Qt Quick-kehitys Maemo-alustalle

Lauri Vastamäki

Opinnäytetyö
Marraskuu 2011
Tietotekniikan koulutusohjelma
Ohjelmistotekniikka
Tampereen ammattikorkeakoulu

TAMPEREEN AMMATTIKORKEAKOULU
Tampere University of Applied Sciences

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tietotekniikan koulutusohjelma
Ohjelmistotekniikan suuntautumisvaihtoehto

VASTAMÄKI, LAURI: Qt Quick-kehitys Maemo-alustalle
Opinnäytetyö 39 s.
Marraskuu 2011

Tässä työssä käsitellään Qt Quick -sovellus kehitystä Maemo alustalle. Työn tarkoitus oli perehtyä Qt Quick -sovelluskehitykseen sekä Maemo-alustaan.

Tässä työssä käsitellään Qt Quick -sovelluskehityksessä huomioon otettavia asioita kehitettäessä Maemo-alustalle sekä Qt Quick -sovelluskehitystä yleisesti. Tässä työssä käsitellään myös olemassa olevia työkaluja kehityksen helpottamiseksi ja nopeuttamiseksi käsitellään asennuksesta konfigurointiin.

Työn sisällön avulla aiheeseen tarkemmin perehtymättömän lukijan toivotaan saavan riittävät taidot Qt Quick -sovelluskehitykseen. Työn lukijalta odotetaan hyviä pohjatietoja Qt-sovelluskehityksestä.

Työssä myös tuodaan esiin ongelmia, jotka tulivat esiin sovelluskehityksen yhteydessä sekä mahdollisia ratkaisuja näihin.

Työn lopussa esitellään myös kokonaisuuden innoittanut Qt Quick -sovellus, Indite, jossa yhdistyy Twitter ja Facebook samaan käyttöliittymään.

Asiasanat: Maemo, Mobiilikehitys, QML, Qt, Qt Quick

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Degree Programme in Information Technology
Option of Software Engineering

VASTAMÄKI, LAURI: Qt Quick Development for the Maemo platform

Bachelor's thesis 39 pages
November 2011

This thesis covers the basic knowledge required to develop a Qt Quick application for the Maemo platform. The motivation behind this thesis was to learn more about Qt Quick programming and the Maemo platform.

The required software and other tools are explained from the install process all the way to configuring. The specific things needed to know about developing for the Maemo platform are introduced and explained. The general basics in Qt Quick development are also explained.

The thesis also brings forth some problems faced in the development and some possible solutions to these problems.

The Qt Quick application, Indite, that inspired this thesis is showcased at the end of the thesis. Indite is a social media smashup application that combines Twitter and Facebook under the same user interface for easy access and usability.

The thesis expects a good basic knowledge in Qt-development from the reader. If the reader has this knowledge, he or she should get a good introduction to the subject by reading this thesis.

Keywords: Maemo, Mobile development, QML, Qt, Qt Quick

Sisällysluettelo

TERMIT JA LYHENTEET

API	Application Programming Interface, käyttöliittymä ohjelman ja käyttöjärjestelmän väliseen kommunikointiin.
D-bus-väylä	Nokia N900-laitteesta löytyvä kommunikointiväylä laitteiston käyttöön.
Graph API	Facebookin käytön mahdollistava käyttöliittymä.
Maemo	Nokia Oyj:n kehittämä älypuhelinkäyttöjärjestelmä joka pohjautuu Linuxiin.
QML	Deklaratiivinen käyttöliittymien luomiseen suunniteltu ohjelmointikieli, perustuu JavaScriptiin.
Qt	Kehitysympäristö alustariippumattomien sovellusten kehittämiseen.
Qt Quick	Tehokkaiden ja näyttävien sovellusten luomiseen tarkoitettu Qt:n laajennettu kehitysympäristö, jossa yhdistyy Qt ja QML.
Qt SDK	Kaikki mitä tarvitaan sovelluskehitykseen Qt Quick-kehitysympäristössä.
QtweetLib	Avoimen lähdekoodin kirjasto joka mahdollistaa Twitterin käytön.
RPC	Remote Process Communication, kommunikointi protokolla jota käytetään D-bus-väylässä.
signal-slot	Qt:n käyttämä kommunikointi tapa ohjelman osien välillä.
SSH-avain	Secure Shell-avain, avain jota käytetään turvallisen etäkäytön autentikointiin.
Symbian	Nokia Oyj:n kehittämä älypuhelinkäyttöjärjestelmä.

1 JOHDANTO

Tämä työ käsittelee Qt Quick -sovelluskehitystä, pääasiassa Maemo alustalla. Työ jakautuu neljään toisistaan riippuvaan osaan. Nämä osat muodostavat kokonaisuuden, jonka avulla lukija pystyy omaksumaan Qt Quick -sovelluskehityksen perusteet.

Toisessa luvussa perehdytään Qt Quick -kehitysympäristön teknisiin ominaisuuksiin ja sovelluskehitykseen. Qt Quick on kehitysympäristö, joka tarjoaa deklaratiivisen tavan luoda uniikkeja ja dynaamisia käyttöliittymiä hienoilla siirtymillä ja tehosteilla.

Kolmannessa luvussa käydään läpi vaadittavat esivalmistelut ja tarvittavat ohjelmat sovelluskehityksen aloittamiseksi. Tämä luku käsittää muun muassa laitteelta vaadittavat ohjelmistoversiot.

Neljännessä luvussa käydään läpi työssä kohdattuja ongelmakohtia ja esitellään joitakin opittuja kikkoja. Tässä luvussa tulee esiin muun muassa laitteiston suorituskykyvaatimukset.

Viidennessä luvussa esitellään tähän työhön innoittanut, sosiaaliset mediat Facebookin ja Twitterin yhdistävä sovellus, Indite.

2 QT QUICK

Qt Quick on Nokia Oyj:n kehittämä kehitysympäristö. Qt Quick yhdistää QML- ja Qt-kielet samaan kehitysympäristöön, helpottaen näin hienon käyttöliittymän ja tehokkaan sovelluslogiikan yhdistämistä samaan sovellukseen.

Qt Quick -kehitysympäristö sisältyy uusimpiin Nokia Oyj:n älypuhelimiin vakiona, mikä luo hyvän asiakaspohjapotentiaalin. Qt Quick mahdollistaa sovelluksen kehittämisen yhtä aikaa Maemolle ja Symbianille. Tässä tulee kuitenkin vastaan eräitä ongelmia, joista enemmän luvussa neljä.

Qt Quick on kokonaisuudessaan erittäin tehokas kehitysympäristö, jonka avulla on mahdollista luoda nopeasti näyttäviä ja tehokkaita sovelluksia. Qt Quick on saatavilla monille alustoille, minkä ansiosta kehittäjä voi säästää aikaa kehittäessään sovelluksia useammille alustoille.

2.1 QML

QML on JavaScriptiin pohjautuva, deklaratiivinen käyttöliittymän toteutukseen suunniteltu kieli (Qt 4.7: Qt Declarative UI Runtime). QML mahdollistaa hyvin nopean prototyyppien toteuttamisen, tätä on selvitetty Qt 4.7: Qt Declarative UI Runtime lähteessä. QML sisältää animaatioita ja tilasiirtymiä varten valmiita funktioita joiden avulla pystyy helposti luomaan näyttäviä käyttöliittymiä. QML-kielellä luodut käyttöliittymät myös näyttävät samalta niin Maemolla kuin Symbianillakin.

```
import QtQuick 1.0

 Rectangle {
     id: canvas
     width: 200
     height: 200
     color: "blue"

     Image {
         id: logo
         source: "pics/logo.png"
         anchors.centerIn: parent
         x: canvas.height / 5
     }
 }
```

Koodiesimerkki 1. QML-kielen käytön esimerkki.

QML-kielen syntaksi on selkeää ja helposti opittavaa (Koodiesimerkki 1). Koodissa näkyvä "id: canvas"-rivi on tärkeä QML-kielessä. Siinä annetaan objektille uniikki tunniste, jonka avulla kyseistä objektia on mahdollista kutsua muista objekteista. Tämä mahdollistaa muun muassa hienoja tehosteita sisältävät käyttöliittymät.

QML-kieli on deklaratiivinen, eli se kertoo suorittavalle laitteelle mitä tehdään eikä tarkasti miten tehdään. Tämä tulee parhaiten esiin puhtaasti QML-kielellä toteutetussa ohjelmassa. Puhdasta QML-sovellusta ei käännetä ajettavaksi ohjelmaksi vaan koodi suoritetaan QML-viewer ohjelmassa, joka on saatavilla monille alustoille, muun muassa Maemolle. Qt Quick -sovelluksen tapauksessa QML-koodi suoritetaan Qdeclarative-View-luokassa.

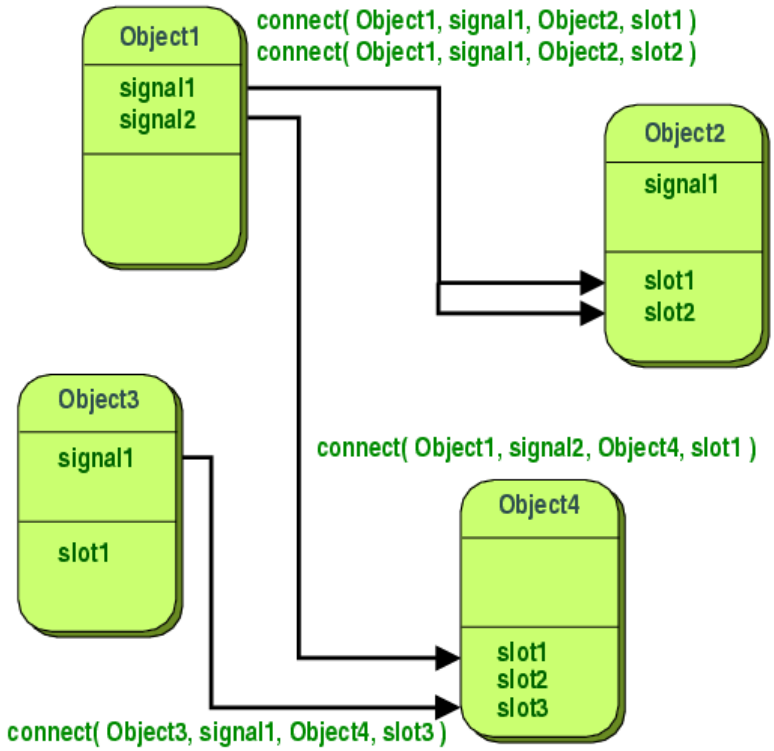
QML-koodi siis kertoo ohjelmalle mitä ohjelman luoja haluaa käyttäjän näkevän ja miten siirrytään tilasta toiseen, vaikka päänäkymästä asetuksiin tai tietoja ohjelmasta-ikkunaan. Kehitysvaiheessa on myös mahdollista asettaa käyttöliittymä käyttämään tiettyä teemaa alustasta riippumatta. Tämä siis mahdollistaa uniikin ja yhdenmukaisen käyttäjäkokemuksen.

2.2 Qt

Qt on alunperin Trolltech nimisen yrityksen luoma ohjelmointikieli, Nokia Oyj osti myöhemmin Trolltechin, jolloin Qt siirtyi Nokian omistukseen. Qt käyttää C++:aa pohjanaan, laajentaen sen toiminnallisuutta. Qt toimii useilla alustoilla, mikä lisää sen käytännöllisyyttä ohjelmistokehittäjän näkökulmasta.

