Metropolia

Jukka Paasonen

HTML5 as Common User Interface Layer in Mobile Device Platforms

Helsinki Metropolia University of Applied Sciences
Master's Degree in Information Technology
Multimedia Communications
Master's Thesis
3 December 2012

PREFACE

The research started back in 2010 with near to native solutions and with a purpose to reduce errors while implementing final user interface from an accepted prototype. While the research moved on, so did the industry and the buzz for HTML5 was getting louder. With this so did the research evolve to include HTML5 to one of the prototyping tools and solutions that could be used in the production code.

The market of mobile device platforms has been fragmenting increasingly, which makes development, let alone testing more tedious. While most of them have software simulator, true testing should be done in a real device. Acquiring all the devices can be expensive, which made the research somewhat slow.

The research should give a fairly good view on which platforms should be targeted today, which tools to use for development, how to test for feature availability and speed performance. With the results the developers can evaluate if HTML5 is the right solution for the given application.

This research would have not been possible without many individuals whom I wish to give my sincere gratitude. These people include, but not limited to Mari Paasonen, Harri Kiljander, Kari Sysimiilu, Petri Kosonen, Yoshinao Nanbu, Ville Jääskeläinen and Karin Paasonen.

I wish I was younger and technologies like HTML5 would mature faster.

Helsinki, October 25, 2012

Jukka Paasonen

Metropolia

Abstract

Author Title Number of Pages Date	Jukka Paasonen HTML5 as Common User Interface Layer in Mobile Device Platforms 74 pages 3 December 2012
Degree	Master of Engineering
Degree Programme	Information Technology
Specialisation option	Multimedia Communications
Instructor(s)	Ville Jääskeläinen, Principal Lecturer Harri Kiljander, Dr. Tech, Interactive Digital Media

HTML5 is spreading from web to mobile applications. This research was started to find out if HTML5 and the related technologies were ready to be used for cross platform mobile device application development in the current market major devices.

The research selected four current mobile device platforms based on the current market shares and on developer surveys for finding the most popular platforms. For each platform their manufacturer's recommended development solution was evaluated against the ability for creating HTML5 based applications. The three most popular third party cross platform solutions based on HTML5 and related technologies were evaluated for their development process and compared to the manufacturers' offering.

Tools for testing against HTML5 and compliances of other standards as well as performance, were evaluated for their ability to test features, benchmark application speed and easy of use.

HTML5 seems the best solution for cross platform development, when there are more than one target platform. The compliance of tools and standards is maturing rapidly and very decent compatibility is offered with the modern development platforms.

Key words	HTML5, CSS3, JavaScript, Mobile, Smartphone, User Interface

Table of Contents

List of figures

List of tables

Abbreviations and Acronyms

API	Application Programming Interface
ASF	Apache Software Foundation
CSS	Cascaded Style Sheets
DN	Developer Network
FPS	Frames Per Second
DOM	Document Object Model
GPL	GNU General Public License
GUI	Graphical User Interface
HTML	Hypertext Markup Language
IE10	Microsoft Internet Explorer 10 web browser
IETF	Internet Engineering Task Force
iOS	Apple Operating System used in iPhone and iPad devices
J2ME	Java 2 Mobile Edition
JIT	Just In Time compilation method used by scripting language engines
JS	JavaScript
LGPL	GNU Lesser General Public License
MathML	Markup Language for Mathematical expressions in HTML
MIME	Multipurpose Internet Mail Extensions
MVC	Model, View and Controller
NDK	Native Development Kit
NGEN	Native Image Creator used by Microsoft in their App Store
OS	Operating System
PNG	Portable Network Graphics
QML	Qt Meta object Language
Qt	C++ cross platform GUI library
S^3	Nokia Symbian^3 Operating System
SDK	Software Development Kit
SGML	Standard Generalized Markup Language
SMIL	Synchronized Multimedia Interaction Language
SVG	Scalable Vector Graphics
UI	User Interface
UX	User Experience
W3C	World Wide Web Consortium

WebKit	Open source web browser engine
WHATWG	Web Hypertext Application Technology Working Group
WRT	Web Runtime, a module in WebKit which enables the use of widgets
WWDC	Apple Worldwide Developer Conference
XAML	Extensible Application Markup Language
XML	Extensible Markup Language

1 Introduction

HTML5 is the recent version of the hypertext markup language standard that makes the Web. Recently it has become a popular topic in the media and in the development of mobile applications. The name HTML5 is often used to refer to the whole collection of related technologies which are not singly used for markup, such as LocalStorage which is a JavaScript feature, or CSS3 which is used for styling and layout.

Software user interface prototyping is often done in technologies such as Adobe Flash and Qt QML, but with the recent popularity around HTML5 has made it a replacement for the previously used technologies. While HTML5 related prototyping is getting more popular, the final implementation is done natively and the prototype code is thrown away. The research wanted to clarify the current situations for building HTML5 hybrid applications with the five most popular platforms.

The upgrade cycle of the mobile device platforms is getting faster and the support for HTML5 related development models is becoming widely available. The native solution from each manufacturer can be expected to have access to the most features that are available in the given platform. With HTML5 there are limitations since the user interface needs to be rendered from a WebView class of the given platform that may or may not allow deeper access on the device.

Research was defined by a broad question:

Could HTML5 and related technologies lower the costs of implementing User Interface design in different mobile platforms today?

This research evaluated the possibility of using a single user interface technology across the most popular mobile device platforms. Up until now, each platform has provided their own way for building applications and their user interfaces. Nokia adopted Qt framework with QML for Meego and more recently Silverlight with XAML for Windows Phone platform, Google uses Java with proprietary XML files for Android and RIM

offers native C++ for their Blackberry devices. Finally, the one platform that made mobile applications popular, Apple uses proprietary Objective-C for iPhone and iPad.
Windows Phone 7 for example has a security model that prevents any additional access to the device hence making the HTML5 application to be as any web site in the native browser. Windows Phone 8 upgraded the development environment remarkably and allows similar HTML5 development path as the other platforms via native wrappers.

The costs of the development are usually made up of engineering time, tools and number of people. This research touches mainly the tool area since technologies are the tools of the engineer. [1:4] One commonly used method in user interface development is the iterative development cycle as shown in Figure 1. The user experience (UX) is designed, prototyped in HTML5, rewritten in native code and tested first against code related tests, then user interface tests by the UX team. There could possibly be user interfaces tests comparing the prototype and the native.

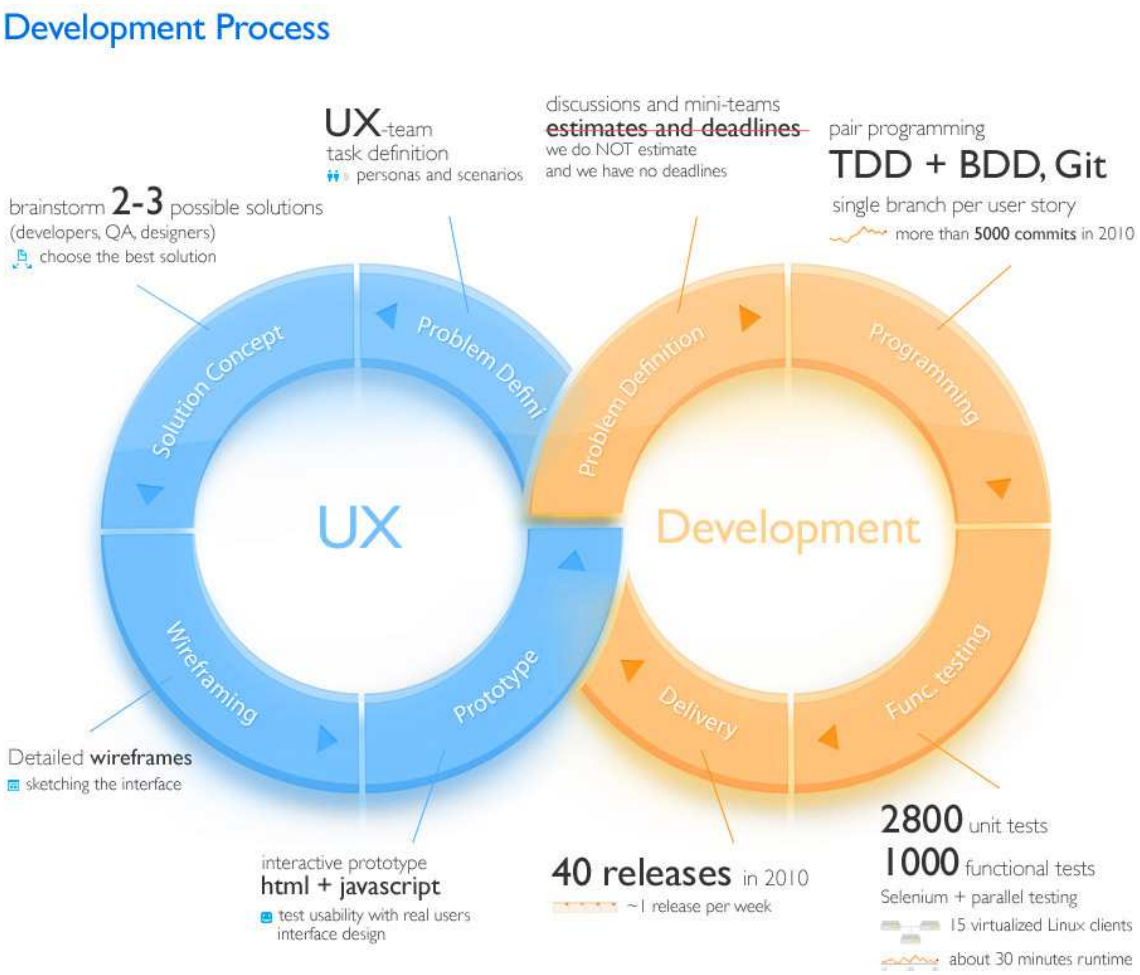


Figure 1. Iterative software development progress [2]

Instead of implementing the user interface with a native code, the existing HTML markup from the prototype with the JavaScript for logic could be used in order to reduce development time. The conversion from prototype to production will never be direct but by using the same API and same rendering capabilities for both of them, the time to product can be reduced.

The environment where prototypes are initially run is a desktop web browser. Later in the development iteration cycles, the prototype is usually used in the targeted mobile device, but still in the web browser surrounding. If the platform makes it possible, finally the prototype is wrapped inside a native application for accessing real device data. This research also covered testing each platform against the current HTML5 specifications in order to evaluate how much of the same code made for prototypes or similar web pages could be used in the mobile device.

The mobile device platform selection was based on the current market shares. The research covered the following four platforms:

- Apple iOS
- Google Android
- Microsoft Windows Phone
- RIM BlackBerry

In all of the selected platforms the HTML5 based application requires a native wrapper application that connects to the device data. The amount of code hence work, varies across platforms but most of the differences come from the JavaScript APIs.

Tablets were not taken within the scope of the research, although all of the selected mobile platform manufacturers have already released or are planning to release tablet devices using the same operating system as their smartphones. CSS3, however, has a module suitable for handling different screen sizes via media queries.

This research consists of four parts. Part one focuses on the selection of the most popular platforms based on their market share and developer interest. The second part describes how application development can be done in these platforms. The third part

evaluates different cross platform solutions and how they can be used to target all of the selected platforms. The fourth part evaluates available tools for measuring HTML5 compatibility, JavaScript performance and other similar tools for the topic.

2 HTML5 and Related Technologies

HTML5 is a standard for creating hypertext markup to be used in modern web sites and more recently in mobile applications. The standard is still incomplete, in a draft status, thus called a living standard. The HTML5 standard is developed by World Wide Web Consortium (W3C), Web Hypertext Application Technology Working Group (WHATWG) and Internet Engineering Task Force (IETF). HTML5 has been developed simplicity in mind. [3:3]

The name HTML5 is often used to refer a set of technologies that are not within the actual HTML5 standard for hypertext markup language. Figure 2 visualises the state of the standard statutes in December 2011 with areas specifying the different standard development groups related to the surrounded specifications. The innermost circle in contains the specifications that were spilt from the original WHATWG HTML5 specification.

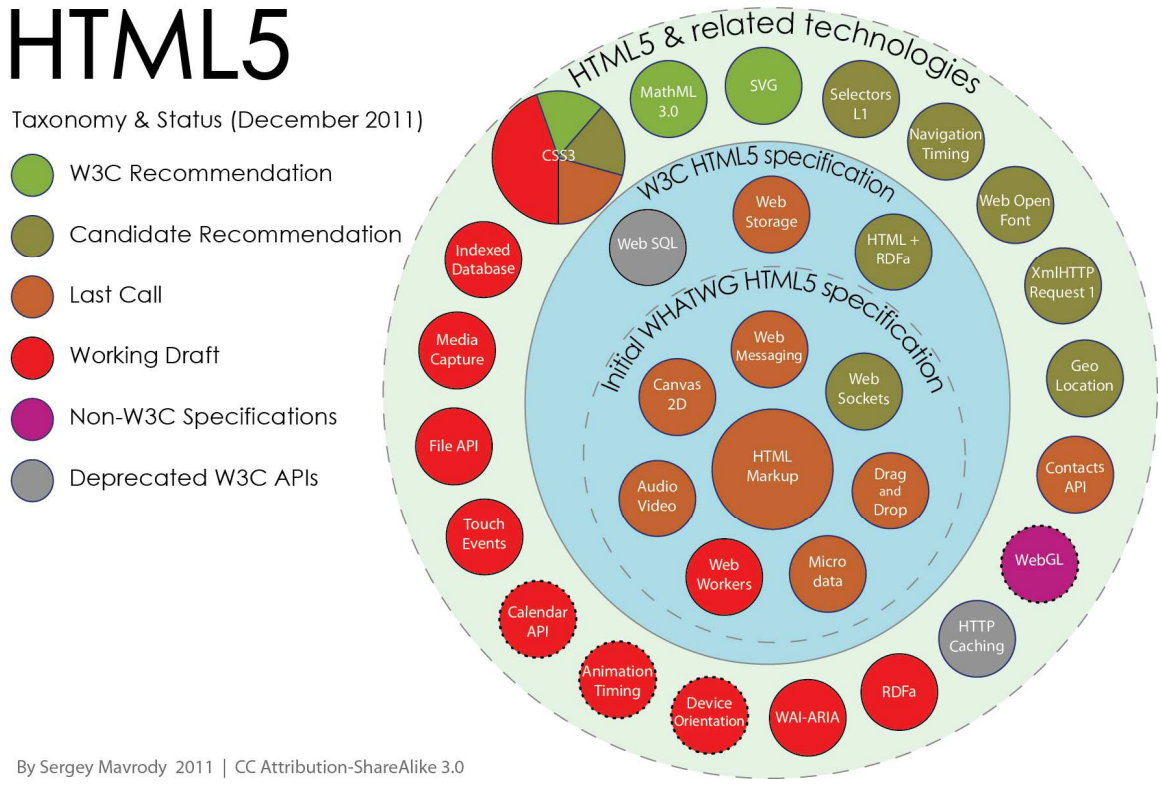


Figure 2. HTML5 taxonomy and status in December 2011 [4]

World Wide Web Consortium (W3C) is made of 385 member organisations and 72 full time staff working all over the world. Their mission is to lead the Web to its full potential by developing standards, protocols and guidelines. [5]

WHATWG is a community of people interested in evolving the Web by focusing on the development of HTML and Web Application APIs. The community was created by individuals from companies like Apple, Mozilla and Opera in 2004 after they had been meeting in a W3C workshop. [6]

IETF has the goal of making the Internet to work better by producing technical documents that designers and developers can use. [7]

HTML5 has become a buzz word that is used to refer to a group of standards and technologies related to web development. Most of them have been created to address certain issues that have risen in the web development. For example Application Cache was built on the idea of Google Gears. Not all of the mentioned HTML5 related technologies are included in the W3C HTML5 specification, but they are part of the WHATWG HTML specification. Other HTML5 related technologies, which are not part of either the W3C HTML5 or the WHATWG HTML specification, are published by W3C separately.

The desktop experience of a successful web site does not translate directly to a smartphone application. Different approach has to be taken due to the different interaction model. Mobile devices are used by fingers, while the desktop is until now been interacted with a mouse and a keyboard. For example there is no hover interaction in the touch interface.

HTML5 has been seen as a user interface layer that can be used similarly targeting the desktop web experience as well as in the mobile device applications. While earlier HTML specifications were based upon what browser vendors wanted to do, HTML5 was specified by researching how Internet was build [3:5]. Developers familiar with web page development are most likely to use HTML5 once adapting to mobile device application development, as the KendoUI survey results from September 2012 show in Figure 3. [8:5]

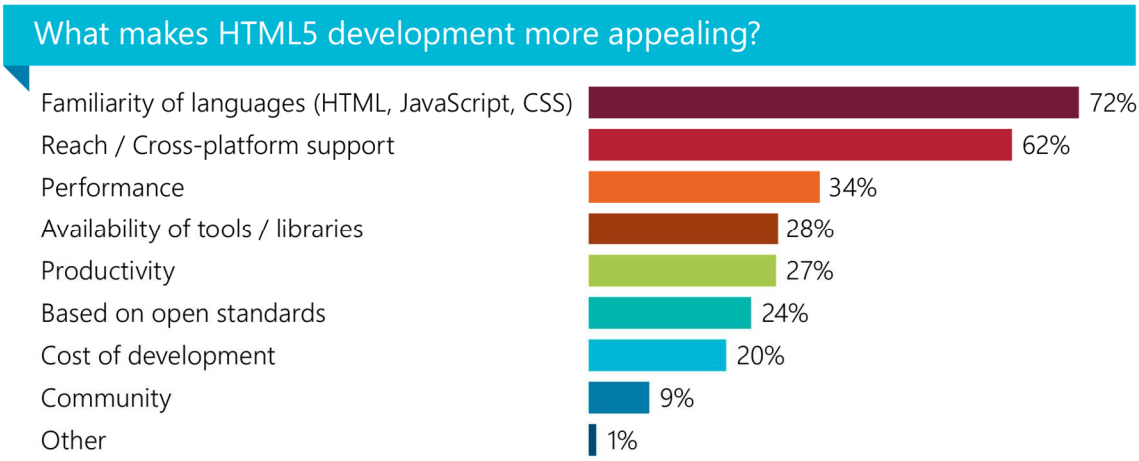


Figure 3. KendoUI developer survey results for HTML5 development interest [8:5]

According to Sergey Mavrody HTML5 has several advantages why it should be used in the modern web development. It is backward compatible, has simpler syntax, contains improved semantics, more productive coding and smaller document size due to new elements and attributes. Additionally it comes with plugin-free video and audio and timed media playback, smart Web Forms 2.0 functionality, ability to use in-line Scalable Vector Graphics (SVG) and MathML [9] in with "text/html" Multipurpose Internet Mail Extensions (MIME) [10] type. Finally it enables easier development and enhanced user experience. [4:10]

When is the right time to use HTML5 instead of native approach? Pure HTML5 application is simply a web page running locally without any additional access to native features than online pages. [11:3] Figure 4 displays the orientation of native against pure HTML5 application. The hybrid application stands in the middle ground, reaching from near to full native to plain WebView containing heavy HTML5 implementation. [12]

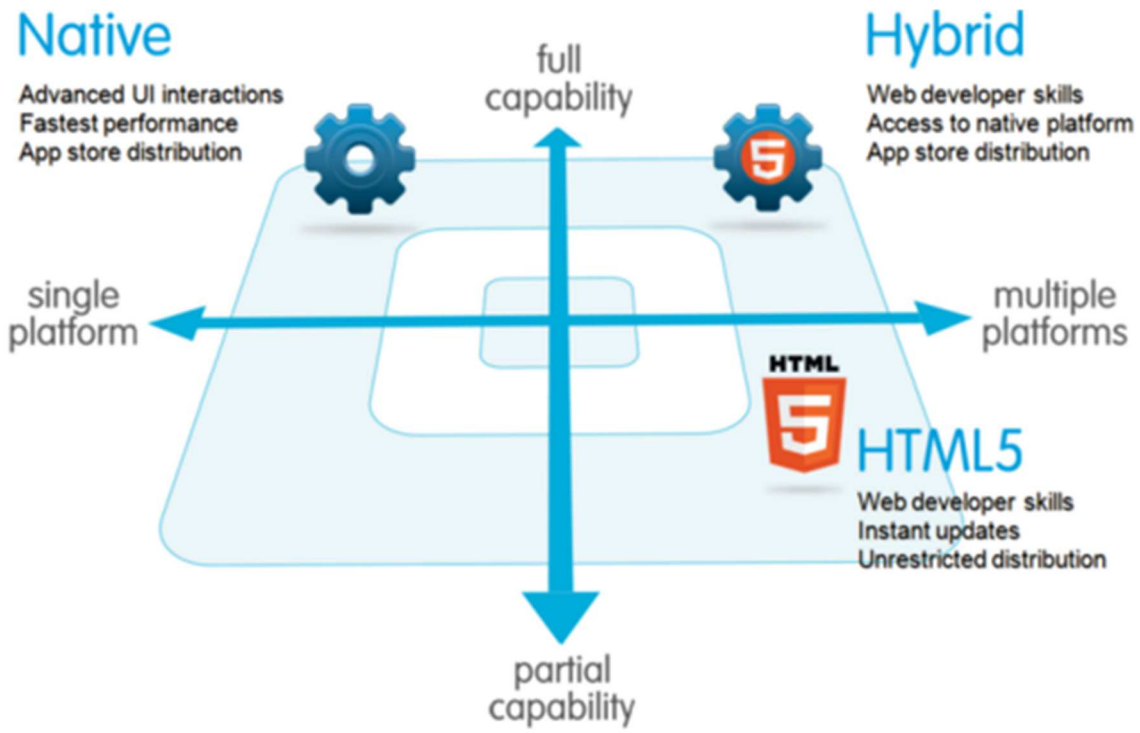


Figure 4. Native, HTML5, or Hybrid: Understanding Your Mobile Application Development Options [12]

HTML5 has been marketed as similar to Java, both promising "write once, run everywhere". HTML5 is also expected to have standardised API for accessing mobile device data such as GPS location. Since each implementation is different, even if using the same rendering engine like WebKit, the feature availability and display layout does not look identical. Even the same platform is not always compatible between versions, thus the fragmentation keeps increasing. [13]

Mobile platform manufactures are working hard to differentiate and thus are not even trying to standardise their features.

