CS 4518 Mobile and Ubiquitous Computing
Lecture 1: Introduction

Emmanuel Agu

About Me

A Little about me

- WPI Computer Science Professor
- Research interests:
    - mobile computing especially mobile health, computer graphics
- Started working in mobile computing, wireless in grad school
- CS + ECE background (Hardware + software)
- Current active research: Mobile health apps
    - E.g: AlcoGait app to detect how drunk Smartphone owner is
        - https://www.youtube.com/watch?v=pwZaoKmfq8c

Administrivia

Administrivia: Schedule

- ***Week 1-4:*** I will introduce class, concepts, Android (Students: Android programming, assigned projects)
  - **Goal:** Students acquire basic Android programming skills to do excellent project
  - Focus on programming mobile & ubicomp components
- ***Week 4:*** Students will present final project proposal
- ***Week 5-7:*** Students work on final project
- ***Week 7:*** Students present + submit final projects
- Quizzes (5) throughout

Requirements to get a Grade

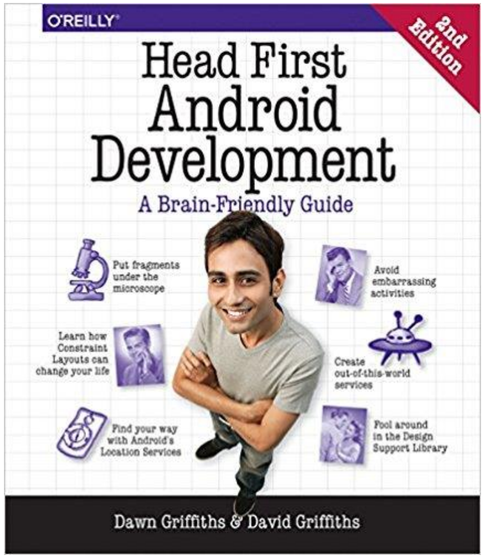
- **Grading policy:**
  - Assigned Projects 40%, Final project: 35%, Quizzes: 25%
- **Final project phases:** (See class website for deadlines)
  1. Pick partners, form project groups
  2. Submit 1-slide of proposed idea (problem + envisioned solution)
  3. Present project proposal

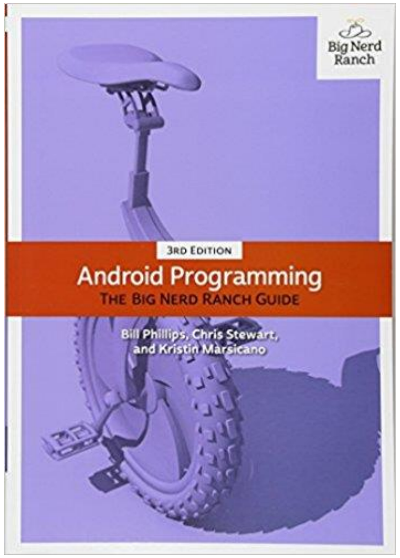
     + plus submit proposal (intro + related work + methodology/design + proposed project plan)
  4. Build app, evaluate, experiment, analyze results
  5. Present results + submit final paper (in week 7)
- **New final project aspects this offering:**
  - Larger teams (5 or 6 members)
  - Points for degree of difficulty of project

Course Texts

- **Android Texts:**
  - *Head First Android Dev, (2nd ed),* Dawn and David Griffiths, O'Reilly, 2017
  - *Android Programming: The Big Nerd Ranch (Third edition)*, Bill Phillips, Chris Stewart and Kristin Marsicano, The Big Nerd Ranch, 2017

Gentle, visual intro





Bootcamp Tutorial

- Will also use official Google Android documentation
- Learn from research papers: Why not text?

Course Assistants

TA: Chai Nimkar

SA: Rachel Plante

Class in 2 Halves

- 2 Halves: About 50 mins each half
- Break of about 10 mins
- Talk to me **at the end NOT during break**
  - I need break too

Poll Question

- How many students:
  1. **Own** recent Android phones (running Android 4.4, 5, 6 , 7 or 8?)
  2. **Can borrow** Android phones for projects (e.g. from friend/spouse)?
  3. **Do not own and cannot borrow** Android phones for projects?

Mobile Devices

Mobile Devices

- Smart phones (Blackberry, iPhone, Android, etc)
- Tablets (iPad, etc)
- Laptops
- Smartwatches

SmartPhone Hardware

- Smart = Communication + Computing + Sensors
  - **Communication:** Talk, text, Internet access, chat
  - **Computing:** Java apps, JVM, apps
    - Powerful processors: Quad core CPUs, GPUs
  - **Sensors:** Camera, video, location, temperature, heart rate sensor, etc
- Google Pixel XL phone: Quad core 1.6 GHz Snapdragon CPU, Adreno 530 GPU, 4GB RAM
  - A PC in your pocket!!
  - Multi-core CPU, GPU
  - Runs OpenGL ES, OpenCL and now Deep learning (Tensorflow)

Smartphone Sensors

- Typical smartphone sensors today
  - accelerometer, compass, GPS, microphone, camera, proximity
- Can sense physical world, inputs to intelligent sensing apps
  - E.g. Automatically turn off smartphone ringer when user walks into a class

Growth of Smartphone Sensors

- Every generation of smartphone has more and more sensors!!

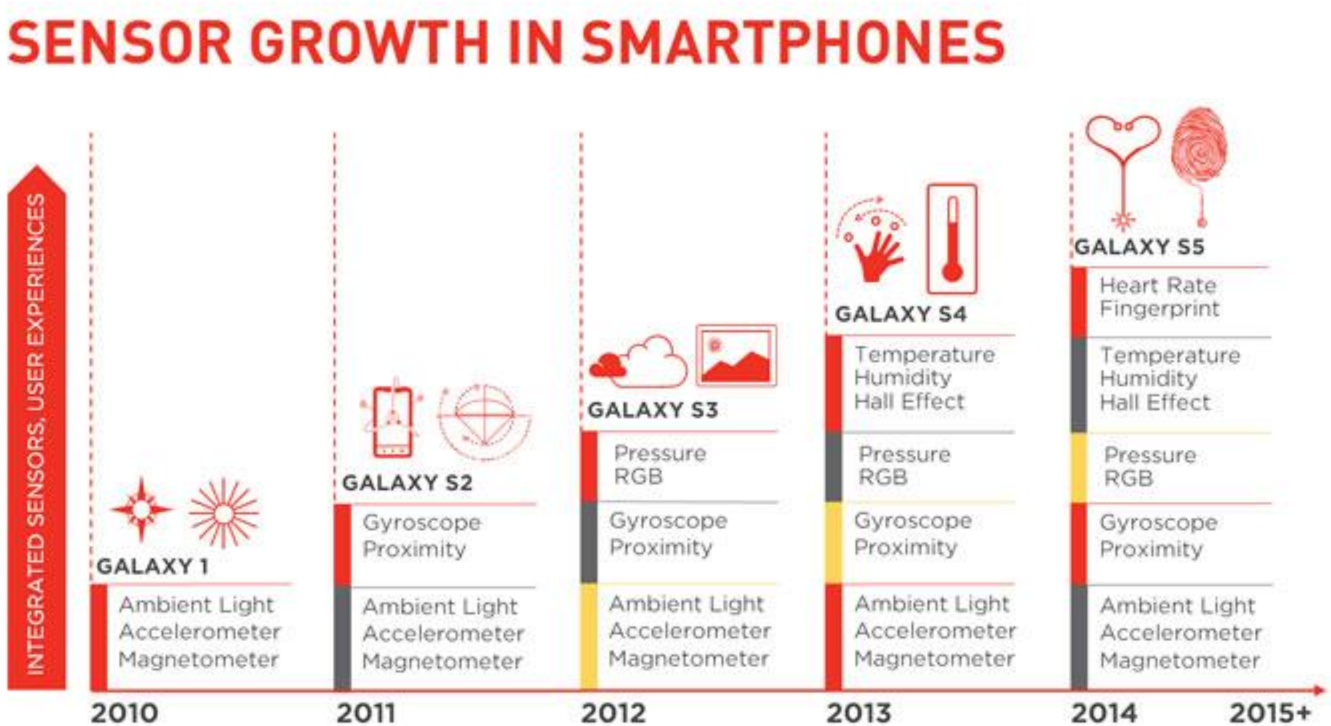


Image Credit: Qualcomm

Future sensors?

