Kemi Tornion ammattikorkeakoulu

PhoneGap-sovelluskehyksen soveltuvuus mobiilisovelluksen tuottamiseen Android-, Windows Phone- ja iPhone-alustoille

Kimmo Satta & Riku Korkeakoski

Kaupan ja kulttuurin opinnäytetyö
Tietojenkäsittelyn koulutusohjelma
Tradenomi

TORNIO 2013

ALKUSANAT

Haluamme kiittää opettajiamme Yrjö "Ykä" Koskenniemeä ja Markku Henrikssonia mahdollisuudesta osallistua mobiilisovelluksen luontiin liittyvään projektiin, sekä tuesta projektin aikana.

Lisäksi haluamme kiittää Ylläksen väkeä avusta sovelluksen kehityksessä.

TIIVISTELMÄ

KEMI-TORNION AMMATTIKORKEAKOULU

Koulutusohjelma:	Tietojenkäsittely
Opinnäytetyön tekijä(t):	Kimmo Satta & Riku Korkeakoski
Opinnäytetyön nimi:	PhoneGap-sovelluskehyksen soveltuvuus mobiilisovelluksen tuottamiseen Android-, Windows Phone- ja iPhone-alustoille
Sivuja (joista liitesivuja):	55 (12)
Päiväys:	13.11.2013
Opinnäytetyön ohjaaja(t):	Yrjö Koskenniemi

Toteutimme työharjoittelussa mobiilisovelluksen Ylläksen Matkailuyhdistykselle. Opinnäytetyömme aiheena on mobiilisovelluksessa käyttämämme PhoneGap-sovelluskehys ja sen soveltuvuus toteutettaessa ohjelmaa useammalle laitealustalle. Tavoitteenamme on käsitellä PhoneGapin toimintaa, ominaisuuksia ja sen suomia etuja ja haittoja. Lisäksi käsittelemme sovelluksen toimintojen toteutusta ja siinä kohdattuja ongelmia.

Tutkimme miten PhoneGap helpottaa sovelluksen siirtämistä alustalta toiselle. Tutkimuksen pohjana käytämme tekemäämme sovellusta, joka aluksi toteutettiin Androidille ja sen jälkeen siirrettiin Windows Phone- sekä iPhone -alustoille. Sovelluksen teossa käytämme jQuery- ja jQuery Mobile -kirjastoja.

Aineistona työssämme on käytetty sovelluksen lisäksi aihetta käsitteleviä internet-lähteitä, jotka ovat pääasiassa PhoneGap- ja jQuery Mobile -dokumentaatiot.

Opinnäytetyötä tehdessämme opimme PhoneGap-sovelluskehyksen käyttöä, sekä sen suomat edut. Sovelluskehys helpottaa ja nopeuttaa sovelluksen siirtämistä alustalta toiselle, koska ohjelmaa ei tarvitse kirjoittaa jokaisen laitteen natiivikielellä, vaan sovelluskehys kääntää HTML- , CSS- ja JavaScript-kielet alustakohtaiselle kielelle.

Asiasanat: PhoneGap, mobiiliohjelmointi, jQuery Mobile, Android, iPhone, Windows Phone, JavaScript

ABSTRACT

KEMI-TORNIO UNIVERSITY OF APPLIED SCIENCES

Degree programme:	Bachelor of Business Administration
Author(s):	Kimmo Satta & Riku Korkeakoski
Thesis title:	The applicability of PhoneGap mobile development framework to develop mobile applications for Android, Windows Phone, and iPhone platforms
Pages (of which appendixes):	55 (12)
Date:	13.11.2013
Thesis instructor(s):	Yrjö Koskenniemi

The outcome of the thesis research is a mobile application for Ylläksen Matkailuyhdistys that we developed during our practical training period. The topic of our thesis is PhoneGap mobile development framework that we used in the mobile application. We studied its applicability in developing an application for multiple platforms. Our objective is to address the functions, features, and the advantages and disadvantages of PhoneGap. We also discuss how we implemented the features and tackled the problems we encountered.

We examine if PhoneGap makes it easier to port an application. The basis of our research is our application that we first developed on Android and afterwards ported to Windows Phone and iPhone. We also utilized the jQuery and jQuery Mobile libraries.

The materials that we utilized are the mobile application and internet resources related to the topic of this research, i.e. the PhoneGap and the jQuery Mobile documentations.

During the process we learned the usage of PhoneGap application framework and the advantages it brings about. The framework makes it easier and faster to port applications to different platforms, mainly because there is no need to program with the native languages of the devices. Instead, the framework compiles the HTML, CSS, and JavaScript languages to device specific languages.

Keywords: PhoneGap, mobile programming, jQuery Mobile, Android, iPhone, Windows Phone, JavaScript

SISÄLLYS

1 JOHDANTO

1.1 Tausta

Aloitimme opinnäytetyön tekemisen toukokuussa 2013 Kemi-Tornion ammattikorkeakoulussa harjoitteluosuuden loppupuolella. Harjoittelun tavoitteena oli luoda mobiilisovellus Ylläksen alueen matkailijoiden käyttöön. Tämän sovelluksen kehittämisestä muotoutui opinnäytetyömme.

Mobiilisovellus on osa TG4NP-projektia (Tourist Guide for Northern Periphery), jossa Kemi-Tornion ammattikorkeakoulu oli mukana. Projektissa, joka on osa EU:n Northern Periphery Programmea, on tarkoitus tukea Euroopan pohjoisten alueiden matkailualaa luomalla mobiilipohjaisia palveluja parantamaan matkailijoiden kokemuksia. (TG4NP Tourist Guide for Northern Periphery 2010, hakupäivä 3.5.2013.)

TG4NP-projekti alkoi kesäkuussa 2010 ja päättyi syyskuussa 2013. Itse liityimme projektiin tammikuussa 2013 aloittaessamme harjoittelun. Tätä ennen koulussamme oli tehty Ylläksen mobiilisovelluksesta demoversio.

Koska meillä ei ollut aikaisempaa kokemusta mobiiliohjelmoinnista, aloitimme tutustumalla kyseisen demoversion koodiin ja toimintaan. Totesimme sen olevan toiminnallisuudeltaan vajava ja päätimme aloittaa sovelluksen tekemisen puhtaalta pöydältä käyttäen aikaisemmin luotua ominaisuusmääritelmää, joka löytyy liitteestä 2.

1.2 Tavoitteet

Tavoitteena harjoittelussa meillä oli luoda toimiva ja helppokäyttöinen sovellus kolmelle eri laitealustalle: Android, Windows Phone ja iPhone, käyttäen PhoneGap-sovelluskehystä. Tämän pohjalta tutkimme opinnäytetyössämme PhoneGapin soveltuvuutta saman sovelluksen tuottamiseen yhtäaikaisesti kyseisille laitealustoille.

Tuomme ilmi PhoneGapin etuja ja haittoja sovelluksen luonnissa, eri laitealustoille käännettäessä kohtaamiamme ongelmia ja niihin löytämiämme ratkaisuja.

1.3 Rakenne

Kerromme aluksi luvussa 2 sovelluksen suunnittelusta, jonka jälkeen käymme läpi työvälineet ja ohjelmointikielet sekä Android-, Windows Phone- ja iPhone-kehitysympäristöjen asennuksen. Kerromme lisäksi PhoneGapin toiminnasta ja käyttämästämme jQuery-kirjastosta.

Luvussa 3 käsitellään sovelluksen rakenne ja toiminta. Luku on jaettu osiin siten, että sovelluksen jokainen sivu on yksi alaluku. Näissä kerromme kyseisen sivun toiminnasta, toteutuksesta, mahdollisista ongelmatilanteista ja ongelmatilanteiden ratkaisuista.

2 SUUNNITTELU JA TYÖVÄLINEET

2.1 Suunnittelu

Tutustuimme jo olemassa olemaan ominaisuusmäärittelyyn, jonka pohjalta suunnittelimme sovellukseen tulevat ominaisuudet. Ensimmäisenä suunnittelimme päävalikon ulkoasun, jossa otettiin huomioon sovelluksen käytettävyys aurinkoisessa rinteessä (tumma värimaailma). Tämän jälkeen lähdimme toteuttamaan sovellusta osa kerrallaan Scrum-menetelmää soveltaen.

2.2 Ohjelmointivälineet

Sovellus on suurilta osin toteutettu käyttäen PhoneGap-sovelluskehystä, joka kääntää HTML-, CSS- ja JavaScript-kielillä kirjoitetun koodin laitteiden natiivikielelle. JavaScriptissä apuna on käytetty jQuery 1.8.3- ja jQuery Mobile 1.2.0 -kirjastoja. PhoneGapin käyttämän Cordova-rajapinnan versio on 2.2.0.

Ohjelmointiympäristöinä käytössä olivat Eclipse Androidille, Microsoft Visual Studio Express 2010 for Windows Phone ja Xcode 4.2 iOS:lle. Lisäksi ohjelmointiin käytettiin Notepad++-editoria.

Testauksessa käytimme ohjelmointiympäristöjen omia testaustyökaluja, Firefox-selaimen Firebug-lisäosaa ja Chrome-selaimen omia kehitystyökaluja. Testipuhelimina olivat Samsung Galaxy S, Galaxy S3, iPhone 4S, Samsung Omnia 7, ZTE Blade ja Samsung Galaxy Tab 2.

2.3 Muut välineet

Kuvankäsittelyssä käytössä oli Adobe Photoshop, tiedostojen hallinnassa Dropbox, dokumentoinnissa Microsoft Visio kaavioiden luontiin sekä OpenOffice- ja LibreOffice-toimistotyökalut tekstinkäsittelyyn.

2.4 PhoneGap

PhoneGap on Adoben ylläpitämä mobiililaitteille suunnattu sovelluskehys, joka mahdollistaa mobiilisovellusten kehittämisen web-tekniikoita (HTML, CSS, JavaScript) käyttäen. Tämä tuo mukanaan myös sen edun, että sovelluksesta pystyy tekemään version usealle laitealustalle vähäisillä ohjelmakoodin muutoksilla. Sovelluksen käyttöliittymä näytetään laitteilla käyttäen hyväksi niiden sisäänrakennettuja internet-selaimia ilman niiden omia käyttöliittymäosia, kuten ikkunan kehyksiä tai osoitepalkkia. Sovellus siis näkyy sellaisena, että se vie laitteen koko leveyden ja pituuden. (Trice 2012, hakupäivä 7.10.2013.)

PhoneGap mahdollistaa myös pääsyn mobiililaitteen toimintoihin, kuten kameraan tai GPS-navigaatioon, sen tarjoaman rajapinnan kautta. Se hoitaa kommunikaation sille annettujen JavaScript-komentojen ja laitteen natiivikielen välillä. (Trice 2012, hakupäivä 7.10.2013.)

2.5 Kehitysympäristöjen asennus

2.5.1 Android

Android-version teossa käytetty Eclipse-ohjelmointiympäristö on vapaasti ladattavissa verkosta, sillä se on avoimen lähdekoodin lisenssillä toteutettu. Lisäksi Eclipseen tarvitaan Android Development Tools -lisäosa, joka mahdollistaa Android-sovellusten kehittämisen. (Installing the Eclipse Plugin 2013, hakupäivä 21.10.2013.)

Eclipse on pakatussa muodossa (.zip), ja se puretaan haluttuun kohdekansioon. Ohjelman käynnistyksen yhteydessä on mahdollista luoda omavalintainen workspace-kansio (työtila), johon projekti tallentuu. Työtilaa voidaan sovellusta käyttäessäkin vaihtaa. (Installing the Eclipse Plugin 2013, hakupäivä 21.10.2013.)

Ohjelman käynnistyttyä asennetaan ADT-lisäosa. Tämä tehdään Help-valikosta löytyvästä Install New Software -toiminnosta. Avautuvassa näkymässä painetaan "Add", laitetaan nimeksi ”ADT plugin” ja osoiteriville ”https://dl-ssl.google.com/android/eclipse/”. (Installing the Eclipse Plugin 2013, hakupäivä 21.10.2013.)

Seuraavaksi asennetaan erinäisiä lisäosia Android-versioihin ja työkaluihin liittyen. Tämä tehdään Eclipsen Window-valikosta valitsemalla Android SDK manager. Avautuvasta näkymästä voi valita asennettavia työkaluja ja API-versioita (rajapinta). Tools-osiosta on oltava asennettuna SDK tools ja platform tools, sekä vähintään yksi API-versio. (SDK Manager 2013, hakupäivä 21.10.2013.)

Lopuksi ladataan PhoneGap-sovelluskehys. Ladattu paketti sisältää projektiin liitettävät JavaScript- ja Java archive -tiedostot. Tiedosto voidaan purkaa vapaavalintaiseen paikkaan.

Tämän jälkeen luodaan Eclipse-ympäristössä uusi projekti. Valitaan File-valikosta New - new project. Android-kansiosta valitaan Android Application Project. Ohjelma avustaa käyttäjän lyhyen prosessin läpi, jossa projekti nimetään ja avataan. Projektin avauduttua luodaan uudet kansiot PhoneGap-sovelluskehyksen tarvitsemia tiedostoja varten. Projektin juurihakemistoon luodaan libs-kansio sekä assets-kansioon www-kansio. Libs-kansioon liitetään PhoneGap-paketissa oleva cordova.jar tiedosto, ja luotuun www-kansioon liitetään cordova.js-tiedosto. Lisäksi PhoneGapin mukana tullut xml-kansio kopioidaan res-hakemistoon. (Getting Started with Android 2013, hakupäivä 24.10.2013.)

Viimeisenä muokataan res-hakemistossa olevaa MainActivity.java-tiedostoa, johon lisätään PhoneGap-paketin käyttöoikeus (taulukko 1), vaihdetaan luokan laajennus (taulukko 2) ja määritetään sovelluksen käynnistyessä avautuva sivu (taulukko 3). Lisäksi cordova.js-tiedoston lataus on lisättävä halutuille sivuille (taulukko 4). (Getting Started with Android 2013, hakupäivä 24.10.2013.)

Taulukko 1. PhoneGap-paketin käyttöoikeuden lisäys (Getting Started with Android 2013, hakupäivä 24.10.2013)

```
Import org.apache.cordova.*;
```

Taulukko 2. Luokan laajennuksen vaihto (Getting Started with Android 2013, hakupäivä 24.10.2013)

```
Public class HelloCordovaActivity extends DroidGap {}
```

Taulukko 3. Sovelluksen käynnistyessä ladattavan sivun määrittäminen (Getting Started with Android 2013, hakupäivä 24.10.2013)

```
Super.loadUrl("file:///android_assets/www/index.html");
```

Taulukko 4. Cordova.js-tiedoston lisääminen sovellukseen (Getting Started with Android 2013, hakupäivä 24.10.2013)

```
<script type="text/javascript" charset="utf-8" src="js/cordova-
2.2.0.js"></script>
```

2.5.2 Windows Phone

Windows phone 7 -version ohjelmointiympäristönä oli Microsoft Visual Studio Express 2010 for Windows Phone. Ohjelma on ilmaiseksi ladattavissa osoitteessa "http://www.microsoft.com/en-us/download/details.aspx?id=27570". Ladattava paketti sisältää tarvittavat työkalut sovellusten tekemiseen Windows Phone 7.1- ja 7.5 -laitteille. Ohjelmaa voi käyttää 30 päivää, jonka jälkeen se on rekisteröitävä. Rekisteröinti on maksuton ja vaatii ainoastaan Microsoft-tilin, esimerkiksi Hotmail-sähköpostitilin. Windows Phone 8:lle ohjelmoitaessa käyttäjällä täytyy olla Windows 8 -käyttöjärjestelmä jolloin ladattava paketti on Windows Phone SDK 8.0. (Windows Phone 7 Platform Guide 2013, hakupäivä 28.10.2013.)

Windows Phone SDK:n asennuksen lisäksi tarvitaan PhoneGap-sovelluskehys. PhoneGap-paketista löytyy CordovaWP7xx_x.zip-niminen tiedosto, jossa ”x.x”-merkintä viittaa ladattuun PhoneGap-versioon. Tämä tiedosto sisältää PhoneGapin ominaisuudet, ja sitä voidaan käyttää suoraan sovelluksen pohjana. Tiedosto kopioidaan Visual Studion ProjectTemplates-kansioon, joka löytyy seuraavan polun takaa: ”My Documents\Visual Studio 2010\Templates\”. (Windows Phone 7 Platform Guide 2013, hakupäivä 28.10.2013.)

