KARELIA-AMMATTIKORKEAKOULU
Tietotekniikan koulutusohjelma

Olli Timonen

WINDOWS PHONEN JA HTML5:N SOVELLUSKEHITYS

Opinnäytetyö
Huhtikuu 2013

Karelia AMMATTIKORKEAKOULU

OPINNÄYTETYÖ
Huhtikuu 2013
Tietotekniikan koulutusohjelma

Karjalankatu 3
80200 JOENSUU
p. (013) 260 6800

Tekijä
Olli Timonen

Nimeke
Windows Phonen ja HTML5:n sovelluskehitys

Toimeksiantaja
Ampparit Oy

Tiivistelmä

Tässä opinnäytetyössä kehitettiin Ampparit Oy:lle Ampparit.com-mobiilisovellus Windows Phone -alustalle. Mobiilisovelluksen haluttiin näyttävän jo olemassa oleva Ampparit.com HTML5-sivusto ja saada se toimimaan hyvin yhteen alustan kanssa. Ampparit Oy oli jo tuonut Android-sovelluksen saataville ja nyt oli Windows Phonen vuoro.

Sovellukset luotiin noudattamalla yksinkertaista ohjelmistotuotantotapaa, jossa käytiin läpi neljä vaihetta: suunnittelu, toteutus, testaus ja julkaisu. Sovellukset toteutettiin omilla laitteilla ja ohjelmistoilla. Windows Phone -sovelluskehitys tehtiin Visual Studiolla käyttämällä XAML- ja C#-ohjelmointikieliä. Sovellukset pyrittiin toteuttamaan mahdollisimman yksinkertaisiksi ja muistiasäästäviksi käyttämällä suoraviivaisia ratkaisuja ja käsittelemällä hyvin taustalle menevät prosessit ja komponentit.

Työt saatiin tehtyä hyvissä ajoin ennen eräpäivää ja Ampparit.com-sovellukset löytyvät Windows Phone -kaupasta. Sovelluksesta piti tehdä erilliset ratkaisut Windows Phone 7- ja Windows Phone 8 -käyttöjärjestelmille, koska Windows Phone 7:n heikko tuki HTML5-sovelluksia kohtaan aiheutti ongelmia. Toimeksiannon ohessa dokumentoitiin yksinkertaiset ohjeet HTML5-sivuston liittämiseen Windows Phone -sovellukseen.

Kieli
suomi

Sivuja 31

Asiasanat
Mobiilipalvelut, Ohjelmistot, WWW-sivustot, HTML

Karelia
UNIVERSITY OF APPLIED SCIENCES

THESIS
April 2013
Degree Programme in
Information Technology

Karjalankatu 3
FI 80200 JOENSUU
FINLAND
Tel. 358-13-260 6800

Author
Olli Timonen

Title
Windows Phone and HTML5 Application Development

Commissioned by
Ampparit Inc.

The purpose of this thesis was to develop Ampparit.com mobile application for Windows Phone platform. The application was supposed to display HTML5 version of Ampparit.com and to make it work well with the platform. Ampparit Inc. had already made an application for Android, so now it was Windows Phone's turn.

The Applications were developed by using a simple development method, which had four stages: design, execution, testing and release. The development for Windows Phone was made with Visual Studio by using XAML and C# programming languages. The Applications were aimed to be simple and memory friendly by using straightforward methods and handling well background work of processes and components.

The work was completed well in advance and therefore Ampparit.com applications are available at Windows Phone Store. Separate versions of applications for Windows Phone 7 and Windows Phone 8 had to be made, because of compatibility issues triggered by lack of HTML5 support of Windows Phone 7. While working on the applications, simple instructions were made for how to implement HTML5 site to Windows Phone application.

Language
Finnish

Pages 31

Keywords
mobile services, softwares, WWW pages, HTML

Sisältö

Termit ja käsitteet

Android	Linux-pohjainen käyttöjärjestelmä, joka on suunnattu kosketusnäyttölaitteille.
Black Box	Testausmenetelmä, joka keskittyy toimintojen testaamiseen.
Breakpoint	Ohjelmoinnissa käytetty kohta, johon ohjelma halutaan testauksessa pysäyttää.
C#	Olio-ohjelmointikieli, jota käytetään erityisesti Windows-alustoilla.
CSS	Cascading Style Sheets -tyylikieli, jolla määritellään web-sivun ulkoasu.
Debug	Ohjelmoinnissa käytetty termi virheiden etsimiseen ja vähentämiseen.
Emulaattori	Ohjelma, joka jäljittelee toista käyttöjärjestelmää.
HTML5	Hypertext Markup Language -merkintäkielen uusin versio, jota käytetään web-sivujen jäsennyksessä.
iOS	Applen kehittämä käyttöjärjestelmä mobiililaitteille.
JavaScript	Ohjelmointikieli, jota käytetään web-sivujen käyttäjäpuolen operaatioissa.
Natiivisovellus	Sovellus, joka on tarkoitettu käytettäväksi tietyllä alustalla tai laitteella.
URI	Uniform Resource Identifier sisältää tietoa, jossa on yleensä verkko-osoite ja lisäparametrejä sen määrittelyyn.
URL	Uniform Resource Locator on osoite Internetistä löytyvälle WWW-sivulle.
Viewport	Web-sivujen kokoon ja skaalaukseen käytetty tunniste.
White Box	Testausmenetelmä, joka keskittyy koodin testaamiseen.
Wi-Fi	Tekniikka tiedon siirtämiseen langattomasti laitteiden välillä.
XAML	Extensible Application Markup Language on XML-merkintäkieli, jolla tehdään visuaalinen määrittely sovelluksesta.

1 Johdanto

Tässä opinnäytetyössä esitellään Windows Phonen ja HTML5:n yhteensopivuuteen tarvittavia ratkaisuja sekä toimeksiannon kehitysprosessi. Siinä luodaan yritykselle Windows Phone -sovellus, joka käyttää jo olemassa olevaa HTML5-sivustoa.

Windows Phonen ja HTML5:n yhteensopivuusratkaisuja esitellään esimerkkiprojekteilla. Projekteja on kaksi, koska Windows Phone 7 ja Windows Phone 8 vaativat kumpikin hieman erilaiset ratkaisut. Esimerkeissä projektit luodaan Visual Studiolla ja sovellukseen tehdään tärkeimmät lisäykset ja muutokset HTML5-sovellusta varten. Lisäksi esimerkkisovelluksia varten luodaan yksinkertaiset HTML5-sivustot, joissa mallinnetaan Windows Phonen normaalia sovellusta web-muodossa. Tarkoituksena tällä on havainnollistaa, kuinka lähelle natiivisovellusta HTML5-sovelluksella voi päästä.

Toimeksiannossa luodaan Windows Phone -sovellus Ampparit Oy:lle. Ampparit Oy on tullut tunnetuksi Ampparit.com-uutisportaalista, jonka HTML5-versiota varten Windows Phone -sovelluskehys tehdään. Sovellus ottaa paljon mallia jo olemassa olevasta Ampparit.com Android-sovelluksesta, jotta käyttäjäkokemus saataisiin yhteneväiseksi sekä että jatkokehitys olisi helpompaa, kun sovellusrakenne on samanlainen eri alustoilla.

2 Ampparit Oy ja mobiilisovelluskehitystyö

2.1 Ampparit Oy

Ampparit Oy on Joensuussa toimiva IT-alan yritys, joka on osa Otava-konsernia. Ampparit perustaa toimintansa tiedonlouhintajärjestelmiin. Tällä hetkellä Amppareilla on kaksi tuotetta, Witpik-mediaseurantatyökalu ja Ampparit.com-uutisportaali. Witpik on ammattikäyttöön tarkoitettu työkalu, joka seuraa asiakasta kiinnostavia lähteitä ja kerää oleellisimman tiedon. Ampparit.com-uutisportaali on yksi Suomen suosituimmista verkkosivustoista, joka tarjoaa uutisvirran kotimaisista medioista sekä muita yleishyödyllisiä palveluja, kuten säätiedot ja tv-opas [1].

2.2 Mobiilisovelluskehityksen tarve

Ampparit Oy on kehittänyt Ampparit.com-uutisportaalista mini- ja lite-versiot. Mini on todella yksinkertainen toteutus, jossa ei ole käytetty ollenkaan tyylejä, mutta se sisältää silti tärkeimmät toiminnot ja on tarkoitettu käytettäväksi yksinkertaisissa laitteissa, kuten pieninäyttöisissä puhelimissa tai tekstipohjaisissa selaimissa [2]. Lite on nimensä mukaisesti kevyempi versio, joka on tarkoitettu mobiililaitteille [3]. Kummankin kevytversion uutiset ovat vain mobiiliuutisia. Kehittyvässä mobiilimaailmassa kuitenkin tarvitaan paremman käyttäjäkokemuksen tuovia ratkaisuja ja pienen yrityksen ei kannata ryhtyä luomaan jokaiselle eri mobiilialustalle omaa natiivisovellusta, koska niiden hallinta vaatii enemmän resursseja. Tämän takia Ampparit Oy päätti kehittää mobiililaitteille HTML5-sovelluksen, joka toimisi suosituimmilla alustoilla ja tarjoaisi yhteneväisen käyttäjäkokemuksen kaikille. Viime vuonna julkaistu Ampparit.com Android-sovellus on toteutettu näyttämällä Android-sovelluksen selaimessa kyseinen HTML5-sivusto. Alustan oma sovellus piti toteuttaa, jotta sovellukseen saataisiin oikeanlainen rajapinta HTML5-sisällön hyvän käyttäjäkokemuksen luomiseksi. Ampparit Oy:llä oli näin ollen tarve toteuttaa sovelluskehysratkaisut muillekin suosituille alustoille.