- Complex activity sensor,
- Pollution sensor,
- etc

Wireless Networks

Wireless Network Types

- **Wi-Fi (802.11):** (e.g. Starbucks Wi-Fi)
- **Cellular networks:** (e.g. Sprint network)
- **Bluetooth:** (e.g. car headset)
- **Near Field Communications (NFC)**

  e.g. Mobile pay: swipe phone at dunkin donut

Bluetooth

Wi-Fi

NFC

Wireless Networks Comparion

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

Different speed, range, power, uses, etc

Mobile Computing

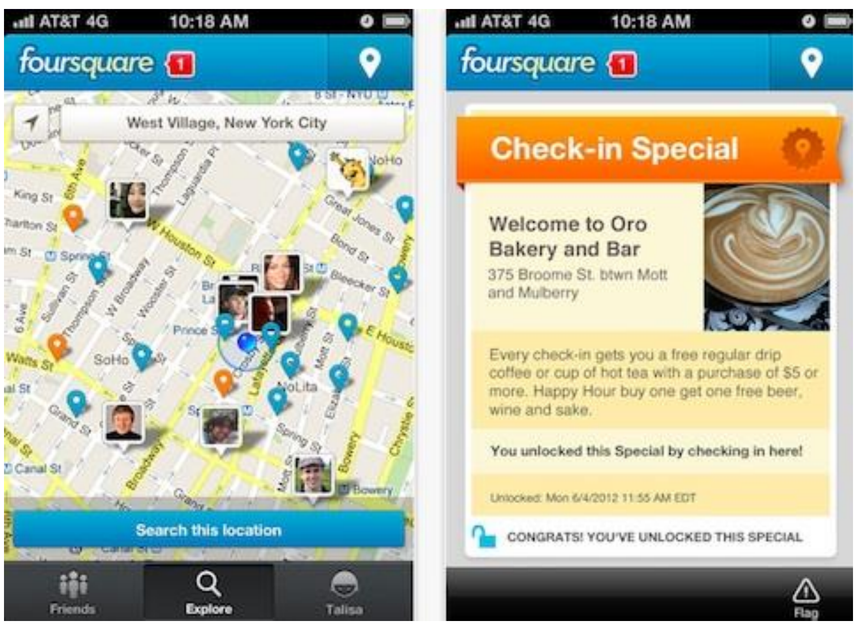
mo·bile

adjective
/ˈmōbəl,ˈmōˌbīl/

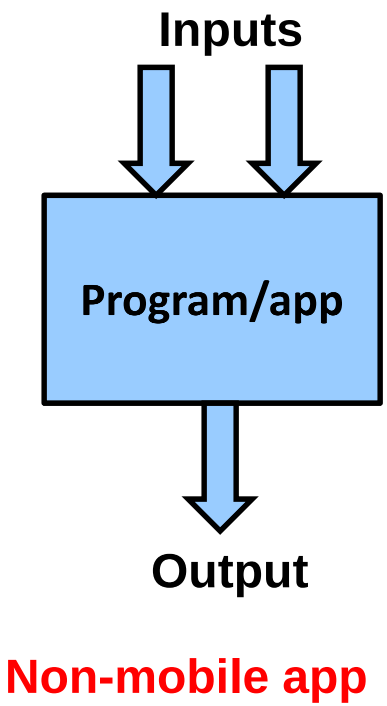
1. able to move or be moved freely or easily.
   "he has a major weight problem and is not very mobile"
   synonyms: able to move (around), moving, walking; motile; ambulant

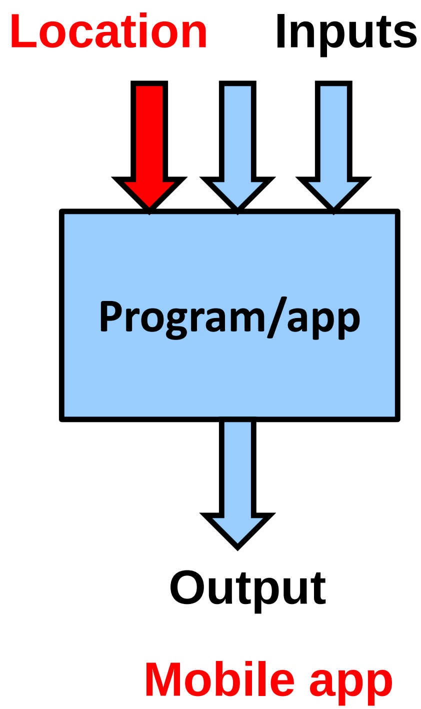
Mobile Computing

- Human computes while moving
  - Continuous network connectivity,
  - Points of connection (e.g. cell towers, WiFi access point) might change
- **Note: Human initiates all activity**, (e.g launches apps)
- Wireless Network is *passive*
- **Example:** Using *foursquare.com* on SmartPhone



Related Concept: Location-Awareness





- Mobile computing = computing while location changes
- **Location-aware:** Location must be one of app/program's inputs
- Different user location = different output (e.g. maps)
- **E.g.** User in California gets different map from user in Boston

Location-Aware Example

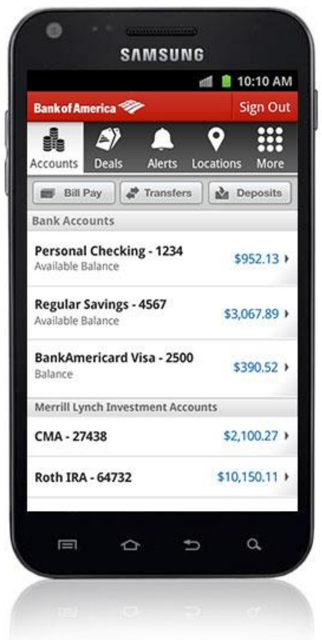
- Location-aware app must have different behavior/output for different locations
- Example: Mobile yelp
  - **Example search:** Find Indian restaurant
  - App checks user's location
  - Indian restaurants **close to user's location** are returned

Example of Truly Mobile App: Word Lens

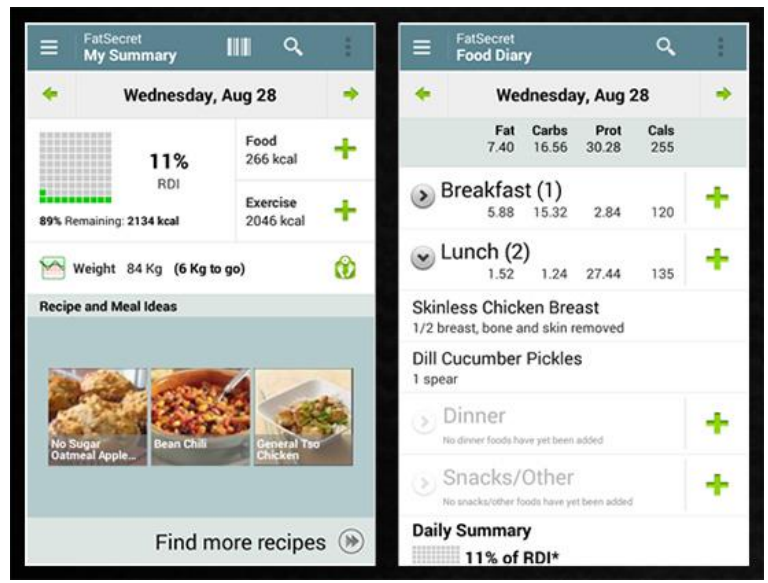
- Translates signs in foreign Language
- Location-dependent because location of sign, language? varies

Some Mobile apps are not Location-Aware

- If output does not change as location changes, not location-aware
- Apps run on mobile phone **just for convenience**
- **Examples:**



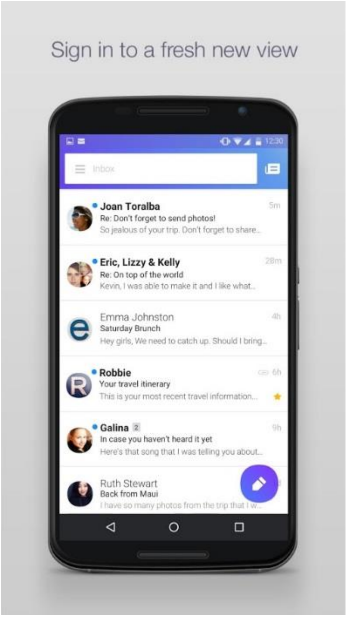
Mobile banking app



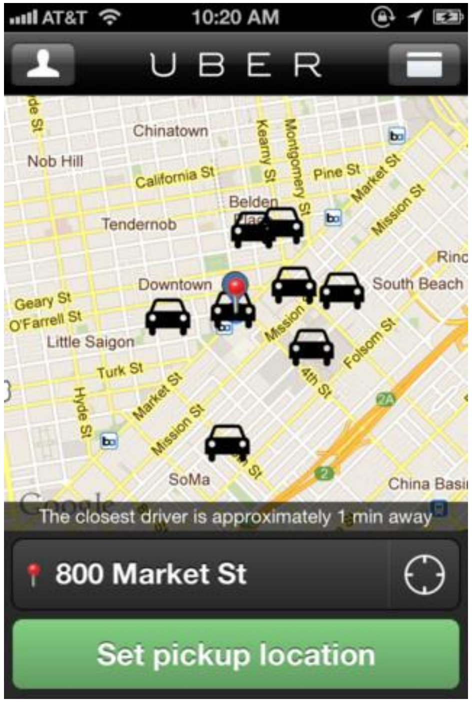
Diet recording app

- Distinction can be fuzzy. E.g. Banking app may display nearest locations

Which of these apps are Location-Aware?



