Mobile Web Appplications Development with HTML5

Lecture 1: Introduction

Claudio Riva

Aalto University - Fall 2012

The Web was Dead!

WIRED AUGUST 2010

The Web was Dead!

WIRED AUGUST 2010

The Web is Dead Again!

> I think the biggest mistake we made as a company is betting too much on HTML5 as opposed to native
>
> **Mark Zuckerberg, Facebook CEO**

What Went Wrong ?

What Went Wrong ?

- No client-side rendering

What Went Wrong ?

- No client-side rendering
- No templating

What Went Wrong ?

- No client-side rendering
- No templating
- No local storage of data

What Went Wrong ?

- No client-side rendering
- No templating
- No local storage of data
- No tolerance for high-latency

What Went Wrong ?

- No client-side rendering
- No templating
- No local storage of data
- No tolerance for high-latency
- No JSON API

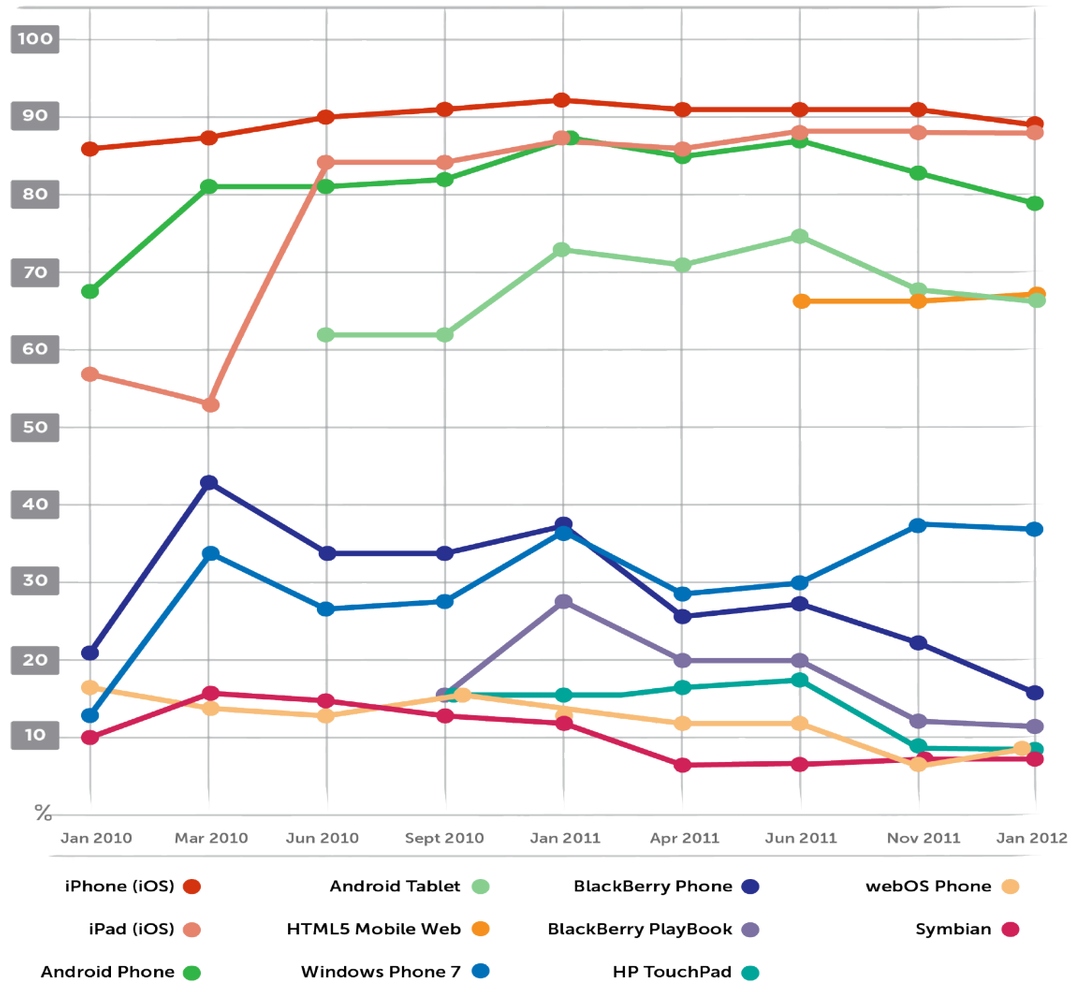
What Went Wrong ?

- No client-side rendering
- No templating
- No local storage of data
- No tolerance for high-latency
- No JSON API
- No mobile optimization

What Really Went Wrong ?

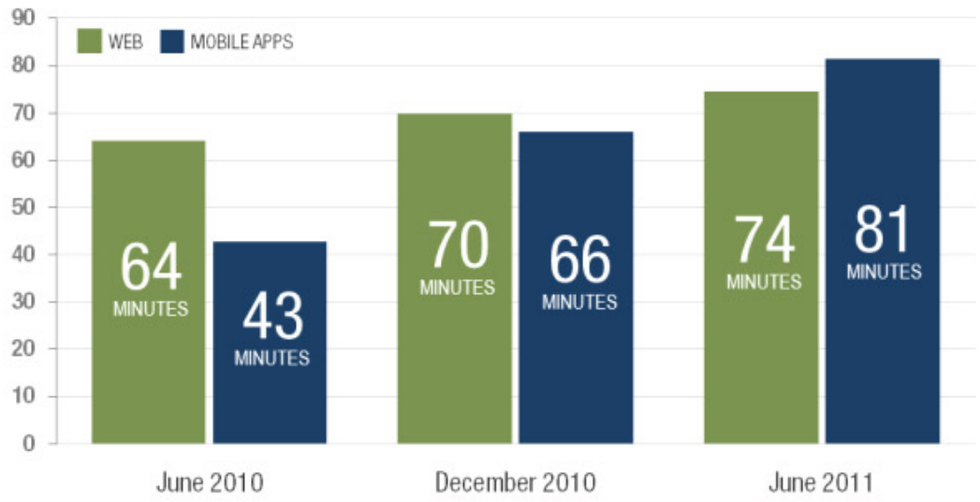
> He still claimed their biggest mistake was betting on HTML5 when their real biggest mistake was incompetent execution
>
> **Yehuda Katz**

Interest in developing for various mobile platforms



More Time is Spent on Mobile Apps

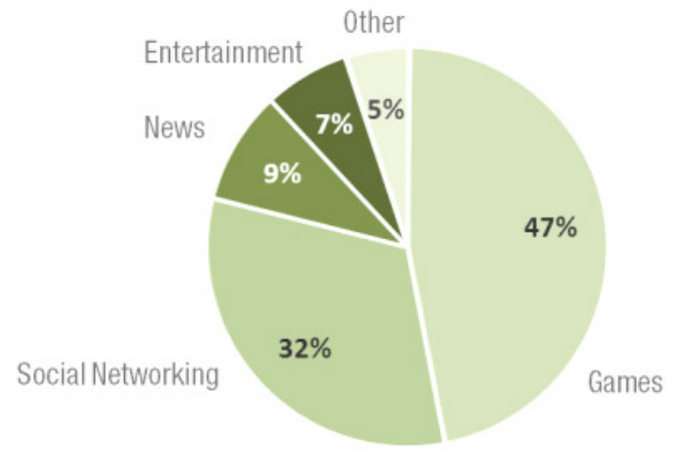
U.S. Mobile Apps vs. Web Consumption, Minutes per Day



FLURRY

Sources: comScore, Alexa, Flurry Analytics

U.S. Mobile App Consumption, Time Spent per Category



FLURRY

Source: Flurry Analytics, May 2011

Is the Web Dead ?

WIRED AUGUST 2010

Is the Web Dead ?

- If you are reading this, it's not.

WIRED AUGUST 2010

Is the Web Dead ?

- If you are reading this, it's not.
- Wired article was read and discussed mostly on the web

WIRED AUGUST 2010

Is the Web Dead ?

- If you are reading this, it's not.
- Wired article was read and discussed mostly on the web
- HTML5 brings a lot of new cool stuff for web developers

WIRED AUGUST 2010

Is the Web Dead ?

- If you are reading this, it's not.
- Wired article was read and discussed mostly on the web
- HTML5 brings a lot of new cool stuff for web developers
- Companies like Google, Apple, Adobe, Microsoft are embracing HTML5

WIRED AUGUST 2010

Is the Web Dead ?

- If you are reading this, it's not.
- Wired article was read and discussed mostly on the web
- HTML5 brings a lot of new cool stuff for web developers
- Companies like Google, Apple, Adobe, Microsoft are embracing HTML5
- Many apps have an HTML5 heart wrapped in a native shell

WIRED AUGUST 2010

But it's true that:

But it's true that:

Building web apps is more like craftsmanship

But it's true that:

Building web apps is more like craftsmanship

Performance was/is (sometimes) not enough

But it's true that:

Building web apps is more like craftsmanship

Performance was/is (sometimes) not enough

Browser support (especially on mobile) is fragmented

But it's true that:

Building web apps is more like craftsmanship

Performance was/is (sometimes) not enough

Browser support (especially on mobile) is fragmented

Web is evolving fast

But it's true that:

Building web apps is more like craftsmanship

Performance was/is (sometimes) not enough

Browser support (especially on mobile) is fragmented

Web is evolving fast

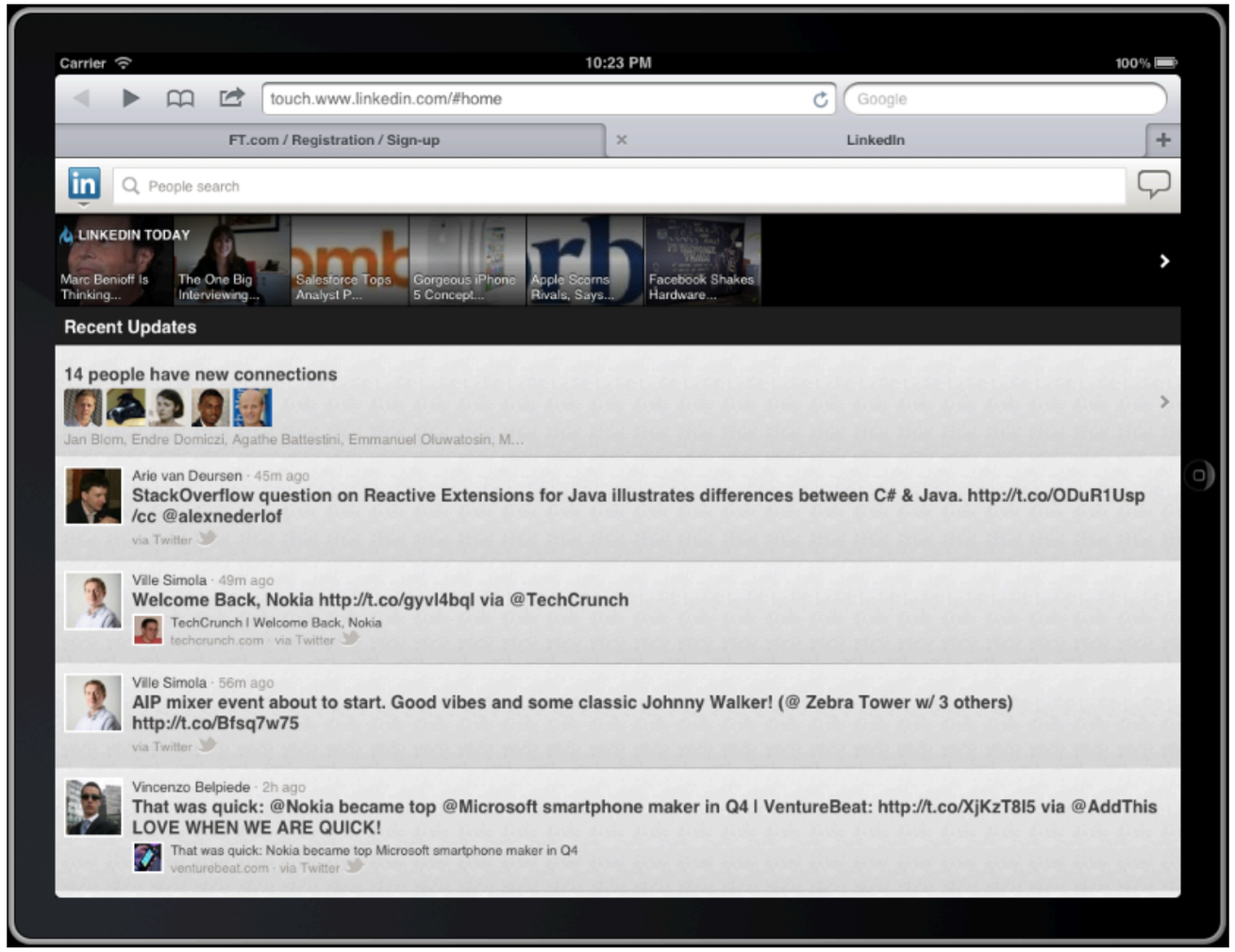
It's the only cross-platform mobile platform

What the heck is a "Mobile Web App" ?

