Mobile- and Web-based Software

Lecture 1: Introduction

David Sik
siktdavid@gmail.com

BME AUT
Department of
Automation and
Applied Informatics

Requirements

- 3+3 small assignments. (20%)
- Large assignment. (20%)
- Midterm test. (20%)
- Endterm exam. (40%)
- Grades: 40-55-70-85% (2-3-4-5)
- Minimum requirement for signature:
  - Each item (+ each assignment) over 40%.
- Minimum requirement for grade:
  - Signature + exam over 40%
- Offered mark (no exam needed): all item over 85%

Requirements

- Assignments with deadlines
- A preliminary specification for the large assignment.
- Android Studio need to be installed to do the assignments and for practicing.
- Android smartphone is not required.
- The projects shall uploaded to the AUT website.
- More info will handed out with the assignments.

Info

- Mobile-based software
  - 1-7th week
- Web-based software
  - 8-14th week

Info, Mobile-based software

- David Sik, siktdavid@gmail.com (with any questions)
- Lectures, practices:
  - Tuesday 10:15-11:45 QBF15
  - Thursday 10:15-11:45 QBF14
- No lessons (some Thursday):
  - 14 September, 28 Sepetember
  - 5 October, (19 October?)
- Lectures with demos and coding together
- https://github.com/daveflat/MobWeb2017
- If you can, bring notebook with Android Studio, Android smartphone, USB cable

Question time

Mobile-based Software

Mobile Devices

- What is a *mobile device*?
  - Earlier: device for radio network based voice communication
  - Nowadays: miniaturized personal computer (smartphone, tablet, smartwatch…) with high processing power, loads of sensors sensors and networking capabilities

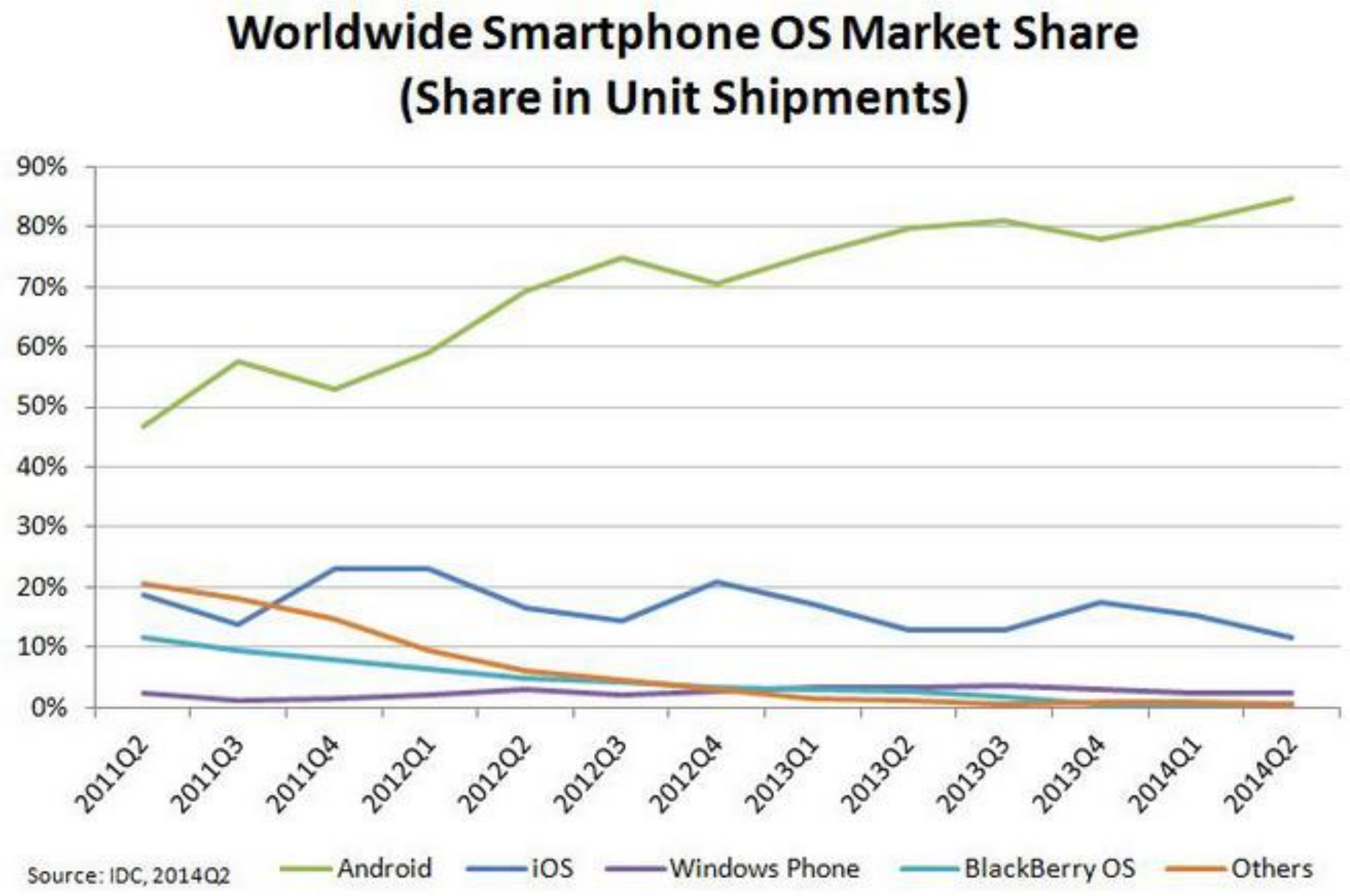
Characteristics of Mobile Devices

- Main characteristics:
  - Battery constrained
  - Have a lot of different sensors and peripherals
  - Wireless network communication
  - Smaller form factor and various screen sizes
  - Touch-based user interface
  - Limited hardware capabilities (compared with a standard desktop/laptop computer)
- Mobile Software Development: creating software for mobile devices via different SDKs and programming languages

Platforms

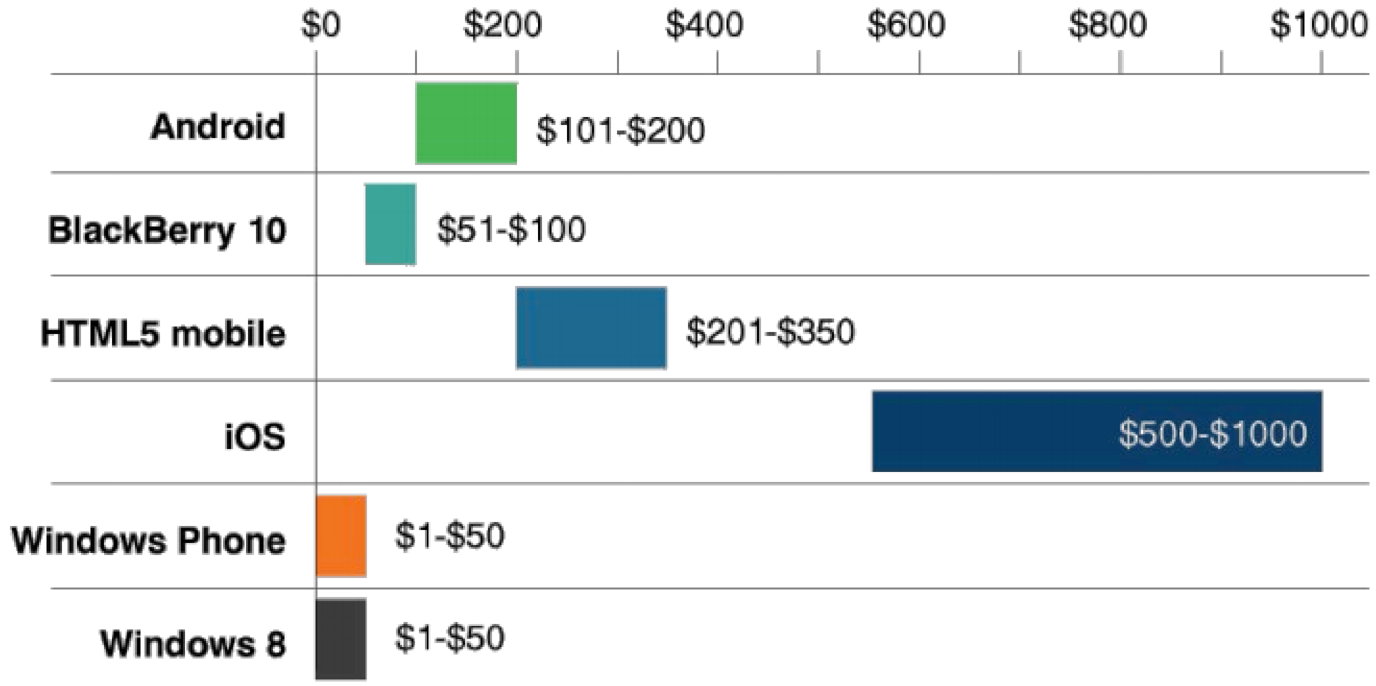
- Current major mobile platforms
  - Android
  - iOS
  - Windows
- Lots of minor platforms

