Introduction to Android

Alessio Vecchio

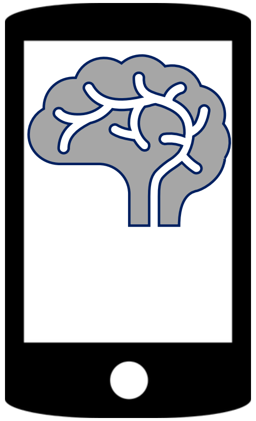
Mobile devices

- Smartwatches
- Smartphones
- Tablets
- Laptops

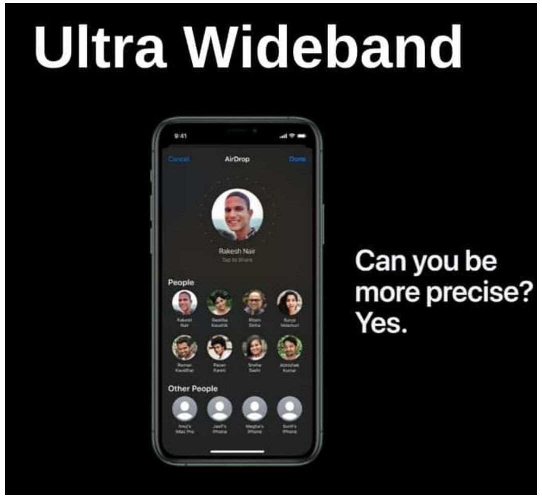
Smartphone

- Smart = Communication + Computing + Sensors
  - Communication: variety of technologies (wi-fi, cellular, bluetooth, NFC)
  - Computing: powerful processors (4-8 cores, GPU), able to execute VMs
  - Sensors: position, temperature, ambient light, pressure, camera, microphone

- Google Pixel 9:
  - Octa-core (1x3.1 GHz Cortex-X4 & 3x2.6 GHz Cortex-A720 & 4x1.92 GHz Cortex-A520), 64 bit
  - 12 GB RAM

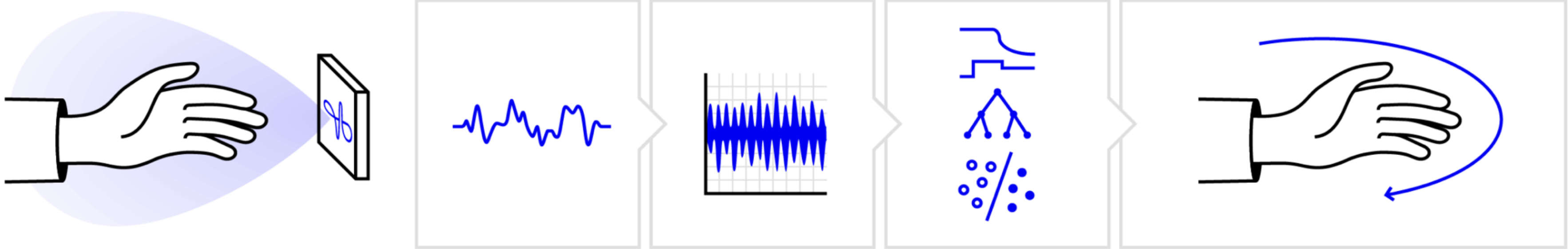


Source: apple.com



Sensors

- Sensors usually available:
  - GPS, microphone, ambient light, proximity, camera, fingerprint
- Other sensors may be present:
  - heart rate, pressure, temperature, motion sensing, UWB
- Information about the physical world -> Intelligent sensing apps
  - Example: amount of burnt calories by measuring physical activity


Gesture

Raw signal

Soli signal transformation

Gesture classification

Detected Swipe gesture

Source: google.com

Communication technologies

- Wi-Fi (IEEE 802.11)
- Cellular networks
- Bluetooth
- Near Field Communications (NFC)

Network Type	Speed	Range	Power	Common Use
WLAN	600 Mbps	45 m – 90 m	100 mW	Internet.
LTE (4G)	5-12 Mbps	35km	120 – 300 mW	Mobile Internet
3G	2 Mbps	35km	3 mW	Mobile Internet
Bluetooth	1 – 3 Mbps	100 m	1 W	Headsets, audio streaming.
Bluetooth LE	1 Mbps	100+ m	.01–.5 W	Wearables, fitness.
NFC	400 kbps	20 cm	200 mW	Mobile Payments

