

**KEMI-TORNIO UNIVERSITY OF APPLIED SCIENCES  
TECHNOLOGY**

Ogunlolu O. Isaac

**HTML5, the future of mobile applications:**

A comparison between HTML5 application development platforms and native  
platforms

The Bachelor's Thesis of degree programme in Information Technology  
Kemi 2012

## PREFACE

This piece of work wouldn't have been a reality without the help of the almighty God who I am indebted to for his goodness, unending grace, favor and strength which served as the *sine qua non* of the success of this thesis project and helped me through the course of my studies. My gratitude is made of speechless compliments which can simply be expressed by saying “**Thank you God**”.

My unreserved thanks go to the thesis commissioner Antti-Pekka Aaltonen, Chief Technical Officer (CTO) of IMSS Oy for giving me the opportunity to write this thesis for a company of such importance and for the flexibility and patience shown during the course to this thesis work. I would also like to thank the extremely brilliant Thai Bui for impeccably supervising this thesis work, for guiding me and for his wise words during the course of my studies.

I extend my profound gratitude to all the teachers and staff-members of Kemi-Tornio University of Applied Sciences for their support throughout the course of my studies. I however specially thank Soili Mäkimurto-Koivumaa, Osmo Kurola, Martta Ruottu, Teppo Aaltto and Thai Bui for the role they all played in the success of this thesis work and that of my studies in general.

I would like to acknowledge Matti Juopperi, Matti & Leena Aha and Pekka & Laina Aläjärvi for their generosity and hospitality towards me right from the start till this very moment, thank you so much and God bless you. I appreciate my friends who have shown me love, affection and emotional support in this cold and sometimes acrimonious environment. Most especially the unique Uchenna Unanka who right from the very start has been sort of a guiding hand and mentor to me, words can't express how grateful I am to you and I sincerely hope our friendship knows no end.

To my lovely family, words can't express how grateful I am to you all. From the start you have served as a reliable source of motivation, encouraging me to push myself from zero to apex in order to succeed. Your prayers, emotional support and love have made this a possibility and I therefore dedicate this thesis to you all. I reserve special thanks to my parents Mr. & Mrs. Ogunlolu, your hustle made these wonderful experiences a possibility and I am sincerely grateful for your efforts, I hope and believe they are not in vain. I love and appreciate you all.

Finally, I would like to extend my gratitude to the stunningly beautiful Laura Häkkinen for her motivation, emotional support, affection, patience and love shown to me during the course of my thesis process and that of my studies in general. What can I say? You have done so much in so little time and I would probably be a better person if I had met you earlier. The past year and a half has been the best days of my life, I love you and I look forward to better days ahead with you.

April 2012

## ABSTRACT

Kemi-Tornio University of Applied Sciences, Technology	
Degree Programme	Information Technology
Name	Isaac O. Ogunlolu
Title	<b>HTML5, the future of mobile applications:</b> A comparison between HTML5 application development platforms and native platforms
Type of Study	Bachelor's Thesis
Date	16 May 2012
Pages	77 + 2 appendices
Instructor	Thai Bui Quoc
Company	IMSS Ltd. Oy Ab
Contact Person/Supervisor from Company	Antti-Pekka Aaltonen, CTO, IMSS Ltd. Oy Ab

HTML has been the most popular mark-up language for the past two decades but the introduction of HTML5 has even made it more popular and has created a buzz amongst IT enthusiasts. HTML5 comes with powerful tools that aim to revolutionize the web and the way web applications are viewed. These tools include advanced application programming interfaces which can be used to solve problems ranging from geolocation to online storage, thus making HTML5 a force to be reckoned with. A good question to be asked is why so much buzz around HTML5? What makes it different from HTML4 its predecessor? That is because it brings with it the “*write one run many*” solution mobile application developers have craved for. Mobile application development has largely been based on native development platforms like J2ME, Android, C++, C# etc. Developing applications with these platforms however come with limitations, the most important limitation being their cross-platform incapability. HTML5's cross-platform capability has impeccably solved this issue and revealed a nagging problem which native application possess.

To fulfil the main objective of this work, one would have to investigate HTML5, analyse the special features it comes with and compare HTML5 applications with native applications, thus drawing out a strictly neutral conclusion. Another objective was to implement a HTML5-based mobile application using RhoElements. Exploratory research and GUI programming methods were used for this work. Exploratory research was used to analyze HTML5 and to compare HTML5 with native applications. RhoElements and jQueryMobile was used to program the implemented application's GUI.

Satisfactory results were derived from this work process. Comparisons between HTML5 apps and native apps were made based on their development costs, cross platform capabilities, application performance and development cycle. Special HTML5 features were also analyzed and a lightweight mobile application was implemented.

**Keywords:** HTML5, Mobile application development, RhoElements, Cross-platform application development, HTML5 vs. Native applications.

## TABLE OF CONTENTS

PREFACE .....	I
ABSTRACT .....	II
EXPLANATION OF CHARACTERS AND ABBREVIATIONS .....	VI
1. INTRODUCTION.....	1
1.1. IMSS Ltd. Oy Ab .....	2
1.2. Project Overview .....	2
1.2.1. Project Objectives .....	3
1.2.2. Project Scope.....	4
2. PROJECT BACKGROUND.....	6
2.1. HTML5.....	6
2.1.1. History.....	7
2.1.2. Semantic Structure .....	9
2.1.3. CSS.....	9
2.1.4. JavaScript .....	11
2.1.5. HTML5's Browser Support .....	12
2.2. HTML5 Development frameworks .....	14
2.2.1. RhoElements .....	15
2.3. Native Development Platforms .....	17
2.3.1. Android OS .....	18
2.3.2. iOS.....	19
2.3.3. Windows Phone OS .....	22
3. SPECIAL HTML5 FEATURES .....	24
3.1. Offline applications .....	24
3.1.1. Application Cache.....	24
3.1.2. Local Storage .....	25
3.2. Geolocation.....	29
3.3. Real Time Communication and Connectivity .....	30
3.3.1. Web Workers .....	30
3.3.2. Web Socket .....	31
3.3.3. Notifications.....	32

3.4.	File Access.....	33
3.4.1.	File API.....	34
3.4.2.	Native Drag and Drop.....	34
3.4.3.	Desktop Drag in and Drag out.....	36
3.5.	Semantics and Markup.....	37
3.5.1.	Improved Semantic Tags.....	37
3.5.2.	New Form Types.....	39
3.6.	Multimedia.....	40
3.6.1.	Video.....	40
3.6.2.	Audio.....	41
3.6.3.	Canvas.....	42
4.	HTML5 APPLICATIONS VS NATIVE APPLICATIONS.....	43
4.1.	Development Costs.....	45
4.2.	Cross Platform Capabilites.....	47
4.3.	Application Performance.....	49
4.4.	Development Cycle.....	51
5.	IMPLEMENTATION AND RESULTS.....	53
5.1.	Background and History.....	53
5.1.1.	Design and Implementation.....	54
5.1.2.	SMS.....	55
5.1.3.	GPRS.....	56
5.1.4.	Hardware.....	56
5.2.	HTML5 Application Design.....	57
5.2.1.	Use Case Diagram.....	57
5.2.2.	Statechart Diagram.....	58
5.2.3.	Class Diagram.....	59
5.3.	GUI Design.....	59
5.3.1.	Old GUI Look (In J2ME).....	60
5.3.2.	New GUI Look (HTML5).....	62
6.	FURTHER DEVELOPMENT.....	64
7.	CONCLUSIONS.....	65
8.	REFERENCES.....	67

9. LIST OF APPENDICES .....77

## **EXPLANATION OF CHARACTERS AND ABBREVIATIONS**

API	Application Programming Interface
C	C Programming Language
CDMA	Code Division Multiple Access
CPU	Central Processing Unit
CSS	Cascading Style Sheets
C++	C Plus Plus
CTO	Chief Technical Officer
DOM	Document Object Model
GPRS	General Packet Radio Service
GSM	Global System for Mobile Communication
GUI	Graphical User Interface
HTML	Hypertext Mark-Up Language
HTTP	Hypertext Transfer Protocol
IETF	Internet Engineering Task Force
IP	Internet Protocol

IT	Information Technology
JS	JavaScript
J2ME	Java Platform, Micro Edition
OS	Operating System
PC	Personal Computer
PDF	Portable Document Format
ROI	Return on Investment
SDK	Software Development Kit
SMS	Short Message Service
TDMA	Time Division Multiple Access
URL	Universal Resource Locator
VM	Virtual Machine
WHATWG	Web Hypertext Application Technology Working Group
WWW	World Wide Web
W3C	World Wide Web Consortium



## 1. INTRODUCTION

Mobile application development has rapidly evolved recently due to the boom in the smartphone market. Software has to be constantly made and updated to cope with the rapid hardware and OS technology advancement the smartphone industry is currently experiencing. Mobile application development has largely been based on native development platforms like J2ME, android, C++, C#, iOS, Windows OS etc. These platforms however come with limitations of every kind, limitations which include cross-platform incapability, development costs, code management etc.

Imagine a situation when developers would develop an application just once and this application would run on motorola phones, nokia phones, android based phones, tablets, mobile PC's etc. irrespective of the device, the screen size, the software architecture and the hardware peripherals. That is what HTML5 is capable of and even though it is still under development /1/, it is increasingly becoming popular amongst developers and IT enthusiasts.

HTML5 is relatively new but it is a platform that would continue to pick up momentum as soon as developers start to realize how powerful and easy to learn it is. It comes with different API's which can be used to solve problems ranging from geo-location to online storage etc. Web applications run in mobile browsers and can also be re-packaged as native applications which can work on various mobile platforms. With the wide range of platforms to support, combined with the sheer power of mobile browsers, HTML5 is becoming the "write one, run many" solution which the mobile industry has craved for. HTML5 is the risk mobile developers should be taking because based on statistics gathered by strategy analytics about one billion HTML5 phones would be sold worldwide in 2013 /2/.

## **1.1. IMSS Ltd. Oy Ab**

IMSS Ltd. Oy Ab is the company who authorized this thesis, and they therefore own this paper, practical results, software and other materials that accompany this research work.

It was established in 2008 though had its origin already from 2004 but from 2008 it was called IMSS. IMSS was conceived in 2006, when Antti-Pekka Aaltonen and Panu Koivurova decided to work together to improve industrial maintenance by making innovative software. A project commenced in 2007, where Outokumpu, Kemi-Tornio University of Applied Sciences (Unit of Technical Education) and the founders of IMSS worked together to make a product to meet Outokumpu's need. The project ended in 2008, and after that IMSS was founded to make business use of the project's results. The main customer was Outokumpu Stainless Tornio. At first, the company had three employees all of whom had stakes at the company. IMSS has since grown having nine employees as of spring 2011.

IMSS provides software for various business uses and the main focus is on work management systems and mobile software. Their main customers are industrial, taxi, security and maintenance companies. IMSS has several software and hardware partners and resellers. In the near future, IMSS will concentrate on constantly improving its current services and software.

## **1.2. Project Overview**

HTML has come a long way and it was when HTML5 appeared one began to see and appreciate what the web with HTML5 is capable of. Personal interest in explicit and interesting technologies like HTML5 and the future prospects of IMSS Oy using HTML5 development platforms to develop their innovative mobile applications drove and inspired this research.

*“HTML5, the future of mobile applications: A comparison between HTML5 application development platforms and native platforms”* is the title of this work and as the title implies this paper explains why HTML5 is the future of mobile applications, compares HTML5 with native applications thereby drawing out positive conclusions and is concluded by a practical work which involves using RhoElements to develop a mobile application for IMSS Oy.

Although HTML5 is generally known to be a web technology, which through powerful development platforms like phonegap and RhoElements can be repackaged into both desktop and mobile applications, but due to the depth of this topic and time limitations the topic was streamlined and focus was placed on just mobile applications. One would in the process try to analyze HTML5, compare it with native development platforms, highlight the special features HTML5 comes with and therefore draw out positive conclusions from a “mobile” point of view. And as mentioned earlier this work also has a practical side to it which would be explained in details later /15/ /74/.

### **1.2.1. Project Objectives**

The main objective of this thesis is to draw out positive conclusions on why IMSS Oy should embrace HTML5 as their mobile application development platform. Other objectives include:

- To develop a mobile application for IMSS Oy using the HTML5 development platform RhoElements. The application would be an HTML5 version of the mobile application developed for the mobile devices project (A school project in which Antti Aaltonen the CTO of IMSS was involved), the purpose of the application would be to send messages through SMS or through the GPRS network to a server application which would in turn carry out the proposed commands of opening doors, closing doors or just querying door statuses.

- To compare HTML5 with native development platforms like android, iOS and Windows OS. The comparison would be based on factors like development costs, application performance, supported platforms and development cycle.
- To create a basis for further research on HTML5, which simply declares one's desire to continue research on HTML5 as it appears to be the future of the web, mobile applications and even desktop applications.

### **1.2.2. Project Scope**

This work is divided into two parts which includes the research work and practical implementation. For the research part, a background look at HTML 5, HTML5 development platforms, Android OS, iOS, Windows OS etc. would be taken. A detailed highlight of special HTML5 features would be given and based on the cost of development, development cycle, cross platform capabilities and application performance. Comparisons between HTML5 and other native languages would be made with the aim of drawing out neutral conclusions. These conclusions should however convince the thesis commissioner that HTML5 is the future of mobile applications.

The second part would involve implementation which includes using HTML5 to implement a mobile application. This part would include the implementation process, results and challenges faced during implementation. Mobile devices project was a final school project and what was done during the course of the project was to make simple software which could turn into something commercial while, at the same time, keeping the application as efficient as possible. The project was a huge chunk of work which had to be broken down, a chunk of the workload was to design the mobile application's GUI and code the mobile application. The function of the application would be to send commands through either SMS or GPRS to a server system which implements the commands and sends back results to the mobile application.

J2ME was used to code this application and even though the results were impeccable from the customer requirements point of view. The application was well below standard when it comes to the performance, user experience, GUI design and responsiveness. The main problem behind this was that the application's look, performance and responsiveness varied on different smartphones and mobile PC's. That was when the idea of coding the same application in HTML5 which would look, perform and respond the same way on all devices came up.

## 2. PROJECT BACKGROUND

### 2.1. HTML5

HTML5 can be simply defined by the formula given below:

$$\text{HTML5} \approx \text{HTML} + \text{CSS} + \text{JS} \text{ /3/}$$

The formula above implies that HTML5 is approximately a mixture of HTML, CSS and JS APIs which is an almost exact literal representation of what HTML5 is. Bringing these technologies together gives developers the ability to create applications that could only be created for desktops /4/. Wikipedia the online encyclopaedia defines HTML as a language used for structuring and presenting content for the WWW. HTML5 being the fifth revision of the HTML mark-up standard which has been evolving since its creation addresses HTML shortcomings while, at the same time, bringing with it rich API's for powerful applications, vast multimedia support and improved mark-up semantics /1/ /4/. The increased capabilities of modern browsers such as Google's chrome and the increasing inability of HTML4 to meet up with the functional demands of the web gave birth to HTML5.

WHATWG explained HTML5 as the new version of HTML4, XHTML1, and DOM Level 2 HTML which addresses many of these specification's issues while, at the same time, improving HTML and XHTML to have more adequate support for web applications. Furthermore, aside from defining a mark-up language which can be written in both HTML and XHTML, it also defines powerful API's (some of which are known as DOM level 0) that form the basis of the web architecture /5/.

*WHATWG which is an acronym for Web Hypertext Application Technology Working Group is a community of individuals from Apple, the Mozilla foundation and the Opera software group who are interested in evolving the web by focusing on the development of HTML and APIs needed for web applications /5/.*

HTML5 is still quite vague; people thus misunderstand it a lot. Though still under development and irrespective of how vague it is, it is increasingly becoming popular among web and mobile developers.

