

Antti Ojanen

# Mobiilisovelluksen kehittämisen vaihtoehdot

Opinnäytetyö  
Tietojenkäsittelyn ko


Toukokuu 2013




**MIKKELIN AMMATTIKORKEAKOULU**

Mikkeli University of Applied Sciences

## KUVAILULEHTI

 <b>MIKKELIN AMMATTIKORKEAKOULU</b> Mikkelin University of Applied Sciences	<b>Opinnäytetyön päivämäärä</b>  30.5.2013		
<b>Tekijä(t)</b> Antti Ojanen	<b>Koulutusohjelma ja suuntautuminen</b> Tietojenkäsittelyn koulutusohjelma		
<b>Nimeke</b> Mobiilisovelluksen kehittämisen vaihtoehdot			
<b>Tiivistelmä</b>  <p>Opinnäytetyön tarkoituksena oli tutkia mobiilisovelluksien eri kehitystapoja sekä Google Maps JavaScript APIn toimivuutta. Työhön sisältyy teoriaosuus, joka koostuu mobiilisovelluksien kehitystapojen vertailusta keskenään tässä työssä sekä itse ohjelmointityö, jossa suunnitellaan sekä toteutetaan hybridi-sovellus, joka julkaistaan Applen, Windows Phonen sekä Googlen sovelluskaupassa. Käydään myös läpi eri tekniikoita, joita vaaditaan valmiin sovelluksen saavuttamiseen.</p> <p>Mobiilisovellusten kehittäjillä on mahdollista tehdä kolme eri tapaa lähteä rakentamaan sovellustaan. Vaihtoehdot ovat HTML5-sovellus, hybridsovellus ja natiivisovellus. Jokaisella vaihtoehdolla on omat vahvuutensa ja rajoitteensa. Esimerkiksi natiivisovelluksen vahvuuksia on mm. suorituskyky ja rikas käyttökokemus, kun taas rajoitteita on esimerkiksi yhden sovelluksen tekeminen periaatteessa kolmesti. Tämä antaa paljon mahdollisuuksia kehittäjille, jotka voivat valita parhaan mahdollisen kehitystavan omalle projektilleen. Työssä myös avataan hieman Google Maps API -rajapintaa, jolla on itse tulevassa sovelluksessa suuri rooli reittien näyttämässä.</p> <p>Toteutin ohjelmointityönä Suomen Lentopalloliitolle ja Powercup -lentopalloturnaukselle mobiilisovelluksen. Sovellus tulee olemaan navigointisovellus, joka ohjastaa käyttäjät peli- sekä majoituspaikoille. Päätin tehdä sen hybridsovelluksena, joka on HTML5-sovelluksen ja natiivisovelluksen välimalli. Sillä päästään käsiksi natiivisovelluksen tapaan laitteiden ominaisuuksiin ja saadaan irti myös HTML5-sovelluksen hyöty, eli sovellus tarvitaan tehdä vain kerran käyttäen HTML5- sekä JavaScript - ohjelmointikieliä. PhoneGap on ohjelmointikehys, joka mahdollistaa pääsyn laitteiden ominaisuuksiin esimerkiksi GPS:ään, jota tässä työssä tarvitaan. Lopuksi käänsin sovelluksen Adoben pilvipalvelulla, joka antaa mm. iOS:n, Androidin sekä Windows Phonen alustoille toimivan kokonaisuuden, joka mahdollista laittaa sovelluskauppoihin jaettavaksi.</p>			
<b>Asiasanat (avainsanat)</b>  mobiililaitteet, mobiilipalvelut, navigointi			
<b>Sivumäärä</b> 40	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;"><b>Kieli</b> Suomi</td> <td style="width: 50%;"><b>URN</b></td> </tr> </table>	<b>Kieli</b> Suomi	<b>URN</b>
<b>Kieli</b> Suomi	<b>URN</b>		
<b>Huomautus (huomautukset liitteistä)</b>			
<b>Ohjaavan opettajan nimi</b>  Janne Turunen	<b>Opinnäytetyön toimeksiantaja</b>  Suomen Lentopalloliitto Powercup		

## DESCRIPTION

 <p><b>MIKKELIN AMMATTIKORKEAKOULU</b> Mikkeli University of Applied Sciences</p>		<b>Date of the bachelor's thesis</b>  30 May 2013	
<b>Author(s)</b> Antti Ojanen		<b>Degree programme and option</b> Business Information Technology	
<b>Name of the bachelor's thesis</b> The alternatives of mobile application development			
<b>Abstract</b>  <p>Purpose of this study was to examine the development of mobile applications in different ways as well as the Google Maps JavaScript API functionality. The work included a theoretical part in which I compared mobile applications' development methods available and the actual programming work which was to design and implement a hybrid application to be published in Apple, Windows Phone and Google's app stores. I also introduced a variety of techniques required for the achievement of the final application</p> <p>Mobile applications developers can choose from three different ways to start building the application. These options are HTML5 application, hybrid application, and native application. Each option has its own strengths and limitations. For example, the strengths of a native application are performance and rich user experience while the constraint is that the application must be programmed three times. This gives a lot of opportunities for developers who can select the best development method for their project. This work also introduced a bit Google Maps API which had a major role in the application displaying the routes.</p> <p>As the practical part of this study I developed an application for Suomen Lentopalloliitto and the Powercup volleyball tournament. This was a navigation application guide participants to the game and accommodation venues. I decided to make a hybrid application, which was an intermediate model between an HTML5 application and a native application. It took advantage both the native application and HTML5 application, which made it possible to program the application only once using the HTML5 and JavaScript programming languages. PhoneGap was an application programming interfaces allow access to the device features such as GPS, which was in this work needed. Finally, I compiled the application in Adobe cloud service which enabled the final version for iOS, Android and Windows Phone platforms to be placed in application stores.</p>			
<b>Subject headings, (keywords)</b>  mobile devices, mobile services, navigation			
<b>Pages</b>  40	<b>Language</b>  Finnish	<b>URN</b>	
<b>Remarks, notes on appendices</b>			
<b>Tutor</b> Janne Turunen		<b>Bachelor's thesis assigned by</b> Suomen Lentopalloliitto Powercup	

## SISÄLTÖ

1	JOHDANTO .....	2
2	TEKNIIKAT .....	3
2.1	HTML5 .....	3
2.2	CSS3 .....	5
2.3	JavaScript.....	6
2.4	jQuery Mobile.....	7
3	MOBIILIALUSTAT .....	7
3.1	Windows Phone .....	8
3.2	Android .....	9
3.3	IOS .....	11
4	GOOGLE MAPS .....	12
4.1	Google Maps JavaScript API.....	13
4.2	Google Maps mobiiliversio .....	14
5	MOBIILISOVELLUKSEN KEHITTÄMISEN VAIHTOEHDOT .....	15
5.1	Natiivisovellus .....	16
5.2	HTML5-sovellus.....	16
5.3	Hybridisovellus.....	17
5.4	PhoneGap.....	17
6	TOTEUTUS .....	18
6.1	Suunnittelu.....	18
6.2	Työkalujen asentaminen ja projektin luominen.....	19
6.2.1	PhoneGapin lataaminen ja projektin luominen.....	22
6.3	Ohjelmointi .....	23
6.4	Julkaisu .....	28
6.4.1	Google play .....	29
6.4.2	Apple App Store .....	30
6.4.3	Windows Store apps .....	31
7	POHDINTAA .....	33
	LÄHTEET .....	35

## 1 JOHDANTO

Tämän opinnäytetyön tarkoituksena on suunnitella ja toteuttaa paikkatietoja hyödyntävän mobiilisovelluksen HTML5-, CSS3- ja JavaScript -ohjelmointikielillä ja kääntää se lopuksi eri mobiilialustoille. Valitsin aiheen siksi, koska mobiililaitteiden kehitys ja niihin saatavien sovelluksien suosio ja määrä on noussut kovasti viime vuosina.

Toinen hieman perinteisempi tapa olisi tehdä tämä natiivisovelluksena eli sisäänrakennettuna sovelluksena. Tarkoittaa konkreettisesti sitä, että se ohjelmoidaan Androidille, Windowsille sekä iOS:lle omana sovelluksena, mutta tätä hankaloittaa juurikin se, että pitää jokaiselle mobiilikäyttöjärjestelmälle ohjelmoida oma sovellus näiden kehittäjien omilla työkaluilla ja kielillä.

Työ siis tehdään hybridisovelluksena, joten se käännetään ohjelmointikehyksellä tai vastaavalla pilvipalvelulla, joka mahdollistaa sen toimivan lähes natiivisovelluksen tapaan. Kysymys siis kuuluu, miksi siis tehdä natiivisovelluksia, vaikka on mahdollista tehdä yksi sovellus ja kääntää se sitten kaikille alustoille ilman, että tekee saman sovelluksen monesti eri kielillä? Vastaus on, että tällä tavalla tehdessä tulee vastaan rajoituksia ja asioita, jotka eivät välttämättä toimi samalla tapaan, kuten natiivisovelluksessa.

Samalla tutkin, kuinka Googlen tarjoama rajapinta Google Maps API toimii mobiilisovelluksena. Google Maps API on rajapinta, joka antaa mahdollisuuden käyttää Googlen tarjoamia karttapalveluita. Tätä rajapintaa on myös tarkoitus käyttää työssäni. Tästä aiheesta on lisää luvussa 4. Selvitän myös mobiilisovelluskehityksen vaihtoehtoja, joita tässä aiemmin hieman mainitsinkin. Eli tutkitaan kolmea eri toteutusvaihtoehtoa: natiivi, HTML5-sovellus ja näiden kahden välimallia eli hybridisovellusta.

Ohjelmointityön tavoitteena on tehdä Power Cup:iin karttapalvelu, joka on jo kehittynyt maailman suurimmaksi lasten sekä nuorten lentopalloturnaukseksi, joka järjestetään Mikkelissä kesällä 2013. Sovelluksen tarkoituksena on pystyä navigoimaan ja löytämään majoitus- sekä pelipaikat, jotka ovat upotettuna valmiiksi siihen, sillä ulkopaikkakuntalaisia tulee olemaan kaupungissa silloin paljon. Valmiin sovelluksen olisi tarkoitus olla käytettävissä itse tapahtumassa.

## 2 TEKNIIKAT

Tässä luvussa käsitellään tekniikoita, joita tässä työssä tarvitaan. HTML5, CSS3 sekä JavaScript ovat pääosassa ohjelmointityön teossa, joten niitä on hyvä hieman tarkentaa, vaikka niistä on varmasti kirjoitettu jo monessa työssä. Nämä kaksi ensin mainittua on verkkosivujen tekemisen kulmakivet. Asiaa voitaisiin havainnollistaa siten, että jos niiden avulla rakennettaisiin talo, olisi HTML5 talon perustukset ja CSS3 sen julkisivu. JavaScriptillä taas sivuista tehdään dynaamisemmat.

Tässä työssä käytetään PhoneGap-sovelluskehystä, joka kääntää verkkosivun ns. hybridisovellukseksi. Huhtamäen mukaan (2011, 6) sovelluskehys on ”puolivalmis” ohjelmisto. Sen tarkoituksena on tarjota erilaisia valmiita toimintoja.

### 2.1 HTML5

HTML5 on viimeisin versio HTML -standardista, jonka ensimmäinen versio tuli vuonna 2008. Se on merkkaukieli, joka jäsentää ja esittää World Wide Webin sisältöä. Se ei ole vielä täysin valmis, mutta tulevaisuuden näkymät hyvät. Se tuo kehittäjille kapasiteettia, jolla he voivat muokata World Wide Webin olemuksen uudelleen. (Lowery & Fletcher 2011, 27–39.)

HTML on lyhenne ja se tulee sanoista HyperText Markup Language. HyperText tarkoittaa periaatteessa eri web-sivujen linkittämistä. Markup Language taas on rykelmä elementtejä eli tageja, joilla voidaan teksti näyttää luettavasti esimerkiksi selaimessa. (Lowery & Fletcher 2011, 35.) HTML koostuu siis elementeistä, joista esim. <body>-tagi määrittää sivun rungon, jossa sitten näkyvät kaikki kirjoitukset, kuvat, linkit jne. Elementti suljetaan tagilla </body>.

Mitäs uutta sitten HTML5 tuo? Uudistuksia ovat mm. elementit <video> ja <audio> sekä paikannusominaisuus, joka on mielenkiintoinen omaa työtäni kohden. Ennen HTML5:sta videoiden katselemiseen tarvittiin verkkoselaimen lisäosa, esimerkiksi Adobe Flash Player, mutta nyt HTML5 tarjoaa natiivin tuen videoiden sekä äänien soittamiseen käyttäen mainitsemaani <video> ja <audio> elementtiä. (Lowery & Fletcher 2011, 39.) Nykypäivänä HTML5 on ottanut ison askeleen varsinkin tablettien ja

mobiililaitteiden kehityksessä, sillä Adoben Flash Player tuki otettiin niistä pois kokonaan.

Tällä hetkellä HTML5 tukee kolmea eri video formaattia:

- MP4
- WebM
- Ogg.

Taulukosta 1 huomaa, ettei mikään näistä kolmesta formaatista tue suurimpia selaimia. Ratkaisu tähän on tekemällä videosta eri tiedostoversiot ja käyttämällä niitä sitten oikean selaimen kanssa (Karlins 2011, 172).

### TAULUKKO 1. Videoformaatin selain tuki (HTML5 Video 2013)

Selain/Formaatti	MP4	WebM	Ogg
Internet Explorer v. 9+	Kyllä	Ei	Ei
Chrome v. 6+	Kyllä	Kyllä	Kyllä
Firefox v. 3.6+	Ei	Kyllä	Kyllä
Safari v. 5+	Kyllä	Ei	Ei
Opera v. 10.6+	Ei	Kyllä	Kyllä

HTML5 tukee MP3, Wav sekä Ogg ääniformaatteja. Myös äänipuolella on samankaltaista konfliktia formaattien kanssa. Kuten taulukosta 2 näkyy, niin lähes kaikki tukevat Wav (Waveform Audio File Format) formaattia.