Platform Market Share



Revenue per Platform

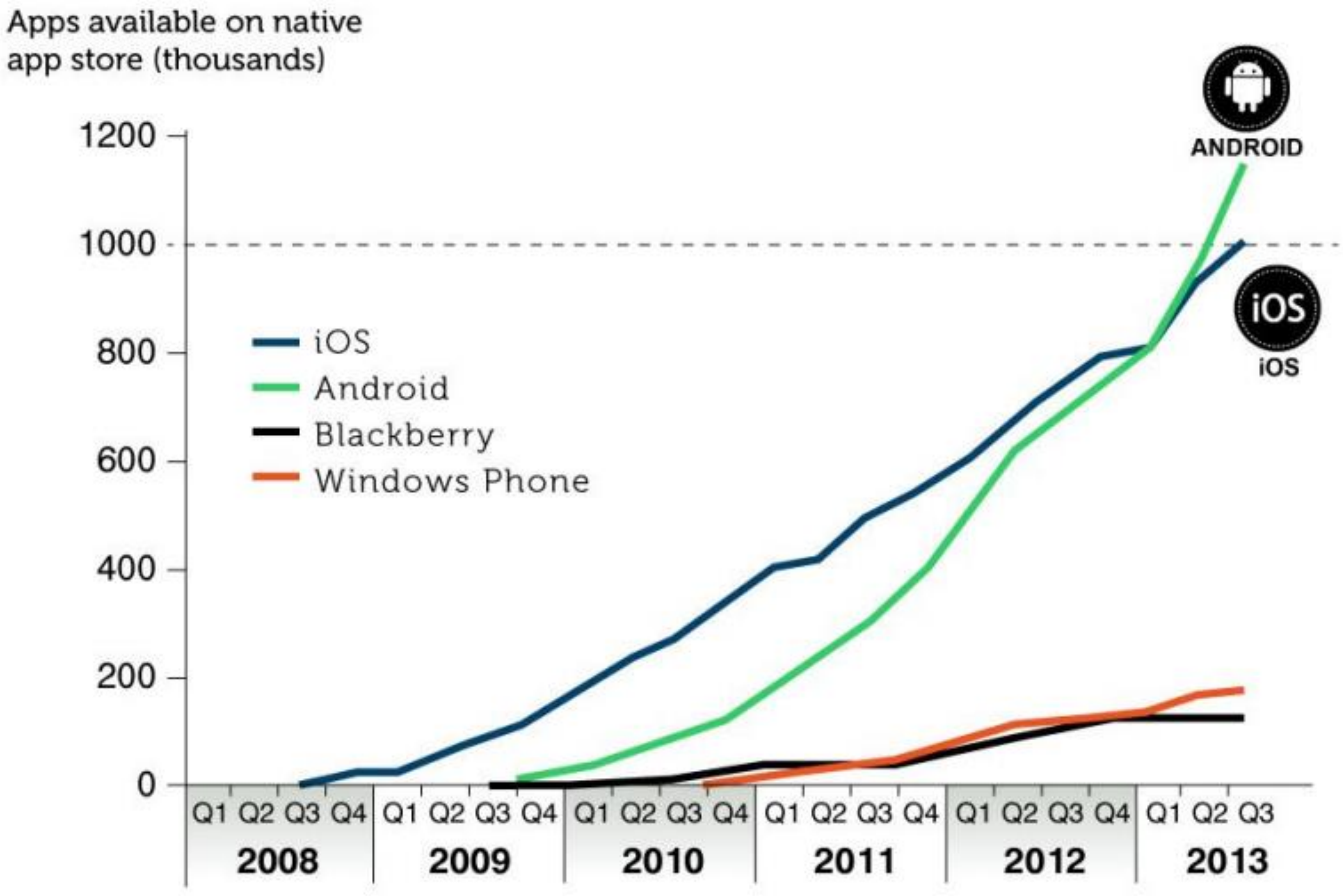
Median revenue per app, per month (n=2,425)*



As most developers use more than one platform, besides their primary platform, part of these revenues may be generated on platforms other than the primary. However they are indicative of the revenue potential of each platform. These figures exclude developers who are not interested in generating revenue.

Source: Developer Economics Q1 2014

Apps in App Stores



Data: VisionMobile estimates

Mobile Software Development

iOS

What devices run iOS?

- iPhone
- iPod Touch
- iPad
- Apple Watch
- Apple TV

iOS as an Operating System

- Based on the same core as OS X
  - Most of the kernel (XNU) code is shared (UNIX/BSD components)
  - iOS and OS X applications are not binary compatible
- Closed system, restrictions apply to how and what applications can be created for iOS
  - Application development is only possible on OS X

Why It Became a Hit

- Novel UI
  - Optimized for touch screens
  - Multitouch
  - UI speed is top priority
- First really popular/usable mobile web browser: Safari
- App Store model: centralized application distribution model
  - All applications are accessible at one place
  - Only small initial investment is needed
- All iOS devices have a unified UI and similar capabilities

Shortcomings/problems of the Platform

- Somewhat restricted APIs, limited background execution and inter-app communication
- Development is only possible on OS X
- Wired connection only via iTunes (no USB mass storage mode)
- Applications can only be acquired from the Apple App Store
  - Apple decides what applications are allowed; lot of criticism
- To get full control, you need to jailbreak your device
- Storage is not extendable
- Connecting to external devices/accessories only with a license (except for Bluetooth Low Energy since iPhone 4S)
- Only premium devices (Android is available for all device segments)

Brief History of iOS Devices

- June 2007: release of iPhone 1.0
- March 2008: the iPhone SDK is released
  - Developing 3rd party apps became possible
- July 2008: iPhone 3G and iOS 2.0 is released, App Store opens
- April 2010: iPad is released (the OS is renamed to iOS from „iPhone OS")
- April 2015: Apple Watch is released
- Currently iPad Air 2, iPhone 7 and 7 Plus
- iOS 11 is here

Apple Watch

- You need an iPhone to use an Apple Watch
- Every WatchKit app must have a companion iOS app
  - > The UI is running on the watch
  - > The logic of the app runs on your iPhone
- Later we might get “independent” Apple Watch apps which run entirely on the watch

What You CANNOT do in your iOS app (in general)

- Keep the app running continously (as a service)
- Access a shared storage on the disk to share data between different apps
- Control phone calls (besides initiating a call)
- Connect to external devices using IR, Bluetooth 3.x or USB/wire
  - Only after applying to the “**Made for iPhone**” program
- Launch an app without any user interaction (on device startup or push notification)
- …