Tämän jälkeen uuden projektin luominen tapahtuu Visual Studion ”New Project” -ominaisuuden kautta. Avautuvalta listalta Template-valikosta valitaan CordovaWP7, ja projektille annetaan nimi sekä tallennuskansio. (Windows Phone 7 Platform Guide 2013, hakupäivä 28.10.2013.)

2.5.3 iPhone

iPhone-sovelluksen ohjelmointiympäristönä käytössä oli xCode 4.2. xCode-ohjelmisto on ladattavissa Applen App Storesta tai Apple Developer-sivustolta. Apple Developer-sivustolle pääsy vaatii rekisteröitymisen sovelluskehittäjäksi (Getting Started with iOS 2013, hakupäivä 29.10.2013).

Ohjelman asennuksen jälkeen ladataan PhoneGap ja asennetaan xCode Command Line Tools, jotta PhoneGap-sovelluskehystä voidaan käyttää. Asennus suoritetaan xCode-ohjelman Menu-valikosta avautuvasta Asetukset-linkistä. Avautuvan valikon Lataukset-välilehdeltä valitaan Command Line Tools asennettavaksi. (Getting Started with iOS 2013, hakupäivä 29.10.2013.)

Uuden projektin luominen aloitetaan OS X -käyttöjärjestelmän Terminal-sovellusta apuna käyttäen. Ladatun PhoneGap-paketin ”\lib\ios”-hakemistosta löytyvä bin-kansio raahataan Terminal.app-kuvakkeen päälle, jolloin avautuu uusi Terminal-ikkuna. Projekti luodaan syöttämällä avautuneeseen ikkunaan taulukko 5. mukainen komento. Tämän jälkeen juuri luotu projekti on avattavissa xCode-sovelluksessa. (Getting Started with iOS 2013, hakupäivä 29.10.2013.)

Taulukko 5. Uuden projektin luonti komentokehotteen kautta (Getting Started with iOS 2013, hakupäivä 29.10.2013)

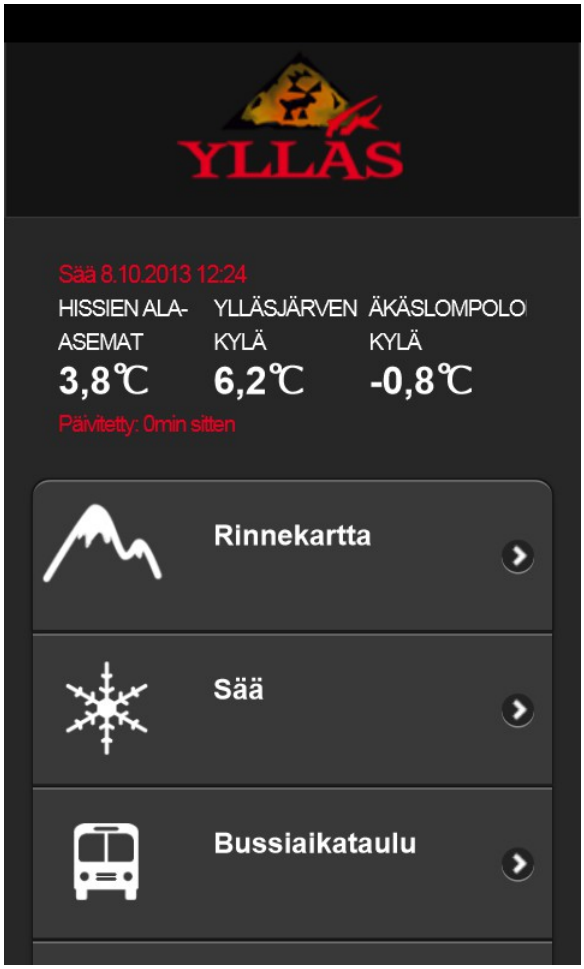
```
/create <project_folder_path><package_name><project_name>

<project_folder_path> is the path to your new Cordova iOS project
(it must be empty if it exists)
<package_name> is the package name, following reverse-domain style
convention
<project_name> is the project name
```

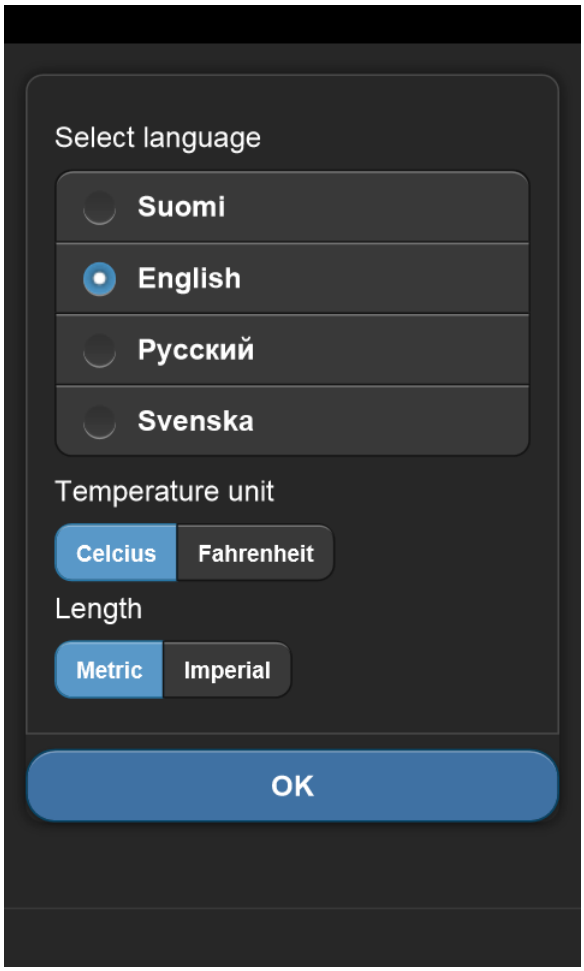
3 SOVELLUKSEN RAKENNE JA TOIMINTA

Liitteessä 1 on kuvattu sovelluksen yleisrakenne ja kunkin sivun toiminta.

3.1 Päävalikko



Kuvio 1. Sovelluksen päävalikko (Ylläs 2013)



Kuvio 2. Kielivalikko (Ylläs 2013)

3.1.1 Päävalikon toiminta

Käynnistymisen aikana Cordova avaa index.html-tiedoston, joka sisältää sovelluksen päävalikon (kuvio 1). Jos ohjelma ajetaan laitteessa ensimmäisen kerran, avautuu käyttäjälle kielivalikko, jossa voidaan määrittää sovelluksessa käytettävä kieli sekä lämpötila- ja pituusyksiköt. Tällä hetkellä valittavina kielinä ovat suomi ja englanti, muut käännökset ovat vielä kesken (ruotsi, venäjä ja japani). Android-versiossa oletuskielenä on laitteessa käytössä oleva kieli, muussa tapauksessa se on englanti.

Päävalikkoon tultaessa sovellus tarkistaa internet-yhteyden tilan. Jos yhteys löytyy, sovellus hakee Ylläksen palvelimelta sää-, rinne- ja hissitiedot JSON-muodossa (JavaScript Object Notation) ja tallentaa ne laitteen paikalliseen tallennustilaan (local storage). Säätiedot tulevat näkyviin päävalikon yläosaan. Jos taas internet-yhteyttä ei löydy, annetaan siitä käyttäjälle ilmoitus. Lisäksi sovellus seuraa yhteyden tilan

muutosta, ja sen muuttuessa joko hakee tiedot tai ilmoittaa käyttäjälle yhteyden tilan muutoksesta. Käyttäjä voi myös päivittää tiedot itse päävalikosta löytyvästä linkistä.

Laitteen takaisin-painike vie käyttäjän sivun hierarkiassa edelliselle sivulle. Kun ollaan ylimmällä hierarkiatasolla päävalikossa, näyttää sovellus viestin, jossa voidaan joko sulkea sovellus tai jatkaa sen käyttöä.

3.1.2 Päävalikon toteutus

Sivun päävalikko on toteutettu jQuery Mobilella (taulukko 6), ja sivujen vaihto tapahtuu AJAX-tekniikalla lukuun ottamatta Kartta-linkkiä. Kartta-linkin takaa löytyy rinnekartta, Sää-linkistä saa tarkemmat säätiedot, Bussiaikataulu näyttää paikalliset SkiBus-aikataulut, Palvelut-linkistä löytyvät ravintolat ja vuokraamot, sekä Ohje näyttää sovelluksen käyttöohjeen.

Taulukko 6. jQuery Mobile sivupohja (Getting Started with jQuery Mobile 2013, hakupäivä 7.10.2013)

```
<!doctype html>
<html>
  <head>
    <title>My Page</title>
    <meta name="viewport" content="width=device-width, initial-
    scale=1">
    <link rel="stylesheet" href="http://code.jquery.com/mobile/
    1.2.0/jquery.mobile-1.2.0.min.css">
    <script src="http://code.jquery.com/jquery-1.8.2.min.js">
    </script>
    <script src="http://code.jquery.com/mobile/1.2.0/jquery.mobile-
    1.2.0.min.js"></script>
  </head>
  <body>
    <div data-role="page">
      <div data-role="header">
      <h1>My Title</h1>
    </div><!-- /header -->

    <div data-role="content">
      <p>Hello world</p>
    </div><!-- /content -->

    <div data-role="footer">
      <h4>My Footer</h4>
    </div><!-- /footer -->

    </div><!-- /page -->
  </body>
</html>
```

Sivun <body>-tunnisteen (engl. tag) latauduttua kutsutaan init()-funktiota, joka odottaa, että sivu on latautunut kokonaan (taulukko 7). Tämän jälkeen kutsutaan onDeviceReady()-funktiota, joka tarkistaa näytetäänkö kielivalinta-sivu, hakee oikean kielen kieli.js-tiedostosta ja päivittää sen mukaan päävalikon, asettaa laitteen takaisin-painikkeen tapahtumankuuntelijan (engl. event listener) (taulukko 8), tallentaa internetistä haettavat tiedot laitteelle, sekä tarkistaa internet-yhteyden tilan ja päivittää sovelluksen tiedot.

Taulukko 7. PhoneGapin tapahtumankuuntelija, joka tarkistaa, että sivu on latautunut ja laite vastaa kutsuihin (Events 2013, hakupäivä 7.10.2013)

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

Taulukko 8. PhoneGapin takaisin-painikkeen tapahtumankuuntelija ja painikkeen toiminta (Events 2013, hakupäivä 7.10.2013; Ylläs 2013)

```
document.addEventListener("backbutton", function(e){
  if($.mobile.activePage.attr('id')=='main'){
    e.preventDefault();
    navigator.notification.confirm(
      '',           // message
      poistu,       // callback to invoke with index of button
pressed */
      kieli.poistu, // title
      kieli.ei+','+kieli.kylla // buttonLabels
    );
  }
  else navigator.app.backHistory();
}, true);
```

Tässä luvussa esitellyt navigaatioon liittyvät toiminnot ovat käytössä myös sovelluksen muilla sivuilla.

Koska sovelluksessa on käytössä useampi kieli, teimme erillisen kieli.js-tiedoston, johon on määritelty kaikki sovellukseen tuleva teksti eri kielillä. Ne on tallennettu JavaScriptin taulukoihin (engl. array), joista ne haetaan tarkistamalla käyttäjän valitsema kieli ja lataamalla kyseinen taulukko. Näin esimerkiksi kutsumalla kieli.suljettu voi antaa tulokseksi ”suljettu” tai ”closed”. Taulukoiden toimintaa on kuvattu suomen ja ruotsin osalta taulukossa 9.

Taulukko 9. Sovelluksen kielimääritysten kuvaus (Ylläs 2013)

```
var kieli;
function maaritaKieli() {
  if(window.localStorage.getItem('kieli')=="suomi") {
    kieli = {
      avoin: "Avoinna",
      suljettu: "Suljettu",
    }
  }
  else if(window.localStorage.getItem('kieli')=="ruotsi") {
    kieli = {
      avoin: " &Ouml;ppet",//Öppet
      suljettu: " St&auml;ngt",//Stängt
    }
  }
}
```

3.1.3 Päävalikon toteutuksen ongelmatilanteet

PhoneGap koostuu kahdesta koodikannasta, laitteen natiivikielestä ja JavaScriptistä, mikä aiheuttaa seuraavan tilanteen: JavaScript-koodia pystytään ajamaan ennen kuin PhoneGap on kokonaan latautunut, mikä taas aiheuttaa virhetilanteen sovelluksessa. Sovelluksen Android- ja iPhone-versioissa tätä ongelmaa ei tullut vastaan, mutta Windows Phone-versiossa funktiot tuli kutsua PhoneGapin deviceready-tapahtumankuuntelijan kautta (ks. Taulukko 7). Se odottaa, että Cordova on latautunut kokonaan, ennen JavaScript-koodin ajamista. (Events 2013, hakupäivä 7.10.2013.)

Kun siirsimme koodia Android-alustalta Windows Phone -alustalle, saimme joka sivulla taulukon 10 mukaisen virheilmoituksen. Tutkittuamme koodia ja kokeiltuamme eri ratkaisuja löysimme virheen aiheuttajaksi sivujen alusta puuttuvan <!DOCTYPE html>-deklaraation. Koska HTML:stä on erilaisia tyyppejä, tulee selaimelle sivun alussa kertoa mitä niistä käytetään. Androidin Webkit-pohjainen selain pystyi näyttämään sivut oikein ilman sitä, mutta Windows Phonen Internet Explorer-selain taas ei tähän pystynyt. (Don't forget to add doctype 2013, hakupäivä 21.10.2013.)

Taulukko 10. Virheilmoitus Windows Phonella <!DOCTYPE>-deklaraation puuttuessa (Microsoft Visual Studio Express 2010 for Windows Phone 2013)

```
A first chance exception of type 'System.SystemException' occurred in
```

```
Microsoft.Phone.Interop.dll
Error calling js to fire nativeReady event. Did you include cordova-
x.x.x.js in your html script tag?
```

Windows Phone -alustalla ilmeni myös seuraavanlainen ongelmatilanne: kun sivulla navigoitiin jQuery Mobilen AJAX:in avulla, laitteen takaisin-painike sulki sovelluksen riippumatta siitä kuinka syvällä sivun hierarkiassa oltiin. Koska karttasivua lukuun ottamatta muita sivuja ei varsinaisesti ladata selaimella suoraan, Windows Phone päättelee, että ollaan sivujen ylimmällä hierarkiatasolla ja näin ollen sulkee sovelluksen. Ongelma ratkesi sillä, että haimme jQuery Mobilelta kyseisen sivun tunnisteen. Sivun ollessa pääsivu, kysyttiin käyttäjältä haluaako hän sulkea sovelluksen. Muussa tapauksessa siirryttiin selainhistoriassa taaksepäin (taulukko 11) (jQuery.mobile.activePage 2013, hakupäivä 21.10.2013).

Taulukko 11. Windows Phonen takaisin-painikkeen toiminta (jQuery.mobile.activePage 2013, hakupäivä 21.10.2013; Ylläs 2013)

```
function handleBackButton() {
  if($.mobile.activePage.attr('id')=='main') {
    navigator.app.exitApp();
  }
  else history.back(-1);
}
```

Tietojen tallennukseen valitsimme HTML5-standardin mukanaan tuoman local storagen koska halusimme kaikille versioille yhtenäisen ratkaisun, eikä Windows Phone ainoana ympäristönä tukenut PhoneGapin tietokanta-rajapintaa (Web Storage 2013, hakupäivä 24.10.2013). Windows Phone-alustalla tietojen tallentaminen kuitenkin kesti minuutteja yrittäessämme tallentaa useita kymmeniä muuttujia kerralla. Suorituskyky parani huomattavasti kun tallensimme muuttujat ensin neljään taulukkoon (engl. array), jotka sitten tallensimme JSON-merkkijonona local storage -muuttujiin (taulukko 12).