Table credit: Nirjoin, UNC

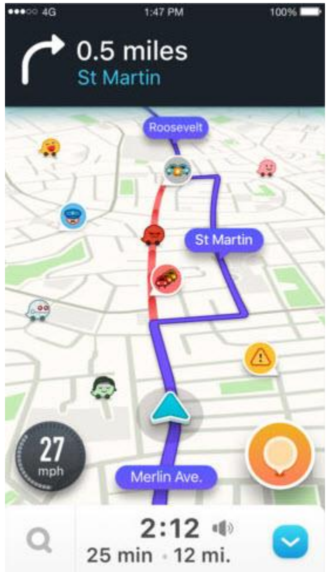
Mobile computing

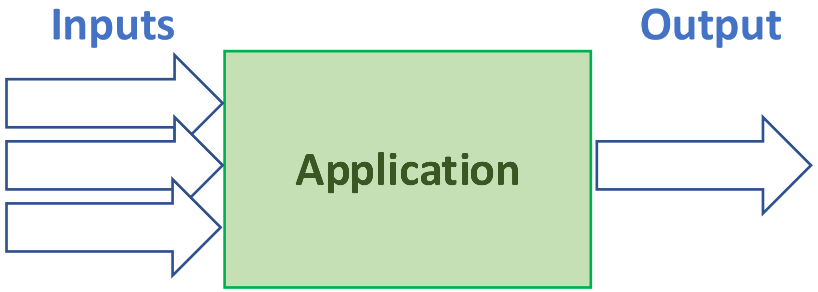
- Humans use a computing device while moving
  - Continuous network connectivity
  - Point of connection might change (Wi-Fi AP, cell tower)
- From the user’s perspective: same approach of desktop machines
  - User always initiates all activities (launching apps)
  - Examples:
    - Web browsing on a train
    - Chatting while walking
- No interaction with surrounding environment
- Infrastructure (network equipment) just for providing connectivity

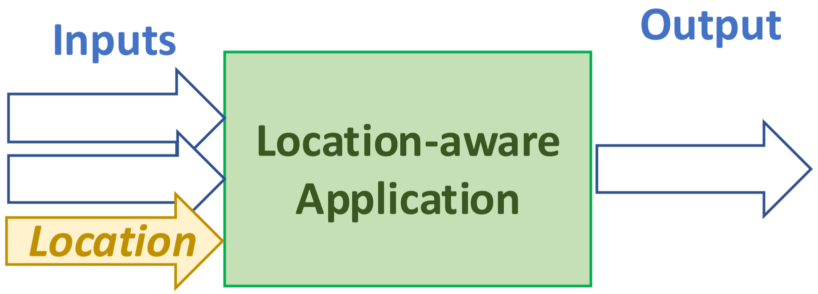
http://www.leparisien.fr/societe/alerte-aux-smombies-ces-zombies-du-smartphone-qui-s-exposent-a-des-accidents-22-04-2019-8058333.php

Location-aware computing

- Mobile computing ≠ Location-awareness
- Location-aware: position must be one of the programs inputs
- Different position –> different output
- Examples:
  - Maps app
  - Museum app







Mobile apps not necessarily location-aware

- If behavior/output does not change when location changes -> not location-aware
- Application executed on smartphones just for convenience

- In some cases classification can be not so clear: ticket reservation app could use location to show the closest movie theaters

Major problem: energy efficiency

- During last decades resources increased significantly with the exception of energy

- Energy saving techniques:
    - CPUs able to adapt to the computational needs (most of the time smartphone is idle)
    - Turn off screen aggressively
    - Application specific methods: in video streaming reduce resolution
    - Make the user aware of remaining energy:
        - switch to energy-saving mode, reducing app functionalities

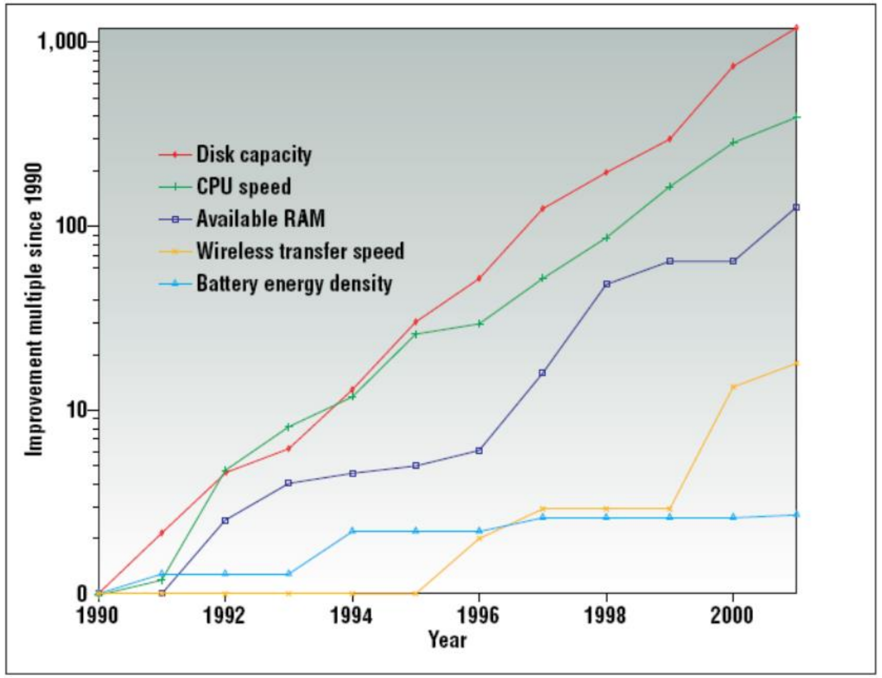


Figure 1. Improvements in laptop technology from 1990–2001.