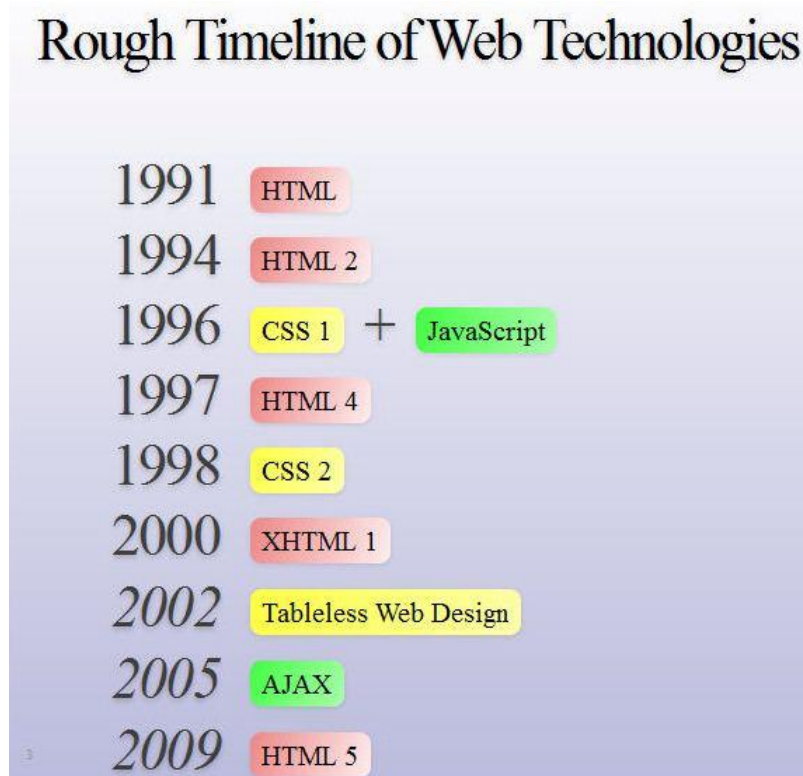


**Fig 1. HTML5's logo /6/**

### **2.1.1. History**

HTML5 was conceived when individuals from various organizations like Apple, Mozilla foundation and the Opera software came together to form WHATWG. These organizations were not impressed with W3C's declined interest in HTML, the direction in which XHTML was being led and the apparent disregard for the need of real-world authors. In response, WHATWG was formed to tackle these issues. With WHATWG's enthusiasm and W3C's willingness to be involved, work started on HTML5 through collaborations between WHATWG and the HTML group within W3C called W3C HTML in 2007. The first working draft of HTML5 was published by W3C in January 2008, the latest draft being an editor's draft which was published in February 2012 and is currently in that state as at the time this paper was written /1/ /5/ /7/.

The history of HTML5 is literally interwoven with the history of the web, and HTML5 is the latest web technology developed for the ever evolving WWW. The image below shows how the web has evolved since HTML was first introduced.



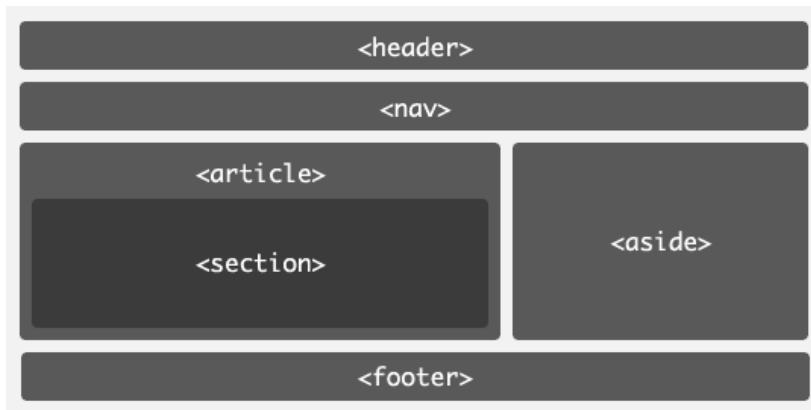
**Fig 2. Rough timeline of web technologies /8/**

WHATWG has since abandoned its HTML5 project and has switched to an unversioned development model for the HTML specification. WHATWG has also renamed its specification from HTML5 to HTML stating that they are just working on HTML and not worrying about version numbers. W3C has however continued to work on HTML5 hoping to reach a full specification status by 2022 /1/ /5/. IT organizations like Google, Microsoft, Apple, Mozilla, Facebook, IBM, HP, Adobe etc. have contributed voluminously to HTML5 /4/.

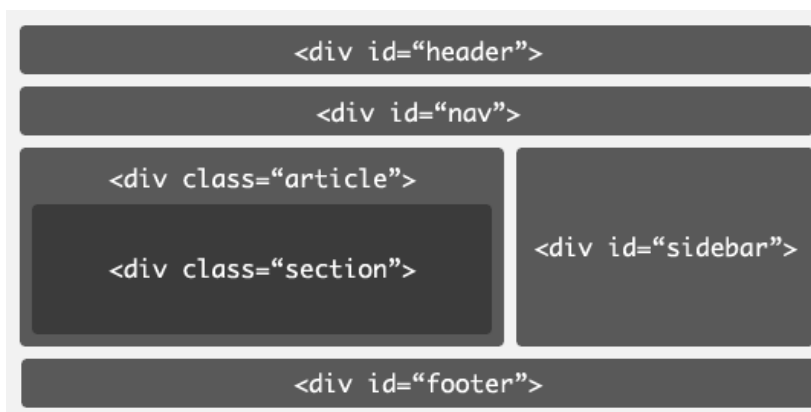


### 2.1.2. Semantic Structure

HTML5 comes with new elements that make it easier for developers to author more semantic web pages and applications. These elements can be used to replace the div element which in turn makes semantic representation easier, see images below /24/.



**Fig 3. HTML5's semantic structure /24/**



**Fig 4. HTML4's semantic structure /24/**

### 2.1.3. CSS

CSS is a mechanism used for adding style which includes font, color and design to web documents. CSS can also be defined as a style sheet language which is used to describe the

look and format of a document written in a markup language, the most common application being webpages written in HTML and XHTML /25/ /26/.

CSS was designed primarily to separate the content of a document from its presentation which includes elements like layout, color and font. The separation improves content accessibility, provides flexibility and control of presentation characteristic and enables multiple pages to share the same format thereby reducing complexity and repetition in the structural content /26/.

It can thus be concluded that HTML structures content while CSS formats the structured content /27/. The code list below shows how CSS is embedded into a HTML document using the `<style>` tag.

```
<! DOCTYPE html>
<html lang="en-us" >
<head>
  <title>Font Test</title>
  <style>
    @font-face {
      font-family: EraserDust;
    }
    body {
      background: #333333;
      font-family: EraserDust;
      font-size: 24pt;
    }
  </style>
</head>
<body>
  <p>This is a test!</p>
</body></html>
```

**Source Code List 1. How to embed CSS into an html document /28/**

### 2.1.4. JavaScript

JavaScript is a cross-platform object-oriented scripting language developed by Netscape. It is a small and lightweight language which cannot operate as a standalone but has to be used within other mediums like web documents and browsers. JavaScript is one of the world's most popular programming languages and is used in millions of web pages and server applications worldwide /29/ /30/ /33/.

Contrary to misconceptions which suggests that JavaScript relates to Java, this is however not the case. JavaScript is a scripting language that resides in HTML documents and therefore runs only on a browser. JavaScript has however since been used outside webpages, it has been used in PDF documents, desktop widget etc /31/ /32/.

The code list below shows how JavaScript is embedded into a HTML document using the `<script>` tag.

```
<html>
<body>
<h1>My First Web Page</h1>
<script type="text/javascript">
document.write("<p>" + Date() + "</p>");
...
</script>
</body>
</html>
```

#### Source Code List 2. How to embed JavaScript into an html document /34/

*Note: A good question would be “so what makes HTML5 so special?” The answer to that question is that HTML5 comes with newly implemented and powerful API's that enables developers to make use of new features like geolocation, application caching, canvas drawing, drag and drop, web storage, file manipulation, media playback for both video*

and audio, offline applications, multithreading, WebSockets etc. Some of these features are discussed in chapter 3.

### 2.1.5. HTML5's Browser Support

It would be completely useless to talk about HTML5 without at least making mention of what actually makes it work. Web browsers present and transverse information on the WWW /78/.

All major browsers like *Google chrome*, *Opera*, *Apple's Safari*, *Internet Explorer* and *Mozilla Firefox* support HTML5 in varying degrees even though the specifications for HTML5 are still under development /75/. The table below shows test results of how well web browsers support HTML5.

#### Desktop Browsers

**Table 1. Test result of how current desktop browsers support HTML5 /76**

Current Browser Version		
Browser	Version number	Points
Google Chrome	17.0	374
Mozilla Firefox	10.0	332
Opera	11.60	329
Apple Safari	5.1	302
Microsoft Internet Explorer	9	141

**Table 2. Test result of how desktop browsers being developed or currently in beta versions support HTML5 /76/**

<b>Development or Beta Version</b>		
<b>Browser</b>	<b>Version number</b>	<b>Points</b>
<b>Apple Safari</b>	5.2	352
<b>Opera</b>	12 alpha	344
<b>Microsoft Internet Explorer</b>	10	344

It can be seen from Table 1 that Google Chrome 17.0 which is the latest version at the moment supports HTML5 more than any other desktop browser by scoring 374 while Internet Explorer 9 scored quite low with 141 points. Table 2 however shows an improvement with Safari 5.2, Opera 12 alpha and Internet Explorer 10, this means their next releases would have better support for HTML5 /76/.

### **Mobile Browsers**

**Table 3. Test result of how current mobile browsers support HTML5 /77/**

<b>Current Browser Version</b>		
<b>Browser</b>	<b>Supported Platform</b>	<b>Points</b>
<b>Opera Mobile 12</b>	Multiple platforms	354
<b>Firefox Mobile 10</b>	Multiple platforms	315
<b>iOS 5</b>	Apple iPhone and iPod Touch	305
<b>MeeGo/Harmattan</b>	Nokia N9 and N950	271
<b>BlackBerry OS 7</b>	BlackBerry Bold 9900 and others	266
<b>Android 4.0</b>	Samsung Galaxy Nexus	256
<b>Bada 2.0</b>	Samsung Wave and others	251
<b>webOS 2.2</b>	Palm Pre 2 and HP Pre 3	201
<b>Android 2.3</b>	Google Nexus S and others	182
<b>Windows Phone 7.5 (Mango)</b>	Samsung Omnia W, LG E906	141

**Table 4. Test result of how mobile browsers being developed or currently in beta versions support HTML5 /77/**

Development or Beta Version		
Browser	Supported Platform	Points
Chrome Beta	All Android 4 devices	343

Table 3 and 4 shows how well mobile browsers support HTML5, opera mobile 12 supports HTML5 the most with 354 points and again Microsoft scores low with Windows Phone 7.5 (Mango) scoring 141 points /77/.

## 2.2. HTML5 Development frameworks

Since the introduction of HTML5 many development platforms have been introduced to help deal with the demands of designers and developers. The mouth-watering cross-platform capabilities HTML5 possesses and the ability for these development platforms to access APIs, take full advantage of JS, HTML5 and CSS, access mobile hardware components like camera and be deployed to multiple mobile platforms have enabled these platforms to carve a niche for themselves in the mobile application development environment /9/ /10/.

Using browser APIs, these frameworks provide a more consistent and comprehensive HTML5 support across mobile browsers. Some HTML5 development frameworks include:

- *Sencha touch* which was the first HTML5 framework for mobile devices.
- *PhoneGap* which enables users to access hardware features including GPS/location data, accelerometer, camera, sound and etc.
- *Rhodes* which is the main competition to PhoneGap and is Ruby based.
- *RhoElements*
- *Worklight*
- *BkRender*

- *DHTMLX Touch*
- *Jo*
- *M-Project*
- *SproutCore*
- *jQTouch*
- *Google Web Toolkit*
- *Titanium*
- *Appcelerator*
- *etc. /11/ /12/ /13/*

IMSS Oy is considering adopting RhoElements as its HTML5 development platform because they currently develop applications for just motorola devices. RhoElements at the moment fits that bill perfectly well because RhoElements is owned by Motorola and currently supports just Motorola devices, though that might change soon as it is being extended to support other devices. RhoElements would therefore be reviewed as the target HTML5 development platform.

### **2.2.1. RhoElements**

RhoElements is a web-based application development framework that is used to develop user-friendly and flexible applications that look, feel and act the same on every supported device. Applications developed using RhoElements are OS-independent, hardware-agnostic and have the versatility to work across several Motorola devices /14/.

#### **2.2.1.1. Features**

##### **Device Support**

Applications developed using RhoElements are device independent which simply implies that the applications will run most motorola devices regardless of the processor type and speed, memory architecture, screen size and resolution or if the device has Wi-Fi or a cellular network /14/ /15/.

**OS Support**

One application developed using RhoElements will operate on motorola devices running Microsoft Windows Mobile, Windows CE or Android. In the nearest future RhoElements would be improved to support more operating systems and non-motorola devices /14/ /15/.

**Application Design Control**

Applications developed with RhoElements are free from restrictions imposed by operating systems. Developers can freely create intuitive and easy to use graphical user interfaces /15/.

**Same Look, Same Feel**

RhoElements applications have a consistent look and feel. Regardless of supported device type, screen size or OS, applications look, feel and behave in a similar manner /14/ /15/.

**Stellar Application Performance**

RhoElements offers the latest technology in application development. RhoElements offers multithreading capabilities through WebWorkers while WebSockets enables faster data communication and transmission /14/ /15/.

**Features of Mobile Devices and Peripherals**

RhoElements captures barcodes, processes signatures and photos, processes debit or credit card transactions, prints out sales receipts etc /15/.

**Automatic Screen Fit**

RhoElements automatically scales up and down as needed to match the display of each device. It also supports screen rotation which allows users to take advantage of the extra screen /15/.



### **Online and Offline**

Applications developed with RhoElements can remain active even when there are lapses in the wireless network coverage /15/.

#### **2.2.1.2. Benefits**

Other benefits that come with RhoElements include:

- Reduced application costs due to the cross platform capabilities of RhoElements
- Reduced programmer costs: one has to master only one skillset and that is HTML5
- Improved mobile computer ROI: every utility and peripheral is utilized
- Reduced end-user training costs
- Improved application performance
- Improved employee productivity: Waiting time for screens to refresh is drastically reduced.
- Minimized use of network bandwidth: RhoElements applications require very little bandwidth /15/.

### **2.3. Native Development Platforms**

Native applications are applications developed to run specifically on operating systems or machine firmware and they typically have to be adapted for different devices. For example, a native application developed for the iPhone or iPad will have to run on its proprietary iOS platform /16/. A native application can also be defined as an application that is designed to run on a particular platform (machine language or OS). These applications are always installed on the platform and rely heavily on the platform in most cases /17/.

Native development platforms are execution environments on which native applications run. There are lots of mobile platforms out there but the most popular platforms include Apple's iOS, Nokia's Symbian platform, Google's Android platform, Microsoft's windows mobile platform and so many more.

### 2.3.1. Android OS

Android OS is a Linux-based OS for smartphones and tablets which was developed by Open Handset Alliance which is led by Google. Android is currently the world's most popular and bestselling mobile platform, having more than 400,000 applications as at October 2011, 10 billion application downloads, and running on over 200 million devices as at November 2011. Android being open source also has a very large community of developers that write applications which extend the functionality of Android devices /18//19/.

#### Architecture

Android is particularly Linux based due to the Linux kernel it runs on. It is also largely based on middleware, collated C/ C++ libraries and APIs and application software running on Java-compatible libraries. Android runs on the Dalvik virtual machine which relies on the Linux kernel for behind the scene functionalities like threading and low-level memory management. The Dalvik VM executes the .dex (Dalvik executable) format which was explicitly optimized for minimal memory footprint. Android relies on its Linux kernel for core system services such as process management, memory management, network stack, driver model and security. Android uses the Linux kernel as an abstraction layer between the hardware and the software stack and has the ARM architecture as its main hardware platform. The diagram below shows the graphical representation of the android architecture /18/ /20/.



**Fig 5. Android Architecture /20/**

### **Application development**

Android applications are developed using the Java programming language and the code is developed and compiled using the Android SDK tools, the result being an Android package which is an archive file with an .apk suffix /21/. Android can also be developed using other development tools /18/.