a. Yahoo mail mobile



b. Uber app

Mobile Device Issue: Energy Efficiency

- Most resources increasing exponentially *except* battery energy (ref. Starner, IEEE Pervasive Computing, Dec 2003)

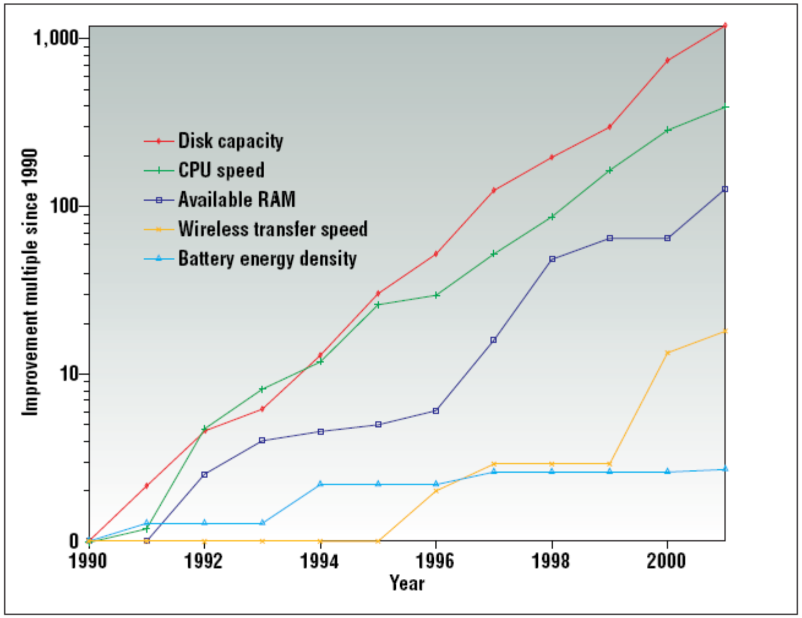


Figure 1. Improvements in laptop technology from 1990–2001.

- Some energy saving strategies:
  - **Energy harvesting:** Energy from vibrations, charging mats, moving humans
  - **Scale content:** Reduce image, video resolutions to save energy
  - **Auto-dimming:** Dim screen whenever user not using it. E.g. talking on phone
  - **Better user interface:** Estimate and inform user how long each task will take
    - E.g: At current battery level, you can either type your paper for 45 mins, watch video for 20 mins, etc

Ubiquitous Computing

u·biq·ui·tous

/yo͞oˈbikwədəs/

adjective

present, appearing, or found everywhere.
"his ubiquitous influence was felt by all the family"
synonyms: omnipresent, ever-present, everywhere, all over the place, pervasive,

Ubiquitous Computing

- Collection of specialized assistants to assist human in tasks (reminders, personal assistant, staying healthy, school, etc)
- App figures out user's current state, intent, assists them
- **How?** array of *active* elements, sensors, software, Artificial intelligence
- Extends *mobile computing* and *distributed systems* (more later)
- **Note:** System/app initiates activities, has intelligence
- **Example:** Google Assistant, feed informs user of
  - Driving time to work, home
  - News articles user will like
  - Weather
  - Favorite sports team scores, etc
- Also supports 2-way conversations

User Context

- Imagine a genie/personal assistant who wants to give you all the "right information" at the right time
  - Without asking you any questions
- Examples:
  - Detect traffic ahead, suggest alternate route
  - Bored user, suggest exciting video, etc
- Genie/personal assistant needs to passively detect user's:
  - Current situation (Context)
  - Intention/plan

Ubicomp Senses User's Context

- Context?
  - *Human:* motion, mood, identity, gesture
  - *Environment:* temperature, sound, humidity, location
  - *Computing Resources:* Hard disk space, memory, bandwidth
  - *Ubicomp example:*
    - *Assistant senses:* Temperature outside is 10F (environment sensing) + Human plans to go work (schedule)
    - *Ubicomp assistant advises:* Dress warm!
- Sensed **environment + Human + Computer resources** = ***Context***
- *Context-Aware* applications adapt their behavior to context

Sensing the Human

- Environmental sensing is relatively straight-forward
  - Use specialized sensors for temperature, humidity, pressure, etc
- Human sensing is a little harder (ranked easy to hard)
  - **When:** time (Easiest)
  - **Where:** location
  - **Who:** Identification
  - **How:** (Mood) happy, sad, bored (gesture recognition)
  - **What:** eating, cooking (meta task)
  - **Why:** reason for actions (extremely hard!)

5 W's + 1 H

- Human sensing (gesture, mood, etc) easiest using cameras
- Research in ubiquitous computing integrates
  - location sensing, user identification, emotion sensing, gesture recognition, activity sensing, user intent

Sensor

- **Example:** E.g. door senses only human motion, opens
- **Sensor:** device that can sense physical world, programmable, multi-functional for various tasks (movement, temperature, humidity, pressure, etc)
- Device that can take inputs from physical word
  - Also includes camera, microphone, etc
- Ubicomp uses data from sensors in phone, wearables (e.g. clothes), appliances, etc.

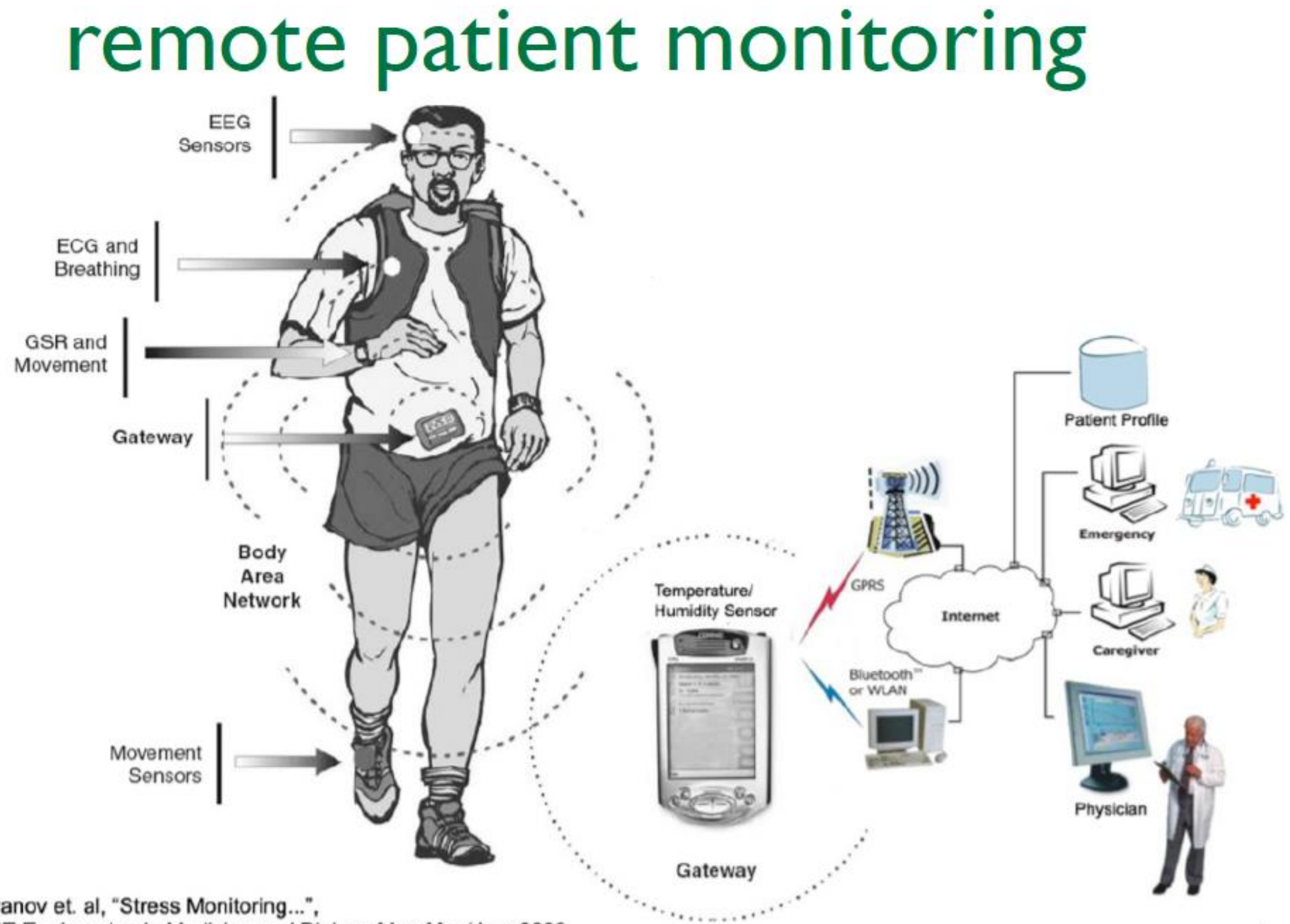
(courtesy of MANTIS project, U. of Colorado)