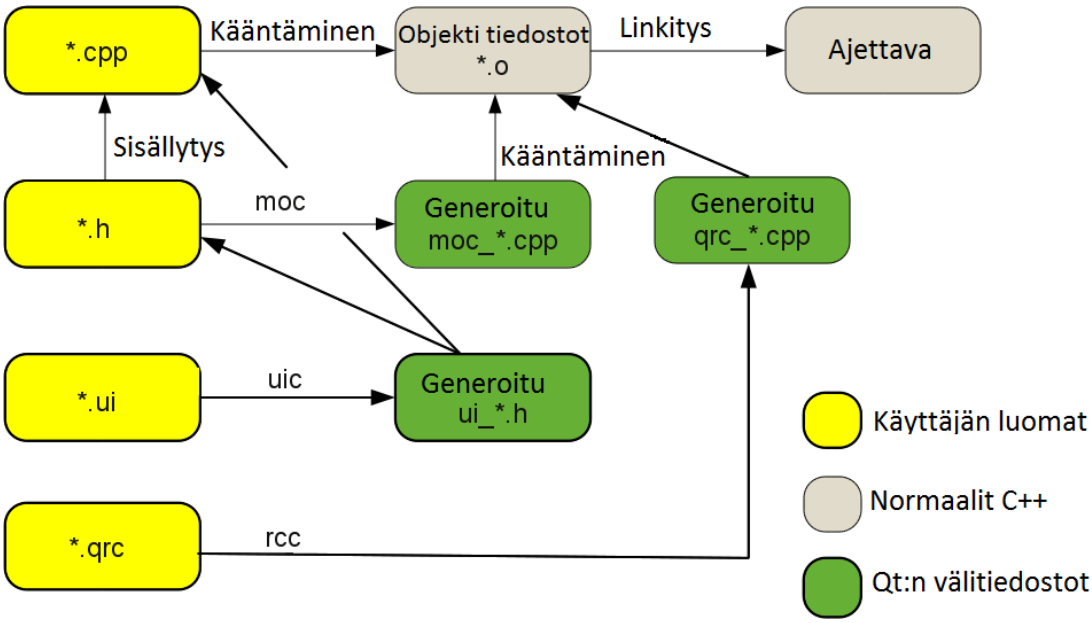
Qt:n parhaita lisäyksiä C++-kieleen on SIGNAL-SLOT kommunikointi (Qt 4.7: Signals & Slots). Ohjelman oliot voivat käyttää signaaleja kommunikoidessaan kuten Qt 4.7: Signals & Slots lähde selvittää. Tällainen signaali lähetetään oliosta ohjelman sisäisen tilan muuttuessa ennalta määritetyllä tavalla, tällainen muutos voi olla vaikka ohjelman sulkemisnapin painaminen. Tässä tapauksessa siis käyttäjä haluaa sulkea ohjelmansa, joten lähetetään esimerkiksi signaali exit. Tällöin halutaan että ohjelman exit()-funktiota kutsutaan. Jos ohjelman pääluokkaan on luotu tarvittava slot (Kuva 1), voidaan yhdistää exit-signaalin ja kyseinen slot. Signaali on mahdollista kytkeä niin moneen slotiin kuin halutaan. Se on mahdollista kytkeä myös toiseen signaaliin, jolloin tämä toinen signaali lähetetään heti ensimmäisen lähetyksen jälkeen.

Slot on normaali C++-funktio, jota voi kutsua perinteisesti, mutta jonka erikoisuus on mahdollisuus liittää siihen signaali mistä tahansa muusta luokasta. Slotiin on mahdollista yhdistää useampikin signaali useammasta luokasta. Koska slotit käyttäytyvät kuin tavalliset funktiot, ne noudattavat C++-kielen normaaleja sääntöjä kun niitä kutsutaan suoraan. Koska slotia voi kutsua mistä tahansa komponentista, saattaa jonkin ohjelman osan lähettämä signaali kulkeutua johonkin ei haluttuun komponenttiin. Tämä taas saattaa aiheuttaa ei toivottua käytöstä, josta saattaa aiheutua jopa sovelluksen kaatuminen.



Kuva 1. Signal-Slot kommunikointikaavio (Qt 4.7: Signals & Slots).

Ohjelman kääntämisvaiheessa kutsutaan Meta-Object Compiler-, eli moc-työkalua (Qt 4.7: Using the Meta-Object Compiler (moc)). Kuten Qt 4.7: Using the Meta-Object Compiler (moc) lähde selivittää, moc-työkalu lukee C++-otsikkotiedostot läpi etsien Q_OBJECT-makroja. Jos työkalu löytää Q_OBJECT-makron yhdestä tai useammasta luokasta, se luo C++-lähdekooditiedoston, joka sisältää meta-objekti koodin kyseisille luokille. Luotu C++-koodi on käännettävä ja linkitettävä luokan implementoinnin yhteydessä (Kuva 2).



Kuva 2. Qt-ohjelman kääntäminen (Using CMake to Build Qt Projects | Qt Developer Network - suomennettu).

Useammalle alustalle helposti syntyvä Qt-sovellus on kilpailuvaltti. Periaatteessa on mahdollista tehdä ohjelma kerran ja kääntää se sitten useammalle alustalle ilman minkäänlaista refaktorointia. Käytännössä kuitenkin ohjelman toimintaan saattaminen useammalla alustalla vaatii jonkin verran alustakohtaista optimointia ja alustan mahdollisten uniikkien ominaisuuksien hyötykäyttöä, jos niillä päästään esimerkiksi nopeampaan suoritukseen tai sulavampaan käyttöön.

2.3 QDeclarativeView

QDeclarativeView on luokka, joka liittää QML-käyttöliittymän Qt-koodiin. QdeclarativeView:lle annetaan aloitusnäkymäksi haluttu QML-tiedosto. Tämä on näkymä, joka näkyy käyttäjälle ohjelman aloituksen yhteydessä.

```
#include <QApplication>
 #include <QDeclarativeView>

 int main(int argc, char *argv[])
 {
     QApplication app(argc, argv);

     QDeclarativeView view;

view.setSource(QUrl::fromLocalFile("application.qml"));
     view.show();

     return app.exec();
 }
```

Koodiesimerkki 2. QDeclarativeView:n käyttöönotto.

QDeclarativeView:lle asetetaan QML-tiedosto, tässä tapauksessa application.qml Qt-ohjelman main-luokassa (Koodiesimerkki 2). Nyt ohjelmassa käyttäjälle avautuva aloitusnäkymä on juuri application.qml-tiedostossa määritetyn näköinen.

2.4 Qt Mobility API

Qt:n mobiilikehittäjien työn helpottamiseksi on kehitetty Qt Mobility API:t (Qt Mobility 1.1). Nämä API:t mahdollistavat älypuhelinten ominaisuuksien helpon hyödyntämisen kuten Qt Mobility 1.1 lähteestä nähdään. Tällaisia ominaisuuksia ovat esimerkiksi GPS ja asennon tunnistimet. Kun laitteelle on asennettu Mobility API:t voidaan sovelluksen koodissa kutsua valmiilla kutsuilla eri osia alustan ominaisuuksista. Esimerkiksi GPS-sijainnin selvitys tai laitteen asennon tunnistaminen onnistuu.

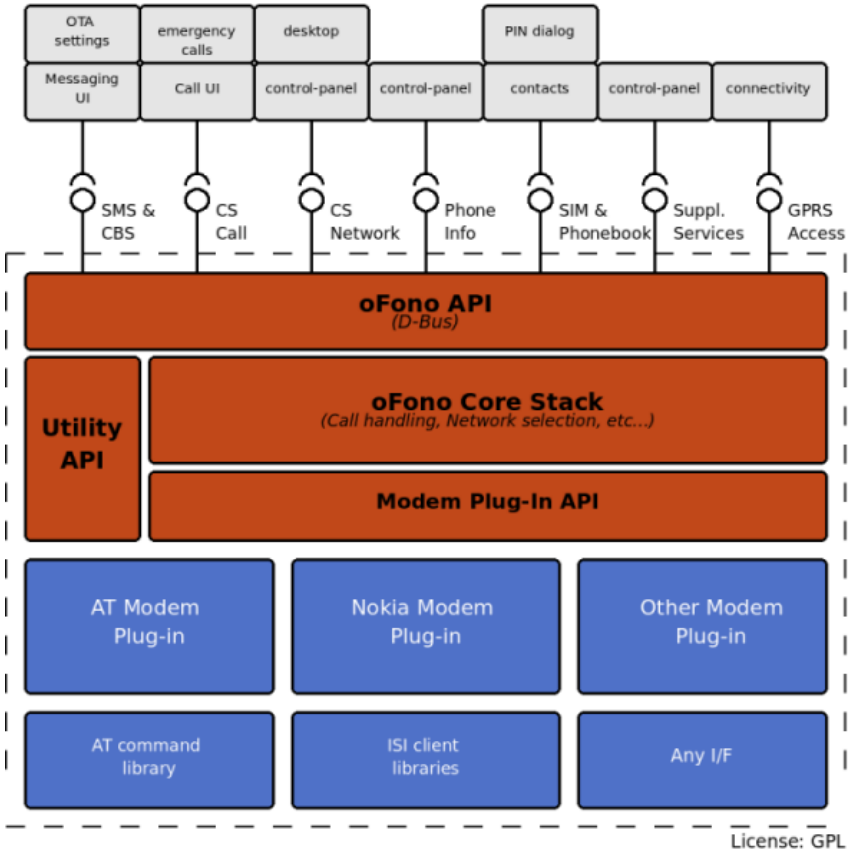
Mobility API:t ovat jatkuvasti kehittyvä osa Qt-kehitysympäristöä. Mobility API:t ovat erillinen osa Qt-kehitysympäristöä, eli ne voi päivittää erillään muusta Qt-kehitysympäristöstä ja näin kannattaa tehdä.

2.5 Maemo-alustan erikoisuudet

Maemo-alustalla sovelluskehittäjältä vaaditaan pelkän Qt Quick hallinnan lisäksi sellaista tietoa ja taitoa alustan uniikeista ominaisuuksista. Oman tapansa toimia vaativat esimerkiksi äänentoisto ja kommunikointi joidenkin alustan ominaisuuksien kanssa.

2.5.1 D-bus

D-bus-väylää käytetään järjestelmän tiedotteiden vastaanottamiseen, esimerkiksi virta vähissä ja järjestelmän sammutus. Myös viesti sovelluksen pakollisesta sammumisesta kulkee tätä väylää pitkin (maemo.org - DBusGuide). D-bus siis toimii välikerroksena laitteiston ja ohjelmiston välissä (Kuva 3).



Kuva 3. D-bus-väylän sijainti järjestelmähierarkiassa (Documentation | oFono).

Kaikki sovellukset Maemo-alustalla täytyy alustaa oikein käynnistyksen yhteydessä, mikä tapahtuu D-bus-väylää käyttämällä. Jos alustusta ei ole tehty oikein on yksi merkki siitä sovelluksen käynnistyminen mutta välitön sulkeutuminen. Alustaminen tapahtuu D-bus-session osso_initialize()-funktiota kutsumalla kuten maemo.org - DBusGuide lähteestä selviää. Tämä tulee tehdä vain kerran sovelluksessa (Koodiesimerkki 3).

```
#include <libosso.h>

int main(int argc, char* argv[])
{
  osso_context_t* osso_context = osso_initialize("example", "0.0.1", FALSE, NULL);
  if(!osso_context)
  {
    printf("osso_initialize() failed\n");
    return OSSO_ERROR;
  }
...
```

Koodiesimerkki 3. osso_initialize()-funktion kutsuminen.

Sovellukset voivat kommunikoida keskenään käyttämällä D-bus-väylää. Tämä tarkoittaa mahdollisuutta lähettää viestejä prosessista toiseen, ja keskustelu tapahtuu Remote Process Communication (RPC)-metodilla.