According to Michael King [13]

> "Enterprises increasingly want a single app platform"

The main reasons for enterprises not using HTML5 are missing standardised APIs for certain mobile device features like camera, poor performance and lack of native UI/UX. The reason for considering the use of HTML5 are lower cost platform deployment costs, different screen sizes share the same code for example tablet versus smartphone. [13] Figure 5 visualises the differences and limitations between native

and HTML5 applications. While HTML5 application is limited to the web browser and custom theme, native application can access all of the operating system (OS) capabilities with native user interface theming and user experience.

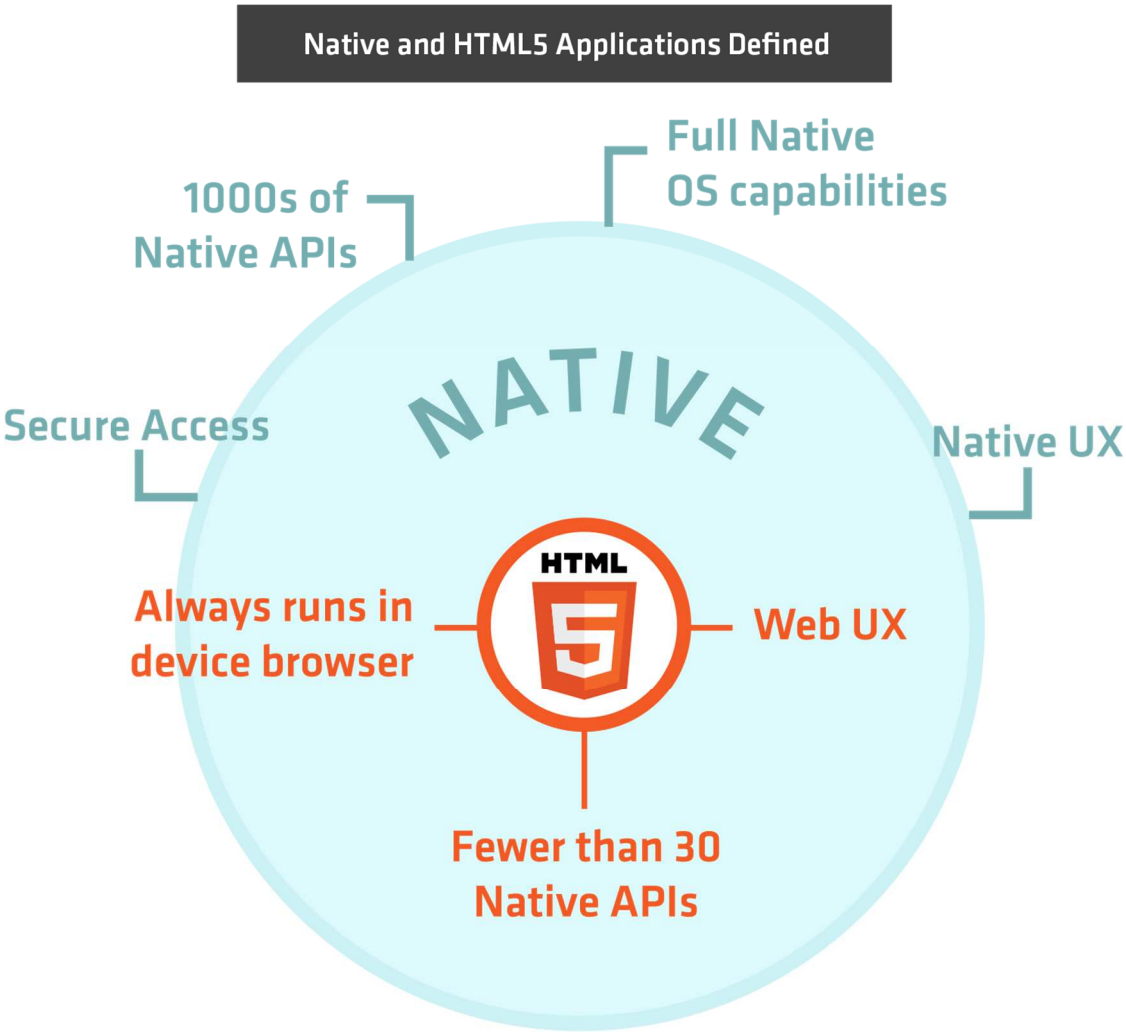


Figure 5. Native functionality as an extension to plain web app [11:4]

Hybrid applications are similar to native applications but use the same rendering engine as the web browser available in the mobile device operating system. Hybrid application can leverage the full native OS capabilities and native APIs to the web based application.

2.1 Brief history of Hypertext Markup Language (HTML)

The first specification for HTML was published by Berners-Lee in 1991 and called HTML Tags. It language was based on another markup language known as Standard Generalized Markup Language (SGML), which is not publically available. The development of following version was relatively rapid until the release of HTML 4.0 by W3C in 1997, which was revised as 4.01 in 1998. Since then there has not been a new recommended version of HTML. [4]

In 1998 W3C decided to stop developing HTML and focus on XHTML as a replacement. XML conforming version of HTML, XHTML 1.0 was published in 2000 and later update to 1.1 in 2002. Version 2.0 was developed and drafts were released from 2002 until 2006 when W3C decided drop the development and focus on HTML5, which was in its early phase called Web Forms 2.0. [14] The first draft of HTML5 was released in 2008. XML flavoured version of HTML5, called XHTML5 has been under development since 2009. [15]

2.2 Current Status and Future of HTML5

HTML5 is a buzz word used with many meanings in the media. This has created many expectations on the technologies, some with lack of actual existence. Therefore the technologies around HTML5 are filled with inflated exceptions, with the peak just being passed in 2011, as shown in Figure 6. [16] One such promise is to push the capabilities of web and mobile applications as far as Flash applications have done in the web. With the current rate of browser updates and implementation improvements however, it is not far. [17]

"The rising star of HTML5. HTML5 has the potential to become a common bridge system across smartphone platform islands and the sea of feature phones. HTML5 is the only common app technology supported by Android, iOS, new versions of BlackBerry OS and Windows Phone platforms. With 225 million Android devices and 146 million iOS devices sold to date, HTML5 is supported by over 371 million mobile devices today, albeit with mixed levels of compatibility." [18]

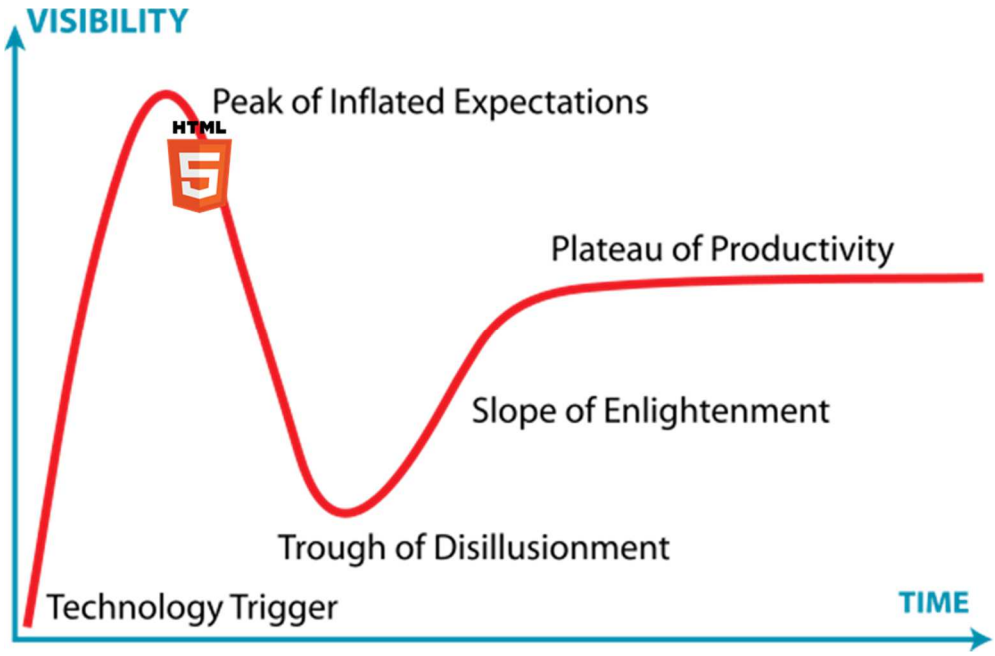


Figure 6. HTML5 is just past the peak of expectations in the eyes of the developers [16]

Recently the schedule for completing the HTML5 standard has been updated. W3C plans to finish it by the end of 2014. [19] On the other hand WHATWG announced in 2011 that they dropped the version number of HTML and call it a living standard, which is continuously evolving. [20]

Big companies such as Adobe and Facebook are placing focus on HTML5 development and related web for mobile. [21] Adobe acquired Nitobi, a company that created PhoneGap which is the most popular cross platform solution for creating HTML5 based mobile applications, thus increasing their development resources towards HTML5 tools and solutions. [22]

The importance of HTML5 support has also been recognised by Adobe. While Adobe has been very successful with its Flash plugin, which is by far the most widely spread browser plugin in desktop environment, they have recently decided to stop developing Flash plugin for mobile devices. Instead they will be focusing on HTML5 in mobile devices and Flash in desktop devices. [23]

With the updated schedule and increased resources for developing tools and applications, HTML5 can be expected to become more important technology in the future.

2.3 Semantic Approach

Web page technologies have been designed to distinct between the roles of the content, how it is presented and what kind of interaction or behaviour it has. Figure 7 describes how HTML is the base element describing the data and content. CSS is used to define how the content is presented, perhaps differently in different context such as desktop versus mobile devices. Finally JavaScript is used to program the behaviour, data binding and other more demanding methods.

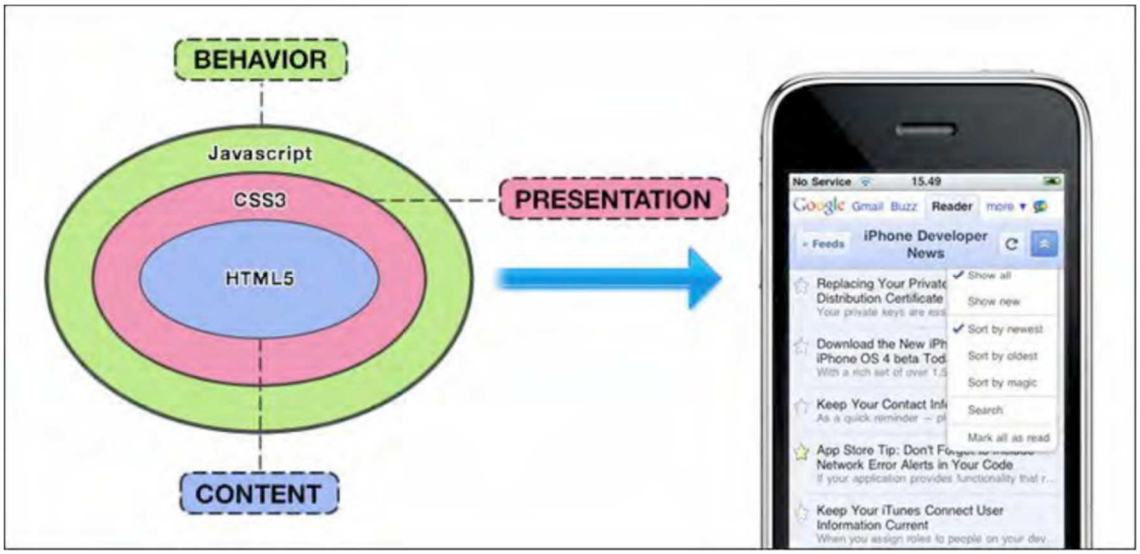


Figure 7. Progressive Enhancement Paradigm on Web Site and WebApp [24:112]

The structure of Model-View-Controller was first used in Smalltalk programming language. [25:72] From there it has been adopted to many programming languages and systems that implemented all the three components. HTML5 updates the existing HTML4 markup with semantically clearer elements, such as "header", "nav" and "footer" as shown in Figure 8 which compares the same layout between these two versions.

Figure 8. HTML4 and HTML5 comparison in two simple iPhone page structures [24:247]

While in HTML4 and earlier, the different sections of the page were defined as "div" elements with distinctive id properties. In HTML5 the new elements are used for the same purpose in order to give better semantics for the page.

2.4 Cascaded Style Sheets 3 (CSS3)

Cascading Style Sheets are used for defining how HTML data is shown, transformed and animated. The standard for CSS3 is defined in several modules while earlier versions were in a single standard definition. The modularisation made it easier for W3C to work on each aspect separately and gave the browser makers better roadmap what to implement. The current CSS3 modules and their statuses are listed in Table 1.

However, the support is very fragmented since some of the modules that are complete as a standard are not yet fully implemented in the major browsers. On the other hand, some modules that are still in the exploring phase, have been implemented by some browser vendors. For example the Grid Layout has only been implemented in Internet Explorer 10 (IE10). [26]

Most often when a browser vendor has implemented a feature or a module that is not a stable recommendation, they add their browser specific prefix in the property names. For example in the case of IE10 and Grid Layout, the properties are starting with the prefix "–ms-grid".

Table 1. CSS3 modules and their statuses

Module	Current Status	Last Update
Media Queries	Recommendation	19 June 2012
Selectors Level 3	Recommendation	29 September 2011
CSS Namespaces	Recommendation	29 September 2011
CSS Color Level 3	Recommendation	7 June 2011
CSS Level 2 Revision 1	Recommendation	7 June 2011
CSS Level 1	Recommendation	11 April 2008
CSS Flexible Box Layout	Candidate Recommendation	18 September 2012
CSS Values and Units Level 3	Candidate Recommendation	28 August 2012
CSS Backgrounds and Borders Level 3	Candidate Recommendation	24 July 2012
CSS Image Values and Replaced Content Level 3	Candidate Recommendation	17 April 2012
CSS Speech	Candidate Recommendation	20 March 2012
CSS Basic User Interface Level 3	Candidate Recommendation	17 January 2012
CSS Multi-column Layout	Candidate Recommendation	12 April 2011
CSS Style Attributes	Candidate Recommendation	12 October 2010
CSS Mobile Profile 2.0	Candidate Recommendation	10 December 2008
CSS Marquee	Candidate Recommendation	5 December 2008
CSS TV Profile 1.0	Candidate Recommendation	14 May 2003
CSS Print Profile	Last Call	13 October 2006
CSS Paged Media Level 3	Last Call	10 October 2006
Filter Effects	Working Draft	25 October 2012
CSS Counter Styles Level 3	Working Draft	9 October 2012
CSS Intrinsic & Extrinsic Sizing Module Level 3	Working Draft	27 September 2012
CSS Conditional Rules Level 3	Working Draft	11 September 2012
CSS Transforms	Working Draft	11 September 2012
CSS Fonts Level 3	Working Draft	23 August 2012
CSS Fragmentation Level 3	Working Draft	23 August 2012
CSS Regions	Working Draft	23 August 2012
Selectors Level 4	Working Draft	23 August 2012
Compositing and Blending	Working Draft	16 August 2012
CSS Text Level 3	Working Draft	14 August 2012
CSS Box Alignment Module Level 3	Working Draft	12 June 2012
CSS Exclusions and Shapes	Working Draft	3 May 2012
CSS Writing Modes Level 3	Working Draft	1 May 2012

CSS Animations	Working Draft	3 April 2012
CSS Transitions	Working Draft	3 April 2012
CSS Grid Layout	Working Draft	22 March 2012
CSS Positioned Layout Level 3	Working Draft	7 February 2012
CSS (Grid) Template Layout	Working Draft	29 November 2011
CSS Generated Content for Paged Media	Working Draft	29 November 2011
CSS Device Adaptation	Working Draft	15 September 2011
CSSOM View	Working Draft	4 August 2011
CSS Object Model	Working Draft	12 July 2011
CSS Lists and Counters Level 3	Working Draft	24 May 2011
CSS Cascading and Inheritance Level 3	Working Draft	15 December 2005
CSS Presentation Levels	Working Draft	13 August 2003

The previous version, CSS 2.1 is not so old. The last update was made in 7th June 2011 when it became a Recommendation. CSS 2 was promoted to the same status in 11th April 2008. The first draft was made already back in November 1997, soon after CSS 1 was released as a Recommendation in December 1996. Any major browser today can be expected to support CSS 2.1. The modules of CSS3 enable progressive enhancement on top of CSS 2.1 support.

2.5 JavaScript

JavaScript was initially created in 1995 by Netscape. In the beginning it was called Mocha, then LiveScript and by the end of 1995 the name JavaScript became permanent. This was only to use the popularity of Java to mislead developers to start using JavaScript and because of this, people are still confusing JavaScript to Java.