2.3 Toimeksiannon tavoite

Tavoitteena oli kehittää Ampparit.comista Windows Phone -sovellus hyödyntäen jo olemassa olevaa Ampparit.comin Android-sovellusta. Sovelluksen tuli olla yksinkertainen ja helppo jatkokehitystä ajatellen. Toimeksiannossa tuli myös kartoittaa mahdolliset jatkokehitysmahdollisuudet alustaa hyväksikäyttäen.

3 Tekniikat

Toimeksiannossa tarvitaan erinäisiä tekniikoita sovelluskehityksen toteutuksessa. Windows Phone ja HTML5 -sovellusta tehtäessä tarvitaan useita eri ohjelmointikieliä, jotka on myös hyvä tuntea.

3.1 Windows Phone

Windows Phone on uusin laajalevikkisistä mobiilikäyttöjärjestelmistä Googlen Androidin ja Applen iOS:n rinnalla. Se on Microsoftin kehittämä ja julkaistiin vuonna 2010 [4]. Muista käyttöjärjestelmistä se erottuu yksinkertaisuudellaan ja yhteneväisellä teemalla. Käyttäjäkokemus on pyritty mukaistamaan Windows-tietokoneella oleva kokemus eli käyttäjällä on työpöytä ja sovellusvalikko. Aloitusnäytössä sovelluksien ns. live-tiilet ilmoittavat käyttäjälle reaaliaikaisesti tilapäivityksistä. Windows Phone keskittyy sovelluksissaan pitkälti omaan teemaansa, jonka noudattamista odotetaan myös kolmannen osapuolen kehittäjiltä. [5.]

3.2 HTML5

HTML5 ei ole yksi asia, vaan se on kokoelma teknologioita, jotka yhdessä luovat modernin web-kokemuksen käyttäjälle. Sellaista ei voinut ennen luoda kuin natiivisovelluksissa. HTML5 koostuu viidennen version HTML-merkintäkielestä, CSS3-tyylikielestä ja JavaScript-ohjelmointikielestä. HTML5 ei kuulu millekään tietylle yritykselle tai rajoitu tiettyyn käyttöjärjestelmään, vaan sitä kehittävät yhdessä lukuisat verkkoyhteisöt ja suurimmat teknologiayritykset. Tällainen liittouma on luotu, jotta saataisiin aikaan yhteinen ja kehittynyt standardi, joka auttaa web-teknologioiden evoluutiossa. HTML5 ei siis ole vielä valmis, vaan se kehittyy koko ajan. [6.]

3.3 Ohjelmointikielet

3.3.1 C#

C# (lausutaan ja tunnetaan myös nimellä CSharp) on olio-ohjelmointiin perustuva ohjelmointikieli, jonka Microsoft kehitti vuonna 2000. C# on kehittynyt C- ja C++ -ohjelmointikielistä ja se muistuttaa syntaksiltaan paljon Javaa. C# on yleistynyt paljon ja on nykyään yksi eniten käytetyistä ohjelmointikielistä. Microsoftin Windows Phone -käyttöjärjestelmän ohjelmointikielenä käytetään C#:ia. [7.]

3.3.2 XAML

XAML on XML-pohjainen merkintäkieli, jonka on kehittänyt Microsoft. XAML-kielellä tehdään visuaalinen määrittely applikaatiosta samaan tapaan kuin HTML:llä määritellään web-sivun näkymä. Näkymä voidaan tehdä koodaamalla tai graafisella editorilla. XAML-tiedostoon voi tehdä koodilinkityksen sitä luotaessa, jolloin määritellään samanniminen tiedosto joko C#- tai Visual Basic -kielillä. Windows Phone kehityksessä luodaan automaattisesti XAML- ja C# -tiedostojen linkitys. [8.]

3.3.3 CSS

CSS on tyylikieli, jolla määritellään HTML-sivun ulkoasu. Tyylejä käytetään määrittelemällä tietyn HTML-elementin tai elementtiryhmän asetukset, esim. fontti tai taustaväri. CSS kehitettiin, koska sitä ennen piti jokaisen HTML-elementin tyyli määritellä erikseen tageilla suoraan HTML-tiedostoon. Tämä oli kehittäjien näkökulmasta todella hidasta ja sivun pituus venyi paljon. [9.]

3.3.4 JavaScript

JavaScript on ohjelmointikieli, jota käytetään web-sivujen käyttäjäpuolen operaatioissa. Se kehitettiin vuonna 1995 ja sitä käytettiin ensimmäisenä Netscape-selaimessa [10]. Se on nykyään sisällytetty valmiiksi yleisimpiin web-selaimiin ja on yksi käytetyimpiä ohjelmointikieliä. JavaScript muistuttaa Javaa vain nimensä perusteella, muuten syntaksi ja rakenne eroavat täysin. Windows Phonen WebBrowser-luokka sisältää yksinkertaiset tavat lähettää ja vastaanottaa JavaScript-komentoja web-sivun kanssa. [11.]

3.4 Työkalut

Microsoftin Visual Studio on ohjelmankehitysympäristö, jolla voi luoda sovelluksia lukuisille eri alustoille. Visual Studion ensimmäinen versio julkaistiin Windows 95:n yhteydessä vuonna 1995 ja on sen jälkeen saanut uuden version muutaman vuoden välein. Uusin versio tällä hetkellä on Visual Studio 2012, joka toimii vain Windows 8 -koneilla. Edeltävä versio Visual Studio 2010 on vielä eniten käytössä. Windows Phone kehityksessä Visual Studio 2010:llä voi luoda sovelluksia vain Windows Phone 7 -versioille. Visual Studio 2012 -versiolla voi luoda mobiilisovelluksia sekä Windows Phone 7:lle ja Windows Phone 8:lle. [12.]

4 Windows Phone ja HTML5

Windows Phonen ja HTML5:n yhteensopivuus on keskeistä toimeksiannon takia. Parhaan tuloksen saavuttamiseksi on ensin kartoitettava, mikä on mahdollista ja miten sekä mikä ei ole mahdollista ja miten siihen voidaan kehittää kompromissiratkaisu. Eri Windows Phone -versiot vaikuttavat myös paljon HTML5-sovelluksen kehitykseen.

4.1 WebBrowser ja keskeisimmät toiminnot

Windows Phonen WebBrowser-ohjain perustuu Internet Exploreriin. Windows Phone 7 -versiossa se perustuu Internet Explorer 9:ään ja Windows Phone 8:ssa se perustuu Internet Explorer 10 -versioon. WebBrowserin toiminnot ovat rajalliset verrattuna työpöytäselaimiin, mutta se sisältää kuitenkin perusselaamiseen tarvittavat toiminnot sekä muutaman apufunktion HTML5-sovellusta varten. [13.]

4.1.1 Navigate

Navigate on funktio, joka ottaa parametrikseen URI:n ja antaa selaimelle käskyn siirtyä URI:n sisältämään URL-osoitteeseen [14]. URI on mahdollista määritellä sisältämään joko absoluuttisen tai relatiivisen URL:n. Navigatella on myös toinen määritys, joka ottaa parametrikseen URI:n lisäksi postDatan ja additionalHeadersin [15]. PostData on tietoa, joka halutaan viestittää serverille HTTP

POST -siirron välityksellä. AdditionalHeaders viestittää sivulle HTTP-tunnisteita. Niiden avulla voidaan esim. esittäytyä työpöytäselaimena, jos halutaan avata työpöydille tarkoitettu nettisivu mobiiliversion sijaan.

4.1.2 InvokeScript

InvokeScriptin avulla voi lähettää Windows Phone -sovelluksen koodista web-sivulle JavaScript-komentoja ja -funktioita. InvokeScript ottaa parametrikseen yhden merkkijonon tai merkkijonon ja merkkijonotaulukon [16]. Merkkijonoon tarvitsee vain syöttää haluttu JavaScript-funktio ja tarpeen vaatiessa taulukkoon pystyy antamaan parametrit. Parametrejä voi käyttää monella eri tapaa ja niistä löytyy hyvin esimerkkejä [17].

4.1.3 ScriptNotify

ScriptNotify on tapahtumakuuntelija, joka vastaanottaa web-sivun lähettämiä viestejä Windows Phonelle [18]. Viestit lähetetään web-sivulta JavaScriptin avulla kuvan 1 tapaan.

```
window.external.notify("test");
```

Kuva 1. Esimerkki viestin lähetyksestä web-sivulta puhelimeen.

Paluuarvojen perusteella sovelluksessa voidaan tehdä tarvittavat viestit tai muutokset käyttäjälle. Esimerkiksi lomaketta lähetettäessä web-sivulta voidaan lähettää viesti käyttäjän selaimeen ja näyttää Windows Phonen oma MessageBox-viesti onnistuneesta lomakkeen lähetyksestä.

