



**UNIVERSITY OF PIRAEUS**

**Digital Systems Department**

**Postgraduate Program**

**“Technology Education & Digital Systems”**

**Network-Oriented Systems**

**MSc Thesis Project**

**“Cross Platform Mobile Application for Mobile Devices with  
Social Media and Geolocation Services”**

Odysseas Tigas  
Student.Id : ME11071

Supervisor Professor: Dr. Marinos Themistokleous

Piraeus 2013

**Dedicated to my family**

Πανεπιστήμιο Πειραιώς

## Acknowledgments

I thank my supervisor professor of Digital Systems department of University of Piraeus, Dr. Marinos Themistokleous for guiding me through the project and encouraging me to use new and innovating software tools. I also thank Professor of Digital Systems and Vice Rector of the University of Piraeus Dr. Georgios Vasilakopoulos for providing me the methods and the inspiration to carry on through the duration of my studies.

Πανεπιστήμιο Πειραιώς

# “Cross Platform Mobile Application for mobile devices with social media and geolocation services”

**Odysseas Tigas**

*Keywords:* **native mobile applications, cross platform mobile applications, fragmentation, social media, geolocation services.**

**Abstract:** Most of the modern mobile devices manufacturers are using different operating systems. One of the manufacturers’ targets is to cover the needs of their users by providing a vast number of applications charged or not in an online market. The ability for the individual developers and the mobile applications developing organizations to make profit can be done by charging an application or adding advertisement modules in the code. There are two ways to develop mobile applications: The first way is the native way of programming and the second way is to develop an application in a cross platform fashion. The former way requires a specific IDE to use and a specific programming language in order to develop applications, while the later way is using web techniques such as the web browser of the mobile device and web based languages like java script, html, and CSS. The issue that developers have to address is to develop applications for more than one platform in order to avoid the fragmentation of the mobile operating systems. The cross platform developing can solve this issue with an efficient and economical way. Organizations associated with mobile developing require specialized staff for every platform with the native way, unlike the second way in which the code is common for all platforms. In addition, it is very useful to develop in a cross platform fashion because the target group of users is specific but the type of the devices they use may differ. Finally with the cross platform fashion the developing time and the cost are reduced. In this postgraduate thesis project we are going to illustrate the latest cross platform tools and compare them to export useful conclusions about their advantages and disadvantages. Next we are going to suggest a cross platform tool in order to build an application step by step, using social media and geolocation services.

## Table of Contents

| Chapter  | Heading  | Page |
|----------|--|------|
| <b>1</b> | <b>Introduction</b>  | 11   |
| <b>2</b> | <b>Problem Description</b>                                   | 11   |
| 2.1      | Problem Statement  | 11   |
| 2.2      | Goals  | 11   |
| 2.3      | Purposes   | 12   |
| 2.4      | Methods  | 12   |
| <b>3</b> | <b>Method</b>  | 12   |
| 3.1      | About The Methodology  | 12   |
| 3.2      | Roles  | 13   |
| 3.3      | XP LifeCycle   | 13   |
| <b>4</b> | <b>Existing Technologies &amp; Practices</b>                 | 14   |
| 4.1      | Introduction   | 14   |
| 4.2      | What is a Native Application                                 | 14   |
| 4.2.1    | Programming Language – IDE for Android OS                    | 15   |
| 4.2.2    | Programming Language – IDE for Apple iOS                     | 16   |
| 4.2.3    | Programming Language – IDE for Blackberry OS                 | 16   |
| 4.2.4    | Programming Language – IDE for Windows Phone                 | 17   |
| 4.2.5    | Setting up the Android ADT Bundle and Running an Application | 17   |
| 4.3      | What is a Cross Platform Application?                        | 26   |
| 4.4      | Native and Cross Platform Comparison                         | 27   |
| 4.5      | Cross Platform Tools Disadvantages                           | 29   |
| 4.6      | Existing Cross Platform Tools                                | 30   |
| 4.6.1    | Appcelerator Titanium  | 30   |
| 4.6.2    | RhoMobile  | 31   |
| 4.6.3    | PhoneGap   | 33   |
| 4.6.4    | Senca Touch  | 34   |
| 4.6.5    | Other Cross Platform Frameworks                              | 35   |
| 4.7      | Cross Platform Tools Comparison                              | 36   |
| 4.8      | Conclusions About Frameworks                                 | 37   |
| <b>5</b> | <b>Requirement Studies</b>                                   | 37   |
| 5.1      | Feasibility Study  | 37   |
| 5.1.1    | Technical Factor   | 38   |
| 5.1.2    | Economic Factor  | 38   |
| 5.1.3    | Legal Factor   | 38   |
| 5.1.4    | Organizational Factor  | 38   |
| 5.1.5    | Scheduling Factor  | 39   |
| 5.2      | Tool Selection   | 39   |
| 5.2.1    | JavaScript   | 40   |
| 5.2.2    | HTML5  | 41   |
| 5.2.3    | CSS  | 42   |

|          |   |     |
|----------|---|-----|
| 5.2.4    | The Progressive Enchantment of Web Technologies   | 43  |
| 5.3      | Tool Installation                                 | 45  |
| 5.3.1    | Download The Tool                                 | 46  |
| 5.3.2    | Open Eclipse – Create Project                     | 47  |
| 5.3.3    | Configuring Project for PhoneGap                  | 55  |
| 5.3.4    | PhoneGap Update Activity Class                    | 61  |
| 5.3.5    | Configure the Project Metadata                    | 63  |
| 5.3.6    | Running an Application                            | 66  |
| 5.4      | Conclusions About Chapter Five                    | 69  |
| <b>6</b> | <b>Application Development</b>                    | 70  |
| 6.1      | Introduction                                      | 70  |
| 6.1.1    | The Application Description                       | 71  |
| 6.1.2    | The Web Diagrams                                  | 72  |
| 6.1.3    | Dummy Screens                                     | 74  |
| 6.2      | Social Media & Geolocation Services               | 75  |
| 6.2.1    | Social Media Services                             | 75  |
| 6.2.2    | Geolocation Services                              | 75  |
| 6.3      | Application Functionality                         | 76  |
| 6.3.1    | The Local Login (SQLite)                          | 76  |
| 6.3.2    | Cloud Registration and Coordinates (Database.com) | 76  |
| 6.3.3    | The ChildBrowser Plugin                           | 94  |
| 6.4.4    | The Social Media Section                          | 96  |
| <b>7</b> | <b>Results</b>                                    | 101 |
| 7.1      | The finished Application                          | 101 |
| 7.2      | Deploy on other platforms                         | 109 |
| 7.3      | Limitations                                       | 117 |
| 7.4      | Future Work                                       | 117 |
|          | <b>References</b>                                 | 118 |
|          | <b>Web References</b>                             | 119 |
|          | <b>Annex (Code)</b>                               | 123 |

## Table of Figures

| Figure name                      | Page |
|----------------------------------|------|
| Figure 1 Download the SDK        | 17   |
| Figure 2 Terms and Conditions    | 18   |
| Figure 3 Save the ADT Bundle     | 19   |
| Figure 4 Extract ADT Bundle      | 19   |
| Figure 5 Extract ADT Bundle II   | 20   |
| Figure 6 Open Eclipse            | 20   |
| Figure 7 Select a Workspace      | 21   |
| Figure 8 The Eclipse Environment | 21   |
| Figure 9 Android SDK manager     | 22   |
| Figure 10 Importing Project      | 23   |

|  |    |
|--|----|
| Figure 11 Project Loaded                                       | 23 |
| Figure 12 Manipulating graphical elements                      | 24 |
| Figure 13 Android Device Chooser                               | 25 |
| Figure 14 Finished Android Application                         | 26 |
| Figure 15 The three types of applications                      | 27 |
| Figure 16 RhoMobile MVC model                                  | 32 |
| Figure 17 Graphical representation of progressive enchancement | 44 |
| Figure 18 Download PhoneGap                                    | 46 |
| Figure 19 Extract PhoneGap                                     | 47 |
| Figure 20 Select a workspace                                   | 48 |
| Figure 21 The Known eclipse Environment                        | 48 |
| Figure 22 Create new Project                                   | 49 |
| Figure 23 Set project name                                     | 50 |
| Figure 24 Final confirm  | 51 |
| Figure 25 Configure launcher icon                              | 52 |
| Figure 26 Create activity                                      | 53 |
| Figure 27 Black Activity                                       | 54 |
| Figure 28 Package explorer                                     | 55 |
| Figure 29 New folder   | 56 |
| Figure 30 Important folders                                    | 57 |
| Figure 31 Files Placement                                      | 58 |
| Figure 32 www new file   | 59 |
| Figure 33 Open text editor                                     | 60 |
| Figure 34 Add to build Path                                    | 61 |
| Figure 35 Changes in HelloGap                                  | 63 |
| Figure 36 Open AndroidManifest                                 | 64 |
| Figure 37 Application Run                                      | 66 |
| Figure 38 Run Virtual  | 67 |
| Figure 39 Run Result   | 67 |
| Figure 40 Physical Result                                      | 68 |
| Figure 41 Cross Platform Demo Finished                         | 69 |
| Figure 42 Application Use Case Diagram                         | 70 |
| Figure 43 Transition Diagram                                   | 73 |
| Figure 44 WEB-WS-DB Diagram                                    | 73 |
| Figure 45 SOA Diagram  | 74 |
| Figure 46 Dummy screens I                                      | 74 |
| Figure 47 Dummy screens II                                     | 75 |
| Figure 48 Database.com sign Up                                 | 77 |
| Figure 49 create new object                                    | 78 |
| Figure 50 New Custom object                                    | 78 |
| Figure 51 New relationships                                    | 79 |
| Figure 52 Remote Access step 1                                 | 80 |
| Figure 53 Remote Access step 2                                 | 80 |
| Figure 54 Remote Access Detail                                 | 81 |
| Figure 55 Child Browser files Placement                        | 95 |

|   |     |
|---|-----|
| Figure 56 Child Browser Style                         | 96  |
| Figure 57 facebook developers create account          | 96  |
| Figure 58 New Application Creation                    | 97  |
| Figure 59 Application information                     | 98  |
| Figure 60 clone git                                   | 99  |
| Figure 61 update git                                  | 99  |
| Figure 62 The launcher icon in android menu           | 101 |
| Figure 63 splash and functionalities                  | 102 |
| Figure 64 First and second functionality.             | 103 |
| Figure 65 functionality of the second option I        | 104 |
| Figure 66 functionality of the second option II       | 105 |
| Figure 67 New Record and edit record with coordinates | 106 |
| Figure 68 Show records                                | 107 |
| Figure 69 Third option and The login result           | 108 |
| Figure 70 get started – build                         | 109 |
| Figure 71 Register with adobe ID                      | 110 |
| Figure 72 plan selection                              | 111 |
| Figure 73 file upload from GIT                        | 111 |
| Figure 74 File uploading                              | 112 |
| Figure 75 name options for our file                   | 112 |
| Figure 76 download our project in various formats     | 111 |
| Figure 77 upload new apk to production                | 113 |
| Figure 78 successful attempt to upload on Google play | 114 |
| Figure 79 Upload our xap file                         | 115 |
| Figure 80 App info                                    | 116 |
| Figure 81 submission successful                       | 116 |



## Table of Tables

| <b>Table name</b>   | <b>Page</b> |
|---|-------------|
| Table 1 Platform General Information                            | 15          |
| Table 2 xml layout file   | 24          |
| Table 3 Java Code Sample  | 25          |
| Table 4 Characteristics of types of applications                | 28          |
| Table 5 Other important characteristics                         | 28          |
| Table 6 Features of Appcelerator Titanium                       | 31          |
| Table 7 Cross platform software support                         | 36          |
| Table 8 TELOS factors   | 37          |
| Table 9 Software licenses                                       | 38          |
| Table 10 scheduling documents                                   | 39          |
| Table 11 JS sample  | 41          |
| Table 12 HTML sample  | 42          |
| Table 13 CSS sample   | 43          |
| Table 14 Software web sites                                     | 45          |
| Table 15 index.html hello world                                 | 61          |
| Table 16 import org.apache.cordova.DroidGap;                    | 62          |
| Table 17 public class HelloGapActivity extends DroidGap {       | 62          |
| Table 18 super.loadUrl("file:///android_asset/www/index.html"); | 62          |
| Table 19 manifest.xml support screens                           | 64          |
| Table 20 manifest.xml uses permission                           | 65          |
| Table 21 manifest.xml configChanges                             | 65          |
| Table 22 manifest.xml activity                                  | 66          |
| Table 23 use case diagram codes                                 | 71          |
| Table 24 Consumer key   | 81          |
| Table 25 Scripts in index.html                                  | 82          |
| Table 26 Deviceready ondevice readyfunctions                    | 83          |
| Table 27 SalesforceWrapper function.                            | 83          |
| Table 28 swf.login  | 85          |
| Table 29 Moustache.js   | 85          |
| Table 30 setupdefaultView                                       | 86          |
| Table 31 HTML templates structure                               | 87          |
| Table 32 resetcontainer and setuphome functions                 | 87          |
| Table 33 home tamplate  | 88          |
| Table 34 SetupFormView  | 88          |
| Table 35 The formtamplate                                       | 89          |
| Table 36 saveformdata function                                  | 90          |
| Table 37 setupListview function                                 | 91          |
| Table 38 datacontainer element                                  | 92          |
| Table 39 Function Queryrecords                                  | 92          |
| Table 40 Function renderlistData                                | 93          |
| Table 41 listitem template                                      | 93          |

|  |    |
|--|----|
| Table 42 onListItemTap getrecordbyId   | 93 |
| Table 43 Cofig.xml                     | 95 |
| Table 44 Plugin.xml                    | 95 |
| Table 45 childbrowser script reference | 95 |
| Table 46 App_ID                        | 98 |

Πανεπιστήμιο Πειραιώς

## 1. Introduction

Mobile devices can help us almost in every sector of our modern world. According to some authors, the mobile devices appearance is a big revolution in human evolution [1] because their appearance in our lives behaves somehow like a new artificial organ with extra functionalities. In the early nineties mobile devices were used only for voice and text communication, but today the majority of the devices are capable of sophisticated functions similar to a modern desktop computer plus the ability of mobility. In this postgraduate thesis project we discuss the elements of the cross platform applications from the scope of terminology, and then we explain the differences between native and cross platform applications. There is also a brief description of existing cross platform tools. Finally we build a cross platform application with social media and geolocation services.

## 2. Problem Description

### 2.1 Problem statement

The main reason to create this project is the need for common code for different platforms. Mobile cross platform frameworks are addressing that challenge by leveraging the ubiquitous browser and Java script or scripting languages such as Lua or Ruby. Apart from the developing part of this postgraduate thesis project we highlight the idea that there are no standards yet for a universal solution when we refer to mobile world fragmentation in contrast with other fragmentations happened in the past, such as the desktop world fragmentation in the previous decade when operating systems of Microsoft<sup>1</sup>, Apple<sup>2</sup> and some Unix like systems appeared in the market. For the desktop fragmentation the Sun's Java<sup>3</sup> provide us the “build once and run anywhere” strategy to give a solution instead of developing in-house solutions by building frameworks in C++ with Operating System (OS)-specific modules abstracted. One other fragmentation is the web browser fragmentation appeared between 2004 and 2008 when browsers not following the specifications outlined by the World Wide Web Consortium (W3C)<sup>4</sup>. The above fragmentation was solved with selection statements during programming (e.g. if internet explorer then do this, else do that) [2]. Lastly there is a need for performance improvements in order the cross platform applications to dominate the mobile world.

### 2.2 Goals

The main goal is to develop a cross platform application using social media and geolocation services and to avoid the known fragmentations of mobile devices world. Secondary goals are to explain the concept of cross platform idea.

## 2.3 Purposes

Our motivation is to apply existing technologies learned during the undergraduate and postgraduate courses, such as HTML, CSS, and java script to develop mobile applications in a cross platform way and see in practice the differences between the cross platform way and the native way where there is previous experience.

## 2.4 Methods

Choosing a methodology is not simple, because there is no methodology to fit in every project. For the purpose of our work we used the agile development [3] which is one of the three major categories of systems development methodologies, the other two are the structure design and the Rapid Application Development (RAD) which are out of the scope of our project. Agile software development is a group of software development methods based on iterative and incremental development, where requirements and solutions evolve through collaboration between self-organizing, cross-functional teams. It promotes adaptive planning, evolutionary development and delivery, a time-boxed iterative approach, and encourages rapid and flexible response to change. It is a conceptual framework that promotes foreseen interactions throughout the development cycle. The Agile Manifesto [4] introduced the term in 2001.

# 3. Method

## 3.1 About the Methodology

In this chapter we refer to the method that we used. The work in this project was done through the agile development due to time limitations and for one more reason: Because these programming centric methodologies have few rules and practices, all of which are fairly easy to follow. In detail these methodologies simplify the SDLC (Systems Development Life Cycle) and its four stages (planning, analysis, Design, implementation) focusing on simple and iterative application development. One characteristic of these methodologies is the absence of occupation with modeling and documentation for long time. Some of the agile development methodologies are the extreme programming (XP), Scrum and the Dynamic systems Development Method (DSDM) For the purpose of our project we used the extreme programming methodology from agile methodologies. The extreme programming [5] was created around 1995 by Kent Beck and Ward Cunningham. XP is a “lightweight, efficient, low-risk, flexible, predictable and scientific. It is based on four values: communication, simplicity, feedback and courage. Extreme programming is made to accept changes, which are very valuable for us. Apart from our selection due to time limitations we mentioned above, this methodology gave us the opportunity to develop our own skills through work and the confidence because of the feedback we had from the supervisor professor.

### 3.2 Roles

Generally the extreme programming methodology includes two basic roles, the customer and the developer. Additionally there are two supplementary roles, the tracker and the couch [6]. The customer drives the whole project with continuous involvement; the highest participation of the customer equals the better result. The developer's work is to produce code based on the customer's oral stories. The tracker work is to keep track of the scheduled jobs and the couch job is to lead the team of developers by directing them with deep knowledge of the extreme programming practices. For our project we used the basic roles including the supervisor professor as a customer and the developer as a pair of developers with core coding skills and UI coding skills respectively.

### 3.3 XP Lifecycle

According to Kent Beck [7], the pioneer of extreme programming, the ideal XP lifecycle project has six stages the exploration, the planning, the iterations to first release, the productionizing, the maintenance and the death. At the first stage we explore if we have the skills, the tools and the confidence to carry on for a good first release. The second stage is where customers or users meet with the development team to create user stories or requirements. The development team converts user stories into iterations that cover a small part of the functionality or features required. A combination of iterations provides the customer with the final fully functional product. In our case the discussions with the supervisor generated the required stories and requirements. The pair of programmers prepares the plan, time, and costs of carrying out the iterations, and individual developers sign up for iterations. The suitable and also learned during our studies method as a planning approach was the critical path method [8]. With this approach the essential iterations for the project are grouped in linear way and arranging for completion of other iterations parallel to the critical path. The next stage is about to break the schedule into weekly iterations with the target to produce a set of functional cases for each of the stories scheduled for that iteration. The first iteration is the base of the entire system and at the end of the last iteration the project is ready for the stage of production. At the stage of production we have the parallel testing before releasing. The last two stages are maintenance and death. At the maintenance stage we have to simultaneously produce new functionality, keep our existing system running and adding new people in the team if it is necessary. Finally the death stage marks the customer's satisfaction about the whole project, in more detail when the customer has nothing more to say about the project or to tell new stories this is the end of an XP project. There is also a bad death stage when the customer wants more features and the team of the project is out of economic schedule. At a normal end of the job the team is responsible to write a small document about the system for future changes.

## 4. Existing Technologies & Practices

### 4.1 Introduction

In this chapter we are going to discuss the existing technologies and practices by making clear the majority of the terms referring to our project. At first we explain the differences between native and cross platform applications, and then we compare them in tables. Next we describe the existing cross platform tools and compare them in tables also. Before we mention any tool we have to understand the mobile application development options because each development scenario has its advantages and disadvantages. According to the scenarios we have many factors to think of, such as our team's developing skills, the required device functionality, the offline capability and the importance of security. So there are many factors we have to care of before we proceed with our project. When a choice for a project is the wrong choice and some factors are miscalculated it may drive us far away from our targets and purposes with bad consequences like choosing a bad Goldilocks zone<sup>5</sup> which tells us that when a planet is too far from the star it is too cold, so close and it is so hot in respective to understand the hardship we have to go through when things go wrong in a software project. Mobile developing scenarios including, native applications, web HTML5 applications and hybrid applications. The explanation of the classification between the three previous terms mentioned is also a part of this chapter.

### 4.2 What is a Native Application?

Our first approach is the native application development. When we talk about native mobile applications we describe binary executable files that are downloaded directly to the device and stored locally. The installation process can be initiated by the user or, in some cases, by the IT department of the organization. The most popular way to download a native app is by visiting an app store, such as Apple's App Store, Android's Marketplace, BlackBerry's App World and Microsoft's windows phone store. Other methods exist and are sometimes provided by the mobile vendor. Once the application has been installed on the device, the user launches it like any other service the device offers. Upon initialization, the native application interfaces directly with the mobile operating system, without any intermediary or container. The native application is free to access all of the application programming interfaces (API's) that are made available by the OS vendor and, in many cases, has unique features and functions that are typical of that specific mobile OS [9]. Once the native application is installed on the mobile device and launched by the user, it interacts with the mobile operating system through proprietary API calls that the operating system exposes. These can be divided into two groups: low-level API's and high-level API's. The low level API's is responsible to connect with the touch screen the keyboard, the microphone, to play sounds through the speakers , to receive images from the camera, to access the Global Positioning System (GPS), and generally access all the hardware elements of the device giving the application all the required functionality. The high level API's providing higher level



services such as web browsing, calendar management, contacts, photo album, phone calls, text messaging and many more.

Although most mobile Operating systems include a set of built-in applications that can execute these services, a set of exposed high-level API's is made accessible for native apps as well, allowing them to access many of the important services mentioned above. Other API's enable downloadable apps to access various cloud-based services that are provided by the OS vendor, such as push notifications or in-app purchases. There is an important set of API's that any mobile Operating system provides and has to do with graphics in order to provide the graphical user interface (GUI) toolkit , such as menus , buttons , dialog boxes etc. The most known mobile operating systems provides a specific Integrated Development Environment (IDE) in order to be possible for the developer to use these graphical elements mentioned above which they inherit the features and functions of a specific OS that we develop the application. When we choose to develop a native mobile application on a specific platform we must have specific skills on the programming language, the IDE, and the specifications of the devices that use this platform. Next we overview some of these platforms discussing the programming language, and the IDE that they use. Additionally, we create a simple “hello world” application for android platform in order to demonstrate the installation of one of these tools. Finally before we proceed with the description of the platforms we provide a brief table with specifications of the platforms grouped by programming language, tools we need, the packaging format of the application and the store where we can download the applications.

|                          | <b>Android</b>    | <b>Apple iOS</b>    | <b>Blackberry OS</b>    | <b>Windows Phone</b>                           |
|--------------------------|-------------------|---------------------|-------------------------|--|
| <b>Languages</b>         | Java (some C,C++) | Objective C, C, C++ | Java                    | C#, VB.NET and others                          |
| <b>Tools</b>             | Android SDK       | Xcode               | BB Java Eclipse Plug-in | Visual studio, Windows Phone development tools |
| <b>Packaging Formats</b> | .apk              | .app                | .cod                    | .xap   |
| <b>Stores</b>            | Google Play       | Apple App Store     | Blackberry App World    | Windows Phone Marketplace                      |

**Table 1 Platform General Information**

#### 4.2.1 Programming Language – IDE for Android OS

To develop for the Android, we can use Windows, Linux, or Mac. Android applications are typically written in Java. Since the release of the Android NDK (Native Development Kit) in June 2009, developers may also create native libraries in C and C++ to reuse existing code or gain performance. The most commonly used and recommended editor is Eclipse<sup>6</sup> with the Android Development Tools plug-in. The plug-in provides a full-featured development environment that is integrated with the emulator. It provides debugging capabilities and let us install multiple versions of the Android platform easily. There is also a capability of using command line tools to create a skeleton application, emulator, debugger and binge to an actual device.

#### **4.2.2 Programming Language – IDE for Apple iOS**

When a developer decides to develop an iPhone application the hardware requirements include an Intel-based Macintosh computer running OS X v10.5.7 or later. From the scope of software there is a need for installation of iPhone SDK<sup>7</sup>, which includes the Xcode IDE, iPhone simulator, and a suite of additional tools for developing applications for iPhone and iPod touch. With these tools the programmer can create and run the application in the simulator. The IDE that a programmer needs is the Xcode which is the Apple’s integrated development environment for developing applications for Mac OS X and the iPhone. The preferred language in Xcode is Objective-C, which is required for iPhone applications, but Xcode also supports a variety of languages such as C, C++, Fortran, Java, Objective-C++, AppleScript, Python, and Ruby. The Xcode IDE has a modified GNU compiler and debugger for its backend. The Xcode suite includes Interface Builder and Instruments. Interface Builder helps to create user interfaces for the Mac and iPhone applications. Instruments provides a thorough analysis of the application’s runtime performance and memory usage, allowing the developer to efficiently find memory leaks and bottlenecks to help improve the user experience.

#### **4.2.3 Programming Language – IDE for Blackberry OS**

BlackBerry apps are developed in Java using MIDP 2.0, CLDC 1.1 and RIM’s proprietary API’s. Extensive documentation, training videos, and downloads are available at the BlackBerry Developers Web Site<sup>8</sup>. The tools to develop for BlackBerry are free. Although the BlackBerry tools are based on Java, only the Windows 32-bit operating system is really supported for development. The learning curve to develop native BlackBerry applications in Java is relatively steep compared to other mobile platforms. The BlackBerry runs a proprietary multitasking operating system. Version 7.1 is the most current version, although developers should be prepared to encounter much older versions since BlackBerry owners sometimes do not upgrade for a while, especially if the devices are being provided from their enterprise. Central to understanding the BlackBerry platform is the BlackBerry Enterprise Server (BES). BES provides advanced functionality for IT administrators. A BES allows administrators to deploy and update applications, set policies for devices, and most importantly, synchronize email, calendar entries, contacts, and tasks wirelessly using push technology. BES is one of the reasons the BlackBerry is so dominant in the enterprise market. A fast Windows machine is recommended. It is possible to develop on a Mac by running these tools inside a Windows virtual machine, but for best performance developers should run Windows natively. Finally developers need to download eclipse and BB java eclipse plug-in to proceed with blackberry application development.



#### 4.2.4 Programming Language – IDE for Windows Phone

Microsoft’s Windows Phone OS is the brand new proposition of Microsoft about the mobile world after the fade out of windows mobile. The initial release was back to 2010 and the most stable release for now is windows phone 8. The main programming languages required to develop windows phone applications is the C# and the Visual basic. The ease of use provided by the visual studio as a tool is a major help even for new developers. The marketplace of windows phone is growing rapidly. One big advantage of windows phone is the strict compatibility between devices of the same OS version and one big disadvantage is the quick fade out of windows phone 7.5 without any upgradability to windows 8 in hardware terms and in tool terms too.

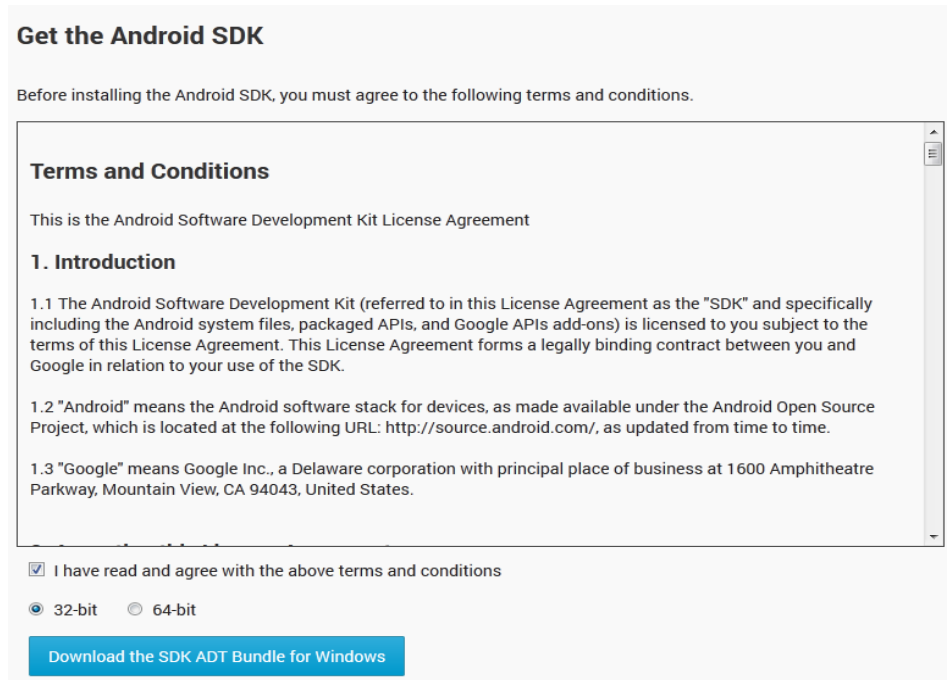
#### 4.2.5 Setting up the Android ADT Bundle and Running an Application

In this paragraph our purpose is to setup the android ADT bundle in order to run a native simple application which is one of our previous projects on this platform. The application is developed for windows Phone OS also with easiness because of the perfect tools of Microsoft Blend and the Visual studio IDE. We decided to use android as an example because of the availability of devices in order to run it on a device and not only virtual, the limited possibilities of our personal computer and the previous knowledge of developing mobile applications. At first we download the required software which is the ADT bundle<sup>9</sup>. We used ADT bundle because according to their site the ADT bundle included, Eclipse and ADT plugin, Android SDK Tools, Android Platform-tools, the latest Android platform, the latest Android system image for the emulator. Next we are going to demonstrate step by step the installation of ADT Bundle and the creation of a simple native application for this platform. First we download from the site mentioned above the ADT Bundle for windows.



**Figure 1 Download the SDK**

Then we choose the 32 bit version and we read the terms and conditions before we proceed. We downloaded the 32 bit version because our system runs windows 7 professional 32 bit. The choice is free depends on your system.



**Get the Android SDK**

Before installing the Android SDK, you must agree to the following terms and conditions.

**Terms and Conditions**

This is the Android Software Development Kit License Agreement

**1. Introduction**

1.1 The Android Software Development Kit (referred to in this License Agreement as the "SDK" and specifically including the Android system files, packaged APIs, and Google APIs add-ons) is licensed to you subject to the terms of this License Agreement. This License Agreement forms a legally binding contract between you and Google in relation to your use of the SDK.

1.2 "Android" means the Android software stack for devices, as made available under the Android Open Source Project, which is located at the following URL: <http://source.android.com/>, as updated from time to time.

1.3 "Google" means Google Inc., a Delaware corporation with principal place of business at 1600 Amphitheatre Parkway, Mountain View, CA 94043, United States.

I have read and agree with the above terms and conditions

32-bit    64-bit

[Download the SDK ADT Bundle for Windows](#)

**Figure 2 Terms and Conditions**

When the browser prompts us to open the file `adt-bundle-windows-x86-2012522.zip`, we select to save the file and click OK. The file is 426 MB and is about 30 minutes to download depending of your internet connection speed.

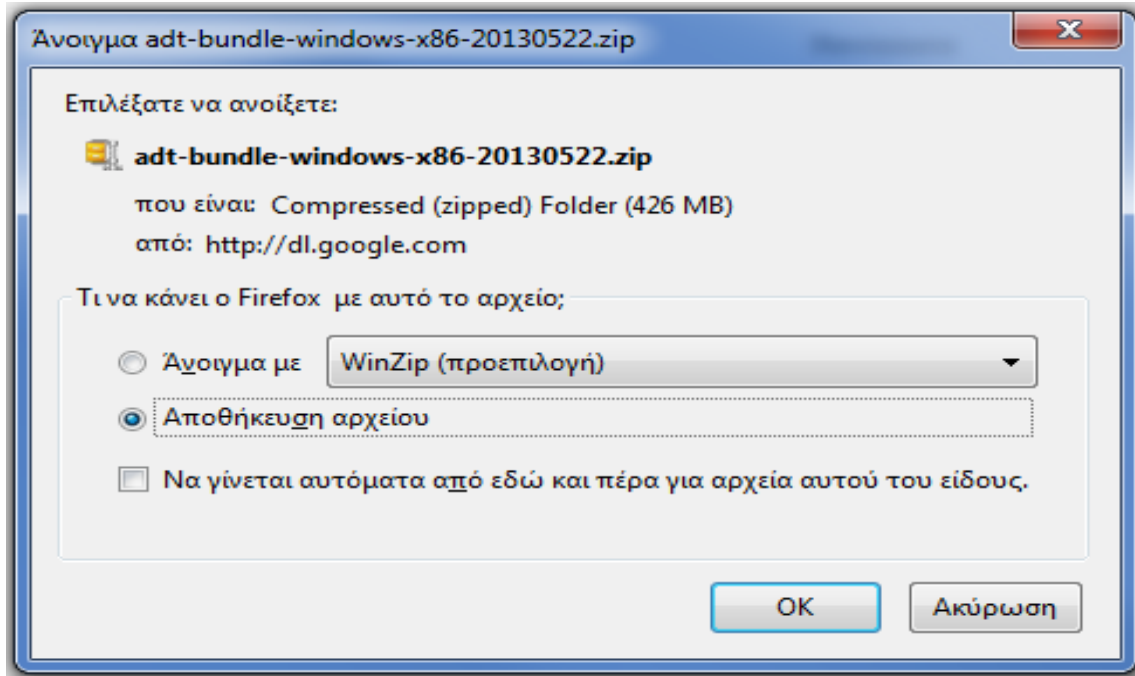


Figure 3 Save the ADT Bundle

After the successful download of the file now is time to install the SDK and the Eclipse IDE. For that reason we unpack the ZIP file adt-bundle-windows-x86-2012522.zip and save it to an appropriate location, such as "ANDROID DEVELOPMENT" directory in your home directory.

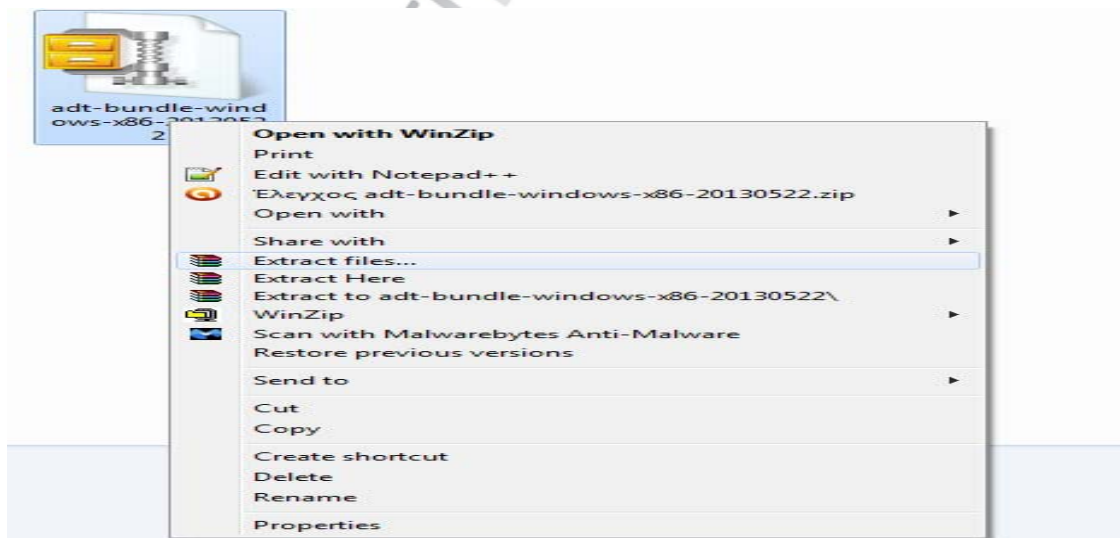


Figure 4 Extract ADT Bundle

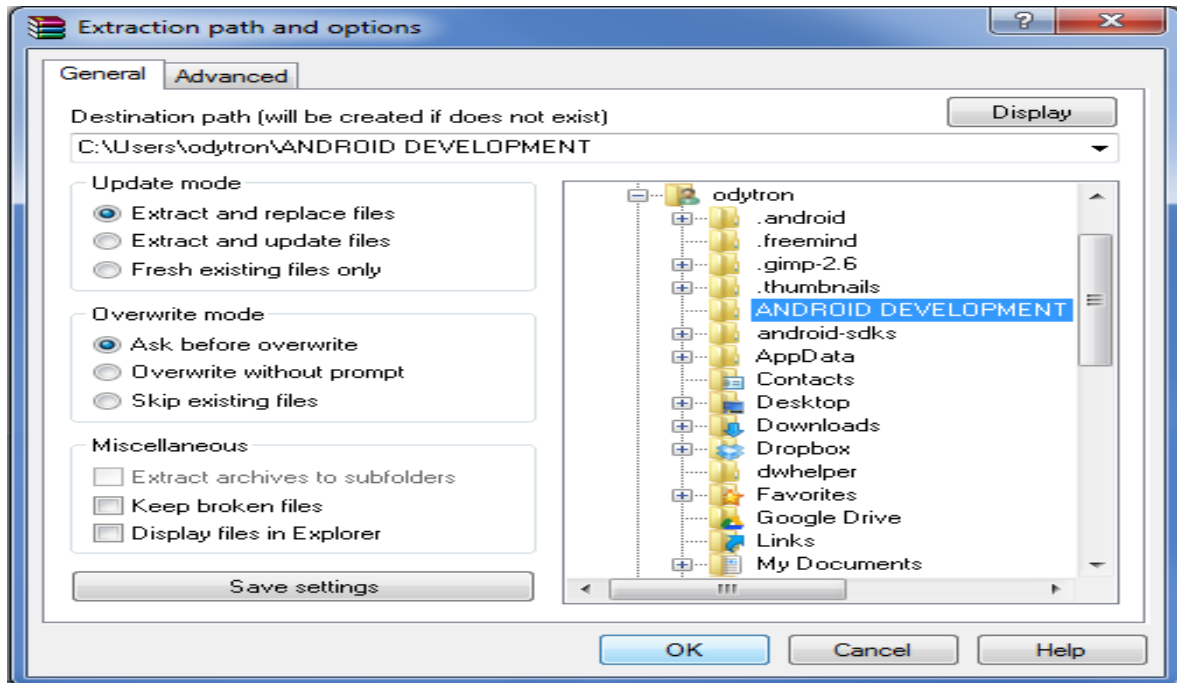


Figure 5 Extract ADT Bundle II

Now everything is ready to begin developing our android applications or open existing applications. The next step is to open the unpacked folder “ANDROID DEVELOPMENT” and run eclipse. After that the program prompts us to select a workspace, in other words where to save our projects. The location of eclipse is located in our case under:

C:\Users\odytron\ANDROIDDEVELOPMENT\adt-bundle-windows-x8620130522\eclipse

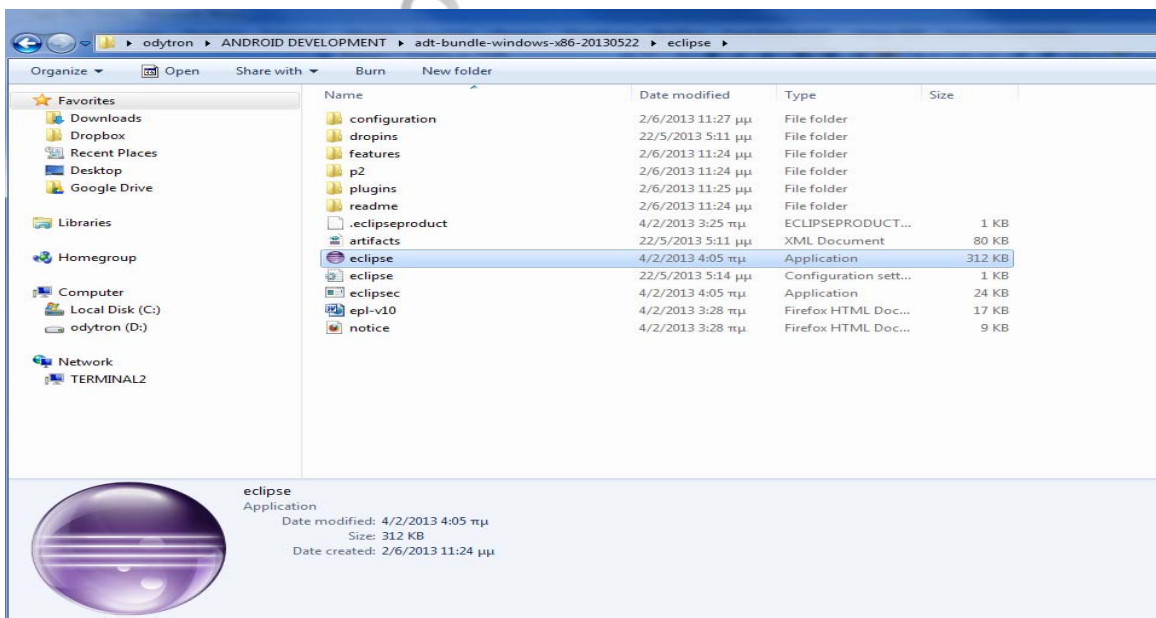


Figure 6 Open Eclipse

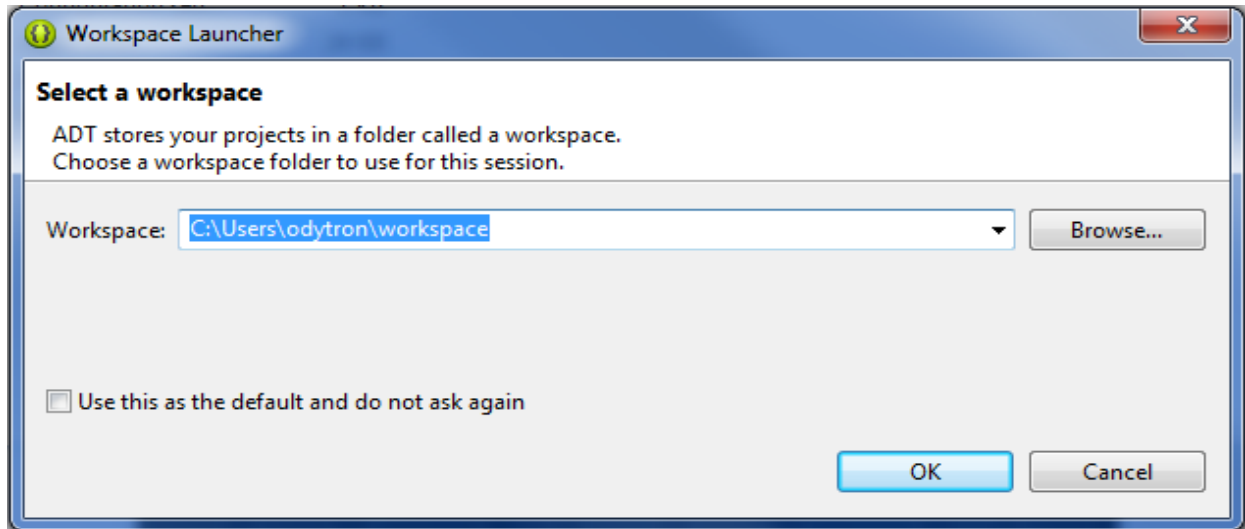


Figure 7 Select a Workspace

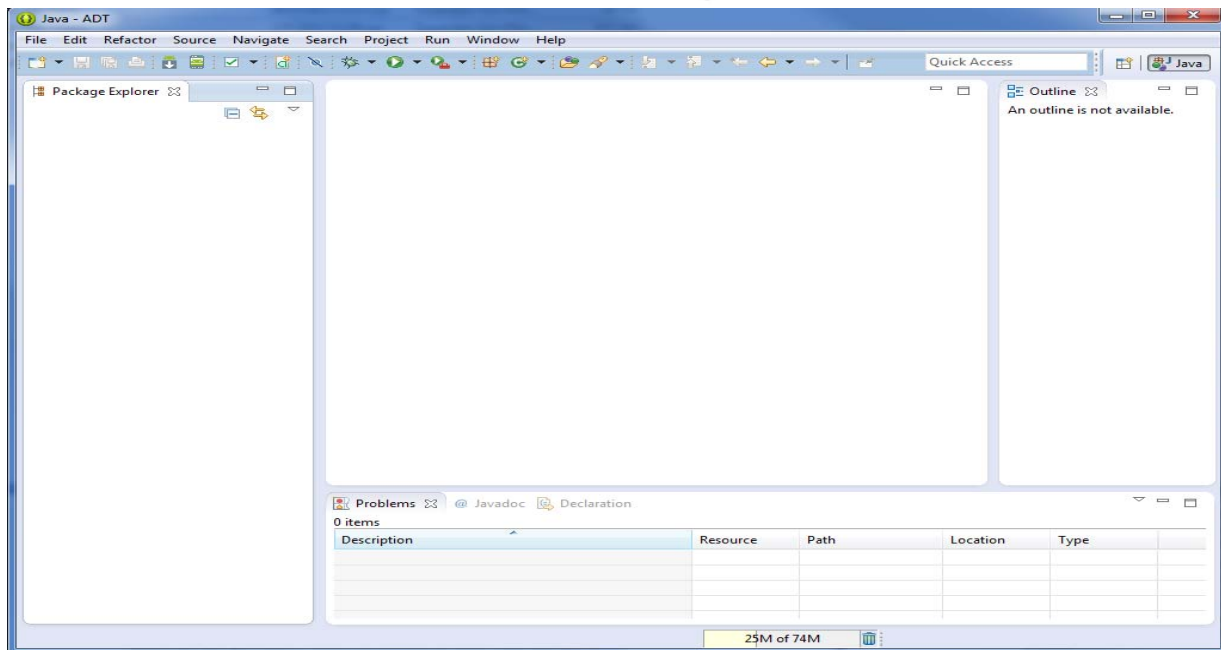


Figure 8 The Eclipse Environment

As we can see the main screen of eclipse is user friendly familiar with other IDE's. There is a menu bar, the tools, the package explorer when we put our files the main code window and other elements. The next step is to update the SDK manager.