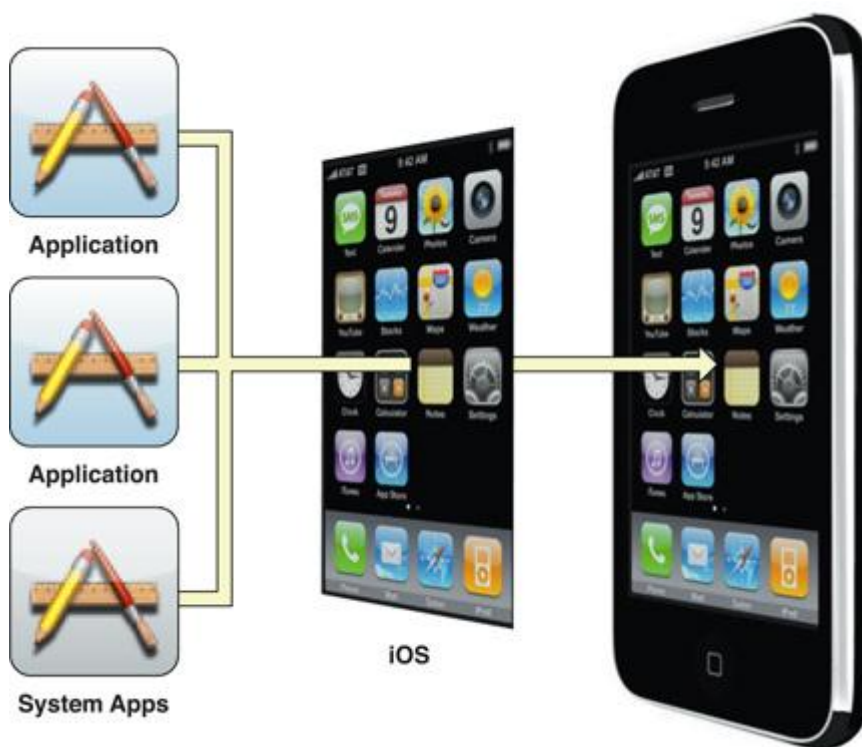
### **2.3.2. iOS**

Apple's iOS formerly known as iPhone OS is a mobile operating system developed by Apple Inc. iOS was originally developed for the iPhone but it has since been extended to support other products from the apple stable, products which include the iPhone, the iPad and the iPod. Apple's iOS is the third most popular mobile platform just behind Google's Android and Nokia's Symbian. It boasts of about 500, 000 app store applications and 24

million applications download as at October 2011. iOS is not open source and is therefore not licensed for installation on non-Apple hardware /22/.

### Architecture

At the highest hierarchy level, iOS acts as the link between the underlying hardware and the applications that appear on the screen, as shown in the image below. It is rare for applications to communicate directly with the underlying hardware, they instead communicate through the system's iOS platform which includes a set of well-defined interfaces that protect the application from hardware changes, thus enabling applications to work consistently on devices with different hardware capabilities.

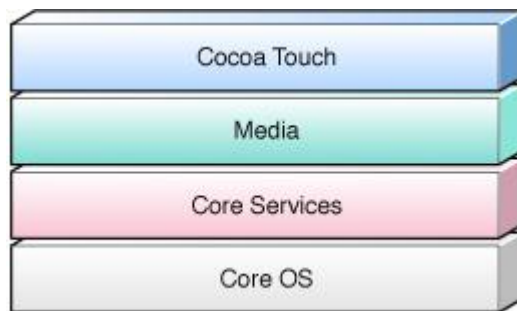


**Fig 6. Applications layered on top of iOS /23/**

The technology behind iOS's implementation can from an abstract point of view be divided into 4 layers. They include the *Core OS* layer, the *Core Services* layer, the *Media* layer and the *Cocoa Touch* layer. The lower system layers contain the fundamental

services and technologies on which all applications rely while the higher levels contain more advanced services and technologies /22/ /23/.

- The *Cocoa Touch* layer contains the key frameworks used for building iOS applications. It is at this layer the basic application infrastructure and support for key technologies is defined. This layer supports technologies such as multitasking, touch-based input, push notifications, and many high-level system services.
- The *Media* layer involves creating the best multimedia experience available on a mobile device. This layer contains graphics, audio and video technologies.



**Fig 7. iOS Layers /23/**

- The *Core Services* layer contains the fundamental services that all applications find useful.
- The *Core OS* layer forms the foundation which contains the low-level features on which other technologies are built upon. /23/

### **Application development**

Applications are specifically written and compiled for iOS and the ARM architecture using the iOS SDK. The Safari browser which comes pre-installed in Apple devices also supports web applications which can be wrapped up into mobile applications. These applications are developed by combining HTML, JS and CSS.

### 2.3.3. Windows Phone OS

Windows Phone is a mobile OS designed and developed by Microsoft. Windows phone comes with an explicit user interface which was designed using metro. It also controls the device hardware on which it runs and integrates the OS with third party software and other Microsoft services. Windows Phone is the latest mobile platform from the Microsoft franchise, the previous platform was Windows Mobile which has completely been replaced by Windows Phone /35/.

#### Architecture

Windows Phone OS has a layered architecture as shown in the image below. Unlike iOS Windows Phone has the ability to run on multiple hardware platforms. In order to provide a consistent user experience a set of defined requirements which include an ARM7 CPU, a DirectX capable GPU, a camera, and a multi-touch capacitive display must be met /36/.

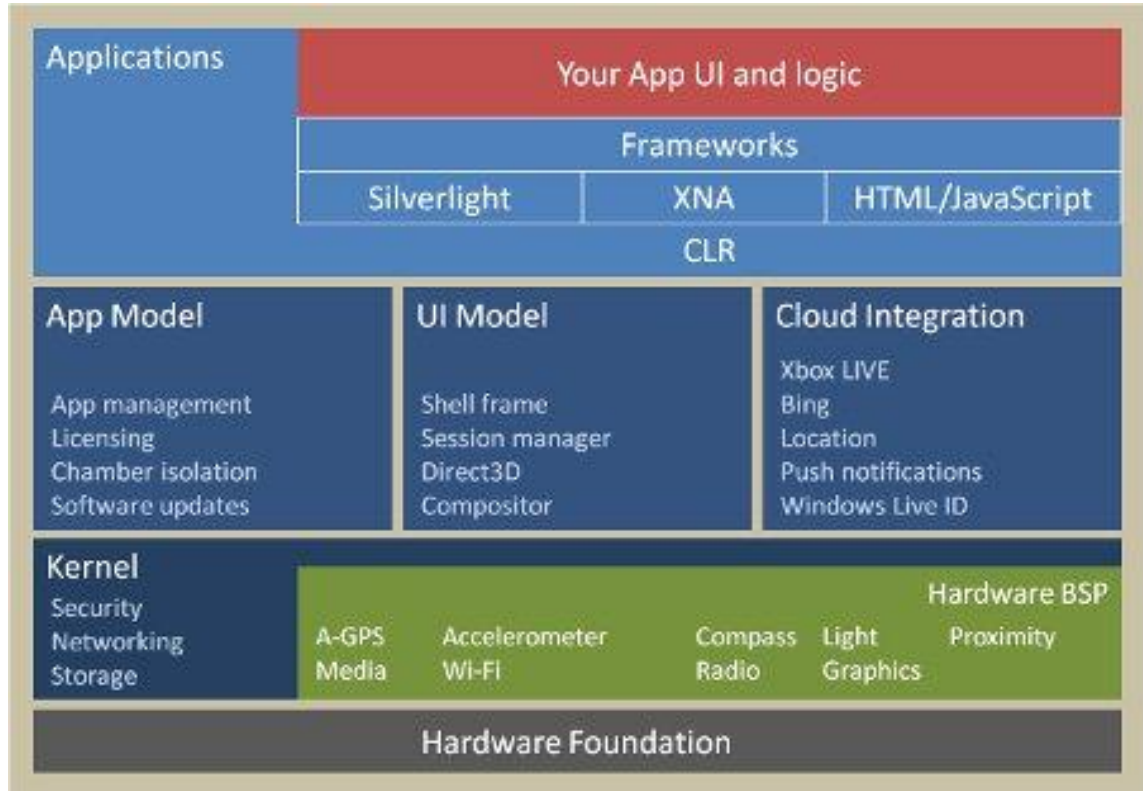


Fig 8. Windows Phone Architecture /36/

**Application Development**

Windows Phone applications are written in C# and are based on XNA or Silverlight.

Applications must be approved by Microsoft before it is submitted to the Windows Phone Marketplace /35/.

### 3. SPECIAL HTML5 FEATURES

HTML5 comes with special features not seen in HTML4, thus making it special. These features have increased its functional capabilities making HTML5 ideal for powerful, efficient and functional web applications.

Taking a thorough look at these features could make this paper rather lengthy, therefore only a brief look at each feature would be taken, in the process, sighting examples and making references to code lists and images when necessary.

#### 3.1. Offline applications

HTML5 is quite synonymous with the web thus making offline sound strange in this context. HTML5 however brings this possibility by making your applications available offline irrespective of network unavailability. When talking about web applications we directly refer to applications that will not function without an internet connection, however HTML5 has solved this problem by providing two solutions: one is a *SQL-based database API* which is used to store data locally and the other an *offline application HTTP cache* which ensures the availability of applications even when not connected to a network /43/.

##### 3.1.1. Application Cache

The `manifest` attribute on the `html` element is responsible for ensuring that web applications remain available even when the user is not connected to the network /43/. Even though most modern browsers have caching capabilities, they are largely unreliable and inefficient. HTML5 addresses these issues with the application cache API /79/.

The application cache API enables offline browsing, stores cached data locally therefore making data load faster and reduces work load on servers /44/. Application caching is quite simple and easy to use, see code list below:



```
CACHE MANIFEST
index.html
help.html
style/default.css
images/logo.png
images/background.png

NETWORK:
server.cgi
```

### **Source code list 3: Example of a manifest file /43/.**

The code list above shows how a manifest file looks like. Files under the cache manifest section (`index.html`, `help.html`, `style/default.css`, `images/logo.png`, `images/background.png`) will all be cached while `server.cgi` under the network section will not be cached /43/.

For the manifest file to function it must be linked to the HTML5 application by declaring it in the HTML5 application like this:

```
<!DOCTYPE HTML>
<html manifest="cache-manifest">
...

```

### **Source code list 4: Referencing a manifest file /43/.**

It is important to note that the `manifest` attribute must be added to every page of the application which is to be cached /44/.

## **3.1.2. Local Storage**

The first thing that comes to mind when dealing with user data is a server, developers would send data to and retrieve data from web servers. HTML5 however changes that with

the storage APIs it offers, they include *Web Storage*, *Web SQL Database*, *Indexed DB* and *File Storage* /45/.

### 3.1.2.1. Web Storage

*Web storage* also known as *Local storage* or *DOM storage* is used to store key-value pairs locally like any JavaScript object. The data saved persists even when users navigates away from the web site, closes the browser tab or even exits the browser. As stated earlier, data is stored based on a named key and data is retrieved using the same key, the key is always a string value. The data itself can be any type supported by JavaScript but it has to be converted into a string value before it can be stored. It can be thus concluded that *Web storage* is used to store unstructured and miscellaneous data.

It can however be argued that old offline storage mechanisms like *cookies* perform the same function. Cookies are however extremely limited in data capacity, are encapsulated within HTTP headers and they are transferred within the traffic between clients and servers, they therefore slow down network activity. *Local storage* also comes with a second object called *Session storage*, this works the same way *Local storage* works but the stored data is cleared when the browser is closed /45/. API functions used for *Local storage* includes `setItem()`, `getItem()`, `removeItem()` and `clear()` /45/ /46/ /47/. See code below for sample code of how to use *Local storage* to store a key-value:

```
if(!localStorage.getItem("checkins")) {  
    localStorage.setItem("checkins", JSON.stringify([]));  
}
```

**Source code list 5: Using local storage for a simple key-value storage /47 slide 8/.**

### 3.1.2.2. Web SQL Database

*Web SQL Database* is an offline SQL database that is largely based on SQLite. The *Web SQL database* is not part of the HTML5 specification but it is part of a suite of specifications that allows developers build rich and powerful web applications /80/. *Web*

*SQL database* comes with the power, functionality, speed and effort of a structured SQL relational database /45/. The *Web SQL database* API is variably supported across web browsers, though web browsers like Google chrome, Opera and Safari fully support the API, Mozilla Firefox decided not to support the API /80/ /81/.

Using the *Web SQL database* is simple and straightforward, code list 6 below shows how to open or create a database and code list 7 shows how execute a transaction. If a developer tries to open a database that does not exist, the API simply creates the database and developers don't have to bother about closing the database /80/.

```
var db = openDatabase('mydb', '1.0', 'my first database', 2 * 1024 * 1024);
```

**Source code list 6: Code list showing how to create and open a database /80/.**

```
var db = openDatabase('mydb', '1.0', 'my first database', 2 * 1024 * 1024);
db.transaction(function (tx) {
    tx.executeSql('CREATE TABLE IF NOT EXISTS foo (id unique, text)');
    tx.executeSql('INSERT INTO foo (id, text) VALUES (1, "synergies)');
});
```

**Source code list 7: Code list showing how to make database transactions /80/.**

### 3.1.2.3. Indexed DB

*Indexed Database* was designed by reaching a compromise between *web storage* and *web SQL database* therefore combining the simplicity that comes with *web storage* with the speed *web SQL database* offers. An *Indexed database* is simply a collection of object

stores which stores objects. The stores are similar to SQL tables but do not have constraints on the object structure making it needless to make definitions upfront /45/ /48/. The code list below shows how to open an indexed database and set its version.

```
setup: function(handler)
{
    var openRequest = indexedDB.open("geomood", "Geo-Mood
Checkins");
    openRequest.onsuccess = function(ev) {
        db = ev.target.result;
        db.onerror = function(ev) { console.log("db error",
arguments, ev.target.webkitErrorMessage); }
        db.version==version ? handler() :
indexedStore.reset(handler);
    }
},

reset: function(handler)
{
    var versionRequest = db.setVersion(version);
    versionRequest.onsuccess = function(ev) {
        if (db.objectStoreNames.contains("checkins"))
db.deleteObjectStore("checkins");
        var checkinsStore = db.createObjectStore("checkins", {
keyPath: "time" });
        checkinsStore.createIndex("moodIndex", "mood", { unique:
false });
        handler();
    }
},
```

**Source code list 8: Code list showing how to set the version and create an indexed database /48/.**

### 3.2. Geolocation

The *Geolocation* API (though not part of the HTML5 specification) is a separate API specification developed by W3C. The *Geolocation* API defines a high level interface used to locate information associated with the hosting device. This information includes the latitude, longitude and sometimes altitude of the device. The *Geolocation* API comes with methods like `getCurrentPosition()`, `watchPosition()` and `clearWatch()` /49/ /82/.

Due to its importance the *Geolocation* API is widely supported by desktop web browsers; browsers like Mozilla Firefox, Google Chrome, Safari, Opera and Internet Explorer all support the *Geolocation* API. There is also a significant amount of support on mobile devices; platforms like Android, iPhone, Opera Mobile, Symbian, Blackberry and Maemo all support the *Geolocation* API /49/. The code list below shows how to test if geolocation services are available:

```
if(navigator.geolocation) {
    /* geolocation is available */
} else {
    alert("I'm sorry, but geolocation services are not supported by
your browser.");
}
```

#### Source code list 9: Checking for Geolocation API support /52/.

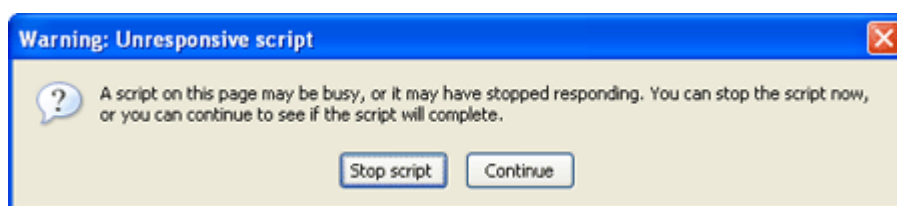
The way a user's position is received varies based on the media being used. A desktop browser uses either Wi-Fi which is accurate to 20m or IP geolocation which is accurate to just the city level and could give wrong information at times while mobile devices uses triangulation technologies like GPS which is accurate to just 10m and is only functional outside, Wi-Fi and GSM/CDMA cell ID's which are accurate to a staggering 1000m /49/.

### 3.3. Real Time Communication and Connectivity

HTML5 has introduced new concepts that make applications function faster by running scripts that are independent of the graphical user interface scripts on the background. This enables multiple scripts to be used in applications thereby making the GUI highly responsive when CPU runtime demanding scripts are being run, allows for bi-directional communication between clients and servers and gives real-time notification update. These concepts include *Web Workers*, *Web Sockets*, and *Notifications*: these concepts combined increases the speed and responsiveness of web applications.

#### 3.3.1. Web Workers

The *web worker* API specification is a separate specification developed by W3C and WHATWG, it does not belong to HTML5 but can however be used with HTML5 /83/. *Web workers* solve JavaScript's inability to run multiple scripts at the same time. JavaScript unlike other native development environments has all its execution processes inside a unique thread (single threaded) thus making applications unresponsive when running different processes. For example, considering a website that processes user events i.e. inputs, queries large amount of API data and processes the DOM, all these cannot be done simultaneously due to limitations to JavaScript /50/ /51/.