Taulukko 12. Tietojen tallentaminen ja lukeminen (Ylläs 2013)

```
// tallennus
window.localStorage.setItem('ys_rinne',JSON.stringify(ys_rinne));

// lukeminen
JSON.parse(window.localStorage.getItem('ys_rinne');
```

Kun sovelluksessa palattiin kartta-sivulta takaisin päävalikkoon, oli päävalikko tämän jälkeen zoomattavissa, mikä ei ollut sovelluksessa haluttu toiminto. Ongelman syyksi ilmeni se, että kartta-sivulla oli määritelty viewport-metatunniste joka mahdollisti kartan zoomaamisen. Päävalikko-sivulla kyseisen ongelman aiheutti metatunnisteen puuttuminen. Kun taulukossa 13 kuvattu metatunniste lisättiin päävalikon <head>-tunnisteen sisään, alkoi sivu toimia normaalisti. Viewport-tunnisteessa user-scalable-määritys hallinnoi sitä, onko zoomaus käytössä, width sivun leveyttä näytöllä ja initial-scale zoomaustasoa sivulle tultaessa (Configuring the Viewport 2013, hakupäivä 24.10.2013).

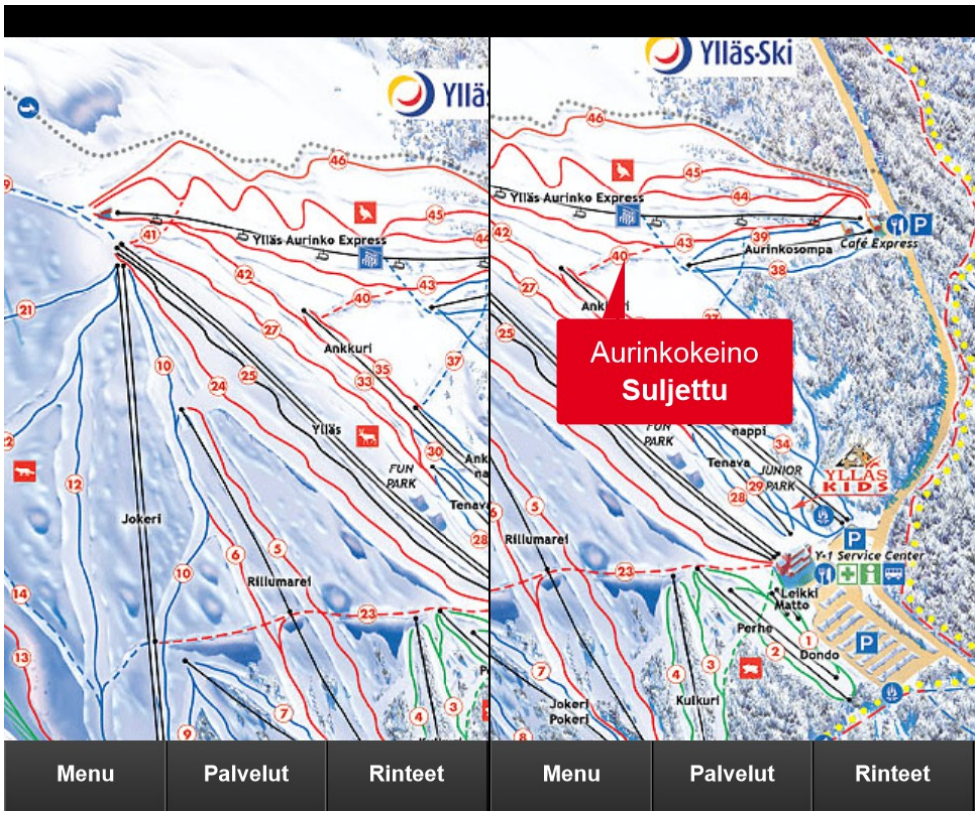
Taulukko 13. Viewport-metatunniste, joka poistaa zoomauksen käytöstä (Configuring the Viewport 2013, hakupäivä 24.10.2013)

```
<meta name="viewport" content="width=device-width, initial-
scale=1.0, user-scalable=false;" />
```

Lisäksi päävalikossa ilmeni Android-alustalla joillakin laitteilla gif-muotoisten animoitujen kuvien toimimattomuus. Tämä johtui siitä, että Android-käyttöjärjestelmän joistakin versioista puuttui kokonaan tuki niiden näyttämiseen. Päädyimme Android-alustalla näyttämään animaation sijaan käyttäjälle tekstimuotoisen ilmoituksen ”Ladataan...”.

Kieli.js-tiedostossa tuli esiin seuraava tilanne: haettaessa skandinaavisia tai kyrillisiä kirjaimia näkyviin tuli virheellinen merkki oikean kirjaimen sijaan. Ongelmana oli vääränlainen merkistökoodaus joka ei tukenut kyseisiä kirjaimia. Kun tiedoston merkistökoodaukseksi vaihdettiin UTF-16BE, näkyivät kirjaimet näytölle tulostettaessa oikein. (HTML5 Input Types 2013, hakupäivä 3.11.2013.)

3.2 Rinnekartta



Kuvio 3. Rinnekartta; oman sijainnin haku ja rinnetiedot (Ylläs 2013)

3.2.1 Rinnekartan toiminta

Sivun latauduttua näytetään rinnekartta, jonka jälkeen sovellus yrittää saada käyttäjän sijainnin (kuvio 3). Jos sijaintia ei löydy tai käyttäjä on kartta-alueen ulkopuolella, annetaan siitä ilmoitus ja vieritetään ruutu kartan keskelle. Sijainnin löytyessä koordinaatit muutetaan pisteiksi kartalla ja piirretään omaa sijaintia kuvaava merkki sekä vieritetään ruutu kyseiseen kohtaan. Käyttäjän sijainti päivitetään 20 sekunnin välein ja sijaintitiedot tallennetaan paikalliseen tallennustilaan.

Käyttäjän koskettaessa kartalla olevaa rinnenumeroa näytetään kyseisen rinteen nimi ja aukiolotieto (kuvio 3). Toista rinnenumeroa kosketettaessa vanha tieto piilotetaan ja uusi näytetään. Tiedot piilotetaan käyttäjän koskettaessa vanhaa "informaatiokuplaa".

Karttaa zoomattaessa tai vierittäessä valikolle lasketaan uusi korkeus ja tekstikoko laitteen käyttöjärjestelmästä ja versionumerosta riippuen. Valikko piilotetaan

zoomauksen tai vierityksen alkaessa ja näytetään sen päättyessä. Valikossa on linkit sovelluksen päävalikkoon, rinne- ja hissi- sekä palveluluetteloihin.

3.2.2 Rinnekartan toteutus

Rinnekartta pohjautuu Ylläksen verkkosivuille tehtyyn karttaan, jonka saimme käyttöömme projektin yhteydessä. Sieltä sovelluksemme hyödyntää piirrettyä karttakuvaa sekä GPS-koordinaatit kartalle pisteiksi muuttavaa funktiota.

Kartan avautuessa sovellus tarkistaa näytetäänkö käyttäjän, rinteen, hissin vai palvelun sijainti local storageen tallennetuista muuttujista. Kartalle tultaessa rinne- ja hissisivulta suoritetaan joko funktio naytaHissit() tai funktio naytaRinteet(). Ne näyttävät informaatiokuplan, josta näkee käyttäjän valitseman rinteen tai hissin sijainnin ja aukiolotiedon. Kyseisiä funktioita kutsutaan myös käyttäjän painaessa rinnenumeroa kartalla. Funktiot tarkistavat local storageen tallennetun hissin tai rinteen numeron, josta switch...case valintarakenteen avulla haetaan sen koordinaatit ja nimi. Koordinaatit on tallennettu karttakuvan jokaista pikseliä kuvaavina x- ja y-koordinaatteina, jotka jouduimme käsin merkitsemään jokaiselle rinteelle ja hissille. Funktiot laskevat myös kartan zoomaustason sekä sovellusikkunan leveyden ja korkeuden, jotta informaatiokuplan koko saataisiin sovitettua sovellusikkunan kokoon. Sovellusnäkymä vieritetään näyttämään valitun rinteen tai hissin sijainti käyttäen JavaScriptin self.scroll()-funktiota, joka on kuvattu taulukossa 14. Käyttäjälle näkyy myös koko ajan viesti latautumisesta, ja funktion suorittamisen jälkeen poistetaan local storagesta väliaikaiset muuttujat sekä piilotetaan viesti. Alueen palvelut näyttävä naytaPalvelu()-funktio toimii samalla tavalla.

Taulukko 14. Ruudun vierityksen ja sovellusikkunan koon laskeminen funktiossa naytaHissit() sovelluksen Android-versiossa (Ylläs 2013)

```
/* vieritetään ruutua näyttämään hissi kartalla. Muuttujat
hissi_coords_x ja hissi_coords_y ovat hissin sijainti karttakuvassa
pikseleinä zoomaustasolla 1 */
var ikkunanLeveys=((self.document.body.offsetWidth));
var ikkunanKorkeus=((self.document.body.offsetHeight));
self.scroll(((hissi_coords_x)-(ikkunan_leveys/2)),((hissi_coords_y)-
(ikkunan_korkeus/2)));
```

Käyttäjän painaessa sormella karttaa rekisteröi taulukossa 15 kuvattu funktio painalluksen, tarkistaa ettei käyttäjä ole painanut auki olevaa informaatiokuplaa (tämä saa aikaan kyseisen kuplan sulkemisen) ja tallentaa painalluksen x- ja y-arvot. Tämän jälkeen tarkistetaan kummalla puolella kuvan x-akselin keskikohtaa painallus on tapahtunut ja haetaan for-silmukkaa käyttäen arvoja lähimmäksi osuva rinne joko Ylläsjärven tai Äkäslompolon puolelta. Seuraavaksi rinteen arvot tallennetaan local storage -muuttujiin, piilotetaan mahdollisesti näkyvissä oleva edellinen informaatiokupla ja tallennetaan local storage -muuttujaan arvo joka estää sivun vierityksen ennenaikaisesti naytaRinteet()-funktiossa, jota kutsutaan viimeisenä.

Taulukko 15. Käyttäjän painallukset rekisteröivä funktio ja for-silmukka (Ylläs 2013)

```
// alustetaan muuttujat
var erox=10000;
var eroy=10000;

/* rekisteröidään käyttäjän painallus ja tallennetaan sen
x- ja y-arvot */
$("#kartta").live("tap", function(e) {
  var clickedX = e.pageX; //painalluksen x-arvo
  var clickedY = e.pageY; //painalluksen y-arvo
});

/* Käydään läpi lista rinteistä ja lasketaan käyttäjän painalluksen
x- ja y-koordinaatien ero rinteen koordinaateihin. Jos erotus < erox
+ eroy tallennetaan rinteen numero muuttujaan lahinRinne */
for(var i=1;i<=38;i++) {
  var eroxnyk = Math.abs(clickedX-eval('iy'+i+'rinx')); // esim. var
iy1rinx
  var eroynyk = Math.abs(clickedY-eval('iy'+i+'riny'));// esim. var
iy1riny
  if((eroxnyk+eroynyk)<(erox+eroy)) {
    erox = eroxnyk;
    eroy = eroynyk;
    lahinRinne = i;
  }
```

Käyttäjän sijainnin laskeminen kartalla tapahtuu funktiossa omaPaikka(), jota kutsutaan kartalle tultaessa ja tämän jälkeen 10 sekunnin välein kartalla oltaessa. Ensin tarkistetaan onko käyttäjän sijaintia ennestään määritelty. Sijaintitietojen puuttuessa annetaan ilmoitus ”Haetaan sijaintia”.

Seuraavaksi yritetään saada sijainti käyttäen PhoneGapin siihen tarkoitettua ja taulukossa 16 kuvattua funktiota. Sen valinnoista maximumAge määrittää, mikä on korkein hyväksytty välimuistiin tallennettu sijainnin ikä millisekunteina ennen kuin

sijaintia haetaan uudestaan. timeout taas määrittää aikavälin milloin kyseistä funktiota kutsutaan uudestaan. enableHighAccuracy puolestaan määrittää käytetäänkö sijainnin hakuun laitteen GPS-toimintoa. Tämän arvon ollessa epätosi, käytetään paikallistamiseen vain matkapuhelinverkkoja sekä langattomia verkkoja, joista saatu sijainti ei ole yhtä tarkka. Funktiota geoLocationSuccess() kutsutaan, mikäli sijainti saatiin haettua onnistuneesti. Se tallentaa saadut x- ja y-koordinaatit muuttujiin, joiden arvoilla kutsutaan nayta_sijainti()-funktiota, joka muuttaa annetut koordinaatit pisteiksi kartalla ja kutsuu näillä arvoilla nayta_sijaintia()-funktiota. Se puolestaan laskee käyttäjän sijainnin kertovan merkin paikan ja vierittää sovellusnäkymän sen kohdalle. nayta_sijainti()-funktion saimme Ylläksen verkkosivujen kartan tekijältä. onError-funktiota taas kutsutaan, mikäli sijaintia ei saatu. Se näyttää käyttäjälle siitä ilmoituksen ja kutsuu eiSijaintia()-funktiota, joka vierittää sovellusnäkymän kartan keskelle. Funktiot geoLocationSuccess() ja onError() ovat kuvattu taulukossa 17.

Taulukko 16. GPS-sijainnin hakeminen (Geolocation 2013, hakupäivä 28.10.2013)

```
navigator.geolocation.watchPosition(geoLocationSuccess, onError,{
  maximumAge : 10000,
  timeout:10000,
  enableHighAccuracy: true
});
```

Taulukko 17. geoLocationSuccess- ja onError-funktiot (Ylläs 2013)

```
function geoLocationSuccess(position) {
  window.localStorage.setItem('sijainti',position);
  hakupohj=position.coords.latitude;
  hakuita=position.coords.longitude;
  setTimeout(function() {nayta_sijainti(hakupohj, hakuita);},1000);
}

// sijainnin haku ei onnistunut, näyttää ilmoituksen
function onError(error) {
  navigator.notification.alert(
  kieli.eiSijaintia, // viesti
  eiSijaintia,       /* funktio, jota kutsutaan kun painetaan paini-
ketta */
  kieli.ilmoitus,    // viestin otsikko
  kieli.ok           // painikkeen teksti
  );
}
```

Tämän jälkeen tarkistetaan onko sijaintia aikaisemmin yritetty epäonnistuneesti hakea. Mikäli näin on, piilotetaan mahdolliset viestit käyttäjälle eikä sijantia yritetä hakea. Jos sijainti on jo aikaisemmin haettu, kutsutaan nayta_sijanti()-funktiota näillä arvoilla.

Kartan alareunassa oleva valikko on osittain toteutettu käyttäen jQuery Mobile-kirjastoa, mutta sen toimintaa on jouduttu muokkaamaan huomattavasti. Tästä kerrotaan lisää luvussa 3.2.3.

3.2.3 Rinnekartan toteutuksen ongelmatilanteet

Aluksi karttakuva toteutettiin 1200 x 765 -resoluutioisena jpg-muotoisena kuvana, mutta karttaa selattaessa vanhemmilla laitteilla tuli vastaan ongelmia suorituskyvyn kanssa. Karttaa zoomattaessa tai vierittäessä sovellus ei vastannut heti, vaan vasteaika saattoi olla jopa useita sekunteja. Seuraavaksi testasimme kuvan tallentamista gif- ja png-formaateissa, joista png-formaatti antoi parhaimman suorituskyvyn, mutta vasteaika oli silti liian suuri. Lopulta päädyimme jakamaan kuvan sataan pienempään palaan ja sijoittamaan sen sivulle -tunnisteiden avulla. Tämä paransi suorituskykyä huomattavasti ja viivettä vieritettäessä sekä zoomattaessa tuli vain harvoin vastaan.

