Born on 23rd September 2008

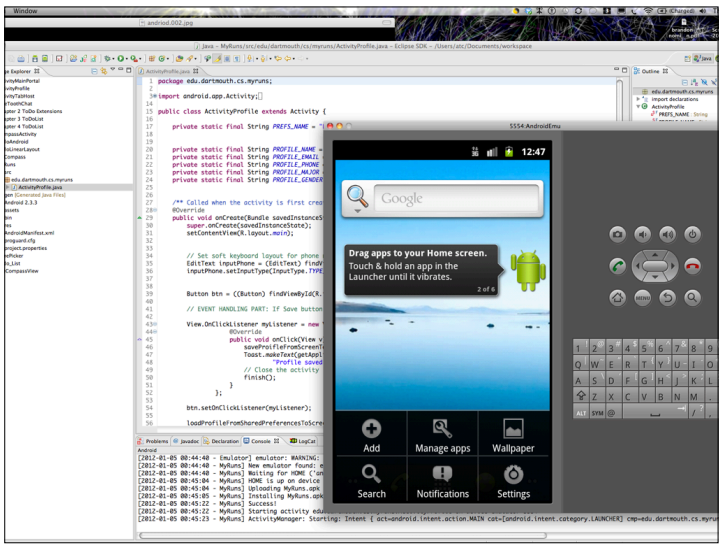
hola android!

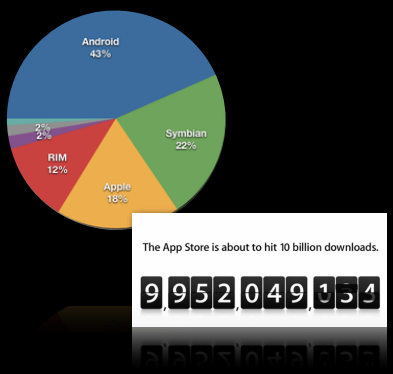
what beauty

Born on 9 January 2007

my first love

You don't need to own an android phone to do the class!





some factoids

What's in a smartphone - the hardware inside?

And, why is it such a revolutionary device?

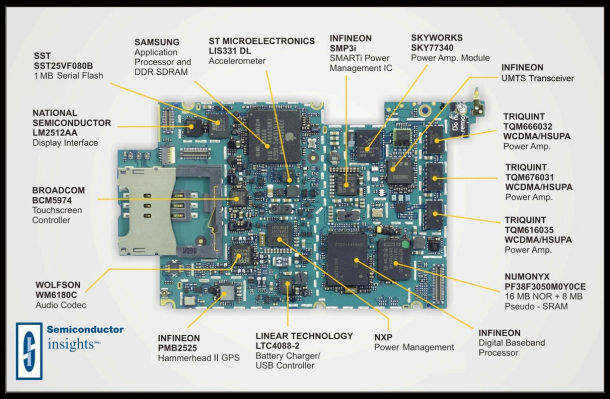
What software runs on phones?

How to I code apps and distribute them to the masses?

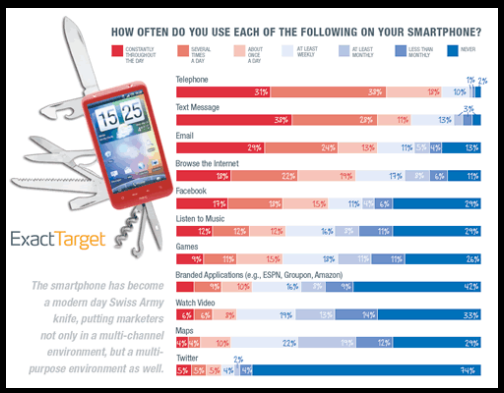
Oh, and I do I get rich... quickly.

Where is does the vision lead?

what's under the hood?



The naked iPhone 3GS



Things you can do

what's your favorite app?