The first standardised version of JavaScript, called ECMA-262, was created during 1996 and 1997 by Ecma International. The standard is often referred as EcmaScript and the first version ES1. The second version, ES3 was created in 1999 and the third version in 2009. The next version ES6 is under way and planned to be released in 2012. ES4 was developed until 2008 it was dropped. [27]

JavaScript is the most fragmented part of the three building blocks for web applications. Each browser vendor has made their own implementations and additions to the language. EcmaScript standards are an effort for reducing the fragmentation and to

make the language match the needs of the developers. Table 2 lists the major browser vendors and the JavaScript rendering engines they are using. Due to this fragmentation several open source JavaScript libraries have been created for simplifying the browser differences, for example jQuery, which is by far the most popular JavaScript library today. [28]

Table 2. Major web browser JavaScript rendering engines [1:116]

Web Browser	JavaScript Engine
Microsoft Internet Explorer	JScript
Mozilla Firefox, up to 3.5	SpiderMonkey
Mozilla Firefox, from 3.6	TraceMonkey
Apple Safari, up to 3.2	JavaScriptCore
Apple Safari, from 4.0	Nitro
Google Chrome	V8
Opera	Futhark

In addition to the JavaScript rendering engines listed in Table 2, in the mobile platforms the most used browser engine is WebKit, which comes with its own JavaScript rendering engine called JavaScriptCore. However it is possible to use any other rendering engine when building a browser that is using WebKit engine. [29]

The fragmentation is one reason why it is not so clear to define version numbers for the language. According to Mozilla, they shipped Firefox 4.0 with JavaScript 1.8.5. [30_26]

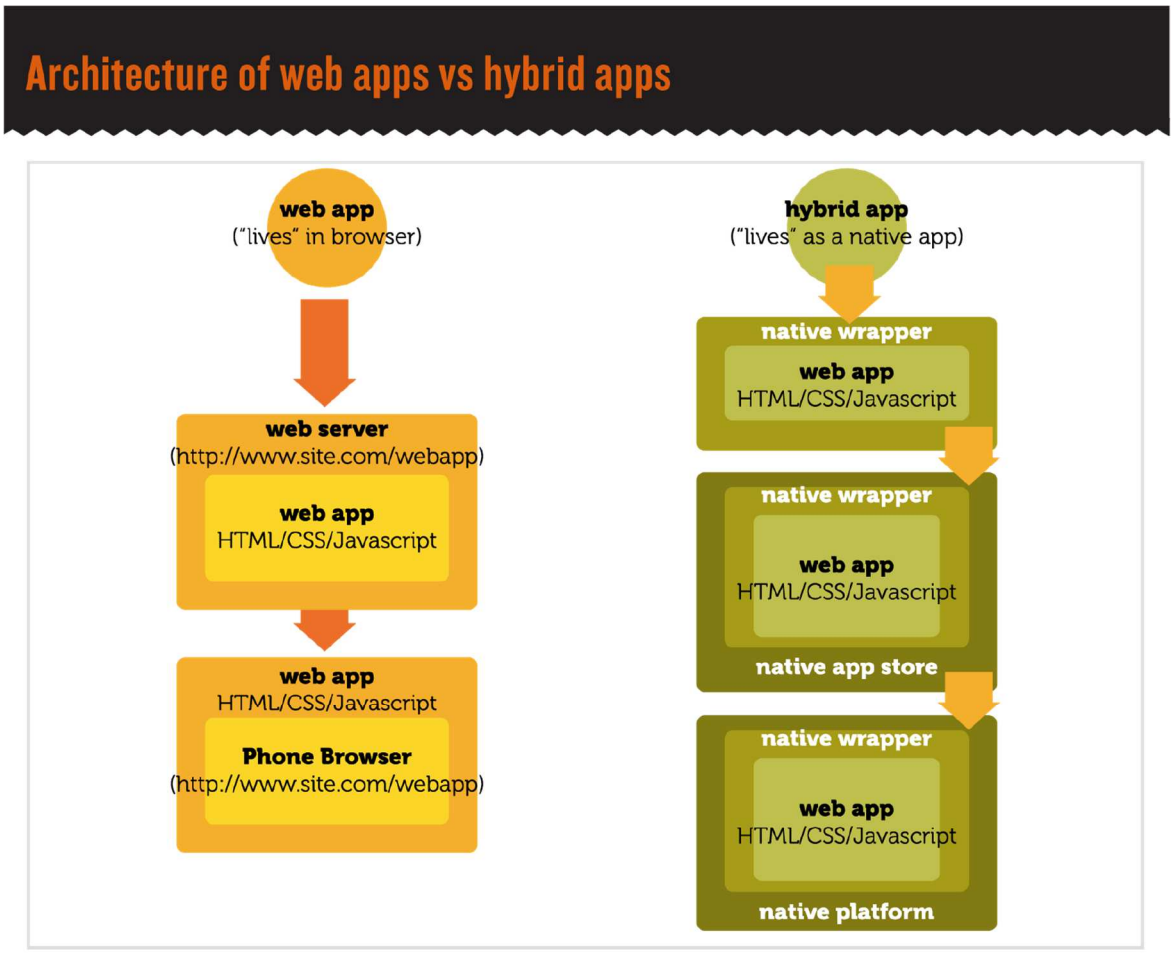
Since the creation of EcmaScript, JavaScipt is not the only language based on it. Also Actionscript 3 used by Adobe Flash is based on the ES5 standard. Popularity of JavaScript has been increasing during the past few years while it has been adopted in many production applications such as Adobe Creative Suite design tools and Qt C++ GUI library user interface scripting. It is no longer the World's most misunderstood language as Douglas Crowford said in 2008. [31]

JavaScript is the part of HTML5 based applications which does the communication between the device and the application. This makes it very important tool and language for the developers. During the last five years, JavaScript has been the most improvised feature in the web browser implementations. [32]

2.6 HTML5 as Mobile Application Platform

The main reason for making HTML5 and related technologies based mobile applications is the assumed existing engineering experience of the web developers. [8] Ideally the same set of tools and best practises could be used in the web and in the mobile device. However the security model is different, since a web site should not be able to access any sensitive data of the device, such as contacts data.

An application that is built for the mobile platform and installed on the device should have access to any reasonable data it would need. Modern mobile platforms have security models for per application per data type access rights. For example Android application manager in the device asks every time from the user which rights will be granted to the application when it is installed or updated.



Source: Cross-Platform Tools 2012 | www.CrossPlatformTools.com | February 2012
Licensed under Creative Commons Attribution 3.0 License
Sponsored by: Marmalade, Runrev LiveCode, Verizon Developer Community, Xamarin | Supported by webinos project

Figure 9. The structure of web apps in comparison to hybrid apps [33:30]

HTML5 applications can be developed in two ways. Plain HTML5 application is made only of text files that describe the HTML markup, CSS styling and JavaScript for interaction. These applications are run as any other web site in the browser with the exception that the files are most likely locally on the device. Hybrid applications are the next step toward more data access and platform specific code. The HTML5 part can be exactly the same but some amount of native code is needed for wrapping the browser rendering engine to the application. Both of these approaches are described in Figure 9, web app living in browser and hybrid app living as a native app. Developer satisfaction for HTML5 mobile applications are shown in Figure 10.

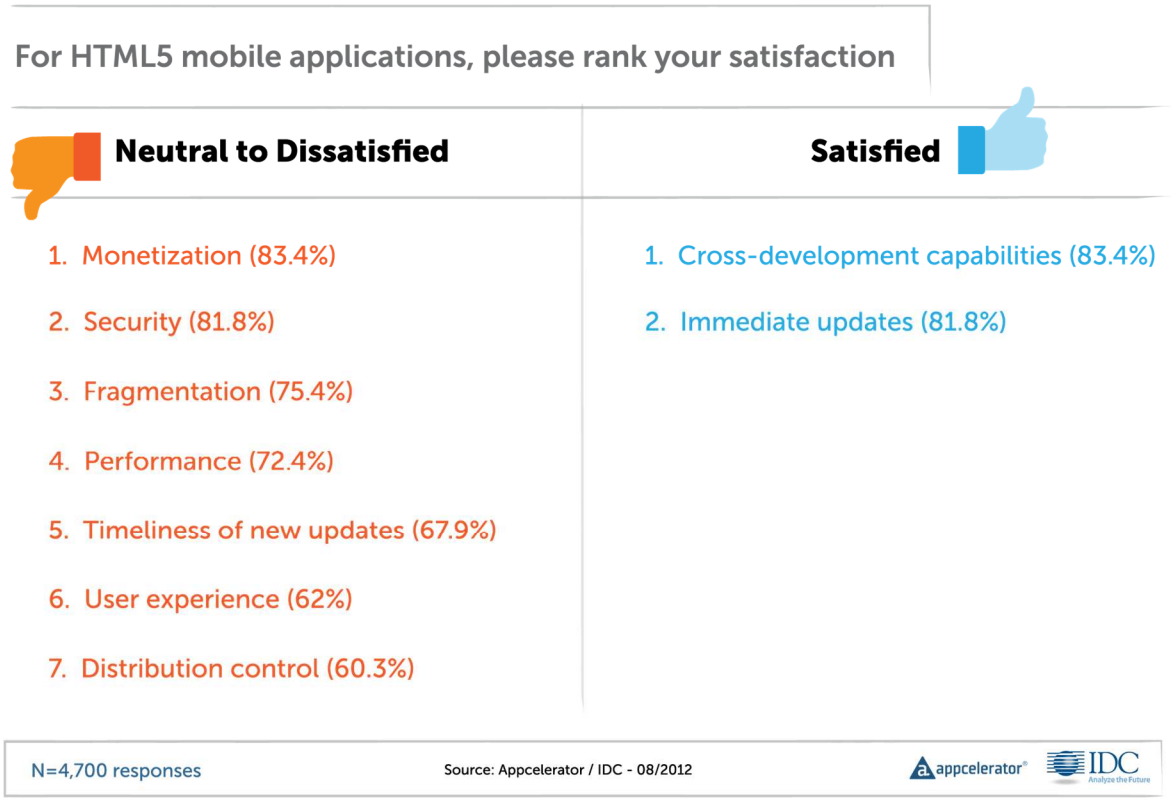


Figure 10. Developer satisfaction for HTML5 based applications [34:4]

According to Appcelerator survey results [34], developers are satisfied with the cross platform capabilities that HTML5 applications offer, but are dissatisfied by the fragmentation of the feature support and performance.

3 Mobile Platform Selection

Given the changing economics in the mobile device platform shares, the research was targeted to a limited selection of platforms. The selection of the mobile device platforms was based upon on their market shares during the last two years. Traditionally one manufacturer made both the software and the hardware for a given device, as it has been the case with Nokia until last year, when they started using an operating system from Microsoft, and with Apple. Recently there has been a separation of those two, for example with Google's Android operating system, with Samsung as the biggest device manufacturer using it. [35]

In addition to the market share, also the developer mindshare was evaluated for making a platform selection. Forthcoming device release, such as Nokia Lumia devices that come with Windows Phone 7 and 8 operating systems, attract the developers.

StatCounter provides global statistics on currently active Internet users. The five biggest operating systems for the last nearly three years according to StatCounter are shown in Figure 11. It can be seen that Symbian from Nokia, still dominant in 2008, lost initially its marker shares to iOS which was first available via iPhone and later via iPad. Starting in summer 2009, Android rocketed while taking market shares from Symbian and iOS, later from Blackberry too. [36]

During the summer 2012, feature phone platform Nokia S40 jumped on the list, mainly due to new Asha devices. Surprisingly there is no Windows Phone present in the statistics. Windows Phone is seen as the rising star, as the Developer Economics state that it is "the new cool". [37]

Back in November 2011, when Windows Phone Marketplace had existed for a year, Distimo, a mobile analytics company made a report of their research of the marketplace. [38] TechCrunch referred the results as:

"Today, Microsoft's Windows Phone is being hyped as the 3rd major mobile ecosystem, with some analysts even predicting it will steal huge chunks of mobile market share

away from Apple's iPhone by 2015. However, in terms of its app store, it's still the 5th largest in size, behind Apple, Android, BlackBerry and Nokia."

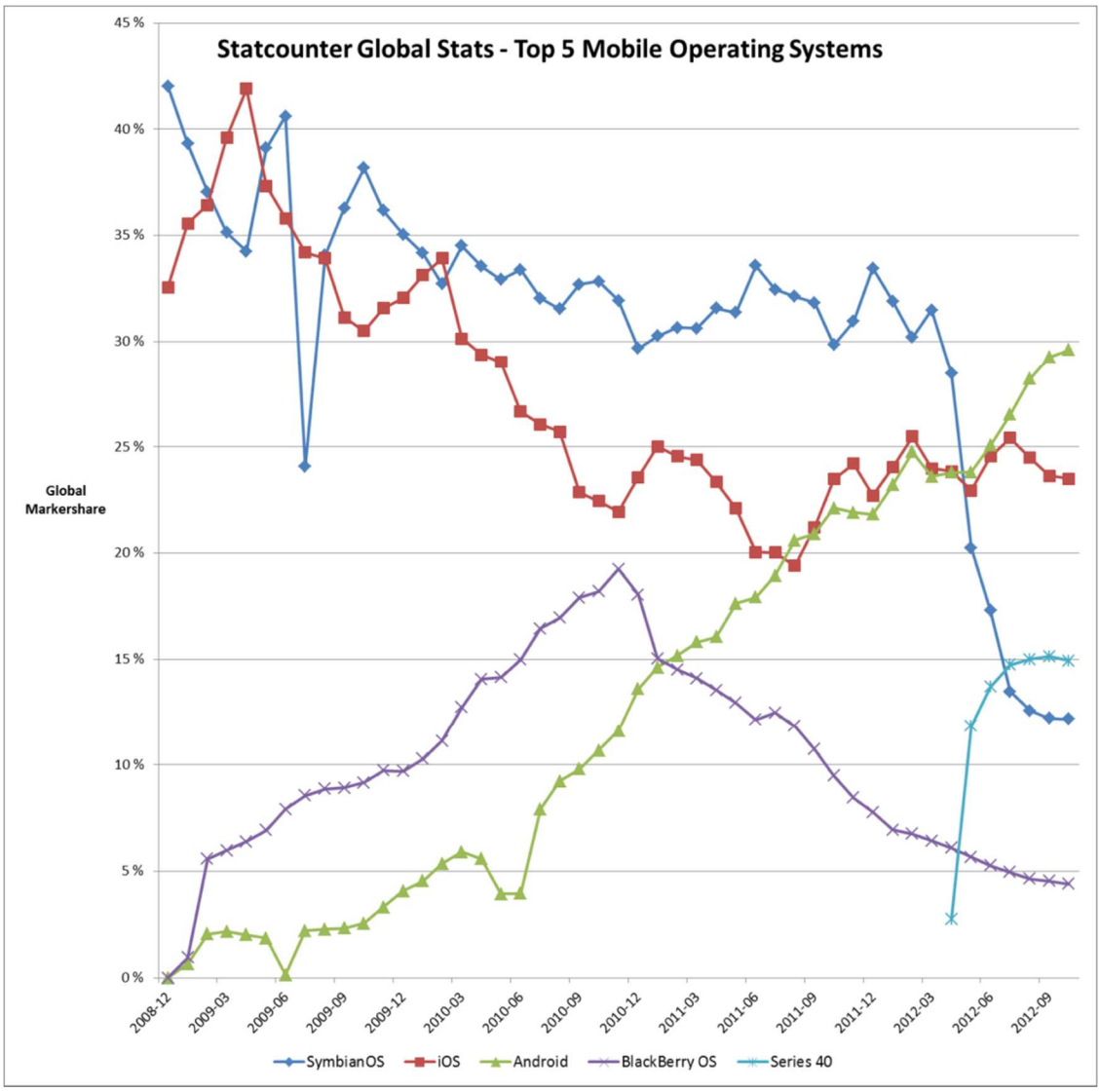


Figure 11. Top five mobile operating systems according to StatCounter [36]

Markets are clearly dominated by two big players, Google Android and Apple iOS. The same can be read from the developer mindshare where the situation has been like so for the last three years. In a similar fashion Blackberry has declined while Windows Phone is gaining interest, as Figure 12 visualises. [37]

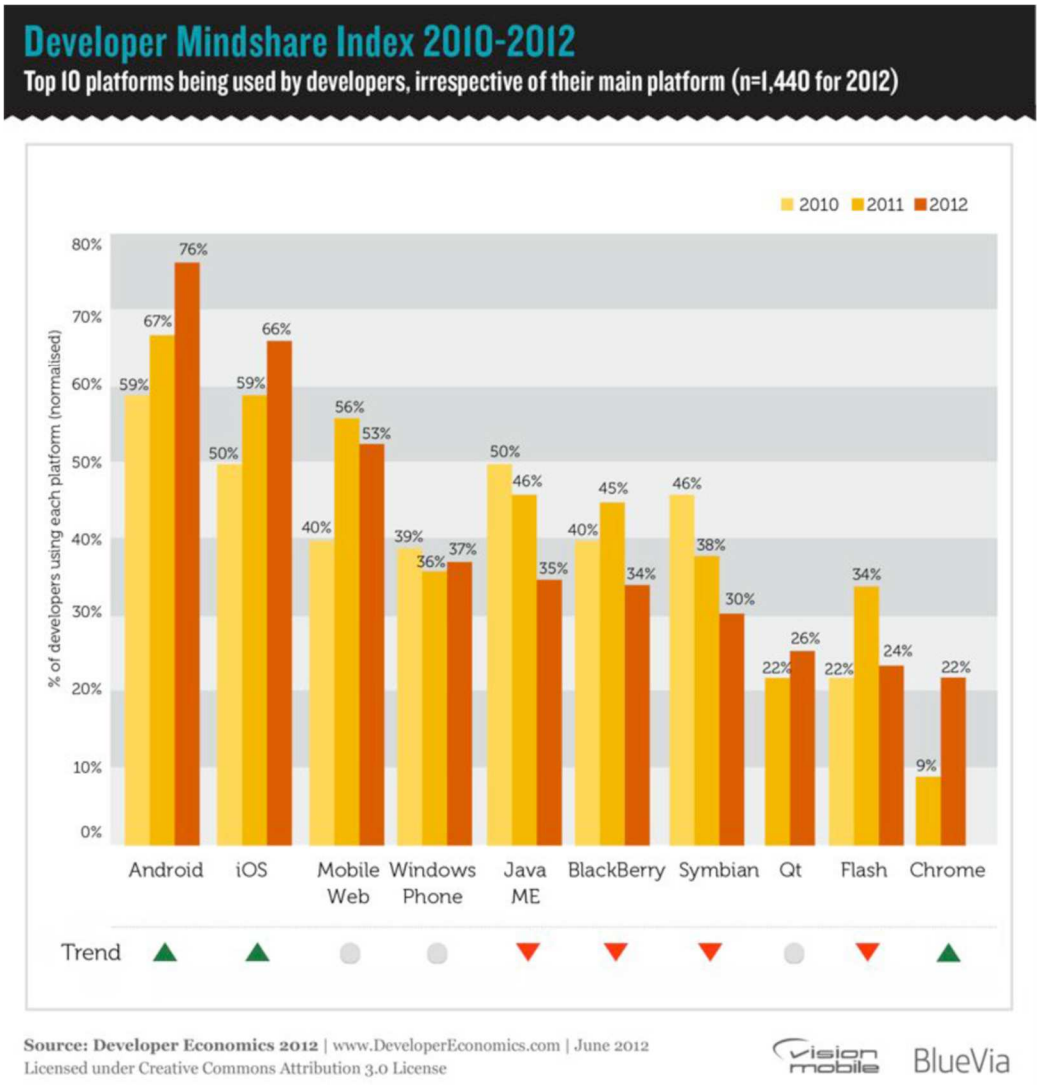


Figure 12. Developer Mindshare index 2010-2012 according to VisionMobile [37]

While the current state of the market share and developer mindshare reflect the past, the future intentions are equally interesting. While Android is slightly losing its attractions, Windows Phone is gaining it, as it can be seen in Figure 13, in the yearly comparison of the developer intentshare.

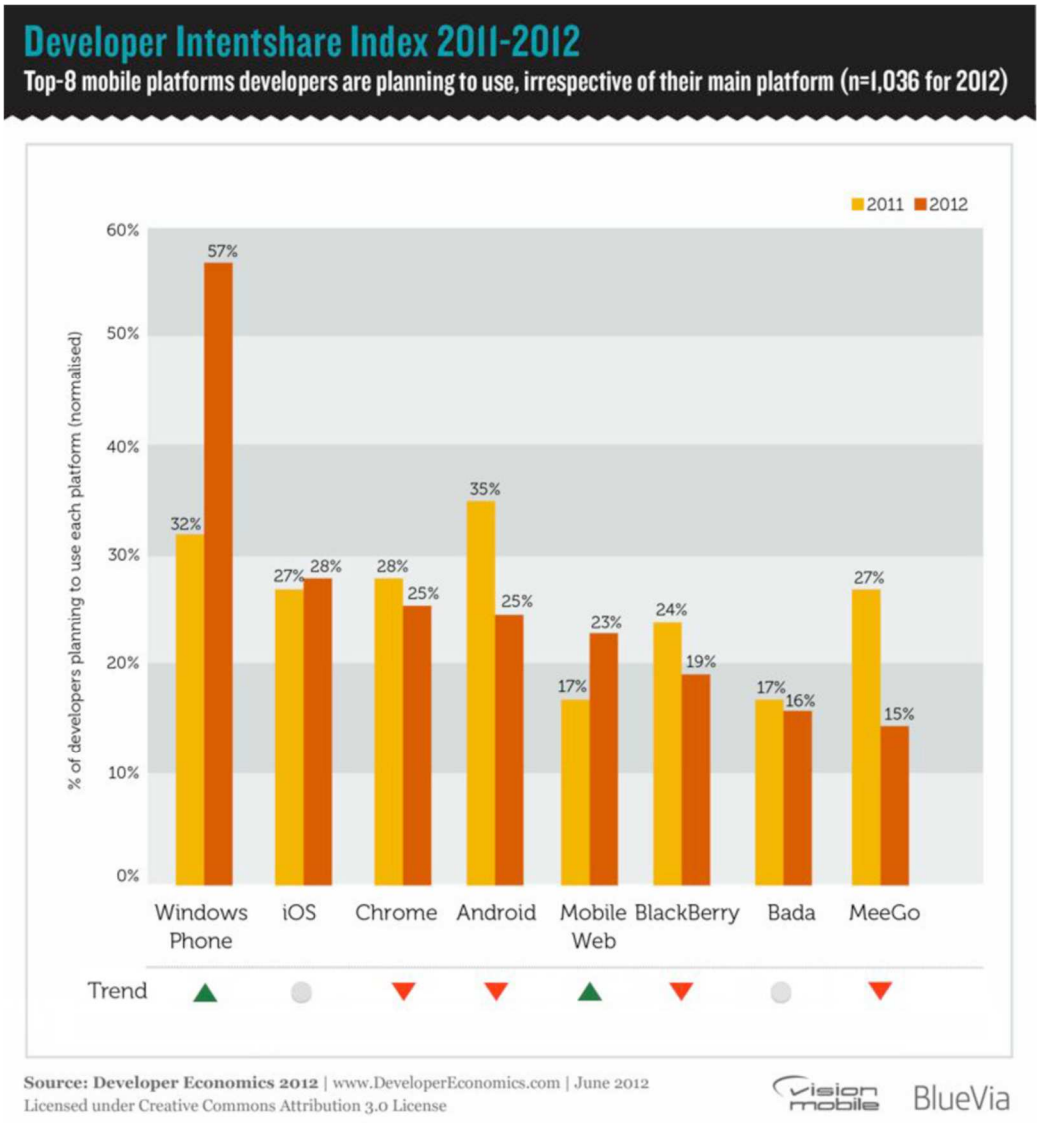


Figure 13. Developer Intentshare 2011-2012 according to VisionMobile [37]

Surprisingly Qt was getting more mindshare while the platforms supporting it, namely Symbian and MeeGo from Nokia, are both discontinued. However BlackBerry 10 with its C++ Cascades, using Qt might affect the numbers, but since the release date is postponed to 2013, it is unlikely. Figure 14 shows similar results from the developer survey by Appcelerator / IDC in the third quarter of 2012.

