A mobile web app technology stack

+Boris Smus

@borismus

Outline

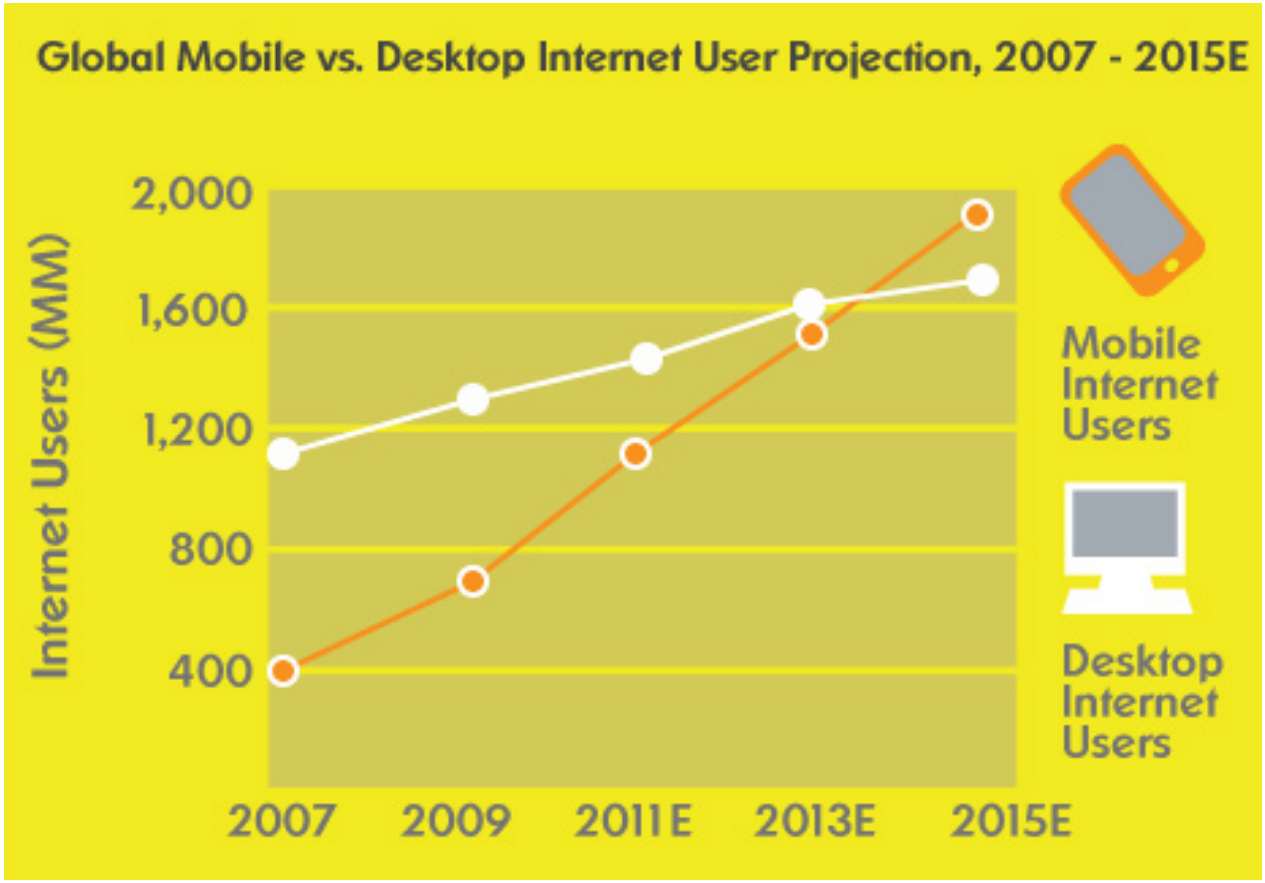
Google™

State of mobile web
Design philosophy
Building a mobile web application
Application demo