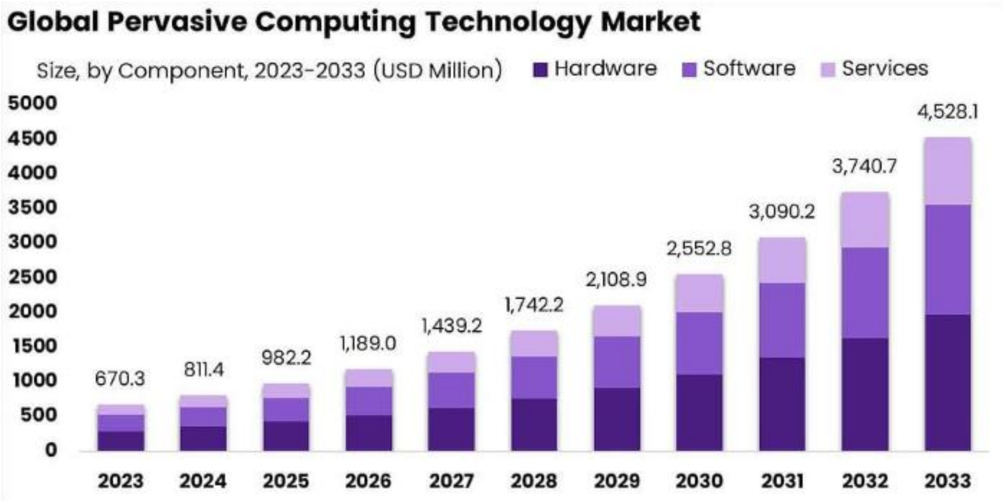
Starner, IEEE Pervasive Computing, Dec 2003

Pervasive computing

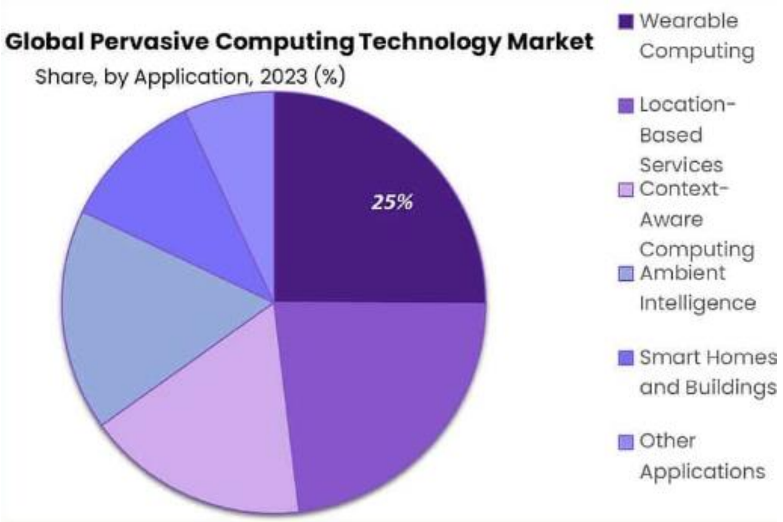
- Ubiquitous computing, «The Computer for the 21st Century», M. Wiser
  - «The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistiguishable from it.»
- Pervasive computing = Ubiquitous computing
- Goal: to assist humans in tasks
  - Reminders, suggestions, ease interaction with the environment, well-being, health, …
- System needs to understand user's intention
- Sensors are generally used to collect information about the physical space
- System initiates activity, pro-active, user's explicit input may be not required
- Examples
  - Advising about driving conditions to home, work
  - Improve fitness exercise
  - automatically detect social relationships

Pervasive computing & wearable devices

- Wearable devices (e.g. smartwatches) provide highly personal information, useful in pervasive computing
- Apple Watch can trigger an emergency SOS call if it detects a hard fall and subsequent inactivity
- Wearables can provide patients with personalized health data, which could assist with self-diagnosis and behavior change interventions
- Monitoring of activity or sleep patterns via an accelerometer in a watch