4.2 Rajoitteet ja ongelmat

Mobiililaitteille tarkoitetuissa HTML5-sovelluksissa tärkeintä on saada web-sivu tuntumaan ns. aidolta sovellukselta ja ensimmäiseksi on kannattavaa saada se skaalautumaan oikein. Skaalaaminen onnistuu käyttämällä viewport-metatagia, jolle annetaan halutut parametrit [19]. Käyttämällä device-width-parametriä pitäisi sivun skaalautua laitteen leveyden mukaan. Microsoft on päättänyt, että device-width palauttaa aina samat arvot. Kun laite on pystyasennossa, niin palautuu arvo 320 ja vaaka-asennossa palautuu arvo 480. Näillä parametreillä

sivu näyttää aina todella lähelle zoomatulta. Microsoft päätti tehdä tällaisen ratkaisun, koska he havaitsivat, että useimmat mobiilisivut on suunniteltu tälle resoluutiolle ja näin ollen he tukisivat tätä näillä oletusarvoilla. Tämä kuitenkin aiheuttaa ongelmia paremmalla resoluutiolla olevilla puhelimilla, koska tällöin sivu ns. kasvaa liian suureksi. Muilla alustoilla device-width palauttaa laitteen resoluution leveyden, mutta tämä ei riitä oikean skaalauksen tuomiseksi, vaan eri resoluutioiden ja näyttöjen tuumakoon yhteensopivuudeksi lisäparametri targetdensityDpi auttaa asiaan. Tätä lisäparametriä ei ole Windows Phonen selaimessa saatavilla. Microsoft on tuomassa lähitulevaisuudessa Full HD -resoluutiotuen Windows Phone -alustalle, jolloin heidän pitäisi samalla reagoida tähän ongelmaan [20].

Windows Phone 7:n selaimesta puuttuu kokonaan CSS:n position: fixed -tuki. Elementit, jotka määritellään kiinteään sijaintiin, vierivät normaalisti muiden elementtien kanssa. Näin ollen kehittäjä ei voi luoda, esim. aina sivun ylälaidassa pysyvää toimintopalkkia. Selaimesta ei voi myöskään poistaa värikorostuksia, jotka esiintyvät harmaina laatikkoina elementtien päällä, kun niitä klikataan. Jos tavoitteena on saada natiivisovellusvaikutelma, niin linkkejä tai painikkeita klikatessa harmaat suorakulmiot pilaavat tämän tavoitteen. Jos Windows Phone 7:lle haluaa tehdä lokaalisen HTML5-sovelluksen, niin kehittäjän kannattaa turvautua kolmannen osapuolen kirjastoihin, koska Windows Phone 7 ei tue lokaalisten tiedostojen käyttöä WebBrowser-luokassa suoraan. Yhden lokaalisen HTML-tiedoston näyttämisessä pitää ensin lukea tiedosto muistiin ja käyttää WebBrowser-luokan NavigateToString-funktiota kuvan 2 tapaan. Tämän ratkaisun kanssa ilmenee ongelmia, jos HTML-tiedosto sisältää tyyli- tai JavaScript-tiedostoviitteitä sillä niiden paikallistaminen ei onnistu näin.

```
var file = Application.GetResourceStream(new Uri("Html/index.html", UriKind.Relative));
string html = new StreamReader(file.Stream).ReadToEnd();
Browser.NavigateToString(html);
```

Kuva 2. Yhden lokaalisen HTML-tiedoston näyttäminen selaimessa.

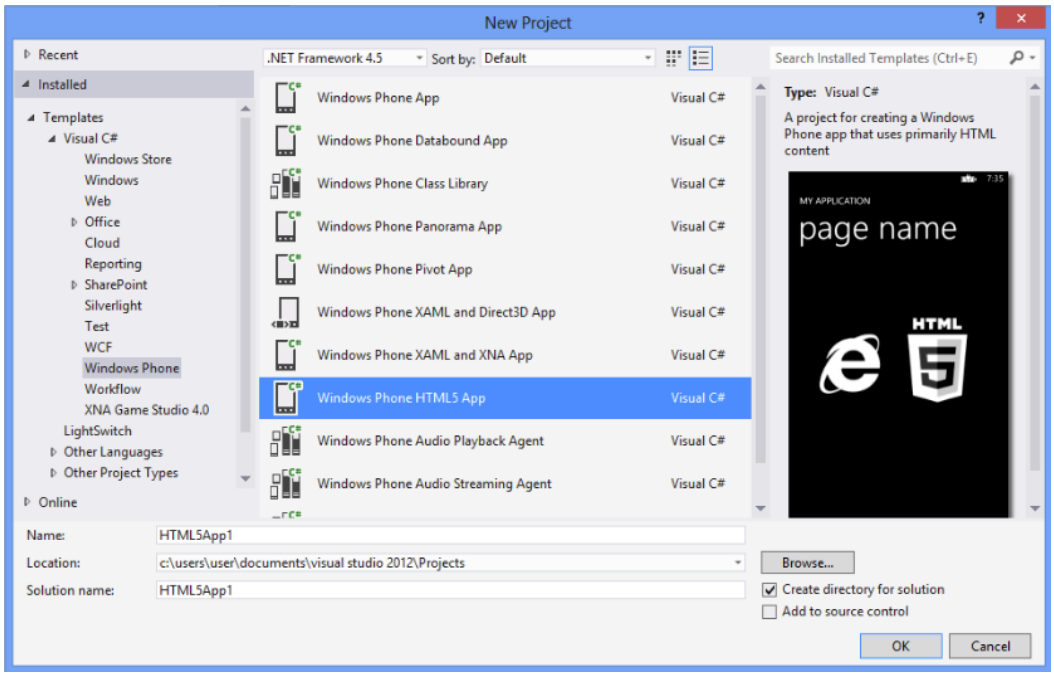
4.3 Esimerkkiratkaisut

Ratkaisuissa esitellään, miten luodaan yksinkertainen Windows Phone -sovelluspohja HTML5-sivustolle. Windows Phone 7- ja Windows Phone 8

-versioille on luotu omat ratkaisut. HTML5-sivustoesimerkit koostuvat vain HTML:stä ja CSS:stä, jotka tuovat Windows Phone -tyylin web-muotoon.

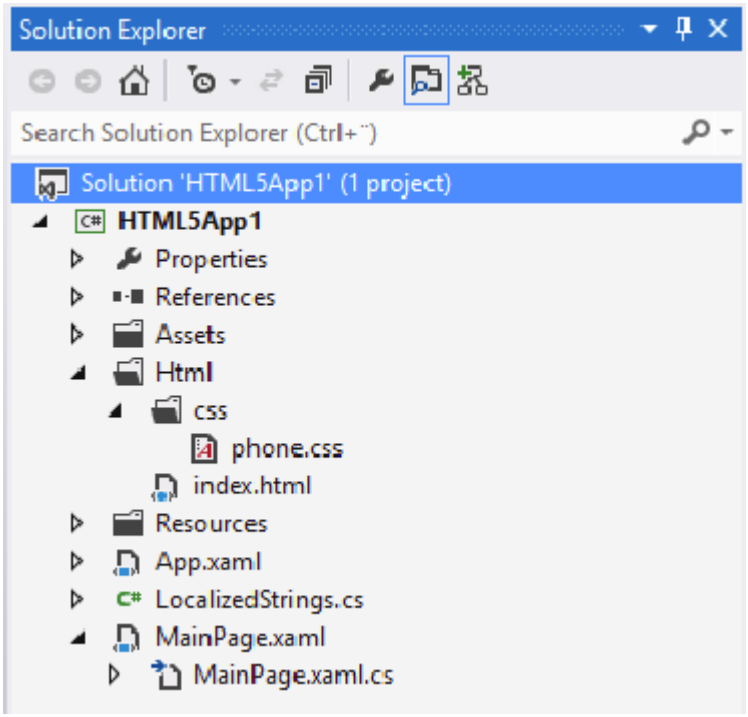
4.3.1 Windows Phone 8

Microsoft Visual Studio 2012 -versiossa on Windows Phone 8:lle oma HTML5-sovellusmalli, joka on valittuna kuvassa 3. Sovelluspohja on tarkoitettu käytettäväksi lokaalisella HTML-sisällöllä, mutta sitä voi hieman muokkaamalla käyttää hyvänä kehyspohjana jo olemassa olevallekin HTML5-sivustolle.



Kuva 3. HTML5-projektin luonti Visual Studio 2012 avulla.

Velho luo automaattisesti WebBrowser-näkymän ja siihen liittyvät toiminnot sekä yksinkertaisen HTML5-sivuston. Sivustoon kuuluu index.html ja tyylitiedosto phone.css. Rakenne on muilla tavoin samanlainen normaalin Windows Phone App -projektin kanssa, niin kuin kuvasta 4 selviää.