RFID tags

Tiny Mote Sensor, UC Berkeley

Ubiquitous Computing: Wearables

Ubiquitous Computing: Wearable sensors for Health



Jovanov et. al, "Stress Monitoring...",
IEEE Engineering in Medicine and Biology Mag.May/June2003

UbiComp: Wearables, BlueTooth Devices

Body Worn
Activity Trackers

Bluetooth
Wellness
Devices

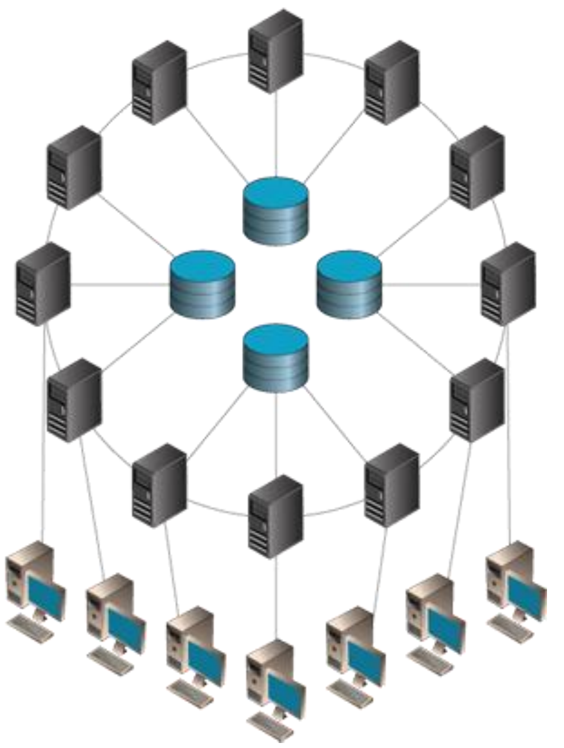
External sources of data for smartphone

Definitions: Portable, mobile & ubiquitous computing

Distributed Computing

- Computer system is physically distributed
- User can access system/network from various points.
- E.g. Unix cluster, WWW
- Huge 70's revolution

- *Distributed computing example:*
  - WPI students have a CCC account
  - Log into CCC machines,
  - Web surfing from different terminals on campus (library, dorm room, zoolab, etc).



- **Finer points:** network is fixed, Human moves

Portable (Nomadic) Computing

- **Basic idea:**
  - Network is fixed
  - device moves and changes point of attachment
  - No computing while moving

- ***Portable (nomadic) computing example:***
  - Mary owns a laptop
  - Plugs into her home network,
  - **At home:** surfs web while watching TV.
  - Every morning, brings laptop to school, plug into WPI network, boot up!
  - **No computing while traveling to school**

Mobile Computing Example

- Continuous computing/network access while moving, automatic reconnection

- *Mobile computing example:*
  - John has SPRINT PCS phone with web access, voice, SMS messaging.
  - He runs apps like facebook and foursquare, continuously connected while walking around Boston

- **Finer points:**
  - John and mobile users move
  - Network deals with changing node location, disconnection/reconnection to different cell towers

Ubiquitous Computing Example

- ***Ubiquitous computing:*** John is leaving home to go and meet his friends. While passing the fridge, the fridge sends a message to his shoe that milk is almost finished. When John is passing grocery store, shoe sends message to glasses which displays “BUY milk” message. John buys milk, goes home.

- **Core idea:** ubiquitous computing assistants **actively** help John

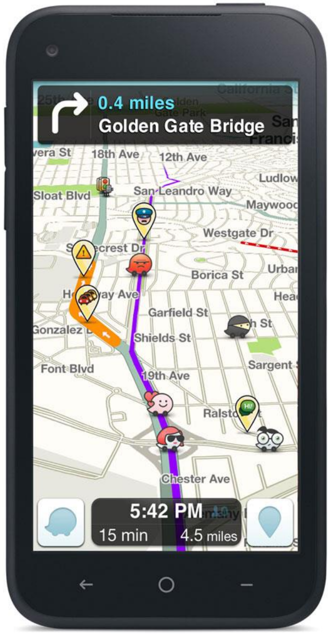
SmartPhone Sensing

Smartphone Sensing

- Smartphone used to sense human, environment

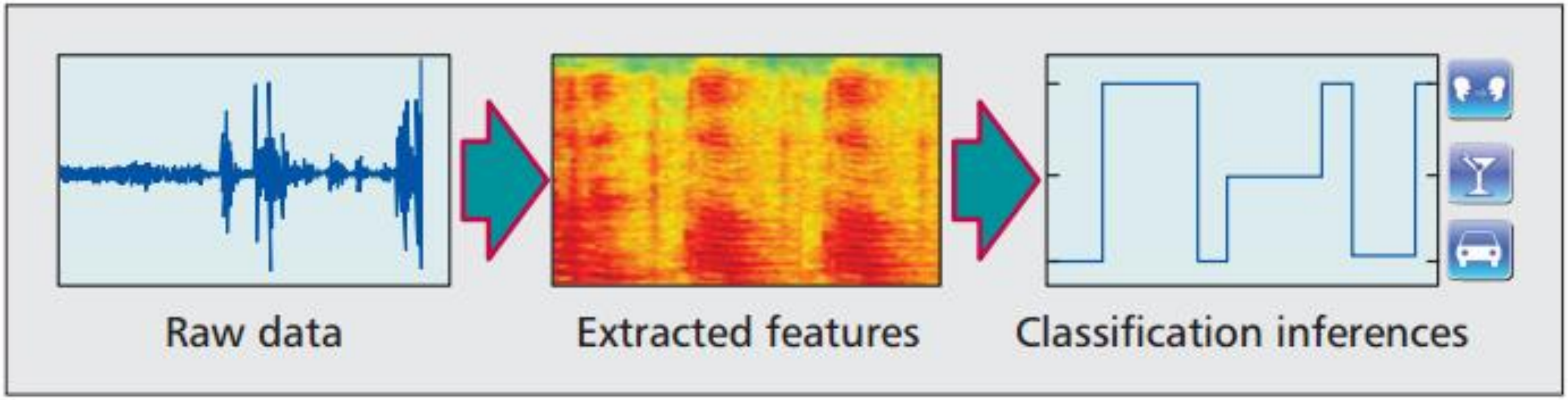
- *Example:* Human activity sensing (e.g. walking, driving, climbing stairs, sitting, lying down)

- *Example 2:* Waze crowdsourced traffic



Sensor Processing

- Machine learning commonly used to process sensor data
  - Action to be inferred is hand-labelled to generate training data
  - Actual data is mined for combinations of sensor readings corresponding to action
- Example: Smartphone detects user's activity (e.g. walking, running , sitting,) by classifying accelerometer sensor data



What Can We Detect/Infer using These Sensors

Smartphone Sensing!!