Android-alustan WebView-näkymän (jota PhoneGap käyttää käyttöliittymän näyttämiseen) zoomaustoiminnon ollessa käytössä, tuli vieritettäessä tai zoomattaessa näkyviin automaattisesti muutaman sekunnin ajaksi kuvion 4 mukaiset Androidin sisäänrakennetetut zoomauskontrollit. Koska karttaa voi zoomata sormielein, ja koska kontrollit asettuvat oikeaan alakulmaan peittämään valikkoa, halusimme estää niiden näkymisen. Zoomauksen saa Androidilla otettua käyttöön sen MainActivity luokassa MainActivity.java tiedostossa, taulukon 18 mukaisesti. Siinä metodi setBuiltInZoomControls() ottaa käyttöön edellä mainitut kontrollit, mutta antaessamme sille arvoksi epätosi, katosi sovelluksen koko zoomaustoiminto käytöstä. Löysimme Androidin dokumentaatiosta metodin setDisplayZoomControls(), jolla kyseiset kontrollit pystyy poistamaan käytöstä ilman, että se vaikuttaa sovelluksen zoomaustoimintoon. Ongelmana oli, että se vaatii Android 3.0 tai sitä suuremman käyttöjärjestelmän version toimiakseen. Loimme kuitenkin funktion (katso taulukko 19), joka ottaa tuetuilla laitteilla kontrollit pois käytöstä, mutta jättää ne näkyviin vanhemmilla laitteilla. Ennen funktiota oleva @TargetApi(11) tarkoittaa, että funktio suoritetaan vain käyttöjärjestelmän version ollessa suurempi kuin 3 (API taso 11 tuli käyttöön Androidin versioissa 3.0) (TargetApi 2013, hakupäivä 4.11.2013; Codenames,

Tags, and Build Numbers 2013, hakupäivä 4.11.2013). (WebSetting 2013, hakupäivä 4.11.2013.)

Kuvio 4. Androidin WebView-näkymän zoomauskontrollit korostettuna (Ylläs 2013)

Taulukko 18. Zoomaustoiminnon ottaminen käyttöön Androidin WebView-luokassa (Ylläs 2013)

```
WebSettings ws = super.appView.getSettings();
ws.setBuiltInZoomControls(true);
ws.setSupportZoom(true);
```

Taulukko 19. Zoomauskontrollien poistaminen käytöstä Android-alustalla (Ylläs 2013)

```
@TargetApi(11)
public void zoomControlsDisable() {
  if(android.os.Build.VERSION.SDK_INT >= 11) {
    WebSettings ws = super.appView.getSettings();
    ws.setDisplayZoomControls(false);
  }
```

iPhone-alustalla kartassa esiintyi lisäksi seuraava ongelma: emme pystyneet zoomaamaan karttaa 1:1 suhdetta lähemmäksi, vaikkakin kauemmaksi zoomaaminen toimi. Tälle ei löytynyt selitystä, vaan korjasimme tilanteen siten, että suurensimme kartan resoluution kaksinkertaiseksi ja tallensimme kuvat kuten aiemmin. Tämän jälkeen muutimme viewport-metatunnisteesta seuraavat arvot: initial-scale=1, minimum-scale=0.5 ja maximum-scale=3.0. Nyt karttaa pystyi zoomaamaan ongelmitta.

Kartan alaosassa näkyvä valikko aiheutti suuren määrän ongelmia. Lähdimme toteuttamaan sitä aluksi Android-alustalle käyttäen jQuery Mobilen fixedtoolbar-liitännäistä, joka mahdollistaa sovellusnäkymän vierittämisen siten, että valikko pysyy ruudun alareunassa (Fixed toolbars 2013, hakupäivä 4.11.2013). Tämä ei kuitenkaan

toiminut halutulla tavalla muilla kuin Androidin 3.0 tai sitä uudemmilla versioilla, vaan se jäi aina näkymän ulkopuolelle vieritettäessä.

Seuraavaksi toteutimme valikon jQuery Mobilella ilman edellä mainittua liitännäistä, eli jätimme valikon sisältävästä -tunnisteesta pois data-role=”footer”- ja data-position=”fixed” -määritykset. Lisäsimme sivulle seuraavat jQuery-liitännäiset: jQuery Events for Start/Stop Scrolling, joka lisää tapahtumankuuntelijat näkymän vierityksen käynnistymiseen ja päättymiseen, sekä jquery.resizestop, joka taas lisää tapahtumankuuntelijan zoomauksen päättymiseen (Fatih 2012, hakupäivä 4.11.2013; Barbagallo 2013, hakupäivä 4.11.2013). Loimme lisäksi footerPosition()- ja footerPosition2()-funktiot, joista ensimmäinen tarkistaa onko käytössä oleva Androidin versio pienempi kuin 3.0 (PhoneGapin device.version -ominaisuus) tai onko funktiota kutsuttu zoomaustapahtuman päätyttyä (Device 2013, hakupäivä 4.11.2013). Tällöin se kutsuu funktiota footerPosition2(), joka puolestaan tarvittaessa laskee valikon koon, fonttikoon ja valikon sijainnin siten, että se sijoittuu sovellusnäkymään oikein. Lisäksi versionumeroa 3.0 vanhemmilla käyttöjärjestelmillä valikko piilotetaan zoomaus- ja vieritystapahtumien alussa ja otetaan uudelleen näkyviin funktion laskutoimitusten päätyttyä, koska muuten se zoomautuu kartan mukana eikä siirry karttaa vieritettäessä. Valikon toimintaa kuvataan taulukossa 20.

Valikkoa piti myös tämän jälkeen muokata Windows Phone- ja iPhone-alustoille, koska mikään alustoista ei täysin tukenut samoja JavaScript-ominaisuuksia, joita tarvittiin valikon koon, fonttikoon ja sijainnin laskemisessa. Esimerkiksi sovelluksen zoomauskerroin saadaan laskettua Androidille kaavalla window.outerWidth / window.innerWidth, Windows Phonelle kaavalla window.screen.deviceXDPI / 1 ja iPhonelle kaavalla window.outerWidth / $(document).width(). Lisäksi iPhonella piti laskea valikon sijainti erikseen versiota 6 vanhemmille laitteille. Windows Phone- ja iPhone -alustojen versiot funktiosta footerPosition2() löytyvät taulukosta 21.

Taulukko 20. Kartan valikon koon, fonttikoon ja sijainnin laskeminen Android-alustalla (Ylläs 2013)

```
// haetaan käyttöjärjestelmän versio
var version=parseInt(device.version);

// kutsutaan footerPosition2()-funktiota kun vieritys pysähtyy
```

```
$(window).bind('scrollstop', function(){
  if(version < 3) footerPosition2();
});


// kutsutaan footerPosition2()-funktiota kun zoomaustapahtuma päättyy
$(window).on('resizestop',200, function () {
   footerPosition2();
});

// valikon fonttikoko
var fonttiKoko=$('span.ui-btn-text').css('font-size');

/* kuuntelee käyttäjän kosketusta -> jos android-versio > 3 ja
multitouch, kutsuu footerposition funktiota. muuten muuttaa ts-
muuttujan arvksi 1 */
document.addEventListener('touchstart', function(e) {
  if (version>=3) {
    var touch1 = e.touches[0];
    var touch2 = e.touches[1];
    // jos multitouch
    if(touch1.pageX>0 && touch2.pageX>0) {
      touchExists='joo';
      footerPosition('resize');
      ts = null;
    }
    else ts = 1;
    touchExists='ei';
  }
  else ts = 1;
}, false);

/* tutkii, onko valikko piilotettu -> jos on, niin tutkitaan painaako
käyttäjä tällä hetkellä näyttöä -> jos ei, niin näytetään valikko */
function footerCheck() {
  if($('#footer').css('visibility')=='hidden') {
    if(touchExists=='ei'){
      document.getElementById('footer').style.visibility="visible";
    }
  }
}
// kutsutaan footerCheck()-funktiota 2 sekunnin välein
setInterval("footerCheck()", 2000);

// Piilottaa valikon jos android-versio < 3 tai jos zoomataan
function footerPosition(event) {
  if(version<3 || event == 'resize') {
    if($('#footer').css('visibility')=='hidden') footerPosition2();
    else {
      document.getElementById('footer').style.visibility="hidden";
    }
  }
};

/* näyttää valikon, laskee sen koon ja sijainnin sekä fonttikoon
zoomauksen mukaan */
function footerPosition2() {
  // zoomaustaso
  var zoomausKerroin = window.outerWidth / window.innerWidth;
  // laskee valikon korkeuden
  document.getElementById('footer').style.height = 4 * pixelRatio /
 (zoomausKerroin ) + "em";
```

```
  // tallentaa valikon korkeuden ja leveyden muuttujiin
  var footerKorkeus=Math.floor(parseFloat($('#footer').height()));
  var footerLeveys=Math.floor(parseFloat($('#footer').width()));

/* tutkitaan puhelimen käyttöjärjestelmän versio, jos <3 lasketaan
sen sijainti ja näytetään se, muuten näytetään */
  if(version<3) {
    $('#footer').css({
      top : $(window).scrollTop() + window.innerHeight-
            footerKorkeus+2+'px',
      left : $(window).scrollLeft() + window.innerWidth-
             footerLeveys+'px'
    });
    $('#footer').css('visibility',"visible");

  }
  else $('#footer').css('visibility',"visible");
  // määritetään valikon linkkien korkeus ja leveys
  $('#linkki,#linkki2').css({
    height : footerKorkeus+'px',
    width : window.outerWidth/zoomausKerroin/3 +'px'
  });
  $('#linkki3').css({
    height : footerKorkeus + 'px',
    width : window.outerWidth/zoomausKerroin/3 + 'px'
  });
  // määritetään valikon fonttikoko
  $('#linkki,#linkki2,#linkki3').css({
    'font-size' : 13 * pixelRatio / zoomausKerroin + 'pt',
    'height' : footerKorkeus+'px'
  });
};
```

Taulukko 21. Sovelluksen footerPosition2()-funktion erot Windows Phone- ja Android -alustoilla (Ylläs 2013)

```
// Windows Phone alkaa

// näyttää valikon, laskee sen koon ja sijainnin sekä fonttikoon
zoomauksen mukaan
function footerPosition2() {
  //zoomaustaso
   var zoomausKerroin =(window.screen.deviceXDPI) / dpi;

$('#alapalkki').css({
    top : Math.ceil(window.pageYOffset+
    document.body.offsetHeight/zoomausKerroin-
    14/zoomausKerroin)+1+'px',
    left : window.pageXOffset+'px',
    height: 3 / zoomausKerroin + 'em',
    width : window.innerWidth / zoomausKerroin+1 +'px',
    'border-top' : 1/zoomausKerroin +'px'
  });

  // määritetään valikon fonttikoko
  $('#linkki,#linkki2,#linkki3').css({
    'font-size' : 11 / zoomausKerroin + 'pt'
  });
```

```
  // valikko näytetään
  $('#alapalkki').show('50');

}

// Windows Phone päättyy
// iPhone alkaa

function footerPosition2() {
  // zoomaustaso
   var zoomausKerroin = window.outerWidth / $(document).width();
   var footerKorkeus=Math.floor(parseFloat($('#footer').height()));
   var footerLeveys=Math.floor(parseFloat($('#footer').width()));


  // laskee valikon korkeuden
   document.getElementById('footer').style.height = 2 *
   resoluutioKerroin / (zoomausKerroin ) + "em";

  // määritetään valikon linkkien korkeus ja leveys
  $('#linkki,#linkki2').css({
  height : footerKorkeus+'px',
    width : window.outerWidth/zoomausKerroin/3 +'px'
   });
  $('#linkki3').css({
    height : footerKorkeus + 'px',
    width : window.outerWidth/zoomausKerroin/3 + 'px'
  });

  // määritetään footerin fonttikoko
  $('#linkki,#linkki2,#linkki3').css({
  'font-size' : 10 / zoomausKerroin + 'pt',
  'height' : footerKorkeus+'px'
  });

  if(device.version<'5') {
    $('#footer').css({
      top:Math.ceil(window.pageYOffset+document.body.offsetHeight /
      zoomausKerroin-footerKorkeus+2)+'px',
      left:window.pageXOffset,
      width:101/zoomausKerroin+'%'
    });
    }
};

// iPhone päättyy
```

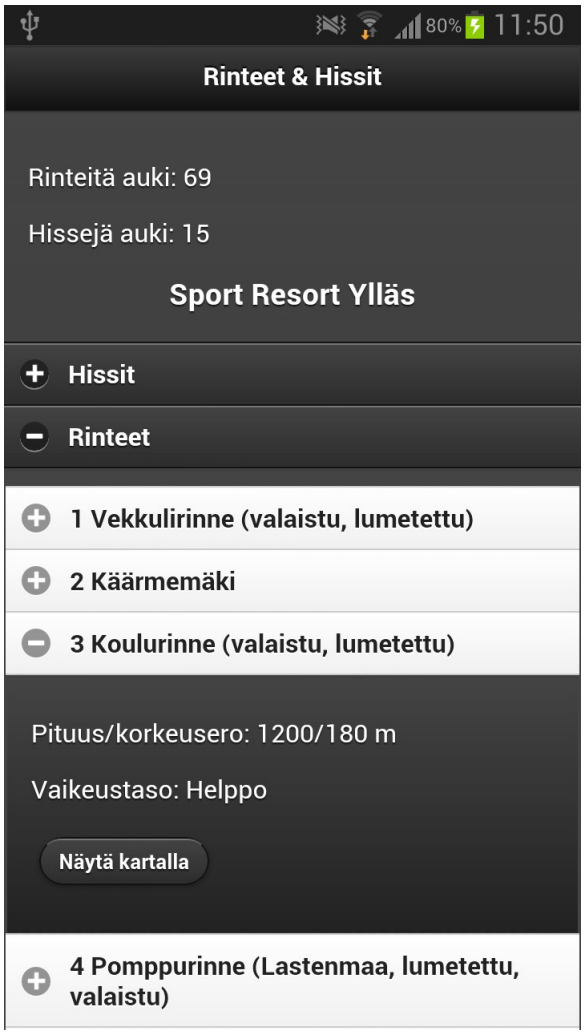
iPhone-versiossa ongelmaksi muodostuivat valikon painikkeiden paikat. Karttakuvaa zoomattaessa tai vierittäessä valikko näytti päällepäin toimivan oikein, eli se pysyi kuvan alareunassa kiinni ja oikean kokoisena. Tästä huolimatta linkkien sisältö, eli uudelle sivulle ohjaavat "href"-määritykset, ei seurannut matkassa, vaan ne jäivät alkusijoilleen näkymättömiksi painikkeiksi. Tämän ongelman ratkaisuksi loimme kartalle click-tapahtuman (engl. event), jossa ensin tarkistetaan painaako käyttäjä alavalikon päällä. Käyttäjän painalluksen osuessa alavalikon ulkopuolelle, funktion

suoritus lopetetaan. Jos taas käyttäjä painaa valikon päällä, tarkistetaan seuraavaksi painalluksen sijainti x-akselilla. Tämän arvon mukaan käyttäjä siirtyy kyseisessä kohdassa olevan linkin osoittamaan paikkaan (taulukko 22).

Taulukko 22. iPhone-version click-tapahtuma alavalikon linkkejä varten (Ylläs 2013)

```
$("#kartta").click(function(event) {
  if(event.clientY >= document.body.offsetHeight / screenCssPixelRa-
tio
  - $('#footer').height() && event.clientY <=
  document.body.offsetHeight / screenCssPixelRatio) {
    if(event.clientX <= document.body.offsetWidth /
    screenCssPixelRatio / 3){
      window.location.href="../../index.html";
    }
    else if(event.clientX >= document.body.offsetWidth /
    screenCssPixelRatio / 3 * 2){
      window.location.href="../slopes/slopes.html";
    }
    else window.location.href="../services/services.html";
  }
});
```

3.3 Rinteet ja hissit



Kuvio 5. Luettelo rinteistä ja hisseistä (Ylläs 2013)

3.3.1 Rinne- ja hissivalikon toiminta

Sovelluksen paikallisesta tallennustilasta haetaan rinne- ja hissitiedot ja luodaan neljä laajennettavaa valikkoa. Valikoissa on lueteltu rinteiden ja hissien nimet ja niitä koskettamalla saadaan näkymään tarkemmat tiedot sekä painike, josta valitun kohteen voi paikantaa kartalla. Kiinni olevat rinteet ja hissit esitetään harmaalla ja auki olevat mustalla tekstillä (kuvio 4).

Jos paikallisesta tallennustilasta ei löydy tietoja tai tiedot ovat vanhentuneita (yli 24h), näytetään tiedoista ainoastaan nimet ja annetaan käyttäjälle ilmoitus niiden vanhentumisesta.