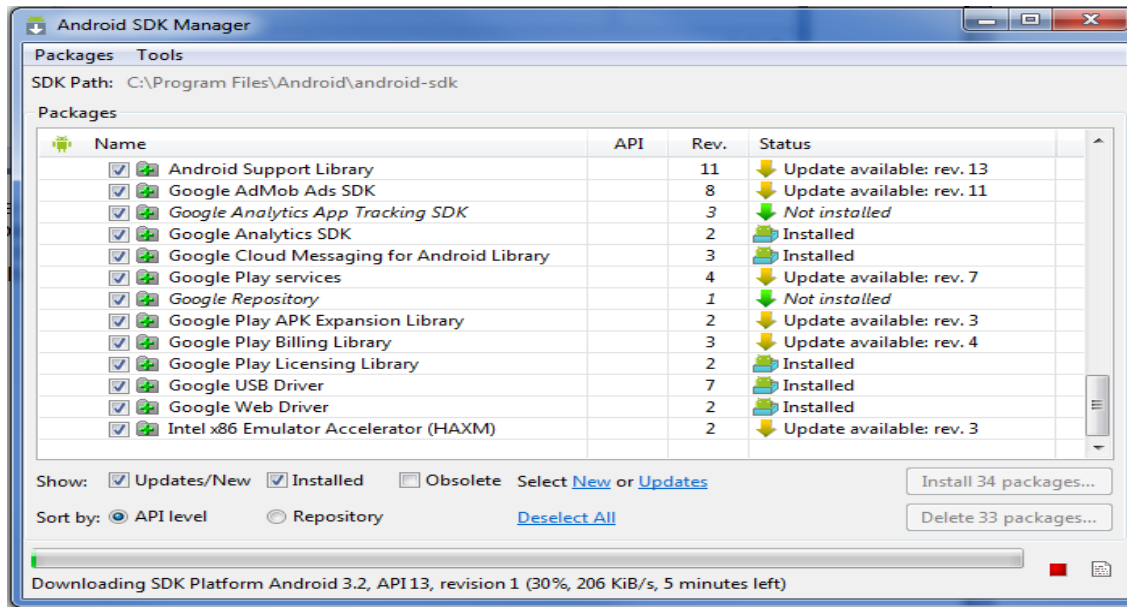


Figure 9 Android SDK manager

Next step is to demonstrate an application developed with native fashion. We choose an application already developed for the mobile devices programming courses, during the third semester of our postgraduate studies. The application is an integer calculator with a simple user interface calculating mathematical operations using web services or local functions. We demonstrate the second version with the functions locally stored. The first thing we have to do is to import our project to eclipse IDE. We select file import and then in the folder General we select Existing Projects into Workspace.

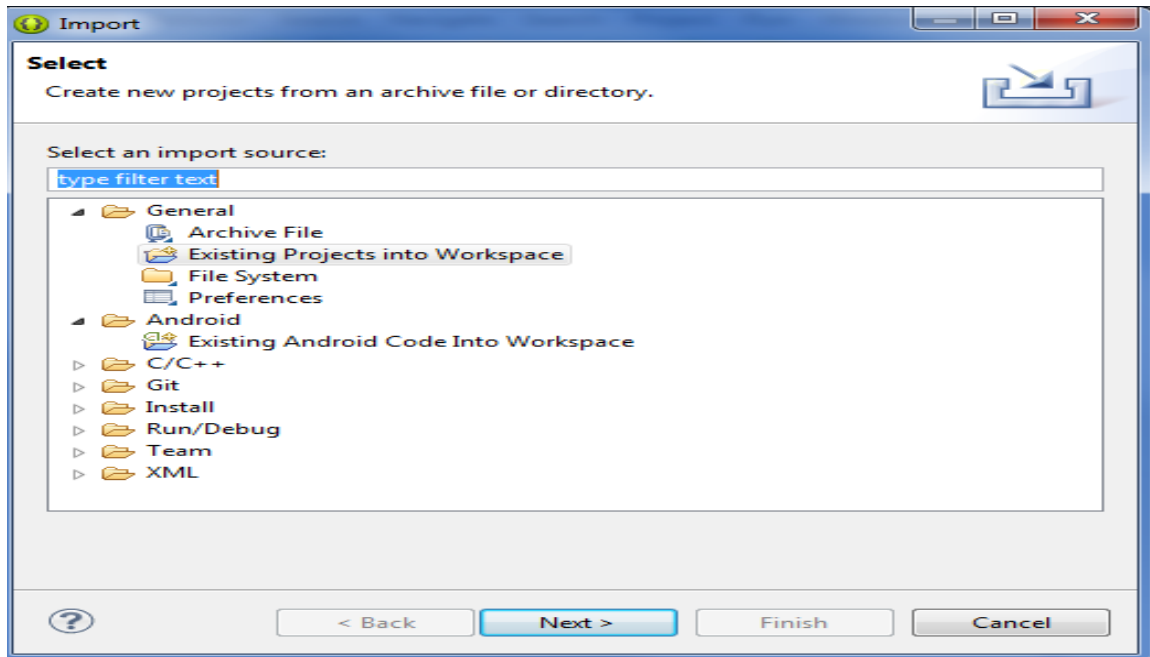


Figure 10 Importing Project

As we can see in the picture below, the project is loaded with all subfolders and other elements.

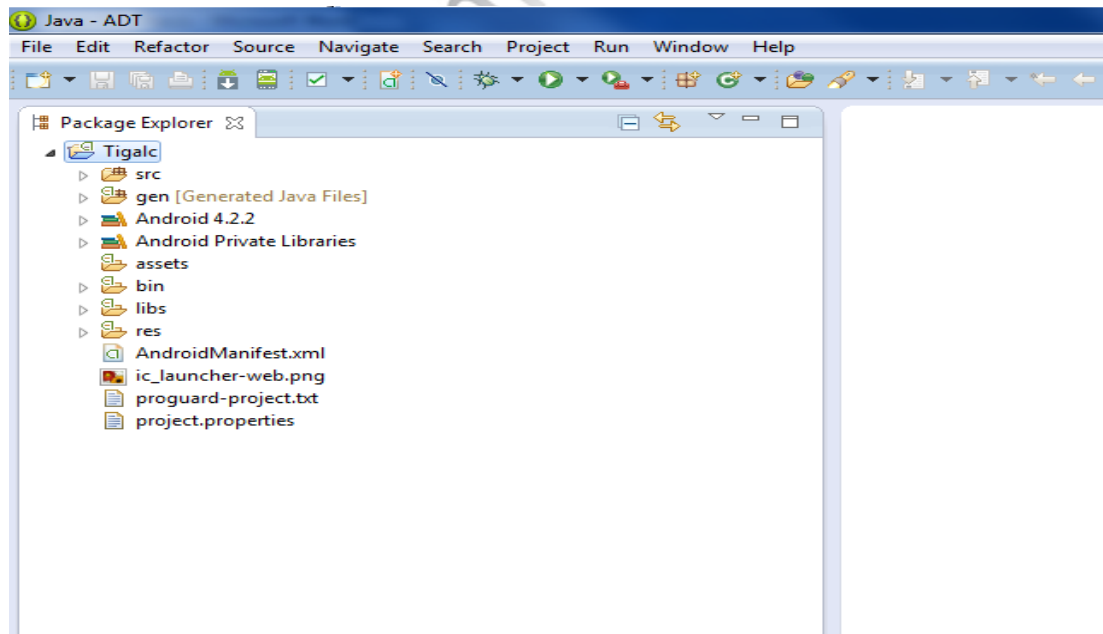


Figure 11 Project Loaded

The first folder is the general project folder, the src folder includes the project packages and it is recommended to be very careful with these names because it is tuff



to change them in the future. Under this general package name are the java files where all the functionality is embedded. The next important folder is the res folder where all the images and graphical elements are saved into drawable folder. Apart from the drawable folder there is another folder named layouts. A layout file is always a .xml format file while a pure code java file is always a .java format. The java files giving the functionality to the layout files in order to be able to act. There is an example below with a button which is in the layout file and the functionality of the button which is in the java file. The button of add operation has an xml code behind which appearing by clicking it. Each xml layout drives to the respective java file and each java file drives to the respective xml layout file.

```
<Button
    android:id="@+id/btnsum"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_x="39dp"
    android:layout_y="275dp"
    android:text="+" />
```

Table 2 xml layout file

As we can see there is an easy user interface for the developer in order to manipulate buttons, texts, menus and other graphical elements.

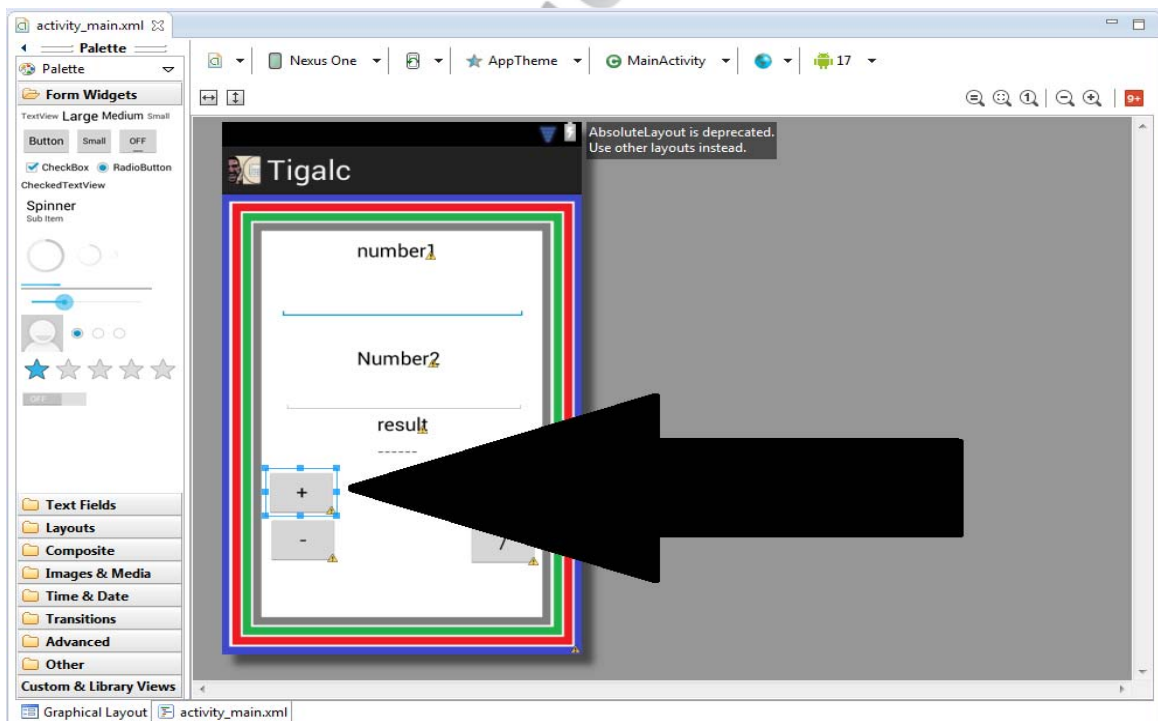


Figure 12 Manipulating graphical elements

Next table illustrates a small part of the respective java code of this specific xml layout. Note that java files beginning with capital letters while xml files with lower case letter.



```

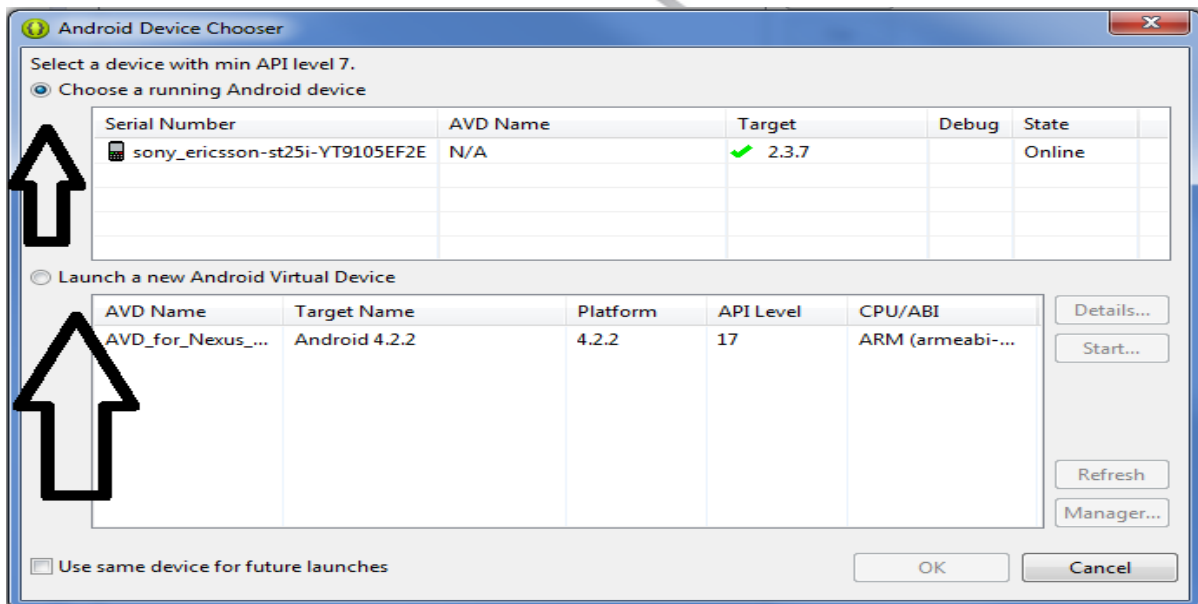
if (v==b3) {

    String a,b;
    Double vis;
    a = ss1.getText().toString();
    b = ss2.getText().toString();
    vis = Double.parseDouble(a) - Double.parseDouble(b);
    shw.setText(vis.toString());

}
    
```

**Table 3 Java Code Sample**

At the Annex we provide all the code of this small native project which includes the java and the xml file. At the Annex there is also the code of the cross platform application which follows at the next chapters. Finally we have to run the finished application with two ways, the first way is to connect your android device to the computer and select it to run on the device or to run virtual with android emulator. Next image shows the options for running configurations. We tested with both ways.



**Figure 13 Android Device Chooser**

Finally the application is running on the emulator. As we mentioned before, it is a calculator, executing four basic mathematical operations, the addition, the subtraction, the multiplication and the division. There is also a notification of error when we try to divide by zero. The paragraph 4.2 was all about native applications, the rest of chapter five has to do with cross platform applications.

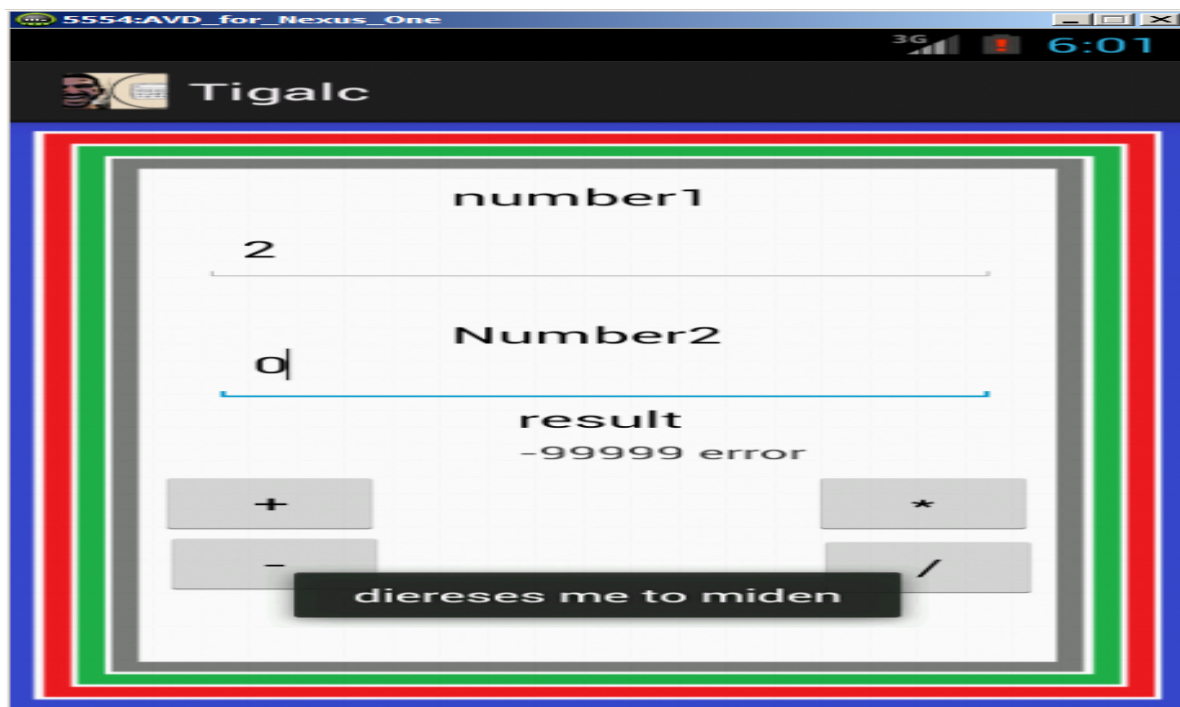


Figure 14 Finished Android Application

#### 4.3 What is a Cross Platform Application?

Our second approach is the cross application development. As we mentioned at this paragraph 4.1, there are two cross platform scenarios from which a developer can select to proceed with a project. The scenarios are the cross platform web application developing and the cross platform framework application developing. The first scenario is to develop an application with web techniques only, which was the initial idea of programmers for developing cross platform applications easily before the appearance of specific cross platform frameworks. Developing with this way means that a web UI is reacting as web page, using all the functionality of a web application, totally avoiding the barriers and specific knowledge requirements of the API's. This way is an easy way to maintain an application anytime because a web application uses the same principles of a web page in terms of navigation. A similar way of control flow with screens acting like pages, buttons and menus is a characteristic for most of the mobile devices. With a web application development we can use a simplified common code in HTML, CSS and Java Script using the web browser of a modern cell phone. With this trend of creating applications we can run the same code across platforms. For organizations that require fast results and applications for all platforms, it is much easier to hire developers with HTML, CSS and Java Script knowledge than to hire developers with specific knowledge on cross platform various tools and frameworks. The second way of developing applications is the cross platform framework usage. Generally there are two trends of tools in this category of framework cross platform developing. The first trend lets the developer to design native applications using a common programming language like C, C++ for cross platform API's through a compiler included in a framework. Usually we

call this way a native cross platform solution. With the second trend the developer can create projects by using HTML, Java Script, CSS and some libraries of the native environment of a specific platform via a native container in order to use the hardware peripherals of the device and other functions. The second way is usually called the hybrid cross platform solution, in other words the hybrid approach combines native development with web technology. In the picture below we illustrate the three main types of applications with their basic elements. The three main types are the native application, the web application and the hybrid application. In the picture below we can see the differences between three types of applications. In the first application which is mentioned in paragraph 4.2 we have the pure native style with only the code e.g. (JAVA, C#, objective C) which reacts with the device API's. In the second image we have a pure web application which is mentioned at the very beginning of this paragraph and in which the mobile browser is the whole container and includes the UI and the business part of the code acting like a web page. At the last image we can see a hybrid application in which there is a native container with the nested web code inside and the reaction with the device API's.

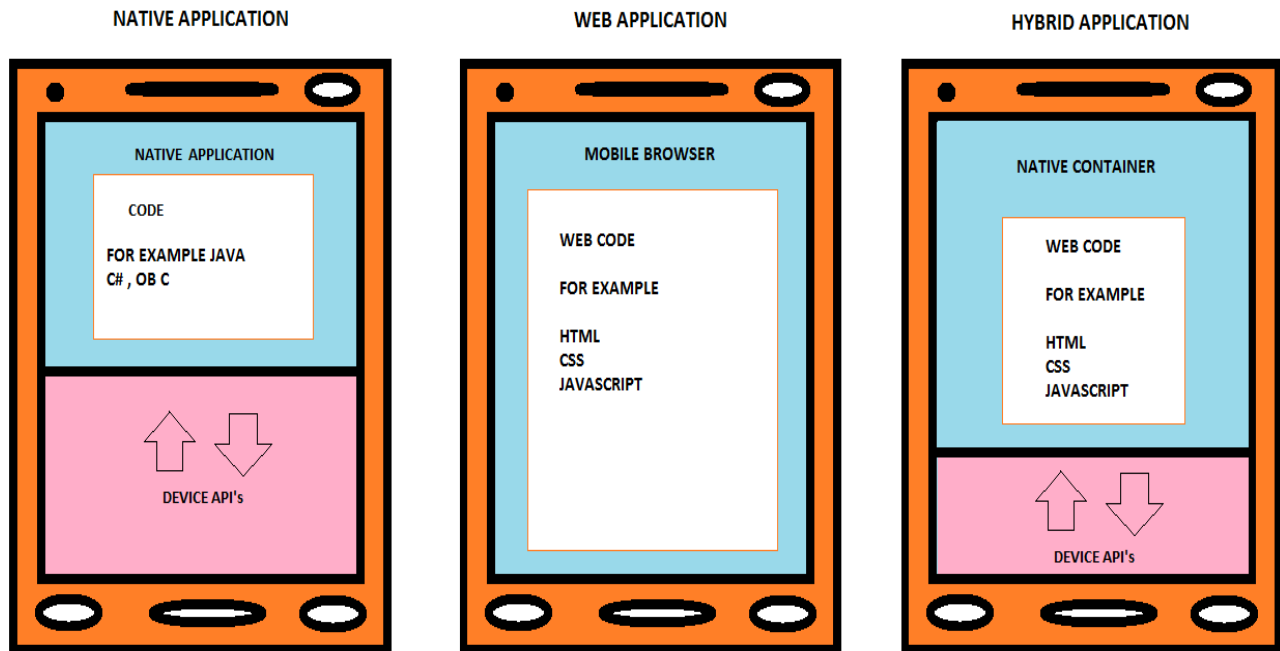


Figure 15 The three types of applications

#### 4.4 Native and Cross Platform Application Comparison

In this paragraph we discuss the advantages and the disadvantages between native and cross platform applications. The native applications are better from the scope of

performance but their main problems are the updating and the cost of developing. It not easy to update a version of an application widely used via the markets of the brand all the time and it is also difficult for an organization to afford the cost of stuff with high developing skills. The web application approach is very easy to be updated and has lower coding skills requirements. The disadvantage of the web approach is the limited functionality compared with the other approaches which are related with the API's. The last approach is the hybrid approach with the advantage of “standing between the two approaches” giving the most efficient result for the developer who wants to program the same application for more than one operating system. In order to make a brief description and comparing all approaches we created two tables. The first table that is following giving the appropriate information for the differences of each approach based on various functions alphabetically with the 4 level characterizations Excellent, Good, Poor and None.

| Function                          | Native application          | Hybrid application          | Web Application                 |
|-----------------------------------|-----------------------------|-----------------------------|---------------------------------|
| Access device hardware            | Excellent                   | Good                        | Poor                            |
| Advanced Graphics                 | Excellent                   | Good                        | Good                            |
| Application Upgrade               | Poor (via the markets)      | Good                        | Excellent (manipulate web code) |
| Code Portability and Optimization | None                        | Excellent                   | Excellent                       |
| Easiness of installation-access   | Excellent (via the markets) | Excellent (via the markets) | Good (via the browsers)         |
| Using existing code elements      | Poor                        | Excellent                   | Excellent                       |

**Table 4 Characteristics of types of applications**

The second table shows some other important functions such as the type of developing language, the running method, the developing skills, the user experience and the performance

| Function            | Native application                 | Hybrid application        | Web Application                                   |
|---------------------|------------------------------------|---------------------------|---|
| Developing Language | Usually Java, C# objective C e.t.c | Web languages             | Web languages                                     |
| Developing Skills   | Java, C                            | HTML, CSS, JS             | HTML, CSS, JS                                     |
| Running Method      | From markets                       | Like a native application | From URL  |
| Performance         | High (locally stored)              | High (locally stored)     | Low ( based on server stored needs network access |
| User Experience     | Excellent                          | Good                      | Low   |

**Table 5 Other important characteristics**

As we discussed in the introduction of this chapter, there is no perfect approach. The tables above confirm that opinion by giving us valuable conclusions in order to think and plan our moves before we decide to create an application. Based on the two tables we discuss some scenarios of usability of each approach. For example if the organization has talented and experienced staff in many programming languages may proceed with native applications. Two other reasons for companies to choose the native way is when the company aims to a specific target group e.g. apple users only, android users only etc, and when the application requires full functionality from the specific OS, e.g. in sector of graphics. The scenario of web mobile application is used when the organization require a fast distribution to be available and visible from everybody and not from a small group of a specific device owners.

Finally the hybrid approach is suitable to combine the two previous approaches. The reasons for an organization to combine the two approaches are: Firstly to use existing knowledge of web technologies in order to accomplish more difficult functions that only the API's can offer. A more difficult function may refer to easy access to plug-ins that can easily be used in other services or tools. These also include offering common links to similar APIs, such as those for the devices cameras, accelerometers, or location sensors. Instead of writing unique code to talk to an iPhone's GPS and an Android smartphone GPS, one centralized set of code will be automatically modified to interface with both devices. Secondly the organization may target to invest in a “non fade out soon” technologies like HTML5, CSS and JS. Apart from the mentioned advantages and disadvantages of the application developing depending to approaches, there are many more new functions in the mobile world. This new functions are emerging the use of cross platform applications because many of the new functions are advantages of cross platform technologies. For example the easy support for enterprise and cloud services is a big advantage for cross platform applications because these frameworks allow the easy integration of cloud services. For example, after exchange integration is coded once, it will work on both platforms. Also, multiple security methods aren't needed, because the applications will function similarly on either the iOS or the Android platform.

#### **4.5 Cross Platform Tools Disadvantages**

In this paragraph we highlight some disadvantages of cross platform tools. The first disadvantage is the lack of documentation comparing to the classic SDK's because many of the cross platform tools and frameworks are too new. The second disadvantage has to do with the supporting features when a specific SDK is changing somehow adding more features. The cross platform framework must be updated also in order to be able to pass all the information required after the changes. The challenge is the bridge between two languages. The third disadvantage is the restrictiveness of the cross platform tools which may force the developers to use specific suites ignoring their preferences. The restrictiveness of the tools may drive the whole stuff learning many things from the scratch. The fourth disadvantage has to do with the inefficiency of the code. The code may be inefficient when the developers not programming in specific native languages on a specific platform, because the final code is determined by the



efficiency of the translation engines of the tools. A translation engine may cause the exported code not to follow the standard coding techniques. If the code is inefficient the possibility for an organization to avoid using this code in the future is big.

#### 4.6. Existing Cross Platform Tools

In this paragraph we are going to make a brief presentation of four popular cross platform frameworks and tools. The classification of these frameworks and tools in this project will follow the categories of the previous paragraph. The categories of tools and frameworks are two, the native solution cross platform solutions which is a category of tools and frameworks where a compiler translates a common programming language code into the specific's device native code [10].

The second category is the hybrid cross platform solution and includes the kind of the tools and frameworks in which the developer uses web technologies as mentioned before, in order to access the phone's advanced features, such as location, orientation, camera, etc.

##### 4.6.1 Appcelerator Titanium

Appcelerator Titanium is a platform for developing mobile, tablet and desktop applications using web technologies [11]. Appcelerator Titanium is developed by Appcelerator Inc. and was introduced in December 2008<sup>10</sup>. Support for developing iPhone and Android-based mobile applications was added in June 2009<sup>11</sup>. Support for developing iPad-based tablet apps was added in April 2010<sup>12</sup>. BlackBerry support was announced on June 2010<sup>13</sup> but it is still in closed beta. Appcelerator Titanium Mobile is one of several phone web based application framework solutions allowing web developers to apply existing skills to create native applications for iPhone and Android. Yet, while using the familiar JavaScript syntax, developers will also have to learn the Titanium API, which is quite different from familiar web frameworks such as jQuery. All application source code gets deployed to the mobile device where it is interpreted (the company's marketing refers to this as being a "cross-compiler")<sup>14</sup> using a JavaScript engine; Mozilla's Rhino is used on Android and BlackBerry, and Apple's Javascriptcore is used on iOS. In 2011 it was announced that a port to Google's V8 JavaScript engine is in development which, when complete, will significantly improve performance<sup>15</sup>. Being interpreted means that some errors in the source code will not be caught before the program runs. Program loading takes longer than it does for programs developed with the native SDKs, as the interpreter and all required libraries must be loaded before interpreting the source code on the device can begin. Some developers have reported that although working with Titanium gives fast results, making Titanium well suited for prototyping, there are issues around differences in behavior of the API cross-platform, stability and memory management, that made them re-write their apps in native code in the end.<sup>16, 17</sup>. However, as of February 28, 2012, there have been over 30,000 applications shipped to the app stores built with Titanium, including NBC Universal's flagship mobile app<sup>18</sup>. Many Appcelerator developers cite the speed of development, native UI, and JavaScript skill set needed as reasons why they choose to use Appcelerator<sup>19</sup>. In June 2011, Appcelerator released Studio and Titanium Mobile 1.7<sup>20</sup>. Titanium Studio is a full open standards IDE that is derived from Aptana Studio

which Appcelerator acquired in January 2011. In April 2010 Appcelerator expanded the Titanium product line with the Titanium Tablet SDK<sup>12</sup>. The Titanium Tablet SDK draws heavily from the existing support for iPhone, but it also includes native support for iPad-only user interface controls such as split views and popovers. Initially the mobile SDK only supported development for iPad, but support now includes Android-based tablets as well. Appcelerator, Inc. also offers cloud-based services for packaging, testing and distributing software applications developed on the Titanium platform<sup>21</sup>. The company expanded its product line in January 2011 by acquiring Aptana, Inc, a developer of open source tools for building web applications<sup>22</sup>.

To make a conclusion with this framework we highlight that this framework use JavaScript with custom APIs to build native applications for iPhone and Android. Titanium is an open-source framework, released under the Apache 2 license<sup>23</sup>. Considering all the above Appcelerator Titanium is classified as a native cross platform solution.

The features of Appcelerator Titanium are concentrated in the table below :

| <b>features of Appcelerator Titanium</b>  |
|---|
| <b>Support of web technologies (HTML , Java Script , CSS) on all platforms along with PHP, Python and Ruby for desktop platforms</b><br><b>Integrated support for popular JavaScript and Ajax Frameworks including jQuery, YUI, MooTools, Scriptaculous and others.</b> |
| <b>A platform-independent API to access native UI components including navigation bars, menus, dialog boxes and alerts, and native device functionality including the file system, sound, network and local database.</b>   |
| <b>API access to native mobile functionality like geolocation, accelerometer and maps.</b>  |
| <b>Extensibility through open interfaces and licensing, allowing developers to introduce support for additional scripting languages, media codecs and device-specific functionality.</b>  |

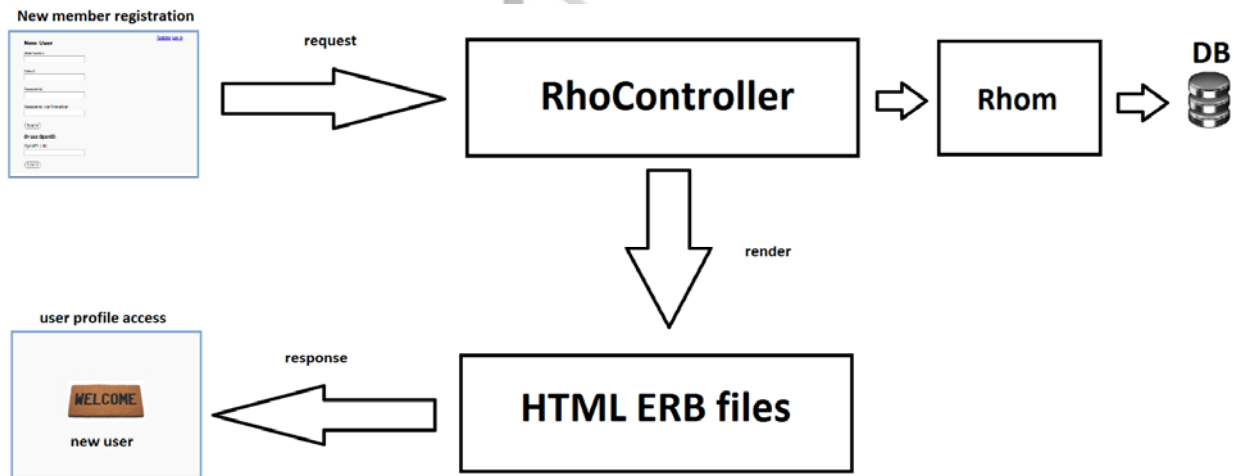
**Table 6 Features of Appcelerator Titanium**

#### **4.6.2 RhoMobile**

The RhoMobile Suite which is formerly known as Rhodes Framework is an open-source framework developed by Motorola and now owned by Motorola Solutions<sup>46</sup> for building native applications that can run on a variety of devices<sup>47</sup>. This means that

regardless of device brand, screen size, or major operating system, an application produced on the RhoMobile Suite framework should look exactly the same when accessed. It is released under the MIT license<sup>48</sup>. Rhodes uses a Model-View-Controller pattern. Views are written in HTML (including HTML5). Controllers are written in Ruby. Rhodes supports iOS, Android, Windows Mobile, Research in Motion (BlackBerry), and Windows Phone 7. Symbian support was dropped after Rhodes 1.2. Rhodes applications are installed and run as native applications. However, the developer is using the web development fashion. The developer defines the user interface of the application in HTML and CSS. Then, at runtime, the HTML and CSS is rendered in a native browser UI control that is embedded in the application by the Rhodes framework. JavaScript may be used for some interaction control the same way as JavaScript usage in a web application.

There is also a possibility to add logic to the applications views using embedded Ruby (ERB), in the same way as in a Ruby on Rails application. ERB files are similar to PHP or JSP, where code can be mixed with markup to create dynamic HTML. Rhodes will generate the complete HTML, evaluating the Ruby code before the HTML is rendered by the browser UI Control, which will then dynamically execute any JavaScript that is on the page. The developers can also write Ruby code for the application logic that implements the flow of control for their application. As we mentioned before, Rhodes follows the Model-View-Controller (MVC) pattern that is similar to Ruby on Rails and other web frameworks. The concept is to implement methods in a controller to define actions that map to HTTP requests. The controller action will typically fetch data from the model (implemented in the Rhodes ORM layer, Rhom) and render a view (implemented as HTML ERB). In the figure below we present the MVC pattern illustrated with the Rhodes object model and an example use case.



**Figure 16 RhoMobile MVC model**

In the example, there is a new member page where the user can fill in the information in the fields to provide values for the new member attributes in order to register to the system. When the user clicks the RegisterButton, a request is made to a lightweight embedded web server in Rhodes that only exists to respond to these UI requests and



RhoController actions. When a user clicks on a URL in the HTML view, a controller action is called. In this example, the MemberController create method is called. The controller action then calls the Member model, implemented with Rhom to save a new Member in the local database. Then a view is rendered to display the result to the user, in our case is the member profile. The entire web response cycle happens locally on the device. Rhodes development files are compiled into a native executable that is installed on the device or run in a desktop simulator using command line tools or the web interface on [www.rhohub.com](http://www.rhohub.com). Since Rhodes apps are native binary applications, they can be submitted and distributed through the Apple iTunes App Store, BlackBerry World, Android Marketplace, and other distribution channels.

To build for a device, developers typically need to sign up for those developer programs and acquire cryptographic keys required to sign applications [12]. Revising this subparagraph we can say that RhoMobile suite is a native crossplatform solution using Ruby and the MVC model. For further reading of the MVC model we providing a link<sup>48</sup> with an MVC project we made during our postgraduate studies.

#### 4.6.3 PhoneGap

PhoneGap is a mobile development framework produced by Nitobi, purchased by Adobe Systems.<sup>24,25</sup> It enables software programmers to build applications for mobile devices using JavaScript, HTML5 and CSS3, instead of device-specific languages such as Objective-C<sup>26</sup>. The resulting applications are hybrid, meaning that they are neither truly native (because all layout rendering is done via web views instead of the platform's native UI framework) nor purely web-based (because they are not just web apps, but are packaged as apps for distribution and have access to native device APIs). From 1.9 version onward it is even possible to freely mix native and hybrid code snippets. The software underlying PhoneGap is Apache Cordova<sup>27</sup>. The software was previously called just "PhoneGap", then "Apache Callback"<sup>28,29</sup>. Apache Cordova is open source software. First developed at an iPhoneDevCamp event in San Francisco. Apple Inc. has confirmed that the framework has its approval, even with the new 4.0 developer license agreement changes<sup>30</sup>. The PhoneGap framework is used by several mobile application platforms like ViziApps,<sup>31</sup> Worklight,<sup>32,33</sup> Convertigo<sup>34,35</sup> and appMobi<sup>36</sup> as the backbone of their mobile client development engine. Adobe officially announced the acquisition of Nitobi Software (the original developer) on October 4, 2011<sup>37</sup>. Coincident with that, the PhoneGap code was contributed to the Apache Software Foundation to start a new project called Apache Cordova. The project original name, Apache Callback, was viewed as too generic<sup>38</sup>. Then it also appears in Adobe Systems as Adobe PhoneGap and also as Adobe Phonegap Build<sup>39</sup>. Early versions of PhoneGap required a person making iOS apps to have an Apple computer, and a person making Windows Mobile apps to have a computer running Windows. After September 2012, the "PhoneGap Build" service allows a programmer to upload his source code to a "cloud compiler" that generates apps for every supported platform<sup>40</sup>. The core of PhoneGap applications use HTML5 and CSS3 for their rendering, and JavaScript for their logic. Although HTML5 now provides access to underlying hardware such as the accelerometer, camera and GPS, browser support for HTML5-based device access is

not consistent across mobile browsers, particularly older versions of Android. To overcome these limitations, the PhoneGap framework embeds HTML5 code inside a native WebView on the device, using a Foreign Function Interface to access the native resources of the device<sup>41</sup>. However, the use of web-based technologies leads many PhoneGap applications to run slower than native applications with similar functionality<sup>42</sup>. Adobe Systems warns that applications built using PhoneGap may be rejected by Apple for being too slow or not feeling "native" enough (having appearance and functionality consistent with what users have come to expect on the platform)<sup>43,44</sup>.

PhoneGap currently supports development for the operating systems Apple iOS, Google Android, LG web OS, Microsoft Windows Phone, Nokia Symbian OS, RIM BlackBerry and Windows phone support<sup>45</sup>. Considering all the above the PhoneGap is a hybrid cross platform solution.

#### 4.6.4 Sencha Touch

Sencha Touch is a user interface (UI) JavaScript library, or framework, specifically built for the Mobile Web. It can be used by Web developers to develop user interfaces for mobile web applications that look and feel like native applications on supported mobile devices. It is fully based on web standards such as HTML5, CSS3 and JavaScript. Sencha Touch aims to enable developers to quickly and easily create HTML5 based mobile apps that work on Android, iOS and BlackBerry devices, and produce a native-app-like experience inside a browser. Sencha Touch is a product of Sencha, which was formed after popular JavaScript library projects Ext JS, jQTouch and Raphaël were combined<sup>49</sup>. The first release of Sencha Touch, version 0.90 beta, was made available on July 17, 2010. This beta release supported devices running Android, and iOS (on iPhone, iPod touch, iPad). Subsequently the first stable version, 1.0, was released in November 2010. Version 1.1.0 added support for devices running BlackBerry OS version 6. The latest release, Sencha Touch 2, was released on March 7, 2012 and is designed to run on WebKit based browsers such as Android, Google Chrome for Android, RIM BlackBerry Browser, Bada Mobile Browser, Kindle Fire Browser, and Safari. Sencha has announced its intention to support Windows Phone in the 2.x line. There are no plans to support Firefox Mobile. Sencha Touch includes a set of graphical user interface GUI-based controls (or components) for use within mobile web applications. These components are optimized for touch input. The components are: buttons with device specific themes and effects; form elements such as text fields for email, date picker, and address; sliders, selectors, and combo-boxes; a list component with momentum-scrolling and an index bar; a minimal icon set; toolbars and menus; movable tabs; bottom toolbars; and a map component with support for multi-touch gestures such as pinch and zoom<sup>50</sup>. All the components can be themed according to the target device. This is done using Sass, a style sheet language built over CSS. Sencha Touch has four in-built transition effects: slide over or under the current element, pop, flip, and cube. It supports common touch gestures built from touch

events, which are Web standards but supported only by Android, iOS, and some touch enabled devices. These are tap, double tap, swipe, scroll, and pinch.

#### 4.6.5 Other Cross Platform Frameworks

Apart from the four frameworks which we are going to compare later, there is a variety of frameworks with similar characteristics. Describing all known frameworks of the market is out of the scope of this project, but in this paragraph we give a quick description for many of those frameworks. QuickConnectFamily<sup>51</sup> is a framework for HTML, CSS, and JavaScript which allows the developers to build an application that runs on iPhone/iPad, Android, BlackBerry, and WebOS.

With the QuickConnectFamily templates the developers can create applications with too close to native applications performance. Bedrock Framework from Metismo<sup>52</sup> founded in August 2007. It is a cross compiler converts J2ME source code to native C++, simultaneously deploying the exported product to Android, iPhone, BREW, Windows Mobile, and more. Bedrock is a set of proprietary libraries and tools. Corona<sup>53</sup> framework is using the Lua scripting language for native iPhone, iPad, and Android apps. Corona is a proprietary framework. MoSync SDK<sup>54</sup> framework uses C or C++ as programming languages and to develop is using MoSync libraries to build for Symbian, Windows Mobile, j2me, Moblin, and Android. MoSync is also a proprietary framework. Unity<sup>55</sup> framework is a very popular game development platform which allows the developers to deploy to Mac, Windows, or iPhone. Unity supports three scripting languages: JavaScript, C#, and a dialect of Python called Boo. iWebKit<sup>56</sup> is a HTML5 and CSS3 framework targeting iOS native and web applications. iWebKit has been released under the GNU Lesser General Public License. WebApp.Net<sup>57</sup> is a lightweight, JavaScript framework to build applications that can take advantage of a WebKit browser control; namely, iOS, Android, and WebOS. Released under the Creative Commons Attribution-ShareAlike License. The DojoToolkit<sup>58</sup> is a flexible and extensible JavaScript framework, primarily used to build web applications. WidgetPad<sup>59</sup> is a collaborative, open-source mobile development environment for creating smartphone applications using standard web technologies, including CSS3, HTML5 and JavaScript. This platform includes project management, source code editing, debugging, collaboration, versioning and distribution. It can be used to create apps for OSes such as iOS, Android and WebOS. WidgetPad is currently in private beta; and there is a need for contact with the creators in order to access. MoSync<sup>60</sup> is another free and open source software (FOSS) cross-platform mobile application development SDK based on common programming standards. The SDK includes tightly integrated compilers, runtimes, libraries, device profiles, tools and utilities. MoSync features an Eclipse-based IDE for C/C++ programming. Support for JavaScript, Ruby, PHP, Python and other languages is planned. The framework supports a large number of operating systems including Android, Symbian, Windows Mobile, BlackBerry and even Moblin, a mobile Linux distro. For a concentrated full list of Cross platform frameworks information and for further reading we suggest the Wikipedia link<sup>61</sup> with the specific topic. In the next paragraph we compare the four frameworks we discussed in paragraph 4.6.

#### 4.7 Cross Platform Tools Comparison

In this paragraph we compare the four cross platform frameworks into a brief table<sup>62</sup>. We compare four frameworks for four different operating systems : iOS, Android, Windows Phone and Blackberry. We selected to present a table separated into three sections. The first section has to do with the basic characteristics of the framework such as the development languages , what type of application we can create and what is the type of the license. In the second section we present the Operating systems and the ability of support of each framework. The third section is a general reference of the capabilities of each framework, some of these capabilities are : Geolocation, Web access, SQLite, Telephone use, Vibration Image library browsing and other capabilities.

| Feature                                | Appcelerator Titanium                                   | PhoneGap                 | RhoMobile              | Senca Touch                  |
|--|---|--------------------------|------------------------|------------------------------|
| <b>Open Source License</b>             | Apache Public License v2, Proprietary                   | Apache Public License v2 | MIT                    | GPL v3 (+commercial edition) |
| <b>Framework target</b>                | Embedded applications                                   | Embedded applications    | Embedded applications  | Web applications             |
| <b>Development languages</b>           | HTML, JavaScript, (PHP, Ruby & Python for Desktop apps) | HTML, JavaScript and CSS | HTML, JavaScript, Ruby | HTML5, CSS3, JavaScript      |
| <b>Platforms</b>                       |   |                          |                        |                              |
| <b>iOS</b>                             | Yes   | Yes                      | Yes                    | Yes                          |
| <b>Android</b>                         | Yes   | Yes                      | Yes                    | Yes                          |
| <b>Windows Phone</b>                   | Not clear yet   | Yes                      | Yes                    | Yes                          |
| <b>BlackBerry</b>                      | Yes   | Yes                      | Yes                    | Yes                          |
| <b>General</b>                         |   |                          |                        |                              |
| <b>Self contained, no web required</b> | Yes   | Yes                      | Yes                    | Yes<br>Offline support       |
| <b>Able to access the web for data</b> | Yes   | Yes                      | Yes                    | Yes                          |
| <b>Geolocation</b>                     | Yes   | Yes                      | Yes                    | No                           |
| <b>Vibration</b>                       | Yes   | Yes                      | Yes                    | No                           |
| <b>Accelerometer</b>                   | Yes   | Yes                      | Yes                    | No                           |
| <b>Sound (play)</b>                    | Yes   | Yes                      | Yes                    | No                           |
| <b>File system IO</b>                  | Yes   | Yes                      | Yes                    | No                           |
| <b>Maps</b>                            | Yes   | Yes                      | Yes                    | No                           |
| <b>Telephone</b>                       | Not clear yet   | Yes                      | Yes                    | No                           |
| <b>SQLite</b>                          | Yes   | Yes apart from           | Yes                    | No                           |

|  |                        |            |     |    |
|--|------------------------|------------|-----|----|
|  |                        | blackberry |     |    |
| <b>File uploading</b>                          | Not clear yet          | Yes        | Yes | No |
| <b>Image library browsing</b>                  | Not clear yet          | No         | Yes | No |
| <b>Distribution Analytics</b>                  | Yes via cloud services | No         | No  | No |
| <b>Native Language Application Development</b> | Yes                    | No         | Yes | No |

**Table 7 Cross platform software support**

#### 4.8 Conclusions About Frameworks

Fortunately for the developers there are numerous platforms for creating mobile cross platform applications. In the previous paragraphs we discussed four different cross platform frameworks the first two were native cross platform development solutions and the last two were hybrid cross platform development solutions. Every system has its advantages depending the needs of the individual developers or the organizations. Appcelerator titanium is a good native solution for those who are experts of some powerful languages such as python, Ruby and PHP. RhoMobile is a great solution for those who are familiar with the Model View Controller (MVC) model. Finally the PhoneGap and the Senca Touch are hybrid solutions suitable for web technology knowledge in combination with fast results.

