Adobe Flex ja Microsoft Silverlight sovelluskehityksessä

HAMK
HÄMEEN AMMATTIKORKEAKOULU

Ammattikorkeakoulun opinnäytetyö

Mediatekniikka

Riihimäki

Teemu Ristivuori

HAMK
HÄMEEN AMMATTIKORKEAKOULU

TIIVISTELMÄ

RIIHIMÄKI
Mediatekniikka
Ohjelmistotekniikka

Tekijä Teemu Ristivuori **Vuosi** 2012

Työn nimi Flex ja Silverlight sovelluskehityksessä

TIIVISTELMÄ

Tässä opinnäytetyössä tutkitaan Adobe Flex:n ja Microsoft Silverlight:n välisiä eroja sekä molempien tekniikoiden vahvuuksia ja heikkouksia. Tekniikat ovat hyvin samankaltaiset ja HAMK haluaa selvittää alustojen soveltuvuuden opetuskäyttöön.

Osana opinnäytetyötä toteutettiin demo-tason ohjelma Metso Mineralsille työnimeltään ”Metso Atlas”. Ohjelma on toteutettu vertailun vuoksi molemmilla tekniikoilla.

Työn toimeksiantajana on Hämeen ammattikorkeakoulun Riihimäen yksikkö sekä Metso Minerals.

Työn teoriaosuudessa käydään opinnäytetyön kannalta olennaiset termit ja käsitteet läpi.

Opinnäytetyön tutkimusosuudessa käydään sekä Flex että Silverlight yksityiskohtaisesti läpi. Metso Atlas-sovelluksen toteutus molemmilla tekniikoilla käydään läpi kappaleessa 5. Pääpaino on vastaan tulleissa ongelmissa, sekä tekniikoiden välisissä eroissa.

Tutkimusosuuden ja Metso Atlas-sovellusten perusteella saatiin lopputulokseksi yleisellä tasolla se, että kumpaakaan tekniikkaa ei voi yksiselitteisesti julistaa paremmaksi. Flex on SDK-komponenteiltaan kypsempi mutta erot ovat pieniä ja tärkeämpi valintakriteeri on projektin kohdealusta.

Opetuskäyttöön molemmat tekniikat soveltuvat hyvin, mutta eri tavalla. Flex/Flash Builder on selkeytensä takia erinomainen valinta ohjelmoinnin peruskursseille. Silverlight puolestaan on Windows Phonen pääkehitysalusta, mikä tekee siitä loogisen valinnan osana mobiili-ohjelmointia.

Avainsanat Adobe Flex, Microsoft Silverlight, sovelluskehitys

Sivut 31 s. + liitteet 31 s.

HAMK
UNIVERSITY OF APPLIED SCIENCES

RIIHIMÄKI
Degree Programme in Media Technology
Software development

Author Teemu Ristivuori **Year** 2012

Subject of Bachelor's thesis Flex and Silverlight in software development

ABSTRACT

This thesis focuses on investigating the differences between Adobe Flex and Microsoft Silverlight and the strengths and weaknesses of both technologies. The thesis was commissioned by HAMK University of Applied Sciences and Metso Minerals. The technologies are very similiar and HAMK University of Applied Sciences wants to find out how viable they would be in educational use.

The demo software called "Metso Atlas" was created as a part of this thesis for Metso Minerals. The software was created twice, using both technologies for comparative purposes.

The theoretical chapters of the thesis focus on explaining all the relevant terms and concepts. The research part of the thesis goes through both Flex and Silverlight in detail. The implementation of Metso Atlas with both technologies is described in Chapter 5. The main focus is on the problems encountered and in the differences between the two technologies.

Based on the research part, and the Metso Atlas-applications, it was found that neither technology can be simply declared as better. Flex is a more mature technology in regards to its SDK-components, however the differences are minor. This makes the project's target platform a far more important criteria.

For educational use both technologies work well, but in different ways. Flex/Flash Builder is an excellent choice for basic/starter courses due to its clarity. Silverlight on the other hand is the main platform for the Windows Phone, which makes it a logical choice as a part of any package involving mobile programming.

Keywords Adobe Flex, Microsoft Silverlight, Software development

Pages 31 p. + appendices 31 p.

SISÄLLYS

1 JOHDANTO 1

2 TEORIA JA TERMIT 1

2.1 Rikkaat internet-sovellukset 1
2.2 Adobe Flex 1
2.3 Microsoft Silverlight 2
2.4 Adobe AIR ja työpöytäsovellukset 2

3 ADOBE FLEX 3

3.1 Flash Builder 3
3.1.1 Käyttöliittymä 4
3.1.2 Työkalut ja ominaisuudet 6
3.2 Yhteensopivuus 9
3.3 Ohjelmointikielet 9
3.3.1 ActionScript 3.0 10
3.3.2 MXML 10
3.4 Yhteisön taso ja levinneisyys 11
3.5 Työpöytäsovellukset 11

4 MICROSOFT SILVERLIGHT 11

4.1 Visual Studio 11
4.1.1 Käyttöliittymä 12
4.1.2 Työkalut ja ominaisuudet 16
4.2 Yhteensopivuus 20
4.3 Ohjelmointikielet 21
4.3.1 C# 21
4.3.2 XAML 22
4.4 Yhteisön taso ja levinneisyys 23
4.5 Työpöytäsovellukset 23

5 METSO ATLAS-ESIMERKKISOVELLUS 23

5.1 Sovellus Adobe Flex:llä 24
5.1.1 MetsoAtlas.mxml 24
5.1.2 kartta.mxml 24
5.1.3 oikeaValikko.mxml 24
5.1.4 vasenValikko.mxml 24
5.1.5 sisaltoPopup.mxml 25
5.1.6 HaeTiedot.as 25
5.1.7 MetsoAtlas-app.xml 25
5.2 Sovellus Microsoft Silverlightillä 25
5.2.1 MainPage.xaml 25
5.2.2 kartta.xaml 25
5.2.3 oikeaValikko.xaml 26
5.2.4 vasenValikko.xaml 26
5.2.5 sisaltoPopup.xaml 26
5.2.6 HaeTiedot.cs 26

5.2.7 kaavioSisalto.cs 26
5.3 Havaitut erot Flexin ja Silverlightin välillä 26

YHTEENVETO 27

5.4 Adobe Flex 28
5.4.1 Vahvuudet 28
5.4.2 Heikkoudet 28
5.4.3 Soveltuvuus opetuskäyttöön 28
5.5 Microsoft Silverlight 29
5.5.1 Vahvuudet 29
5.5.2 Heikkoudet 29
5.5.3 Soveltuvuus opetuskäyttöön 30

LÄHTEET 31

Liite 1 Flex-sovelluksen lähdekoodi luokittain
Liite 2 Silverlight-sovelluksen lähdekoodi luokittain

1 JOHDANTO

Adobe Flex ja Microsoft Silverlight ovat molemmat ohjelmien kehittämisympäristöjä (SDK) rikkaiden internet-sovellusten tuotantoon.

Tässä opinnäytetyössä tutkitaan näiden tekniikoiden välisiä eroja, vahvuuksia ja heikkouksia. Opinnäytetyön pääpaino on työpöytäsovelluksissa mutta myös internet-puolta ja tekniikoiden teoriaa käydään läpi.

Osana opinnäytetyötä on myös demo-tason ohjelma Metso Mineralsille työnimeltään ”Metso Atlas”. Tämä ohjelma on toteutettu vertailun vuoksi molemmilla tekniikoilla.

Lisäksi tutkitaan Flexin ja Silverlightin soveltuvuutta opetuskäyttöön HAMK:ssa.

Lukijan oletetaan ymmärtävän ohjelmointia ja sovelluskehitystä, joten alan perustermejä ei selitetä erikseen. Lisäksi monia työssä käytettyjä termejä ei käännetä suomeksi, koska alan kieli on englanti ja lopputuloksena olisi dokumentti, jota ei helposti ymmärtäisi alan osaaja eikä maallikko.

2 TEORIA JA TERMIT

Tässä osiossa käydään läpi opinnäytetyön kannalta olennaiset tekniikat ja termit yleisellä tasolla.

2.1 Rikkaat internet-sovellukset

Rikkaat internet-sovellukset eli RIA-sovellukset ovat web-sovelluksia joilla on monia työpöytäsovelluksien piirteitä. Adobe Flash, JavaFX ja Microsoft Silverlight ovat tämän hetken yleisimmät alustat.

Yhteistä näillä alustoilla on vaatimus käyttäjän koneelle asennettavasta ajoympäristöstä (Adobe Flash, Java, Microsoft Silverlight). Toinen tapa on käyttää JavaScript-pohjaisia vaihtoehtoja kuten Ajaxia.

Rikkaat internet-sovellukset käyttävät niin sanottua ”Rich Client”-mallia eli valmiiksi käännetty ohjelma välitetään käyttäjälle selaimen kautta. (Labriola, Tapper & Boles, Adobe Flex 4.5 Fundamentals, 2012, 3-9.)

Opinnäytetyössä käytetään RIA-lyhennettä tästä eteenpäin selkeyden takia.

2.2 Adobe Flex

Adobe Flex on Adobe Flashille RIA-sovellusten kehittämiseen tarkoitettu SDK (software development kit). Alun perin Flexin julkaisi Macromedia vuonna 2004 auttaakseen kehittäjiä kirjoittamaan web-sovelluksia Flash-alustalle.

Virallisena kehitysympäristönä käytetään Adoben itse julkaisemaa Adobe Flash Builderia ja käyttöliittymän suunnitteluun Adobe Flash Catalystia. (Labriola, Tapper & Boles, Adobe Flex 4.5 Fundamentals, 2012, 13-14.)

Flex käydään yksityiskohtaisesti läpi osiossa 3.

2.3 Microsoft Silverlight

Microsoft Silverlight on RIA-sovelluksien toteuttamiseen tarkoitettu ohjelmointiympäristö (SDK). Silverlight vaatii selainliitännäisen asentamisen ja on siten suoraan verrattavissa Adobe Flashiin.

Silverlightin pääasiallisena kehitysympäristönä toimii Microsoft Visual Studio. Graafiseen käyttöliittymä-suunnitteluun voi käyttää myös Expression Blend-ohjelmaa. (MacDonald, Pro Silverlight 4 in C#. 2010, xxv-xxxii)

Silverlight käydään yksityiskohtaisesti läpi osiossa 4.

2.4 Adobe AIR ja työpöytäsovellukset

Adobe AIR, koko nimeltään Adobe Integrated Runtime, on alustariippumaton ajoympäristö, jonka tarkoituksena on ajaa RIA-sovelluksia työpöytäsovelluksina tai mobiililaitteilla.

Adobe AIR tukee seuraavia käyttöjärjestelmiä:

- Windows
- Linux (virallinen tuki lopetettu versiossa 2.6)
- Mac OS

Lisäksi ainakin seuraavat mobiilikäyttöjärjestelmät ovat tuettuja:

- BlackBerry Tablet OS
- iOS
- Android

AIR:n avulla voidaan toteuttaa työpöytäsovelluksia käyttäen RIA-ohjelmointitekniikoita, kuten Flash, ActionScript, HTML ja JavaScript. AIR:lle kehitetty sovellus toimii missä tahansa laitteessa, johon on asennettu AIR-ajoympäristö, kunhan laite on tarpeeksi tehokas ajamaan ohjelman ja ohjelmoija on noudattanut hyviä ohjelmointikäytäntöjä.

Jos sovellus kehitetään suoraan AIR-sovellukseksi, voidaan käyttää hyväksi lisäominaisuuksia, joita RIA-sovelluksista ei normaalisti löydy:

- Tiedostojärjestelmä-integraatio
- Native client extensions
- Native window/screen integration
- Taskbar/dock integration
- Tuki laitteiden kiihtyvyysantureille ja GPS:lle.
- Sisäänrakennettu SQLite-tuki

(Wikipedia 2012a.)

Osa yllä olevasta listasta on tarkoituksella jätetty kääntämättä suomeksi, koska käännöksistä ei suoraan ymmärrä, mistä on kyse, kun taas alan osaaja ymmärtää englanninkieliset termit ongelmitta.

Varsinkin sisäänrakennettu SQLite-tuki ansaitsee erityismaininnan. Suora tuki ohjelman sisäiselle tietokannalle, jota ei tarvitse käynnistää ja joka toimii ilman mitään palvelinta, on äärimmäisen hyödyllinen ominaisuus työpöytäsovellukselle.

3 ADOBE FLEX

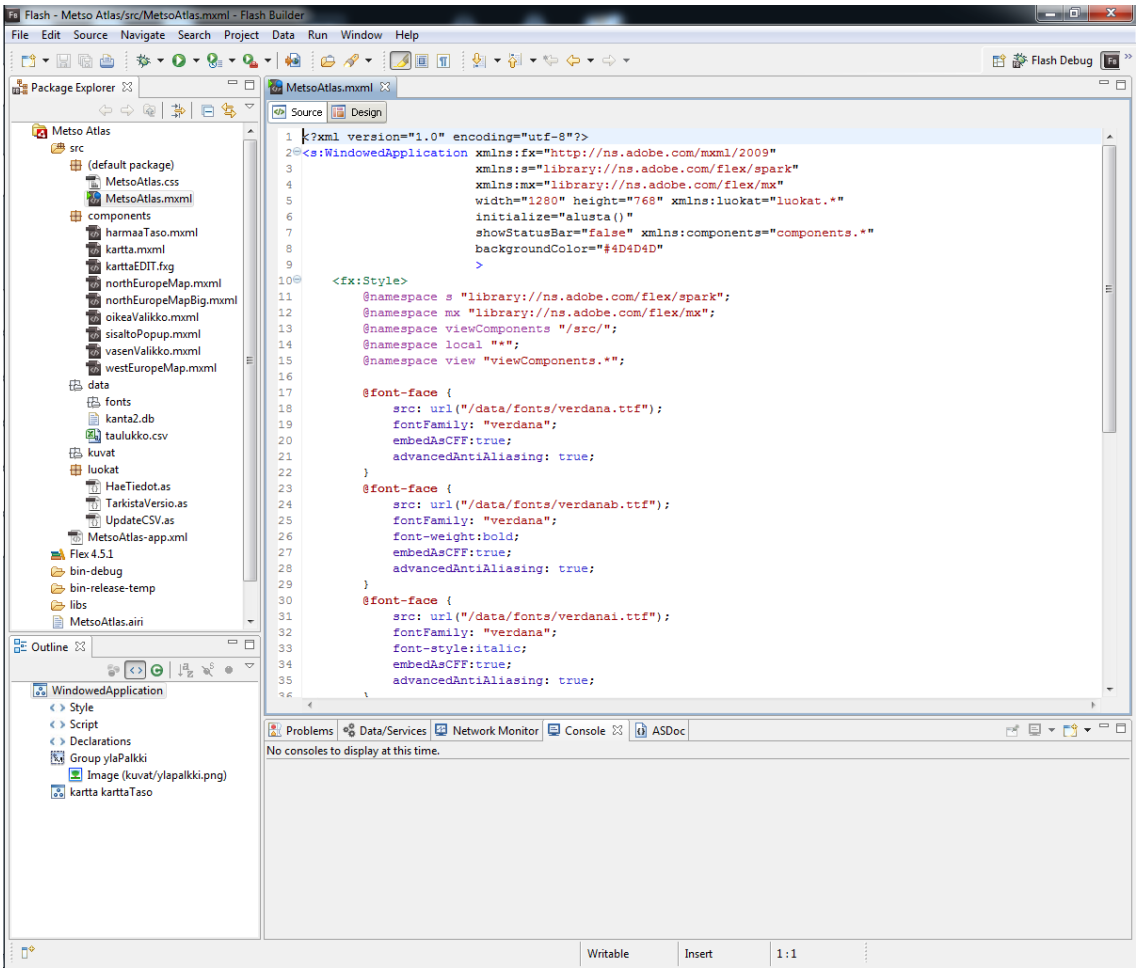
Tässä osiossa käydään Adobe Flex tarkemmin läpi tekniikkana, sekä tutkitaan sen soveltuvuutta yleisesti ohjelmistokehitysalustaksi.

3.1 Flash Builder

Käytetyn kehitysympäristön helppokäyttöisyys ja toiminnallisuudet ovat erittäin tärkeitä kriteereitä, kun valitaan eri ohjelmointialustojen välillä. Tässä osiossa tutustutaankin tarkemmin Flex-alustan ensisijaiseen kehitysympäristöön, eli Adobe Flash Builderiin.

Flash Builderin rinnalla on Adobe Catalyst, joka on suunniteltu käyttöliittymän graafiseen toteutukseen. Catalyst on täten verrattavissa Microsoftin Expression Blendiin. Catalystin yleisin käyttötarkoitus on projekteissa, joissa ohjelmoinnin ja graafisen puolen toteuttavat eri henkilöt. Jos ohjelmoija tekee itse myös ulkoasun, niin nämä ohjelmat eivät yleensä tarjoa lisäarvoa ohjelmoijalle.