What You CAN do in iOS apps

- Wake up your app when it enters a location (geofencing) or receives a push notification
    - To show the UI, user interaction is still needed
- Launch other apps from your app
- Create custom keyboards
- Create “widgets” (Home Screen Extensions)
- Connect to Bluetotth Low Energy devices
- Vibrate the phone
- Create fancy visual effects using physical simulation and 3D graphics either using low level (Open GL) or high level (UIKit) APIs
- …

iOS Native Applications

- **Native**: written in Swift or Objective-C, compiled to machine code
  - iOS devices use ARM CPUs
- Great performance, no virtual machine or on-device compilation is required

iOS Development without Swift or Objective-C 1/2

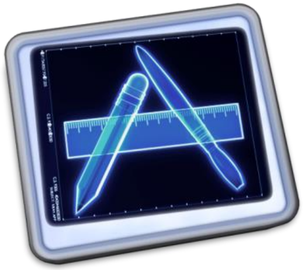
- Xamarin.iOS / MonoTouch
  - C# / .NET
  - Ahead-Of-Time (AOT) compilation: IL code is compiled on the development computer to iOS binaries (no JIT or interpreter)
  - Binding to most Cocoa Touch APIs, e.g. MonoTouch.UIKit.UIButton
  - .NET garbage collection, .NET class libraries (Web Services, System.Xml, System.Threading, etc.)
  - Development still requires OS X

iOS Development without Swift or Objective-C 2/2

- Phonegap
  - HTML/CSS/JavaScript
  - Application is hosted in a Web View
  - Platform specific features are accessed via a JS library
  - Standard web based JS libraries can also be used (jQuery)
  - iOS/Android/Windows Phone/Windows 8/…
- Appcelerator Titanium
  - JavaScript
  - Precompiled JS code executed by an interpreter, generates native UI
  - iOS/Android/Blackberry (no WP)

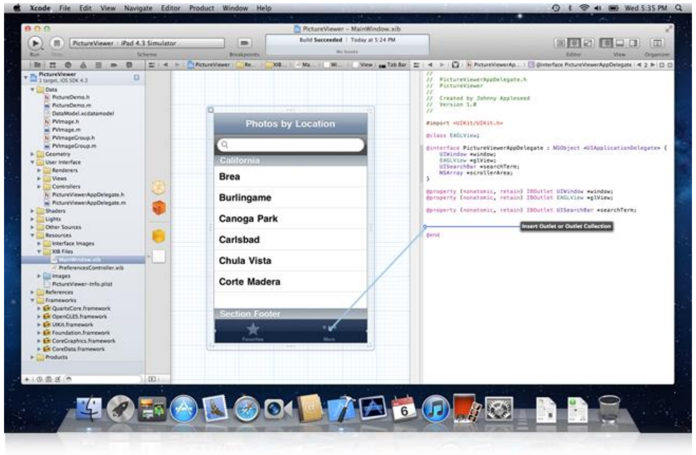
Development Tools

- Xcode (downloadable from OS X App Store), contains everything
    - Xcode app: IDE, debugger, interface builder
    - iOS Simulator
    - Instruments: profiling tool, analyzes application during running (e.g. finding memory leaks)
- Xcode is free and updated with each new iOS version



Xcode the IDE

- Editor
- Interface builder
- Compiler: LLVM (Clang)
- Debugger: LLDB
- Used for both iOS and OS X development



```
//
//  PictureViewerAppDelegate.h
//  PictureViewer
//
//  Created by Johnny Appleseed
//  Version 1.0
//

#import <UIKit/UIKit.h>

@class EAGLView;

@interface PictureViewerAppDelegate : NSObject <UIApplicationDelegate> {
    UIWindow *window;
    EAGLView *glView;
    UISearchBar *searchTerm;
    NSArray *scrollerArea;
}

@property (nonatomic, retain) IBOutlet UIWindow *window;
@property (nonatomic, retain) IBOutlet EAGLView *glView;

@property (nonatomic, retain) IBOutlet UISearchBar *searchTerm;


@end
```

Xcode File Edit View Navigate Editor Product Window Help
Wed 5:35 PM
PictureViewer – MainWindow.xib
Build Succeeded | Today at 5:24 PM
Editor
View
Organizer
Photos by Location
California
Brea
Burlingame
Canoga Park
Carlsbad
Chula Vista
Corte Madera
Section Footer
Insert Outlet or Outlet Collection

OS X vs. iOS Development

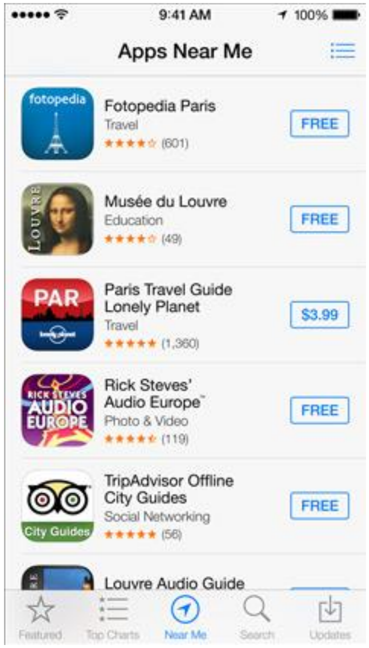
- Lots of common APIs, but also several differences
  - Cocoa vs. Cocoa Touch
    - iOS lacks several classes or classes are dumbed down
    - Separate documentation, separate example code
  - 3D graphics: OpenGL vs. OpenGL ES
  - iOS uses (automatic) reference counting instead of garbage collection
- The two worlds are getting closer and closer
  - OS X apps are now also sandboxed
- Separate developer program
  - OS X apps can be distributed outside the Mac App Store without developer program access

Distributing Apps

- Apple App Store
  - Main distribution channel
- Ad-Hoc
  - Maximum 100 devices, restricted by UDID
  - Installer package (.IPA file) can be shared via email/websites/etc. Only devices with the given UDIDs can install the app
- Enterprise
  - In-house distribution for companies; similar to ad-hoc distribution but no limit on device number

App Store

- Distribution channel for apps, accessed via the device/iTunes or web
  - Developer can pick the price
  - 70% of all sales revenue goes to developer
  - Apps must comply with the App Store rules
    - All apps are evaluated by Apple before they are released to the public (~ 5 days)
- Developer must sign up for the Apple Developer Program
  - $99 / year



Apps Near Me

Fotopedia Paris
Travel
★★★★☆ (601)
FREE

Musée du Louvre
Education
★★★★☆ (49)
FREE

Paris Travel Guide Lonely Planet
Travel
★★★★★ (1,360)
$3.99

Rick Steves' Audio Europe™
Photo & Video
★★★★★ (119)
FREE

TripAdvisor Offline City Guides
Social Networking
★★★★★ (56)
FREE

Louvre Audio Guide

Apple Services

- Push Notification
- In-App Purchase (StoreKit)
  - 70% goes to developer
- iCloud
  - Cloud storage, all Apple IDs gets 5GB for free
  - API: file, settings, database (Core Data) storage
- iAd
  - In-app ads (similar to Google AdMob)
  - 70% goes to developer
  - Per click and per impression
- Game Center
  - Matchmaking, leaderboards, achievements

Mobile Software Development

Android

Origin of Android

- Android is one of the most popular mobile operating systems of present days
  - It become a brand by intent (TV spots, ads, etc.)
  - Google stands behind the platform
- Android offers the picture of a well functioning OS
- Revolutionized the picture of mobile operating systems

Android devices

- Mobile phone and tablet manufacturers
- Car navigation and multimedia systems
- Android Wear
- Industrial solutions
- Android has potential on many other fields:
  - Small and large screen
  - Limited resource capabilities (e.g. battery)
  - Touch screen support
  - Android@Home

Android in cars

Where does Android came from?

