



SAVONIA

■ OPINNÄYTETYÖ - AMMATTIKORKEAKOULUTUTKINTO
TEKNIIKAN JA LIIKENTEEN ALA

KUOPIONKIRPPARI.FI

Android-sovellus

TEKIJÄ: Riku Jokinen

Koulutusala Tekniikan ja liikenteen ala	
Koulutusohjelma Tietotekniikan koulutusohjelma	
Työn tekijä(t) Riku Jokinen	
Työn nimi Kuopionkirppari.fi Android-sovellus	
Päiväys 21.11.2013	Sivumäärä/Liitteet 31 + 12
Ohjaaja(t) lehtori Jussi Koistinen	
Toimeksiantaja/Yhteistyökumppani(t) yrittäjä Mikko Hopia, Tmi Mikko Hopia	
<p>Tiivistelmä</p> <p>Opinnäytetyön aiheena oli toteuttaa Android-sovellus kuopionkirppari.fi-sivuston käyttämiseksi. Sovelluksen tilasi kuopiolainen Tmi Mikko Hopia.</p> <p>Sovelluksella tuli pystyä selaamaan ja hakemaan käyttäjien ilmoituksia kuopionkirppari.fi-palvelussa sekä lisäämään uusia ilmoituksia kuvineen. Sovelluksesta oli aiemmin tehty versio Applen iOS-käyttöjärjestelmälle ja nyt tarvittiin versio, joka toimii myös Android-laitteissa. Sovelluksen tarkoituksena oli helpottaa Android-laitteiden omistajien palvelun käyttöä.</p> <p>Sovellus kehitettiin käyttäen hyväksi Eclipse-kehitysympäristöä, johon oli asennettu Android Developer Tools -lisäosa. Sovelluksen ja palvelun välisessä tiedonvaihdoissa käytettiin hyväksi jo valmiiksi toteutettua JSON-rajapintaa.</p> <p>Opinnäytetyön tuloksena julkaistiin valmis versio sovelluksesta Google Play -sisältöpalveluun. Jatkokehityksestä ja sovelluksen päivittämisestä sovitaan erikseen projektin jälkeen.</p>	
Avainsanat Android, JSON, Eclipse, Java, kirpputori, kirppari	

Field of Study Technology, Communication and Transport			
Degree Programme Degree Programme in Information Technology			
Author(s) Riku Jokinen			
Title of Thesis Android Application for kuopionkirppari.fi			
Date	21 November 2013	Pages/Appendices	31 + 12
Supervisor(s) Mr Jussi Koistinen, Lecturer			
Client Organisation /Partners Mr Mikko Hopia, t/a Mikko Hopia			
<p>Abstract</p> <p>The purpose of this thesis was to create an Android application to ease the use of the kuopionkirppari.fi service. The application was ordered by t/a Mikko Hopia.</p> <p>The application had to be able to browse, search and add new adverts. There was already an iOS version made of this application and now there was a need for an Android version. The purpose of this application was to give users an easier way to use the service.</p> <p>The application was developed with the Eclipse development environment. An additional Android Developer Tools plugin for Eclipse was installed to help with the Android development. The existing JSON interface was used for sending and receiving data from the server.</p> <p>As a result of this thesis a finished version of the application was released in the Google Play digital application distribution platform. Plans for further development will be made after this project.</p>			
Keywords Android, JSON, Eclipse, Java, rummage sale, flea market			

SISÄLTÖ

TERMIT JA LYHENTEET	6
1 JOHDANTO	7
2 VAATIMUSMÄÄRITTELY	8
2.1 Toiminnalliset vaatimukset	8
2.2 Ulkoasuun ja käyttökokemukseen liittyvät vaatimukset	9
2.3 Vanhempien Android-versioiden tukeminen	9
3 ANDROID	10
3.1 Android-käyttöjärjestelmä	10
3.1.1 API-versiot	10
3.1.2 Aktiviteetit ja fragmentit	10
3.1.3 Käyttöliittymät	10
3.2 Android-pohjaiset laitteet	11
3.3 Google Play	11
4 KÄYTETTY TYÖKALUT JA LAITTEET	12
4.1 Eclipse	12
4.2 Adobe Photoshop	12
4.3 Fyysiset ja virtuaaliset laitteet	12
5 TYÖN TOTEUTUS	14
5.1 Sovelluksen rakenne	14
5.1.1 Vaatimusten saavuttaminen vaadituilla API-versioilla	14
5.1.2 AndroidManifest.xml-tiedosto	14
5.1.3 Aktiviteettien välillä navigointi	15
5.1.4 Omat käyttöliittymäelementit	16
5.1.5 Näkymät ja siirtymät	17
5.1.5.1 Latausikkuna	17
5.1.5.2 Ilmoitusten selaus	17
5.1.5.3 Yksittäinen ilmoitus	19
5.1.5.4 Kuvien esikatselu	20
5.1.5.5 Ilmoituksen lisäys	21
5.1.5.6 Ilmoitusten haku	24
5.1.5.7 Ohjeet-näkymä	25

5.1.5.8 Ei yhteyttä -näkö	25
5.2 Sovelluksen käyttäjältä vaadittavat oikeudet	26
5.3 Valmiiden käyttöliittymäelementtien ja luokkien hyödyntäminen	27
5.4 JSON-rajapinta	28
5.5 Testaus	29
6 YHTEENVETO	30
LÄHTEET	31
LIITTEET	
Liite 1 Projektisuunnitelma	

TERMIT JA LYHENTEET

ADT	Android Developer Tools eli Android-kehittäjän työkalut on liitännäinen, joka lisää kehitysympäristöön toiminnallisuuksia, jotka helpottavat Android-sovelluksen kehittämistä.
API	Application programming interface eli ohjelmointirajapinta mahdollistaa eri ohjelmien pyyntöjen ja tietojen vaihdon.
AVD	Android Virtual Device eli virtuaalinen Android-laite mahdollistaa Android-sovellusten suorittamisen virtuaalisesti ilman fyysisiä päätelaitteita.
JSON	JavaScript Object Notation on laajalti käytetty yksinkertainen tiedonsiirtomuoto, joka nimestään huolimatta ei ole riippuvainen JavaScriptistä.
URI	Uniform Resource Identifier on merkkijono, jota käytetään esim. nimen tai verkkoresurssin tunnistamiseen.
XML	Existensible Markup Language on merkintäkieli, jolla tieto voidaan esittää ihmisen sekä koneen ymmärtämässä muodossa. Tiedolle voidaan merkitä sen tarkoitusperä.

1 JOHDANTO

Tämän työn tilaaja on kuopiolainen Tmi Mikko Hopia, joka ylläpitää kuopionkirppari.fi-sivustoa. Työn aiheena on tehdä sivuston käytön helpottamiseksi Android-sovellus, jolla ilmoituksia pystyy sekä selaamaan että lisäämään. Sama sovellus on toteutettu Applen iOS-käyttöjärjestelmälle ja toteutetaan myös Microsoftin Windows Phone -käyttöjärjestelmälle.

Kuopionkirppari.fi on vuonna 2009 perustettu verkossa toimiva yksityisille henkilöille tarkoitettu ilmoituskanava. Kuopionkirppari.fi:ssä vierailee päivittäin jopa 14 000 eri kävijää. Palvelun käyttö on ilmaista eikä se vaadi käyttäjätunnuksen rekisteröintiä.

Opinnäytetyön aihe syntyi Tmi Mikko Hopian tarpeesta saada kuopionkirppari.fi -sivustolle Android-sovellus, jotta mobiili käyttäjäkokemus saataisiin myös Android-laitteiden käyttäjille.

Työn tavoitteena on luoda julkaisuvalmis sovellus, joka mukailee jo aiemmin tehtyä iOS-version sovellusta niin ulkoasultaan kuin toiminnallisuudeltaan. Valmis sovellus julkaistaan Google Play -sisältöpalvelussa.

2 VAATIMUSMÄÄRITTELY

Sovelluksen tuli pohjautua käyttökokemukseltaan ja ulkoasultaan jo olemassa olevaan Applen iOS-käyttöjärjestelmälle tehtyyn versioon. Ulkoasun tuli mukailla valmista iOS-sovellusta kuitenkin niin, että Androidille ominaiset piirteet näkyivät sovelluksessa. Esimerkiksi Androidille tavanomaiset välilehdet, niillä navigointi sekä valikot ovat useille käyttäjille tuttuja muista Android-sovelluksista. On järkevää mukailla totuttuja käyttötapoja, jotta käyttäjäkokemus olisi helppo ja miellyttävä.

Yhteys kuopionkirppari.fi-sivuston tietokantaan tuli hoitaa käyttäen hyväksi valmista JSON-rajapintaa. Rajapinta mahdollistaa ilmoitusten selauksen, osioiden ja kategorioiden hakemisen sekä ilmoitusten lisäyksen kuvineen. Ellei toteutettavaan sovellukseen olisi ollut valmiina jo aiemmin eri alustalle tehtyä versiota, olisi tullut tutkia muita sovelluksen toteutustapoja.

Mikäli sovellukselle ei olisi ollut valmista versiota, olisi tietokantayhteyden voinut toteuttaa eri tavalla. Yksi mahdollisuus toteuttaa tietokantayhteys olisi ollut käyttää MySQL Connector -luokkaa. Tietokantakyselyitä olisi voinut suorittaa suoraan sovelluksesta. Tätä tapaa ei kuitenkaan suositella, koska ei ole viisasta käyttää Javan tietokanta connector-luokkaa suojaamattomissa verkoissa, kuten 3G-verkossa tai langattomassa lähiverkossa (MySQL Connector).

Toinen mahdollisuus tehdä hieman erilainen versio tietokantayhteydestä olisi ollut ottaa palvelimen vastaus XML-muodossa JSON-muodon sijaan. Käytännössä lopputulos olisi ollut aivan sama ja toteutus vaatinut yhtä paljon koodirivejä.

2.1 Toiminnalliset vaatimukset

Sovellukseen tuli toteuttaa helposti selattava navigointi eri toiminnallisuuksien välillä. Sovellusta tuli pystyä käyttämään vain näyttö pystysuunnassa.

Toteutettavat toiminnallisuudet olivat:

- ilmoitusten selaus
- ilmoitusten lisäys
- ilmoitusten haku
- yksittäisen ilmoituksen selaus
- yksittäisen kuvan esikatselu
- tietoja-sivu.

Ilmoitusten selaus tuli toteuttaa ns. loputtomana listana, eli käyttäjän selatessa listan loppuun tuli ilmoituksia ladata automaattisesti lisää. Ilmoituksia tuli myös pystyä selaamaan osioittain ja kategoriittain. Ilmoitusta painettaessa tuli avautua yksittäisen ilmoituksen tarkemmat tiedot sekä ilmoituksessa mahdollisesti olevat muut kuvat.