Mobile trends

Google

Mobile internet usage to overtake desktop by 2014



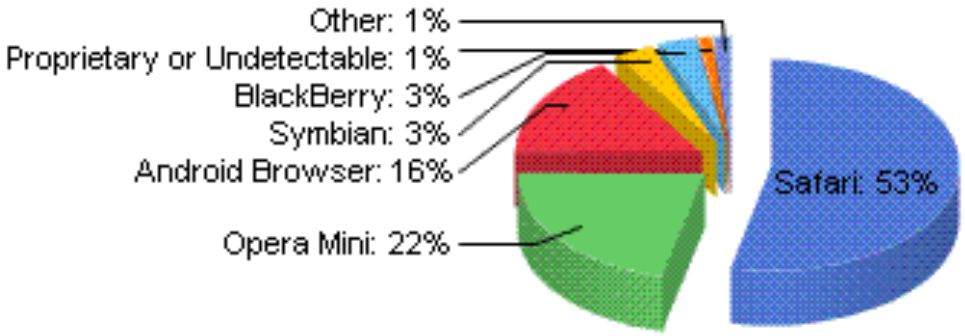
Mobile browsers

Google™

Mobile is 95% WebKit

Missing features:

- WebGL
- Camera
- Indexed DB
- Web Audio
- others



iOS 5 MobileSafari is great
Android 4 Browser improving
Opera Mini not a true browser

Web vs native

Google

Web as a unified platform - build for single target
Native is a moving target: iOS, Android, WP7, etc

Native pushes the boundary (closer to the metal)
Standardized web moves more slowly, but catching up!

The case for enterprise

Google™

Largely data-driven applications!

Examples:

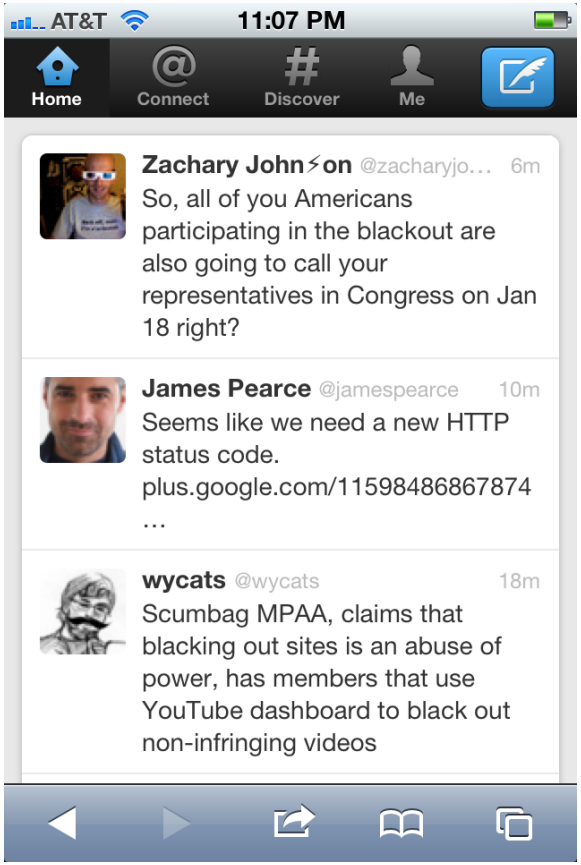
- app.ft.com
- mobile.twitter.com

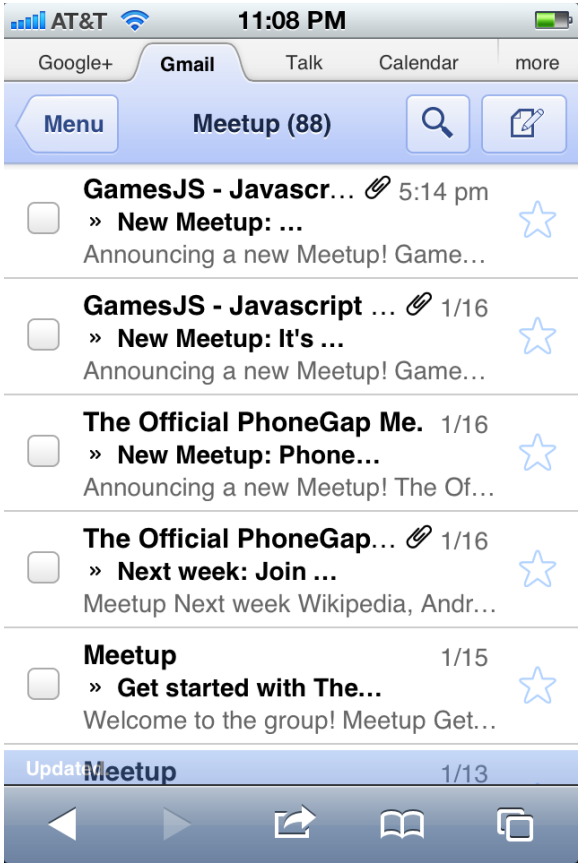
The mobile web is ready for apps.

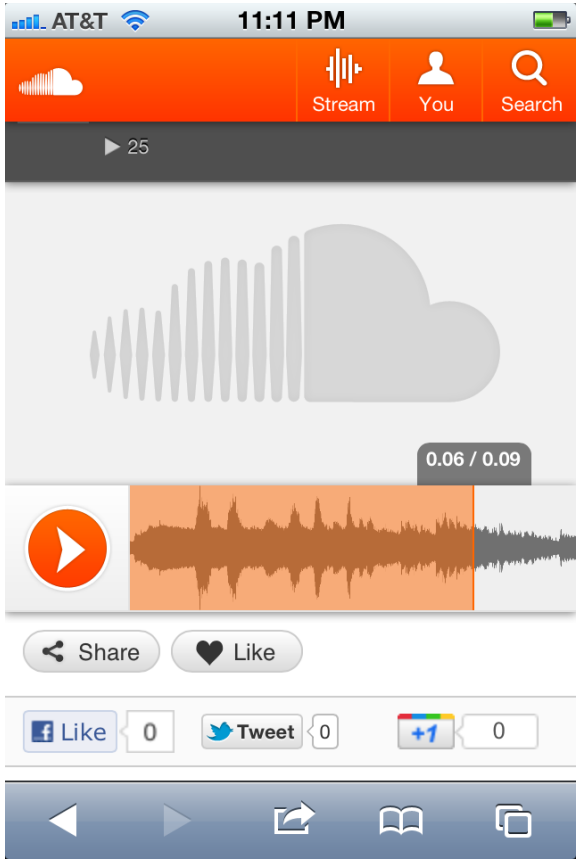
Users may even prefer web to native
Some reports claim that 87% actually do.

Great examples

Google™







Designing

Design is important

Google

Think about it up-front!

