

Mobile Dilemma: Native Apps vs. Web-based Apps

Dino **Esposito**

Content

- Let's make sense of opinions and people's demand to (try to) build up a few facts

At the root of the question

The Crux of the Matter

Companies' perspective

- What should we invest in? Mobile site? Mobile app?
 - Marketers and developers are still struggling with the question
 - Since the opening of Apple's AppStore back in 2008
- Companies wonder whether ...
 - They really need both a mobile site *and* a mobile application
 - They better funnel resources into one fully-funded project
- Companies wonder what's best for their users
 - Native experience or generic Web-like experience
 - Native experience means replicating it for various platforms
 - Web-like experience is *less-than-cool* and often *less-than-ideal*

The Crux of the Matter

Architects' perspective

- Just one right answer: **It depends**
 - Concrete answer depends on business goals, objectives, and overall mobile strategies
- Just too many forces being applied
 - Executives impressed by iPhone apps they've seen (and want)
 - PMs impressed by the amount of work
 - Devs willing to do this and/or that (**Pet Technology** anti-pattern)
 - Users demanding for more and more

Dilemma and decision-makers

- Company defines its mobile strategy
- Architects suggest solutions based on their acknowledgment of the company strategy
- Strategies are related to numbers
 - Selling/giving more services to existing users
 - Reaching out and attract more users
- Look at numbers in a **context-sensitive** way

Looking at numbers

Numbers are just numbers

- A really successful (iPhone) app can reach about 10% of the entire mobile population
 - Not to mention device fragmentation and capabilities that could lower this percentage
 - Your app may not run on just any iPhone or Android devices
- A mobile site can cover 100% of the mobile space

Looking at numbers

Numbers are relative to the business

- A large percentage of your users may be using iPhone or Android devices
 - 10% of general population, but likely 60-70% of your audience
- Business model
 - Need to reach out more people or make current users happier?
 - Is your business compelling to people with lower-end devices?
- Examples
 - If you sell ringtones you may want to set up a mobile site
 - If you sell games, you don't want a mobile site (a classic site may be enough)
 - If you do business, you might want to choose *one* platform

Fragmentation

- **Smartphones** fragmentation
 - Differences between versions of an OS (BB, iOS, WP7)
 - Differences between firmware (Android)
 - Devs using undocumented features (or bugs)
 - Devs pushing devices to the limit
- **Mobile devices** fragmentation
 - Properly addressed by WURFL (scientiamobile.com)

Aspects of a mobile site

Mobile sites

What's really good

- Entirely server-side and developed with known products and technologies
 - ASP.NET, PHP
- Hassle-free user experience
 - Users point the browser and go: no setup/config required
 - (Even though just pointing the browser may be problematic)
- Maximize SEO experience
 - The same as desktop Web

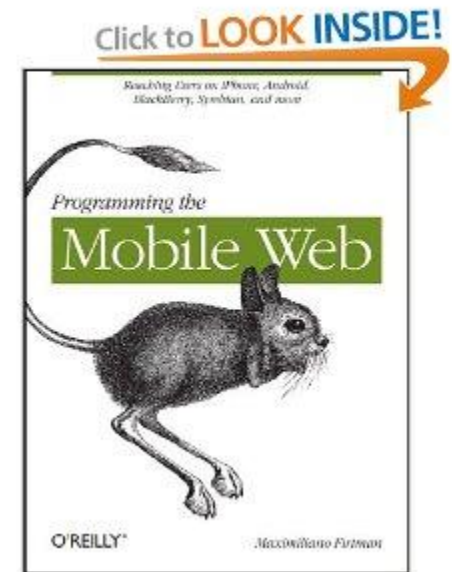
Mobile sites

What's not-so-good

- Radically different browser capabilities
 - Spread reducing for smartphones (WebKit-equipped browsers)
 - Site development **must** address device fragmentation
- Web-based navigation
 - Limited offline work
 - More traffic (without flat rates/wifi may be more expensive)
 - Network latency may slow down the application

Mobile myths

- You just need a 240px HTML page
- Native apps will kill the mobile Web
- People don't like mobile sites
 - (or was it developers?)
- You just need one desktop site
- OK, you just need one desktop and one mobile site



Building a mobile site

Issues and tasks

- Screen size and resolution
 - Resolution ranges from 128x128 to 480x800
- Plethora of browsers
 - Preinstalled browsers and additional browsers (i.e., Opera Mini)
 - Only in advanced OS preinstalled browsers are upgraded
- Before you start, define sitemap and navigation paths
 - Golden rule: **80%** of desktop sites is ignored by mobile users
- Identify use-cases and give them a priority
 - Interviews, statistics, guesses, be-ready-to-change
 - Make it easy to get anywhere
- Be focused on context
 - Where's the user, why's using, what's needed, how to help

Building a mobile site

Pragmatic rules

- No more than 3 clicks for each feature
 - No more than 3 sections per each page
 - If impossible, add a new page
- Skip welcome pages; be direct
- Link the desktop site
- Minimize forms and typing
- Geo-localize users
 - If that information can be treated as input for your app
- Guess next steps
 - Use cookies to track operations and learn from them
- Progressive enhancement
 - Opposed to *graceful degradation* of desktop