## 5. Requirement Studies

### 5.1 Feasibility Study

The theoretical part of this project was mentioned till chapter four. From now the rest of the project is the practical part. Before we proceed with the development part of our project there is a need to check some critical factors. These factors are the TELOS [13] factors widely used in project management and specifically in feasibility studies [14]. The acronym TELOS refers to the five areas of feasibility - Technical, Economic, Legal, Operational, and Scheduling. In the table below we present these factors and then we are going to make a feasibility study according to our project.

|                    |   |
|--------------------|---|
| T - Technical      | Is the project technically possible?                  |
| E - Economic       | Can the project be afforded? Will it increase profit? |
| L - Legal          | Is the project legal?                                 |
| O - Organizational | Will the organization accept the change?              |
| S - Scheduling     | Can the project be done in time?                      |

**Table 8 TELOS factors**



### 5.1.1 Technical Factor

From Technical scope the project based in common software and hardware. The Hardware infrastructure is an IBM compatible PC with 2.1Ghz Dual Core Processor, 4 Giga bytes of RAM and Internet access. The mobile device we used for our tests is a Sony Xperia U with dual core processor and Android Operating system. As a software infrastructure we based the whole project on Windows 7 - 32 bit professional edition. The integrated development environment will be eclipse with all the add-ons and the framework setup methods required.

Additional tools like notepads and graphical tools like GNU Image Manipulation Program (Gimp) will be used for supplementary reasons. It is useful to mention that the backups of the whole project including useful references, code etc was done by the free Google drive cloud storage service.

### 5.1.2 Economic Factor

From Economic scope the project costs where increased because of the open source software usage, the universities contracts with software companies and its own technical infrastructures such as laboratories and an updated library. The only cost where an android Sony xperia U device which is considered to be a low end device compared with other devices. Profit can be achieved with ways which we describe in paragraph 7.4.

### 5.1.3 Legal Factor

One big issue when we create projects is the legitimacy of the tools we use and the whole project also. Our tools are legal because they are FOSS based tools with open licenses. The operating system of our personal computer is licensed through the Digital Systems department of university of Piraeus with Microsoft Corporation via a Dream spark Academic account. Below there is a brief table with licenses. The framework we used for creating the cross platform application was Phone Gap not only for license reasons but for many other reasons explained in paragraph 5.2. The whole project is using new practices to cover an example of cross platform development methods and the content of the project is not offending to any social group or ideology.

| Software                      | Type                 | License                                      |
|-------------------------------|----------------------|--|
| Windows 7 Professional 32 bit | Proprietary Software | Academic Dream spark Account (serial number) |
| Eclipse                       | Open Source software | Eclipse Public License (EPL)                 |
| PhoneGap                      | Open Source software | Apache 2.0 License                           |
| Android                       | Open Source software | Apache License 2.0                           |

|            |                      |                                       |
|------------|----------------------|---------------------------------------|
|            |                      | Linux kernel patches under GNU GPL v2 |
| Notepad ++ | Open Source software | GPL                                   |
| Gimp       | Open Source software | GNU (L)GPLv3+ v2.7+                   |

**Table 9 Software licenses**

#### 5.1.4 Organizational Factor

From organizational scope the project is a part of our master thesis for the Digital Systems Department of the university of Piraeus with title “Cross Platform Mobile Application for Mobile Devices with Social Media and Geolocation Services”. Considerations , issues and proposals are made during the discussion with the supervisor Dr Marinos Themistokleous. A three member professor committee will exam the whole project at the end of the second academic semester of 2013.

#### 5.1.5 Scheduling Factor

Lastly the schedule of the project was made with two axis the first axis was the often meetings and email exchange with the supervisor in order to improve the project. The second axis where the important docs in order the supervisor to check the important milestones of the project. The important documents which we shared with the supervisor where displayed at the table below.

| Document           | Purpose                                   |
|--------------------|---|
| The daily Progress | A daily and weekly record of what is done |
| The targets        | The general targets of the month          |
| The Thesis draft   | The draft of the thesis project           |

**Table 10 scheduling documents**

The daily progress document is a record of a daily and weekly project process. The targets document includes the targets of the month and the milestones. The thesis draft document is the whole progress of the project till a specific moment without corrections. The deadline we set for the completion of the project is the middle of October 2013.

#### 5.2 Tool Selection

After one week of considerations we decided to select the PhoneGap framework. We based on four criteria in order to make our final selection, the innovation, the ease of use, the new knowledge gain and the cost. From the scope of innovation PhoneGap is a complete hybrid cross platform application framework purchased by Adobe which is one of the leading software companies worldwide. From the scope of ease of use we state that PhoneGap framework has a good documentation with is a plug-in way of setup on eclipse IDE, which is one of our favorite environments. There is also a full support for android devices in which there is previous experience. The most important reason for selecting PhoneGap was the knowledge gain. With PhoneGap frameworks we have the opportunity to test our skills and learn more coding practices in web technologies such

as Java Script , HTML5 and CSS. The cost is one critical factor and in our case phonegap uses the free Apache 2.0 License. The hybrid way is definitely a new way of developing mobile applications using the native assets of the known platforms of the market. In next subparagraphs we describe the three web technologies and the way they function before we proceed with PhoneGap.

### 5.2.1 JavaScript

JavaScript is the programming language of the Web. The overwhelming majority of modern websites use JavaScript, and all modern web browsers—on desktops, game consoles, tablets, and smart phones—include JavaScript interpreters, making JavaScript the most ubiquitous programming language in history. JavaScript is part of the triad of technologies that is a must for the most developers: HTML to specify the content of web pages, CSS to specify the presentation of web pages, and JavaScript to specify the behavior of web pages [15].

JavaScript (JS) is an interpreted computer programming language. It was originally implemented as part of web browsers so that client-side scripts could interact with the user, control the browser, communicate asynchronously, and alter the document content that was displayed. More recently, however, it has become common in both game development and the creation of desktop applications. JavaScript is a prototype-based scripting language that is dynamic, is weakly typed, and has first-class functions. Its syntax was influenced by the language C. JavaScript copies many names and naming conventions from Java, but the two languages are otherwise unrelated and have very different semantics. The key design principles within JavaScript are taken from the Self and Scheme programming languages.<sup>64</sup> It is a multi-paradigm language, supporting object-oriented,<sup>65</sup> imperative, and functional programming styles. JavaScript's use in applications outside of web pages—for example, in PDF documents, site-specific browsers, and desktop widgets—is also significant. Newer and faster JavaScript VMs and frameworks built upon them (notably Node.js) have also increased the popularity of JavaScript for server-side web applications. JavaScript was formalized in the ECMAScript language standard and is primarily used as part of a web browser (client-side JavaScript). This enables programmatic access to computational objects within a host environment. JavaScript was originally developed in Netscape, by Brendan Eich. Battling with Microsoft over the Internet, Netscape considered their client-server solution as a distributed OS, running a portable version of Sun Microsystems' Java. Because Java was a competitor of C++ and aimed at professional programmers, Netscape also wanted a lightweight interpreted language that would complement Java by appealing to nonprofessional programmers, like Microsoft's Visual Basic. Developed under the name Mocha, LiveScript was the official name for the language when it first shipped in beta releases of Netscape Navigator 2.0 in September 1995, but it was renamed JavaScript<sup>66</sup> when it was deployed in the Netscape browser version 2.0B3. The change of name from LiveScript to JavaScript roughly coincided with Netscape adding support for Java technology in its Netscape Navigator web browser. The final choice of name caused confusion, giving the impression that the language was a spin-off of the Java programming language, and the choice has been characterized by many as a marketing ploy by Netscape to give JavaScript the cachet of what was then the hot new



web programming language. "JavaScript" is a trademark of Oracle Corporation. It is used under license for technology invented and implemented by Netscape Communications and current entities such as the Mozilla Foundation. In the table below we demonstrate a simple function written in JavaScript and how to call it:

```
var divideByThree = function (number) {  
  
    var val = number / 3;  
  
    console.log(val);  
  
}; // then we have to call this simple function  
  
divideByThree(9);
```

**Table 11 JS sample**

### 5.2.2 HTML5

HTML5 is a markup language for structuring and presenting content for the World Wide Web and a core technology of the Internet. It is the fifth revision of the HTML standard (created in 1990 and standardized as HTML 4 as of 1997)<sup>67</sup> and, as of December 2012, is a W3C Candidate Recommendation<sup>68</sup>. Its core aims have been to improve the language with support for the latest multimedia while keeping it easily readable by humans and consistently understood by computers and devices (web browsers, parsers, etc.). HTML5 is intended to subsume not only HTML 4, but also XHTML 1 and DOM Level 2 HTML<sup>67</sup>. Following its immediate predecessors HTML 4.01 and XHTML 1.1, HTML5 is a response to the observation that the HTML and XHTML in common use on the World Wide Web are a mixture of features introduced by various specifications, along with those introduced by software products such as web browsers, those established by common practice, and the many syntax errors in existing web documents<sup>69</sup>. It is also an attempt to define a single markup language that can be written in either HTML or XHTML syntax. It includes detailed processing models to encourage more interoperable implementations; it extends, improves and rationalizes the markup available for documents, and introduces markup and application programming interfaces (APIs) for complex web applications. For the same reasons, HTML5 is also a potential candidate for cross-platform mobile applications. Many features of HTML5 have been built with the consideration of being able to run on low-powered devices such as smartphones and tablets. In particular, HTML5 adds many new syntactic features. These include the new <video>, <audio> and <canvas> elements, as well as the integration of scalable vector graphics (SVG) content (that replaces the uses of generic <object> tags) and MathML for mathematical formulas. These features are designed to make it easy to include and handle multimedia and graphical content on the web without having to resort to proprietary plugins and APIs. Other new elements, such as <section>, <article>, <header> and <nav>, are designed to enrich the semantic content of documents. New attributes have been introduced for the same purpose, while some elements and attributes have been removed. Some elements, such as <a>, <cite> and <menu> have been changed, redefined or

standardized. The APIs and Document Object Model (DOM) are no longer afterthoughts, but are fundamental parts of the HTML5 specification<sup>70</sup>. HTML5 also defines in some detail the required processing for invalid documents so that syntax errors will be treated uniformly by all conforming browsers and other user agents<sup>71</sup>. In general HTML5 is still used for element interconnection [16]. In the table below there is an example of HTML5 code as presented by w3schools.com<sup>72</sup>.

```
<!DOCTYPE html>

<html>

<body>

<video width="320" height="240" controls autoplay>
  <source src="movie.mp4" type="video/mp4">
  <source src="movie.ogv" type="video/ogg">
  Your browser does not support the video tag.
</video>

</body>

</html>
```

**Table 12 HTML sample**

### 5.2.3 CSS

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation semantics (the look and formatting) of a document written in a markup language. Its most common application is to style web pages written in HTML and XHTML, but the language can also be applied to any kind of XML document, including plain XML, SVG and XUL. CSS is designed primarily to enable the separation of document content (written in HTML or a similar markup language) from document presentation, including elements such as the layout, colors, and fonts<sup>73</sup>. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple pages to share formatting, and reduce complexity and repetition in the structural content (such as by allowing for table less web design). CSS can also allow the same markup page to be presented in different

styles for different rendering methods, such as on-screen, in print, by voice (when read out by a speech-based browser or screen reader) and on Braille-based, tactile devices. It can also be used to allow the web page to display differently depending on the screen size or device on which it is being viewed. While the author of a document typically links that document to a CSS file, readers can use a different style sheet, perhaps one on their own computer, to override the one the author has specified. CSS specifies a priority scheme to determine which style rules apply if more than one rule matches against a particular element. In this so-called cascade, priorities or weights are calculated and assigned to rules, so that the results are predictable. The CSS specifications are maintained by the World Wide Web Consortium (W3C). Internet media type (MIME type) text/css is registered for use with CSS by RFC 2318 (March 1998), and they also operate a free CSS validation service<sup>74</sup>. Concluding Cascading Style Sheets (CSS) are the tool that web designers and developers use alongside markup languages such as HTML and XHTML to build websites.

CSS provides web browsers with the information they need to control the visual aspect of a web page, such as the position of HTML elements, text styles, backgrounds, colors and images, and much more. Advanced CSS techniques give website authors the ability to tailor layouts and designs for mobile web browsers, as well as the skills they need to create websites for regular desktop browsers. I will introduce you to the basics of writing CSS for mobile devices [17]. In the table below there is an example of CSS code as presented by w3schools.com<sup>75</sup>.

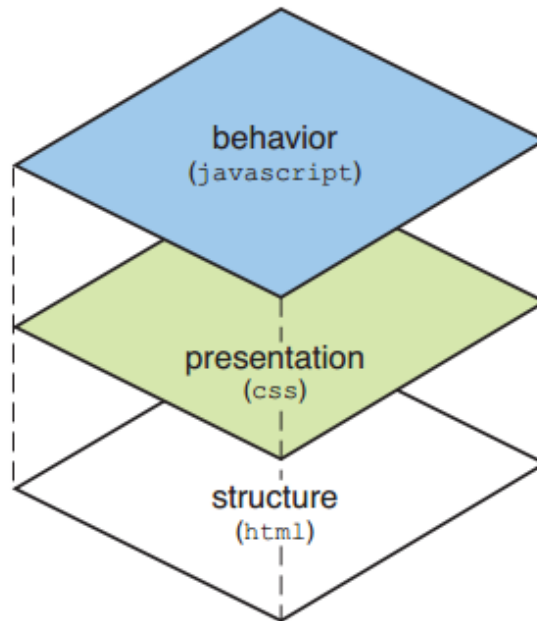
```
body
{
background-color:#d0e4fe;
}
h1
{
color:orange;
text-align:center;
}
p
{
font-family:"Arial";
font-size:20px;
}
```

**Table 13 CSS sample**

#### **5.2.4 The Progressive Enchantment of Web Technologies**

Progressive enhancement is the fundamental base for all front-end development. At its most basic level, it is creating a functional separation between HTML, CSS, and JavaScript. Progressive enhancement is a layered approach to Web design, where focus is put on content, the user, and accessibility. The first step is keeping your HTML, CSS, and JavaScript separated, but we don't refer to them as HTML, CSS, and

JavaScript. We refer to these three “layers” as structure, presentation, and behavior, probably so the methodology can be accurately applied to other areas beyond the current state of Web design. Regardless, it is a bottom-up or inside out building model for a website or application. We first focus on the content and mark it up with semantic and meaningful HTML. This is the first layer, “structure.” After the content is properly marked up, we can move onto layer two, “presentation.” On the presentation layer, we deal with CSS. The third layer of progressive enhancement, “behavior,” we deal with last. This is where we will be spending a lot of time because this is where the JavaScript lives. The next figure shows the different layers of Web design [18].



**Figure 17 Graphical representation of progressive enchantment**

The interesting thing about having these three layers is that they are never intended to touch each other, yet they’re all integrated—as we move up the ladder, the next layer is dependent on the previous. JavaScript needs CSS, and CSS needs HTML. This ordering is set up so we can remove each layer from top to bottom and we never lose the most important aspect of our site: the content. As long as we keep our layers separate, we make our site work with only HTML, pretty it up with CSS, and then smooth out the behavior with JavaScript, we will make sure that your content is always accessible. For the history we mention that in 2003 in Austin, TX, at South by Southwest, a new term was coined that realigned the Web with the original path of Tim Berners-Lee; this term was announced as progressive enhancement, and ever since, the face of Web design has been changed. The way we build and think moved from focusing on machines (browsers) to a friendlier model, which centered around people (users). We started realizing that content was more important than decoration. We cared about users and we cared about getting them to the content they were seeking as

quickly and as easily as possible. Progressive enhancement changed the Web. The purpose of progressive enhancement is to put the importance on content, which makes

Perfect sense; it's why people visit our site. Although the main focus is on content, it also takes technology into account. Suppose we're building a site with progressive enhancement and beautiful JavaScript animations, and users visit our site with JavaScript turned off in their browser. They won't see any of the JavaScript enhancements we added in with such care, but it shouldn't matter because we can always peel back the layers and effectively get to the content.

Keeping our layers in separate files will also guarantee that if users were to visit without JavaScript, because it's in an external file, the users wouldn't have to waste the bandwidth to download all that code that they won't be using. So it addresses performance issues as well. Putting so much emphasis on content makes sure that our content is always accessible, no matter what. When we mention the term accessibility, we probably thinking about disabled users, and we would be right to assume that, but there is more to it than that. Maybe there's a slow connection or tight bandwidth, but 9 out of 10 times the content will still be able to render because it will be meaningful and lightweight by not being bogged down with other layers. Keeping our HTML clean, meaningful, and semantic will ensure that our content is easily consumable by everything from a constricted bandwidth to a disabled user accessing our site through screen reading software. Keeping all our layers separated raises the reusability factor of our code. We will be able to not only use the same code over and over throughout the same site, but as we move forward in our design and developing career, we'll be able to reuse code from past projects. If the layer were not separated, a specific line of JavaScript or CSS would be very difficult to locate in a project and reused somewhere else, unless the development conditions were exactly the same—and they never are. Concluding the progressive enchantment of web technologies we can say that the use of progressive enhancement in conjunction with HTML, CSS, and JavaScript not only serve our users well, but provide them with a fantastic experience, no matter what browser or device they are using to access it [19].

### 5.3 Tool Installation

This step requires some knowledge of HTML, JavaScript, CSS, XML, and Eclipse IDE. For the purposes of installation we downloaded the software mentioned in the table below.

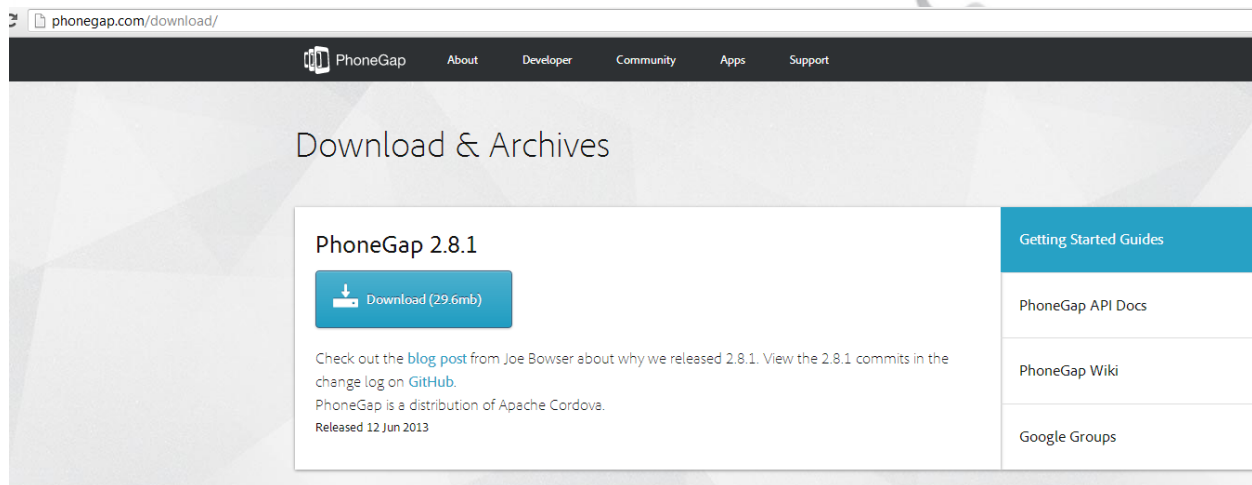
| Software Name   | Site  |
|-----------------|---|
| Eclipse classic | <a href="http://www.eclipse.org/downloads/">http://www.eclipse.org/downloads/</a>                     |
| Android SDK     | <a href="http://developer.android.com/sdk/index.html">http://developer.android.com/sdk/index.html</a> |
| PhoneGap        | <a href="http://phonegap.com/download">http://phonegap.com/download</a>                               |

Table 14 Software web sites

The software in the above table is recommended for our project but the installation of Eclipse and android SDK was done via the ADT bundle in chapter four. We are going to work with the existing installation of eclipse in order to install the PhoneGap.

### 5.3.1 Download the Tool

The first step is to download PhoneGap from <http://phonegap.com/download>. We are going to use PhoneGap 2.8.1 which is the release of 12 June 2013. From this point there is a requirement of knowledge of basic JavaScript, XML, HTML.



**Figure 18 Download PhoneGap**

Then we save the file in C:\Users\yourdir\Downloads and we extract it for later use.



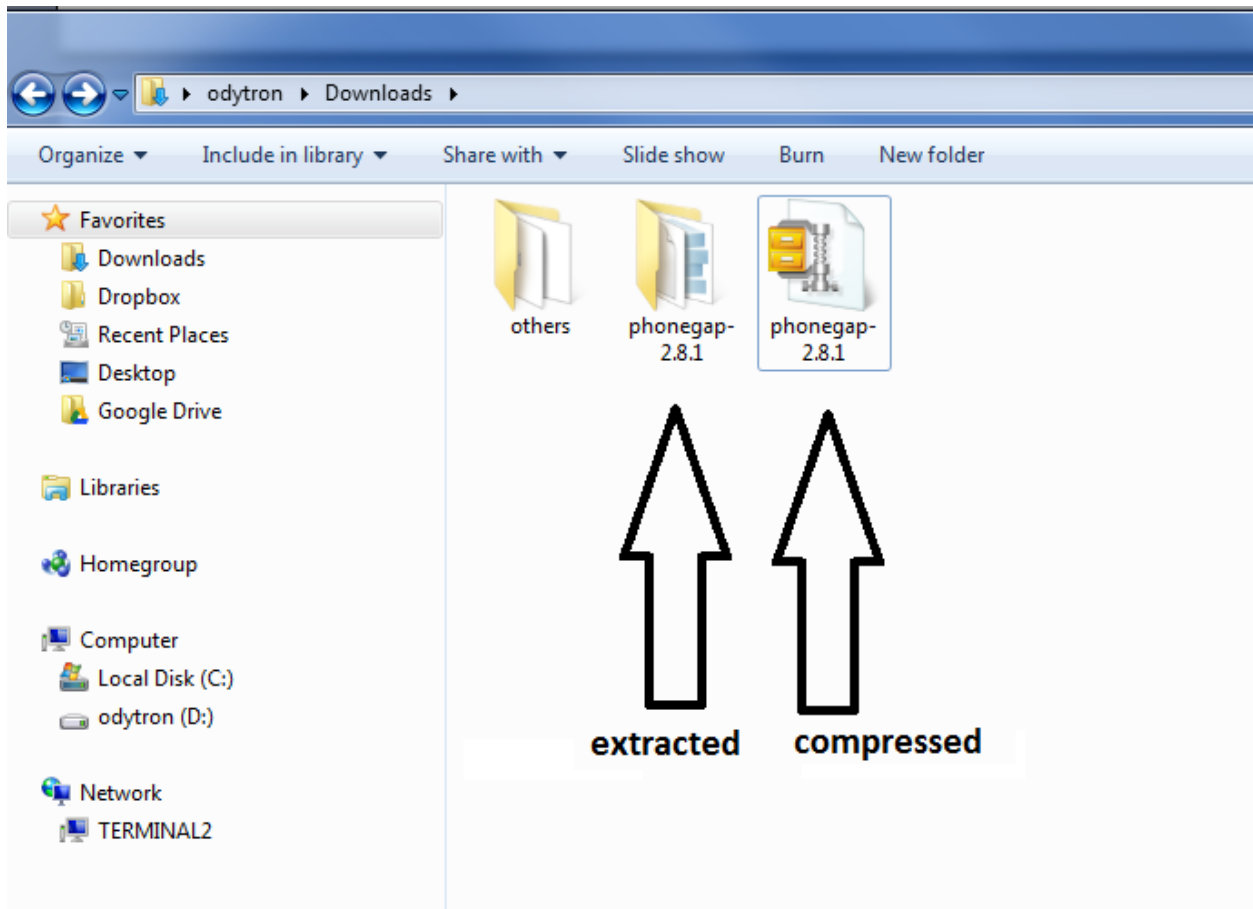
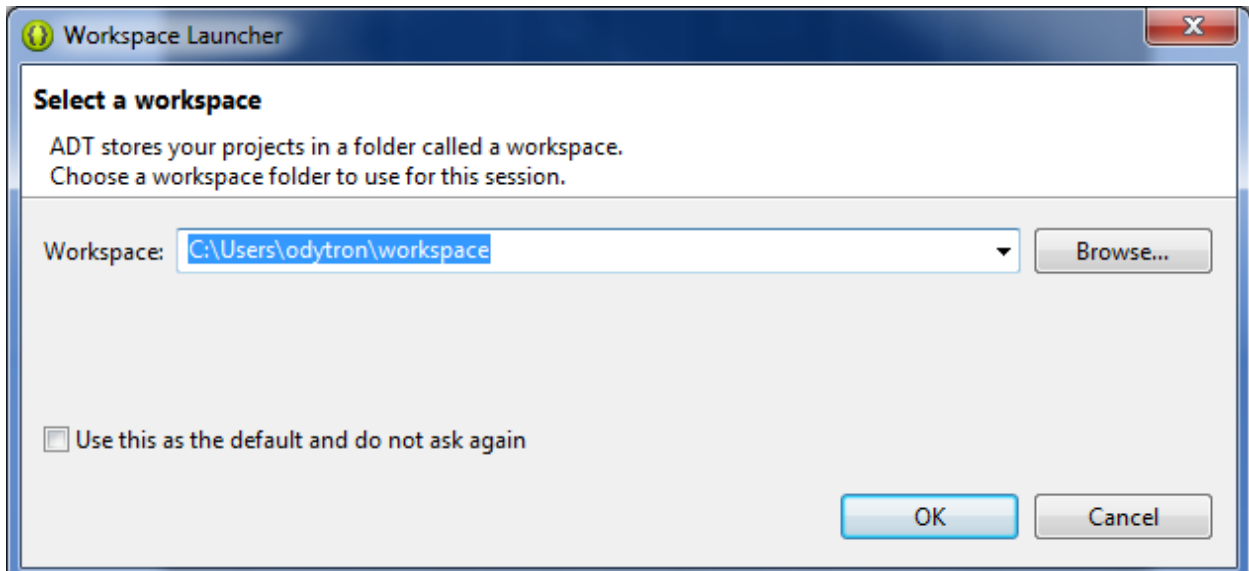


Figure 19 Extract PhoneGap

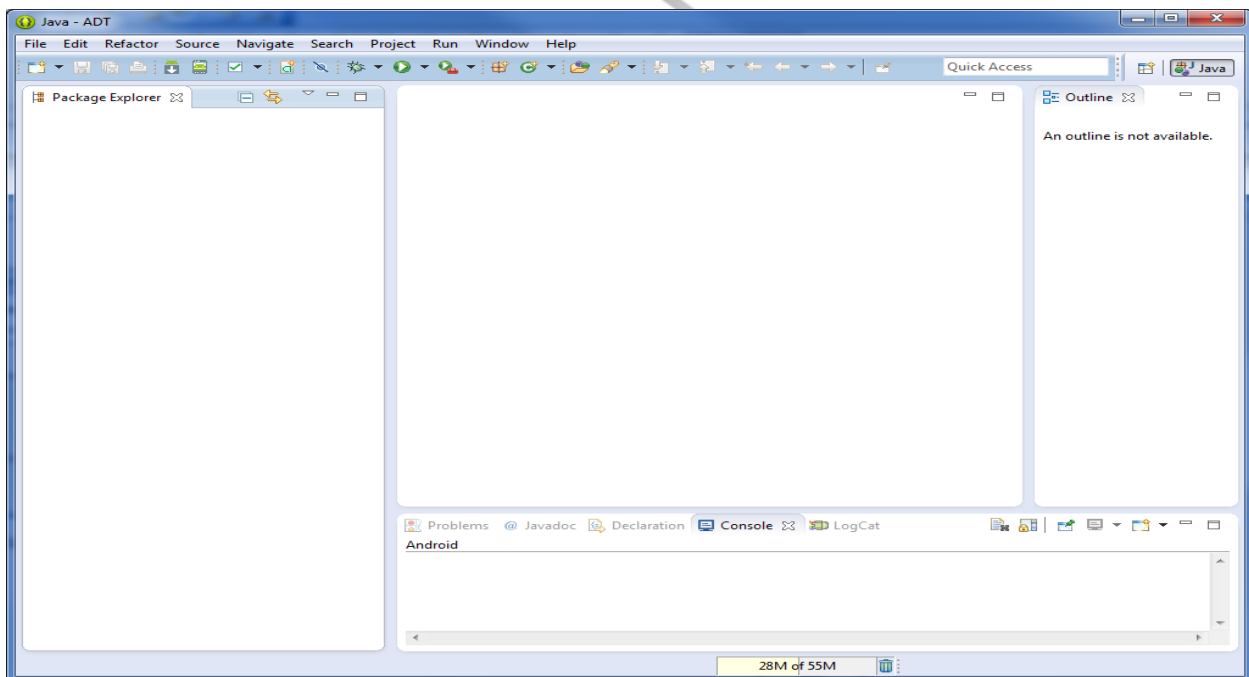
### 5.3.2 Open Eclipse - Create Project

Then we open the eclipse and we select the location of our workspace for the rest of the procedure.



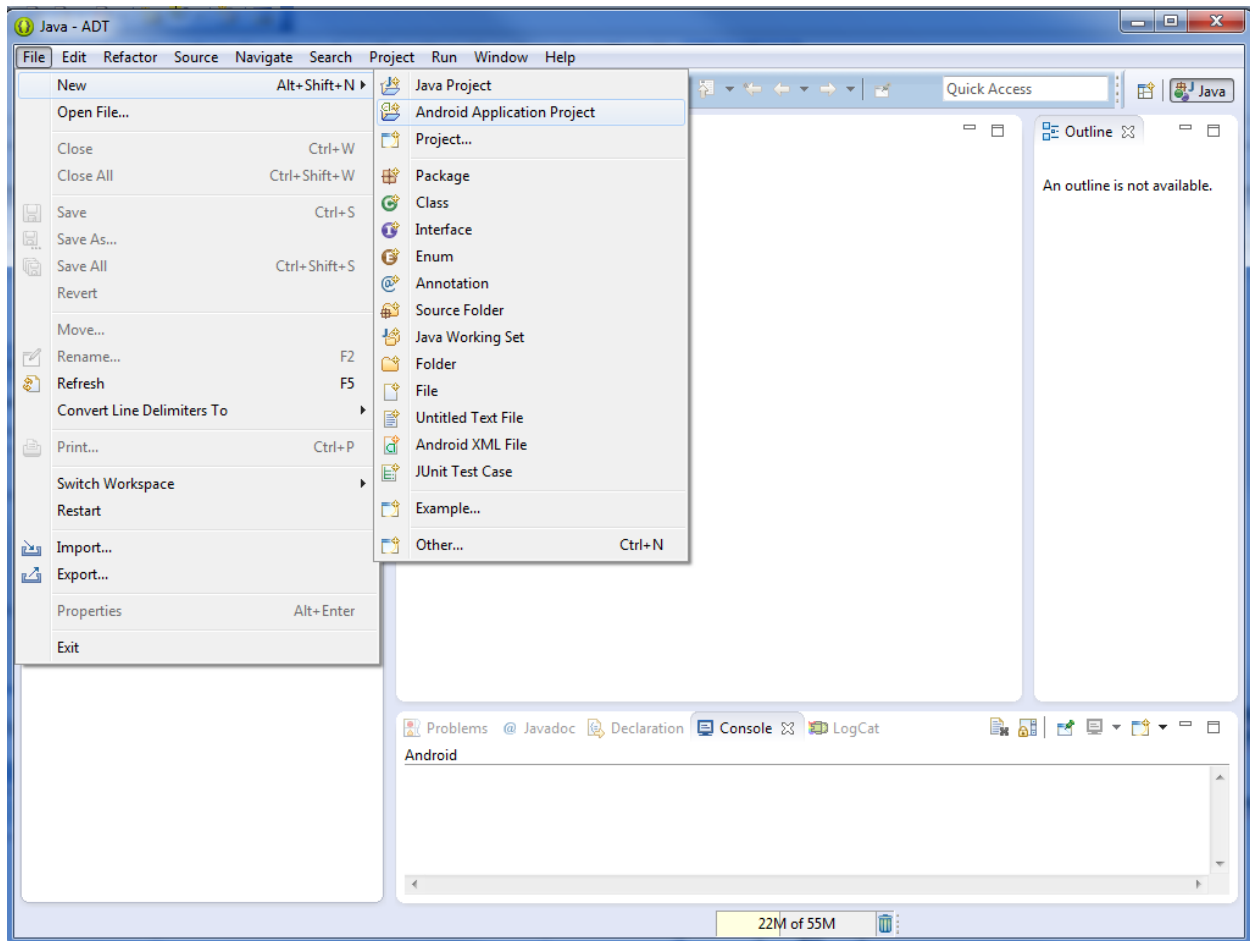
**Figure 20 Select a workspace**

Then we face the known eclipse environment



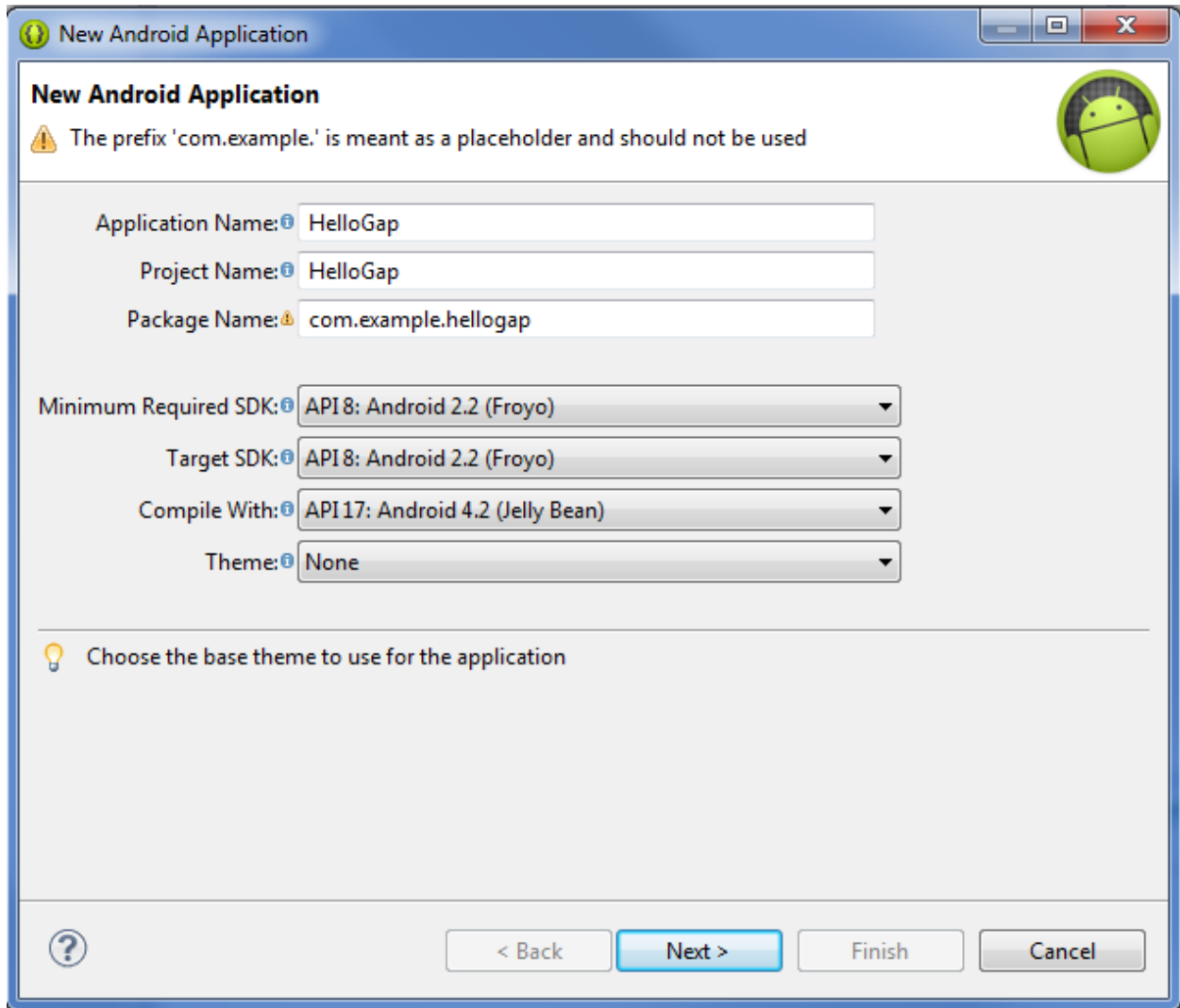
**Figure 21 The Known eclipse Environment**

After this we choose File → New → Android Application Project



**Figure 22 Create new Project**

Then we configure the project by giving an application name, a project name, a package name, the minimum required SDK, The target SDK , the compile with API and the theme. We choose API18: Android 2.2 (Froyo) for the first two options.



**Figure 23 Set project name**

**Next** we configure the icon , the activity and the location of the project

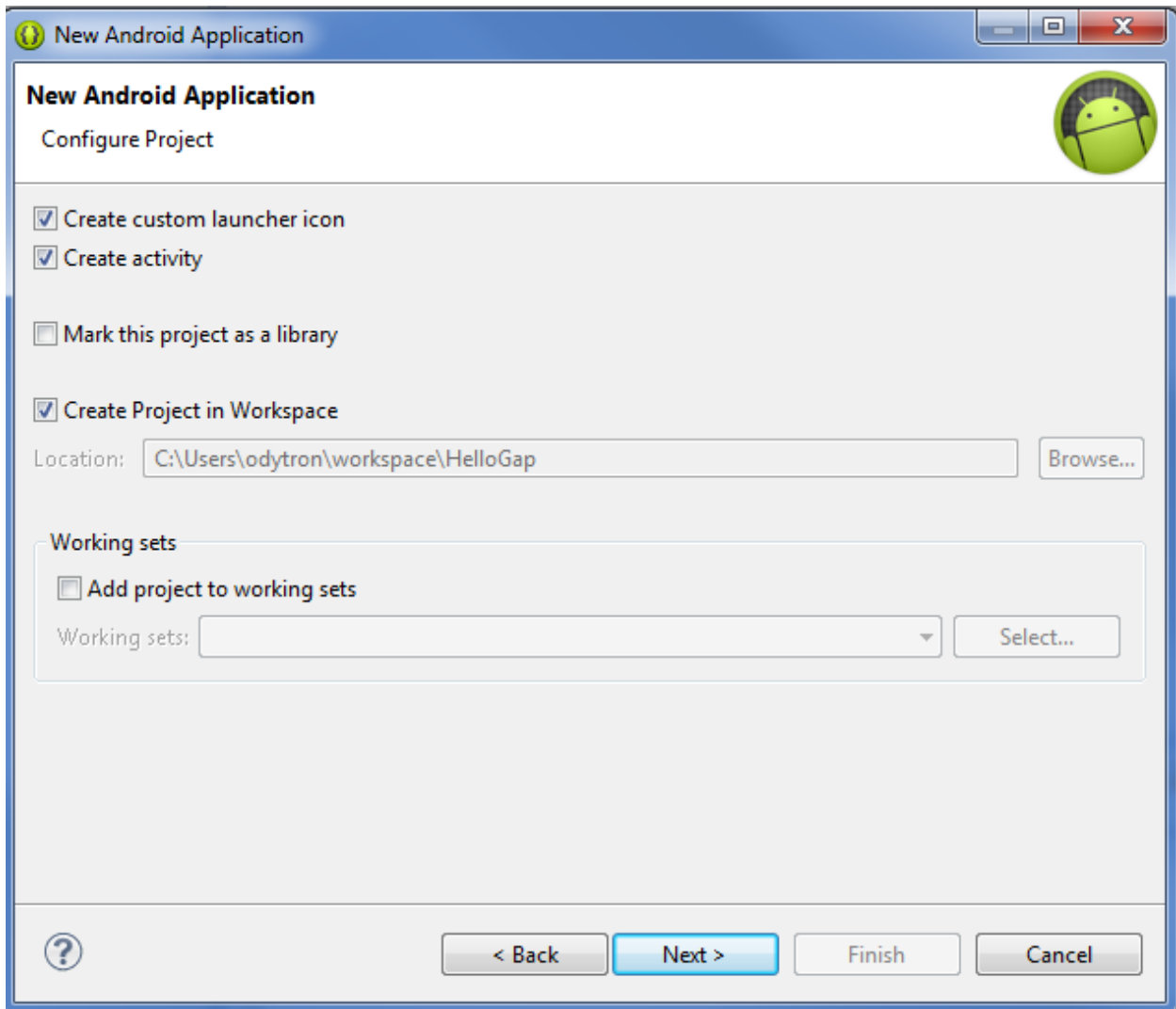


Figure 24 Final confirm

What comes next is to configure of the launcher icon for our application, we leave the defaults.

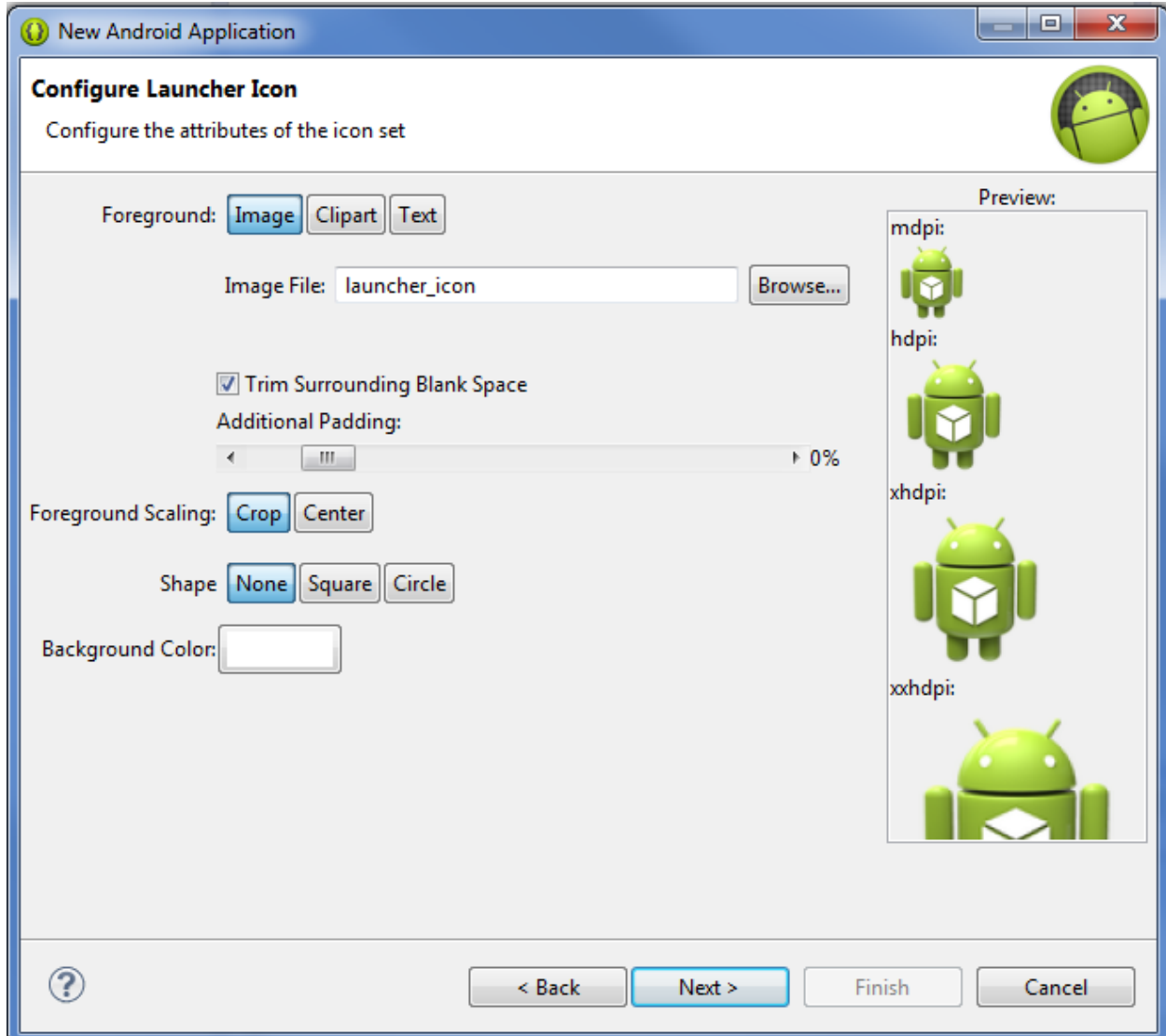


Figure 25 Configure launcher icon

The last configuration for now is to set the kind of activity. We also leave the defaults and press finish.