Ilmoituksia tuli pystyä lisäämään helposti omalta välilehdeltä. Ilmoituksessa piti olla tietyt pakolliset kentät eikä tietoja saanut lähettää eteenpäin, ennen kuin kentissä oli tietoa. Ilmoitukseen oli myös pystyttävä lisäämään kuvia suoraan laitteen kamerasta tai galleriasta.

Ilmoitusten haun piti toimia siten, että haettaessa hakusanalla vain oikeat hakutulokset näytetään. Listauksen tuli toimia samalla lailla kuin ilmoitusten selauksen, ns. loputtomana listana.

Yksittäisestä ilmoituksesta piti voida lukea ilmoituksen tiedot, selata kuvia, soittaa ja lähettää tekstiviesti ilmoittajalle sekä voida lähettää ilmoittajalle sähköpostiviesti, jos sähköpostiosoite oli annettu. Yksittäisessä ilmoituksessa kuvia pystyi olemaan maksimissaan viisi.

2.2 Ulkoasuun ja käyttökokemukseen liittyvät vaatimukset

Kuopionkirppari.fi on yhdenmukainen ulkoasultaan niin verkossa kuin mobiilissakin. Ulkoasun vaatimukset olivat tarkat ja selvät. Valmiin iOS-sovelluksen ulkoasu ja kuopionkirppari.fi:n logo oli annettuna projektin alussa ja niitä tuli käyttää Android-versiossa.

Sovelluksen ja sen käytön oli oltava käyttäjälle mahdollisimman suoraviivaista ja vaivatonta. Siirtymät aktiviteettien välillä tuli animoida siten, että käyttäjälle ei jää epäselväksi, mistä aktiviteetista tullaan ja mihin ollaan siirtymässä. Käytettävän laitteen ulkoisista mitoista riippumatta sovellusta tulisi pystyä käyttämään jopa yhdellä kädellä. Käyttökokemukseen vaikuttaa myös jokaisen aktiviteetin yhdenmukainen ulkoasu, mikä tuli myös ottaa huomioon käyttöliittymää toteuttaessa.

2.3 Vanhempien Android-versioiden tukeminen

Sovelluksen tuli toimia mahdollisimman monilla erilaisilla potentiaalisen käyttäjäryhmän suosimilla Android-laitteilla. Vähimmäis-API-versioksi valittiin API 8, koska sitä vanhempia laitteita on enää hyvin vähän käytössä (Android Developers Dashboard). Tuetuissa laitteissa tuli olla Internet-yhteys, kamera ja mahdollisuus puheluihin. Sovelluksen toteutuksessa tuli ottaa huomioon, että sovellus toimii ja käyttäytyy samalla tavalla tuetusta laitteesta ja versiosta riippumatta.

3 ANDROID

Android on Googlen kehittämä mobiililaitteiden käyttöjärjestelmä. Androidin ohjelmistopino sisältää itse käyttöjärjestelmän lisäksi myös väliohjelmistoja ja valmiiksi asennettuja laitteen käytön mahdollistavia perusohjelmia. Androidin kehittämisestä vastasi alun perin Android Inc., jonka Google myöhemmin osti. Nykyisin Androidin kehittämisestä vastaa Open Handset Alliance, joka koostuu yli 80 ohjelmisto- ja laitevalmistajasta ja teleoperaattorista. (Open Handset Alliance Members.)

3.1 Android-käyttöjärjestelmä

Android-käyttöjärjestelmän suosio johtuu sen helppokäyttöisyydestä, muokattavuudesta ja laitteen saatavilla olevien ohjelmien suuresta määrästä. Sovellusten kehittäjien keskuudessa Android on suosittu mm. siksi, että se pohjautuu avoimeen lähdekoodiin. Pohjimmiltaan Androidin ohjelmistopino pohjautuu Linuxiin. Android-käyttöjärjestelmän ohjelmistokehitys sijaitsee Java-kirjastojen päällä. Java on Sun Microsystemsin kehittämä tietoverkkoihin suunnattu olio-ohjelmointikieli (Vesterholm & Kyppö 2008).

3.1.1 API-versiot

API-versio on kokonaisluku, joka kasvaa versionumeroiden ja päivitysten mukana. Uusi API-versio sisältää uusia toiminnallisuuksia, joita ei ole välttämättä vanhemmissa versioissa tai ne on toteutettu uudella tavalla. Android-sovellukselle tulee määrittää minimi-API-versio, jota vanhemmissa laitteissa sovellus ei toimi. Android versiot on myös nimetty. Nimeämisperiaatteena on aloittaa uusi version nimi aina seuraavalla aakkosella ja nimetä versio jonkin herkullisen leivoksen tai muun makean mukaan. Esimerkiksi Android 4.0 on nimeltään Ice Cream Sandwich ja Android 4.1 Jelly Bean. (Android Developers, Build Numbers.)

3.1.2 Aktiviteetit ja fragmentit

Aktiviteetti on näkyvissä oleva yksi näkymä, jota käyttäjä voi käyttää. Ohjelma voi vaihtaa aktiviteettia käyttäjän siirtyessä toiminnallisuudesta toiseen. Aiemmat aktiviteetit voidaan jättää taustalle ja niihin voidaan palata uudelleen tai ne voidaan sulkea ja tarvittaessa ladata uudelleen.

Android-käyttöjärjestelmän 3.0 (API 11) versiosta alkaen aktiviteetit pystyvät hyödyntämään fragmentti-luokkia saadakseen käyttäjälle mm. paremman, suuria näyttöjä tukevan käyttöliittymän (Android Developers, Fragments). Fragmentit auttavat myös sovelluksen skaalaamisessa pienille ja suurille näytöille.

3.1.3 Käyttöliittymät

Android-käyttöliittymän sovellukset voivat toimia joko pysty- tai vaakasuunnassa. Käyttöliittymä voi mukautua laitetta käännettäessä. Esimerkiksi käännettäessä pystysuunnasta vaakasuuntaan voidaan valikko siirtää ylhäältä vasemmalle. Kehittäjä voi myös lukita näytön asennon, jolloin se toimii vain

tietyssä asennossa. Android-käyttöjärjestelmän käyttöliittymäkomponentit ovat kehittyneet versio-numeroiden kasvaessa, joten vanhemmat versiot eivät välttämättä tue uudempia muutoksia. Tämä saattaa aiheuttaa ongelmatilanteita kehittäjien keskuudessa, kun halutaan sovelluksen tukevan niin useaa Android-laitetta kuin mahdollista.

3.2 Android-pohjaiset laitteet

Android-käyttöjärjestelmää käyttäviä laitteita ovat mm. älypuhelimet, tablettitietokoneet ja tv-mediatoistimet. Suurimpia Android-puhelinlaitteiden valmistajia ovat HTC, LG, Motorola, Samsung ja SonyEricsson (SamMobile, Samsung is the top manufacturer of Android in Q2). Erilaisia Android-laitteita ja niiden eri versioita on 3 921 kpl (Google Play Developer Console).

3.3 Google Play

Google Play on Googlen omistama digitaalinen sisältöpalvelu. Pääasiassa palvelusta ladataan Android-sovelluksia, mutta sieltä voi ladata myös mm. musiikkia, elokuvia, kirjoja ja lehtiä. Elokuvat ja TV-palvelut ovat saatavilla vain muutamissa maissa. Aiemmin Android-sovellukset ladattiin Android Marketista, joka yhdistyi muiden Googlen palveluiden kanssa maaliskuussa 2012 uudeksi Google Play -palveluksi. (Hyvästi Android Market, Tietokone.fi.) Google Play on useimmiten esiasennettuna Android-laitteissa.

4 KÄYTETYT TYÖKALUT JA LAITTEET

Sovelluksen kehittämiseen tarvittiin kehitysympäristö, joka tukee Java-sovellusten kehittämistä ja johon tuli pystyä asentamaan ADT-liitännäinen. Valmiin ulkoasun elementit olivat Adobe Photoshop tiedostomuodossa, joten elementtien erotteluun tuli valita kyseisten tiedostojen avaamiseen kykenevä ohjelma.

4.1 Eclipse

Sovellus kehitettiin Eclipse sovelluskehittimellä, johon ladattiin Android kehittäjän työkalut (Android Developer Tools). Android suosittelee käyttämään Eclipseä vedoten sen monipuolisuuteen. Eclipsen positiivisia puolia ovat mm. kehitettävän sovelluksen testaus niin fyysisillä, kuin virtuaalisillakin laitteilla sekä tehokas debuggaus.

4.2 Adobe Photoshop

Sovelluksen ulkoasun elementtien erotteluun valittiin Adobe Photoshop CS5.5. Adobe Photoshop on kuvankäsittelyohjelma, jonka on kehittänyt Adobe Systems. Photoshop on yksi suosituimmista kuvankäsittelyohjelmista.

4.3 Fyysiset ja virtuaaliset laitteet

Sovelluksen kehittämisessä ja testauksessa käytettiin sekä fyysisiä, että virtuaalisia Android-laitteita. Fyysisillä laitteilla kehittäminen ja testaaminen oli paljon lähempänä loppukäyttäjän kokemusta kuin pelkillä virtuaalilaitteilla testaaminen kehitysympäristössä.

Sovellusta kehitettiin ja testattiin seuraavilla fyysisillä laitteilla:

- HTC Desire HD (Android-versio 2.3.5, API 10)
- Samsung Galaxy Mini (Android-versio 2.2, API 8)
- Samsung Galaxy Note 10.1 3G (Android-versio 4.1.2, API 16)
- Samsung Galaxy S3 (Android-versio 4.1.2, API 16)
- Samsung Galaxy S4 (Android-versio 4.2, API 17).

Jatkuvasti käytössä olivat Samsung Galaxy Note 10.1 3G suuren näyttökoon vuoksi, HTC Desire HD vanhemman Android-version (2.3.5) vuoksi sekä Samsung Galaxy Mini pienen näyttökoon ja resoluution vuoksi. Eri resoluutioiden, näyttökokojen ja Android-versioiden yhtäaikainen kehittäminen säästi aikaa projektin lopusta, kun käyttöliittymä ja toiminnallisuudet oli jo testattu hyvin laajalaisesti.

Android Development Tools:n AVD Managerissa pystyy määrittämään virtuaalisia laitteita. Laitteita ajetaan kehitysympäristössä emulaattorilla. Android Virtual Device (AVD) Managerissa pystyy määrittämään laitteen ominaisuudet hyvin tarkasti. Tarjolla on myös valmiita asetuskokoonpanoja, esimerkiksi Googlen omia Nexus laitteita.