FT Web App (more info)

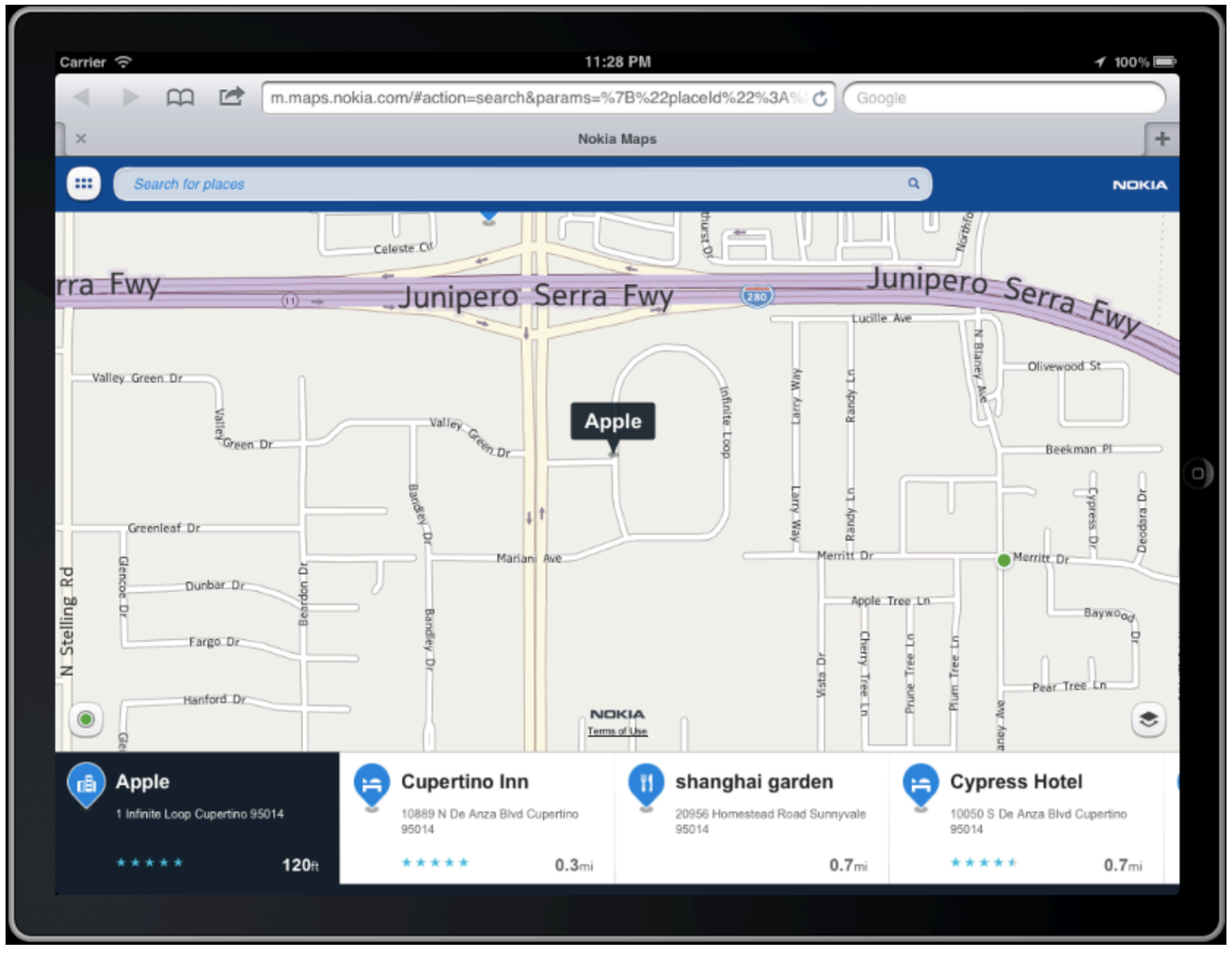
- Developed in 8 months by 3 people
- Hugely optimized for iOS
- Great and responsive UI design
  - Content balancing based on device type
  - Audio playing while moving to other pages
  - Continuos carousel
  - Preloading of content
  - Swipes using touch gestures
- Offline access
- More engagement than native app