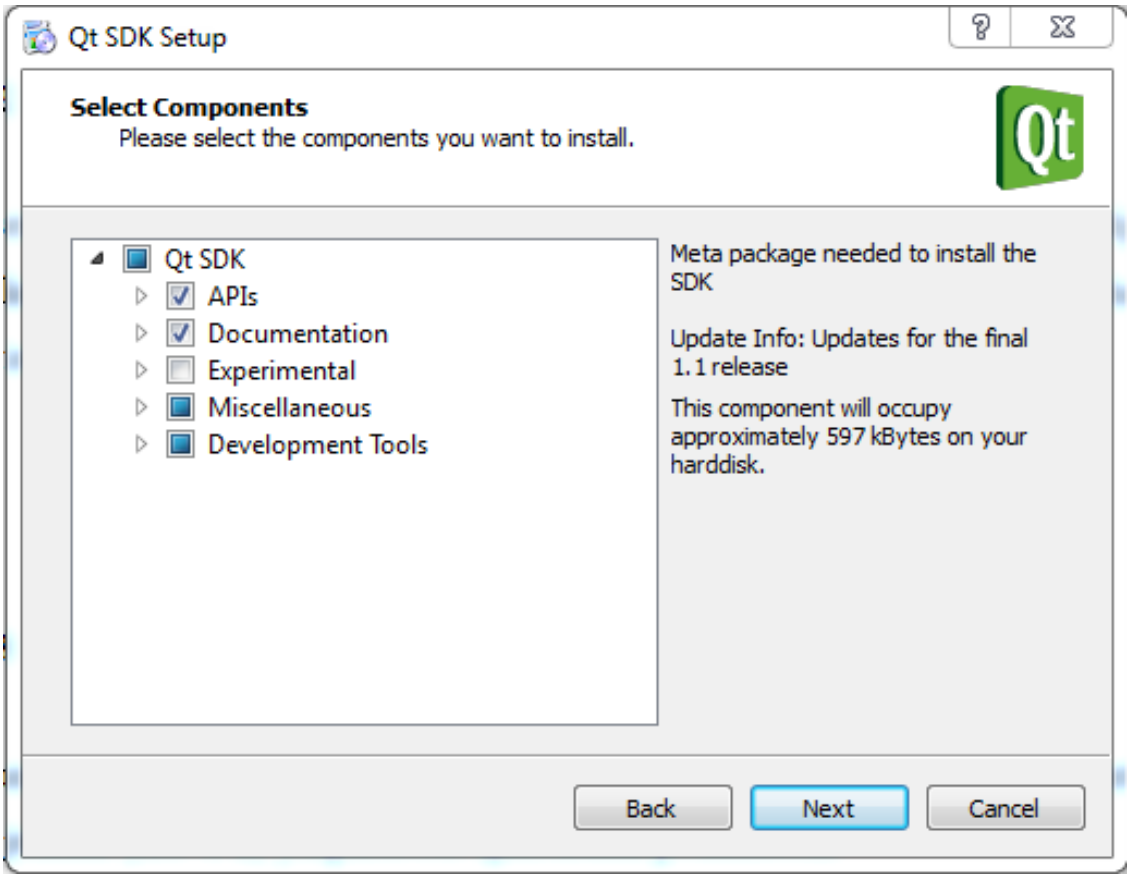
3 VALMISTELUT

3.1 Yleistä

Helpoimmin sovelluskehityksen alkuun pääsee asentamalla uusimman version Nokia Qt SDK:sta. Tämän työn tekemistä inspiroinut ohjelma, Indite-ohjelma, on tehty käyttämällä kyseisen SDK:n tech preview-versiota. Nykyään saatavana on jo virallinen versio. Nokia Qt SDK sisältää kaiken tarvittavan Qt Quick -kehittämisen aloittamiseen. Kun kehittäjä on asentanut SDK:n, voi hän kääntää ohjelmansa niin Maemolle kuin Symbianille. Kehitystyötä helpottaa Mad Developer -ohjelman asennus N900-laitteeseen. Mad Developer mahdollistaa ohjelman suoran asentamisen Qt Creatorista käsin. Jos kehittäjä luo Qt Creatorissa SSH-avaimen ja julkaisee sen laitteeseen, voi hän tämän jälkeen asentaa pakettinsa helposti ja vaivattomasti laitteeseen, vaikka se olisi jossain muualla, kunhan wlan-asetukset ovat kunnossa.

3.2 Asennus

Asennus kannattaa suorittaa lataamalla Nokia Qt SDK Online Installer-ohjelma, joka hakee uusimmat saatavilla olevat versiot komponenteista (Kuva 4).

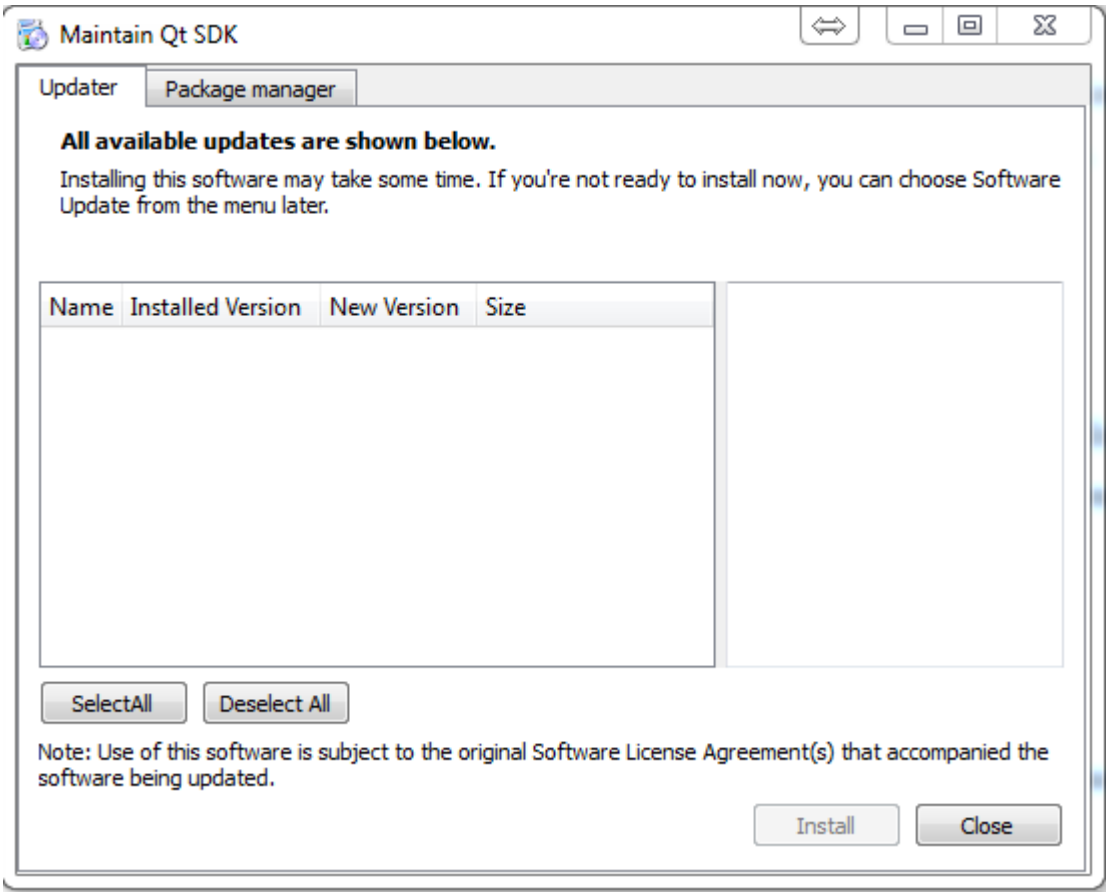


Kuva 4. Haluttujen komponenttien valinta.

Kun halutut komponentit on valittu, asennusohjelma lataa ja asentaa ne. Tämän vaiheen kesto riippuu käytettävissä olevan internetliittymän nopeudesta ja palvelimien kuormasta. Lataamiseen ja asentamiseen on siis syytä varata riittävästi aikaa.

3.3 Ylläpito

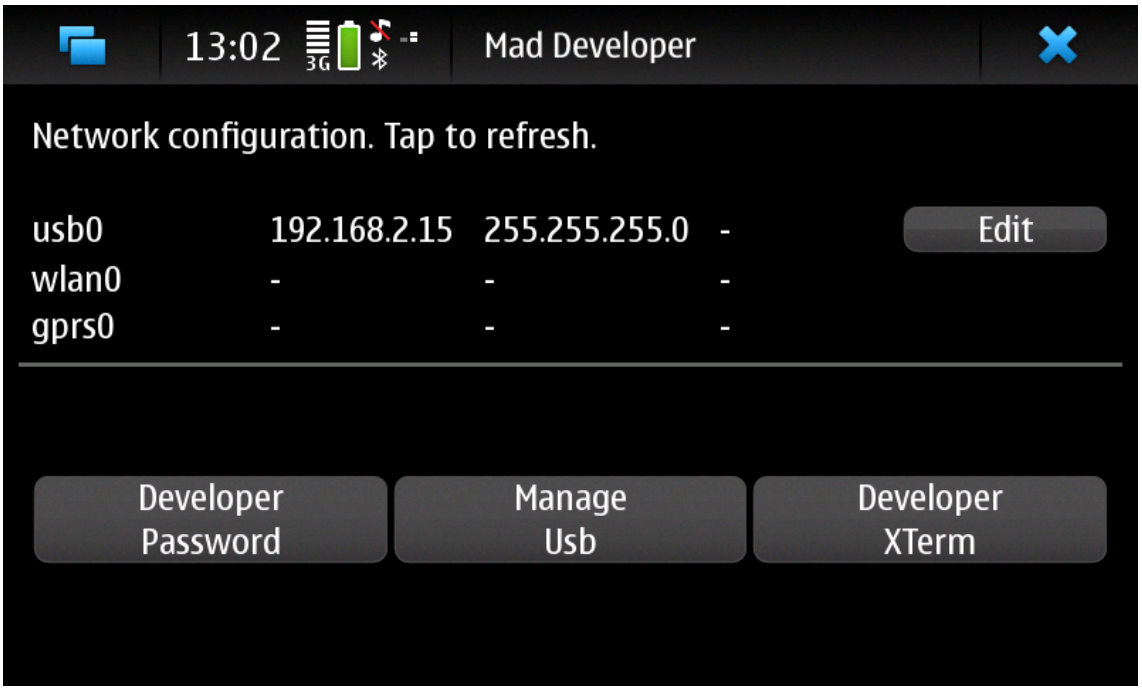
Kun SDK on asennettu, voi käyttäjä käyttää siinä mukana olevaa SDKMaintenanceTool-ohjelmaa (Kuva 5) kehitysympäristönsä ajan tasalla pitämiseen.



Kuva 5. SDKMaintenanceTool.

3.4 Mad Developer

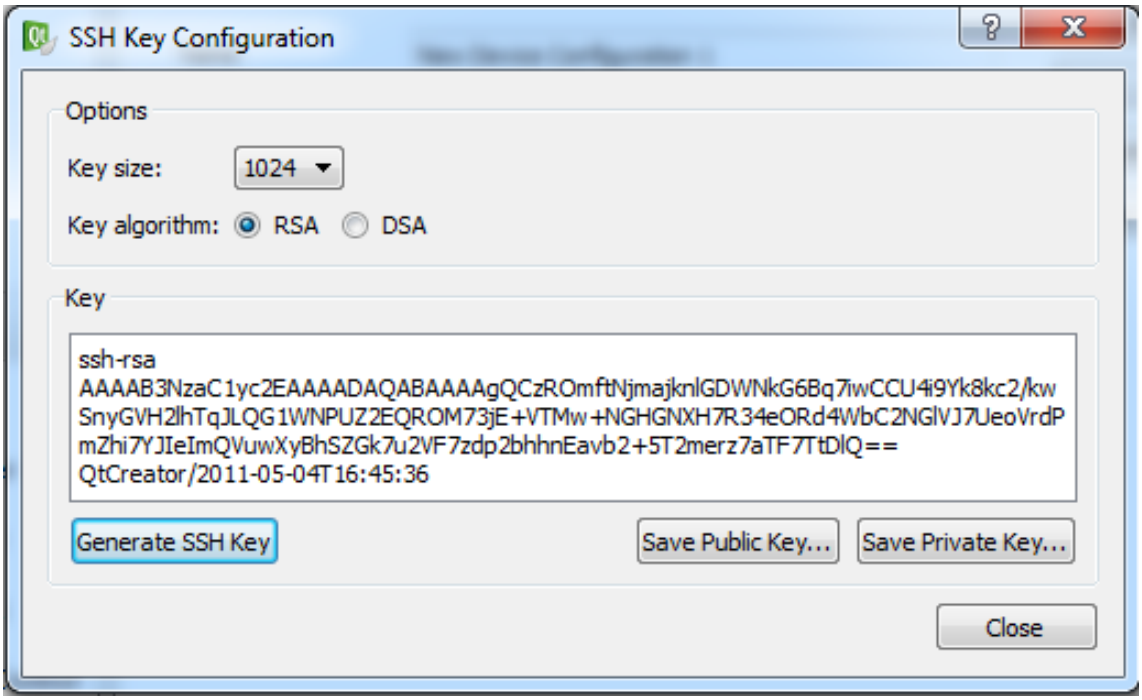
Mad Developer on ohjelma, jonka tarkoitus on helpottaa Qt-kehittäjän työtä Maemo-alustalle sovelluksia kehitettäessä. Ohjelman asennuksen jälkeen on mahdollista yhdistää tietokone ja N900-laite usealla eri tavalla, seuraavaksi esitellään wlan-yhdistäminen. Ohjelman päänäkymässä (Kuva 6) valitsemalla Developer Password, saa salasanan (Kuva 7), jonka avulla Qt Creator pääsee asentamaan laitteeseen sovelluksen.



