# Toolkits for Mobile Devices & Smartphones

Presented by **Yutong Xie**

Advanced User Interface Software, 2017 Spring

# Outline

- From history to present
- Native, web & hybrid mobile apps
- Cross-platform mobile development
- Tools Examples: PhoneGap, RhoStudio, Titanum, Automobile

# From history to present

# Dated back to early 1970's

1 kg,
1 hour,
30 phone numbers

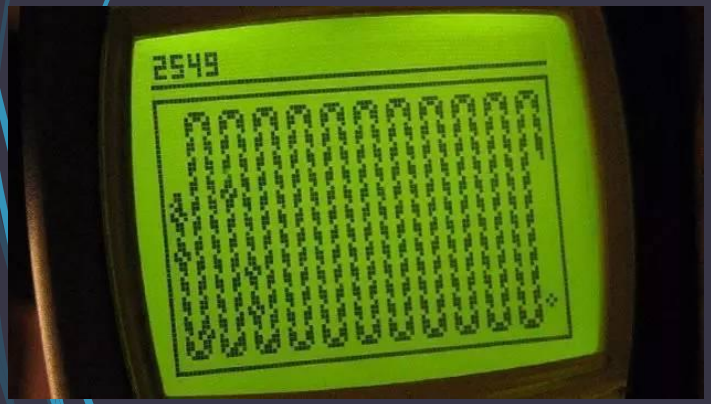
https://www.youtube.com/watch?v=DMbGrakUHc4

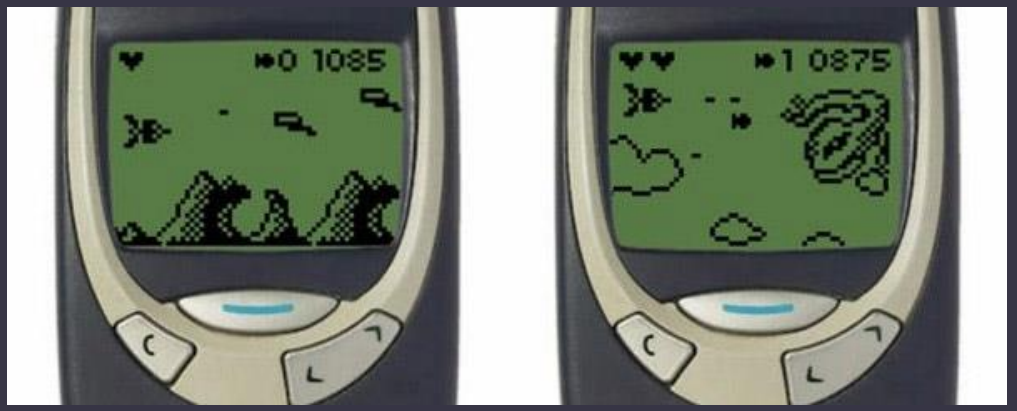
# First “smartphone” by IBM (1994)


- Smartphone: combines features of PC and other useful features for mobile
- Calculator, world clock, calendar and contact book
- Touchscreen, virtual keyboard

# The first apps

- Developed by the handset manufacturers **in-house,** pre-installed (rooted to uninstall)
- Changed the way people thought about communication
- Prices dropped, batteries improved, reception area grew