Kuva 4. HTML5 -projektin rakenne

Sovelluksen pääsivulle generoitui automaattisesti yksinkertainen näkymäratkaisu, johon ei tarvinnut tehdä isompia muutoksia. Oleelliset määrittelyt selviävät kuvasta 5. Ainut lisäys oli Back-näppäimen valvontamäärittely. Lisäksi alapalkki oli turha ja sen pystyi poistamaan kokonaan.

```
shell:SystemTray.IsVisible="True" BackKeyPress="PhoneApplicationPage_BackKeyPress_1">

<!--LayoutRoot is the root grid where all page content is placed-->
<Grid x:Name="LayoutRoot" Background="Transparent">
    <phone:WebBrowser x:Name="Browser"
                      HorizontalAlignment="Stretch"
                      VerticalAlignment="Stretch"
                      Loaded="Browser_Loaded" />
</Grid>
```

Kuva 5. Näkymän määrittely XAML-tiedostossa.

Pääsivun koodipuolella ei tarvinnut myöskään tehdä paljoa muutoksia, niin kuin kuvassa 6 on esitetty. Turhat funktiot poistettiin ja lisättiin Back-näppäimen määrittely. Back-näppäintä painamalla sivuhistoriassa siirrytään taaksepäin, jos se on mahdollista ja muussa tapauksessa sovellus suljetaan.

```
// Url of Home page
private string MainUri = "/Html/index.html";

// Constructor
public MainPage()
{
    InitializeComponent();
}

private void Browser_Loaded(object sender, RoutedEventArgs e)
{
    // Add your URL here
    Browser.Navigate(new Uri(MainUri, UriKind.Relative));
    Browser.IsScriptEnabled = true;
}

private void PhoneApplicationPage_BackKeyPress_1(object sender, CancelEventArgs e)
{
    if (Browser.CanGoBack)
    {
        e.Cancel = true;
        Browser.GoBack();
    }
}
```

Kuva 6. Sovelluksen pääsivun koodimääritelmät.

HTML-tiedostoon lisättiin metatagi, joka poistaa klikkauskorostukset. Normaalisti linkin päälle tulee harmaa korostuslaatikko, kun sitä klikkaa. Valmiiksi generoitujen otsikoiden lisäksi koodissa on myös sisältö-elementti, johon voi laittaa haluamansa datan. Tässä esimerkissä se täytetään vain tekstiriveillä, jotta testatessa voi käyttää vertikaalista vieritystä. Nyt myös jokainen div-elementti on merkitty id:llä. Koko HTML-tiedoston sisältö löytyy kuvasta 7.

```
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
        <meta name="msapplication-tap-highlight" content="no"/>
        <link rel="stylesheet" type="text/css" href="/html/css/phone.css" />
        <title>Windows Phone</title>
    </head>
    <body>
        <div id="title-panel">
            <p>MY APPLICATION</p>
            <div id="page-title">
                <p>page title</p>
            </div>
        </div>
        <div id="content">
            <p>Row1</p><p>Row2</p><p>Row3</p><p>Row4</p><p>Row5</p>
            <p>Row6</p><p>Row7</p><p>Row8</p><p>Row9</p><p>Row10</p>
            <p>Row11</p><p>Row12</p><p>Row13</p><p>Row14</p><p>Row15</p>
        </div>
    </body>
</html>
```

Kuva 7. Windows Phone 8:n HTML5-sovelluksen muokattu HTML-tiedosto.

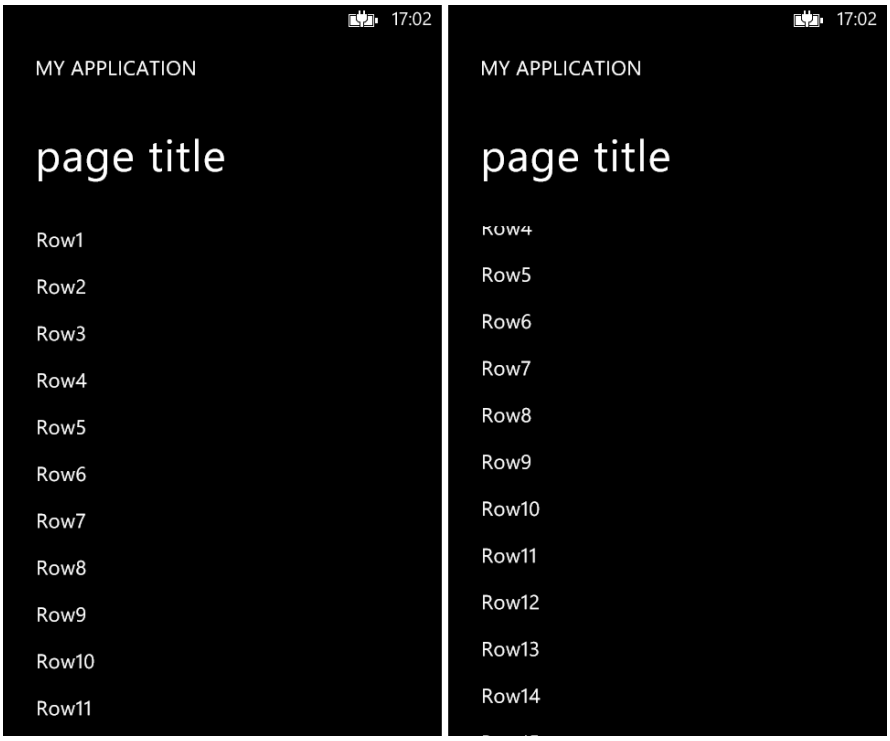
CSS-tyylitiedostossa body-elementti sisältää nyt -ms-user-select: none; -komennon, joka estää tekstin valitsemisen, jotta sovelluksessa näkyvää tekstiä ei voi kopioida leikepöydälle, kuten normaalissakin Windows Phone -sovelluksessa. Body-elementissä esiintyi outo ongelma, jos se sisälsi margin-left määritelmän. Margin-left siirtää elementin vasenta laitaa tietyn määrän oikealle. Tässä tapauksessa se aiheutti sivun vieritysongelman, jolloin sivua pystyi liikuttelemaan sivulle ja yläviistoon. Ratkaisuna tähän ongelmaan oli lisätä margin-left vain tekstiä sisältäviin elementteihin, jolloin sivua pystyi vain vierittämään ylös ja alas. Uutena tyylimääritelmänä oli myös title-panel, joka määritteli sivun otsikkopalkin sijainnin olemaan aina ylimpänä. Vastaisuudessa sisältömääritelmä eli content alkaa tietyn verran alempaa, jotta vieritettäessä kaikki elementit sijoittuvat oikein. CSS-tiedosto tiivistetty kuvaan 8.

```
@media screen and (orientation:portrait) {
    @-ms-viewport {
        width:320px;
        user-zoom: fixed;
        max-zoom: 1;
        min-zoom: 1;
    }
}

@media screen and (orientation:landscape) {
    @-ms-viewport {
        width:480px;
        user-zoom: fixed;
        max-zoom: 1;
        min-zoom: 1;
    }
}

html, body {
    height: 100%;
    width: 100%;
    margin: 0px;
    padding: 0px;
}
```

```
body {
    -ms-user-select: none;
    font-size: 11pt;
    font-family: "Segoe WP";
    letter-spacing: 0.02em;
    background-color: #000000;
    color: #ffffff;
}

#title-panel {
    background-color: inherit;
    position: fixed;
    top: 0;
    margin-left: 24px;
}

#page-title {
    font-size: 24pt;
}

#content {
    margin-left: 24px;
    margin-top: 140px;
}
```

Kuva 8. CSS-tiedoston määritelmät.

Kun ohjelman ajoi puhelimessa, niin tulokseksi saatiin sivu, jossa yläpalkki pysyi paikallaan ja sisällössä olevat tekstirivit menevät vieritettäessä yläpalkin alle. Kuvassa 9 on kaksi kuvankaappausta sovelluksesta, jossa vasemmanpuoleisessa sivu on aloituskohdassa ja oikean puoleisessa sivua on vieritetty hieman.



Kuva 9. Kuvankaappaukset esimerkkisovelluksesta.

4.3.2 Windows Phone 7

Tässä esimerkkiratkaisussa käytettiin palvelinta web-sivuston näyttämiseen. Lokaaliselle palvelimelle on laitettu Windows Phone 8 -esimerkissä käytetty sivustopohja, joka sisältää vain HTML- ja CSS-tiedostot, joita on muokattu hieman Windows Phone 7:lle sopivammiksi. HTML-tiedostoon on lisätty metatagi viewportille, joka skaalaa web-sisällön koon. Pystyssä ruudun leveys siis 320px ja vaaka asennossa 480px. HTML-tiedosto kokonaisuudessaan kuvassa 10.

```
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
        <meta name="viewport" content="width=device-width, user-scalable=no">
        <link rel="stylesheet" type="text/css" href="css/phone.css" />
        <title>Windows Phone</title>
    </head>
    <body>
        <div>
            <p>MY APPLICATION</p>
        </div>
        <div id="page-title" >
            <p>page title</p>
        </div>
    </body>
</html>
```