### TAULUKKO 2. Ääniformaatin selain tuki (HTML5 Audio 2013)

Selain/Formaatti	MP3	Wav	Ogg
Internet Explorer v. 9+	Kyllä	Ei	Ei
Chrome v. 6+	Kyllä	Kyllä	Kyllä
Firefox v. 3.6+	Ei	Kyllä	Kyllä
Safari v. 5+	Kyllä	Kyllä	Ei
Opera v. 10+	Ei	Kyllä	Kyllä

Paikannusominaisuus (geolocation) on ominaisuus, jolla voidaan paikantaa esimerkiksi verkkosivut IP:n avulla. Se ei kuitenkaan ole välttämättä aina täysin tarkka, mutta pitäisi olla ymmärrettävissä. Se ei myöskään näytä sijaintia ilman käyttäjän lupaa, sillä selain kysyy käyttäjältä lupaa näyttää sijainti. Tämä ominaisuus on myös rajapinta,

joten se tarvitsee esimerkiksi JavaScriptiä toimiakseen. (Lowery & Fletcher 2011, 279.)

## 2.2 CSS3

CSS on tyyli pohja, jolla käyttäjä voi määrittää kehittämänsä verkkosivun, yleensä HTML-pohjaisen verkkosivun tyylin ja ulkonäön. Sitä voi myös hyödyntää esimerkiksi XML-dokumenteissa. CSS on suunniteltu ensisijaisesti sitä varten, että voitaisiin erotella sisältö ja ulkoasu toisistaan. Se parantaa sivun esteettömyyttä ja tyylien ominaisuuksien kontrollointia. Se myös mahdollistaa saman merkkauksielisen sivuston näyttämisen erilaisilla tyyliillä eri selaimissa. Myös voidaan vaikuttaa miten se näkyy erikokoisilla näytöillä tai laitteilla. (Cascading Style Sheets 2013.)

Mitä uutta CSS3 sitten on tuonut? Se mahdollistaa esimerkiksi varjojen laittamisen, kulmien pyöristämisen sekä läpinäkyvyyden säätämisen. Nämä tuovat lisää mahdollisuuksia tehdä sivuista näyttävämmät. Myös interaktiiviset toiminnot, kuten esimerkiksi hiiren viemisen tietyn objektin päälle ja siitä syntyvän animaatio tuovat kehittäjälle helpotusta, sillä ennen tarvittiin samankaltaisen efektin tekoon esimerkiksi JavaScriptin apua. (Karlins 2011, 124.)

### TAULUKKO 3. CSS3 ominaisuuksien yhteensopivuus (HTML5 & CSS3 Support)

	Firefox	Opera	Chrome	Safari	Internet Explorer			
Versio	11	11.61	18	5.1	6	7	8	9
Läpinäkyvyys	Kyllä	Kyllä	Kyllä	Kyllä	Ei	Ei	Ei	Kyllä
Heijastukset	Ei	Ei	Kyllä	Kyllä	Ei	Ei	Ei	Ei
Reunojen	Kyllä	Kyllä	Kyllä	Kyllä	Ei	Ei	Ei	Kyllä

Kuten taulukon 3 esimerkeistä huomataan, niin yhteensopivuus vaihtelee Windows-pohjaisten selainten kesken. Seuraavasta linkistä <http://fmbip.com/litmus/> voi käydä katsomassa tarkemmin CSS3 sekä HTML5 yhteensopivuuksia selainten kanssa.

CSS3 on myös HTML5 tapaan keskeneräinen, joten yhteensopivuus ongelmia on eri selainten kanssa. Pääselaimista uusimmat versiot Firefox, Chrome, Opera sekä Safari on tuettu hyvin, mutta Internet Explorerin kanssa saattaa olla ongelmia, varsinkin vanhempia versioita 6, 7 ja 8 ei ole tuettu juuri ollenkaan.



## 2.3 JavaScript

JavaScript on komentosarjakieli, joka alun perin toimi osana verkkoselainta, jotta selainohjelman komentosarjat ns. scriptit voivat olla vuorovaikutuksessa käyttäjän kanssa, esimerkiksi hiiren klikkaukset tai lomakkeiden tarkistuksessa. JavaScriptin käyttö on huomattavaa myös muissa sovelluksissa kuin pelkästään verkkosivuilla, esimerkiksi PDF-dokumenteissa tai työpöydän ”widgeteissä”. JavaScriptiä ei pidä sekoittaa Javaan, sillä näillä kahdella ei ole periaatteessa mitään yhteistä, vaikka molemmat ovat olio-pohjaisia ja molempien nimessä on yhteneväisyyttä. (JavaScript 2013.)

Joskus puhutaan, että JavaScript on sama kuin ECMAScript. ECMA (European Computer Manufacturers Association) on yksityinen organisaatio, joka kehittää standardeja informaatio- ja tietokonesysteemeille. JavaScript on yksi heidän standardeista, jota he kutsuvat ECMAScriptiksi. (Wilton & McPeak 2010, 4.)

Muitakin scriptauskieliä on olemassa, esimerkiksi VBScript ja Perl. Mutta mikä tekee JavaScriptistä parhaimman? Suurin syy on siinä, että Javascriptin levinneisyys ja saatavuus tekevät siitä sen numero ykkösen, sillä kaikista suosituimmat selaimet tukevat sitä. Jos verrataan näiden kahden muun kielen tukia selaimissa, niin VBScript toimii vain Internet Explorerissa ja Perl ei toimi ollenkaan. (Wilton & McPeak 2010, 4.)

Luullaan, että JavaScripti on pelkkä muutaman kätevän funktion tekemiseen tarkoitettu kieli. Todellisuudessa sillä voi tehdä myös kehittyneitäkin sovelluksia. Wiltonin mukaan (2010, 4) Google Maps on tehty lähes pelkästään JavaScriptillä, joka siis tarjoaa karttapalvelun. Myös Sheong presisoi (2008, 19), että Google Maps on koodattu JavaScriptia sekä XML:llä käyttäen, ja että Google tarjoaa ilmaisen Google Maps API:n kehittäjille, jotta he voivat integroida Google Mapsin omiin sovelluksiinsa. Täytyy muistaa kuitenkin, vaikka JavaScript tarjoaa huikean käyttöliittymän, niin se ei itsessään liikuta minkäänlaista dataa, vaan sen homman tekevät palvelimet. Eli kyseessä on tehokas, mutta silti tiettyjä rajoituksia omaava kieli.

## 2.4 jQuery Mobile

jQuery Mobile on monelle eri alustalle kehitetty ohjelmointikehys, joka helpottaa ja tehostaa HTML5-sovellusten sekä hybridisovellusten tekoa mobiililaitteille. Plotzin (2012) mukaan se yhdistää HTML5, CSS3, jQuery:n sekä jQuery UI:n yhteen ja samaan ohjelmointikehykseen.

Ensimmäisen kerran jQuery Mobilesta kuultiin elokuussa 2011, kun jQuery:n kehitysryhmän blogissa julkaistiin uutinen, jossa haluttiin tuoda jQuery omana ohjelmointikehyksenä mobiililaitteille. Isoin työ oli saada jQuery toimimaan kaikilla merkittävimmillä verkkoselaimilla. Se näkyi monina bugi- sekä parannuspäivityksillä, joilla pyrittiin pitämään laatu korkealla. (Giulio 2011, 8.)

jQuery Mobile on rakennettu jQuery:n päälle, joka on JavaScriptin kirjasto, joten jos jQuery on yhtään tuttu ennestään, niin ei ole ongelmia saada jQuery Mobile toimimaan. Se on myös yhteensopiva kaikkien merkittävimpien alustojen sekä modernien selaimien kanssa. Sen yksi hyvä ominaisuus on pieni koko (n. 20 kilotavua pakattuna), joka tekee siitä suorituskyvyltään nopean. (Plotz 2012.)

Pähkinänkuoressa jQuery Mobile antaa siis mahdollisuuden tehdä kätevästi käyttöliittymiä mobiililaitteille. Se tarjoaa paljon uusia toimintoja, kuten välttämättömän kosketustoiminnon esim. sormen painalluksen tai pyyhkäisyn. Valmiiden komponenttien, kuten esimerkiksi painikkeet ja taulukot sekä niiden tyylit tulevat perusteeman kera. On myös mahdollista tehdä omia teemoja esimerkiksi jQuery Mobilen kotisivuilta löytyvältä generaattorilta. (Ortiz, 2011.)

## 3 MOBIILIALUSTAT

Kolmas luku käsittelee mobiilialustoja, eli älypuhelimissa toimivia käyttöjärjestelmiä, joille valmis verkkosovellus on tarkoitus kääntää. Mobiilikäyttöjärjestelmien kehitys on ollut huimaa, sillä niiden ominaisuuksia on mm. kosketusnäyttötuki, kameratuki ja GPS-navigointituki. Nämä alustat toimivat myös tableteissa sekä PDA-laitteissa. Kaikilla alustoilla on myös oma SDK eli Software Development Kit, joka sisältää tarvittavat työkalut esimerkiksi puhelimen simulaattorin.

Markkinoilla on kova kilpailu myös käyttöjärjestelmien osalta. Simonsonin mukaan (2010) noin kymmenen mobiilikäyttöjärjestelmää kilpailee keskenään markkinoista ja ennakoi, että joka toisella on kysyntää myös tulevaisuudessa. Taulukosta 4 nähdään käyttöjärjestelmien vuoden 2012 viimeisen neljänneksen markkinointi osuus. Vertailu kohteena on vuoden 2011 viimeisen neljänneksen osuus.

**TAULUKKO 4. Käyttöjärjestelmien markkinaosuus (Devilla 2013)**

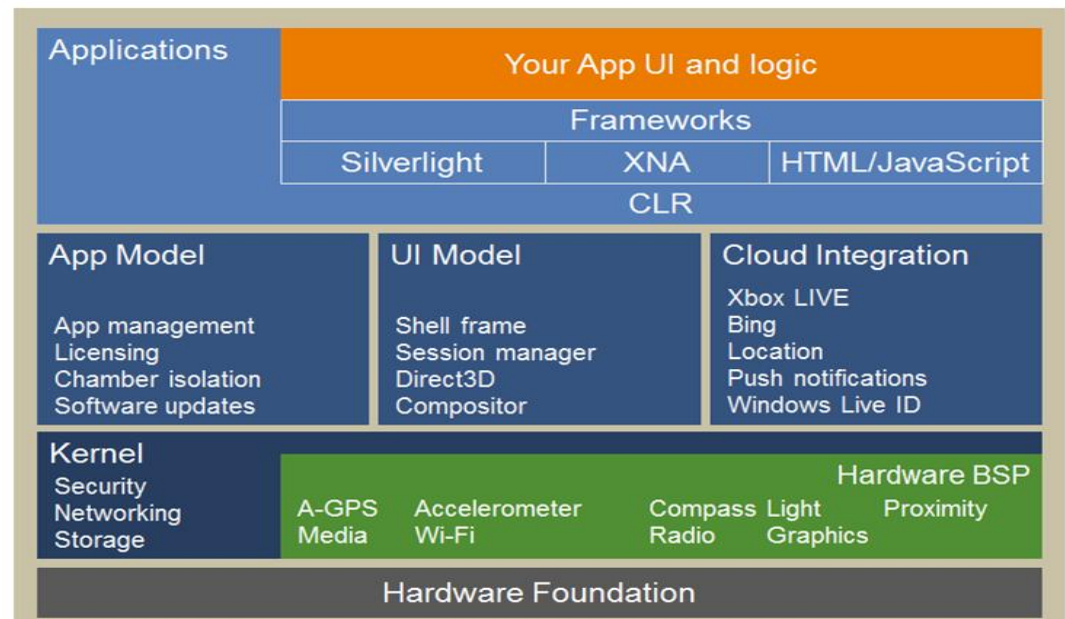
Käyttöjärjestelmä	4/4 2012 (%)	4/4 2011 (%)	Kehitys (%)
Android	70,1	52,9	88
iOS	47,8	23	29,2
Blackberry	3,2	8,1	– 43,1
Windows Phone	2,6	1,5	150
Linux	1,7	2,4	– 2,6
Muut	1,3	12,1	– 84,6

### 3.1 Windows Phone

Windows Phone -käyttöjärjestelmä on Microsoftin kehittämä alusta, jonka tarkoituksena on haastaa markkinoita hallussa pitäviä Androidia sekä iOS:ia. Se kehitettiin alun perin pelkästään Nokian puhelimille, mutta myös taiwanilainen HTC käyttää sitä lanseeraamassaan 8X-puhelimessaan. Vaikka tuntuukin siltä, että Windows Phone alusta tuli markkinoilla hieman jälkijunassa, niin asia kuitenkin niin ole. Lecrenskin (2011, 2) mukaan ensimmäinen kosketus tuli tehtyä vuonna 2000, ei kuitenkaan mobiililaitteen muodossa, vaan Pocket PC:n alustana. Silloin Microsoft julkaisi myös kehitysohjelmuja, joilla pystyi kehittämään natiivisovelluksia C++:lla Pocket PC:lle.

Applen sekä Androidiin pohjautuvien laitteiden menestyksen innoittamana Microsoft päätti kehittää uuden käyttöjärjestelmän, nykyisen Windows Phonen ja sen ensimmäinen versio 7, julkaistiin vuonna 2010. Tärkeää kuitenkin oli, että se tarjosi kehitysalustan, joka hyödynsi .NET, Silverlight- sekä XNA -tekniikoita, sillä niillä oli jo valtava määrä kehittäjiä. Tämä siis helpottaa suunnattomasti kehittäjiä, sillä heidän ei tarvitse ryhtyä oppimaan uusia tekniikoita kehittääkseen Windows Phone alustalle sovelluksia. (Lecrenski & Watson & Fonseca-Ensor 2011.) Kuvasta 1 nähdään Windows Phonen ohjelmistokehityksen arkkitehtuuri.

**KUVA 1. Windows Phone arkkitehtuuri (Microsoft & Nokia 2011)**



Uusin versio, Windows Phone 8, nimeltään ”Apollo”, julkaistiin kesäkuussa 2012. Nokia, Huawei, Samsung sekä HTC ottaa käyttöön tämän version uusimmissa puhelimissaan. Windows Phone 7.x versiota ei pysty päivittämään ”Apolloon”, sillä se suurin osa sen uudistuksista on laitteisto riippuvaisia. Se vaatii moniytimisiä Windows Phoneja, joihin on rakennettu NFC-tuki, eli lyhyen kantaman tiedonsiirtotekniikka, sekä uusi Windows Phone -ydin, joka ei ole upotettu kiinteä ydin. (Mary Jo Foley 2012.) Foley (2012) mukaan kuitenkin jotain hyvää luvassa Windows Phone 7.x-älypuhelimien omistajille, sillä Windows Phone 7.8 -versio on tulossa, joka tuo vähintään yhden Windows Phone 8 -ominaisuuden näille älypuhelimille.