Kuva 6. Mad Developer päänäkymä.

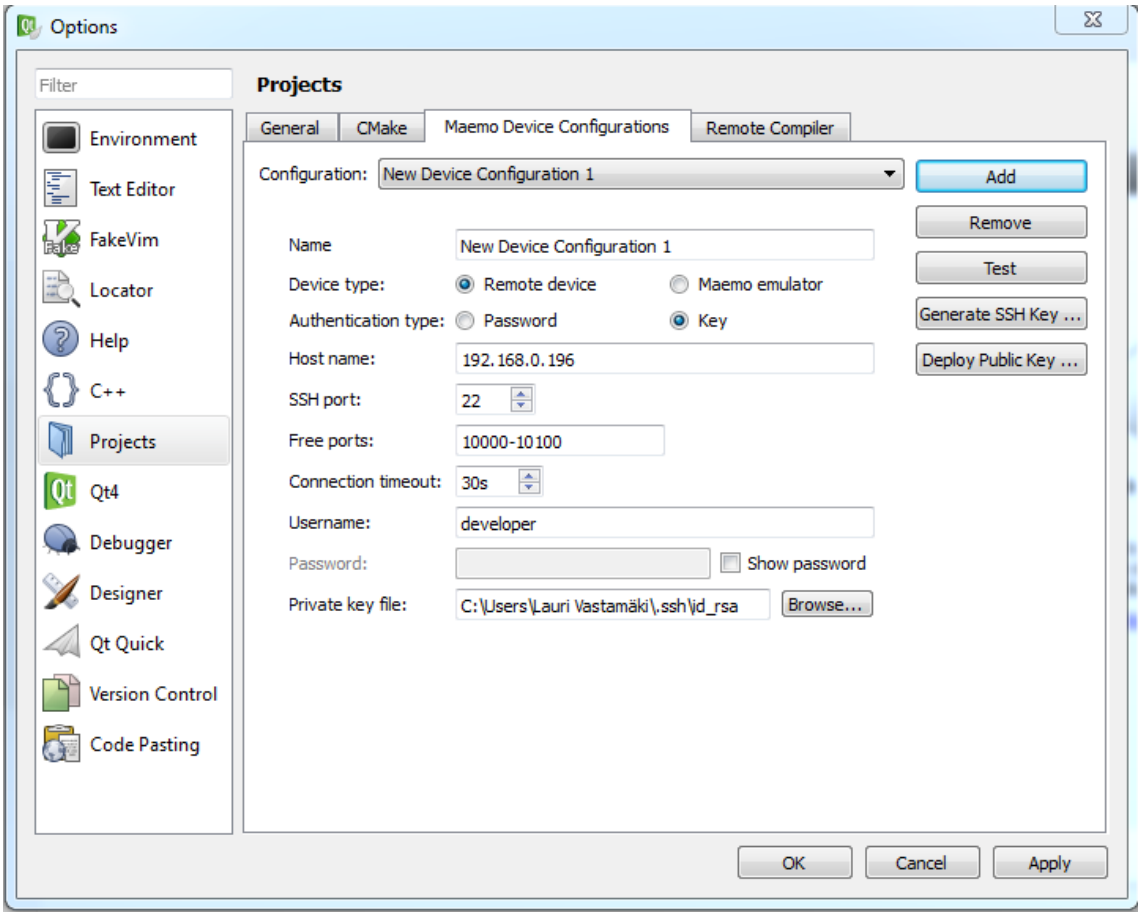
Kuva 7. Mad Developer Developer Password näkymä.

Kannattaa myös luoda Qt Creatorilla SSH-avain (Kuva 8) ja viedä se laitteeseen salasanan avulla. Kun SSH-avain on viety laitteeseen, ei enää tarvitse pitää Mad Developeria jatkuvasti käynnissä, tämä puolestaan vapauttaa resursseja.



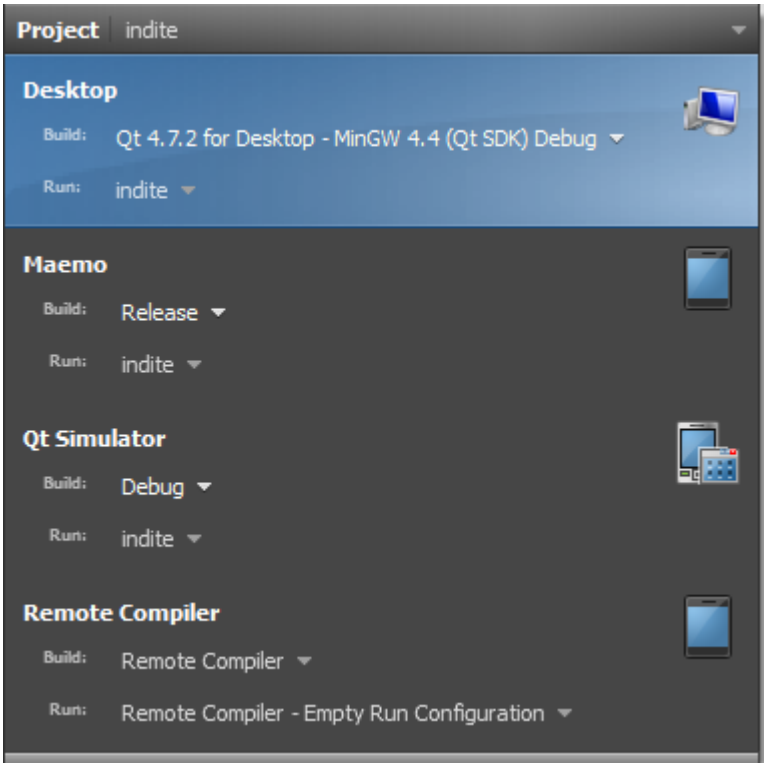
Kuva 8. SSH-avaimen luonti.

Kun sekä julkinen että yksityinen avain on luotu, viedään julkinen avain laitteeseen Deploy Public Key napilla (Kuva 9).



Kuva 9. Qt Creator etälaitteen asetukset.

Kun nämä asetukset on säätänyt kuntoon, tarvitsee enää valita ohjelmalle kyseinen laite ajokohteeksi (Kuva 10). Kun valitaan Run-valinta, Qt Creator kääntää ohjelman ja lataa asennuspaketin laitteelle sekä asentaa ja käynnistää ohjelman. Tämän vaiheen kesto riippuu kääntövaiheen kestosta sillä paketin asennus ja käynnistys toimivat nopeasti.



Kuva 10. Ajokohteen valinta.

4 ONGELMAT

4.1 Tehon tarve

Ohjelman tekemisen yhteydessä tuli esiin myös ongelmia. Suurin ongelma tässä projektissa oli Nokian N900-laitteen suhteellisen heikko teho. Tehon puutteesta johtuen muun muassa suurimmasta osasta animaatioita jouduttiin luopumaan, koska ne yksinkertaisesti näyttivät rumilta ja hidastivat huomattavasti ohjelman käyttöä.

4.2 Ohjelmointivirheet Qt:n versiossa 4.7.0

Qt:n versiossa 4.7.0 on joitakin ohjelmointivirheitä, jotka tulivat esiin ohjelman kehityksen aikana. Eniten näkynyt ongelma oli listojen esittämisessä ollut ohjelmointivirhe, jonka seurauksena tiedot tulivat näkyviin vasta rullatessa listaa. Tämä ongelma ratkaistiin suorittamalla koodissa rullausta samalla, kun luotiin lista.

4.3 Syvät QML-rakenteet

Ohjelman teossa lähdettiin liian voimakkaaseen komponenttien pilkkomiseen, minkä takia QML-rakenteista tuli liiankin syviä. Siitä johtuen signaalien välittäminen ohjelman osista toisiin oli hankalampaa kuin sen olisi tarvinnut olla. Käytännössä tämä tarkoitti suurta määrää signaaleja jotka kulkivat ohjelman tasolta toiselle välittäen informaatiota. Tällaisessa rakenteessa virheiden jäljittäminen osoittautui haastavaksi ja aikaa vieväksi työksi.

4.4 Facebookin API:t

Facebook-palvelua kehitetään jatkuvasti ja siksi sen API:tkin muuttuvat. Tämä ei sinänsä aiheutta suurempia ongelmia, elleivät niiden peruselementit muuttuisi samaan aikaan. Facebookin API:t muuttuvat useamman kerran vuodessa niin perusteellisesti, että kolmansien osapuolien ohjelmat lopettavat toimintansa. Mobiilikehittäjien työtä hidastava seikka on selaimesta riippumattoman autentikoinnin puute. Myös Inditessä on autenti-

kointia varten jouduttu luomaan ohjelmaan sisään ikkuna, jossa näkyy Facebookin internetkirjautumissivu jolla käyttäjä kirjautuu sisään Facebook-profiiliinsa ja antaa ohjelmalle luvan käyttää tietojaan ja julkaista sisältöä profiiliin. Ohjelmassa tästä saatava autentikointiavain tallennetaan, minkä jälkeen käyttäjän ei tarvitse enää syöttää tietojaan uudestaan, ellei turvallisuussyistä halua.

5 INDITE

5.1 Yleistä

Indite on sosiaalisen median käyttöön tehty ohjelma, jolla voi käyttää Facebookia ja Twitteriä samasta käyttöliittymästä. Tarkoituksena oli luoda helposti käytettävä ja monipuolinen sosiaalisen median asiakasohjelma, jolla käyttäjä voi käyttää sekä Facebookia että Twitteriä yhtä aikaa tarvitsematta siihen useampia ohjelmia ja niiden käytön vuorottelua. Myös palveluiden uniikit ominaisuudet oli tarkoitus sisällyttää toimivina ohjelmaan. Inditen luonnin alkuvaiheessa keskityttiin Twitteriin, koska siihen löytyi avoimen lähdekoodin kirjasto QTweetLib, jolla palvelun API:n käyttäminen oli helppoa ja mukavaa. Myös Facebookin Graph API:n käyttöön löytyi lopulta avoimen lähdekoodin kirjasto. Tämä kirjasto oli melko tuore johtuen Facebookin jatkuvasta uudistumisesta.