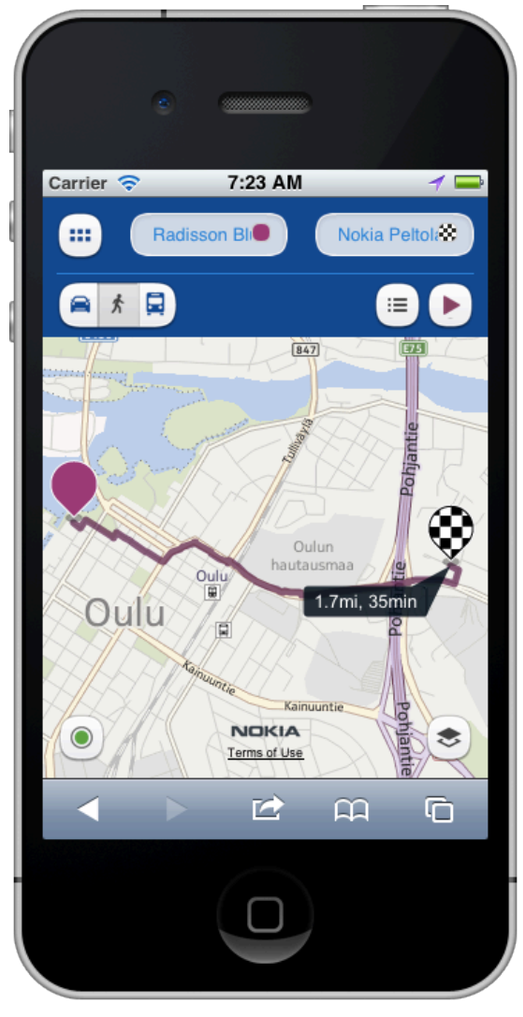
LinkedIn App



Cut the Rope

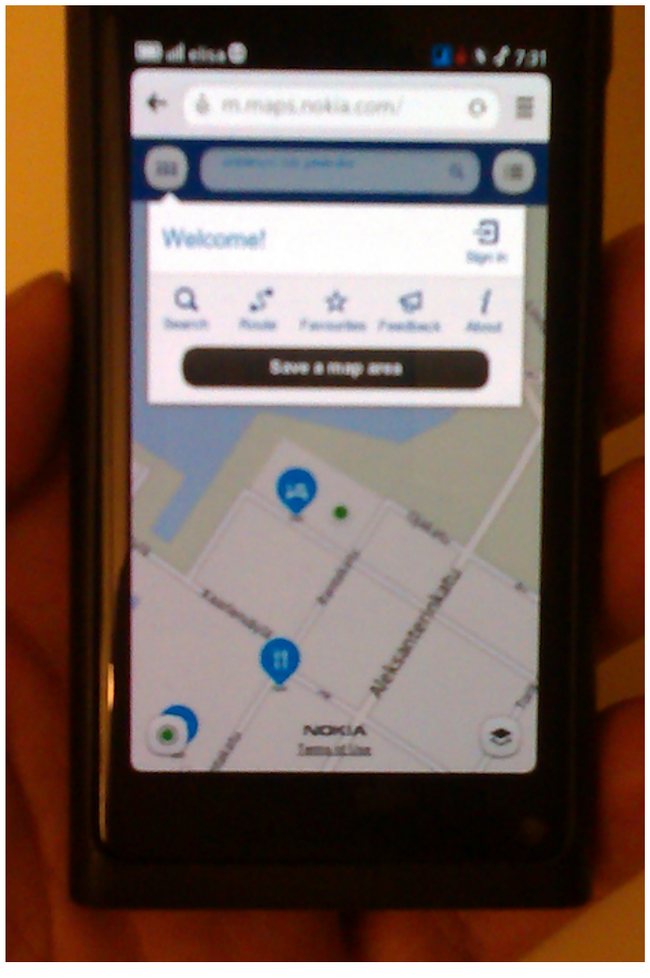
Nokia Mobile Maps



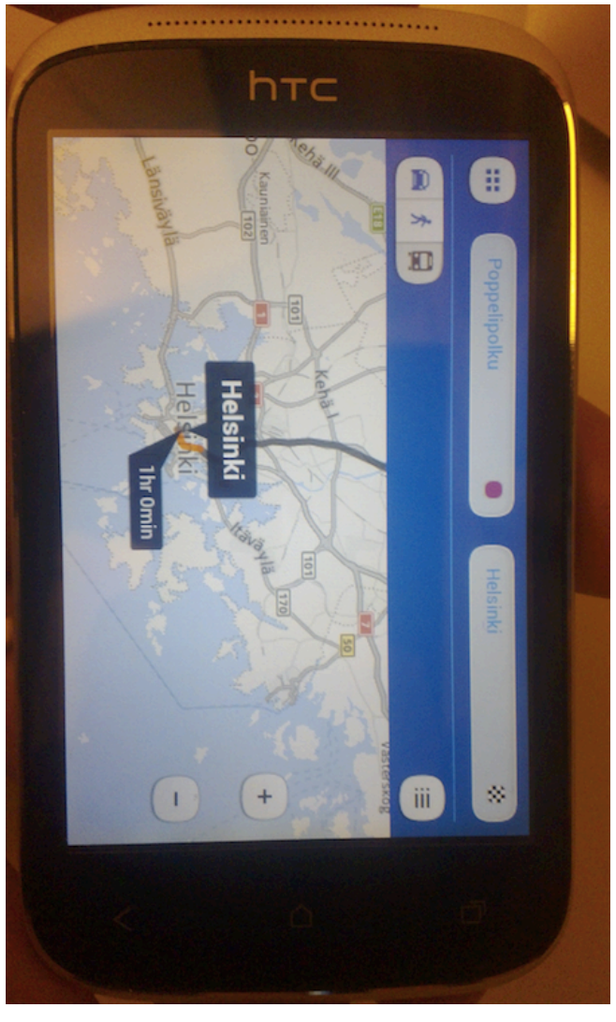


Nokia Mobile Maps

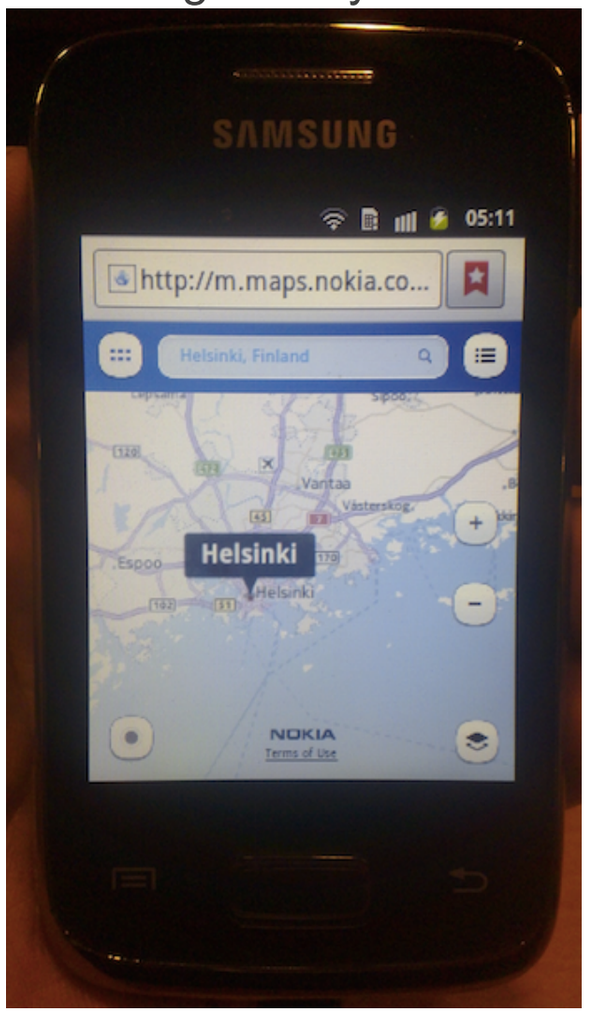
Nokia N9 - 400€



HTC Desire C - 250€

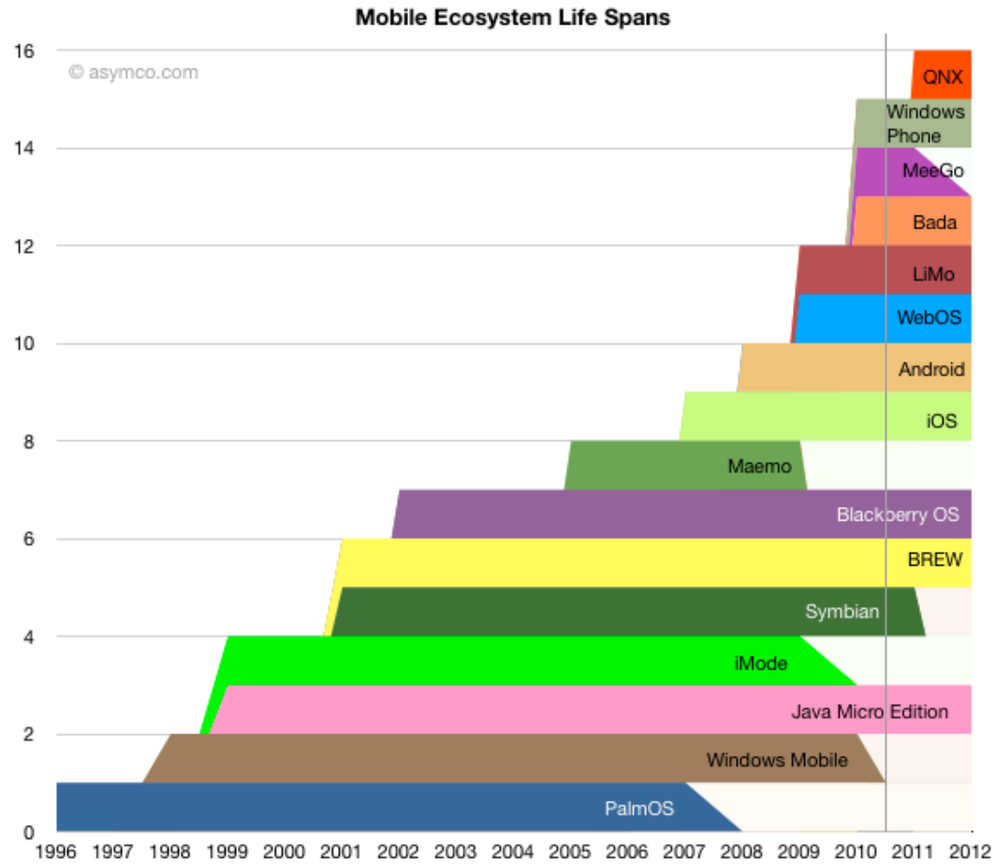


Samsung Galaxy Y - 99€



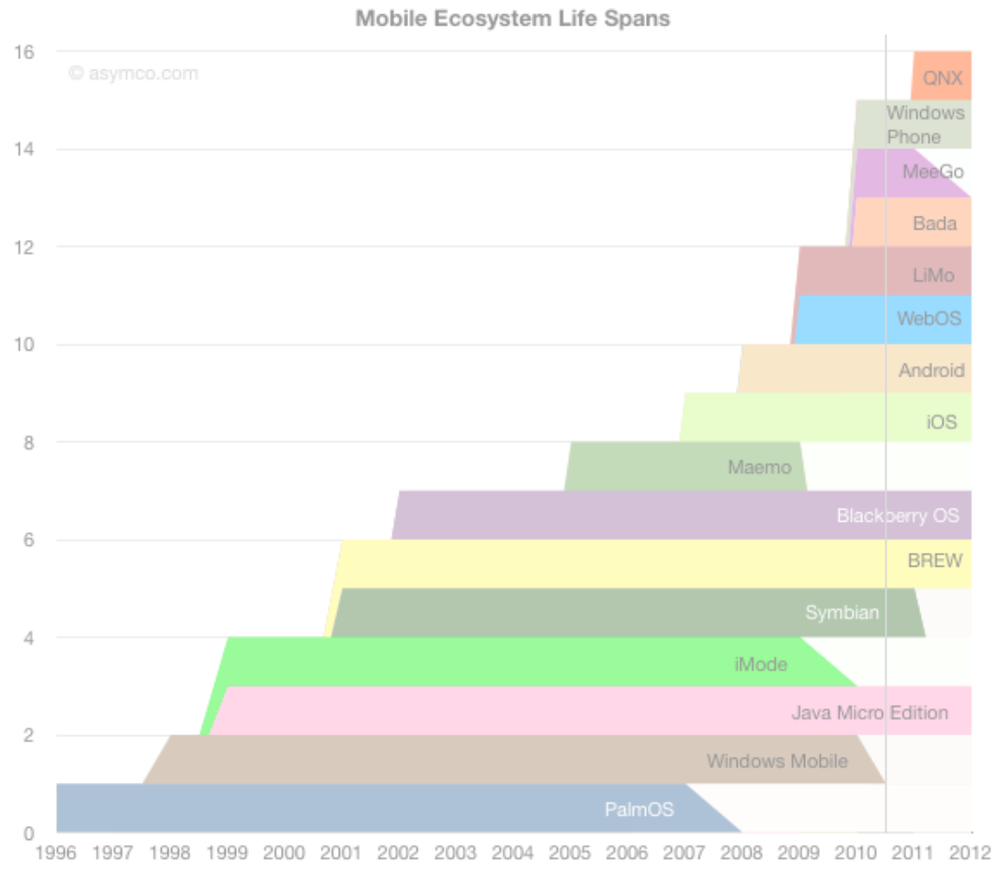
Developing Native Apps for Multiple Platforms

Developing Native Apps for Multiple Platforms



ASYMCO FEB 2011

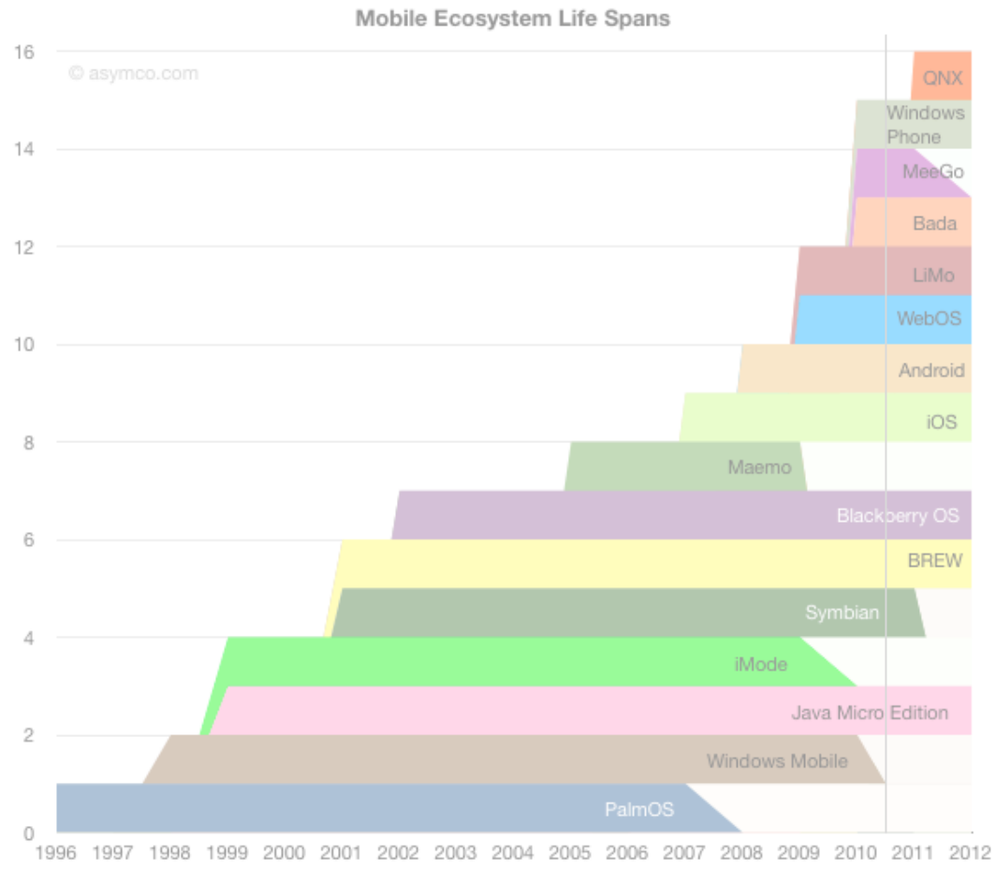
Developing Native Apps for Multiple Platforms



ASYMCO FEB 2011

We must have:

Developing Native Apps for Multiple Platforms

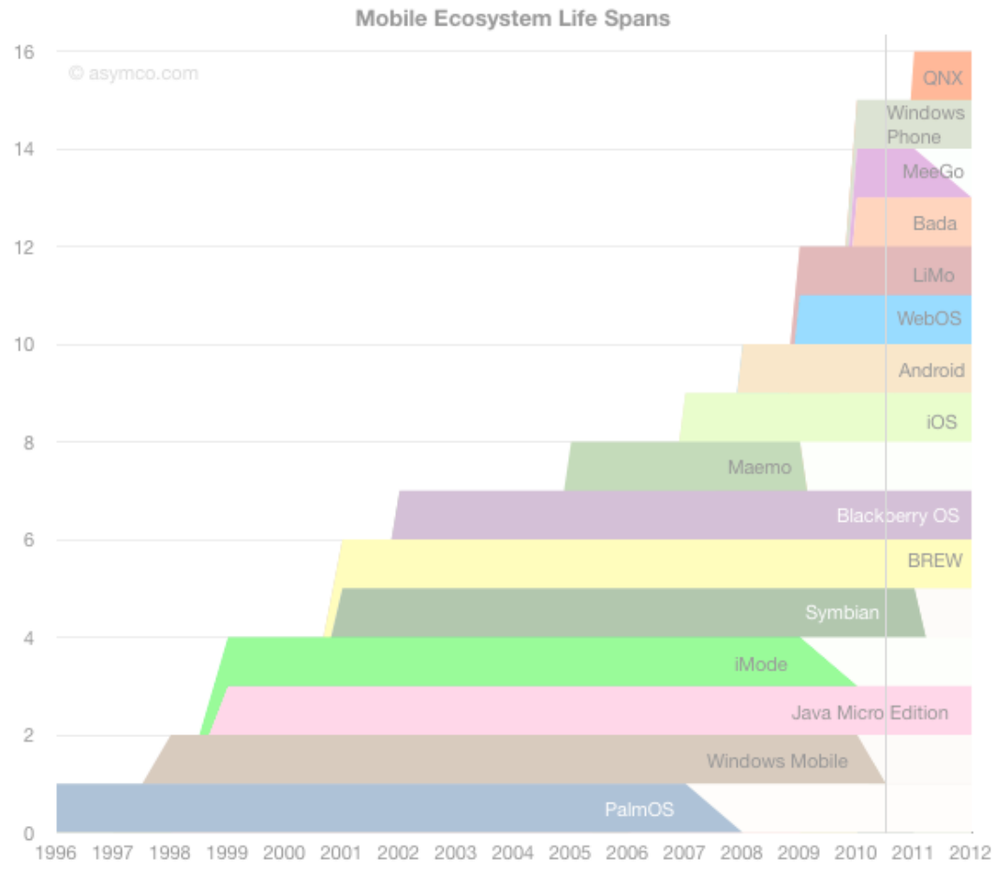


ASYMCO FEB 2011

We must have:

- 2008 - iPhone Apps

Developing Native Apps for Multiple Platforms

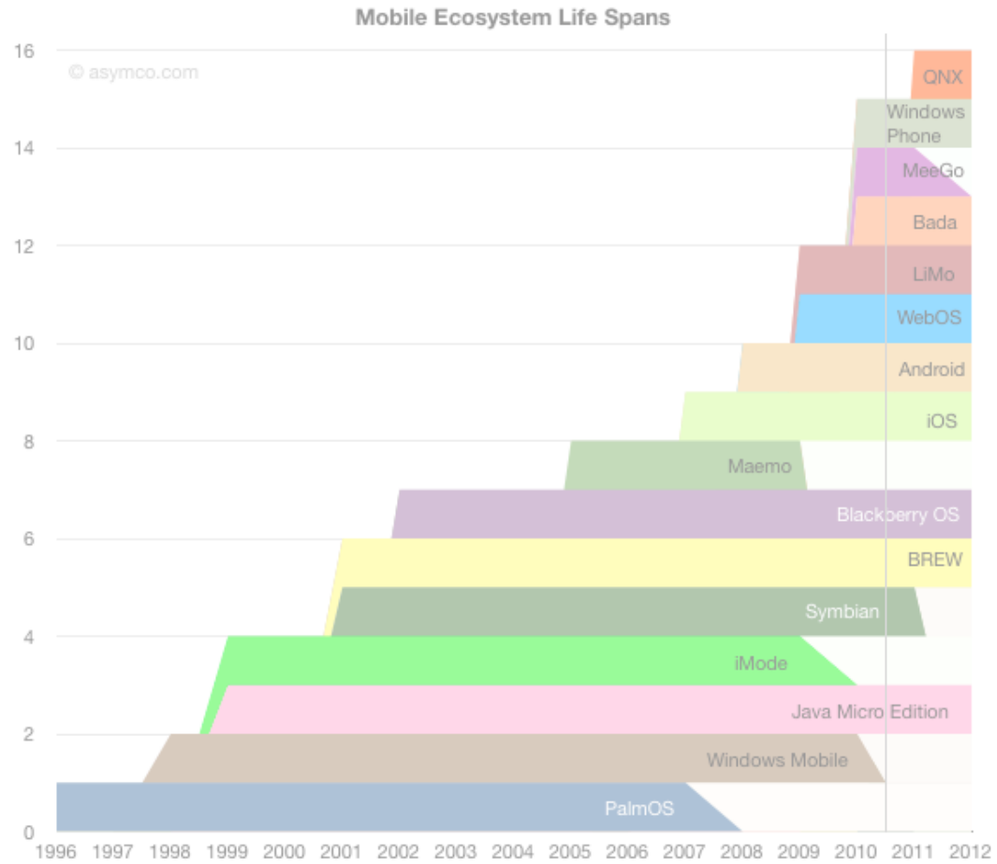


ASYMCO FEB 2011

We must have:

- 2008 - iPhone Apps
- 2009 - Android App

Developing Native Apps for Multiple Platforms

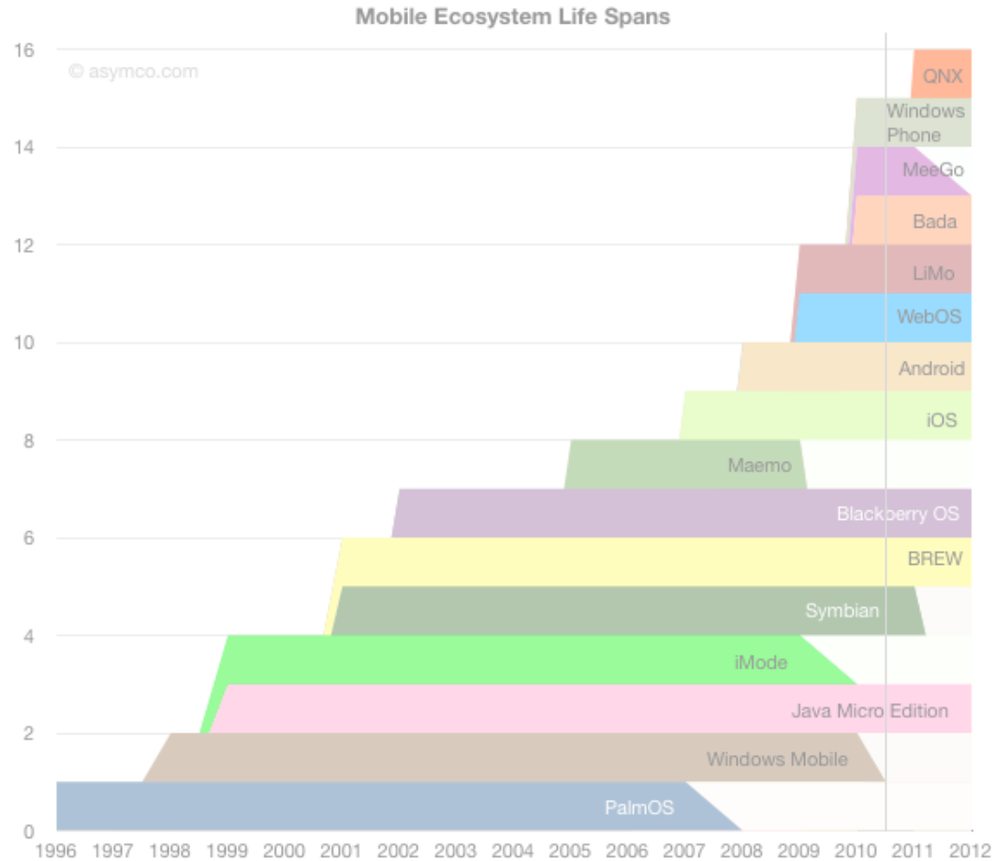


ASYMCO FEB 2011

We must have:

- 2008 - iPhone Apps
- 2009 - Android App
- 2010 - iPad App

Developing Native Apps for Multiple Platforms



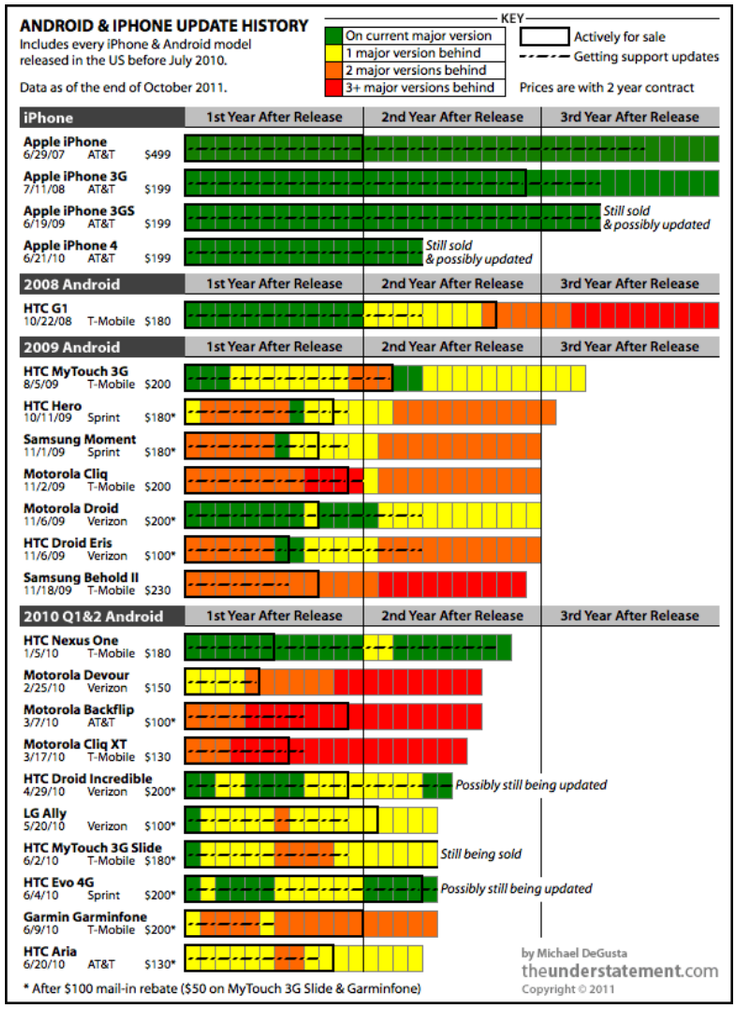
ASYMCO FEB 2011

We must have:

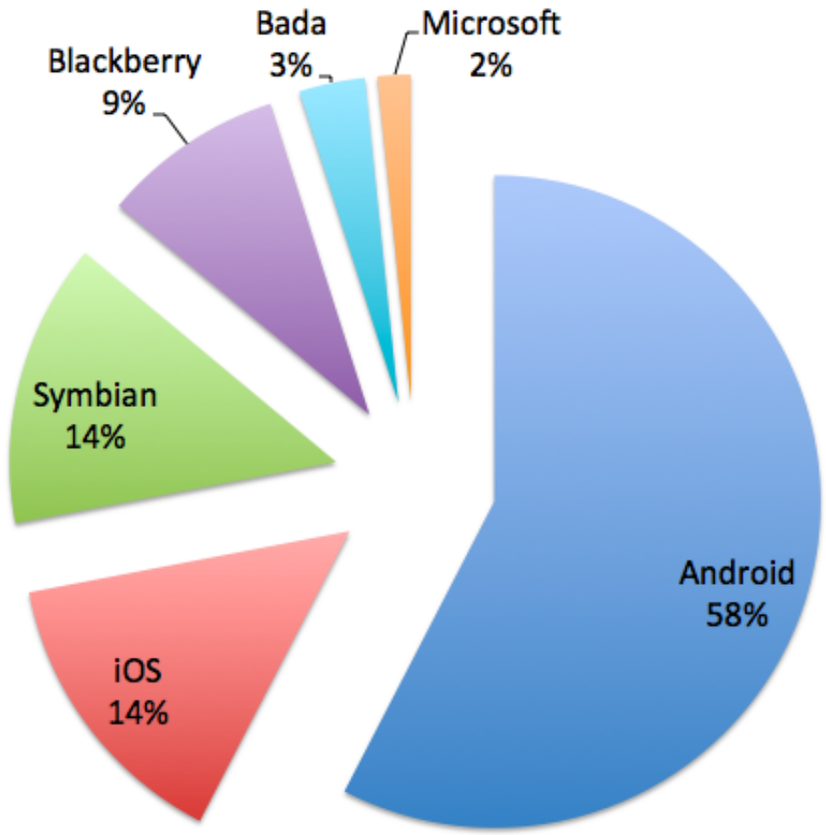
- 2008 - iPhone Apps
- 2009 - Android App
- 2010 - iPad App
- 2011 - maintain them ???