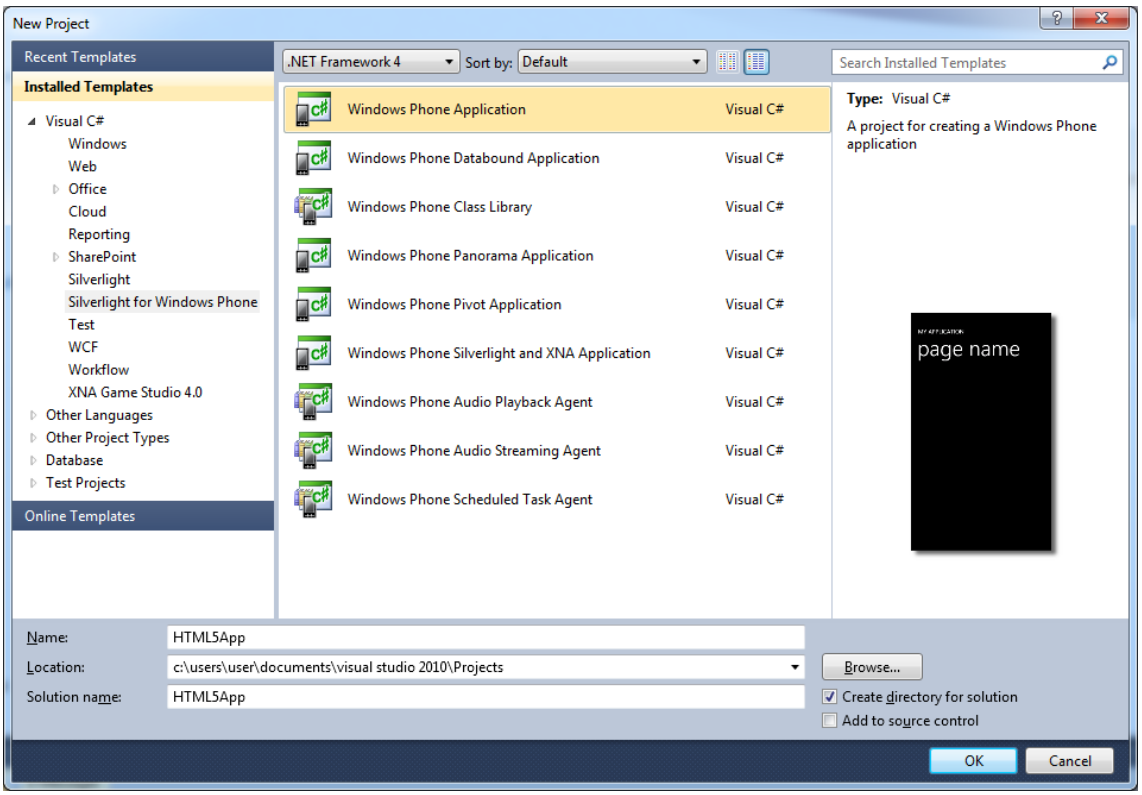
Kuva 10. Windows Phone 7 -sovelluksessa käytetyn HTML-tiedoston sisältö.

CSS-tiedostosta on poistettu media-tyypit ja sivun kokoon vaikuttavat leveys ja korkeus komennot, joten tiedostoon jäi jäljelle vain fonttien ja sivun ulkonäköön vaikuttavat määritteet. CSS-määritelmät kuvassa 11.

```
body {
    font-size: 11pt;
    font-family: "Segoe WP";
    letter-spacing: 0.02em;
    background-color: #000000;
    color: #ffffff;
    margin-left: 24px;
}

#page-title {
    font-size: 24pt;
}
```

Kuva 11. Windows Phone 7 -sovelluksessa käytetyn CSS-tiedoston sisältö.

Vanhemmalle Windows Phone -alustalle sovelluksen luontiin voidaan käyttää joko Visual Studio 2010- tai uudempaa versiota. Projekti luodaan käyttämällä normaalia Windows Phone Application -sovelluspohjaa, joka on valittuna kuvassa 12.



Kuva 12. Windows Phone 7 -projektin luonti.

Sovelluspohjassa automaattisesti luodaan vain pääsivu, jonka näkymä sisältää vain pari tekstiriviä, jotka voi heti poistaa. Ratkaisussa käytetään Windows Phonen komponenteista vain yhtä web-selainta, joten näkymä voidaan määritellä hyvin yksinkertaisesti. Selain on määritelty täyttämään koko ruutu ja seuraamaan Loaded-, Navigating- ja Navigated -tapahtumia, jotka näkyvät kuvassa 13.

```
<Grid x:Name="LayoutRoot" Background="Transparent">
    <phone:WebBrowser x:Name="Browser"
                      HorizontalAlignment="Stretch"
                      VerticalAlignment="Stretch"
                      Loaded="Browser_Loaded"
                      Navigating="Browser_Navigating"
                      Navigated="Browser_Navigated" />
</Grid>
```

Kuva 13. Sovelluksen näkymä määriteltynä XAML-tiedostoon.

Koodin puolella on tehty manuaalisesti historian selaus Back-näppäimellä ja poistettu web-sivun zoomaus. Jokaisella web-sivu navigoinnin onnistumisella tallennetaan sen URI pinoon. Jos historia-pino sisältää useamman kuin yhden web-sivun ja käyttäjä painaa Back-näppäintä niin pinon ylin sivu poistetaan ja toinen käytetään uuteen navigointiin ja poistetaan pinosta samalla. Kun navigointi taas onnistuu, niin sivu laitetaan uudestaan pinoon, kuten kuvasta 14 selviää.

```
private void Browser_Navigated(object sender, NavigationEventArgs e)
{
    history.Push(e.Uri);
}

private void PhoneApplicationPage_BackKeyPress(object sender, CancelEventArgs e)
{
    if (history.Count > 1)
    {
        e.Cancel = true;
        history.Pop();
        Browser.Navigate(history.Pop());
    }
}
```

Kuva 14. Selaushistorian käsittely Windows Phone 7 -sovelluksessa.

Web-sivun zoomauksen poisto on tehty kolmannen osapuolen VisualTreeAdapter-luokalla, jonka avulla estetään selaimen skaalaus- ja zoomaus -arvojen muuttuminen sekä tuplanapautuksen käyttö selaimessa. Ratkaisu kokonaisuudessaan kuvassa 15. [21.]

```
private void Browser_Loaded(object sender, RoutedEventArgs e)
{
    var border = Browser.Descendants<Border>().Last() as Border;

    border.ManipulationDelta += Border_ManipulationDelta;
    border.ManipulationCompleted += Border_ManipulationCompleted;
    border.DoubleTap += Border_DoubleTap;

    Browser.Navigate(new Uri("http://localhost:8080/index.html", UriKind.Absolute));
    Browser.IsScriptEnabled = true;
}

private void Border_DoubleTap(object sender, GestureEventArgs e)
{
    e.Handled = true;
}

private void Border_ManipulationCompleted(object sender, ManipulationCompletedEventArgs e)
{
    if (e.FinalVelocities.ExpansionVelocity.X != 0.0 ||
        e.FinalVelocities.ExpansionVelocity.Y != 0.0)
        e.Handled = true;
}

private void Border_ManipulationDelta(object sender, ManipulationDeltaEventArgs e)
{
    if (e.DeltaManipulation.Scale.X != 0.0 ||
        e.DeltaManipulation.Scale.Y != 0.0)
    {
        e.Handled = true;
    }
}
```

Kuva 15. Selaimen käynnistys ja zoomauksen esto.

Tuloksena saadaan aikaan ongelmiin nähden hyvä Windows Phone 7 ja HTML5 yhteensopivuus sekä sovelluspohja, jota hieman muokkaamalla voi luoda yksinkertaisia web-sovelluksia. Pysyvän sijainnin tuen puute Windows Phone 7:n selaimesta huononsi esimerkkiratkaisun käytettävyyttä. Sovelluksen ulkonäkö ja käyttötuntuma saadaan kuitenkin toteutettua halutunlaiseksi yhteensopivuusongelmista huolimatta. Kuvassa 16 näytetään sovellus ilman mitään sisältöä ja sitä voisi hyvinkin luulla ns. natiivisovellukseksi.

Kuva 16. Windows Phone 7 -esimerkkisovellus emulaattorissa.

4.4 Valmiit kirjastot

Mobiililaitteille HTML5 kehitystä varten on saatavilla ilmaisia ohjelmistorunkoja. Näistä tunnetuimmat ovat PhoneGap ja Apache Cordova. Nämä voi usein sekoittaa toisiinsa, koska PhoneGap on johdannaisversio Cordovasta. Kummatkin ovat ilmaisia ja niillä voi luoda hybridi-sovelluksia suosituimmille alustoille. Jos yrityksellä tai organisaatiolla on tarve luoda tyhjästä mobiilisovellus, niin PhoneGap tai Apache Cordova ovat tutustumisen arvoisia. [22.]

5 Ampparit.com-sovelluksen kehitysprosessi

5.1 Sovelluksen suunnittelu

Suunnittelu alkoi Ampparit.com Android-sovellukseen tutustumisella, jonka jälkeen kartoitettiin Windows Phone -alustan kykenevyys sovelluksen vaatimuksiin nähden. Ampparit.com-sovelluksen suunnittelussa on pyritty noudattamaan Android-sovelluksen luokkarakennetta niin hyvin kuin mahdollista.