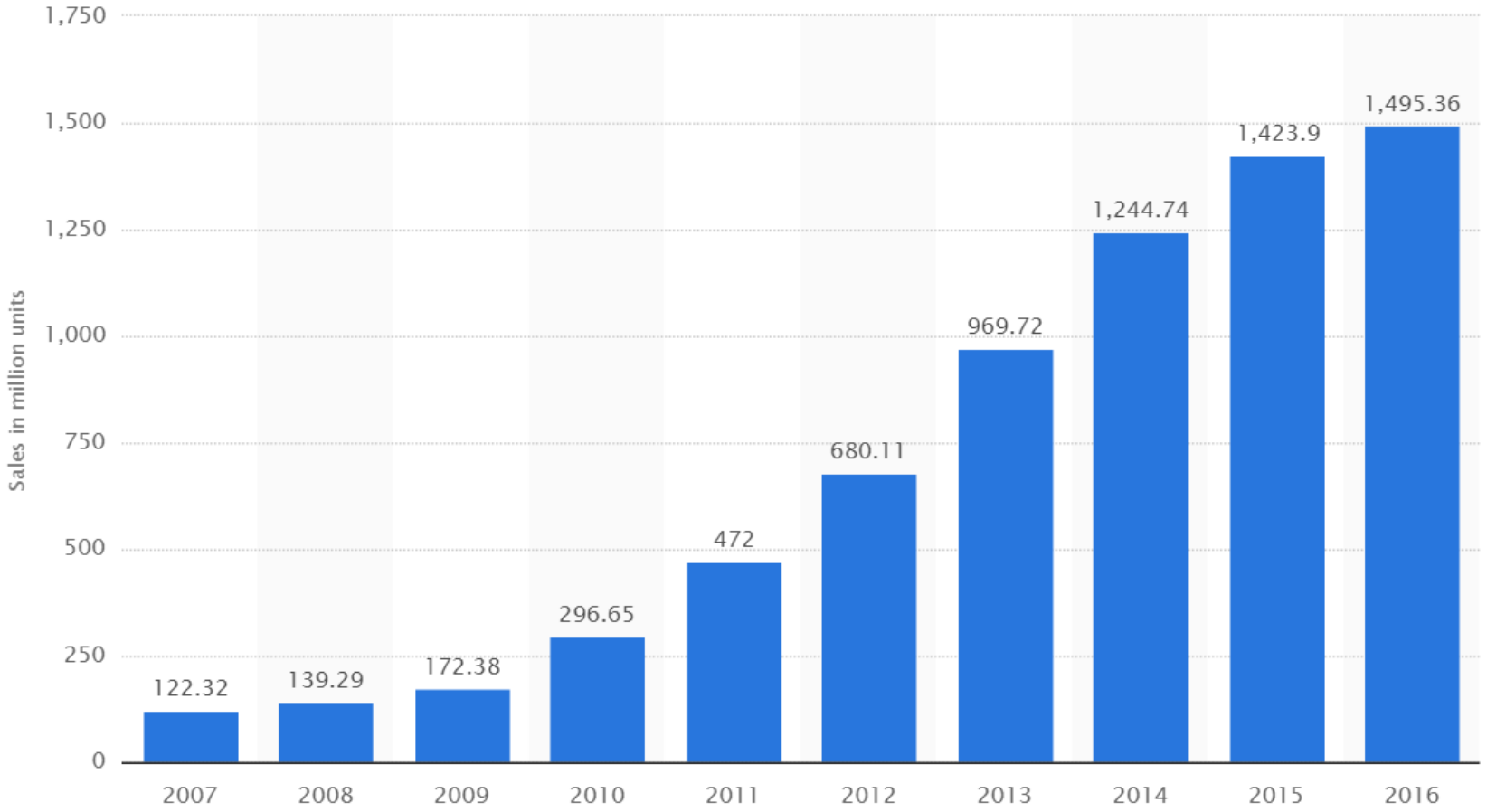


# First iPhone (2007)

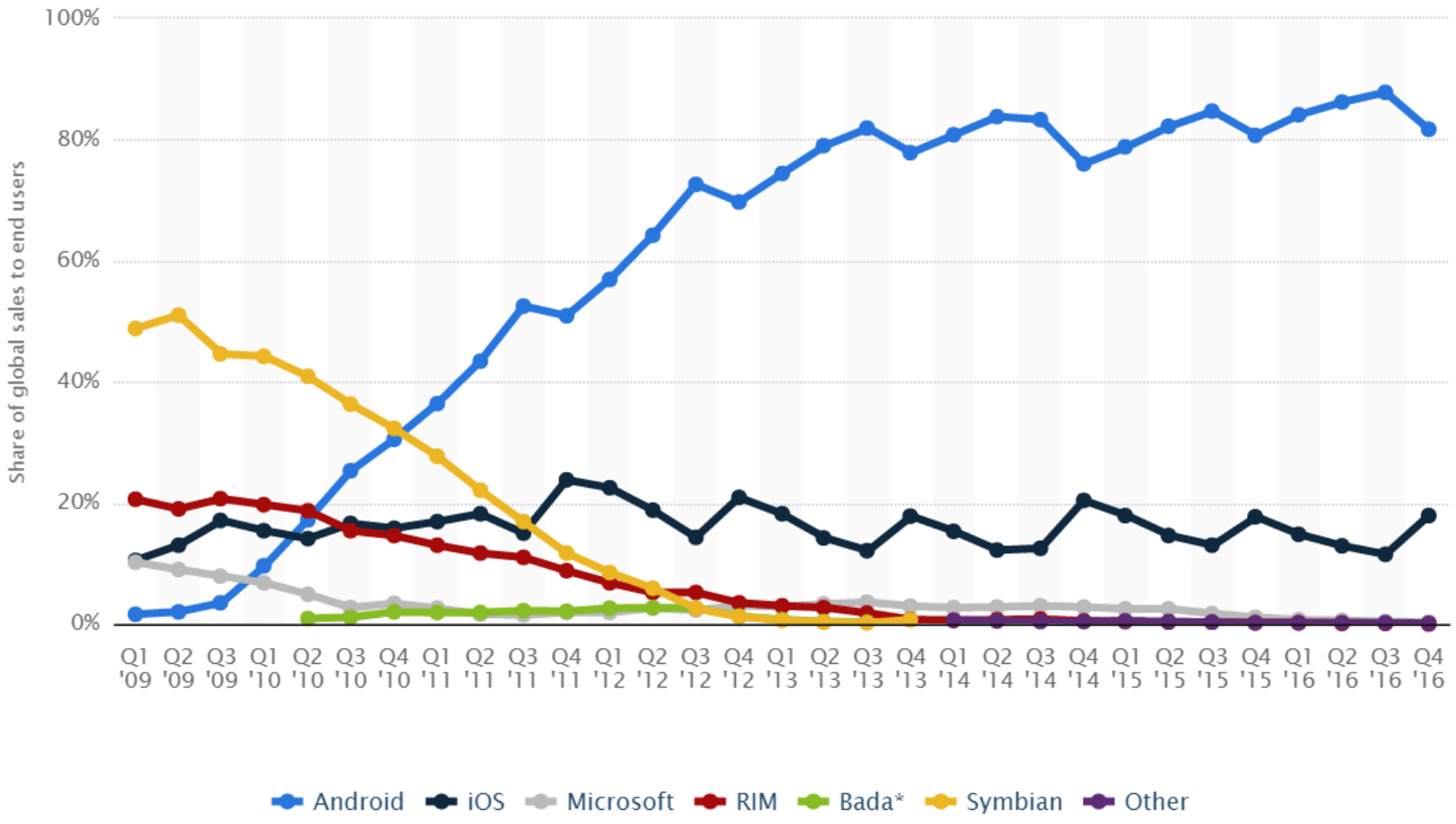
Revolutionary ~~resistive~~ capacitive touchscreen

## Number of smartphones sold to end users worldwide from 2007 to 2016 (in million units)




https://www.statista.com/statistics/263437/global-smartphone-sales-to-end-users-since-2007/

## Global mobile OS market share in sales to end users from 1st quarter 2009 to 1st quarter 2016




https://www.statista.com/statistics/266136/global-market-share-held-by-smartphone-operating-systems/

# Native, web or hybrid

*How to make choices when developing mobile apps*

# Type 1: Native apps

- Different languages and tools for different platforms
  - Java + Android SDK/Studio -> Android
  - Objective-C or Swift + Xcode -> iOS
  - C# or VB.NET + Visual Studio -> Windows
- Good user experience
- Access to device APIs
- Cons: High cost ($ and time)