3.1.1 Käyttöliittymä



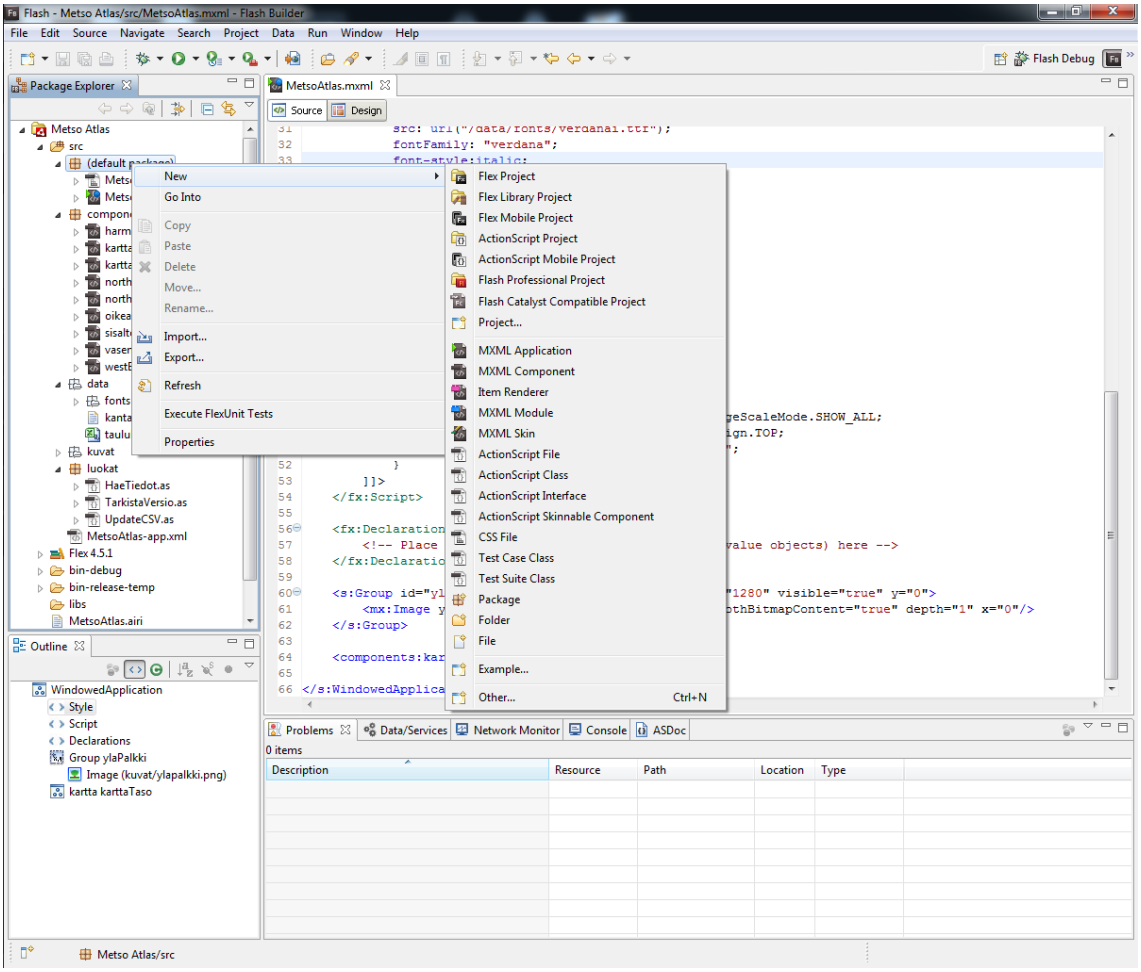
Kuva 1 Flash Builder - käyttöliittymä

Kuvassa 1 on Adobe Flash Builderin normaali työskentelynäkymä. Yleisilme on selkeä ja ympäristö on jaettu selkeisiin kokonaisuuksiin. Vasemmassa reunassa käyttöliittymää on Package Explorer. Sen sisältä löytyvät kaikki projektit ja niiden kaikki tiedostot paketti(package)- ja hakemistorakenteineen.

Projektit ja niiden tiedostot ovat selkeässä puurakenteessa. Puun juuri on projektin nimi eli esim. ”Metso Atlas” ja kun projektin puun avaa, sen edellä olevasta kolmiosta tulee näkyviin projektin hakemistorakenne. Oletushakemistot ovat seuraavat:

- src
  - Kaikki ohjelman lähdekoodit löytyvät tämän puurakenteen alta.
- bin-debug ja bin-release-temp
  - Näissä hakemistoissa säilytetään ajamista varten käännettyjä tiedostoja eikä käyttäjän tarvitse niistä piitata. (Huom. Ei tule sekoittaa lopulliseen käännettyyn ohjelmapakettiin.)
- libs
  - Tämä hakemisto on tarkoitettu ohjelman käyttämille ulkoisille kirjastoille

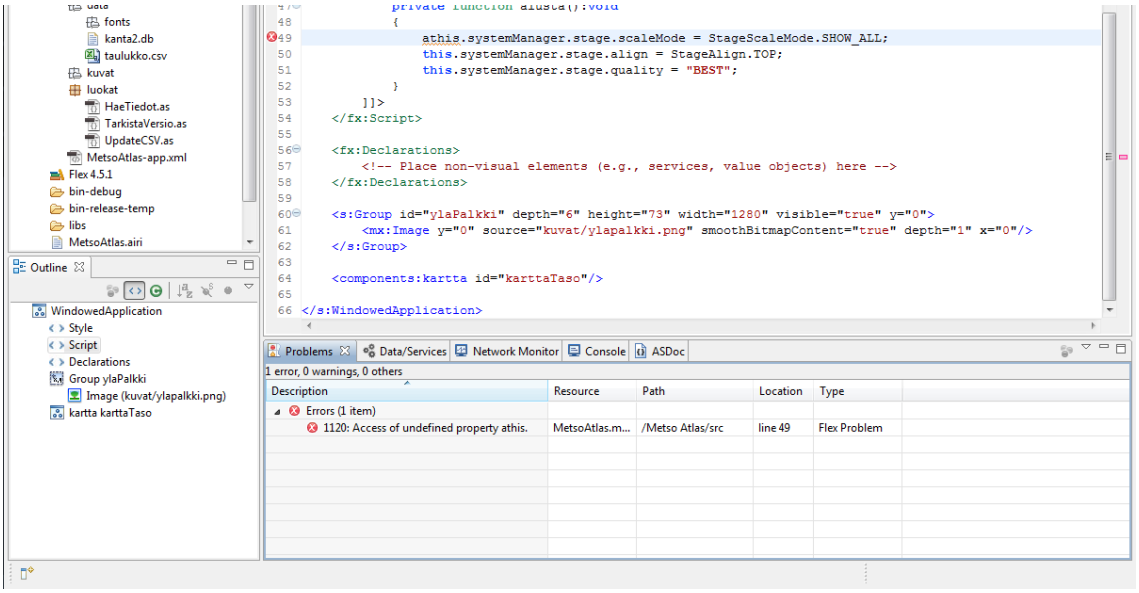
Nämä hakemistot luodaan automaattisesti ympäristön toimesta kun uusi projekti lisätään. Bin-debug- ja bin-release-temp-hakemistoihin ei tule koskea itse. Sen sijaan src-hakemiston rakenteella ei ole mitään rajoituksia lukuun ottamatta AIR-asetustiedostoa, jonka tulee sijaita src:n perusjuuressa. Package Explorerista lisätään myös uudet rakenteet ja tiedostot, kuten alla olevasta kuvasta 2 ilmenee.



Kuva 2 Flash Builder - Package Explorer ja tiedoston lisääminen

Package Explorerin oikealla puolella on Flash Builderin työskentelytila. Tilassa näkyy Package Explorerista valittu tiedosto. Työskentelytilalla on kaksi eri moodia, Source-moodi ja Design-moodi. Source-näkymässä näkyy kyseisen tiedoston lähdekoodi ja sitä käytetään ohjelmointiin. Design-näkymää käytetään käyttöliittymän graafiseen suunnitteluun.

Työskentelytilan alapuolella on tila erilaisille ilmoituksille ja työkaluille joita voi vaihtaa siinä olevien välilehtien avulla. Yleisimmin tässä osiossa näkyvät konsoli, debug-viestit ja virheilmoitukset. Esimerkki tästä löytyy kuvasta 3.



Kuva 3 Flash Builder - ala-osio

Package Explorerin alapuolella on Outline-osio, joka näyttää valitun tiedoston rakenteen yksinkertaistetusti. Tämä auttaa navigoimaan tiedoston sisällä nopeasti.

Ohjelman oikeasta ylänurkasta löytyy painikkeita, joilla voi vaihtaa koko ohjelman kokonaisnäkymää eri moodeihin. Oletuksena on perusnäkymä ja Debug-näkymä mutta muitakin löytyy, kuten esim. CVS näkymä.

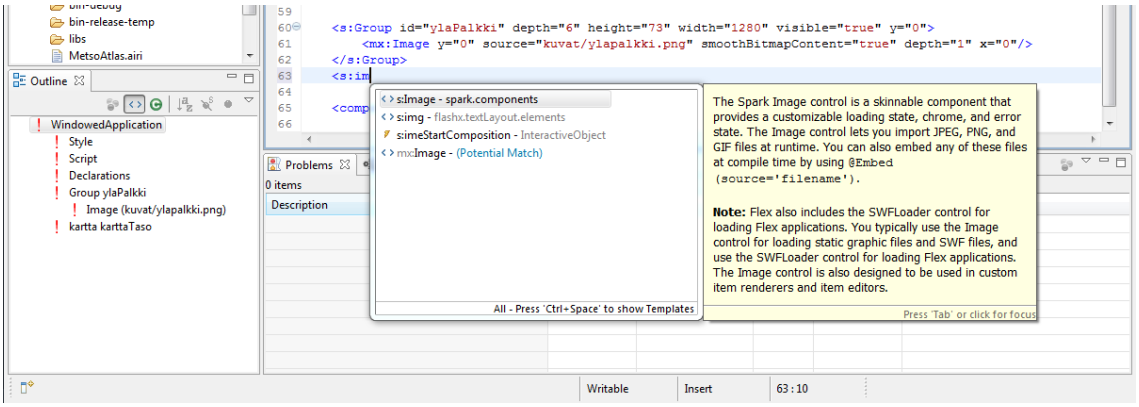
Debug-osion tarkoitus on analysoida käynnissä olevan ohjelman toimintaa. Flash Builder vaihtaa tähän osioon automaattisesti, kun työn alla oleva ohjelma käynnistetään debug-tilassa. Debug-osio käydään läpi tarkemmin Työkalut ja ominaisuudet-osiossa(3.1.2).

3.1.2 Työkalut ja ominaisuudet

Koodiehdotus on yksi minkä tahansa kehitysalustan tärkeimmistä ominaisuuksista. Kooditäydennyksen/ehdotusten idea on se, että ympäristö tunnistaa mitä ohjelmoija on tekemässä ja ehdottaa täydennystä. Tämä nopeuttaa ohjelmointia huomattavasti.

Kooditäydennyksen voi Flash Builderissä avata myös käsin painamalla ctrl+välilyöntiä. Tämä on erittäin hyödyllistä, koska ympäristö ei aina ehdota asioita tarpeeksi aikaisin. Täydennyksen avaaminen käsin on tarpeen myös, jos ohjelmoija haluaa etsiä listasta jotain tiettyä toimintoa. Listassa valittuna olevan toiminnon/komponentin tarkemmat tiedot ovat näkyvissä listan oikealla puolella. Tämä vähentää tarvetta etsiä toimintojen tarkoitusta dokumentaatiosta ja parantaa siten ajankäyttöä.

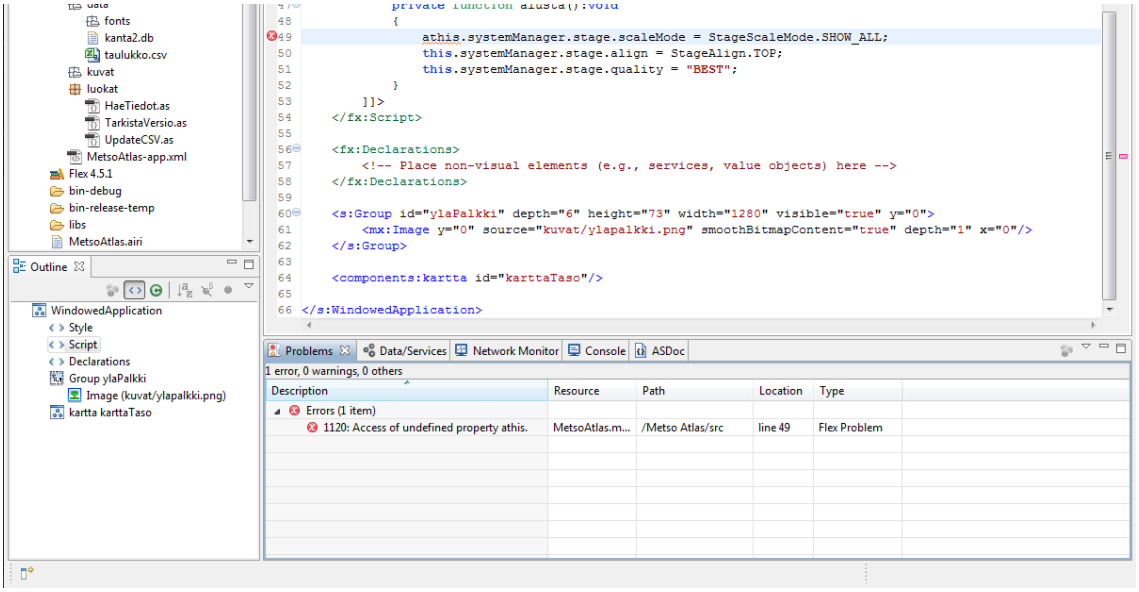
Esimerkki kooditäydennyksestä kuvassa 4.



Kuva 4 Flash Builder - esimerkki kooditäydennyksestä

Reaaliaikainen koodintarkistus on myös olennainen osa mitä tahansa vakavasti otettavaa kehitysympäristöä. Ympäristö lukee ohjelmoijan koodia, ilmoittaa mahdollisista virheistä tai huonoista käytännöistä ja merkitsee ongelmakohdat suoraan koodiin.

Flash Builder tukee tätä ominaisuutta, tosin välillä käsiteltävä tiedosto täytyy tallentaa (ctrl+s), että koodin tarkistus päivittyy. Lisäksi editoitavan tiedoston tulee olla oikeasti osa projektia, muuten virheentarkistus ei lue sitä. Eli ohjelman täytyy oikeasti käyttää kyseistä tiedostoa jollain tavalla. Tiedoston sijainti projektin puurakenteessa ei yksinään riitä.



Kuva 5 Flash Builder - reaaliaikainen koodintarkistus

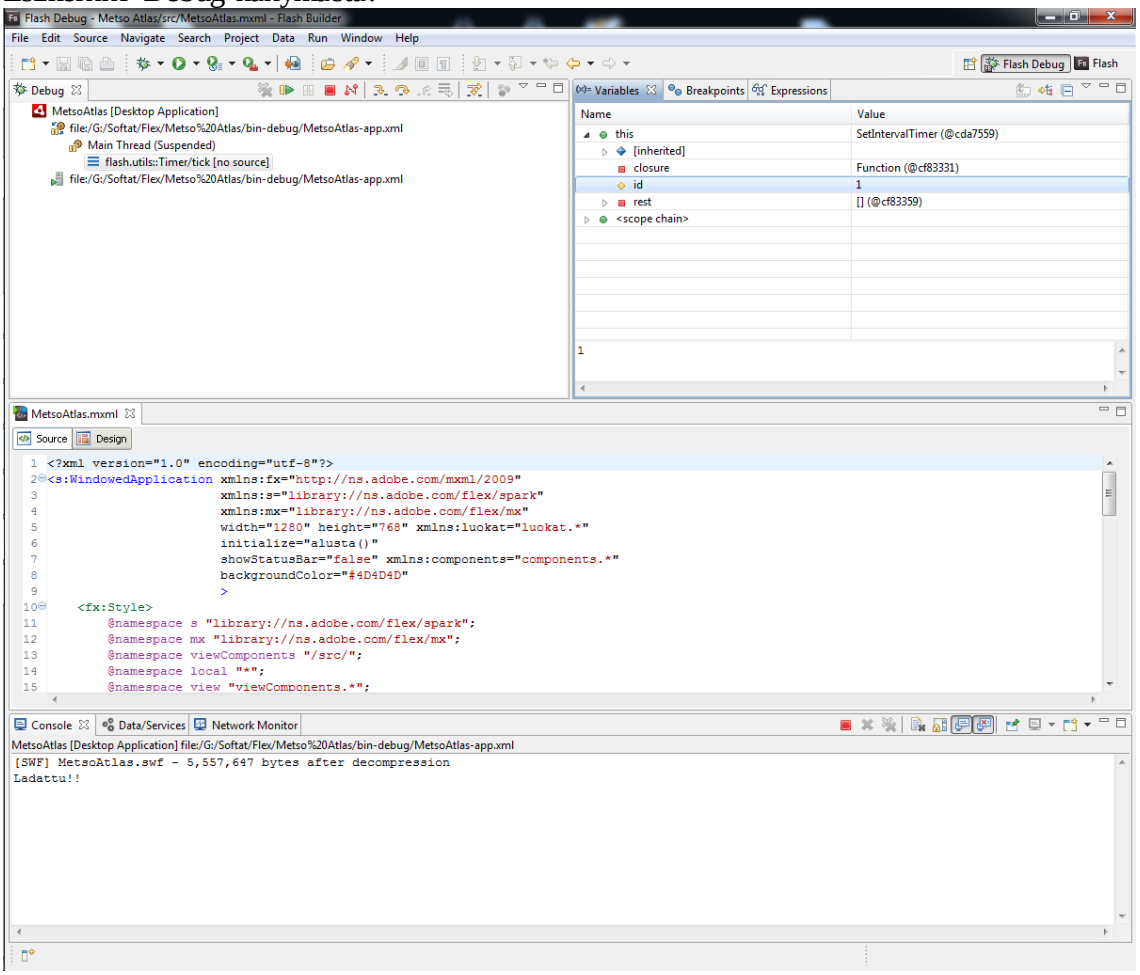
Debug-moodi. Debug-moodin tarkoitus on antaa ohjelmoijalle lisätietoa siitä, mitä sovelluksen sisällä tapahtuu. Ohjelmoija voi laittaa koodiin ns. break point-merkintöjä. Kun ohjelman suoritus saavuttaa rivin jolla on break point, niin suoritus pysähtyy ja debug-näkymä aukeaa. Ohjelman suorituksen voi myös käsin keskeyttää debug-näkymästä haluamallaan hetkellä. Ohjelma jatkaa kyseisestä kohdasta, kun debug-näkymästä painetaan vihreän nuolen näköistä ”play” nappia. Suoritus jatkuu normaalisti kunnes vastaan tulee seuraava break point, tai kunnes käyttäjä keskeyttää ohjelman ajon itse.

Debug-näkymässä näkyy muun muassa kaikkien käytössä olevien muuttujien sisällöt sillä hetkellä ja tämä onkin ylivoimaisesti debug-ominaisuuden yleisin käyttötarkoitus. Flash Builderin debug-toteutus on selkeä ja helposti omaksuttava. Muuttujat löytyvät oikealta ylhäältä olevasta ”Variables” osiosta selkeässä puurakenteessa. Puurakenteen selaaminen toimii samalla tavalla kuin Package Explorerissa.