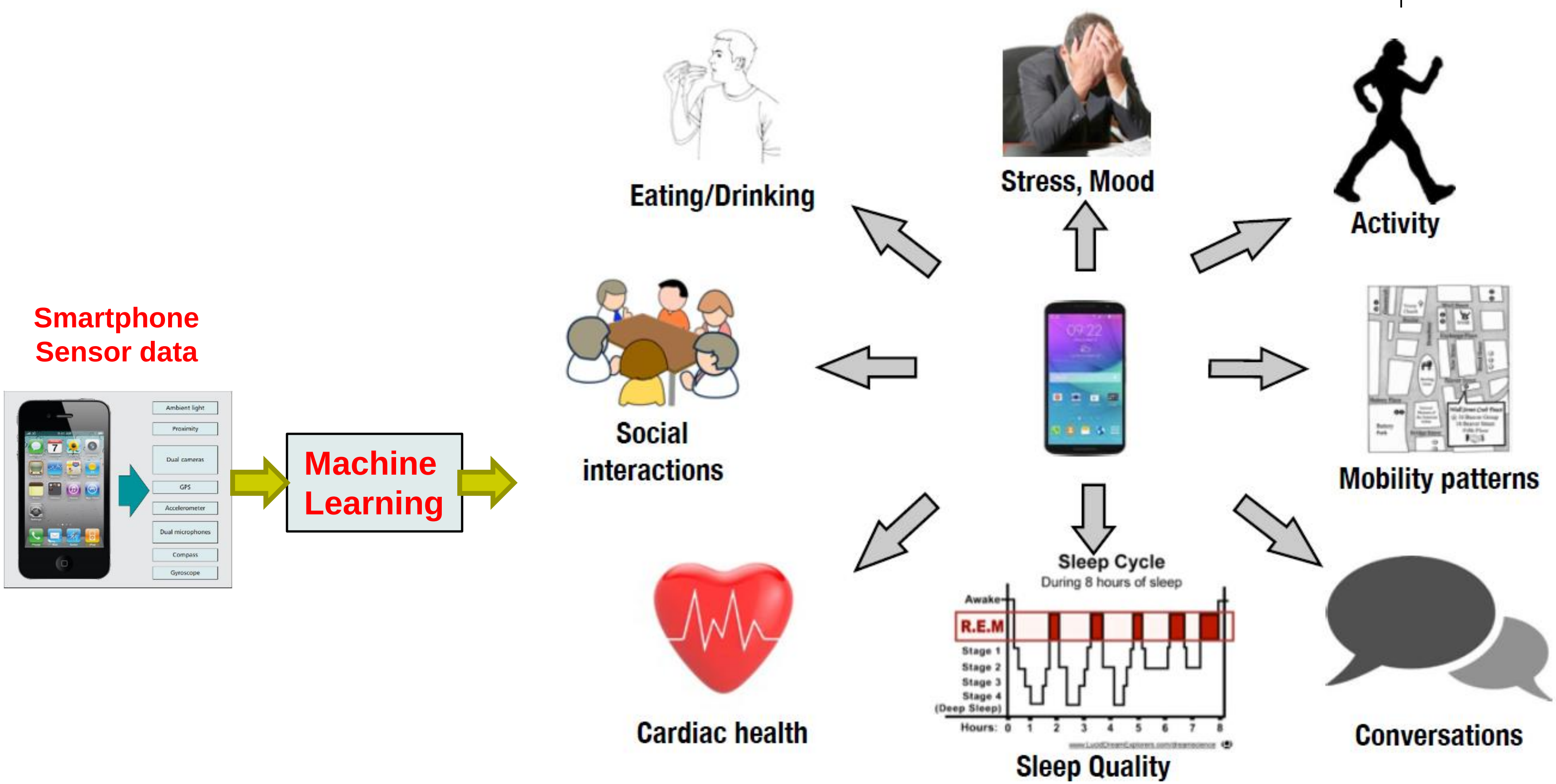
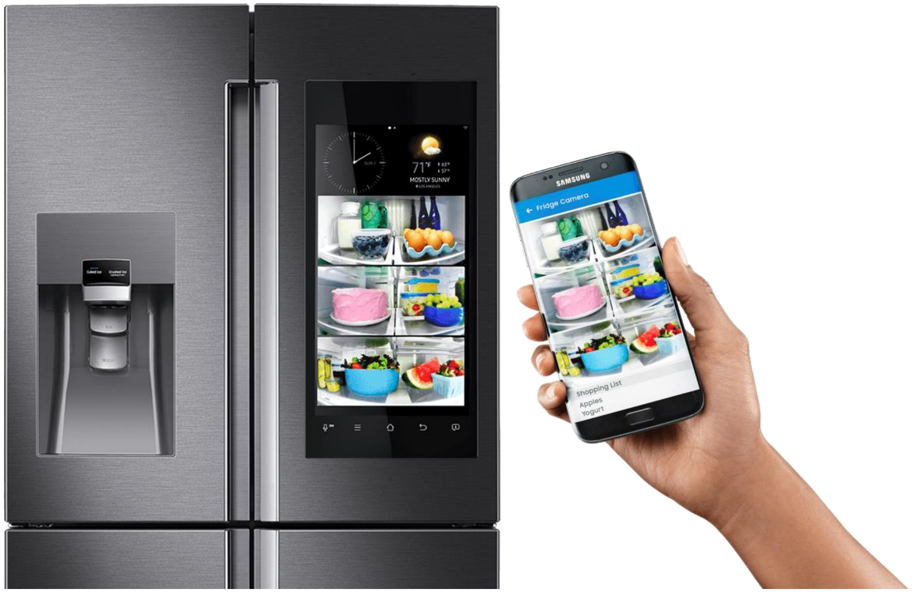


Image Credit: Deepak Ganesan, UMass

Internet of Things (IoT)

IoT: Networked Smart Things (Devices)

- Smart things: Can be accessed, controlled over the network, learns users patterns



Nest Smart thermostat
- Learns owners manual settings
- Turns down heat when not around

Smart Fridge
- See groceries in fridge from anywhere

Other Ubicomp Systems

- **Smart Homes:** Continuously monitors elders who live in smart home, automatically dials 911 if elder ill, fall
  - Falls kill many old people who live alone
- **Smart buildings:** Senses presence of people, ambient temperature, people flow, dynamically adjusts heating/cooling
  - Can save over 40% of energy bill
- **Smart Cities:** Real time data from Sensors embedded in street used to direct drivers to empty parking spots
  - About 30% of traffic jam caused by people hunting for parking

Introduction to Android

What is Android?

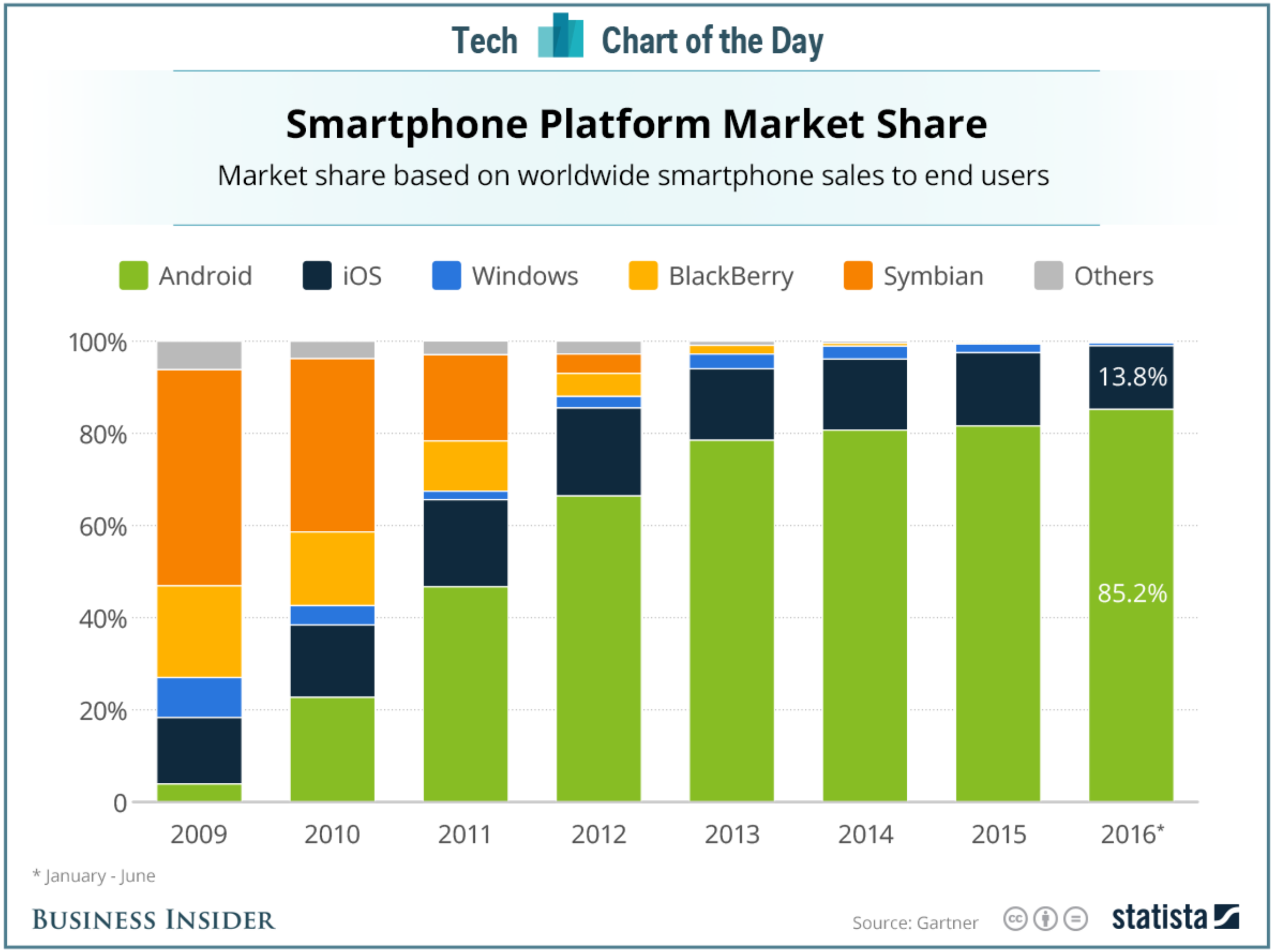
- Android is world's leading mobile operating system
  - Open source (https://source.android.com/setup/)

- **Google:**
  - Owns Android, maintains it, extends it
  - Distributes Android OS, developer tools, free to use
  - Runs Android app market