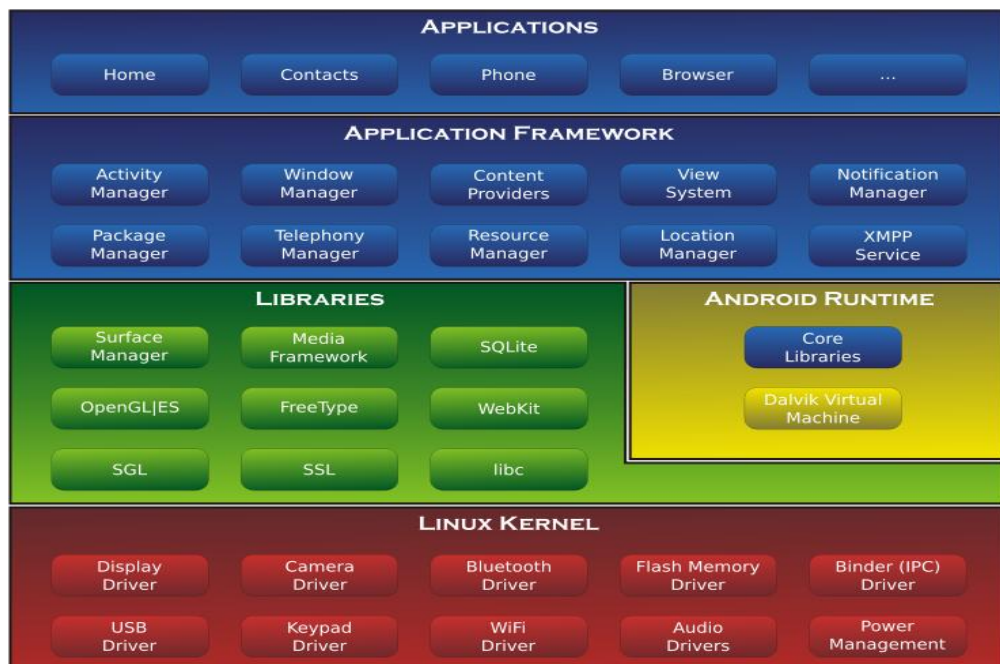
Helppo tapa lähteä kehittämään sovelluksia Windows Phone -alustalle on Visual Studio avulla, jonka voi ladata ilmaiseksi. Myös siihen saatava SDK ja muut työkalut on saatavissa ilmaiseksi. Pieni osaaminen C# -ohjelmointikieltä ei ole pahitteeksi, sillä se on pääkieli Windows Phone -sovelluksissa.

### 3.2 Android

Android on käyttöjärjestelmä, joka on muokattu versio Linuxista. Vuodesta 2007 lähtien Androidia on kehittänyt Open Handset Alliance -niminen yhtymä (Android

2013). Se on pääasiassa suunniteltu kosketusnäytöllisille mobiililaitteille esim. älypuhelimelle. Alun perin sitä kehitti samanniminen Android -kehitysryhmä, mutta vuonna 2005 Google osti Androidin kehitystiimeineen. Lee (2011, 2) toteaa, että suurin osa Androidin koodista julkaistiin vapaaseen lähdekoodiin perustuvan Apache-lisenssin alla, sillä Google halusi Androidin koodin olevan ilmainen ja vapaasti muokattavasti.

Ensimmäinen Android -versio julkaistiin syyskuussa 2008, kun laite valmistaja HTC julkaisi Dream -puhelimensa. Samalla Android julkaisi ensimmäisen SDK:n, versio 1.0 (Morris 2008). Julkaistuja Android -versioita on tullut 17 kappaletta, joista uusin on 4.2, joka tunnetaan myös nimeltä ”Jelly Bean” ja se julkaistiin lokakuun lopulla 2012. LG:n Nexus 4 ja Samsungin Nexus 10 oli ensimmäiset laitteet, jotka tuki tätä versiota. (Android version history 2013.) Trenholmin (2013) mukaan Google kehittää uutta 5.0 versiota Androidista ja sen on huhuiltu tulevan markkinoille vuoden 2013 toisella neljänneksellä.



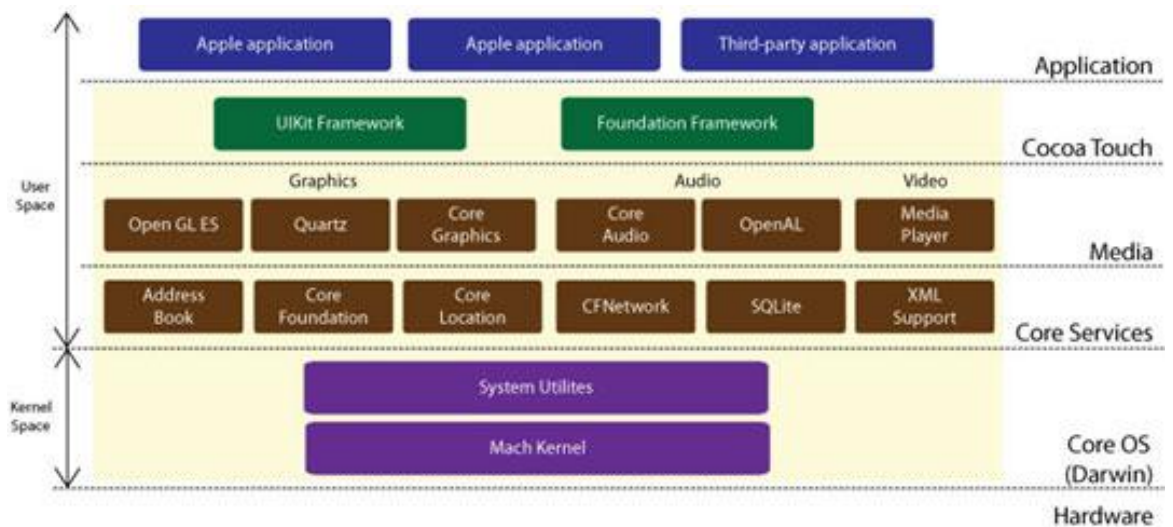
**KUVA 2. Androidin arkkitehtuuri (Android – System architecture 2007)**

Kuvasta 2 nähdään malli Androidin arkkitehtuurista. Käyttöjärjestelmä koostuu kerroksista, joiden alemmat kerrokset tarjoavat eri palveluita ylemmälle kerrokselle. Sovellusten kehitystapa on käyttää Java -ohjelmointikieltä sekä Androidin SDK:ta. Kaikki Androidin sovellusten kehitykseen tarvittavat työkalut ovat ilmaisia ja ovat saatavissa Internetistä. Suosituin kehitysympäristö on Eclipse, jota voi käyttää Windowsilla, Linuxilla tai MAC:lla.

### 3.3 IOS

Kolmas tässä työssä esiteltävä mobiilialusta Applen kehittämä iOS, joka pohjautuu OS X -käyttöjärjestelmään, jota käytetään Applen tietokoneissa. Sen ensimmäinen versio tuli ulos vuonna 2007 iPhoneille, joka sisälsi paljon hienoja ominaisuuksia, mutta ei 3G:tä. Ensimmäinen version myötä tuli myös Applelta uusi laite iPod Touch, joka myös toimi iOS -käyttöjärjestelmällä. IOS:in toisen version myötä tuli App Store, eli kauppapaikka, joka mahdollisti sovelluskehittäjien tekemien sovellusten myymisen. Samaan versioon tuli myös tuki 3G:lle. IOS:in tuki lajeeni iPadin ensimmäisen sukupolven myötä, jossa pyöri iOS 3.2 -versio, kun se julkaistiin vuonna 2010. Viimeisin versio on iOS 6, joka julkaistiin syyskuussa 2012, jonka myötä ei enää iPhone 3GS, iPod Touch 4. sukupolven ja iPad 2 vanhempia laitteita tueta. (IOS version history 2013.)

Apple ei kuitenkaan ole antanut lupaa iOS:in asentamisesta muihin kuin oman Apple perheen laitteisiin toisin kuin Android ja Windows Phone, jotka voidaan asentaa useisiin eri valmistajan laitteisiin.



**KUVA 4. IOS arkkitehtuuri (Bada Developers 2011)**

Kuvasta 4 nähdään malli iOS käyttöjärjestelmän arkkitehtuurista. Objective-C -ohjelmointikieli, jolla on kehitetty iOS -käyttöjärjestelmä ja joka pohjautuu muihin C-kieliin (C, C#, C++). Vähintään Mac-tietokone tarvitaan, sillä sovelluksia ei voi tehdä muilla alustoilla. Applella on oma ilmainen kehitysympäristö nimeltään Xcode ja

SDK, jotka tarvitaan sovelluskehityksessä iOS:lle. Jos halutaan lopullinen sovellus ajaa oikeassa laiteessa, niin silloin pitää maksaa \$99 dollarin provisio Applelle.

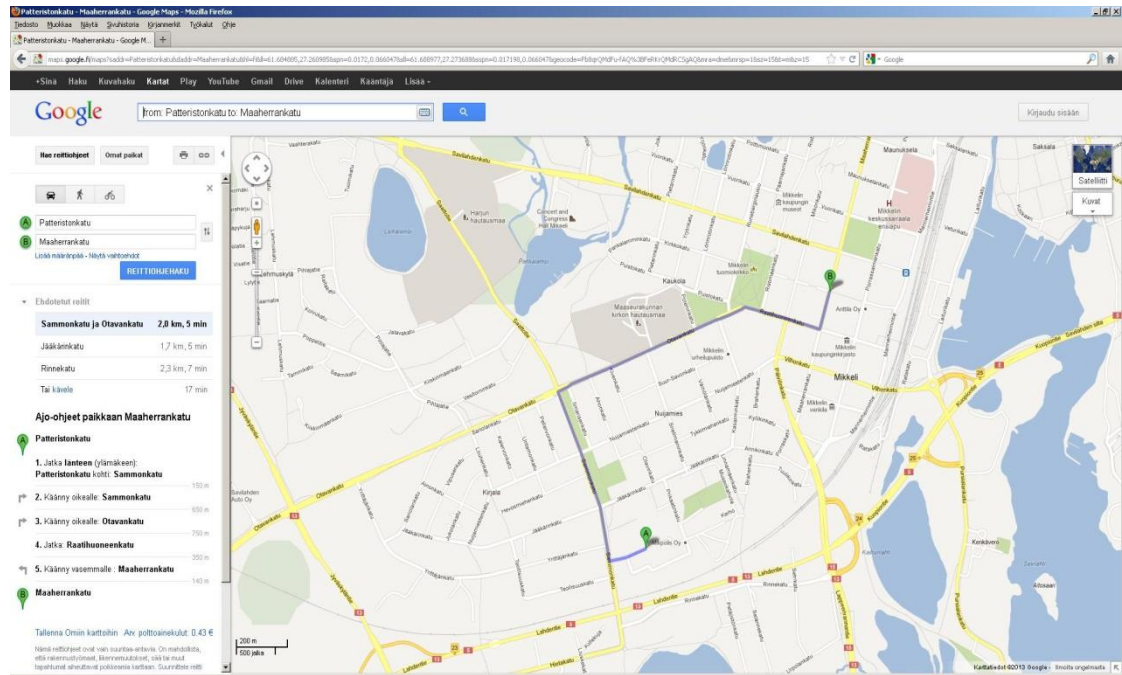
#### 4 GOOGLE MAPS

Google Maps on tehokas karttapalvelu, jonka on kehittänyt Google. Se tarjoaa monta hyvää karttoihin liittyvää palvelua mm. Google Maps -verkkosivun ja Google Maps API -rajapinnan, jota tässä työssä käytetään. Tarkemmin nämä palvelut sisältävät ominaisuuksia, joilla voidaan esimerkiksi tehdä reittihaku paikasta A paikkaan B ja valita haluamansa kulkuneuvo. Se myös sisältää Street View -ominaisuuden, joka antaa näkymän katukuvasta. (Google Maps API 2013.)

On mahdollista myös vaihtaa kartan kuva vaikkapa satelliittikuvapohjaiseksi, jolloin saa rikkaamman kokemuksen kartasta. Tosin kaikki paikat ei ole kuvattu yhtä terävänä, sillä vähempi asutetut paikat sisältävät vähemmän yksityiskohtia. Monet hallitukset ovat valittaneet tästä, sillä he pitävät sitä mahdollisena uhkana, koska terroristit pystyvät käyttämään sitä avukseen suunnitellessaan hyökkäyksiä. Google on esimerkiksi sumentanut joitain Yhdysvalloissa sijaitsevia tärkeitä alueita tämän vuoksi mm. Valkoisen Talon ja U. S Capitol -alueet (Blurred Out: 51 Things You Aren't Allowed to See on Google Maps 2008).

Gauthamin (2012) mukaan moni muukin Googlen kehittämä verkkosovellus, myös Google Maps käyttää laajalti. Kun käyttäjä raahaa karttaa, niin sovelluksen kartan kuvat ladataan palvelimelta ja asetetaan ne ruudulle ilman, että sivu lataantuu uudelleen. Kuva 5 on näkymä Google Maps verkkosivulta, jossa on tehty reittihaku Mikkelin ammattikorkeakoululta kauppatorille.

## KUVA 5. Näkymä Google maps verkkosivulla



Google Maps API -rajapinta ei ole ainut Googlen tarjoama rajapinta. Esimerkiksi iOS -käyttöjärjestelmälle on oma rajapintansa, jota hyödyntäen voidaan tehdä karttapalveluja natiivisovelluksina. Google Maps SDK for iOS -rajapinta sisältää perus reitti ja navigointi ominaisuuksien lisäksi myös 3D -mallinnettuja rakennuksia. Myös Androidille on oma samantapainen rajapinta. (Google Maps SDK for iOS 2013.)

### 4.1 Google Maps JavaScript API

Vuonna 2005 Google julkaisi Google Maps JavaScript API -rajapinnan, joka antaa kehittäjille mahdollisuuden integroida Google Maps -palvelun omille verkkosivuilleen. Se on ilmainen ja tällä hetkellä ei sisällä mainoksia, mutta on mahdollista, että Google saattaa käyttää oikeutta näyttää mainoksia tulevaisuudessa (Google Maps API – Terms of Use 2012).