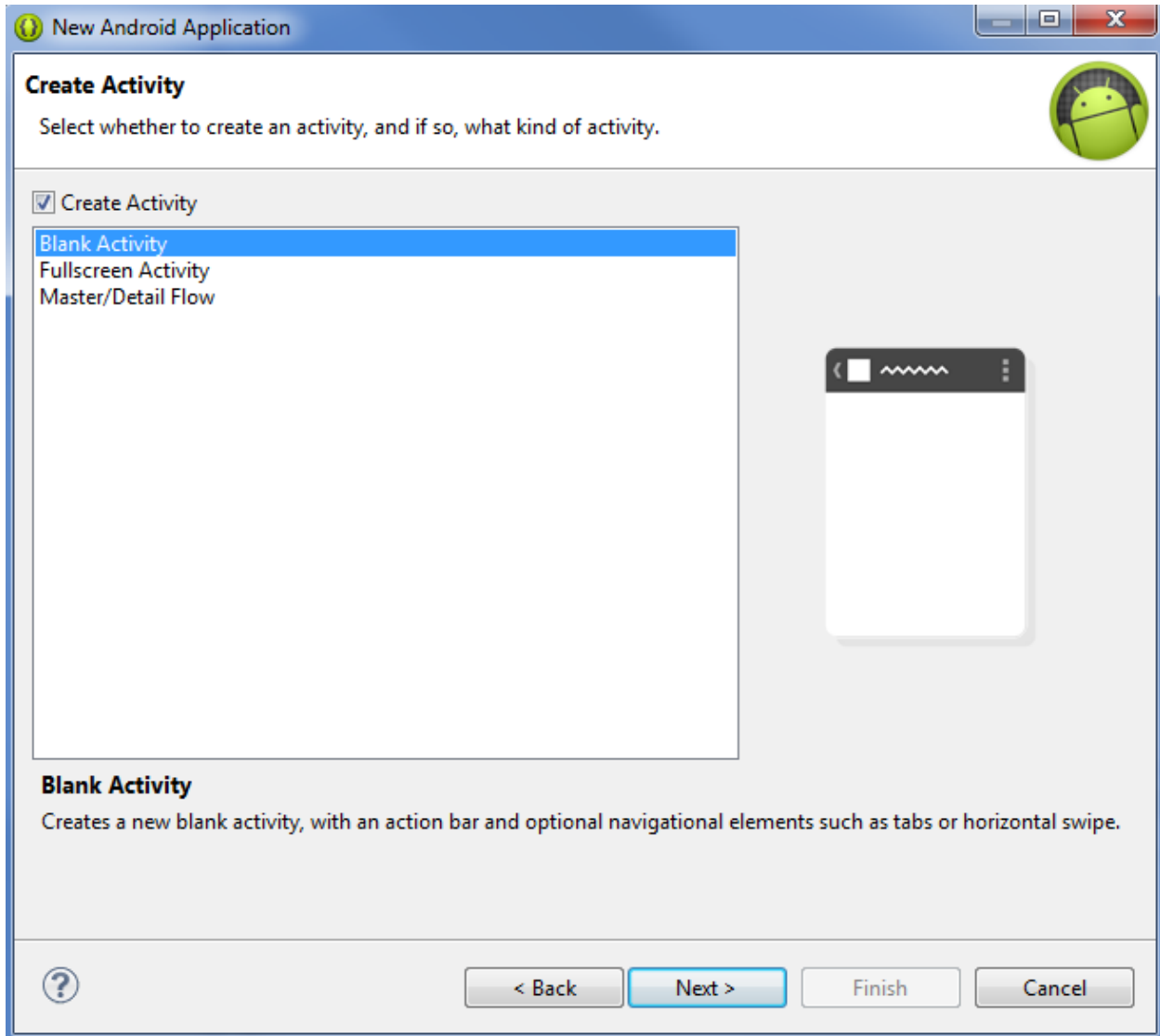
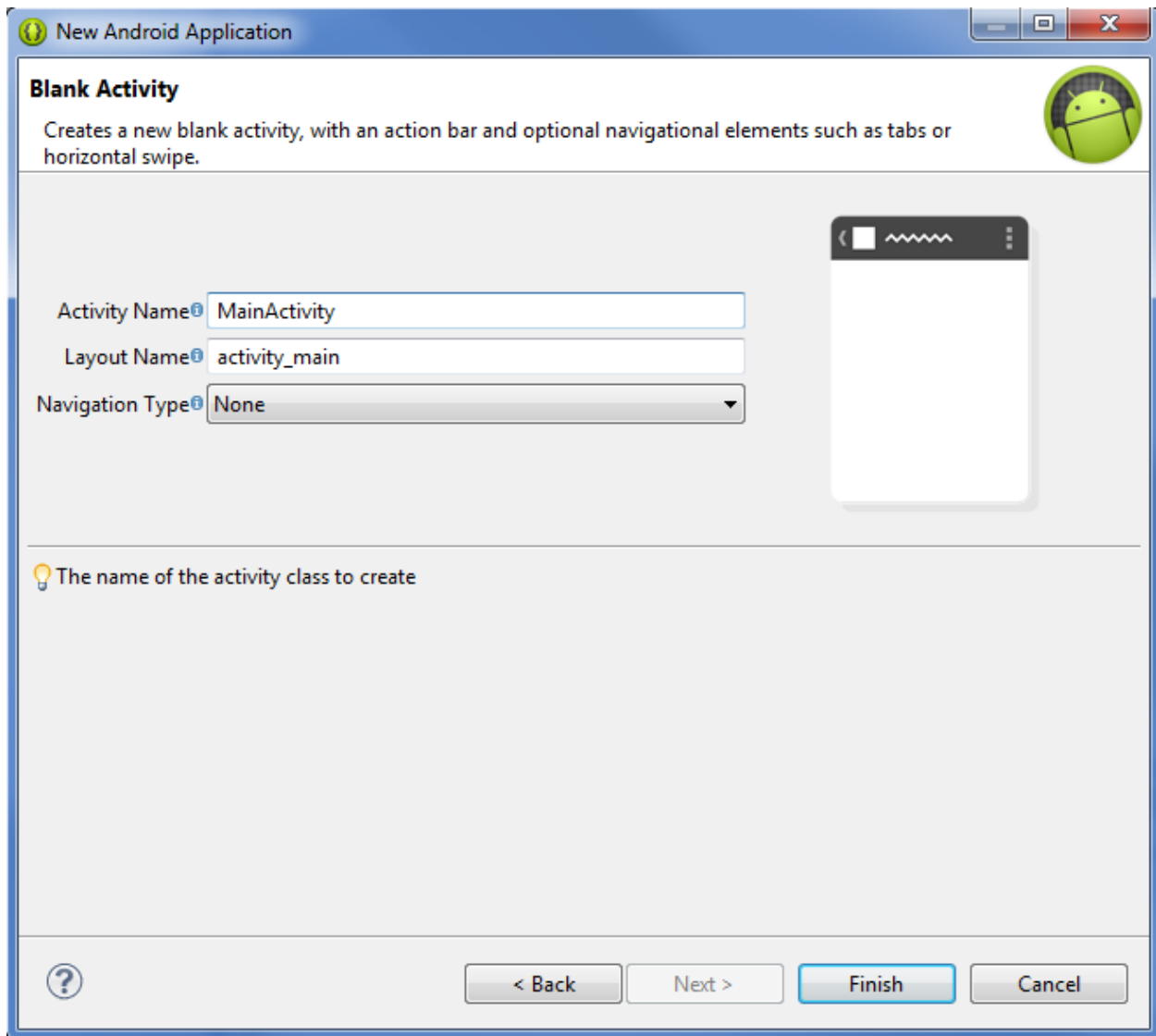


Figure 26 Create activity



**Figure 27 Black Activity**

Now as we can see from the picture below the project in our eclipse environment is ready as an empty project. However the project is not a PhoneGap project and we need to make some changes.

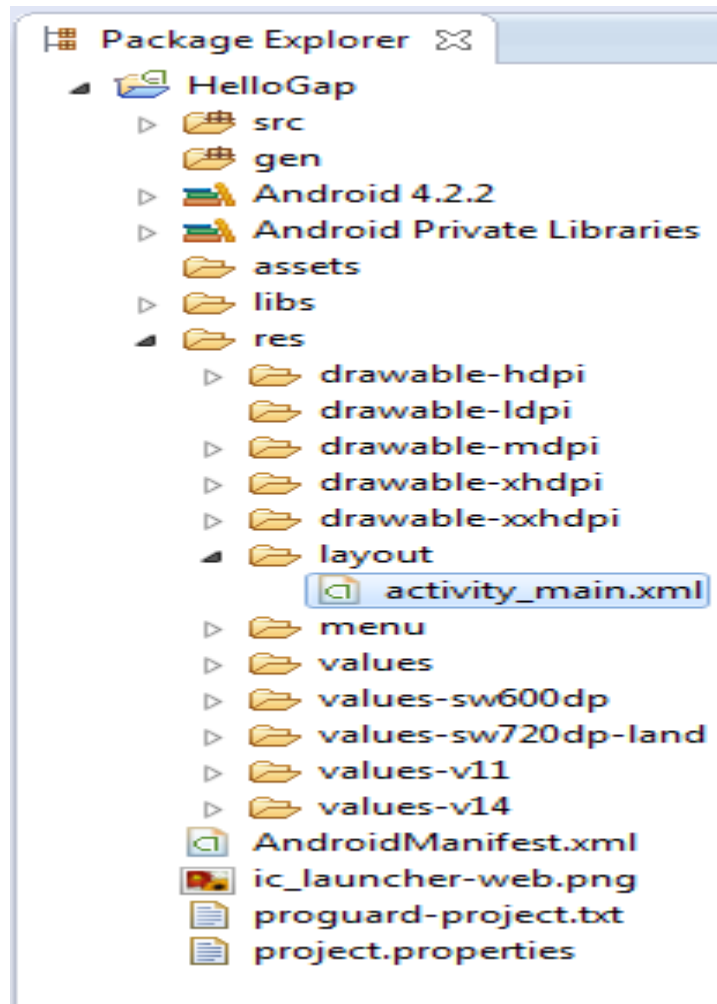
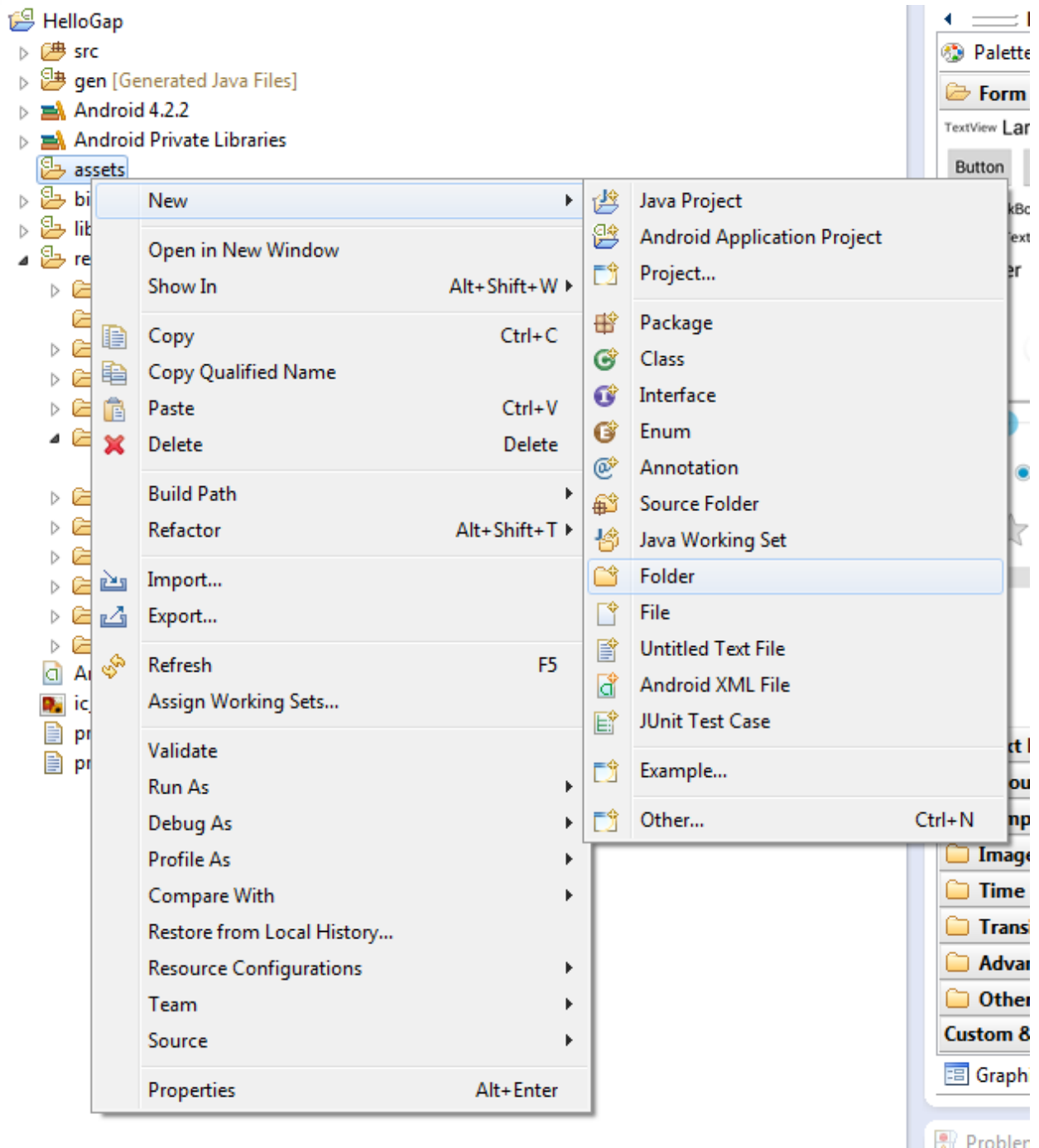


Figure 28 Package explorer

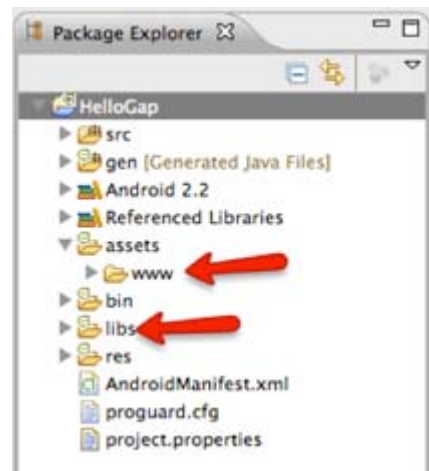
### 5.3.3 Configuring Project for PhoneGap

Next we are configuring the project to create a PhoneGap application in the folder assets we create a folder called “www”



**Figure 29** New folder

Then we have a folder with the name libs which is very important and we will explain why. All of the HTML and JavaScript for your PhoneGap application interface will reside within the assets/www

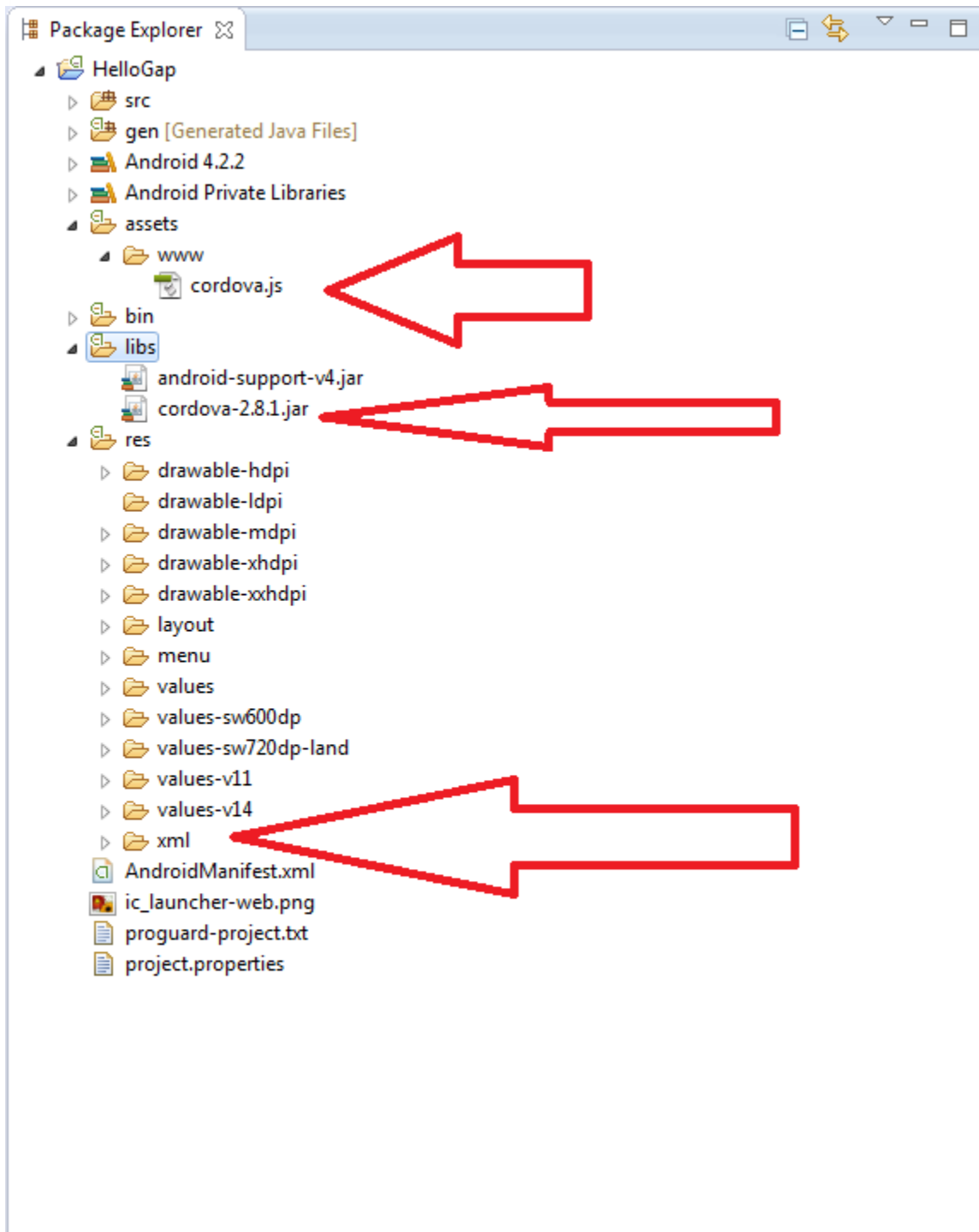


**Figure 30 Important folders**

Then we have to copy the files of PhoneGap into our project. We navigate to the folder where we downloaded the PhoneGap and we proceed with the next steps

1. We copy cordova.js from C:\Users\odytron\Downloads\phonegap-2.8.1\lib\android\ to the assets/www directory within our Android project.
2. We copy cordova-2.8.1.jar from C:\Users\odytron\Downloads\phonegap-2.8.1\lib\android\ to the libs directory within our Android project.
3. We copy the xml directory from C:\Users\odytron\Downloads\phonegap-2.8.1\lib\android\ into the res directory within our Android project.

After the above steps we press F5 to refresh our project and show the folders with the new files in them. If with the refresh the eclipse still doesn't show the files then restart eclipse. The picture below shows the placement of required files and folders in our eclipse project.



**Figure 31 Files Placement**

Next, we create a file named index.html in the assets/www folder. This file will be used as the main entry point for our PhoneGap application's interface.



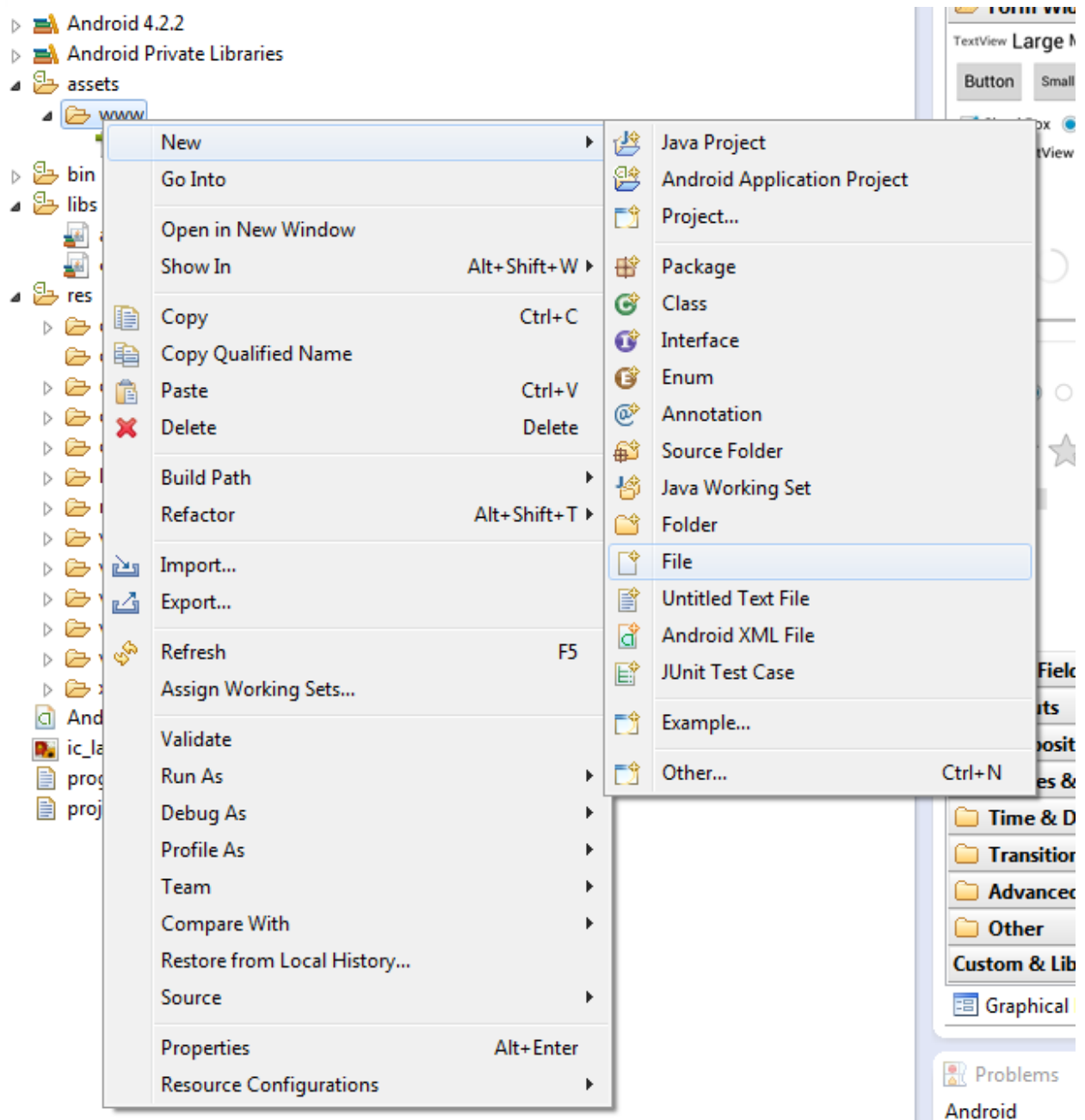
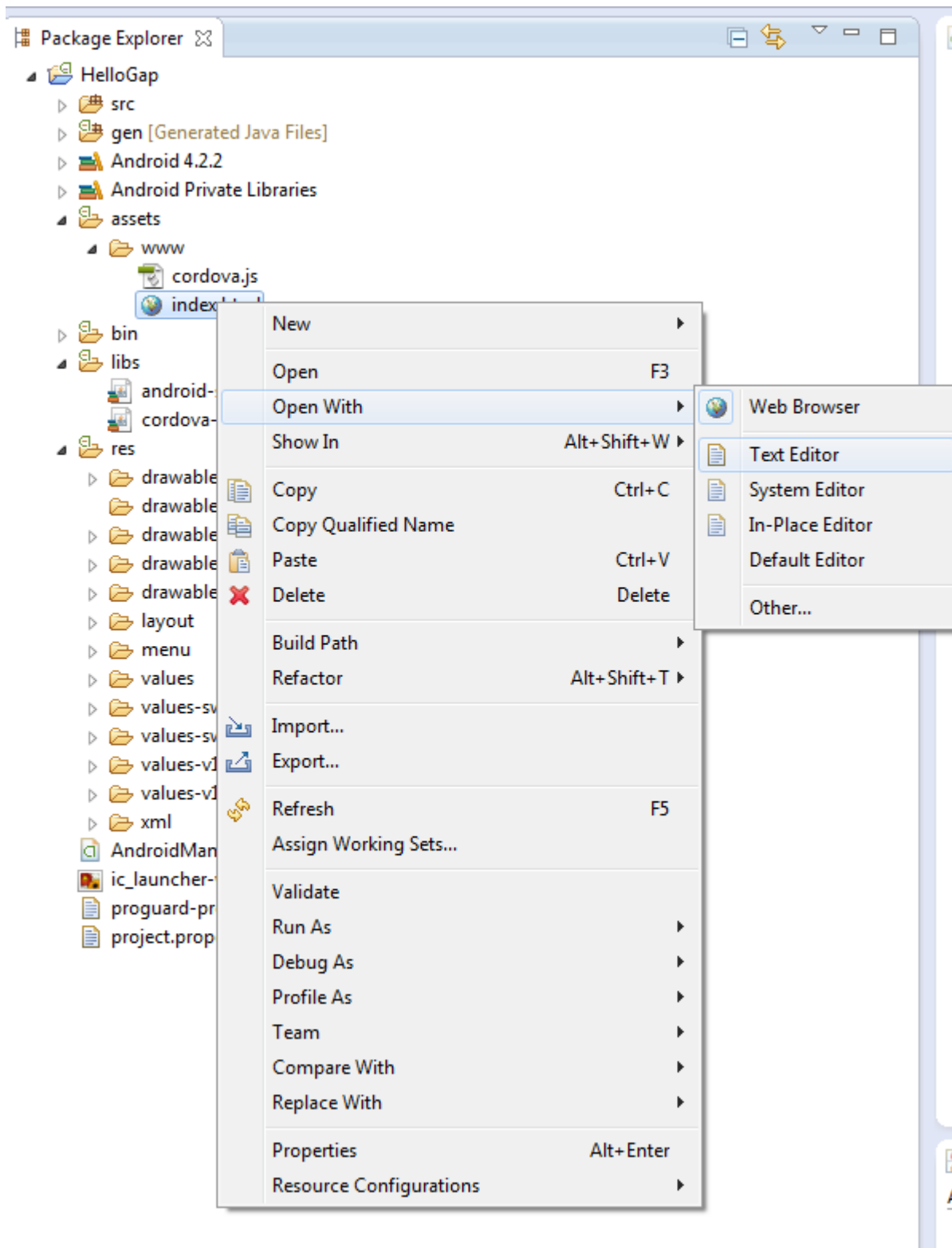


Figure 32 www new file

We create the folder with the above picture's way and then we open the file with eclipse text editor in order to paste and save the HTML code given later.



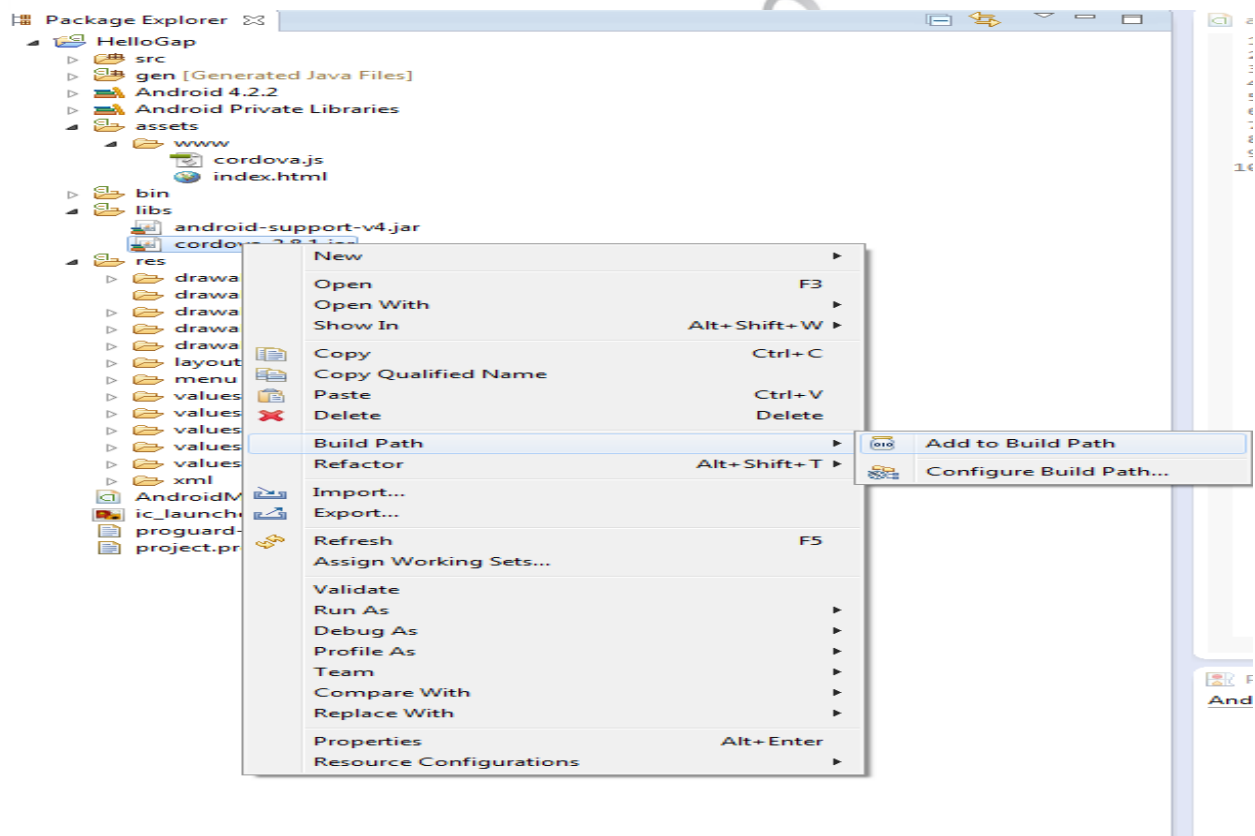
**Figure 33 Open text editor**

Then we add the following HTML code of the following table to act as a starting point for our user interface development:

```
<!DOCTYPE HTML>
<html>
<head>
<title>PhoneGap</title>
<script type="text/javascript" charset="utf-8" src="cordova.js"></script>
</head>
<body>
<h1>Hello Odytron team WELCOME TO PHONEGAP</h1>
</body>
</html>
```

**Table 15 index.html hello world**

Then we need to add the cordova-2.8.1.jar library to the build path for the Android project. Right-click cordova-2.8.1.jar and select Build Path > Add To Build Path



**Figure 34 Add to build Path**

### 5.3.4 Update the Activity Class

Now we are ready to update the Android project to start using PhoneGap. We open our main application Activity file. This file will have the same name as our project, followed by the word "Activity". It will be located under the src folder in the project package that we specified earlier in this process.

For our project, which we named HelloGap, the main Android Activity file is named HelloGapActivity.java, and is located in the package com.example.helloGap, which we specified in the New Android Project dialog box. In the main Activity class, we add an import statement

```
import org.apache.cordova.DroidGap;
```

**Table 16 import org.apache.cordova.DroidGap;**

We Change the base class from Activity to DroidGap ; this is in the class definition following the word extends :

```
public class HelloGapActivity extends DroidGap {
```

**Table 17 public class HelloGapActivity extends DroidGap {**

We Replace the call to setContentView() with a reference to load the PhoneGap interface from the localassets/www/index.html file, which we created earlier.

```
super.loadUrl("file:///android_asset/www/index.html");
```

**Table 18 super.loadUrl("file:///android\_asset/www/index.html");**

In PhoneGap projects, we can reference files located in the assets directory with a URL referencefile:///android\_asset, followed by the path name to the file. The file:///android\_asset URI maps to the assets directory. All the changes made are shown to the next image.

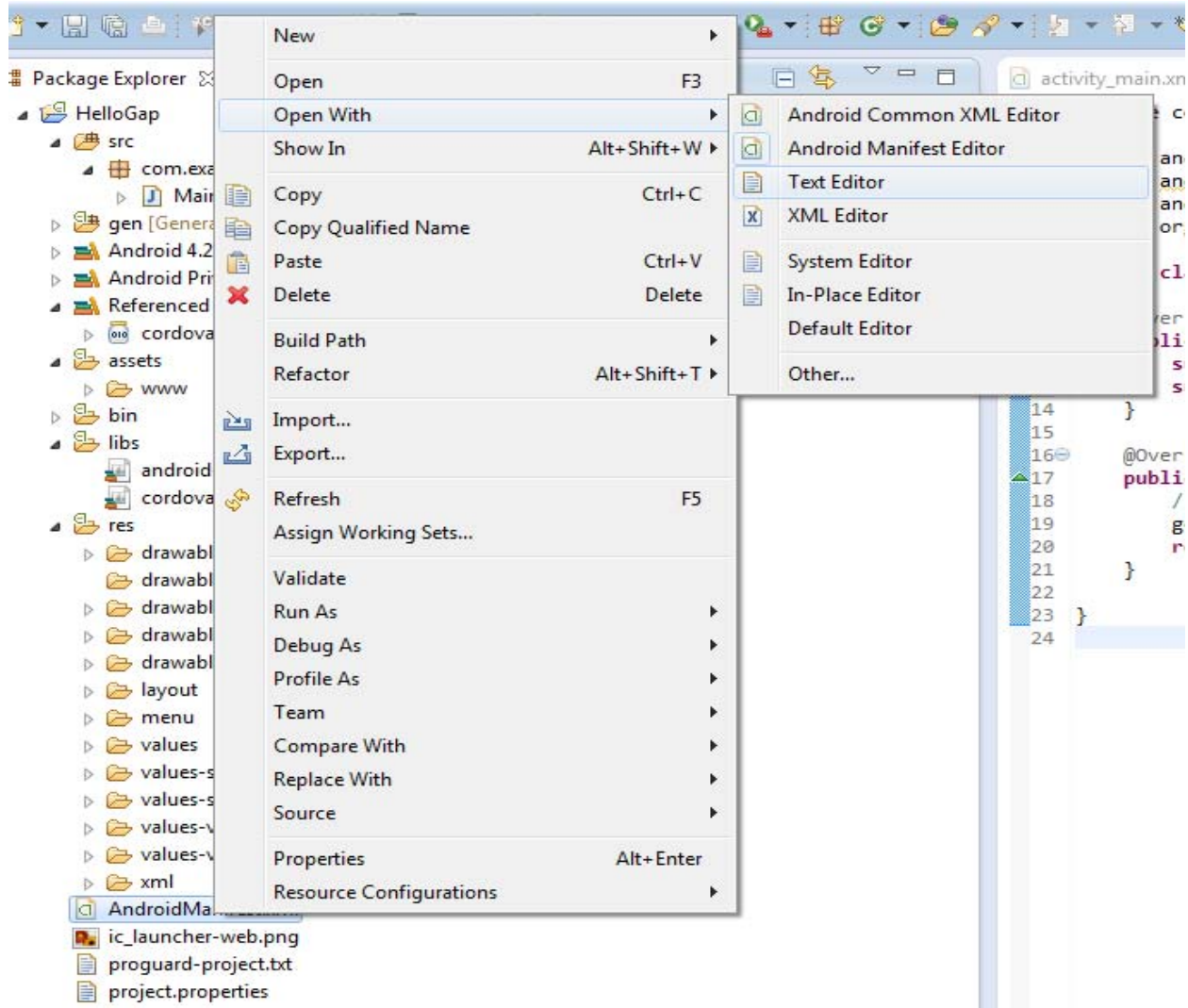
ΠΑΝΕΠ

```
1 package com.example.hellogap;
2
3 import android.os.Bundle;
4 import android.app.Activity;
5 import android.view.Menu;
6 import org.apache.cordova.DroidGap;
7
8 public class MainActivity extends DroidGap {
9
10     @Override
11     public void onCreate(Bundle savedInstanceState) {
12         super.onCreate(savedInstanceState);
13         super.loadUrl("file:///android_asset/www/index.html");
14     }
15
16     @Override
17     public boolean onCreateOptionsMenu(Menu menu) {
18         // Inflate the menu; this adds items to the action bar if it is present.
19         getMenuInflater().inflate(R.menu.main, menu);
20         return true;
21     }
22 }
23 }
24 }
```

Figure 35 Changes in HelloGap

### 5.3.5 Configure the Project Metadata

The last step is to configure the project metadata to enable PhoneGap to run. We Begin by opening the AndroidManifest.xml file in our project root. We Use the Eclipse text editor by right-clicking the AndroidManifest.xml file and selecting Open With > Text Editor.



**Figure 36 Open AndroidManifest**

In AndroidManifest.xml, we add the following supports-screen XML node as a child of the root manifest node:

```
<supports-screens
    android:largeScreens="true"
    android:normalScreens="true"
    android:smallScreens="true"
    android:resizeable="true"
    android:anyDensity="true"
/>
```

**Table 19 manifest.xml support screens**



The `supports-screen` XML node identifies the screen sizes that are supported by our application. We can change screen and form factor support by altering the contents of this entry. Next, we need to configure permissions for the PhoneGap application. We Copy the following `<uses-permission>` XML nodes and we paste them as children of the root `<manifest>` node in the `AndroidManifest.xml` file:

```
<uses-permission android:name="android.permission.CAMERA" />
<uses-permission android:name="android.permission.VIBRATE" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"
/>
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission
android:name="android.permission.ACCESS_LOCATION_EXTRA_COMMANDS" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.RECEIVE_SMS" />
<uses-permission android:name="android.permission.RECORD_AUDIO" />
<uses-permission android:name="android.permission.MODIFY_AUDIO_SETTINGS"
/>
<uses-permission android:name="android.permission.READ_CONTACTS" />
<uses-permission android:name="android.permission.WRITE_CONTACTS" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"
/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.GET_ACCOUNTS" />
<uses-permission android:name="android.permission.BROADCAST_STICKY" />
```

**Table 20** `manifest.xml` uses permission

The `<uses-permission>` XML values identify the features that we want to be enabled for our application. The lines above enable all permissions required for all features of PhoneGap to function. After we have built our application, we may want to remove any permission that we are not actually using; this will remove security warnings during application installation. After we have configured application permissions, we need to modify the existing `<activity>` node. We Locate the `<activity>` node, which is a child of the `<application>` XML node. We Add the following attribute to the `<activity>` node:

```
android:configChanges="orientation|keyboardHidden"
```

**Table 21** `manifest.xml` configChanges

Next, we need to create a second `<activity>` node for the `org.apache.cordova.DroidGap` class. We add the following `<activity>` node as a sibling of the existing `<activity>` XML node:

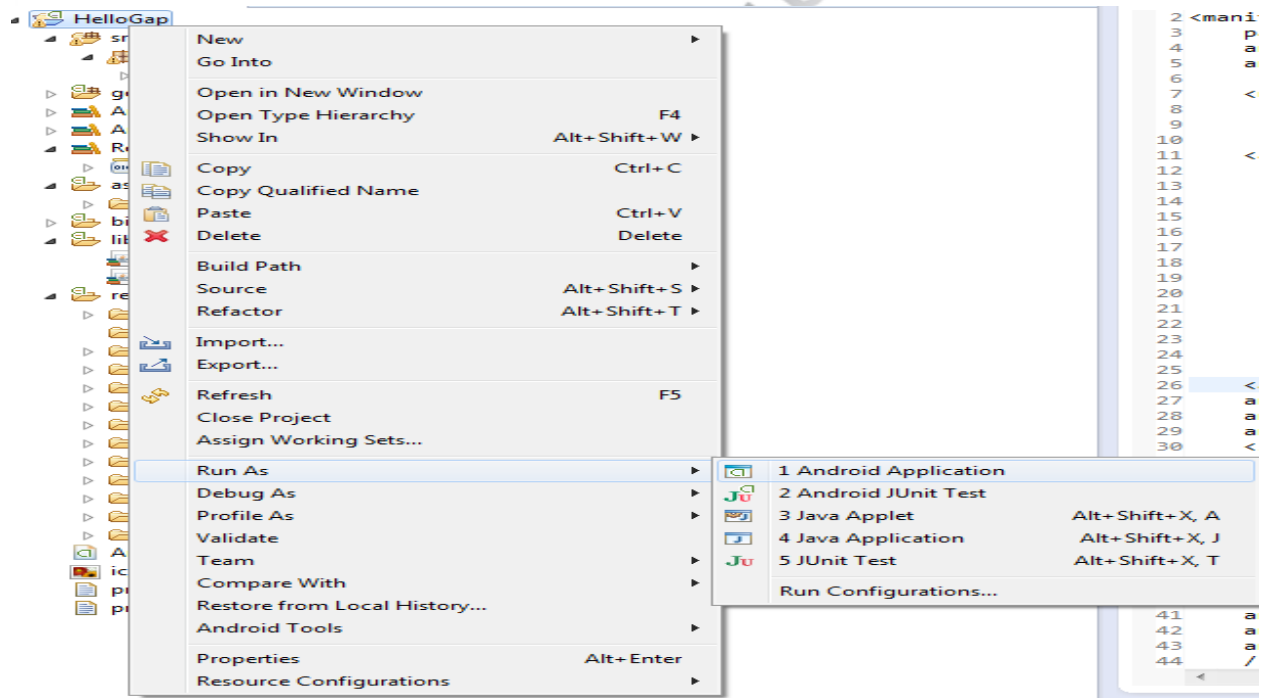
```
<activity
  android:name="org.apache.cordova.DroidGap"
  android:label="@string/app_name"
  android:configChanges="orientation|keyboardHidden">
  <intent-filter></intent-filter>
</activity>
```

**Table 22 manifest.xml activity**

At this point, our project is configured to run as a PhoneGap project for Android.

### 5.3.6 Running an Application

To launch our PhoneGap application in the Android emulator, we right-click the project root, and select Run As > Android Application



**Figure 37 Application Run**

Then the system prompts us to choose a device. The first choice is the physical device and the second is the virtual device via the emulator. We run first virtually and then we will run physically.

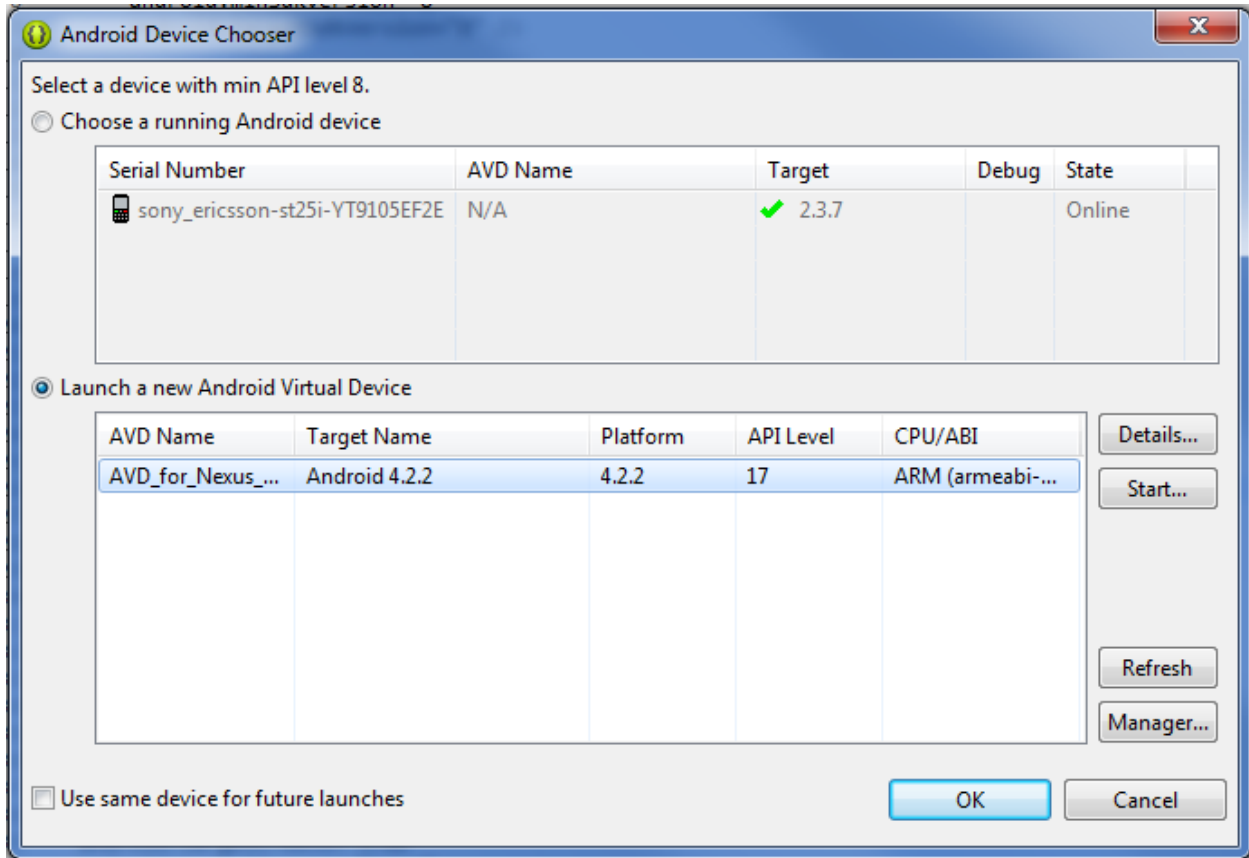


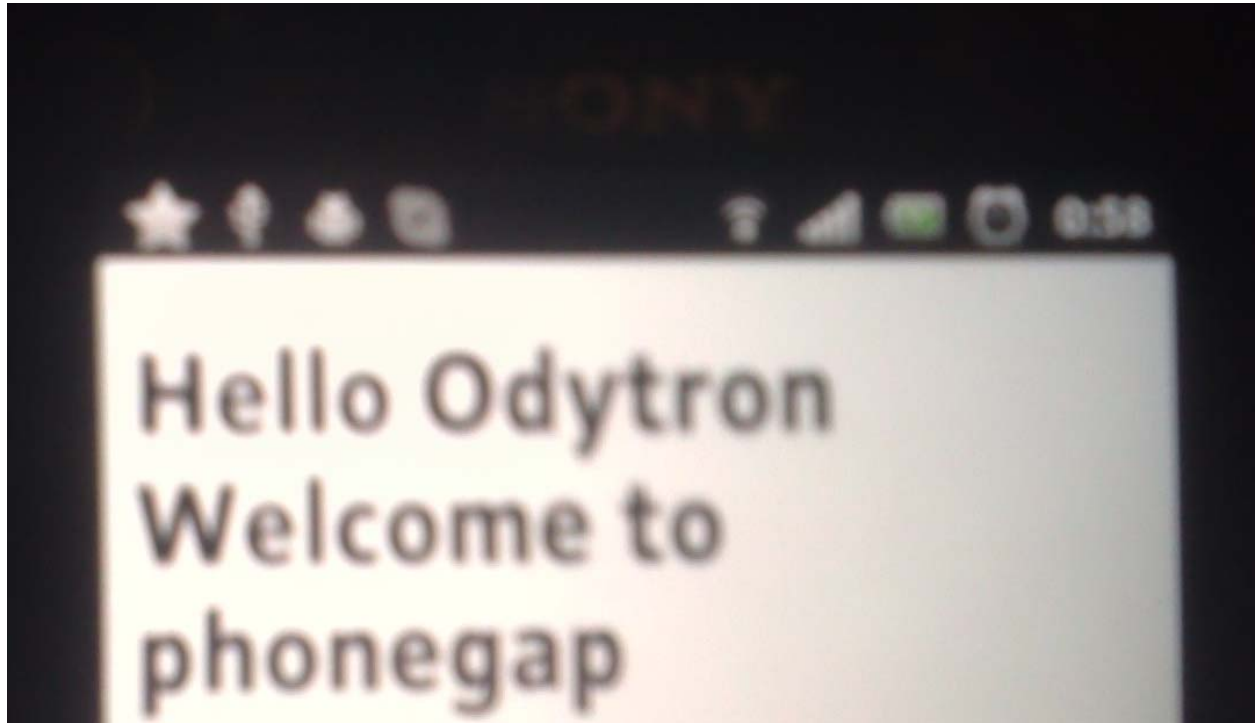
Figure 38 Run Virtual

Next image shows the virtual device result



Figure 39 Run Result

Next image shows the physical device result



**Figure 40 Physical Result**

Now its time to put more functionality to our first PhoneGap application. For this reason we are going to change the code in index.html and adding a CSS file into the www folder. The application will take a given name and then will give us back a greeting with our name. The code for the index.html and the master.css files are given in the annex of this project.