Debug-näkymässä voi myös suoraan editoida koodia, eli muutokset voi toteuttaa heti. Ohjelma täytyy kuitenkin ajaa uudelleen, että saadaan nämä muutokset käyttöön. Debuggerin koodinäkymä myös pysähtyy sille riville, missä taukopiste on, joten oikeaa kohtaa ei tarvitse etsiä.

Jos ohjelman ajon aikana tapahtuu virhe, niin debug-näkymä pysähtyy automaattisesti siihen kohtaan, missä virhe tapahtui.

Esimerkki Debug-näkymästä:



Kuva 5 Flash Builder - debug-näkymä

Console eli konsoli löytyy Flash Builderin alaosasta, kun ohjelma ajetaan debug-tilassa. Konsoliin tulostuu käyttäjälle näkymättömiä viestejä. Ohjelmoija voi esimerkiksi laittaa viestin tulostumaan konsoliin kun tiedosto on ladattu. Tämä on hyödyllistä, koska ohjelman toimintaa voi tarkkailla ilman että sen suoritusta tarvitsee pysäyttää taukopisteillä. Tämän takia sovellukset tuleekin ajettua käytännössä aina debug-tilassa vaikka taukopisteitä ei edes olisi käytössä.

Konsoliin voi tulostaa rivejä komennolla trace(”haluttu viesti”+muuttuja);.

Projektin kääntäminen lopulliseksi versioksi tapahtuu ylävalikosta Project→Export Release Build. Jos ollaan tekemässä AIR-työpöytäsovellusta, niin pakettiin tulee lisätä sertifikaatti. Jos virallista sertifikaattia ei ole saatavilla, niin sellaisen voi tehdä itse. Lisäksi tulee valita tehdäänkö AIR-paketti vai suora asennusohjelma joka sisältää myös AIR-ajoympäristön. Mahdollista on myös tehdä välitason AIRI-paketti, joka pitää sertifioida myöhemmin.

3.2 Yhteensopivuus

Flex toimii web-puolella kunhan käyttöjärjestelmälle ja käytetylle selaimelle on tuki Flash Playerille.

Flash Playerin tukemat käyttöjärjestelmät:

- Microsoft Windows
- Mac OS X
- Linux
- Solaris
- BlackBerry Tablet OS
- Android
- Pocket PC

Työpöytäsovellukset toimivat Adobe AIR:n tukemissa käyttöjärjestelmissä:

- Microsoft Windows
- Mac OS X
- Android
- iOS
- BlackBerry Tablet OS
- Linux (virallinen tuki lopetettu versiosta 2.6 lähtien)

3.3 Ohjelmointikielet

Adobe Flex käyttää kahta eri kieltä. Varsinainen ohjelmointi tapahtuu käyttäen ActionScript 3.0:a. Graafisten elementtien toteutukseen käytetään MXML-kuvauskieltä. Graafisia elementtejä voi luoda myös suoraan ActionScript 3.0:lla.
(Labriola, Tapper & Boles, Adobe Flex 4.5 Fundamentals, 2012, 13-14.)

Yleisesti ottaen graafiset elementit toteutetaan MXML-kieltä käyttäen ja näihin elementteihin viitataan ActionScript 3.0-koodissa. Jos elementtejä tarvitsee luoda dynaamisesti, niin ne pitää luoda ActionScriptillä.

3.3.1 ActionScript 3.0

ActionScript 3.0 on olio-ohjelmointikieli jonka alun perin kehitti Macromedia Inc. (nykyisin Adobe Systemsin omistuksessa). ActionScript pohjautuu ECMAScriptiin ja on Flash-ympäristön pääohjelmointikieli.

ActionScript 2.0 oli vielä skriptikieli mutta muuttui versio 3.0:n myötä olio-ohjelmointikieleksi ja soveltuu nykyään paljon paremmin monimutkaiseen ohjelmointiin antaen kehittäjälle huomattavasti enemmän hallintaa. AS 3.0 on myös jopa 10 kertaa nopeampi kuin edeltäjänsä.

AS 3.0 on Flex-ohjelmointirajapinnan ydin. (Wikipedia 2012b.)

Hello World-ohjelma toteutettuna ActionScript 3.0-kielellä:

```
var helloLabel:Label = new Label();
helloLabel.text = "Hello World";
this.addChild(helloLabel);
```

Kuva 6 koodiesimerkki AS3-kielestä

ActionScript 3.0 on kaikin puolin huomattava parannus Flash-alustalle ohjelmoijan näkökulmasta. Siirtyminen olio-ohjelmointikieleksi on tuonut Flash/Flex:n varteenotettavaksi työkaluksi ohjelmistotuotantoon ja jopa työpöytäsovelluksiin, kun käytetään Adobe AIR:ia.

ActionScript 3.0:n ”taakkana” on sen skriptikielitausta. Kaikki toiminnot ja käytännöt eivät aina toimi siten kuin muissa ohjelmointikielissä mutta tämä on enimmäkseen vain totuttelukysymys. Joillekin olio- ja luokkaohjelmointikäytännöille ei ole suoraa tukea, kuten esimerkiksi Singleton-luokat.

3.3.2 MXML

MXML on XML-pohjainen kuvauskieli käyttöliittymän toteuttamiseen. MXML:n nimelle ei ole annettu virallista selitystä mutta todennäköisesti se on lyhenne sanoista ”Macromedia eXtensible Markup Language”.

MXML on syntaksiltaan hyvin selkeä ja sen lukeminen ei vaadi kovin suurta ymmärrystä ohjelmoinnista. MXML:n tarkoitus on määrittää graafisia elementtejä selkeällä XML-pohjaisella kuvauskielellä.
(Labriola, Tapper & Boles, Adobe Flex 4.5 Fundamentals, 2012, 13.)

Hello World-ohjelma toteutettuna MXML-kielellä:

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml">
  <mx:Label text="Hello World!" />
</mx:Application>
```

Kuva 7 koodiesimerkki MXML-kielestä

MXML-komponentteihin voi viitata suoraan AS3.0-koodin puolelta, kunhan kyseisille MXML-komponenteille on annettu id. MXML-komponentit eivät ole staattisia, joten niitä voi muokata AS3.0:n puolella tehdyillä komennoilla viitaten halutun komponentin id-nimeen.

Jos joitain osia käyttöliittymästä täytyy luoda ajon aikana dynaamisesti, niin tämä ei ole mahdollista MXML:llä. Tälläiset komponentit tulee luoda kokonaan AS3.0:n puolella. Tämä on tehty hyvin helpoksi ja AS3.0:n puolella voi tehdä komponentteja dynaamisesti ja lisätä ne haluamaansa kohtaan käyttöliittymää.

3.4 Yhteisön taso ja levinneisyys

Flash/Flex on alustana erittäin suosittu ja löytyy suurimmasta osasta tietokoneita. Johtuen suuresta suosiosta myös kehitysyhteisö on erinomaisella tasolla ja verkko onkin täynnä ohjeita ja laajennuksia alustalle.

3.5 Työpöytäsovellukset

Työpöytäsovellukset Flexillä toteutetaan Adobe AIR:n kautta. Käyttäjän tarvitsee asentaa AIR-ajoympäristö koneelleen, että Flex:llä tehdyt työpöytäsovellukset toimivat. Sovellukset ovat alustariippumattomia eli sama sovellus toimii kaikissa järjestelmissä mihin on asennettu AIR-ajoympäristö. Tämä tietenkin olettaen, että ohjelmoija pysyy hyvissä ohjelmointikäytännöissä eikä esimerkiksi käytä suoria polkuviittauksia, jotka tietenkin eroavat eri järjestelmien välillä.

4 MICROSOFT SILVERLIGHT

Tässä osiossa käydään Microsoft Silverlight tarkemmin läpi tekniikkana, sekä tutkitaan sen soveltuvuutta yleisesti kehitysalustaksi.

4.1 Visual Studio

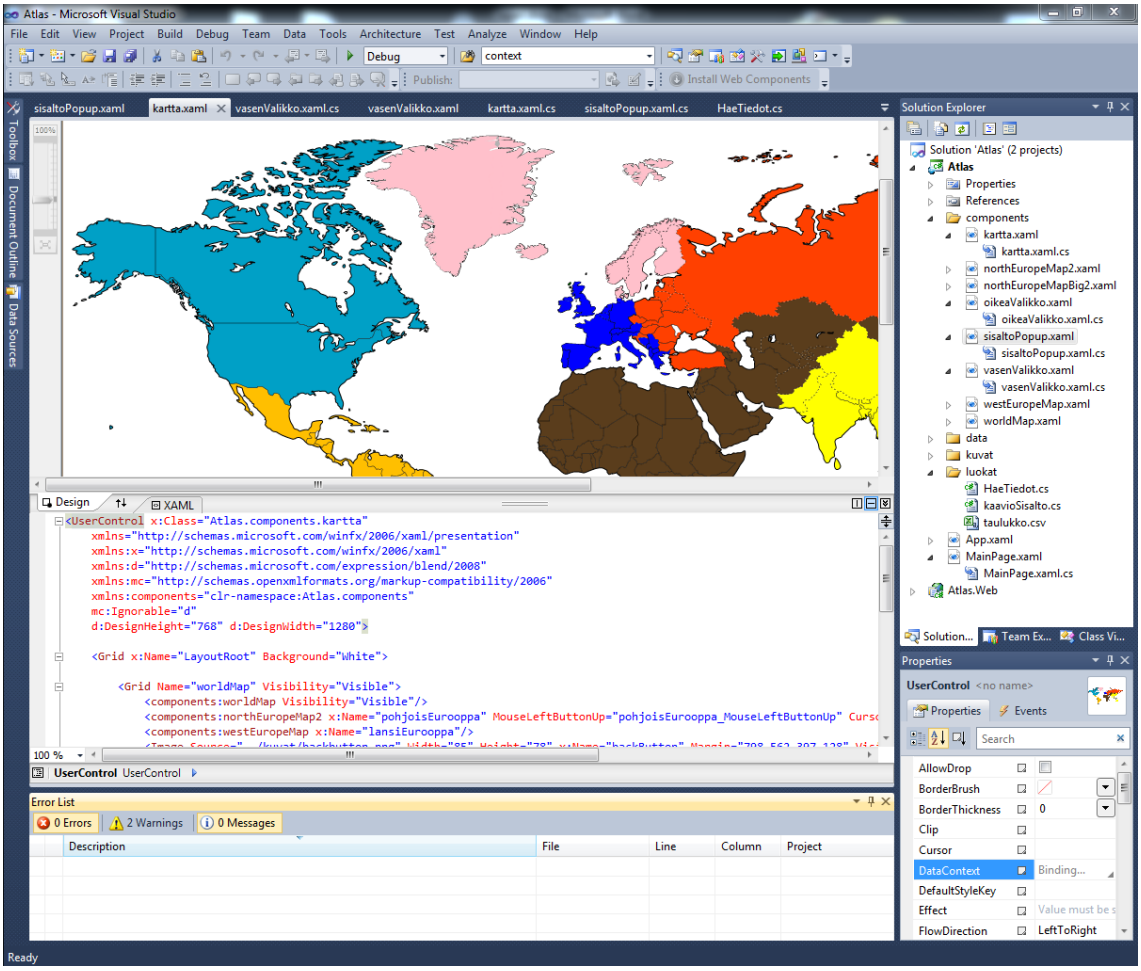
Käytetyn kehitysympäristön helppokäyttöisyys ja toiminnallisuudet ovat erittäin tärkeitä kriteereitä, kun valitaan eri ohjelmointialustojen välillä. Tässä osiossa tutustutaankin tarkemmin Silverlight-alustan ensisijaiseen kehitysympäristöön, eli Microsoft Visual Studioon.

Vaikka Visual Studio on Silverlightin pääasiallinen kehitysympäristö, niin on huomionarvoista, että Silverlight tukee kaikkia .NET-kieliä ja tästä johtuen kehitysympäristönä voi periaatteessa käyttää mitä tahansa .NET kielen kehitysalustaa, kunhan se tukee Silverlight SDK:ta.

Graafisen ulkoasun suunnitteluun löytyy Microsoft Expression Blend, jota voi verrata Adoben Catalystiin. Expression Blendin yleisin käyttötarkoitus on projekteissa, joissa ohjelmoinnin ja graafisen puolen toteuttavat eri

henkilöt. Jos ohjelmoija tekee itse myös ulkoasun, niin näistä ohjelmista ei kovin suurta lisäarvoa löydy.

4.1.1 Käyttöliittymä



Kuva 8 Visual Studio - käyttöliittymä

Kuvassa 8 on Microsoft Visual Studion normaali työskentelynäkymä kun editoidaan XAML-tiedostoa. Yleisilme yhdellä näytöllä on ahdas ja hieman sekava. Käytännössä tila loppuu kesken. Eri osioita voi onneksi irrottaa ja siirtää toiseen näyttöön. Tehokasta työskentelyä varten tarvitsee käytännössä kaksi näyttöä tai yhden kooltaan ja resoluutioltaan suuren näytön.

Visual Studio avaa oletuksena yhden projektin kerrallaan yhteen ikkunaan. Jos käyttäjä avaa toisen projektin, niin se avautuu kokonaan uuteen Visual Studio-ohjelmaan.

Oikeassa yläreunassa käyttöliittymää on Solution Explorer. Sen sisältä löytyy kyseisen projektin kaikki tiedostot hakemistorakenteineen.

Tiedostot ovat selkeässä puurakenteessa. Puun juuri on projektin nimi eli esim. ”Atlas” ja kun projektin puun avaa sen edellä olevasta kolmiosta tulee näkyviin projektin hakemistorakenne. Oletushakemistot ovat seuraavat:

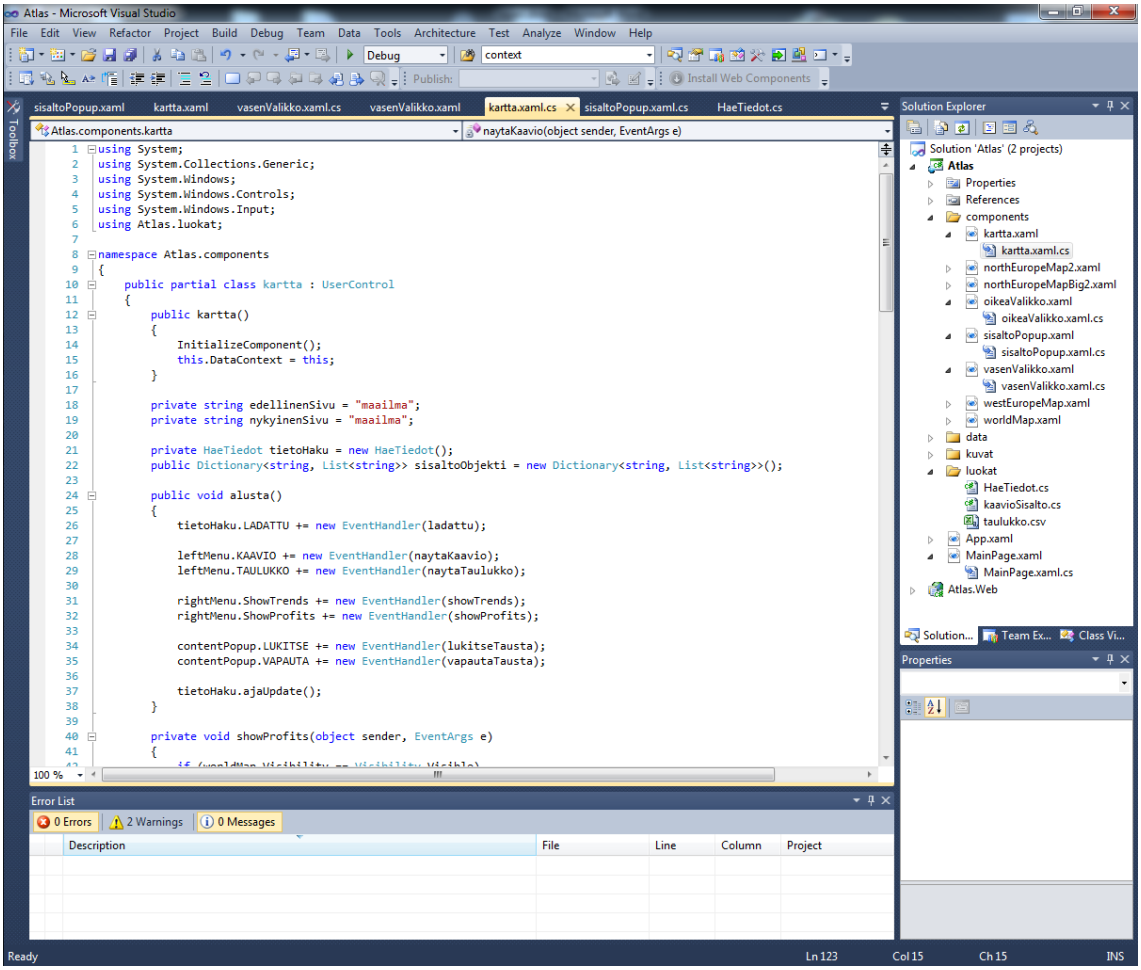
- Properties

- Sisältää projektiin liittyviä asetus-tiedostoja
- References
  - Sisältää projektin käyttämät viittaukset. Silverlight-projektiin tarvitsee yleensä lisätä käsin tarvittavia viittauksia, koska kaikki kirjastot yms. eivät ole oletuksena mukana.

Solution Explorerin vasemmalla puolella on Visual Studion varsinainen työskentelytila. XAML-tiedostoa editoidessa ylimpänä tässä tilassa on graafinen näkymä, josta näkee suoraan miltä kyseinen komponentti näyttää. Tästä näkymästä voi myös editoida ulkoasua.

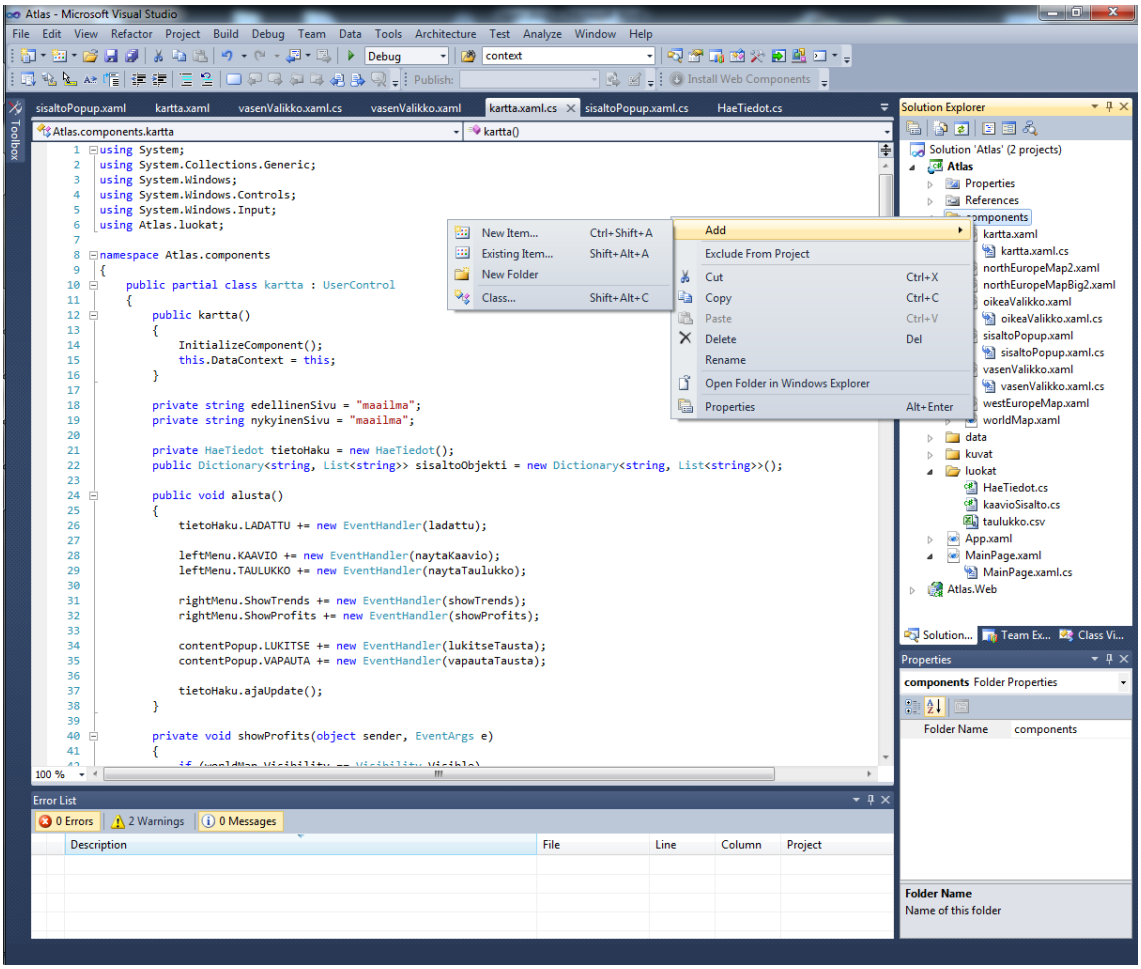
Visual Studion työkalut ohjelman UI:n toteutukseen ovat hyvin selkeät ja toimivat erinomaisesti. Ulkoasu-editorin alapuolella on kyseisen tiedoston koodinäkymä, josta voi suoraan editoida XAML-koodia.

Näiden alapuolella on oletuksena piilotettu osio, jossa näkyvät esimerkiksi konsoli-tulosteet ja virheilmoitukset.



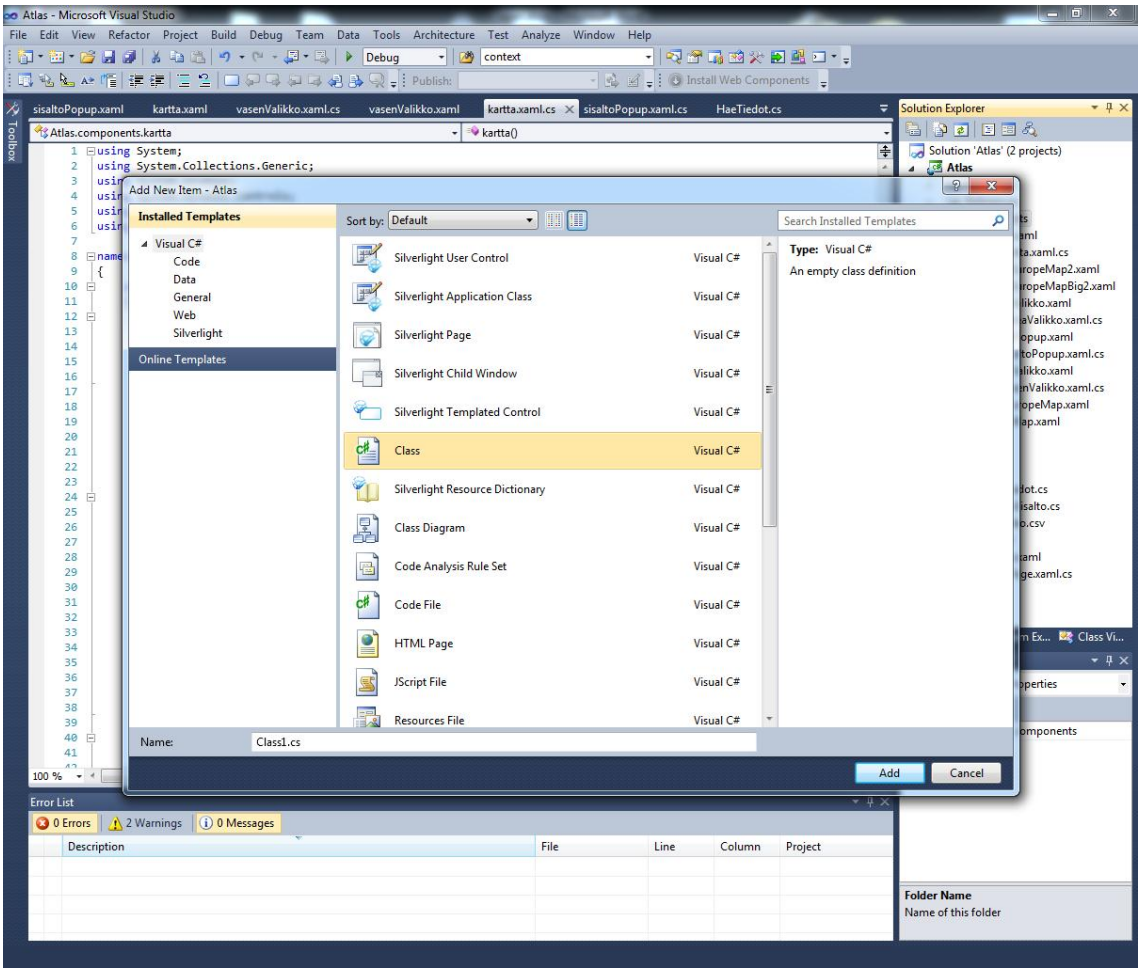
Kuva 9 Visual Studio - käyttöliittymä ilman graafista näkymää

Kuvassa 9 on Visual Studion normaali työskentelynäkymä, kun editoidaan normaalia .cs kooditiedostoa. Näkymä on täsmälleen samanlainen kuin XAML-tiedostoa editoitaessa lukuun ottamatta ulkoasu-editoria, mitä ei tässä näkymässä ole.



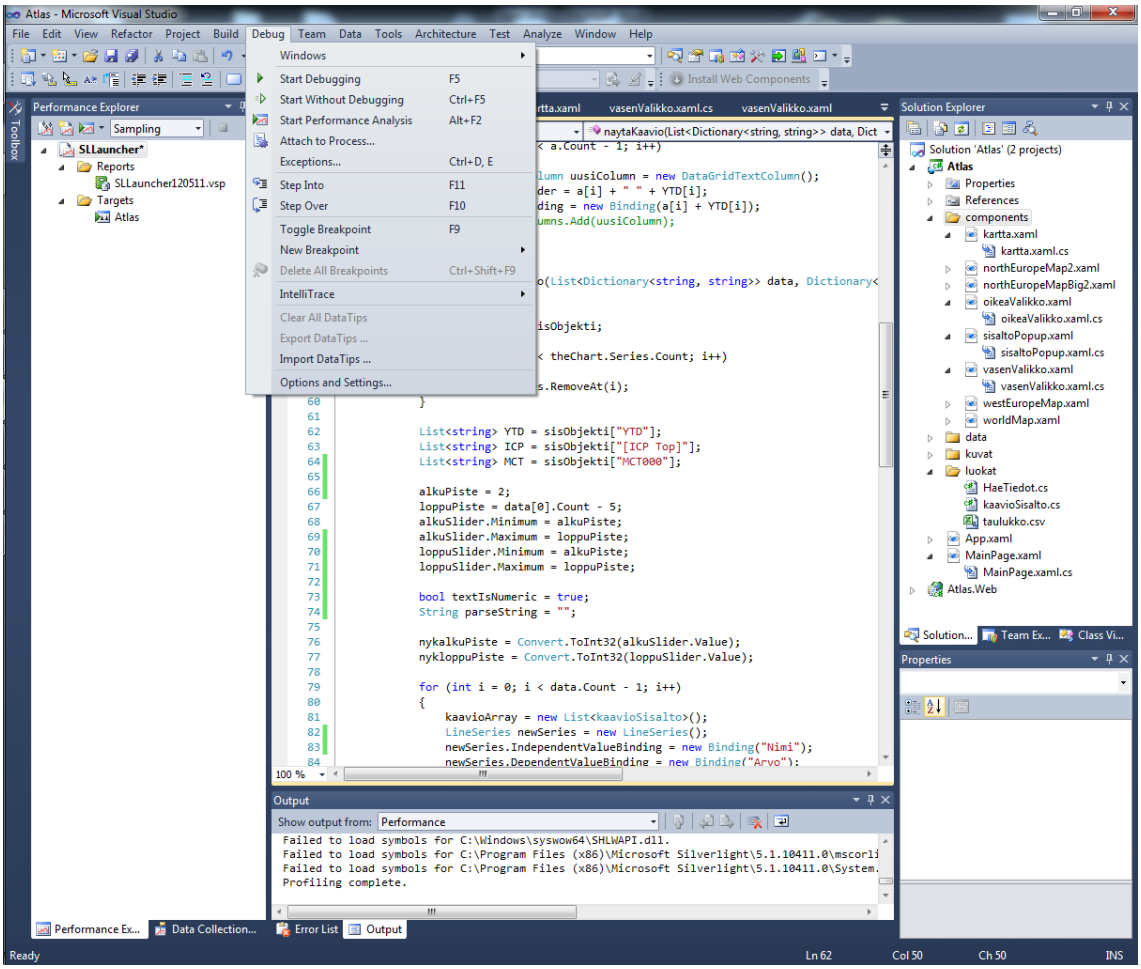
Kuva 10 Visual Studio - uuden tiedoston lisääminen

Uusien tiedostojen tai kansioiden lisääminen onnistuu helposti painamalla hiiren oikeaa nappia Solution Explorerissa haluamassaan kohdassa. Tämä avaa uuden ikkunan, josta voi lisätä uusia tiedostoja ja tehdä myös muita tiedostoihin liittyviä toimintoja kuten kopiointi, leikkaus, liittäminen ja niin edelleen. Kun valikosta valitaan Add→New Item, niin aukeaa kuvassa 11 näkyvä popup-ikkuna.



Kuva 11 Visual Studio - uuden tiedoston lisääminen(2)

Tästä pop-up ikkunasta valitaan haluttu tiedostotyyppi. Alhaalla olevaan ”Name”-syötekenttään annetaan tiedoston haluttu nimi.



Kuva 12 Visual Studio - Debug-valikko

Kuvassa 12 näkyy Debug-valikko avattuna. Tämä toimii esimerkkinä ohjelman ylävalikoiden toiminnasta. Valikoiden toiminnot käydään tarkemmin läpi seuraavassa osiossa(4.1.2).

4.1.2 Työkalut ja ominaisuudet

Visual Studio on erittäin monipuolinen ja tehokas kehitysympäristö. Tässä osiossa käydään läpi tärkeimmät työkalut ja ominaisuudet mutta kaikkia toiminnallisuuksia ei ole tarkoituksenmukaista käydä läpi tämän opinnäytetyön puitteissa. Ominaisuuksien lisäksi käydään läpi havaittuja puutteita ja erityisen hyvin toimivia asioita.

Visual Studion monipuolisuus on sen suurin etu mutta myös yksi sen suurimmista haitoista. Ympäristö on kieltämättä erittäin tehokas ja siitä löytyy työkalut käytännössä kaikkiin mahdollisiin asioihin. joita ohjelmistokehityksessä tarvitsee. Varjopuolena tämä tekee siitä ulkoasultaan paljon sekavamman ja aloituskynnys on huomattavan paljon korkeammalla kuin esim. Flash Builderissä. Tämä ei ole kuitenkaan kovin suuri ongelma, koska Visual Studio on kehitysalustana de facto-standardi kaikessa Windows-ohjelmoinnissa ja .NET kielissä.

Visual Studion ylävalikot sisältävät valtaosan ympäristön toiminnoista. Ne käydään seuraavaksi yleisellä tasolla läpi, koska kaikkien ominaisuuksien kattava läpikäynti olisi laajuudeltaan jo erillinen opinnäytetyö.

Huomionarvoista on se, että ylävalikot muuttuvat sen mukaan minkälainen tiedosto on kyseisellä hetkellä avoinna.

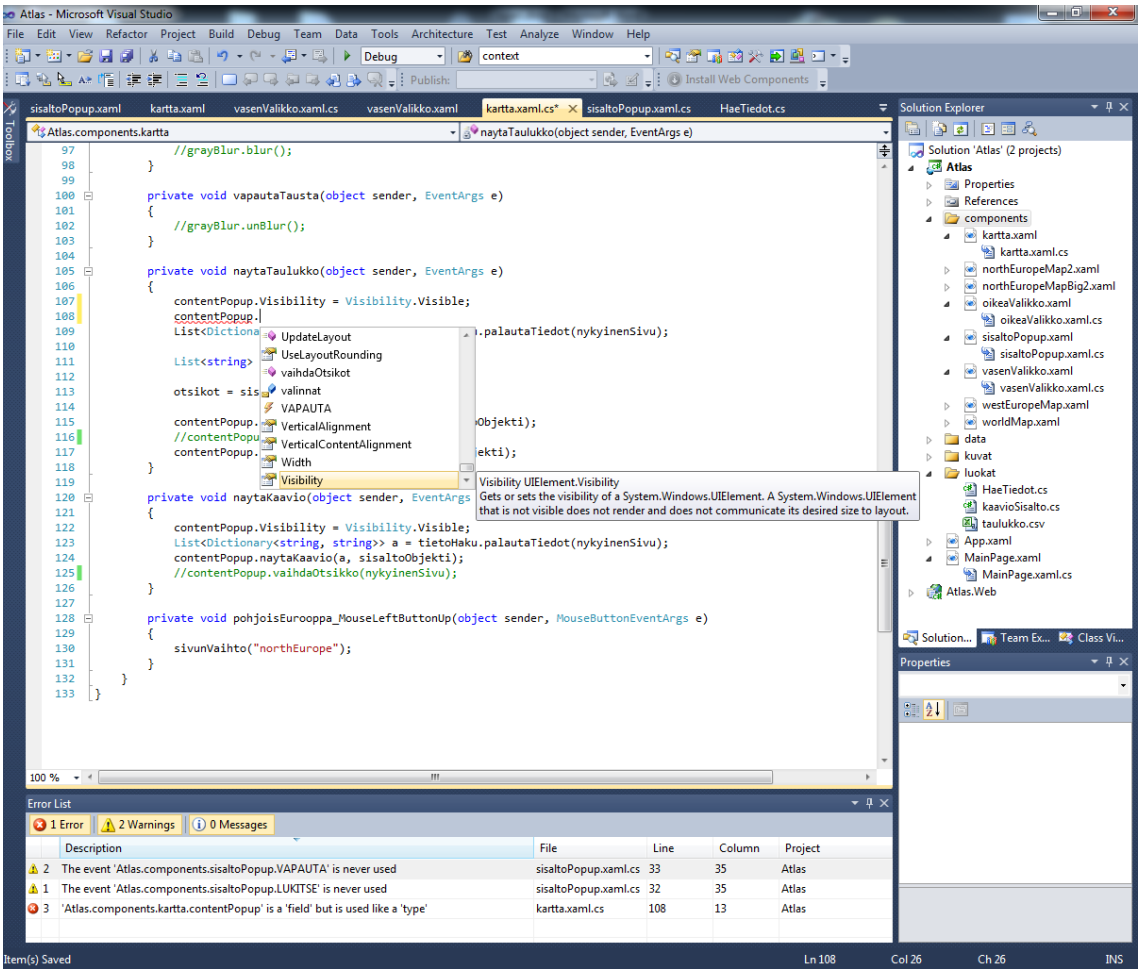
Visual Studion ylävalikot:

- File
  - Yleisiä toiminnallisuuksia mitä File-valikoista yleensäkin löytyy. Esim. ”New”, ”Open”, ”Close”, ”Save” jne.
- Edit
  - ”Undo”, ”Redo”, ”Copy”, ”Paste” ja muita vastaavia toiminnallisuuksia.
- View
  - Visual Studion käyttöliittymän eri osia saa näkyviin tai piiloon tästä valikosta. Läheskään kaikki käyttöliittymän osat eivät ole oletuksena käytössä.
- Refractor

  Ei näkyvissä XAML-tiedostoja editoitaessa. Työkaluja koodin uudelleennimeämiseen, kapselointiin, parametrien uudelleenjärjestelyyn jne.
- Project
  - Projektiin liittyviä toimintoja. Oikeastaan tarpeeton koska kaikki nämä onnistuvat myös suoraan Solution Explorerista.
- Build
  - Sovelluksen kääntäminen tapahtuu täältä.
  - Varsinkin ”Rebuild Solution” ja ”Clean Solution” ovat jatkuvassa käytössä, koska korjatut virheet eivät aina kuittaudu korjatuiksi ellei projektia käännä uudelleen.
- Debug
  - Debug-ympäristön toimintoja.
  - ”Start Debugging” käynnistää ohjelman debug-tilassa.
  - ”Start Without Debugging” ajaa ohjelman ilman debug-tilaa.
  - “Start Performance Analysis” tarkkailee miten sovellus käyttää järjestelmäresursseja ja luo havainnoistaan raportin kun sovellus suljetaan.
- Team
  - Team Foundation Server:iin liittyviä asioita.
- Data
  - Data-lähteiden hallintaa. Käytännössä SQL-tietokanta-asetuksia.
- Format
  - ”Bring to Front” ja ”Send to Back”. Vaihtavat valitun XAML-komponentin z-syvyyttä.
- Tools
  - Paljon erilaisia työkaluja, makroja yms.
- Architecture
  - Työkaluja ohjelmistosuunnitteluun, esim. UML-kaavio.

- Test
  - Sisältää toiminnallisuuksia sovelluksen testaamiseen.
- Analyze
  - Sisältää työkaluja sovelluksen analysointiin ja profilointiin.
- Window
  - Sisältää valitun välilehden asetuksia. Esim. mahdollisuus irrottaa välilehti omaksi ikkunakseen.
  - Optio palauttaa UI oletusnäkymään.
- Help
  - Yleinen help-valikko.

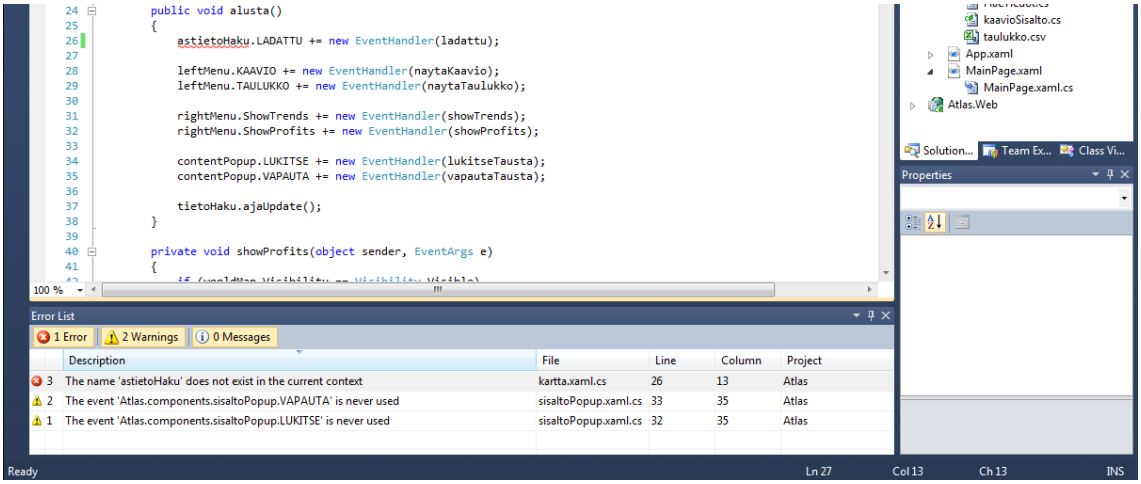
Ylävalikkojen alla on useampi rivi pikakuvakkeita. Nämä ovat kaikki valikoista löytyviä eniten käytettyjä toimintoja, joten niitä ei käydä läpi sen tarkemmin.



Kuva 13 Visual Studio - esimerkki kooditäydennyksestä

Kuvassa 13 näkyy Visual Studion kooditäydennys-toiminnallisuus. Kooditäydennys nopeuttaa ohjelmointia huomattavasti, koska se ehdottaa automaattisesti ratkaisua eikä kaikkea tarvitse kirjoittaa. Samoin eri komponenttien parametrit löytyvät helposti kooditäydennyksen avulla. Listan oikealla puolella näkyy myös valitun vaihtoehdon lyhyt kuvaus. Tässä suhteessa Visual Studio häviää Flash Builderille, koska Flash Builderissä avautuu kokonaan luettavissa oleva versio kyseisen kohdan dokumentaatiosta.

Kooditäydennyksen voi aktivoida myös manuaalisesti painamalla Ctrl+Välilyöntiä.

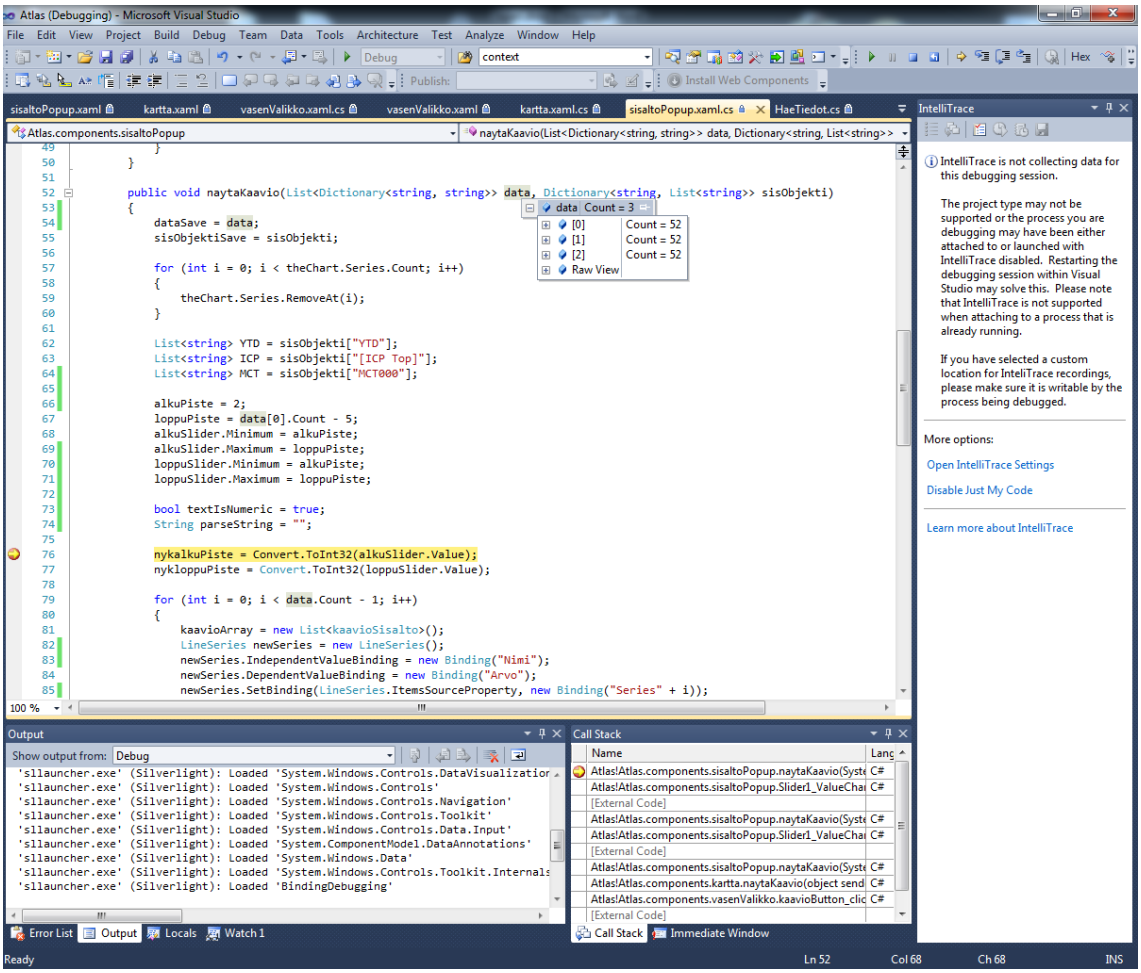


Kuva 14 Visual Studio – reaaliaikainen virheentunnistus

Kuvassa 14 näkyy esimerkki reaaliaikaisesta virheentunnistuksesta. Jos ohjelmoija tekee esimerkiksi kirjoitusvirheen, niin ympäristö havaitsee sen ja alleviivaa virheen, sekä lisää virhe-ilmoituksen Error Listiin joka löytyy ohjelman alareunasta. Visual Studio tosin ei käännä projektia automaattisesti uudestaan eli monet korjatut virheet eivät kuittaudu korjatuiksi ennen kuin ohjelmoija kääntää ohjelman uudelleen.

Visual Studion reaaliaikainen koodintarkistaja ei läheskään aina reagoi muutoksiin suoraan, vaan ohjelma pitää kääntää uudelleen, että koodintarkastajakin päivittyy. Varsinkin korjatut virheet jäävät kummittelemaan ja pysyvät virheiksi merkittyinä mikä voi aiheuttaa sekaannusta.

Käyttöliittymässä on paljon hyviä näppäinoikoteitä, jotka nopeuttavat työskentelyä huomattavasti. Monet niistä ovat ctrl+k:n takana. Esimerkiksi ctrl+k→ctrl+c kommentoi valitut koodirivit ja ctrl+k → ctrl+u poistaa valitut kommentit.



Kuva 15 Visual Studio - debug-näkymä

Kuvassa 15 on Visual Studion Debug-näkymä. Debug-näkymän tarkoitus on antaa ohjelmoijalle työkalu pysäyttää sovelluksen ajo halutuissa kohdissa ja tutkia esimerkiksi muuttujien sisältöä. Visual Studion debug-tilassa riittää, että vie hiiren minkä tahansa muuttujan tai vastaavan päälle. Ympäristö avaa listan samaan kohtaan mistä näkee muuttujan sen hetkisen tilan ja sisällön.

4.2 Yhteensopivuus

Toimii web-puolella kunhan käyttöjärjestelmälle ja käytetylle selaimelle on tuki Silverlightille.

Silverlightin tukemat käyttöjärjestelmät (sekä Web, että työpöytäohjelmat):

- Microsoft Windows
- Mac OS X
- Windows Phone
- Symbian OS
- Linux (Moonlight)

4.3 Ohjelmointikielet

Silverlightiä voi ohjelmoida millä tahansa .NET ohjelmointikielellä. Lista yleisemmistä .NET ohjelmointikielistä:

- A#
- Boo
- C#
- C++/CLI
- Cobra
- Component Pascal
- F#
- IronPython
- IronRuby
- IronLisp
- J#
- JScript .NET
- L#
- Managed Extensions for C++
- Managed JScript
- Nemerle
- Oxygene
- P#
- Phalanger
- Phrogram
- PowerBuilder
- Team Developer
- VBx
- VB.NET
- Windows PowerShell

Näistä yleisimmin käytössä ovat C# ja VB.NET.

4.3.1 C#

Kaikkien .NET-kielien läpikäymiseen menisi sivumääräisesti enemmän, kuin koko tähän opinnäytetyöhön, joten käytämme alustan suosituinta kieltä esimerkkinä.

C# tai C Sharp on Silverlightin kanssa eniten käytetty ohjelmointikieli, joka kehitettiin Microsoftin .NET-konseptia varten. Lähtökohtana oli yhdistää C++-kielen tehokkuus ja Java-kielen helppokäyttöisyys.

C# on moderni yleiskäyttöinen ohjelmointikieli.

Hello World-ohjelma toteutettuna C#-kielellä:

```
using System;

class Program
{
    static void Main()
    {
        Console.WriteLine("Hello world!");
    }
}
```

Kuva 16 koodiesimerkki C#-kielestä

(Wikipedia 2012c.)

C# on yksi maailman käytetyimmistä ohjelmointikielistä ja soveltuu käytännössä kaikkeen ohjelmistotuotantoon. Toisin kuin ActionScript, C# on kehitetty alusta alkaen olio-ohjelmointikieleksi. Tämän lisäksi C# on kuusi vuotta vanhempi ja siltä osin kypsempi ohjelmointikieli.

Tämä ei tosin ole etu kaikissa asioissa. Esimerkiksi arrayt eivät ole dynaamisia lainkaan. Niiden sijaan pitää käyttää jotain muuta toiminnallisuutta kuten List tai Dictionary.

Olio- ja luokamallien ominaisuudet ovat tuettuna, eikä esimerkiksi singleton-luokkien toteutus vaadi mitään erikoisia ”patentteja” kuten AS3.0:n puolella.

4.3.2 XAML

Silverlight käyttää kuvauskielenä XML-pohjaista XAML-kieltä(Extensible Application Markup Language), joka on hyvin samankaltainen XHTML:n ja Adoben MXML:n kanssa.

Jokainen elementti XAML-tiedostossa on yhteydessä Silverlight-luokan instanssiin. Elementin nimi vastaa luokan nimeä täsmälleen. Esimerkiksi elementti <Button> ohjeistaa Silverlightiä luomaan Button-objektin. (MacDonald, Pro Silverlight 4 in C#. 2010, 33-34.)

Tässä osiossa keskitytään myös Silverlightin omiin komponentteihin.

Hello World-ohjelma toteutettuna XAML-kielellä:

```
<?xml version=”1.0″ ?>
  <Canvas xmlns=”http://schemas.microsoft.com/client/2007″>
    <TextBlock Text=”Hello World!” />
  </Canvas>
```

Kuva 17 koodiesimerkki XAML-kielestä

Valmiista Silverlight-komponenteista löytyi puutteita, eivätkä kaikki toiminnallisuudet onnistuneet yhtä helposti kuin Flexillä. Esimerkiksi Image-komponentille on pakko määritellä koko käsin, koska oletusarvoisesti ympäristö skaalaa sen niin suureksi kuin mahdollista. Tämä on vain pieni ärsyttävyys mutta tämänlaiset asiat hidastavat kehitystyötä yllättävän paljon.

Toinen ongelma on x ja y-koordinaattien puute. Näiden sijasta kaikki komponentit tarvitsee asetella paikalleen marginaaleja käyttäen. Tämä ei ole ongelma, jos käytetään Visual Studion omaa työkalua UI-suunnitteluun.

Silverlightin DataGrid-komponentissa on huomattava puute. Se on suunniteltu toimimaan staattisena, eli ohjelman tulee tietää otsikkojen nimet etukäteen. DataGridin sisältö järjestäytyy näiden otsikkojen mukaan. Esimerkiksi kaksiulotteisen taulukon antaminen sisällöksi ei toimi. Suoraa tukea täysin dynaamiselle DataGridille ei siis ole. Atlas-sovelluksen perusedellytys on täysin dynaaminen DataGrid, joten tämä puute esti Atlas-sovelluksen toteuttamisen käyttäen pelkästään Silverlightin omia komponentteja ja ominaisuuksia.

Tämä ei tietenkään tarkoita ettei Silverlightilla voi tehdä dynaamista DataGridia mutta puuttuvat toiminnallisuudet pitää toteuttaa itse tai tarvitsee turvautua kolmannen osapuolen tekemään toteutukseen. Molemmat vaihtoehdot voivat olla erittäin työläitä.

4.4 Yhteisön taso ja levinneisyys

Silverlight ei ole alustana läheskään yhtä levinnyt kuin Flash mutta yhteisö on erittäin hyvällä pohjalla johtuen siitä, että Silverlight käyttää .NET-ohjelmointikieliä. Lisäksi tulee ottaa huomioon, että Silverlight on Windows Phone-mobiilikäyttöjärjestelmän kehitysalusta. Tämä fakta jo yksinään riittää perusteluksi Silverlightin opetteluun ja käyttöön.

4.5 Työpöytäsovellukset

Silverlight tukee suoraan työpöytäsovelluksia, eikä erillistä ajoympäristöä Silverlightin lisäksi tarvitse asentaa.

5 METSO ATLAS-ESIMERKKISOVELLUS

Tässä osiossa on Metso Atlas-sovellus olennaisilta osiltaan tehtynä Flexillä sekä Silverlightilla käyttäen vain kyseisten tekniikoiden sisältämiä komponentteja ja ominaisuuksia. Eli ei lähdetä käyttämään kolmansien osapuolien ratkaisuja, koska tarkoitus on nimenomaan selvittää molempien SDK-pakettien vahvuudet ja heikkoudet.

Alustan komponenttien soveltuvuus, mukautuvuus ja helppokäyttöisyys ovat ensiarvoisen tärkeitä kriteereitä, kun mitataan alustan tuotantotehokkuutta. Ohjelmointi-projekti valmistuu huomattavasti nopeammin, kun voi

käyttää jo olemassa olevia komponentteja ja ominaisuuksia. Jos alustasta puuttuu joku tarvittava komponentti tai sen toteutus ei vastaa sovelluksen tarpeita, niin kehitysajat kasvavat moninkertaisiksi hyvin nopeasti. Tästä johtuen Atlas toteutetaan käyttäen pelkästään alustojen omia komponentteja ja toiminnallisuuksia.

Sisältötiedostoja, kuvia ja muita vastaavia ei sisällytetä tähän esimerkkiin. Osassa 5.3 käydään läpi esiin tulleet eroavuudet ja mahdolliset ongelmat.

Metso Atlas-sovelluksen tarkoitus on lukea annetusta CSV-tiedostosta esim. talouslukuja ja näyttää ne kartalla graafisessa muodossa. Lisäksi ohjelman avulla voi katsella dataa normaalissa taulukkomuodossa tai erilaisina graafisina käyrinä. Ohjelma suodattaa datan automaattisesti vastaamaan valittua aluetta kartalta.

Silverlight-sovelluksessa keskityttiin pelkästään toteuttamaan vastaavat toiminnallisuudet olennaisilta osilta, koska kehityksen aikana havaittiin että tämänlaista sovellusta ei pysty järkevästi toteuttamaan Silverlight 4.0:n omilla komponenteilla ja ominaisuuksilla. Lisätietoja tästä asiasta löytyy osiossa 5.3.

5.1 Sovellus Adobe Flex:llä

5.1.1 MetsoAtlas.mxml

Sovelluksen päätiedosto. Ei itsessään sisällä paljoa koodia mutta toimii kehyksenä sovellukselle ja alustaa tarvittavat toiminnot.

Lähdekoodi löytyy liitteestä 1.

5.1.2 kartta.mxml

Tämä komponentti sisältää ohjelman pääkäyttöliittymän ja ulkoasun.

Lähdekoodi löytyy liitteestä 1.

5.1.3 oikeaValikko.mxml

Tämä komponentti sisältää oikealla olevan valikon ulkoasun ja toiminnallisuuden.

Lähdekoodi löytyy liitteestä 1.

5.1.4 vasenValikko.mxml

Tämä komponentti sisältää vasemmalla olevan valikon ulkoasun ja toiminnallisuuden.

Lähdekoodi löytyy liitteestä 1.

5.1.5 sisaltoPopup.mxml

Tämä komponentti sisältää popup-ikkunan ulkoasun ja toiminnallisuuden.

Lähdekoodi löytyy liitteestä 1.

5.1.6 HaeTiedot.as

Tämä luokka vastaa CSV-tiedoston parseroimisesta ja tietokantatoiminnoista.

Lähdekoodi löytyy liitteestä 1.

5.1.7 MetsoAtlas-app.xml

AIR-asetustiedosto, jossa määritellään ohjelman yleisiä asetuksia AIR-ympäristöä varten.

Lähdekoodi löytyy liitteestä 1.

5.2 Sovellus Microsoft Silverlightillä

Silverlight erottaa jokaisen XAML-tiedoston koodin omaksi tiedostokseen. Esim. kartta.xaml tiedoston koodi-tiedosto on kartta.xaml.cs. XAML-tiedosto sisältää UI-komponenttien määrittelyt ja .cs-tiedostossa on kyseisen komponentin koodi C#-kielellä. Selkeyden takia tässä osiossa ei erikseen mainita XAML-tiedostojen omia .cs tiedostoja.

Sen sijaan alla listatut .cs-tiedostot ovat erikseen tehtyjä luokka-tiedostoja.

5.2.1 MainPage.xaml

Sovelluksen päätiedosto. Ei itsessään sisällä paljoa koodia mutta toimii kehyksenä sovellukselle ja alustaa tarvittavat toiminnot.

Lähdekoodi löytyy liitteestä 2.

5.2.2 kartta.xaml

Tämä komponentti sisältää ohjelman pääkäyttöliittymän ja ulkoasun.

Lähdekoodi löytyy liitteestä 2.

5.2.3 oikeaValikko.xaml

Tämä komponentti sisältää oikealla olevan valikon ulkoasun ja toiminnallisuuden.

Lähdekoodi löytyy liitteestä 2.

5.2.4 vasenValikko.xaml

Tämä komponentti sisältää vasemmalla olevan valikon ulkoasun ja toiminnallisuuden.

Lähdekoodi löytyy liitteestä 2.

5.2.5 sisaltoPopup.xaml

Tämä komponentti sisältää popup-ikkunan ulkoasun ja toiminnallisuuden.

Lähdekoodi löytyy liitteestä 2.

5.2.6 HaeTiedot.cs

Tämä luokka vastaa CSV-tiedoston parseroimisesta ja tietokantatoiminnoista.

Lähdekoodi löytyy liitteestä 2.

5.2.7 kaavioSisalto.cs

LineChart-kaaviota varten tarvittava luokka, jossa määritellään kuvaajan datarakenne. Kaavioon tuleva data databindataan tässä luokassa määriteltyihin muuttujiin.

5.3 Havaitut erot Flexin ja Silverlightin välillä

DataBinding toimii huomattavasti paremmin ja selkeämmin Flex:ssä. Flex:n puolella DataBinding-toteutus vaatii parhaimmillaan ainoastaan sen, että kerrotaan komponentille mikä sen databind-kohde on ja sen jälkeen merkitään kyseinen kohde [Bindable]-merkinnällä. Muita toimenpiteitä ei yleensä tarvita ja alusta huolehtii kaikesta taustalla.

Silverlight:n toteutus on kömpelömpi. Se toimii periaatteessa samalla tavalla mutta jos data muuttuu, niin komponentti ei suoraan päivity vastaamaan uutta sisältöä. Tämä toiminnallisuus tulee toteuttaa itse käyttäen INotifyPropertyChanged-luokkaa. Käytännössä tarvitsee siis toteuttaa erillisiä luokkia jotka huolehtivat siitä, että databindaukset oikeasti toimivat. Tämä on huomattavasti monimutkaisempaa ja muuttaa sekuntien työn Flexillä minuuttien työksi Silverlightillä.

Ehdottomasti tärkein havainto on se, että Atlas-sovellusta ei ole mahdollista toteuttaa Silverlight-ympäristössä käyttäen sen omia komponentteja ja ominaisuuksia. Silverlightin DataGrid on suunniteltu siten, että otsikoiden(columns) tulee olla tiedossa etukäteen ja sisältö databindataan näihin otsikoihin. Atlas-sovelluksen perusedellytys on täysin dynaaminen DataGrid, koska kaikki tiedot haetaan ulkoisesta CSV-tiedostosta, eikä sen rakenteesta tai sisällöstä ole varmuutta. Eli ohjelma ei voi tietää DataGridin otsikkoja tai sisältöä etukäteen.

Kyseisen toteutuksen voi tietenkin ohjelmoida itse tai käyttää kolmannen osapuolen tekemää DataGrid-toteutusta. Tämä ei kuitenkaan ole tarkoituksenmukaista tämän opinnäytetyön kannalta, koska tarkoituksena on vertailla Flexin ja Silverlightin välisiä etuja ja haittoja. Molemmilla alustoilla voi toteuttaa melkein minkä tahansa toiminnallisuuden, jos se suunnitellaan ja toteutetaan itse puhtaalta pöydältä.

Kolmannen osapuolen toteutuksen opettelu tai koko toiminnallisuuden toteuttaminen tyhjästä ovat kuitenkin todella paljon työläämpiä vaihtoehtoja joten tämä on pakko merkitä puutteeksi Silverlightille.

Flexillä sovelluksen toteutus onnistuu helposti suoraan sen omilla komponenteilla.

YHTEENVETO

Kumpaakaan alustaa ei ole mahdollista julistaa yksiselitteisesti paremmaksi. Alustat ovat erittäin lähellä toisiaan ominaisuuksiltaan ja toteutukseltaan. Flex on kuitenkin alustana kypsempi, sisältää paremmat komponentit ja asioiden toteutus on yleisesti ottaen Flex:llä selkeämpää ja yksinkertaisempaa.

Käytännössä molemmilla ympäristöillä voi kuitenkin toteuttaa melkein minkä tahansa toiminnallisuuden, mutta niin sanotusti ”out of box” Flex on tehokkaampi alusta, koska suurin osa asioista onnistuu sen omilla komponenteilla eikä omaa toteutusta tai kolmannen osapuolen toteutuksia tarvita.

Tämä ei kuitenkaan riitä julistamaan Flex:iä selkeästi parhaaksi alustaksi kaikkiin tilanteisiin.

Tärkein valintaperuste on halutun projektin/ohjelman kohdealusta, esim. Windows Phone-sovellukseen ainoa järkevä valinta on Silverlight.

Jos kohdealusta soveltuu molemmille järjestelmille, niin toiseksi tärkein valintaperuste on tekijöiden aiempi ohjelmointikokemus. Jos kehittäjillä on aiempaa kokemusta Flash tai Java-ympäristöstä, niin Flex on siltä osin parempi valinta. Vastaavasti .NET-osaajat ovat enemmän kotonaan Silverlightin ja Visual Studion kanssa.

Web-puolella alustan levinneisyys on tärkeä kriteeri ja Flashin valtavan suosion takia Flex on selkeästi etulyöntiasemassa. Työpöytä-sovelluksissa eroa ei ole, koska molemmissa tapauksissa tarvitsee asentaa erillinen ajoympäristö. Flexin tapauksessa tämä on Adobe AIR. Silverlight sisältää itse tarvittavan ajoympäristön.

5.4 Adobe Flex

5.4.1 Vahvuudet

- Flashin levinneisyys
- selkeämpi ympäristö, paremmat oletuskomponentit
- yhteisö, valmiiksi tehtyjä palikoita löytyy valtavasti
- AIR
- alustariippumaton
- Flash Builder on erittäin selkeä kehitysalusta
- Android ja iOS-tuki AIR:n kautta

5.4.2 Heikkoudet

- Vain yksi ohjelmointikieli (AS3.0), joka ei ole yhtä kypsä kuin monet .NET-kielet
- AS3.0 on paljon parempi kuin edeltäjänsä mutta ei pärjää vertailussa esim. C#:lle.
- Ei tukea Windows Phonelle (ainakaan vielä)

5.4.3 Soveltuvuus opetuskäyttöön

Jos minulta kysyttäisiin yleisellä tasolla, mitä ohjelmointikieltä/ympäristöä suosittelisin ohjelmoinnin aloittamiseen, niin Flex olisi korkealla listan kärkipäässä. Tähän on useita syitä:

1. Flash Builder ympäristönä on selkeä, helppo oppia ja sisältää silti käytännössä kaikki yleisemmin tarvittavat toiminnallisuudet. Ohjelmoinnin opettelu on tarpeeksi vaikeaa muutenkin, eikä siihen tarvitse lisätä taistelua kehitysympäristöä vastaan.

2. ActionScript 3.0 on tehokas ja syntaksiltaan selkeä olio-ohjelmointikieli. Olio-ohjelmointikielien yksi suurimpia hienouksia on se, että kun opit yhden kunnolla, niin toiseen siirtyminen on suhteellisen helppoa.

3. Flex:n ulkoasu-toteutus tapahtuu suurelta osalta XML-pohjaisella XAML-kielellä. Tämä on trendi nykyään käytännössä kaikkialla ja perusidea on hyvä oppia jo aikaisessa vaiheessa.

4. Ensimmäisen ohjelmointialustan on suositeltavaa olla graafinen, että opiskelija näkee työnsä tuloksen. Lisäksi harjoituksista ja tehtävistä saa tehtyä paljon mielenkiintoisempia ja hauskempia, kun voi käyttää animaatioita ynnä muita graafisia elementtejä.

5. Flex sisältää valtavan määrän erilaisia valmiita komponentteja, joten jo ohjelmointia aloittelevien opiskelijoiden kanssa voi tehdä oikeasti toimivia ja käyttökelpoisia sovelluksia. Ei tarvitse tyytyä teoria-tason esimerkkeihin. Tämä tekee ohjelmoinnin oppimisesta mielekkäämpää ja tehokkaampaa.

Mielestäni ohjelmoinnin peruskurssin yksi tärkeimpiä tavoitteita tulisi olla saada opiskelijat kiinnostumaan ohjelmoinnista ja havaitsemaan, että se ei ole niin kauheaa kuin he todennäköisesti luulevat. Itse käymälläni peruskurssilla kielenä oli C ja vaikutus oli käytännössä päinvastainen. C on teorian tasolla hyvä valinta aloituskieleksi, koska se on C-kielien ”isä”. Käytännössä valinta ei ole osuva, koska C on kielenä hyvin vanhanaikainen, ei olio-pohjainen kieli, joka ei sisällä mitään graafista puolta. Jos tarkoituksena on luoda innostusta ohjelmointiin ensimmäisellä kurssilla, niin C on yksi huonoimmista mahdollisista valinnoista.

Yhteenvetona siis suosittelen Flex/Flash Builder-yhdistelmää varsinkin ohjelmoinnin perusopetukseen. Flexistä on helppo siirtyä johonkin muuhun oliopohjaiseen kieleen, koska perusasiat pysyvät käytännössä samana.

5.5 Microsoft Silverlight

5.5.1 Vahvuudet

- Kaikki .NET-ohjelmointikielet, kehittäjän ei tarvitse opetella uutta kieltä. .NET osaajalle looginen valinta.
- WPF-kehittäjät ovat kuin kotonaan
- Visual Studio on erittäin kehittynyt kehitysympäristö
- Windows Phonen pääkehitysympäristö

5.5.2 Heikkoudet

- Silverlight ei ole yhtä laajalle levinnyt kuin Flash.
- Vaikka tuki löytyy usealle eri ohjelmointikielelle, niin yksi niistä pitää kuitenkin valita. Tämä voi olla ongelma kun kehittäjiä on useampi.
- Visual Studiossa on huomattava oppimiskynnys, eikä se ole niin selkeä kuin Flash Builder. Vastapainona Visual Studio on paljon monipuolisempi.
- Linux-tuki Moonlightin kautta on epävirallinen, eikä ilmeisesti täysin toimiva.
- Ei yhtä laajaa työpöytä-tukea kuin AIR:lla.
- Ei tukea Androidille (tuki tulossa Moonlight-tiimiltä)

- Ei tukea iOS:lle
- Käytännössä virallinen mobiilituki ainoastaan Windows Phonelle mutta siinä Silverlight onkin koko Windows Phone-alustan virallinen kehitysympäristö.

5.5.3 Soveltuvuus opetuskäyttöön

Silverlight ja Visual Studio soveltuvat alustaksi opiskelijoille joilla on jo jonkin verran ohjelmointipohjaa. Alusta on erityisen soveltuva mobiilikehitykseen.

Perusteluja:

1. Visual Studio on yksi tehokkaimmista ohjelmointiympäristöistä, mutta tämän takia se on myös yksi monimutkaisimmista. Pelkästään ympäristön oppimiseen menee huomattavan paljon aikaa, varsinkin jos taustalla ei ole aiempaa ohjelmointikokemusta.

2. Silverlight tukee käytännössä kaikkia .NET-kieliä, joten käytetyn kielen suhteen on suurempia vapauksia. Käytännössä kuitenkin useimmat projektit toteutetaan joko C#- tai VB.NET-ohjelmointikielillä, koska näille on eniten dokumentaatiota.

3. Silverlight, C# ja XNA ovat Windows Phonen pääohjelmointialustat, joten Silverlight-osaajien tarve tuskin on katoamassa. Pikemminkin päinvastoin, riippuen siitä miten Microsoftin mobiilialusta menestyy.

On suositeltavaa, että opiskelijoilla on jo perustason ymmärrys ohjelmoinnista ennen Silverlightiin siirtymistä. Esimerkiksi Flex:stä on helppo siirtyä Silverlightiin, koska alustat ovat huomattavan samanlaisia, mutta Flex/Flash Builder soveltuu paljon paremmin ohjelmoinnin ensikertalaisille.

Yhteenvetona siis suosittelen Silverlightin opettamista osana mobiilipakettia käyttäen kielenä nimenomaan C#:ia.

LÄHTEET

Labriola M., Tapper J., Boles M. 2012, Adobe Flex 4.5 Fundamentals: Training from the Source. Berkeley, CA: Peachpit (Labriola, Tapper & Boles, 2012)

MacDonald M. 2010. Pro Silverlight 4 in C#. USA: Apress. (MacDonald 2010)

Wikipedia 2012a. Adobe AIR. Viitattu 27.2.2012. http://en.wikipedia.org/wiki/Adobe_air (Wikipedia 2012a)

Wikipedia 2012b. ActionScript. Viitattu 27.2.2012. http://en.wikipedia.org/wiki/ActionScript (Wikipedia 2012b)

Wikipedia 2012c. C sharp (programming language). Viitattu 31.3.2012. http://en.wikipedia.org/wiki/C_Sharp_(programming_language) (Wikipedia 2012c)

Liite 1

Flex-sovelluksen lähdekoodi luokittain

MetsoAtlas.mxml

```
<?xml version="1.0" encoding="utf-8"?>
<s:WindowedApplication xmlns:fx="http://ns.adobe.com/mxml/2009"
xmlns:s="library://ns.adobe.com/flex/spark"
xmlns:mx="library://ns.adobe.com/flex/mx"
width="1280" height="768"
xmlns:luokat="luokat.*"
initialize="alusta()"
showStatusBar="false"
xmlns:components="components.*"
backgroundColor="#4D4D4D">

  <fx:Style>
            @namespace s "library://ns.adobe.com/flex/spark";
            @namespace mx "library://ns.adobe.com/flex/mx";
            @namespace viewComponents "/src/";
            @namespace local "*";
            @namespace view "viewComponents.*";

            @font-face {
                        src: url("/data/fonts/verdana.ttf");
                        fontFamily: "verdana";
                        embedAsCFF:true;
                        advancedAntiAliasing: true;
            }
            @font-face {
                        src: url("/data/fonts/verdanab.ttf");
                        fontFamily: "verdana";
                        font-weight:bold;
                        embedAsCFF:true;
                        advancedAntiAliasing: true;
            }
            @font-face {
                        src: url("/data/fonts/verdanai.ttf");
                        fontFamily: "verdana";
                        font-style:italic;
                        embedAsCFF:true;
                        advancedAntiAliasing: true;
            }

            mx|Alert
            {
                        chrome-color: #e05314;
                        corner-radius: 10;
            }
  </fx:Style>

  <fx:Script>
     <![CDATA[

        private function alusta():void
        {
          this.systemManager.stage.scaleMode = StageScaleMode.SHOW_ALL;
          this.systemManager.stage.align = StageAlign.TOP;
          this.systemManager.stage.quality = "BEST";

        }
       ]]>
   </fx:Script>
```

```
   <fx:Declarations>
   <!-- Place non-visual elements (e.g., services, value objects) here -->
   </fx:Declarations>

   <s:Group id="ylaPalkki" depth="6" height="73" width="1280" visible="true" y="0">
      <mx:Image y="0" source="kuvat/ylapalkki.png" smoothBitmapContent="true"
depth="1" x="0"/>
   </s:Group>

   <components:kartta id="karttaTaso"/>

</s:WindowedApplication>
```

kartta.mxml

```
<?xml version="1.0" encoding="utf-8"?>
<s:Group xmlns:fx="http://ns.adobe.com/mxml/2009"
xmlns:s="library://ns.adobe.com/flex/spark"
xmlns:mx="library://ns.adobe.com/flex/mx" width="1280" height="768"
xmlns:components="components.*"
xmlns:luokat="luokat.*"
xmlns:graffat="kuvat/vektori.*"
initialize="alusta()" xmlns:vektori="kuvat.vektori.*"
>
   <fx:Declarations>
   <!-- Place non-visual elements (e.g., services, value objects) here -->
   </fx:Declarations>

   <fx:Script>
      <![CDATA[
            import luokat.HaeTiedot;
            import mx.controls.Alert;

            private var edellinenSivu:String = "maailma";
            private var nykyinenSivu:String = "maailma";

            private var sisaltoObjekti:Object;
            private var toimi:Image;

            private var tietoHaku:HaeTiedot = new HaeTiedot();

            private function alusta():void
            {
              tietoHaku.addEventListener(HaeTiedot.LADATTU,ladattu);
              leftMenu.addEventListener(vasenValikko.TAULUKKO, naytaTaulukko);
              leftMenu.addEventListener(vasenValikko.KAAVIO, naytaKaavio);
              contentPopup.addEventListener(sisaltoPopup.LUKITSE, lukitseTausta);
              contentPopup.addEventListener(sisaltoPopup.VAPAUTA, vapautaTausta);

              rightMenu.addEventListener(oikeaValikko.SHOW_PROFITS, showProfits);
              rightMenu.addEventListener(oikeaValikko.SHOW_TRENDS, showTrends);

              tietoHaku.alustaKanta();
              tietoHaku.ajaUpdate();
            }
            private function showProfits(e:Event):void
            {
              if(worldMap.visible==true)
              {
                pohjoisEurooppa.changeColor(0x008B07);
                lansiEurooppa.changeColor(0xB2C809);
              }else
              {
                pohjoisEurooppaIso.changeColors("profits");
              }
            }
```