Google Maps JavaScript API on tehty JavaScriptillä, joka helpottaa sen upottamista verkkosivulle. Sillä on mahdollista tehdä oma karttapalvelu, johon on esimerkiksi tehty valmiiksi kehittäjän haluamansa reittivalinnat tai muita toimintoja. Ilmaisen version rajoitteena on 25 000 lataus kerran raja, mutta saatavilla on myös maksullinen lähinnä yrityksille suunnattu Google Maps API for Business. (Google Maps API FAQ 2013.)



Kaikkien Maps API -rajapintojen pitäisi ladata kartat käyttäen Googlen antamaa API – avainta. Se mahdollistaa kehittäjän tarkkailla oman rajapintansa käyttöä ja myös antaa Googlelle mahdollisuuden ottaa kehittäjään yhteyttä tarvittaessa mm. rajapinnan käyttölimiittien ylittyessä. Sen saa tekemällä Google Account –tilin, jolloin Google antaa API –avaimen ja jota käyttämällä omassa koodissa saa luotua yhteyden karttoihin. Kuvasta 6 nähdään kuinka API – avainta käytetään koodissa. Key - parametri on kehittäjän API -avain ja sensor -parametri tarkoittaa sitä, että käyttäkö sovellus sensoreita eli mm. GPS – paikannusta. (Google Maps JavaScript API v3 2013.)

#### **KUVA 6. API -avaimen näkymä koodissa**

```
<script type="text/javascript"
  src="https://maps.googleapis.com/maps/api/js?key=AIzaSyCDYtYU6Qo3aEc7JpiE3wzGJPoVBhdGvLQ&sensor=true">
</script>
```

Muita Google Maps JavaScript API -rajapinnan tarjoamia ominaisuuksia on esimerkiksi ”Overlays”. Sen avulla voidaan laittaa objekteja karttaan haluttuun koordinaatti kohtaan, jotka pysyvät paikallaan vaikka karttaa liikuteltaisiin tai zoomattaisiin. Toinen tärkeä ominaisuus ovat tapahtumat, joilla voidaan hiiren tai sormen painalluksilla määritellä haluttu toiminto. (Google Maps JavaScript API v3 2013.)

#### **4.2 Google Maps mobiiliversio**

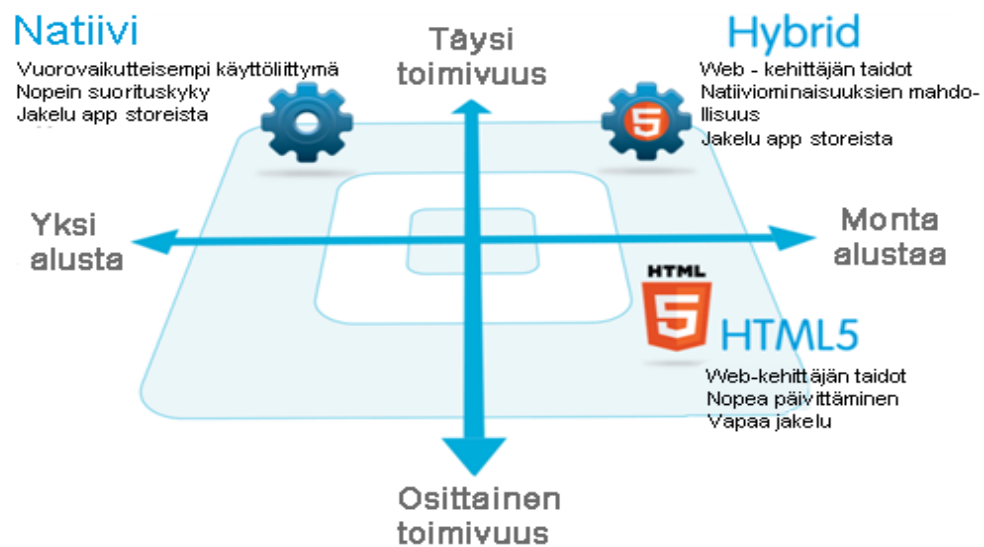
Google esitti vuonna 2006 Java sovelluksen nimeltä Google Maps for Mobile, jonka tarkoituksena oli toimia missä tahansa mobiililaitteessa, joka tukee Javaa. Monet Google Mapsin verkkopalvelun ominaisuuksista toimii myös tässä sovelluksessa. Vuonna 2007 Google Maps for Mobilen toinen versio näki päivänvalon, jonka suurimpana uudistuksena oli paikannus hyödyntäen GPS:ää tai lähintä mahdollista langatonta verkkoa. Vuoteen 2012 mennessä sitä tukee mm. Android, iOS, Windows Phone BlackBerry ja Symbian mobiilialustat. (Google Maps Mobiililaitteille 2013.)

Apple kuitenkin päätti vuonna 2012 kesällä korvaavansa Google Mapsin omalla karttapalvelulla iOS 6:een. Tosin ilman ongelmia ei tämä onnistunut, sillä muutamissa miinuteissa julkaisun jälkeen käyttäjät raportoivat virheistä Esimerkiksi Helsingin rautatieasema oli muuttunut puistoksi ja Dubliniin oli tullut uusi lentokenttä. Joulukuussa 2012 Google julkaisi Google Mapsin iOS:lle vaihtoehtona Applen omalle kart-

tapalvelulle. Parissa päivässä sitä oli ladattu jo yli 10 miljoonaa kertaa. (Google Maps 2013.)

## 5 MOBIILISOVELLUKSEN KEHITTÄMISEN VAIHTOEHDOT

Kun tarvitaan tehdä oma mobiilisovellus, silloin yleensä mietitään kokonaisuutta ja kuinka lähteä ylipäätään rakentamaan sovellusta. Tehdäänkö natiivisovellus, jolla saadaan paras suorituskyky, mutta joudutaan tekemään eniten töitä sen eteen? Vai tehdäänkö perusverkkosovellus tutuilla standardeilla verkkotekniikoilla, jolloin tarvitsee kerran ohjelmoida sovellus, joka sitten toimii monilla eri alustoilla ja laitteilla, mutta samalla menetetään mahdollisuus hyödyntää laitteen ominaisuuksia? Onko sittenkin paras vaihtoehto tehdä hybridisovellus, jolloin HTML5:lla koodatulla verkkosovelluksella voidaan yhdistää natiivisovelluksen ominaisuuksia? Joka tapauksessa jokaisella näistä kolmesta vaihtoehdosta on hyvät sekä huonot puolensa. Yleensä kokeilemalla selviää haluttu vaihtoehto, joka täyttää ruutunsa parhaiten. (Korf & Oksman 2012.)



**KUVA 7. Kehitystapojen vertailua mukailen (Korf & Oksman 2012)**

Kuvasta 7 voidaan nähdä näiden kolmen vaihtoehdon ominaisuuksien vertailut toisiinsa. Korfin (2012) mukaan asiat, jotka auttavat löytämään oikean vaihtoehdon, ovat esimerkiksi kehitystiimin taidot, tietoturva ja mitä laitteen ominaisuuksia halutaan hyödyntää.

## 5.1 Natiivisovellus

Natiivisovellus tarjoaa kaikista parhaimman käytettävyyden, ominaisuudet sekä suorituskyvyn. Ne kehitetään yleensä kehitysympäristössä, joka tarjoaa työkaluja sovelluksen testaamisesta aina projektin hallitsemiseen asti. IOS-, Android- ja Windows Phone -sovellukset ohjelmoidaan eri kehitysympäristöissä ja kielillä, joka tuo oman haasteensa natiivisovellusten kehittämiseen. (Korf & Oksman 2012.)

Myös DuPont (2012) toteaa, että natiivisovelluksen suurin puute on sen koodin siirrettävyys, sillä yhdelle alustalle kehitetty sovellus toimii vain sillä alustalla ja jos halutaan tehdä eri alustalle sama sovellus, on se tehtävä kokonaan alusta ja eri kehitysympäristössä. Hän myös kehuu natiivisovelluksen ominaisuuksia, kuten suorituskykyä ja niiden tuomaa hyvää ns. rikasta käyttökokemusta. Hammondsin (2012) mukaan natiivisovelluksen hyötyjä ovat esimerkiksi mahdollisuus käyttää laitteen ominaisuuksia, yhteydetön tallennustila sekä runsaampi käyttöliittymä. Rajoitteita taas ovat eri kehittämistekniikat ja ympäristöt, saman työn ”monistaminen” sekä lisääntyneet kustannukset.

## 5.2 HTML5-sovellus

HTML5-sovellus on periaatteessa verkkosivu tai yhdistelmä niitä, jotka ovat kehitetty toimimaan pienellä ruudulla. Niitä voidaan näyttää millä tahansa modernilla mobiiliselaimella. Koska sovelluksesi sisältö on verkossa, niin sen sisältö on hakukoneiden saatavilla, joten tämä voi olla suuri hyöty joillekin sovelluksille. (Korf & Oksman 2012.)

Kehittäjän on helppo lähteä tutulla HTML5-tekniikalla kehittämään näitä sovelluksia ja vaikka olisikin aivan aloittelija, niin rima on paljon alempana, kuin esimerkiksi jos lähtisi aloittelijana kehittämään natiivisovellusta. Toki pientä ongelmaa tulee olemaan testaamisessa, sillä jokaisella laitteella on omat näytön koot ja resoluutiot. Tärkeä osa HTML5-sovellusta on niiden jakelu ja tuki, joka on helpompaa jos verrataan natiivisovellukseen, sillä esimerkiksi bugin korjaaminen sovelluksessa tapahtuu yhtä aikaisesti kaikille. (Korf & Oksman 2012.)

Vaikka näiden sovellusten kehittäminen kuulostaa hyvältä, niin niillä ei kuitenkaan päästä käsiksi laitteen ominaisuuksiin. Niillä ei saada aikaiseksi natiivisovelluksen vaikutelmaa, vaikka CSS3 tarjoaa hienoja animaatioita ja graafisia hienouksia. (Korf & Oksman 2012.) Myös yhteydetön tallentaminen ja tietoturvan vajavuus ovat isoja HTML5-sovellusten rajoitteita.

### 5.3 Hybridisovellus

Hybridisovellus yhdistää natiivisovelluksen ja HTML5-sovelluksen ominaisuudet yhdeksi kokonaisuudeksi. Korfin (2012) mukaan hybridisovellus määritetään verkkosovellukseksi, sillä se on suurimmaksi osaksi tehty HTML5:lla sekä JavaScriptilla, joka sitten käännetään natiivisovellukseksi, joka pääsee käsiksi laitteen ominaisuuksiin. Phonegap on yksi esimerkki kääntöohjelmasta, jolla voidaan tehdä hybridisovelluksia. Hammondsin (2012) mukaan, hybridisovellus poistaa joitain HTML5-sovelluksen rajoitteita, kuten esimerkiksi yhteydettömän tallennuksen ja antaa mahdollisuuden käyttää laitteen ominaisuuksia esim. kameraa.

### 5.4 PhoneGap

PhoneGap on vapaaseen lähdekoodiin perustuva sovelluskehys, joka antaa verkkosivujen kehittäjälle mahdollisuuden käyttää tuttuja verkko-ohjelmointikieliä ja kehittääkseen niillä natiivisovelluksia mobiililaitteisiin esimerkiksi Androidille. (Lunny 2011, 16.) Lyhyesti sanottuna PhoneGap auttaa kääntämään verkkosivun mobiilisovellukseksi ilman, että kehittäjän tarvitsee osata mitään mobiilialustoille suunnattua ohjelmointikieltä. Se myös antaa hyödyntää yksittäisen mobiilipuhelimen ominaisuuksia esimerkiksi kameraa tai paikkatietoja. (La Counte 2011, 25.) Myös Rodger (2011, 76) toteaa kirjassaan, että nämä hybridisovellukset tukevat mobiilipuhelimen ominaisuuksia.

PhoneGap siis toteuttaa hybridisovelluksia, jotka periaatteessa toimivat tavallisen verkkosovelluksen ja natiivin mobiilisovelluksen välissä. Nämä siis toimivat verkkoselaimessa itsessään ja ajavat HTML5 -koodia, jotka sitten hyödyntävät mobiililaitteen ominaisuuksia ja käyttäytyvät natiivisovelluksen tapaan. (Rodger 2011, 7.)

PhoneGapistä löytyy kuitenkin yksi rajoite sillä, jos halutaan kääntää iOS - mobiilialustalle sovellus, niin siihen tarvitaan tietokone, jossa on Mac-käyttöjärjestelmä. Muille, esimerkiksi Windows Phonelle kelpaa PC, missä pyörii Windows -käyttöjärjestelmä. Androidille käy PC, missä on Mac, Windows tai Linux - käyttöjärjestelmä. Finleyn (2012) mukaan Adobe kuitenkin toi ratkaisun tuohon rajoitukseen sillä, että he kehittivät verkkosovellusten kääntämiseen tarkoitetun pilvipalvelun Adobe PhoneGap Buildin syyskuussa 2012.

## **6 TOTEUTUS**

Tämä luku kertoo ohjelmointityön suunnittelusta ja toteutuksesta. Toimeksiantajana toimii Suomen Lentopalloliitto sekä PowerCup, joka on kehittynyt maailman suurimmaksi nuorten ja lasten lentopalloturnaukseksi. Kun sain tietää, että nyt on mahdollista tehdä mobiilisovellus opinnäytetyönä, niin en hetkeäkään epäröinyt, sillä oma kiinnostukseni mobiilisovelluksia kohtaan on suuri. En pidä itseäni hyvänä ohjelmoijana, mutta halusin ottaa työn haasteena ja tätä kautta oppia uusia asioita ja kehittyä ohjelmoijana.

### **6.1 Suunnittelu**

Projekti lähti liikkeelle palaverilla toimeksiantajien kanssa, jossa määriteltiin, mitä tuleva sovellus sisältää. Päätettiin, että valmis mobiilisovellus sisältäisi navigointijärjestelmän, joka näyttäisi peli- ja yöpymispaikat Mikkelissä, jotta kaupunkiin tulevien vierailijoiden olisi helpompaa päästä perille. Bonuksena tekisin vielä tarkemman navigoinnin raviradalta, joka toimii pääpelipaikkana, mutta ongelmaksi saattaa muodostua koordinaattien paikkansa pitävyys.