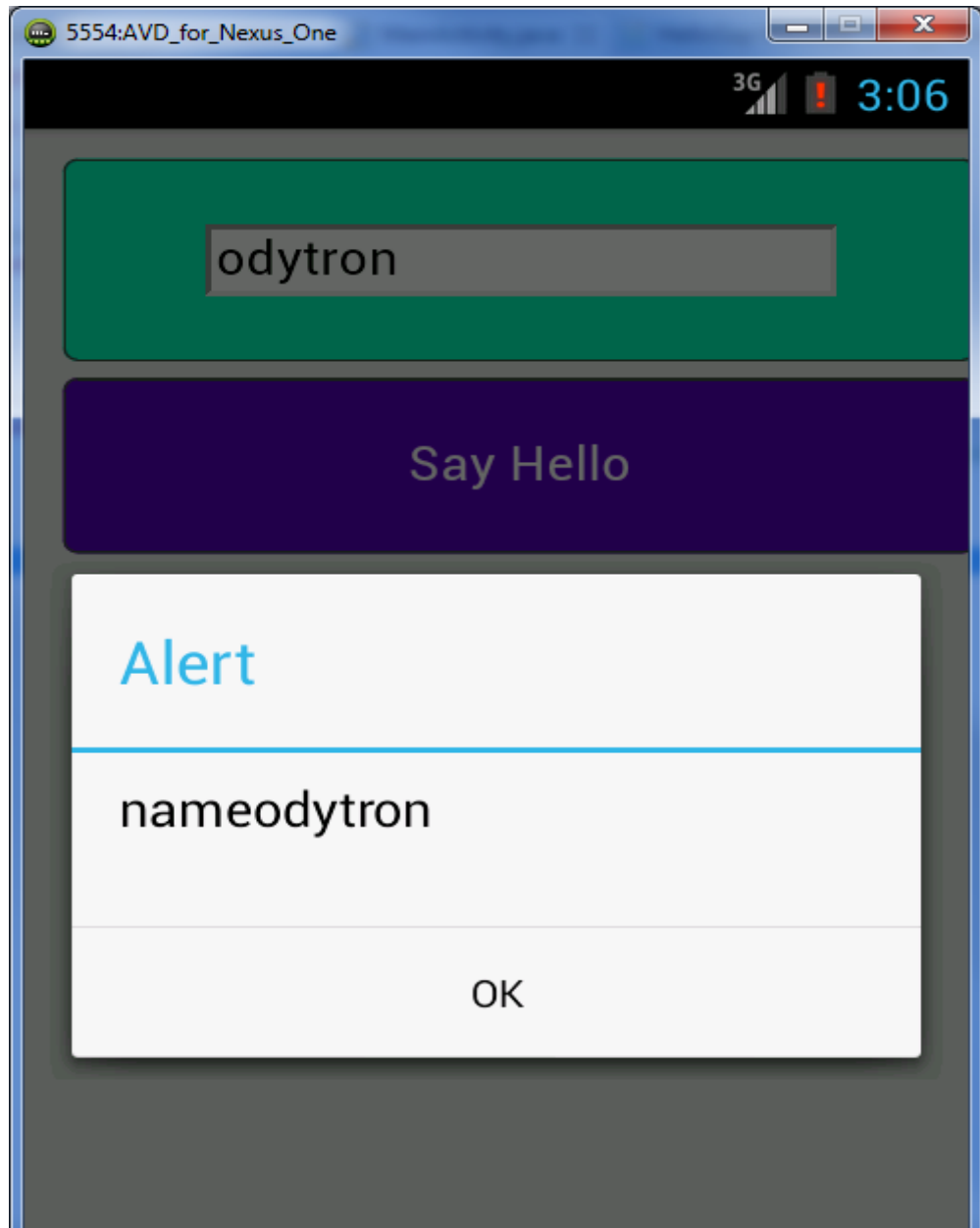


Figure 41 Cross Platform Demo Finished

#### 5.4 Conclusions about Chapter Five.

In chapter five we made a feasibility study for our project. Then we explain why we selected PhoneGap and we described the basic web technologies HTML, JavaScript and CSS. Finally we installed PhoneGap step by step with a simple “Hello World” application.

## 6. Application Development

### 6.1 Introduction

In this chapter we are going to describe the functionality of the application with use case diagram, and other project related diagrams such as SOA diagram and transition diagram. Apart from the diagrams we present the Dummy screens of the application. Then we are going to explain what is a geolocation service and what is a social media service. Finally we are going to describe the steps of developing according to the methodology we used, giving the critical path diagram.

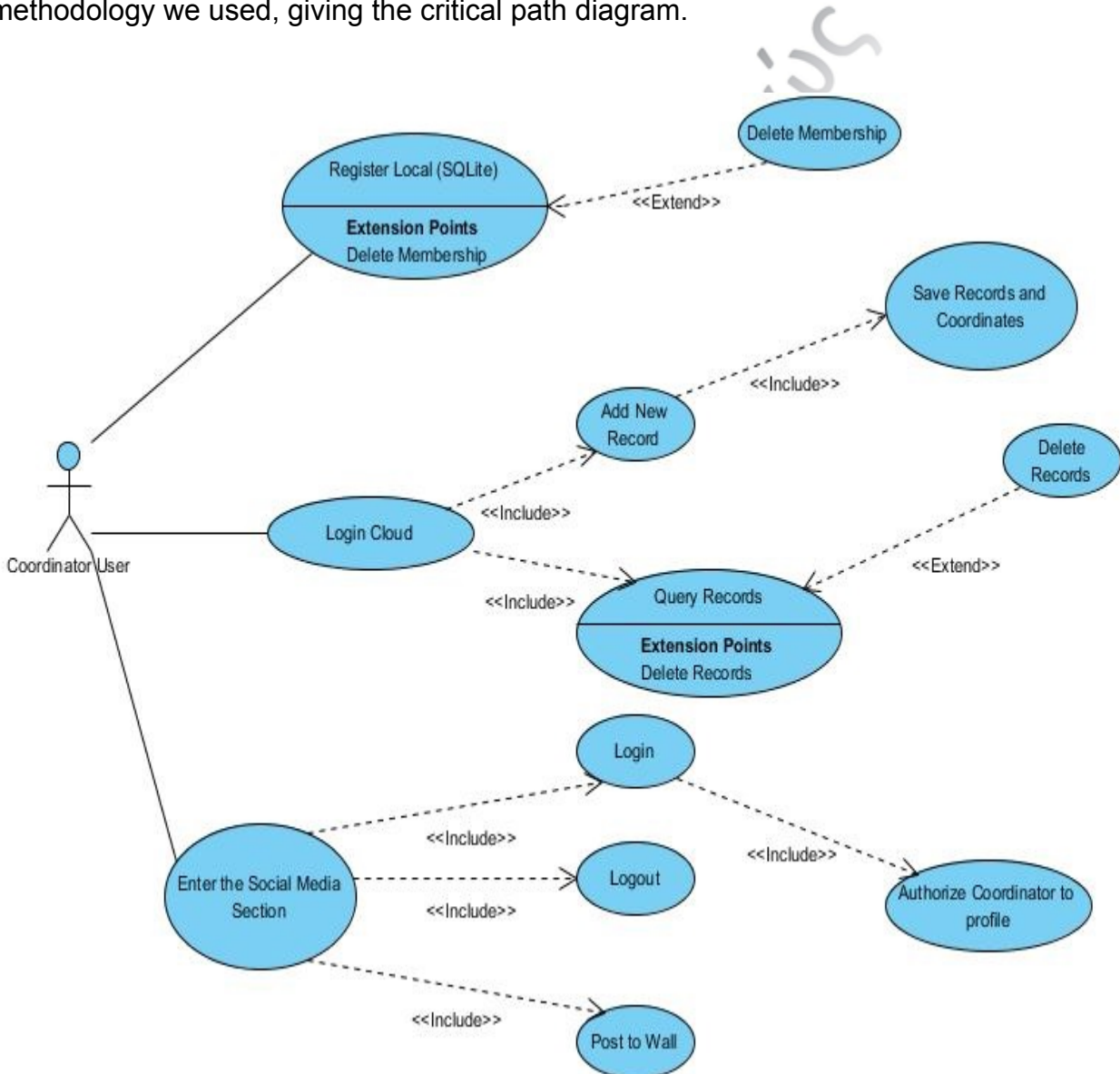


Figure 42 Application Use Case Diagram

### 6.1.1 Application Description

The major target of the project is to guide on how the web technology can help us to create one code for all mobile platforms. Apart from the major target mentioned above there are three minor purposes. The first purpose is to use a local database for user registration with SQLite which is a classic way to use databases with mobile devices. The second purpose is to register to a service with a cloud technology and show our geolocation coordinates via the mobile device GPS and other wireless network. The third purpose of the application is to enter a social media service. The use case diagram of the application is given in the figure above and includes twelve use cases and it's made with Visual paradigm for UML software which is a Proprietary with Free Community Edition software.

In the table below we give the names of the five use cases

| Use case number | Use case name                               |
|-----------------|---|
| 1               | REGISTER LOCAL SQLite <b>UC001</b>          |
| 2               | DELETE MEMBERSHIP <b>UC002</b>              |
| 3               | LOGIN CLOUD <b>UC003</b>                    |
| 4               | ADD NEW RECORD <b>UC004</b>                 |
| 5               | SAVE RECORDS AND COORDINATES <b>UC005</b>   |
| 6               | QUERY RECORDS <b>UC006</b>                  |
| 7               | DELETE RECORDS <b>UC007</b>                 |
| 8               | ENTER THE SOCIAL MEDIA SECTION <b>UC008</b> |
| 9               | LOGIN <b>UC009</b>                          |



|    |   |
|----|---|
| 10 | AUTHORIZE APPLICATION TO PROFILE <b>UC010</b> |
| 11 | LOGOUT <b>UC011</b>                           |
| 12 | POST TO WALL <b>UC012</b>                     |

**Table 23 use case diagram codes**

### 6.1.2 Web Diagrams

The three click rule of web applications according to Jeffrey Zeldman is "based on the way people use the Web" and "the rule can help you create sites with intuitive, logical hierarchical structures" [20]. The three click rule was a practice we used during our postgraduate studies and we also applied in our application. The web technology allows us to use web based diagrams , such as the transition diagram , the web –web service – database diagram and the Service architecture oriented (SOA) diagram. The web –web service – database shows an architecture which includes the web page the related web service and the relation with the database. Finally the Service architecture oriented (SOA) diagram includes five layers :

1. UI layer.
2. Business Process layer.
3. WS layer.
4. Basic service layer.
5. Database Layer.

The next diagram is the transition diagram which shows the control flow between pages of our application

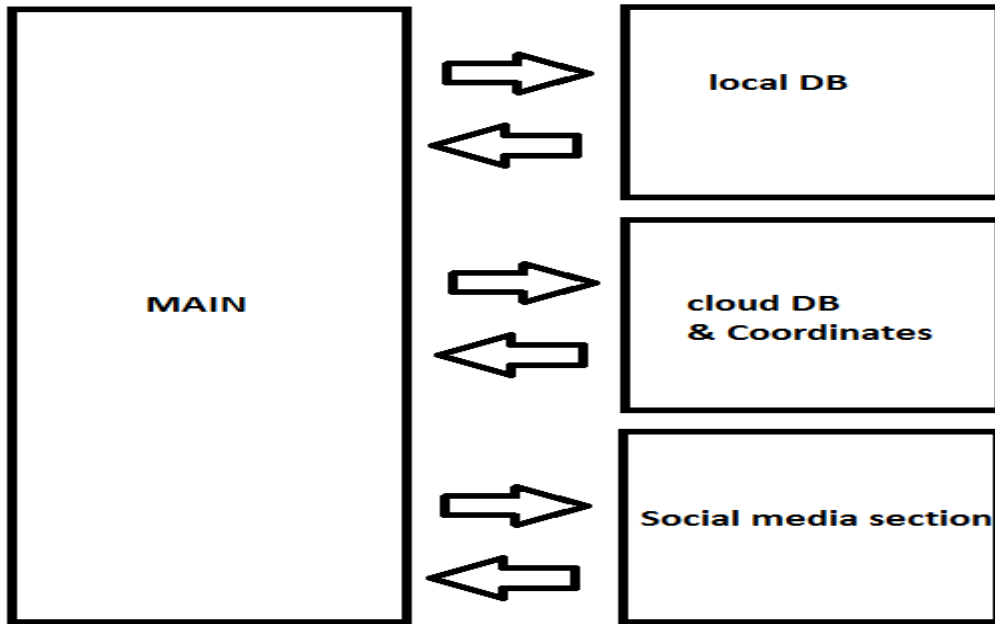


Figure 43 Transition Diagram

The next diagram is the WEB- WEB- SERVICE- DATABASE LAYER which shows the relationship between these layers.

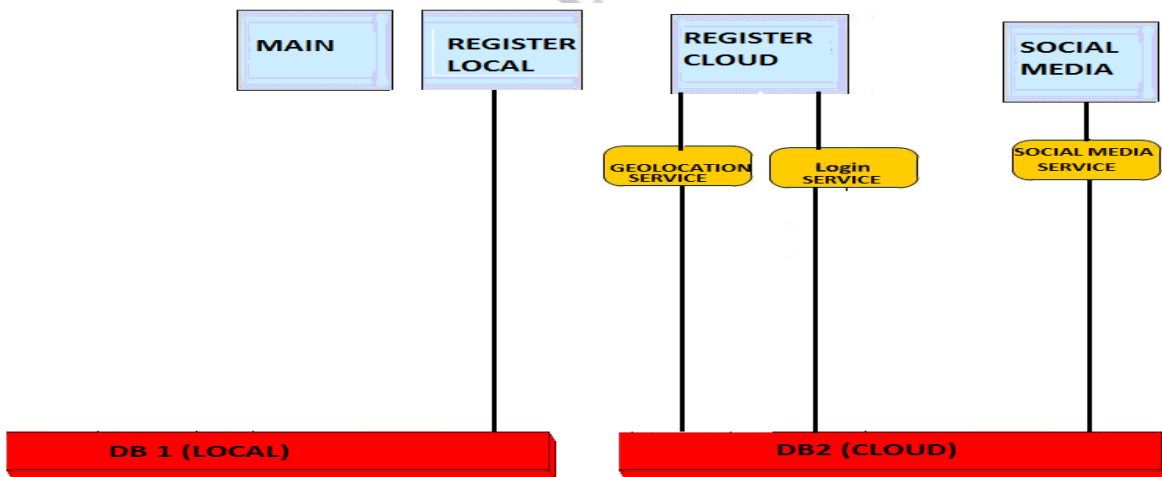


Figure 44 WEB-WS-DB Diagram

The next diagram is the Service Oriented Architecture (SOA) diagram which shows the relationship between processes and the services.

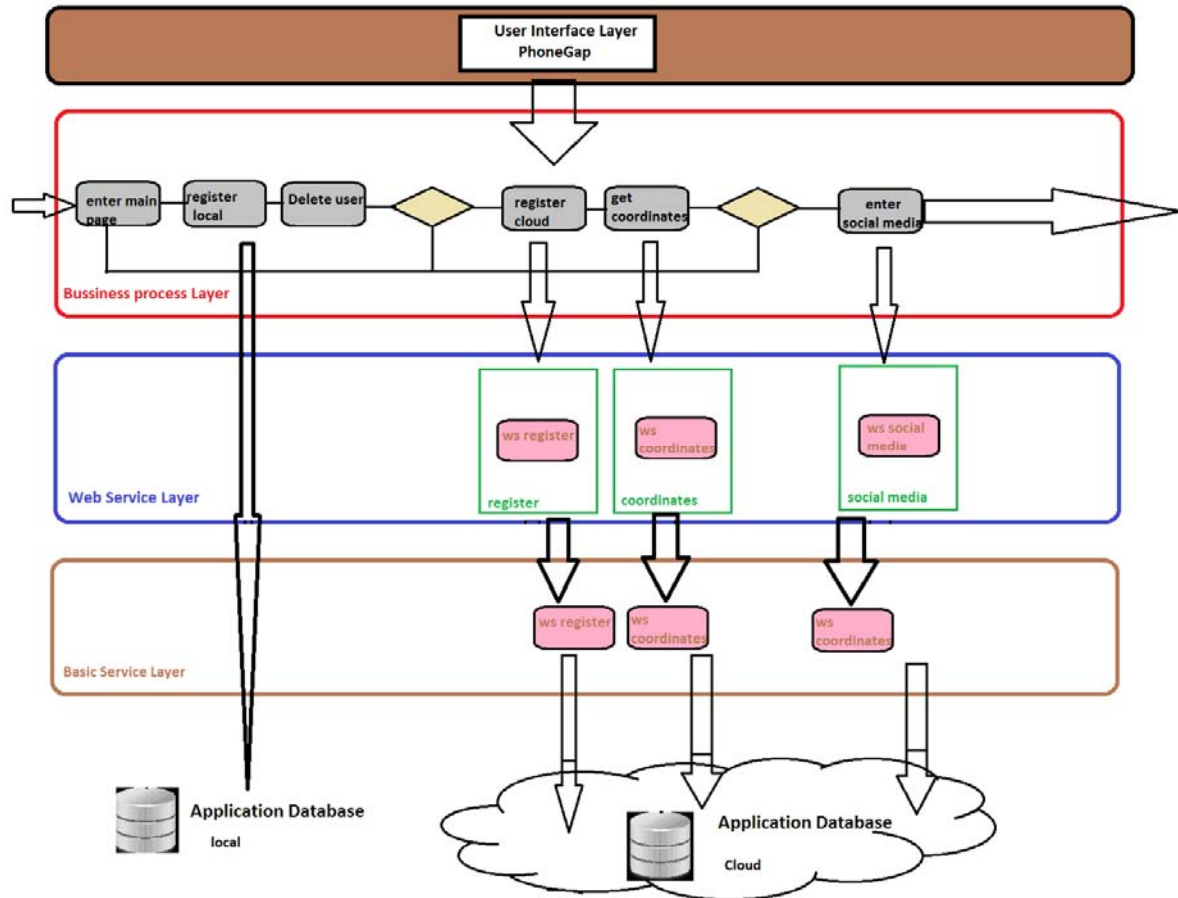


Figure 45 SOA Diagram

### 6.1.3 Dummy Screens

Next we illustrate the dummy screens of our application

#### Dummy Screens

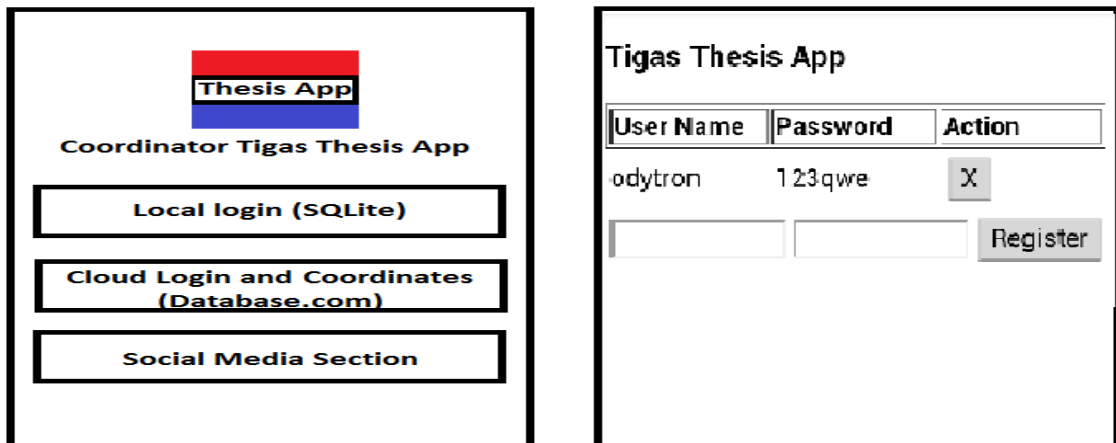


Figure 46 Dummy screens I

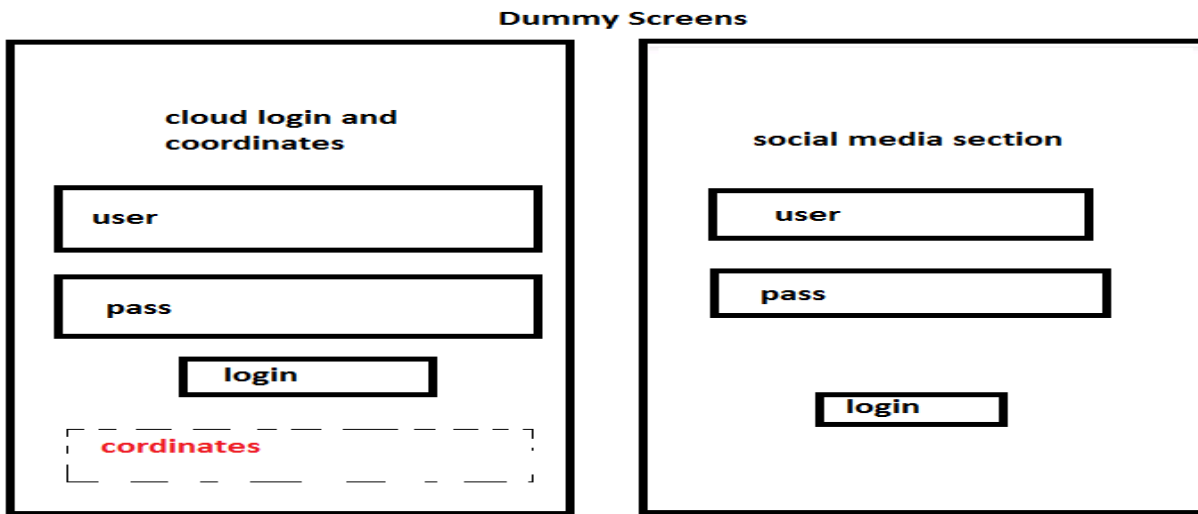


Figure 47 Dummy screens II

## 6.2 Social Media & Geolocation Services

### 6.2.1 Social media Services

There are many definitions to refer to a social media service. According to Andreas Kaplan and Michael Haenlein define social media as "a group of Internet-based applications that build on the ideological and technological foundations of Web 2.0, and that allow the creation and exchange of user-generated content [21]. Furthermore, social media depends on mobile and web-based technologies to create highly interactive platforms through which individuals and communities share, co-create, discuss, and modify user-generated content. It introduces substantial and pervasive changes to communication between organizations, communities, and individuals [22]. Social media differentiates from traditional/industrial media in many aspects such as quality [23], reach, frequency, usability, immediacy, and permanence. There are many effects that stem from internet usage. For content contributors, the benefits of participating in social media have gone beyond simply social sharing to building reputation and bringing in career opportunities and monetary income, as discussed in Tang, Gu, and Whinston (2012) [24].

### 6.2.2 Geolocation Services

Geolocation services or Location-based services (LBS) are a general class of computer program-level services used to include specific controls for location and time data as control features in computer programs. As such LBS is an information service and has a number of uses in social networking today as an entertainment service, which is accessible with mobile devices through the mobile network and which uses information on the geographical position of the mobile device. This has become more and more important with the expansion of the smartphone and tablet markets as well.[25].

LBS are used in a variety of contexts, such as health, indoor object search,[26] entertainment,[27] work, personal life, etc. LBS include services to identify a location of a person or object, such as discovering the nearest banking cash machine (a.k.a. ATM) or the whereabouts of a friend or employee. LBS include parcel tracking and vehicle tracking services. LBS can include mobile commerce when taking the form of coupons or advertising directed at customers based on their current location. They include personalized weather services and even location-based games. They are an example of telecommunication convergence.

### **6.3 Application Functionality**

The application has three different functions to demonstrate. The first function has to do with the use of SQLite for local registration in order to save a user in the local storage of the mobile phone and proceed with other functions of a simple CRUD system. The second function has to do with the cloud registration and the user's ability to save the location coordinates in to it. The third and the last function has to do with the social media integration of the application which is a good way to make an application famous and earn the maximum gain from such an effort.

#### **6.3.1 The Local Login (SQLite)**

In this subparagraph we are going to discuss the first functionality of our application in brief. SQLite is a relational database management system contained in a small (~350 KB) C programming library. In contrast to other database management systems, SQLite is not a separate process that is accessed from the client application, but an integral part of it. SQLite is ACID-compliant and implements most of the SQL standard, using a dynamically and weakly typed SQL syntax that does not guarantee the domain integrity. SQLite is a popular choice as embedded database for local/client storage in application software such as web browsers. It is arguably the most widely deployed database engine, as it is used today by several widespread browsers, operating systems, and embedded systems, among others. SQLite has many bindings to programming languages. The source code for SQLite is in the public domain.<sup>76</sup> In general we can say that SQLite has some limitations. It is slow, it locks the whole file for writing, the database is restricted to 2 GB in most cases and it's not very scalable.

#### **6.3.2 Cloud Registration and Coordinates (Database.com)**

Database.com is the cloud database offering from salesforce.com. Database.com enables hosted relational database capabilities that we easily can tap into within our applications, without having to manage the infrastructure on our own. We can use the Web-based administrative interface to create objects and relationships, and use the REST or SOAP APIs to access our data. We can learn more about the capabilities of Database.com through the Developer Center. Our target is to consume data from the cloud and especially from Database.com. The concept of this function is to collect contacts from various locations.

It can be used to gather contact information, as well as notes, and it can record the user's GPS coordinates to identify where the information was captured. It also allows us to go back and edit data that we previously captured. The advantage is the cross platform fashion which allows us to use any compatible smartphone. In this subparagraph we explain how we implemented the functionality of this page including all the source code in the annex section. For data storage, this application uses a single custom table on Database.com. Although this is a basic one-table example, Database.com can also support large, multi-relationship data structures. The step by step tutorial on how to interconnect the files of this functionality can found in Andrew Trice article of adobe's web site<sup>77</sup>. This functionality starts with a login page. Then the childbrowser redirects us to the salesforce authentication page to insert our credentials. Next we have to options, the first is to add a new record and the second is to query our records. When we add a new record the GPS is used to record our coordinates also. Lastly our personal information and out coordinates are saved In the cloud. This function can found practical use in many sectors such us sales, education , etc. The next images describe how to setup our database for the above purpose.

**Get Started Developing on the Database.com Platform**

Signing up for your free Database.com account is easy. Just fill out the form below, and you'll be up and running in a minutes.

**About you**

First name:\*

Last name:\*

Email Address:\*

Primary Job Role:\*

**About your company**

Country:\*

Postal Code:\*

Company:\*

Number of employees\*

**Database.com Username**

Your username must be in the form of an email address, which can be actual or hypothetical. For example, you can use something like mike@db.data. All email correspondence related to this account will be sent to the address provided in the "Email Address" field above.

Username:\*  Username must be in the form of an email

**For your security**

I have read and agreed to the [Master Subscription Agreement](#) and the [Supplemental Terms for Database.com](#)

Figure 48 Database.com sign Up

log in, and click Create A New Object on the System Overview screen.

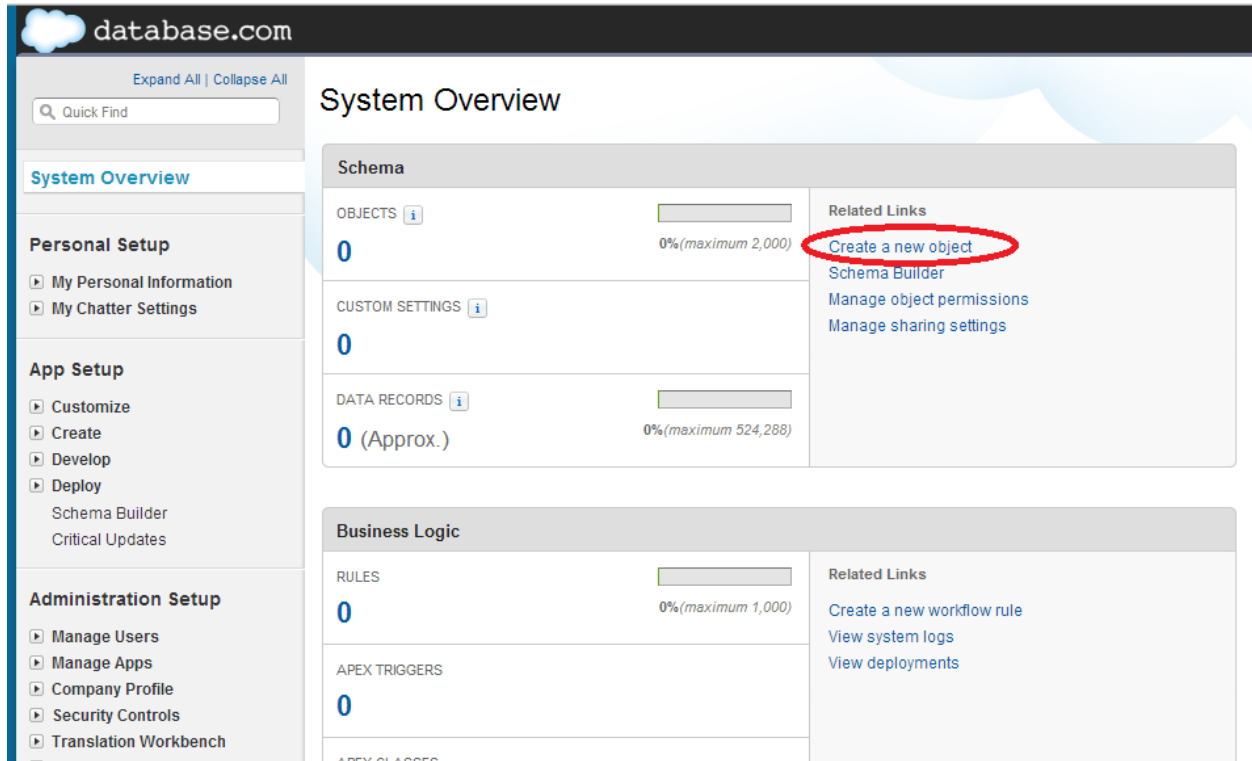


Figure 49 create new object

This shows the *New Custom Object* dialog. Here's where we enter the label/name for our objects. I created a Lead object to contain the leads captured by the PhoneGap application.

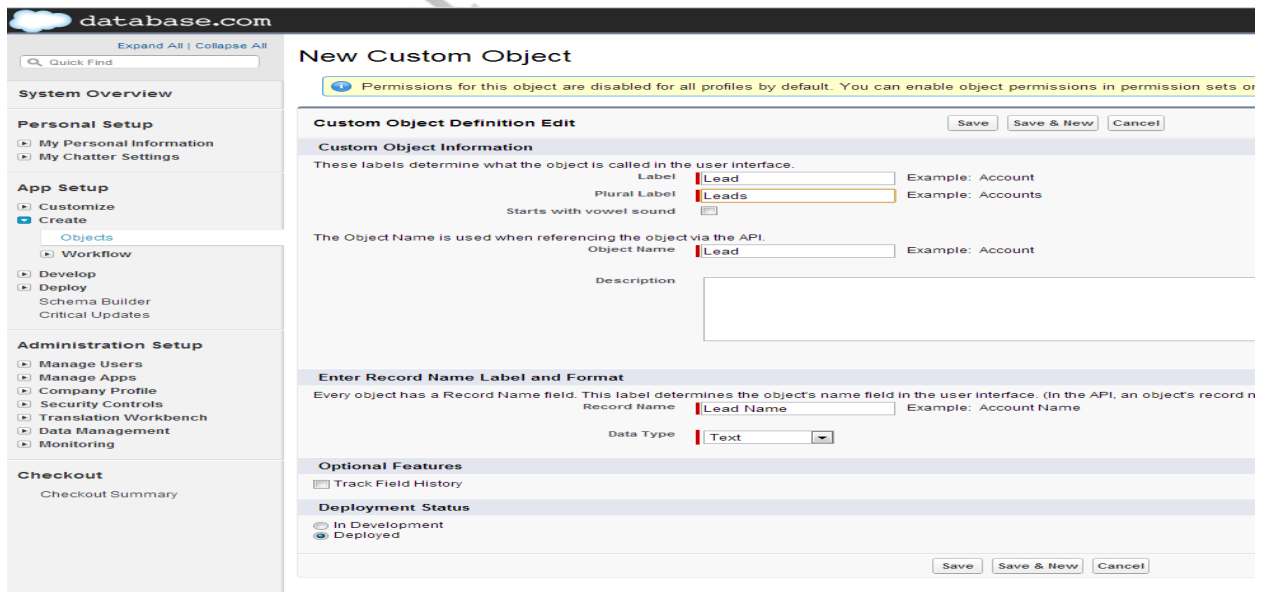


Figure 50 New Custom object



The next step is to add custom fields to the Lead object. Under the *Custom Fields & Relationships* section, click **New** to add data fields. We added fields for “First Name,” “Last Name,” “Latitude,” “Longitude,” “Notes,” “Email,” and “Telephone.” Then we followed the steps in the online wizard and it guides us through this process.

Created By [Odysseas Tigas](#), 6/24/2013 4:27 AM

**Standard Fields**

| Action               | Field Label                      | Field Name     | Data Type          |
|----------------------|----------------------------------|----------------|--------------------|
|                      | <a href="#">Created By</a>       | CreatedBy      | Lookup(User)       |
|                      | <a href="#">Last Modified By</a> | LastModifiedBy | Lookup(User)       |
| <a href="#">Edit</a> | <a href="#">Lead Name</a>        | Name           | Text(80)           |
| <a href="#">Edit</a> | <a href="#">Owner</a>            | Owner          | Lookup(User,Queue) |

**Custom Fields & Relationships** New [Field Dependencies](#)

No custom fields defined

**Related Lookup Filters**

No related lookup filters defined.

**Validation Rules** New

No validation rules defined.

---

**Custom Fields & Relationships** New [Field Dependencies](#)

| Action                                     | Field Label               | API Name     | Data Type      | Controlling Field | Modified By  |
|--|---------------------------|--------------|----------------|-------------------|--|
| <a href="#">Edit</a>   <a href="#">Del</a> | <a href="#">Email</a>     | Email__c     | Email          |                   | <a href="#">Odysseas Tigas</a> , 6/24/2013 4:32 AM |
| <a href="#">Edit</a>   <a href="#">Del</a> | <a href="#">First</a>     | First__c     | Text(100)      |                   | <a href="#">Odysseas Tigas</a> , 6/24/2013 4:33 AM |
| <a href="#">Edit</a>   <a href="#">Del</a> | <a href="#">Last</a>      | Last__c      | Text(100)      |                   | <a href="#">Odysseas Tigas</a> , 6/24/2013 4:34 AM |
| <a href="#">Edit</a>   <a href="#">Del</a> | <a href="#">Latitude</a>  | Latitude__c  | Number(3, 10)  |                   | <a href="#">Odysseas Tigas</a> , 6/24/2013 4:35 AM |
| <a href="#">Edit</a>   <a href="#">Del</a> | <a href="#">Longitude</a> | Longitude__c | Number(3, 10)  |                   | <a href="#">Odysseas Tigas</a> , 6/24/2013 4:36 AM |
| <a href="#">Edit</a>   <a href="#">Del</a> | <a href="#">Notes</a>     | Notes__c     | Text Area(255) |                   | <a href="#">Odysseas Tigas</a> , 6/24/2013 4:40 AM |

**Figure 51 New relationships**

At this point, we’ve created the data objects used to persist data captured within the application. However, there is one more step before we can use these objects within a PhoneGap application. To access the data remotely, we must create a remote access “application.” The configuration of this application includes a unique key used to correctly authenticate users and identify our data objects. To create a new remote access “Application,” expand the *Develop* category and select *New*.” The *Remote Access Edit* form displays, where we specify an application name, contact email, and callback URL configuration for our application. The application name is used as descriptive text when logging into our application, and the callback URL is used to redirect the user’s browser once they have authenticated successfully.

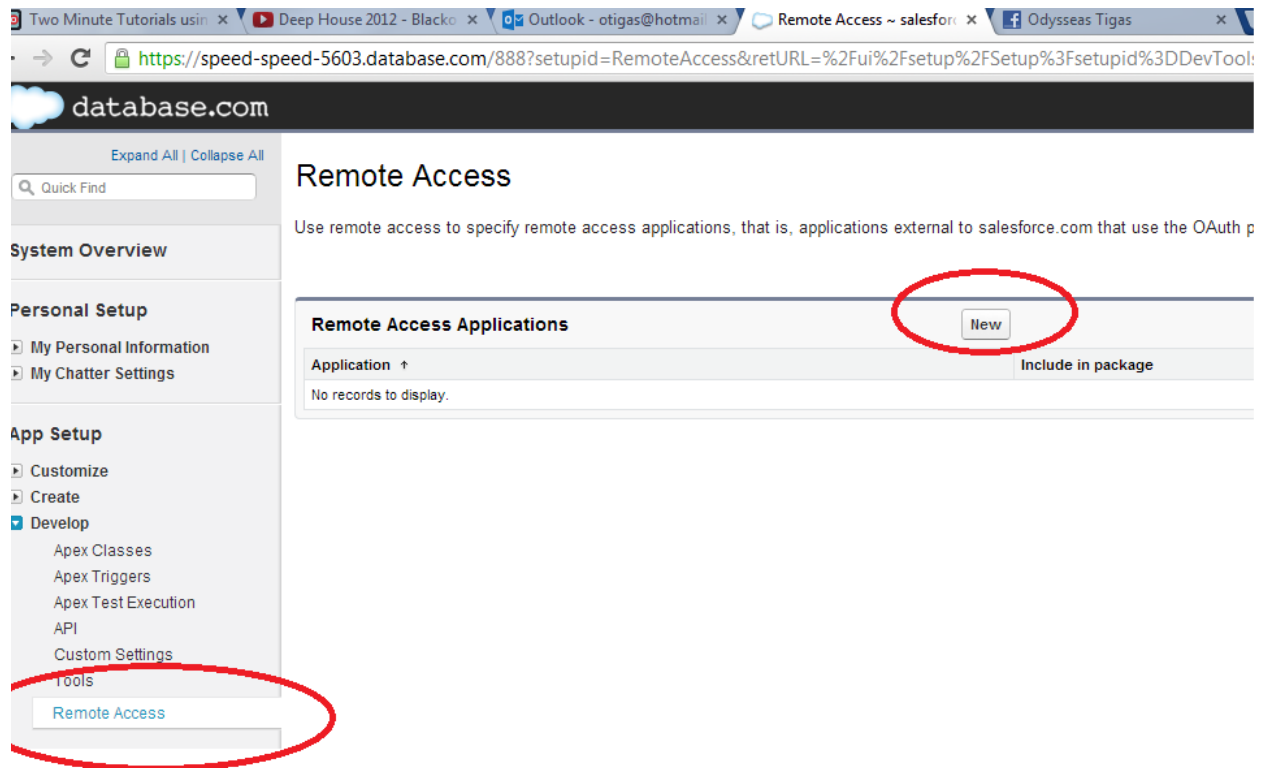


Figure 52 Remote Access step 1

### Remote Access

Specify the remote access application to be used with salesforce.com. Salesforce.com generates the Consumer Key and Consumer Secret on successful save.

A screenshot of the "Remote Access Edit" form in Salesforce. The form is divided into several sections: "Basic Information" with fields for Application (PhoneGapAccess), Description (Access to phonegap application), Logo Image URL, Info URL, Contact Phone, and Contact Email (otigas@hotmail.com); "Integration" with a Callback URL (sfdc://success); "Policies" with a checked box for "No user approval required for users in this organization"; and "Authentication" with a checked box for "Use digital signatures". "Save" and "Cancel" buttons are visible at the bottom.

Figure 53 Remote Access step 2

Once we save this information, we are provided with a “consumer key,” which we need later when logging in and accessing data services from Database.com. It is important to make a note of the consumer key. We need this later in our JavaScript configuration.

## Remote Access

[← Back to List: Remote Access](#)

### Remote Access Detail

[Edit](#) [Delete](#)

| Basic Information  |  |
|--|--|
| Application  | PhoneGapAccess   |
| Description  | Access to phonegap application   |
| Logo Image URL   |  |
| Info URL   |  |
| Contact Phone  |  |
| Contact Email  | <a href="mailto:otiqas@hotmail.com">otiqas@hotmail.com</a>                           |
| Integration  |  |
| Callback URL   | sfdc://success   |
| Policies   |  |
| No user approval required for users in this organization | <input checked="" type="checkbox"/> <a href="#">i</a>                                |
| Authentication   |  |
| Use digital signatures                                   |  |
| Consumer Key   | 3MVG99qusVZJwhsmjZIIeUsFRnadOib8Kv_MPwooFMEi.XpChrZ5cVEcKU_7NR1zfQjjmdHI7wMARXnLlgku |
| Consumer Secret  | <a href="#">Click to reveal</a>  |
| Created Date   | 6/24/2013 4:55 AM  |

**Figure 54 Remote Access Detail**

|  |
|--|
| Consumer key   |
| 3MVG99qusVZJwhsmjZIIeUsFRnadOib8Kv_MPwooFMEi.XpChrZ5cVEcKU_7NR1zfQjjmdHI7wMARXnLlgku |
| Consumer secret  |
| 5170453360847162969  |

**Table 24 Consumer key**

The application will communicate with Database.com's REST API using the forcetk.js JavaScript wrapper. Forcetk.js (Force.com JavaScript REST Toolkit) is an open source wrapper to the REST API that simplifies the consumption of data from Force.com/Database.com inside of JavaScript-based applications. Forcetk.js provides the hooks for OAuth2 authentication, and helper methods that make retrieving and updating data incredibly easy. When building applications in PhoneGap, will be able to access the force.com REST API directly without need for a proxy (like a standard browser-based application).

When leveraging PhoneGap, we create our applications entirely in HTML, CSS, and JavaScript. We can develop these applications in any text editor. IDE's like eclipse , visual studio and xcode allow us to deploy PhoneGap applications directly onto a device using a USB connection. However, we can also use PhoneGap Build, a cloud-based PhoneGap compiler, which allows us to simply upload our HTML, CSS, and JavaScript code, and it will generate platform-specific binaries for us. Because the application interface is being built using HTML, CSS, and JavaScript, it is possible to leverage existing tools and frameworks to improve developer productivity. This example will leverage the following tools to speed up the developer process:

**Zepto.js** – A development accelerator library that provides utility and shortcut functions for creating interactive and dynamic JavaScript experiences. Zepto.js has a jQuery compatible syntax, with a mobile-centric optimizations.

**Twitter Bootstrap** – A UI styling library that provides CSS styles and HTML/JavaScript components to make our application feel more like an "app" instead of "just a web page".