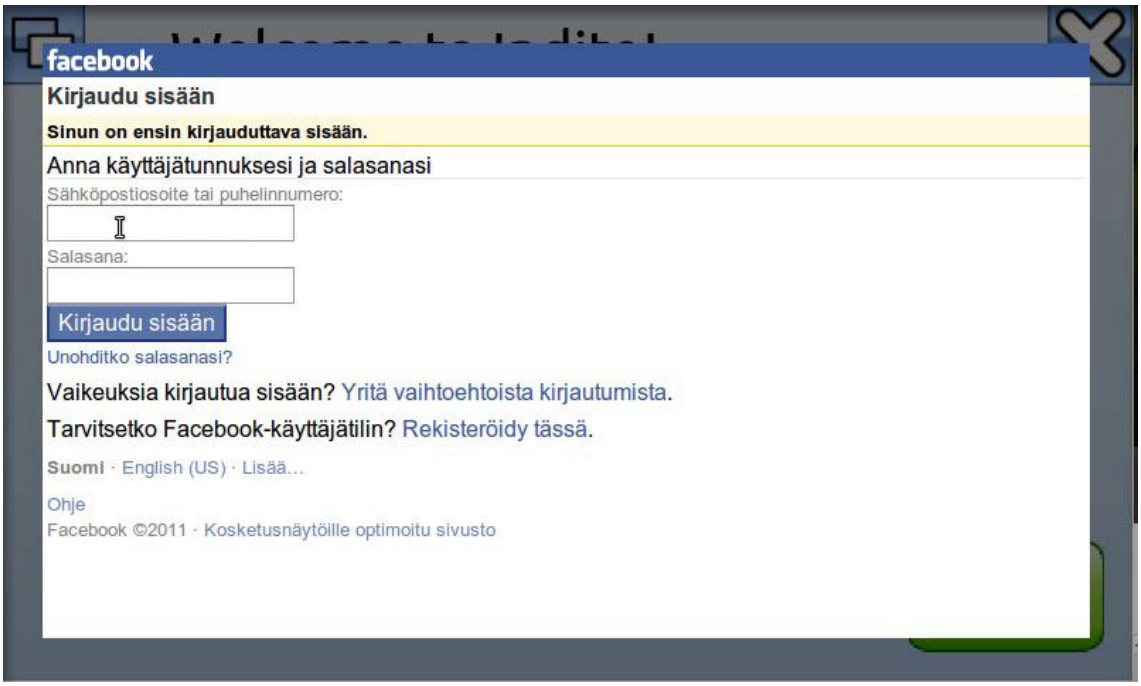
Sovelluskehittäjän näkökulmasta Facebook on ikävä palvelu. Jatkuvasti muuttuvat API:t ovat suuri haaste kehittäjälle ja toimiva ohjelma saattaa yhtäkkiä lopettaa toimintansa API-muutoksista johtuen. Tämä tarkoittaa sitä, että sovelluskehittäjän on jatkuvasti päivitettävä ohjelmaansa varmistaakseen sen toiminnan.

5.2 Ohjelman käyttäminen

Kun Inditen käynnistää ensimmäisen kerran aukeaa palveluiden kirjautumisnäkymä (Kuva 11) käyttäjälle. Käyttäjän painaessa joko Twitter- tai Facebook-logoa aukeaa kyseisen palvelun kirjautumisikkuna (Kuvat 12 ja 13).

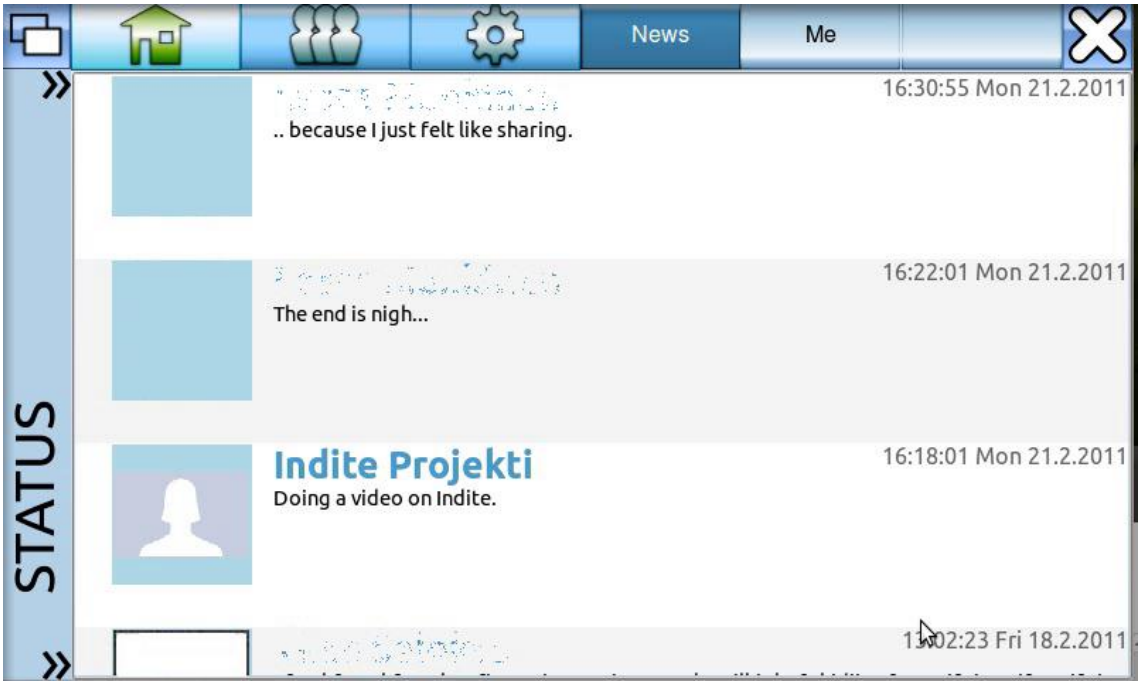
Kuva 11. Inditen alkunäkymä.

Kuva 12. Twitterin kirjautumisnäkymä.



Kuva 13. Facebookin kirjautumisnäkymä.

Kun käyttäjä on onnistuneesti kirjautunut palveluun kohtaa hän ohjelman päänäkymän (Kuva 14), jossa on nähtävissä käyttäjän Facebook-seinä ja Twitterin hometimeline. Listaa voi selata alaspäin, jos haluaa nähdä vanhempia tweettejä tai seinän kirjoituksia. Klikkaamalla listan elementtiä aukeaa siitä laajempi versio (Kuva 15), joka sisältää sen tekstin kokonaisuudessaan.



Kuva 14. Inditen päänäkymä.

Kuva 15. Elementin laajennettu näkymä.

Täältä käyttäjä voi myös lukea kommentit (Kuva 16) tai kirjoittaa niitä (Kuva 17), jakaa sen omille kavereilleen ja tarkistaa "Tykkäysten" lukumäärän (Kuva 18), tämä siis Facebookin tapauksessa. Twitter-viestin ollessa kyseessä, käyttäjä voi merkitä viestin suosikikseen tai jakaa sen omille kavereilleen.

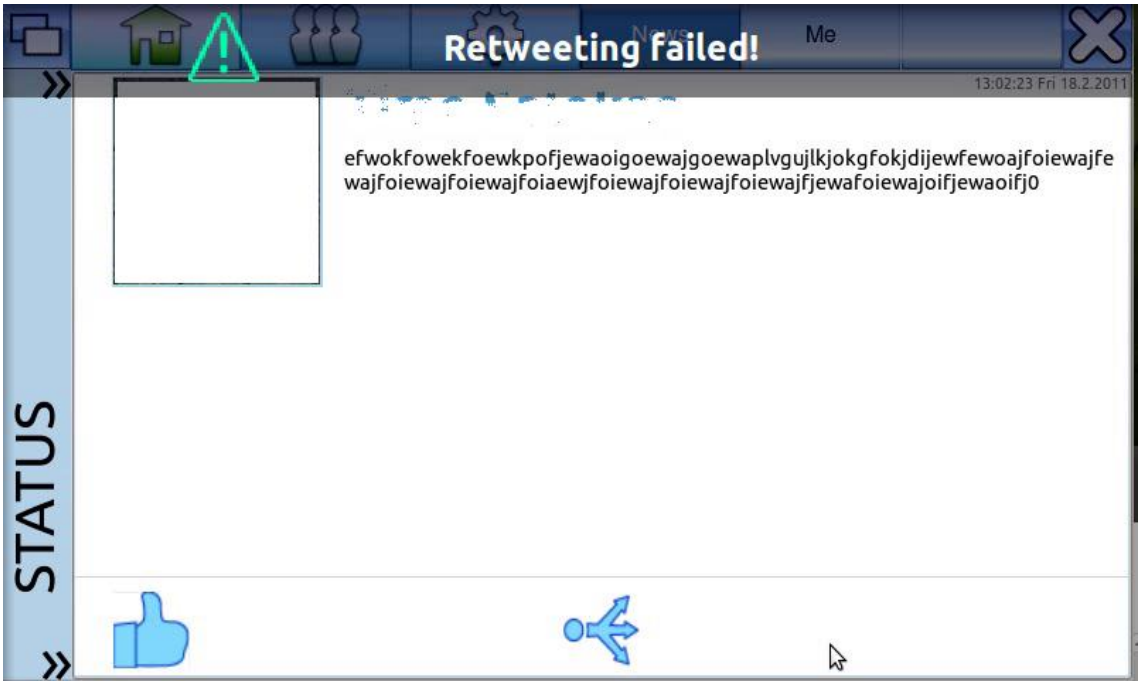
Kuva 16. Facebookin kommenttien luku.

Kuva 17. Facebook-kommentin kirjoitus.

Kuva 18. Facebookin "tykkääjät"-listattuna

Indite-ohjelma myös antaa käyttäjälle palautetta onnistuneista ja epäonnistuneista tapahtumista. Tälläisiä ilmoituksia ovat esimerkiksi onnistunut kommentin lähetys (Kuva 19) ja epäonnistunut Twitter-jako (retweet) (Kuva 20). Inditessä on myös ilmoitukset muun muassa onnistuneesta Twitter-jaosta ja Facebookiin lisätystä "Tykkäyksestä".

Kuva 19. Onnistuneesta kommentin lähetyksestä saatava ilmoitus.



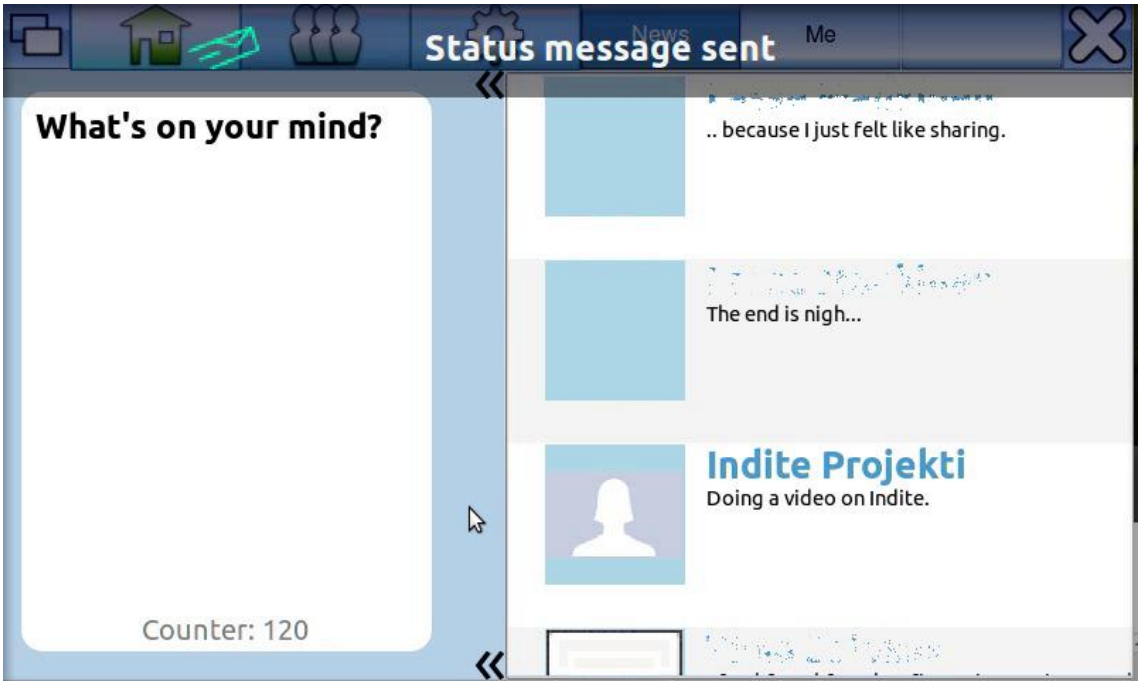
Kuva 20. Epäonnistuneesta Twitter-jaosta kertova ilmoitus.