Virtuaaliselle laitteelle pystyy AVD Managerissa määrittelemään ominaisuuksia, kuten laitteen API-version, suorittimen tyypin, fyysisen näppäimistön, etu- ja takakameran, RAM-muistin määrän, laitteen sisäisen tallennustilan sekä SD-muistikortin ja sen tallennustilan.

5 TYÖN TOTEUTUS

Työn suorituksen ja testaamisen kannalta parasta oli tehdä yksi toiminnallisuus kerrallaan ja testata sen toimivuus perusteellisesti ennen seuraavan toiminnallisuuden toteutusta. Tämä auttoi seuraavissa työvaiheissa. Toiminnallisuudet tehtiin seuraavassa järjestyksessä: latausikkuna, ilmoitusten selaus, yksittäisen ilmoituksen esikatselu, kuvan esikatselu, ilmoitusten haku, tietoja-sivu, ilmoituksen lisäys, kuvan lisäys ilmoitukseen.

Ilmoitusten selauksen toteuttaminen vei eniten aikaa, koska samalla tuli luoda käyttöliittymäelementit ja yhtenäinen ulkoasu, jota mukailtiin myöhempien toiminnallisuuksien luonnissa.

Sovelluksen testaaminen fyysisillä ja virtuaalisilla laitteilla jokaisen toiminnallisuuden luomisen yhteydessä säästi aikaa työn lopputestaukselta ja lopussa havaittavien virheiden korjaukselta. Sovelluksen oltua lähes valmis toteutettiin kokonaisvaltainen testaus erilaisilla Android-laitteilla.

5.1 Sovelluksen rakenne

Androidille tyypillisesti sovelluksen rakenne toteutettiin siten, että eri toiminnallisuudet sijaitsivat eri välilehdillä. Määrittelyvaiheessa sovellukselle määriteltiin, että sen tulee toimia vain pystyasennossa. Vaikka hyvänä käytänteenä pidetään, että sovelluksen tulisi toimia kaikissa orientaatioissa, tähän sovellukseen tämä rajoite tuntui sopivan hyvin ja se helpotti käyttöliittymän luomista huomattavasti.

5.1.1 Vaatimusten saavuttaminen vaadituilla API-versioilla

Android-sovellukset ovat tukeneet välilehtiä vasta API-versiosta 11 lähtien ja luotavan sovelluksen tuli tukea API-versioita alaspäin versioon 8 asti, joten välilehdet tuli toteuttaa hyväksikäyttäen ActionBarSherlock liitännäistä. ActionBarSherlock on Android support libraryn liitännäinen. Android support library mahdollistaa uudempien API-versioiden ominisuuksien käytön vanhemmissa versioissa. ActionBarSherlock on laajasti käytetty ja se on helppokäyttöinen ja muokattavissa omien tarpeiden mukaan.

Sovellus toteutettiin siten, ettei API-versiosta riippuen mitään toiminnallisuutta täytynyt tehdä kahdennettuna eri versioille, vaan kaikki toiminnallisuus toimii sellaisenaan kaikilla laitteilla.

Käyttöliittymän toteutuksessa ei käytetty staattisia mittoja, vaan käyttöliittymä skaalautuu näytön koosta ja resoluutiosta riippumatta samannäköisenä kuitenkin niin, että suuremman näytön kuva ei ole vain suurennos pienikokoisemman näytön kuvasta.

5.1.2 AndroidManifest.xml-tiedosto

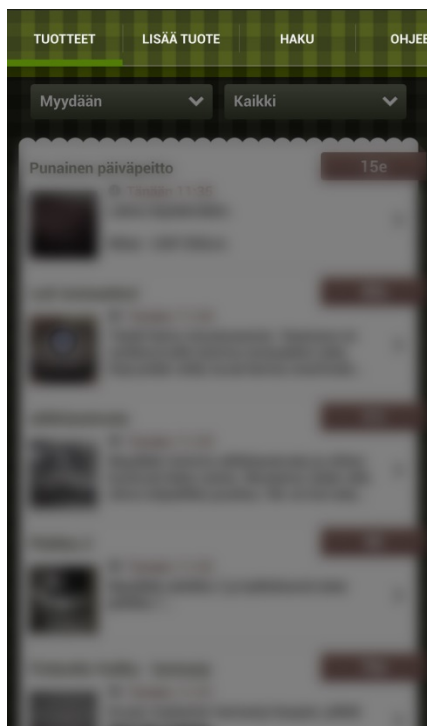
Jokaisella Android-sovelluksella on oma AndroidManifest.xml-tiedosto, josta käy ilmi koko sovelluksen ominaisuudet. Tiedostossa käsitellään mm. seuraavat asiat:

- sovelluksen paketin nimi

- sovelluksen versiokoodi ja versionimi
- minimi ja kohde API-versiot
- käyttäjältä varmistettavat laitteen käyttöoikeudet
- sovelluksen nimi
- sovelluksen kuvake
- kaikki näkymät ja niiden tyylit, orientaatiot ja otsikot.

5.1.3 Aktiviteettien välillä navigointi

ActionBarSherlockin muokattiin sovelluksen ulkoasun mukaiseksi ylikirjoittamalla perusteema. Sovelluksesta poistettiin näkyvistä Android-sovellukselle ominainen otsikkorivi, jotta itse sovellukselle ja sen listalle saatiin mahdollisimman paljon tilaa. Välilehdillä navigointi tapahtuu näytön yläreunassa sijaitsevalla palkilla ja sen ulkoasu mukailee muuta sovellusta.



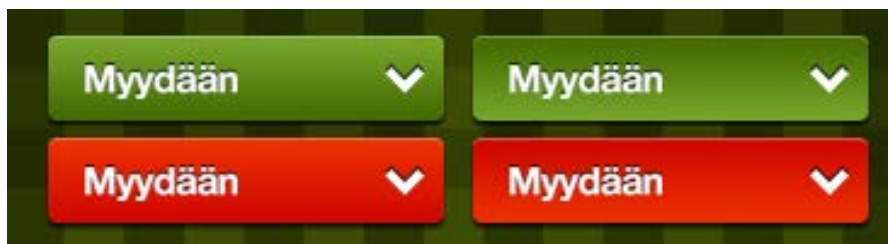
KUVA 1. Välilehdillä navigointi

Yllä olevassa kuvassa (kuva 1) näkyy, kuinka navigointipalkki mukailee sovelluksen teemaa siten, että se on lähes läpinäkyvä paljastaen taustakuvan. Palkissa valitun välilehden merkinä on sovelluksen värimaailmaa mukaileva tumman vihreä viiva.

Välilehtien sisällössä käytettiin hyväksi fragment-luokkia. Välilehteä vaihdettaessa nykyinen, jo luotu välilehti jää taustalle. Tämä parantaa käyttökokemusta eikä välilehden uudelleen avaaminen aiheuta koko aktiviteetin uudelleen latautumista.

5.1.4 Omat käyttöliittymäelementit

Tehtaessa ilmoitusten selaus -välilehteä tuli määritellä joidenkin sovelluksessa käytettävien käyttöliittymäelementtien ulkoasu ja käyttäytyminen.



KUVA 2. Käyttöliittymän painikkeet ja niiden painallukset

Kuva 2 näyttää kaksi eri sovelluksessa käytettävää painiketta. Painikkeisiin tuli tehdä tekstin lisäksi pyöristykset reunoihin sekä oikeassa reunassa sijaitseva alaspäin osoittava nuoli. Tämä saatiin aikaan määrittelemällä button-elementille oma tyyli tiedostossa. Tyyliä kuvataan XML-merkintäkielellä.

```
<?xml version="1.0" encoding="utf-8"?>
<layer-list xmlns:android="http://schemas.android.com/apk/res/android">
    <item>
        <shape android:shape="rectangle">
            <gradient android:startColor="#76a429" android:endColor="#446c02" android:angle="270" />
            <corners android:radius="3dp" />
        </shape>
    </item>
    <item android:right="14dp" android:bottom="2dp">
        <bitmap android:src="@drawable/camera_small_white" android:gravity="right|center_vertical" />
    </item>
</layer-list>
```

KUVA 3. Painikkeen tyyli tiedostossa

Kuvassa 2 oikealla olevien painikkeiden tila on alas painettu (pressed). Painikkeen taustaväri muuttuu painalluksen merkiksi. Kuvassa 3 on esitetty tyylin muodostaminen erillisessä tyyli tiedostossa. Layer-list-elementin sisäiset item-elementit kerrostetaan siten, että ensimmäisenä merkitty elementti näytetään pohjalta alkaen. Jokaiselle painikkeen tilalle tulee määrittää oma tyyli tiedostonsa.

Painikkeen tyylin muuttaminen esimerkiksi painaessa toteutetaan selector-elementillä, joka sisältää kaikkien tarvittavien tilojen tyyli tiedostot. Selector on kuvattu myös XML-merkintäkielellä.

```
<?xml version="1.0" encoding="utf-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:state_pressed="true" android:drawable="@drawable/button_camera_onpress" />
    <item android:state_enabled="false" android:drawable="@drawable/button_camera_disabled" />
    <item android:state_enabled="true" android:drawable="@drawable/button_camera_default" />
</selector>
```

KUVA 4. Painikkeen selector-tiedosto

Esimerkki selector-tiedostosta on esitetty kuvassa 4. Painikkeen tyyliksi annetaan selector-elementin sisältävä tiedosto, joka pitää huolen painikkeen eri tilojen tyyleistä.

5.1.5 Näkymät ja siirtymät

Tässä luvussa esitellään sovellukseen luodut näkymät ja niiden toiminnallisuus. Toiminnallisuudet ovat lähes samat kuin sovelluksen iOS-versiossa.

Sovellukseen luotiin seuraavat näkymät:

- latausikkuna
- ilmoitusten selaus
- yksittäinen ilmoitus
- kuvien esikatselu
- ilmoituksen lisäys
- ilmoitusten haku
- ohjeet
- ei yhteyttä -näkö.

5.1.5.1 Latausikkuna

Sovelluksen käynnistyksen yhteydessä ladataan JSON-rajapintaa hyväksi käyttäen osastot ja eri kategoriat, jotka tallennetaan laitteen tietokantaan muiden näkymien hyödynnettäväksi.