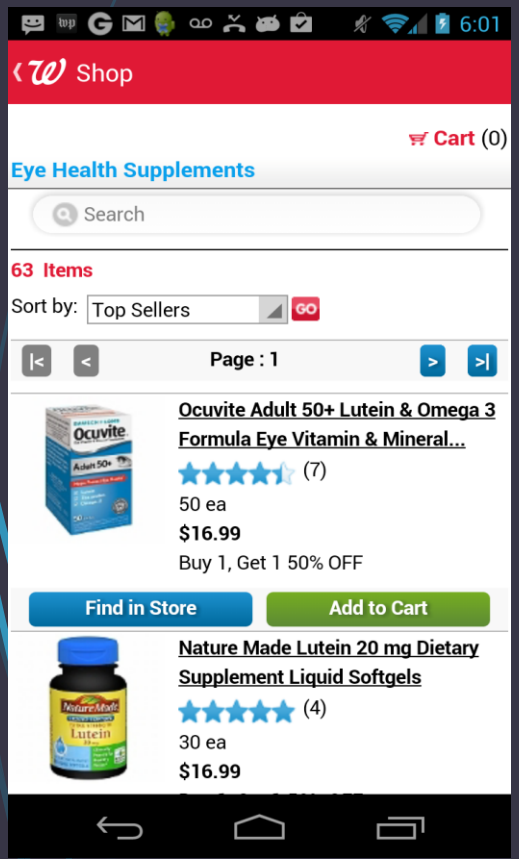
# Type 2: Web apps

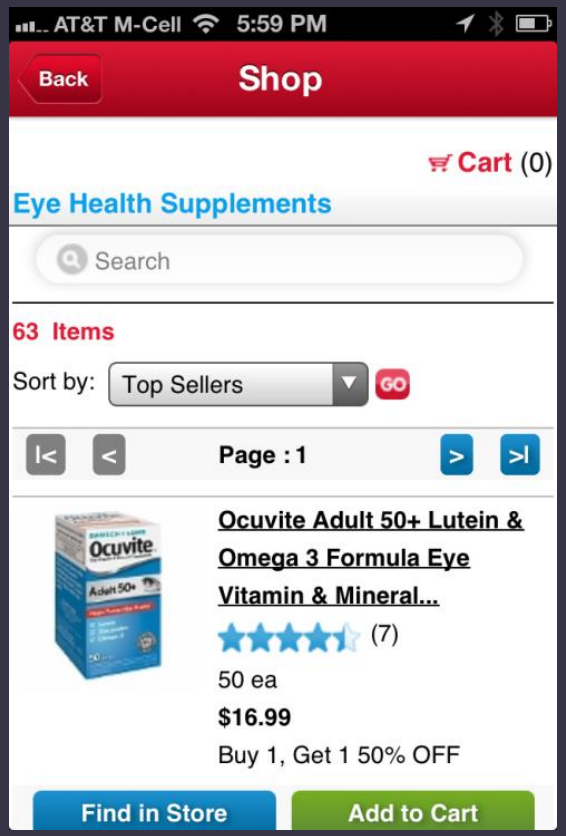
- Use HTML5, CSS3, JavaScript
- Live in mobile browsers (URL & network), no installation
- Cheap, simplified deployment, cover different platforms
- Immediate availability (publication)
- Cons: poor user experience, limited access to APIs

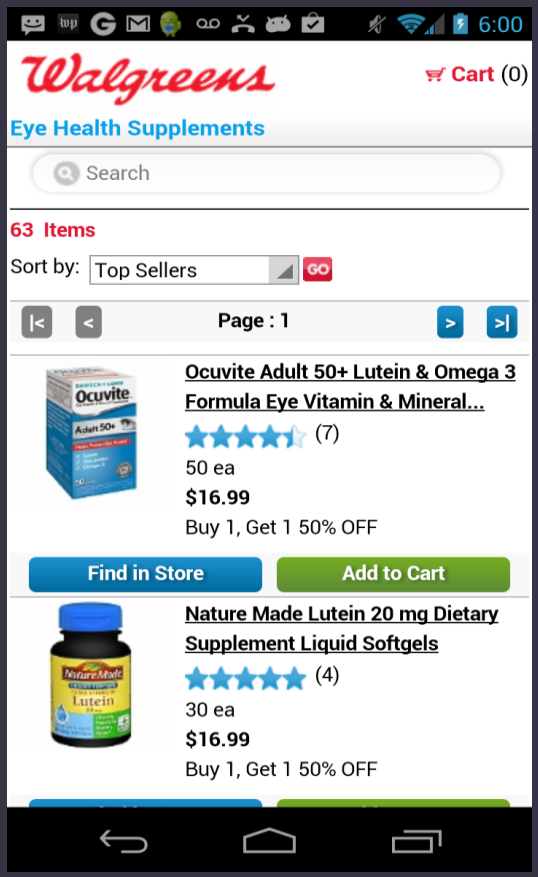
# Type 3: Hybrid apps

- Native features + web technologies + wrapper –> app store
- Rely on middleware functionality and updates
- User experience: not fast, smooth as native apps