Sivun yläosassa näytetään avoimien rinteiden ja hissien lukumäärä.

3.3.2 Rinne- ja hissivalikon toteutus

Sivun sisältö on toteutettu jQuery Mobilella, ja sivulle tultaessa kirjoitetaan otsikko ja luodaan tyhjät -tunnisteet tunturin molempien puolien (Äkäslompolo ja Ylläsjärvi) rinteille ja hisseille. Aluksi sovellus tarkistaa milloin tiedot on tallennettu paikalliseen tallennustilaan, jonka jälkeen tallennusaikaa verrataan nykyiseen aikaan. Jos tiedot ovat vanhentuneita, annetaan käyttäjälle siitä ilmoitus.

Rinne- ja hissitiedot kirjoitetaan käyttäjän avatessa jonkin laajennettavista valikoista. Tämä on toteutettu jQuery Mobilen expand- ja collapse-tapahtumia käyttämällä. Käyttäjän avatessa valikon toteutuu expand-tapahtuma (taulukko 23), jossa kutsutaan toista funktiota, joka kirjoittaa osion tiedot avautuvaan listaan.

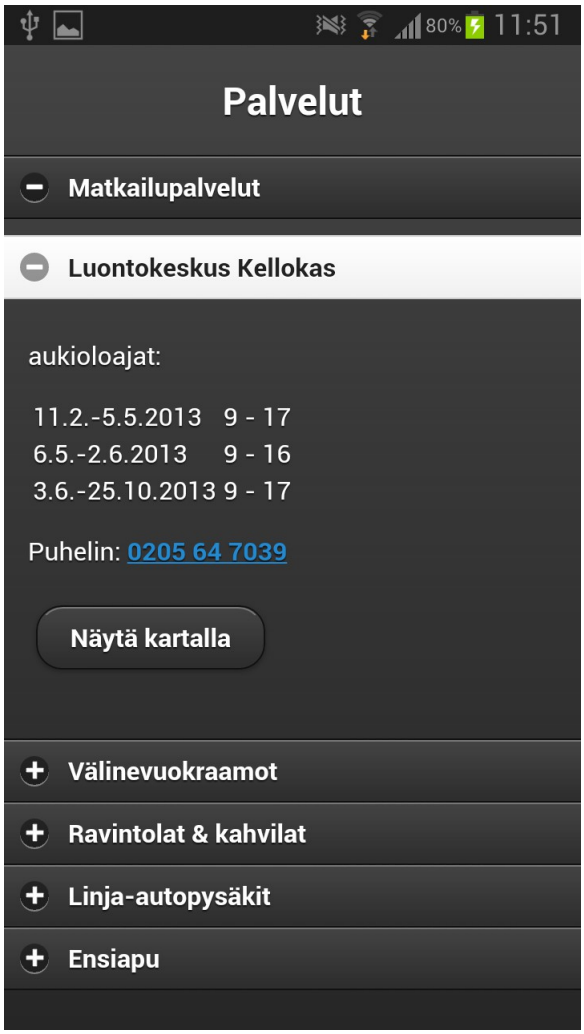
Taulukko 23. jQuery Mobile expand- ja collapse-funktiot (Collapsible content on-click events 2010, hakupäivä 21.10.2013)

```
$('div').live('expand', function(){
  function KirjoitaRinneTiedot();
}).live('collapse', function(){
});
```

3.3.3 Rinne-ja hissivalikon toteutuksen ongelmatilanteet

Aluksi toteutimme rinne- ja hissitietojen kirjoittamisen heti sivun latautuessa. Tämä toimi hyvin uudemmassa ja tehokkaammassa Android-puhelimessa, mutta vanhemmalla puhelimella listauksen luonti kesti todella pitkään. Windows Phone-alustalla tämä aiheutti valikoiden ulkoasun latautumiseen ongelmia. Laajennettavat valikot eivät kaikki näkyneet oikein sovelluksessa.

3.4 Palvelut



Kuvio 6. Listaus palveluista (Ylläs 2013)

3.4.1 Palvelut-valikon toiminta

Palvelut.js-tiedostosta luetaan palveluiden tiedot, lukuun ottamatta ravintolatietoja, jotka haetaan Ylläksen palvelimelta. Nämä tiedot kirjoitetaan laajennettaviin valikoihin, jotka on lajiteltu palveluiden tyyppien mukaan. Muista tiedoista löytyy palvelusta riippuen aukioloajat, puhelinnumero sekä paikannuspainike (kuvio 6).

3.4.2 Palvelut-valikon toteutus

Sivulla listatut palvelut ovat ravintoloita lukuun ottamatta poimittu Ylläksen kotisivulta ja kirjattu staattisesti palvelut.js-tiedostoon. Tiedostosta löytyy palvelun nimi ja sijainti, sekä mahdolliset aukioloajat ja puhelinnumero (taulukko 24). Sijaintitiedot, eli coordx- ja coordy-muuttujien arvot, ovat yksittäisten pikseleiden x- ja y-koordinaatit rinnekartalla, jotka haimme kartalta manuaalisesti. Sovelluksen pääsivulla kutsuttavassa

haePalvelut()-funktiossa laitteen paikalliseen tallennustilaan tallennetut ravintolatiedot kirjoitetaan palvelut.js-tiedostosta löytyvien tietojen kanssa data-muuttujaan, joka lopuksi kirjoitetaan sivulle käyttämällä trigger(”create”)-funktiota (kuvattuna osiossa 3.7.2).

Taulukko 24. Esimerkki palvelut.js-tiedoston sisällöstä (Yates 2012, hakupäivä 6.11.2013; Ylläs 2013)

```
var keskus1 = {
  nimi: "Luontokeskus Kellokas",
  auki: "<br /><table><tr><td>11.2.-5.5.2013</td><tr></table>",
  puhelin: '<a href="tel:0205 64 7039">0205 64 7039</a>',
  coordx: "1980",
  coordy: "936"
}
```

Sijaintitietojen kerääminen mahdollisti "Näytä kartalla"-painikkeen lisäämisen palveluihin. Tätä painamalla siirrytään kartta-sivulle ja palvelun tiedot lähetetään naytaPalvelu()-funktiolle, joka keskittää kartan palvelupisteen kohdalle ja piirtää informaatiokupla palvelupisteen sijaintikohtaan kartalla (taulukko 25). Puhelinnumeron linkki-tunnisteeseen on lisätty taulukon 24 mukainen tel-määritelmä, joka mahdollistaa puhelun soittamisen linkkiä painettaessa (Yates 2013, hakupäivä 6.11.2013).

Taulukko 25. Palvelutietojen lähetys sekä infokuplan piirtäminen kartalle (Ylläs 2013)

```
// palvelutietojen lähetys
function palveludata(palvelu) {
  window.localStorage.setItem('palvelupaikka','1');
  // käytetään karttaindexissä
  window.localStorage.setItem('palvelunimi',eval(palvelu).nimi);
  window.localStorage.setItem('palvelu_x',eval(palvelu).coordx);
  window.localStorage.setItem('palvelu_y',eval(palvelu).coordy);

  // naytaPalvelu()-funktiosta:
  // vieritetään ruutua näyttämään rinne kartalla
  self.scroll(((palvelu_coords_x*kerroin2)-(ikkunan_leveys/2)),((pal-
velu_coords_y*kerroin2)-(ikkunan_korkeus/2)));

  // piirretään infokupla palvelu-diviin
  $('#palvelu').html('<p class="speech" style="left:' + vasen+';
top:'
  + yla+'; height:auto; font-size:' + fonttiKoko + '"><span
  class="bold">' + palvelunimi + '</span></p>');
  $('p.speech').css('width', $('p.speech').width()+15);
}
```

3.5 Säätiedot

3.5.1 Säätietojen toiminta

Sovelluksen paikallisesta tallennustilasta haetaan ja näytetään sinne tallennetut säätiedot. Jos tietoja ei löydy tai tiedot ovat vanhentuneet, annetaan käyttäjälle siitä ilmoitus. Lämpötila- ja korkeusyksiköt muunnetaan samalla tarpeen vaatiessa Fahrenheiteiksi ja jaloiksi.

3.5.2 Säätietojen toteutus

Sivulle tultaessa tarkistetaan laitteen paikallisesta tallennustilasta paivamaara-muuttujan tila. Tämä muuttuja tallennetaan samalla, kun haeSaa()-funktio tallentaa JSON-muodossa olevat säätiedot paikalliseen tallennustilaan sovelluksen päävalikkoon tultaessa. Jos tätä muuttujaa ei internet-yhteyden puuttumisen tai jonkin muun tapahtuneen ongelman takia ole tallentunut, annetaan käyttäjälle ilmoitus, että säätietoja ei ole saatavilla. Muussa tapauksessa säätiedot kirjoitetaan sivulle. Käyttäjän kielivalikossa tekemän valinnan mukaan lämpötilat ja pituudet muunnetaan tietoja kirjoitettaessa oikeaan yksikköön (taulukko 26).

Taulukko 26. Lämpötilojen muunnos valittuun yksikköön (Ylläs 2013)

```
function muunnaLampotila(value) {
  /* jos kielivalikosta valittuna fahrenheit, lampotilan muunnos
  fahrenheitiksi */
  if(window.localStorage.getItem('ltilaYksikko')=='fahrenheit') {
    value = value.replace(",",".");
    var F = Math.round((parseFloat(value) * 9/5+32)*10)/10 +
    '&#8457;';
  }
  // muuten palautetaan celsiuksina
  else var F = value +'&#8451;';
  return F;
}
```

3.6 Bussiaikataulu

3.6.1 Bussiaikataulun toiminta

Kun sovelluksen päävalikosta avataan Bussiaikataulut-linkki, näkyviin tulevat linkit Äkäslompolon ja Ylläsjärven paikallisiin SkiBus-aikatauluihin.

3.6.2 Bussiaikataulun toteutus

Muusta navigaatiosta poiketen, linkkejä paikallisiin aikatauluihin ei ladata AJAX-tekniikalla, vaan ne ladataan uusina sivuina (taulukko 27). Aikataulut esitetään sivuilla png-formaatissa olevina kuvina. iPhone-versiossa aikataulusivujen zoomaus on toteutettu käyttämällä muokattua iScroll 4 -skriptiä. Android- ja Windows Phone-versioissa aikataulusivulta paluu tapahtuu laitteen takaisin-painikkeen kautta, mutta iPhone-versiossa sivun alalaitaan on luotu erillinen takaisin-painike.

Taulukko 27. Sivun lataus external-attribuuttia käyttäen (Linking pages 2013, hakupäivä 6.11.2013)

```
<a href="akaslompolo.html" rel="external">
```

3.6.3 Bussiaikataulun toteutuksen ongelmatilanteet

Navigointi aikatauluihin toteutettiin aluksi AJAX-tekniikalla, mutta tämän jälkeen kuvan zoomaus ja vierittäminen eivät toimineet. Päätimme toteuttaa sivunvaihdon käyttäen jQuery Mobilen rel="external"-määritystä, jolloin koko sivu latautuu ja näin ollen <head>-tunnisteen sisällä olevat viewport-määritykset mahdollistavat kuvan vierityksen ja zoomauksen.

iPhone-laitteessa ei ole fyysistä takaisin-painiketta, joten tähän versioon lisäsimme sivun alareunaan painikkeen tätä toimintoa varten. Sivua zoomattaessa painikkeen koko kasvoi ja sivua vierittäessä painike katosi näkyvistä. Tämän ongelman ratkaisuna käytimme iScroll 4 -skriptiä, joka mahdollistaa sivun eri osien, esimerkiksi -tunnisteen (taulukko 28), zoomauksen määritysten mukaan (Spinelli 2012, hakupäivä 31.10.2013). Aikataulukuvan sisältävä -tunniste on tehty zoomattavaksi ja vieritettäväk-

si, jolloin erilliseen -tunnisteeseen tehty takaisin-painike pysyy määritetyssä paikassa ja sen koko ei muutu.

Taulukko 28. iScroll 4 -määritys -tunnisteelle (Spinelli 2012, hakupäivä 31.10.2013)

```
<script type="text/javascript">
  var myScroll;
    function loaded() {
    myScroll = new iScroll('scroller');
  }
  document.addEventListener('DOMContentLoaded', loaded, false);
</script>
<div id="scroller">
  <!--zoomattavan ja vieritettävän div:n sisältö-->
</div>
```

3.7 Ohje

3.7.1 Ohjeen toiminta

Ohje-sivulla luodaan käyttöohje sovelluksen toimintoihin. Ohjeen sisältö näytetään käytössä olevan kielen mukaan.

3.7.2 Ohjeen toteutus

Sivulle tultaessa suoritetaan kirjoitaOtsikko()- ja kirjoitaDivit()-funktiot. kirjoitaOtsikko()-funktiossa luodaan sivulle pääotsikko. kirjoitaDivit()-funktiossa luodaan kolme laajennettavaa valikkoa joiden sisältö haetaan kieli.js-tiedostosta käytössä olevan kielen mukaan. Valikoiden ja sisällön luonti tapahtuu jQuery Mobilen trigger(”create”)-funktiota käyttäen (taulukko 29).

Taulukko 29. Sivun sisällön määritys ja kirjoitus (Ylläs 2013)

```
// sisällön määritys
data=data+'<div data-role="collapsible" data-theme="a" id="ohje" data-
inset="false"><h3 id="karttaOhje">'+kieli.rinnek+'</h3>';
data=data+'<div data-theme="c" >'+kieli.karttaohje+'</div>';
data=data+'</div>';
// trigger('create'), datan kirjoitus
$('#ohje).replaceWith(data);
$('#collapsibleSet').find('div[data-role=collapsible]').trigger('cre-
```

```
ate');
```

4 JOHTOPÄÄTÖKSET JA POHDINTA

PhoneGapilla pystyy kohtalaisen helposti tuottamaan sovellusta kaikille käyttämillemme alustoille, kun ensin saa selvitettyä laitekohtaiset erot niiden JavaScript-ohjelmakoodin käsittelyssä. Android-alustalla HTML-tiedostoon sijoitetut funktiot eivät toimineet Windows Phone-alustalla, vaan ne täytyi sijoittaa erilliseen JavaScript-tiedostoon. Lisäksi Androidilla ohjelma toimi ilman koodin lataamista deviceready-tapahtumankuuntelijan kautta, kun taas Windows Phone tämän vaati toimiakseen. Android- ja iPhone-alustojen molempien käyttämät Webkit-pohjaiset selaimet toimivat paremmin yhteen kuin Windows Phonen käyttämä Internet Explorer-pohjainen selain. Myös JavaScriptin ajaminen PhoneGapin kautta käyttää laitteen resursseja enemmän kuin suoraan laitteiden natiivikielille kirjoitetut ohjelmat. Käyttäen esimerkkinä sovelluksen yksinkertaisempia ominaisuuksia, kuten päävalikkoa tai sää-sivua, voidaan todeta PhoneGap-sovelluskehyksen soveltuvan tällaisten luontiin hyvin. Monimutkaisempien toimintojen, esimerkiksi kartta-sivun, toteutus kuitenkin vaatii paljon enemmän koodin muokkausta laiteympäristöjen välillä.

Saimme omasta mielestämme hyvin tuotua esille PhoneGap-sovelluskehyksen etuja ja erityisesti sen aiheuttamia ongelmia. Löysimme myös kohtaamiimme ongelmatilanteisiin omassa sovelluksessa toimivat ratkaisut.

Aineiston keruu onnistui kohtalaisen helposti internetin hakukoneita käyttäen. Aiheesta on kirjoitettu internetissä paljon, ja monilla on ollut samoja kysymyksiä kuin meille ilmeni projektin aikana. Kirjalähteiden käytössä ongelmana tällä alalla on tiedon nopea vanheneminen, mikä pitää paikkansa PhoneGapin kohdalla varsin hyvin, sillä siitä julkaistaan lyhyin väliajoin uusia versioita. Aineiston käsittelyssä huomioitavaa on tiedon muokkaaminen oman sovelluksen tarpeita vastaamaan, sillä harvoin löytämämme esimerkit sopivat sellaisenaan. Raportin teossa hankalinta oli alan runsas englanninkielinen termistö ja sen käyttäminen suomenkielisessä raportissa. Termeillä on harvoin vakiintuneita suomenkielisiä vastineita, ja usein täytyi miettiä miten raportista saataisiin kaikkein ymmärrettävin.