SmartPhone OS

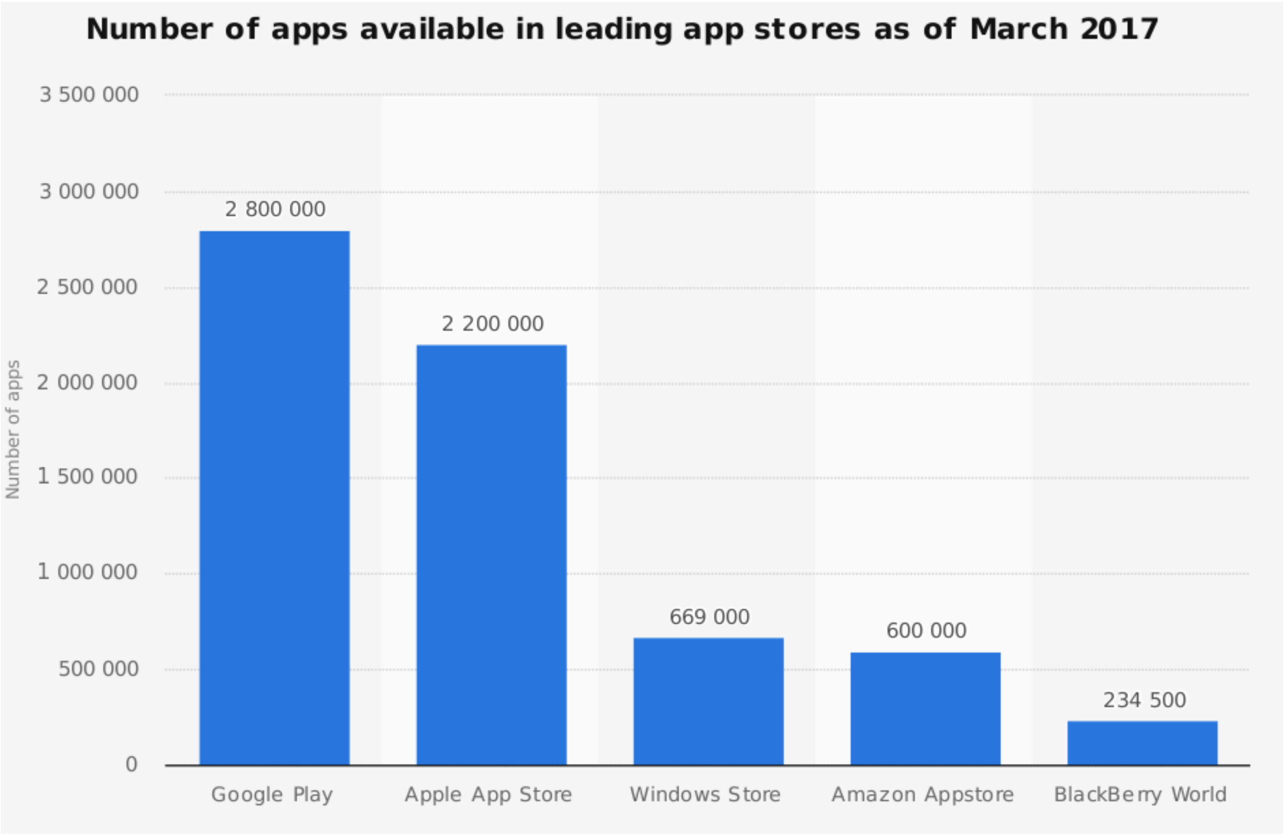
- Over 80% of all phones sold are smartphones
- Android share 86% worldwide

Source: Statista



Android Growth

- Over 2 billion Android users, March 2017 (ref: the verge)
- 2.8 million apps on the Android app market (ref: statista.com)
  - Games, organizers, banking, entertainment, etc



Android is Multi-Platform

Google Glass
(being redone)

In-car console

Smartwatch

Android runs on
all these devices

Tablet

Smartphone

This Class: Focuses
Mostly on Smartphones!

Television

Android for Mobile Computing and Ubicomp

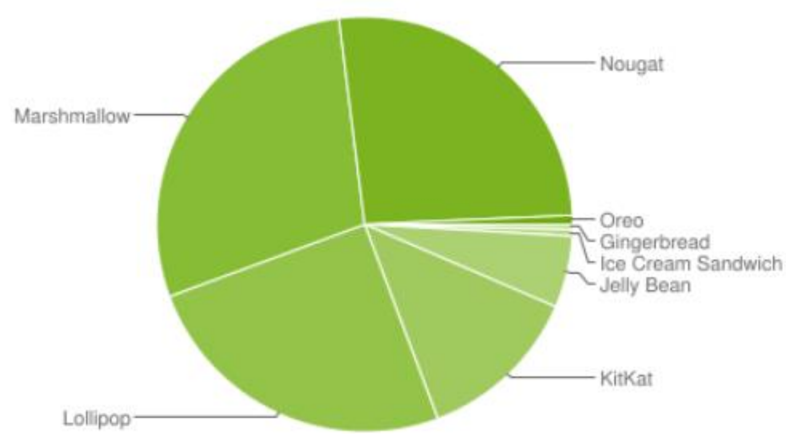
- Android for Mobile programmable modules
  - Audio/video playback, taking pictures, database, location detection, maps

- Android for Ubicomp programmable modules
  - Sensors (temperature, humidity, light, etc), proximity
  - Face detection, activity recognition, place detection, speech recognition, speech-to-text, gesture detection, place type understanding, etc
  - Machine learning, deep learning

Android Versions

- Class will use Android 7 (“Nougat”)
- Officially released December 5, 2016
- Latest version is Android 8 (Oreo), released August 2017
- Below is Android version distribution as at January 8, 2018

Version	Codename	API	Distribution
2.3.3 - 2.3.7	Gingerbread	10	0.4%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	0.5%
4.1.x	Jelly Bean	16	1.9%
4.2.x		17	2.9%
4.3		18	0.8%
4.4	KitKat	19	12.8%
5.0	Lollipop	21	5.7%
5.1		22	19.4%
6.0	Marshmallow	23	28.6%
7.0	Nougat	24	21.1%
7.1		25	5.2%
8.0	Oreo	26	0.5%
8.1		27	0.2%