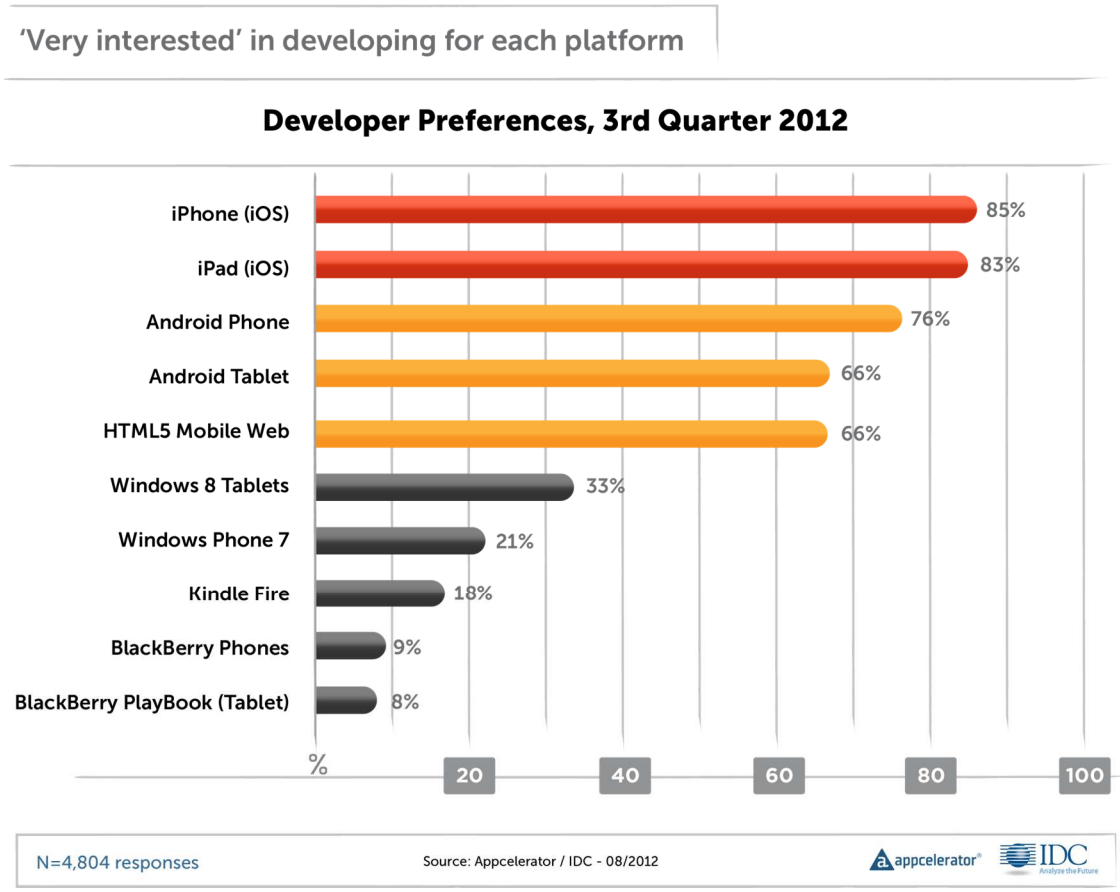


Figure 14. Developer platform preference results for the third quarter of 2012 by Appcelerator / IDC [34]

Appcelerator and IDC surveyed 5526 Appcelerator Titanium developers in August 2012 on their perceptions about current debates in mobile, social, and the cloud as well as their development priorities. The results show a similar trend towards the major platforms, as shown in Figure 14. Plain HTML5 applications are the most interesting development platform after the two dominant native platforms, iOS and Android. [34]

Figure 15 shows the comparison of the developer preference results between the quarterly surveys from January 2010 to August 2012.

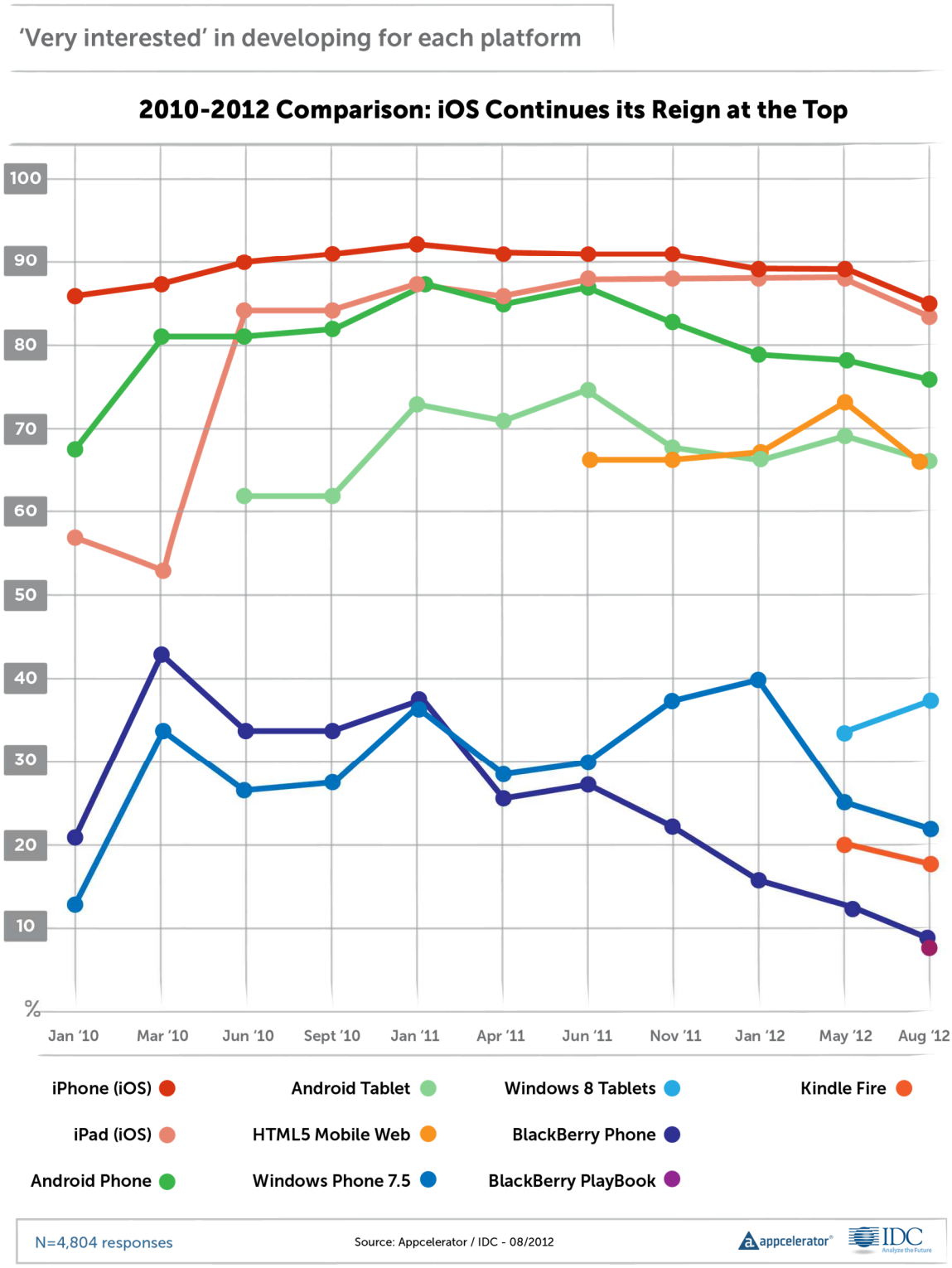


Figure 15. Interest rate of developer platforms during 2010-2012 [34]

Obviously the results differ between the companies making them, but one thing is clear, iOS and Android clearly dominate the market and developer interest.

It is hard to say how the questions have been set and what have been the options to choose from but, anyhow Windows Phone is there as the third platform and with that position it has more value than many other longer existing platforms. Figure 16 shows the number of devices shipped and devices currently in the market.

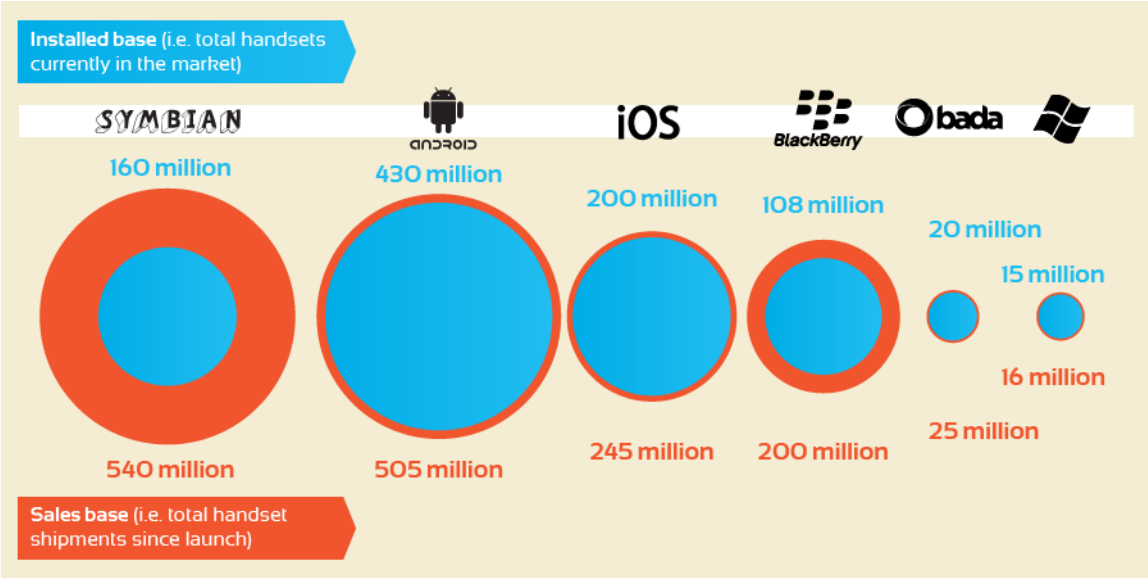


Figure 16. VisionMobile 100 Million Club infographic at the end of first half 2012 [39]

According to mobile device analytics company Gartner, the sale numbers for smartphones are only expected to grow, while in the third quarter of 2011 the sales increase was 42%. [40] Specifically Google Android is the most growing platform.

The platform versions tested were the latest publicly released versions that are already available or expected to be available soon in the devices on the market. The selected four mobile device platforms based on their popularity are:

- Apple iOS 6
- Google Android 4.2
- Microsoft Windows Phone 8
- RIM BlackBerry 10

The versions selected are the most recent versions made available for developers. At the moment only BlackBerry 10 devices are not available at the market. Developers are able to get an Alpha device from RIM but only by being personally present at any of

the BlackBerry Jam events. All of the four platforms come with a software simulator in the platform SDK which makes it possible to test some of the development aspects. However, true performance can only be tested on a real device. Figure 17 shows the developer survey results of the targeted form factors, placing smartphones first, tablets second with over half of developers targeting them.

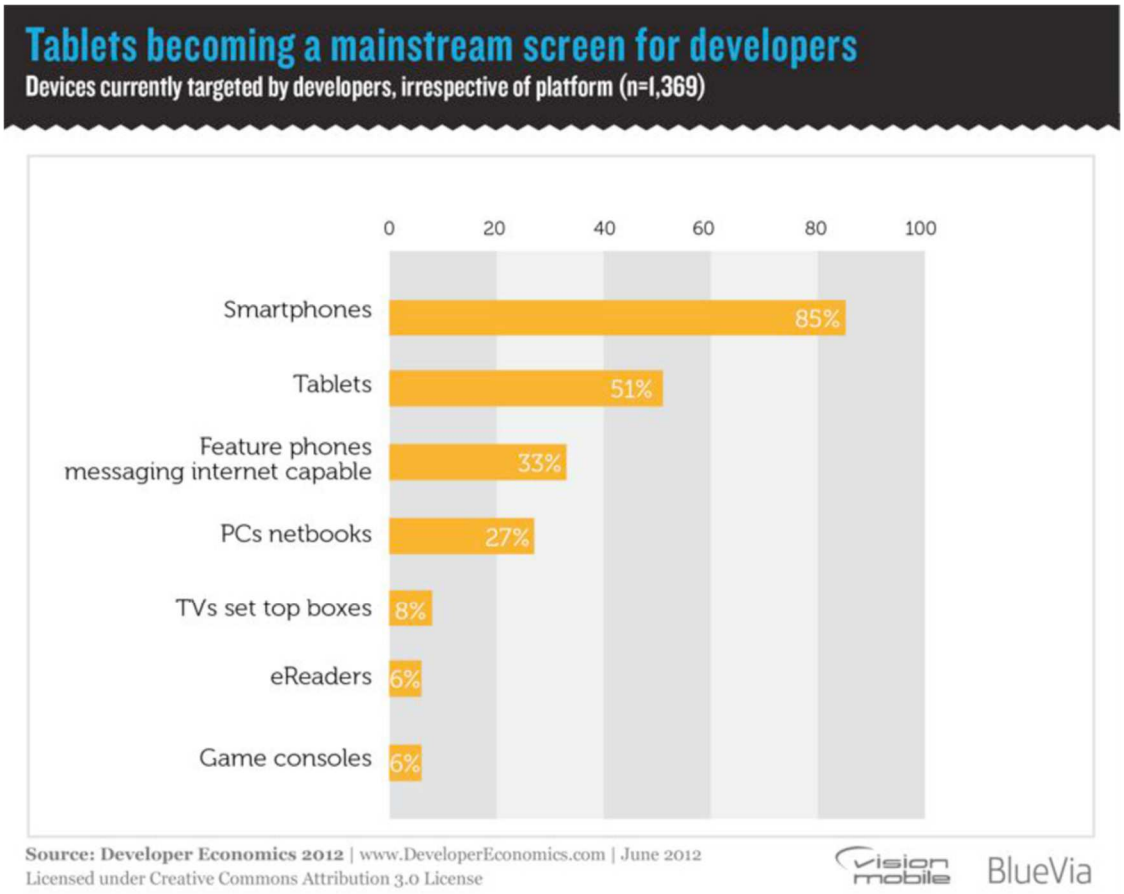


Figure 17. Form factor targets in the order of priority for developers [37]

Adding tablets and other form factors to the consideration, they are outside the scope of this research. However HTM5 enables for liquid layouts that realign and resize automatically via CSS method called media queries.

4 Mobile Platform Native Solution Analysis

This section evaluates the selected four platforms for their native application development solution and the ability to use HTML5 with related technologies. The native solution offers easy access to native look and feel via visual building blocks and theming. The ability to reach similar user interface by using HTML5 was evaluated.

The main platforms in today's market according to Mobile Developers Guide to the Galaxy are described in Table 3 [41], with their programming language options, supported form factors and developer access URL.

Table 3. The main mobile device platforms in the market today [41]

Platform	Language(s)	Form factor	URL
Aliyun	Java, C, C++	Smartphone	developer.aliyun.com
Android	Java, C, C++	Smartphone, Tablet, TV	developer.android.com
bada	C, C++	Smartphone	developer.bada.com
BlackBerry	Java, Web Apps	Smartphone	developer.blackberry.com
BlackBerry Playbook OS (QNX)	ActionScript, C++, HTML	Tablet	developer.blackberry.com
BlackBerry 10	C, C++, HTML, AIR	Smartphone, Tablet	developer.blackberry.com
Brew MP	C	Featurephone	brewmp.com
Firefox OS	HTML	Smartphone	mozilla.org/en-US/b2g
iOS	Objective-C, C	Smartphone, Tablet	developer.apple.com/devcenter/ios
Mer	HTML, C/Qt	Smartphone	merproject.org
Nokia OS	Java ME	Featurephone	developer.nokia.com/Develop/Series_40
Symbian	C, C++, Java, Qt, Web Apps, others	Smartphone	forum.nokia.com/symbian

Tizen	HTML	Smartphone	tizen.org
Windows 8	C#/VB.NET, C++, JavaScript	Tablet, PC	dev.windows.com
Windows Phone 7	C#, VB.NET, C++	Smartphone	dev.windowsphone.com

In order to develop applications for more than a single platform, most likely desktop computers running with Apple Mac and Microsoft Windows are needed. Table 4 clarifies the requirements for developer environments.

Table 4. Development Requirements [41]

Mobile OS	Operating System for development	Software/IDEs	Programming Language
Android	Windows/Mac/Linux	Eclipse/Java/Android Development Tool (ADT)	Java
BlackBerry	Windows mainly	Eclipse/JDE, Java	Java
iOS	Mac only	Xcode	Objective C
Symbian	Windows/Mac/Linux	Carbide.c++	C++
WebOS	Windows/Mac/Linux	Eclipse/WebOS plugin	HTML/JavaScript/C++
Windows Phone 7	Windows mainly	Visual Studio 2010	C#, .NET, Silverlight or WPF

In addition to the platforms listed in Tables 3 and 4, Windows Phone 8 follows the similar approach as Windows Phone 7, except the development is limited to a 64 bit Windows 8 and Visual Studio 2012. [42]

4.1 Apple iOS

Apple became immediately popular once it entered the mobile device market space with iPhone. With iPhone Apple introduced iOS, their mobile device operating system which is used in iPhone, iPad, iPod Touch and Apple TV devices. The programming language used for development is called Objective-C, which is a superset of C with

syntax of Smalltalk. Development tools Xcode and iOS SDK are only available for Mac OS X. [43]

The latest version of the operating iOS 6 was released in September 2012. Apple delivers it as an upgrade for iPhone versions starting from 3GS and iPad 2 onward. Not all features of the latest version will be supported in the earlier hardware versions since the clock speed, memory and other resources differ. Complete release history of the platform is listed in Table 5. The name iOS did not appear until 2010 when introduced with the 4.0 release. Before that it was called iPhone OS since iPhone was the only device using it.

Table 5. iOS version release dates and the corresponding devices

iOS Version	Xcode Version	Date	Devices initially installed with the iOS version
1.0		2007-06-29	First iPhone released
2.0		2008-07-11	iPhone 3G
3.0		2009-06-17	iPhone 3GS
3.2		2010-03-02	First iPad released
4.0	3.2.3	2010-06-21	iPhone 4
4.1	3.2.4	2010-08-17	iPod Touch 4th generation, Apple TV 2nd generation
4.2	3.2.5	2010-11-15	
4.3	3.2.6	2011-03-11	iPad 2
5.0	4.3	2011-10-12	iPhone 4S
5.1	4.4	2012-03-16	iPad 3rd generation, Apple TV 3rd generation
6.0	4.5	2012-09-19	iPhone 5, iPad 4th generation, iPad Mini

Xcode comes with an editor, compilers and other tools required for application development. iOS SDK contains the framework libraries, software simulator and other version specific tools. [24] Only registered developers can access the SDK, but the registration for Apple Developer Center is free of charge. In order to publish applications to the iTunes Store and iOS devices, subscription to the iOS Developer Program is needed. At the moment it costs 75 EUR per year. [44]

iOS applications can be developed as native, hybrid and plain HTML5. Plain HTML5 applications are launched in Safari web browser and hybrid applications use the same rendering engines as Safari. The software architecture in both native and hybrid applications are structured as shown in Figure 18. [45]

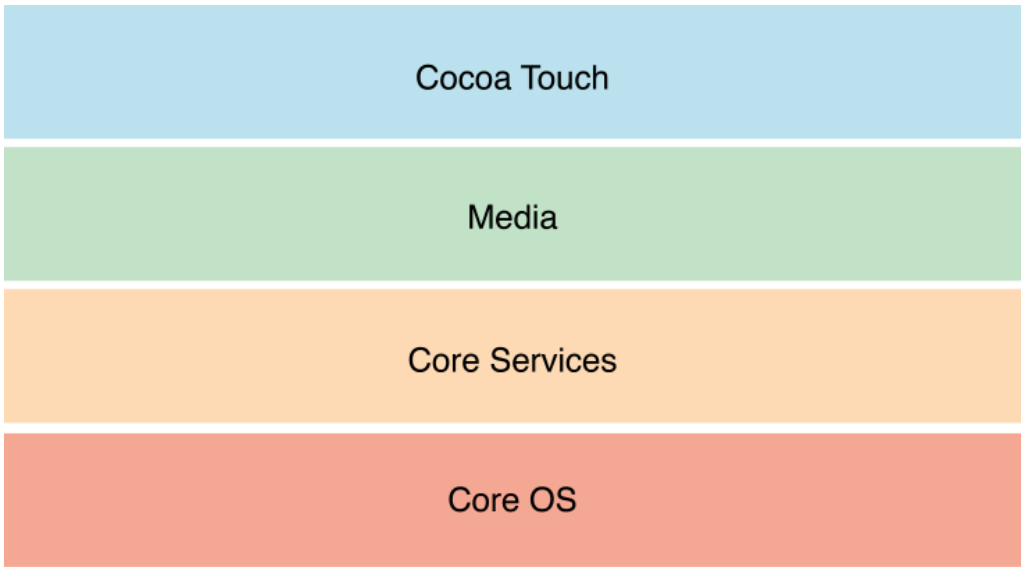


Figure 18. iOS software architecture layers [45]

The iOS software APIs are packaged as a set of frameworks. Each framework is assigned to a certain layer of the architecture, thus some frameworks are low level, near to hardware and some high level abstractions. Many of the frameworks are shared in both developer platforms iOS and Mac OS. The latter is for desktop development and not discussed in the research.