5:44
Refill by Scan
ENTER RX NUMBER
Center prescription barcode in marked area
Prescription
Barcode
QR Code


6:01
Shop
Cart (0)
Eye Health Supplements
Search
63 Items
Sort by: Top Sellers
GO
Page : 1
Ocuvite Adult 50+ Lutein & Omega 3 Formula Eye Vitamin & Mineral...
(7)
50 ea
$16.99
Buy 1, Get 1 50% OFF
Find in Store
Add to Cart
Nature Made Lutein 20 mg Dietary Supplement Liquid Softgels
(4)
30 ea
$16.99


AT&T M-Cell
5:59 PM
Back
Shop
Cart (0)
Eye Health Supplements
Search
63 Items
Sort by: Top Sellers
GO
Page : 1
Ocuvite Adult 50+ Lutein & Omega 3 Formula Eye Vitamin & Mineral...
(7)
50 ea
$16.99
Buy 1, Get 1 50% OFF
Find in Store
Add to Cart


6:00
Walgreens
Cart (0)
Eye Health Supplements
Search
63 Items
Sort by: Top Sellers
GO
Page : 1
Ocuvite Adult 50+ Lutein & Omega 3 Formula Eye Vitamin & Mineral...
(7)
50 ea
$16.99
Buy 1, Get 1 50% OFF
Find in Store
Add to Cart
Nature Made Lutein 20 mg Dietary Supplement Liquid Softgels
(4)
30 ea
$16.99
Buy 1, Get 1 50% OFF


Native app
Native application
Device APIs
Web app
Mobile browser
Web code
Hybrid app
Native container
Web code
Device APIs

# Cross-platform mobile development

*Develop once and run anywhere*

# Cross-platform mobile development

- Around 50%-80% code reuse
- Consistent user experience & similar behavior across different devices and platforms
- Open source/commercial frameworks and tools
- Provide a bridge/middleware between the web app & OS APIs

# Existing approaches of cross-platform mobile development

1. **Hybrid approach**:
   - Use browser engine to render HTML in native container
   - May lack native look
   - e.g. PhoneGap

# Existing approaches of cross-platform mobile development

2. **Interpreted approach**:

- Written in a specific language
- At runtime the tool interprets it for different platforms
- May degrade the performance

- e.g. Rhodes, Appcelerator Titanium

# Existing approaches of cross-platform mobile development

3. **Model driven approach**:

- Use DSL (Domain Specific Language) and UML (Unified Modeling Language), generate source code from defined models
- Still in early stage