Raportointi onnistui ongelmitta, koska sovellus on itse tekemämme ja sen johdosta sitä on helppo käsitellä. Alusta asti meille oli selkeää miten raportti rakentuu ja mitä asioita käymme läpi.

Sovellusta tehdessä opimme käyttämään PhoneGap-sovelluskehystä, sekä tietotaitomme käyttämistämme web-tekniikoista kehittyi huomattavasti. Eri alustojen ohjelmointiympäristöt tulivat myös tutuksi. Saimme kokemusta mobiilisovelluksien kehittämisestä ja jos tulevaisuudessa tulemme työskentelemään niiden parissa, pystymme arvioimaan PhoneGapin käytön kannattavuutta verrattuna laitteen natiivikieleen sovellusten vaatimusten perusteella. Lisäksi saimme kokemusta pidempikestoisesta projektista ja työskentelystä tiimissä.

Mielestämme tätä raporttia voidaan hyödyntää opetuskäytössä, jos kurssin aiheena on mobiiliohjelmointi ja PhoneGap-sovelluskehys. Lisäksi raportti voi olla apuna ensimmäistä PhoneGap-sovellusta tehdessä.

Jatkotutkimuksen aiheena voisi olla saman sovelluksen kehittäminen kyseisille alustoille niiden natiivikielillä tai käyttäen jotain muuta mobiilisovelluskehystä.

LÄHTEET

Barbagallo, John Paul 2013. jQuery Events for Start/Stop Scrolling. Coderwall. Hakupäivä 4.11.2013. <https://coderwall.com/p/b1wxgg>

Codenames, Tags, and Build Numbers 2013. Android Developers. Android. Hakupäivä 4.11.2013. <http://source.android.com/source/build-numbers.html>

Collapsible content on-click events 2010. jQuery Forum. jQuery. Hakupäivä 21.10.2013. <http://forum.jquery.com/topic/collapsible-content-on-click-events>

Configuring the Viewport 2013. Safari Web Content Guide. iOS Developer Library. Apple Inc. Hakupäivä 24.10.2013. <https://developer.apple.com/library/ios/documentation/AppleApplications/Reference/SafariWebContent/UsingtheViewport/UsingtheViewport.html>

Device 2013. PhoneGap Documentation. PhoneGap. Hakupäivä 4.11.2013. <http://docs.phonegap.com/en/2.2.0/cordova_device_device.md.html>

Don't forget to add doctype 2013. Quality Web Tips. W3C. Hakupäivä 21.10.2013. <http://www.w3.org/QA/Tips/Doctype/>

Events 2013. PhoneGap Documentation. PhoneGap. Hakupäivä 7.10.2013. <http://docs.phonegap.com/en/2.2.0/cordova_events_events.md.html#Events>

Fatih, Akin 2012. f/jquery.resizestop. GitHub. Hakupäivä 4.11.2013 <https://github.com/f/jquery.resizestop>

Fixed toolbars 2013. jQuery Mobile Framework. jQuery Mobile. Hakupäivä 4.11.2013. <http://jquerymobile.com/demos/1.2.0/docs/toolbars/bars-fixed.html/>

Geolocation 2013. PhoneGap Documentation. PhoneGap. Hakupäivä 28.10.2013. <http://docs.phonegap.com/en/2.2.0/cordova_geolocation_geolocation.md.html#Geolocation>

Getting Started With Android 2013. PhoneGap Documentation. PhoneGap. Hakupäivä 24.10.2013. <http://docs.phonegap.com/en/1.8.1/guide_getting-started_android_index.md.html#Getting%20Started%20with%20Android>

Getting Started with iOS 2013. PhoneGap Documentation. PhoneGap. Hakupäivä 29.10.2013. <http://docs.phonegap.com/en/2.2.0/guide_getting-started_ios_index.md.html>

Getting Started with jQuery Mobile 2013. jQuery Learning Center. jQuery Mobile. Hakupäivä 7.10.2013. <http://learn.jquery.com/jquery-mobile/getting-started>

HTML5 Input Types. W3Schools. Hakupäivä 3.11.2013. <http://www.w3schools.com/html/html5_form_input_types.asp>

Installing the Eclipse Plugin 2013. Android Developers. Android. Hakupäivä 21.10.2013. <http://developer.android.com/sdk/installing/installing-adt.html>

jQuery.mobile.activePage 2013. jQuery Mobile API Documentation. jQuery Mobile. Hakupäivä 21.10.2013. <http://api.jquerymobile.com/jQuery.mobile.activePage>

Linking pages 2013. jQuery Mobile Docs. jQuery Mobile. Hakupäivä 6.11.2013. <http://jquerymobile.com/demos/1.2.0/docs/pages/page-links.html>

Microsoft Visual Studio Express 2010 for Windows Phone 2013. Ohjelmointiympäristö Windows Phone -alustalle.

SDK Manager 2013. Android Developers. Android. Hakupäivä 21.10.2013. <http://developer.android.com/tools/help/sdk-manager.html>

Spinelli, Matteo 2012. iScroll 4. Cubiq.org. Hakupäivä 31.10.2013. <http://cubiq.org/iscroll-4>

TargetApi 2013. Android Developers. Android. Hakupäivä 4.11.2013. <http://developer.android.com/reference/android/annotation/TargetApi.html>

TG4NP Tourist Guide for Northern Periphery. Northernperiphery.eu 2010. Hakupäivä 3.5.2013. <http://www.northernperiphery.eu/en/projects/show/&tid=82>

Trice, Andrew 2012. PhoneGap Explained Visually. PhoneGap Blog. PhoneGap. Hakupäivä 24.10.2013. <http://phonegap.com/2012/05/02/phonegap-explained-visually/>

WebSettings 2013. Android Developers. Android. Hakupäivä 4.11.2013. <http://developer.android.com/reference/android/webkit/WebSettings.html>

Web Storage 2013.W3C . Hakupäivä 24.10.2013. <http://www.w3.org/TR/webstorage/>

Windows Phone 7 Platform Guide 2013. PhoneGap Documentation. PhoneGap. Hakupäivä 28.10.2013. <http://docs.phonegap.com/en/edge/guide_platforms_wp7_index.md.html#Windows%20Phone%207%20Platform%20Guide>

Yates, Ian 2012. Quick Tip: Make Telephone Numbers Do Something. Tuts+. Hakupäivä 6.11.2013. <http://webdesign.tutsplus.com/tutorials/htmlcss-tutorials/quick-tip-make-telephone-numbers-do-something/>

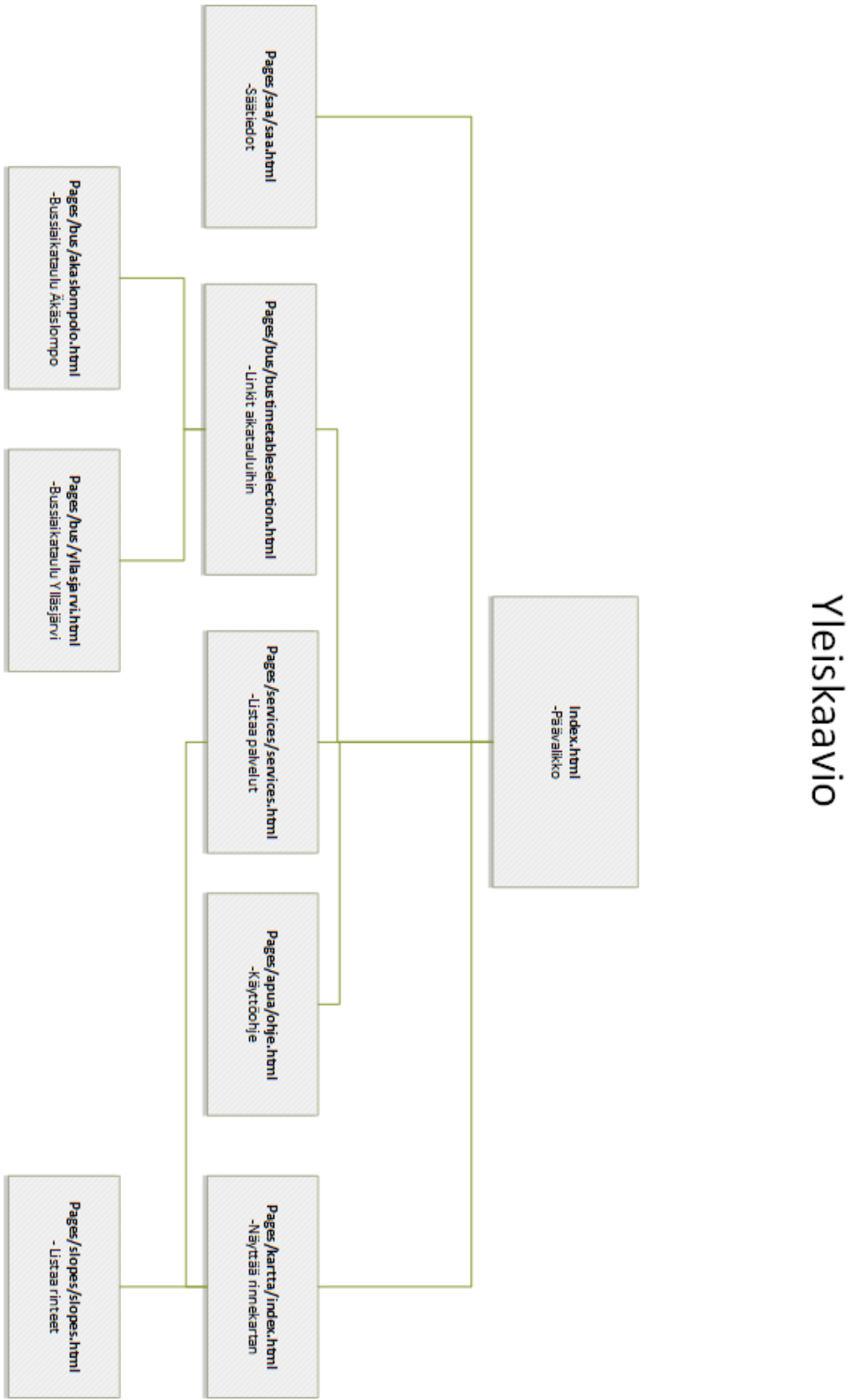
Ylläs 2013. Mobiilisovellus Ylläkselle.

LIITTEET

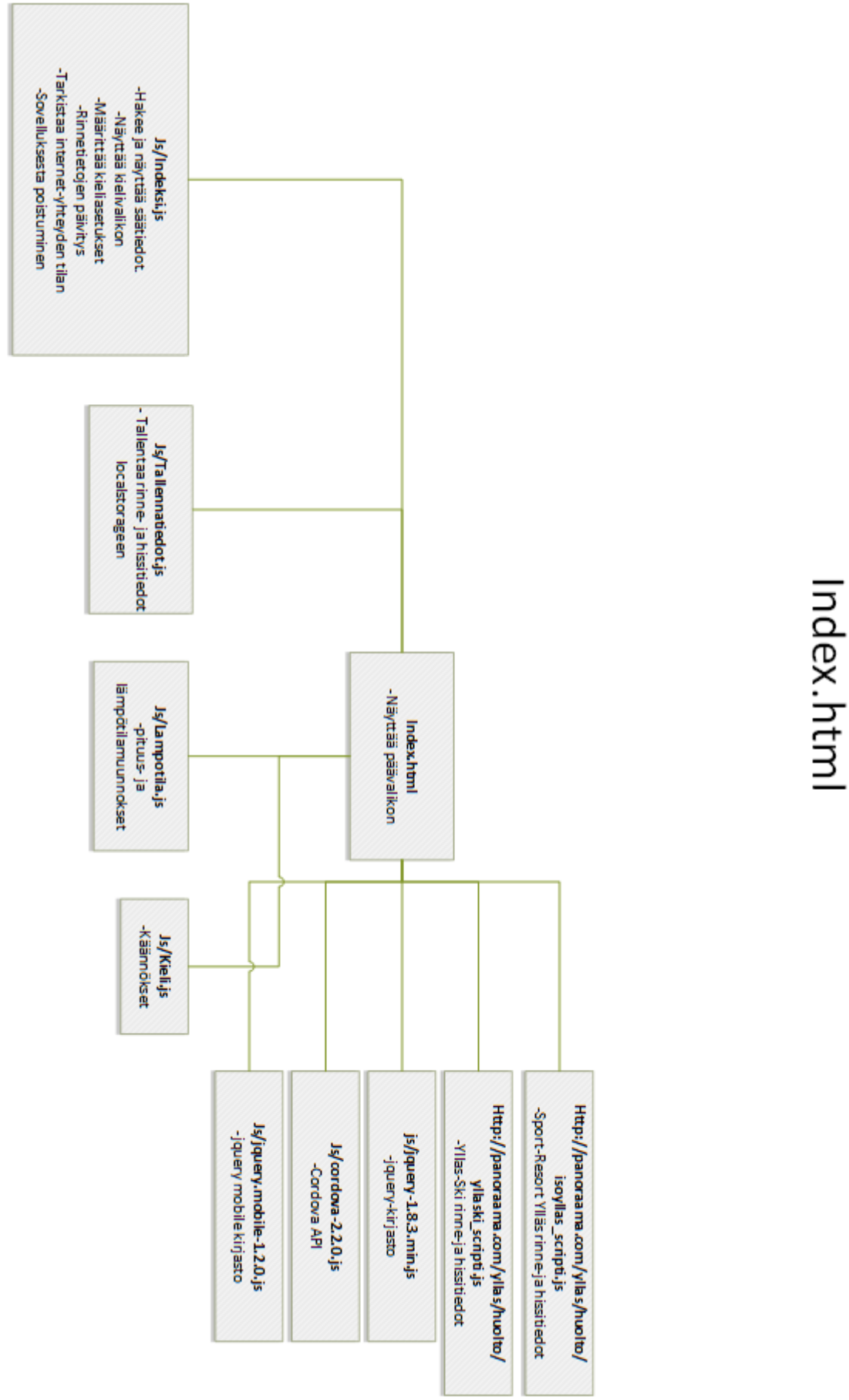
Liite 1. Sovelluksen rakenteen ja toiminnan kaavio