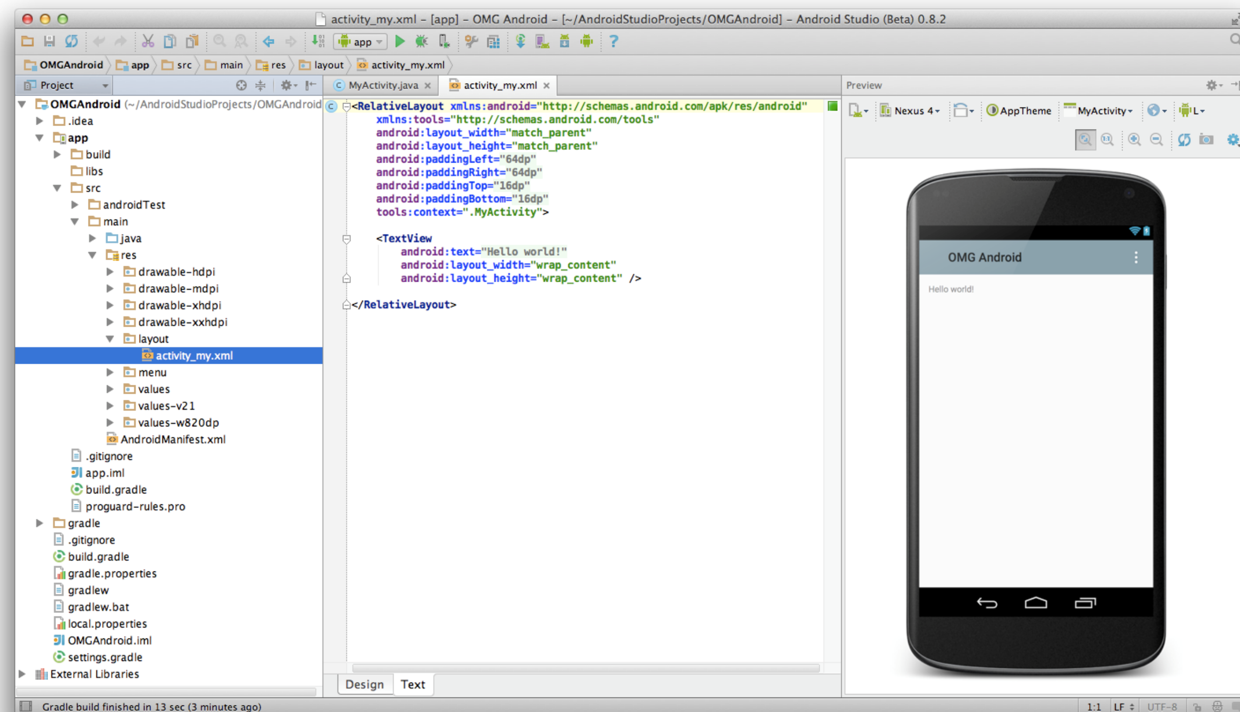


Source: http://developer.android.com/about/dashboards/index.html

Android Developer Environment

New Android Environment: Android Studio

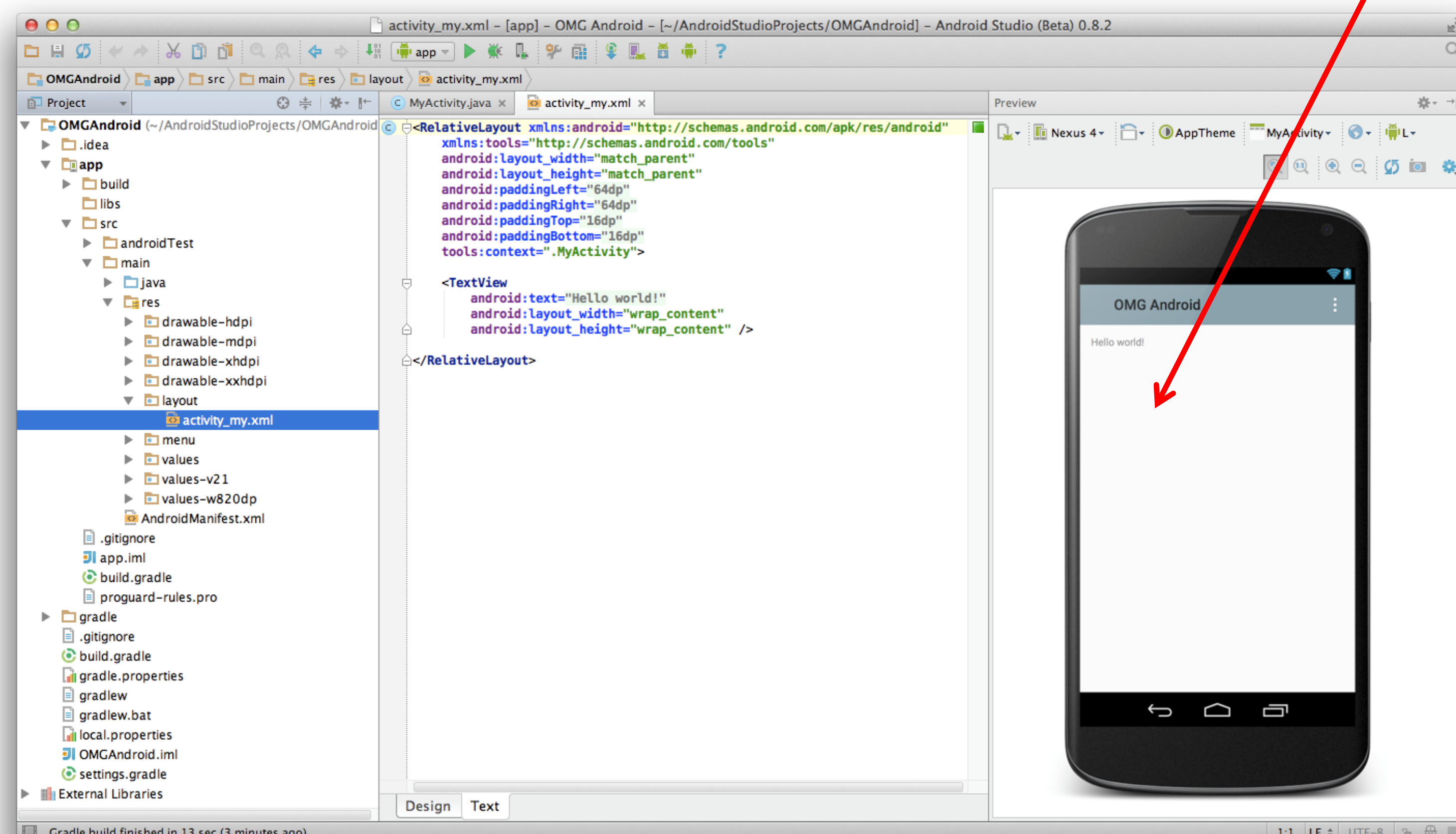
- Old Android dev environment used **Eclipse + plugins**
- Google developed it's own IDE called **Android Studio**
- Integrated development environment, cleaner interface, specifically for Android Development (e.g. drag and drop app design)
- In December 2014, Google announced it will stop supporting Eclipse IDE



Where to Run Android App

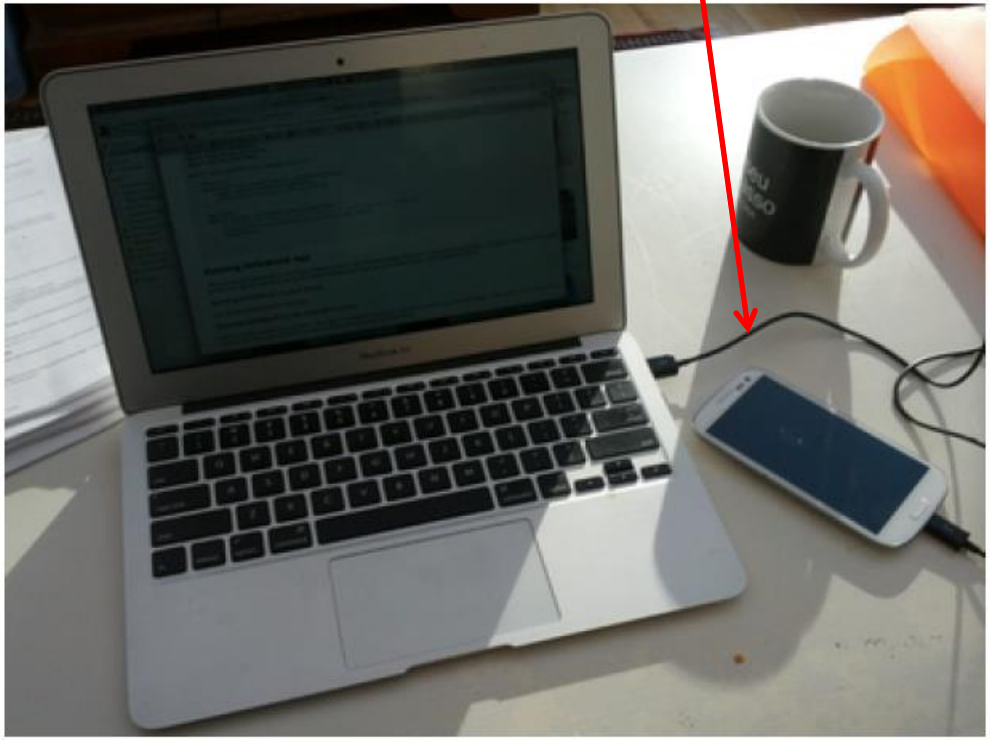
- Android app can run on:
  - Real phone (or device)
  - Emulator (software version of phone)

Emulated phone in Android Studio



Running Android App on Real Phone

- Need USB cord to copy app from development PC to phone



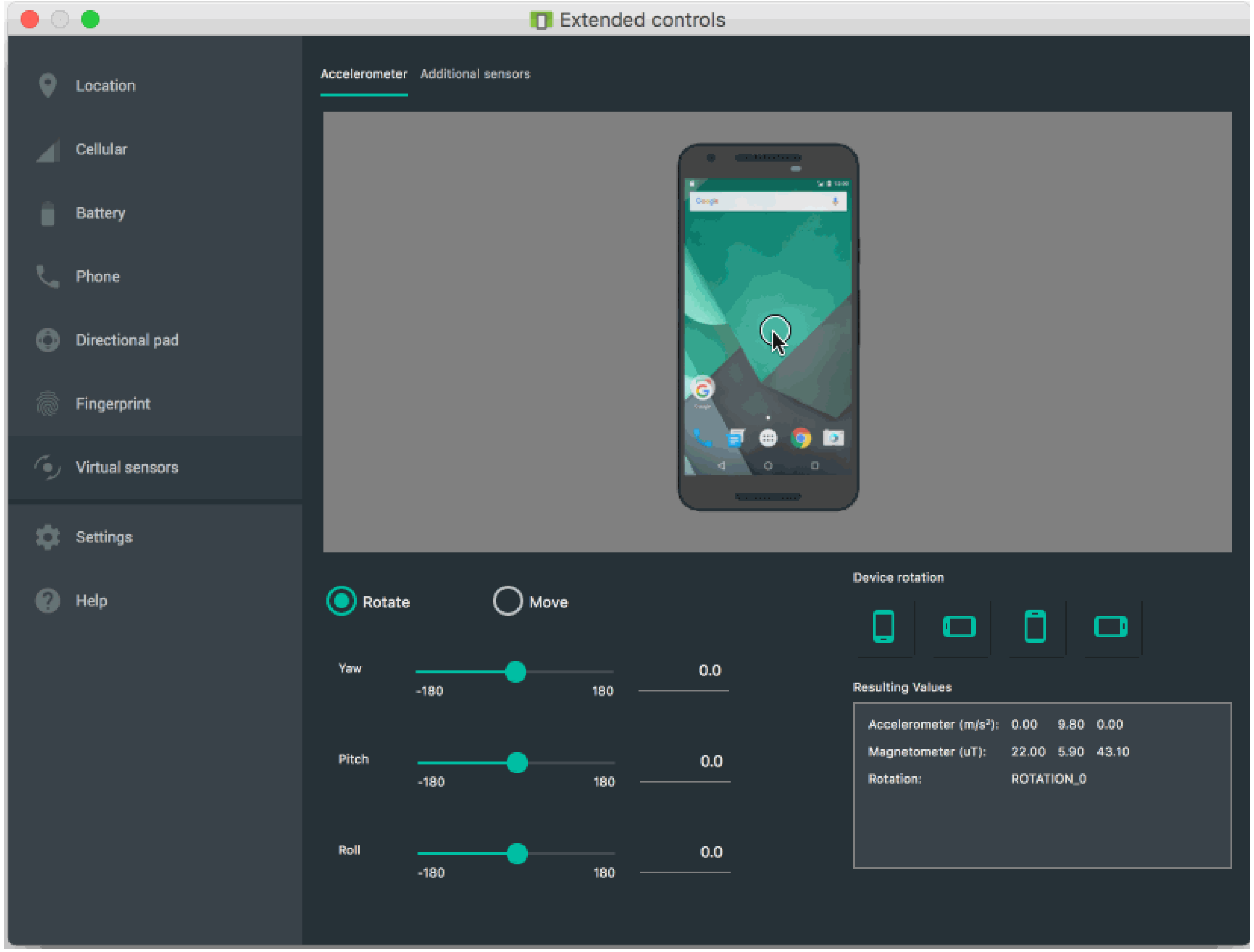
Emulator Pros and Cons (Vs Real Phone)