Lähdin rakentamaan sovellusta ensin pohtimalla, minkälainen rakenne toimisi parhaiten tässä työssä. Suljin heti pois natiivisovelluksen, sillä aika, taidot ja resurssit eivät riittäisi, sillä saman työn tekeminen kolmelle eri alustalle ja kielellä ei ole se kannattavin vaihtoehto. Vaikka natiivilla saisi parhaimman suorituskyvyn sekä sulavimman käyttöliittymän, niin tässä työssä ei tarvitse kuitenkaan käyttää koko puhelimen resursseja. Kamppailu paikasta auringossa on siis hybridisovelluksen sekä selaimessa toimivan HTML5-verkkosovelluksen välillä. Hyötyjä ja rajoitteita on molemmissa rakenteissa. Hybridisovellus toimisi parhaiten tässä työssä, koska sillä päästäisiin kä-

siksi laitteen ominaisuuksiin ja sen kääntäminen tapahtuisi helposti PhoneGapin avulla kaikille laitteille. Rajoitteena on kuitenkin valmiin sovelluksen jakaminen käyttäjille, sillä valmis sovellus pitäisi saada kauppapaikoille esim. Google Playhin, jotta Android käyttäjät saisivat sen ladattua itselleen. Nämä paikat kuitenkin ovat maksullisia, joten ilman rahallista panostusta täytyy tämä vaihtoehto jättää pois. HTML5-sovellus olisi kaikista helpointa toteuttaa, sillä jokainen käyttäjä pääsisi verkkosivuille oman laitteensa avulla. Vaikka HTML5 sisältää paikannusominaisuuden, niin se ei kuitenkaan ole tarpeeksi tarkka, sillä esimerkiksi navigoidessa autolla on tärkeää tietää tarkkaan mistä risteyksestä käännetään.

Kun toimeksiantaja näytti vihreää valoa hybridisovelluksesta aiheutuville kustannuksille, niin ei ollut vaikeaa päättää rakenteen kokoonpanoa. Seuraavaksi oli vuorossa tarvittavien ohjelmien lataaminen, eli tarkemmin PhoneGapin asentaminen. Valmiin työn kääntäminen on mahdollista tehdä kahdella eri tapaa, omalla koneella tai pilvessä. Ajattelin pohtia sitä myöhemmin, sillä tärkeämpää on saada ensimmäinen demo valmiiksi ja sitä kautta koko sovellus valmiiksi.

## 6.2 Työkalujen asentaminen ja projektin luominen

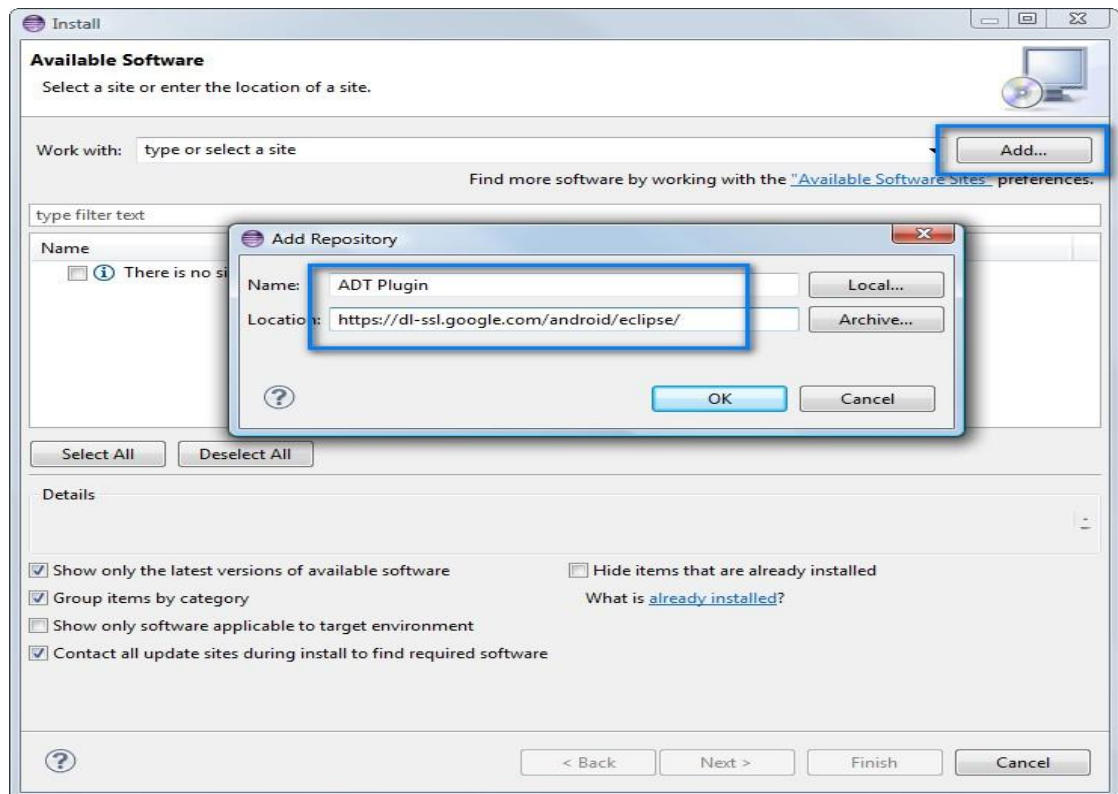
Asennukseen on olemassa englanninkieliset ohjeet PhoneGapin kotisivuilla, mutta saattavat olla hieman sekavat aloittelijalle, jos ei osaa polkuja asettaa, niin päätin kääntää ohjeet suomenkielelle ja vähän tarkentaa vaikeita kohtia. Itsellä oli muutamia ongelmia juuri näiden kanssa. Tämä ohje on tehty käyttäen Windows 7 -käyttöjärjestelmää ja Androidin -alustaa. Tätä ohjetta tehdessä PhoneGap tukee Androidin 2.1 - 4.x -versioita. Seuraavia työkaluja tarvitaan tässä työssä ja niiden asennus ohjeet:

- Eclipse Classic
- Apache Ant
- Java JDK
- Android SDK.

### *Eclipse Classic*

Eclipse Classic on IDE eli kehitysympäristö, joka tukee pääasiassa Javaa, mutta myös muita kieliä. Se voidaan ladata osoitteesta <http://www.eclipse.org/downloads/>. Muiste-

taan valita myös oikea järjestelmälaji (32- tai 64-bittinen). Kun lataus on valmis, puretaan se esimerkiksi kansioon ”C:\eclipse\”. Seuraavaksi ladataan Eclipseen Android Plugin, joka mahdollistaa kätevästi esim. uusien Android projektien luomisen. Avataan seuraavaksi Eclipse ja valitaan ”Help” valikosta ”Install new software..”. Seuraavaksi painetaan ”Add” ja lisätään alla olevan kuvan mukaan tiedot. Kuvassa 8 havainnollistetaan ADT Pluginin lataaminen.



**KUVA 8. Android plug-inin lataaminen (Ricardo Ferreira 2012)**

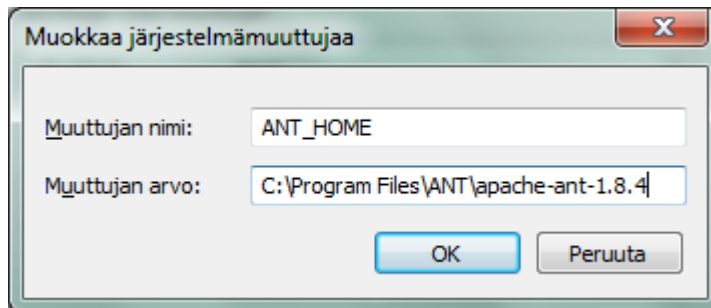
Painetaan seuraavaksi ”OK”, kun ollaan valmiita. Seuraavaksi tarkistetaan, että paketit ovat valittuna ja painetaan ”Next”. Käynnistetään Eclipse uudestaan ja asennuksen pitäisi näin ollen olla valmis.

### *Apache Ant*

Apache Ant on Java -kirjasto ja komentokehotetyökalu, joka helpottaa sovelluksen tekoa. Ladataan se osoitteesta <http://ant.apache.org/bindownload.cgi> ja puretaan se kansioon ”C:\Android\ANT\”. Kopioidaan seuraavaksi sen polku (minulla se on tässä muodossa C:\Android\ANT\apache-ant-1.8.4) ja asetetaan se ANT\_HOME poluksi. Tämä tehdään menemällä ”Käynnistä > Ohjauspaneeli > Järjestelmä ja suojaus > Jär-

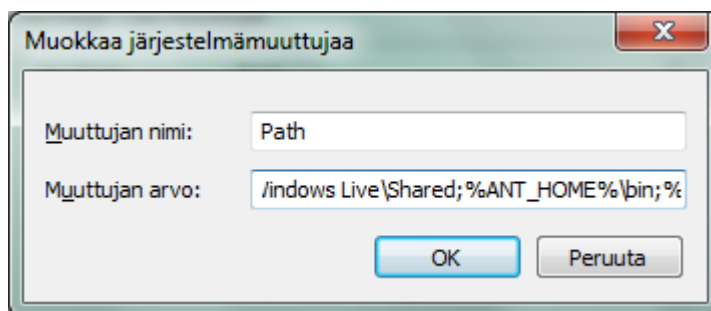
jestelmä > Järjestelmän lisäasetukset > Ympäristömuuttujat”. Sieltä painetaan järjestelmämuuttujista ”Uusi” ja laitetaan muuttujan nimeksi ”ANT\_HOME” ja muuttujan arvoksi kopioitu polku. Tarkistetaan tiedot kuvasta 9.

### KUVA 9. Järjestelmämuuttujan tiedot



Seuraavaksi muokataan muuttujaa ”Path” ja lisätään siihen ”%ANT\_HOME%\bin”. Muuttujat voidaan erotella puolipisteellä ”;” toisistaan. Kuva 10 auttaa havainnollistamaan asian.

### KUVA 10. Polun määrittely



### *Java JDK*

Seuraavaksi ladataan Java JDK. Ladataan tietokonettasi vastaava 32- tai 64-bittinen versio. Ajetaan asennus oletus kansioon, jonka asennusohjelma ehdottaa. Minulla se on ” C:\Program Files\Java\jdk1.7.0\_15”. Nyt tehdään samalla tavalla kuin Apache Ant asennuksen kanssa, eli kopioidaan polku ja asetetaan se muuttujaan JAVA\_HOME ja ”Path” muuttujaan %JAVA\_HOME%\bin.



## *Android SDK*

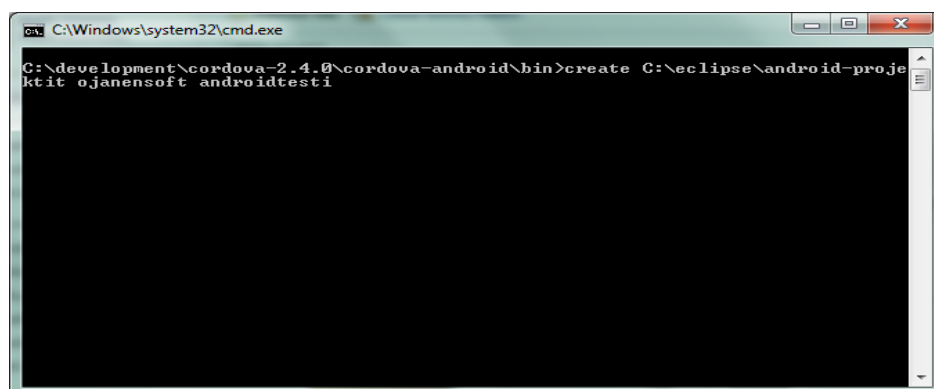
Ladataan seuraavaksi Android SDK ja puretaan se kansioon ”C:\Android\”. Kopioidaan polku ja tehdään uusi muuttuja ”ANDROID\_HOME”, niin kuin Apache Ant esimerkissä. Tällä kertaa ”Path” muuttujaan lisätään kaksi kohtaa ”%ANDROID\_HOME%\platform-tools” sekä ”%ANDROID\_HOME%\tools”.

### **6.2.1 PhoneGapin lataaminen ja projektin luominen**

Ladataan Apache Cordova ja puretaan se kansioon ”C:\development\”. Sitten puretaan kansista ”cordova-android” -niminen paketti samaan kansioon. Nyt pitäisi olla ”cordova-android” -niminen kansio, jonka sisältä löytyy ”bin” -kansio. Painetaan seuraavaksi ”bin” kansion päällä Shift-näppäin pohjassa hiiren kakkosnäppäintä ja valitaan ”Avaa komentoikkuna tähän”.

Uuden projektin voidaan luoda ”create” -komennolla, johon määritellään kansio mihin projekti tulee, yhtiön nimi ja projektin nimi. Yhtiön nimi täytyy eritellä vähintään keran pisteellä, muuten tulee virheilmoitus. Esimerkiksi itse tein komennolla ”create C:\eclipse\android-projektit ojanen.soft androidtesti”. Kansiota ei pidä tehdä itse etukäteen valmiiksi, vaan ”create” -komento luo itse sen esim. tässä tapauksessa ”android-projektit” kansioon. Muuten tulee virheilmoitus, että projekti on jo olemassa. Kuvasta 11 nähdään esimerkki projektin luomisesta.

### **KUVA 11. Projektin luominen komentokehottessa**



Avataan seuraavaksi Eclipse ja paina CTRL + N ja valitaan ”Android project from existing code > polku android-projektit kansioon > valitaan äsken tehty projekti “ ja

painetaan “finish”. Jos Android SDK Manager käynnistyy Eclipsen käynnistyksen yhteydessä, niin annetaan sen ladata puuttuvat tiedostot.

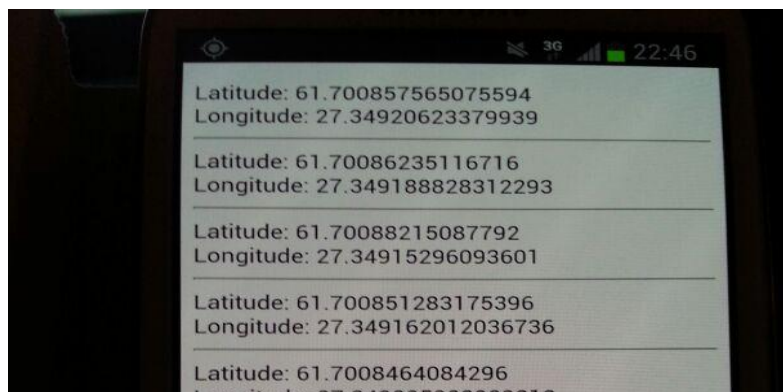
### 6.3 Ohjelmointi

Ohjelmointi osuus alkoi tutkimalla PhoneGap -koodin toimivuutta ja tarkemmin kuinka saada GPS -ominaisuus älypuhelimesta toimimaan. PhoneGapin kotisivuilta löytyneen oppaan mukaan lähdin hakemaan ensi tuntumaan työhöni, josta löytyi paikannuksesta oma lukunsa. Sieltä löytyi kolme eri metodia, joita voi käyttää:

- `geolocation.getCurrentPosition`
- `geolocation.watchPosition`
- `geolocation.clearWatch`.