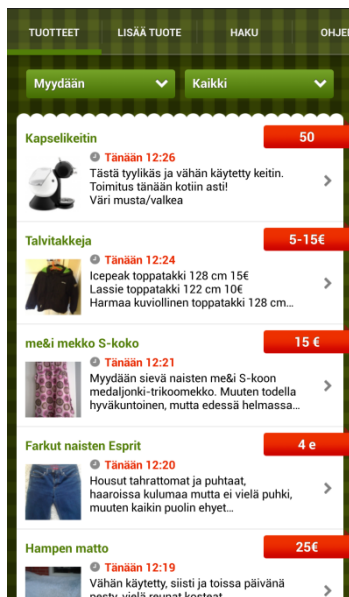


KUVA 5. Sovelluksen latausikkuna

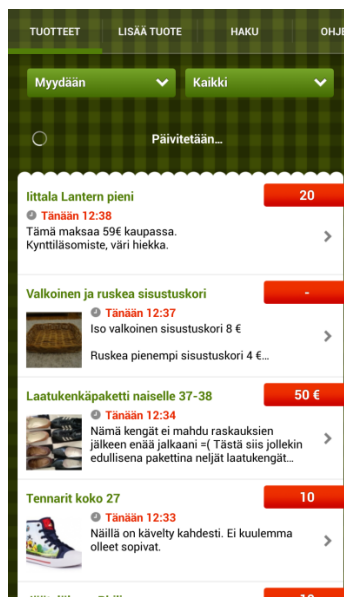
Yllä olevassa kuvassa (kuva 5) näkyy latausikkunan ulkoasu. Se on yksinkertainen ja siinä käytetään hyväksi Androidin natiivia latauspalkkia.

5.1.5.2 Ilmoitusten selaus

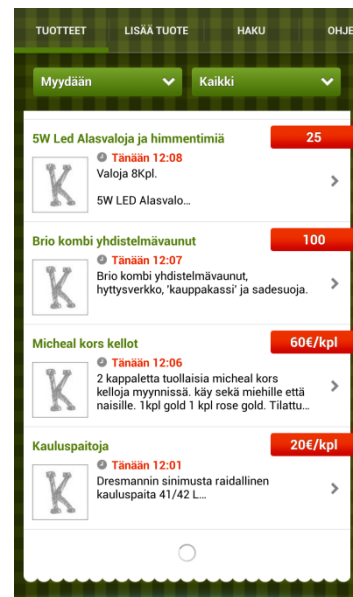
Sovelluksen alunäkymänä latausikkunan jälkeen näytetään ilmoitusten selaus. Sovelluksen käynnistyessä haetaan JSON-rajapintaa hyväksikäyttäen listan ensimmäiset 20 ilmoitusta.



KUVA 6. Ilmoitusten selaus



KUVA 7. Ilmoitusten päivitys



KUVA 8. Uusien ilmoitusten haku

Kuvassa 6 näkyy ilmoitusten selauksen oletusnäkymä. Jokaisella ilmoituksella on mahdollista olla lisäotsikko, hinta, kuva, lisäsjankohta ja esittelyteksti. Painettaessa yksittäistä ilmoitusta siirrytään seuraavaan näkymään liu'uttamalla se oikealta vasemmalle. Samaan aikaan vanha näkymä liukuu vasemmalta puolelta ruudusta ulos. Tämä auttaa käyttäjää ymmärtämään senhetkisen sijainnin sovelluksessa ja selkeyttämään käyttökokemusta.

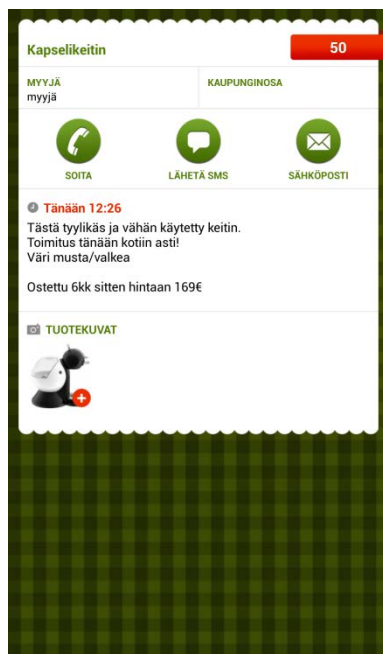
Lista voidaan päivittää ns. vedä ja vapauta päivittääksesi -toiminnolla, joka on yleistynyt mobiilisovelluksissa. Kuvassa 7 näkyy lista alas vedettynä ja juuri vapautettuna. Lista on toteutettu loppumattomana listana, eli kun käyttäjä selaa listan loppuun, ladataan lisää ilmoituksia, ks. kuva 8. Uusia ilmoituksia haetaan kerralla 20 kappaletta, jotta latausajat ovat mahdollisimman lyhyet ja käyttäjä kokee listan jatkuvan saumattomasti.

Listan yläpuolella voidaan valita painikkeista osio ja sen kategoria. Painiketta painettaessa avautuu lista, josta voidaan valita osio tai kategoria, joka ladattiin laitteen muistiin latausikkuna-näkymässä. Välittömästi valinnan jälkeen lista päivitetään siten, että haetaan 20 ensimmäistä ilmoitusta kyseisestä osiosta ja kategoriasta.

Ilmoituksen tietojen latauduttua siitä luodaan Product-luokka, joka lisätään ArrayList-listaan. Kaikkien tarvittavien ilmoitusten latauduttua ArrayList-lista välitetään ProductAdapterille, joka vastaa sovelluksen listan muodostamisesta. ProductAdapterissa on määritelty yksittäisen listan elementin käyttöliittymäelementit ja niiden tyylit sekä sijainnit. Tiedot näihin käyttöliittymäelementteihin ladataan ProductAdapterissa. ProductAdapter huolehtii myös ilmoituksen kuvan latauksesta. Kuvat listan ilmoituksiin ladataan eri säikeissä, mikä mahdollistaa useamman kuvan yhtäaikaista latauksen kerrallaan. Kuvat ladataan ja näytetään vasta, kun listaa on selattu siten, että ilmoitus on näkyvässä.

5.1.5.3 Yksittäinen ilmoitus

Käyttäjän valittua yksittäisen ilmoituksen joko "Ilmoitusten lisäys" -näkyellä tai "Ilmoitusten haku" -näkyellä avataan "Yksittäinen ilmoitus" -näkymä jolle sen luotaessa annetaan parametrina ilmoituksen yksilöity tunnus (ID).



KUVA 9. Yksittäinen ilmoitus

Näkymän latautuessa haetaan saadulla yksilöidyllä tunnuksella ilmoituksen tiedot ja näytetään ne kuvan 9 esittämällä tavalla. Ilmoittajaan voidaan ottaa yhteyttä suoraan sovelluksesta joko soittamalla, tekstiviestillä tai sähköpostilla.

Monipuolisuutta ja käyttäjäystävällisyyttä sovelluksessa lisäävät nopeat ja helpot tavat ottaa yhteyttä ilmoituksen jättäneeseen henkilöön. Jos ilmoituksessa on puhelinnumero, näytetään "Soita" ja "Lähetä sms" -painikkeet. "Soita"-painiketta painettaessa kutsutaan Androidin puhelunäkymää. Vastavasti "Lähetä sms"-painiketta painettaessa kutsutaan Androidin viestinäkymää.

```
imgBtnCall.setOnClickListener(new OnClickListener() {
    @Override
    public void onClick(View v) {
        if (((TelephonyManager) getSystemService(Context.TELEPHONY_SERVICE)).getPhoneType() != TelephonyManager.PHONE_TYPE_NONE) {
            Intent call_intent = new Intent(Intent.ACTION_CALL);
            call_intent.setData(Uri.parse("tel:" + phone));
            startActivity(call_intent);
        } else {
            Toast.makeText(getApplicationContext(), R.string.cant_call, Toast.LENGTH_LONG).show();
        }
    }
});
```

KUVA 10. Puhelunäkymän kutsu

Kuvassa 10 näkyy esimerkki puhelunäkymän kutsumisesta. Aluksi testataan onko laitteella mahdollisuutta soittaa puhelua tai lähettää tekstiviestiä. Jos laitteessa ei ole puhelumahdollisuutta näytetään käyttäjälle virheilmoitus, muuten kutsutaan näkymää. Käyttöjärjestelmä itse päättää minkä sovelluk-

sen puhelun soittamiseksi käynnistää. Puhelunäkymälle välitetään URI-muodossa mm. puhelinnumero.

```
imgBtnSms.setOnClickListener(new OnClickListener() {
    @Override
    public void onClick(View v) {
        if (((TelephonyManager) getSystemService(Context.TELEPHONY_SERVICE)).getPhoneType() != TelephonyManager.PHONE_TYPE_NONE) {
            Intent sms_intent = new Intent(Intent.ACTION_VIEW);
            sms_intent.setData(Uri.parse("sms:" + phone));
            sms_intent.putExtra("sms_body", product.get_title());
            startActivity(sms_intent);
        }
        else {
            Toast.makeText(getApplicationContext(), R.string.cant_sms, Toast.LENGTH_LONG).show();
        }
    }
});
```

KUVA 11. Viestinäkymän kutsu

Kuvassa 11 näkyy esimerkki viestinäkymän kutsumisesta. Aluksi testataan onko laitteella mahdollista soittaa puhelua tai lähettää tekstiviestiä. Jos laitteella ei voi lähettää tekstiviestiä näytetään käyttäjälle virheilmoitus, muuten kutsutaan näkymää. Myös viestin lähetyksessä järjestelmä päättää mikä sovellus tekstiviestin lähettämiseksi käynnistetään. Viestinäkymälle välitetään URI-muodossa puhelinnumero, sekä extrana viestin sisältö. Sovellus ei pysty lähettämään tekstiviestiä, vaan ainoastaan avaamaan viestin kirjoituksen ennalta syötetyillä tiedoilla.

Jos ilmoituksessa on sähköpostiosoite, näytetään "Sähköposti"-painike. Painiketta painettaessa kutsutaan Androidin SENDTO-näkymää, joka päättää saadun tiedon perusteella, millä sovelluksella tieto tulee lähettää. SENDTO-näkymälle annetaan URI-muodossa vastaanottaja (mailto) sekä viestin aihe (subject). Näistä tiedoista käyttöjärjestelmä päättää, millä sovelluksella viesti tulisi välittää eteenpäin.

Ilmoituksen kuvien määrä on rajattu maksimissaan viiteen kuvaan. Kuvat näkyvät ilmoitusnäköm alalaidassa. Kuvaa painettaessa avataan "Kuvien esikatselu"-näköm, jossa kuvia voi helposti selata.

5.1.5.4 Kuvien esikatselu

Näkömän latautuessa ladataan ilmoituksen kaikki kuvat parempilaatuisena ImagePagerAdapter-luokkaan, joka välitetään ViewPager-elementtiin, jotta kuvia voidaan selata näyttöä pyyhkäisemällä. ViewPager-elementissä yksittäinen kuva on TouchImageView-luokassa, joka lisää kuviin kuvien skaalaus-toiminnallisuuden kuvaa nipistämällä.



KUVA 12. Ilmoituksen kuvat

Katseltavan kuvan järjestysnumero näkyy näkymän alalaidassa, kuten kuvassa 12 näkyy. Näkymässä on myös painike, jolla se voidaan sulkea ja palata edelliseen näkymään. Siirtymä tähän luokkaan tapahtuu horisontaalisen siirtymän sijaan vertikaalisesti. Avattaessa näkymä ilmestyy näytön alalaidasta ja suljettaessa se liukuu takaisin alas.