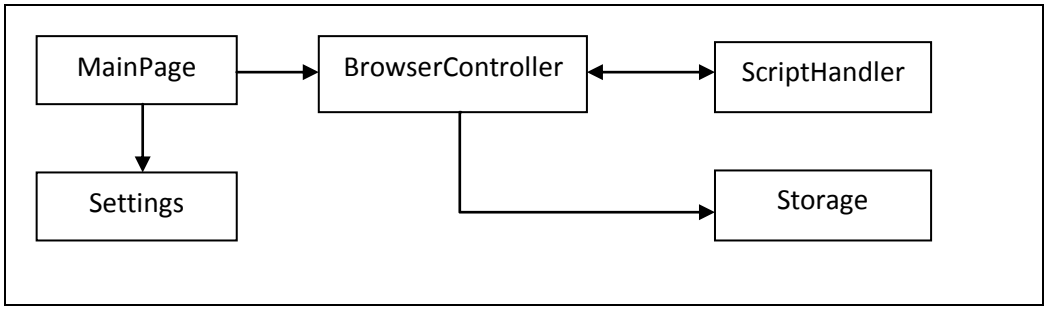
5.1.1 Lähtötilanne

Kehitysprosessi alkoi Ampparit.com Android-sovelluksen testikäytöllä, jolloin oli mahdollista saada hyvä kuva siitä mitä Windows Phone -sovellus tulisi tarvitsemaan. Millaisia käyttöjärjestelmän omia toimintoja sovellus tarvitsisi ja millaisia ratkaisuja tietyissä tilanteissa oli käytetty? Ampparit.com Android-sovellus on toteutettu näyttämällä käyttäjälle HTML5-sivusto, joka näyttää ja tuntuu natiivisovellukselta. Ampparit Oy tarjosi lähdekoodit Android-sovellukseen ja niistä kävi ilmi sovelluksen yksinkertaisuus. Sovellus käyttää kahta nettiselainta. Selain Ampparit.com HTML5-sivustoa varten ja toinen selain avattavia uutisia varten. Windows Phonelle oli mahdollista tehdä vastaavanlainen ratkaisu ja jopa lainata suoraan muutamia algoritmejä.

Android-sovelluksessa uutista luettaessa oli mahdollista siirtyä asetuksiin, jonka sivu oli toteutettu alustan omilla komponenteilla. Tällainen vastaava ratkaisu tulisi toteuttaa myös Windows Phone -sovelluksessa alustan omaa tyyliä noudattaen. Tämä olisi ainoa sivu, jossa Windows Phone:n oma tyyli tulisi näkyviin.

5.1.2 Määrittely

Alustava luokkarakenne luotiin paperille, joka noudatti pitkälti Android-sovelluksen linjaa. Luokkarakenne on kuvattu kuviossa 17.



Kuvio 17. Windows Phone -sovelluksen alustava luokkarakenne.

Sovelluksen pohjana toimii MainPage eli pääsivu, joka on oletussivu Windows Phone -sovellusta luotaessa. Se käsittäisi kaikki komponentit mitä näytölle laitetaan näkyviin eli mm. latauspalkki ja selaimet. Kaaviossa MainPagesta oikealle päin löytyvät pääsivun alaiset luokat. Ensimmäinen alaluokka on BrowserController, joka nimensä mukaisesti hallinnoi selaimia. Kaikki sivujen navigointiin liittyvät ja sivun hallintaa käsittelevät toiminnot ja algoritmit löytyvät siitä. ScriptHandler-luokka käsittelee kaikki JavaScriptiin liittyvät toiminnot eli kaikki

HTML5-sivulle lähetettävät ja vastaanotettavat komennot. Storage-luokka hoitaa tarvittavien tietojen tallentamisen puhelimen muistiin. Settings-sivulta voi muuttaa uutistenluku-selaimen asetuksia.

5.2 Toteutus

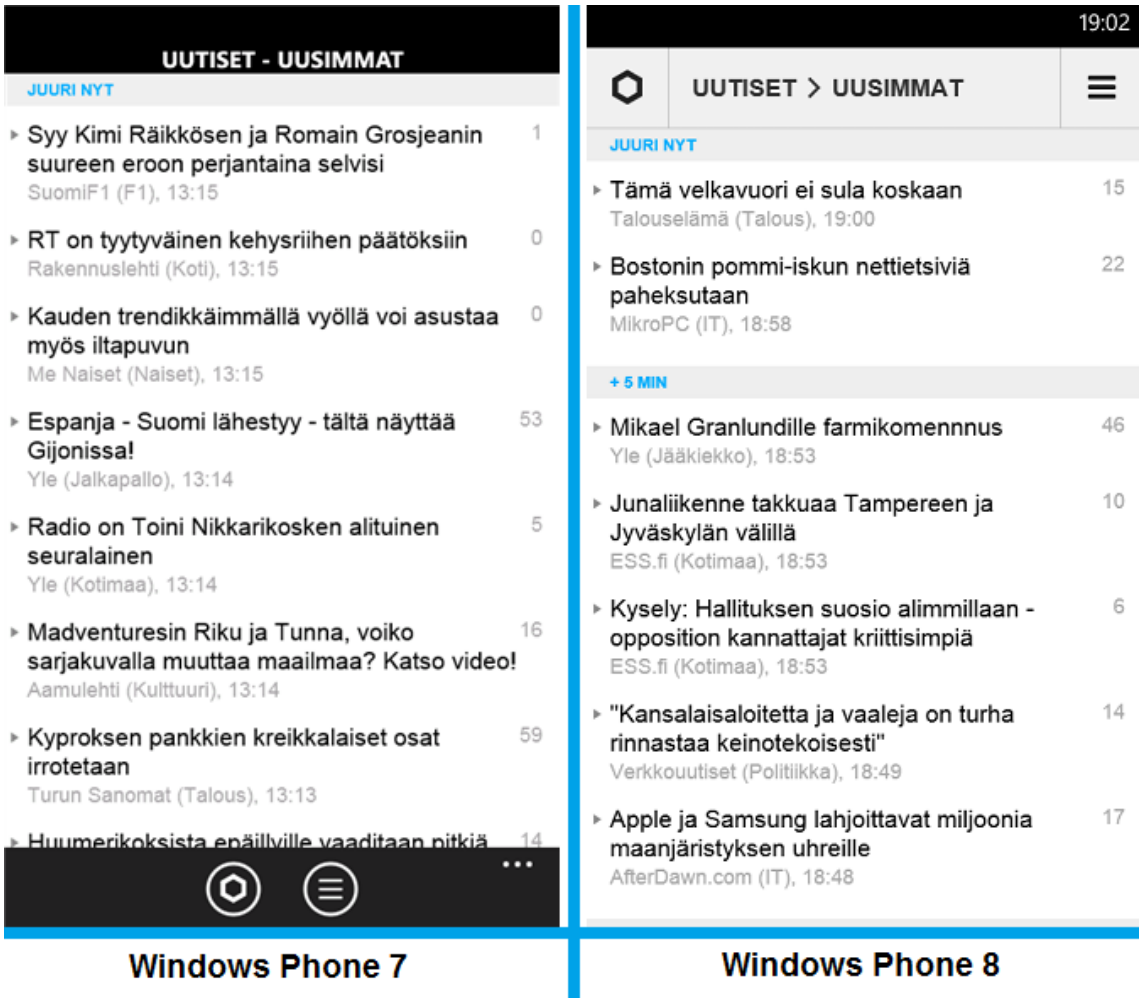
Ampparit.com-sovelluksen luominen alkoi kokeilemalla Windows Phone -alustan rajoja ja miten hyvin Android-alustan sovellusratkaisua voisi käyttää hyväksi. Windows Phone 7 -alustan rajoitteet HTML5-käytössä aiheuttivat kehityksen keskeytymisen ja Ampparit Oy:n toive oli keskittyä Windows Phone 8 -kehitykseen, jossa vastaavia ongelmia ei ilmennyt. Windows Phone 7 -kehitys nostettaisiin uudestaan esille Windows Phone 8 -sovelluksen valmistuttua.

Ampparit Oy käytti testilaitteena Lumia 820 -puhelinta, jossa on Windows Phone 8 -käyttöjärjestelmä. Ampparit.com HTML5-sivusto toimi muokkaamattomana Lumia 820:n web-selaimessa suhteellisen hyvin. Ongelmia esiintyi vain esteettisissä ominaisuuksissa, jotka olivat korjattavissa.

Windows Phone 8 -sovelluksen toteutus alkoi noudattamalla ennalta suunniteltua luokkakaaviota ja kääntämällä Android-sovelluksen lähdekoodeja C#-muotoon. Sovellus kääntyi nopeasti, mutta testiajossa ilmeni ongelma, joka esiintyi kahden web-selaimen ja toimintopalkin yhteensopivuudessa. Jos HTML5-sivustoa vieritti alaspäin ja avasi uutislinkin ja uutisen lukemisen jälkeen palasi takaisin sovellukseen, niin sivu ei ollut enää käyttäjän selaamassa kohdassa, vaan sivu kelautui automaattisesti ylös. Ongelman aiheuttaja oli toimintopalkin piilottaminen ja näyttäminen, jonka tapahtuessa taustalla oleva selain kelautui ylös. Käyttäjän näkökulmasta tällainen ongelma voi olla erittäin ärsyttävä ja ongelmaan tuli keksiä ratkaisu. Helppoa ratkaisua siihen ei löytynyt, vaan sovelluksen koko rakennetta oli pakko muuttaa. Rakennetta piti muuttaa niin, että se sisälsi kaksi sivua kullekin selaimelle. Näin ollen tiettyyn selaimeen liittyvät toiminnot eivät sotkeneet toisen toimintaa eli tässä tapauksessa toimintopalkki vain uutistenluku-selaimelle. Muutos oli iso, mutta tarpeellinen, koska sovellus toimisi tällä muutoksella jopa nopeammin ja olisi tietenkin käyttäjäystävällisempi. Sovelluksen toteutuksessa ei ilmennyt muita suurempia ongelmia ja näin ollen kehitysprosessi eteni seuraaviin vaiheisiin.