- An operating system not only for mobile phones!
- The collection of all devices that support Android
- It is in the hands of Goolge
- **2005**: Google buys a company, called: *Android Incorporated*
- **2007**: The first news have been leaked that Google will enter into the mobile market
- **November 5, 2007:** The newly formed Open Handset Alliance has introduced the Android platform
- **End of 2008:** The HTC G1 (Dev Phone) was available on the market by T-Mobile

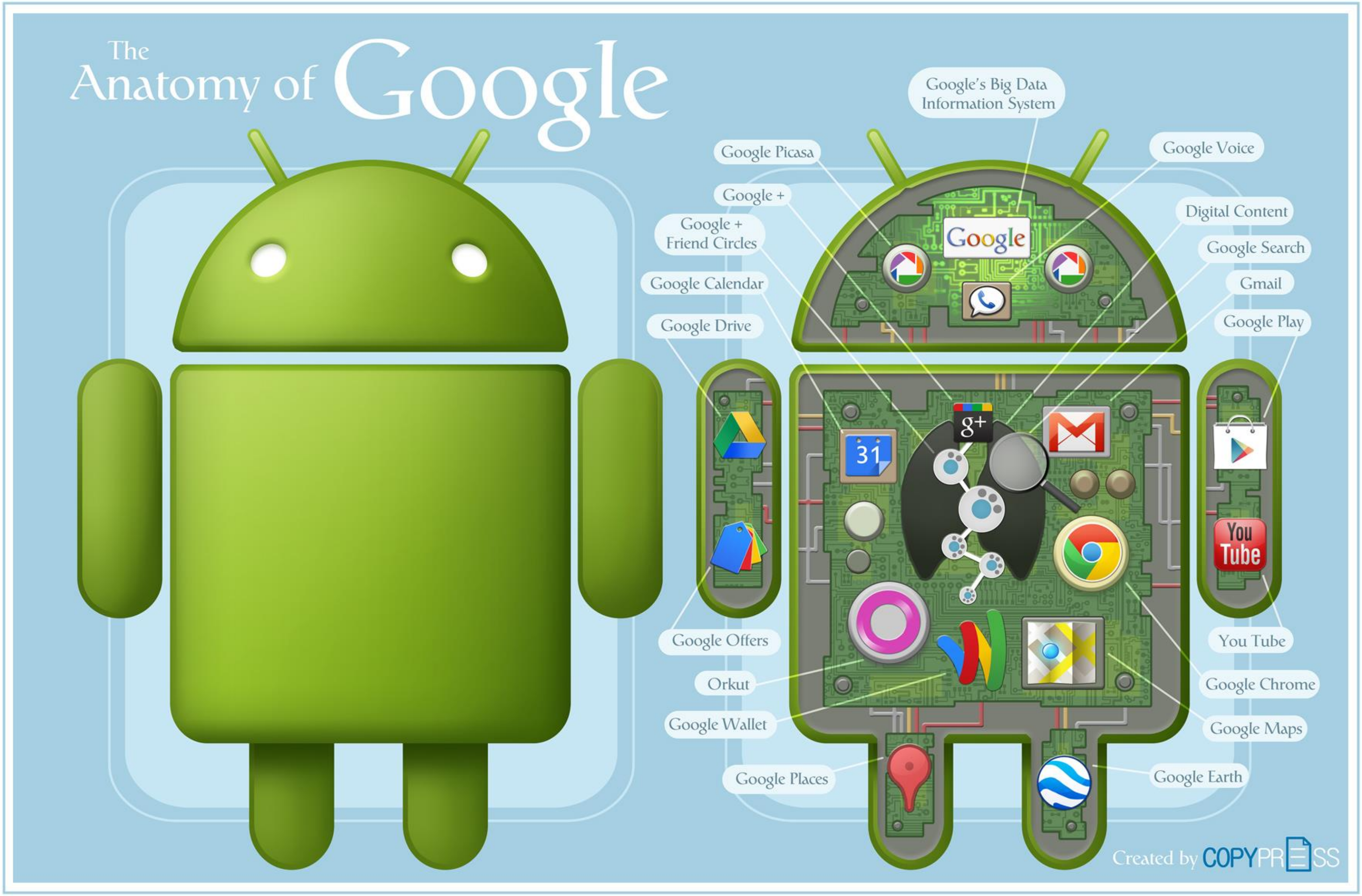
Open Handset Alliance

- Android is not only Google’s development
  - More than 50 member companies
  - 11 manufacturer (HTC, LG, Motorola, Samsung, Asus, Garmin, Huawei, Sony Ericsson, Ericsson, Toshiba, Acer)
  - 13 semiconductor manufacturer (not at least Intel, Marvell, nVidia, Broadcom, SiRF, ARM and Atheros)
  - 10 operator (example T-Mobile and Vodafone)
  - 12 software company (Google, eBay)
  - 7 other corporation
- Members are changing frequently
- Android:
  - Product of huge knowledge and lot of work

Why Android become successful?

- User experience (iPhone like…)
- Several devices (high end and low end)
- Small hardware requirements
- Open source, free to use (with some limitations)
- Fast application development
- Marketing

Android and Google integration



Android: from the perspective of larger manufacturers

- Their sales were decreasing
- It was necessary to support Android for them
- They did not have resource to develop their own system (expect Samsung Bada)
- Android is free

CSL
SAMSUNG
hTC smart mobility
LG
HUAWEI
Sony Ericsson
Google
acer
MOTOROLA

Android: from the perspective of smaller manufacturers

- Easy to reach market: free and popular operating system
- Excellent value for money: open doors to the market
- New mobile brands (e.g.: Huawei, ZTE)
- Open doors to reach operator markets (e.g.: Alcatel)

And the others?

- **Apple:** own successful solution (iOS)
  - Based on their success they introduces new features on their devices, that were previously on Web (e.g.: map)
- **Microsoft**: can not support Android directly, but it is good that there is something against iOS while Microsoft develops Windows Phone further
- **Nokia**: too long lifetime for Symbian
  - Refused Android
  - Partnership with Microsoft
  - … after Microsoft?

Android versions

- Android 1.0 – 2008. October
- Android 1.1 – 2009. February
- Android 1.5 (Cupcake) – 2009. April
- Android 1.6 (Donut) – 2009. September
- Android 2.0 and 2.1 (Eclair) – 2009. October
- Android 2.2 (Froyo) – 2010. May
- Android 2.3 (Gingerbread) – 2010. December
- Android 3.0-3.2 (Honeycomb) – 2011 January-July
- Android 4.0 (Ice Cream Sandwich) – 2011. October
- Android 4.1 (Jelly Bean) – 2012. July
- Android 4.2 (Jelly Bean) – 2012. November
- Android 4.3 (Jelly Bean)
- Android 4.4 (KitKat)
- Android 5.0, 5.1 (Lollipop)
- Android 6.0 (Marshmallow)
- Android 7.0 (Nougat)

Android versions

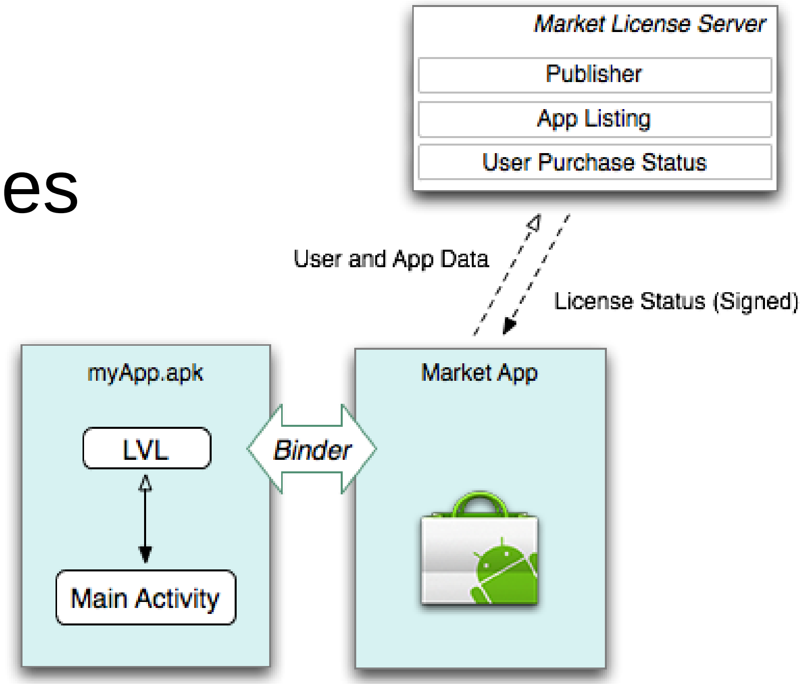
- Developers must follow version changes
- Could be significant differences between versions
- Google pursue backward compatibility, but there are deep gaps (v 3.0)
- Before developing, it's neccesary to choose the minimum supported version well
- Version codename: some candy ☺

Google Play (earlier: Android Market)

- https://play.google.com/store
- Available since 22. October, 2008.
- Application store for Android devices
- The Play Store application is not open source!
- Pre-installed on all major Android device
- One of the biggest application store nowadays

Google Play Services

- Android Licensing Service
- In-app Billing
- Google authentication
- Google+ integration
- Google Play Gaming Serivces
- Google Cloud Messaging
- Fused Location
- …



The capabilities of the Android platform 1/2

- Rich application behavior capabilities (background services, lazy connected components, easy to reach system services)
- Full JavaSE class library + rich Java API for Android functions like network handling, telco, etc.
- Rich UI widget library, easy to develop custom widgets, styles and themes
- Dynamic UI behavior, animations, rich resource capabilities

The capabilities of the Android platform 2/2

- Secure file management, SQLite and easy-to-use key-value storage
- Linux based; native development is supported
- Rich external library collection (Java and native):
  - Always check first! – GitHub, Google Code, etc.
- Rich Google Services for publishing, advertising and for development
- … **there are always new APIs, tools and libraries!**

Application behavior

- When the application starts and activates
  - Manual start (clicking on an icon)
  - Activation by some external event (incoming call, battery status, boot ready)
  - Services and contents provided by the application
- Application running type
  - Foreground running
  - Background services
- Special needs
  - Widget support
  - Live wallpaper

How do we handle these on Android?