Develop, Test and Maintain

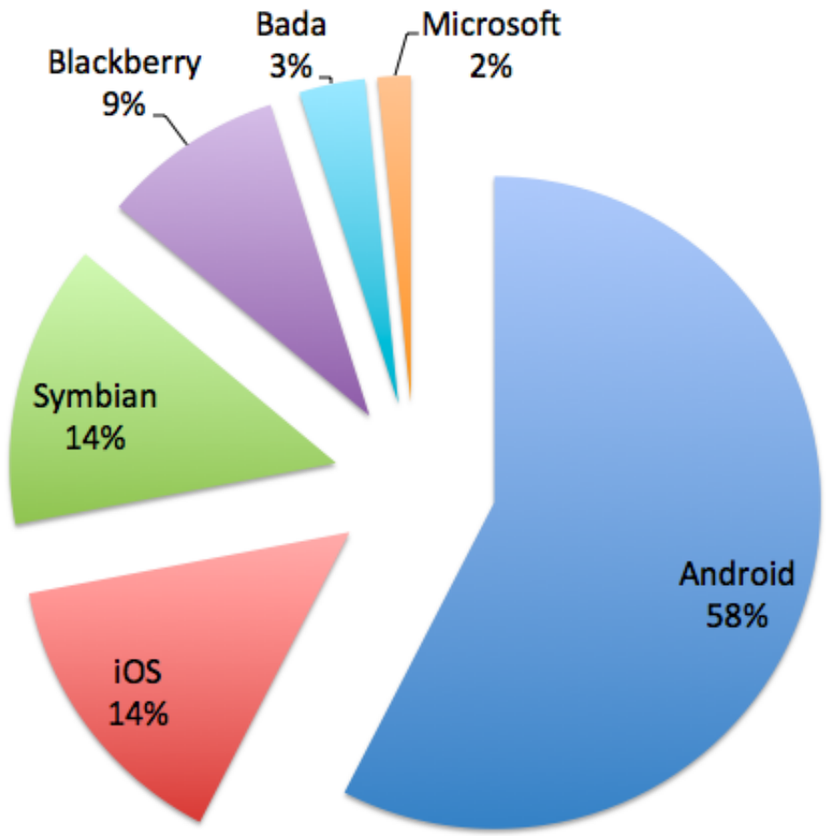
Fragmentation of OS Versions

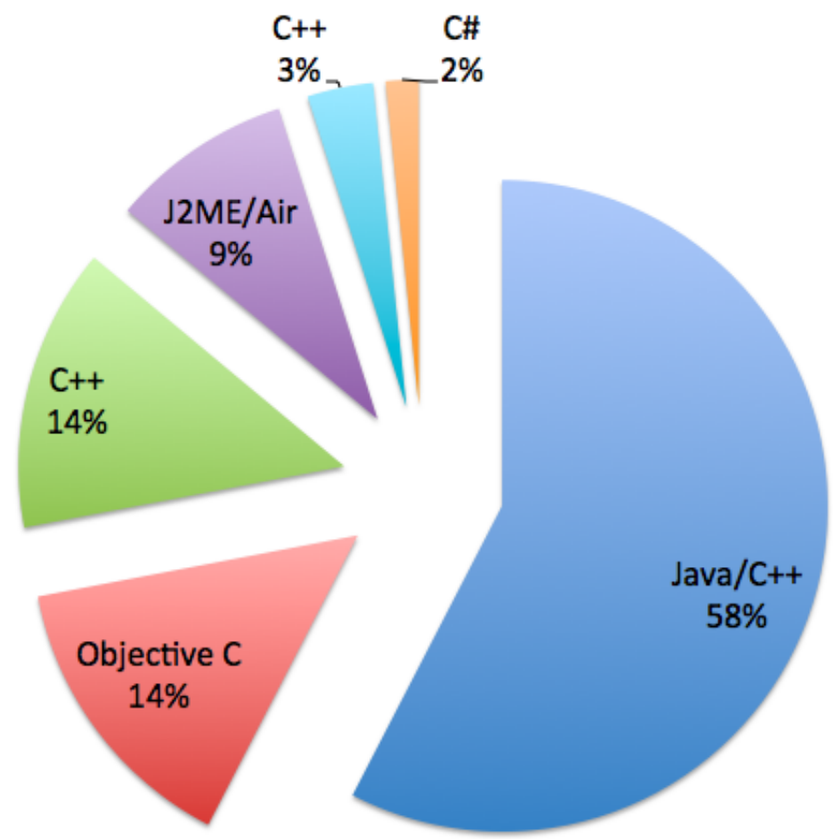


Smartphone Platforms Shares (units shipped Q4 2011)

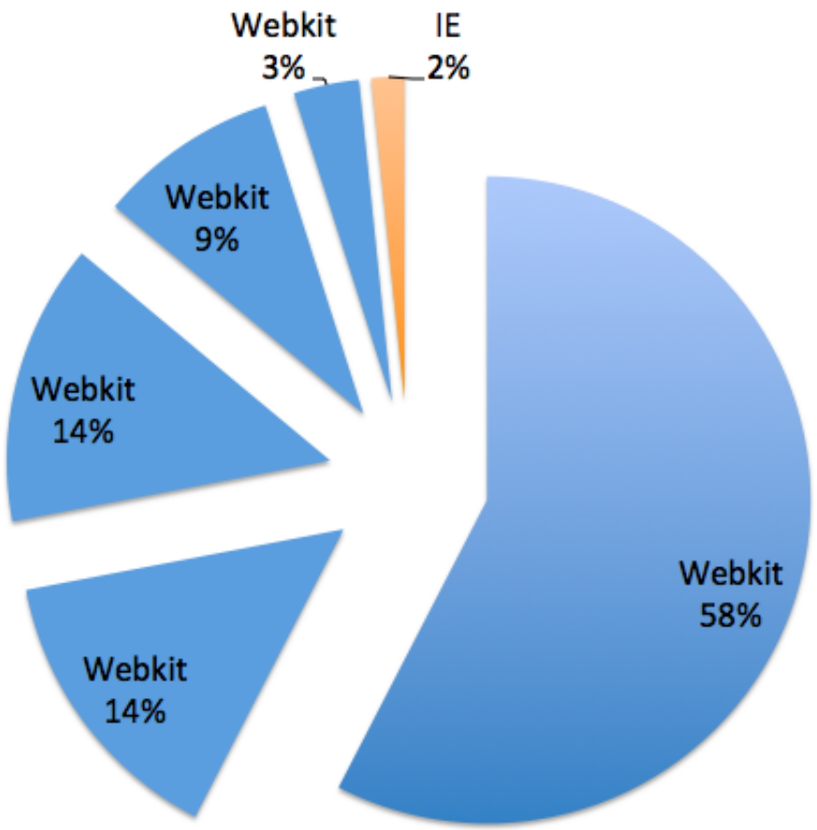


Smartphone Platforms Shares (units shipped Q4 2011)

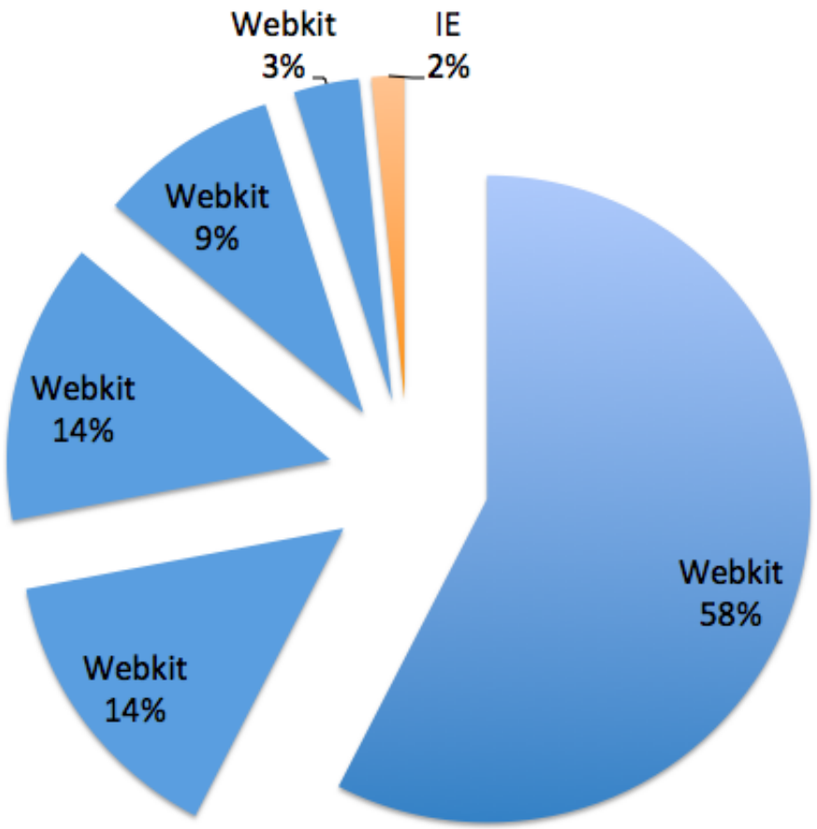


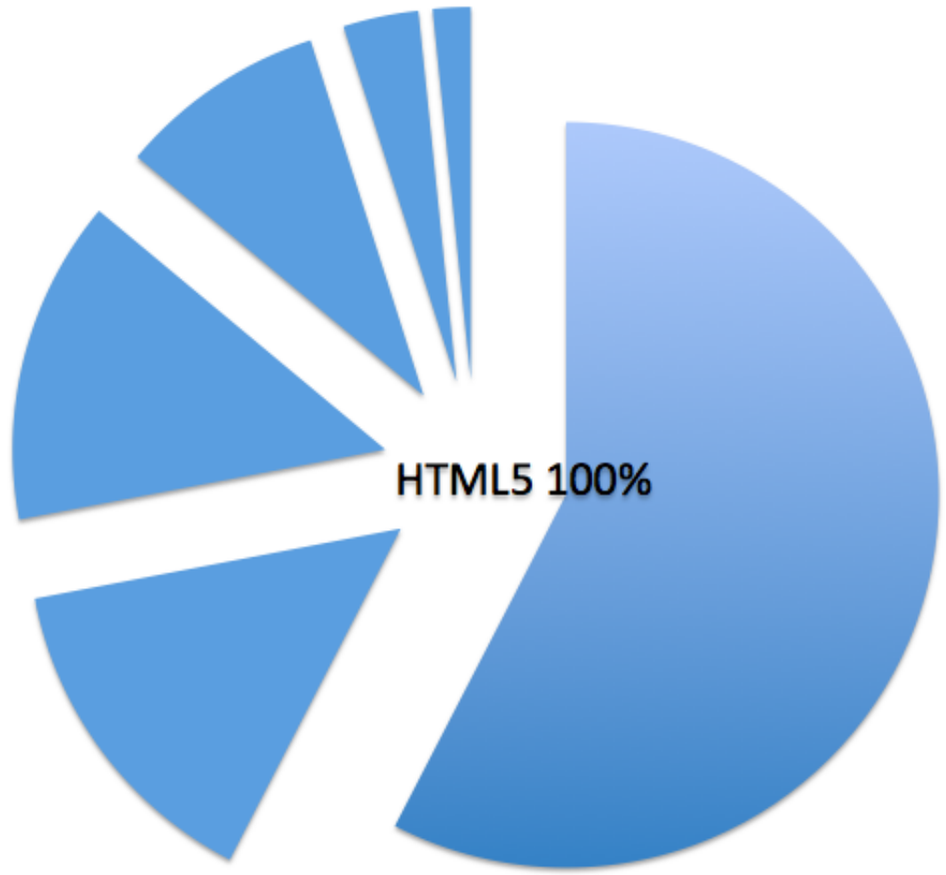


Smartphone Browsers Shares (units shipped Q4 2011)

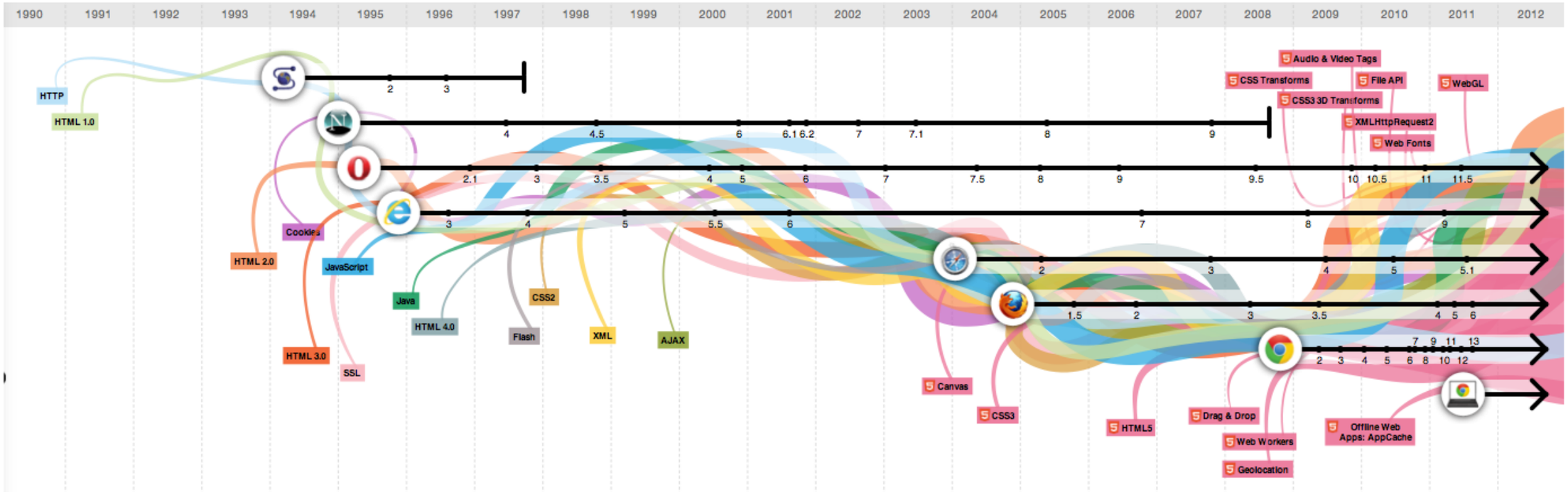


Smartphone Browsers Shares (units shipped Q4 2011)





Evolution of the Web (Browsers)



Web Evolution

Shifting from the "Document Web" to the "App Web"

Shifting from the "Document Web" to the "App Web"