Apple organises once a year the Worldwide Developer Conference (WWDC) where they share the latest updates to the developer platforms. In June 2012 was the 23rd time of this event which makes it the longest running developer event in the world. In the keynote they revealed that there are over 400 million iTunes Store accounts, 650 000 iOS based applications of which 225 000 specifically for iPad. In regards for the iOS version fragmentation already over 80% of all the 365 million devices are running iOS 5 as shown in Figure 19. [46] In the Apple Special Event for publishing new devices in October 2012, they announced that the latest version, iOS 6 is already in over 200 million devices, just after being available for one month. According to Apple this is the fastest upgrade rate in any software in history. [47]

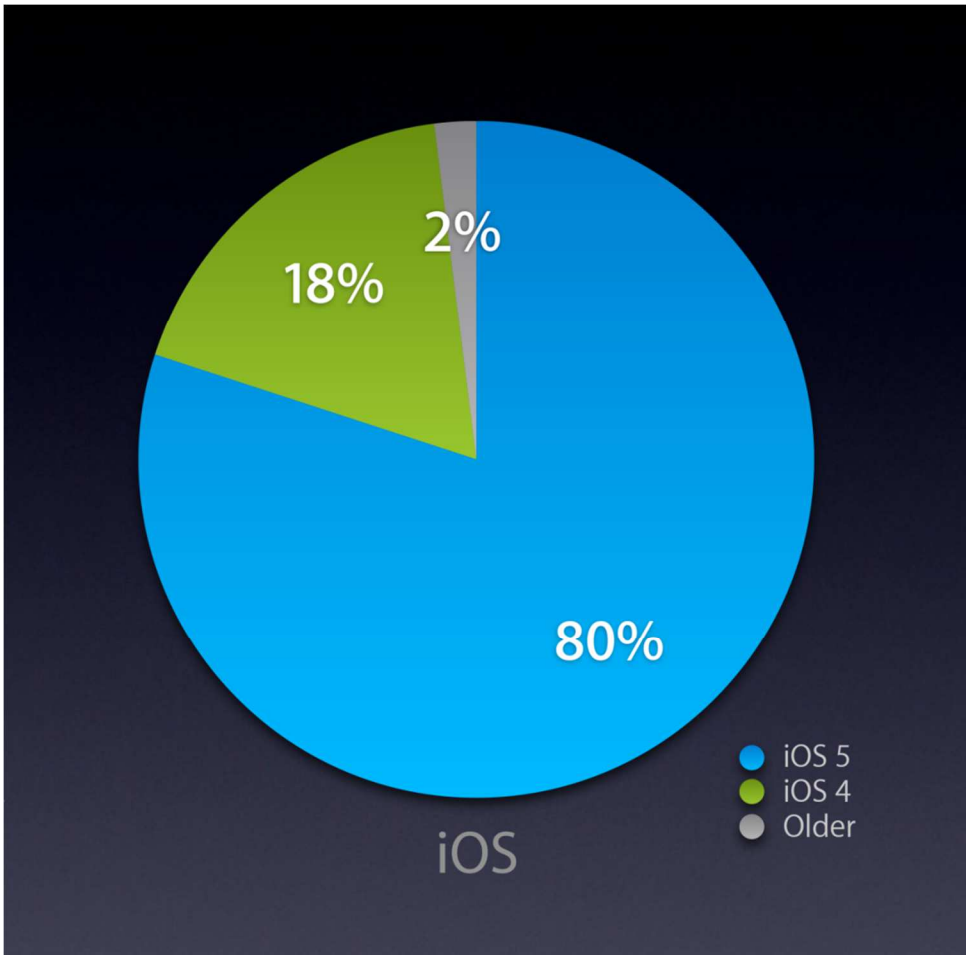


Figure 19. iOS version adoption in June 2012 of all sold devices [46]

The fast upgrade cycle to the older devices makes the platform fragmentation smaller and allows developers to use the latest features available in the platform.

4.2 Google Android

Android is a Linux based mobile device operating system created by Google in 2008. The latest version 4.2 was released in November 2012 but most of the devices are running older versions, as shown in Table 6 and in Figure 20. Even with the version 4.0 being released in 2011 for smartphones and tables, 55% of the devices in the market are still using the version 2.3 in their devices. [48]

Table 6. Android Platform versions in October 1st 2012 [48]

Version	Codename	Initially released	API Level	Distribution
1.5	Cupcake		3	0.1%
1.6	Donut		4	0.4%
2.1	Eclair		7	3.4%
2.2	Froyo	May 2010	8	12.9%
2.3 - 2.3.2	Gingerbread	Dec 2010	9	0.3%
2.3.3 - 2.3.7		Feb 2011	10	55.5%
3.1	Honeycomb (tablet only)	May 2011	12	0.4%
3.2		Jul 2011	13	1.5%
4.0.3 - 4.0.4	Ice Cream Sandwich	Dec 2011	15	23.7%
4.1	Jelly Bean	Jun 2012	16	1.8%

Android is the most rapidly evolving development environment, updates for the SDK and tools are published almost every second month. As a negative side to this, device manufacturers find it hard to keep up and that is why the devices in shops are already outdated. New platform version migration to older devices is fairly slow and dependant of the device manufacturers thus creating fragmentation.

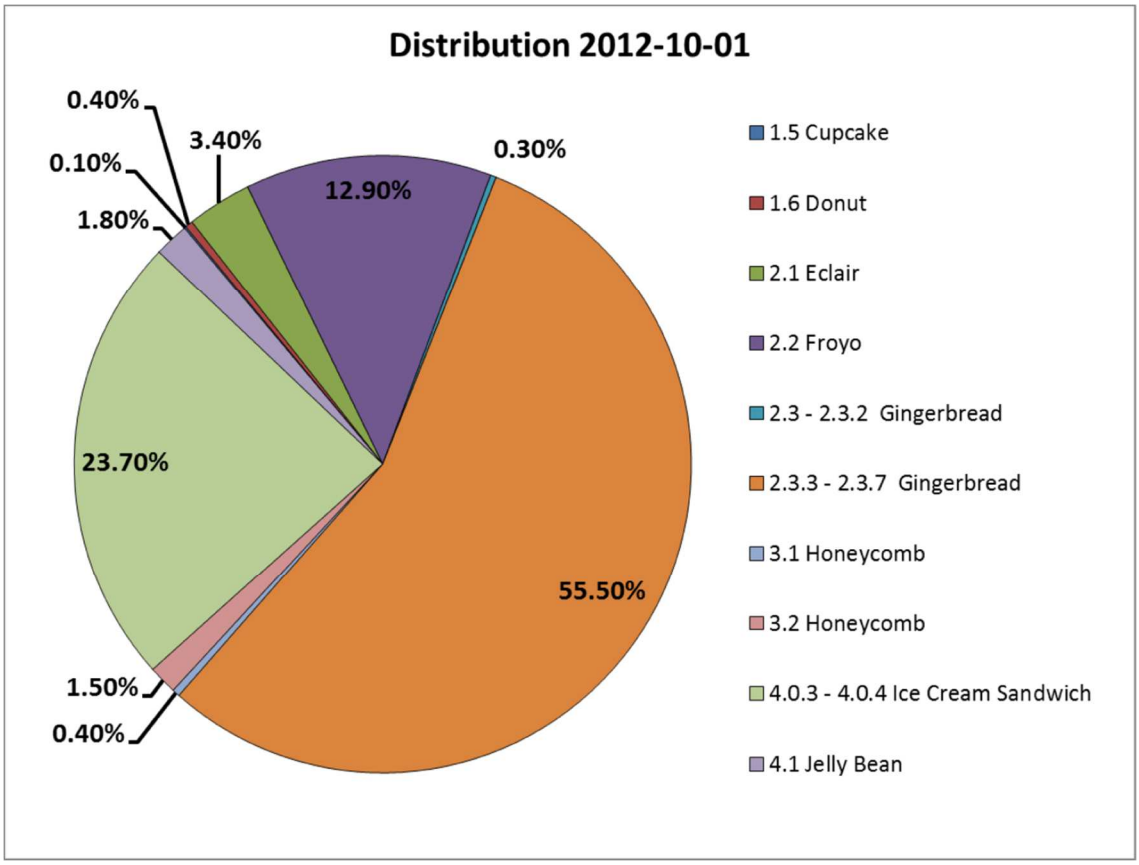


Figure 20. Android platform version distribution in 1st October 2012

Applications are developed in Java, while the user interface is written in Android SDK specific XML. HTML5 development is possible with a wrapper application that defines the rendering engine and its layout. [49] The class used for this is called WebView that is using the same WebKit rendering engine as the native browser. By default it does not enable JavaScript processing. [50] The code required for enabling it is shown in Figure 21.

```
13          setContentView(R.layout.activity_main);
14
15          WebView myWebView = (WebView) findViewById(R.id.webview);
16          WebSettings webSettings = myWebView.getSettings();
17          webSettings.setJavaScriptEnabled(true);
18          webSettings.setAllowFileAccess(true);
```

Figure 21. Enabling JavaScript processing in Android WebView

The SDK comes with separately downloadable APIs for different API levels and the software simulator Android Virtual Device. The virtual devices can be configured to

emulate different platform versions, screen sizes, memory and other hardware resources. The preferred IDE is the open source code editor Eclipse for which Google provides the Android Developer Tools plugin. [51:23] All of the tools including Eclipse can be downloaded free of charge and without registration.

4.3 Microsoft Windows Phone

Windows Phone is a new platform created by Microsoft. It started with version number 7, most likely for marketing purposes for promoting the desktop platform Windows 7. The earlier mobile device platform from Microsoft, Windows Mobile, existed until version 6.5 which is often the source for confusion. The operating system was said to be rebuilt and as a development environment very different. [52] Windows Phone 7.0 and 7.5 were relatively strict platforms that allow application development only in Silverlight and with very limited capabilities to access native features.

The research originally included the Windows Phone 7.5 (WP7), but as Windows Phone 8.0 (WP8) was published during the Microsoft developer event called Build, the research was quickly adopted to these remarkable changes.

Tools can be downloaded for free, including the Windows Phone 8 SDK and Visual Studio 2012 Express. These can be installed only to a 64 bit Windows 8 desktop operating system. Testing can be done with a software emulator that comes with the SDK. [42] Testing on a real device as well as publishing applications to the Windows Phone App Store requires the developer subscription from Dev Center that has annual 75 EUR fee.

WP8 was released in late October 2012 and allows application development in several ways with much richer APIs and features in comparison to the earlier versions. [52] HTML5 applications in WP8 are using the Internet Explorer 10 rendering engine for mobile. Earlier version applications will run in IE9 compatibility mode. [53]

While WP7 only had Silverlight as the developer environment, WP8 has additionally C++ that uses DirectX APIs, similar to the desktop environment. Silverlight applications are written in C# and XAML. C# is a programming language developed by Microsoft and is similar to Java. XAML is a dialect of XML that is used to define user interfaces.

These models and the high level view of the available APIs are displayed in Figure 22. Note that the name Silverlight is no longer used. WP7 had a subset of Silverlight 4.0 with limited set of APIs from the desktop version, but additional mobile device specific APIs. [54:5]

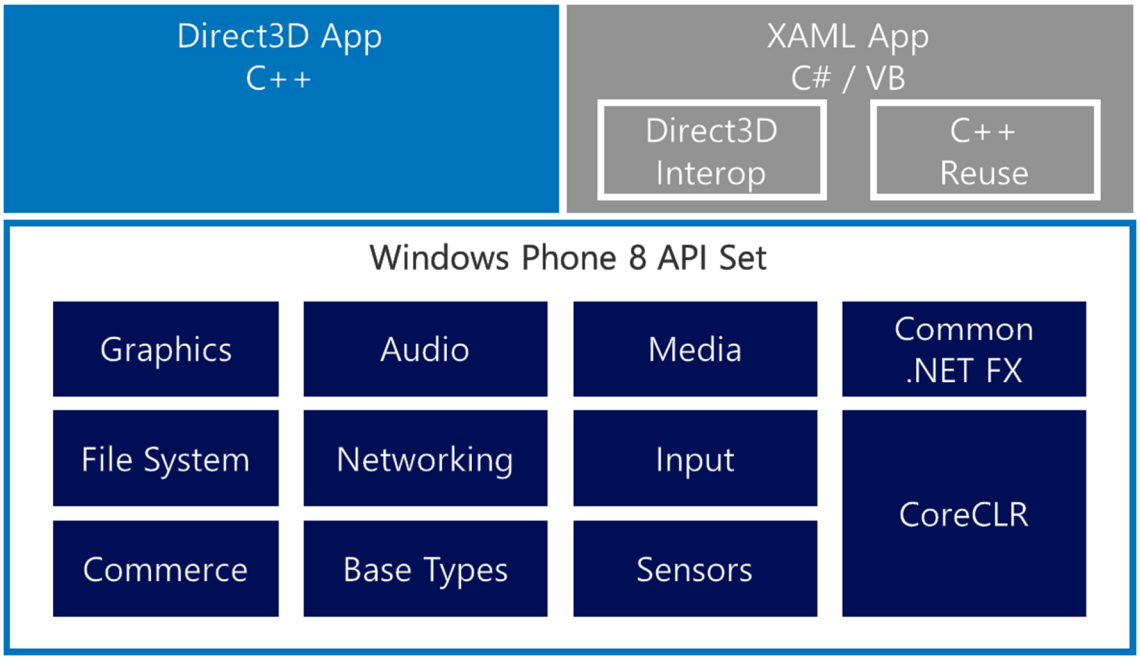


Figure 22. Windows Phone 8 Application models [55]

Most of the WP8 APIs are shared with the new version of the desktop Windows 8. Applications written for WP7 are recompiled by Microsoft in their App Store before offered as a download for WP8 and all applications that are published to App Store are also recompiled with the Native Image Creator (NGEN). This "compile in the cloud" feature is promised to dramatically increase the speed of the applications by removing the need for Just In Time (JIT) compilation on the device. [56] XAML text files are also compiled to binary files making them faster to parse.

HTML5 based hybrid applications are developed inside the XAML App approach, by using the WebBrowser control. It is possible to link C++ code to JavaScript via XAML and WebBrowser, thus enabling same access level to HTML5 as native applications. However all the required data needs to be passed through by the application which creates additional work for the applications developers. By default the WebBrowser control does not allow JavaScript to run thus the developer needs to enable it. Addi-

tionally the application needs to have permission to use the control and possibly access the network. [53]

Earlier version of the platform had only one target frame size. WP8 supports three different frame sizes thus the application needs to be aware of the surrounding resources. Different sizes are listed in Table 7 with the comparison to WP7 resolution. Existing applications for WP7 are scaled while for WP8 it is recommended to use three different sets of graphical assets, each for available resolution. [57]

Table 7. Resolutions and aspect ratios that are supported in Windows Phone 8 [57]

Resolution name	Resolution pixels	Aspect ratio	Delta from WP7	Scaled resolution
WVGA	480 × 800	15:9	None. This is the only supported resolution for WP7.	480 × 800
WXGA	768 × 1280	15:9	1.6x scale	480 × 800
720p	720 × 1280	16:9	1.5x scale, 80 pixels taller (53 pixels, after scaling)	480 × 853

It should be noted that devices running WP7 are not upgradable to WP8 since the device requirements are different. This increases the version fragmentation and requires developers to create two versions of their applications in case they wish to target the whole market of Windows Phone customers.

Communication from WebBrowser control to JavaScript and back is done by using a single available method that can only pass strings as its arguments. This limitation requires binary data such as images to be Base64 encoded in the calling side and decoded in the receiving side.

4.4 RIM Blackberry

Research In Motion (RIM) will launch BlackBerry 10 (BB10) in the end of January 2012. [58] The platform has been available for some time as a developer alpha and beta releases. The third beta iteration of the OS was released in late September 2012 and while the alpha hardware built by RIM was updated, it is only available to developers

participating to the worldwide developer events called BlackBerry Jam. The final version of the SDK is announced to be released in 11th December 2012. [59] The BlackBerry App World market place for applications has already been opened to accept BB10 applications.

The platform offers several ways to develop applications, such as WebWorks that uses HTML, CSS and JavaScript, and on the opposite side Cascades, which enables native C++ development with Qt GUI library. More specific details are described in Table 8, which lists the programming languages available development solutions. PlayBook 2, the tablet OS from RIM also supports the same development solutions as listed in Table 8, thus reducing the amount of work needed when targeting both smartphones and tablets. [60]

Table 8. Available development solutions for BlackBerry 10 [60]

BlackBerry 10 Platform	Language(s)	Notes
Native SDK	C, C++	Close as hardware as possible. Uses QNX Momentics IDE
Cascades	Qt C++	UI can be written in QML. Can use Cascades builder which is available via QNX Momentics IDE
WebWorks	HTML5, CSS3, JavaScript	Built on top of WebKit rendering engine with access to native APIs
Adobe AIR	ActionScript	Access to native APIs only via building an AIR Native Extension
Android Runtime	Java	Repackaging tools for existing Android 2.3.3 applications

WebWorks is the solution of choice for HTML5 hybrid application development. It dives down to native API via WebKit rendering engine as visualised in Figure 23, starting from native API to WebWorks platform which then uses WebKit for rendering the HTML5, CSS3 and JavaScript files.

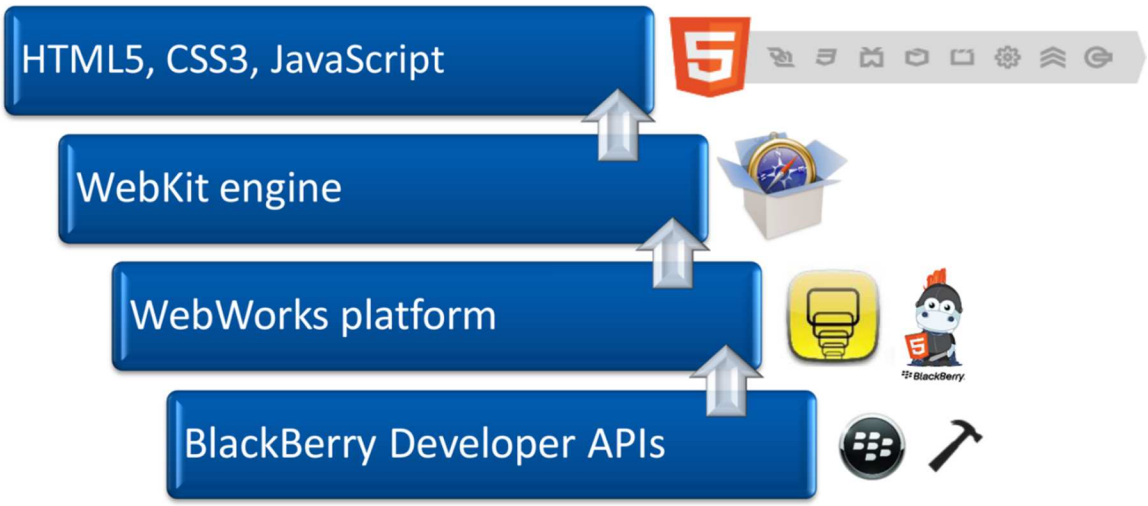


Figure 23. BlackBerry 10 WebWorks native to HTML5 layers [61]

The tools can be downloaded for free and publishing applications to the App World only requires free registration. Software simulator and each of the development solution SDKs needs to be downloaded separately. The BB10 simulator comes as a virtual OS image that requires VMware Player or similar virtual machine software to run. One such free software is VirtualBox, available at https://www.virtualbox.org/.

Finally, BlackBerry 10 is very promising platform, but as it is still in beta and devices unavailable, further investigation is not possible.

5 Cross Platform Solutions

Growing fragmentation in the mobile device market has created the need for cross platform solutions that would reduce the amount of work required while developing applications for more than one platform.