(Not just smart) Devices

- The definition of "mobile" is ... mobile
- Arranging a definition:
 - Personal device
 - You can bring it with you
 - You bring it with you most of the time
 - You use it instantly and in nearly any condition (vs. laptop)
 - Provides some type of connectivity

(Not just smart) Devices

- Low-end devices (\$100 devices, e.g., Nokia S40)
 - Basic HTML browser
 - QVGA screen (quarter VGA, 240x320)
 - Camera
 - MP3 player
 - Games & utility apps
- High-end devices / smartphones
 - Accelerometer
 - Excellent camera
 - Bluetooth
 - Good/high quality of Web navigation
 - Touch, apps, GPS, video

Addressing fragmentation

- One page for each device
 - Impractical
- One-site-fits-all
 - Ideal more than practical
 - Comes at the cost of leaving a lot of legacy devices behind and give up on advanced features on smart phones
- Multi-serving
 - Group devices in classes based on capabilities
 - Build a version of the site per classes you intend to support
 - Define a strategy to serve the right site to each device that connects to the site
- Responsive design

- XML-based repository of device capabilities
 - 500+ different capabilities of 7000+ devices
- Open-Source product with very strict (AGPL v3) license
 - AGPL = open-source all of the source code on your server
 - Commercial license from ScientiaMobile (**scientiamobile.com**)
- Adopters
 - Facebook, Google, AdMob
 - Numerous mobile (virtual) network operators
- WURFL in .NET
 - Official API from ScientiaMobile (scientiamobile.com/downloads)
 - API from 51Degrees (with uncertainties around the licensing terms for the WURFL repository)

WURFL for ASP.NET

- Start from user agent string
- Query the database to match UA to a class of devices
- Grab capabilities of that device

- Instantiate once in global.asax
- Grab the request (UA+headers) and get a device
- Query capabilities

- Logically similar (but MUCH more powerful) than ASP.NET browser capabilities

WURFL Capabilities

- Much more than just telling you whether a device is mobile or not
 - Tells whether the community considers it mobile
- Device OS
- Whether it is a tablet
- Image inlining
- Model name
- Video streaming capabilities
 - There are situations like same device/OS run different codecs
- Human brains behind WURFL data

WURFL for the Cloud

- Each site (Java, PHP, ASP.NET) can incorporate up-to-date version of WURFL database
- Cloud option uses a different API to place the query to ScientiaMobile cloud and just (and cache) response
- In beta, enroll @ ScientiaMobile.com

Aspects of a native app

Mobile Continents

- Application created specifically for a mobile OS
- Each mobile OS is a closed environment
 - Different languages, frameworks, hardware
- A native app works only devices with a given OS
- Write the same application over and over again
 - Functionally speaking it is always analogous to web sites

Costs of development

- Skills about different largely heterogeneous platforms
- No code reuse
- Nearly no parallelism in development
- Multiple projects to maintain/outsource
 - Build the app for the primary platform (mostly iOS)
 - Everything else is a cut-down version

Sex-appeal of Native Apps

- Users of smartphones love native apps (no matter what)
- No need to point browsers to URLs
- Nicer user interface and experience
- Makes your phone look cooler
- Lacking a mobile app can alienate users from your brand

- Outsourcing ...

TYPICAL
IPHONE USER



HOW SHE SEES
HERSELF



HOW SHE'S SEEN BY
BLACKBERRY USERS



HOW SHE'S SEEN BY
ANDROID USERS



TYPICAL
ANDROID USER



HOW HE SEES
HIMSELF



HOW HE'S SEEN BY
IPHONE USERS



HOW HE'S SEEN BY
BLACKBERRY USERS



TYPICAL
BLACKBERRY USER



HOW HE SEES
HIMSELF



HOW HE'S SEEN BY
IPHONE USERS



HOW HE'S SEEN BY
ANDROID USERS



Functional comparison

- Web applications can't access native device features such as accelerometer, gyroscope, camera, audio, GPS
- If the browser supports WebKit you can do some of these things via modernizr
- Native controls may look nicer but don't really offer additional features
- Offline storage is possible also in WebKit/HTML5
- Native app downloads in a single package
 - Web app can be written to minimize roundtrips (sprites/inlining)

One (Generic) Strategy that Fits All?

Starting Point

- **Assumption:** *you don't have unlimited resources*
- Start with a mobile site
- Add mobile apps for most popular platforms
 - #1 is iOS
 - #2 is Android
- Everything else is kind of the *long-tail* of mobile
 - BlackBerry
 - Windows Mobile
 - Windows Phone 7.x
 - Symbian
 - Nokia QT

Summary

- *Mobile vs. Native is not a matter of preference*
- *If you have limited resources it may be savvier to funnel them to a mobile site*
- *Go native if there are enough resources*
- *And keep more than an eye on cross-platform solutions such as PhoneGap and Titanium*