5.1.5.5 Ilmoituksen lisäys

Käyttäjä voi helposti lisätä uuden ilmoituksen palveluun sekä liittää ilmoitukseen kuvia kamerasta tai valita niitä laitteen muistista. Ilmoituksen osion ja kategorian valinta tapahtuu ilmoitusten selauksesta tutuilla painikkeilla. Ilmoituksen lisäyksessä vaaditut tiedot tulee syöttää ennen kuin ilmoituksen lähettäminen palvelimelle on mahdollista.

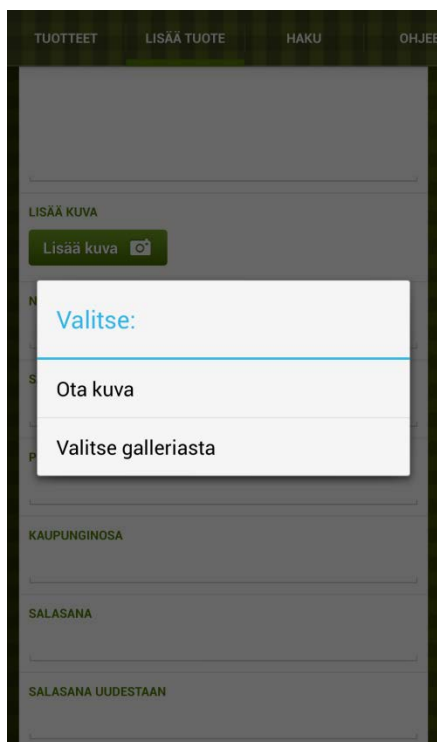
KUVA 13. Ilmoituksen lisäys -näkymä

Tietojen syöttämistä varten tekstikenttiin on valittu sisältötyypit tallennettavan tiedon mukaan. Otsikko-, hinta-, ilmoitusteksti-, nimi- ja kaupunginosa-kenttiin tieto syötetään tekstinä. Sähköpostikentässä on käytetty hyväksi Androidin sähköposti-sisältötyyppiä, jolloin näppäimisön ulkoasu muuttuu sähköpostin osoitteen syöttöä varten. Puhelinnumero-kentässä on käytetty hyväksi Androidin puhelinnumero-sisältötyyppiä, jolloin näppäimistön ulkoasu muuttuu siten, että vain numeronäppäimet ovat käytössä. Salasana-kentissä teksti näytetään tähtimerkein tai laittekohtaisilla merkeillä, jotka käyttöjärjestelmässä on määritelty. Salasanoiden vastaavuus tarkistetaan, ennen kuin ilmoituksen voi lähettää.

KUVA 14. Ilmoituksen lisäys -näytymän alaosa

Kuten kuvassa 14 näkyy, läheta-painike on "disabled"-tilassa, kunnes kaikki vaaditut tiedot on syötetty. Salasana kysytään varmistuksena kahdesti. Ilmoituksen lisäyksessä valinnaisia kenttiä ovat:

- hinta
- sähköpostiosoite
- kaupunginosa.



KUVA 15. Kuvan lisäys

Kuvien lisäys tapahtuu Lisää kuva -painikkella. Painiketta painettaessa voidaan valita otetaanko kuva kameralla vai valitaanko se laitteen muistista. Yhtä ilmoitusta kohden voidaan lisätä viisi kuvaa.

Kuvassa 15 näkyy valikko, joka avautuu Lisää kuva -painiketta painettaessa.

```
ContentValues values = new ContentValues();
values.put(MediaStore.Images.Media.TITLE, "Image File Name");
mCapturedImageURI = getSherlockActivity().getContentResolver().insert(MediaStore.Images.Media.EXTERNAL_CONTENT_URI, values);

Intent cameraIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
cameraIntent.putExtra(MediaStore.EXTRA_OUTPUT, mCapturedImageURI);
startActivityForResult(cameraIntent, 1);
```

KUVA 16. Kuvan lisäys ilmoitukseen laitteen kameralla

Kuten kuvassa 16 näkyy, kuvaa lisättäessä kutsutaan Androidin MediaStore.ACTION_IMAGE_CAPTURE -näkyä. Käyttöjärjestelmä päättää, mikä sovellus avataan kuvan ottamista varten. Kuvan ottamisen jälkeen kuvan EXIF-tiedoista tutkitaan, missä orientaatioissa kuva on otettu. Jos kuva ei ole oikeinpäin, se tulee kääntää ennen kuvan lisäystä ilmoitukseen.

```
Intent intent = new Intent();
intent.setType("image/*");
intent.setAction(Intent.ACTION_GET_CONTENT);
startActivityForResult(Intent.createChooser(intent, (CharSequence) getResources().getString(R.string.choose_from_gallery)), 2);
```

KUVA 17. Kuvan lisäys ilmoitukseen laitteen muistista

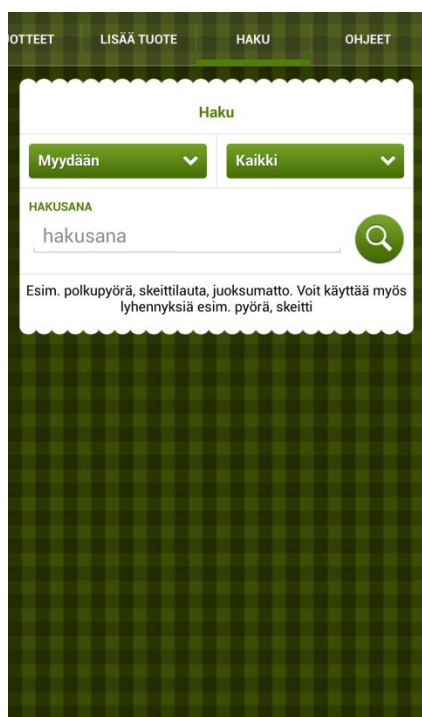
Kuva voidaan lisätä ilmoitukseen myös laitteen muistista. Kuvassa 17 näkyy näkymän kutsu, jossa tyypiksi määritellään kuvatiedostot. Käyttöjärjestelmä luo listan sovelluksista, joista voidaan tuoda kuvatiedostoja. Yksi näistä sovelluksista on laitteen oma kuvagalleria. Kuvia voidaan tuoda myös kolmannen osapuolen sovelluksista, kuten esimerkiksi Dropboxista tai Instagramista.

Kun ilmoituksen kaikki vaaditut kentät on täytetty, voidaan ilmoitus lähettää palvelimelle POST-tietona käyttäen hyväksi JSON-rajapintaa. Ilmoitusta lähettäessä lasketaan tiedonsiirtoon tarvittava tiedon määrä, jonka perusteella käyttäjälle näytetään ilmoituksen lisäyksen edistyminen latausikkunassa. Tämä lisää käyttäjäystävällisyyttä ja varmuutta, koska suurien kuvatiedostojen lähetyksessä hitaalla yhteydellä voi kestää hetken. Käyttäjälle on hyvä näyttää edistyminen, jos toiminnon suorittamisessa kestää useita sekunteja.

Ilmoituksen lisäyksen yhteydessä syötetyt nimi, puhelinnumero, sähköpostiosoite ja kaupunginosa tallentuvat lähetyksen jälkeen laitteen tietokantaan. Uutta ilmoitusta lisättäessä nämä tiedot haetaan valmiiksi laitteen tietokannasta niille varattuihin kenttiin. Tämä toiminnallisuus mahdollistaa jatkossa helpomman ilmoitusten lisäyksen, kun osa tiedoista on jo valmiiksi täytetty. Laitteeseen tallennetut tiedot voidaan poistaa Ilmoituksen lisäys -näköymästä sen tyhjennä-painikkeella.

5.1.5.6 Ilmoitusten haku

Ilmoitusten haku -näköymässä voidaan hakea ilmoituksia hakusanalla eri osioista ja kategorioista. Hakuehdot lähetetään palvelimelle POST-tietona ja palvelin palauttaa hakutulokset JSON-tietona. Näköymässä käytetään hyväksi samaa listaa ja adapteria, jota käytetään myös Ilmoitusten selaus -näköymässä. Listan toiminnallisuus on identtinen molemmissa näköymissä.



KUVA 18. Haku-näköymä

Käyttäjälle annetaan ohje hakusanojen käytöstä, jotta hakutuloksista olisi käyttäjälle mahdollisimman paljon hyötyä. Ohje ja sen asettelu näkyy kuvassa 18.

5.1.5.7 Ohjeet-näkymä

Sovellukseen toteutettiin myös ohjeet-osio, jossa kerrotaan ilmoituksen voimassaolosta, sen muokkaamisesta sekä sisällön rajoitteista. Näkymässä näytetään myös palvelun tarjoajan, suunnittelijan ja kehittäjän tiedot linkkeinä, jotka avautuvat selaimeen.

```
Intent browserIntent = new Intent(Intent.ACTION_VIEW, Uri.parse("http://www.rikujokinen.com"));
startActivity(browserIntent);
```

KUVA 19. Selainnäkömän kutsu

Avattaessa linkkiä selaimeen kuvan 19 esittämällä tavalla kutsutaan näkymää, jolle annetaan URI-muodossa sivuston osoite (URL). Jälleen käyttöjärjestelmä päättää, mikä sovellus käynnistetään.



KUVA 20. Ohjeet-näkymä

Ohjeet-näkymässä oikeassa yläkulmassa näytetään myös sovelluksen senhetkinen versio, kuten kuvassa 20 näkyy. Versionumero haetaan AndroidManifest.xml-tiedostosta.

5.1.5.8 Ei yhteyttä -näkömä

Jos laitteen verkkoyhteys menetetään, näytetään käyttäjälle Ei yhteyttä -näkömä. Sovelluksen tarkkailija kuuntelee verkkoyhteyden tilan muutosta ja tarkastaa, onko sovellus yhteydessä verkkoon tilan muuttuessa.



KUVA 21. Ei yhteyttä -näkymä

Käyttäjälle näytetään viesti yhteyden puuttumisesta (kuva 21). Käyttäjä voi yhteyden palautuessa palata edelliseen näkymään laitteen takaisin-painikkeella.

5.2 Sovelluksen käyttäjältä vaatimat oikeudet

Android-käyttöjärjestelmässä käyttäjän asennettaessa sovellusta tulee hänen hyväksyä sovelluksen oikeudet käyttää laitteen tiettyjä toimintoja. Sovellus ei voi poiketa oikeuksista ja niiden rajoituksista. Kun sovellusta päivitetään ja käyttäjä lataa uuden version sovelluksesta, tulee käyttäjän hyväksyä mahdolliset uudet pyydettyt oikeudet. Sovellus ei saa pyytää oikeuksia, joita se ei sovelluksen käytössä tarvitse.