Hyperlinked documents

Page reloading

Static pages

Web servers

Static content

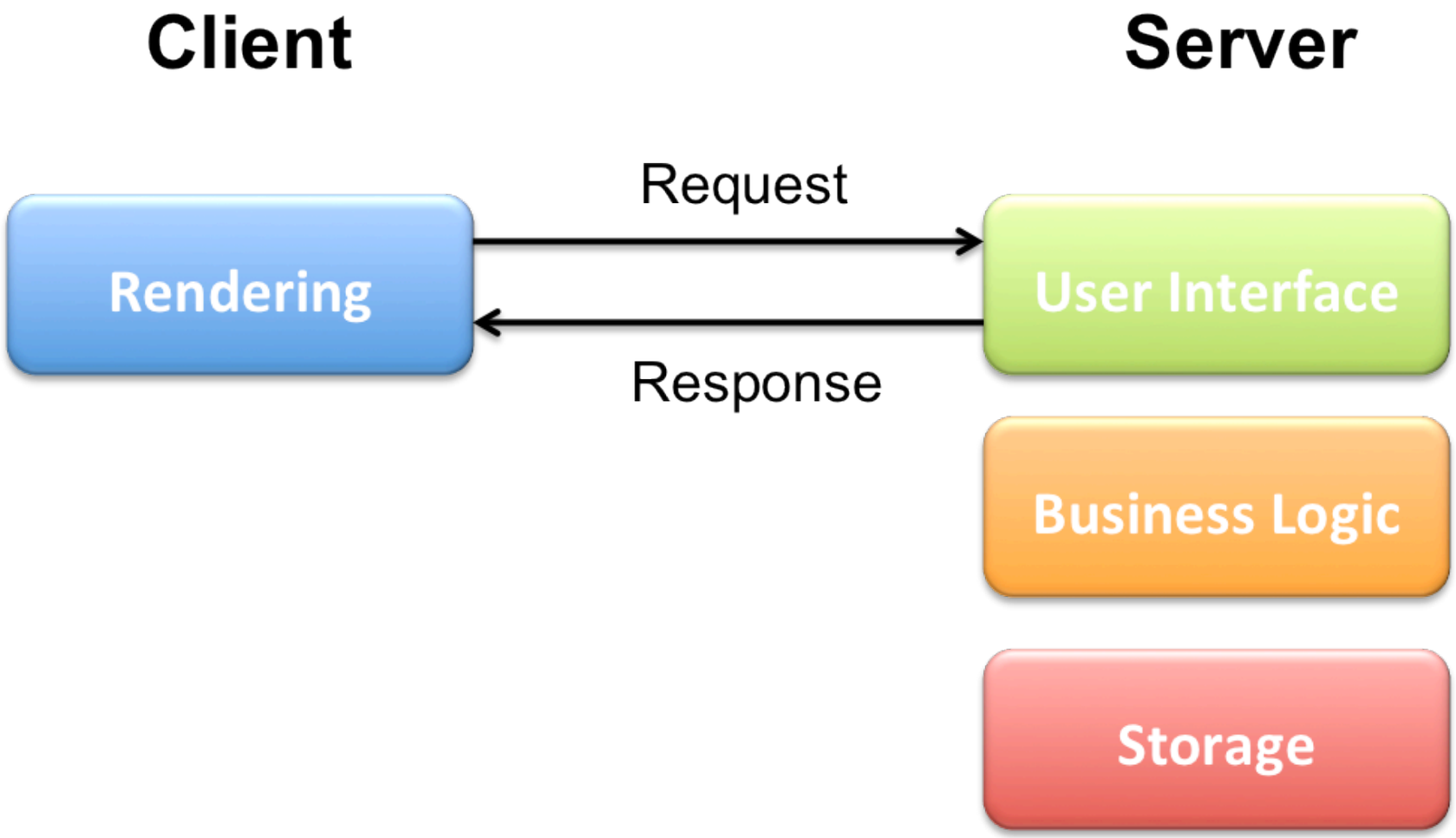
Shifting from the "Document Web" to the "App Web"

Hyperlinked documents	Interactive web apps
Page reloading	Asynchronous API calls
Static pages	HTML + CSS + JS
Web servers	Data hubs (web apis)
Static content	Real-time communication

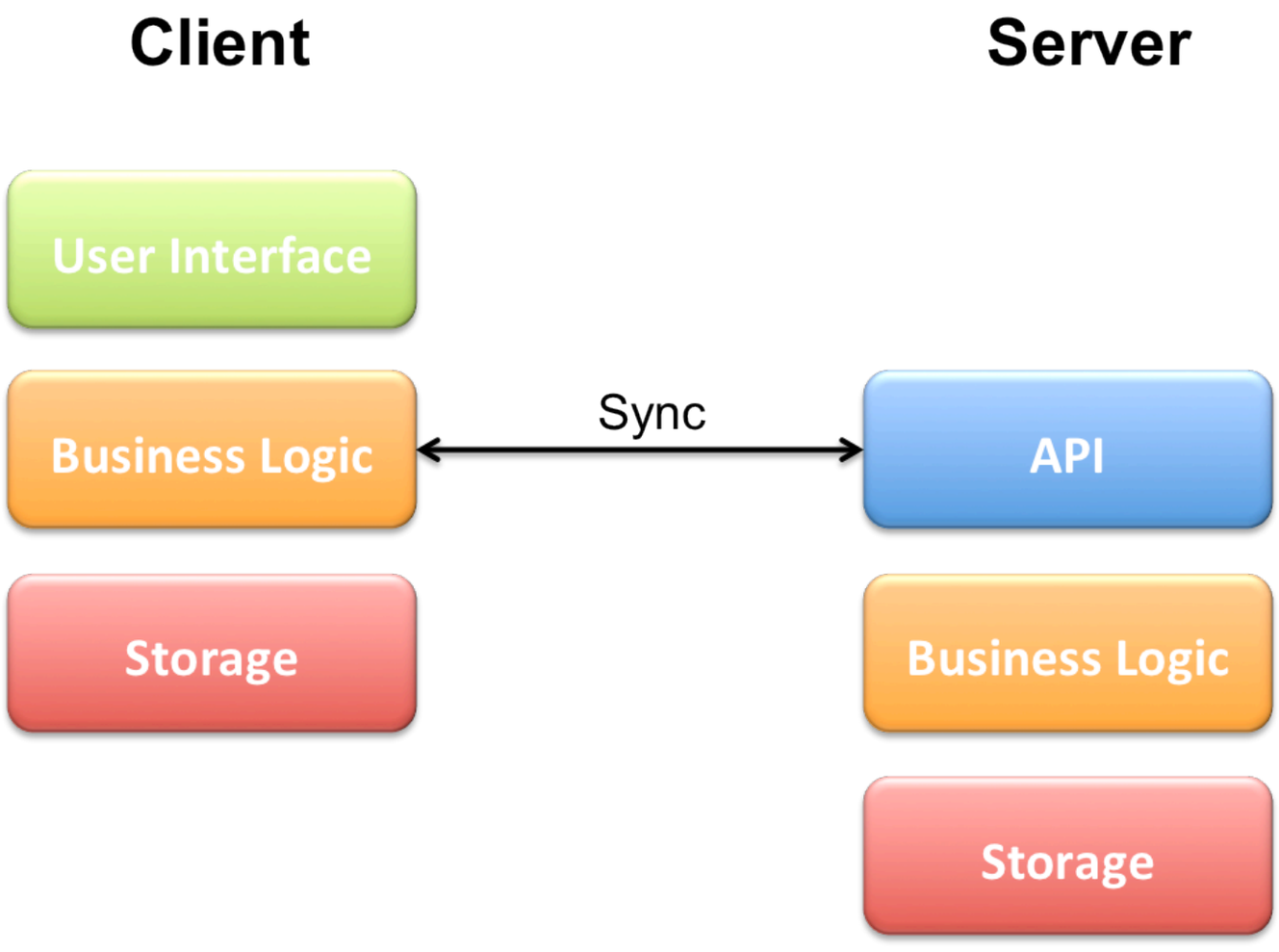
Hyperlinked documents	Interactive web apps
Page reloading	Asynchronous API calls
Static pages	HTML + CSS + JS
Web servers	Data hubs (web apis)
Static content	Real-time communication

Web apps are built with open standards that are refered as "Web Technologies"

The Classic Web Architecture



Towards a New Web Architecture



What is HTML version 5 ?

1998 - W3C decided they will stop evolving HTML beyond version 4.01

1999 - HTML 4.01 becomes a recommendation

2000 - W3C released XHTML 1.0 and force the world to use XML

2002 - W3C released first draft of XHTML 2.0, no backwards compatibility

2004 - WHATWG started working on HTML v5 (Opera, Mozilla and Apple)

2006 - W3C agrees to use WHATWG proposal for HTML5

2009 - W3C stops works on XHTML 2.0 and resources are diverted to HTML5

2012 - HTML5 gets forked up

2014 - HTML 5.0 will become a recommendation (according to W3C)

2016 - W3C plans to release HTML 5.1

Philposphy of HTML5

Philosophy of HTML5

- Specify undocumented features (e.g. XMLHttpRequest)
- Browser behaviour with invalid markup
- Support web applications
- Define an open standard (opposed to Flash)
- Don't break the Web

HTML5 Living Spec at WHATWG

HTML

Living Standard — Last Updated 11 June 2012

Table of contents

HTML5 Specs at W3C

HTML5

- New semantics
- Connectivity
- CSS3
- Device Access
- 3D, Graphics and Effects
- Multimedia
- Offline & Storage
- Performance & Integration

New Semantics

New Semantics
section, header, footer, nav, ...

New Form Controls & Types
date, range, email, url, tel, ...

New Form Validation
by type, required, :valid, :invalid, :required

Offline Storage

Offline Usage

- Install a package on the device
- Buggy on some platforms
- online/offline events
- iOS full screen metatag

Storage

- Persistent and Session Storage
- key/value (strings)
- limited to 5Mb
- IndexDB and SQL storage

Multimedia

Audio and Video Tags

Javascript API & events

Some codecs supported

3D, Graphics and Effects

2D Canvas API

SVG support

WebGL

Device Access

Accelerometer / gyroscope / magnetometer

Orientation change

Touch events (touchstart, touchmove, touchend)

File API and File Reader

(Media Camera API)

CSS3 and Styling

New styling
Rounded borders, shadows, opacity

2D & 3D transforms
rotate, scale, skew, translate

Transitions
basic animations between 2 states

keyframe animations

prefixes (-webkit, -o, -moz, -ms)

WebSockets

Web Sockets

Server-sent Events

Performance & Integration

Notifications API

XMLHttpRequest 2

Web Workers

Browser Support for HTML5

Offline web applications - Working Draft

Method of defining web page files to be cached using a cache manifest file, allowing them to work offline on subsequent visits to the page

*Usage stats:	Global
Support:	57.04%
Partial support:	0.4%
Total:	57.44%

Resources: Sitepoint tutorial | Dive Into HTML5 article | Mozilla Hacks article/demo

Show all versions	IE	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Opera Mobile	Android Browser	
3 versions back	6.0	6.0	13.0	3.2	11.0	3.2		10.0		
2 versions back	7.0	7.0	14.0	4.0	11.1	4.0-4.1		11.0	2.1	
Previous version	8.0	8.0	15.0	5.0	11.5	4.2-4.3		11.1	2.2	
Current	9.0	9.0	16.0	5.1	11.6	5.0	5.0-6.0	11.5	2.3	3.0
Near future		10.0	17.0		12.0				4.0	
Farther future	10.0	11.0	18.0	6.0	12.1					

Note: Currently not supported in some versions of Chrome on Linux.

Feedback

WHEN CAN I USE ...