Windows Phone 8 -sovelluskehityksen valmistuttua Windows Phone 7 -sovellus nousi takaisin työlistalle. Windows Phone 7 -rajoiteongelmat ohitettiin luomalla hybridi-sovellus. Hybridi-sovelluksella tarkoitetaan sitä, että se sisältää kahta eri tekniikkaa toiminnallisuus toteutuksessa eli tässä tapauksessa natiivi ja HTML5. Ongelmakohta sovelluksessa oli yläpalkki, joka tuli korvata Windows Phone -alustan omalla toimintopalkilla, jonka painikkeet linkitettiin lähettämään sivustolle niitä vastaavat komennot. Sovelluksen ylälaitaan lisättiin otsikko-tekstipalkin, joka muuttuu aina kun vastaanotetaan web-sivulta uusi otsikkoviesti. Sovelluksessa oli näin ollen sovelluspalkki kokoajan näkyvissä, joten sen rakennetta ei tarvinnut muuttaa niin kuin Windows Phone 8 -sovelluksessa.

Windows Phone 7 -sovelluksesta piti karsia pois lukuisia ominaisuuksia yhteensopivuusongelmien vuoksi. Jotkin yhteensopivuusongelmat oli mahdollista korjata palvelinmuutoksilla. Radikaalein muutos oli jättää TV-opas kokonaan pois, koska sitä ei sellaisenaan voinut käyttää. TV-opas on kuitenkin mahdollista palauttaa tulevaisuudessa, kun sitä muokataan sopivammaksi. Windows Phone 7 -sovellus valmistui tarvittavilla muutoksilla ja oli näin ollen valmis testaukseen ja julkistamiseen. Kummankin sovelluksen etusivu löytyy kuvasta 18.



Kuva 18. Ampparit.comin Windows Phone 7- ja Windows Phone 8 -sovellukset.

5.3 Testaus

Sovellus oli yksinkertainen, joten testitkin olivat suhteellisen yksinkertaisia. Sovellusta testattiin Black box- ja White box -testausmenetelmillä. Black box -testaus on ns. sovelluksen kokeilua tietämättä yhtään mitä taustalla tapahtuu. White box -testaus on hallittua koodin testausta. [23.]

5.3.1 Black Box

Windows Phone 8 -kehitys ja testaus tehtiin kokonaisuudessaan Lumia 820:llä, joten laitetestausta tapahtui kokoajan. Sovelluksen yksinkertaisuus mahdollisti nopeat testaukset esim. siirtymiset, eleet yms. Mahdolliset nettiongelmatilanteet testattiin myös. Laitteen kanssa siirryttiin katvealueelle, jossa puhelimen kentät laskivat vähiin. Tällaisen tilanteen käsittely toteutettiin vain sovellusta käynnistettäessä, jossa sovellus ensin tarkistaa nettiyhteyden saatavuuden 15 sekun-

nin aikakatkaisulla. Jos aikakatkaisu menee umpeen niin käyttäjä saa virheilmoituksen. Vastaavanlainen skenaario oli mahdollista luoda yhdistämällä puhelin Wi-Fiin ja katkaisemalla sen nettiyhteys. Puhelin näyttää olevansa yhteydessä verkkoon, muttei tajua että se ei pääse internetiin.

5.3.2 White Box

Sovelluksen koodi oli yksinkertaista, koska suurin osa työstä tapahtui palvelimen puolella. Toteutuksen yhteydessä tehtiin yksinkertaisia debug-testejä. Breakpointteja asetettiin testattavan muuttujan muutoksiin ja kokeiltiin eri skenaarioita sovelluksessa. Monimutkaisemmat funktiot tarkastettiin vielä hieman tarkemmin ns. double-check-tyylillä.

5.4 Julkaisu

Microsoftin Windows Phone -sovelluskauppa on Windows Phone -sovellusten ainoa jakopiste, joten julkaisu tehtiin vain sinne. Kehittäjän osalta sovelluksen lisääminen kauppaan on varsin yksinkertaista. Sovelluksesta pitää antaa perustiedot, kuvankaappaukset yms. sekä tietenkin luodun sovelluksen XAP-tiedosto. Kun tarvittavat tiedot on annettu, niin sovelluksen hyväksymisprosessi voi alkaa. Hyväksymisprosessi voi kestää 5–7 päivää, jonka jälkeen kehittäjä saa sähköposti-ilmoituksen prosessin päättymisestä. [24.]

Ampparit.com -sovelluksia tehtiin kaksi kappaletta, sillä Windows Phone 7 vaati erilaisen ratkaisun. Windows Phone -kaupassa onnistuu usean sovelluksen liittäminen samaan tuotteeseen. Kun käyttäjä etsii sovelluskaupasta Ampparit.com-sovellusta Windows Phone 7 -puhelimella, niin käyttäjälle näytetään vain Ampparit.comin Windows Phone 7 -sovellus. Jos käyttäjä etsii Ampparit.com-sovellusta Windows Phone 8:n tai uudemman version puhelmella, niin käyttäjälle näytetään Windows Phone 8 -versio sovelluksesta. [25.]

Ampparit.com Windows Phone 8 -sovelluksen julkaisu sujui pääpiirteittäin ongelmitta, mutta nopeasti alkoi tulla palautetta, että fontti olisi liian pieni. Sovellus oli siis testattu vain Lumia 820 -puhelimella ja ongelmaksi paljastui viewportin skaalaus Lumia 920 -puhelimella. Ongelma korjattiin suhteellisen nopeasti palvelimelle ja sovellus näin ollen skaalautui oikein kaikille puhelimille. Windows

Phone 7:n julkaisussa vastaavaa ongelmaa ei ilmennyt, koska Windows Phone 7 tukee vain yhtä resoluutiota.

Ampparit Oy teki kummastakin sovelluksesta Ampparit.com -sivustolle kehitysblogi viestit, joissa informoitiin käyttäjiä juuri julkaistuista uutuuksista. Kehitysblogi on hyvä viestimiskeino käyttäjäkunnan kanssa, johon käyttäjät voivat antaa palautetta ja kehitysideoita. [26; 27.]

6 Tulokset

Esimerkkiprojektit esittelevät, miten web-sivuista saadaan Windows Phone yhteensopivat ilman kolmannen osapuolen kirjastoja. Jos yrityksellä on jo olemassa oleva HTML5-sivusto, niin näillä vinkeillä ne saadaan nopeasti tukemaan myös Windows Phonea. Esimerkeissä myös näytetään, miten saadaan perus Windows Phone -sovellusmalli web-muotoon, jota näin ollen voi hyvin käyttää muillakin alustoilla.

Windows Phone 8:n ratkaisussa hyviä tuloksia saa jo pelkästään metatageilla. Viewportin lisäys ja tap-highlightin poistaminen käytöstä tuovat jo natiivisovelluksen tuntua. Sovelluksen Back-näppäimen käsittelyyn on lisätty ehto, joka antaa luvan poistua sovelluksesta, jos sivuhistoriassa ei pääse taaksepäin. Windows Phone 7:ssä ei ole tap-highlightin poistomahdollisuutta, mutta viewport toimii samalla tapaa. Myös Back-näppäimen käsittely onnistuu, mutta sivuhistorian joutuu tekemään itse.

Ampparit.com-sovellukset valmistuivat suhteellisen nopeasti. Projekti käynnistyi virallisesti tammikuun lopulla ja sovellukset olivat sovelluskaupassa maaliskuun lopulla. Viivästyksiä tuli jonkin verran, esim. aikaa meni Windows Phone 8 -laitteen saamisessa ja Windows Phone -kehittäjätilin valmistumisprosessissa.

Esimerkkiprojektit ja Ampparit.com-sovellukset ovat kummatkin tehty erikseen Windows Phone 7:lle ja Windows Phone 8:lle. Vaikka Windows Phone 7:n tuki HTML5-sovelluksia kohtaan onkin huono, niin sitä kannattaa silti tukea ns. hybridiratkaisuilla.

Ampparit.com-sovelluksen Windows Phone 8 -versiossa käyttäjä näkee etusivulla vain selaimen, mutta Windows Phone 7 -version hybridiratkaisussa käyttäjälle näytetään tekstikenttä, selain ja toimintopalkki. Ampparit.com HTML5-sivustosta on tehty omat määritteet Windows Phone 7:ää varten, josta on piilotettu otsikko ja toimintopainikkeet, koska ne eivät liiku käyttäjän vierityksen mukana. Otsikko, johon tulee selattavan sivun nimi, esim. UUTISET – SUOSITUIMMAT lähetetään Windows Phone -sovellukseen window.external.notify -komennolla ja laitetaan otsikko näkymään sovelluksen ylälaidassa olevaan tekstikenttään. Näin ollen käyttäjä näkee kokoajan millä sivulla uutisia selailee, niin kuin Android- ja Windows Phone 8 -sovelluksissa. Toimintopalkissa olevat painikkeet valikko ja valinnat on linkitetty lähettämään HTML5-sivulle JavaScript-funktio kutsu, josta sivu aukaisee halutun valikon.