The cross platform solutions tested in this research were selected on the basis of their popularity. The popularity was determined by the developer mindshare shown in Figure 24 and the developer intentshare shown in Figure 25.

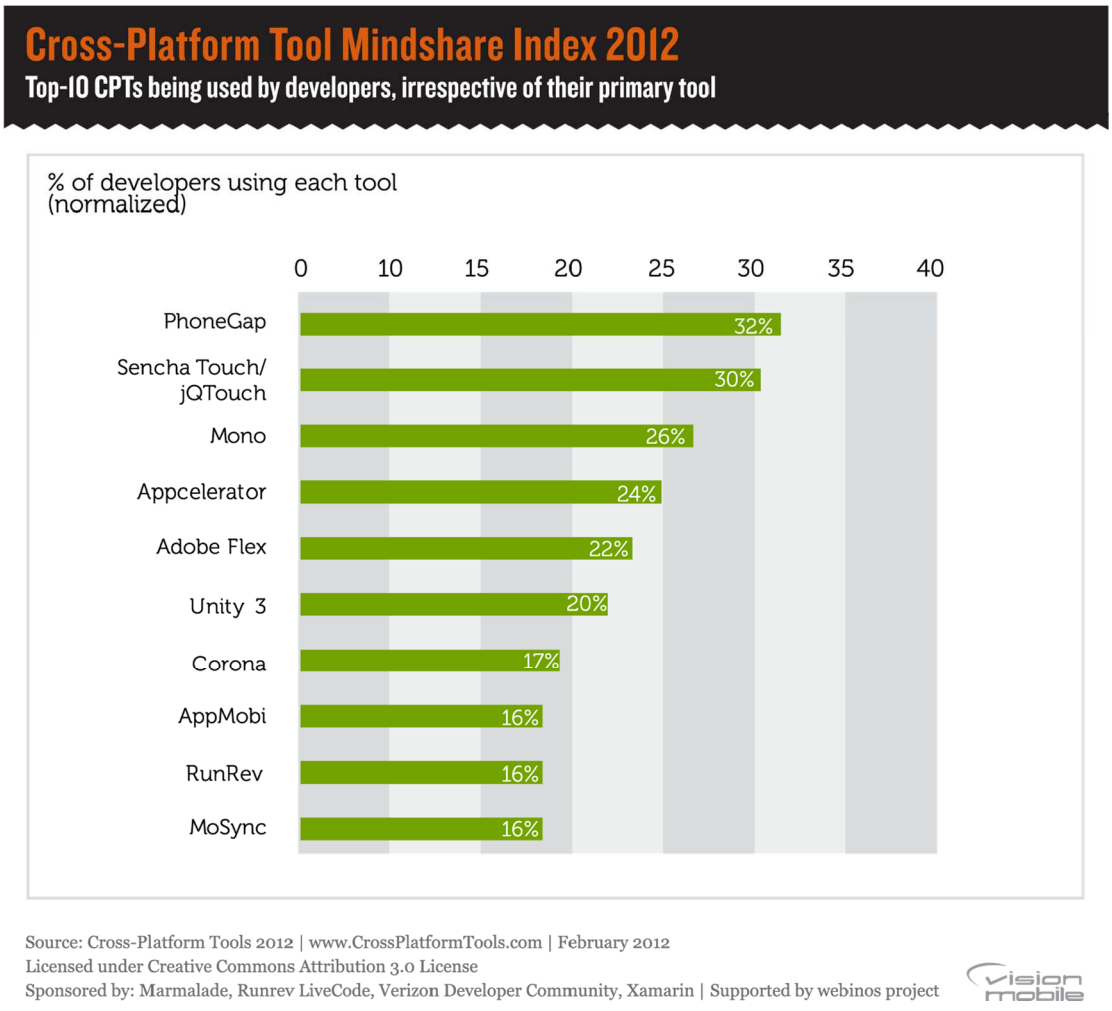


Figure 24. Developer mindshare of the cross platform tools [33:34]

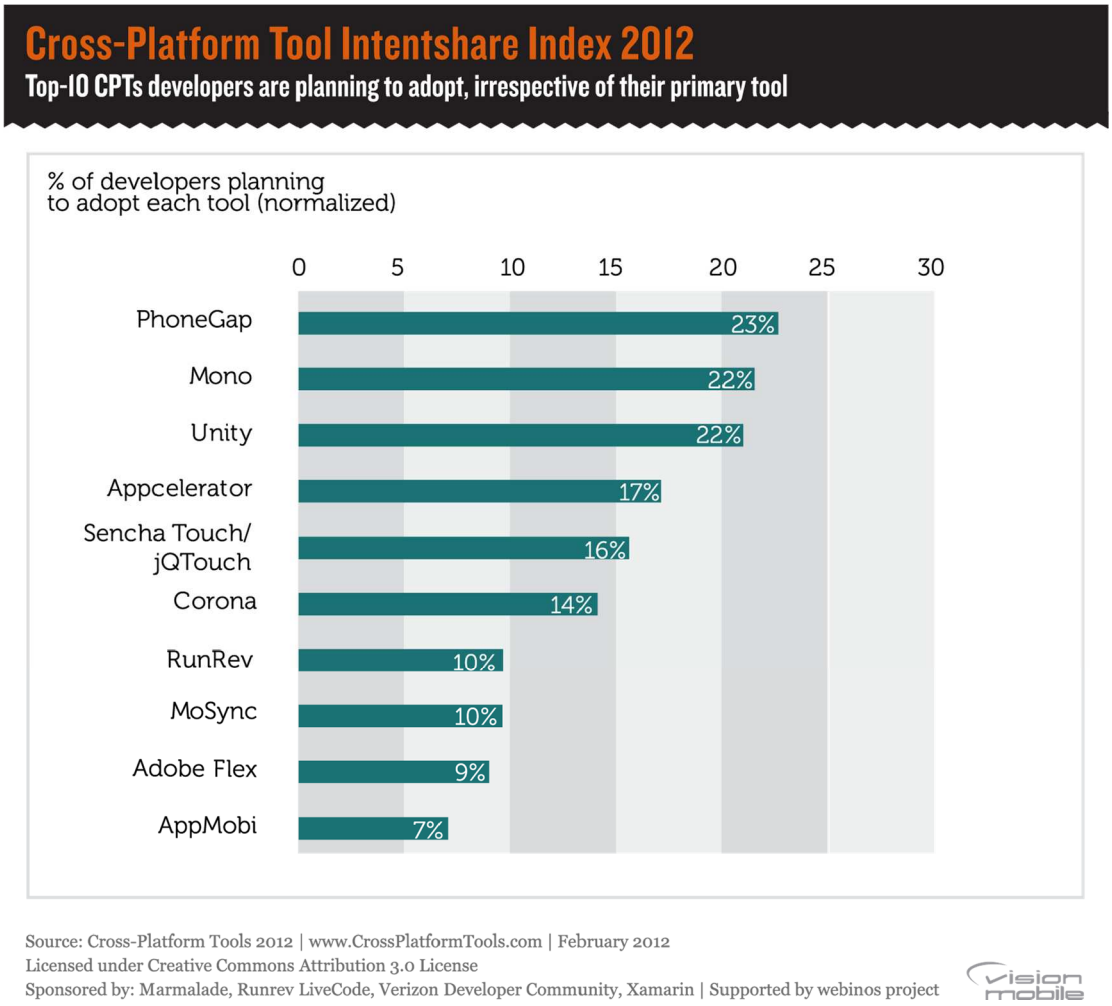


Figure 25. Developer intentshare of the cross-platform tools [33:35]

Four most popular frameworks were evaluated for their use as replacement for native application development. Popularity was based on the Cross-Platform Tools 2012 research made by VisionMobile. [33] The following frameworks were selected:

- Adobe PhoneGap
- Appcelerator Titanium
- Sencha Touch
- Xamarin Mono

In the context of the four selected mobile device platforms, most of the solutions have support for most of the platforms, but only PhoneGap is reported to support all of them. Table 9 shows the mobile device operating system support per cross platform solution.

Table 9. Mobile device operating system support per cross platform solution

	Apple iOS	Google Android	Microsoft Windows Phone	RIM Blackberry
Adobe PhoneGap	Yes	Yes	Yes	Yes
Appcelerator Titanium	Yes	Yes		Beta
Sencha Touch	Yes	Yes		Yes
Xamarin Mono	Yes	Yes		

As Table 9 indicates, at the moment only PhoneGap supports all of the four most popular platforms. It should also be noted that most of the cross platform frameworks that were evaluated before the selection only support the two most popular platforms, Android and iOS. Most of the solutions included in the research have been or are planning to expand their reach for other platforms as well. Table 10 lists the licensing, pricing and additional notes on the tooling of the cross platform solutions.

Table 10. Licensing, pricing and tools of the cross platform solutions

	License	Price	Tools
Adobe PhoneGap	MIT	Free	Source code, libraries, plugins. No IDE
Appcelerator Titanium	Apache 2 License	Free, requires registration. Possible for commercial licensing	Eclipse based IDE
Sencha Touch	GPL v3	Free	Available for purchase
Xamarin Mono	LGPL v2	Starting from 249 USD	

While most of the solutions are free, some of them come with a cost if used commercially, as can be seen in Table 10. The costs are added on top of the native as all of the solutions are using the platform specific SDK for the binary creation. HTML5 and specifically JavaScript is the technology used as the common language which the developer should be familiar with in all of the solutions.

In the most likelihood, in case of targeting iOS and any other platform, both Mac and Windows desktop computer operating systems are needed as a developer environment, as described in the Mobile Platform Analysis section. [62]

5.1 Appcelerator Titanium

The cross platform framework from Appcelerator, Titanium Platform includes Titanium SDK and Titanium Studio. Applications using Titanium SDK are written in HTML, CSS and JavaScript, which are then interpreted via Python, Ruby and PHP. Titanium Studio is built on top of Eclipse, an open source IDE and it is based on Aptana Studio which was similar product developed by Aptana, a company that Appcelerator acquired in 2011. According to Appcelerator, Titanium can speed up to 70% of the development time when comparing to native iOS or Android development. [33]

The version evaluated in the research was 2.1.3 that was released in October 3rd 2012. First public version with Android and iOS support was released in 2009. Earlier versions were meant for cross platform development in the desktop computers. Blackberry support is according to VisionMobile Cross-platform Developer Tools 2012 report in beta, but not mentioned at the Appcelerator web site. Having said that, BlackBerry was one of the main sponsors of the Titanium focused developer event called Codestrong in late October 2012, thus there might be some truth in the beta statement.

Support for Windows 8 and Windows Phone 8 are planned to be delivered during the first half of 2013. [63] Figure 26 illustrates the Titanium platform and the related services.

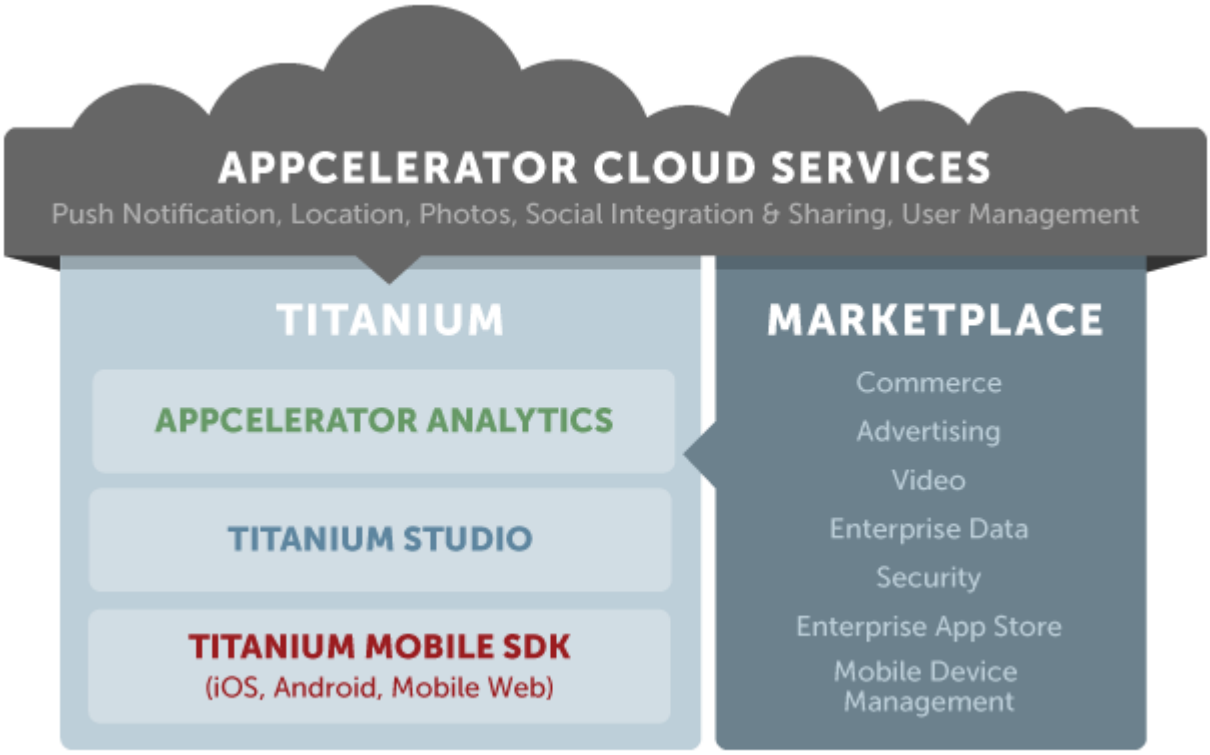


Figure 26. Titanium platform chart [64]

Titanium SDK comes with several APIs for gaming, cloud services and for native features of the given mobile platform. Many of the additional APIs have been added to the SDK via acquiring smaller companies that were delivering those services, such as a company called Cocoafish which was acquired in 2012 for gaining the cloud services. [33]

The API for accessing native features is specific to Titanium and does not necessarily conform the related W3C and WHATWG standards. Applications use JavaScript in the development phase, but once the application is compiled, the SDK turns it to bytecode that is then compiled to a native binary for the target platform by using the native SDK of that platform. The initial JavaScript code is said to be 80% cross platform compliant, thus leaving the 20% platform specific.

The final binary is said to be mostly native, thus performing as near to native implementation. The application contains independent JavaScript rendering engine and an implementation of the WebKit engine.

According to Appcelerator, more than 35000 applications built with Titanium have been published and deployed to over 40 million devices. The number of registered developers has peaked during last 12 months and reached 300 thousand developers in April 2012. While 97% of the developers target iOS and 92% Android, there is a demand for additional platforms. [33]

Appcelerator offers certification for developers in order for them to be recognised with the skills for developing with Titanium Platform. Two types of certificates exist, Titanium Certified Application Developer (TCAD) and Titanium Certified Mobile Developer (TCMD) of which the prior is required for completing the latter. Training for the certification via webinars and online courses is free and the certification exam is available online. The price for either of the certificates at the time of writing is 49.99 USD. [65]

Installing and setting up the Titanium Studio requires the Android SDK and the API level 8 plugins. It is not clear why specifically this version, as it is for Android 2.2.

5.2 Adobe PhoneGap

PhoneGap has become one of the most popular cross platform solutions since its creation 2008. It was originally developed by Nitobi, a software company, which was acquired by Adobe in the latter half of 2011. [66] Since then, the source code has been released to Apache Software Foundation in order to keep it open for the community. With the acquisition, the existing employers of Nitobi were enabled to focus solely on the development and innovation. The source code is available at https://github.com/callback

"PhoneGap is an HTML5 app platform that allows a developer to author native applications with web technologies and get access to APIs and app stores. PhoneGap leverages web technologies developers already know the best... HTML and JavaScript."
[33]

Additionally Adobe has shown already before the acquisition its commitment to HTML5 by adding PhoneGap support for their main Web Publishing tool, the Dreamweaver, onwards from version CS 5.5. [67]

Once the source was moved under Apache Software Foundation, the name of the project was changed to Callback. Soon after the name was changed again and current is Cordova. In general the framework is still known as PhoneGap. Figure 27 shows the PhoneGap application development process with cloud based build service and third party tools such as JavaScript libraries and PhoneGap plugins.

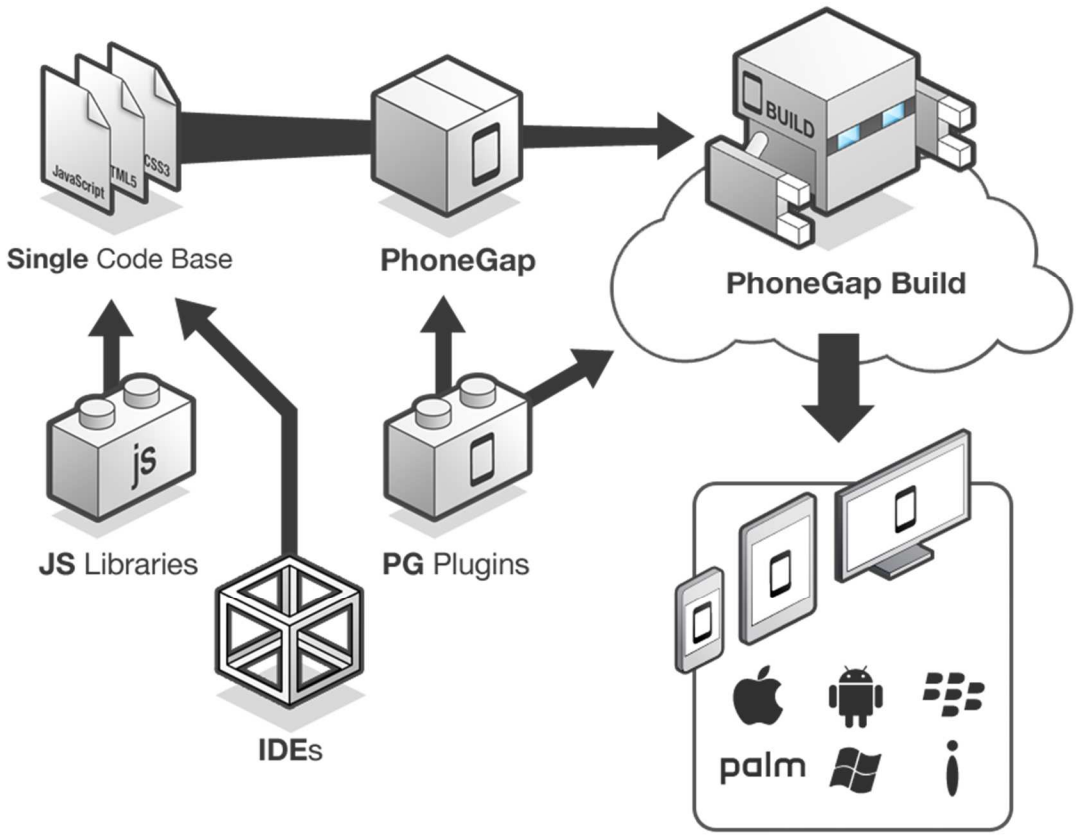


Figure 27. PhoneGap application development process [68]

PhoneGap applications are so called hybrid applications that contain the HTML5, CSS and JavaScript of which the JavaScript is linked to the given native API via PhoneGap specific API implementation. In Figure 26 the cloud service provided by Adobe, called PhoneGap build, is used for compiling to the target platforms. The platform specific

native code is compiled with the platform SDK while the HTML5 part is kept as original and rendered via platform specific WebView class. PhoneGap was the first cross platform solution to employ this kind of a hybrid approach. This might be the reason why few other less popular solutions use PhoneGap within their framework. Having said that, by using the native browser renderer the use of standard features is limited to the support given in the platform. This is considered to be the most negative downside of the hybrid approach.

The main reason for popularity is the amount of platforms supported, since it covers all major platforms. The popularity can only be expected to increase after Adobe announced to stop the development of Flash Player for mobile devices and made PhoneGap available via Flash Builder and DreamWeaver. Before that there has not been any specific IDE or other editing tools available for PhoneGap. [33]

The version evaluated in the research was 2.2.0 that was released in November 1st 2012. The first release was made in 2008 and in the last 12 months there has been a point release nearly every month. Table 11 shows the PhoneGap API support for different mobile device platforms.