Painamalla päänäkymän vasemmassa reunassa olevaa palkkia, liukuu esiin tilaviestin kirjoitustila (Kuva 21). Kun käyttäjä on kirjoittanut haluamansa, hän valitsee ensin palvelut joissa haluaa tekstin julkaista (Kuva 22) ja lopuksi kirjekuoren kuvan lähettääkseen tai rasti peruakseen lähettämisen (Kuva 23). Status-päivityksen lähetttämisen onnistuessa, tulee näkyviin ilmoitus onnistumisesta (Kuva 24).

Kuva 21. Tilaviestin kirjoitustila.

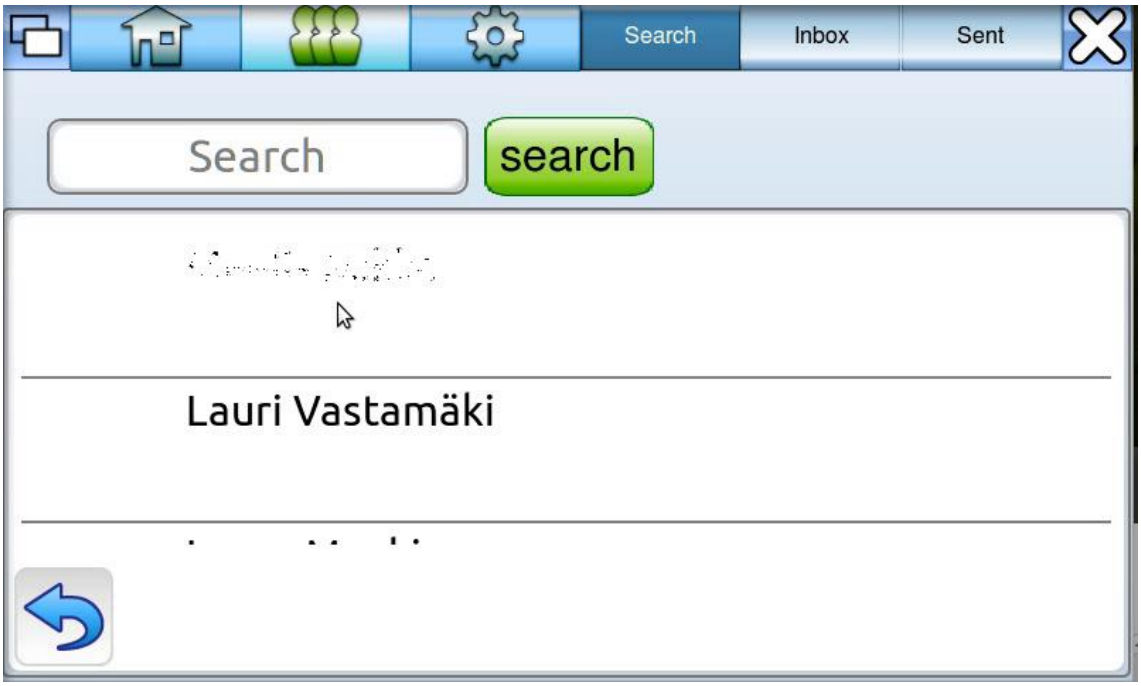
Kuva 22. Tilaviestin kohteen valinta.

Kuva 23. Tilaviestin lähetys tai peruutus.

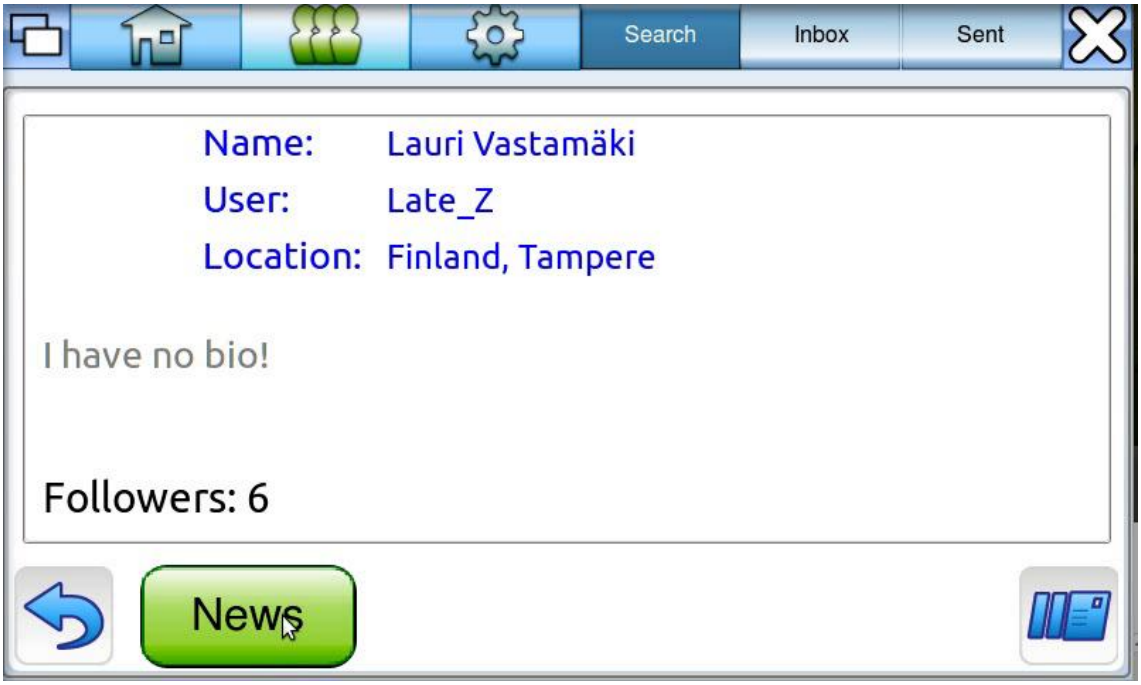


Kuva 24. Ilmoitus onnistuneesta tilapäivityksestä.

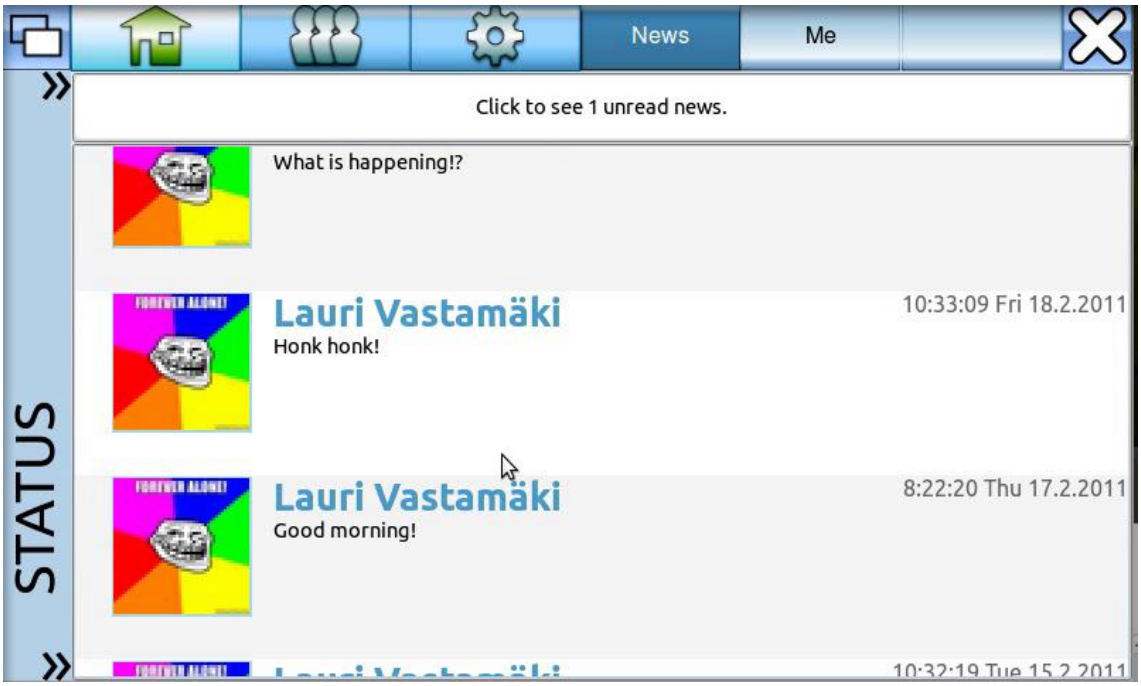
Yläpalkin Kaverit-napista, jossa näkyy kolme rinnakkaista hahmoa, pääsee näkemään Twitter-kaverinsa (Kuva 25) ja hakemaan Twitter-käyttäjiä. Painamalla kaverinsa nimeä, saa näkyviin tämän tarkempia tietoja (Kuva 26). Täältä voi myös lähettää yksityisviestin kaverille Twitterin välityksellä tai tarkastella kaverin tekemiä Twitter-päivityksiä (Kuva 27).



Kuva 25. Twitter-kaverit.



Kuva 26. Twitter-kaverin tarkemmat tiedot.

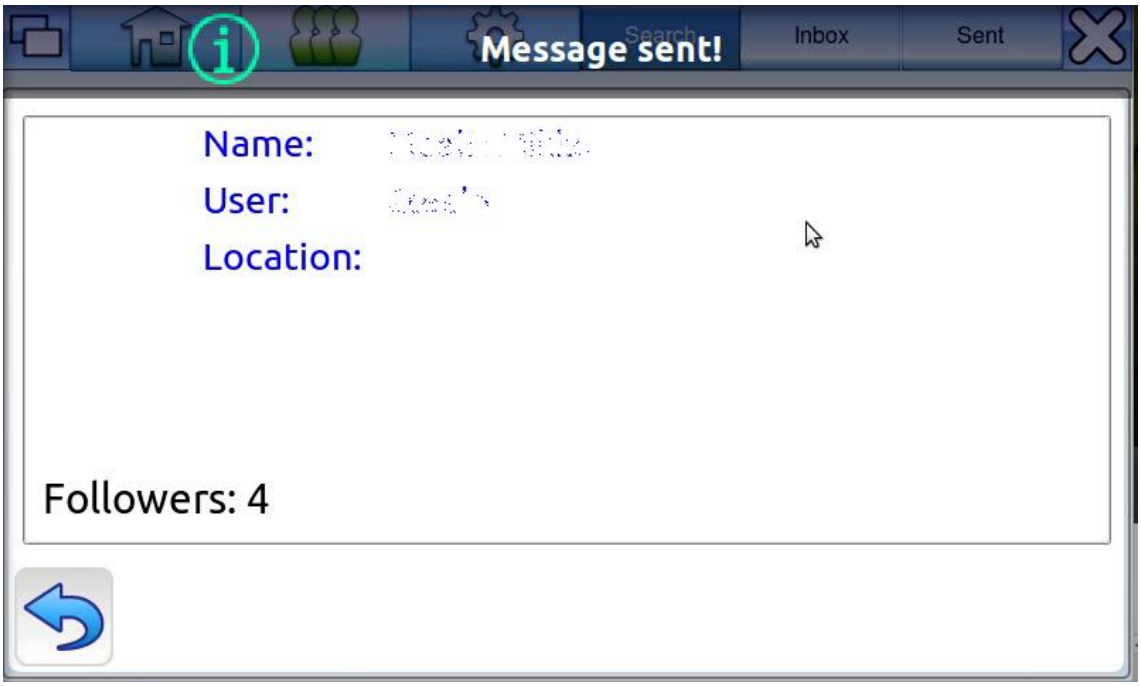


Kuva 27. Twitter-kaverin päivitykset.

Painamalla päänäkymän yläpalkin Me-kohtaa, saa näkyviin vain omat statuksensa aikajärjestyksessä. Painamalla News-kohtaa palaa ohjelma alkunäkymään.

