# Building Mobile Apps with Web Standards

## Hello, my name is Tim Rogers.

- I'm a 17 year old student from Essex.
- I'm studying for my A Levels, but in my spare time, I like to make cool stuff, mainly for the web and for mobile platforms.
- I develop iPhone and Android application for their respective app stores - but there's a twist!
  - Web standards (and technologies)



Basically go through what it says on the slide. Short explanation of what web standards are for those who do not know - technologies defined by W3C as central to the creation of web content and applications. This includes things that most of us will use in our daily work, like HTML5, CSS and Javascript, but also lesser-used technologies like SVG and the Canvas API.

# The challenges of mobile development

## What are our challenges?

- What platforms should I develop for?
- Do I have the right skills?
- How do I created a good experience for mobile users that is consistent with their expectations?
- What features are available, and which should I choose to implement?

## And what are the possible solutions?

- Web applications
- Native applications



Platforms: Today as developers, one of the most common tasks that we want to complete is creating mobile applications, generally for the iOS and Android platforms. But what about all the other platforms, like Windows Mobile, Palm WebOS and Blackberry?

Skills: One of the problems which has occured from the divergence of platforms is differences in the languages require to program for each device. For example, iOS traditionally demands that we develop in Objective-C, whereas Android has a strong preference for Java applications.

Experience: Developing for mobile is somewhat different to developing desktop applications or traditional web applications. We have to adapt our user interfaces to smaller screen sizes, on-screen keyboards and touch screen interactions, rather than the traditional keyboard, mouse and monitor that we would see on a PC or Mac.

Features: Each platform that you could develop an application for has a different set of features, or generally at the very least a different way of accessing them. The mobile ecosystem is full of “buzzword” technologies like geolocation, push notifications and multitasking. But which of these are available and which will actually be useful for your application.

We could build the applications that we want to build in two ways - using the native frameworks and languages provided by the devices, or by instead creating web applications. When you think about it, many of these challenges could be avoided by using web applications. But traditionally, there are some differences between the capabilities of applications running natively and those running in the browser.

# Native vs Web App

## Web app

Using existing web development skills
Compatible with a variety of platforms
No expensive fees

## Native

Access to device APIs
Generally better performance
Opportunity to market your application in a store
Easy to charge for your app

**But what about if you could merge the best bits together...**

jQTouch + PhoneGap



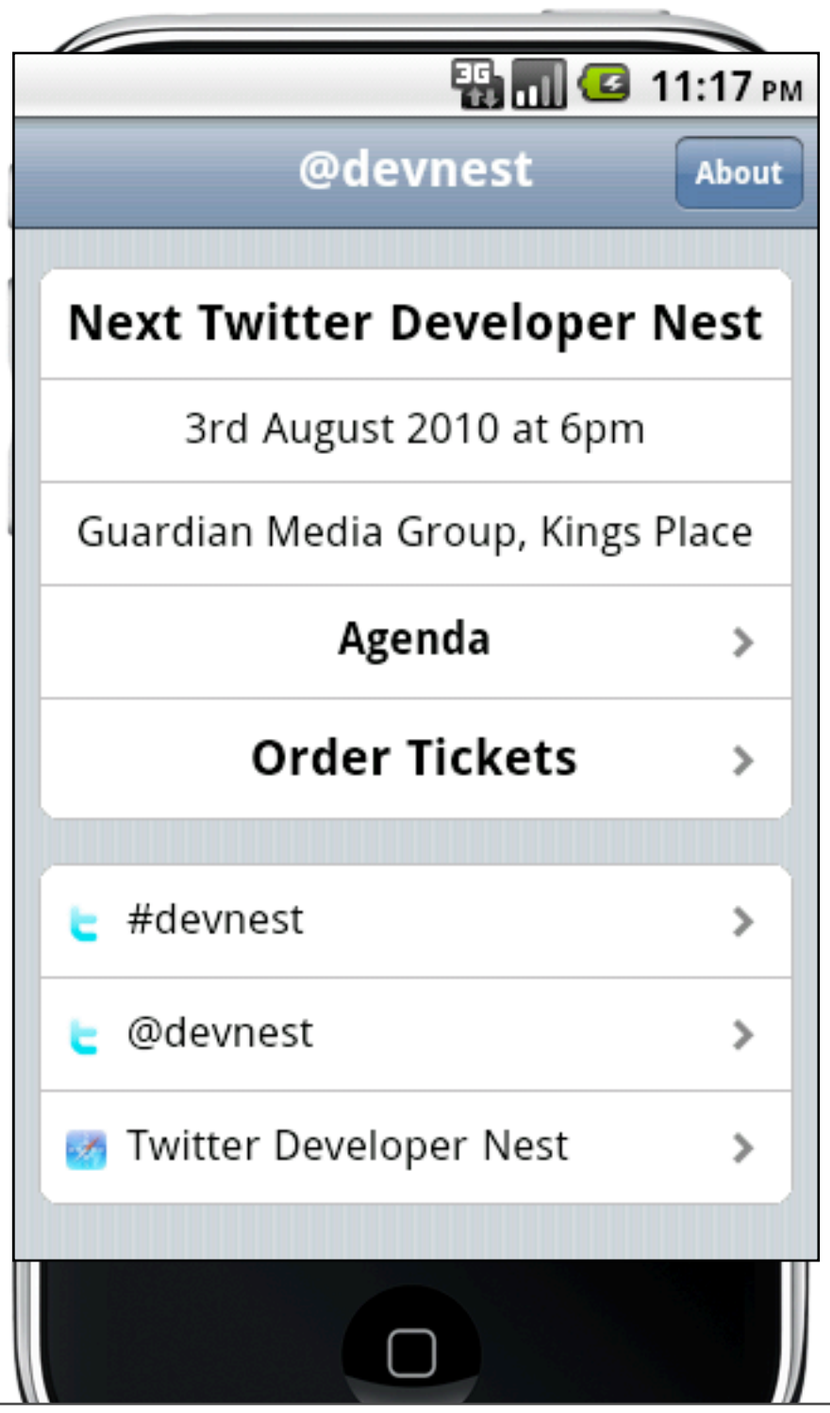
Explain the traditional differences between native and web development. Mention WebKit on iPhone and Android when speaking about web apps. Go through the advantages in a little more detail.