Liite 2. Sovelluksen ominaisuusmäärittely

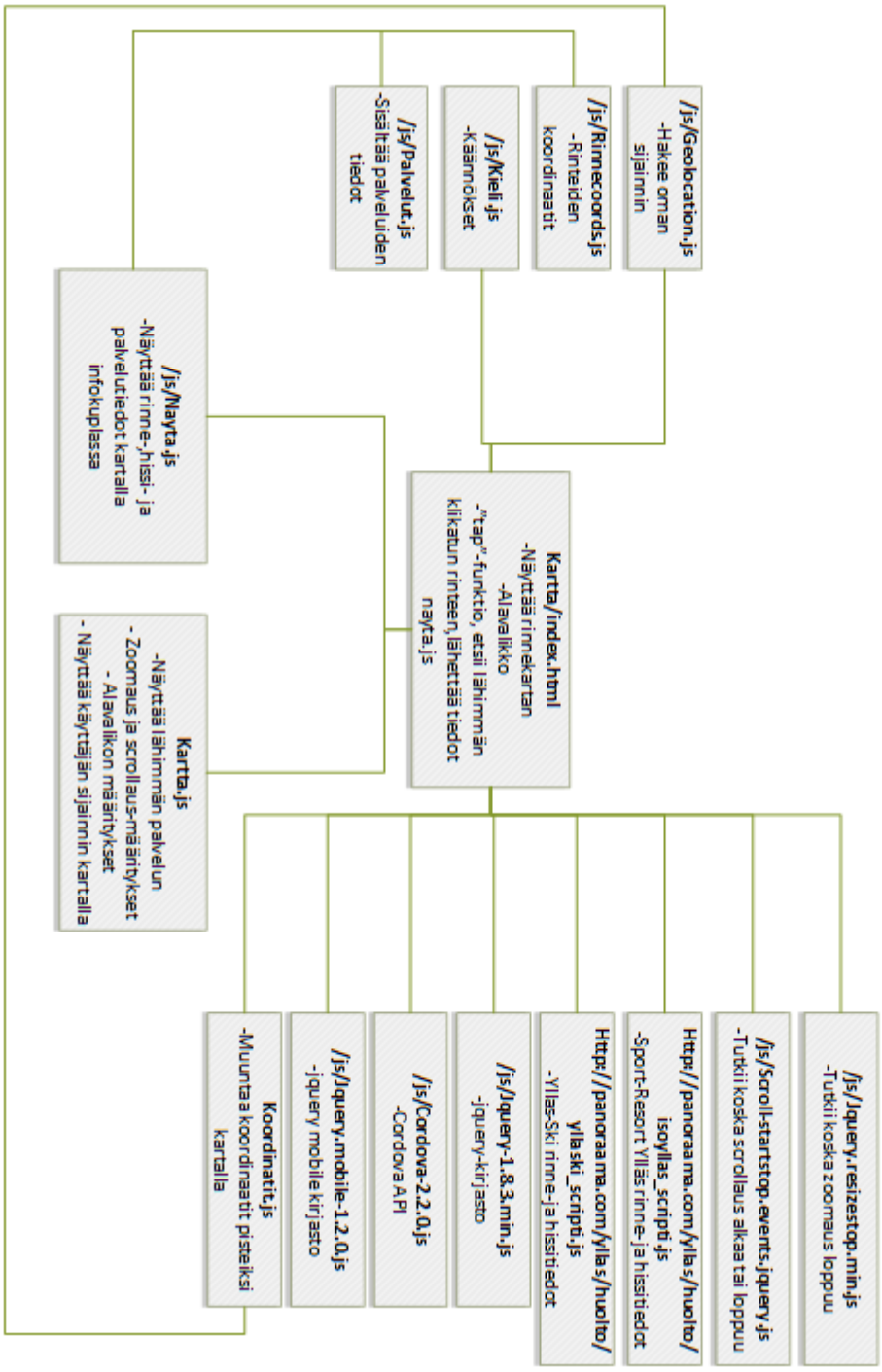
Liite 1 1(8)



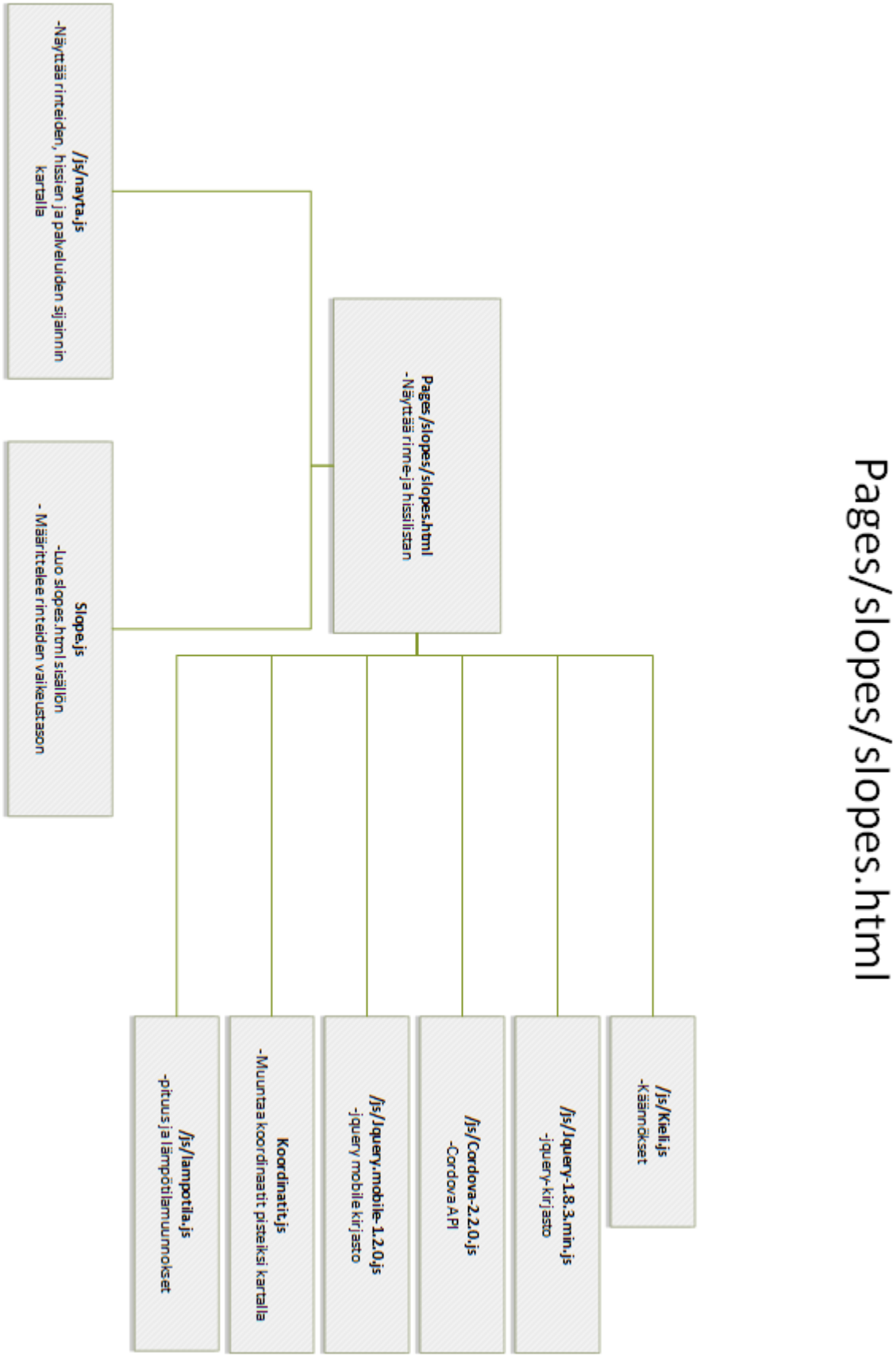
Liite 1 2(8)


Index.html
Index.html
- Näyttää päävalikon
Js/Indeksi.js
-Hakee ja näyttää säätiedot
-Näyttää kielivalikon
-Määrittää kieliasetukset
-Rinnetietojen päivitys
-Tarkistaa internet-yhteyden tilan
-Sovelluksesta poistuminen
Js/Tallennatiedot.js
- Tallentaa rinne- ja hissitiedot
localstorageen
Js/Lampotila.js
-pituus- ja
lämpötilamuunnokset
Js/Kieli.js
-Käännökset
Http://panoraama.com/yllas/huolto/
isoyllas_scripti.js
-Sport-Resort Yllas rinne-ja hissitiedot
Http://panoraama.com/yllas/huolto/
yllaski_scripti.js
-Yllas-Ski rinne-ja hissitiedot
js/jquery-1.8.3.min.js
-jquery-kirjasto
Js/cordova-2.2.0.js
-Cordova API
Js/jquery.mobile-1.2.0.js
- jquery mobile kirjasto

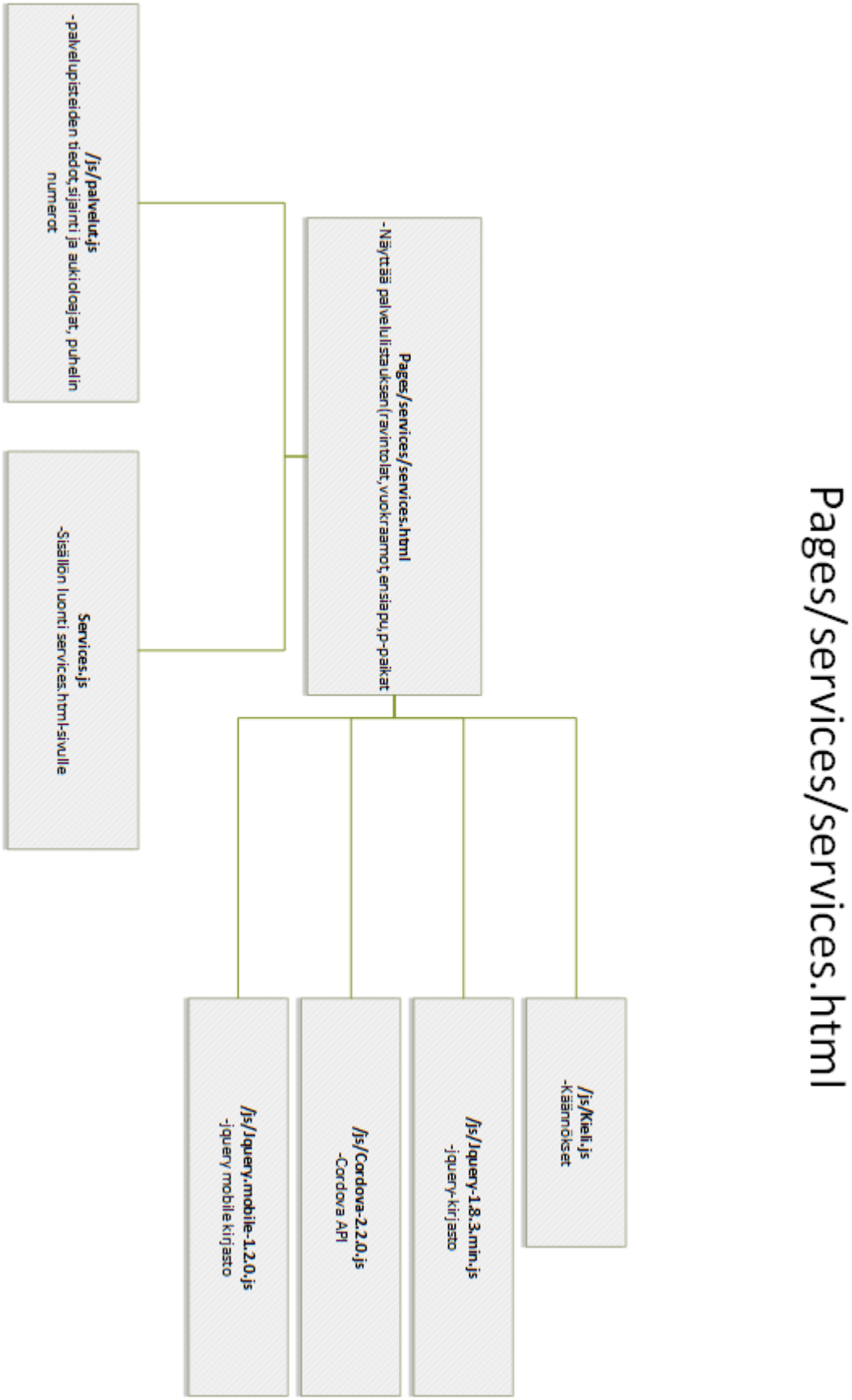
Liite 1 3(8)


Pages/kartta/index.html
/js/Geolocation.js
-Hakee oman sijainnin
/js/Rinnecoords.js
- Rinteiden koordinaatit
/js/Kieli.js
-Käännökset
/js/Palvelut.js
-Sisältää palveluiden tiedot
/js/Nayta.js
-Näyttää rinne-,hissi- ja palvelutiedot kartalla infokuplassa
Kartta/index.html
-Näyttää rinnekartan
-Alavalikko
-"tap"-funktio, etsii lähimmän klikatun rinteen,lähettää tiedot nayta.js
Kartta.js
-Näyttää lähimmän palvelun
- Zoomaus ja scrollaus-määritykset
- Alavalikon määritykset
- Näyttää käyttäjän sijainnin kartalla
/js/Jquery.resizestop.min.js
-Tutkii koska zoomaus loppuu
/js/Scroll-startstop.events.jquery.js
-Tutkii koska scrollaus alkaa tai loppuu
Http://panoraama.com/yllas/huolto/isoyllas_scripti.js
-Sport-Resort Ylläs rinne-ja hissitiedot
Http://panoraama.com/yllas/huolto/yllaski_scripti.js
- Yllas-Ski rinne-ja hissitiedot
/js/Jquery-1.8.3.min.js
-jquery-kirjasto
/js/Cordova-2.2.0.js
-Cordova API
/js/Jquery.mobile-1.2.0.js
- jquery mobile kirjasto
Koordinatit.js
-Muuntaa koordinaatit pisteiksi kartalla

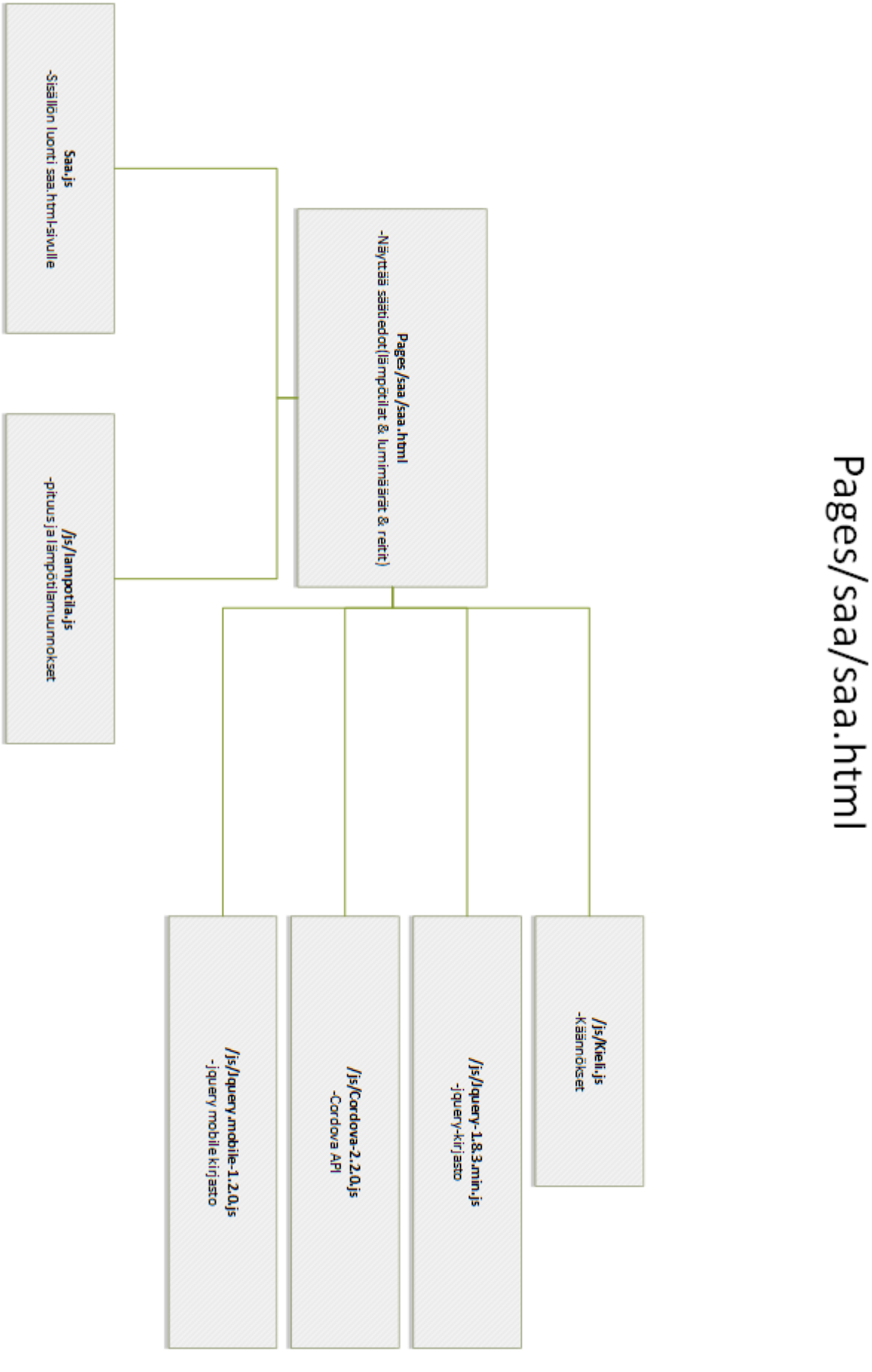
Liite 1 4(8)