Simple Multi-Tasking

I hings you can do

what makes things different is the embedded sensors

accelerometer

emotion

gyroscope

air quality

compass

near-field

barometer

proximity

light

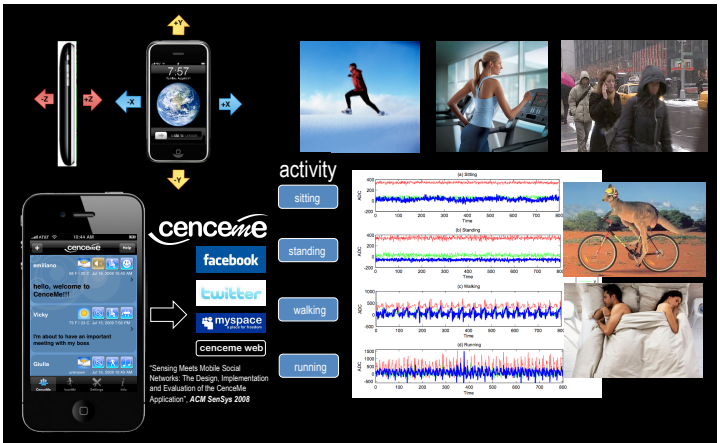
radios

microphone

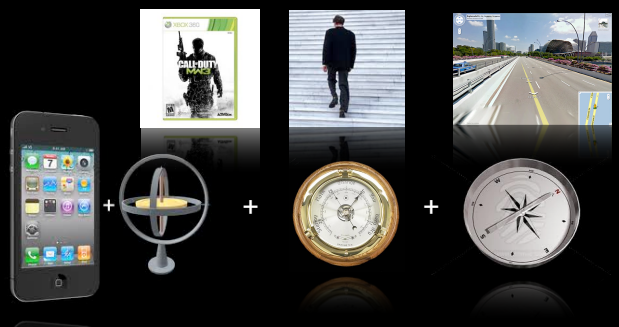
location

front/back cameras

Nokia workshop on large-scale sensor networks, February 2005, Kuusamo, Finland

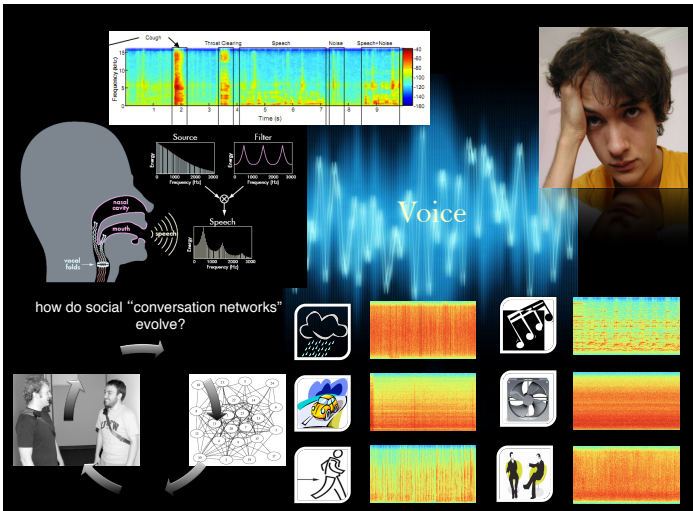


accelerometer



gyroscope, barometer, compass

two microphones



SoundSense

Hong Lu, Wei Pan, Nicholas D. Lane, Tanzeem Choudhury, Andrew T. Campbell

SoundSense

location (GPS, wifi, cellular)

front and back cameras

Face Unlock

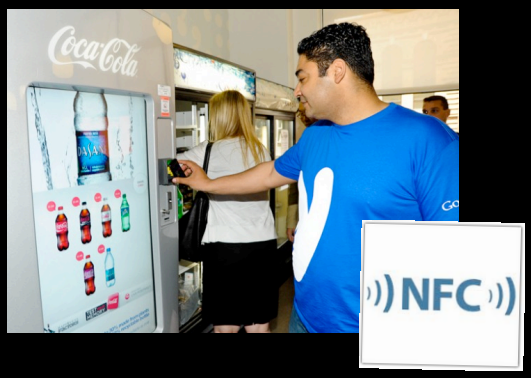
WalkSafe:

a pedestrian safety app for mobile users who walk and talk while crossing roads

Tianyu Wang, Giuseppe Cardone,
Antonio Corradi, Lorenzo Torresani,
Andrew T. Campbell

