in rough order of preference)
  - Android Developer's Cookbook (Steele & To)
  - Busy Coder's Guide to Android Development (Murphy)
    - Online only: http://commonsware.com/Android/
  - Android in Action, 2nd Edition (Ableson, Sen, & King)
  - Android Application Development for Dummies (Felker)
- Online references
  - http://developer.android.com/
    - By far the most important single reference.
  - Android forum on StackOverflow
    - http://stackoverflow.com/questions/tagged/android
  - Android widget gallery
    - http://www.droiddraw.org/widgetguide.html
  - Marty Hall's Tutorials
    - http://www.coreservlets.com/android-tutorial/

Questions?