**Fig 9. Unresponsive script dialogue /51/**

*Web workers*, however, bring threading to JavaScript by defining an API that runs background scripts thereby aiming to get rid of situations shown in the image above. *Web workers* helps long-running scripts handle computationally intensive tasks without disrupting the fine user experience i.e. GUI and user interaction /51/. Code list 10 below

shows how to check if the browser supports *web workers* and code list 11 shows how to create and start a *web worker*:

```
/* Check if Web Workers are supported */  
function getWebWorkerSupport() {  
    return (typeof(Worker) !== "undefined") ? true:false;  
}
```

**Source code list 10: Check if browser supports web workers /84/.**

```
var worker = new Worker('task.js');  
worker.postMessage(); // Start the worker.
```

**Source code list 11: Working with Web Workers /51/.**

### 3.3.2. Web Socket

Web applications nowadays rely on event driven communications which are full duplexed and have minimal latency. Online games, social networking sites, collaborative platforms etc. demand fast communication medium to perform well. Sockets have been on the web in the form of HTTP for a while, but HTTP came with limitations which needed to be addressed. HTTP is half-duplexed therefore it allows one way traffic flow at a time and adds latency and message overhead. These factors put together have made HTTP less suitable for modern web applications thereby bringing a need for an upgrade /53/.

*Web Socket* which was defined in the communications section of the HTML5 specification is a W3C/IETF standard that provides a true full duplex communication channel thereby establishing an open connection between a client which could be a web browser and a server enabling both parties to send and receive data simultaneously and instantaneously. Only the data itself is sent, ignoring the HTTP overhead thereby reducing bandwidth usage drastically. *Web sockets* use `ws` as its new URL schema and `wss` *secure web sockets* for a

secure connection which is quite similar to HTTPS /53/ /54/ /85/. The code list shown below shows how to open and make use of *web sockets*:

```
var connection = new
WebSocket('ws://html5rocks.websocket.org/echo');

// When the connection is open, send some data to the server
connection.onopen = function () {
    connection.send('Ping'); // Send the message 'Ping' to the
server
};

// Log errors
connection.onerror = function (error) {
    console.log('WebSocket Error ' + error);
};

// Log messages from the server
connection.onmessage = function (e) {
    console.log('Server: ' + e.data);
};
```

### **Source code list 12: Working with Web Sockets /55/.**

The code list above shows how a web socket is opened by calling the `WebSocket` constructor, the code list also includes some event handlers which helps to monitor socket state, process received information and process errors.

### **3.3.3. Notifications**

Though not yet standardized, *the notifications API* is a specification proposed and currently being implemented by Google. It is not an HTML5 specification but it can be used with HTML5 /87/. It allows both passive notifications like new mail, tweets, posts or new calendar events and real-time user interaction notifications to be displayed to users.



The code list below *code list 13* shows how to check for notifications support while *code list 14* shows how to create notifications:

```
// check for notifications support
// you can omit the 'window' keyword
if(window.webkitNotifications) {
    console.log("Notifications are supported!");
}
else {
    console.log("Notifications are not supported for this Browser/OS
version yet.");
}
```

**Source code list 13: Code list showing how to check for notification API support /56/.**

```
function createNotificationInstance(options) {
    if (options.notificationType == 'simple') {
        return window.webkitNotifications.createNotification(
            'icon.png', 'Notification Title', 'Notification
content...');
    } else if (options.notificationType == 'html') {
        return
window.webkitNotifications.createHTMLNotification('http://someurl.
com');
    }
}
```

**Source code list 14: Code list showing to create notifications /56/.**

### 3.4. File Access

HTML5 through the help of some powerful JavaScript APIs now have a deeper communication and integration with the operating system.

### 3.4.1. File API

*File APIs* allow web applications to create, write, read and navigate a sandboxed section of a local file system. The API is divided into 3 themes which include:

- Reading and manipulating files which includes interfaces like `File/Blob`, `FileList` and `FileReader`.
- Creating and writing files which include interfaces like `BlobBuilder` and `FileWriter`.
- Directories and file system access which includes interfaces like `DirectoryReader`, `FileEntry/DirectoryEntry` and `LocalFileSystem`.

As of the time this paper was written, Google chrome was the only web browser supporting file API /57/.

### 3.4.2. Native Drag and Drop

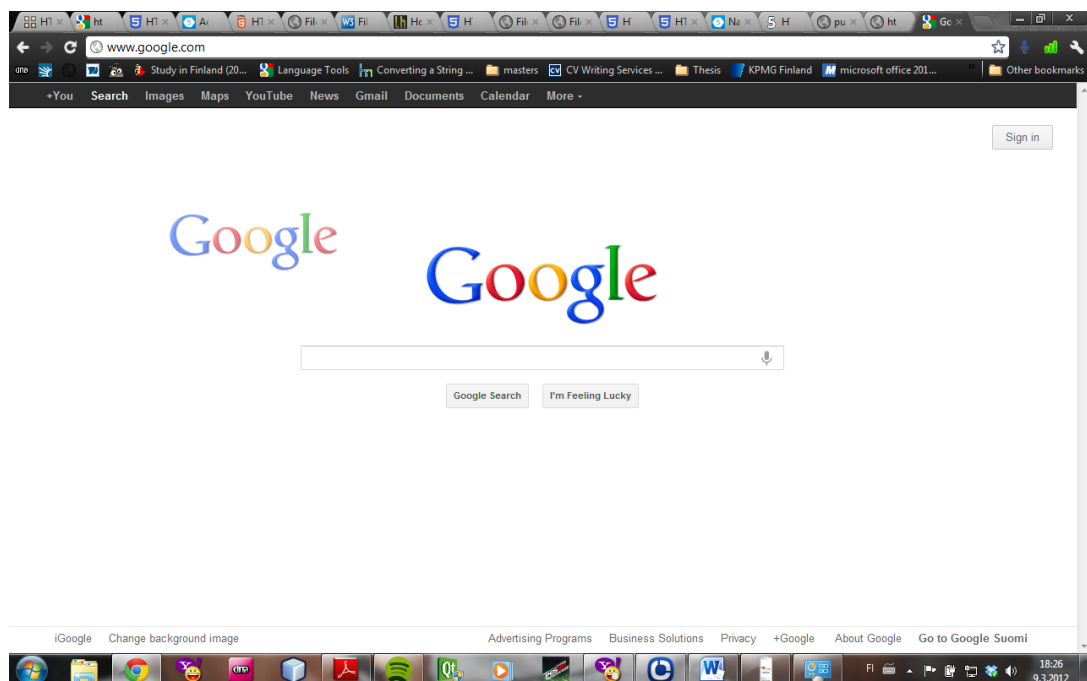
*Drag and drop* is a very important feature HTML5 has. *Drag and drop* which is also known as DnD is a powerful JavaScript API, event-based mechanism and also a new markup that is used to determine `draggable` elements on a page. Most browsers support DnD but to make use of it, it must be confirmed that the browser supports the functionality; *Code list 15* shows how to check for DnD functionality. An object can easily be made draggable by setting the `draggable` attribute to `true` as shown in *Code list 16* /58/.

```
if(Modernizr.draganddrop) {  
    // Browser supports HTML5 DnD.  
} else {  
    // Fallback to a library solution.  
}
```

**Source code list 15: Code list showing how to check for drag and drop functionality /58/.**

```
<div id="columns">  
    <div class="column" draggable="true"><header>A</header></div>  
    <div class="column" draggable="true"><header>B</header></div>  
    <div class="column" draggable="true"><header>C</header></div>  
</div>
```

**Source code list 16: Code list showing how to make an object draggable/58/.**



**Fig 10. Example implementation of drag and drop on Google's homepage /86/**

### 3.4.3. Desktop Drag in and Drag out

The DnD API also comes with a possibility of dragging files in and out of the desktop. Users can drag files from the desktop to the web browser and from the web browser to the desktop. Almost all browsers support dragging files from the desktop to a web application or a browser, dragging files from a web browser or a web application to the desktop was implemented by Google and only Google's chrome browser currently supports this feature, a good example being Gmail users ability to drag files out from their Gmail accounts to their desktops /58/ /89/. The code lists below shows how to implement simple desktop drag ins and outs.

```
function handleDrop(e) {
    e.stopPropagation(); // Stops some browsers from redirecting.
    e.preventDefault();

    var files = e.dataTransfer.files;
    for (var i = 0, f; f = files[i]; i++) {
        // Read the File objects in this FileList.
    }
}
```

**Source code list 17: Code list showing how to drag files from the desktop to the web browser /58/.**

```
document.querySelector('#dropzone').addEventListener('drop',
function(e) {
    var reader = new FileReader();
    reader.onload = function(evt) {
        document.querySelector('img').src = evt.target.result;
    };
    reader.readAsDataURL(e.dataTransfer.files[0]);
}, false);
```

**Source code list 18: another method used to drag files from the desktop to the web browser /88/.**

```
<a href="src/star.mp3" draggable="true" class="dragout"
  data-
downloadurl="MIMETYPE:FILENAME:ABSOLUTE_URI_TO_FILE">download</a>

var files = document.querySelectorAll('.dragout');
for (var i = 0, file; file = files[i]; ++i) {
  file.addEventListener('dragstart', function(e) {
    e.dataTransfer.setData('DownloadURL',
this.dataset.downloadurl);
  }, false);
}
```

**Source code list 19: Code list showing how to drag files from the web browser to the desktop /89/.**

### 3.5. Semantics and Markup

HTML5 has added more meaningful elements and provided developers with tools that make life easier. Tools like:

- New media elements
- New structural elements
- New semantics
- New link relation types
- New attributes
- New form types
- New microdata syntax

#### 3.5.1. Improved Semantic Tags

HTML5 provides a new set of meaningful elements which can be used to describe a typical web page layout. These meaningful elements which literally describes the content they

contain makes codes easier to read and organize and for search engines to search for web content /59/. *Code list 20* below shows the new semantic organization:

```
<body>
  <header>
    <hgroup>
      <h1>Page title</h1>
    </hgroup>
  </header>
  <nav>
    <ul>
      Navigation...
    </ul>
  </nav>
  <section>
    <article>
      <section>
        Content...
      </section>
    </article>
  </section>
  <aside>
    Top links...
  </aside>
  <figure>
    
    <figcaption>Chart 1.1</figcaption>
  </figure>
  <footer>
    Copyright ©
    <time datetime="2010-11-08">2010</time>.
  </footer>
```

**Source code list 20: HTML5 semantic structure showing new elements /60/.**

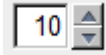
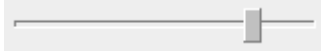
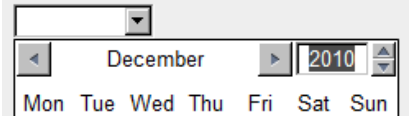
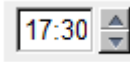


As shown in *code list 20* new tags like `<hgroup>`, `<nav>`, `<section>`, `<article>`, `<aside>`, `<figure>`, `<footer>`, `<figcaption>` and `<time>` makes more sense than having lots of `<div>` tags thus making page structure more presentable, accessible, readable and searchable .

### 3.5.2. New Form Types

#### Controls

New form controls have been introduced in HTML5, these makes data collection and collation easier compared to previous HTML versions. They include:

**Table 5. New HTML5 form input types**

<code>&lt;input type="number"&gt;</code>	
<code>&lt;input type="range"&gt;</code>	
<code>&lt;input type="date"&gt;</code>	
<code>&lt;input type="time"&gt;</code> Other controls in this group include datetime, month and week	
<code>&lt;input type="color"&gt;</code>	
<code>&lt;input type="search"&gt;</code>	

Other form controls include `<datalist>`, `input type="tel">`, `input type="email">` and `input type="url">`. HTML5 forms also come with new

attributes which include: `placeholder`, `autofocus`, `min` and `max`, `step`. Output mechanisms like `output`, `progress` and `meter` and form validation entities like `required` and `pattern` /61/.

## 3.6. Multimedia

Until HTML5, web application developers relied heavily on browser plugin technologies like Adobe's *flash* and Microsoft's *silverlight* to provide cutting edge multimedia effects which includes audio, video, 2D and 3D animations and graphic styling. HTML5 has however come with some astonishing and powerful effects that enable developers to do without these technologies which are sometimes very expensive compared to HTML5's free and open standard.

### 3.6.1. Video

HTML5's new video tag `<video>` is one of such powerful and much needed tools HTML5 came with. *Adobe flash* has been the undisputed plugin used to deliver videos to the web, the introduction of HTML5's `<video>` tag has however made it possible for developers to integrate videos directly into web pages without the need for any plugin-based solution.

Integrating a video into a web page before HTML5 was a big issue for developers due to its unreliability. Using the `<video>` tag brings simplicity and is refreshingly straightforward compared to the previous method used to add videos. *Code list 21* and *code list 22* shows the difference /62/.



```
<object width="425" height="344">
  <param name="movie" value="http://www.youtube.com/v/</param>
  <param name="allowFullScreen"
    value="true"></param>
  <param name="allowscriptaccess"
    value="always"></param>
  <embed src="http://www.youtube.com/v/9sEI1AUFJKw&hl=en_GB&fs=1&"
    type="application/x-shockwave-flash"
allowscriptaccess="always"
    allowfullscreen="true" width="425"
    height="344"></embed>
</object>
```

**Source code list 21: Embedding a video into a webpage before the <video> tag /62/.**

```
<video src=turkish.ogv
  width=320
  height=240
  controls
  poster=turkish.jpg>
</video>
```

**Source code list 22: Embedding a video using the <video> tag /62/.**

### 3.6.2. Audio

Developers relied on plugins most especially adobe flash due to its ubiquitous nature to deliver audio and video contents to the web. The introduction of HTML5's <audio> tag has however just like the <video> tag enabled audio playback on the web without the help of plugins /63/.

```
<audio controls preload="auto" autobuffer>
  <source src="elvis.mp3" />
  <source src="elvis.ogg" />
  <!-- now include flash fall back -->
</audio>
```

**Source code list 23: Embedding an audio content using the <audio> tag /63/.**

### 3.6.3. Canvas

With HTML5, graphics on the web could only get better. The <canvas> element enables 2D and 3D image rendering comes with features that include:

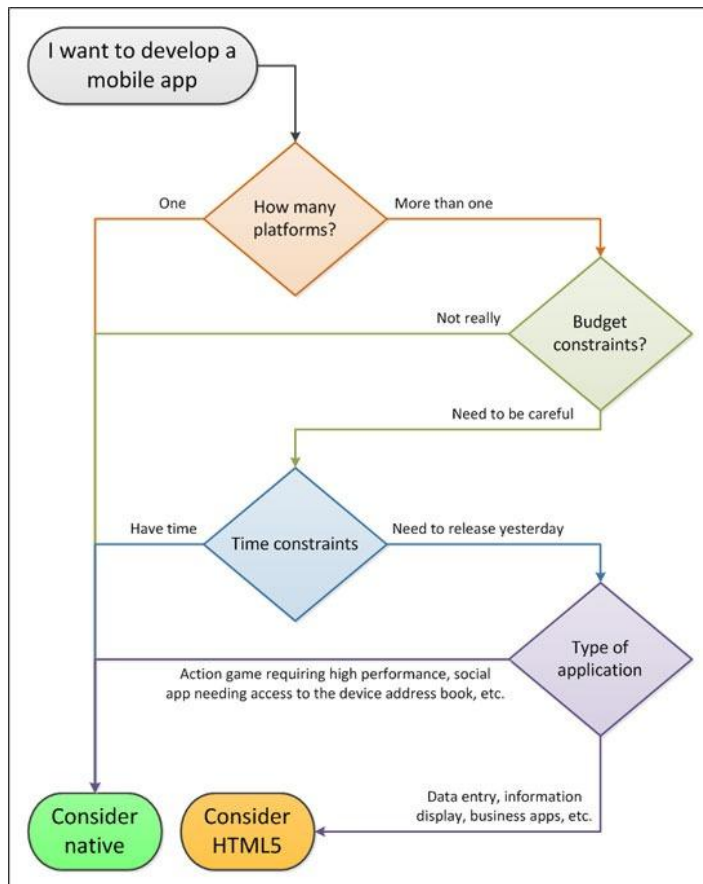
- shape drawing
- color filling
- creating gradients and patterns
- text rendering
- manipulation of images, video frames, pixels and other canvases /64/

## 4. HTML5 APPLICATIONS VS NATIVE APPLICATIONS

Native applications have been around for a while and they will continue to thrive as time goes on, HTML5 applications on the other hand are quite new, promising and thriving to create a market niche of their own. Since the introduction of HTML5 there have been lots of arguments on which path is best to follow, is it HTML5 applications or native applications?

This chapter joins a long list of arguments on this same topic, this is, however, strictly from a personal point of view and is a result of the extensive research which has been done on this topic. These arguments are based on facts and future projections because it is quite obvious HTML5 will not take over from native applications overnight. Important application development factors like development costs, application performance, supported platforms, development cycle and GUI design would be the theme of these arguments. It is also important to note that even though the purposed aim is to convince the thesis commissioner to pick HTML5 applications in favor native applications, it would, however, be as neutral as possible thereby declaring native applications as the better one if need arises.