Näistä kolmesta `geolocation.watchPosition` -metodi täytti tarpeeni täydellisesti. Se paikantaa sijaintisi ja seuraa sitä myös ja päivittää aina muutoksen tullen. `Geolocation.getCurrentPosition` -metodi paikantaa sijainnin, mutta ei päivitä sitä muutoksen ja `geolocation.clearWatch` -metodi taas pysäyttää koko paikannuksen. Oppaasta löytyneen esimerkkiä käyttäen lähdin kokeilemaan metodin tarkkuutta ja muutenkin tutustumaan koodiin. Tuloksena syntyi toimiva ja oikein tarkka ohjelma. Testi alustana toimi Samsung Galaxy S3 -älypuhelin.

#### KUVA 12. Koodin ja paikannuksen testausta

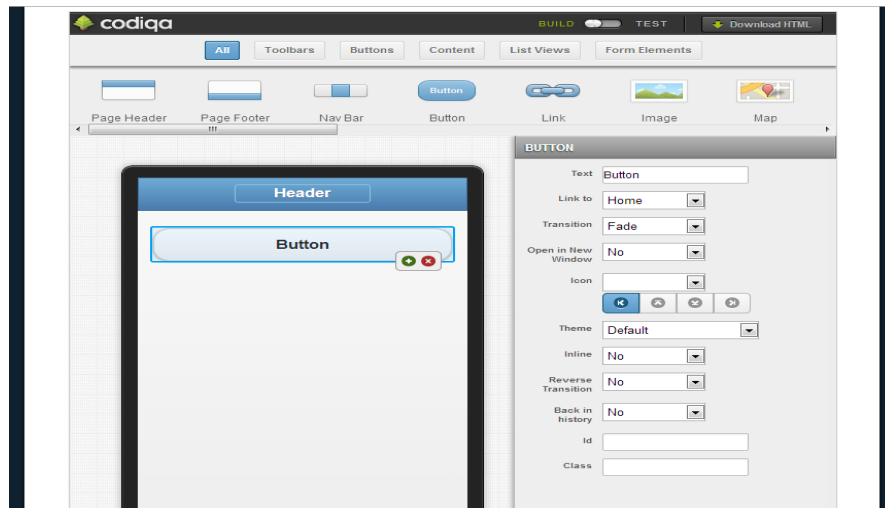


Kuten kuvasta 12 nähdään, niin ohjelma antoi minulle koordinaatteja, jotka sitten syötin Google Maps -verkkopalveluun ja tuloksena olivat noin muutaman metrin heiton tekevät koordinaatit. Tulokseen olin erittäin tyytyväinen, sillä nyt valmis työ tulisi

olemaan tarpeeksi tarkka ainakin Android -alustalla. Myöhemmässä vaiheessa tullaan näkemään myös muiden alustoiden ja laitteiden tuomat tarkkuus heitot.

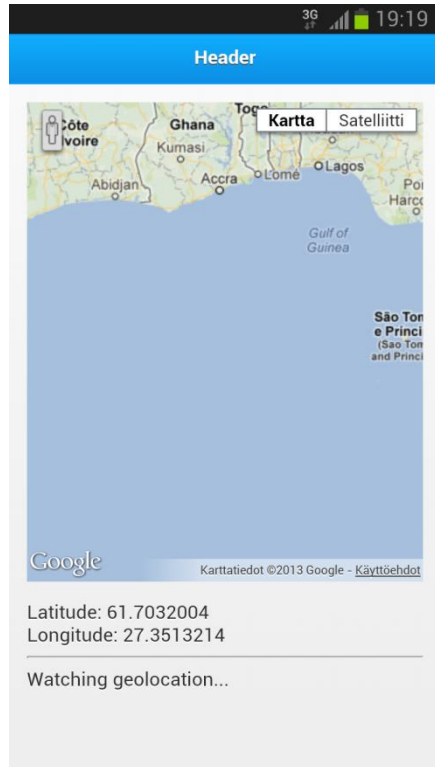
Nyt kun on huomattu ja kokeiltu koordinaattien toimivuus, niin seuraavaksi tehdään tulevaan työhön käyttöliittymä. Se onnistui helposti jQuery Mobilen kotisivuilta löytyvän Codiqa -sovelluksen avulla, jossa voidaan tehdä ”drag and drop” menetelmällä halutun näköinen pohja. Mahdollista on siis laittaa esimerkiksi painikkeita ja taulukoi- ta haluamansa muodon sekä koon mukaan, sekä valita koko teemalla haluttu värimaailma. Valmis teema sitten ladataan itselle, jonka tiedostot linkitetään omaan työhön, jonka jälkeen voidaan aloittaa lopullinen ohjelmointi. Kuvassa 13 nähdään esimerkki miten Codiqa -sovellus toimii.

### KUVA 13. Codiqa esimerkki



Käyttöliittymän halusin olevan mahdollisimman yksinkertainen, joten pääsivulla ei olisi kartan lisäksi muuta kuin valikko, jossa on kaikki kohteet, missä pelataan ja majoituspaikat. Kohde kun on valittu, kartta näyttää autolla ajettavan reitin kohteeseen. Kuten kuvasta 14 nähdään, niin päänäkymä tulisi olemaan tämän näköinen. Kartan alapuolella tällä hetkellä näkyvien koordinaattien tilalle tulee kohdevalikko. Tietenkin paikannus tulee olemaan Mikkelin kohdalla.

## KUVA 14. Alustava suunnitelma päänäkymän käyttöliittymästä



Seuraavaksi huomasin ongelman testaamisessa, sillä mitenkä voidaan testata Android-simulaattorissa GPS -ominaisuutta. Pienen tutkimisen jälkeen sen pitäisi olla mahdollista, jos syöttää koordinaatit manuaalisesti, mutta totesin olevan helpompaa ajaa sovellus älypuhelimessa, vaikka se saattaa tuottaa ylimääräistä työtä, sillä joka kerta joudutaan sovellus lataamaan älypuhelimeen, kun halutaan testaa pientäkin koodin muutosta. Se ainakin välillä turhautti, sillä kun sovellus ei kymmenennenkään testauskerran jälkeen toiminut haluamalla tavalla. Kuitenkin tämä tie on mielestäni varmin tapa nähdä toimivuus.

Seuraava askel kohti valmista työtä oli onnistuminen jatkuvan paikantamisen kanssa. Tämän vaiheen kanssa oli pitkään ongelmia, mutta tämä kuitenkin pitkälti keskittyi tuolin ja näppäimistön väliin. Oma osaaminen oli siis kovilla tässä vaiheessa, mutta kuitenkin tästäkin ongelmasta selvittiin voittajana.

## KUVA 15. Navigointia varten tarvittava koodi

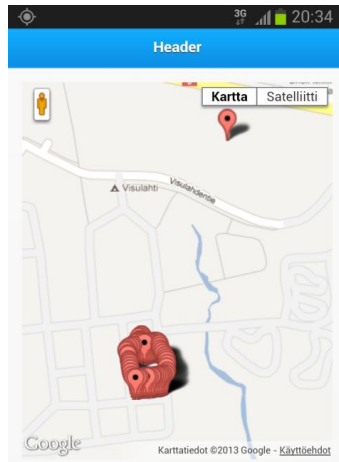
```

60 | $(document).ready(function() {
61 |   $('#map_canvas').gmap().bind('init', function(evt, map) {
62 |     navigator.geolocation.watchPosition( function(position) {
63 |       var clientPosition = new google.maps.LatLng(position.coords.latitude, position.coords.longitude);

```

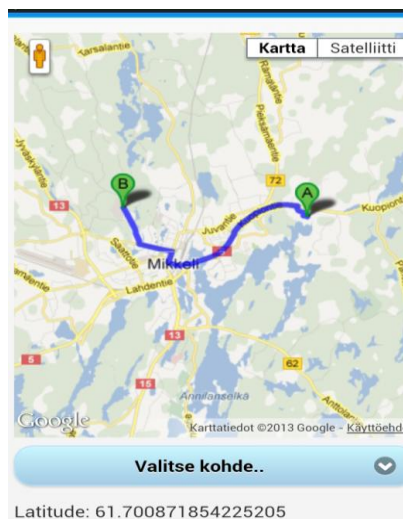
Kuvasta 15 nähdään koodirivejä työstä, jotka tarvitaan koordinaattien hakemiseen ja piirtämiseen kartalle. Rivillä 62 - 63 olevalla koodilla haetaan koordinaatit käyttäen älypuhelimien GPS -ominaisuutta ja tallennetaan ne ”clientPosition” muuttujaan. Muuttujaa käytetään myöhemmässä vaiheessa piirtämään koordinaatit kartalle. Kuten kuvasta 16 nähdään pienen kävely retken näkymä. Myös huomataan, että sovellus jättää vanhat koordinaattien paikat näkyviin, mutta tämän ei pitäisi olla iso ongelma.

### KUVA 16. Navigoinnin toimivuus



Seuraava askel oli saada tehtyä reittiohjaus käyttäjän sekä haluamansa kohteen välille. DisplayDirections metodilla voidaan tehdä Googlelle pyyntö, jolla saadaan haluttujen pisteiden välille reittihaku. Samalla ratkesi myös ongelma merkkapisteiden kanssa, sillä koodia piti muuttaa juuri tällä kohdalla. Ongelman ratkaisun löysin Googlen omista ohjeista. Kuten kuvasta 17 nähdään esimerkki reitin näyttämisen Visulahdesta Kalevankaan raviradalle.

### KUVA 17. Esimerkki reitin näyttäminen



Tämän toiminnan onnistumisen jälkeen on seuraavaksi saatava valikko toimimaan kartan kanssa, joissa olevien kohteiden koordinaattien avulla saataisiin tehtyä uusi reittiohje. Täytyy muistaa huomioida uuden reitin laskeminen aina kohteen vaihtuessa. Myös pientä hienosäätöä on kartan päivittämisen kanssa, sillä nyt sovellus keskittää itsensä aina uudestaan tietylle zoomaus tasolle. Näiden asioiden ruutuun saamisen jälkeen alkaa olla lähellä ensimmäistä julkaisu valmista versiota.

Zoomaus ongelma ratkesi siten, että ensin piti estää ”preserveViewport” metodilla ”displayDirection” metodin oman kartan keskittämisen sekä zoomauksen. Sen jälkeen määritetään itse haluama zoomaustaso sekä keskityksen kohta. Kuvassa 18 nähdään ”preserveViewport” metodin tehtävä käytännössä ja riveillä 73-74 tapahtuu kartan keskittäminen käyttäjän tämän hetkiseen sijaintiin ja zoomaustaso.

### KUVA 18. Näkymä koodista

```

71  'travelMode': google.maps.DirectionsTravelMode.DRIVING }, {'preserveViewport':true}, function(success, result) {
72      var center = new google.maps.LatLng(position.coords.latitude, position.coords.longitude);
73      $('#map_canvas').gmap('option', 'center', center);
74      $('#map_canvas').gmap('option', 'zoom', 14);
75      $('#map_canvas').gmap('refresh');

```

Seuraavaksi oli vuorossa tehdä valikko, missä löytyy kaikki toimeksiantajan osoitteet, mihinkä käyttäjä halutaan ohjata. Valikko tulee kartan alapuolelle, josta käyttäjä valitsee haluamansa kohteen ilman, että hänen täytyisi tietää kohteen osoitetta. Tämä helpottaa suunnattomasti käyttäjän navigointia, sillä paikat löytyvät nyt suoraan sovelluksesta. Ongelma kohdaksi tuli uuden reitin päivittäminen, sillä kartta ei päivittänyt uutta reittiä, kun valikosta valittiin uusi kohde. Tämä ratkesi kuitenkin hieman eri tavalla mitä odotin. Kohteiden koordinaatteja sekä niiden paikkaansa pitävyyttä sovelluksessa testattiin pelkästään tietokoneen selaimessa, sillä ei ollut järkevää joka kohteen lisäämisen jälkeen ajaa sitä älypuhelimessa. Monen tunnin yrittämisen jälkeen saada ongelma ratkaistuksi, päädyin kokeilemaan sitä älypuhelimessa ja sovellus toimi aivan täydellisesti.

Tässä vaiheessa sovellus toimii toiminnallisuuden puolesta hyvin, mutta pientä hienosäätöä on vielä ulkoasun kanssa. Lisäsin vielä sovelluksen ulkoasuun ominaisuuden, että kun käyttäjä saapuu etusivulle, niin ruutuun ilmestyy Powercupin logo sekä käyttöohjeet, joissa kehoitetaan laittamaan GPS päälle ennen navigoinnin aloittamista. Seuraavaksi on jäljellä vain sovelluksen kääntäminen iOS, Android sekä Windows Phone

-alustoille. Vaihtoehtoina olisi ladata kaikkien alustoiden työkalut, joita sovelluksen kääntäminen vaatisi tai käyttää Adoben pilvipalvelua, jolla kätevästi saa käännettyä kaikille alustoille. Valitsin tämän Adoben pilvipalvelun, sillä nyt ei tarvitse ladata monia eri työkaluja. Adobe PhoneGap Build toimii siten, että ensin rekisteröidytään ja sitten syötetään oman sovelluksen koodi joko zip -tiedostona tai GitHubin kautta. Myös config.xml tiedosto luodaan itse ennen käännöksen aloitusta samaan paikkaan, missä index.html sijaitsee. Siihen laitetaan esimerkiksi tulevan iOS- sekä Windows Phonen -sovellusten kuvakkeiden sijainti, joita ilman sovellusta ei hyväksytä. Lopuksi syötin koodin zip -tiedostona, joka osoitettiin helpoimmaksi vaihtoehdoksi.

### KUVA 19. Adoben pilvipalvelu



Kuvasta 19 nähdään valmiit käännetyt sovellukset eri alustoille, jotka eivät ole täysin julkaisukelpoisia vielä. Androidille ja iOSille julkaistaessa vaaditaan pientä hienosäätöä, mutta Windows Phonelle julkaistaessa ei tarvitse muuta kuin käydä syöttämässä käännetty sovellus Windows Storeen.