- e.g. AutoMobile Project

# Criteria for tools selection*

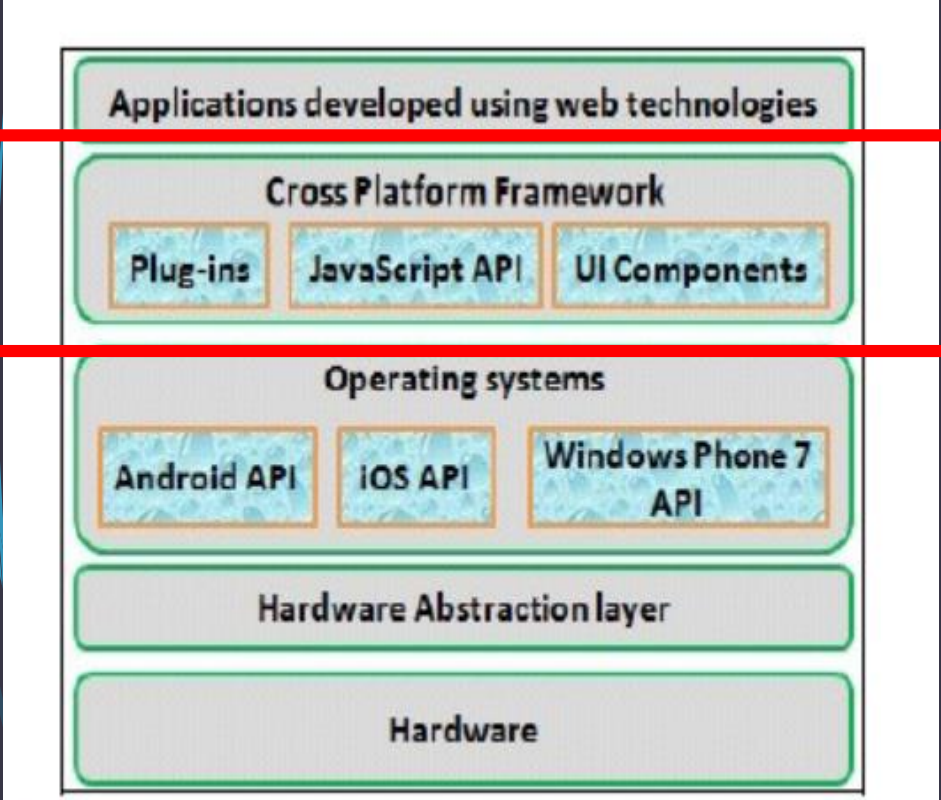



Figure 1: General architecture of cross platform mobile application development.

1. Multiple mobile platforms supported
2. Development Environment: IDE, debugger, emulator, GUI, deployment etc.
3. Availability of OS API's: what can be accessed
4. Documentation
5. User eXperience/ User Interface
6. Architecture
7. Resource consumption

** Summarized from multiple papers*

# Tools for cross-platform mobile development

PhoneGap,

Rhodes, Appcelerator Titanium,

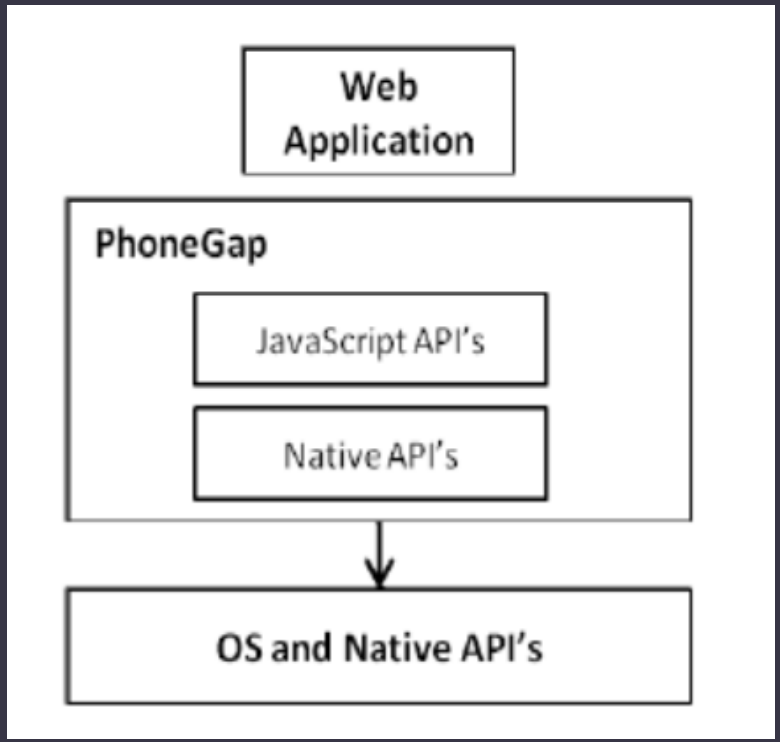
Automobile project

# PhoneGap

- Free open source framework from Nitobi (2008), acquired later by Adobe
- Using only HTML, CSS, JavaScript
- Providing access to device functionality through JavaScript API
- Does not provide an IDE, but provide a "cloud-compiler" called PhoneGap Build