```
private function showTrends(e:Event):void
{
  if(worldMap.visible==true)
  {
    pohjoisEurooppa.changeColor(0x79C809);
    lansiEurooppa.changeColor(0xC85309);
  }
  else
  {
    pohjoisEurooppaIso.changeColors("trends");
  }
}

private function ladattu(e:Event):void
{
  sisaltoObjekti = tietoHaku.palautaObjekti();
  lataaToimipaikat();
}

private function lataaToimipaikat():void
{
//näiden tiedot haetaan HaeTiedot-luokan avulla
  var toimet:Array;
  var i:int;
  toimet=tietoHaku.toimiPaikat();

  for(i=0; i<toimet.length; i++)
  {
    toimi = new Image();
    toimi.source="kuvat/toimipaikka2.png";
    toimi.smoothBitmapContent=true;
    toimi.width=37;
    toimi.height=46;
    toimi.visible=true;
    toimi.buttonMode=true;
    toimi.addEventListener(MouseEvent.MOUSE_OVER, toimiOver);
    toimi.addEventListener(MouseEvent.MOUSE_OUT, toimiOut);
    toimi.addEventListener(MouseEvent.CLICK, toimiClick);
    toimi.x=toimet[i]["x"];
    toimi.y=toimet[i]["y"];

    toimi.id=toimet[i]["alue"];

    if(toimet[i]["taso"]=="suomi")
    {
     //toimiSuomi.addElement(toimi);
    }else if(toimet[i]["taso"]=="eurooppa")
    {
     //toimiEurooppa.addElement(toimi);
    }else if(toimet[i]["taso"]=="maailma")
    {
     //toimiMaailma.addElement(toimi);
    }
  }
}

private function toimiClick(event:MouseEvent):void
{
  //tietoRuutu(event.currentTarget.id);
}

private function toimiOver(event:MouseEvent):void
{
  event.currentTarget.source="kuvat/toimipaikkaOver.png";
}
```

```
        private function toimiOut(event:MouseEvent):void
        {
          event.currentTarget.source="kuvat/toimipaikka2.png";
        }

        private function parseri(e:MouseEvent):String
        {
          var dataString:String;
          dataString=e.currentTarget.toString();
          var taulu:Array = dataString.split(".");
          var tulos:String = taulu[taulu.length-1];
          return tulos;
        }

        protected function mouseOver(event:MouseEvent):void
        {
          //var kohde:String = parseri(event)+"Kuva";
          //trace("KOHDE: "+kohde);
          //this[kohde].alpha=1;
          //event.currentTarget.alpha=1;
        }

        protected function mouseOut(event:MouseEvent):void
        {
          //var kohde:String = parseri(event)+"Kuva";
          //this[kohde].alpha=0;
          //event.currentTarget.alpha=0;
        }

        protected function mouseClick(event:MouseEvent):void
        {
          if(event.currentTarget.id=="suljeOhjelmaNappi")
          {
            //tietoHaku.suljeKanta();
            //close();
          }
          else
          {
            sivunVaihto(event.currentTarget.id);
          }
        }

        protected function sivunVaihto(sivu:String):void
        {
          edellinenSivu=nykyinenSivu;
          nykyinenSivu=sivu;
          backButton.visible=true;

          if(sivu=="maailma")
          {
            closerMaps.visible=false;
            backButton.visible=false;
            maailmaSpotit.visible=true;
            worldMap.visible=true;
          }else
          {
            worldMap.visible=false;
            maailmaSpotit.visible=false;
            closerMaps.visible=true;
            if(sivu=="northEurope")
            {
              pohjoisEurooppaIso.visible=true;
```

```
            }
          }

          naytaToimipaikat(sivu);
          piilota(edellinenSivu);
        }

        public function naytaToimipaikat(alue:String):void
        {
          if(alue=="suomi")
          {
            //toimiSuomi.visible=true;
          }else if(alue=="eurooppa")
          {
           //toimiEurooppa.visible=true;
          }else if(alue=="maailma")
          {
           //toimiMaailma.visible=true;
          }
        }

        protected function piilota(alue:String):void
        {
          piilotaToimipaikat(alue);

          if(alue=="eurooppa")
          {
            //suomi.visible=false;
          }
          else if(alue=="maailma")
          {
            /* maailmaTaso.visible=false;
            maailmaSpotit.enabled=false;
            maailmaSpotit.mouseEnabled=false; */
          }
        }

        protected function piilotaToimipaikat(taso:String):void
        {
          if(taso=="suomi")
          {
            //toimiSuomi.visible=false;
          }else if(taso=="eurooppa")
          {
            //toimiEurooppa.visible=false;
          }else if(taso=="maailma")
          {
            //toimiMaailma.visible=false;
          }
        }

        private function lukitseTausta(e:Event):void
        {
          grayBlur.blur();
        }

        private function vapautaTausta(e:Event):void
        {
          grayBlur.unBlur();
        }

        private function naytaTaulukko(e:Event):void
        {
          var a:Array = tietoHaku.palautaTiedot(nykyinenSivu);
          var otsikot:Array = sisaltoObjekti["[ICP Top]"];
          contentPopup.naytaTaulukko(a);
```

```
            contentPopup.vaihdaOtsikko(nykyinenSivu);
            contentPopup.vaihdaOtsikot(otsikot,sisaltoObjekti);
          }

          private function naytaKaavio(e:Event):void
          {
            var a:Array = tietoHaku.palautaTiedot(nykyinenSivu);
            contentPopup.naytaKaavio(a, sisaltoObjekti);
            contentPopup.vaihdaOtsikko(nykyinenSivu);
          }

          protected function backButton_clickHandler(event:MouseEvent):void
          {
            if(contentPopup.visible==true)
            {
              contentPopup.piilota();
            }
            if(nykyinenSivu == "eurooppa")
            {
              sivunVaihto("maailma");
            }
            else if(nykyinenSivu == "maailma")
            {}
            else
            {
              sivunVaihto(edellinenSivu);
            }
          }

          protected function rightMenu_overHandler(event:MouseEvent):void
          {
            rightMenu.y=500;
          }

          protected function rightMenu_mouseOutHandler(event:MouseEvent):void
          {
            rightMenu.y=736;
          }

          protected function pohjoisEurooppa_clickHandler(event:MouseEvent):void
          {}

      ]]>
   </fx:Script>

   <s:Group id="worldMap" visible="true" x="41" y="-2">
            <vektori:karttaEDITuus id="karttaTesti" depth="900" />
            <components:northEuropeMap id="pohjoisEurooppa" depth="1000" />
            <components:westEuropeMap id="lansiEurooppa" depth="1000" />
   </s:Group>

   <s:Group id="closerMaps" visible="false">
            <components:northEuropeMapBig id="pohjoisEurooppaIso" visible="false"
x="130" y="50" depth="900"/>
   </s:Group>

   <components:harmaaTaso id="grayBlur" depth="2000"/> <!-- tämä peittää taustan kun
tuodaan popup esiin, estää taustalla olevien juttujen käytön -->
   <components:vasenValikko id="leftMenu" x="19" y="420" height="237" width="278"
depth="2001"/> <!-- Vasemmalla oleva valikko, napit taulukon/kaavion näyttämiselle -->
   <components:sisaltoPopup id="contentPopup" x="330" y="120" width="924" height="627"
visible="false" depth="2001"/> <!--  -->
   <components:oikeaValikko id="rightMenu" depth="1999" x="1029" y="736" mouseO-
ver="rightMenu_overHandler(event)" mouseOut="rightMenu_mouseOutHandler(event)"/>
   <mx:Image x="210" y="686" source="kuvat/backbutton.png" id="backButton" visi-
ble="false" smoothBitmapContent="true" click="backButton_clickHandler(event)"
depth="2001" buttonMode="true"/>
```

```
    <s:Group id="maailmaSpotit" enabled="true" mouseEnabled="true" alpha="0"
    depth="2000">
              <s:Group id="northEurope" x="18" y="10" buttonMode="true"
              click="mouseClick(event)" mouseEnabledWhereTransparent="false"
              mouseOut="mouseOut(event)"mouseOver="mouseOver(event)">
                 <s:Path x="-226" y="-21" alpha="0"
                          data="
                          M 600 100
                          L 900 100
                          L 880 250
                          L 850 252
                          L 830 280
                          L 770 250
                          L 650 250
                          L 600 250
                          L 540 150      Z">
                    <s:stroke>
                       <s:SolidColorStroke color="#333333" caps="square" joints="miter"/>
                    </s:stroke>
                    <s:fill>
                       <s:SolidColor color="#00CCFF"/>
                    </s:fill>
                 </s:Path>
              </s:Group>
    </s:Group>
</s:Group>
```

oikeaValikko.mxml

```
<?xml version="1.0" encoding="utf-8"?>
<s:TitleWindow xmlns:fx="http://ns.adobe.com/mxml/2009"
               xmlns:s="library://ns.adobe.com/flex/spark"
               xmlns:mx="library://ns.adobe.com/flex/mx" width="242" height="300"
               chromeColor="#E05314" cornerRadius="10" borderVisible="true"
               dropShadowVisible="true" textAlign="left"
               initialize="titlewindow1_initializeHandler(event)">
   <s:layout>
            <s:VerticalLayout gap="20" paddingTop="20" paddingLeft="10" horizonta-
lAlign="left"/>
   </s:layout>

   <fx:Script>
      <![CDATA[
            import mx.core.windowClasses.TitleBar;
            import mx.events.CloseEvent;
            import mx.events.FlexEvent;
            public static const SHOW_TRENDS:String = "show_trends";
            public static const SHOW_PROFITS:String = "show_profits";


            protected function clickHandler(event:MouseEvent):void //yleinen click-
handler, lähettää tiedon käyttäjän valinnasta eventtinä pääohjelmalle
            {
               switch(event.target.id)
               {
                  case "asetus1":
                        dispatchEvent(new Event(SHOW_TRENDS));
                        break;
                  case "asetus2":
                        dispatchEvent(new Event(SHOW_PROFITS));
                        break;
               }
            }

            protected function titlewindow1_initializeHandler(event:FlexEvent):void
            {
               closeButton.visible=false;
               title="Settings and filters";
            }
      ]]>
   </fx:Script>

   <fx:Declarations>
            <!-- Place non-visual elements (e.g., services, value objects) here -->
   </fx:Declarations>

   <!-- Näillä napeilla tehdään valintoja siitä mitä karttapohjalla näytetään -->
   <s:Label id="asetus1" text="Show Trends" fontFamily="verdanaNormi" fontSize="20"
click="clickHandler(event)" buttonMode="true"/>
   <s:Label id="asetus2" text="Show Profits" fontFamily="verdanaNormi" fontSize="20"
click="clickHandler(event)" buttonMode="true"/>
   <s:Label id="asetus3" text="asetus3" fontFamily="verdanaNormi" fontSize="20"
click="clickHandler(event)" buttonMode="true"/>
   <s:Label id="asetus4" text="asetus4" fontFamily="verdanaNormi" fontSize="20"
click="clickHandler(event)" buttonMode="true"/>

</s:TitleWindow>
```

sisaltoPopup.mxml

```
<?xml version="1.0" encoding="utf-8"?>
<s:TitleWindow xmlns:fx="http://ns.adobe.com/mxml/2009"
               xmlns:s="library://ns.adobe.com/flex/spark"
               xmlns:mx="library://ns.adobe.com/flex/mx"
               chromeColor="#E05314" cornerRadius="10"
               close="titlewindow1_closeHandler(event)"
               height="627" width="924”>

   <s:states>
            <s:State id="tekstiState" name="TekstiState"/>
            <s:State id="taulukkoState" name="TaulukkoState"/>
            <s:State id="kaavioState" name="KaavioState"/>
   </s:states>

   <fx:Script>
      <![CDATA[
            import mx.charts.chartClasses.Series;
            import mx.charts.series.LineSeries;
            import mx.collections.ArrayCollection;
            import mx.controls.dataGridClasses.DataGridColumn;
            import mx.events.CloseEvent;
            import mx.events.FlexEvent;

            private var taulukkoOtsikot:Array;
            private var sisaltoArray:Array;
            private var sisaltoObjekti:Object;
            private var nykalkuPiste:int=0;
            private var nykloppuPiste:int=20;
            private var netSales:Array = new Array;
            private var newOrders:Array = new Array;


            [Bindable]
            private var alkuPiste:int;
            [Bindable]
            private var loppuPiste:int;

            private var taulukkoTesti:Array;
            private var tila:String="net sales";

            [Bindable]
            private var taulukkoArray:Array;
            [Bindable]
            private var kaavioArray:Array;

            public static const LUKITSE:String = "lukitse";
            public static const VAPAUTA:String = "vapauta";

            public function vaihdaOtsikko(s:String):void
            {
               switch(s)
               {
                  case "eurooppa":
                        title="Europe";
                        break;
                  case "aasia":
                        title="Asia";
                        break;
                  case "afrikka":
                        title="Africa";
                        break;
                  case "pohjoisamerikka":
                        title="North America";
                        break;
                  case "etelaamerikka":
                        title="South America";
                        break;
```

```
                case "maailma":
                        title="World";
                        break;
            }
        }

        public function piilota():void
        {
            this.visible=false;
            dispatchEvent(new Event(VAPAUTA));
        }

        private function resetoi():void
        {
            try
            {
                linechart1.series = null;
            }catch(e:Error)
            {}
        }

        public function naytaTaulukko(a:Array):void
        {
            if(this.visible==false)
            {
                dispatchEvent(new Event(LUKITSE));
            }
            resetoi();
            currentState="TaulukkoState";
            alkuPiste=2;
            loppuPiste=a[2].length-2;

            taulukkoTesti=a;

            var tempArray:Array = new Array;

            for(var i:int=0;i<=a.length-2;i++)
            {
                        var soluArray:Array = new Array;
                        soluArray.push(trimWhitespace(a[i][0]));
                        soluArray.push(trimWhitespace(a[i][1]));
                        for(var j:int=2;j<=a[i].length-2;j++)
                        {
                            soluArray.push(parseInt(removeSpaces(a[i][j])));
                        }
                        tempArray.push(soluArray);
            }

            taulukkoArray=tempArray;
            this.visible=true;
        }

        public function vaihdaOtsikot(a:Array,o:Object):void
        {
            var YTD:Array = o["YTD"];
            var columnArray:Array = taulukko.columns;
            var colTest:Array = new Array;

            for(var i:int=0;i<a.length-1;i++)
            {
                        var uusiColumn:DataGridColumn = columnArray[i];
                        uusiColumn.headerText = a[i]+" "+YTD[i];
                        colTest.push(uusiColumn);
            }
            taulukko.columns = colTest;
        }
```

```
private function trimWhitespace($string:String):String {
   if ($string == null) {
            return "";
   }
   return $string.replace(/^\s+|\s+$/g, "");
}

public function naytaKaavio(a:Array,o:Object):void
{
   currentState="KaavioState";
   resetoi();

   var YTD:Array = o["YTD"];
   var ICP:Array = o["[ICP Top]"];
   var MCT:Array = o["MCT000"];

   for(var i:int=0;i<MCT.length;i++)
   {
            if(MCT[i]=="NET SALES")
            {
               netSales.push(i);
            }else if(MCT[i]=="NEW ORDERS")
            {
               newOrders.push(i);
            }
   }

   if(tila=="net sales")
   {
            alkuPiste=2;
            loppuPiste=newOrders[0]-5;
   }else if(tila=="new orders")
   {
            alkuPiste=newOrders[0];
            loppuPiste=a[2].length-6;
   }

   sisaltoArray=a;
   sisaltoObjekti=o;

   nykalkuPiste=alkuSlider.value;
   nykloppuPiste=loppuSlider.value;

   if(this.visible==false)
   {
            dispatchEvent(new Event(LUKITSE));
   }

   kaavioArray = new Array;

   for(i=nykalkuPiste;i<=nykloppuPiste;i++)
   {
            var sisObjekti:Object = new Object;

            for(var j:int=0;j<=a.length-2;j++)
            {
               var otsikkoString:String = trimWhitespace(a[j][1]);
               sisObjekti[otsikkoString] = parseInt(removeSpaces(a[j][i]));
            }
            sisObjekti.kuukausi = ICP[i]+" "+YTD[i];
            kaavioArray.push(sisObjekti);
   }

   //luodaan LineSeriesit
   for(j=0;j<=a.length-2;j++)
   {
            otsikkoString = trimWhitespace(a[j][1]);
            var testSeries:LineSeries = new LineSeries;
```

```
                        testSeries.xField="kuukausi";
                        testSeries.yField=otsikkoString;
                        testSeries.displayName=otsikkoString;
                        linechart1.series.push(testSeries);
                }
                this.visible=true;
            }

            public static function removeSpaces(str:String):String{
                var text_arr:Array=str.split(' ');
                return(text_arr.join(''));
            }

            protected function titlewindow1_closeHandler(event:CloseEvent):void
            {
                dispatchEvent(new Event(VAPAUTA));
                this.visible=false;
            }

            private function sliderControl():void
            {
                var tempArray:Array = new Array;
                nykalkuPiste=alkuSlider.value;
                nykloppuPiste=loppuSlider.value;

                if(currentState=="TaulukkoState")
                {
                    //tehdään taulukkoArray uudestaan säätimien asennon mukaan
                    for(var i:int=0;i<=taulukkoTesti.length-2;i++)
                    {
                            var soluArray:Array = new Array;
                            soluArray.push(trimWhitespace(taulukkoTesti[i][0]));
                            soluArray.push(trimWhitespace(taulukkoTesti[i][1]));
                            for(var j:int=nykalkuPiste;j<=nykloppuPiste;j++)
                            {
                                soluArray.push(parseInt(removeSpaces(taulukkoTesti[i][j])));
                            }
                            tempArray.push(soluArray);
                    }
                    taulukkoArray=tempArray;
                }
                else if(currentState=="KaavioState")
                {
                    naytaKaavio(sisaltoArray,sisaltoObjekti);
                }
            }

            protected function mouseClick(event:MouseEvent):void
            {
                if(event.currentTarget.id=="netSalesValinta" ||
event.currentTarget.id=="netSalesNuoli")
                {
                            tila="net sales";
                            naytaKaavio(sisaltoArray,sisaltoObjekti);
                }
                else if(event.currentTarget.id=="newOrdersValinta" ||
event.currentTarget.id=="newOrdersNuoli")
                {
                            tila="new orders";
                            naytaKaavio(sisaltoArray,sisaltoObjekti);
                }
            }
        ]]>
    </fx:Script>
    <fx:Declarations>
            <!-- Place non-visual elements (e.g., services, value objects) here -->
    </fx:Declarations>
```