Don't let "web" and "enterprise" be an excuse for poor UX

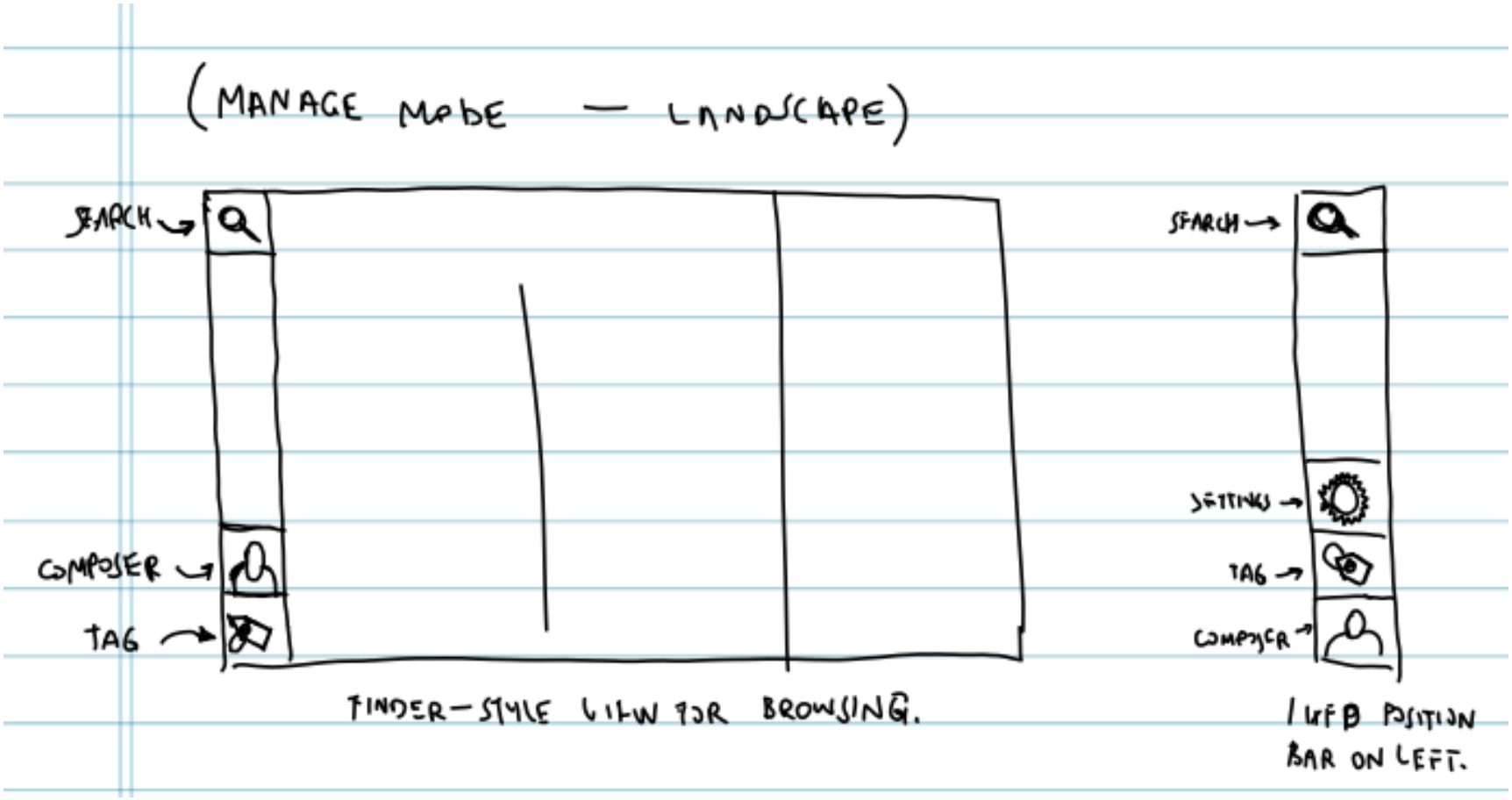
Be inspired by mobile patterns. Don't reinvent the wheel:

http://pttrns.com/

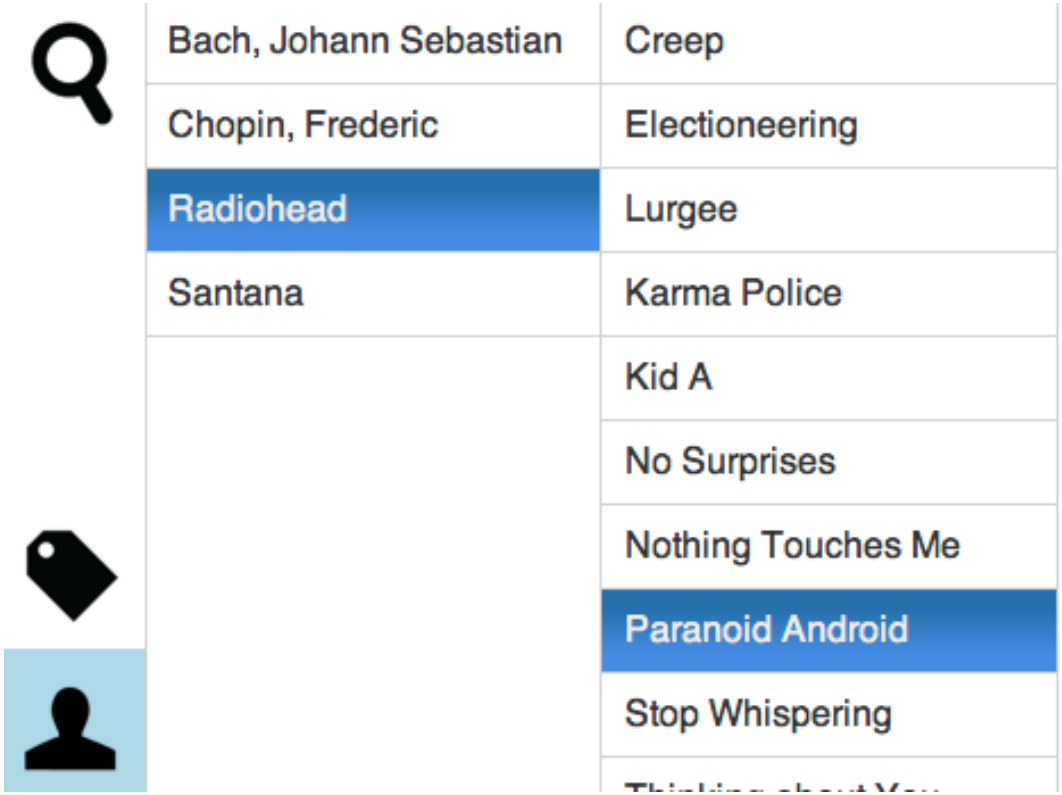
http://mobile-patterns.com/

Low fidelity

Google™



Mid fidelity




```
Paranoid Android
by
Radiohead

Chords:
A#sus5  A7Sus4  Dm7  Dm*
  -0-    -0-     -1-  -0-
  -3-    -3-     -3-  -3-
  -3-    -0-     -2-  -2-
  -2-    -2-     -0-  -3-
  -0-    -0-     -3-  -0-
  -0-    -0-     -0-  -0-

Intro:
Cm   Eb  F  Eb  Gm  Bb
A#sus5  A7Sus4 Gm  Bb
A#sus5  A7Sus4

Verse:
Cm       Eb        F
Eb          (Gm        Bb
A#sus5 A7Sus4) x2
Please could you stop
the noise, I'm trying to
```

Tablet != Phone != ...

Philosophy: Adaptive Apps

Google™

Too hard to use one DOM tree for all form factors

Adaptive apps – Custom views and templates for each form factor, shared model

Use responsive design within the form factor (eg. same layout on iPad 4:3 and Galaxy Tablet 16:9)

Building

Model view controller

Google

Fundamental pattern for separating concerns

Model handles data and server persistence layer
View handles user input and rendering

Used to be on the server. Now moving to the thicker client.

Paradox of choice

Google™

There are many MVC frameworks

TodoMVC - one app written in all of them
Comparison blog post - a high level comparison

I use Backbone.js for relatively simple apps

Templating engines

Google

Complex apps require complex DOM
DOM manipulation is relatively slow

Answer: JS templating

1. Embed `<script id="my-template" type="text/my-template-language">` into HTML, with text contents of the template.

2. Use template library to populate template with data.

Mustache.js

Google

Mustache is a logic-less templating engine

```
{{#items}}
  {{#link}}<li><a href="{{url}}">{{name}}</a></li>{{/link}}
{{/items}}
```

template

+

```
{
  "items": [{"name": "green", "link": true, "url": "#Green"},
      {"name": "blue", "link": true, "url": "#Blue"}],
}
```

data

=

```
<li><a href="#Green">green</a></li>
<li><a href="#Blue">blue</a></li>
```

Mustache.to_html(template, data);

CSS Frameworks

Google™

Augmented CSS-style languages

- $variables: true
- .nesting { .allowed { font-color: bold; }}
- mixins/inheritance

Many variants of syntax, but basically the same.