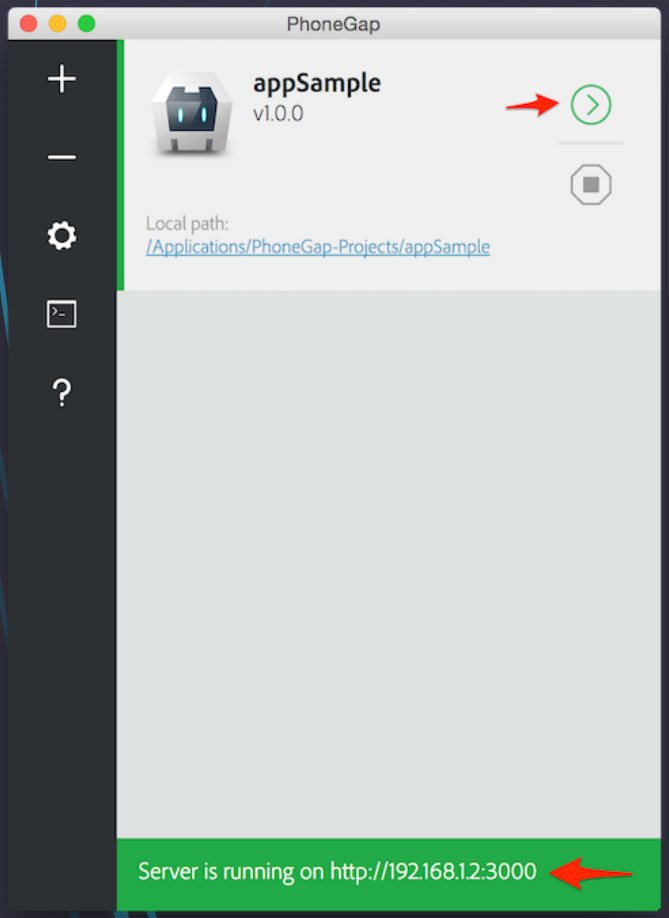
# PhoneGap

- Desktop App + Developer Mobile App
- 85-90% code reuse (at best)
- Doc: http://docs.phonegap.com/