Table 11. PhoneGap feature support in different mobile platforms [62:10, 68]

Feature	iPhone / iPhone 3G	iPhone 3GS and newer	Android 1.0 – 4.0	Windows Phone 7	BlackBerry 5.x–6.0+	Bada	Symbian	webOS	Tizen
Accelerometer	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Camera	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Compass	N/A	Yes	Yes	Yes	N/A	Yes	N/A	Yes	Yes
Contacts	Yes	Yes	Yes	Yes	Yes	Yes	Yes	N/A	N/A
File	Yes	Yes	Yes	Yes	Yes	N/A	N/A	N/A	N/A
Geolocation	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Media	Yes	Yes	Yes	Yes	N/A	N/A	N/A	N/A	Yes
Network	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

Notification (alert)	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Notification (sound)	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Notification (vibration)	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	N/A
Storage	Yes	Yes	Yes	Yes	Yes	N/A	Yes	Yes	Yes

Support for Windows Phone 8 has been under development already before the publication of the Windows Phone 8 SDK and is expected to land to a near future release. [69]

PhoneGap installation to the target development environment requires copying files since there is no installer for that task. Depending of the target platform, the native tools need to be configured for inclusion of the PhoneGap files. [62:20]

5.3 Sencha Touch

Sencha Touch is a hybrid application cross platform solution created by Sencha in 2007. Originally the project was combining the knowledge of three popular JavaScript libraries, ExtJS, jQTouch and Raphael. Later it evolved to more complete solution with native WebView interfaces. Therefore it is considered to a very similar solution with PhoneGap but due to the better tooling, namely with Sencha SDK Tools and Sencha Architect IDE, the development process is smoother. Sencha has 300 thousand registered developers. VisionMobile survey revealed that Sencha Touch is used to target BlackBerry devices more than average. [33]

The version evaluated this research was 2.1 that was released in 6th November 2012. The platforms supported by this version are shown in the Table 12. By the end of the year 2012, Sencha Touch is expected to have support for Windows Phone 8. [70] Support for Blackberry 10 theme is available but currently in beta state. [71] Table 12 shows the mobile device platforms with the least version supported by Sencha Touch 2.

Table 12. Mobile operating systems and their versions supported by Sencha Touch 2 [72]

Operating System	Version
Android	2.3+
BlackBerry OS	6+
BlackBerry Tablet OS	1
iOS	4+

In order to download the framework, user needs to register to the Sencha Forum. There exist two versions of the framework, open source and commercial.

"Sencha Touch is a cross-platform framework aimed at next generation, touch enabled, devices. It is currently compatible with Apple iOS 3+, Android 2.1+, and BlackBerry 6+ devices. Together these devices represent over 95% of current US mobile traffic." [33]

5.4 Xamarin Mono

Xamarin Mono is the only solution in the research which does not use HTML5 and related technologies for cross platform development. Instead it builds on top of two open source frameworks, *MonoTouch* for iOS and *Mono for Android*, and uses C# language and .NET libraries for development. This solution is based on the Mono project which is an open source project with the motivation to implement Microsoft .NET framework and C# language for other operating systems that Microsoft Windows. The popularity of Xamarin Mono is based on the existing knowledge of .NET in the enterprise developers and thus it has received the high scores in the developer surveys. [33]

Due to the nature of this solution, it is outside the HTML5 scope of this research and thus left without further evaluation.

6 Feature Availability and Performance

Several tools and libraries have been developed for browser based testing. These libraries can be used to test the feature parity and standard compliance of the mobile platform rendering engines.

The standard compliance and richness in features are the limiting factors once it comes to designing the HTML5 application. The limitations vary between platforms and before the development begins, it should be decided which are the target platforms. This research focused on the most popular platforms and tested against the feature specifications via several benchmarking tools. This gives a good idea of which features are generally available and which are found specifically from a certain platforms.

Certain features such as file access might be limited to the ones owned by the application. These limitations are usually based on the security model of the platform and cannot be overcome by native solution. Testing for these limitations are outside the scope of this research but should be taken into consideration once application development is under planning.

Most of the browser implementation used in the Mobile Platforms is based on the WebKit open source browser engine. With this the feature parity cannot be guaranteed as each implementation uses WebKit differently, thus building a different rendering engine.

The WebKit project is an open source web page rendering engine which is widely used in the desktop as well as in the mobile environment. One of the big mobile device companies, Nokia has used WebKit in all of its device platforms and has been supporting the project with engineering and financial resources. [25] Table 13 shows the rendering engines used in the major mobile device platforms.

Table 13. Mobile Browser rendering engines [62]

Mobile OS	Browser
Android	Webkit-based
iPhone	Webkit-based

BlackBerry 6.0 +	Webkit-based
Windows Phone 7	IE 7-based *
WebOS	Webkit-based
Nokia	Webkit-based
BADA	Webkit-based

PhoneGap uses these modern browser rendering engines as the platform for building HTML5 based applications by embedding them within the native wrapper application.

Opposing to the information in Table 13, Microsoft evangelised that their browser used in Windows Phone 7 matches the desktop version of Internet Explorer 9, which in turn was released at the same time with Windows Phone 7. A similar approach was made in Windows Phone 8, which uses Internet Explorer 10. [56]

Testing against HTML5 standards and the ability of accessing device data via JavaScript are the main tasks for measuring the maturity of HTML5 support in the given platform. It should also be evaluated how easily the look and feel of the given platform can be achieved via HTML5.

Mobile devices have rather limited resources when it comes to CPU, memory and other hardware related to computing performance. That is the main reason why mobile device platforms lack a real multitasking operation model. Instead they are using application states which can but the application to idle while it is not directly used by the user. In the last quarter of 2011, most of the published smart phones have a CPU capable to run 1 GHz or faster. Also the amount of cores are increasing which enables better multitasking.

This section discusses the different methods and tools for feature testing and standards compliance. In the end of the section there are results of the selected tests.

6.1 Feature Detection

Historically web development has concentrated in evaluating which browser the client is using and made assumptions based on that. Opposing method of browser detection is feature detection.

Since the market has completely changed from the time when Internet Explorer was the dominant browser and sites were built specifically for it, the need for different approach became relevant. The feature detection method has been suggested to be used in web development for several years now and with mobile applications the case is not different.

Feature detection is based on the idea to test against a single feature at a time. For example if the application is to use a specific feature which is known to be unimplemented everywhere, then it needs to checked whether that feature is available in the current environment. The JavaScript code shown in Figure 28 demonstrates how the feature detection is used for detecting the availability of the application cache feature.

```
1  if (typeof applicationCache !== 'undefined') {
2      // Feature found
3  } else {
4      // Feature not found
5  }
```

Figure 28. Feature detection method in JavaScript

Feature detection does not guarantee that the feature in question is actually compliant with any standard or would have the properties or methods available. Feature detection is good for detecting the presence of a possible feature but for checking the correct implementation additional testing is required. However according to the collected feature detection libraries not so many browsers expose any misleading object names. [62, 73]

6.2 Benchmarking Tools

All major browser manufacturers have published tools for testing against standards compliance as well as processing speed.

Each platform has certain API with programming language specific classes for displaying HTML5 content. Since every platform comes in several different versions due to updates, those are taken into consideration here.

Some of test can be run by just visiting the web site, where others require implementing a minimal test application. The Table 14 lists the test solutions evaluated in the research. Additionally there exist several tests that are not included in the list because of their similarity with other more popular tests, their development has stopped or they are considered deprecated by their creators.

Table 14. Comparison table of the several testing tools that are available online

Name	URL	Notes
Acid3	http://acid3.acidtests.org/	Web Standards test from early 2008. Detailed explanation available at http://www.webstandards.org/action/acid3/
Are We Playing Yet?	http://areweplayingyet.org/	This is an open and public initiative to bring more harmony into HTML5 Audio implementations
Benchmark.js	http://benchmarkjs.com/	A robust benchmarking library that works on nearly all JavaScript platforms, supports high-resolution timers, and returns statistically significant results. Source available at https://github.com/bestiejs/benchmark.js

Browser-scope	http://www.browserscope.org/	Browserscope is a community-driven project for profiling web browsers. The goals are to foster innovation by tracking browser functionality and to be a resource for web developers.
Can I use...	http://caniuse.com/#agents=mobile	Compatibility tables for support of HTML5, CSS3, SVG and more in mobile browsers.
CSS selectors test	http://tools.css3.info/selectors-test/test.html	After starting the test-suite it will automatically run a large number of small tests which will determine if your browser is compatible with a large number of CSS selectors. Runs total of 574 tests.
Dromaeo	http://dromaeo.com/	Mozilla JavaScript and DOM tests created by John Resig. Source available at https://github.com/jeresig/dromaeo
ECMAScript Language test262	http://test262.ecmascript.org/	test262 is a test suite intended to check agreement between JavaScript implementations and ECMA-262, the ECMAScript Language Specification (currently 5.1 Edition). The test suite contains thousands of individual tests, each of which tests some specific requirements of the ECMAScript Language Specification. Runs 11571 tests.
haz.io	http://haz.io/	Uses Modernizr to give visually better looking feedback of the features supported in the current browser
jsPerf	http://jsperf.com/browse	Collection of different JavaScript performance tests

Kraken	http://krakenbenchmark.mozilla.org/	JavaScript benchmark by Mozilla
Mobile HTML5	http://mobilehtml5.org/	Trying to understand HTML5 compatibility on mobile and tablet browsers. Shows API detection statistics for browsers available in the mobile space
Modernizr	http://modernizr.github.com/Modernizr/test/	Feature detection library for features that are new in HTML5 and CSS3. Runs over 40 tests. Source available at https://github.com/Modernizr/Modernizr
Octane	https://developers.google.com/octane/	Made by Google. The JavaScript Benchmark Suite for the Modern Web. Source code available http://code.google.com/p/octane-benchmark/
Peacekeeper	http://peacekeeper.futuremark.com	Universal browser performance test
Ringmark	http://www.rng.io/	The Ringmark test suite has been developed by Facebook and Bocoup. One of the tests used in Browserscope. Recently open sourced, available at https://github.com/coremob/coremob-tests/
RoboHornet	http://www.robohornet.org/	RoboHornet is a benchmark designed around performance pain point real web developers care about. Source available at https://github.com/robohornet/robohornet

SunSpider	http://www.webkit.org/perf/sunspider/sunspider.html	This benchmark tests the core JavaScript language only, not the DOM or other browser APIs.
The CCS3 Test	http://css3test.com/	Runs 944 tests to check whether any of the 221 features are available
The HTML5 Test	http://html5test.com/	The HTML5 test score is an indication of how well your browser supports the upcoming HTML5 standard and related specifications
V8 Benchmark Suite	http://v8.googlecode.com/svn/data/benchmarks/current/revisions.html	Created by Google V8 JavaScript Engine team.

The tests listed in Table 14 were evaluated and the following tests were selected due to their approach for testing certain features. Many of the tests are using same or very similar methods and thus overlapping.

The selection of suitable test solutions for this research was made based on the availability of the sources, testing procedure and popularity. The tests that replicated each other were dropped. The popularity was evaluated by the number of public test results and major publisher blog posts.

The speed related tests were not tested with the software simulators, since the environment of a real target device usually has lower resources and the tests would give misleading results. The tests related to features and standard parity were run in the software simulators and the results are available in the end of this section.

The test selection was limited to the following solutions, all except Ringmark are mainly for speed related performance tests:

- Dromaeo, includes SunSpider and V8
- Kraken
- Octane, includes V8
- Ringmark
- RoboHornet
- SunSpider

Further performance testing could measure and calculate the power used by certain features, as suggested by the proceeding "Who Killed My Battery: Analyzing Mobile Browser Energy Consumption" at Mobile Web Performance event. [74] In the scope of this research, only software based testing is evaluated.

6.2.1 Dromaeo

Dromaeo consists of a few different test suites. JavaScript performance suite is designed to allow real-world application testing with binary tree traversal, string and array processing, and prime number computation. [1] The JavaScript tests include other open source tests such as SunSpider and V8 tests.

Another test suite is for benchmarking DOM traversal and manipulation. These types of tests are crucial for any application that interacts with the DOM. Due to certain implementation bugs in Internet Explorer, these tests will not run in versions 6, 7 and 8.

Test suites are available at http://dromaeo.com/ and the source codes are available at https://github.com/jeresig/dromaeo.

6.2.2 Kraken

Kraken is a JavaScript performance test suite made by Mozilla.

Google Chrome team is providing an updated source version of the Kraken tests at http://kraken-mirror.googlecode.com/svn/trunk/kraken/hosted/index.html. According to them, the online Mozilla version at http://krakenbenchmark.mozilla.org/ does not reflect the latest changes made by Mozilla. [75]

6.2.3 Octane [76]

Octane is a JavaScript Benchmark Suite for the Modern Web created by Google. It is based on the V8 Benchmark suite created also by Google, with additional tests that are taken from other open source projects.

The 3rd party tests were added since the team developing this solution believes that a high performance JavaScript rendering engine should be able to perform well on real world code, besides the usual benchmarking tools.

Octane is available for download at http://code.google.com/p/octane-benchmark/source/checkout and can be run directly via browser at https://developers.google.com/octane/.

6.2.4 Ringmark

Ringmark was initially developed by Facebook and Bocoup, and later by W3C CoreMob group. CoreMob stands for "Core Mobile Web Platform Community Group" which is a group organised within W3C with the goal to accelerate the development of modern mobile web applications in HTML5 related solutions. The group currently has 279 members and has several major companies involved, such as Mozilla, Qualcomm, ST-Ericsson, Google and jQuery. The reason for making W3C as the parent for this solution is to help W3C figure out how they could use Ringmark for improving their standard process. [77]

The core concept behind Ringmark is based around the question of "what do developers need?". The tests are grouped in three sets, called rings. Each ring represents certain set of features that developers would need for certain types of applications. The rings can be considered as a software versioning. Ring 0 represents the base functionality that most mobile platforms have today. As reference platforms, Ring 0 represents the feature level of iOS 5.0 and Android 2.2, which were the dominant platforms when Ringmark was released in early 2012. [78]

Ring 1 represents what functionality is needed to create applications using features such as camera access. It is primarily designed to test against features that could be used for developing cross platform consistent applications. One of such features prefix free CSS3 properties.

Ring 2 is still work in progress and is planned to include features such as WebRTC and WebGL once they become more mature. It should be noted that the next ring tests will not be run in case the earlier ring has even a single partially failing test. The reason for this is to avoid any fragmentation in the test results.

The tests are built by going deeper than just simple feature detection method, thus each API requires very specific tests. Currently Ringmark contains more than 400 tests in order to fully test all the aspects of the given feature or API. Many of the tests are using methodology from other solutions such as Modernizr, caniuse.com and W3C tests. The APIs and features tested in Ring 0 and 1 are listed in Figure 28.

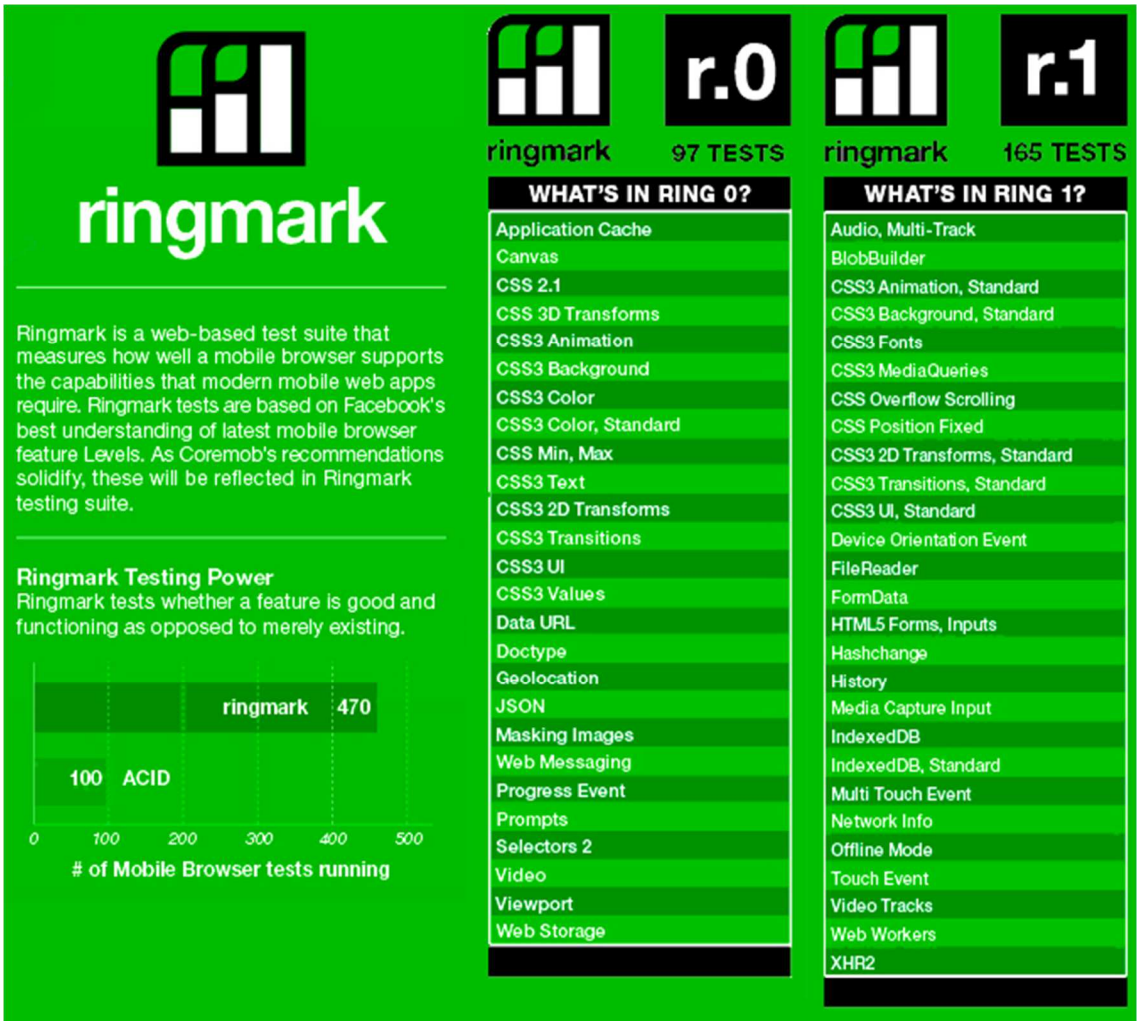


Figure 29. APIs and features used in Rings 0 and 1 of Ringmark [79]

The structure is made future proof as more rings can be added once more features, specifications and APIs mature to a certain level. The performance is essentially measured in drawing speed. Ring 1 requires the rendering engine to draw and animate 50 sprites at the speed of 30 frames per second (FPS). Similarly Ring 2 requires 100 sprites at 30 FPS. More performance related tests are under development and not included in the research. [80]

The source is available at https://github.com/coremob/coremob-tests/, while the current release of Ringmark can be accessed and run via http://rng.io. Ringmark is also integrated to the Browserscope community driven test collection available at http://www.browserscope.org/ringmark/test. The statistics at Browserscope are automatically collected of the test ran at http://rng.io. [81]

6.2.5 RoboHornet

RoboHornet is a set of benchmarks created by open source community. Source code for the project is available at https://github.com/robohornet/robohornet where any developer can contribute. The contributors of the project are developers distinguished by their achievements in other web related open source projects. The project is designed around performance pain points which the community considers important. [82]

The goal of the project is to gather the best benchmarking tests for helping the browser vendors to build better implementation of the features needed by real web applications. The tests are not limited to JavaScript but to all aspects that a web application might touch. Individual tests are given votes by the developer community in order to have more weight on the ones that matter the most.

RoboHornet has one design flaw, it requires allowing pop-up windows which are disabled by default in most modern platforms and web browsers. During the application tests, Robohornet got stuck several times and did not complete all the tests in any of the evaluated platforms.

6.2.6 SunSpider [1]

SunSpider is a widely used JavaScript performance suite created by WebKit open source project. It focuses only to test JavaScript functionality, in terms of the language structure and mathematical algorithms. However there are no Document Object Model (DOM) related tests. The current version from WebKit is available at http://www.webkit.org/perf/sunspider/sunspider.html

Google Chrome team made few improvements to the tests and released the source, available at http://sunspider-mod.googlecode.com/svn/data/hosted/sunspider.html. [75]

6.3 Test Results

Ringmark was the only test out of the selection which checks for features and against standard specifications. Other tests did include very limited feature checking while focusing on the implementation performance. Since the tests were run in the software simulators, the speed related results are most likely to give wrong results due to additional resources of the hardware environment. The results are listed in Table 15. For comparison The HTML5 Test and The CSS3 Test results are included.

The tests were run in a hybrid application, which uses the HTML5 related solution for the platform, as evaluated in the Chapter 3. Table 15 shows the results for software simulators that were tested against three tests.

Table 15. test suite results for software simulators

	The HTML5 Test	The CSS3 Test	Ringmark
Android 4.2	298 + 3 bonus	64% (369/937)	Ring 0: all 101 passed. Ring 1: 90 passed, 33 failed
BlackBerry 10	484 + 11 bonus	63% (558/937)	Ring 0: all 101 passed. Ring 1: 147 passed, 3 failed (audio, sprite and Web Workers performance)

iOS 6	386 + 9 bonus	55% (500/937)	Ring 0: all 101 passed. Ring 1: 141 passed, 9 failed
Windows Phone 8	319 + 6 bonus	54% (444/937)	Ring 0: 100 passed, one failed (prompt support)

While The HTML5 Test and The CSS3 Test only use the feature detection method, they are often referred to in the news when platform updates are becoming available. They can give a fairly good sense of direction and as the results in Table 15 point out, BlackBerry 10 is leading the competition with the highest points.