My preference: SCSS

App view layout

Best practice: avoid tables, relative positioning, absolute positioning, floats

Use CSS flex-box!


```
#flexbox {
  display: box;
  box-orient: horizontal;
}
#flexbox > p:nth-child(2),
#flexbox > p:nth-child(3) {
  box-flex: 1;
}
```

Caveat: new API just landed, but relatively few changes

More layout

Google™

What about headers/footers? Use `position: fixed;`

How to scroll inside elements? Use `overflow: scroll;`

Inertial scrolling? (Note: iOS 5 only)

```
-webkit-overflow-scrolling: touch;
```

Touch input

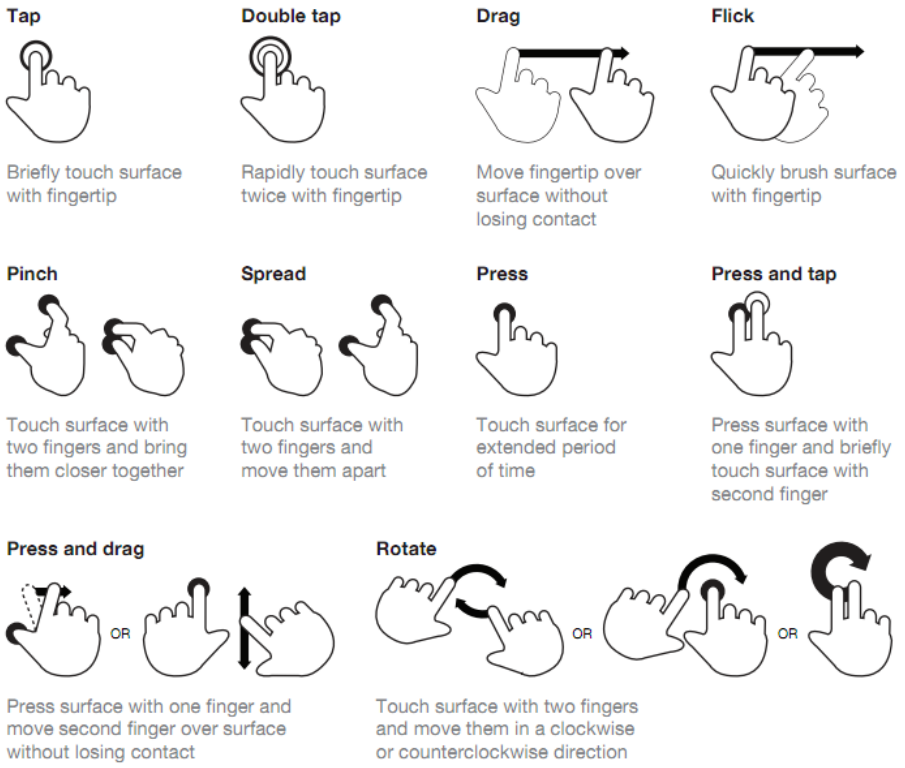
Google

Fingers != mouse. Design for multi-touch!

```
touchstart
touchmove
touchend
```

Click delayed by 300ms on mobile. Use [fast click](#).

More info, [read article](#)



Make it work offline

Google

Storing assets: AppCache, *Filesystem*
Storing data: localStorage, WebSQL, *IndexedDB*

Incomplete support in mobile.

Offline first – pretend that there's no internet connection, implement a sync layer that works only when online.

Offline/online events:

```
navigator.onLine & window.(ononline|onoffline)
```

Unit testing

Google™

MVC provides separation of concerns
Views are hard to test, but

Test your models!

QUnit start/stop mechanisms for testing async code

Tips and tricks

Google™

Enable Safari console for logging on iOS
(*Settings/Safari/Developer*)

Simulate touch events on desktop with MagicTouch.js

Remote debugging hack with jsconsole.com

A similar tech stack, all packed up for you: thorax.js

Demo

Thanks for your time!