The image below shows a simple flowchart which in a modest way describes this argument graphically:



**Fig 11. Flowchart showing how to choose between native apps and HTML5 apps /69/**

Fig 11 graphically summarizes the whole argument by bring together the major factors that need to be considered when trying to compare these two development platforms. The flowchart describes the best path to take when trying to choose which development platform suits your application best. These factors come in form of simple questions that needs answers, for instance, when going native, the developer might be thinking of supporting just **one platform**, is not **constrained** to a particular **budget**, has **enough time** to implement the application and is making a **high performance application** with an **exquisite user interface**. Developing an HTML5 application on the other hand depicts a completely different scenario when compared to the first scenario, in this scenario the developer wants the application to be **cross-platform**, is **constrained to a budget**, does **not** have so much **time** and is **just making a good enough application with a simple user interface**.

## 4.1. Development Costs

This is where HTML5 applications come out as the most definite and probably undisputed winner. Native applications are fragmented between many different mobile platforms. Platforms which include *Android, iOS, Windows Phone, Blackberry, Symbian* etc all use different code bases. Developing applications for these platforms therefore costs a significant amount when compared to the amount used to develop HTML5 applications. When developing native applications many cost factors are put into consideration, factors which include how many platforms are supported, how much skill-set the developer should possess etc. These factors being a very important part of the whole development process very much affect how much is being invested into making an application /65/ /66/.

For example, a company making a native application that would run on all major smartphones in the market (*targeting just one platform will not help with sales*) would have to develop (*from the scratch*) for Android, iOS, Windows Phone, Blackberry, Symbian and so on. Developing this application for different platforms costs a lot to do and not just that, there might be a possible need to make various versions of the application for one single platform due to hardware differences, architectural differences, screen size, orientation and so on. It does not end there, there would be patches, updates etc. which would have to be coded and released to the application store. These costs a fortune compared to HTML5 applications *can also be called hybrid applications* which are easier and faster to develop, platform independent and can simply be updated by refreshing the webpage, by restarting the web browser or by releasing the update via an app store /65/ /66/ /68/.

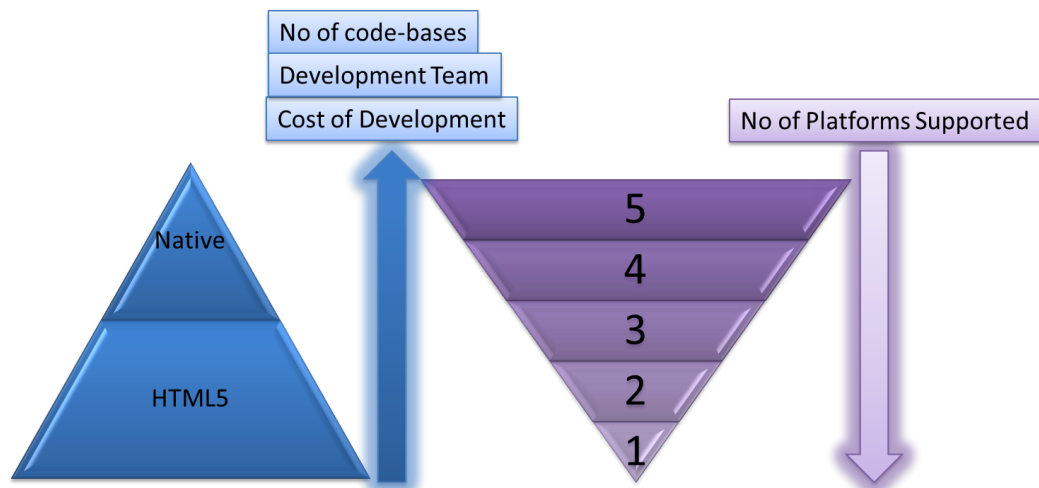
*HTML5 applications in this context can also be referred to as hybrid applications. Hybrid applications are HTML5 applications that are wrapped into native applications by cross-platform development platforms like phonegap and RhoElements giving them access to some device specific peripheral APIs and sensors /65/.*

HTML5 in this case is the clear winner when costs used to develop HTML5 applications is considered, compared to native applications. HTML5 applications are written once and deployed on all devices that have a browser and have access to the internet /67/. HTML5 is

also not so complicated to learn and implement, thus this makes it much easier for developers to write applications and manage a single code-base that is deployed to several OS platforms therefore eliminating the stress and difficulties of managing different code-bases for different OS platforms. It also, at the same time, makes it easier for organizations to develop cost-effective applications at a very fast pace using very little budgets. So, rather than spending a fortune on developing for many platforms and updating the application across all platforms, firms can invest in other productive and innovative projects /65/ /66/.

The cost of employing professionals with specialized skill-sets specific to various platforms, acquiring different development tools, making use of complicated API's and making use of different programming languages. All these leads to the company hiring more people who would manage the different platforms being supported, so for instance if a company supports Android, iOS, Blackberry and Windows Phone, four different teams might have to be employed. They would also have to manage enormous code bases for the different platforms being supported. These factors make native applications cost more to develop when compared to HTML5 (*hybrid*) applications /69/.

The diagram below depicts the cost effects faced when supporting native applications. It shows the relationship between the development platforms, the number of platforms supported and their cost effects. The pyramids show that the more platforms supported the more the cost of development, the number of code-bases supported and the larger the development team gets. For example, if a company develops native applications and supports two different platforms, the cost is therefore two times the cost of supporting just one platform. It would be five times the cost of one platform if the company supports five different platforms. The image also shows that when developing native applications, it is highly probable that more than one platform is supported. The story is, however, completely different with HTML5 applications, which have fewer code-bases (just one in most cases), have just one development team and significantly costs less when compared to native applications.

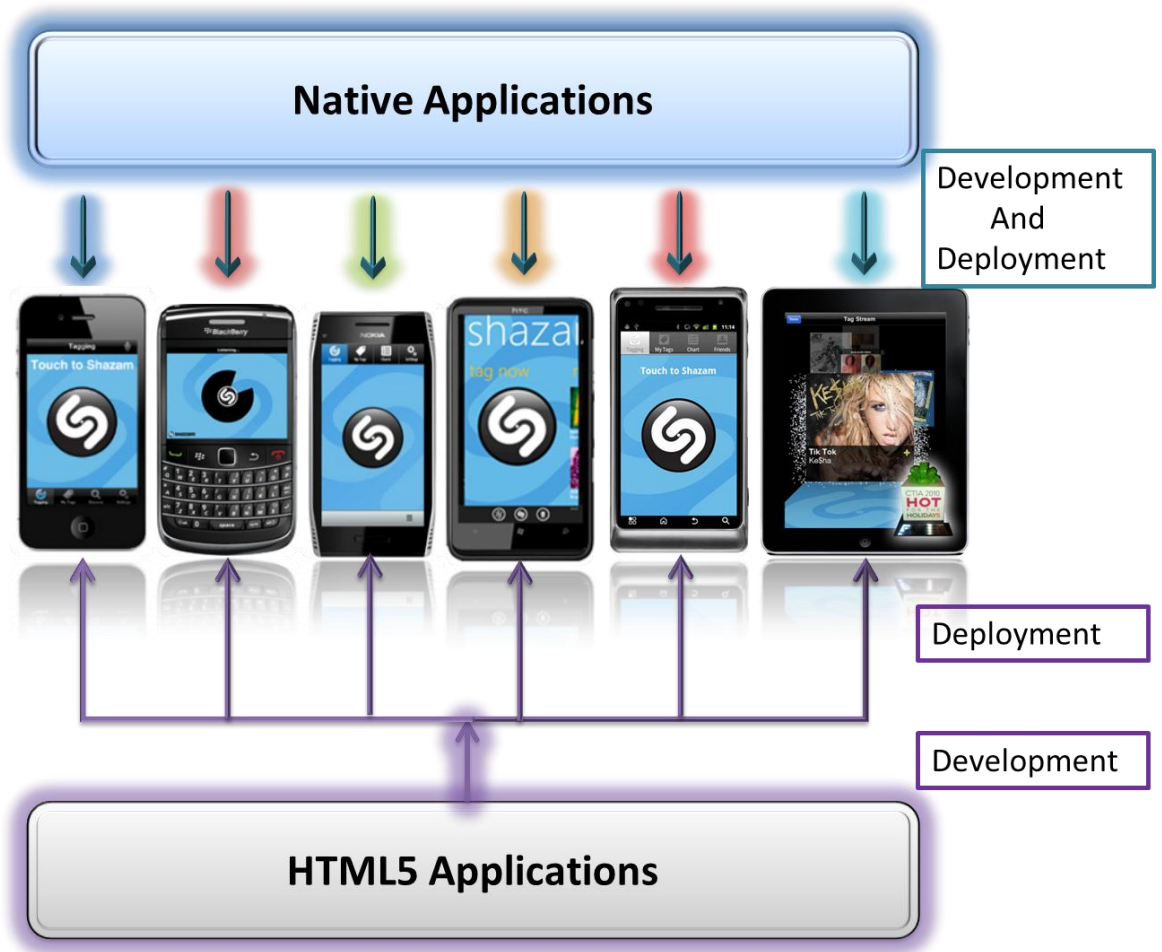


**Fig 12. Pyramid model showing the relationship between costs and number of platforms supported**

## **4.2. Cross Platform Capabilities**

HTML5 once again is a clear and definite winner in this aspect and, that is why, it is being embraced by developers and enthusiasts as the future of mobile application development. Native applications are not cross platform, they need to be developed from scratch to support different platforms due to their platform dependency. An application written for iOS would never work on an Android device and vice versa, this has been the issue with native applications until HTML5 was introduced.

The introduction of HTML5 created a great buzz, this buzz was mainly due to one discovery made and that is the cross-platform capabilities HTML5 possesses when wrapped into a native application with cross-platform development tools like PhoneGap, RhoElements etc. The image below graphically explains the cross-platform capabilities of HTML5:



**Fig 13. Image showing the cross-platform capabilities of HTML5**

The self-designed diagram above is self-explanatory, the colorful arrows pointing downwards from the native applications bar indicate that for all devices, a new application is developed from scratch and deployed. This amounts to a very large code-base which is expensive, takes a significant amount of time to be completed and is very difficult to manage. HTML5 applications on the other hand as depicted in the diagram above (arrows pointing upwards) is developed once and then deployed to all devices. This means there is only one code-base to manage making it easy to update applications and this also makes it significantly cheaper when compared to native applications development. HTML5 is the *sine qua non* of cross-platform mobile applications and rightly so because HTML5 impeccably offers the “*write once, run many*” solution /66/ /70/.

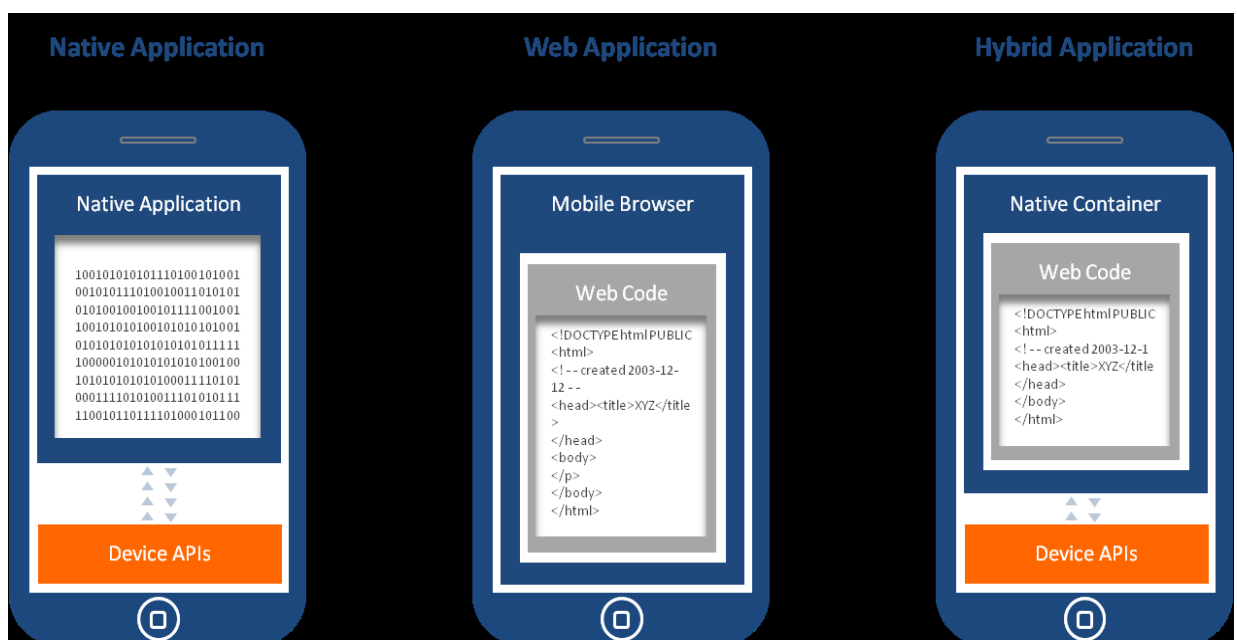


### 4.3. Application Performance

Native applications in this aspect emerge as the *de facto* winner. HTML5 applications have not been able to compete with native applications when it comes to application performance. Native applications have better look and feel, are more secure, runs faster and have more access to core device API's like camera, Bluetooth, GPS etc /70/ /71/.

#### OS Integration

Native applications integrate more with the OS, thereby gaining access to core device APIs like the camera API, GPS API, Bluetooth API etc. These APIs allow native applications to interact directly with the device's touchscreen, keyboard, microphone, graphics, camera, files etc.



**Fig 14. Image showing the OS integration levels of the different application types /72/.**

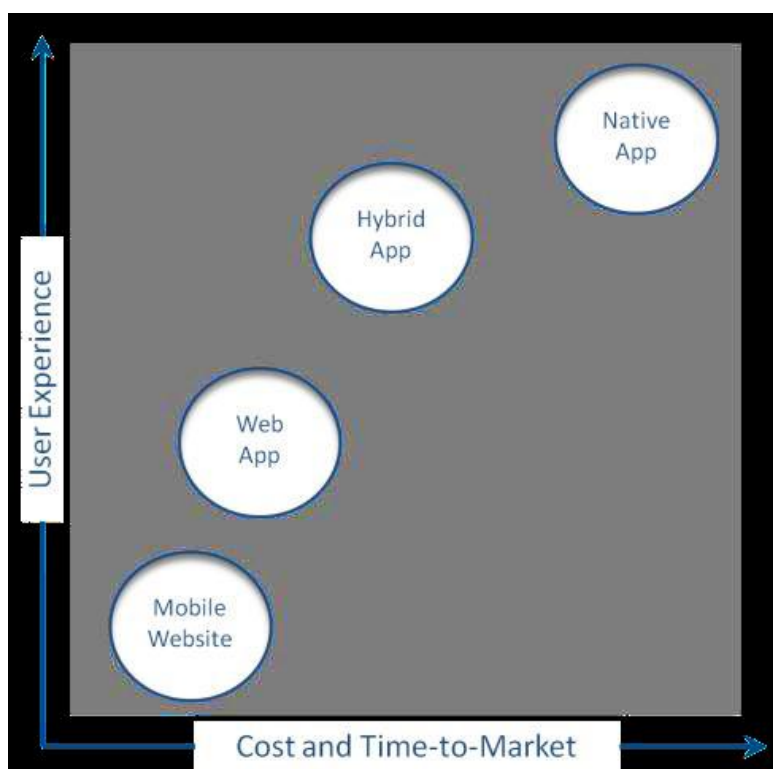
As shown in the image above, native applications have direct access to the device APIs thus allowing developers to fully express themselves making use of what is needed in a very efficient manner. Ordinary web applications runs through a web browser thus have no access to any device API. Hybrid (HTML5) applications however combine native

development with web technologies run on a web browser but are wrapped in a native container by cross platform technologies like phonegap and RhoElements thus gaining access to specific device APIs /72/.

### User Experience

HTML5 is still under development and might not be completed until 2014. Till its completion when its capabilities can be seen and fully understood, HTML5 applications will keep playing second fiddle to native applications when it comes to user experiences.

User experience in this context is a collection of features that includes the application's look and feel, touchscreen responsiveness, speed, graphic performance etc. Native applications possess these features in abundance due to their ability to feel natural on a device thereby taking full advantage of the device capabilities whereas HTML5 applications due to limited access to APIs do not possess the slick user experience native applications like games deliver /69/. The drawback with this is that as shown in the image below the greater the user experience the greater the cost and time of development.