©Market.us

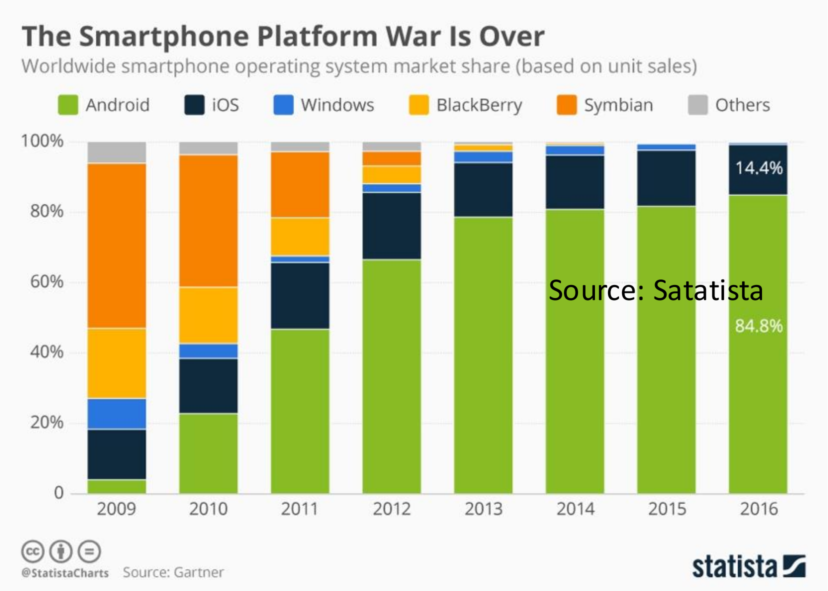


Android

- Android is world’s most popular mobile operating system
  - Open source (https://source.android.com)
- Google:
  - Owns Android, maintains it, extends it
  - Distributes Android OS, developer tools, free to use
  - Runs Google Play store
- Open Handset Alliance: 80+ companies
  - Mobile operators: Vodafone, Telefonica, Telecom Italia, NTT Docomo, Sprint, …
  - Handset manufacturers: Samsung, Sharp, Sony, HTC, Acer, Asus, Nec, …
  - Semiconductor companies: ARM, Freescale, Nvidia, TI, …
  - Software companies: Google, eBay, …
  - Other companies: Accenture, Wind River

Android’s history and present

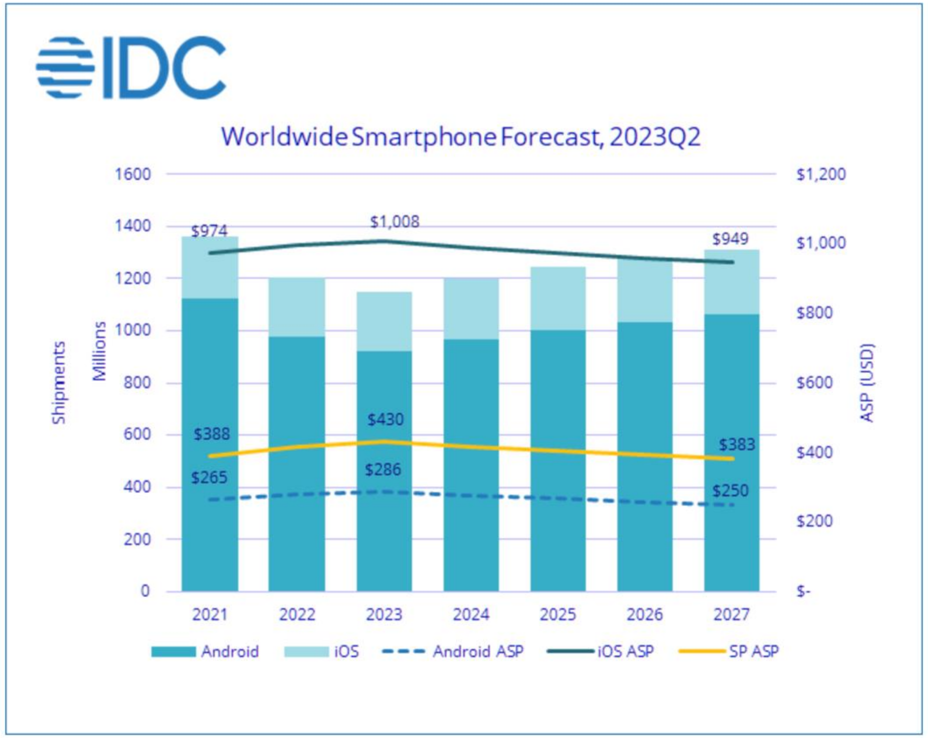
- History of evolution:
  - 2005 Google buys Android Inc.
  - 2007 OHA is announced and Android is presented
  - 2008 Android SDK 1.0 is released
  - 2010 Nexus One is commercialized
  - 2012 800K+ Android devices activated every day
- Some recent numbers (sales)
- Share changes significantly depending on country


The Smartphone Platform War Is Over

Worldwide smartphone operating system market share (based on unit sales)