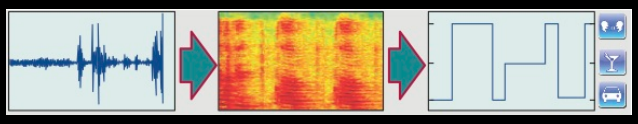
Dartmouth College University of Bologna

collocation nets (BlueTooth)



near-field communications (NFC)

Beam My Stuff



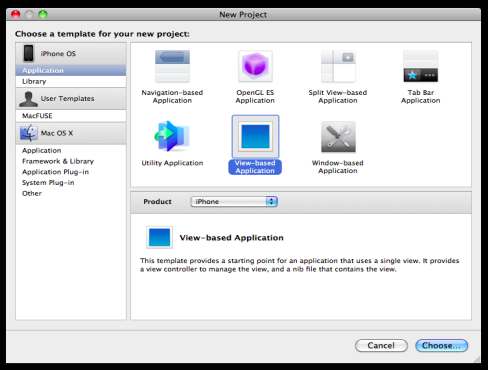
raw sensor data

extracted features

classification inference

making sense of data

where does the vision lead?



phones are open and programmable

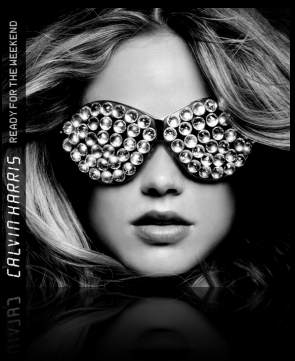
"massive markets driving innovation - some 15 year old will release the equivalent of facebook for phones in the next 3 years", Andrew T. Campbell, January 5, 2011

In your lifetime ...

smartphones are getting smarter; at some point they'll:

-understand our behavioral patterns
-mimics human perception
-anticipate our every move
-help us navigate our day
-become integrated into the fabric of our lives

ultimately leading to the "cognitive phones".



things will look different

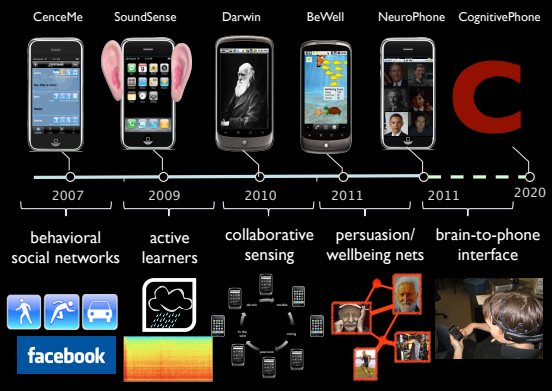
Brain to Mobile Phone Interface

Andrew T. Campbell, Tanzeem Choudhury, Shaohan Hu, Hong Lu,
Matthew K. Mukerjee*, Mashfiqui Rabbi, Rajeev D. S. Raizada

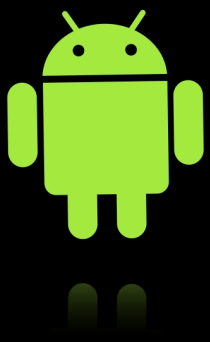
"NeuroPhone"

* Contact Author

we'll interact with smartphones in new ways



smartphone sensing group



Let's take a look at our little green friend's software

factoids on androids ;-)

Android applications are written in the Java

The Android SDK tools compile the code—along with any data and resource files—into an Android package (file.apk) which is considered to be one application and is the file that Android devices use to install the application.