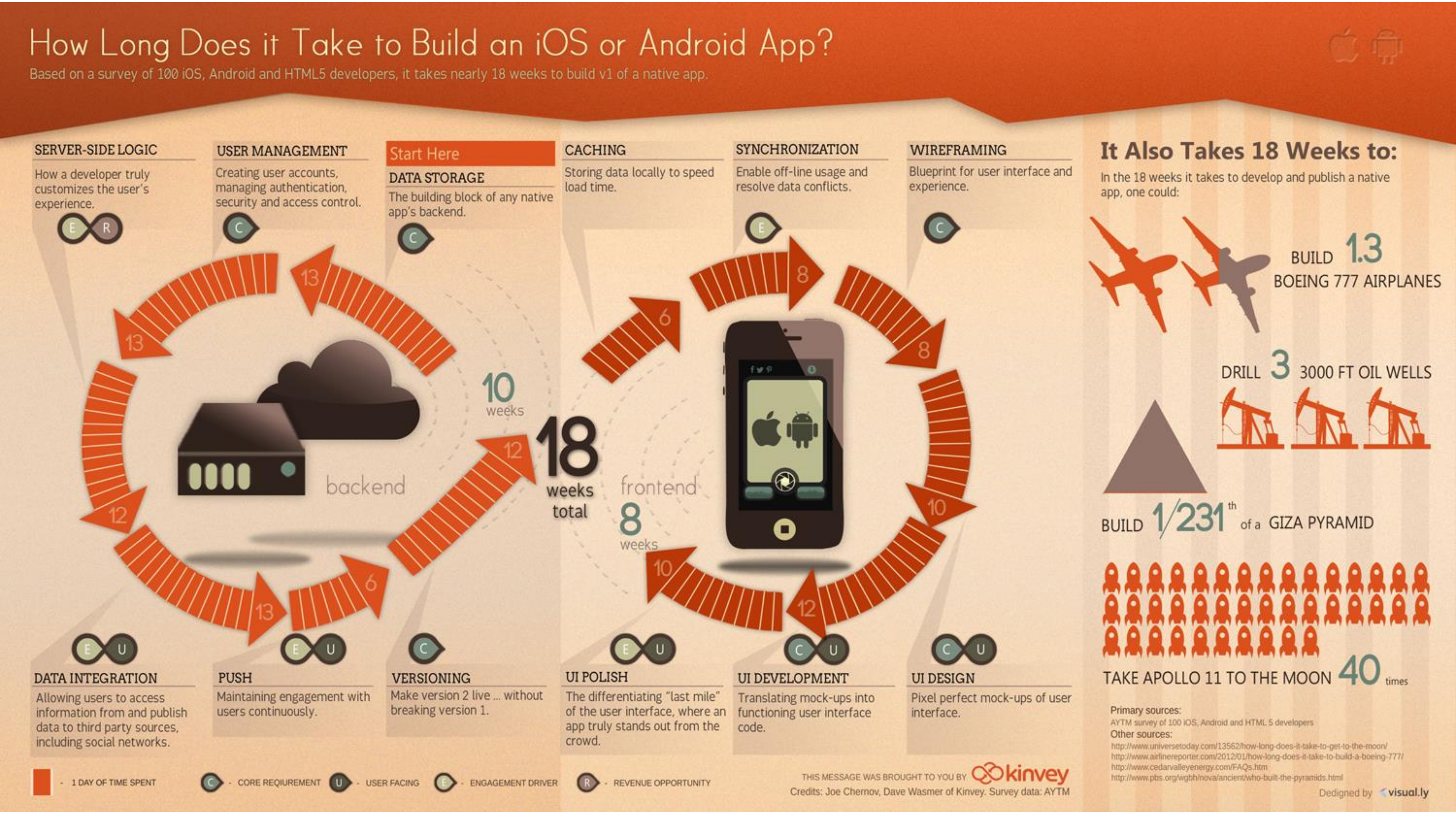
<u>Android application components</u>

Some Learning Materials

- Java
  - Head First Java
    - http://www.headfirstlabs.com/books/hfjava/
- Android programming
  - Professional Android 4 Application Development
    - http://www.wrox.com/WileyCDA/WroxTitle/Professional-Android-4-Application-Development.productCd-1118102274.html
- Android developer portal
  - http://developer.android.com/develop/index.html

Android platform structure

How long does it take to build a mobile app?



How Long Does it Take to Build an iOS or Android App?

Based on a survey of 100 iOS, Android and HTML5 developers, it takes nearly 18 weeks to build v1 of a native app.

SERVER-SIDE LOGIC
How a developer truly customizes the user's experience.

USER MANAGEMENT
Creating user accounts, managing authentication, security and access control.

Start Here

DATA STORAGE
The building block of any native app's backend.

CACHING
Storing data locally to speed load time.

SYNCHRONIZATION
Enable off-line usage and resolve data conflicts.

WIREFRAMING
Blueprint for user interface and experience.

DATA INTEGRATION
Allowing users to access information from and publish data to third party sources, including social networks.

PUSH
Maintaining engagement with users continuously.

VERSIONING
Make version 2 live ... without breaking version 1.

UI POLISH
The differentiating "last mile" of the user interface, where an app truly stands out from the crowd.

UI DEVELOPMENT
Translating mock-ups into functioning user interface code.

UI DESIGN
Pixel perfect mock-ups of user interface.

It Also Takes 18 Weeks to:

In the 18 weeks it takes to develop and publish a native app, one could:

BUILD 1.3 BOEING 777 AIRPLANES

DRILL 3 3000 FT OIL WELLS

BUILD 1/231th of a GIZA PYRAMID

TAKE APOLLO 11 TO THE MOON 40 times

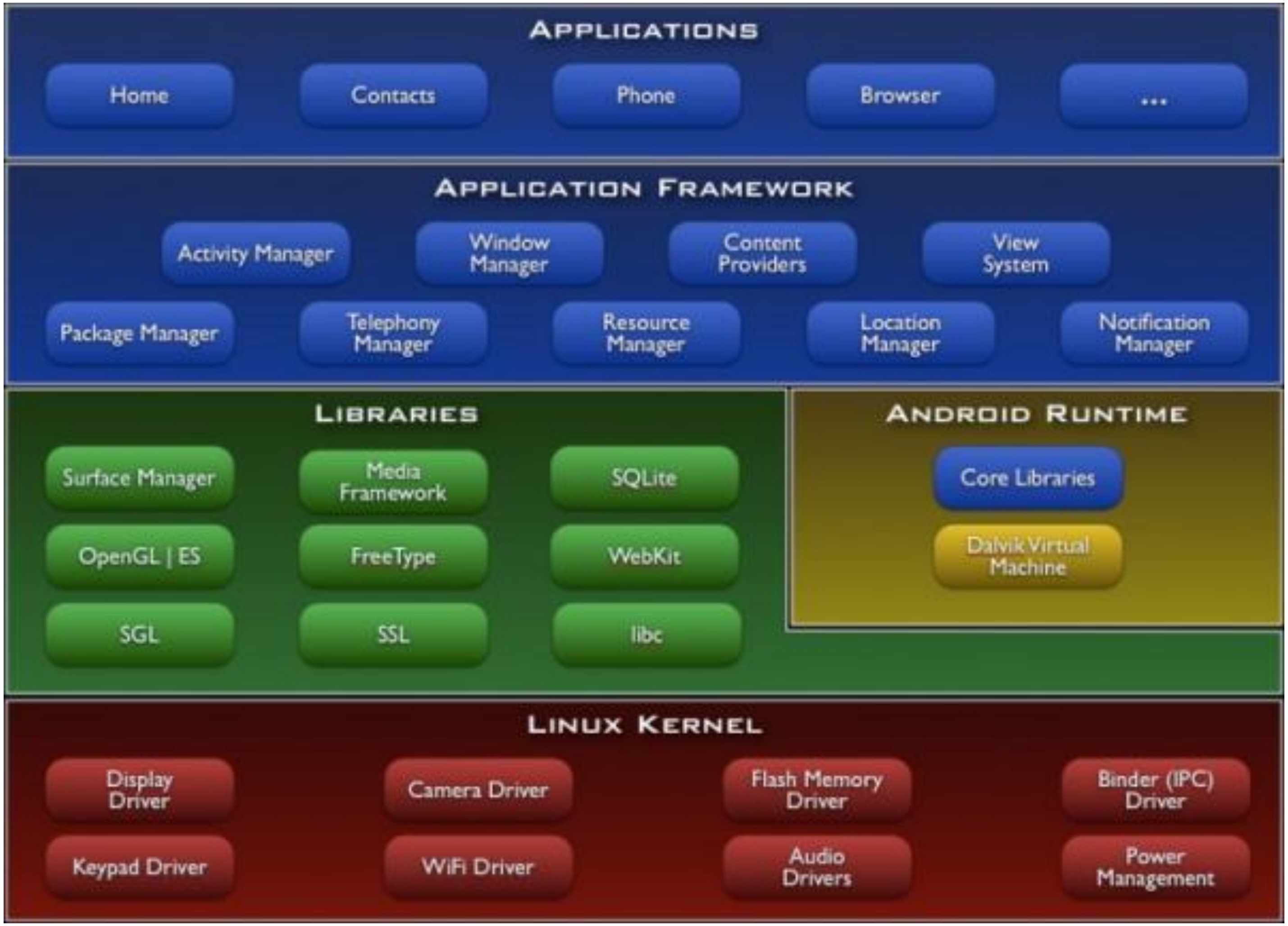
Primary sources:
AYTM survey of 100 iOS, Android and HTML 5 developers
Other sources:
http://www.universetoday.com/13562/how-long-does-it-take-to-get-to-the-moon/
http://www.airlinereporter.com/2012/01/how-long-does-it-take-to-build-a-boeing-777/
http://www.cedarvalleyenergy.com/FAQs.htm
http://www.pbs.org/wgbh/nova/ancient/who-built-the-pyramids.html

Dedigned by visual.ly

🟥 - 1 DAY OF TIME SPENT
C - CORE REQUIREMENT
U - USER FACING
E - ENGAGEMENT DRIVER
R - REVENUE OPPORTUNITY

THIS MESSAGE WAS BROUGHT TO YOU BY kinvey
Credits: Joe Chernov, Dave Wasmer of Kinvey. Survey data: AYTM

Structure of Android platform



Platform build-up 1/3

- The structure of the platform is simple and clean
- The **red** part is the Linux kernel, which contains:
  - Drivers of the hardware components
  - These parts are created by the device manufacturers, because they know best „what is in the box”
  - Memory management, process scheduling, low energy consumption performance management

Platform build-up 2/3

- The **green** parts are program libraries/services which running on the Linux kernel, like: libc, SSL, SQLite, etc. These are implemented in C/C++.
- The Dalvik virtual machine in the **yellow** part is based on these parts.
- Not compatible with Sun's virtual machine
  - Totally different instruction set
  - It runs different binary programs
  - After compiling, Java programs are not compiled into .class files, but to a Dalvik Executable format, which extension is .dex. It is usually smaller then .class files size.
  - Java is only the programmig language!

Platform build-up 3/3

- On the **blue** parts are only Java source code
- Java source is runned by the virtual machine. This is the essence of Android:
  - A visible and touchable operating system
  - Running programs
- The virtual machine can hide the whole Linux file system, and users can see only the file system provided by Android Runtime

Managed code

- **Managed code** runs on a virtual machine (Dalvik)
- It makes the application run secure
- An application can't ruin the whole system
- Memory management is happening with a garbage collector
- But! Be careful to create „good” source code

What is **not** true about Android platform?

A. Linux based system

B. Each standard application runs on a virtual machine instance

C. Built-in applications (e.g. phone, contacts) are part of the operating system

D. Supports only managed code

http://babcomaut.aut.bme.hu/votes/

Development tools

Software Development Tools on Android

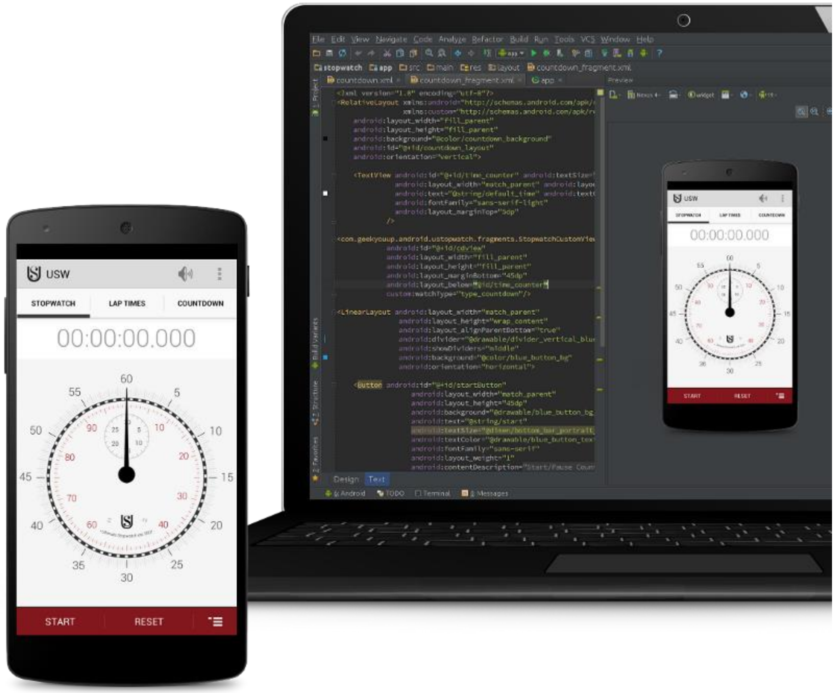
- **Android SDK (Software Development Kit)**:
  - Development tools, documentation, all Android version
  - Emulator management (AVD Manager)
  - Always up to date
  - Java, rich API
  - Android Studio and Eclipse plugin
- **Android NDK (Native Development Kit)**:
  - Enables to compile native code
  - C++
- **Android ADK (Accessory Development Kit)**:
  - Support for designing accessories like: docking station, health devices, etc.)
  - Android Open Accessory protocol (USB and Bluetooth)

Android Cross Platfrom Development

- Phonegap
  - HTML/CSS/JavaScript
  - Application is hosted in a Web View
  - Platform specific features are accessed via a JS library
  - Standard web based JS libraries can also be used (jQuery)
  - Android/iOS/Windows Phone/Windows 8/…
- Appcelerator Titanium
  - JavaScript
  - Precompiled JS code executed by an interpreter, generates native UI
  - Android/iOS/Blackberry (no WP)