Androidilla tarvitaan tehdä vielä keystore -tiedosto, joka yhdistää henkilön tai yrityksen kyseiseen sovellukseen ja validiteetin, joka täytyy olla yli 25 -vuotta pitkä kesto- taan. Tämä keystore-tiedosto syötetään Adoben pilvipalvelun käännösvaiheessa. IO- Silla puolestaan tarvitaan MAC-tietokoneesta saatava avainkoodi.

## 6.4 Julkaisu

Julkaisu tapahtuu laittamalla sovellus Androidin, Windows Phonen sekä Applen so- velluskauppoihin, joista käyttäjät voivat omille laitteilleen käydä lataamassa valmiin sovelluksen. Tämä kuitenkin oli alussa pieni ongelma, sillä tätä asiaa juuri mietittiin kovasti opinnäytetyöni ohjaajani kanssa. Sillä jos sovellus julkaistaan näissä kolmessa sovelluskaupassa, niin siitä syntyy kustannuksia n. 175 € euron verran. Jos valmis

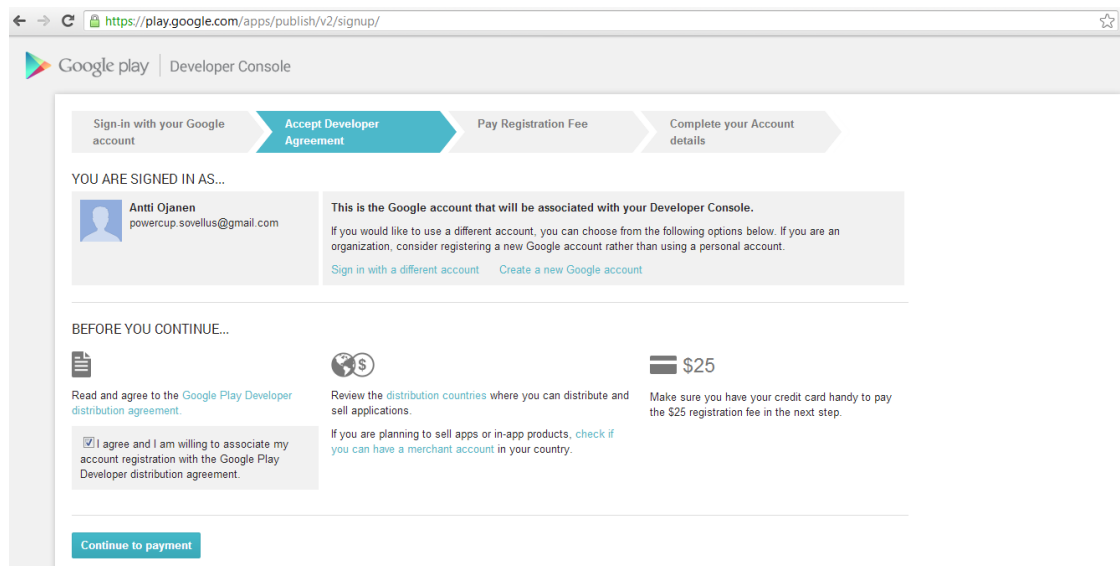
sovellus toimisi HTML5 -sovelluksena, niin silloin sen jakaminen olisi ilmaista ja helppoa, sillä jokainen kävijä menisi vaan sovelluksen verkko-osoitteeseen ja näin ollen pystyisi käyttämään sitä sen kautta. Mutta näiden kahden kehitystavan välillä olikin ratkaisevinta sen toiminnallisuus, joka kallistui hybridisovelluksen puolelle.

Keskustelimme toimeksiantajien kanssa tästä kustannusongelmasta, niin tulimme siihen tulokseen, että ne tullaan julkaisemaan näissä sovelluskaupoissa ja kustannuksista vastaa toimeksiantajat. Tämä oli hyvä asia varsinkin käyttäjien puolesta, sillä he saavat nyt toiminnallisuudeltaan paremman hybridisovelluksen, kuin HTML5-sovelluksen, joka kärsii paikannuksen epätarkkuudesta.

### 6.4.1 Google play

Google play julkaisee kaikki Android -pohjaiset sovellukset. Sinne rekisteröityminen tapahtuu tekemällä ensin Google-tilin, jonka sähköpostiosoitetta voidaan käyttää rekisteröitymiseen. Google playn maksu on 25\$ dollaria, joka on näistä kolmesta kaikista halvin. Kuvasta 20 nähdään yksi rekisteröitymisen vaiheista.

#### KUVA 20. Näkymä Google Playn rekisteröitymisestä



Tein tätä sovellusta varten uuden sähköpostin tulevaisuutta varten, jos esimerkiksi ensi vuonna halutaan käyttää samaa sovellusta, niin voidaan käyttää tätä samaa tiliä sen julkaisemiseen. Kuvan 19 näkymän jälkeen suoritetaan tämä 25\$ dollarin vuosit-

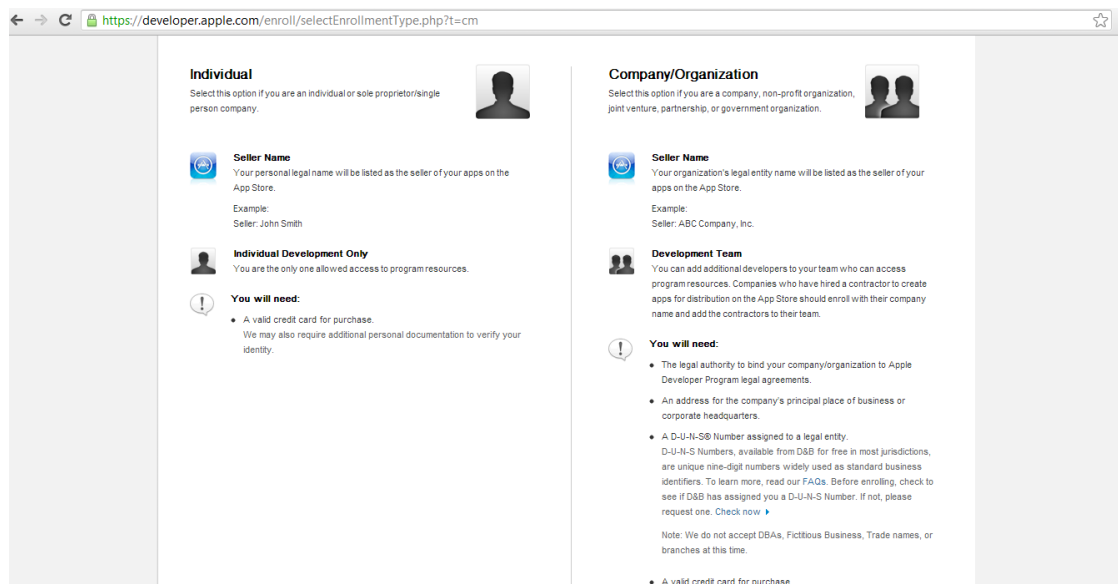


tainen maksu, joka antaa valtuuden julkaista omia sovelluksia Google Playssa. Tämän jälkeen onkin sitten mahdollista lisätä oma sovellus Google Play -sovelluskauppaan. Adobe PhoneGap Build -pilvipalvelu antaa apk -tiedoston, joka ladataan Google Playn palveluun. Sen jälkeen syötetään sovelluksen tietoja ja ruudunkaappauksia. Kun on valmista, niin sovellus tulee muutaman tunnin sisällä käyttäjien ulottuville.

## 6.4.2 Apple App Store

Applen App store julkaisee iOS -pohjaiset sovellukset Applen laitteille. Rekisteröityminen suoritetaan ensin tekemällä Apple ID, jonka tunnuksena toimii sähköpostiosoite, jota käytin aikaisemmin Google Playn yhteydessä. Rekisteröinnin yhteydessä pyydetään ilmoittaa, että onko kyseessä yksityinen henkilö vai yritys, joka haluaa julkaista sovelluksia. Periaatteessa sovelluksen julkaisevat toimeksiantajani, eli Suomen Lentopalloliitto sekä Powercup, mutta jos sen vaihtoehdon valitsee, niin silloin tarvitaan erilaisia todistuksia, osoitteita. Helpommaksi vaihtoehdoksi havaitsin yksityishenkilö-vaihtoehdon.

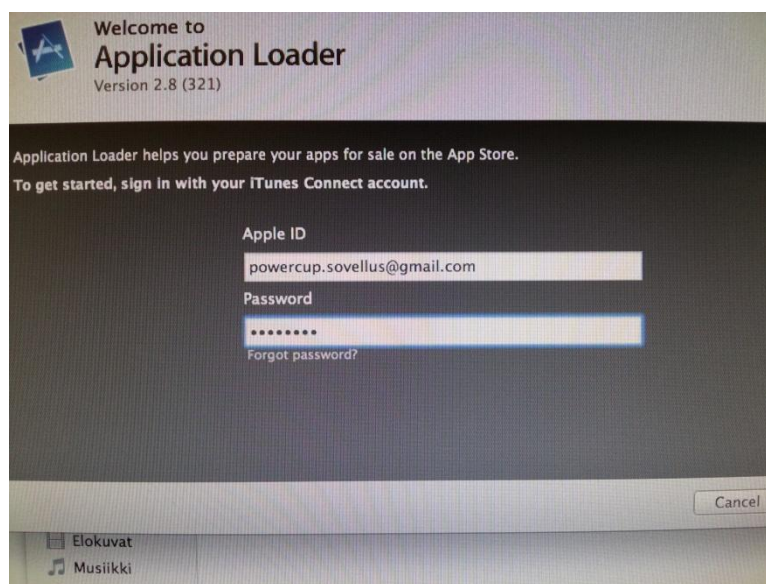
### KUVA 21. Rekisteröinti valinta näkymä



Kuvassa 21 on havainnollistettu vaihtoehtojen ehdot. Apple kehittäjän maksu on 99\$ dollaria vuodessa. Onnistuneen rekisteröitymisen jälkeen joutuu odottamaan 24 tuntia ennen kuin oston hyväksyminen tulee sähköpostiin. Odottelun jälkeen tulee aktivointikoodi, jonka syöttämällä omalta tililtä löytyvään aktivointilomakkeeseen saadaan tili näin ollen julkaisu valmiiksi.

Jotta sovellus saadaan käyttäjille ladattavaksi, tulee tehdä muutama asia mitä en olisi aivan heti arvannut tulevan. Ensinnäkin tarvitaan kuitenkin se Applen MAC - tietokone, jotta voidaan tehdä tarvittavat sertifikaatit sekä avaimet. Omalta Apple -tililtä käydään luomassa sertifikaattia sekä sopimusehdot, jotka syötetään Adobe Phonegap Builder -pilvipalveluun samalla, kun sovellus käännetään iOS -alustalle. Sovelluksen julkaisemista varten tarvitaan oma sovellus nimeltään Application Loader, jolla itse sovellus lähetetään App storeen.

## KUVA 22. Application Loaderiin kirjautuminen



Kuvasta 22 nähdään esimerkki kirjautuminen Application Loader palveluun. Sovelluksen tiedot syötetään erikseen Apple -tililtä löytyvään iTunes Connect -palveluun. Tämä vaihe oli monimutkaisempi, kuin millään muulla käyttämälläni sovelluskaupalla.

### 6.4.3 Windows Store apps

Windowsin Phone alustalle sovellukset julkaistaan sille tehdyille Windows Store app -nimisessä sovelluskaupassa. Rekisteröinti tapahtuu tekemällä oma Windows ID, jonka tunnuksena toimii sähköposti, jota olen käyttänyt muissakin sovelluskaupoissa. Jos halutaan julkaista sovelluksia Windows Phonelle, niin täytyy myös tänne maksaa kehittäjänmaksu, joka on myös 99\$ dollaria vuodessa. Rekisteröinnissä kysytään toimii-

ko kehittäjänä yritys vai yksityishenkilö. Tähän valitsin samalla tavalla, kuin Applen App Storessa, eli yksityishenkilö.

Windows Phonella on myös mahdollisuus rekisteröityä DreamSpark tilillä ilmaiseksi, joka on suunnattu opiskelijoille. Kuitenkaan en julkaise tätä sovellusta yksin, sillä mukana ovat myös toimeksiantajat, joten en siksi valitse tätä vaihtoehtoa.

### KUVA 23. Valinta näkymä rekisteröinnissä

Get started

Welcome to the Windows Phone Dev Center. Your subscription gives you the ability to publish apps in the Windows Phone Store.

To become a member you need a valid credit card, PayPal account, or promo code. Students with a verified DreamSpark account can register for free.

You'll need to provide your contact info, and if you're registering as a company we'll also need your legal entity name.

[Learn more](#) about registration.

Country/region of residence or business\*

Finland

Account type\*

Company  
Select this option if you want to sell apps as a business or government organization.

Individual or student  
Select this option if you want to sell apps as a sole proprietor or using your DreamSpark student account.

Legal Terms\*

By selecting this box, I agree to be bound by the Windows Phone Store Application Provider Agreement. Further, if accepting on behalf of a company, then I represent that I am authorized to act on my company's behalf.

Kuten kuvassa 23 voidaan nähdä, että valita voidaan joko yritys tai yksityishenkilön käyttäjätili. Jos halutaan, että oma sovellus on maksullinen, niin silloin täytyy tehdä erillinen veroprofiili. Nyt tili on valmis ja näin ollen voidaan jatkossa julkaista omia sovelluksia.

Windows Storeen sovelluksen julkaisu onnistuu siten, että kehittäjältä kysytään sovellukseen liittyviä tietoja ja xap -tiedosto, joka sisältää itse sovelluksen, sekä siihen liittyviä tietoja esimerkiksi missä maassa kyseinen sovellus julkaistaan tai määritellään sen hinta. Kuten kuvasta 24 voidaan havaita, että mitä eri tietoja kysytään lähettäessä sovellusta. Valitsin vain Suomen Windows Storen, sillä ei ole järkeä laittaa muiden maiden kauppapaikkaa, koska ulkomaalaisia maita ei ole tulossa turnaukseen.


## KUVA 24. Valmiin sovelluksen lähettäminen

App

Only show changes I made

Field	Published info	New info	
Category	-	travel + navigation	
Subcategory	-	navigation	
Publish as hidden	-	No	
Allow trial downloads	-	No	
Base price	-	0.00 EUR	<a href="#">Edit</a>
Markets	-	1 total	<a href="#">Edit</a>

XAPs

XAP name	Version	OS	Resolution	Language
 Powercup.xap	1.0.0.0	7.1	WVGA	English

Sovelluksen lähettämisen jälkeen tarvitsee vain odottaa sen hyväksyntä, jossa saattaa mennä muutama päivä, sillä kaikki sovellukset tarkistetaan, etteivät ne sisällä mitään epäasianmukaista materiaalia.