**Fig 15. Image showing relationship between cost and user experience /72/.**

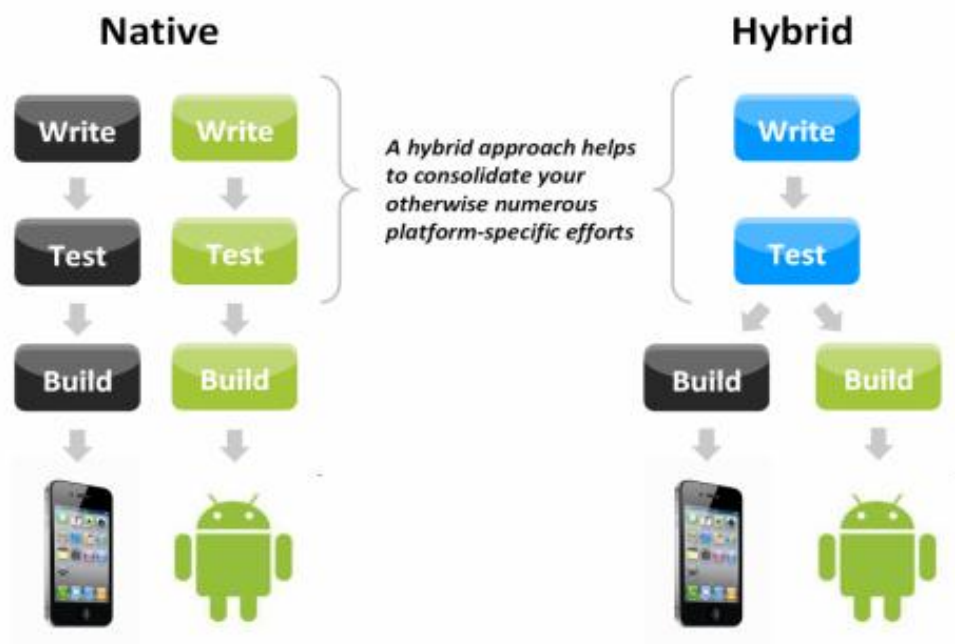
## Speed

Native applications taking advantage of their proximity to the device's underlying hardware and performance boosters like multithreading and GPU acceleration tend to perform faster when compared to HTML5 applications.

HTML5 applications are however trying to catch up and might catch up in a few years' time due to the technological advancements in browser technologies. HTML5's multithreading web workers and mobile web browser's powerful JavaScript engines might also make this a possibility /66/ /70/.

## 4.4. Development Cycle

Developing native applications are more demanding and more expensive when compared to the development of HTML5 applications. The image below gives an overview of the differences between developing multiple native applications and developing a HTML5 application.



**Fig 16. Image showing the major differences between multiple native application development cycle and hybrid (HTML5) application development cycle /73/**

Supporting just one native platform might not be a good marketing strategy, so for native applications to do well, at least two of the major OS platforms (*android, iOS, Windows Phone and blackberry*) should be supported. This brings a significant amount of overhead as other factors like testing and maintenance which includes updates and patches has to be considered. This also results in large chunks of multiple code-bases which will be managed by multiple teams of developers; this cost a significant amount and is therefore not cost-effective. The development cycle of native applications is therefore a very long, expensive and complicated process.

However, HTML5 applications are written just once and can be deployed to as many platforms as possible. This means there is only one code-base to be managed, making the whole development cycle as simple as it can be. HTML5 presents a cost-effective solution because it eliminates the overhead that comes with developing for many platforms. Another advantage is that the cost of employing professionals and the time the development and maintenance process takes is drastically reduced /69//73/.

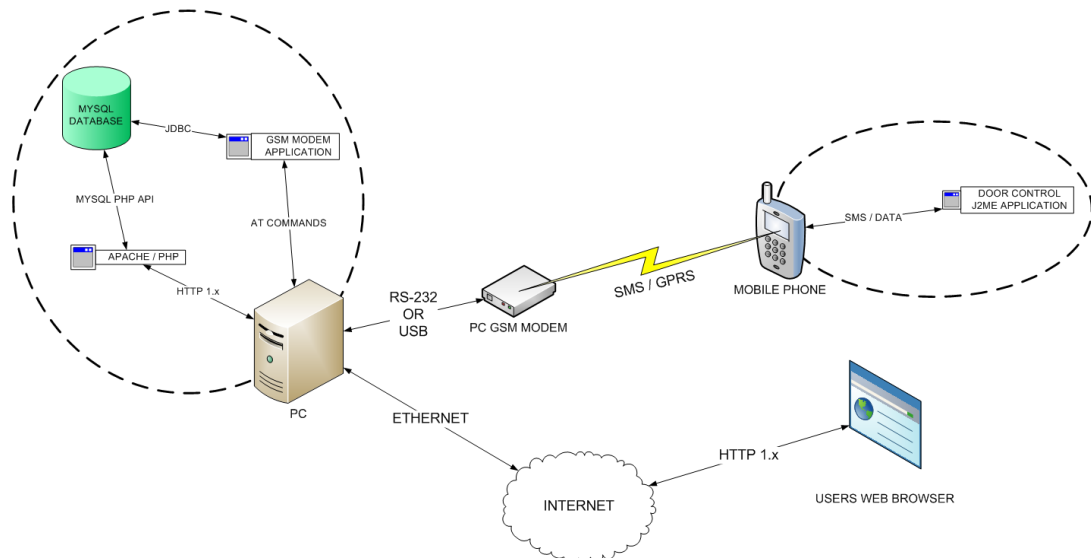
*Summary: It can therefore be deduced from the arguments that it is not about which is best, i.e. the argument of comparing HTML5 applications vs. native applications might not be valid after all. A more concise argument would be based on the purpose of the application, for instance if a rich user experience is the prime target e.g. games, then going native suits that situation but if the application is one where an explicit user experience is not needed and is meant to be cross-platform, then HTML5 applications suits that situation best.*

## 5. IMPLEMENTATION AND RESULTS

### 5.1. Background and History

During the penultimate semester (autumn 2011) of studies at school came a project that would later form a large part of this work. The aim of the project was to create a system which incorporated a mobile client with a GPRS modem to form a Security tool. The main purpose of the system was to remotely control electric door locks by sending commands from mobile devices to a server application which in turn carries out the command and sends responses back to the mobile device. An assigned task which was successfully designed and implemented was the task of developing the mobile application.

More attention would be paid to just the mobile application part of the project, a project which also included a database implementation, a web application and a server application. For more information on the project's description check APPENDIX 1.



**Fig 17. System Architecture**

The mobile application was to have at least the following features, but since the scrum development method was used these features were developed into more explicit functions. The basic features the customer required were:

- The application should be a java application (It was coded using J2ME)
- The application should support sending and receiving messages via GPRS data transfer and SMS.
- Simple text-based UI for control. Users must be able to open / close doors and see door statuses.

### 5.1.1. Design and Implementation

#### Statechart Diagram

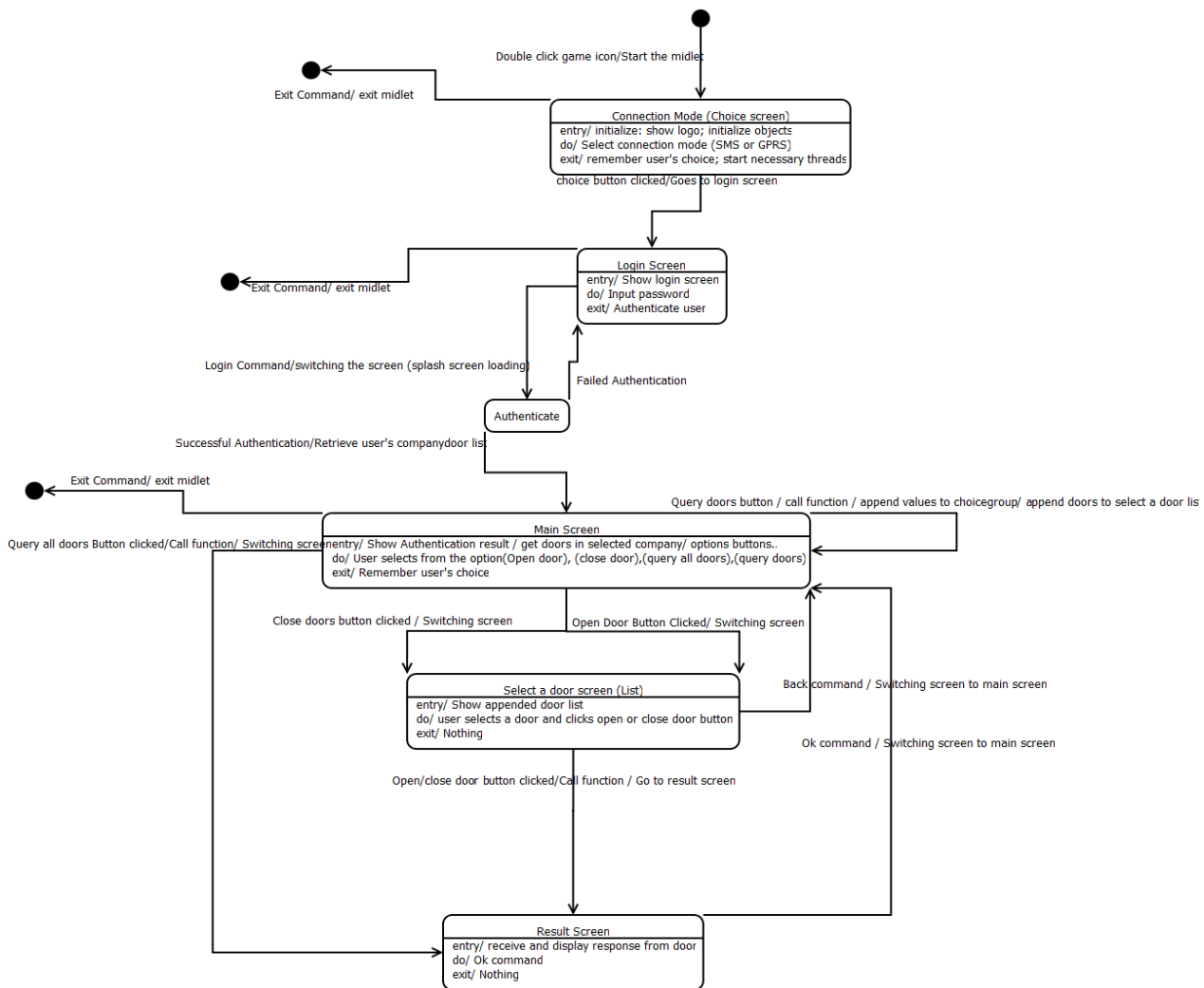
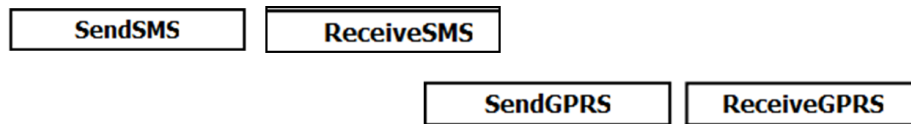


Fig 18. Mobile Application's Statechart Diagram

The diagram above shows the proposed application flow which gave a blueprint-like clue of what the GUI would look like.

### Class Diagram



**Fig 19. Mobile Application's Class Diagram**

The application had 4 classes as shown in fig 19 above. *SendSMS* class deals with the SMS part of the application which sends SMS messages (commands) to the PC (or Server) application through the modem. *ReceiveSMS* class receives responses from the server application. The *SendGPRS* sends commands through the GPRS network while the *ReceiveGPRS* receives messages through a GPRS network. Detailed class attributes and methods are shown in APPENDIX 2.

### 5.1.2. SMS

SMS also known as text-messaging is a digital cellular network feature that allows short text and numeric messages to be exchanged between mobile phone devices, from PCs to mobile phones through SMS messaging gateways on the internet /37/.

AT commands are synonymous with SMS communication. AT commands are used in modems and GSM phones to enhance SMS communication. AT commands most commonly used includes:

- AT+CMGS (used to send message)
- AT+CMSS (used to send message from storage),
- AT+CMGL (used to list messages)
- AT+CMGR (used to read message)

Though a single SMS is limited to just 160 characters it has however grown to be an integral part of communication in the world due to its convenience and cost effectiveness /38/.

### 5.1.3. GPRS

GPRS is a non-voice 2.5G technology upgrade that was added to existing TDMA networks which is the underlying transport mechanism used by GSM networks. GPRS transmits data packets over existing cellular networks /39/.

GPRS which extends the capabilities of the GSM packet switched data provides services which include SMS, continuous internet access, point to point service, instant messaging and etc /40/.

### 5.1.4. Hardware

RhoElements currently supports just motorola devices; therefore the application would be built for motorola devices. The developed application would be tested on two devices as specified by the thesis commissioner, testing would be done on either Motorola MC65 or Motorola ES400. Both devices run Windows Mobile /41/ /42/.