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.CALL_PHONE" />
<uses-feature android:name="android.hardware.telephony" android:required="false" />
<uses-permission android:name="android.permission.SEND_SMS" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<uses-permission android:name="android.permission.FLASHLIGHT" />
<uses-permission android:name="android.permission.CAMERA" />
<uses-feature android:name="android.hardware.camera" android:required="false" />
<uses-feature android:name="android.hardware.camera.flash" android:required="false" />
<uses-feature android:name="android.hardware.camera.autofocus" android:required="false" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
```

KUVA 22. Sovelluksen käyttämät laitteen ominaisuudet

Kuvassa 22 näkyy vaadittujen oikeuksien osa AndroidManifest.xml-tiedostosta. Sovellus tarvitsee toimiakseen Internet-yhteyden, sekä sen tilan muuttuessa pääsyt tutkimaan langattoman verkon ja mobiiliverkon tilaa.

Sovellus tarvitsee myös oikeuden käyttää laitteen puheluominaisuutta, jotta ilmoituksen jättäneeseen henkilöön voidaan sovelluksen kautta ottaa yhteyttä. Tämä ominaisuus on merkitty valinnaiseksi ominaisuudeksi, jotta sovellusta olisi mahdollista käyttää, vaikkei laitteella voisiakaan soittaa.

Sovellus pyytää myös oikeutta lähettää tekstiviestejä ja käyttää laitteen kameraa. Joissain laitteissa kameran ominaisuudet ovat vähäisemmät, ja tästä syystä oikeudelle on määriteltä valinnaisiksi ominaisuuksiksi salaman ja automaattisen tarkennuksen käyttö.

Sovelluksen tulee myös päästä käsiksi laitteen tallennustilaan, jotta kuvia ilmoituksiin voidaan ladata laitteen muistista.

5.3 Valmiiden käyttöliittymäelementtien ja luokkien hyödyntäminen

Käyttöliittymän toteutuksessa käytettiin useita valmiita ylikirjoitettuja käyttöliittymäelementtejä ja luokkia. Sovelluksen näkyvin osa, ilmoitusten listaus on toteutettu ylikirjoitetulla ListView-elementillä. RefreshableListView nimellä kulkeva luokka on luotu vuonna 2011 ja se löytyy GitHub verkkopalvelusta. RefreshableListView lisää listaan toiminnallisuuden, jolla listaan saadaan uusi toiminnallisuus vetämällä listaa alaspäin listan alussa ja vapauttamalla se. Tätä käytettiin listan elementtien päivittämisessä. Luokka ei toiminut sellaisenaan halutulla tavalla vaan sitä tuli muokata, jotta sen tyyli ja toiminnallisuus vastasivat sovelluksen tarkoitusta.

Kuvien lataamiseen, näyttämiseen ja niiden hallitsemiseen välimuistissa käytettiin BitmapManager-luokkaa. Se mahdollistaa mm. kuvien lataamisen eri säikeissä, jotta sovellus voi suorittaa useita eri toimintoja yhtä aikaa, eikä sen toiminta pysähdy jokaisen kuvan latauksen kohdalla.

Androidin TextView-elementin tekstin katkaisu toimii vain kahden rivin tai alle kahden rivin teksteissä. Sovelluksen tulee näyttää ilmoituksen tekstistä kolme tai neljä riviä ja katkaista teksti sen jälkeen kolmella pisteellä (engl. ellipsize). Tämä toiminnallisuus saatiin käyttämällä GitHub-verkkopalvelusta ladattua valmista vuonna 2012 luotua EllipsizingTextView-luokkaa.

Kuvien esikatselussa kuvat esitetään TouchImageView-luokan avulla, jossa on ylikirjoitettu joitakin ImageView-luokan metodeita. TouchImageView on kirjoitettu 2011 ja se löytyy GitHub-verkkopalvelusta. Luokka tuo lisää toiminnallisuutta kuvan käsittelyyn, kuten tässä tapauksessa kuvien skaalaamisen nipistämällä.

Palvelimelta ladattavien osioiden ja kategoroiden nimien tallentamiseen laitteen tietokantaan käytettiin itse kirjoitettua DatabaseHandler-luokkaa, jolla luodaan sopivat tietokannan taulut ja kyselyt. Tietokanta tyhjennetään jokaisen käynnistyksen yhteydessä ja ladataan uusimmat tiedot osioista ja kategorioista.

Palvelimelta saapuvan JSON-tiedon jäsentämiseen käytettiin valmista JSONParser-luokkaa. Luokka pystyy jäsentämään saapuvat JSON-taulukot Java-ohjelmointikielen ymmärtämään muotoon. Luokka on kirjoitettu vuonna 2012 ja se löytyy androidhive.info-verkkosivustolta (AndroidHive JSONParser).

Ilmoitusta lisättäessä tulee laskea tiedonsiirtoon tarvittava tiedon määrä. Tämä saataisiin toteutettua MultiPartEntity luokkaa hyväksikäyttämällä, jollei kuvia olisi useampia. Koska kuvia on voitava lähettää kerralla useampia, käytettiin tiedon lähettämiseen palvelimelle itse kirjoitettua CustomMultiPartEntity-luokkaa, jolla kyseinen toiminnallisuus saatiin toteutettua.

5.4 JSON-rajapinta

Tiedonsiirto sovelluksen ja palvelimen välillä tapahtuu palvelimella sijaitsevan PHP-ohjelmointikielillä ohjelmoidun rajapinnan kautta. Rajapinta palauttaa tiedot JSON-tiedonsiirtomuodossa. Rajapinta sisältää kolme toiminnallisuutta: osioiden ja kategorioiden hakeminen, ilmoitusten hakeminen tietokannasta tietyillä hakuehdoilla sekä ilmoituksen tallentaminen tietokantaan. Hakuehdot rajapintaan syötetään GET-tietona, eli ne sisällytetään haettavan tiedoston URL-osoitteeseen. Rajapinta palauttaa kerrallaan 20 ilmoitusta.

Ilmoituksia hakiessa voidaan määritellä seuraavat hakuparametrit:

- yksittäisen ilmoituksen ID
- ilmoituksen tyyppi (myydaan, ostetaan, lahjoitetaan, kyydit, asunnot)
- yksittäinen kategoria
- vapaa tekstihaku.

Ilmoituksen tyypit ja kategoriat haetaan sovelluksen käynnistyessä, jotta tietojen päivittyessä palveluun sovellukseen ei tarvitse tehdä muutoksia, vaan sovellus mukautuu kaikkeen palvelussa sijaitsevaan staattisen tiedon muuttumiseen. Hakuparametrejä voidaan käyttää useampia yhtäaikaaisesti. Hakutulosten alkamiskohta voidaan myös määritellä erikseen. Tämä mahdollistaa seuraavien 20 ilmoituksen hakemisen. Hakutulokset palautetaan palvelimelta JSON-taulukkona, joka parsitaan sovelluksessa erillistä luokkaa hyväksikäyttäen.

Ilmoituksen lisäyksessä palvelimelle lähtettävä tieto lähetetään POST-tietona. Lisättävän ilmoituksen parametrit ovat seuraavat:

- tyyppi (myydaan, ostetaan, lahjoitetaan, kyydit, asunnot)
- yksittäisen kategorian ID
- ilmoituksen otsikko
- ilmoituksen teksti
- tuotteen hinta
- ilmoituksen lisääjän nimi
- ilmoituksen lisääjän puhelinnumero
- ilmoituksen lisääjän sähköpostiosoite
- tuotteen sijainti (kaupunginosa)

- kuva 1
- kuva 2
- kuva 3
- kuva 4
- kuva 5
- salasana.

Palveluun voidaan tallentaa viisi kuvaa yhtä ilmoitusta kohden. Salasanan puuttuessa palvelin generoi salasanan ja lähettää sen käyttäjän sähköpostiin, jos hän on sellaisen määrittänyt. Lisättyä ilmoitusta pystyy muokkaamaan verkkopalvelussa joko puhelinnumerolla ja salasanalla tai sähköpostiosoitella ja salasanalla.

5.5 Testaus

Sovellukselle ei laadittu erikseen testaussuunnitelmaa, vaan sitä testattiin jokaisen toiminnallisuuden valmistuttua. Testauksessa otettiin huomioon laajalti käyttäjän mahdollisesti tekemät virheet ja niistä syntyvät virhetilanteet. Virheiksi luettiin myös käyttäjäkokemusta alentavat tekijät. Uusien toiminnallisuuksien valmistuttua testattiin kokonaisuutta, kuitenkin keskittyen uusimpaan toiminnallisuuteen.