Explain that these are the advantages of each platform, but we can all easily think of the relative disadvantages of each. Eg. web apps tend to be quite slow, and native apps often require you to retool your skillset. Also mention psychological aspect - native apps **feel** different.

But what about if you could merge the best bits of each option together, leaving the disadvantages behind. Explain that this is exactly what you can do with PhoneGap and jQTouch.

# jQTouch

## What is jQTouch?




- A jQuery plugin for creating mobile interfaces, created by David Kaneda, now taken over by Sencha Inc.
  - Free and open source
  - Available on GitHub
- Works in the WebKit engine on both Android and iOS.
  - A great platform consistent user experience
- Access to jQuery, as well as the ability to use all your favourite Javascript libraries like Underscore.js to do anything you like



jQTouch is a free and open source plugin for jQuery which allows you to get the classic iPhone navigation interface in your application. With jQuery and jQTouch, you can build your application using HTML, CSS and Javascript. As you can probably tell by my lovely little screenshot, jQTouch creates a great UI for iPhone, but it is also perfect and looks really great on Android devices. The image depicts the Twitter Developer Nest app on iPhone, which I will show you later, but you can also run the same application on an Android phone and see the same interface, with a few different fonts. jQTouch is designed to be fully compatible with the WebKit browser used on iOS and on Android. And there we go – the same great looking interface is also perfect for Android.

Anyway, back to the advantages of using web standards to build your app. With these technologies, you won't need to worry too much about your skillset – if you're proficient in building for the web, you can build for mobile with jQTouch once you've got used to how it works, which I will show you in a second. You don't have to worry about the platform issue, because jQTouch supports Apple and Android devices, and it can also work with the upcoming Blackberry OS 6 and Palm's WebOS. Using the standard interface features, you also go a long way towards ensuring a good experience for the user, and thanks to the trio of Javascript, HTML5 and CSS3, you can access a whole range of device capabilities including Geolocation. There are some functions which you cannot access with these web technologies, but this is a problem I will go into, and solve, later.

You will also be glad to know that you can use all your favourite Javascript libraries. This makes it really easy to implement great levels of interactivity and connectivity into your application. With jQuery, you can use AJAX, and you don't have to worry about cross-domain requests: your application is run from the file protocol, so you can access any resources on the web that you want! I'm sure as you're attending this event, you'll all be very glad to hear that this certainly means you can access the Twitter API! You can also make use of all the other great jQuery features, including its effects and DOM manipulation. And beyond jQuery, you're also free to include other libraries like Mootools or Underscore.