- Pros:
  - Conveniently test app on basic hardware by clicking in software
  - Easy to test app on various emulated devices (phones, tablets, TVs, etc), various screen sizes

- Cons:
  - Limited support, access to hardware, communications, sensors
  - E.g. GPS, camera, video recording, making/receiving phone calls, Bluetooth devices, USB devices, battery level, sensors, etc
  - Slower than real phone

New Support for Sensors

- Can now emulate some sensors (e.g. location, accelerometer), but still limited



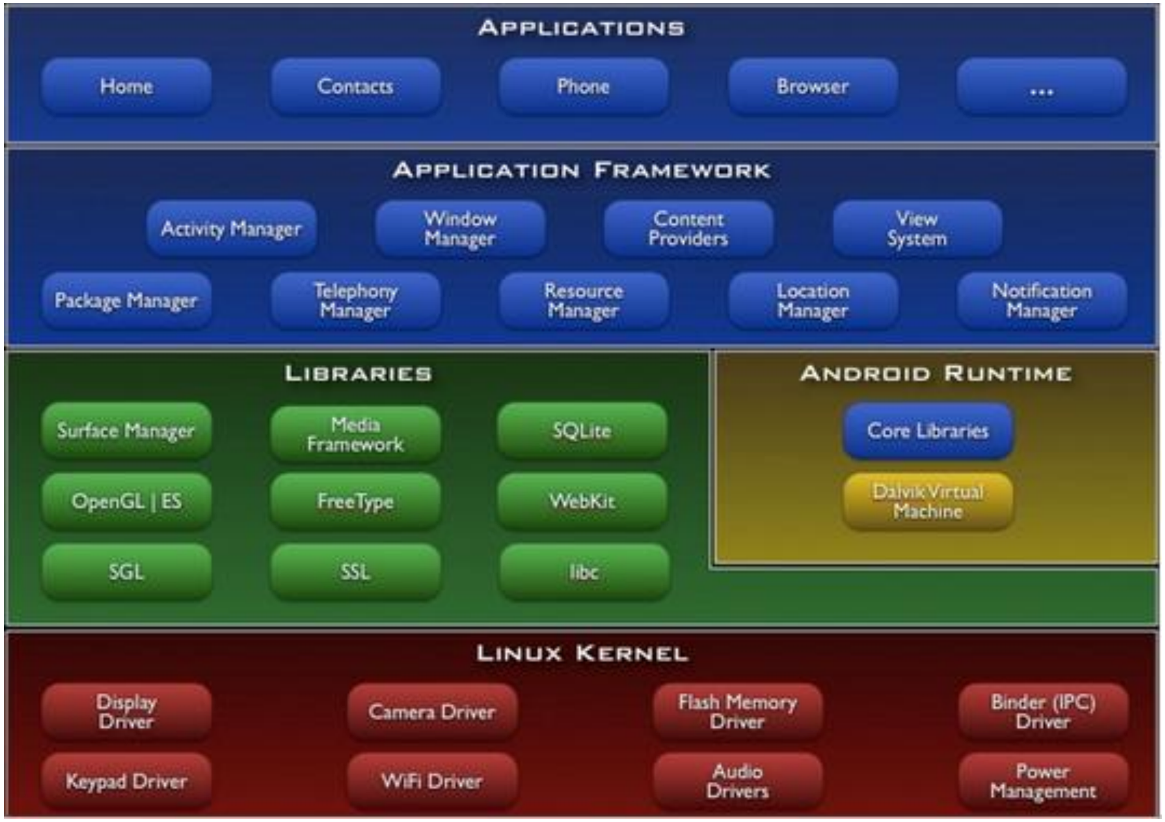
Android Software Framework

Android Functionality as Apps

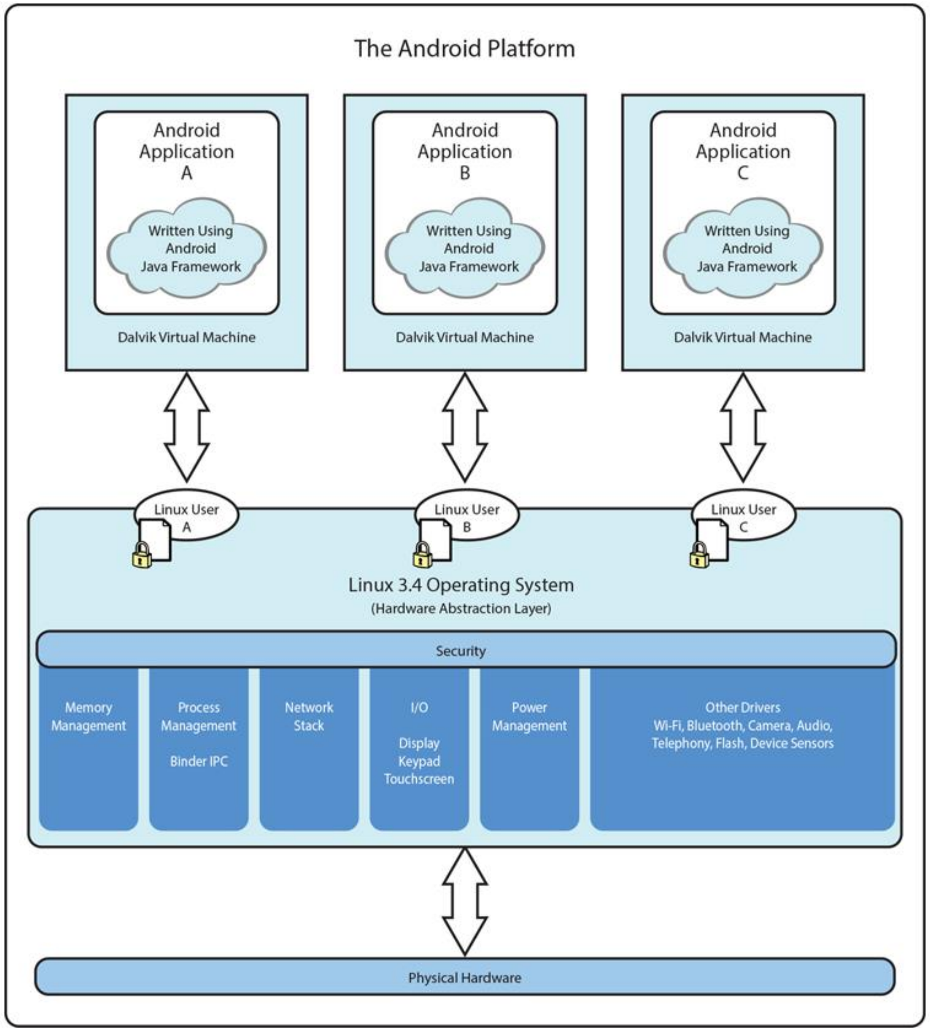
- Android functionality: collection of mini-applications (apps)
- Even dialer, keyboard, etc

Android Software Framework

- **OS:** Linux kernel, drivers
- **Apps:** programmed & UI in Java
- **Libraries:** OpenGL ES (graphics), SQLite (database), etc



Android Software Framework



- Each Android app runs in its own security sandbox (VM, minimizes complete system crashes)
- Android OS multi-user Linux system
- Each app is a different user (assigned unique Linux ID)
- Access control: only process with the app's user ID can access its files

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

References

- Android App Development for Beginners videos by Bucky Roberts (thenewboston)
- Ask A Dev, Android Wear: What Developers Need to Know, https://www.youtube.com/watch?v=zTS2NZpLyQg
- Ask A Dev, Mobile Minute: What to (Android) Wear, https://www.youtube.com/watch?v=n5Yjzn3b_aQ
- Busy Coder's guide to Android version 4.4
- CS 65/165 slides, Dartmouth College, Spring 2014
- CS 371M slides, U of Texas Austin, Spring 2014