Sovelluksen ollessa lähes valmis sitä testattiin myös ulkopuolisella käyttäjällä ns. musta laatikko -testauksena. Käyttäjä ei siis tiennyt kuinka sovelluksen tulisi toimia tai mitä kooditasolla tapahtuu. Tällaisessa testauksessa saadaan mahdollisesti aikaan oikeita loppukäyttäjän virhetilanteita. Ennen sovelluksen julkaisemista Google Play -sisältöpalvelussa sitä käytettiin päivittäin ja siitä alkoi kehittyä tapa. Voitiin siis hyvillä mielin todeta, että sovellus on hyödyllinen, toimii jouhevasti ja käyttäjäystävällisesti ja sitä on hauska käyttää.

6 YHTEENVETO

Projektin tavoitteena oli luoda kuopionkirppari.fi-sivuston käyttöä varten Android-sovellus, joka julkaistaisiin Google Play -sisältöpalvelussa. Sovelluksen tuli mukailla jo aiemmin luotua iOS-käyttöjärjestelmän versiota kuitenkin mukaillen Android-käyttöjärjestelmälle ominaisia piirteitä. Projektin lopputuloksena syntyi valmis sovellus, joka julkaistiin heti sen valmistuttua. Sovellusta on ladattu 15.9.2013 mennessä Google Play -sisältöpalvelusta 308:aan eri Android-laitteeseen.

Projekti antoi sopivasti haasteita ja mietittävää sekä kiinnostava aihe varmasti joudutti työn valmistumista ennen tavoiteaikaa. Hyvin pieniä muutoksia lukuun ottamatta en tekisi mitään sovelluksessa toisin. Toiminnallisuusmäärittely sekä valmis rajapinta tietokantaan asiakkaan puolesta olivat hyvin tehdyt. Jos sovellus olisi tullut toteuttaa alusta alkaen, olisin tehnyt asiat varmasti samalla tavalla, kuin ne nyt on toteutettu. Olen tyytyväinen lopputulokseen ja käytän sovellusta edelleen päivittäin.

Sovellusta voisi jatkokehittää laajentamalla sen toimintoja muihin kaupunkeihin. Sovelluksen Ohjeet-näkymään voisi toteuttaa valikon, josta käyttäjä voisi valita haluamansa kaupungin. Tämä toiminnallisuus on sovelluksen uudemmassa iOS-käyttöjärjestelmän versiossa.

LÄHTEET

ANDROID DEVELOPERS DASHBOARD. [Viitattu 2013-10-02.] Saatavissa:
<http://developer.android.com/about/dashboards/index.html>

ANDROID DEVELOPERS, BUILD NUMBERS. [Viitattu 2013-10-02.] Saatavissa:
<http://source.android.com/source/build-numbers.html>

ANDROID DEVELOPERS, FRAGMENTS. [Viitattu 2013-10-02.] Saatavissa:
<http://developer.android.com/guide/components/fragments.html>

ANDROIDHIVE JSONPARSER. [Viitattu 2013-10-02.] Saatavissa:
<http://www.androidhive.info/2012/01/android-json-parsing-tutorial/>

GOOGLE PLAY DEVELOPER CONSOLE. [Viitattu 2013-10-02.] Saatavissa:
<https://play.google.com/apps/publish/>

KOTILAINEN, Samuli 2012-03-07. Hyvästi Android Market. [Viitattu 2013-11-04.] Saatavissa:
http://www.tietokone.fi/artikkeli/uutiset/hyvasti_android_market_google_play_yhdistaa_palvelut

OPEN HANDSET ALLIANCE MEMBERS. [Viitattu 2013-10-02.] Saatavissa:
http://www.openhandsetalliance.com/oha_members.html

SAMSUNG IS THE TOP MANUFACTURER OF ANDROID IN Q2. [Viitattu 2013-10-02.] Saatavissa:
<http://www.sammobile.com/2013/07/23/report-samsung-is-the-top-manufacturer-of-android-in-q2/>

SIVA, Praga 2011-07-14. MySQL Connector. [Viitattu 2013-07-17.] Saatavissa:
<http://appinventor.blogspot.fi/2011/07/android-mysql.html>

VESTERHOLM M. ja KYPPÖ J. 2008. Java-ohjelmointi. Talentum Media Oy: Helsinki.

LIITE 1

PROJEKTISUUNNITELMA



Projektisuunnitelma

Suunnitelman laatija:	Riku Jokinen
Suunnitelman lyhyt kuvaus:	Dokumentti määrittää projektin kuopionkirppari.fi Android sovellus
Päivämäärä:	13.4.2013
Versio:	1.0

Päivämäärä	Versio	Tehtävät	Tekijä
10.4.2013	0.1	Projektisuunnitelman teko alkaa	Riku Jokinen
13.4.2013	1.0	Projektisuunnitelman ensimmäinen versio valmis	Riku Jokinen

SISÄLLYS

1	JOHDANTO	36
1.1	Yleiskuvaus	36
1.2	Tuote	36
1.3	Suunnitelman ylläpito	36
1.4	Määritelmät, termit ja lyhenteet	36
2	PROJEKTIN ORGANISOINTI	38
2.1	Projektin vaiheistus	38
2.1.1	Esitutkimus	38
2.1.2	Määrittely	38
2.1.3	Suunnittelu	38
2.1.4	Toteutus	38
2.1.5	Testaus	39
2.1.6	Käyttöönotto ja ylläpito	39
2.2	Organisaation rakenne	39
2.3	Sidosryhmien kuvaus	39
2.4	Vastuuhenkilöt	39
3	PROJEKTIN OHJAAMINEN	40
3.1	Tavoitteet ja priorisointi	40
3.2	Riskien hallinta	40
3.3	Seuranta ja ohjaus	40
3.4	Henkilöresurssien käytön suunnittelu	40
4	TEKNIikka	41
4.1	Menetelmät ja työkalut	41
4.2	Dokumentointi	41
4.3	Laadunvarmistus	41
5	VAIHEET, AIKATAULUT JA BUDJETTI	42
5.1	Projektin osittaminen	42
5.2	Resurssien käyttö ajan funktiona	42
5.3	Budjetti ja resurssien allokointi	42
5.4	Aikataulu	42

1 JOHDANTO

1.1 Yleiskuvaus

Dokumentti määrittää, kuinka kuopionkirppari.fi Android sovellus projekti toteutetaan. Projektin tarkoituksena on luoda android sovellus, jolla voidaan käyttää kuopionkirppari.fi sivustoa android laitteissa.

1.2 Tuote

Sovellus myötäilee ulkoasultaan ja toiminnallisuudeltaan jo toteutettua vastaavaa iOS käyttöjärjestelmälle tuotettua sovellusta. Sovellus on tarkoitettu henkilöille, jotka omistavat android laitteen ja haluavat käyttää kuopionkirppari.fi palvelua mobiilisti.

1.3 Suunnitelman ylläpito

Sovelluksen määrittelyjen muuttuessa myös projektisuunnitelmaa päivitetään. Suunnitelman versio-numeroa päivitetään ja tehdyt muutoksen kuvataan pääosin dokumentin sivulla 2.

1.4 Määritelmät, termit ja lyhenteet

Android

Android on mobiililaitteille suunniteltu ohjelmistopino, joka sisältää käyttöjärjestelmän, väliohjelmistoja ja perusohjelmia. Käyttöjärjestelmäydin on avoimen lähdekoodin GPLv2 lisensoitu Linux. Nykyään androidin omistaa Google ja sen kehittämisestä vastaa Open Handset Alliance.

iOS

iOS on Applen kehittämä käyttöjärjestelmä, joka on käytössä Applen mobiililaitteissa sekä Apple TV-laitteissa. iOS perustuu Darwin BSD –käyttöjärjestelmään. Lähdekoodi on pääosin suljettua.

kuopionkirppari.fi

kuopionkirppari.fi on Tmi Mikko Hopian tuottama ja ylläpitämä verkkopalvelu. Palvelu on yksityisille henkilöille tarkoitettu ilmoituskanava. Ilmoitusten jättäminen ja lukeminen on maksutonta, eikä edellytä rekisteröitymistä.

JSON rajapinta

Ohjelmointirajapinta (engl. Application programming interface, API) mahdollistaa eri ohjelmien keskustelun keskenään. Ohjelmat voivat tehdä pyyntöjä ja vaihtaa tietoja keskenään. JSON on yksinkertainen tiedonsiirtomuoto. Nimestään huolimatta JSON (JavaScript Object Notation) on JavaScriptistä riippumaton.

Eclipse

Eclipse Foundationin kehittämä ohjelmointiympäristö, joka tukee mm. seuraavia ohjelmointikieliä: Java, C, C++ ja PHP. Ympäristöä kehitetään avoimen lähdekoodin lisenssillä.

Google Play

Googlen omistama digitaalinen sisältöpalvelu. Pääasiassa palvelusta ladataan Android sovelluksia, mutta sieltä löytyy myös mm. musiikkia, elokuvia, kirjoja ja lehtiä. Aiemmin Android sovellukset lattiin Android Marketista, joka yhdistyi muiden Googlen palveluiden kanssa maaliskuussa 2012 uudeksi Google Play -palveluksi.

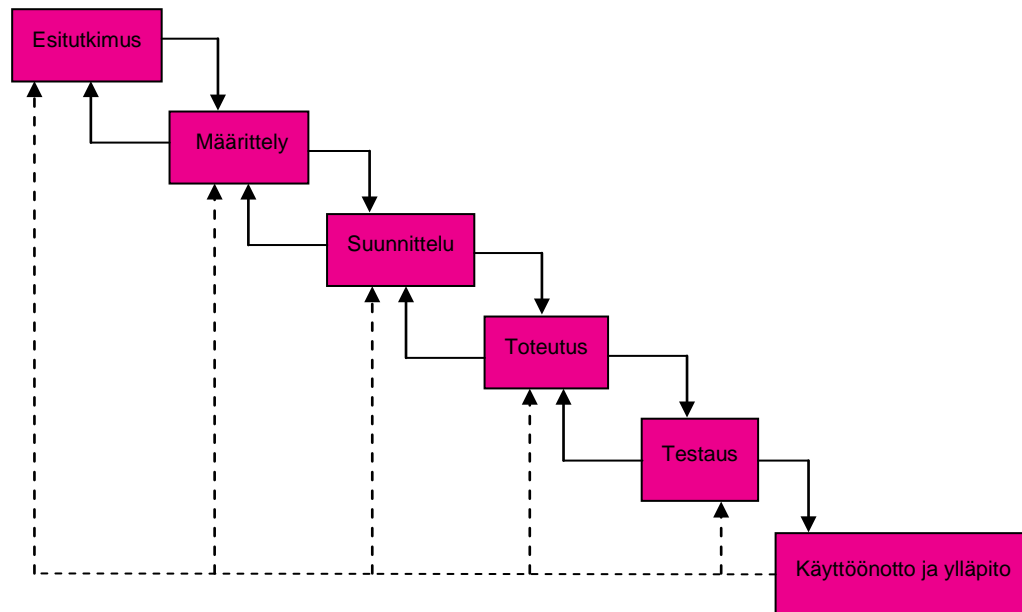
AVD