## 7 POHDINTAA

Työtä tehdessä tuli esille monta eri kysymystä, mutta myös niille vastauksia. Huomasin tutkiessani eri kehitysvaihtoehtoja, että jokaisella vaihtoehdolla on omat vahvuutensa sekä heikkoutensa. Esimerkiksi, jos verrataan hybridisovellusta sekä HTML5-sovellusta, niin ainakin tässä työssä sovelluksen tärkeimmällä ominaisuudella eli paikantamisen tarkkuudella oli merkittävä ero. Hybridisovellus paikansi n. muutaman metrin tarkkuudella, kun taas HTML5-sovelluksenta heitto saattoi olla satoja metrejä. Tässä tapauksessa natiivi- ja hybridisovellus olivat yhtä tarkkoja, mutta hybridisovellus taas oli huomattavasti helpompi tapa tehdä. Syy näihin eroavaisuuksiin löytyy siitä, että HTML5-sovellus käyttää paikantamiseen tietokoneen IP -osoitetta sekä Wi-Fi -asemia. Esimerkiksi älypuhelimissa paikantamiseen käytetään vielä lisäksi radiomasoja. Hybridi- ja natiivisovellukset käyttävät laitteiden GPS -ominaisuutta hyväksi, jolloin saadaan se ratkaiseva tarkempi tulos.

Toinen iso asia mikä herätti paljon kysymyksiä, oli sovellusten julkaiseminen. Siinä missä HTML5-sovellus oli ylivoimainen näihin kahteen muuhun verrattuna, on juuri-kin tässä kategoriassa, sillä HTML5-sovelluksen käyttö tapahtuisi puhelimen selaimessa eikä sitä tarvitsisi julkaista missään sovelluskaupassa. Myös sovelluksen vi-

kojen korjaaminen ja päivittäminen onnistuu helpommin, sillä kun tehdään päivitys, päivittyy se myös jokaiselle käyttäjälle samanaikaisesti. Myös eri alustojen sovelluskauppojen byrokratiassa on isoja eroja. Esimerkiksi Google Playn odotusaika on noin muutama tunti, kun taas Applella saattaa mennä jopa viikkoja. Muutenkin Applen toiminta on äärimmäisen tiukkaa, sillä täytyy kaikenlaisia sertifikaatteja tehdä ja muuta toimintaa, mikä toki selittyy tiukalla tietoturvalla. Tästä esimerkkinä sovellukseni hylkääminen Applelta, jonka syynä oli sen yksinkertaisuus. Sovelluksen olisi pitänyt olla monipuolisempi ja viihdyttävämpi, vaikka he sanoivat pitävän viestissään yksinkertaisuudesta.

Itse ohjelmointiin että työn lopputulokseen olen ihan tyytyväinen. En saanut ihan kaivattua loppu tulosta, sillä omat taidot tulivat vastaan. Mutta Google Maps JavaScript API:n toimivuuteen olen tosi tyytyväinen. Sitä on helppo käyttää ja mahdollistaa todella näyttävienkin karttojen teon. Sovelluksen toimivuutta ei JavaScript -pohjainen rajapinta häirinnyt. Havaitsin myös PhoneGapin olevan kattavampi, mitä aluksi luulin, sillä alussa olin siinä uskossa, ettei sillä pääse käsiksi esimerkiksi puhelin kameraan.

Tulevaisuudessa kehittäisin sovellusta hieman käytännöllisemmäksi esimerkiksi valikkojen ja muiden pikanäppäinten luomisella. Esimerkiksi paluu näppäin etusivulle olisi hyvä uudistus. Myös olisi hyvä saada reitti ohjeet ihan tekstinä esimerkiksi kartan alapuolelle. Eikä olisi poissa suljettua myös mahdollinen ääniohjaus. Myös jos Applen app storeen halutaan saada julkaistua, niin sovelluksesta täytyy saada rikkaampi kokemus.

## LÄHTEET

Adarsh 2013. WWW-dokumentti. Which Mobile Platform to Develop For. Päivitetty 25.1.2013. Luettu 15.2.2013.

Android. 2013 Wikipedia. WWW-dokumentti.  
[http://en.wikipedia.org/wiki/Android\\_%28operating\\_system%29#cite\\_note-AndroidAnnouncement-10](http://en.wikipedia.org/wiki/Android_%28operating_system%29#cite_note-AndroidAnnouncement-10). Päivitetty 14.2.2013. Luettu 16.2.2013.

Android - System architecture 2007. Keskusteluryhmän artikkeli.  
<http://www.anddev.org/open-news-f1/android-system-architecture-in-words-t7.html>.  
Päivitetty 15.2.2007. Luettu 18.2.2013.

Android version history 2013. Wikipedia. WWW-dokumentti.  
[http://en.wikipedia.org/wiki/Android\\_version\\_history#cite\\_note-93](http://en.wikipedia.org/wiki/Android_version_history#cite_note-93). Päivitetty:  
18.2.2013. Luettu: 20.2.2013.

Bada Developers 2011. The basic Architecture and UI of bada. WWW-dokumentti.  
<http://developer.bada.com/article/The-Basic-Architecture-and-UI-comparisons-between-bada-and-iOS>. Päivitetty 23.8.2011. Luettu 20.2.2012.

Bai, Giulio 2011. jQuery Mobile First Look. Olton Birmingham: Packt Publishing Ltd

Blurred Out: 51 Things You Aren't Allowed to See on Google Maps 2008. WWW-dokumentti.  
<http://www.itsecurity.com/features/51-things-not-on-google-maps-071508/>. Päivityksestä ei tietoa. Luettu 23.2.2013.

Cascading Style Sheets 2013. Wikipedia. WWW-dokumentti.  
[http://en.wikipedia.org/wiki/Cascading\\_Style\\_Sheets](http://en.wikipedia.org/wiki/Cascading_Style_Sheets). Päivitetty 2.2.2013. Luettu 6.2.2013.

Devilla, Joey 2013. IDC's Smartphone Stats for 4Q 2012, and a Review of Their Mobile OS Share Prediction for 2015. <http://www.globalnerdy.com/2013/02/14/idcs-smartphone-stats-for-4q-2012-and-a-review-of-their-mobile-os-share-prediction-for-2015/>. Päivitetty 14.2.2013. Luettu 18.2.2013.

DuPont, Ben 2012. Options for MobileApp Development. Information Week. PDF-dokumentti.<http://reports.informationweek.com/...Options-for-Mobile-App-Development.html>... Päivitetty 3.3.2012. Luettu 25.2.2013.

Ferreira, Ricardo 2012. Clean Code Development - Quality Seal. WWW-dokumentti. <http://cleancodedevelopment-qualityseal.blogspot.fi/2012/11/how-to-install-phonegap-for-android.html>. Päivitetty: 11.11.2012. Luettu 6.3.2013.

Finley, Klint 2012. Adobe Launches Hosted PhoneGap Build Service For Creating Cross-Platform Mobile Apps. WWW-dokumentti. <http://techcrunch.com/2012/09/24/adobe-launches-hosted-phonegap-build-service-for-creating-cross-platform-mobile-apps/>. Päivitetty. 24.9.2012. Luettu 12.2.2013.

Gautham A. S. 2012. Google Revises Their Map, Adds Offline Version and 3D imaging. WWW-dokumentti. <http://www.techgau.org/2012/06/google-revises-their-map-adds-offline.html>. Päivityksestä ei tietoa. Luettu 23.2.2013.

Garside, Juliette 2012. Apple Maps service loses train stations, shrinks tower and creates new airport. WWW-dokumentti. <http://www.guardian.co.uk/technology/2012/sep/20/apple-maps-ios6-station-tower>. Päivitetty 20.9.2012. Luettu 23.2.2013.

Google Maps JavaScript API v3 2013. WWW-dokumentti. <https://developers.google.com/maps/documentation/javascript/events>. Päivitetty 14.3.2013. Luettu 20.3.2013.

Google Maps SDK for iOS 2013. WWW-dokumentti. <https://developers.google.com/maps/documentation/ios/>. Päivitetty 21.2.2013. Luettu 20.3.2013.

Google Maps Mobiililaitteille 2013. WWW-dokumentti. <http://www.google.com/mobile/maps/>. Päivityksestä ei tietoa. Luettu 23.2.2013.

Google Maps 2013. Wikipedia. WWW-dokumentti.

[http://en.wikipedia.org/wiki/Google\\_Maps\\_%28application%29](http://en.wikipedia.org/wiki/Google_Maps_%28application%29). Päivitetty: 7.2.2013.  
Luettu 23.2.2013.

Google Maps/Google Earth APIs Terms of Service 2012. Käyttöehdot.  
<https://developers.google.com/maps/terms?hl=fi>. Päivitetty: 13.12.2012. Luettu:  
23.2.2013.

Hammonds, Mark 2012. Mobile App Development Options: Which Way To Go?  
WWW-dokumentti. <http://mobile.tutsplus.com/tutorials/mobile-web-apps/mobile-app-development-options-which-way-to-go/>. Päivitetty: 31.5.2012. Luettu: 25.2.2013.

HTML5 & CSS3 Support. Find me by IP. WWW-dokumentti.  
<http://fmbip.com/litmus/>. Päivityksestä ei ole tietoa. Luettu 6.2.2013.

HTML5 Video 2013. W3Schools. WWW-dokumentti.  
[http://www.w3schools.com/html/html5\\_video.asp](http://www.w3schools.com/html/html5_video.asp). Päivityksestä ei tietoa. Luettu  
5.2.2013.

HTML5 Audio 2013. W3Schools. WWW-dokumentti.  
[http://www.w3schools.com/html/html5\\_audio.asp](http://www.w3schools.com/html/html5_audio.asp). Päivityksestä ei tietoa. Luettu  
5.2.2013.

Humalamäki, Antti 2011. PHP-pohjaiset ohjelmointikehykset. Jyväskylän ammatti-  
korkeakoulu. Tekniikan ja liikenteen ala. Opinnäytetyö.

IOS version history 2013. Wikipedia. WWW-dokumentti.  
[http://en.wikipedia.org/wiki/IOS\\_version\\_history](http://en.wikipedia.org/wiki/IOS_version_history). Päivitetty: 20.2.2013. Luettu  
20.2.2013.

JavaScript 2013. Wikipedia. WWW-dokumentti.  
<http://en.wikipedia.org/wiki/JavaScript>. Päivitetty 8.2.2013. Luettu 8.2.2013.

Karlins, David 2011. Dreamweaver CS5.5 Mobile and Web Development with  
HTML5, CSS3, and JQuery. Birmingham: Packt Publishing Ltd.



Korf, Mario & Oksman Eugene. Developer Force. WWW-dokumentti. Native, HTML5, or Hybrid: Understanding Your Mobile Application Development Options. [http://wiki.developerforce.com/page/Native,\\_HTML5,\\_or\\_Hybrid:\\_Understanding\\_Your\\_Mobile\\_Application\\_Development\\_Options](http://wiki.developerforce.com/page/Native,_HTML5,_or_Hybrid:_Understanding_Your_Mobile_Application_Development_Options). Päivityksestä ei tietoa. Luettu 25.2.2013.

La Counte, Scott 2011. Going Mobile : Developing Apps for Your Library Using Basic HTML Programming. Chigaco: ALA Editions.

Lecrenski, Nick Watson, Karli Fonseca-Ensor, Robert 2011. Beginning Windows Phone 7 Application Development : Building Windows Phone Applications Using Silverlight and XNA. Hoboken New Jersey: Wrox.

Lee, Wei-Meng 2011. Beginning Android Application Development. Hoboken New Jersey: Wrox.

Lowery, Joseph W & Fletcher, Mark 2011. HTML5 24-Hour Trainer. Hoboken New Jersey: Wrox.

Lunny, Andrew 2011. PhoneGap Beginner's Guide. Olton Birmingham: Packt Publishing Ltd.

Makzan 2011. HTML5 Games Development by Example: Beginner's Guide. Olton Birmingham: Packt Publishing Ltd.

Mary Jo Foley 2012. Microsoft's Windows Phone 8: There's good news and bad news. WWW-dokumentti.<http://www.zdnet.com/blog/microsoft/microsofts-windows-phone-8-theres-good-news-and-bad-news/12977>. Päivitetty: 20.6.2012. Luettu 20.2.2013.

Microsoft & Nokia. Windows Phone Guide for Symbian Qt Application Developer. [http://windowsphone.interoperabilitybridges.com/media/58803/windows\\_phone\\_guide\\_for\\_symbian\\_qt\\_application\\_developersformatted.pdf](http://windowsphone.interoperabilitybridges.com/media/58803/windows_phone_guide_for_symbian_qt_application_developersformatted.pdf). Opaskirja. PDF-dokumentti. Päivitetty 16.9.2011. Luettu 18.2.2013.

Morril, Dan 2008. Announcing the Android 1.0 SDK, release 1. WWW-dokumentti. <http://android-developers.blogspot.fi/2008/09/announcing-android-10-sdk-release-1.html>. Päivitetty: 23.9.2008. Luettu: 20.2.2013.

Ortiz, Enrique C, 2011. Introduction to jQuery Mobile. WWW-dokumentti. <http://www.ibm.com/developerworks/library/wa-jqmobile/>. Päivitetty. 1.2.2011. Luettu 8.3.2013.

Plotz, Marc 2012. Introduction To jQuery Mobile. WWW-dokumentti. <http://www.htmlgoodies.com/html5/tutorials/introduction-to-jquery-mobile.html#fbid=ES9jbrwwfuH>. Päivityksestä ei tietoa. Luettu 8.3.2013.

Rodger, Richard 2011. Beginning Mobile Application Development in the Cloud. Hoboken, New Jersey: Wrox

Simonson, Rick 2010. Nokia: Joka toinen mobiilialusta jää henkiin. WWW-dokumentti. <http://www.digitoday.fi/bisnes/2010/01/04/nokia-joka-toinen-mobiilialusta-jaa-henkiin/201069/66>. Päivitetty 4.10.2010. Luettu 13.2.2013.

Wilton, Paul and McPeak, Jeremy 2010. Beginning JavaScript (4<sup>th</sup> Edition). Hoboken, New Jersey: Wrox