Pages/slopes/slopes.html
Pages /slopes/slopes.html
-Näyttää rinne-ja hissilistan
/js/näytä.js
-Näyttää rinteiden, hissien ja palveluiden sijainnin kartalla
Slope.js
-Luo slopes.html sisällön
- Määrittelee rinteiden vaikeustason
/js/Kieli.js
-käännökset
/js/Jquery-1.8.3.min.js
-jquery-kirjasto
/js/Cordova-2.2.0.js
-Cordova API
/js/Jquery.mobile-1.2.0.js
-jquery mobile kirjasto
Koordinatit.js
-Muuntaa koordinaatit pisteiksi kartalla
/js/lampotila.js
-pituus ja lämpötilamuunnokset

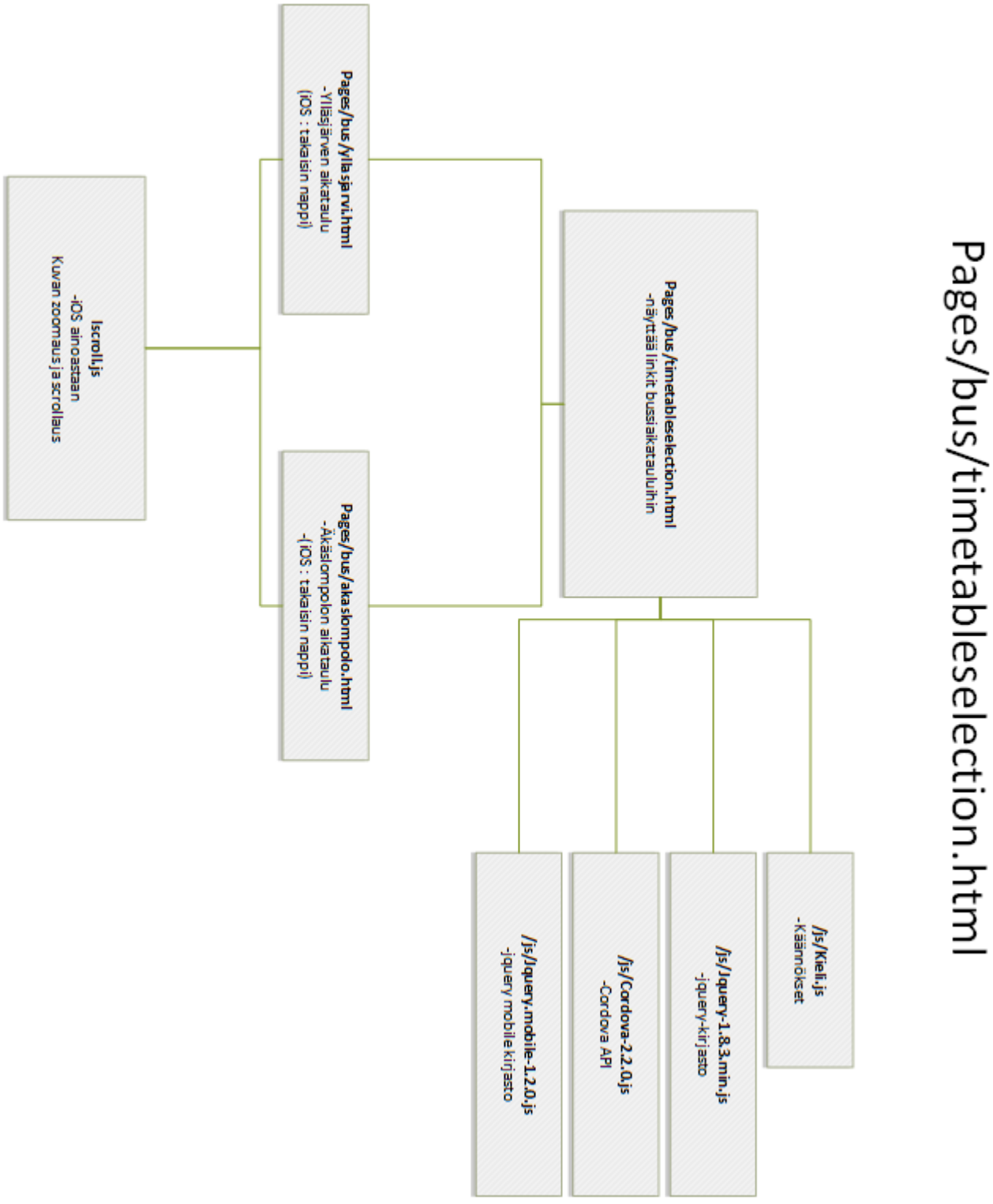
Liite 1 5(8)


Pages/services/services.html
/js/palvelut.js
-palvelupisteiden tiedot,sijainti ja aukioloajat, puhelin numerot
Pages/services/services.html
-Näyttää palvelulistauksen(ravintolat,vuokraamot,ensiapu,p-paikat
Services.js
-Sisällön luonti services.html-sivulle
/js/Kieli.js
-Käännökset
/js/Jquery-1.8.3.min.js
-jquery-kirjasto
/js/Cordova-2.2.0.js
-Cordova API
/js/Jquery.mobile-1.2.0.js
-jquery mobile kirjasto

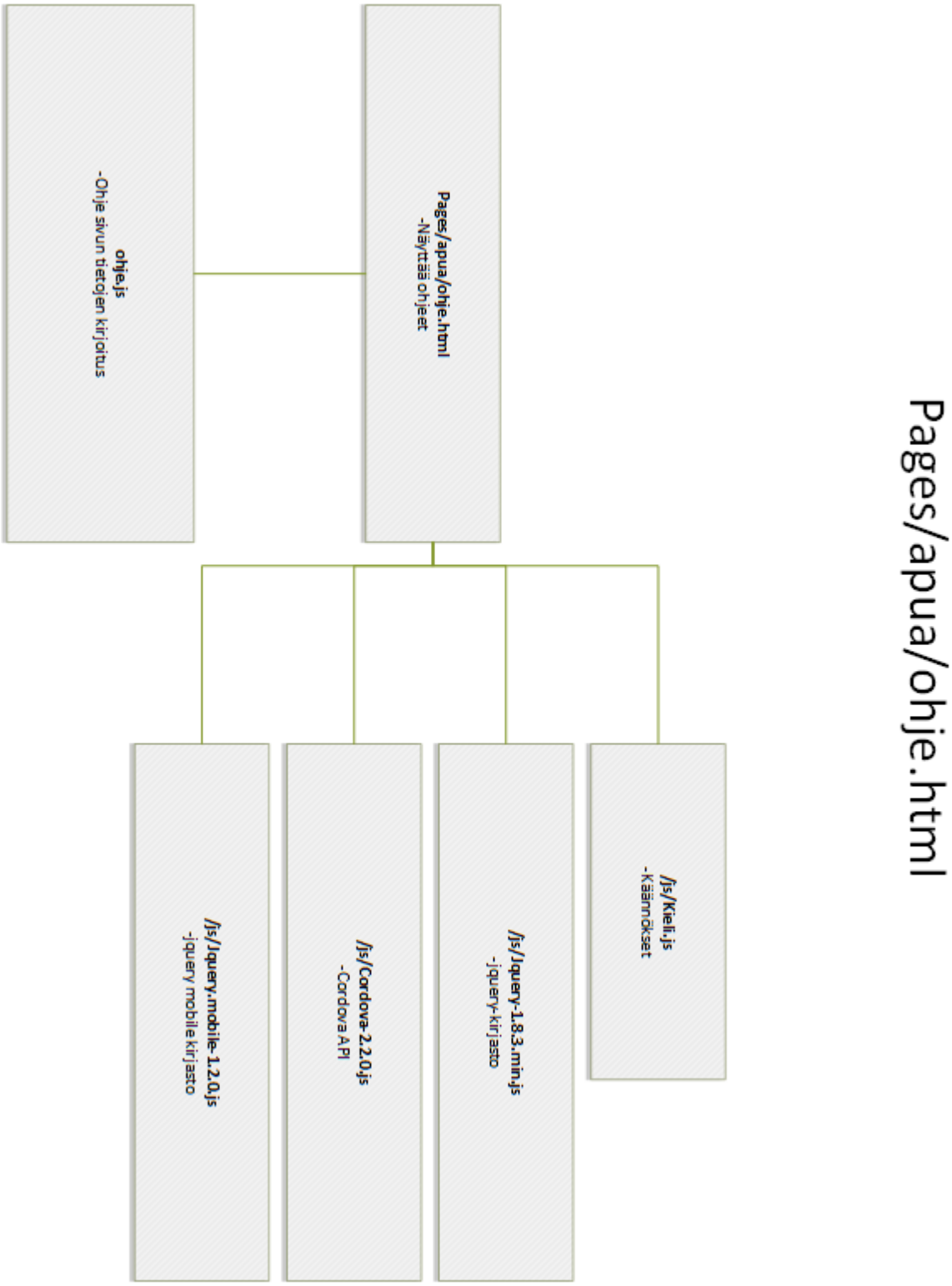
Liite 1 6(8)


Pages/saa/saa.html
Saa.js
-Sisällön luonti saa.html-sivulle
Pages/saa/saa.html
-Näyttää säätiedot(lämpötilat & luminäärät & reitit)
/js/lampotila.js
-pituus ja lämpötilamuunnokset
/js/Kieli.js
-Käännökset
/js/jquery-1.8.3.min.js
-jquery-kirjasto
/js/Cordova-2.2.0.js
-Cordova API
/js/jquery.mobile-1.2.0.js
-jquery mobile kirjasto

Liite 1 7(8)


Pages/bus/timetableselection.html
Pages/bus/timetableselection.html
-näyttää linkit bussiaikatauluihin
Pages/bus/yllasjarvi.html
-Yllässjärven aikataulu
(iOS : takaisin nappi)
Pages/bus/akaslompolo.html
-Äkäslompolon aikataulu
-(iOS : takaisin nappi)
Iscroll.js
-iOS ainoastaan
Kuvan zoomaus ja scrollaus
/js/Kieli.js
-Käännökset
/js/Jquery-1.8.3.min.js
-jquery-kirjasto
/js/Cordova-2.2.0.js
-Cordova API
/js/Jquery.mobile-1.2.0.js
-jquery mobile kirjasto

Liite 1 8(8)


Pages/apua/ohje.html
Pages/apua/ohje.html
-Näyttää ohjeet
ohje.js
-Ohje sivun tietojen kirjoitus
/js/Kieli.js
-Käännökset
/js/jquery-1.8.3.min.js
-jquery-kirjasto
/js/Cordova-2.2.0.js
-Cordova API
/js/jquery.mobile-1.2.0.js
-jquery mobile kirjasto

Liite 2 1(4)

YLLÄS MOBIILISOVELLUS

Ominaisuusmäärittely

versio 1

Yrjö Koskenniemi
Markku Henriksson
Timo Jurvelin
Toni Lamsijärvi

21.02.2012

Liite 2 2(4)

Tavoite *Projektissa suunnitellaan ja toteutetaan pilottiversio Ylläksen alueen matkailijoiden käyttöön tulevasta mobiilisovelluksesta. Ensimmäisessä vaiheessa keskitytään laskettelukeskusten (Sport Resort Ylläs ja Ylläs-Ski) asiakkaiden tarpeisiin. Myöhemmin sovellus laajennetaan koko alueen matkailijoita palvelevaksi*.

Yleistä sovelluksesta

Sovelluksesta tehdään ensin yksi pilottiversio (Android tai Windows Phone tai iPhone), jotta voidaan testata sovelluksen toimivuutta. Varsinaisesta sovelluksesta julkaistaan kaikkiin eri käyttöjärjestelmiin omat versiot.

Pilottisovelluksen ominaisuudet

Rinnekartta

- Reaaliaikainen tieto rinteistä ja hisseistä
- Aukiolojat yms
- Nykyinen päivityssysteemi -> päivitysaika?

Säätiedot

- Nykyiset säätiedot kuten yllas.fi –sivuilla
- Lumitilanne, sademäärät yms
- Varoitukset (lumivyöry, tuuli, lumipyry tms)
- Revontuliennusteet

Tapahtumat

- Tapahtumat hiihtokeskuksessa
- Tapahtumat alueella yleensä

Palvelut keskuksissa

- Ravintolat (menu, aukioloajat)
- Lipunmyynti
- WC:t
- Lastenhoito
- Kaupat
- Majoitus
- Hiihtokoulun palvelupisteet

Ensiapu/Terveydenhoito

- Ensiapupisteiden sijainti
- Yhteystiedot
- Hätänumero

Liite 2 3(4)

GPS-paikannus

- Käyttäjän sijainti kartalla

Bussiaikataulut

- Skibussit alueella (nykyisin pdf-dokumentti)

Ylläs Facebook tilapäivitykset näkyville

- Näytetään Ylläksen Facebook tilapäivitykset

Kieli

- Pilottisovelluksessa kielenä suomi
- Huomioidaan sovellusta tehtäessä, että eri kieliversioita on tulossa

Yleiset yhteystiedot

- Hiihtokeskusten palvelunrot yms, Ylläksen Matkailu, Taxi jne

Muuta huomioitavaa

- Käyttäjälle näytetään aika, jolloin tiedot viimeksi sovellukseen haettu
- Käyttäjälle mahdollisuus päivittää tiedot uudestaan sovellukseen

Pilottisovelluksen käyttöliittymä

Toni Lamsijärvi ja Timo Jurvelin tekevä viikolla 8 demon sovelluksen käynnistyskuvakkeesta ja aloitusnäkymästä.

Sovelluksen aloituskuvake

- Esim Ylläs on ykkönen –logo => sovellus käynnistyy => päävalikkoon/päänäkymään

Sovelluksen päänäkymä

- Sovelluksen kaikki päätoiminnot näkyvät kuvakkeina
- Värimaailma sama kuin yllas.fi
- Taustakuvana jokin Ylläs-aiheinen kuva (esim yllas.fi yläreunassa)
- Kielivalinta pääsivulla esim alareunassa (tai jatkossa Asetukset/Settings-kuvake)

Liite 2 4(4)

Käyttäjärooli	Sovelluksen ominaisuus	Testaaminen (How to demo)	Lisätieto	Prioriteetti
Peruskäyttäjä	Aloitus	Sovelluksen kuvakkeen painaminen	Sovellus ladattu puhelimeen ja valmiina käytettäväksi	
	Sovelluksen osan valinta päänäkymästä	Toimintokuvakkeen painaminen päänäkymässä	Kaikki kuvakkeet näkyvät päänäkymässä	
	Kielivalinta	Päänäkymässä painetaan kielivalinta kuvaketta	Kielivalinnat näkyvillä (Ei pilotissa!)	N/A N/A
	Rinnekartta	Valitaan päänäkymästä rinnekartta	Avautuu karttanäkymä, jossa näkyy käyttäjän sijainti ja kartta	
	Säätiedot - Yleissäätieto näkyvillä päänäkymässä?	Tarkat säätiedot saa näkyville valitsemalla toiminnon Sää alueella	Näytetään samoja säätietoja kuin yllas.fi -webbisivuilla	
	Tapahtumat	Valitsee Tapahtumat	Näytetään hiihtokeskuksen lähiajan tapahtumat. Valittavissa myös kaikki tapahtumat alueella (jos saadaan tiedot)	
	Palvelut keskuksissa	Valitsee palvelut päänäkymässä tai karttanäkymässä valitsee haluamansa palvelun näkyville	Palvelut näytetään kartalla sijaintitietoineen ja lisätietoa saa klikkaamalla	
	Ensiapu	Valitsee päänäkymässä ensiapu tai karttanäkymässä valitaan ensiapukuvake	Näytetään kartalla ensiapupisteet ja lisätiedot saa klikkaamalla	
	GPS-paikannus	Karttanäkymissä näytetään sijainti	Kaikissa karttanäkymissä näytetään käyttäjän sijainti mikäli GPS on päällä	
	Bussiaikataulut	Valitsee päänäkymässä bussiaikataulut	Näytetään bussiaikataulut. Nykyinen versio pdf-dokumentti.	
	Yhteystiedot	Valitsee päänäkymässä yhteystiedot	Näytetään kaikki tärkeät puhelinnrot yms alueen osalta	
	Sovelluksen tietojen päivitys	Päänäkymässä valitaan päivitä tiedot	Haetaan uusimmat tiedot palvelimilta ja päivitetään sovellukseen.	