Motorola MC65



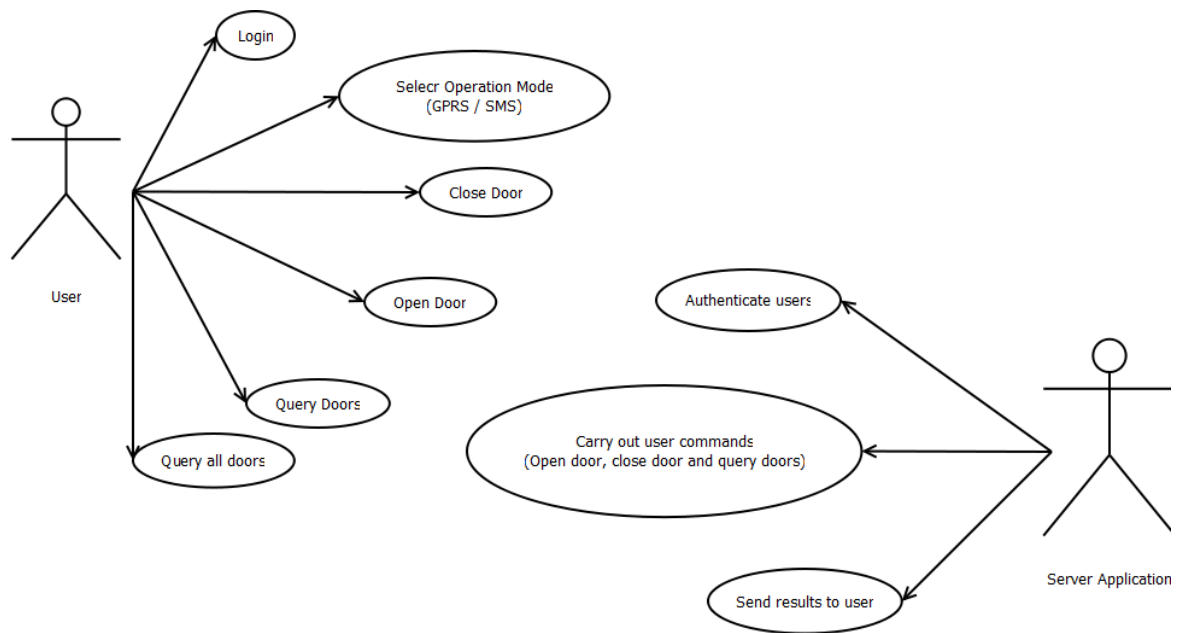
Motorola ES400

**Fig 20. Motorola MC65 and Motorola ES400 /41/ /42/**



## 5.2. HTML5 Application Design

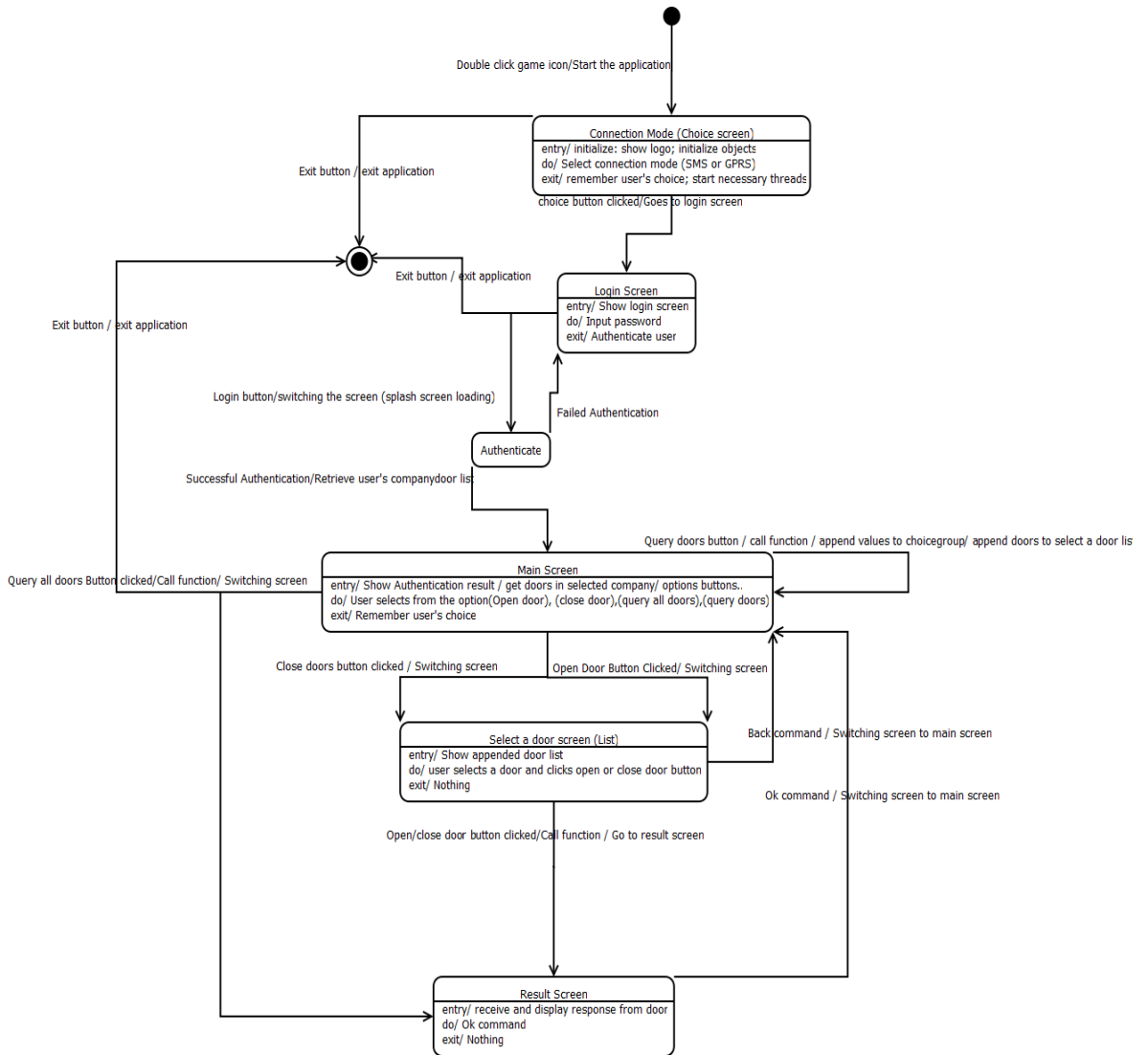
### 5.2.1. Use Case Diagram



**Fig 21. HTML5 application's use case diagram**

The diagram above is quite self-explanatory; it diagrammatically explains the actions carried out by both actors involved. The diagram depicts two actors but focus is, however, placed on just the **user** who makes use of the mobile application. The user can perform six actions with the application; they include logging in, selecting the operation mode (SMS or GPRS), opening a door, closing a door and querying door statuses. The **server application** which less focus is placed on authenticates users, carries out the user's commands and sends results to the user.

### 5.2.2. Statechart Diagram

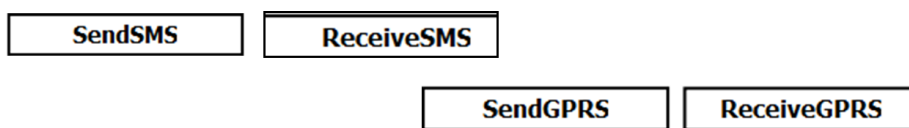


**Fig 22. HTML5 application's statechart diagram**

The diagram above shows the different states the application would possess. It also gives a clue to the application flow and to what the graphical user interface would look like. The states include the connection mode state, the login screen state, the main screen state, the select a door list state and the result screen state. Each state is triggered by an action and that state triggers another action which leads to another state.

### 5.2.3. Class Diagram

The class diagram keeps its original structure. A decision against changing the class diagram was made, due to the application's future development even though focus would be placed on designing the GUI for now. This would serve as a blueprint for the application's future and make further development easier due to careful planning. The detailed class diagram is shown in APPENDIX 2.



**Fig 23. Mobile Application's Class Diagram**

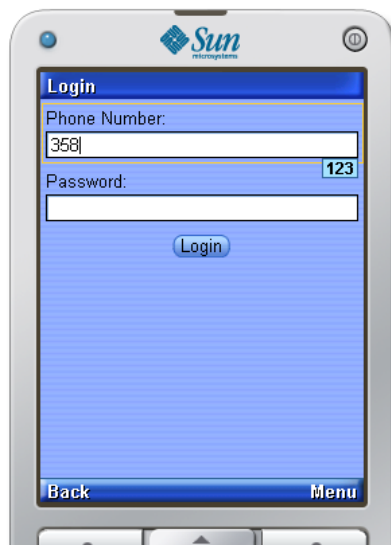
### 5.3. GUI Design

This is the most important part of the implementation part of this work. After the java-based version of this application was completed, the development team agreed that what needed to be worked on the most was the GUI and that is exactly what has been done. This also happened to be the most successful and almost only successful part of the implementation process.

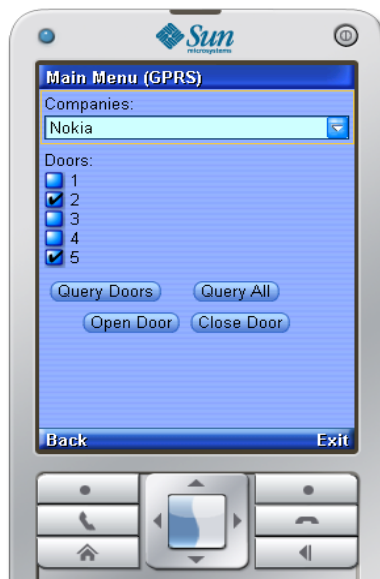
### 5.3.1. Old GUI Look (In J2ME)



**Fig 24. Connection Mode Choice Screen**



**Fig 25. Login Screen**



**Fig 26. Main Menu**

The figures above show results from the J2ME version of the application which as explained earlier was developed as a school project. The program was run on a MIDP-based simulator which worked perfectly but when transferred to a mobile device, the performance declined and the GUI became irritable and inconsistent.

These problems prompted thoughts of the write once and run on many solution, a solution that would run across various mobile devices and, at the same time, maintain its look and performance across these devices.

### 5.3.2. New GUI Look (HTML5)

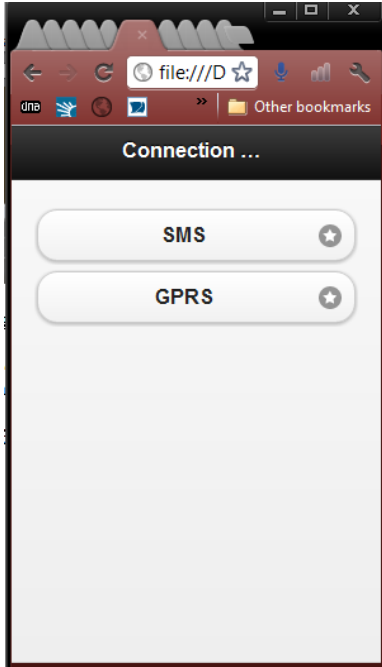


Fig 27. Connection Mode's Choice Screen

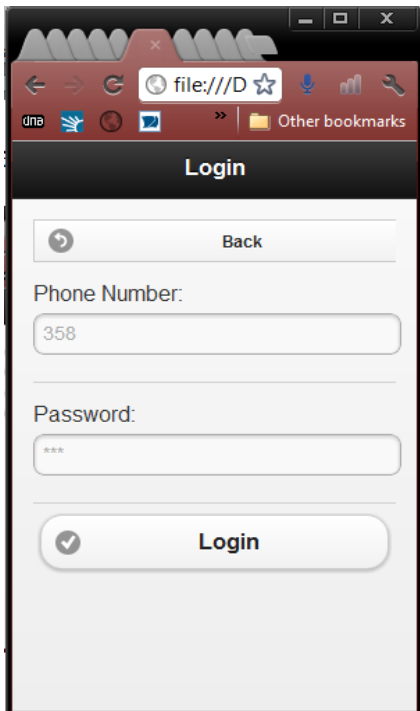
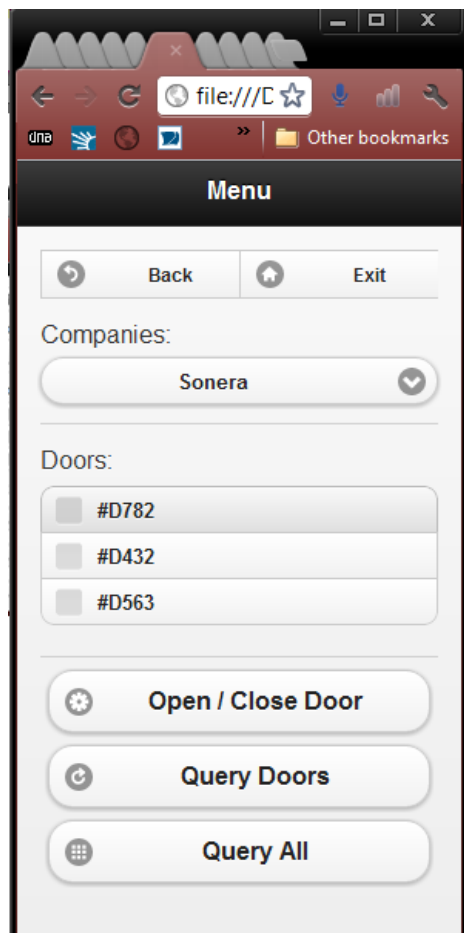


Fig 28. Login Screen



**Fig 29. Main Menu**

The figures above display the HTML5 mobile application's graphical user interface which was developed using jQueryMobile which is a unified, HTML5-based graphical user interface system that is built on the jQuery and jQuery UI foundation /90/.

When the old GUI is compared with the new GUI, there are great discrepancies, discrepancies which point to the positive side of the line. These differences include a better look and feel, better GUI performance and better responsiveness.

The application was simulated using Google chrome browser and was also tested on the Motorola MC65 device via RhoElements. The results achieved were astonishing, astonishing in the sense that the GUI was fast, smooth and responsive.

## 6. FURTHER DEVELOPMENT

The application did not get near the stage of perfection. This in a way opens up the opportunity to further develop the application and therefore make it functional. The current end result is a prototype GUI which is currently static and non-functional. That does not mean the implementation part of this project was unsuccessful, it was quite successful because much was achieved in a very limited amount of time and also considering the difficulties faced during the course of this work process e.g. licensing RhoElements.

A blueprint for further development has been designed: the statechart diagram, the use case diagram and the class diagram all serve as a guide to further develop the application. As mentioned earlier, the application is currently non-functional. It can however be improved by creating the proposed classes, linking them and making them functional. The communication mode between the application and the server should also be addressed. For the SMS communication mode the SMS library should be consistent and should function more efficiently. Furthermore, a server side implementation which supports web sockets should be implemented for the GPRS communication mode.

Due to the incomplete nature of HTML5 some features are not yet standardized e.g. when using web sockets, one has to use special server side implementations. That was an issue as the ready-made java-based server side implementation which was used during the project could not be used for this work. Implementations such as Node.js (Socket.IO, WebSocket-Node, ws), Java (Jetty), Ruby (EventMachine), python (pywebsocket, Tornado), Erlang (Shirasu), C++ (libwebsockets) and .NET (SuperWebSocket) can be used on the server side /55/.

The application also has the potential of being a commercial application if IMSS Oy decides to make it commercial. The idea package alone makes it a capable application within the software industry and that put together with its sleek GUI and its functionality. It might be an invaluable asset to security firms and even individuals.



## 7. CONCLUSIONS

HTML5 is a pretty new concept which happens to be a hot talking point due to expectations placed on it but there is surprisingly very limited number of written materials to get information from, one had to largely depend on the internet for scraps of information. This however makes this work an almost invaluable asset which is made up of some very serious research work combined with a valid and relatively conclusive argument of a topic that has been an issue from day one (i.e. the comparison between HTML5 apps and native apps), thereby bringing together a paper that can be impeccably referred to as a research masterpiece.

Even though this work could be termed as a success, it came with some challenges some of which were solved and some of which remained a problem. Time was a constant menace, a menace which led to the implementation part of this work coming short of its objectives. Though something was in this aspect achieved from a radical point of view, it however still came short of expectations due to the limited time limit. Another issue which rocked the work process was the RhoElements licensing issue. One could not use the RhoElements development platform un-licensed, it took a pretty long time to license the product. The licensing process though in the end was not successful, this however took valuable time and even posed a threat to the successful completion of this project. Due to the time issue, one had to settle for a light-weight static application which does nothing but navigates through pages and has to be further developed to achieve a standalone application status. Another issue was lack of materials as mentioned in the previous paragraph, but this did not do too much of a damage due to thorough research of materials from the web.

The objectives of this work was fully accomplished: HTML5 and its special features was thoroughly researched, HTML5 applications was compared with native applications from a neutral point of view and a light-weight application was designed and implemented, an application which would be considered for further development.

In conclusion, it is a pleasing experience to have been given this chance by IMSS Oy. One was given the chance to explore what has been learnt over the years, one was able to

manage the project process in an efficient manner and one matured as an individual during the course of this project process. One is ready to continue as a professional with an interest in HTML5 because from a personal point of view it is currently the best “write one run many solution” and the *future of mobile applications*.

## 8. REFERENCES

- /1/ Wikipedia, HTML5, [WWW-document], [<http://en.wikipedia.org/wiki/HTML5>] 15 March 2012
- /2/ Shai Neil, One Billion HTML5 Phones to be Sold Worldwide in 2013, [WWW-document], [<http://www.strategyanalytics.com/default.aspx?mod=pressreleaseviewer&a0=5145>], 7 December 2011
- /3/ Chrome Developer Relations, HTML5: Web development to the next level, [WWW-document], [<http://slides.html5rocks.com/#formula-intro-slide>], July 2011
- /4/ html5rocks, Why HTML5 rocks, [WWW-document], [<http://www.html5rocks.com/en/why>], 25 February 2012
- /5/ WHATWG Wiki Page, FAQ, [WWW-document], [<http://wiki.whatwg.org/wiki/FAQ>], 2 March 2012
- /6/ W3C, HTML5 logo, [WWW-document], [<http://www.w3.org/html/logo/>], 29 February 2012
- /7/ W3C, HTML5: A vocabulary and associated APIs for HTML and XHTML, [WWW-document], [<http://dev.w3.org/html5/spec/Overview.html#history-1>], 15 March 2012
- /8/ Chrome Developer Relations, HTML5: Web development to the next level, [WWW-document], [<http://slides.html5rocks.com/#timeline-slide>], July 2011
- /9/ Dumas Lab, Best Mobile Web HTML5 Framework For Mobile App Development, [WWW-document], [<http://dumaslab.com/2011/05/best-mobile-web-html5-framework-for-mobile-app-development/>], 1 May 2011

- /10/ Wikipedia, Mobile Application Development, [WWW-document],  
[[http://en.wikipedia.org/wiki/Mobile\\_application\\_development](http://en.wikipedia.org/wiki/Mobile_application_development)], 12 March 2012
- /11/ O'Dell Jolie, 5 Cross-Platform Mobile Development Tools You Should Try, [WWW-document], [<http://mashable.com/2010/08/11/cross-platform-mobile-development-tools/>], 11 August 2010
- /12/ CSS Reflex, Best 7 HTML5 Frameworks for Mobile Apps, [WWW-document], [<http://www.cssreflex.com/2012/02/7-html5-frameworks-mobile-apps.html/>], 2 March 2012
- /13/ Wikipedia, Frameworks for app development, [WWW-document], [[http://en.wikipedia.org/wiki/HTML5\\_in\\_mobile\\_devices#Frameworks\\_for\\_app\\_development](http://en.wikipedia.org/wiki/HTML5_in_mobile_devices#Frameworks_for_app_development)], 5 March 2012
- /14/ Motorola, RhoElements: Web Based Application Development Framework, [WWW-document], [<http://www.motorola.com/Business/US-EN/Business+Product+and+Services/Software+and+Applications/Application+Development+Framework/RhoElements>], 7 March 2012
- /15/ Motorola, RhoElements Brochure, [WWW-document], [[http://www.motorola.com/web/Business/Products/Software%20and%20Applications/Application\\_Development\\_Framework/RhoElements/Documents/StaticFiles/RhoElements-Application-Development-Framework-Brochure.pdf](http://www.motorola.com/web/Business/Products/Software%20and%20Applications/Application_Development_Framework/RhoElements/Documents/StaticFiles/RhoElements-Application-Development-Framework-Brochure.pdf)], 7 March 2012
- /16/ MobiThinking, Mobile applications: native v Web apps – what are the pros and cons?, [WWW-document], [<http://mobithinking.com/native-or-web-app>], 7 March 2012
- /17/ PC Mag, Definition of: native application, [WWW-document], [[http://www.pcmag.com/encyclopedia\\_term/0,2542,t=native+application&i=47651,00.asp](http://www.pcmag.com/encyclopedia_term/0,2542,t=native+application&i=47651,00.asp)], 8 March 2012