Test your Browser for HTML5 support

THE HTML5 TEST

Mobile Browsers

Mobile Browsers

Too many

Mobile Browsers

Too many

Some are limited

Mobile Browsers

Too many

Some are limited

Some are too innovative

Mobile Browsers

Too many

Some are limited

Some are too innovative

Some are proxy based

Mobile Browsers

Too many

Some are limited

Some are too innovative

Some are proxy based

Most have little documentation

Mobile Browsers

Too many

Some are limited

Some are too innovative

Some are proxy based

Most have little documentation

Most have little debugging support

Too many

Some are limited

Some are too innovative

Some are proxy based

Most have little documentation

Most have little debugging support

Various input methods (focus, cursor, touch, multi-touch)

Some are limited

Some are too innovative

Some are proxy based

Most have little documentation

Most have little debugging support

Various input methods (focus, cursor, touch, multi-touch)

API support

Mobile HTML5 Compatibility Table

Feature	Safari on iOS	Android Browser		BlackBerry Browser		Nokia Browser		Internet Explorer	Opera		Firefox	webOS Browser
Version tested	iPhone, iPad	Phones (1-2.3, 4.0)	Tablets (3.0+)	Phones	Tablet	Meego - Nokia N9	Symbian	Windows Phone	Mobile	Mini	Android	
Minimum version tested	3.2	1.5	3.0	5.0	1.0	1.2	^3	9	11	5	6	1.4
Application Cache W3C API Offline package installation.	✓	✓ 2.1+	✓	✓ 6.0+	✓	✓			✓		✓	✓
Web storage W3C API Persistent and session storage.	✓	✓ 2.0+	✓	✓ 6.0+	✓	✓		✓	✓		✓	✓
Web SQL storage W3C API (no active) Persistent SQLite storage.	✓	✓ 2.0+	✓	✓ 6.0+	✓	✓			✓			✓
Geolocation W3C API Geolocation & tracking using GPS, cells or Wi-Fi.	✓	✓ 2.0+	✓	✓ 6.0+	✓	✓		✓	✓		✓	✓
Multimedia W3C API Video & Audio Players	✓	✓ 2.3+	✓	✓ 7.0+	✓	✓		✓	✓		✓	✓
Server-Sent Events W3C API EventSource pattern to mantain the connection to the server open	✓ 4.1+					✓			✓		✓	
Web Sockets W3C API Newbidireccional protocol over HTTP	✓ 4.2+			✓ 6.1+	✓				✓		✓ 7+	

Mobile Web Challenges

Mobile Web Challenges

Adaptation and Optimization

Mobile Web Challenges

Adaptation and Optimization

Responsive design and server-side detection

Mobile Web Challenges

Adaptation and Optimization

Responsive design and server-side detection

Mobile usability

Mobile Web Challenges

Adaptation and Optimization

Responsive design and server-side detection

Mobile usability

Touch vs Click

Mobile Web Challenges

Adaptation and Optimization

Responsive design and server-side detection

Mobile usability

Touch vs Click

The viewport, full screen and pixel density

Mobile Web Challenges

Adaptation and Optimization

Responsive design and server-side detection

Mobile usability

Touch vs Click

The viewport, full screen and pixel density

Semantic HTML5

Mobile Web Challenges

Adaptation and Optimization

Responsive design and server-side detection

Mobile usability

Touch vs Click

The viewport, full screen and pixel density

Semantic HTML5

Mobile Web is slow

Mobile Web Challenges

Adaptation and Optimization

Responsive design and server-side detection

Mobile usability

Touch vs Click

The viewport, full screen and pixel density

Semantic HTML5

Mobile Web is slow

- Reduce HTTP request
- Leverage CSS3 and GPU
- Inline images
- Optimize JS performance
- Hybrid apps

HTML5 Frameworks & Tools

UI

HTML

Elements
forms

2d Canvas
Svg
Webgl

Styling
Transformations
Animations
Transitions

jQuery mobile framework

Sencha

PhoneGap
Notifications
(alert, sound,
vibration)

Logic

HTML

Touch events
Orientation

Notifications
Web Workers
XHR 2

BACKBONE.JS

handlebars

Data

HTML

Offline usage
Storage
IndexDB

Web sockets
Server-sent
events

File API

File API
Contacts API

PhoneGap

Device

HTML

Accelerometer
Geolocation
Media Capture API

Multimedia

Accelerometer
Media
Camera
Compass
Network

PhoneGap

Backend

RAILS

Sinatra

node.js

django

CoffeeScript

Haml.

Sass.

Database

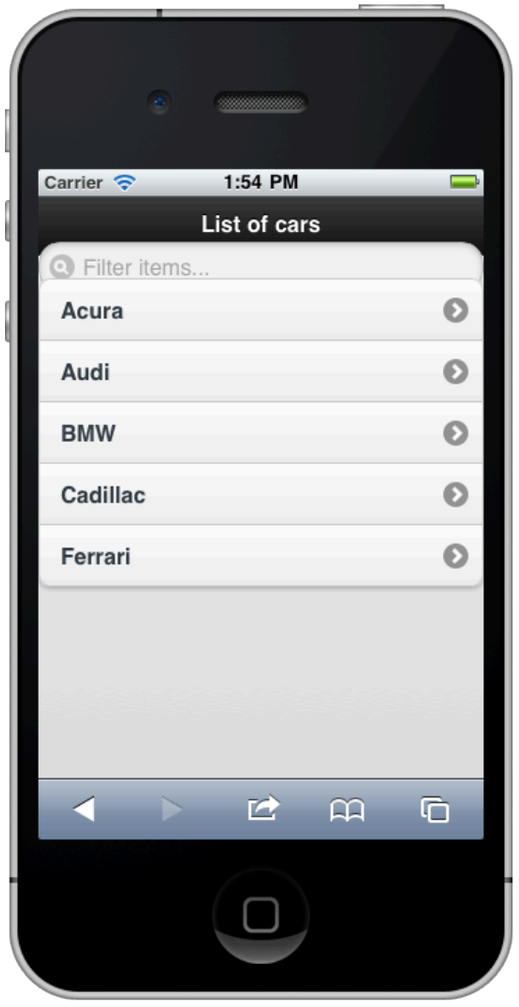
mongoDB

apache CouchDB

jQuery Mobile Example jQuerySimple

```
<!DOCTYPE html>
<html>
  <head>
  <title>Hello World</title>
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="http://code.jquery.com/mobile/1.0/jquery.mobile-1.0.min.css" />
  <script type="text/javascript" src="http://code.jquery.com/jquery-1.6.4.min.js"></script>
  <script type="text/javascript" src="http://code.jquery.com/mobile/1.0/jquery.mobile-1.0.min.js"></script>
</head>
<body>
  <div data-role="page">
    <div data-role="header">
      <h1>List of cars</h1>
    </div><!-- /header -->
    <ul data-role="listview" data-inset="true" data-filter="true">
      <li><a href="#">Acura</a></li>
      <li><a href="#">Audi</a></li>
      <li><a href="#">BMW</a></li>
      <li><a href="#">Cadillac</a></li>
      <li><a href="#">Ferrari</a></li>
    </ul>
  </div><!-- /page -->
</body>
</html>
```

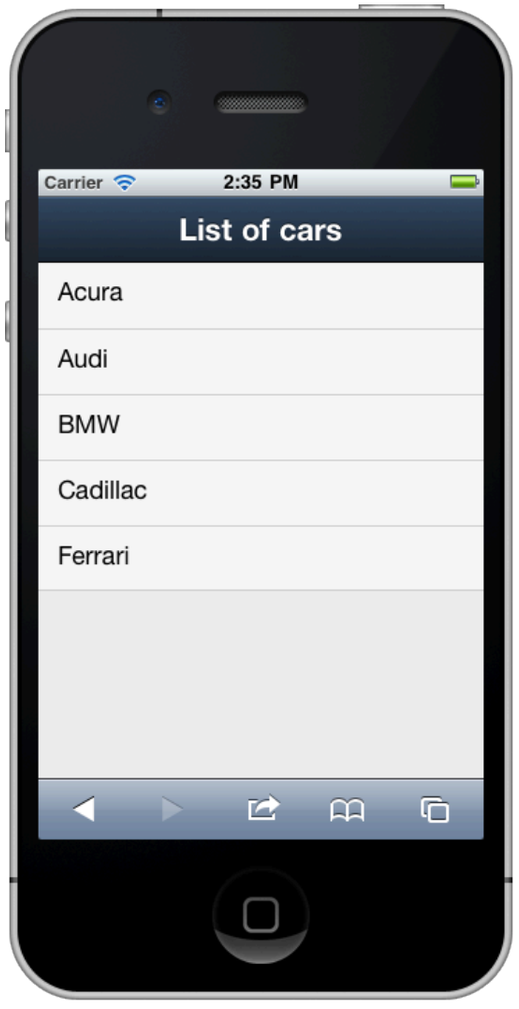
jQuerySimple on a iPhone and iPad

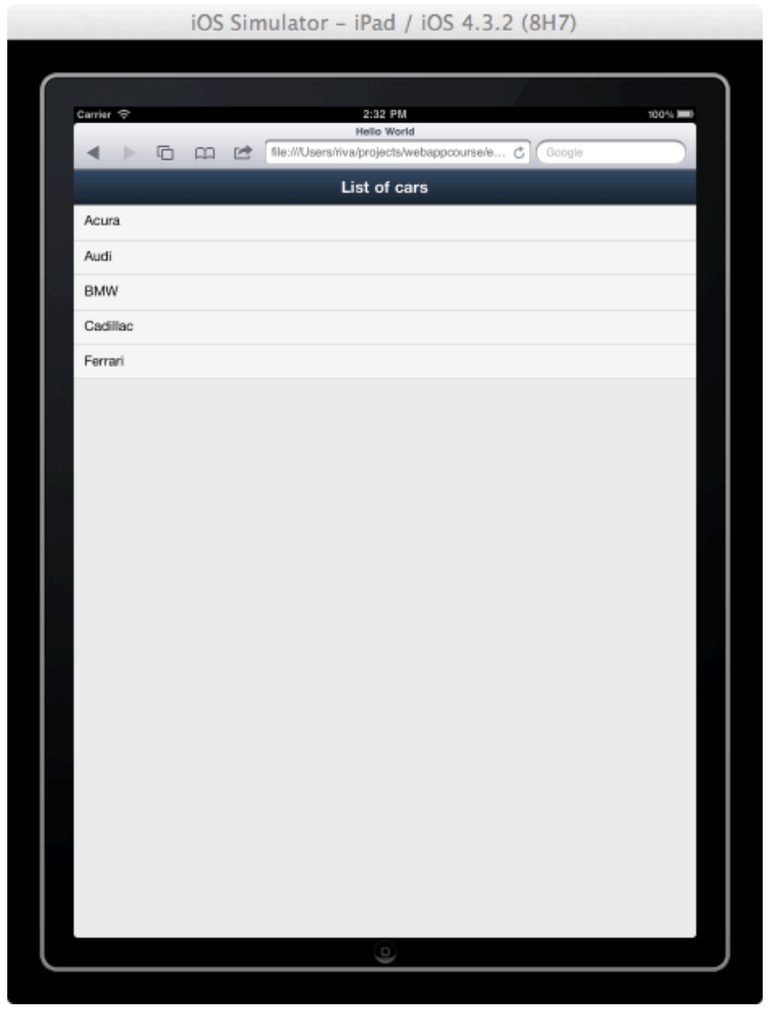


Sencha Example senchaSimple

```
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
        <title>Hello World</title>
        <script src="sencha-touch.js" type="text/javascript"></script>
        <link href="sencha-touch.css" rel="stylesheet" type="text/css" />
         <script type="text/javascript">
            new Ext.Application({
                launch: function() {
                    var cars = new Ext.data.Store({
                        model: Ext.regModel('', {fields: ['name', 'link']}),
                        data: [
                            {name: 'Acura', link:'na'},
                            {name: 'Audi', link:'sa'},
                            {name: 'BMW', link:'eu'},
                            {name: 'Cadillac', link:'eu'},
                            {name: 'Ferrari', link:'eu'}
                        ]
                    });
                    new Ext.Panel({
                        fullscreen: true,
                        dockedItems: [{ xtype: 'toolbar', title: 'List of cars', }],
                        items: [{ xtype: 'list', store: cars, itemTpl: '{name}' }]
                    });
                }
            });
        </script>
    </head>
    <body></body>
</html>
```

senchaSimple on a iPhone and iPad





Organization of the course

Lectures : Introduction of key concepts, technologies and frameworks

Labs : Hands-on classes with practical exercices on selected topics

Challenge : Team work development of a mobile web app

Lectures & Labs

Lecture 1: Introduction	Lab 1: The Challenge Kick-off
Lecture 2: jQuery Mobile	Lab 2: Concepting the Webapp
Lecture 3: Templating and MVC	Lab 3: The first mobile web app
Lecture 4: CSS3 and Responsive UI	Lab 4: Structuring your web app
Lecture 5: HTML5 APIs	Lab 5: Work on the challenge
Lecture 6: Websockets & Webworkers	Lab 6: Lab exercise
Lecture 7: Developing native apps	Lab 7: Work on the challenge

Course Material and Communication

Slides are available at aaltowebapps.com

The slides are best viewed with Google Chrome

Exercises are available at GitHub

Communication channels:

- Twitter: @aaltowebapps
- Noppa

The Fundamentals

Git & GitHub

Javascript & Json

jQuery

Ruby & Sinatra

HAML

Tools

Git

Start using git

```
cd yourproject
git init
```

```
git clone git@github.com...
```