# PhoneGap: Preview app

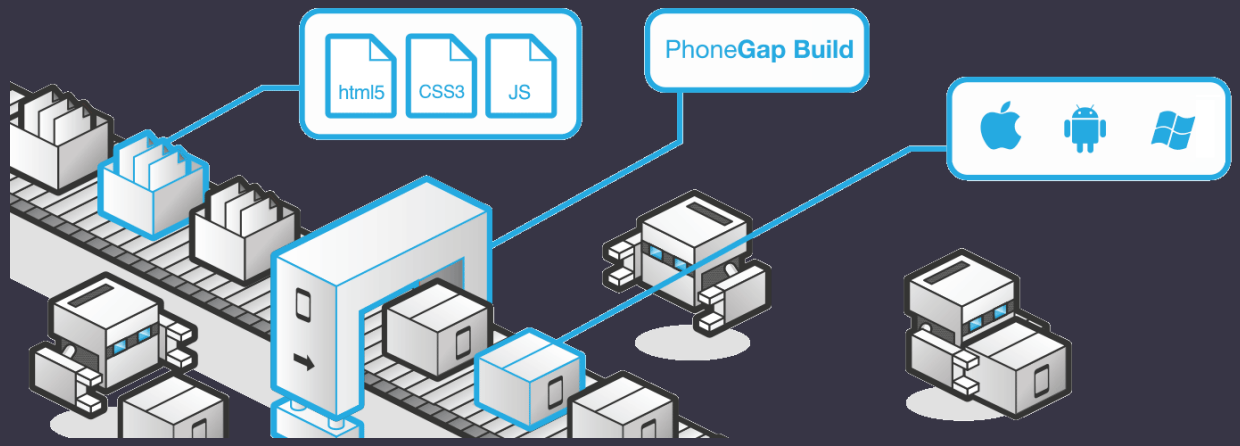
https://youtu.be/pggw-9b8RVY





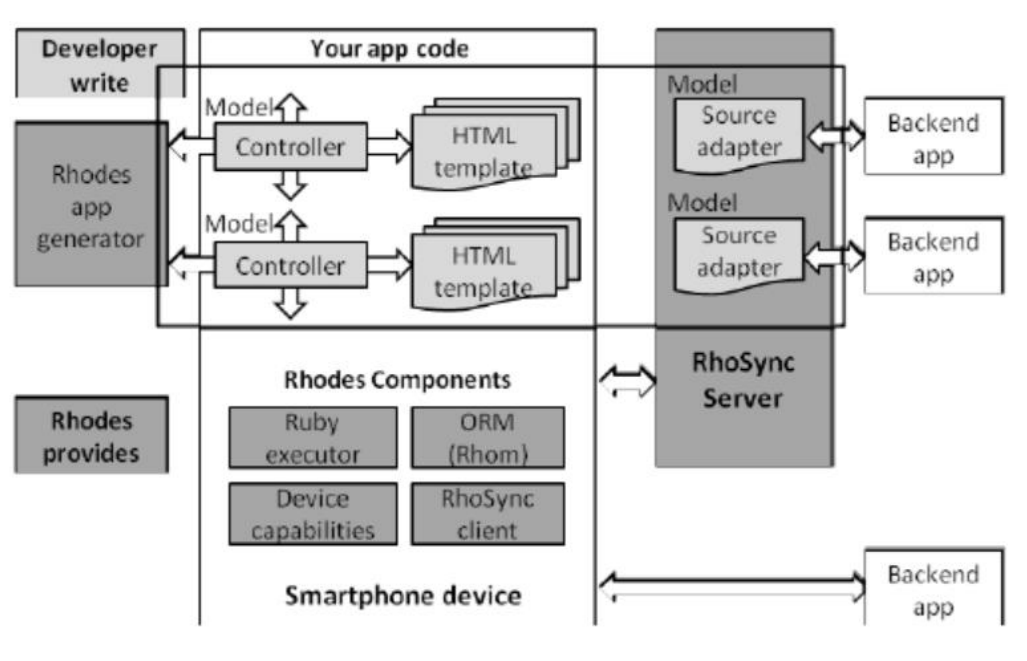
# PhoneGap: pros & cons

- Pros:
  - Native wrapper source code is provided, can be customized further;
  - Simple ‘drop-in libraries’ concept
- Cons:
  - Lack of UI components – can be combined with Sencha Touch 2.0 to achieve better UI



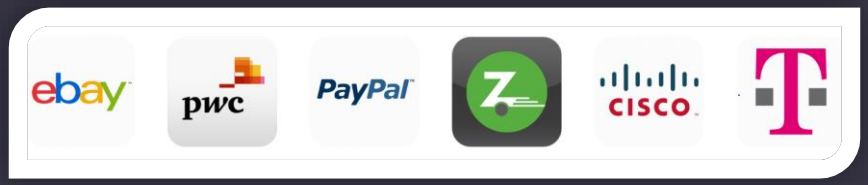
# Rhodes / RhoMobile suite

- IDE: RhoStudio
- Only framework with MVC support
- controller + HTML template + source adapter



# Appcelerator Titanium


- Providing access to device functionality (e.g. GPS) through JavaScript APIs and platform APIs
- Compile to bytecode, platform SDK builds the package for target platform
- Titanium Studio as an IDE



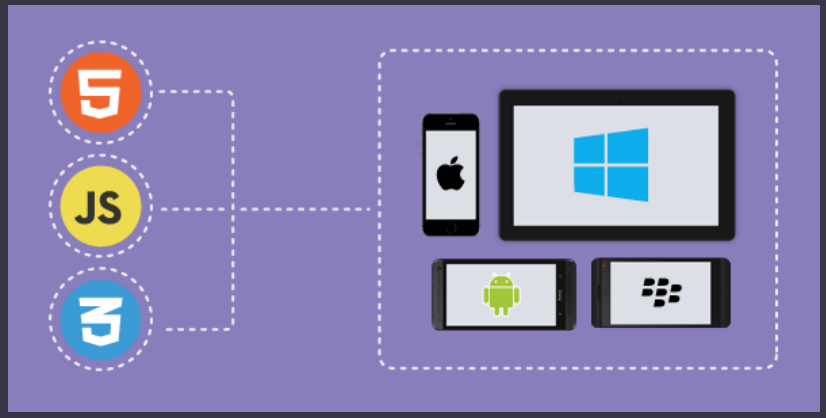
# AutoMobile Project

- Model-Driven Engineering, model-to-code generator
- Appeared in 2014, new and promising
- Only Android & iOS now, still growing
- Relies on modelling languages such as IFML (Interaction Flow Modelling Languages) & tools like WebRatio

- http://automobile.webratio.com/

# Cross-platform mobile development: Pros

- High code reuse, faster development, reduced costs
- Easy to maintain, fix bugs
- Unit tests written only once for common code
- No need for specific platform development language skills
- Ideal for B2B apps where efficiency > look & feel



# Cross-platform mobile development: Cons

- Sluggish UI, HTML5 animations
- Browser components have evolved with OS
- Have to catch up with new OS version
- More battery consumption compared to native apps
- GUI needs to be coded multiple times to obtain specific look and feel

“The biggest mistake we’ve made as a company is betting on HTML5 over native.”

- Mark Zuckerberg, 2012


Thank you!