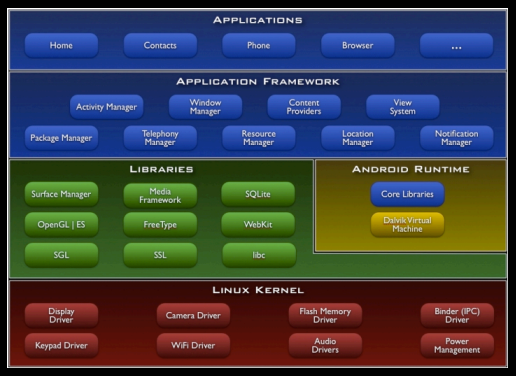
Once installed on a device, each Android application lives in its own security sandbox:

- The Android operating system is a multi-user Linux system in which each application is a different user.
- By default, the system assigns each application a unique Linux user ID (the ID is used only by the system and is unknown to the application). The system sets permissions for all the files in an application so that only the user ID assigned to that application can access them.
- Each process has its own virtual machine (VM), so an application's code runs in isolation from other applications.
- By default, every application runs in its own Linux process. Android starts the process when any of the application's components need to be executed, then shuts down the process when it's no longer needed or when the system must recover memory for other applications.

This creates a very secure environment in which an application cannot access parts of the system for which it is not given permission

There are ways for an application to share data with other applications and for an application to access system services; for example, an application can request permission to access device data such as the GPS, user's contacts, SMS messages, the mountable storage (SD card), camera, Bluetooth, and more.

All application permissions must be granted by the user at install time.



android architecture

Android 4.0, Ice Cream, API 14 Sandwich

API numbers and OS targets

Each version of the Android OS is identified by an API level number:

Two separate targets are applicable:

•Android SDK Platform contains all the Android APIs
•Google APIs by Google Inc contains all the Android APIs and Google Maps APIs

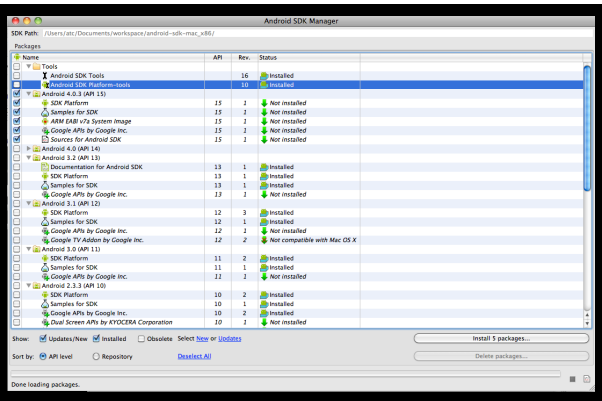
Platform Version	API Level	VERSION_CODE	Notes
Android 4.0.3	15	ICE_CREAM_SANDWICH_MR1	Platform Highlights
Android 4.0, 4.0.1, 4.0.2	14	ICE_CREAM_SANDWICH	
Android 3.2	13	HONEYCOMB_MR2	
Android 3.1.x	12	HONEYCOMB_MR1	Platform Highlights
Android 3.0.x	11	HONEYCOMB	Platform Highlights
Android 2.3.4 Android 2.3.3	10	GINGERBREAD_MR1	Platform Highlights
Android 2.3.2 Android 2.3.1 Android 2.3	9	GINGERBREAD	
Android 2.2.x	8	FROYO	Platform Highlights
Android 2.1.x	7	ECLAIR_MR1	Platform Highlights
Android 2.0.1	6	ECLAIR_0_1	
Android 2.0	5	ECLAIR	
Android 1.6	4	DONUT	Platform Highlights
Android 1.5	3	CUPCAKE	Platform Highlights
Android 1.1	2	BASE_1_1	
Android 1.0	1	BASE	

Android 1.0, the first commercial version of the software, was released on 23 September 2008

getting started: installing the platform

See http://developer.android.com/sdk/installing.html

Step 1: You might need to install the JDK
Step 2: You might need to install Eclipse Classic
http://www.eclipse.org/downloads/
Step 3: Install Android SDK
Step 4: Install Android Development Tools (ADT) custom plugin for the Eclipse IDE