7 Johtopäätökset

Windows Phone ja HTML5 sovelluskehitys alkaa Windows Phone 8 -version myötä näyttää hyvältä. Microsoft on panostanut paljon selaimeensa ja tuonut hyviä asetuksia paremman käyttäjäkokemuksen luomiseksi. Parannettavaa on silti paljon ja kehittäjänäkökulmasta herää paljon kysymyksiä. Tietyt asetukset, jotka ovat Android- ja iOS-selaimissa jo perusjuttuja puuttuvat edelleen Windows Phone:n Internet Exploreriin perustuvasta selaimesta. Selaimessa jostain syystä voi vierittää sivua yli sen mittojen jolloin paljastuu, että sovellushan on selain. Tähän ongelmaan ei ole helppoa ratkaisua, mutta Microsoft voisi sen helposti korjata. Sen lisäksi eleitä pitäisi pystyä muokkaamaan, esim. multitouchin avulla voisi luoda todella hyvän käyttäjäkokemuksen web-sivustoille. Suunta on kuitenkin hyvä ja voi olettaa, että Windows Phone 9 ja Internet Explorer 11 voivat olla jo loistavia HTML5-sovellusten tukemisessa.

Windows Phone 7 on HTML5-sovellusten kannalta huono alusta, mutta kuten Ampparit.com ratkaisusta selviää, niin hybridisovellus on hyvä kompromissi. Sovellus pitää vain luoda niin, että itse sovellukseen tulevat palvelinkutsut ovat vakioita, joita ei tarvitse muuttaa jokaisessa päivityksessä. Näin ollen saadaan minimoitua itse sovelluksen päivitystarve ja voidaan keskittyä HTML5-sivuston kehittämiseen.

HTML5 on pitkään ollut uutisotsikoissa ja aina luvattu, että kohta se yleistyy ja natiivisovellukset kuolevat. Näin ei ole kuitenkaan vielä käynyt, sillä natiivisovellukset ovat edelleen paljon nopeampia ja sulavampia. Isot yritykset käyttävät edelleen natiivisovelluksia mobiilikäyttöjärjestelmissä, koska ne ovat paljon nopeampia ja näin ollen käyttäjät ovat tyytyväisiä. Tekniikka kuitenkin kehittyy kokoajan ja HTML5 saavuttaa pikkuhiljaa nopeudessa. HTML5:n parhain puoli on mielestäni sen päivitettävyys varsinkin palvelinkäytössä, koska kun tekee muokkauksen ja päivittää sen palvelimelle, niin muutos on voimassa kaikilla käyttäjillä heti. Natiivisovelluksissa se on aina paljon isompi operaatio, joka voi kestää montakin päivää. Natiivisovelluksessa tehdään ensin muokkaus ja pistetään sovellus menemään sovelluskaupan hyväksymisprosessin läpi, jonka jälkeen käyttäjän pitää vielä asentaa päivitys.

HTML5 on näillä näkymin kuitenkin tulevaisuuden alusta ja sen käytön määrä tulee kasvamaan paljon, kunhan saadaan yhteneväiset ratkaisut kaikille alustoille. Tämä tarkoittaa taas sitä, että jokaisella alustalla olisi sama tai samanlainen selain, joka käsittää kaikki samat toiminnot, kuin mitä HTML5-standardi sisältää. Sellaista tukea saattaa joutua odottelemaan jonkin aikaa, koska esim. suurimmat käyttöjärjestelmäkehittäjät Apple, Google ja Microsoft eivät varmaankaan ikinä tule olemaan täysin samaa mieltä tietyistä asioista. Tällä hetkellä kuitenkin Google ja Apple ovat HTML5-näkemyksissään lähellä toisiaan. Toivottavasti Microsoft kuulee HTML5-kehittäjien toiveita ja alkaa noudattaa standardia paremmin.

Lähteet

1. Ampparit Oy. Ampparit – Sensing the Web. 2013. http://ampparit.fi/. 23.4.2013.

2. Ampparit Oy. Tietoa – Ampparit Mini. 2013. http://mini.ampparit.com/tietoa. 24.4.2013.

3. Ampparit Oy. Tietoa – Ampparit Lite. 2013. http://lite.ampparit.com/tietoa. 24.4.2013.

4. Microsoft Corporation. Microsoft Unveils Windows Phone 7 Series. 2010. https://www.microsoft.com/en-us/news/press/2010/feb10/02-15mwc10pr.aspx. 23.4.2013.

5. Microsoft Corporation. Windows Phone -ominaisuudet. 2013. http://www.windowsphone.com/fi-fi/features. 23.4.2013.

6. HTML5Rocks. Why HTML5 rocks. 2013. http://www.html5rocks.com/en/why. 23.4.2013.

7. Mayo Joe. What is C#? 2013. http://csharp-station.com/. 23.4.2013.

8. Microsoft Corporation. What is XAML? 2013. http://msdn.microsoft.com/en-us/library/cc295302.aspx. 23.4.2013.

9. HTML.net. What is CSS? 2013. http://www.html.net/tutorials/css/lesson1.php. 23.4.2013.

10. W3Schools. JavaScript Introduction. 2013. http://www.w3schools.com/js/js_intro.asp. 23.4.2013.

11. Chapman Stephen. What Is JavaScript? 2013. http://javascript.about.com/od/reference/p/javascript.htm. 23.4.2013.

12. Microsoft Corporation. Getting Started with Visual Studio. 2013. http://msdn.microsoft.com/library/vstudio/ms165079.aspx. 23.4.2013.

13. Microsoft Corporation. WebBrowser control for Windows Phone. 2013. http://msdn.microsoft.com/en-us/library/windowsphone/develop/ff431797(v=vs.105).aspx. 23.4.2013.

14. Microsoft Corporation. WebBrowser.Navigate Method (URI). 2013. http://msdn.microsoft.com/en-us/library/windowsphone/develop/ff403407(v=vs.105).aspx. 23.4.2013.

15. Microsoft Corporation. WebBrowser.Navigate Method (URI, Byte[], String). 2013. http://msdn.microsoft.com/en-us/library/windowsphone/develop/ff626636(v=vs.105).aspx. 23.4.2013.

16. Microsoft Corporation. WebBrowser.InvokeScript Method. 2013. http://msdn.microsoft.com/en-us/library/windowsphone/develop/ff403616(v=vs.105).aspx. 23.4.2013.

17. Gross, C. Using WebBrowser.Document.InvokeScript() to mess around with foreign JavaScript. 2010. http://www.codeproject.com/Tips/60924/Using-WebBrowser-Document-InvokeScript-to-mess-aro. 23.4.2013.

18. Microsoft Corporation. WebBrowser.ScriptNotify Event. 2013. http://msdn.microsoft.com/en-us/library/system.windows.controls.webbrowser.scriptnotify(v=vs.95).aspx. 23.4.2013.

19. IE Mobile Team. The IE Mobile Viewport on Windows Phone 7. 2010. http://blogs.msdn.com/b/iemobile/archive/2010/11/22/the-ie-mobile-viewport-on-windows-phone-7.aspx. 23.4.2013.

20. Warren Tom. Windows Phone 8 will support 1080p displays by the end of the year. 2013. http://www.theverge.com/2013/4/9/4196022/windows-phone-8-1080p-resolution-support-gdr3-update. 23.4.2013.

21. Eberhardt Colin. Suppressing Zoom and Scroll interactions in the Windows Phone 7 WebBrowser Control. 2011. http://www.scottlogic.co.uk/blog/colin/2011/11/suppressing-zoom-and-scroll-interactions-in-the-windows-phone-7-browser-control/. 23.4.2013.

22. PhoneGap. PhoneGap, Cordova, and what's in a name. 2012. http://phonegap.com/2012/03/19/phonegap-cordova-and-what%E2%80%99s-in-a-name/. 23.4.2013.

23. Tutorialspoint. Software Testing Methods. 2013. http://www.tutorialspoint.com/software_testing/testing_methods.htm. 23.4.2013.

24. Microsoft Corporation. Upload and describe your XAP(s). 2013. http://msdn.microsoft.com/en-us/library/windowsphone/help/jj206723(v=vs.105).aspx. 23.4.2013.

25. Microsoft Corporation. How to target multiple versions with your app for Windows Phone. 2013. http://msdn.microsoft.com/en-us/library/windowsphone/develop/jj206997(v=vs.105).aspx. 24.4.2013.

26. Ampparit Oy. Windows Phone 8 -sovellus julkaistu! 2013. http://www.ampparit.com/blogi/11614552. 24.4.2013.

27. Ampparit Oy. Windows Phone 7 -sovellus julkaistu. 2013. http://www.ampparit.com/blogi/10681210. 24.4.2013.