Source: Satatista

@StatistaCharts Source: Gartner


Worldwide Smartphone Forecast, 2023Q2

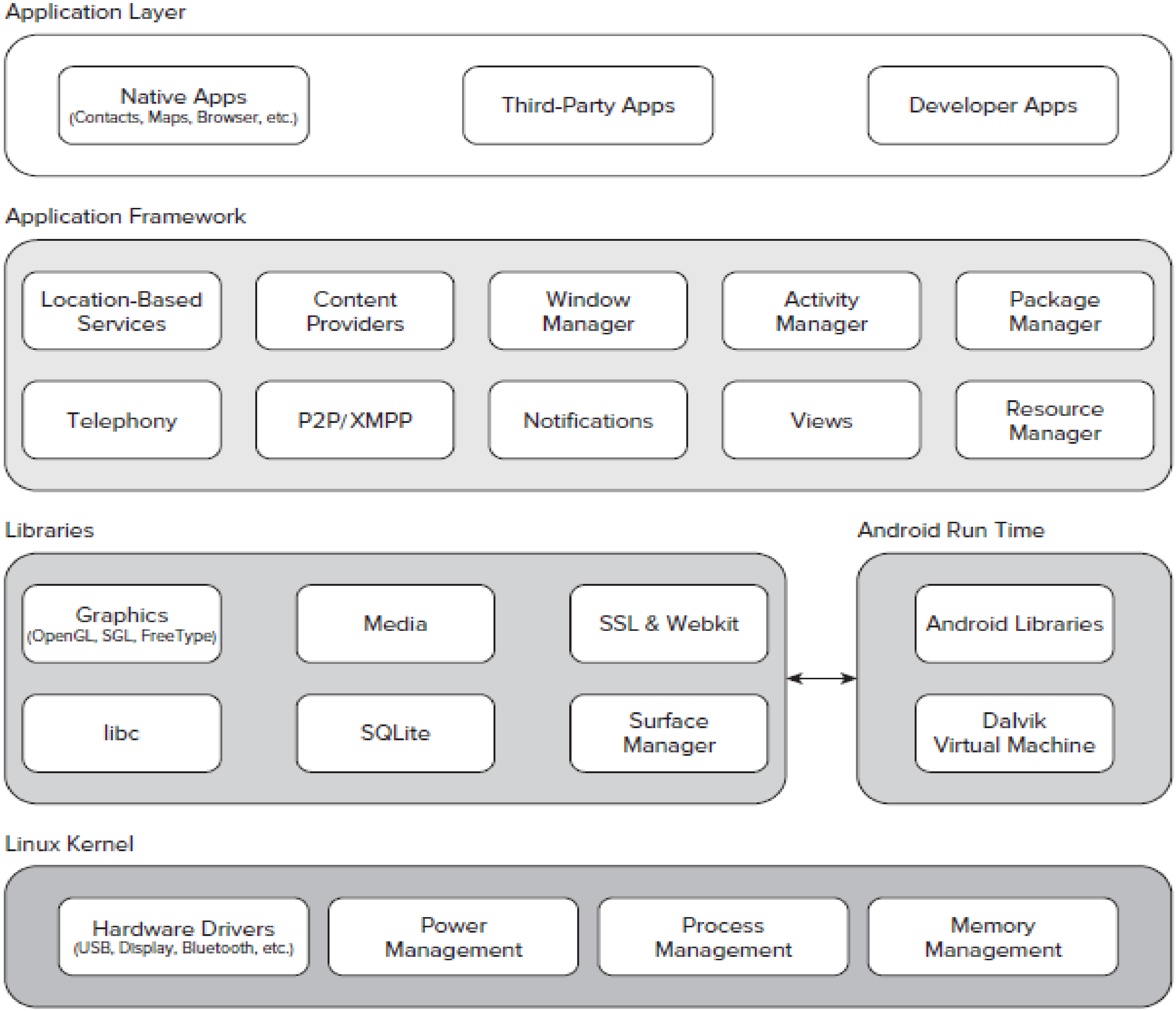
Multiple sizes and purposes

- A range of devices (form factors):
  - Smartphone (from the beginning)
  - Tablet (support introduced in Android 3.0)
  - TV (launch June 2014, Android 5.0)
  - Auto (launch June 2014, Android 5.0)
  - Wear (support introduced in Android 4.4)
  - Glass (experimental project, now only limited distribution with selected companies)