Android Virtual Device (AVD) on emulaattori, jonka voi konfiguroida vastaamaan fyysistä Android laitetta laitteistoa, ohjelmistoa ja ohjelmistorajapintaa myöten.

2 PROJEKTIN ORGANISOINTI

2.1 Projektin vaiheistus

Projektin vaiheita voidaan kuvata seuraavanlaisella kaaviolla.



Kuva 23. Projektin vaiheistus

Kuva 1:n mukaan projektin vaiheistus määritellään seuraavasti:

2.1.1 Esitutkimus

Esitutkimus koostuu pääosin tutkimisesta, kuinka hyvin jo olemassa olevan iOS version pohjalta kaikki toiminnallisuus ja käyttäjäystävällisyys saadaan toteutettua android ympäristössä. Ennen määrittelyä täytyy myös tutustua jo valmiina olevaan JSON rajapintaan ja sovelluksen ulkoasuun ja sen eri osiin.

2.1.2 Määrittely

Määrittelyvaiheessa määritellään toteutettavat toiminnallisuudet. Sovelluksen määrittelyä ohjaa vahvasti iOS version toiminnallisuus. Android sovelluksesta on tarkoitus tehdä mahdollisimman yhdenmukainen iOS version kanssa, kuitenkin mukailen Androidin käyttökokemusta ja käyttöjärjestelmän ominaisratkaisuja esim. käyttöliittymäkomponenteissa ja valikoissa.

2.1.3 Suunnittelu

Suunnitteluvaiheessa mietitään, kuinka käyttöliittymän eri näkymät kannattaa suunnitella, jotta skaalautuvuus ja käyttökokemus olisi laitteesta riippumatta sama.

2.1.4 Toteutus

Suunnitteluvaiheen jälkeen sovellus toteutetaan Eclipse ohjelmointiympäristössä. Toteutuksen aikana sovelluksen ajamiseen ja testaukseen käytetään virtuaalisia android laitteita (AVD) sekä Android

4.1.2 ja Android 2.3.5 versioilla olevia fyysisiä laitteita. Tämä helpottaa reaaliaikaisen näkemyksen siitä, miten sovellus milläkin Android alustalla käyttäytyy ja skaalautuu.

2.1.5 Testaus ja mahdolliset korjaukset

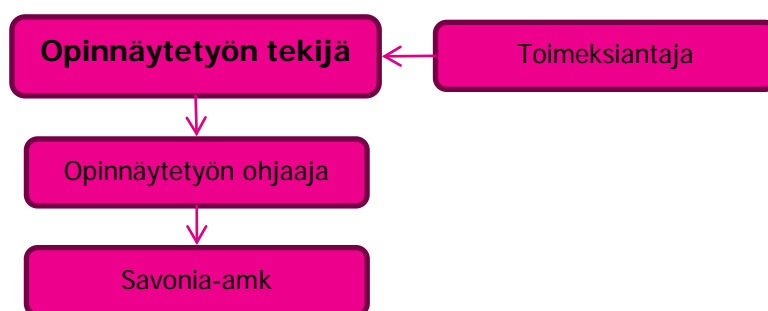
Sovellusta testataan jatkuvasti toiminnallisuus kerrallaan. Lopuksi suoritetaan sovelluksen kokonaisvaltainen testaus mm. Google Play sovelluskaupan kriteerien mukaisesti ja korjataan mahdollisesti esiintyneet virheet ja puutteelliset toiminnot.

2.1.6 Käyttöönotto ja ylläpito

Kun projekti on valmis ja hyväksytty, laitetaan sovellus jakeluun Googlen Play sovelluskauppaan. Sovellus siirtyy toimeksiantajalle projektin päätteeksi ja jatkokehityksestä sekä ylläpidosta sovitaan erikseen jälkikäteen.

2.2 Organisaation rakenne

Organisaatiota voidaan kuvata seuraavalla kaaviolla:



Kuva 24. Organisaation rakenne.

Kuva 2:n esittämällä tavalla projektista vastuussa on opinnäytetyön tekijä. Opinnäytetyön tekijä on yhteydessä toimeksiantajaan. Opinnäytetyön ohjaaja pidetään tietoisena työn etenemisestä.

2.3 Sidosryhmien kuvaus

Toimeksiantajana toimii Tmi Mikko Hopia, kuopionkirppari.fi-sivuston luoja ja ylläpitäjä. Opinnäytetyön ohjaajana toimii Savonia-ammattikorkeakoulun lehtori Jussi Koistinen.

2.4 Vastuuhenkilöt

Opinnäytetyön toteutuksesta kokonaisuudessaan vastaa opinnäytetyön tekijä. Projektin hyväksynnästä vastaavat toimeksiantaja sekä opinnäytetyön ohjaaja.

3 PROJEKTIN OHJAAMINEN

3.1 Tavoitteet ja priorisointi

Projektin tavoitteena on toteuttaa tuotannollinen android sovellus laitettavaksi yleiseen jakeluun. Sovelluksen reunaehtojen täyttymisen jälkeen suurimpana prioriteettina on sovelluksen helppokäyttöisyys ja hyvä käyttökokemus.

3.2 Riskien hallinta

Projektin toteutuksen kannalta riskejä on hyvin vähän. Mahdolliset riskit kuvattu alla:

Riski	Hallintakeinot	Prioriteetti	Todennäköisyys (1-5)
Tekniset ongelmat	Ei rikota laitteita. Suunnitelmaan projektin vaiheet huolellisesti ja varmistetaan käytettävien tekniikoiden soveltuvuus etukäteen.	2	2
Epärealistinen aikataulu	Ei aseteta valmistumisajankohtaa liian aikaiseksi.	1	3
Muut työt	Keskitytään projektiin, eikä oteta muita yhtäaikaista töitä.	3	2

Kaavio 1. Riskit ja hallintakeinot.

3.3 Seuranta ja ohjaus

Projektin edistymisestä tiedotetaan toimeksiantajaa viikon välein, ja opinnäytetyön ohjaaja 2 kertaa kuussa. Tarvittaessa useammin. Edistystä tarkastellaan palaverissa kerran kuussa. Seurannan tarkoituksena on huomata mahdolliset virheet ja suunnitelmasta poikkeavuudet, ennen kuin niistä muodostuu ongelmia. Seurannassa voidaan myös sopia projektin muutoksista tarvittaessa.

3.4 Henkilöresurssien käytön suunnittelu

Opinnäytetyön tekemisestä vastaa yksin opinnäytetyön tekijä, toimeksiantajan avustuksella. Opinnäytetyötä pyritään tekemään mahdollisuuksien mukaan täysipäiväisesti.

4 TEKNIikka

4.1 Menetelmät ja työkalut

Työn dokumentoimiseen käytetään Microsoft Wordia ja muita Microsoft Office –paketin työkaluja. Sovelluksen toteutuksessa käytetään Eclipseä, Adobe Photoshopia sekä muita tarvittavia työkaluja. Sovelluksen ajamiseen käytetään Android virtuaalilaitteita, HTC Desire HD –puhelinta, jossa on androidin versio 2.3.5 sekä Samsung Galaxy Note 10.1 –taulutietokonetta, jossa on androidin versio 4.1.2. Opinnäytetyön tekijän jaksamiseen käytetään työkaluna Philipsin kahvinkeitintä.

4.2 Dokumentointi

Opinnäytetyöstä kirjoitetaan laaja raportti, joka käy läpi työvaiheet ja menetelmät. Raportista jätetään pois kaikki arkaluontoinen tieto, jota toimeksiantaja ei halua yleiseen jakeluun.

4.3 Laadunvarmistus

Kuukausittaiset palaverit ja viikoittaiset tiedottamiset projektin edetessä varmistavat, että projekti on oikeilla raiteilla ja menossa oikeaan suuntaan. Lopullinen ohjelma testataan perusteellisesti ja korjataan virheet ennen toimeksiantajalle luovutusta.

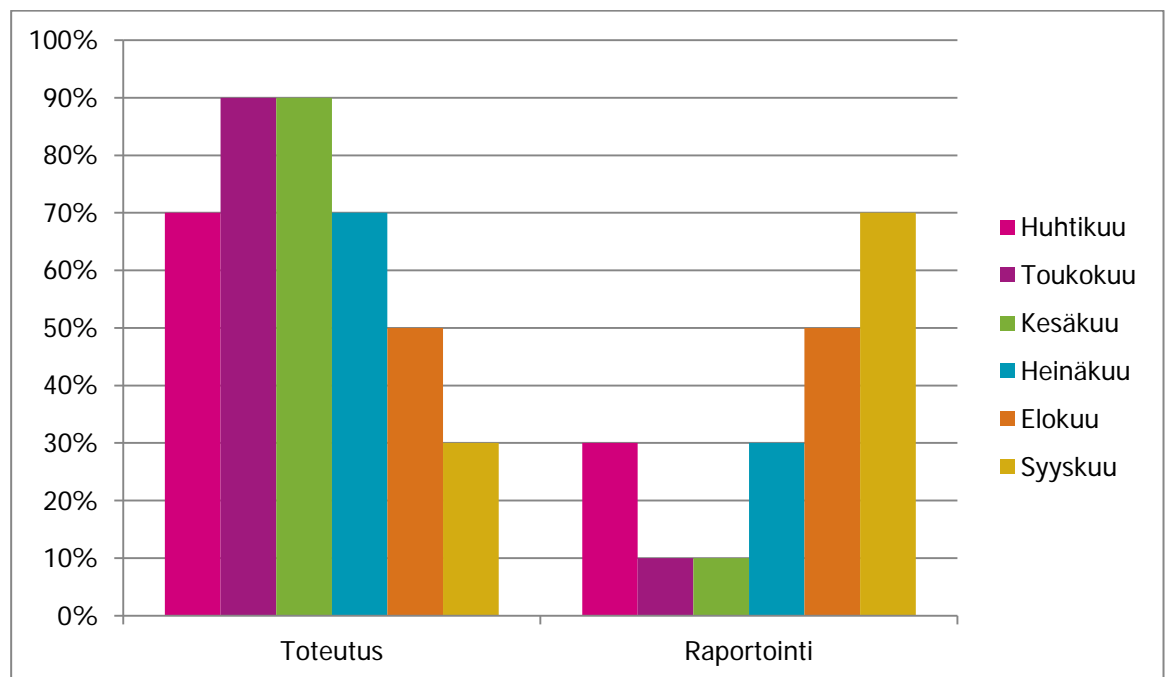
5 VAIHEET, AIKATAULUT JA BUDJETTI

5.1 Projektin osittaminen

Projekti ositetaan pääosin kuten mainittu kohdassa 2.1 (Projektin vaiheistus). Toteutus ositetaan siten, että toteutetaan yksi näkymä kerrallaan. Kun näkymä ja sen toiminnallisuus on toteutettu, siirrytään seuraavaan näkymään. Testausta toteutetaan lomittain toteutuksen kanssa. Lopuksi sovellus testataan kaikkien virheiden varalta ja kirjoitetaan raportti projektista.

5.2 Resurssien käyttö ajan funktiona

Resurssien käyttö jakautuu suunnitellusti seuraavasi:



Kaavio 2. Resurssien käyttö kuukausittain.

Kaavio 1 näyttää kuukausittain käytetyt resurssit toteutuksen ja raportoinnin osalta. Alussa tehdään projektisuunnitelma ja sen saatua hyväksynnän, aletaan toteuttaa sovellusta. Toteutusta tehdään touko- ja kesäkuussa mahdollisuuksien mukaan täysipäiväisesti ja heinäkuusta syyskuuhun raportoinnin osuus kasvaa työmäärästä.

5.3 Budjetti ja resurssien allokointi

Työ toteutetaan opinnäytetyönä josta annetaan stipendi työn valmistuttua. Työn vaiheet ja työssä käytettävän laitteiston kustannukset ovat 0,00€.

5.4 Aikataulu

Opinnäytetyön alustava aikataulu on seuraavanlainen:

Työtehtävä	Aloituspäivä	Lopetuspäivä
Esitutkimus	1.3.2013	1.4.2013
Määrittely	27.3.2013	5.4.2013
Suunnittelu	10.4.2013	21.4.2013
Toteutus	22.4.2013	15.7.2013
Testaus	22.4.2013	31.7.2013
Käyttöönotto	1.8.2013	
Raportointi	1.4.2013	31.9.2013

Kaavio 3. Opinnäytetyön aikataulu.