- /18/ Wikipedia, Android (operating system), [WWW-document],  
[\[http://en.wikipedia.org/wiki/Android\\_\(operating\\_system\)#Foundation\]](http://en.wikipedia.org/wiki/Android_(operating_system)#Foundation), 14 March 2012
- /19/ Android, Android, the world's most popular mobile platform, [WWW-document],  
[\[http://www.android.com/about/\]](http://www.android.com/about/), 5 February 2012
- /20/ Android Developers, What is Android?, [WWW-document],  
[\[http://developer.android.com/guide/basics/what-is-android.html\]](http://developer.android.com/guide/basics/what-is-android.html), 8 March 2012
- /21/ Android Developers, Application Fundamentals, [WWW-document],  
[\[http://developer.android.com/guide/topics/fundamentals.html\]](http://developer.android.com/guide/topics/fundamentals.html), 8 March 2012
- /22/ Wikipedia, iOS, [WWW-document], [\[http://en.wikipedia.org/wiki/IOS\]](http://en.wikipedia.org/wiki/IOS), 13 March 2012
- /23/ iOS Developer Library, iOS Technology Overview, [WWW-document],  
[\[https://developer.apple.com/library/ios/#DOCUMENTATION/Miscellaneous/Conceptual/iPhoneOSTechOverview/IPhoneOSOverview/IPhoneOSOverview.html\]](https://developer.apple.com/library/ios/#DOCUMENTATION/Miscellaneous/Conceptual/iPhoneOSTechOverview/IPhoneOSOverview/IPhoneOSOverview.html), 12, October 2011
- /24/ Hunt Lachlan, A preview of HTML5, [WWW-document],  
[\[http://www.alistapart.com/articles/previewofhtml5\]](http://www.alistapart.com/articles/previewofhtml5), 4 December 2007
- /25/ W3C, Cascading Style Sheets, [WWW-document], [\[http://www.w3.org/Style/CSS/\]](http://www.w3.org/Style/CSS/), 13 March 2012
- /26/ Wikipedia, CSS, [WWW-document],  
[\[http://en.wikipedia.org/wiki/Cascading\\_Style\\_Sheets\]](http://en.wikipedia.org/wiki/Cascading_Style_Sheets), 6 March 2012
- /27/ HTML.net, What is CSS?, [WWW-document],  
[\[http://www.html.net/tutorials/css/lesson1.php\]](http://www.html.net/tutorials/css/lesson1.php), 9 March 2012

- /28/ Lung Chad, Embedding fonts with CSS for your next HTML5 application, [WWW-document], [<http://www.giantflyingsaucer.com/blog/?p=2250>], 22 February 2011
- /29/ Mozilla Developer Network, About JavaScript, [WWW-document], [[https://developer.mozilla.org/en/JavaScript/About\\_JavaScript](https://developer.mozilla.org/en/JavaScript/About_JavaScript)], 6 October 2011
- /30/ Crockford Douglas, JavaScript: The World's Most Misunderstood Programming Language, [WWW-document], [<http://javascript.crockford.com/javascript.html>], 9 March 2012
- /31/ Oracle, What is JavaScript and how is it different from Java Technology?, [WWW-document], [[http://java.com/en/download/faq/java\\_javascript.xml](http://java.com/en/download/faq/java_javascript.xml)], 9 March 2012
- /32/ Wikipedia, JavaScript, [WWW-document], [<http://en.wikipedia.org/wiki/Javascript>], 1 March 2012
- /33/ Mozilla Developer Network, JavaScript Overview, [WWW-document], [[https://developer.mozilla.org/en/JavaScript/Guide/JavaScript\\_Overview](https://developer.mozilla.org/en/JavaScript/Guide/JavaScript_Overview)], 21 October 2011
- /34/ W3Schools, JavaScript How To, [WWW-document], [[http://www.w3schools.com/js/js\\_howto.asp](http://www.w3schools.com/js/js_howto.asp)], 9 March 2012
- /35/ Wikipedia, JavaScript, [WWW-document], [[http://en.wikipedia.org/wiki/Windows\\_Phone](http://en.wikipedia.org/wiki/Windows_Phone)], 15 March 2012
- /36/ Microsoft, Windows Phone 7 Platform introduced to iPhone application developers, [WWW-document], [<http://windowsphone.interoperabilitybridges.com/articles/chapter-1-windows-phone-7-platform-introduced-to-iphone-application-developers>], 1 January 2011

- /37/ ZdNet.com, Short Messaging Service (SMS) explained, [WWW-document], [[http://www.zdnet.com.au/short-messaging-service-sms-explained\\_p6-120248392.htm](http://www.zdnet.com.au/short-messaging-service-sms-explained_p6-120248392.htm)], 31 August 2001.
- /38/ Sherman Jim, SMS 101 - Short Message Service Explained, [WWW-document], [<http://jimshe.wrytestuff.com/swa12363.htm>], 26 July 2005
- /39/ Freitas M, What is GPRS?, [WWW-document], [<http://www.geekzone.co.nz/content.asp?contentid=207>], 4 February 2003.
- /40/ Wikipedia, General Packet Radio Service, [WWW-document], [[http://en.wikipedia.org/wiki/General\\_Packet\\_Radio\\_Service](http://en.wikipedia.org/wiki/General_Packet_Radio_Service)], 10 March 2012
- /41/ Motorola, Specification Sheet: MC65, [WWW-document], [<http://www.motorola.com/web/Business/Products/Mobile%20Computers/MC65/Document/StaticFiles/MC65-Spec-Sheet-0810.pdf>], 12 March 2012.
- /42/ Motorola, Specification Sheet: The Motorola Global ES400 EDA, [WWW document], [[http://www.motorola.com/web/Business/Products/Mobile%20Computers/ES400/Documents/ES400-Spec-Sheet\\_english.pdf](http://www.motorola.com/web/Business/Products/Mobile%20Computers/ES400/Documents/ES400-Spec-Sheet_english.pdf)], 12 March 2012
- /43/ W3C, Offline Web Applications, [WWW document], [<http://www.w3.org/TR/offline-webapps/#sql>], 30 May 2008
- /44/ Bidelman Eric, A beginner's guide to sing the application cache, [WWW document], [<http://www.html5rocks.com/en/tutorials/appcache/beginner/>], 27 May 2011
- /45/ Mahemoff Michael, "Offline": What does it mean and why should i care?, [WWW document], [<http://www.html5rocks.com/en/tutorials/offline/whats-offline/>], 13 October 2011

- /46/ Dive into HTML, The Past, Present & Future of Local Storage for Web Applications, [WWW document], [<http://diveintohtml5.info/storage.html>], 12 March 2012
- /47/ Green Ido, Going off the grid: HTML5 Offline, [WWW document], [<http://offline-11.appspot.com/#7>], November 2011
- /48/ Mahemoff Michael, Client-Side Storage, [WWW document], [<http://www.html5rocks.com/en/tutorials/offline/storage/>], 1 October 2010
- /49/ Devlin Ian, Finding your position with Geolocation, [WWW document], [<http://html5doctor.com/finding-your-position-with-geolocation/>], 14 June 2011
- /50/ W3C, Web Workers, [WWW document], [<http://dev.w3.org/html5/workers/#introduction>], 13 March 2012
- /51/ Bidelman Eric, The Basics of Web Workers, [WWW document], [<http://www.html5rocks.com/en/tutorials/workers/basics/>], 24 May 2011
- /52/ Mozilla Developer Network, Using geolocation, [WWW document], [[https://developer.mozilla.org/en/Using\\_geolocation](https://developer.mozilla.org/en/Using_geolocation)], 27 February 2012
- /53/ Lindkvist David, The HTML5 WebSocket API, [WWW document], [<http://www.slideshare.net/ffdead/the-html5-websocket-api>], 21 April 2010
- /54/ Chrome Developer Relations, HTML5: Web development to the next level, [WWW-document], [<http://slides.html5rocks.com/#web-sockets>], July 2011
- /55/ Ubl Malte & Eiji Kitamura, Introducing WebSockets: Bringing Sockets to the Web, [WWW-document], [<http://www.html5rocks.com/en/tutorials/websockets/basics/>], 13 March 2012



- /56/ Delgado Ernest, Using the Notifications API, [WWW-document],  
[<http://www.html5rocks.com/en/tutorials/notifications/quick/>], 24 February 2010
- /57/ Bidelman Eric, Exploring the Filesystem APIs, [WWW document],  
[<http://www.html5rocks.com/en/tutorials/file/filesystem/>], 27 January 2012
- /58/ Bidelman Eric, Native HTML5 Drag and Drop, [WWW document],  
[<http://www.html5rocks.com/en/tutorials/dnd/basics/>], 17 February 2012
- /59/ Jones Alexander, HTML5 Semantic Markup, [WWW document],  
[<http://www.html5tuts.co.uk/tutorials/semantics/>], 10 March 2011
- /60/ Chrome Developer Relations, HTML5: Web development to the next level, [WWW-document], [<http://slides.html5rocks.com/#semantic-tags-1>], July 2011.
- /61/ Lauke H. Patrick & Mills Chris, New form features in HTML5, [WWW-document],  
[<http://dev.opera.com/articles/view/new-form-features-in-html5/#newoutput>], 16  
December 2010
- /62/ Lawson Bruce, Introduction to HTML5 video, [WWW-document],  
[<http://dev.opera.com/articles/view/introduction-html5-video/>], 11 February 2010
- /63/ Boas Mark, Native Audio in the browser, [WWW-document],  
[<http://html5doctor.com/native-audio-in-the-browser/>], 29 July 2009
- /64/ Sharp Remy, An introduction to the Canvas 2D API, [WWW-document],  
[<http://html5doctor.com/an-introduction-to-the-canvas-2d-api/>], 3 August 2010
- /65/ Rowinski Dan, Hybrid HTML5 Apps Are Less Costly to Develop Than Native,  
[WWW-document], [<http://www.readwriteweb.com/mobile/2012/01/hybrid-html5-apps-are-more-les.php>], 9 January 2012

- /66/ Marshall Matt, How HTML5 will kill the native app, [WWW-document],  
[\[http://venturebeat.com/2011/04/07/how-html5-will-kill-the-native-app/\]](http://venturebeat.com/2011/04/07/how-html5-will-kill-the-native-app/), 7 April 2011
- /67/ Bottom Line Performance, mLearning devices and platforms: What you need to know,  
[WWW-document],  
[\[http://www.bottomlineperformance.com/lolblog/index.php/tag/mobile-learning/\]](http://www.bottomlineperformance.com/lolblog/index.php/tag/mobile-learning/), 31  
August 2011
- /68/ Saxena Ashutosh, HTML5 Vs. Native App Debate – Here Is What You Need To  
Know, [WWW-document], [\[http://www.pluggd.in/html5-vs-native-app-297/\]](http://www.pluggd.in/html5-vs-native-app-297/), 14  
March 2012
- /69/ Gadodia Vaibhav, HTML5 mobile apps vs native mobile apps, [WWW-document],  
[\[http://www.nagarro.com/blog/html5-mobile-apps-vs-native-mobile-apps/\]](http://www.nagarro.com/blog/html5-mobile-apps-vs-native-mobile-apps/), 28 August  
2011
- /70/ Mahemoff Michael, HTML5 vs. Native: The Mobile App Debate, [WWW document],  
[\[http://www.html5rocks.com/en/mobile/nativedebate/\]](http://www.html5rocks.com/en/mobile/nativedebate/), 3 June 2011
- /71/ Vikash, HTML5 vs. Native Apps, [WWW document],  
[\[http://microreviews.org/html5-vs-native-apps/\]](http://microreviews.org/html5-vs-native-apps/), 20 September 2011
- /72/ Worklight, HTML5, Hybrid or Native Mobile App Development, [WWW document],  
[\[http://www.worklight.com/assets/files/HTML5,%20Hybrid%20or%20Native%20Mo  
bile%20App%20Development.pdf\]](http://www.worklight.com/assets/files/HTML5,%20Hybrid%20or%20Native%20Mobile%20App%20Development.pdf), 16 March 2012
- /73/ Jacobs Mike, Living on the Edge of Mobile Development, [WWW document],  
[\[http://iphone.sys-con.com/node/1719019\]](http://iphone.sys-con.com/node/1719019), 17 March 2011
- /74/ Phonegap, How PhoneGap Works, [WWW document], [\[http://phonegap.com/about\]](http://phonegap.com/about),  
16 March 2012

- /75/ Wikipedia, Comparison of web browsers, [WWW document],  
[[http://en.wikipedia.org/wiki/Comparison\\_of\\_web\\_browsers?utm\\_source=sihirlielma&utm\\_medium=cpc&utm\\_campaign=sponsorship#HTML5\\_support](http://en.wikipedia.org/wiki/Comparison_of_web_browsers?utm_source=sihirlielma&utm_medium=cpc&utm_campaign=sponsorship#HTML5_support)], 18 February 2012
- /76/ HTML5 Test, The HTML5 test – How well does your browser support HTML5?, [WWW document], [<http://html5test.com/results.html>], 16 March 2012.
- /77/ HTML5 Test, The HTML5 test – How well does your browser support HTML5?, [WWW document], [<http://html5test.com/results-mobile.html>], 16 March 2012
- /78/ Wikipedia, Web Browser, [WWW document],  
[[http://en.wikipedia.org/wiki/Web\\_browser](http://en.wikipedia.org/wiki/Web_browser)], 19 March 2012
- /79/ W3C, Application cache API, [WWW document],  
[<http://www.w3.org/TR/html5/offline.html#application-cache-api>], 26 March 2012
- /80/ Sharp Remy, Introducing Web SQL Databases, [WWW document],  
[<http://html5doctor.com/introducing-web-sql-databases/>], 26 March 2012
- /81/ Wikipedia, Web SQL Database, [WWW document],  
[[http://en.wikipedia.org/wiki/Web\\_SQL\\_Database](http://en.wikipedia.org/wiki/Web_SQL_Database)], 26 March 2012
- /82/ W3C, Geolocation API Specification, [WWW document],  
[<http://dev.w3.org/geo/api/spec-source.html>], 26 March 2012
- /83/ Wikipedia, Web worker, [WWW document],  
[[http://en.wikipedia.org/wiki/Web\\_worker](http://en.wikipedia.org/wiki/Web_worker)], 27 March 2012
- /84/ CodeDielsel, Introducing HTML5 Web Workers, [WWW document],  
[<http://www.codediesel.com/javascript/introducing-html5-web-workers/>], 27 March 2012

- /85/ Lubbers Peter & Greco Frank, HTML5 Web Sockets: A Quantum Leap in Scalability for the Web, [<http://websocket.org/quantum.html>], 27 March 2012
- /86/ Google.com, Google, [WWW document], [www.google.com], 14 March 2012
- /87/ The Chromium Projects, API Specification, [WWW document], [<http://www.chromium.org/developers/design-documents/desktop-notifications/api-specification>], 28 March 2012
- /88/ Chrome Developer Relations, HTML5: Web development to the next level, [WWW-document], [<http://slides.html5rocks.com/#drag-in>], July 2011
- /89/ Chrome Developer Relations, HTML5: Web development to the next level, [WWW-document], [<http://slides.html5rocks.com/#drag-out>], July 2011
- /90/ jQueryMobile, [WWW-document], [<http://jquerymobile.com/>], April 2012

## **9. LIST OF APPENDICES**

APPENDIX 1	Mobile devices project's description
APPENDIX 2	Mobile application's class diagram

## APPENDIX 1

**Name of project/product:** Mobile Security / lock control.

**The aim of project:** To create a system this incorporates a mobile client with a GPRS modem to form a Security tool. The main purpose of the system is to remotely control electric door locks.

Security companies and guards need to get access or give access to areas that are locked. The doors usually have electric locks that can be controlled remotely. This is good when the guard has no key to the door or he/she has to give access to someone else to go through the door. The guard can open the door with remote access from mobile device.

The mobile application has to have at least the following features:

- Java application
- support for sending/receiving via GPRS data transfer / SMS
- Mobile UI for door control, must see lock status (open / closed)

The web client/database/server should have at least the following

- user management (id and password)

Other demands:

- Has to work with GSM modem which controls the door electrics.

## APPENDIX 2

Mobile application's class diagram which shows the detailed class attributes and methods.