# Getting started with jQTouch

## It is very easy to get started with jQTouch.

- Include the jQuery and jQTouch Javascript files
- Include the jQTouch and Apple theme CSS stylesheets
- Add the initialisation Javascript...

```
var jQT = new $.jQTouch();
```

code/slide5-1.js

- And then write the code to create your 'screens'...

```
<div id="jqt">
   <div id="home" class="current">
      <!-- Toolbar and menus !-->
   </div>
   <div id="settings">
      <!-- Another page !-->
   </div>
</div>
```

code/slide5-2.html



The jQTouch download comes with everything you need to get going with making your application. I don't recommend you download it from jQTouch.com - it is better to clone it from the Git repository, which I will link you to at the end of the presentation. A suitable version of jQuery is included, along with the jQTouch.js plugin file, a jQTouch CSS file and two themes. The makers of jQTouch provide an Apple theme, as seen on the previous slide, and a darker and more industrial theme to use as an alternative. In the header of your script, you just need to include all these files, including the CSS file for the theme you want to use. Once you've included these, the classes you use will be generally the same, regardless of the theme you choose. To get jQTouch to work and do all of its important magic, allowing you to create screens in your app, you just need to add the line of Javascript "var new = new $.jQTouch();" into some script tags. There are a variety of initialization options that you can also include within this line, but they are beyond the scope of this presentation and can be read about on the jQTouch website. In your application, each 'page', like the one you saw on the previous slide in the DevNest app, is held inside a  tag with a unique ID attribute. Within this tag, you will generally expect to have a toolbar, and a collection of unordered lists that make up your content. All these page DIVs are then held inside a container div with the ID of jqt.

Now I'm actually going to show you a very quick sample of a jQTouch application, before moving swiftly onto PhoneGap and finally a full sample application, designed for iPhone.

# A sample jQTouch application

```
<html>
    <head>
        <title>A jQTouch sample application</title>
        <script type="text/javascript" src="jqtouch/jquery-1.4.2.js"></script>
        <script type="text/javascript" src="jqtouch/jqtouch.js"></script>
        <link rel="stylesheet" type="text/css" href="jqtouch/jqtouch.css" />
        <link rel="stylesheet" type-"text/css" href="themes/apple/theme.css">
        <script type="text/javascript">
            var jQT = new $.jQTouch();
        </script>
    </head>
    <body>
        <div id="jqt">
            <div id="home" class="current">
                <div class="toolbar">
                    <h1>jQTouch</h1>
                </div>
                <ul class="rounded">
                    <li class="arrow"><a href="#settings" class="flip">Settings</a></li>
                    <li>A menu item</li>
                    <li>And another again.</li>
                </ul>
            </div>
            <div id="settings">
                <div class="toolbar">
                    <h1>Settings</h1>
                    <a class="button back flip" href="#home">Back</a>
                </div>
                <ul class="edgetoedge">
                    <li>More menu items! Hooray!</li>
                </ul>
        </div>
    </body>
</html>
```

code/slide6-1.html



Talk through what each part of the code is for, to explain how jQTouch works. Mention linking between pages, and included classes like ‘edgetoedge’, ‘rounded’ and ‘arrow’. Show it running in the iPhone simulator for a few seconds

If you want to have a look at this code in full at home, and try it and update it yourself, it is included in the full package of this presentation that I will give you at the end. At the bottom right of each code snippet, you can see the path to the file in the release. An executable, based upon WebKit, is included for Windows, Mac and Linux as well.

# PhoneGap

## What is PhoneGap?

- PhoneGap is a framework by Nitobi Software Inc. which allows you to package apps build in HTML for a variety of platforms
- Provides sample projects and scripts for creating your application from your front-end code
- Just include the phonegap.js library to access device functions...

```
<script type="text/javascript" src="phonegap.js"></script>
```

code/slide7-1.html

- For example, native device alerts and vibration...

```
navigator.notification.alert("Message here", "Title here", "Button Label Here");
navigator.notification.vibrate();
```

code/slide7-2.js



PhoneGap is the other side of this presentation. It takes an application created in HTML, Javascript and CSS and allows you to package it for iOS (including the iPad) and Android, and also other less successful application platforms like Palm WebOS, Blackberry and Windows Mobile, although generally these platforms are more difficult to achieve a good result on.