Development tools - Android Studio

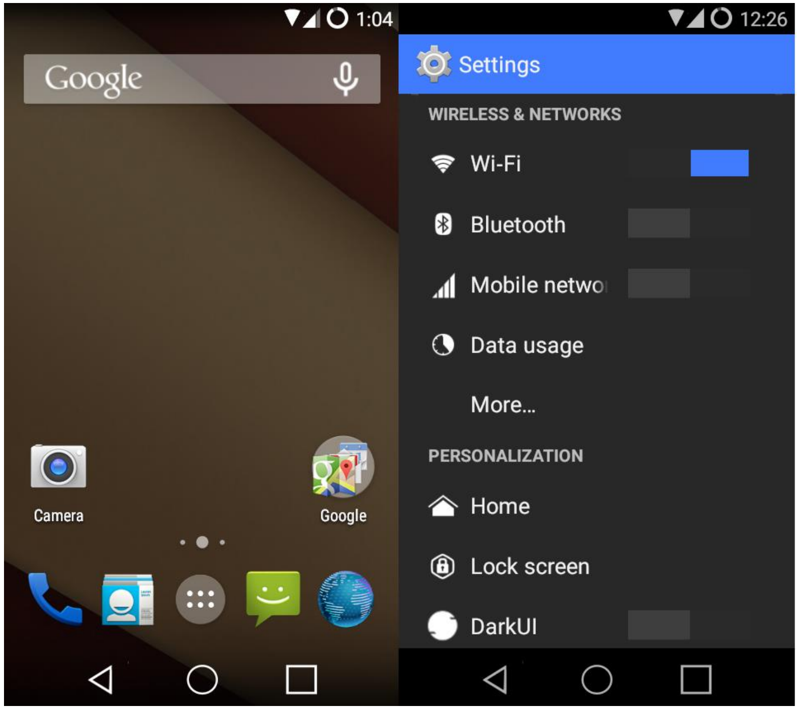
- Flexible Gradle-based build system
- Build variants and multiple APK generation
- Expanded template support for Google Services and various device types
- Rich layout editor with support for theme editing
- Lint tools to catch performance, usability, version compatibility, and other problems
- ProGuard and app-signing capabilities
- Built-in support for Google Cloud Platform
- Version control integration


File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
stopwatch app src main res layout countdown_fragment.xml
countdown.xml countdown_fragment.xml app
Preview
Design Text
Android TODO Terminal Messages
USW
STOPWATCH LAP TIMES COUNTDOWN
00:00:00.000
START RESET

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/
                xmlns:custom="http://schemas.android.com/apk/r
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="@color/countdown_background"
    android:id="@+id/countdown_layout"
    android:orientation="vertical">

    <TextView android:id="@+id/time_counter" android:textSize="
              android:layout_width="match_parent" android:layou
              android:text="@string/default_time" android:textC
              android:fontFamily="sans-serif-light"
              android:layout_marginTop="5dp"
        />

<com.geekyouup.android.ustopwatch.fragments.StopwatchCustomView
        android:id="@+id/cdview"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_marginBottom="45dp"
        android:layout_below="@id/time_counter"
        custom:watchType="type_countdown"/>

<LinearLayout android:layout_width="match_parent"
              android:layout_height="wrap_content"
              android:layout_alignParentBottom="true"
              android:divider="@drawable/divider_vertical_blu
              android:showDividers="middle"
              android:background="@color/blue_button_bg"
              android:orientation="horizontal">