Add all files in a directory

```
git add .
```

Check status

```
git status -s
```

Commit

```
git commit -m "First commit"
git commit -am "First commit"
```

Show all branches

```
git branch
git branch -a
```

Create a new branch

```
git branch newfeature
```

Switch branch

```
git checkout newfeature
```

Add a remote

```
git remote add
```

View the remotes

```
git remote -v
```

Link a local to a remote branch

```
git branch --track local remote
```

Fetch changes from remotes

```
git fetch
```

Push changes to remotes

```
git push
```

Review changes between different branches

```
git log origin/master ^master
```

Merge changes

```
git checkout master
git merge origin/master
```

Must Read: Git Reference

GitHub

A social hub for:

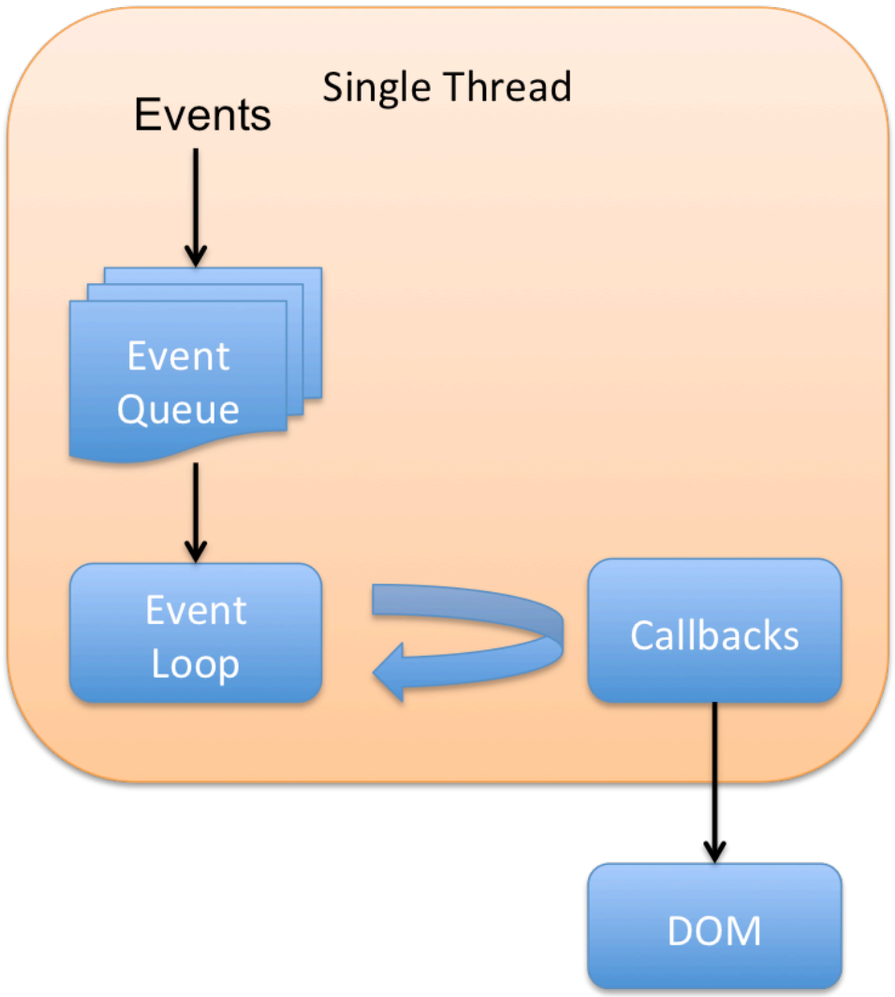
- Source code repository
- Issues
- Wiki
- Pull requests

It's important that you learn how to

- Branch
- Create Pull Requests

Commit early and commit often.

Javascript Event Loop



- One thread of execution. If a function is blocked, the UI is frozen.
- Asynchronous operations
- Blocking I/O operations must not block the UI thread (callbacks)
- Keep callbacks short and fast
- Use closures for creating callbacks

Javascript: Asynchronous calls

Javascript has one thread of execution

Blocking operations (e.g. network) must use callbacks

```
var jqxhr = $.getJSON("example.json", function() {
  alert("success");
});
```

```
$('#target').click(function() {
  alert('Handler for .click() called.');
});
```

Javascript: Lambda Functions

Lambda functions are created at runtime when the execution reaches that point of the flow. This allows functions to be defined conditionally

Function Declaration

```
...
function square(x) {
  return x * x;
}
var b = square(2); //b gets assigned 4
...
```

Function Statement

```
...
var square = function(x) {
  return x * x;
};
var b = square(2); //b gets assigned 4
...
```

Lambda functions are normal objects and can be passed as function parameters (e.g. callbacks)

```
$('#target').click( square );
```

Javascript: Iterators (jQuery)

$.each : iterates over a list

```
var list = [1,2,3,4,5];
var a = $.each(list, function(index, value) {
  alert(index + ": " + value);
}); //It produces 5 alerts
```

or a map

```
var map = {'a': 'b', 'c': 'd'};
var a = $.each(list, function(key, value) {
  alert(key + ": " + value);
}); //It produces 2 alerts
```

$.map : applies a function to each element of the array and maps the results into a new array

```
var list = [1,2,3,4,5];
var a = $.map(list, function(value, index) {
  return square(value);
}; // a gets [1,4,9,16,25]
```

Javascript: Iterators (underscore.js)

each : _.each (list, iterator, [context])

```
_.each([1, 2, 3], function(n){ alert(n); });
_.each({one : 1, two : 2, three : 3},
function(n, k){ alert(n); });
```

find : _.find(list, iterator, [context])

```
_.find([1, 2, 3, 4, 5, 6], function(n){ return n
% 2 == 0; });
=> 2
```

map : _.map(list, iterator, [context])

```
_.map({one : 1, two : 2, three : 3},
function(n, k){ return n*3; });
=> [3, 6, 9]
```

filter : _.filter(list, iterator, [context])

```
_.filter([1, 2, 3, 4, 5, 6], function(n){ return
n % 2 == 0; });
=> [2, 4, 6]
```

reduce : _.reduce(list, iter, memo, [context])

```
var sum = _.reduce([1, 2, 3], function(memo, n)
{ return memo + n; }, 0);
=> 6
```

groupBy : _.groupBy(list, iterator)

```
_.groupBy(['one', 'two', 'three'], 'length');
=> {3: ["one", "two"], 5: ["three"]}
```

Javascript: Closure

A closure is formed by returning a function object that was created within an execution context of a function call from that function call and assigning a reference to that inner function to a property of another object. Or by directly assigning a reference to such a function object to, for example, a global variable, a property of a globally accessible object or an object passed by reference as an argument to the outer function call (Closures).

```
function closureBuilder(arg1, arg2){  //outer function (builder)
    var localVar = 8;
    function exampleReturned(innerArg){  //inner function
        return ((arg1 + arg2)/(innerArg + localVar));
    }
    return exampleReturned;  //return a reference to the inner fucntion
}
var globalVar = closureBuilder(2, 4);
var secondGlobalVar = exampleClosureForm(12, 3);

globalVar(2);   //Result ?
secondGlobalVar(5);   // Result ?
```

Javascript: Closure Example with Google Maps

```
var infoWindow = new google.maps.InfoWindow();

function addInfoWindow(marker, id, name) {
  google.maps.event.addListener(marker, 'click', function () {
      infoWindow.close();
      infoWindow.setContent('<p>'+ name + '</p>');
      infoWindow.open(map, marker);
    });
}

function addMarkers(json) {
  var markers = [];
  for (i=0; i<json.length; i++) {
      var place = json[i];

      var point = new google.maps.LatLng(place.coordinates[0], place.coordinates[1]);
      var marker = new google.maps.Marker({position:point, title:place.name});

      addInfoWindow(marker, place.slug, place.name);
      ...
```

Javascript Object Notation (JSON)

Lightweight data-interchange format

Human readable and easy to comprehend

Easy for machines to parse

Array

```
["first", "second", 3, 4]
```

Hash table

```
{"name":"John", "address":"London", "age":38}
```

Serialize/Deserialize

```
JSON.stringify( ['first', 'second', 3, 4] );
=> ["first","second",3,4]

JSON.parse( ' ["first", "second", 3, 4] ' );
=> Object
```

jQuery more info

Fast Javascript library for **document traversing** , **event handling** , **animations** and **Ajax interactions**

HTML

```
...
<body>
  <div id="button">Click me</div>
</body
```

JS

```
$(document).ready(function(){
  $("#button").click( function(event) {
    alert("Button pressed");
    $("#button").toggleClass("red");
    $("#button").fadeOut().fadeIn();
  });
});
```

Test here: Click me

Ruby

Ruby is a dynamic programming language that combines functional and imperative programming (a mix of Perl, Smalltalk, Eifel, Ada and Lisp)

Online resources to start with:

- Ruby in Twenty Minutes
- Ruby on One Page
- Enumerables
- Programming Ruby

Ruby Blocks

```ruby
3.times { puts 'Hello' }
Hello
Hello
Hello
=> 3
```

```ruby
array = [1, 2, 3, 4]
array.collect do |n|
  n ** 2
end
=> [1, 4, 9, 16]
```

```ruby
@names = %w(John Anne Lukas)
@names.each do |name|
  puts "Hello #{name}!"
end
Hello John!
Hello Anne!
Hello Lukas!
=> ["John", "Anne", "Lukas"]
```

```ruby
#iterator for Fibonacci numbers
def fibUpTo(max)
    n1, n2 = 1, 1
    while n1 <= max
        yield n1  # invoke block with value
        n1, n2 = n2, n1+n2  # and calculate next
    end
end

fibUpTo(1000) { |term| print term, " " }
=>
1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987

a = []
fibUpTo(1000) { |term| a << term }
a
=> [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144,
233, 377, 610, 987]
```

Sinatra more info

Sinatra is a DSL for quickly creating web applications in Ruby

Single file web app

Runs on Rack (fast Ruby web server)

Supports many template engines

```ruby
require 'sinatra'

get '/hi' do
  "Hello World!"
end
```

Sinatra - HTTP Methods (REST)

```ruby
get '/' do
  .. show something ..
end

post '/' do
  .. create something ..
end

put '/' do
  .. replace something ..
end

delete '/' do
  .. annihilate something ..
end
```

Sinatra - Routes

Simple Routes

```
get '/hello' do
  'Hello World'
end
```

Named parameters

```
get '/hello/:name' do
  # matches "GET /hello/foo" and "GET /hello/bar"
  # params[:name] is 'foo' or 'bar'
  "Hello #{params[:name]}!"
end
```

Sinatra - Views

```ruby
# erb => renders /views/index.erb (using /views/layout.erb if it exists)
get '/' do
  erb :index
end
```

```ruby
#haml => renders /views/index.haml embedded in the views/post.haml
get '/' do
  haml :index, :layout => :post
end
```

```ruby
#Inline templates
get '/' do
  haml '%div.title Hello World'
end
```

HAML more info

Template Language

- Mix Ruby code with HAML
- Haml generates HTML code

Clarity

- Indentation = structure
- Tags begin with %
- Tags close themselves
- Use of hashes for attributes

HAML - Syntax

```
%tag content
```

```
<tag>content</tag>
```

```
%h1 Hello HAML
%p HAML is
%ul
  %li Beautiful
  %li Easy
  %li Well-indented
```

```
<h1>Hello HAML</h1>
<p>HAML is</p>
<ul>
  <li>Beautiful</li>
  <li>Easy</li>
  <li>Well-indented</li>
</ul>
```

HAML - Attributes

```
%a{href => "http://bbc.com", id => "title"}
  BBC
```

```
<a href="http://bbc.com" id="title">BBC</a>
```

```
%a(href="http://haml-lang.com" id="title")
  BBC
```

```
<a href="http://bbc.com" id="title">BBC</a>
```

HAML - Shortcuts

```
%p{class => "bio"} Hello
%p.bio Hello
```

```
<p class = "bio">Hello</p>
```

```
%p{id=> "title"} Hello
%p#title Hello
```

```
<p id = "title">Hello</p>
```

```
#title
  .red
    %p Hello
```

```
<div id = title>
  <div class = red>
    <p>Hello</p>
  </div>
</div>
```

HAML - Ruby Evaluation

```
%h1 My Blog
- for article in @articles
  .post
    .title
      %h1= article.title
    .date
      =article.date.strftime("%d %m %Y")
    .article
      %p= article.body
```