**Mustache.js** – An easy-to-use templating library that allows us to create HTML "templates" for use within our dynamic JavaScript applications. Templating enables us to easily separate our HTML UI layer from the application logic written in JavaScript. The first thing to do within the application is create our root HTML file that will be the entry point into the application. All PhoneGap applications start with an "index.html" file, which is the "root" of the application. Within the index.html file, We included all of the appropriate libraries, with a blank HTML <body>. The HTML <body> is blank because the entire user interface will be created dynamically by JavaScript.

```
<html>
  <head>
    <link rel="stylesheet" href="assets/css/bootstrap.css" type="text/css"
  />
    <link rel="stylesheet" href="assets/css/styles.css" type="text/css" />
    <script type="text/javascript" src="js/libs/zepto.js"></script>
    <script type="text/javascript" src="js/libs/forcetk.js"></script>
    <script type="text/javascript" src="cordova-1.7.0.js"></script>
    <script type="text/javascript"
src="js/libs/ChildBrowser.js"></script>
    <script type="text/javascript" src="js/libs/mustache.js"></script>
    <script type="text/javascript"
src="js/salesforceWrapper.js"></script>
    <script type="text/javascript" src="js/application.js"></script>
  </head>
  <body></body>
</html>
```

**Table 25 Scripts in index.html**

When a PhoneGap application is initialized, a "deviceready" event is dispatched. This event indicates that the contents of the application have been sufficiently loaded, and all PhoneGap APIs have been initialized.

```
document.addEventListener( "deviceready", onDeviceReady );

var sfw;

function onDeviceReady( event ) {
    console.log( "deviceready" );

    //initialize salesforce wrapper
    sfw = new SalesforceWrapper();
}
```

**Table 26 Deviceready ondevice readyfunctions**

The first thing that the application does is initialize a JavaScript class that we created called "SalesforceWrapper". The SalesforceWrapper class wraps the forcetk.js library to streamline authentication, and includes all of the configuration information needed to access the force.com REST API. In this class, we will need to set the clientID value with the consumer key that we obtained through the "Remote Access" configuration that was discussed earlier.

```
function SalesforceWrapper() {
    /* AUTHENTICATION PARAMETERS */
    this.loginUrl = 'https://login.salesforce.com/';
    this.clientId = 'YOUR_KEY_GOES_HERE';
    this.redirectUri =
'https://login.salesforce.com/services/oauth2/success';

    /* CLASS VARIABLES */
    this.cb = undefined; //ChildBrowser in PhoneGap
    this.client = undefined; //forceTk client instance

    this.init();
}

SalesforceWrapper.prototype.init = function() {
    this.client = new forcetk.Client(this.clientId, this.loginUrl);
    this.cb = window.plugins.childBrowser;
}

SalesforceWrapper.prototype.login = function (successCallback) {
    this.loginSuccess = successCallback;
    var self = this;
```

```

self.cb.onLocationChange = function (loc) {
    if (loc.search(self.redirectUri) >= 0) {
        self.cb.close();
        self.sessionCallback(unescape(loc));
    }
};

self.cb.showWebPage(self.getAuthorizeUrl(self.loginUrl, self.clientId,
self.redirectUri));
}

SalesforceWrapper.prototype.getAuthorizeUrl = function (loginUrl,
clientId, redirectUri) {
    return loginUrl + 'services/oauth2/authorize?display=touch' +
    '&response_type=token&client_id=' + escape(clientId) + '&redirect_uri=' +
    escape(redirectUri);
}

SalesforceWrapper.prototype.sessionCallback = function(loc) {
    var
    oauthResponse = {};

    var fragment = loc.split("#")[1];

    if (fragment) {
        var nvps = fragment.split('&');
        for (var nvp in nvps) {
            var parts = nvps[nvp].split('=');
            oauthResponse[parts[0]] = unescape(parts[1]);
        }
    }

    if (typeof oauthResponse === 'undefined' || typeof
    oauthResponse['access_token'] === 'undefined') {
        console.log("error");
    } else {
        this.client.setSessionToken(oauthResponse.access_token, null,
        oauthResponse.instance_url);
        if ( this.loginSuccess ) {
            this.loginSuccess();
        }
    }
    this.loginSuccess = undefined;
}

```

**Table 27 SalesforceWrapper function.**

The SalesforceWrapper class simplifies the example from forcetk.js, and enables us to authenticate with a single line of code in our application:

```
sfw.login( setupHomeView );
```

### Table 28 swf.login

The login function of the SalesforceWrapper just needs a single parameter – a reference to a function that will be invoked once the user has successfully logged in. We also noticed that the SalesforceWrapper refers to the ChildBrowser JavaScript class. The ChildBrowser class is part of the ChildBrowser PhoneGap native extension, which is available for iOS, Android, BlackBerry, and Windows Phone. This enables our PhoneGap application to have a "child" web view, which in this case, is used for authenticating the Force.com API. Once the SalesforceWrapper class is initialized, the Moustache.js templates are initialized. Each template is a separate HTML file that will be used to render the interface, and they must be loaded into memory before they can be consumed.

```
var templates = {
  structure:"views/structure.html",
  home:"views/home.html",
  form:"views/formView.html",
  list:"views/dataView.html",
  listItem:"views/listItem.html",
  loaded: 0,
  requested: 0,
};

function onDeviceReady( event ) {
  console.log("deviceready");

  //initialize salesforce wrapper
  sfw = new SalesforceWrapper();

  //load Moustache HTML templates
  for (var key in templates) {
    (function() {
      var _key = key.toString();
      if ( _key != "loaded" && _key != "requested" ){
        templates.requested ++;

        var templateLoaded = function( template ){
          onTemplateLoaded( template, _key );
        }
      }
    })();
  }
}
```



```
        $.get( templates[ _key ], templateLoaded );
    }
    }());
}
}

function onTemplateLoaded(template, key) {

    console.log( key + ": " + template);
    templates[ key ] = template;
    templates.loaded ++;

    if ( templates.loaded == templates.requested ) {
        setupDefaultView();
    }
}
```

**Table 29 Moustache.js**

Once the templates have been loaded, the setupDefaultView() function gets invoked. This sets up the initial user interface, based upon the templates.structure template.

```
var header, container;

function setupDefaultView() {
    console.log("setupDefaultView");
    $("body").html( templates.structure );
    header = $("body").find("#header");
    container = $("body").find("#content");

    $('#login').tap(function (e) {
        e.preventDefault();
        sfw.login( setupHomeView );
    });
}
```

**Table 30 setupdefaultView**

It then sets a reference to the "header" and "container" elements from that template for future usage, then adds an event handler to the "login" button so that when the user taps the button, the login functionality from the SalesforceWrapper class is invoked. We can view the HTML from the templates.structure template below:

```
<div id="header">Welcome</div>
<div id="content">
  <h3 style="padding-top:1.5em; padding-bottom:1.5em;" class="alert
alert-info">Press the "Login" button to authenticate via
Database.com.</h3>
  <br/><br/>
  <a id="login" class="btn btn-success">Login</a>
</div>
```

**Table 31 HTML templates structure**

All authentication is handled within the ChildBrowser, and is completely maintained by Database.com. As developers, we don't have to worry about user account management or login functionality, as force.com handles this for us. The user simply must have permission to access your data objects (database) in Database.com. Once the user is successfully authenticated, the setupHomeView() function will be invoked, and will display the "Home" screen of the application. The setupHomeView() function resets/clears the contents of the container element, and then fills it with the contents of the templates.home template and adds the appropriate event handlers.

```
function resetContainer() {
  //this removes child elements and cleans up event handlers
  container.children().remove();
  container.removeClass("nopadding");
}
function setupHomeView() {
  resetContainer();
  container.html( templates.home );
  header.html( "Welcome" );
  $('#addNew').tap(function (e) {
    setupFormView();
    e.preventDefault();
    e.stopPropagation();
    return false;
  });
  $('#queryMyRecords').tap(function (e) {
    setupListView();
    e.preventDefault();
    e.stopPropagation();
    return false;
  });
}
```

**Table 32 resetcontainer and setuphome functions**

We can view the templates.home template below:

```
<h3>Please select an option:</h3>

<a id="addNew" class="btn btn-info">Add New Record</a>
<br/>
<a id="queryMyRecords" class="btn btn-info">Query My Records</a>
```

**Table 33 home tamplate**

On a device, the user will see the rendered output with the new entry select and the query select. There are two buttons, one for adding a new record, and another to query existing records from Database.com. Next, let's examine what happens when the user clicks on the "Add New Record" button. When the user clicks this button, the `setupFormView()` button will be invoked, which creates a new form for gathering data from the user.

```
function setupFormView(data) {
    resetContainer();
    var html = Mustache.to_html( templates.form, data );
    container.html( html );
    currentLead = data;

    //request current location
    if ( !(data && data.Id) ) {
        header.html( "New Lead" );
        navigator.geolocation.getCurrentPosition(onGeoSuccess, onGeoError
    );
    }
    else {
        header.html( "Edit Lead" );
    }

    $('#save').tap( saveFormData );
    $('#cancel').tap( navigateBackFromFormView );
}
```

**Table 34 SetupFormView**

The `setupFormView()` function will clear the container element, and fill it with HTML from the `templates.form` template. This is where we can see that templating become very useful. Next, let's examine the form template:

```

<div id="form">
  <label for="first">First Name</label>
  <input id="first" type="text" value="{{First__c}}" />

  <br/>
  <label for="last">Last Name</label>
  <input id="last" type="text" value="{{Last__c}}" />

  <br/>
  <label for="phone">Telephone</label>
  <input id="phone" type="text" value="{{Telephone__c}}" />

  <br/>
  <label for="email">Email</label>
  <input id="email" type="text" value="{{Email__c}}" />

  <br/>
  <label for="notes">Notes</label>
  <textarea id="notes" type="text">{{Notes__c}}</textarea>

  <br/>
  <span id="location" class="alert alert-info">Location:
  {{Latitude__c}},{{Longitude__c}}</span>

  <br/>
  <br/>
  <a id="save" class="btn btn-success">Save</a>
  <a id="cancel" class="btn btn-danger">Cancel</a>
</div>

```

**Table 35 The formtemplate**

The form contains the HTML that will be used to generate the user interface. Values wrapped in double brackets "{{" and "}}" will be populated by data passed into the Mustache templating engine. Each value inside of the brackets corresponds to an attribute of a data object passed into Mustache.js. The HTML string for the UI is generated using the Mustache.to\_html() function. As we can see in the setupFormView function above that the to\_html() function uses a data parameter to generate the template HTML. When creating a new Lead, an empty object is passed into this function, so the form's HTML has blank values. When editing an existing lead, this exact function gets invoked, however a populated data object is passed in. This reuses the exact same HTML template, however populates it with the data that was passed in. When rendered within the PhoneGap application, we will see the form displayed as shown below. The GPS location is obtained through the PhoneGap API when capturing a new Lead. The user can enter appropriate data, and click either "Save" or "Cancel".

If the user cancels, the application will take the user back. However, if the user saves, this is where the application pushes data to Database.com. Inside of the saveFormData() JavaScript function, the data is retrieved from the input form, and assigned to a "data" object, which will be sent to Database.com. The saveFormData() function is used for both creating a new Lead, as well as updating and existing lead. If the currentLead variable exists, then the user is currently editing an existing Lead, otherwise the user is creating a new lead. If the user is creating a new Lead, the forcetk client.create function is invoked, otherwise, the client.update function is invoked.

```
function saveFormData( event ) {  
  
    var data = {};  
    data.First__c = $("#first").val();  
    data.Last__c = $("#last").val();  
    data.Telephone__c = $("#phone").val();  
    data.Email__c = $("#email").val();  
    data.Notes__c = $("#notes").val();  
    if ( currentLead ) {  
        //copy it back to the object in memory  
        currentLead.First__c = data.First__c;  
        currentLead.Last__c = data.Last__c;  
        currentLead.Telephone__c = data.Telephone__c;  
        currentLead.Email__c = data.Email__c;  
        currentLead.Notes__c = data.Notes__c;  
        //use the original lat/lon location  
        data.Latitude__c = currentLead.Latitude__c;  
        data.Longitude__c = currentLead.Longitude__c;  
    }  
    else if ( lastCoords ) {  
        data.Latitude__c = lastCoords.latitude;  
        data.Longitude__c = lastCoords.longitude;  
    }  
    try {  
        if ( currentLead == undefined ) {  
            sfw.client.create("Lead__C", data, saveDataSuccess,  
saveDataError );  
        } else {  
            sfw.client.update("Lead__C", currentLead.Id, data,  
saveDataSuccess, saveDataError );  
        }  
    }  
    catch(e){  
        console.log(e);  
    }  
}
```

```
function saveDataSuccess( result ) {  
    alert("Data Saved");  
    navigateBackFromFormView();  
}  
  
function saveDataError( request, status, error){  
    console.log( request.responseText );  
    alert( request.responseText );  
}
```

**Table 36 saveformdata function**

When calling `client.create()`, we just need to pass the type of object, the data object, and success and error callback functions. When referencing the type of object, we have noticed that it is "Lead\_\_c", instead of "Lead", as we may have expected. This is because custom objects and custom data fields in Database.com must use a "\_\_c" suffix. When calling `client.update()`, we need to pass the type of object, the ID of the object being updated, the data object containing new values, and the success and error callback functions. If there is an error when saving data, a message will be displayed to the user. If there were no errors, the user will be taken back to the previous view. Next, let's examine the workflow for retrieving data from Database.com. From the application home screen click on the "Query My Records" button. This will invoke the `setupListView()` JavaScript function.

```
function setupListView() {  
    resetContainer();  
  
    var html = templates.list;  
    container.html( html );  
    header.html( "Leads" );  
  
    if(lastData) {  
        renderListData();  
    }  
    else {  
        queryRecords();  
    }  
  
    $('#cancel').tap( setupHomeView );  
}
```

**Table 37 setupListview function**

The `setupListView()` function will clear the container, and populate it with HTML from the `templates.list` template. This template doesn't actually display the data, instead it sets up the `dataContainer` element where list data will be displayed.

```
<div id="dataContainer">loading...</div>
<br/><br/>
<a id="cancel" class="btn btn-danger" style="width:70%">Cancel</a>
```

**Table 38 datacontainer element**

If the user is navigating back from an edit form, data that is already in memory will be rendered. However, for a new request, the `queryRecords()` function will be invoked. Querying data from Database.com is very easy. When using the `forcetk.js` toolkit, we simply need to invoke the `client.query()` method, and pass in a SOQL query with success and error callback functions. SOQL is the Salesforce Object Query Language, with is very similar to SQL (Structured Query Language) used by other database offerings. SOQL enables us to create countless custom data queries from related objects, just as we may with SQL. Database.com also has an online Workbench tool that lets us test SOQL queries before putting them into our actual application. The `queryRecords()` function is below, where we can see it passing in the SOQL query to retrieve data:

```
function queryRecords() {
    var query = "SELECT
Email__c,First__c,Id,Last__c,Latitude__c,Longitude__c,Notes__c,Telephone__c
"+
    "FROM Lead__c " +
    "ORDER BY Last__c, First__c"

    sfw.client.query( query, onQuerySuccess, onQueryError );
}

function onQuerySuccess( response ) {
    lastData = { "records": response.records };
    renderListData();
}

function onQueryError( request, status, error ) {
    $("#dataContainer").html( "Error loading data: <br/>" +
request.responseText );
}
```

**Table 39 Function Queryrecords**



Once data is returned from Database.com, the `onQuerySuccess` function will be invoked, which invokes the `renderListData()` function.

```
function renderListData() {
  if ( lastData ) {
    container.addClass("nopadding");
    var html = Mustache.to_html( templates.listItem, lastData );
    $("#dataContainer").html( html );
    $("#dataContainer").find("li").tap( onListItemTap );
    $("#cancel").tap( navigateBackFromListView );
  }
}
```

**Table 40 Function renderlistData**

The `renderListData()` function uses the `templates.listItem` template to generate a HTML list based on the data the gets returned from the server, and then adds a "tap" event handler to all `<li>` elements. Next, let's examine the contents of the `templates.listItem` template:

```
<ul>
{{#records}}
  <li id="{{Id}}">
    <strong>{{Last_c}}, {{First_c}}</strong>
    <div class="subtext">{{Email_c}} </div>
  </li>
{{/records}}
</ul>
```

**Table 41 listitem template**

This template instructs Mustache.js to loop over all "records" and will output a `<li>` element containing the Lead's name and email address. Once the data is rendered, we can see the interface. All list formatting is handled in CSS, so the visual presentation looks more like a mobile application list, instead of a bulleted HTML list. When the user taps on a list item, the `onListItemTap` function will be invoked. This function will obtain a reference to the JavaScript object corresponding to the list item that was tapped, and then will invoke the `setUpFormView()` function discussed earlier in this article, passing in the appropriate object.

```
function onListItemTap( event ) {
  var target = $( event.target )
  while (target.get(0).nodeName.toUpperCase() != "LI" ) {
    target=target.parent();
  }
}
```

```

var id = target.attr("id");

var data = getRecordById(id);
setupFormView( data );

event.preventDefault();
event.stopPropagation();

function getRecordById( id ) {

    if ( !lastData ) return;
    var records = lastData.records;

    for (var x=0; x<records.length; x++ ) {
        if (records[x].Id == id ) {
            return records[x];
        }
    }
}
}

```

**Table 42 onListItemTap getrecordbyId**

Since it reuses the `setupFormView()` function, it will reuse the `templates.form` template, however populating it with the data retrieved from Database.com. From now on we can carry on with the consuming section. The complete code of the project is included in a CD-ROM. In subparagraph 6.3.5 we setup step by step the `childbrowser` plugin.

### 6.3.3 The `childBrowser` Plugin

Once we have our consumer key, we're ready to start pushing and pulling data in and out of Database.com. After this database creation the greatest challenge is to successfully access this database from our application. For this reason we must install The `childBrowser` plugin into `phonegap` project which enables our `PhoneGap` application to have a "child" web view. This process took us about one day because of the bugs and the incompatibility of versions. The code for running an external web view is given in the annex. (`works.js`). At first we need to download the `childBrowser` project from the github site<sup>78</sup>. Then we have to add a new package in our `src` folder of our project with the name : `Com.phonegap.plugins.childBrowser`. This new package is made because we will add in it the `ChildBrowser.java` file. We also need the navigation image buttons for our external browser and for this reason we put the folder `childbrowser` into the `assets/www`. Last steps for `childbrowser` installation is the file `ChildBrowser.js` which we add into the folder `js`. In the folder `xml` there are two `xml` files , the first is the `config.xml` and the second the `plugins.xml`. We set the tags as seen in the next tables for each file.

```
<plugin name="ChildBrowser" value="com.phonegap.plugins.childBrowser.ChildBrowser"/>
```

**Table 43 Cofig.xml**

```
<plugins>  
<plugin name="ChildBrowser" value="com.phonegap.plugins.childBrowser.ChildBrowser"/>  
</plugins>
```

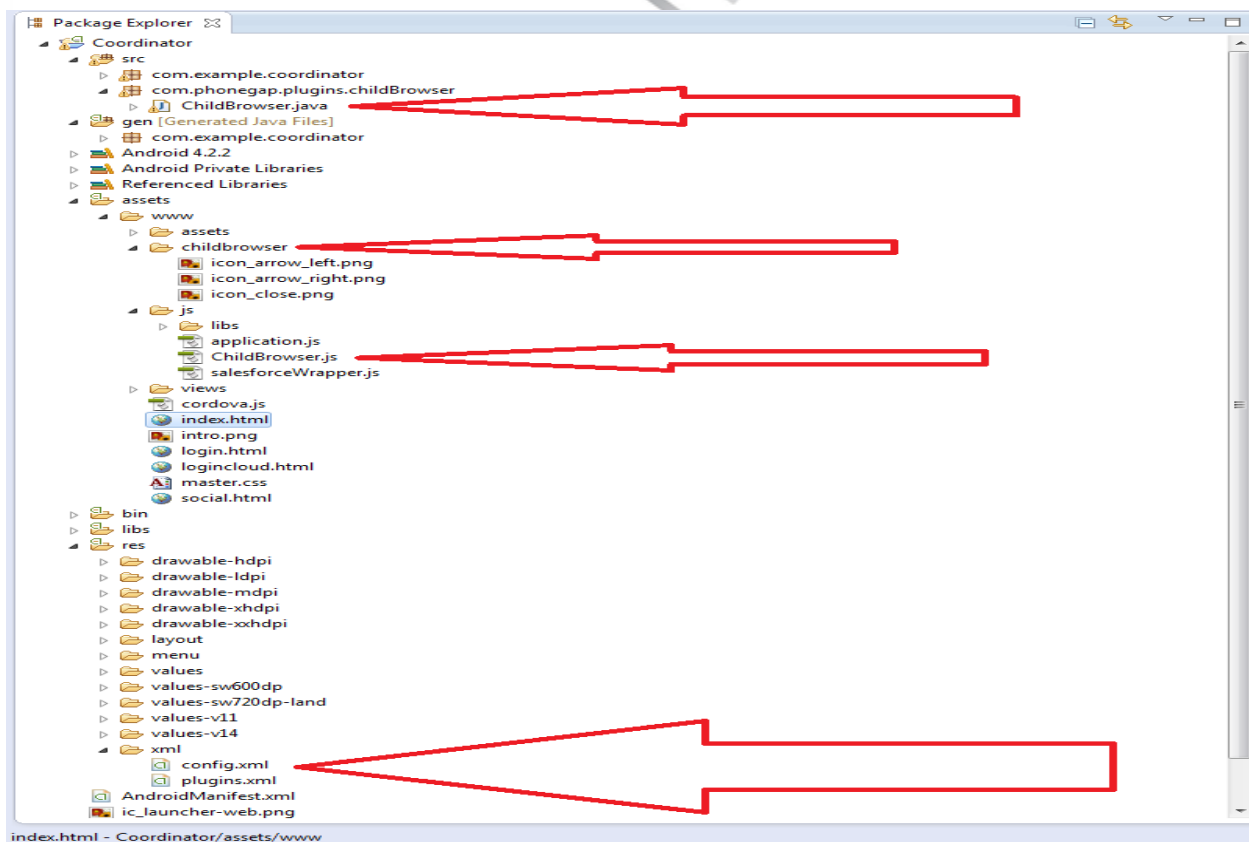
**Table 44 Plugin.xml**

When we want to use the function of the childbrowser plugin we put the next line of code in our file

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

**Table 45 childbrowser script reference**

The files must be placed as the picture below



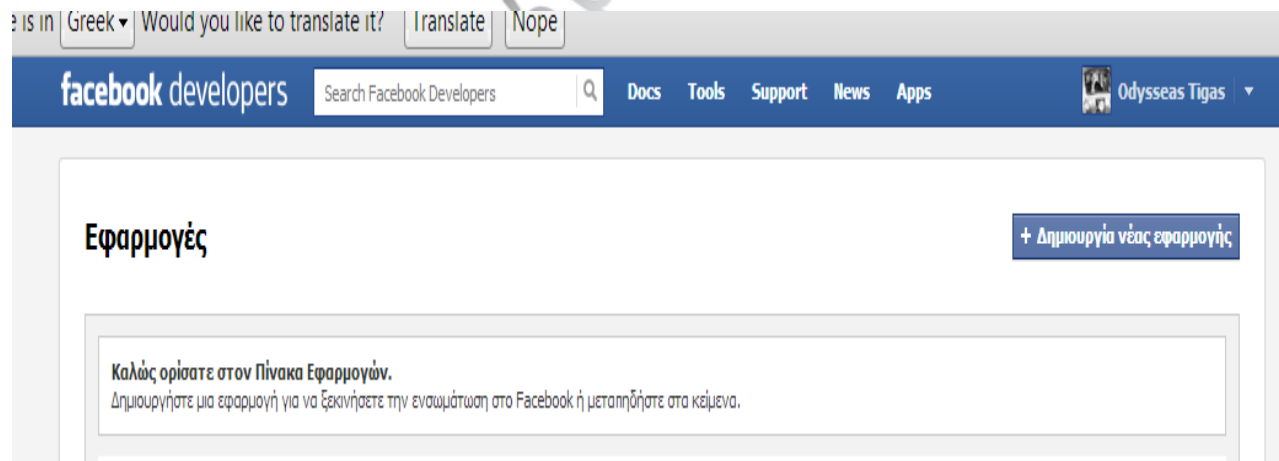
**Figure 55 Child Browser files Placement**



**Figure 56 Child Browser Style**

### 6.3.4 Social Media Section

The last part of this project has to do with the social media service integration. There is a variety of solutions according to native applications for all platforms. When we talk about cross platform applications things are very tuff to manipulate. In terms of functionality there are no serious borders, but the implementation of these integrations is a good reason to avoid this adventure. Plugins like childbrowser and cloud connectivity are new knowledge for us and our average coding skills because of our academic past (electronics engineering department with no more than three programming classes). The most desired part of this project is not the cloud integration but the social media integration via Facebook, because it can make our applications famous. Without social media integration the application will stay in marketplace with very few views and downloads. The social media networks such as Facebook and twitter can offer us better marketing strategies with wide acceptance. In the next subparagraph we explain our efforts to register as developers in Facebook, proceed with creating an application to Facebook in order to get an app ID and install the Facebook plugin into our existing project. We must register our application first from the developer’s page of Facebook. (<https://developers.facebook.com/apps>).



**Figure 57 facebook developers create account**

Then we create our new application and we provide our info.

The screenshot shows the 'Δημιουργία νέας εφαρμογής' (Create New Application) form in the Facebook App Center. The form includes the following fields and options:

- App Name:** A text input field with a question mark icon.
- Χώρος ονόματος εφαρμογής:** A dropdown menu with 'Προαιρετικό' (Optional) selected and a question mark icon.
- App Category:** A dropdown menu with 'Άλλο' (Other) selected and a question mark icon, followed by a 'Choose a sub-category' dropdown menu.
- Φιλοξενία ιστοσελίδας:** A checkbox labeled 'Ναι, θα ήθελα τη δωρεάν φιλοξενία ιστοσελίδων που προσφέρεται από το Heroku' (Yes, I would like the free website hosting provided by Heroku) with a question mark icon and a '(Learn More)' link.

At the bottom of the form, there is a disclaimer: 'Αν προχωρήσετε, συμφωνείτε με τις Πολιτικές της πλατφόρμας του Facebook' (If you proceed, you agree to the Facebook platform's Policies). To the right of the disclaimer are two buttons: 'Συνέχεια' (Next) and 'Ακύρωση' (Cancel).

**Figure 58 New Application Creation**

After these basic steps we have to keep in mind that we must keep our ID in a safe place for later use. The credentials are shown in the image below.

Facebook Developers | Docs | Tools | Support | News | Apps | Odysseas Tigas

φαρμογές ▶ Coordinator ▶ Βασικές

**Coordinator**  
**App ID:** 550915941631409  
**App Secret:** e0fd3374a1cfacca5e6ef5b60c258b90 (reset)  
 This app is in **Sandbox Mode** (Only visible to Admins, Developers and Testers)

**Βασικές πληροφορίες**

**Display Name:** [?] Coordinator

**Namespace:** [?] [ ]

**Contact Email:** [?] otigas@hotmail.com

**App Domains:** [?] Enter your site domains and press enter

**Hosting URL:** [?] You have not generated a URL through one of our partners (Get one)

**Sandbox Mode:** [?]  Enabled  Disabled

**Select how your app integrates with Facebook**

- Website with Facebook Login** Log in to my website using Facebook.
- App on Facebook** Use my app inside Facebook.com.
- Σύνδεση στο Διαδίκτυο από κινητό** Bookmark my web app on Facebook mobile.
- Native iOS App** Publish from my iOS app to Facebook.

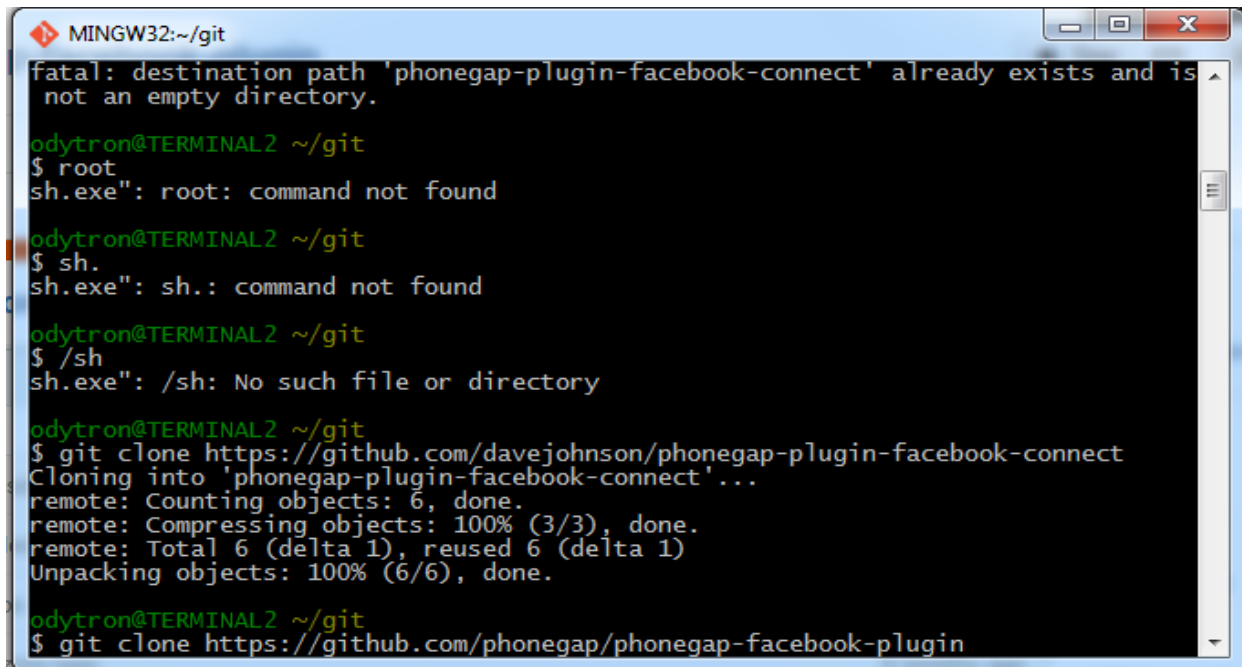
**Figure 59 Application information**

In the table below we provide the credentials in editable format.

|             |   |
|-------------|---|
| Coordinator |   |
| App ID:     | 550915941631409                         |
| App Secret: | e0fd3374a1cfacca5e6ef5b60c258b90(reset) |

**Table 46 App\_ID**

Then we have to download the Facebook plugin from and extract it in our desktop from : <https://github.com/phonegap/phonegap-facebook-plugin>. The images below has to do with the step by step process of the github project clone and setup in our project.



```
MINGW32:~/git
fatal: destination path 'phonegap-plugin-facebook-connect' already exists and is
not an empty directory.

odytron@TERMINAL2 ~/git
$ root
sh.exe": root: command not found

odytron@TERMINAL2 ~/git
$ sh.
sh.exe": sh.: command not found

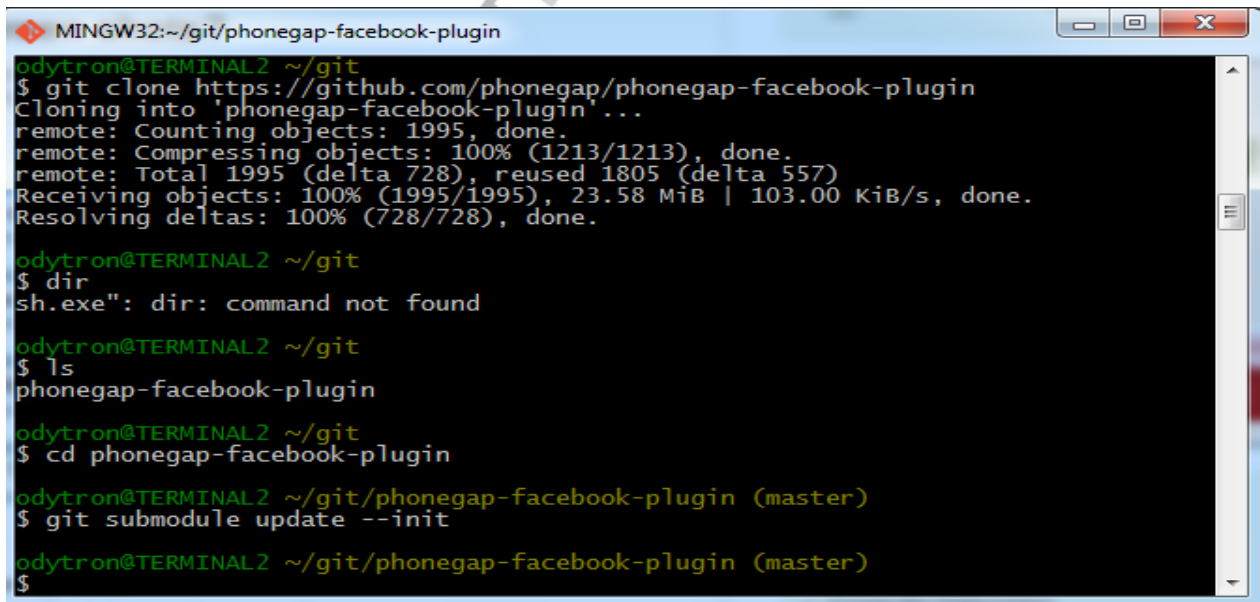
odytron@TERMINAL2 ~/git
$ /sh
sh.exe": /sh: No such file or directory

odytron@TERMINAL2 ~/git
$ git clone https://github.com/davejohnson/phonegap-plugin-facebook-connect
Cloning into 'phonegap-plugin-facebook-connect'...
remote: Counting objects: 6, done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 6 (delta 1), reused 6 (delta 1)
Unpacking objects: 100% (6/6), done.

odytron@TERMINAL2 ~/git
$ git clone https://github.com/phonegap/phonegap-facebook-plugin
```

Figure 60 clone git

Then we have to update our repository



```
MINGW32:~/git/phonegap-facebook-plugin

odytron@TERMINAL2 ~/git
$ git clone https://github.com/phonegap/phonegap-facebook-plugin
Cloning into 'phonegap-facebook-plugin'...
remote: Counting objects: 1995, done.
remote: Compressing objects: 100% (1213/1213), done.
remote: Total 1995 (delta 728), reused 1805 (delta 557)
Receiving objects: 100% (1995/1995), 23.58 MiB | 103.00 KiB/s, done.
Resolving deltas: 100% (728/728), done.

odytron@TERMINAL2 ~/git
$ dir
sh.exe": dir: command not found

odytron@TERMINAL2 ~/git
$ ls
phonegap-facebook-plugin

odytron@TERMINAL2 ~/git
$ cd phonegap-facebook-plugin

odytron@TERMINAL2 ~/git/phonegap-facebook-plugin (master)
$ git submodule update --init

odytron@TERMINAL2 ~/git/phonegap-facebook-plugin (master)
$
```

Figure 61 update git



The next steps are going through the procedure in detail.

1. We Move `cdv-plugin-fb-connect.js` into `android phonegap webroot assets/www`
2. We Include it in `index.html` `<script type="text/javascript" charset="utf-8" src="cdv-plugin-fb-connect.js"></script>`
3. We Add `<plugin name="org.apache.cordova.facebook.Connect" value="org.apache.cordova.facebook.ConnectPlugin" />` into `res/xml/plugins.xml`
4. We Download `facebook_js_sdk.js` from <https://github.com/davejohnson/phonegap-plugin-facebook-connect/tree/master/lib>
5. We Create package `org.apache.cordova.facebook` and class `ConnectPlugin.java`. Copy contents of this file into it <https://github.com/davejohnson/phonegap-plugin-facebook-connect/blob/master/native/android/src/org/apache/cordova/facebook/ConnectPlugin.java>
6. We Add package `com.facebook.android` and `.java` files from <https://github.com/facebook/facebook-android-sdk/tree/master/facebook/src> to your java source. This enables `ConnectPlugin.java` to compile.
7. We Add import to `com.package.your.R`; or whatever is our android package so that our `R.java` will be imported to `FbDialog.java`.
8. We Merge facebook icons from <https://github.com/facebook/facebook-android-sdk/tree/master/facebook/res> to our project `res` folder. This enables Facebook SDK sources to compile. Especially is needed this image <https://github.com/facebook/facebook-android-sdk/blob/master/facebook/res/drawable/close.png>
9. We Verify the facebook using javascript methods in here : <https://github.com/davejohnson/phonegap-plugin-facebook-connect/blob/master/example/www/index.html>

It is important not to forget the `App_ID` in our `index.html` file.

## 7. Results

### 7.1 The Finished Application

In this paragraph we are going to present the screenshots of the application. Next image shows the application icon in our android menu.

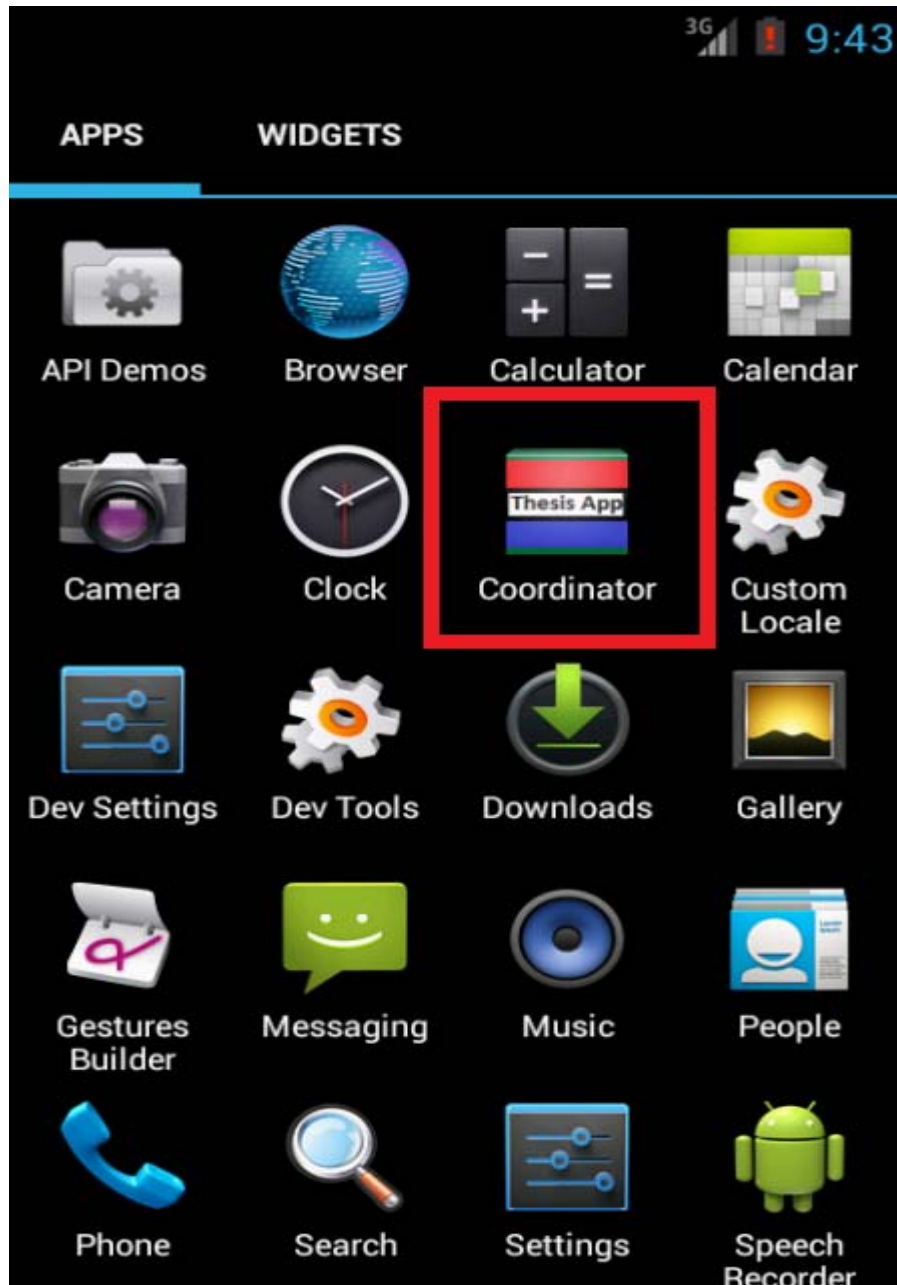


Figure 62 The launcher icon in android menu

The first page the user can view is the promotional splash image and then the central menu from which the user can select the functionalities.



**Figure 63 splash and functionalities**

The first functionality is the SQLite local save. The second functionality has to do with the cloud registration and the coordinates.

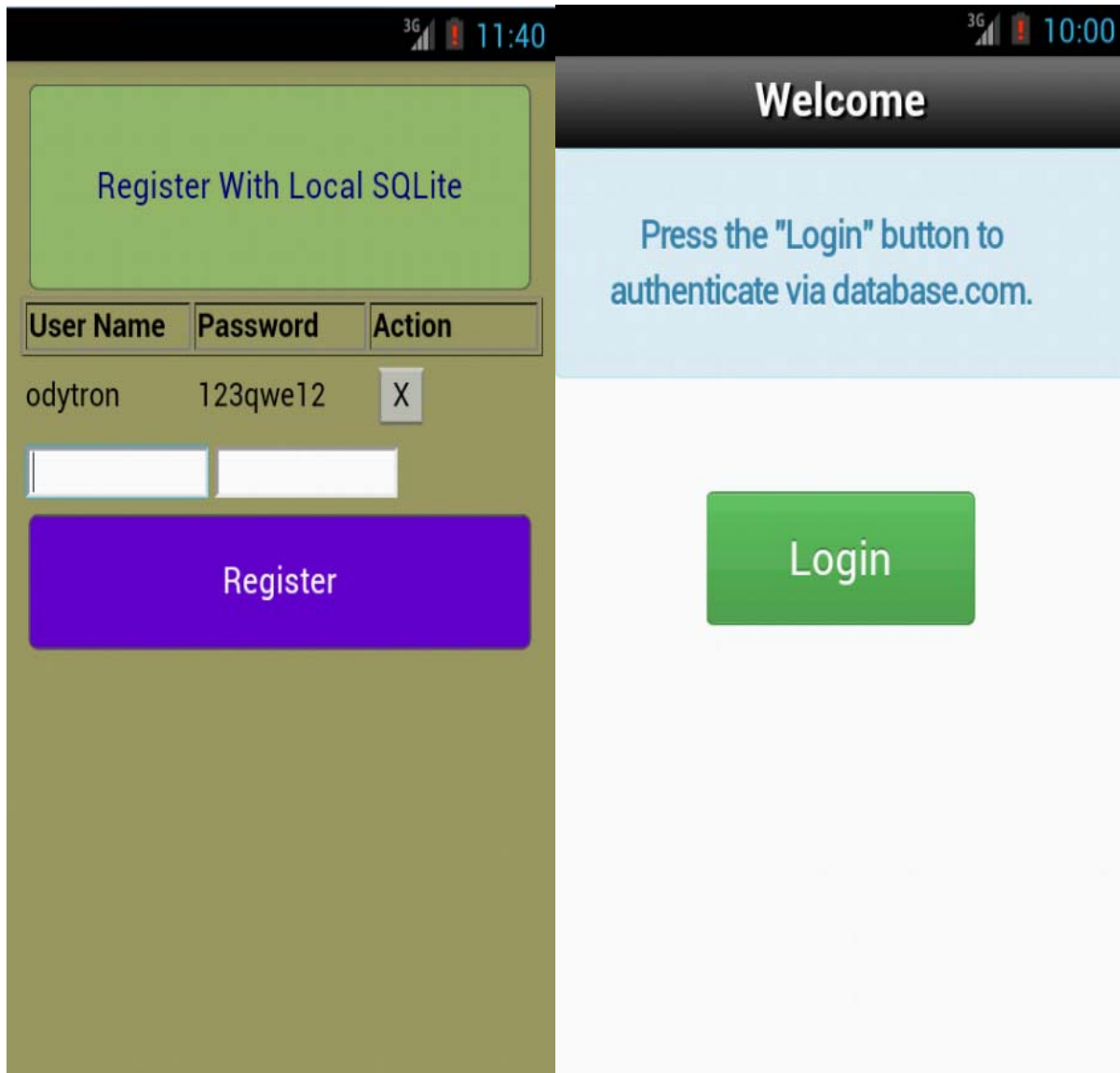


Figure 64 First and second functionality.