    <Button android:id="@+id/startButton"
                android:layout_width="match_parent"
                android:layout_height="45dp"
                android:background="@drawable/blue_button_bg
                android:text="@string/start"
                android:textSize="@dimen/bottom_bar_portrait
                android:textColor="@drawable/blue_button_tex
                android:fontFamily="sans-serif"
                android:layout_weight="1"
                android:contentDescription="Start/Pause Coun
```

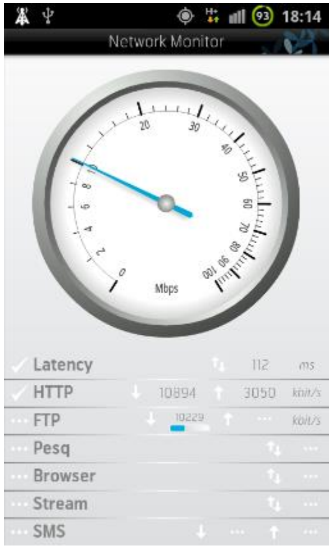
Emulator

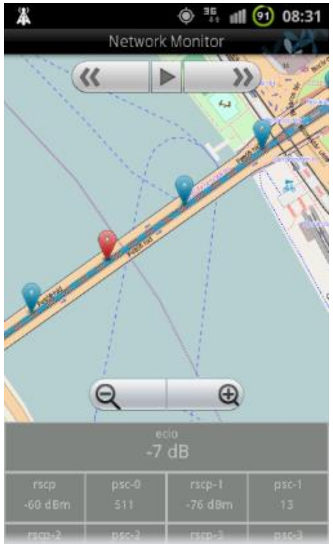
- Emulates the whole Android OS
- Hardware acceleration can be configured
- Built-in applications are available



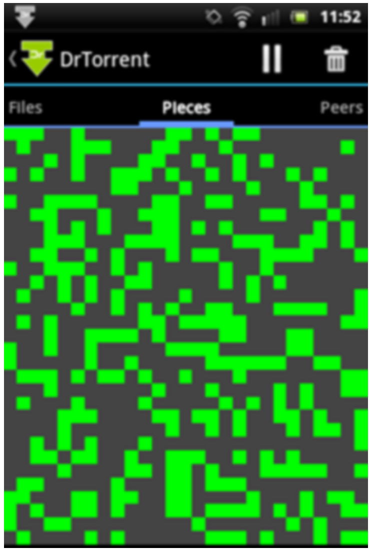
Case Studies

NetworkMonitor

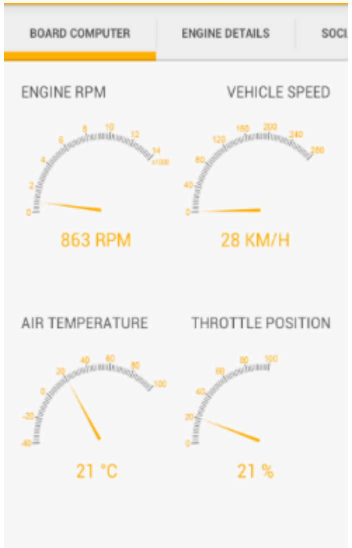


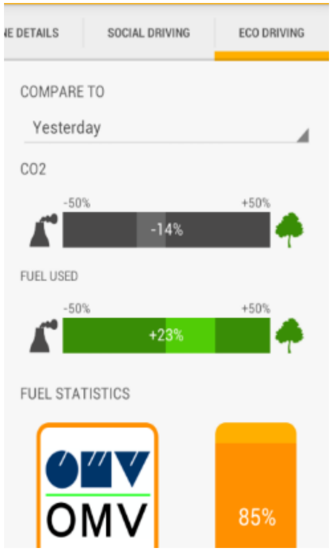


DrTorrent



VehicleICT





Let’s see some examples

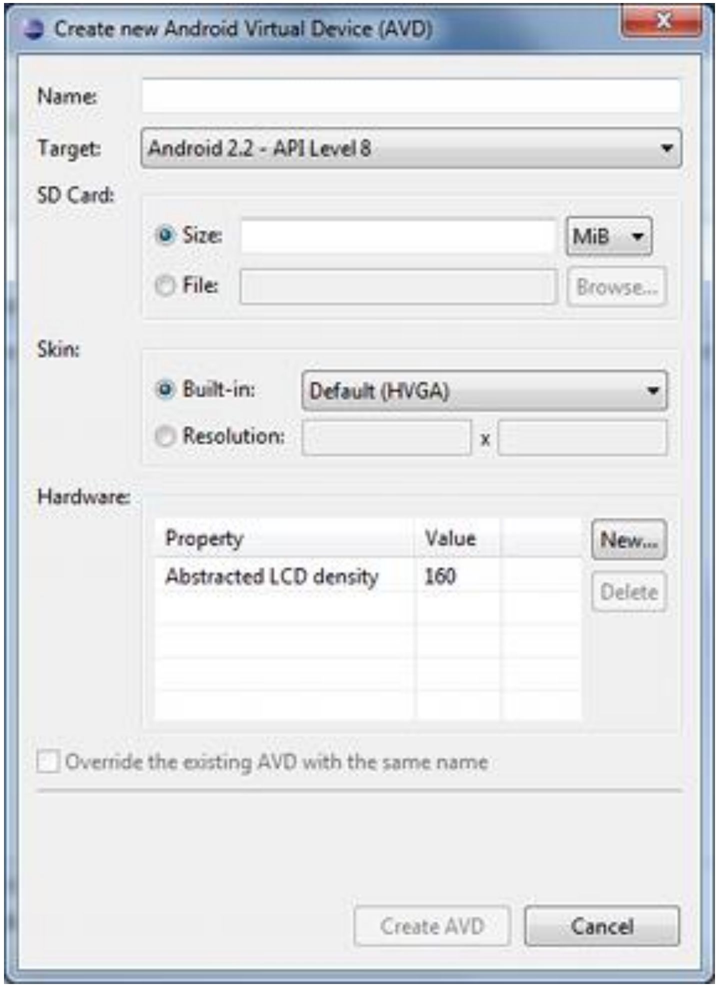
- Let’s exam the structure of the Android SDK!
- Lets create and start an emulator (AVD – Android Virtual Device)!

SDK Manager

- Download and update SDK components
- Check installed versions
- USB Drivers
- Hardware acceleration
- Google Play APIs
- Support libraries

AVD Manager - Emulator configuration

- For launching the emulator an AVD (Android Virtual Device) must be configured
- AVDs can be created via the AVD Manager application that is part of the SDK
- AVD properties:
  - Name
  - Android platform version
  - SD card size
  - Skin
  - Screen properties



Create new Android Virtual Device (AVD)

Name:

Target: Android 2.2 - API Level 8

SD Card:
● Size: MiB
○ File:

Skin:
● Built-in: Default (HVGA)
○ Resolution: x

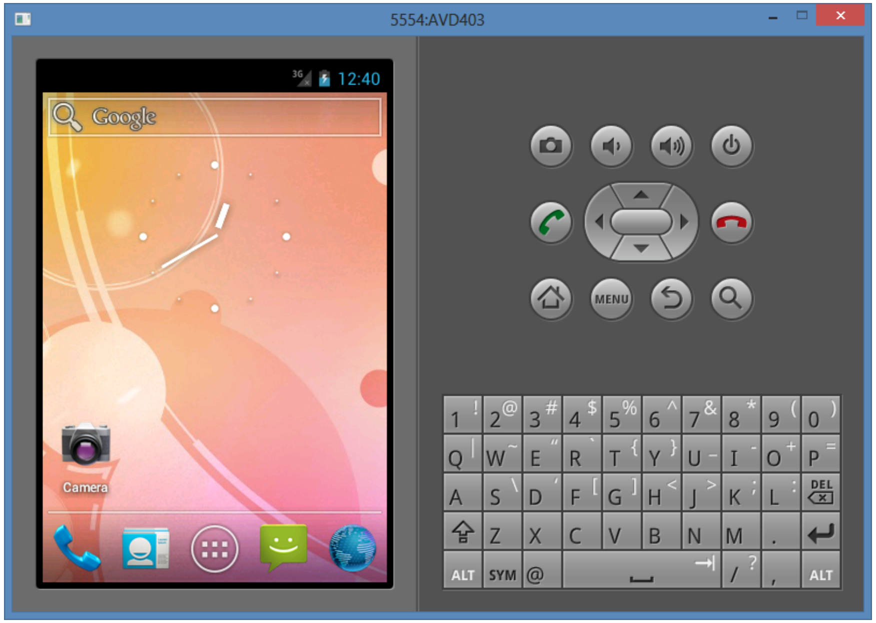
Hardware:

Property	Value
Abstracted LCD density	160

□ Override the existing AVD with the same name

Emulator

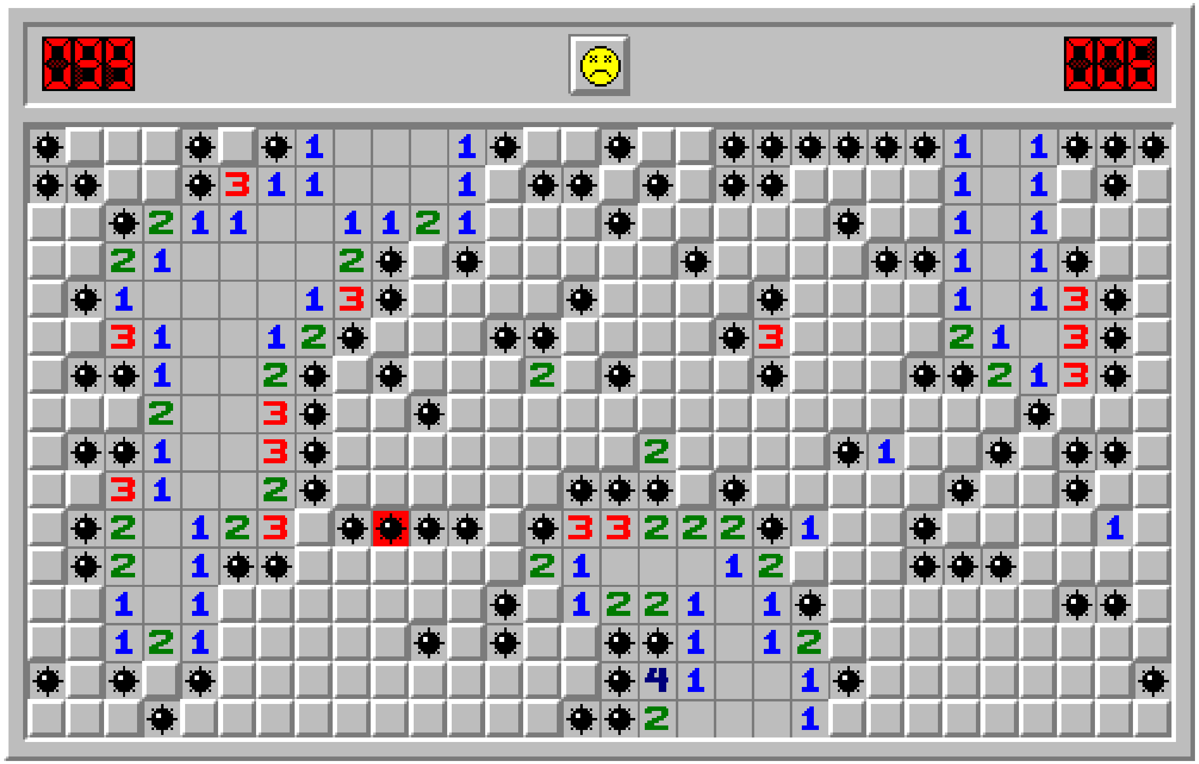
- Emulates the whole Android OS
- Hardware acceleration can be configured
- Built-in applications are available

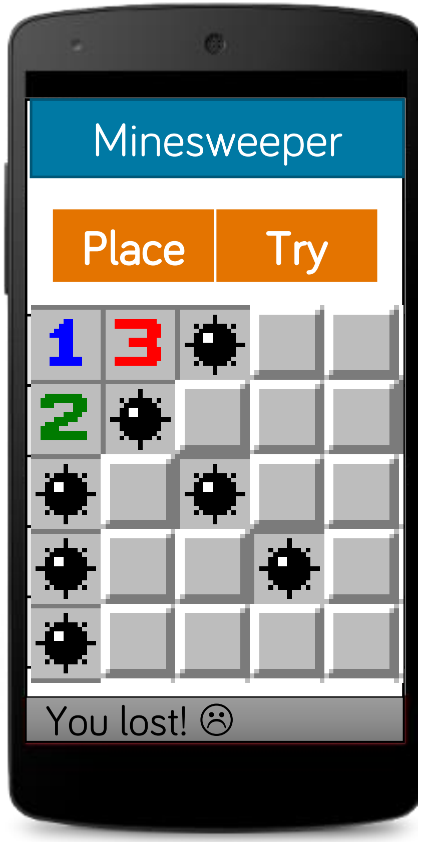


Summary

- Android platform structure
- Development tools: SDK, Emulator, Android Studio
- First Android Appilcation
- First Homework Assignment ☺

First assignment





- Deadline: *September xxth* (23:59)
- Next-(next) time: TicTacToe demo!
  (base for Minesweeper)

Thank you!