Supported platforms

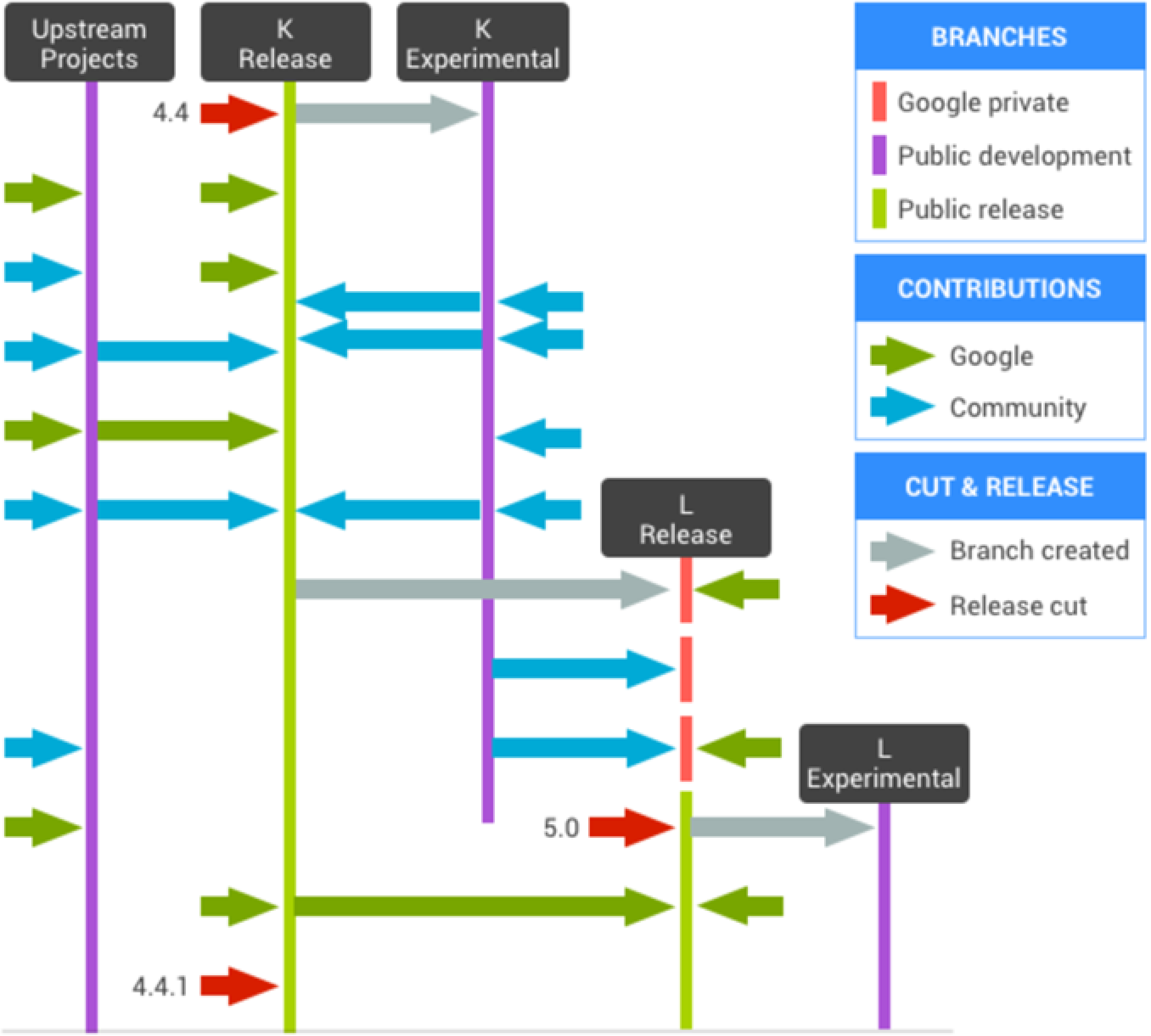
- Supported HW platforms:
  - Supports ARM, x86, MIPS architectures (ARM from the beginning)
  - Both 32bit and 64bit (the latter since Android 5.0)
  - ARM/x86/MIPS or 32/64, transparent at the application level (unless you need native (C) code)
- In 2023 google announced RISC-V as a tier-1 supported platform

Android


Application Layer
Native Apps
(Contacts, Maps, Browser, etc.)
Third-Party Apps
Developer Apps
Application Framework
Location-Based Services
Content Providers
Window Manager
Activity Manager
Package Manager
Telephony
P2P/XMPP
Notifications
Views
Resource Manager
Libraries
Graphics
(OpenGL, SGL, FreeType)
Media
SSL & Webkit
libc
SQLite
Surface Manager
Android Run Time
Android Libraries
Dalvik Virtual Machine
Linux Kernel
Hardware Drivers
(USB, Display, Bluetooth, etc.)
Power Management
Process Management
Memory Management

Android

- AOSP maintains the software
- Software is ported by OEMs to run on their own hardware

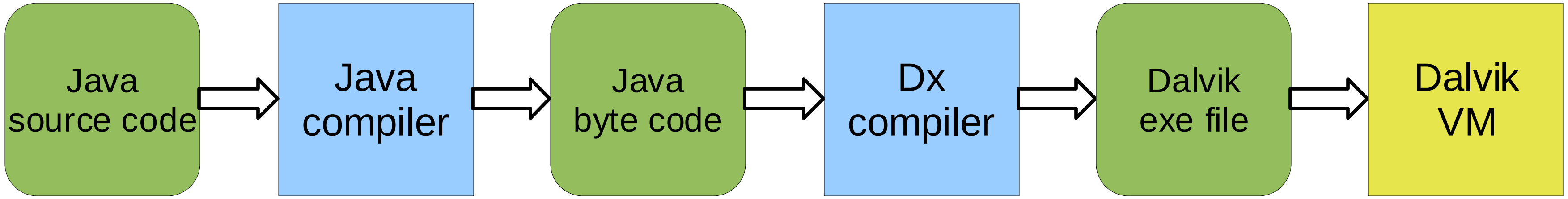


Android app development

- Development tools: free, no specific license is needed
- Distributed as APK file. Each file contains:
  - *dex* executable
  - resources (xml files, images, …)
  - native libs
- Distribution
  - app stores: Google Play Store and others from third parties
  - the Web: just link the APK on a webpage
  - You can also send the APK by email