Next images describe the functionality of the second option

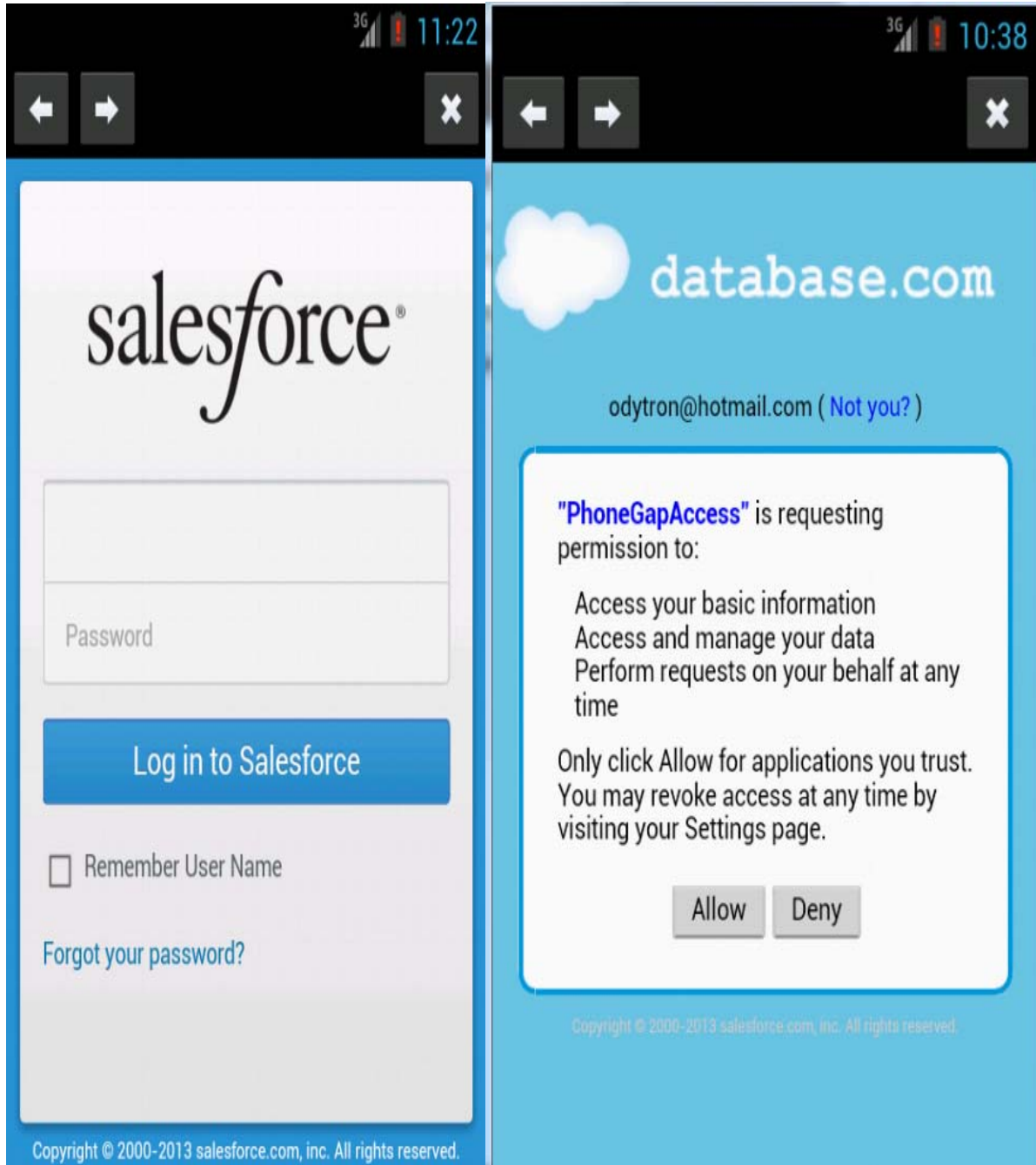
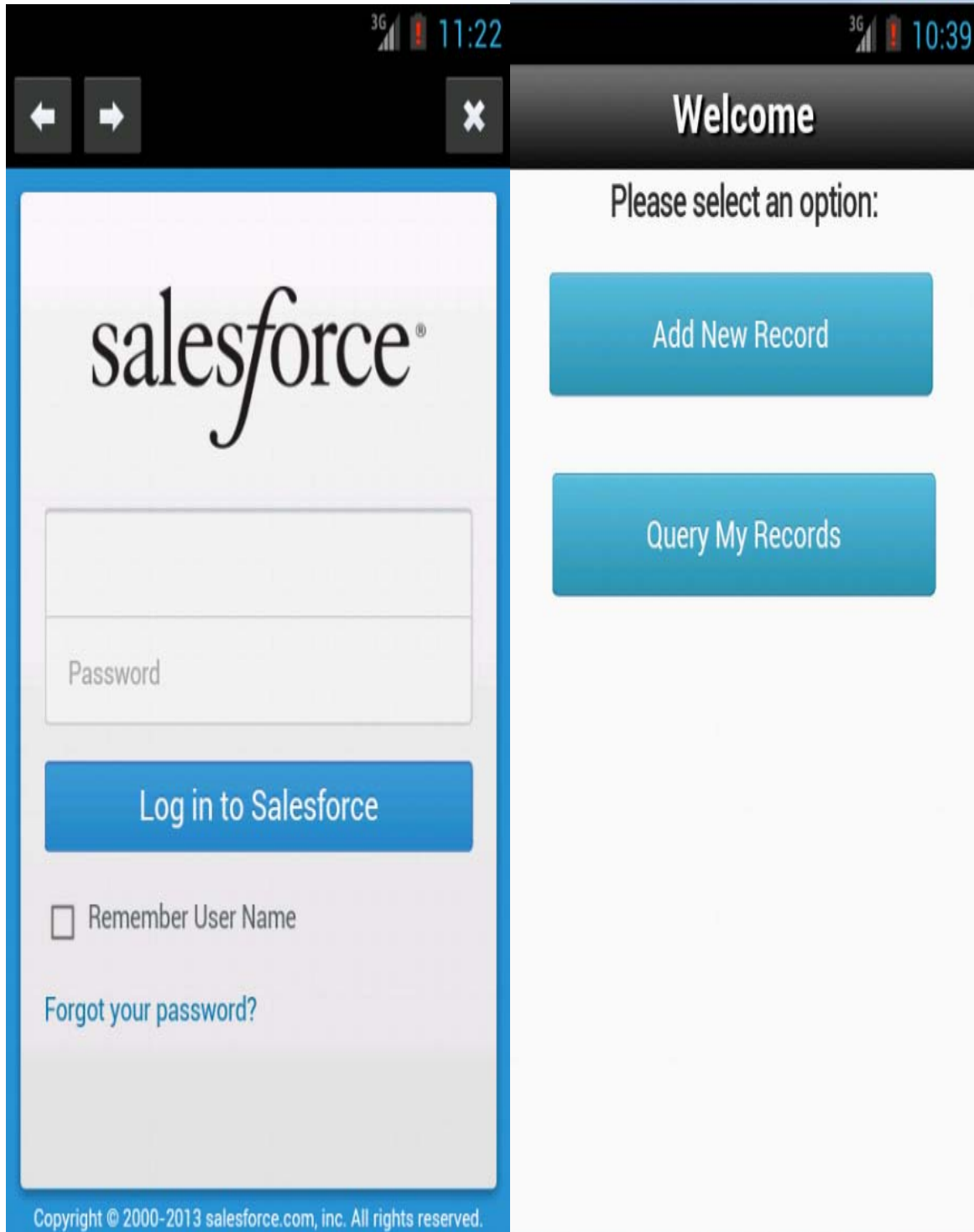


Figure 65 functionality of the second option I



**Figure 66 functionality of the second option II**

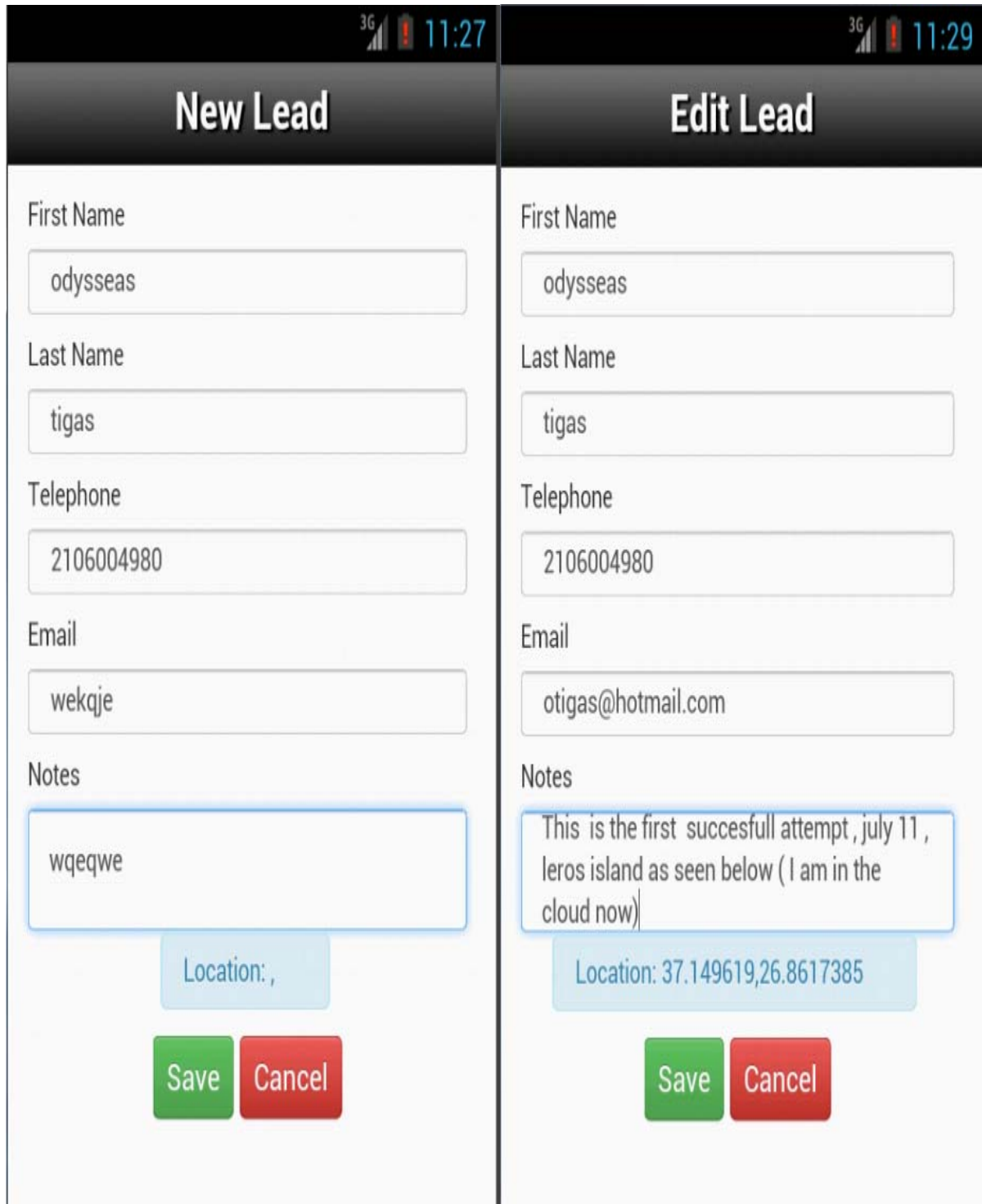


Figure 67 New Record and edit record with coordinates



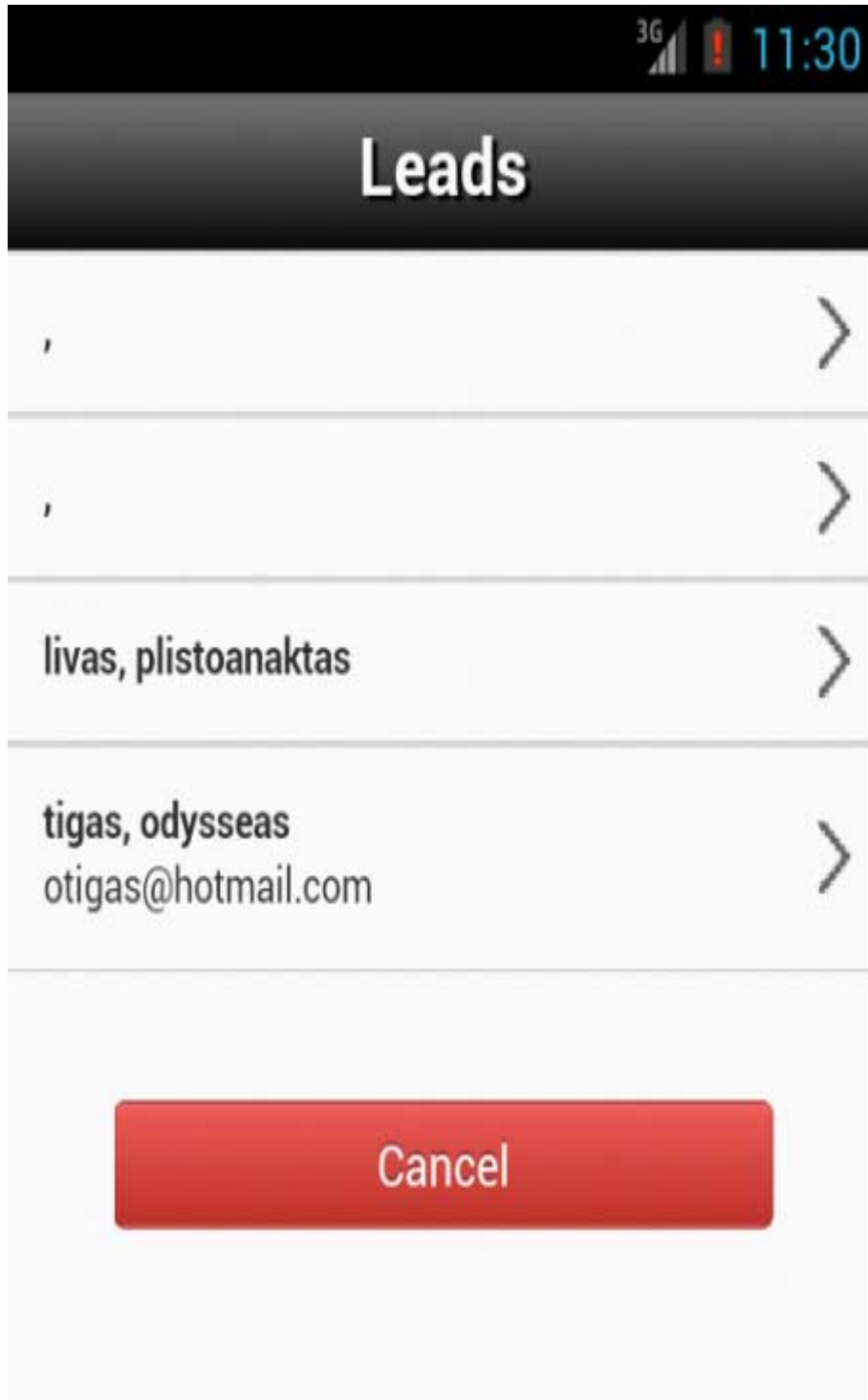


Figure 68 Show records

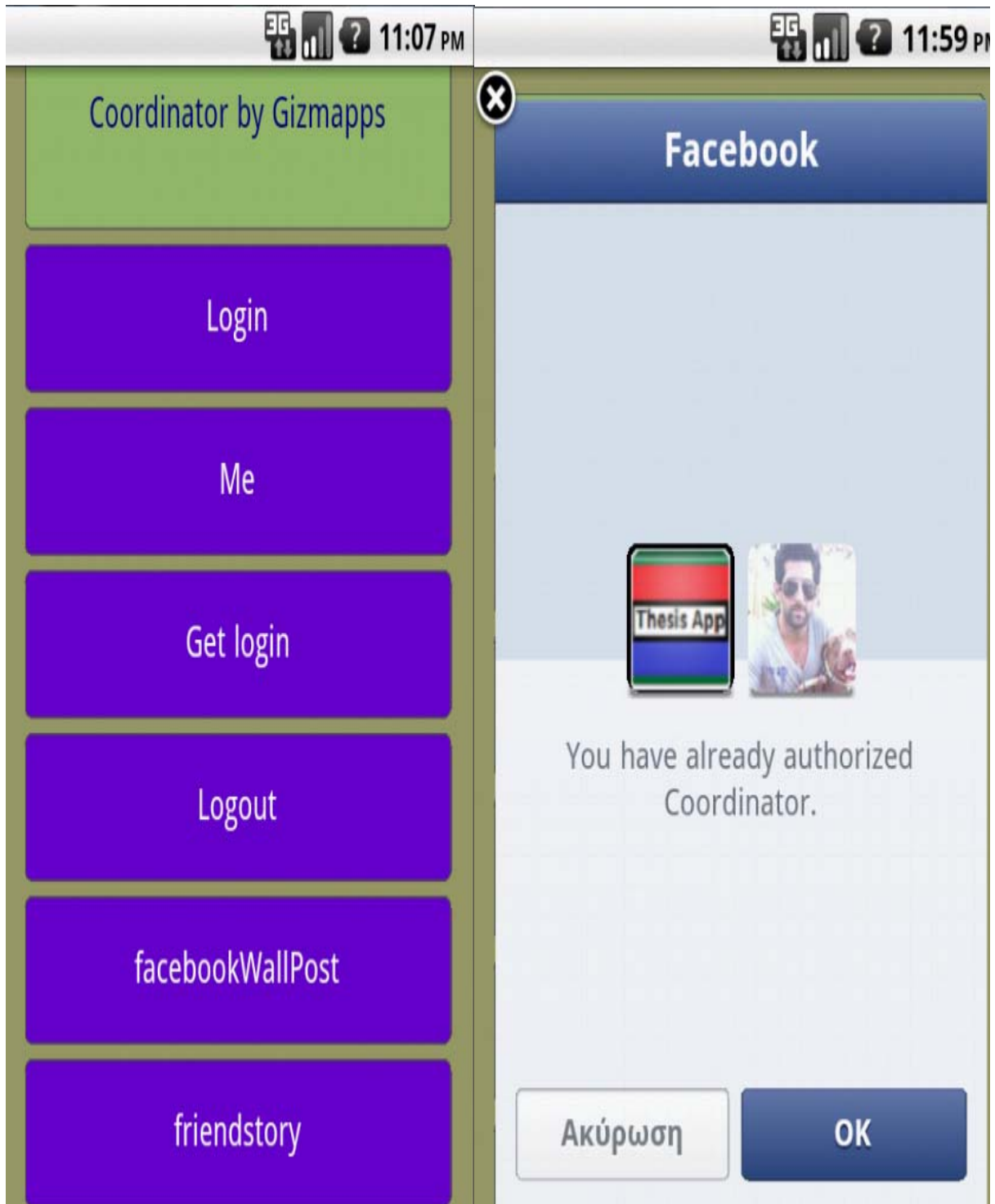
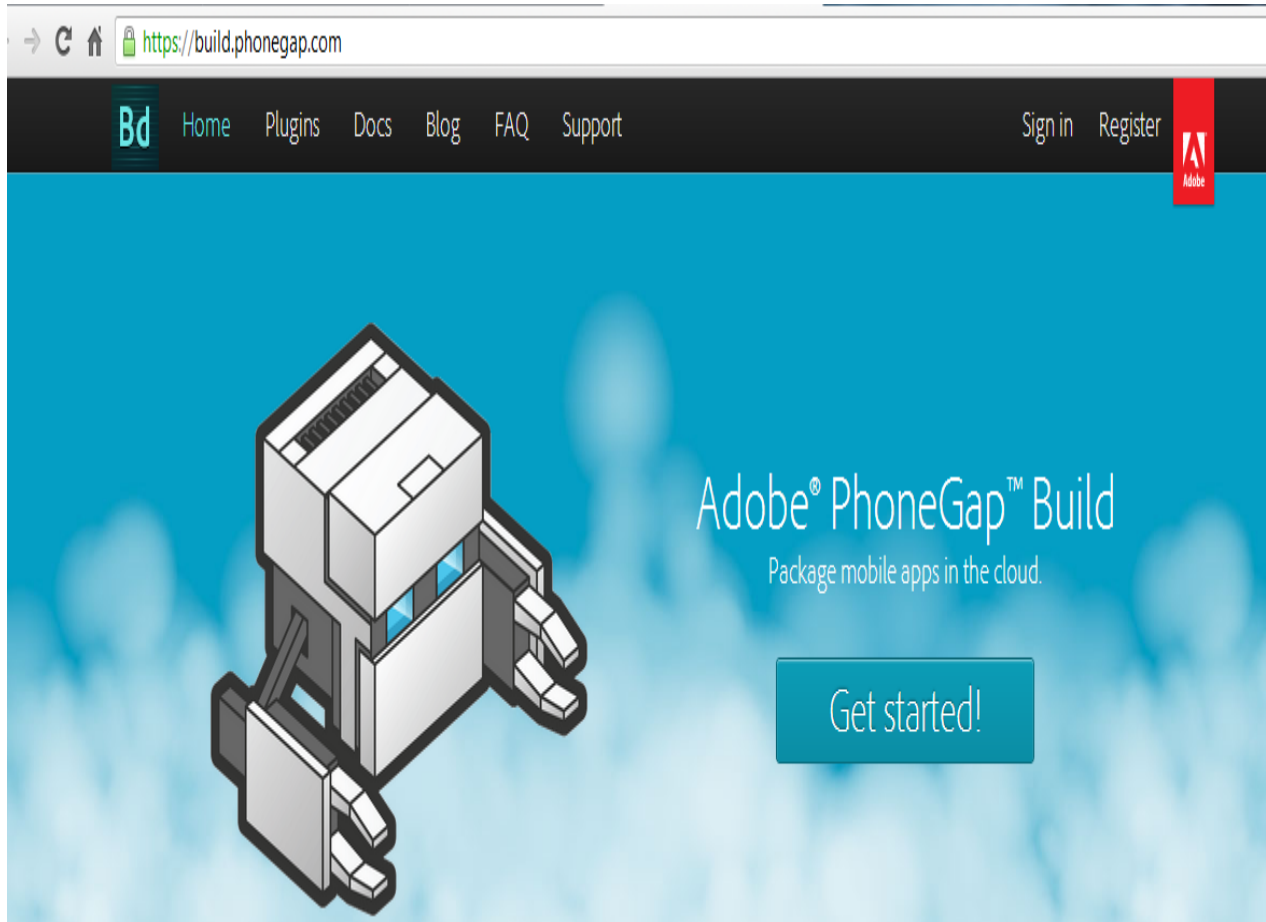


Figure 69 Third option and The login result

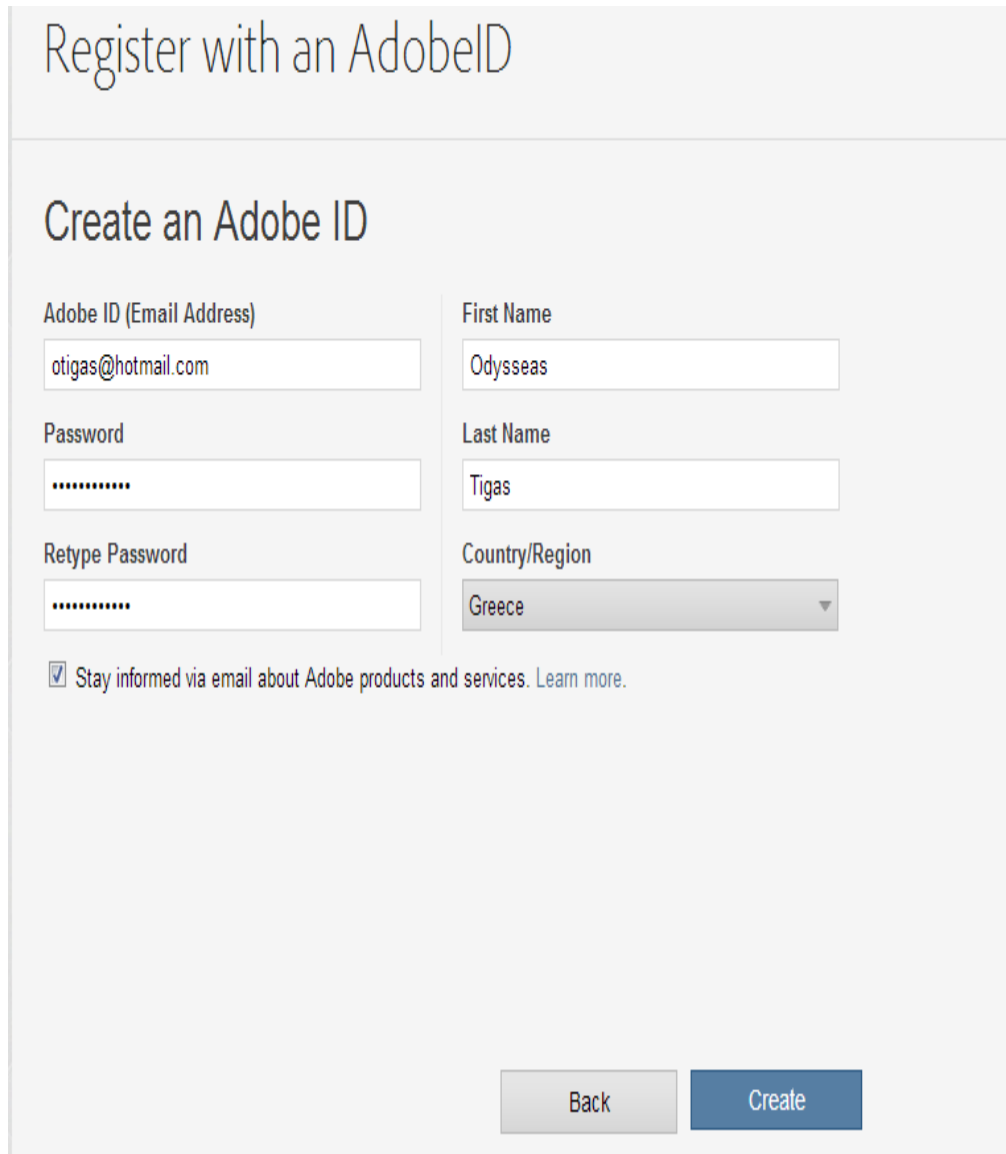
## 7.2 Deploy on Other Platforms

In this paragraph we are going to describe how to deploy on other platforms. The importance of this step is very important for this thesis. We can build our projects from phonegap site <https://build.phonegap.com>



**Figure 70 get started – build**

Then we have to choose the method of registration.



The screenshot shows the Adobe ID registration interface. At the top, it says "Register with an AdobeID". Below that, the main heading is "Create an Adobe ID". The form is divided into two columns. The left column contains three input fields: "Adobe ID (Email Address)" with the value "otigas@hotmail.com", "Password" with masked characters ".....", and "Retype Password" also with masked characters ".....". The right column contains three input fields: "First Name" with the value "Odysseas", "Last Name" with the value "Tigas", and "Country/Region" which is a dropdown menu currently showing "Greece". Below the input fields, there is a checkbox that is checked, with the text "Stay informed via email about Adobe products and services. [Learn more.](#)". At the bottom right of the form, there are two buttons: a grey "Back" button and a blue "Create" button.

**Figure 71 Register with adobe ID**

And then choose our plan. There are three possible plans as seen in the picture below.

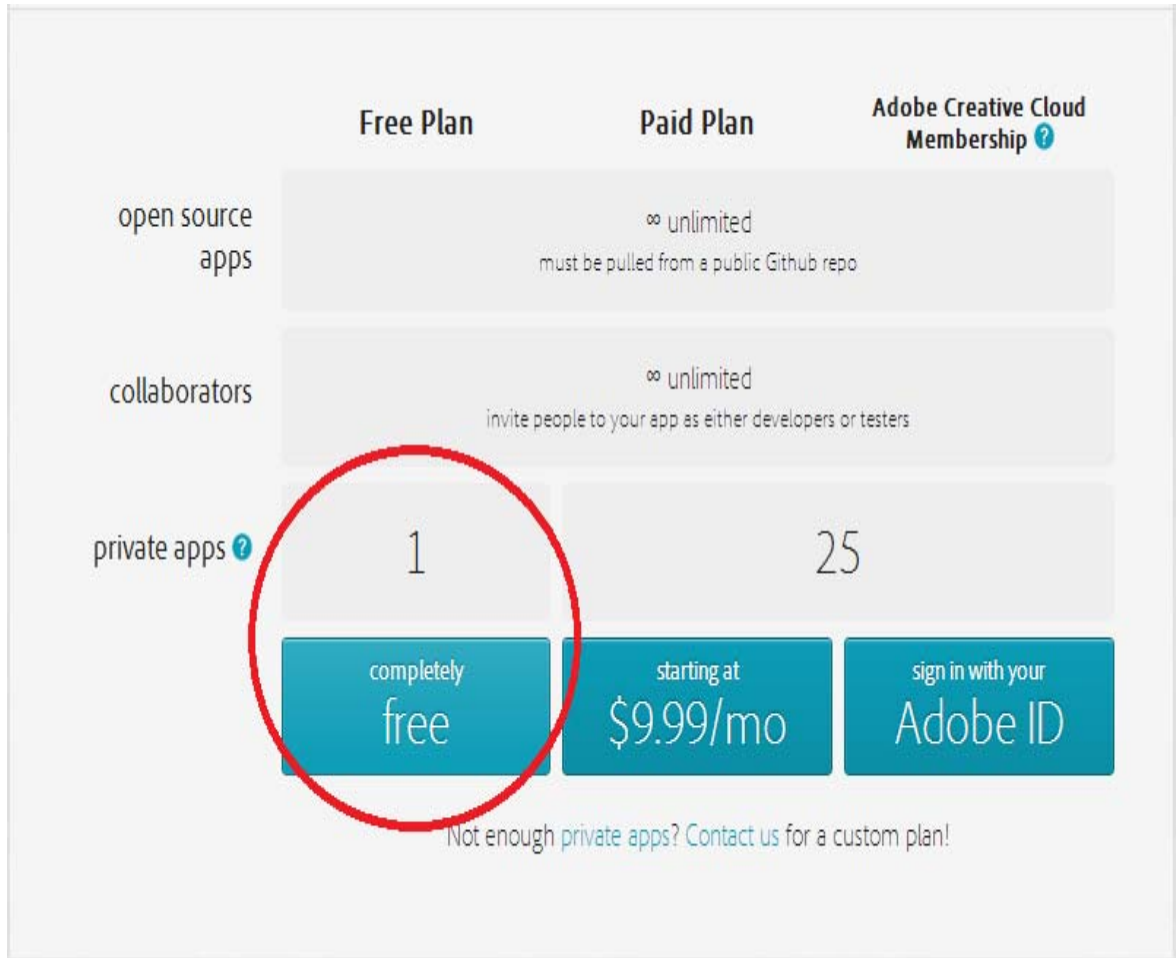


Figure 72 plan selection

Upload our file (zip format or connect to github)

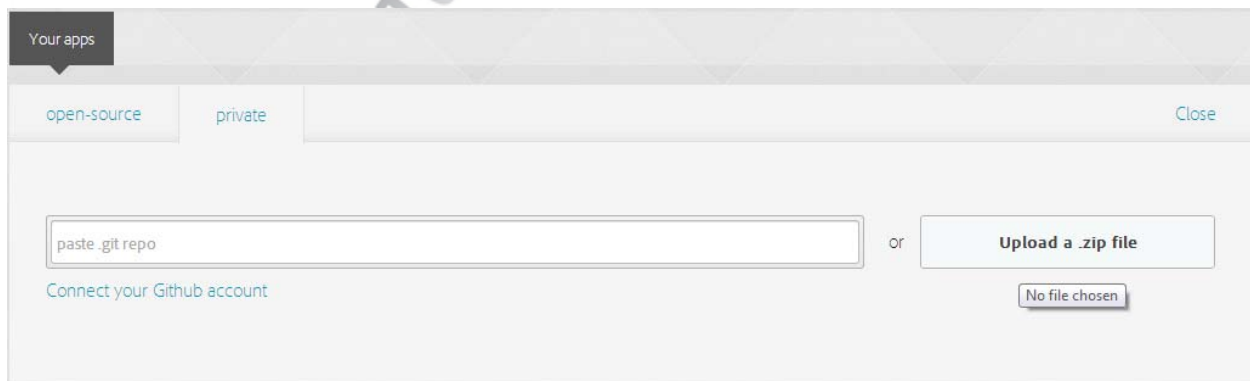


Figure 73 file upload from GIT

Then we have to wait for some moments.

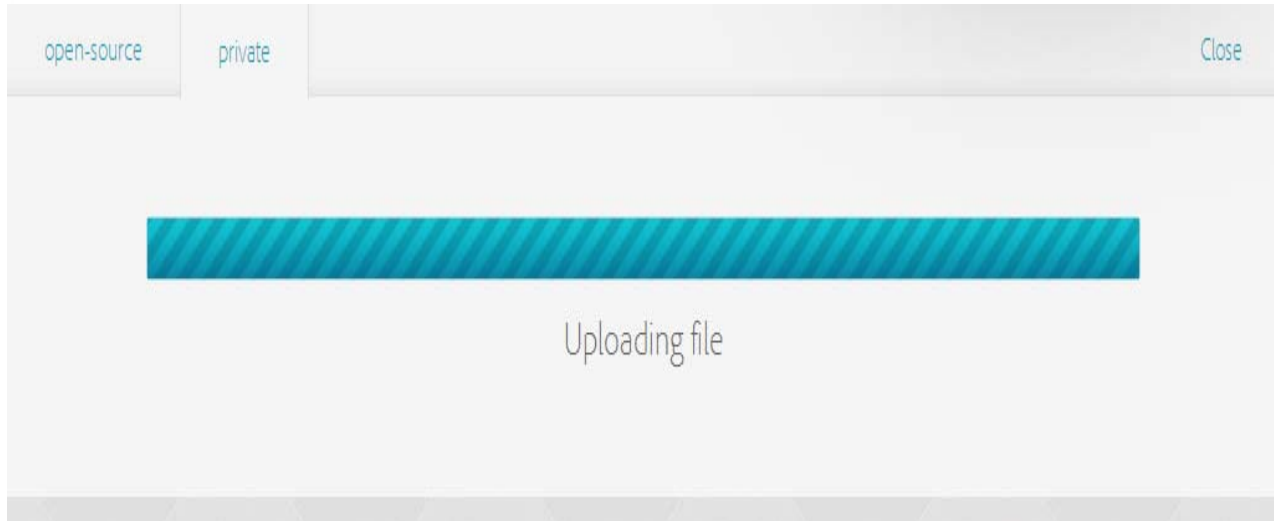


Figure 74 File uploading

The next step is to name our application

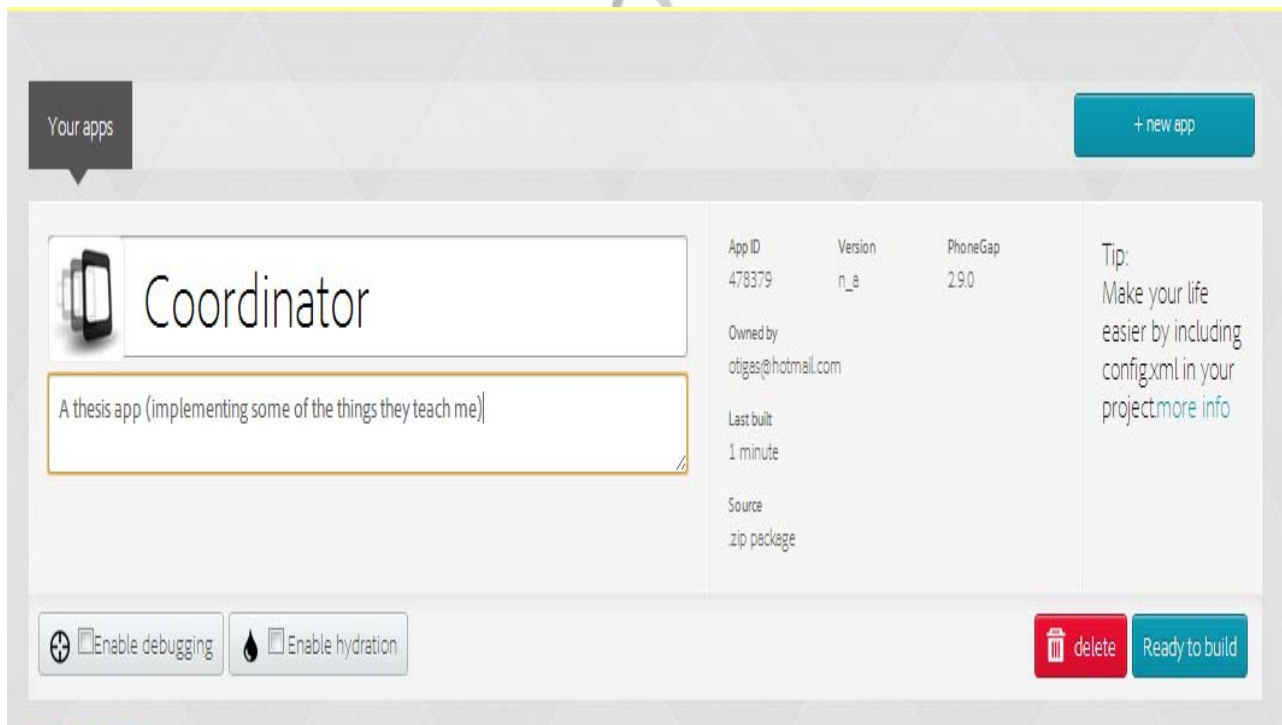


Figure 75 name options for our file

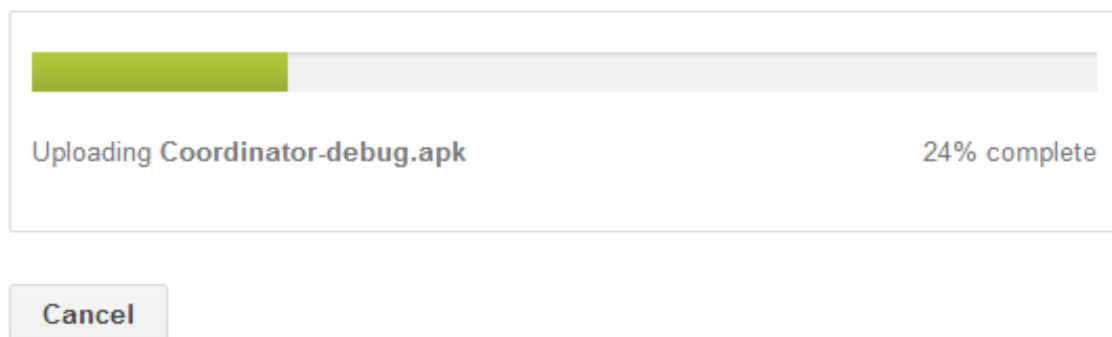
The next image shows the possibility of downloading our application in various formats.



**Figure 76 download our project in various formats**

Then we have to upload our file to our manufacturer market, here is android Google play example picture.

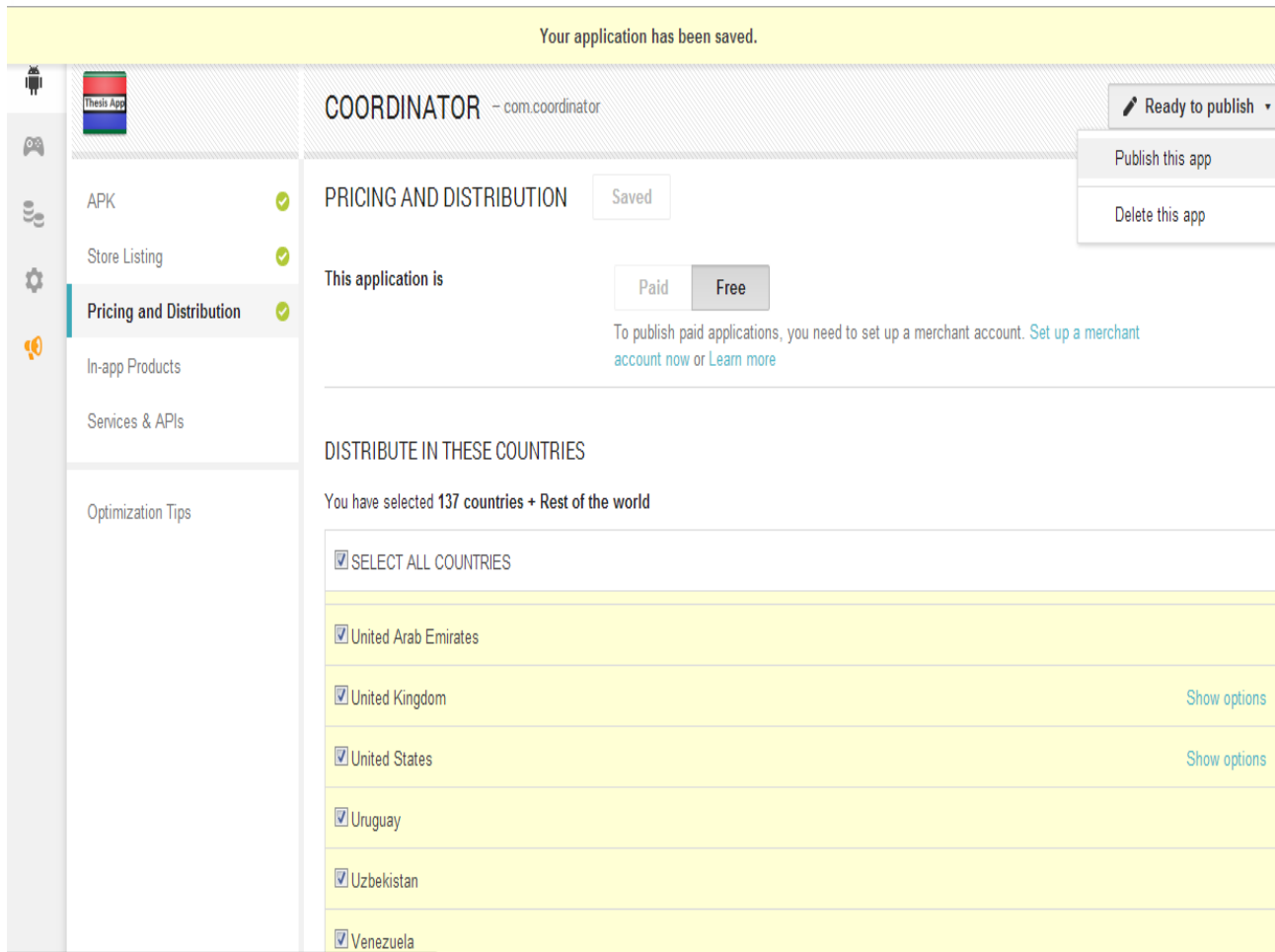
#### UPLOAD NEW APK TO PRODUCTION



**Figure 77 upload new apk to production**

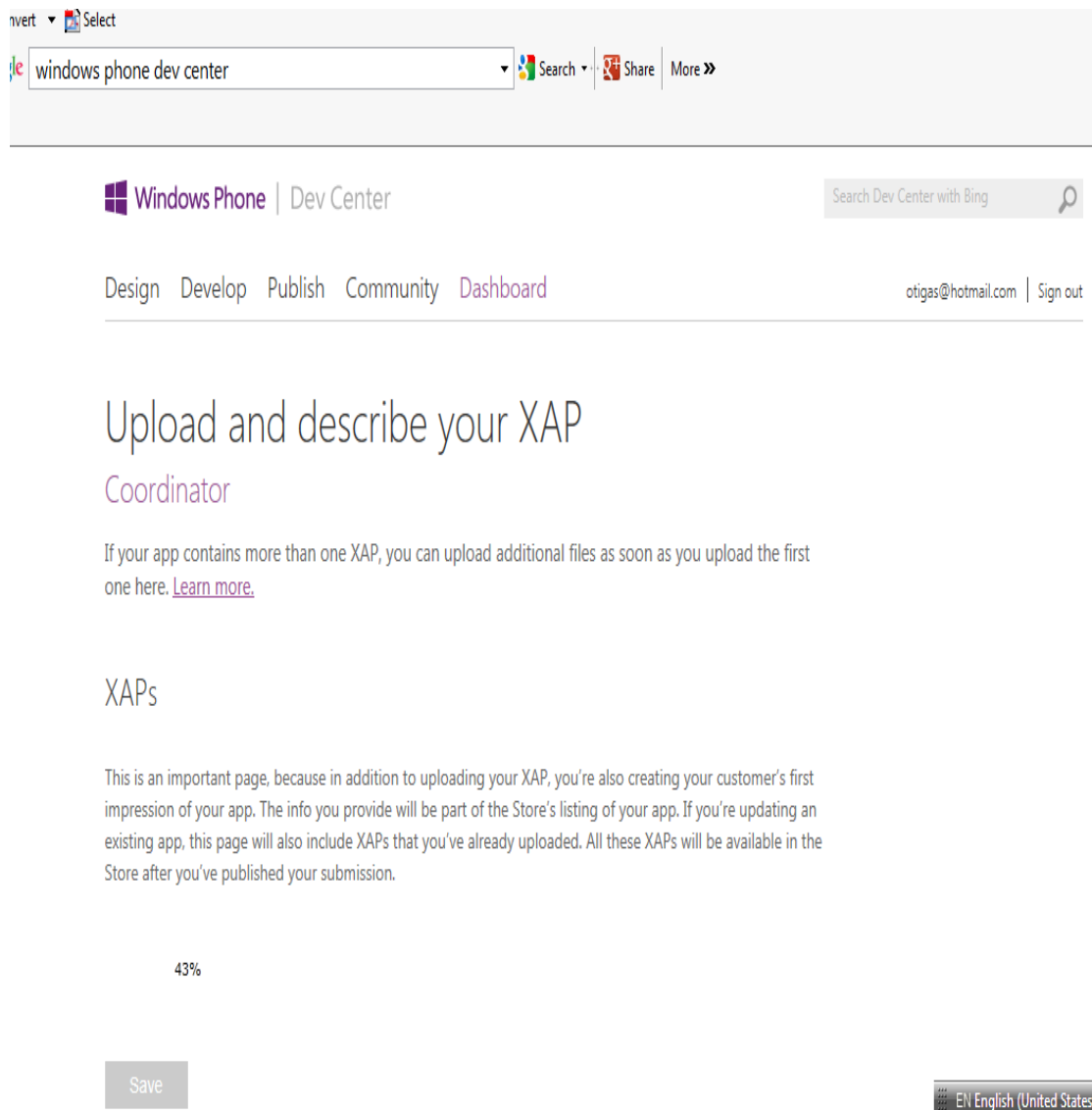


Next image shows the successful uploading of our application in Google play. The application is transformed into .apk format even if the structure is totally different in terms of approach.