step 5: adding platforms and other components

Name	Description
add-ons/	Contains add-ons to the Android SDK development environment, which let you develop against external libraries that are available on some devices.
docs/	A full set of documentation in HTML format, including the Developer's Guide, API Reference, and other information. To read the documentation, load the file offline.html in a web browser.
platform-tools/	Contains platform-dependent development tools that may be updated with each platform release. The platform tools include the Android Debug Bridge (adb) as well as other tools that you don't typically use directly. These tools are separate from the development tools in the tools/ directory because these tools may be updated in order to support new features in the latest Android platform.
platforms/	Contains a set of Android platform versions that you can develop applications against, each in a separate directory.
<platform>/	Platform version directory, for example "android-11". All platform version directories contain a similar set of files and subdirectory structure. Each platform directory also includes the Android library (android.jar) that is used to compile applications against the platform version.
samples/	Sample code and apps that are specific to platform version.
tools/	Contains the set of development and profiling tools that are platform-independent, such as the emulator, the Android SDK and AVD Manager, ddms, hierarchyviewer and more. The tools in this directory may be updated at any time using the Android SDK and AVD Manager and are independent of platform releases.
SDK Readme.txt	A file that explains how to perform the initial setup of your SDK, including how to launch the Android SDK and AVD Manager tool on all platforms.
SDK Manager.exe	Windows SDK only. A shortcut that launches the Android SDK and AVD Manager tool, which you use to add components to your SDK.

application components

activities

An *activity* represents a single screen with a user interface. For example, an email application might have one activity that shows a list of new emails, another activity to compose an email, and another activity for reading emails

services

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

content providers

A *content provider* manages a shared set of application data. You can store the data in the file system, an SQLite database, on the web, or any other persistent storage location your application can access. Through the content provider, other applications can query or even modify the data (if the content provider allows it).

broadcast receivers

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. Applications can also initiate broadcasts—for example, to let other applications know that some data has been downloaded to the device and is available for them to use

the manifest file

Before the Android system can start an application component, the system must know that the component exists by reading the application's AndroidManifest.xml.

Your application must declare all its components in this file.

The manifest does a number of things in addition to declaring the application's components, such as:

-Identify any user permissions the application requires, such as Internet access or read-access to the user's contacts.

-Declare the minimum API Level required by the application, based on which APIs the application uses.

-Declare hardware and software features used or required by the application, such as a camera, bluetooth services, or a multitouch screen.

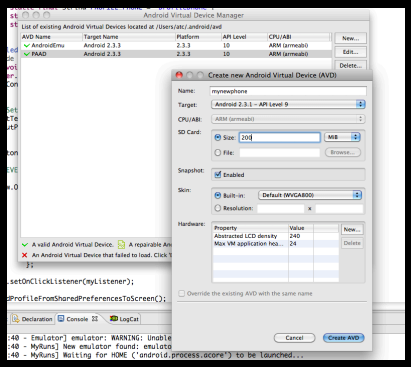
-API libraries the application needs to be linked against (other than the Android framework APIs), such as the Google Maps library.

create android virtual devices (AVD)

AVD is an emulator instance that enables you to model an actual device

- Consists of a hardware profile
- Mapping to a phone limitations (e.g., screen size, cellular wireless speeds)
- Emulated storage, such as Secure Digital (SD) card, etc.

You can create as many AVDs as you like
Use different AVDs to test your applications under different scenarios



application resources

An Android application is composed of more than just code—it requires resources that are separate from the source code, such as images, audio files, and anything relating to the visual presentation of the application.

You should define animations, menus, styles, colors, and the layout of activity user interfaces with XML files.

Using application resources makes it easy to update various characteristics of your application without modifying code and—by providing sets of alternative resources—enables you to optimize your application for a variety of device configurations (such as different languages and screen sizes).

For every resource that you include in your Android project, the SDK build tools define a unique integer ID, which you can use to reference the resource from your application code or from other resources defined in XML. For example, if your application contains an image file named logo.png (saved in the res/drawable/ directory), the SDK tools generate a resource ID named R.drawable.logo, which you can use to reference the image and insert it in your user interface.

Hello, world

Create the Project

Construct the UI

Run the Code

Upgrade the UI to an XML Layout

Debug Your Project

course

papers + coding

MyRuns app

Lab 0 - Set up environment do Hello World, ++

Lab 1 - Construct the UI

Lab 2 - Database (SQLite) and adapters

Lab 3 - Maps, location (GPS) and services

Lab 4 - Motion sensors and exercising stats

Lab 5 - Cloud side using App Engine

group projects

collaborative working

no quizzes, just code

cool apps, jokes .. yes.

and most importantly