Java and Dalvik

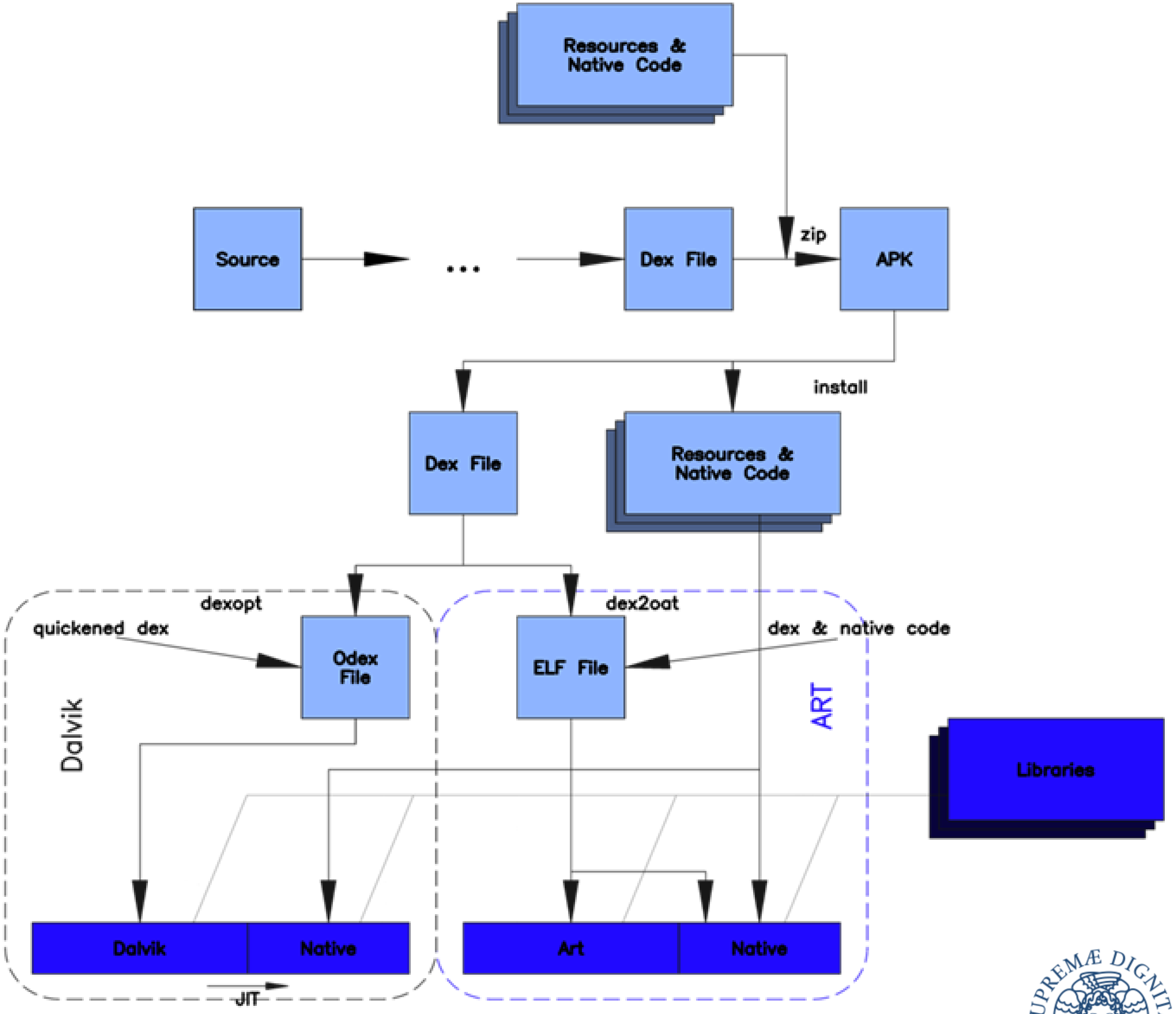
- Dalvik VM ensures developers never have to worry about a particular HW implementation
- The Dalvik VM executes Dalvik executable files, a format optimized to ensure minimal memory footprint
- Developers create *.dex* executables by transforming Java language compiled classes using the tools supplied within the SDK



- Source files: which version of Java? Approximately JSE without GUI, plus Android specific libraries

ART

- ART: Android Run-Time (default since 5.0)
- Dalvik vs ART
  - JIT vs AOT
  - ART faster (depends on application, some benchmarks 2x), less energy (no compilation)