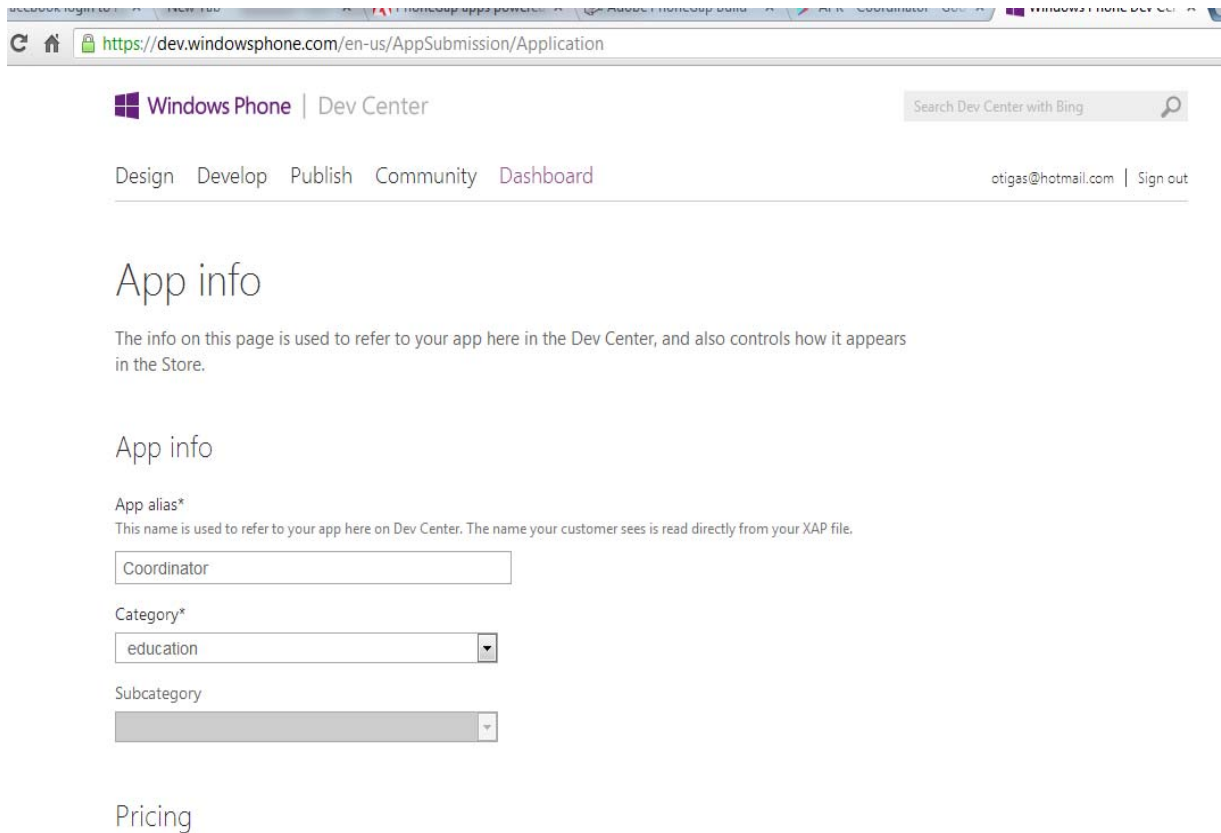


**Figure 78 successful attempt to upload on Google play**

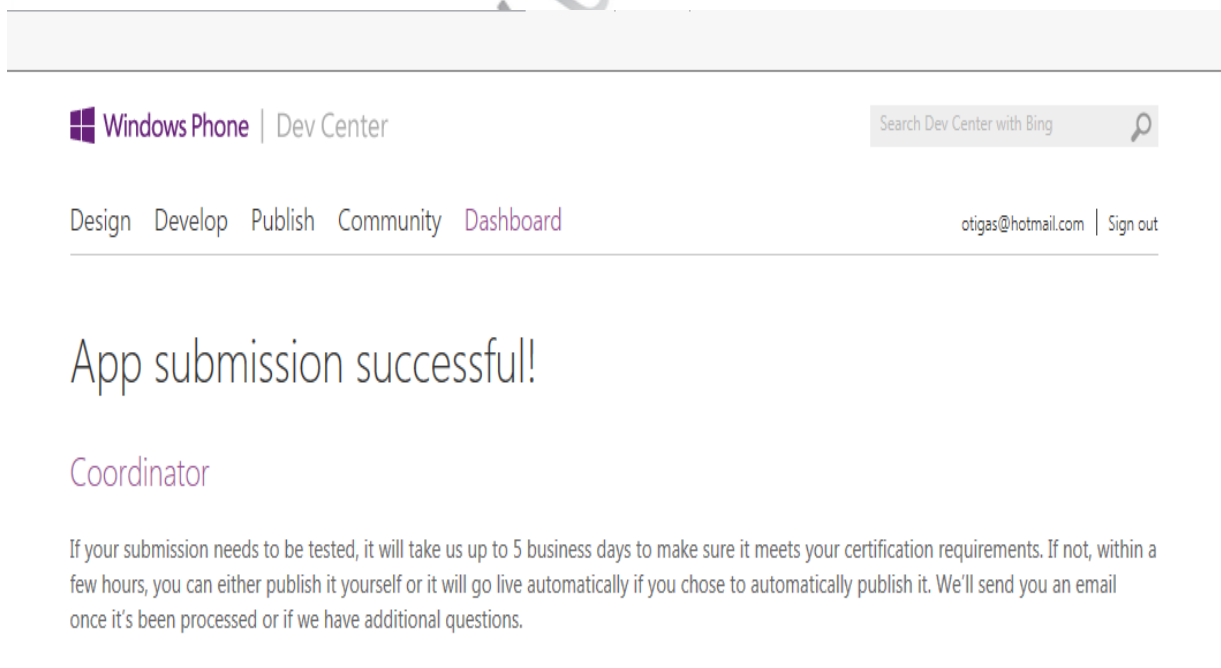
The next attempt has to do with the windows phone store.



**Figure 79 Upload our xap file.**



**Figure 80 App info**



**Figure 81 submission successful**

Then after the proper certifications and permissions our application can be downloaded from the specific manufacturer online market.

### **7.3 Limitations**

Unfortunately the plugins usage via the github and other repositories is a complicated way to add functionality to an application. The native way can be more specific with ready to use plugins without so many bugs and lost repositories. In our example the childbrowser plugin and the Facebook plugin where a big challenge for our knowledge and experience. One other issue is the different requirements of software companies acquired in order a cross platform application to pass the test certification before become available to the users in a specific market. We have noticed that if there are advanced monetizing possibilities of the application via a platform there is a difficulty to pass the test. There are no standards yet for a universal solution on cross platform applications and finally the performance of the cross platform way of developing applications is not equal with the native way in terms of speed and documentation. Even if the cost and the time of developing is reduced during our effort we believe that many improvements must take place before the cross platform way dominates the mobile world.

### **7.4 Future Work**

In general there is a lot of work must be done on the cross platform concept. The standardization is a big issue in our opinion. The plugins adoption is a tuff procedure in many cases, so the automation of some steps would be a good a good idea in the future. In most cases the Integrated Development Environments (IDE's) are following the strict rules of their native purpose when creating applications with the result to be very hard to adopt the web technologies (HTML, JS, CSS). The need of user friendly IDE's for cross platform developing is emerging in this case. One great idea for further research is the automated uploading the application generated files from phonegap to the different markets. Our application adopts some of the most useful functions developed with web technologies, but there is no a specific connection with a commercial application. The use of these functions in a commercial application or a game is a challenge for us. Finally the need for more interaction with the user is an interesting concept by adding push notifications or in-application purchases.

## References

- [1] Laouris, Y., & Eteokleous, N. (2005). We need an educationally relevant definition of mobile learning. *4th World Conference on Mobile Learning*, 2-3.
- [2] Rohit G, Yogesh P (2012) *PhoneGap Mobile web Framework for JavaScript and HTML5*. New York: Apress, 1-2.
- [3] Alan D, Barbara HW, David T (2010) *Systems Analysis and Design with UML. An Object-Oriented Approach-Third Edition*. New Jersey: Wiley, 14-15.
- [4] Beck Kent; et al. (2001). "Manifesto for Agile Software Development". Agile Alliance. Retrieved 14 June 2010.
- [5] John D (2011) *Software Development and Professional Practice*. Apress: New York, 15-16.
- [6] Chromatic (2003) *Extreme Programming Pocket Guide*. Sebastopol: O'Reilly, 59-64.
- [7] Beck, K (2000) *Extreme Programming Explained: Embrace Change*. Boston MA: Addison Wesley, 100- 104.
- [8] Jennifer G, Andrew S (2007) *A brain friendly Guide, Head first Project Management Professional*. Sebastopol: O'Reilly,1-6.
- [9] IBM Software Thought Leadership White Paper (2012) *Native Web or Hybrid mobile-app*. USA: IBM, 1-4.
- [10] Maximiliano F (2013) *Programming the Mobile Web, Second Edition*. Sebastopol: O'Reilly, 69-71.
- [11] John, A . (2013) *Appcelerator Titanium Up & Running*. Sebastopol: O'Reilly, 1-7.
- [12] Sarah A, Vidal G, Lee L (2010) *Pro Smartphone Cross-Platform Development: iPhone, Blackberry, Windows Mobile and Android Development and Distribution*. Apress: New York , 84-85.
- [13] Rodney,O (2007) *Fusibility studies made simple Australia* ,Boat Harbor: Martin Books Ltd. 6-7
- [14] Matson, James (2000) *Cooperative Feasibility Study Guide*, USA department of Agriculture , Rural Business-Cooperative Service: 1-2.
- [15] David, F (2011) *JavaScript: The Definitive Guide, Sixth Edition*, Sebastopol: O'Reilly, 1-2.
- [16] Brett, M (2011) *What Is HTML5? A New Way to Look at the Web*, Sebastopol: O'Reilly, 2-3.
- [17] Ian P, Richard Y (2011) *Beginning CSS: Cascading Style Sheets for Web Design*, Third Edition Indiana: Wiley Publishing: 25-26.
- [18] Tim, W (2013) *Learning JavaScript A Hands-On Guide to the Fundamentals of Modern JavaScript*, Boston: Addison-Wesley, 3-6.
- [19] Aaron, G (2011) *Adaptive Web Design Crafting Eich Experiences with Progressive Enchantment*, Tennessee: Easy Readers, 2-3.
- [20] Zeldman, J (2001) *Taking Your Talent to the Web: Making the Transition from Graphic Design to Web Design*. Indiana: New Riders, 448.
- [21] Kaplan Andreas M., Haenlein Michael, (2010), Users of the world, unite! The challenges and opportunities of social media, *Business Horizons*, Vol. 53, Issue 1 (page 61).

- [22] H. Kietzmann, Jan; Kristopher Hermkens (2011). "Social media? Get serious! Understanding the functional building blocks of social media". Business Horizons 54: 241–251.
- [23] Agichtein, Eugene; Carlos Castillo. Debora Donato, Aristides Gionis, Gilad Mishne (2008). "Finding high-quality content in social media". WSDM'08 - Proceedings of the 2008 International Conference on Web Search and Data Mining: 183–193.
- [24] Tang, Qian; Gu, Bin; Whinston, Andrew B. (2012). "Content Contribution for Revenue Sharing and Reputation in Social Media: A Dynamic Structural Model". Journal of Management Information Systems 29: 41–75.
- [25] Quercia, Daniele; Lathia, Neal; Calabrese, Francesco; Di Lorenzo, Giusy; Crowcroft, Jon (2010). "Recommending Social Events from Mobile Phone Location Data". 2010 IEEE International Conference on Data Mining. p. 971.
- [26] B. Guo, S. Satake, M. Imai. Home-Explorer: Ontology-based Physical Artifact Search and Hidden Object Detection System. Mobile Information Systems, Vol. 4 No.2 (2008), 81–103, IOS Press, 2008.
- [27] B. Guo, R. Fujimura, D. Zhang, M. Imai (2011). "Design-in-Play: Improving the Variability of Indoor Pervasive Games". Multimedia Tools and Applications.

### Web References

- <sup>1</sup> [www.microsoft.com](http://www.microsoft.com)
- <sup>2</sup> [www.apple.com](http://www.apple.com)
- <sup>3</sup> [www.java.com](http://www.java.com)
- <sup>4</sup> [www.w3.org](http://www.w3.org)
- <sup>5</sup> [http://en.wikipedia.org/wiki/Goldilocks\\_zone](http://en.wikipedia.org/wiki/Goldilocks_zone). It's the zone where a planet can be habitable depending the distance from a star.
- <sup>6</sup> <http://www.eclipse.org/>
- <sup>7</sup> <http://developer.apple.com/iphone>
- <sup>8</sup> <http://na.blackberry.com/eng/developers>.
- <sup>9</sup> <http://developer.android.com/sdk/index.html>
- <sup>10</sup> <http://techcrunch.com/2008/12/09/appcelerator-raises-41-million-for-open-source-ria-platform/>
- <sup>11</sup> [http://www.infoworld.com/d/developer-world/appcelerator-enables-iphone-android-app-dev- 655](http://www.infoworld.com/d/developer-world/appcelerator-enables-iphone-android-app-dev-655)
- <sup>12</sup> <http://mashable.com/2010/04/05/titanium-tablet-sdk/>

<sup>13</sup> <http://blackberryrocks.com/2010/05/06/appcelerator-announces-titanium-mobile-beta-support-blackberry-news/>

<sup>14</sup> <http://developer.appcelerator.com/blog/2010/12/titanium-guides-project-js-environment.html>

<sup>15</sup> <http://developer.appcelerator.com/blog/2011/09/platform-engineering-android-runtime-performance-improvements.html>

<sup>16</sup> <http://usingimho.wordpress.com/2011/06/14/why-you-should-stay-away-from-appcelerators-titanium/>

<sup>17</sup> <http://pasamio.com/2011/07/02/a-few-months-with-titanium-appcelerator/>

<sup>18</sup> <http://www.nbc.com/news/2011/06/23/nbc-launches-new-nbccom-app-for-ipad-now-available-on-the-app-store/>

<sup>19</sup> <http://www.slideshare.net/aaronksaunders/why-appcelerator>

<sup>20</sup> <http://readwrite.com/2011/06/13/appcelerator-launches-titanium-studio-mobile-desktop-web-development-in-one>

<sup>21</sup> <http://www.appcelerator.com/products/cloud-services/>

<sup>22</sup> <http://techcrunch.com/2011/01/18/appcelerator-acquires-web-app-development-suite-aptana/>

<sup>23</sup> <http://www.appcelerator.com>

<sup>24</sup> <http://phonegap.com/about/license/>

<sup>25</sup> <http://html.adobe.com/edge/phonegap-build/faq.html>

<sup>26</sup> <http://www.adobe.com/aboutadobe/pressroom/pressreleases/201110/AdobeAcquiresNitobi.html>

<sup>27</sup> <http://www.quora.com/Andre-Charland/PhoneGap/answers>

<sup>28</sup> <http://gigaom.com/2009/04/05/phonegap-seeks-to-bridge-the-gap-between-mobile-app-platforms/>

<sup>29</sup> <http://www.h-online.com/open/news/item/Apache-Cordova-gets-a-new-look-1440114.html>



- <sup>30</sup> <http://phonegap.com/2010/04/14/phonegap-and-the-apple-developer-license-agreement/>
- <sup>31</sup> <http://www.viziapps.com/>
- <sup>32</sup> <http://www-03.ibm.com/software/products/us/en/worklight>
- <sup>33</sup> <http://phonegap.com/2011/06/27/how-phonegap-plays-an-important-part-in-our-enterprise-offering/>
- <sup>34</sup> <http://www.convertigo.com/>
- <sup>35</sup> <http://phonegap.com/2011/07/07/convertigo-mobilizer-uses-phonegap-build-apis/>
- <sup>36</sup> <http://www.appmobi.com/>
- <sup>37</sup> [http://news.cnet.com/8301-30685\\_3-20114857-264/adobe-buys-phonegap-typekit-for-better-web-tools/](http://news.cnet.com/8301-30685_3-20114857-264/adobe-buys-phonegap-typekit-for-better-web-tools/)
- <sup>38</sup> <http://markmail.org/message/vcrw2swiwiwcojsd>
- <sup>39</sup> <https://build.phonegap.com/>
- <sup>40</sup> <http://techcrunch.com/2012/09/24/adobe-launches-hosted-phonegap-build-service-for-creating-cross-platform-mobile-apps/>
- <sup>41</sup> <http://software.intel.com/en-us/articles/the-development-of-mobile-applications-using-html5-and-phonegap-on-intel-architecture-based>. *"However, HTML5 has some limitations. Most prominent, is the lack of API to access device hardware and sensors such as accelerometer, compass, GPS, etc. While native applications can access device hardware, they lack the portability that web apps provide. Thus, a solution is to code a hybrid application, which cumulatively uses the benefits of native and web apps."*
- <sup>42</sup> <http://www.sapandiwakar.in/technical/api-research-study-iphone-and-android-applications/>
- <sup>43</sup> <http://www.adobe.com/devnet/phonegap/articles/apple-application-rejections-and-phonegap-advice.html>
- <sup>44</sup> [http://en.wikipedia.org/wiki/Adobe\\_Systems](http://en.wikipedia.org/wiki/Adobe_Systems)
- <sup>45</sup> <https://github.com/mrlacey/phonegap-wp7>
- <sup>46</sup> <http://www.motorolasolutions.com/US-EN/Business+Product+and+Services/Software+and+Applications/RhoMobile+Suite>

- <sup>47</sup> <http://devproconnections.com/development/product-review-rhomobiles-rhombus-smartphone-app-development-product-suite>
- <sup>48</sup> <http://www.gizmapps.blogspot.gr/2013/04/hello-mvc-asp.html>
- <sup>49</sup> <http://www.sencha.com/blog/ext-js-jqtouch-raphael-sencha>
- <sup>50</sup> <http://notes.sencha.com/post/709462805/intro-to-sencha-touch>
- <sup>51</sup> [http://www.quickconnectfamily.org/qc\\_hybrid/](http://www.quickconnectfamily.org/qc_hybrid/)
- <sup>52</sup> <http://www.softwareag.com/corporate/default.asp>
- <sup>53</sup> <http://anscamobile.com/corona/>.
- <sup>54</sup> <http://www.mosync.com>
- <sup>55</sup> <http://unity3d.com/>.
- <sup>56</sup> <http://iWebkit.net>
- <sup>57</sup> <http://WebApp.net>
- <sup>58</sup> <http://www.dojotoolkit.org>
- <sup>59</sup> <http://widgetpad.ndl.cc/>
- <sup>60</sup> <http://www.mosync.com/>
- <sup>61</sup> [http://en.wikipedia.org/wiki/Mobile\\_application\\_development](http://en.wikipedia.org/wiki/Mobile_application_development)
- <sup>62</sup> [http://en.wikipedia.org/wiki/Multiple\\_phone\\_web\\_based\\_application\\_framework](http://en.wikipedia.org/wiki/Multiple_phone_web_based_application_framework)
- <sup>63</sup> <http://www.gimp.org/>
- <sup>64</sup> <http://www.ecmascript.org/es4/spec/overview.pdf>
- <sup>65</sup> <http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-262.pdf>
- <sup>66</sup> <http://web.archive.org/web/20070916144913/http://wp.netscape.com/newsref/pr/newsrelease67.html>
- <sup>67</sup> <http://www.w3.org/TR/2011/WD-html5-diff-20110405/>
- <sup>68</sup> [http://www.w3.org/QA/2012/12/html5\\_smile\\_its\\_a\\_snapshot.html](http://www.w3.org/QA/2012/12/html5_smile_its_a_snapshot.html)

<sup>69</sup> <http://validator.w3.org/>

<sup>70</sup> <http://www.w3.org/TR/html5-diff/>

<sup>71</sup> <http://www.w3.org/html/wg/drafts/html/master/introduction.html#syntax-errors>

<sup>72</sup> [http://www.w3schools.com/html/tryit.asp?filename=tryhtml5\\_video\\_bear](http://www.w3schools.com/html/tryit.asp?filename=tryhtml5_video_bear)

<sup>73</sup> <http://www.w3.org/standards/webdesign/htmlcss#whatcss>

<sup>74</sup> <http://jigsaw.w3.org/css-validator/>

<sup>75</sup> <http://www.w3schools.com/css/default.asp>

<sup>76</sup> <http://www.sqlite.org/copyright.html>

<sup>77</sup> <http://www.adobe.com/devnet/phonegap/articles/phonegap-apps-powered-by-developercom.html>

<sup>78</sup> <https://github.com/phonegap/phonegap-plugins/tree/master/Android/ChildBrowser>

## Annex

In this section we quote the code of our project.

The code for the simple calculator project (Native application of chapter four). The first table shows the MainActivity.java where all the code is stored.

```
package com.example.tigalc;

import android.app.Activity;
import android.os.Bundle;

import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import android.widget.EditText;

//h klasi mou pou klironomite apo tin Activity kai h opia efarmozi tin Onclick
```

```
//listener

public class MainActivity extends Activity implements OnClickListener {

    // setaro ta stihia mou os metavlites stin java shedon panta

    private Button b1;

    private Button b3;

    private Button b5;

    private Button b7;

    EditText ss1;

    EditText ss2;

    TextView shw;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        // TODO Auto-generated method stub
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // Setaro tis anafores sta koumpia kai sto stihia

        b1 = (Button)findViewById(R.id.btnsum);

        b3 = (Button)findViewById(R.id.btnabs);

        b5 = (Button)findViewById(R.id.btnmul);

        b7 = (Button)findViewById(R.id.btndiv);

        ss1 = (EditText)findViewById(R.id.txtbox1);

        ss2 = (EditText)findViewById(R.id.txtbox2);

        shw = (TextView)findViewById(R.id.lbl1);

        // click listeners gia afti tin klasi (this)
```

```
b1.setOnClickListener(this);
b3.setOnClickListener(this);
b5.setOnClickListener(this);
b7.setOnClickListener(this);

}

@Override
// mesa stin methodo onclick tha kano tis sigrasis mou
// shetika me ta parapano koumpia pou eftiata tous listeners

public void onClick(View v) {
    // TODO Auto-generated method stub

    if (v==b1) {

        // se afta ta block kodika vriskete h litourgikotita
        // vevea den ine toso sosto na dilonis metavlites eki mesa

        String a,b;
        Double vis;
        a = ss1.getText().toString();
        b = ss2.getText().toString();
        vis = Double.parseDouble(a) + Double.parseDouble(b);
        shw.setText(vis.toString());

    }

    if (v==b3) {

        String a,b;
        Double vis;
        a = ss1.getText().toString();
        b = ss2.getText().toString();
        vis = Double.parseDouble(a) - Double.parseDouble(b);
        shw.setText(vis.toString());

    }

}
```

```
        if (v==b5) {  
            String a,b;  
            Double vis;  
            a = ss1.getText().toString();  
            b = ss2.getText().toString();  
            vis = Double.parseDouble(a) * Double.parseDouble(b);  
            shw.setText(vis.toString());  
        }  
  
        if (v==b7) {  
            String a,b;  
            Double vis;  
            a = ss1.getText().toString();  
            b = ss2.getText().toString();  
  
            if(b.equals("0")){  
                vis = Double.parseDouble(a) / Double.parseDouble(b);  
                shw.setText("-99999 error");  
                Toast toast=Toast.makeText(this, "diereses me to  
miden", Toast.LENGTH_LONG);  
                toast.show();  
            }  
            else {  
                vis = Double.parseDouble(a) / Double.parseDouble(b);  
                shw.setText(vis.toString());  
            }  
        }  
    }  
}
```

**TABLE 1 MainActivity.java.**

The next table shows the xml layout named activity\_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<AbsoluteLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/txt1"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="@drawable/kentriki" >

<EditText
    android:id="@+id/txtbox1"
    android:layout_width="220dp"
    android:layout_height="50dp"
    android:layout_x="51dp"
    android:layout_y="75dp"
    android:ems="10"
    android:inputType="number"
    android:textSize="18sp" >

    <requestFocus />
</EditText>

<TextView
    android:id="@+id/textView1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_x="120dp"
    android:layout_y="46dp"
    android:text="number1"
    android:textAppearance="?android:attr/textAppearanceMedium" />

<Button
    android:id="@+id/btndiv"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_x="218dp"
    android:layout_y="325dp"
    android:text="/" />

<Button
    android:id="@+id/btnabs"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_x="40dp"
    android:layout_y="322dp"
    android:text="-" />

<Button
    android:id="@+id/btnsum"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_x="39dp"
    android:layout_y="275dp"
```



```
        android:text="+" />
<Button
    android:id="@+id/btnmul"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_x="217dp"
    android:layout_y="275dp"
    android:text="*" />
<TextView
    android:id="@+id/lbl1"
    android:layout_width="141px"
    android:layout_height="wrap_content"
    android:layout_x="138dp"
    android:layout_y="249dp"
    android:text="-----" />
<TextView
    android:id="@+id/textView3"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_x="138dp"
    android:layout_y="220dp"
    android:text="result"
    android:textAppearance="?android:attr/textAppearanceMedium" />
<EditText
    android:id="@+id/txtbox2"
    android:layout_width="216dp"
    android:layout_height="56dp"
    android:layout_x="54dp"
    android:layout_y="163dp"
    android:ems="10"
    android:inputType="number"
    android:textSize="18sp" />
<TextView
    android:id="@+id/textView2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_x="120dp"
    android:layout_y="154dp"
    android:text="Number2"
    android:textAppearance="?android:attr/textAppearanceMedium" />
</AbsoluteLayout>
```

## TABLE 2 activity\_main.xml

### The next three tables referring to HelloGap Application

```
<!DOCTYPE HTML>
```

```
<html>
  <link rel="stylesheet" href="master.css" type="text/css" media="screen" title="no
  title" charset="utf-8">
  <head>
    <meta name="viewport" content="width=320; user-scalable=no" />
    <meta http-equiv="Content-type" content="text/html; charset=utf-8">
    <title>PhoneGap</title>
    <script type="text/javascript" charset="utf-8"
src="cordova.js"></script>
    <script type="text/javascript" charset="utf-8">
      var displayHello = function() {
        var name = document.getElementById("firstname").value;
        navigator.notification.alert("name" + name);
      }
    </script>
  </head>
  <body onload="init();" id="bdy" >
    <div id="txt">
      <input type="text" name="firstname" id="firstname" />
    </div>
    <div id="btn">
      <a href="#" class="btn" onclick="displayHello();">Say Hello</a>
    </div>
  </div>
</body>
</html>
```

**TABLE 3** `index.html`

```
#bdy
{
    background:#F0F0F0;
}
#btn a{
    border: 1px solid #555;
    -webkit-border-radius: 5px;
    border-radius: 5px;
    text-align:center;
    display:block;
    float:left;
    background:#6600CC;
    width:308px;
    color:#FFF;
    font-size:1.1em;
    text-decoration:none;
    padding:1.2em 0;
    margin:3px 0px 3px 5px;
}
#txt{
    border: 1px solid #555;
    -webkit-border-radius: 5px;
```

```
border-radius: 5px;
text-align:center;
display:block;
float:left;
background:#00FFCC;
width:308px;
color:#9ab;
font-size:1.1em;
text-decoration:none;
padding:1.2em 0;
margin:3px 0px 3px 5px;
```

**Table 4 master.css**

```
package com.example.hellogap;

import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import org.apache.cordova.DroidGap;

public class MainActivity extends DroidGap {

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        super.loadUrl("file:///android_asset/www/index.html");
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }

}
```

**Table 5 MainActivity.java**

```
<!DOCTYPE html >
<html lang="en">
<head>

<meta http-equiv="content-type" content="text/html; charset=UTF-8" />
<meta name="viewport" content="width=device-width; initial-scale=1.0; maximum-
scale=1.0; user-scalable=no" />
<meta name="format-detection" content="telephone=yes" />
```

```
<title>Test App</title>

<script type="text/javascript" charset="utf-8" src="cordova.js"></script>
<script type="text/javascript" charset="utf-8" src="childbrowser.js"></script>
<script language="text/javascript" type="text/javascript">
/**
 * Called when initial web page is loaded
 */
function onBodyLoad()
{
console.log("onBodyLoad()");

// Listen for PhoneGap framework to finish init
document.addEventListener("deviceready",onDeviceReady,false);
}

/**
 * When this function is called, PhoneGap has been initialized and is ready to roll
 */
function onDeviceReady() {
console.log("onDeviceReady()");
}
</script>

</head>
<body onload="onBodyLoad();">

<a href="#"
onclick="window.plugins.childBrowser.showWebPage('http://www.google.com');">Open
Google.com in child browser</a>

</body>
</html>
```

#### Table 6 Works.js

```
package com.phonegap.plugins.childBrowser;

/**
 * PhoneGap is available under *either* the terms of the modified BSD license *or* the
 * MIT License (2008). See http://opensource.org/licenses/alphabetical for full text.
 *
 * Copyright (c) 2005-2011, Nitobi Software Inc.
 * Copyright (c) 2010-2011, IBM Corporation
 */
```

```
import java.io.IOException;
import java.io.InputStream;

import org.apache.cordova.api.Plugin;
import org.apache.cordova.api.PluginResult;
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import android.app.Dialog;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.net.Uri;
import android.text.InputType;
import android.util.Log;
import android.util.TypedValue;
import android.view.Gravity;
import android.view.KeyEvent;
import android.view.View;
import android.view.Window;
import android.view.WindowManager;
import android.view.WindowManager.LayoutParams;
import android.view.inputmethod.EditorInfo;
import android.view.inputmethod.InputMethodManager;
import android.webkit.WebChromeClient;
import android.webkit.WebSettings;
import android.webkit.WebView;
import android.webkit.WebViewClient;
import android.widget.EditText;
import android.widget.ImageButton;
import android.widget.LinearLayout;
import android.widget.RelativeLayout;

public class ChildBrowser extends Plugin {

    protected static final String LOG_TAG = "ChildBrowser";
    private static int CLOSE_EVENT = 0;
    private static int LOCATION_CHANGED_EVENT = 1;

    private String browserCallbackId = null;

    private Dialog dialog;
```

```
private WebView webview;
private EditText edittext;
private boolean showLocationBar = true;

/**
 * Executes the request and returns PluginResult.
 *
 * @param action The action to execute.
 * @param args JSONArray of arguments for the plugin.
 * @param callbackId The callback id used when calling back into JavaScript.
 * @return A PluginResult object with a status and message.
 */
public PluginResult execute(String action, JSONArray args, String callbackId) {
    PluginResult.Status status = PluginResult.Status.OK;
    String result = "";

    try {
        if (action.equals("showWebPage")) {
            this.browserCallbackId = callbackId;

            // If the ChildBrowser is already open then throw an error
            if (dialog != null && dialog.isShowing()) {
                return new PluginResult(PluginResult.Status.ERROR, "ChildBrowser is
already open");
            }

            result = this.showWebPage(args.getString(0), args.optJSONObject(1));

            if (result.length() > 0) {
                status = PluginResult.Status.ERROR;
                return new PluginResult(status, result);
            } else {
                PluginResult pluginResult = new PluginResult(status, result);
                pluginResult.setKeepCallback(true);
                return pluginResult;
            }
        }
        else if (action.equals("close")) {
            closeDialog();

            JSONObject obj = new JSONObject();
            obj.put("type", CLOSE_EVENT);

            PluginResult pluginResult = new PluginResult(status, obj);
            pluginResult.setKeepCallback(false);
            return pluginResult;
        }
    }
}
```

```
    }
    else if (action.equals("openExternal")) {
        result = this.openExternal(args.getString(0), args.optBoolean(1));
        if (result.length() > 0) {
            status = PluginResult.Status.ERROR;
        }
    }
    else {
        status = PluginResult.Status.INVALID_ACTION;
    }
    return new PluginResult(status, result);
} catch (JSONException e) {
    return new PluginResult(PluginResult.Status.JSON_EXCEPTION);
}
}

/**
 * Display a new browser with the specified URL.
 *
 * @param url The url to load.
 * @param usePhoneGap Load url in PhoneGap webview
 * @return "" if ok, or error message.
 */
public String openExternal(String url, boolean usePhoneGap) {
    try {
        Intent intent = null;
        if (usePhoneGap) {
            intent = new Intent().setClass(this.cordova.getActivity(),
org.apache.cordova.DroidGap.class);
            intent.setData(Uri.parse(url)); // This line will be removed in future.
            intent.putExtra("url", url);

            // Timeout parameter: 60 sec max - May be less if http device timeout is less.
            intent.putExtra("loadUrlTimeoutValue", 60000);

            // These parameters can be configured if you want to show the loading dialog
            intent.putExtra("loadingDialog", "Wait,Loading web page..."); // show loading
dialog
            intent.putExtra("hideLoadingDialogOnPageLoad", true); // hide it once page
has completely loaded
        }
        else {
            intent = new Intent(Intent.ACTION_VIEW);
            intent.setData(Uri.parse(url));
        }
        this.cordova.getActivity().startActivity(intent);
    }
}
```



```
        return "";
    } catch (android.content.ActivityNotFoundException e) {
        Log.d(LOG_TAG, "ChildBrowser: Error loading url "+url+" "+ e.toString());
        return e.toString();
    }
}

/**
 * Closes the dialog
 */
private void closeDialog() {
    if (dialog != null) {
        dialog.dismiss();
    }
}

/**
 * Checks to see if it is possible to go back one page in history, then does so.
 */
private void goBack() {
    if (this.webview.canGoBack()) {
        this.webview.goBack();
    }
}

/**
 * Checks to see if it is possible to go forward one page in history, then does so.
 */
private void goForward() {
    if (this.webview.canGoForward()) {
        this.webview.goForward();
    }
}

/**
 * Navigate to the new page
 *
 * @param url to load
 */
private void navigate(String url) {
    InputMethodManager imm =
(InputMethodManager)this.cordova.getActivity().getSystemService(Context.INPUT_ME
THOD_SERVICE);
    imm.hideSoftInputFromWindow(edittext.getWindowToken(), 0);

    if (!url.startsWith("http") && !url.startsWith("file:")) {
```

```
        this.webview.loadUrl("http://" + url);
    } else {
        this.webview.loadUrl(url);
    }
    this.webview.requestFocus();
}

/**
 * Should we show the location bar?
 *
 * @return boolean
 */
private boolean getShowLocationBar() {
    return this.showLocationBar;
}

/**
 * Display a new browser with the specified URL.
 *
 * @param url The url to load.
 * @param jsonObject
 */
public String showWebPage(final String url, JSONObject options) {
    // Determine if we should hide the location bar.
    if (options != null) {
        showLocationBar = options.optBoolean("showLocationBar", true);
    }

    // Create dialog in new thread
    Runnable runnable = new Runnable() {
        /**
         * Convert our DIP units to Pixels
         *
         * @return int
         */
        private int dpToPixels(int dipValue) {
            int value = (int) TypedValue.applyDimension(
                TypedValue.COMPLEX_UNIT_DIP,
                (float) dipValue,
                cordova.getActivity().getResources().getDisplayMetrics()
            );

            return value;
        }
    }
}
```

```
public void run() {
    // Let's create the main dialog
    dialog = new Dialog(cordova.getActivity(), android.R.style.Theme_NoTitleBar);
    dialog.getWindow().getAttributes().windowAnimations =
android.R.style.Animation_Dialog;
    dialog.requestWindowFeature(Window.FEATURE_NO_TITLE);
    dialog.setCancelable(true);
    dialog.setOnDismissListener(new DialogInterface.OnDismissListener() {
        public void onDismiss(DialogInterface dialog) {
            try {
                JSONObject obj = new JSONObject();
                obj.put("type", CLOSE_EVENT);

                sendUpdate(obj, false);
            } catch (JSONException e) {
                Log.d(LOG_TAG, "Should never happen");
            }
        }
    });

    // Main container layout
    LinearLayout main = new LinearLayout(cordova.getActivity());
    main.setOrientation(LinearLayout.VERTICAL);

    // Toolbar layout
    RelativeLayout toolbar = new RelativeLayout(cordova.getActivity());
    toolbar.setLayoutParams(new
RelativeLayout.LayoutParams(LayoutParams.FILL_PARENT, this.dpToPixels(44)));
    toolbar.setPadding(this.dpToPixels(2), this.dpToPixels(2), this.dpToPixels(2),
this.dpToPixels(2));
    toolbar.setHorizontalGravity(Gravity.LEFT);
    toolbar.setVerticalGravity(Gravity.TOP);

    // Action Button Container layout
    RelativeLayout actionButtonContainer = new
RelativeLayout(cordova.getActivity());
    actionButtonContainer.setLayoutParams(new
RelativeLayout.LayoutParams(LayoutParams.WRAP_CONTENT,
LayoutParams.WRAP_CONTENT));
    actionButtonContainer.setHorizontalGravity(Gravity.LEFT);
    actionButtonContainer.setVerticalGravity(Gravity.CENTER_VERTICAL);
    actionButtonContainer.setId(1);

    // Back button
    ImageButton back = new ImageButton(cordova.getActivity());
```

```
        RelativeLayout.LayoutParams backLayoutParams = new
RelativeLayout.LayoutParams(LayoutParams.WRAP_CONTENT,
LayoutParams.FILL_PARENT);
        backLayoutParams.addRule(RelativeLayout.ALIGN_LEFT);
        back.setLayoutParams(backLayoutParams);
        back.setContentDescription("Back Button");
        back.setId(2);
        try {

back.setImageBitmap(loadDrawable("www/childbrowser/icon_arrow_left.png"));
        } catch (IOException e) {
            Log.e(LOG_TAG, e.getMessage(), e);
        }
        back.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                goBack();
            }
        });

        // Forward button
        ImageButton forward = new ImageButton(cordova.getActivity());
        RelativeLayout.LayoutParams forwardLayoutParams = new
RelativeLayout.LayoutParams(LayoutParams.WRAP_CONTENT,
LayoutParams.FILL_PARENT);
        forwardLayoutParams.addRule(RelativeLayout.RIGHT_OF, 2);
        forward.setLayoutParams(forwardLayoutParams);
        forward.setContentDescription("Forward Button");
        forward.setId(3);
        try {

forward.setImageBitmap(loadDrawable("www/childbrowser/icon_arrow_right.png"));
        } catch (IOException e) {
            Log.e(LOG_TAG, e.getMessage(), e);
        }
        forward.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                goForward();
            }
        });

        // Edit Text Box
        editText = new EditText(cordova.getActivity());
        RelativeLayout.LayoutParams textLayoutParams = new
RelativeLayout.LayoutParams(LayoutParams.FILL_PARENT,
LayoutParams.FILL_PARENT);
        textLayoutParams.addRule(RelativeLayout.RIGHT_OF, 1);
```

```
textLayoutParams.addRule(RelativeLayout.LEFT_OF, 5);
edittext.setLayoutParams(textLayoutParams);
edittext.setId(4);
edittext.setSingleLine(true);
edittext.setText(url);
edittext.setInputType(InputType.TYPE_TEXT_VARIATION_URI);
edittext.setImeOptions(EditorInfo.IME_ACTION_GO);
edittext.setInputType(InputType.TYPE_NULL); // Will not except input...
Makes the text NON-EDITABLE
edittext.setOnKeyListener(new View.OnKeyListener() {
    public boolean onKey(View v, int keyCode, KeyEvent event) {
        // If the event is a key-down event on the "enter" button
        if ((event.getAction() == KeyEvent.ACTION_DOWN) && (keyCode ==
KeyEvent.KEYCODE_ENTER)) {
            navigate(edittext.getText().toString());
            return true;
        }
        return false;
    }
});

// Close button
ImageButton close = new ImageButton(cordova.getActivity());
RelativeLayout.LayoutParams closeLayoutParams = new
RelativeLayout.LayoutParams(LayoutParams.WRAP_CONTENT,
LayoutParams.FILL_PARENT);
closeLayoutParams.addRule(RelativeLayout.ALIGN_PARENT_RIGHT);
close.setLayoutParams(closeLayoutParams);
forward.setContentDescription("Close Button");
close.setId(5);
try {
    close.setImageBitmap(loadDrawable("www/childbrowser/icon_close.png"));
} catch (IOException e) {
    Log.e(LOG_TAG, e.getMessage(), e);
}
close.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        closeDialog();
    }
});

// WebView
webview = new WebView(cordova.getActivity());
webview.setLayoutParams(new
LinearLayout.LayoutParams(LayoutParams.FILL_PARENT,
LayoutParams.FILL_PARENT));
```

```
webView.setWebChromeClient(new WebChromeClient());
WebViewClient client = new ChildBrowserClient(edittext);
webView.setWebViewClient(client);
WebSettings settings = webView.getSettings();
settings.setJavaScriptEnabled(true);
settings.setJavaScriptCanOpenWindowsAutomatically(true);
settings.setBuiltInZoomControls(true);
settings.setPluginsEnabled(true);
settings.setDomStorageEnabled(true);
webView.loadUrl(url);
webView.setId(6);
webView.getSettings().setLoadWithOverviewMode(true);
webView.getSettings().setUseWideViewPort(true);
webView.requestFocus();
webView.requestFocusFromTouch();

// Add the back and forward buttons to our action button container layout
actionButtonContainer.addView(back);
actionButtonContainer.addView(forward);

// Add the views to our toolbar
toolbar.addView(actionButtonContainer);
toolbar.addView(edittext);
toolbar.addView(close);

// Don't add the toolbar if its been disabled
if (getShowLocationBar()) {
    // Add our toolbar to our main view/layout
    main.addView(toolbar);
}

// Add our webview to our main view/layout
main.addView(webview);

WindowManager.LayoutParams lp = new WindowManager.LayoutParams();
lp.copyFrom(dialog.getWindow().getAttributes());
lp.width = WindowManager.LayoutParams.FILL_PARENT;
lp.height = WindowManager.LayoutParams.FILL_PARENT;

dialog.setContentView(main);
dialog.show();
dialog.getWindow().setAttributes(lp);
}

private Bitmap loadDrawable(String filename) throws java.io.IOException {
    InputStream input = cordova.getActivity().getAssets().open(filename);
```

```
        return BitmapFactory.decodeStream(input);
    }
};
this.cordova.getActivity().runOnUiThread(runnable);
return "";
}

/**
 * Create a new plugin result and send it back to JavaScript
 *
 * @param obj a JSONObject contain event payload information
 */
private void sendUpdate(JSONObject obj, boolean keepCallback) {
    if (this.browserCallbackId != null) {
        PluginResult result = new PluginResult(PluginResult.Status.OK, obj);
        result.setKeepCallback(keepCallback);
        this.success(result, this.browserCallbackId);
    }
}

/**
 * The webview client receives notifications about appView
 */
public class ChildBrowserClient extends WebViewClient {
    EditText edittext;

    /**
     * Constructor.
     *
     * @param mContext
     * @param edittext
     */
    public ChildBrowserClient(EditText mEditText) {
        this.edittext = mEditText;
    }

    /**
     * Notify the host application that a page has started loading.
     *
     * @param view The webview initiating the callback.
     * @param url The url of the page.
     */
    @Override
    public void onPageStarted(WebView view, String url, Bitmap favicon) {
        super.onPageStarted(view, url, favicon);
        String newloc;
    }
}
```

```
if (url.startsWith("http:") || url.startsWith("https:") || url.startsWith("file:")) {
    newloc = url;
} else {
    newloc = "http://" + url;
}

if (!newloc.equals(edittext.getText().toString())) {
    edittext.setText(newloc);
}

try {
    JSONObject obj = new JSONObject();
    obj.put("type", LOCATION_CHANGED_EVENT);
    obj.put("location", url);

    sendUpdate(obj, true);
} catch (JSONException e) {
    Log.d("ChildBrowser", "This should never happen");
}
}
}
```

**Table 7 Childbrowser.java**

```
/*
 * cordova is available under *either* the terms of the modified BSD license *or* the
 * MIT License (2008). See http://opensource.org/licenses/alphabetical for full text.
 *
 * Copyright (c) 2005-2010, Nitobi Software Inc.
 * Copyright (c) 2011, IBM Corporation
 */

/**
 * Constructor
 */
function ChildBrowser() {
};

ChildBrowser.CLOSE_EVENT = 0;
ChildBrowser.LOCATION_CHANGED_EVENT = 1;

/**
 * Display a new browser with the specified URL.
 * This method loads up a new web view in a dialog.
 *
 * @param url The url to load
 */
```



```
* @param options    An object that specifies additional options
*/
ChildBrowser.prototype.showWebPage = function(url, options) {
  if (options === null || options === "undefined") {
    var options = new Object();
    options.showLocationBar = true;
    options.locationBarAlign = "top";
  }
  cordova.exec(this._onEvent, this._onError, "ChildBrowser", "showWebPage", [url,
options]);
};

/**
 * Close the browser opened by showWebPage.
 */
ChildBrowser.prototype.close = function() {
  cordova.exec(null, null, "ChildBrowser", "close", []);
};

/**
 * Display a new browser with the specified URL.
 * This method starts a new web browser activity.
 *
 * @param url        The url to load
 * @param usecordova Load url in cordova webview [optional]
 */
ChildBrowser.prototype.openExternal = function(url, usecordova) {
  if (usecordova === true) {
    navigator.app.loadUrl(url);
  }
  else {
    cordova.exec(null, null, "ChildBrowser", "openExternal", [url, usecordova]);
  }
};

/**
 * Method called when the child browser has an event.
 */
ChildBrowser.prototype._onEvent = function(data) {
  if (data.type == ChildBrowser.CLOSE_EVENT && typeof
window.plugins.childBrowser.onClose === "function") {
    window.plugins.childBrowser.onClose();
  }
  if (data.type == ChildBrowser.LOCATION_CHANGED_EVENT && typeof
window.plugins.childBrowser.onLocationChange === "function") {
    window.plugins.childBrowser.onLocationChange(data.location);
  }
};
```

```
    }  
};  
  
/**  
 * Method called when the child browser has an error.  
 */  
ChildBrowser.prototype._onError = function(data) {  
    if (typeof window.plugins.childBrowser.onError === "function") {  
        window.plugins.childBrowser.onError(data);  
    }  
};  
  
/**  
 * Maintain API consistency with iOS  
 */  
ChildBrowser.prototype.install = function(){  
};  
  
/**  
 * Load ChildBrowser  
 */  
  
if(!window.plugins) {  
    window.plugins = {};  
}  
if (!window.plugins.childBrowser) {  
    window.plugins.childBrowser = new ChildBrowser();  
}
```

**Table 8 Childbrowser.js**

```
function SalesforceWrapper() {  
    /* AUTHENTICATION PARAMETERS */  
    this.loginUrl = 'https://login.salesforce.com/';  
    this.clientId =  
'3MVG99qusVZJwhsmjZIIeUsFRnadOib8Kv_MPwooFMEi.XpChrZ5cVEcKU_7NR1zfQ  
jjmdHI7wMARXnLlgku';  
    this.redirectUri = 'https://login.salesforce.com/services/oauth2/success';  
  
    /* CLASS VARIABLES */  
    this.cb = undefined; //ChildBrowser in PhoneGap  
    this.client = undefined; //forceTk client instance  
  
    this.init();  
}  
  
SalesforceWrapper.prototype.init = function() {
```

```
this.client = new forcetk.Client(this.clientId, this.loginUrl);
// line 17 with the error
this.cb = window.plugins.childBrowser;
}

SalesforceWrapper.prototype.login = function (successCallback) {
  this.loginSuccess = successCallback;
  var self = this;
  self.cb.onLocationChange = function (loc) {
    if (loc.search(self.redirectUri) >= 0) {
      self.cb.close();
      self.sessionCallback(unescape(loc));
    }
  };
  self.cb.showWebPage(self.getAuthorizeUrl(self.loginUrl, self.clientId, self.redirectUri));
}

SalesforceWrapper.prototype.getAuthorizeUrl = function (loginUrl, clientId, redirectUri) {
  return loginUrl + 'services/oauth2/authorize?display=touch' +
  '&response_type=token&client_id=' + escape(clientId) + '&redirect_uri=' +
  escape(redirectUri);
}

SalesforceWrapper.prototype.sessionCallback = function(loc) {  var oauthResponse =
{};

  var fragment = loc.split("#")[1];

  if (fragment) {
    var nvps = fragment.split('&');
    for (var nvp in nvps) {
      var parts = nvps[nvp].split('=');
      oauthResponse[parts[0]] = unescape(parts[1]);
    }
  }

  if (typeof oauthResponse === 'undefined' || typeof oauthResponse['access_token']
  === 'undefined') {
    console.log("error");
  } else {
    this.client.setSessionToken(oauthResponse.access_token, null,
    oauthResponse.instance_url);
    if ( this.loginSuccess ) {
      this.loginSuccess();
    }
  }
}
```

```
    }  
  }  
  this.loginSuccess = undefined;  
}
```

**Table 9 SalesforceWrapper.js**

ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