7 Results and Analysis

In order to answer the question of using HTML5 as a user interface (UI) for major mobile platforms, the selection of platforms was made, the tools evaluated and the test procedure defined. Depending on the application type and what the application needs to achieve, in most cases the HTML5 seems to be the UI of choice when it comes to cross platform development.

Each platform has different native programming languages, architecture and libraries for binding the UI and the backend together. In all except Windows Phone this is done via a WebKit implementation that is the core for the native web browser and the rendering engine for hybrid HTML5 applications. However each implementation of the WebKit core differs and thus creates feature support fragmentation.

In case all of the four selected platforms are to be targeted and the UI is built in HTML5, CSS3 and JavaScript, the choice of cross platform solution would be PhoneGap. Cross platform development with JavaScript is limited to the platform specific API, but with cross platform solutions, the API is unified. This includes BlackBerry 10 which has the most direct approach to HTML5 application development via its WebWorks solution. It has, however, its own API, quite as the other platforms, but is supported by PhoneGap.

The feature support of the given platform can be tested with several benchmarking tools, which were evaluated in this research. Today the most reliable solution for testing these features is Ringmark. Other benchmarking solutions mainly test the rendering speed of various features and JavaScript methods. The results of those tests vary and are partly dependant on the available hardware resources. Therefore any speed related tests should be done within a real device corresponding to the majority of the devices in the market.

Cross platform solutions can narrow the gap between various platforms, but can only come as far as the commonly supported features. PhoneGap has the widest support for different mobile device platforms and is the recommended solution to be used if the application development is done with HTML5 and its related standards. PhoneGap of-

fers a single API that significantly reduces the amount of JavaScript code needed for cross platform development. Without a cross platform solution the amount of JavaScript code increases with the number of targeted platforms. HTML5 markup and CSS3 styling works well on different platforms as long as the features used are supported.

BlackBerry 10 received the best results for its HTML5 and related standards support. It is also the latest platform and at the time of the research, still unavailable at the market. The market is dominated by Apple iOS and Google Android, which both offer very decent support for HTML5 based hybrid applications.

8 Discussion and Conclusions

HTML5 is a markup standard that has been developing for over a decade. In general when talking about HTML5 also CSS3 and JavaScript is included, so it is not only about markup, it is also about interaction. Using HTML5 and related technologies makes sense in case there is more than one platform to be implemented for the given application. The support for HTML5 related APIs is fragmented, thus the application in question needs to be evaluated by the developer to decide whether the HTML5 is the correct approach.

The aim of the research was to clarify and give sufficient information for making decisions on the strategy of cross platform mobile application development based on HTML5 and the related technologies. The rendering engines are maturing quickly and new features are being implemented rapidly. The amount of native code needed depends on the application needs case by case.

The four most popular mobile device platforms were selected based on their market share and developer interest. All these four platforms enable native and hybrid HTML5 application development with their own APIs. In the case of the hybrid development, native tools and SDK are required. In the case of Apple iOS, the development can only be done in Mac and with Microsoft Windows Phone 8, Windows 8 64 bit desktop operating system. With Google Android and RIM BlackBerry the environment is not this limited, while it can be done in Mac, Linux and Windows.

Cross platform application development with HTML5 is possible in all selected mobile device platforms. Due to the API fragmentation in device specific JavaScript implementation, the most cost efficient to do this is via a cross platform solution such as PhoneGap, which at the moment has the widest support for the most popular mobile device platforms.

The research evaluated the benchmarking tools for testing HTML5 based applications, the standard completeness in the web browser renderers used to render the HTML5 based applications. The available and functional features of the HTML5 and related standards in the rendering engine limit the cross platform portability when the devel-

oper or the designer is using features not in the main stream. These limitations needs to be considered once the application is being designed and the target platforms have been selected. One of the most reliable solutions is the Ringmark feature benchmarking tool.

Further investigation related to HTML5 application development could be done by using real devices that have hardware similar to the ones most popular. This would give realistic figures for speed related performance tests and other resource dependant tests. The power consumption of the device should also be taken into account.

The present research only evaluated many of the speed related benchmarking tools. Further investigation on specific HTML5 related features should be done on real device hardware that is matching the majority of the market. With the similar hardware resources and isolated testing, the results would give a clear picture of how well and how poorly certain features are implemented between the mobile platforms. Once the speed can be compared to the energy consumption, true performance metrics can be made available and possibly standardised for device manufacturers to use in their marketing.

References

1 Pro JavaScript RIA Techniques: Best Practices, Performance, and Presentation
Den Odell
ISBN-13: 978-1-4302-1935-4, 2009, Apress

2 TargetProcess (2011), *TargetProcess: Infographics* (February 2011) http://www.targetprocess.com/blog/2011/02/targetprocess-inforgraphics.html (Accessed 8th November 2011)

3 Pro HTML5 Programming, 2nd Edition
Peter Lubbers, Brian Albers and Frank Salim
ISBN 13: 978-1-4302-3864-5, 2011, Apress

4 Sergey's HTML5 & CSS3: Quick Reference. HTML5, CSS3 and APIs. 2nd Edition
Sergey Mavrody
ISBN-13: 978-0-9833-8672-8, 2012, Belisso

5 W3C (2012), *About W3C*, http://www.w3.org/Consortium (Accessed 6th November 2012)

6 WHATWG (2012), *FAQ - What is the WHATWG?* (October 2012) http://wiki.whatwg.org/wiki/FAQ#What_is_the_WHATWG.3F (Accessed 6th November 2012)

7 IETF, *About the IETF* https://www.ietf.org/about/ (Accessed 6th November 2012)

8 Telerik (2012), *KendoUI - HTML5 Adoption Survey* (November 2012) http://www.kendoui.com/surveys/html5-adoption-survey-2012.aspx (Accessed 8th November 2012)

9 W3C (2011), *W3C Math Home* (June 2011) http://www.w3.org/Math/ (Accessed 9th November 2012)

10 IETF (1996), *Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types* (November 1996), http://www.ietf.org/rfc/rfc2046.txt (Accessed 9th November 2012)

11 Appcelerator (2012), *Native vs. HTML5 Mobile App Development: Which option is best?*, (August 2012) http://pages.appcelerator.com/html5whitepaper.html (Accessed 5th October 2012)

12 Salesforce.com, *Native, HTML5, or Hybrid: Understanding Your Mobile Application Development Options* http://wiki.developerforce.com/page/Native,_HTML5,_or_Hybrid:_Understanding_Your_Mobile_Application_Development_Options (Accessed 5th October 2012)

13 Appcelerator (2012), *Appcelerator Webinar with Michael King: The Great Debate : Native vs. HTML5* (September 2012)

14 W3C (2009), *XHTML 2 Working Group Expected to Stop Work End of 2009, W3C to Increase Resources on HTML 5* (July 2009) http://www.w3.org/News/2009#entry-6601 (Accessed 12th October 2012)

15 Introducing HTML5, Second Edition
Bruce Lawson and Remy Sharp
ISBN 13: 978-0-321-78442-1, 2012, New Riders

16 VisionMobile (2012). *Mobile Megatrends Webinar,* (May 2012) http://www.visionmobile.com/blog/2012/05/report-mobile-megatrends-2012/ (Accessed 12th May 2012)

17 VisionMobile (2011). *HTML5 and what it means for the mobile industry,* (June 2011) http://www.visionmobile.com/product/html5-and-what-it-means-for-the-mobile-industry/ (Accessed 18th July 2011)

18 VisionMobile (2011), *Clash of Ecosystems*, (November 2011) http://www.visionmobile.com/rsc/researchreports/VisionMobile-Clash-of-Ecosystems_v1.pdf (Accessed 8th November 2012)

19 W3C (2012), *HTML Working Group draft plan* (September 2012) http://dev.w3.org/html5/decision-policy/html5-2014-plan.html (Accessed 6th October 2012)

20 WHATWG (2011), *HTML Living standard – Introduction* (November 2011), http://www.whatwg.org/specs/web-apps/current-work/multipage/introduction.html#is-this-html5 (Accessed 11th November 2011)

21 Mobile Business Briefing (2012*), Facebook focuses on mobile web initiatives*, (27 Feb 2012 http://www.mobilebusinessbriefing.com/articles/facebook-focuses-on-mobile-web-initiatives/22560 (Accessed 27th February 2012)

22 Mike Chambers (2011), *Clarifications on Flash Player for Mobile Browsers, the Flash Platform, and the Future of Flash,* (November 2011) http://www.mikechambers.com/blog/2011/11/11/clarifications-on-flash-player-for-mobile-browsers-the-flash-platform-and-the-future-of-flash/ (Accessed 12th November 2011)

23 Adobe (2011), *Flash to Focus on PC Browsing and Mobile Apps; Adobe to More Aggressively Contribute to HTML5* (November 2011) http://blogs.adobe.com/conversations/2011/11/flash-focus.html (Accessed 9th November 2011)

24 Pro iOS Web Design and Development: HTML5, CSS3, and JavaScript with Safari
Andrea Picchi
ISBN 13: 978-1-4302-3247-6, 2011, Apress

25 Beginning Nokia Apps Development: Qt and HTML5 for Symbian and MeeGo
Ray Rischpater and Daniel Zucker
ISBN 13: 978-1-4302-3179-0, 2010, Apress

26 Can I use... (2012), *Compatibility tables for support of HTML5, CSS3, SVG and more in desktop and mobile browsers – CSS Grid Layout* (November 2012) http://caniuse.com/#feat=css-grid (Accessed 3rd November 2012)

27 Brendan Eich (2012), *The State of JavaScript - A very brief history* http://brendaneich.github.com/Strange-Loop-2012/#/1 (Accessed 9th October 2012)

28 BuiltWith.com (2012), *JavaScript usage statistics* (October 2012) http://trends.builtwith.com/javascript (Accessed 18th October 2012)

29 Wingo (2011), *JavaScriptCore, the WebKit JS implementation* (November 2011) http://wingolog.org/archives/2011/10/28/javascriptcore-the-webkit-js-implementation (Accessed 14th October 2012)

30 Mozilla (2011) *New in JavaScript 1.8.5* (October 2011) https://developer.mozilla.org/en-US/docs/JavaScript/New_in_JavaScript/1.8.5 (Accessed 11th November 2011)

31 YUI Theather (2008), *YUI Theater: Douglas Crockford, "Web Forward"* (October 2008) http://www.yuiblog.com/blog/2008/10/17/video-crockford-webforward/ (Accessed 25th November 2008)

32 u-double-u (2012), *SPEED-BATTLE statistics and browser comparison* (October 2012) http://www.speed-battle.com/statistics_e.php (Accessed 17th October 2012)

33 VisionMobile (2012). *Cross-Platform Developer Tools 2012,* (February 2012) http://www.visionmobile.com/product/cross-platform-developer-tools-2012/ (Accessed 1st March 2012)

34 Appcelerator / IDC (2012), *Q3 2012 Mobile Developer Report* (October 2012) http://pages.appcelerator.com/Q32012AppceleratorIDCSurveyReport.html (Accessed 5th October 2012)

35 Reuters (2012), *In a Samsung Galaxy far, far away ... will Android still rule?*, (May 2012) http://www.reuters.com/article/2012/05/03/us-samsung-android-idUSBRE8420BU20120503 (Accessed 8th October 2012)

36 StatCounter (2012), *StatCounter Global Statistics* (October 2012) http://gs.statcounter.com/#mobile_os-ww-monthly-200812-201210 (Accessed 25th October 2012)

37 VisionMobile (2012). *Developer Economics 2012,* (June 2012) http://www.visionmobile.com/product/developer-economics-2012/ (Accessed 6st October 2012)

38 TechCrunch (2011), *Windows Phone Marketplace: One Year In*, (November 2011) http://techcrunch.com/2011/11/22/windows-phone-marketplace-one-year-in/ (Accessed in February 8th 2012)

39 VisionMobile (2012), *[Infographic] The Mobile Industry in Numbers* (October 2012) http://www.visionmobile.com/blog/2012/10/infographic-the-mobile-industry-in-numbers/ (Accessed 22nd October 2012)

40 Gartner (2011), *Gartner Says Sales of Mobile Devices Grew 5.6 Percent in Third Quarter of 2011; Smartphone Sales Increased 42 Percent* (November 2011), http://www.gartner.com/it/page.jsp?id=1848514 (Accessed 15th November 2011)

41 Enough Software (2012). *Mobile Developer's Guide To The Galaxy, 11th edition*, (October 2012) http://www.enough.de/fileadmin/uploads/dev_guide_pdfs/Guide_11thEdition_WEB-1.pdf (Accessed 5th October 2012)

42 Microsoft (2012), *Download Center - Windows Phone SDK 8.0* (November 2012) http://www.microsoft.com/en-us/download/details.aspx?id=35471 (Accessed 8th November 2012)

43 Creating iOS 5 Apps: Develop and Design
Richard Warren
ISBN 13: 978-0-321-76960-2, 2012, Peachpit Press

44 Apple (2011), *iOS Dev Center* (December 2011) https://developer.apple.com/devcenter/ios/index.action (Accessed 10th December 2011)

45 Apple (2012), *iOS Technology Overview: About the iOS Technologies* (September 2012) https://developer.apple.com/library/ios/#documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/Introduction/Introduction.html (Accessed 5th November 2012)

46 Apple (2012), *WWDC 2012 Keynote* (June 2012) https://developer.apple.com/videos/wwdc/2012/ (Accessed 29th October 2012)

47 Apple (2012), *Apple Special Event October 2012* (October 2012) http://www.apple.com/apple-events/october-2012/ (Accessed 30th October 2012)

48 Google (2012), *Bashboards – Platform Versions* (October 2012) http://developer.android.com/about/dashboards/index.html (Accessed 30th October 2012)

49 Learn HTML5 and JavaScript for Android
Gavin Williams
ISBN 13: 978-1-4302-4348-9, 2012, Apress

50 Google (2012), *WebView | Android Developers* (November 2012) http://developer.android.com/reference/android/webkit/WebView.html (Accessed 19th November 2012)

51 Pro Android 4
Satya Komatineni and Dave MacLean
ISBN 13: 978-1-4302-3931-4, 2012, Apress

52 Microsoft (2012), *Windows Phone 8 developer platform highlights* (November 2012) http://blogs.windows.com/windows_phone/b/wpdev/archive/2012/11/05/windows-phone-8-developer-platform-highlights.aspx (Accessed 7th November 2012)

53 Microsoft (2012), *WebBrowser Class* (November 2012) http://msdn.microsoft.com/en-us/library/windowsphone/develop/microsoft.phone.controls.webbrowser%28v=vs.105%29.aspx (Accessed 12th November 2012)

54 Windows Phone 7 Developer Guide: Building connected mobile applications with Microsoft Silverlight
Dominic Betts, Federico Boerr, Scott Densmore, Jose Gallardo Salazar and Alex Homer
ISBN: 978-0-7356-5609-3, 2010, Microsoft Press

55 Microsoft (2012), *Build 2012 - Windows Phone 8: XAML Application Development* (October 2012) http://channel9.msdn.com/Events/Build/2012/2-021 (Accessed 1st November 2012)

56 Microsoft (2012), *Build 2012 - Windows Phone 8: Application Model* (October 2012) http://channel9.msdn.com/Events/Build/2012/2-013 (Accessed 1st November 2012)

57 Microsoft (2012), *Multi-resolution apps for Windows Phone 8* (Novermber 2012) http://msdn.microsoft.com/en-us/library/windowsphone/develop/jj206974%28v=vs.105%29.aspx (Accessed 12th November 2012)

58 RIM (2012), *BlackBerry 10 Launch Event to be Held on January 30th 2013* (November 2012) http://press.rim.com/newsroom/press/2012/blackberry-10-launch-event-to-be-held-on-january-30th-2013.html (Accessed 14th November 2012)

59 RIM (2012), *One More Step Towards BlackBerry 10 – BlackBerry 10 Gold SDKs* (November 2012) http://devblog.blackberry.com/2012/11/blackberry-10-gold-sdks/ (Accessed 14th November 2012)

60 RIM (2012), *BlackBerry Developer - Platform Choice* (October 2012) https://developer.blackberry.com/develop/platform_choice/index.html (Accessed 15th October 2012)

61 RIM (2012), *BlackBerry 10 Jam session - Building HTML5 apps with native capabilities* (August 2012) http://www.blackberryjamworldtour.com/presentations (Accessed 25th October 2012)

62 Beginning PhoneGap: Mobile Web Framework for JavaScript and HTML5
Rohit Ghatol and Yogesh Patel
ISBN 13: 978-1-4302-3904-8, 2012, Apress

63 Microsoft (2012), *New tools for Windows Phone 8 save developers time and money* (October 2012)
http://blogs.windows.com/windows_phone/b/wpdev/archive/2012/10/30/new-tools-for-windows-phone-8-save-developers-time-and-money.aspx (Accessed 30th October 2012)

64 Appcelerator (2012), *Mobile Platform Overview*
http://www.appcelerator.com/platform (Accessed 10th October 2012)

65 Appcelerator (2012), *Appcelerator Empowers the Mobile Developer Community With Even More Choice and Flexibility* (October 2012)
http://thinkmobile.appcelerator.com/press-releases/bid/236017/Appcelerator-Empowers-the-Mobile-Developer-Community-With-Even-More-Choice-and-Flexibility (Accessed 30th October 2012)

66 Andre Charland (2011), *Nitobi enters into Acquisition Agreement with Adobe* (October 2011) http://phonegap.com/2011/10/03/nitobi-enters-into-acquisition-agreement-with-adobe (Accessed 5th March 2012)

67 Andre Charland (2011), *Adobe Dreamweaver 5.5 Supports PhoneGap* (April 2011) http://phonegap.com/2011/04/12/adobe-dreamweaver-5-5-supports-phonegap/ (Accessed 5th March 2012)

68 PhoneGap (2012), *About the Project* (September 2012) http://www.phonegap.com/about/ (Accessed 15th October 2012)

69 Microsoft (2012), *Added support for Windows Phone 8 in Apache Cordova, Sencha Touch, Cocos2D, Ogre3D and other open source frameworks* (October 2012) http://blogs.msdn.com/b/interoperability/archive/2012/10/30/added-support-for-wp8-in-apache-cordova-sencha-touch-and-other-open-source-frameworks.aspx (Accessed 30th October 2012)

70 Sencha (2102), *Sencha Touch with Windows Phone 8* (October 2012) http://www.sencha.com/blog/sencha-touch-with-windows-phone-8 (Accessed 1st November 2012)

71 Sencha (2102), *Sencha Touch 2.1 is Here with New Charting & Tools* (November 2012) http://www.sencha.com/blog/introducing-sencha-touch-2-1-and-more/ (Accessed 9th November 2012)

72 Sencha (2102), *Sencha Touch Build Mobile Web Apps with HTML5* (2012) http://www.sencha.com/products/touch/features (Accessed 7th November 2012)

73 Mobile JavaScript Application Development
Adrian Kosmaczewski
ISBN 13: 978-1-449-32785-9, 2012, O'Reilly Media

74 Who Killed My Battery: Analyzing Mobile Browser Energy Consumption
Narendran Thiagarajan, Gaurav Aggarwal, Angela Nicoara, Dan Boneh, Jatinder Singh
WWW 2012 – Session: Mobile Web Performance April 16–20, 2012, Lyon, France

75 Google (2011), *Updating JavaScript Benchmarks for Modern Browsers* (May 2011) http://blog.chromium.org/2011/05/updating-javascript-benchmarks-for.html (Accessed 15th October 2012)

76 Google (2012), *The Benchmark* (August 2012) https://developers.google.com/octane/benchmark (Accessed 11th October 2012)

77 Facebook (2012), *Announcing Ringmark, a Mobile Browser Test Suite* (February 2012) https://developers.facebook.com/html5/blog/post/2012/02/27/announcing-ringmark--a-mobile-browser-test-suite/ (Accessed 3rd November 2012)

78 Facebook (2012), *The Methodology Behind Ringmark* (April 2012) https://developers.facebook.com/html5/blog/post/2012/04/04/the-methodology-behind-ringmark/ (Accessed 3rd November 2012)

79 Visual.ly (2012), *IDC Next Level of Mobile Web* (May 2012) http://visual.ly/idc-next-level-mobile-web (Accessed 20th October 2012)

80 Facebook (2012), *Ringmark Performance Tests and Coremob Meeting* (June 2012) https://developers.facebook.com/html5/blog/post/2012/06/14/ringmark-performance-tests-and-coremob-meeting/ (Accessed 3rd November 2012)

81 Facebook (2012), *W3C and Ringmark Updates* (May 2012) https://developers.facebook.com/html5/blog/post/2012/05/01/w3c-and-ringmark-updates/ (Accessed 3rd November 2012)

82 RoboHornet (2012), *RoboHornet Alpha* (October 2012) http://www.robohornet.org/ (Accessed 15th October 2012)