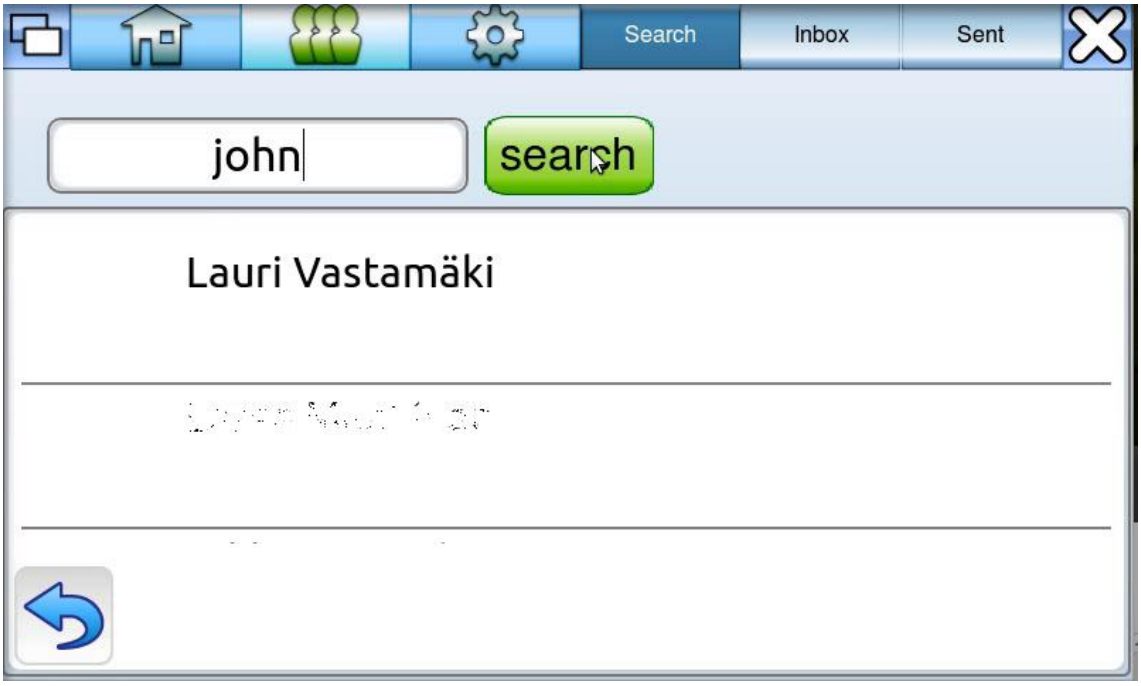
Yksityisviestin lähettäminen tapahtuu klikkaamalla tarkemmat tiedot näyttävässä tilassa, oikeassa alakulmassa näkyvää kirjekuori-nappia. Tämän jälkeen aukeaa yksityisviestin kirjoittamiseen luotu näkymä (Kuva 28). Twitterin 140-merkin rajasta johtuen lisättiin tänne käyttöä helpottava laskuri, joka päivittyy käyttäjän kirjoittaessa viestiään reaaliaikaisesti. Onnistuneesta viestin lähettämisestä annetaan ilmoitus käyttäjälle (Kuva 29).

Kuva 28. Yksityisviestin kirjoitus.

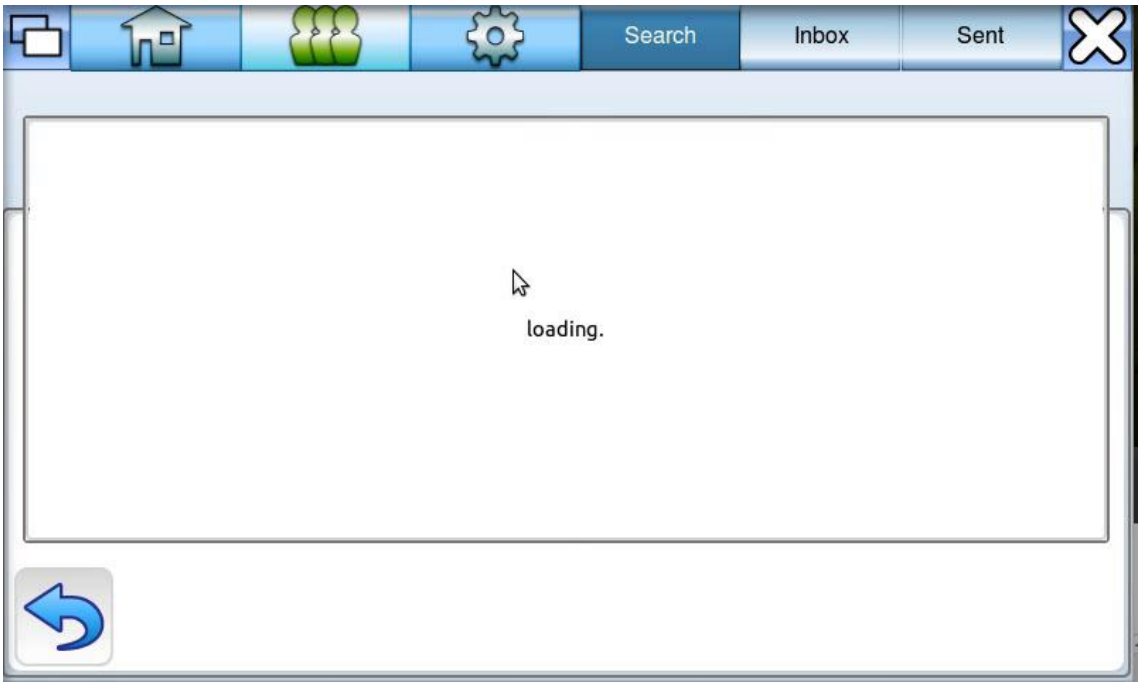


Kuva 29. Yksityisviesti lähetetty onnistuneesti ilmoitus käyttäjälle.

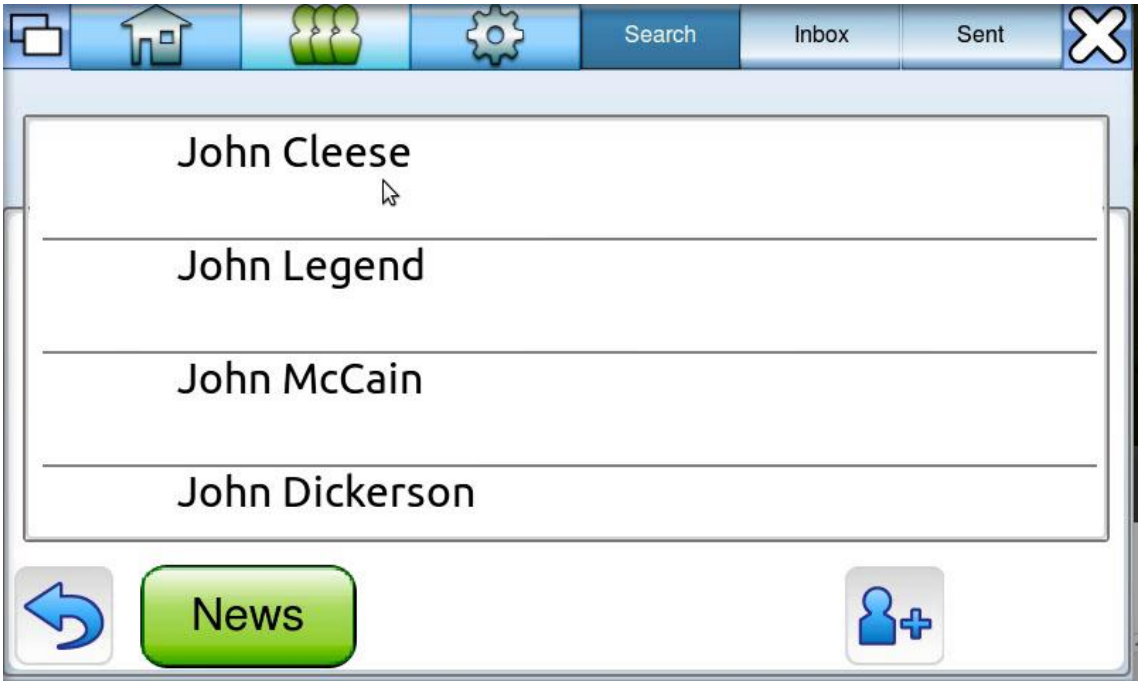
Twitter-käyttäjien haku tapahtuu kirjoittamalla hakutermi hakukenttään ja painamalla Search-nappia (Kuva 30). Kun haku on käynnistetty kerrotaan käyttäjälle haun olevan käynnissä (Kuva 31) ja haun valmistuttua annetaan käyttäjälle tulokset listattuna (Kuva 32). Tästä listasta painamalla saadaan tarkemmat tiedot näkyviin (Kuva 33). Tästä voi lisätä käyttäjän omiin seurattaviin, jolloin tämän tilapäivitykset tulevat käyttäjän omaan Twitter-näkymään tai voidaan katsella kyseisen henkilön tilapäivityksiä samalla tavalla kuin Twitter-kavereiden tapauksessa.



Kuva 30. Hakeminen Twitteristä.



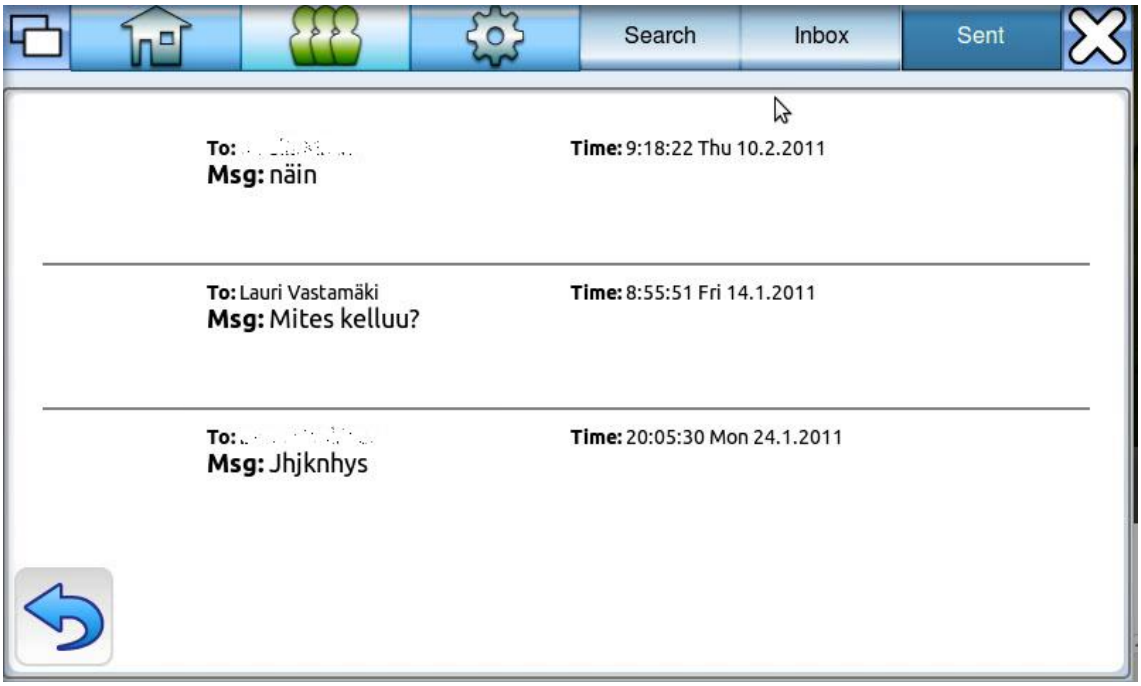
Kuva 31. Twitter-haku käynnissä.



Kuva 32. Twitter-haun tulokset.

Kuva 33. Twitter-haun tuottaman henkilön tarkemmat tiedot.