Google vs Oracle

- Technical decisions influenced by Google vs Oracle legal battle
- Google wrote its own version of Java: different implementation but same API
- Oracle sued Google in 2010 and, in 2012, the judge ruled that APIs are not subject to copyright
- In 2014 Oracle appealed and previous decision was reverted: Java APIs are copyrightable, but leaving open the possibility that Google's was fair use
- In 2016 a jury unanimously agreed that Google's use of the Java APIs was fair use
- Oracle filed another appeal, and the verdict was Google's use was not a fair use
- In 2019 Google filed a petition to Supreme Court and in 2021, the Court issued an opinion holding that Google's use of the Java APIs was fair use

Now Kotlin (+Java)

- Kotlin:
  - Statically typed language
  - Introduced in 2019 to the Android platform
  - Increasingly pushed by Google
  - Interoperable 100% with Java code
  - New content is Kotlin-first
  - Claimed to be more concise, avoid boilerplate code
  - Not limited to Android
- In Android
  - kt -> bytecode -> dalvik

Why Android?

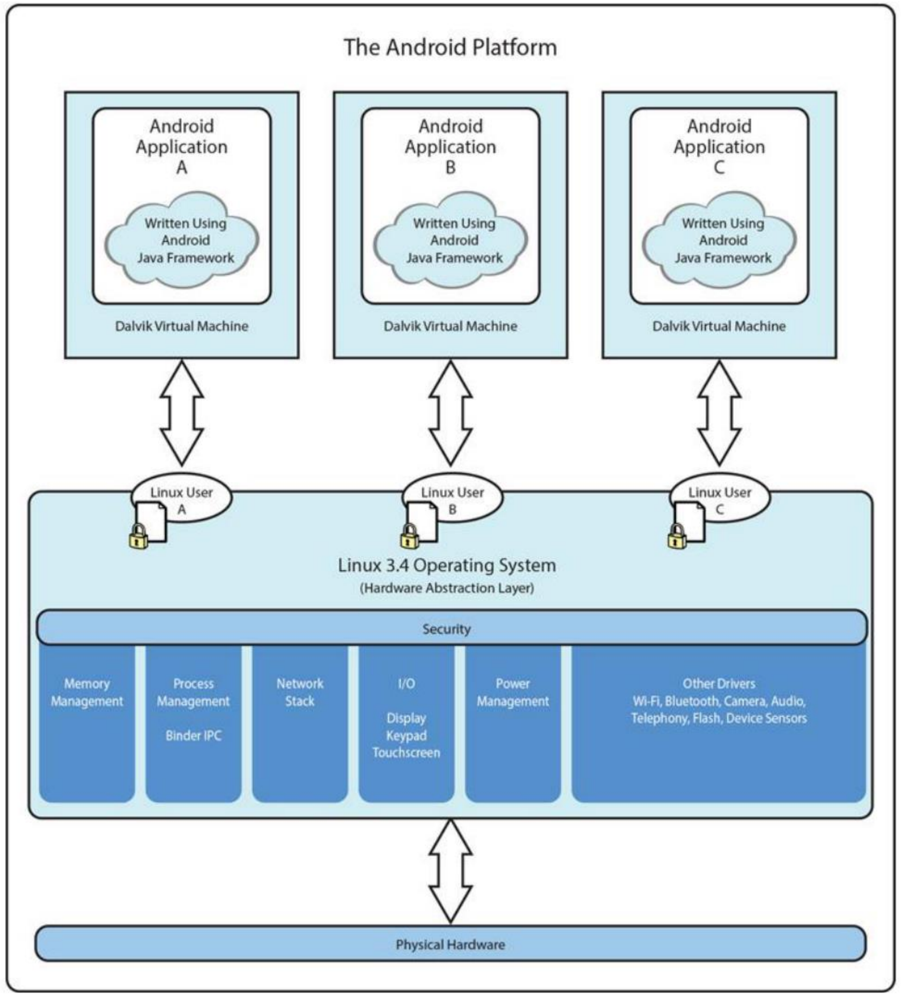
- Development tools are free of charge and available for different OSes
- Larger user base
- Code is open source
    - Actually some components are proprietary (e.g. Maps, Play store)

Additional development problems

- Many versions of Android out there
  - Different APIs
  - Available hardware depends on devices (even if same API level)
  - Different UI models (some buttons have disappeared)
  - Some vendors introduce their own customizations to the UI and may provide additional libraries
  - If you use native code, you have to provide compiled binaries for the processor architectures you want to support
- Several form factors: smartphones, tablets, TVs, smartwatches
- We will focus on smartphones

Isolation between apps

- Android is a multi-user Linux system
- Each application is a different user, ID assigned by the system
- Access control: only process with same ID can access files
- An application's code runs in isolation from other applications
- Every application runs in its own Linux process.
  - Started when any of the app components needs to be executed
  - Shut down when no longer used



Ref: Introduction to Android Programming, Annuzzi, Darcey & Conder

References

- http://developer.android.com