PhoneGap provides all the tools you need to make your applications on these platforms. Mainly, we are concerned in this talk with Android and iOS. For the iPhone and iPad, a project template for Apple's Xcode tool is provided, into which you simply import your web application. After that, you simply have to have an inevitable fight with the often painful Xcode, until you eventually manage to get it packaged and running. For Android, a shell script is provide called droidgap, which allows you to set a number of parameters like version, package identifier and application name, and then have a project folder build which you can then build using the Android SDK tools.

As I mentioned back when we were talking about jQTouch, there are some features that you might find useful. which are not available through the normal Javascript APIs. The beauty of the PhoneGap framework is that it abstracts the APIs for a whole range of functions like filesystem control, native alerts and vibrating the device into a simple Javascript API which is the same across all the platforms, hence allowing you to use the same code everywhere. To make these work, you simply need to include the phonegap.js file into your project, which will be copied into the folder by PhoneGap in most cases. Then you can really get to work!

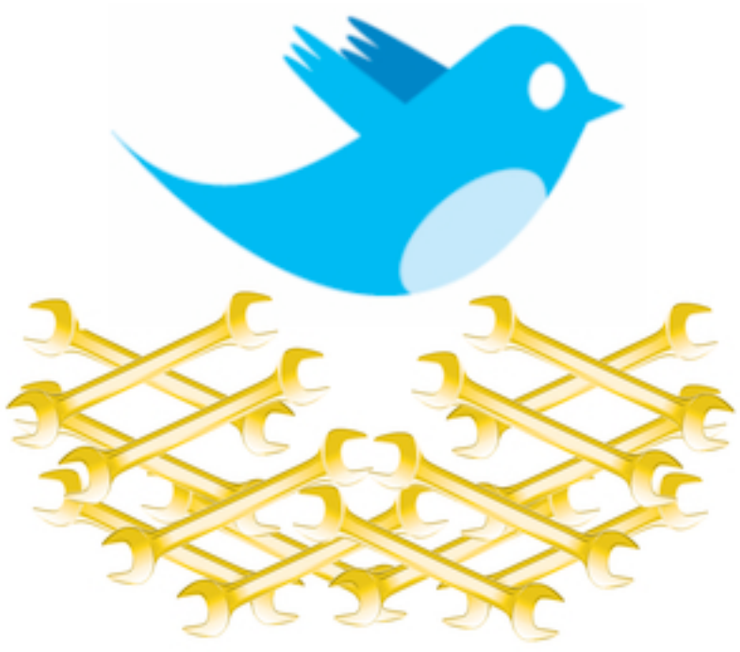
Two of the most obvious features that you might like to access, native alerts (as opposed to rubbishy Javascript ones) and vibration are very easy to use. As you can see on screen, phonegap.js makes it incredibly easy, requiring only a Javascript function with a few parameters to do these tasks.

# DevNest.app

## A mobile application for Twitter Developer Nest

- Created for iPhone and Android
- I will now show you a quick demo of the application running on the iPhone simulator...

twitter developer nest





At last month's DevNest, I decided I was going to use my knowledge of jQTouch and PhoneGap in order to create an app for this event, Twitter Developer Nest. This later evolved into my plan to do a presentation! This app is very simple, relying on the jQTouch features I showed you early and some jQuery magic like load() and getJson() to load remote data, including from the Twitter API, to keep the app up to date. I then used PhoneGap to package the application up for Android and iOS. I am now going to quickly show you the app as it runs on the iPhone simulator.

# Thank you!

## Thank you for watching the presentation.

- The DevNest app is available now on your local app store.

Available on the App Store

iOS: http://bit.ly/devnest-iphone

Android: Search DevNest on the Market

- The DevNest app source code is available on GitHub.

iOS: http://bit.ly/devnest-iphone-source

Android: http://bit.ly/devnest-android-source

- You can also get this presentation and all the related resources from GitHub in seconds.

```
git clone git://github.com/timrogers/building-mobile-apps-with-web-standards.git
```

or visit http://bit.ly/devnest-mobileapps



Thank you for listening to this presentation this evening. Just to inform you, you can already grab the DevNest application that I've been showing you from the Apple App Store or from the Android Market. The source code of that application is also available on GitHub.

If you'd like to view this presentation again, you can clone it, and all the resources including the code samples from my git repository, or you download the package from the GitHub website. If you don't want to remember all these long links, I'm sure I can get them put up on the event's website later.

Does anyone have any questions?