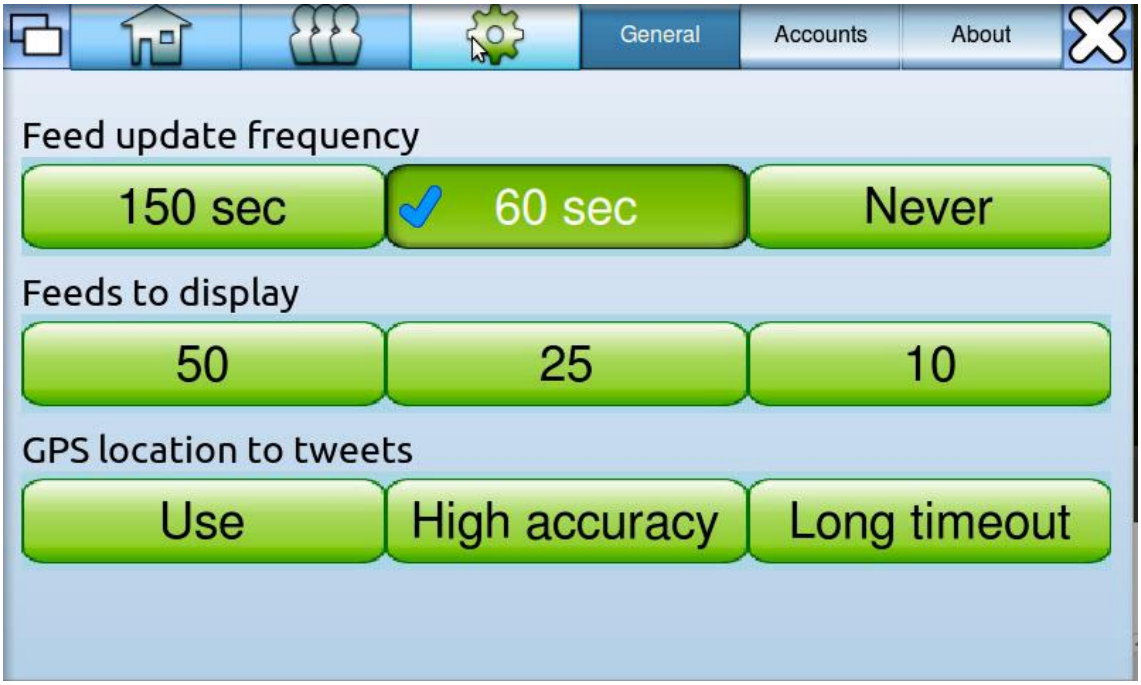
Kaverit-näkymässä yläpalkin oikealla sijaitsevat Inbox- ja Sent-napit tuovat esiin käyttäjän lähettämät ja hänelle lähetetyt yksityisviestit Twitterissä. Nämä esitetään rullattavassa listassa (Kuva 34).



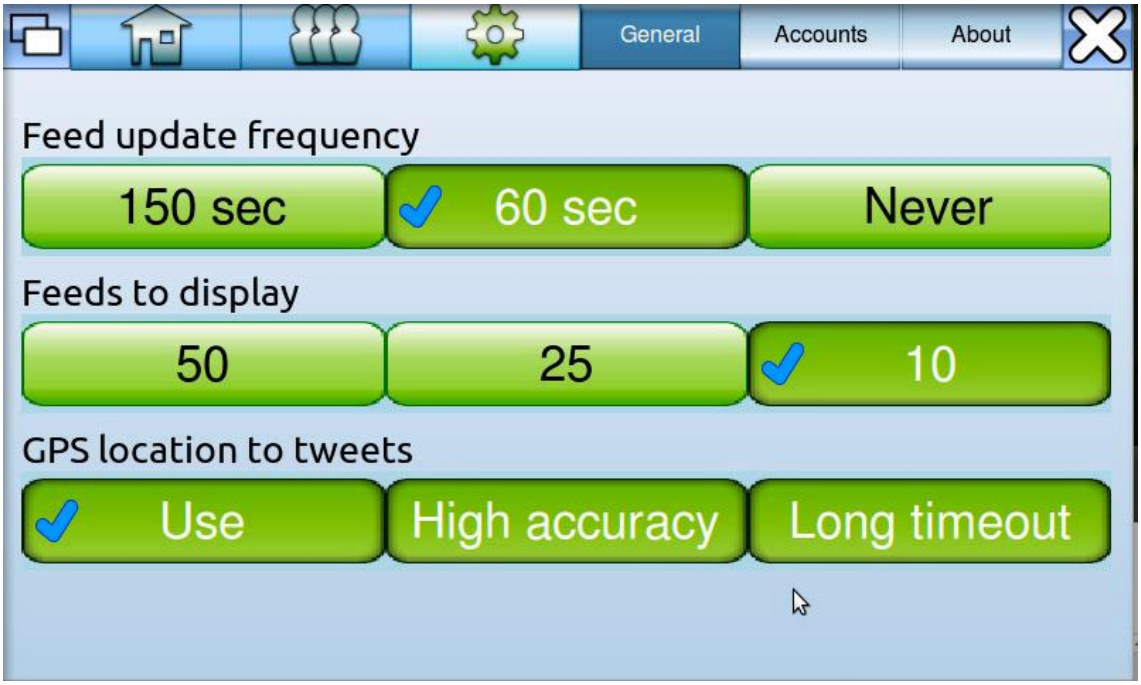
Kuva 34. Twitterin kautta lähetetyt yksityisviestit.

Inditen asetuksia pääsee säätämään Asetukset-napista, napissa rattaan kuva. Asetukset pitävät sisällään tietojen päivitysvälin, tiedon kuinka monta tilaa pidetään maksimissaan näkyvissä päänäkymässä ja sen, käytetäänkö Twitter-viestien yhteydessä paikkatietoa

GPS-sirulta ja jos käytetään, niin miten tarkasti ja millaisella aikakatkaisulla (Kuva 35). Asetetun ja muiden asetusten tilan huomaa helposti, koska nappi on pohjassa ja joissain tapauksissa sisältää myös sinisen "oikein"-merkin (Kuva 36). Tämä kaikki siis "General"-välilehdellä.



Kuva 35. Asetukset näkymä.

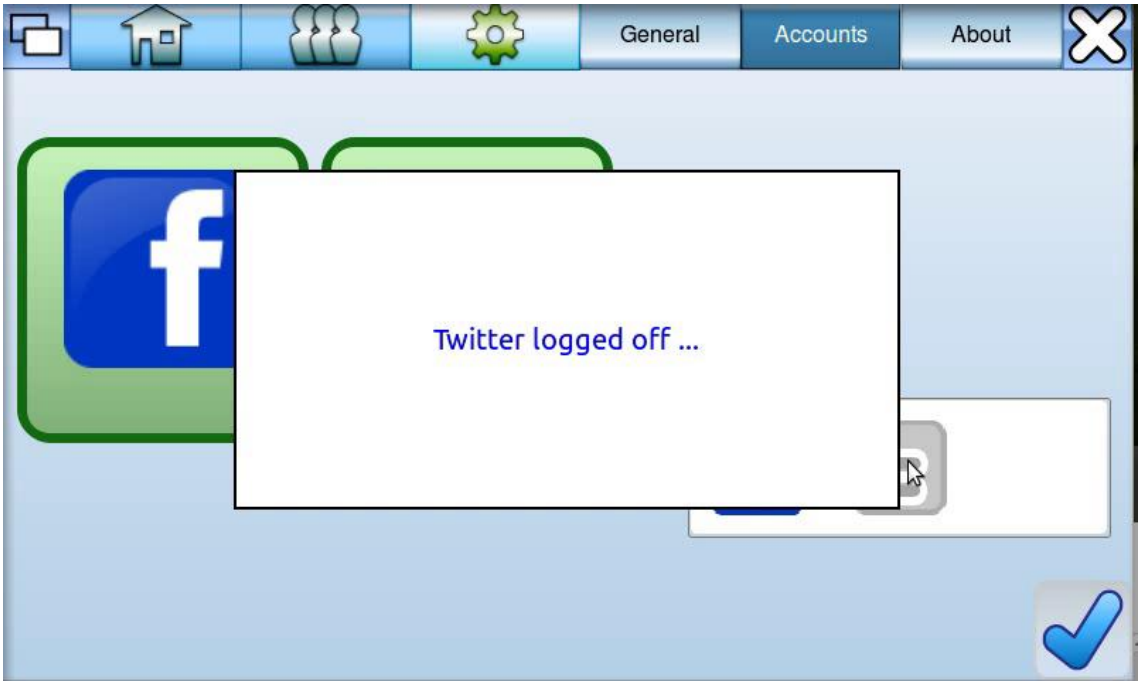


Kuva 36. Asetukset näkymä muutetuilla asetuksilla.

Asetusten "Accounts"-välilehdellä näkyy, mihin palveluihin käyttäjä on kirjautunut (Kuva 37). Täältä voi myös vaihtaa tunnusta, jolla on kirjauduttu tai kirjautua kokonaan

ulos haluamastaan palvelusta. Käyttäjän kirjautuessa ulos palvelusta, annetaan siitä tieto (Kuva 38), myös sisäänkirjautumisen onnistumisesta tiedotetaan käyttäjää (Kuva 39).

Kuva 37. "Accounts"-välilehti asetuksissa.



Kuva 38. Tieto uloskirjautumisesta.

Kuva 39. Tieto onnistuneesta sisäänkirjautumisesta.

Asetusten viimeisellä, "About"-välilehdellä (Kuva 40), kerrotaan Inditen tekijät ja internetosoite, josta voi ladata ohjelman lähdekoodin. Myös Indite-ohjelman hieno logo löytyy isona täältä.

Kuva 40. "About"-välilehti.

Oikeassa yläkulmassa näkyy tilasta riippumatta rasti, jolla ohjelman voi sulkea. Vasemmassa yläkulmassa puolestaan on Maemon pienennä ohjelma ikoni, josta ohjelman saa pienennettyä sulkematta sitä.

6 YHTEENVETO

Qt Quick -kehitys on nopeaa ja pienellä opettelulla helppoa. Mikäli kehittäjällä on ennestään vankka pohjaosaaminen Qt-kehityksestä, tai vaihtoehtoisesti C++-kehityksestä, on hänen helppoa päästä sisään Qt Quick -kehitykseen. Qt Quick mahdollistaa näyttävät ja toiminnallisesti erinomaiset ohjelmat, ilman suurempaa tarvetta graafiselle osaamiselle.

Maemo-alusta edellyttää omat uniikit piirteensä ohjelmalta, mutta tämän ei pidä antaa lannistaa kehitysintoa. Maemo-alusta on saavuttanut suuren suosion, joten mahdollisiin ongelmiin löytää vastauksia ja kehitykseen tukea kokeneemmilta kehittäjiltä kun sitä tarvitsee.

Qt Quick -kehityksen aloitus on tehty helpoksi, kaiken mitä tarvitsee voi ladata ilmaiseksi verkosta hyvien ohjeiden kera. Myös Maemo-alustalle kehitys on helppoa, työkalut sekä tietokoneelle että esimerkiksi N900-laitteelle ovat helppokäyttöisiä ja intuitiivisia.

Toki kaikessa kehityksessä on omat ongelmansa, mutta Qt Quick Maemo-alustalla ei ole niitä ongelmallisimpia. Esimerkiksi N900-laitteen tehon puute vaikuttaa kehittäjän ratkaisuihin, mutta tämä ei ole Qt Quick -kielen ongelma. Qt:n version 4.7.0 ohjelmointivirheet aiheuttavat ylimääräistä tekemistä, mutta version päivityksellä näistäkin pääsee.

LÄHTEET

Shaver, M. 2009. Documentation | oFono. Luettu 7.11.2011.
http://ofono.org/documentation.

maemo.org - DBusGuide. Luettu 7.11.2011.
http://maemo.org/community/wiki-closed/dbusguide/.

Qt 4.7: Qt Declarative UI Runtime. 2010. Luettu 7.11.2011.
http://doc.qt.nokia.com/4.7/qmlruntime.html.

Qt 4.7: Signals & Slots. 2010. Luettu 7.11.2011.
http://doc.qt.nokia.com/4.7/signalsandslots.html.

Qt 4.7: Using the Meta-Object Compiler (moc). 2010. Luettu 7.11.2011.
http://doc.qt.nokia.com/latest/moc.html.

Qt Mobility 1.1. 2010. Luettu 7.11.2011.
http://doc.qt.nokia.com/qtmobility-1.1.0-beta/index.html.

Thelin, J. 2010. Using CMake to Build Qt Projects | Qt Developer Network. Luettu 7.11.2011.
http://developer.qt.nokia.com/quarterly/view/using_cmake_to_build_qt_projects.