```
   <s:Group id="tietoIkkuna" visible="true" height="594" width="923" depth="4">

            <s:Label x="32" y="14" text="Metso" id="tietoOtsikko" fontSize="38" fontFami-
ly="Calibri" fontWeight="bold" color="#005D55"/>
            <s:Label x="386" y="21" text="Otsikko 2" id="otsikko2" fontSize="24" font-
Weight="normal" fontStyle="normal" textDecoration="none" includeIn="TekstiState"/>
            <s:Label x="386" y="261" text="Otsikko 3" id="otsikko3" fontSize="24" in-
cludeIn="TekstiState"/>
            <s:TextArea x="389" y="52" width="415" height="180" id="tekstiAlue1" edita-
ble="false" borderVisible="false" includeIn="TekstiState"/>
            <s:TextArea x="386" y="291" width="415" height="199" id="tekstiAlue2" edita-
ble="false" borderVisible="false" includeIn="TekstiState"/>

            <mx:Image x="32" y="62" width="270" height="212" id="tietoKuva" smoothBitmap-
Content="true" includeIn="TekstiState"/>
            <s:Group id="valinnat" x="20" y="519" height="67" width="878">

               <mx:Image id="netSalesNuoli" source="kuvat/valintanuoli.png" smoothBitmap-
Content="true" width="30" height="23" buttonMode="true" click="mouseClick(event)" x="23"
y="4"/>
               <s:Label text="Net Sales" id="netSalesValinta" fontSize="30" fontFami-
ly="Calibri" fontWeight="normal" color="#005D55" buttonMode="true"
click="mouseClick(event)" x="63"/>

               <mx:Image id="newOrdersNuoli" source="kuvat/valintanuoli.png" smoothBit-
mapContent="true" width="30" height="23" buttonMode="true" click="mouseClick(event)"
x="23" y="34"/>
               <s:Label text="New Orders" id="newOrdersValinta" fontSize="30" fontFami-
ly="Calibri" fontWeight="normal" color="#005D55" buttonMode="true"
click="mouseClick(event)" x="63" y="30"/>

            </s:Group>

            <mx:DataGrid id="taulukko" variableRowHeight="true" wordWrap="true" dataPro-
vider="{taulukkoArray}" horizontalScrollPolicy="auto" verticalScrollPolicy="auto" in-
cludeIn="TaulukkoState" x="32" y="60" width="866" height="451">
            </mx:DataGrid>


            <mx:LineChart includeIn="KaavioState" dataProvider="{kaavioArray}" show-
DataTips="true" x="5" y="60" id="linechart1" width="798" height="451">
               <mx:horizontalAxis>
                        <mx:CategoryAxis categoryField="kuukausi">
                        </mx:CategoryAxis>
               </mx:horizontalAxis>

               <mx:series>
               </mx:series>

            </mx:LineChart>
            <mx:Legend includeIn="KaavioState" dataProvider="{linechart1}" x="781" y="61"
width="132"/>
            <s:HSlider includeIn="TaulukkoState, KaavioState" x="329" y="21" width="569"
id="alkuSlider" minimum="{alkuPiste}" maximum="{loppuPiste}" value="{alkuPiste}" chang-
eEnd="sliderControl()"/>
            <s:HSlider includeIn="TaulukkoState, KaavioState" x="329" y="41" width="569"
id="loppuSlider" minimum="{alkuPiste}" maximum="{loppuPiste}" value="{loppuPiste}" chang-
eEnd="sliderControl()"/>

   </s:Group>
</s:TitleWindow>
```

HaeTiedot.as

```
package luokat
{
   import flash.data.SQLConnection;
   import flash.data.SQLResult;
   import flash.data.SQLStatement;
   import flash.events.*;
   import flash.events.Event;
   import flash.events.EventDispatcher;
   import flash.events.SQLErrorEvent;
   import flash.events.SQLEvent;
   import flash.events.StatusEvent;
   import flash.external.ExternalInterface;
   import flash.filesystem.File;
   import flash.net.LocalConnection;
   import flash.net.URLLoader;
   import flash.net.URLRequest;
   import flash.sampler.NewObjectSample;
   import flash.security.RevocationCheckSettings;
   import mx.utils.OnDemandEventDispatcher;

   public class HaeTiedot extends EventDispatcher
   {
            private var kanta:SQLConnection;
            private var dbFile:File;
            private var dbStatement:SQLStatement;
            private var hakuTulos:Array;
            private var sqlQuery:String;
            private var omaArray:Array=[];
            private var mainArr:Array=[];

            public static const LADATTU:String = "ladattu";

            private var csvObjekti:Object = new Object;

            public function HaeTiedot()
            {}

            public function palautaTiedot(s:String):Array
            {
               var sisaltoArray:Array = new Array;
               var kategoriat:Array;
               switch(s)
               {
                  case "northEurope":
                        kategoriat = new Array("Northern Europe","SD Finland","SD Scandi-
navia");
                        break;
                  case "europe":
                        kategoriat = new Array("Northern Europe","SD Finland","SD Scandi-
navia","Great Britain","DE","PL","BE","NL","Southern Europe","SD
France","Iberica","Central Europe");
                        break;
                  case "asia":
                        kategoriat = new Array("Asia Pacific
MCT","IN","CN","Australasia","Oceania","Turkey","Middle East","Russia and CIS");
                        break;
                  case "africa":
                        kategoriat = new Array("Western North Africa","Eastern North Af-
rica","Southern Africa","Africa West","Africa Central");
                        break;
                  case "northAmerica":
                        kategoriat = new Array("North America MCT","SD
USA","US","CA","Central America");
                        break;
                  case "southAmerica":
                        kategoriat = new Array("South America MCT","BR","South
Cone","Andean Area","North West","South West");
                        break;
```

```
                case "maailma":
                        kategoriat = new Array("EMEA","Americas","Asia Pacific MCT");
                        break;
                default:
                        kategoriat = new Array("nollaArpa");
                        break;
            }

            for(var i:int=0; i<=kategoriat.length; i++)
            {
                        sisaltoArray[i] = csvObjekti[kategoriat[i]];
            }
            return sisaltoArray;
        }

        public function palautaObjekti():Object
        {
                    return csvObjekti;
        }

        public function palautaOtsikot():Array
        {
                    var otsikkoArray:Array = new Array;
                    return otsikkoArray;
        }

        public function ajaUpdate():void
        {
                    var myLoader:URLLoader = new URLLoader();
                    myLoader.load(new URLRequest("data/taulukko.csv"));
                    myLoader.addEventListener(Event.COMPLETE, parseMyData);

        }

        private function parseMyData(e:Event):void
        {
            var myText:String = e.target.data;
            var rows:Array = myText.split("\n");

            var i:Number=0;
            var j:Number=0;

            for each(var arvo:String in rows)
            {
                j=0;
                omaArray[i] = new Array();
                var rivi:Array = arvo.split(";");
                for each(var solu:String in rivi)
                {
                        omaArray[i][j] = solu;
                        j++;
                }

                var trimString:String = omaArray[i][0];

                while (trimString.charAt(0) == " ") {
                        trimString = trimString.substr(1, trimString.length);
                }
                while (trimString.charAt(trimString.length - 1) == " ") {
                        trimString = trimString.substr(0, trimString.length - 1);
                }
                for (var k:Number = 0; k < trimString.length; k++) {
                    while (trimString.charAt(i) == " " && trimString.charAt(i+1) == " ")
                    {
                            trimString = trimString.substr(0, i) + trimString.substr(i+1);
                    }
                }
```

```
                    csvObjekti[trimString]=omaArray[i];
                    i++;
                }
                dispatchEvent(new Event(LADATTU));
            }

            public function alustaKanta():void
            {
                dbFile = File.applicationDirectory.resolvePath("data/kanta2.db");
                kanta = new SQLConnection();
                kanta.addEventListener(SQLErrorEvent.ERROR, errorHandler);
                kanta.open(dbFile);
            }

            public function suljeKanta():void
            {
                kanta.close();
            }

            private function lataaTiedot(taulu:String,alue:String):void
            {
                dbStatement = new SQLStatement();
                dbStatement.sqlConnection = kanta;
                if(taulu=="toimipaikat")
                {
                        sqlQuery = "SELECT * FROM "+taulu+" WHERE toimiID = '"+alue+"'";
                }else if(taulu=="ikonit")
                {
                        sqlQuery = "SELECT * FROM ikonit";
                }

                dbStatement.text = sqlQuery;
                dbStatement.addEventListener(SQLEvent.RESULT, haunTulokset);
                dbStatement.execute();
            }

            private function haunTulokset(event:SQLEvent):void
            {
                var tulos:SQLResult = dbStatement.getResult();
                if (tulos != null)
                {
                        hakuTulos = tulos.data;
                }
            }

            private function errorHandler(event:SQLEvent):void
            {}

            public function toimiPaikat():Array
            {
                lataaTiedot("ikonit", "");
                return hakuTulos;
            }


            public function tietoRuutu(alue:String):Array
            {
                lataaTiedot("toimipaikat",alue);
                return hakuTulos;
            }
    }
}
```

MetsoAtlas-app.xml

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<application xmlns="http://ns.adobe.com/air/application/2.6">

<!-- Adobe AIR Application Descriptor File Template.

            Specifies parameters for identifying, installing, and launching AIR applica-
tions.

            xmlns - The Adobe AIR namespace: http://ns.adobe.com/air/application/2.6
                                    The last segment of the namespace specifies the ver-
sion
                                    of the AIR runtime required for this application to
run.

            minimumPatchLevel - The minimum patch level of the AIR runtime required to run
                                    the application. Optional.
-->

            <!-- A universally unique application identifier. Must be unique across all
AIR applications.
            Using a reverse DNS-style name as the id is recommended. (Eg.
com.example.ExampleApplication.) Required. -->
            <id>Atlas</id>

            <!-- Used as the filename for the application. Required. -->
            <filename>Atlas</filename>

            <!-- The name that is displayed in the AIR application installer.
            May have multiple values for each language. See samples or xsd schema file.
Optional. -->
            <name>Atlas</name>

            <!-- A string value of the format <0-999>.<0-999>.<0-999> that represents ap-
plication version which can be used to check for application upgrade.
            Values can also be 1-part or 2-part. It is not necessary to have a 3-part val-
ue.
            An updated version of application must have a versionNumber value higher than
the previous version. Required for namespace >= 2.5 . -->
            <versionNumber>0.0.1</versionNumber>

            <!-- A string value (such as "v1", "2.5", or "Alpha 1") that represents the
version of the application, as it should be shown to users. Optional. -->
            <versionLabel>Demo 1</versionLabel>

            <!-- Description, displayed in the AIR application installer.
            May have multiple values for each language. See samples or xsd schema file.
Optional. -->
            <!-- <description></description> -->

            <!-- Copyright information. Optional -->
            <!-- <copyright></copyright> -->

            <!-- Publisher ID. Used if you're updating an application created prior to
1.5.3 -->
            <!-- <publisherID></publisherID> -->

            <!-- Settings for the application's initial window. Required. -->
            <initialWindow>
                        <!-- The main SWF or HTML file of the application. Required. -->
                        <!-- Note: In Flash Builder, the SWF reference is set automatical-
ly. -->
                        <content>[This value will be overwritten by Flash Builder in the
output app.xml]</content>

                        <!-- The title of the main window. Optional. -->
                        <!-- <title></title> -->
```

```
<!-- The type of system chrome to use (either "standard" or "none"). Optional. Default
standard. -->
                        <!-- <systemChrome></systemChrome> -->

                        <!-- Whether the window is transparent. Only applicable when sys-
temChrome is none. Optional. Default false. -->
                        <!-- <transparent></transparent> -->

                        <!-- Whether the window is initially visible. Optional. Default
false. -->
                        <!-- <visible></visible> -->

                        <!-- Whether the user can minimize the window. Optional. Default
true. -->
                        <!-- <minimizable></minimizable> -->

                        <!-- Whether the user can maximize the window. Optional. Default
true. -->
                        <!-- <maximizable></maximizable> -->

                        <!-- Whether the user can resize the window. Optional. Default
true. -->
                        <!-- <resizable></resizable> -->

                        <!-- The window's initial width in pixels. Optional. -->
                        <!-- <width></width> -->

                        <!-- The window's initial height in pixels. Optional. -->
                        <!-- <height></height> -->

                        <!-- The window's initial x position. Optional. -->
                        <!-- <x></x> -->

                        <!-- The window's initial y position. Optional. -->
                        <!-- <y></y> -->

                        <!-- The window's minimum size, specified as a width/height pair in
pixels, such as "400 200". Optional. -->
                        <!-- <minSize></minSize> -->

                        <!-- The window's initial maximum size, specified as a width/height
pair in pixels, such as "1600 1200". Optional. -->
                        <!-- <maxSize></maxSize> -->

        <!-- The initial aspect ratio of the app when launched (either "portrait" or "land-
scape"). Optional. Mobile only. Default is the natural orientation of the device -->

        <!-- <aspectRatio></aspectRatio> -->

        <!-- Whether the app will begin auto-orienting on launch. Optional. Mobile only.
Default false -->

        <!-- <autoOrients></autoOrients> -->

        <!-- Whether the app launches in full screen. Optional. Mobile only. Default false -
-->

        <!-- <fullScreen></fullScreen> -->

        <!-- The render mode for the app (either auto, cpu, or gpu). Optional. Mobile only.
Default auto -->

        <!-- <renderMode></renderMode> -->

                        <!-- Whether or not to pan when a soft keyboard is raised or low-
ered (either "pan" or "none").  Optional.  Defaults "pan." -->
                        <!-- <softKeyboardBehavior></softKeyboardBehavior> -->
            <autoOrients>false</autoOrients>
```

```
<fullScreen>false</fullScreen>
        <visible>false</visible>
    </initialWindow>

            <!-- We recommend omitting the supportedProfiles element, -->
            <!-- which in turn permits your application to be deployed to all -->
            <!-- devices supported by AIR. If you wish to restrict deployment -->
            <!-- (i.e., to only mobile devices) then add this element and list -->
            <!-- only the profiles which your application does support. -->
            <!-- <supportedProfiles>desktop extendedDesktop mobileDevice extendedMo-
bileDevice</supportedProfiles> -->

            <!-- The subpath of the standard default installation location to use. Optional.
-->
            <!-- <installFolder></installFolder> -->

            <!-- The subpath of the Programs menu to use. (Ignored on operating systems
without a Programs menu.) Optional. -->
            <!-- <programMenuFolder></programMenuFolder> -->

            <!-- The icon the system uses for the application. For at least one resolution,
            specify the path to a PNG file included in the AIR package. Optional. -->
            <!-- <icon>
                        <image16x16></image16x16>
                        <image32x32></image32x32>
                        <image36x36></image36x36>
                        <image48x48></image48x48>
                        <image72x72></image72x72>
                        <image114x114></image114x114>
                        <image128x128></image128x128>
            </icon> -->

            <!-- Whether the application handles the update when a user double-clicks an
update version
            of the AIR file (true), or the default AIR application installer handles the
update (false).
            Optional. Default false. -->
            <!-- <customUpdateUI></customUpdateUI> -->

            <!-- Whether the application can be launched when the user clicks a link in a
web browser.
            Optional. Default false. -->
            <!-- <allowBrowserInvocation></allowBrowserInvocation> -->

            <!-- Listing of file types for which the application can register. Optional. -->
            <!-- <fileTypes> -->

                        <!-- Defines one file type. Optional. -->
                        <!-- <fileType> -->

                                    <!-- The name that the system displays for the regis-
tered file type. Required. -->
                                    <!-- <name></name> -->

                                    <!-- The extension to register. Required. -->
                                    <!-- <extension></extension> -->

                                    <!-- The description of the file type. Optional. -->
                                    <!-- <description></description> -->

                                    <!-- The MIME content type. -->
                                    <!-- <contentType></contentType> -->

                                    <!-- The icon to display for the file type. Optional. -
->
                                    <!-- <icon>
                                                <image16x16></image16x16>
                                                <image32x32></image32x32>
```

```
<image48x48></image48x48>
                                                <image128x128></image128x128>
                                    </icon> -->

                        <!-- </fileType> -->
            <!-- </fileTypes> -->

    <!-- iOS specific capabilities -->
            <!-- <iPhone> -->
                        <!-- A list of plist key/value pairs to be added to the application
Info.plist -->
                        <!-- <InfoAdditions>
            <![CDATA[
                <key>UIDeviceFamily</key>
                <array>
                    <string>1</string>
                    <string>2</string>
                </array>
                <key>UIStatusBarStyle</key>
                <string>UIStatusBarStyleBlackOpaque</string>
                <key>UIRequiresPersistentWiFi</key>
                <string>YES</string>
            ]]>
        </InfoAdditions> -->
        <!-- <requestedDisplayResolution></requestedDisplayResolution> -->
            <!-- </iPhone> -->

            <!-- Specify Android specific tags that get passed to AndroidManifest.xml file.
-->
            <!--<android>
                        <manifestAdditions>
                        <![CDATA[
                                    <manifest android:installLocation="auto">
                                                <uses-permission an-
droid:name="android.permission.INTERNET"/>
                                                <uses-permission an-
droid:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
                                                <uses-permission an-
droid:name="android.permission.ACCESS_FINE_LOCATION"/>
                                                <uses-configuration an-
droid:reqFiveWayNav="true"/>
                                                <supports-screens an-
droid:normalScreens="true"/>
                                                <uses-feature android:required="true" an-
droid:name="android.hardware.touchscreen.multitouch"/>
                                                <application android:enabled="true">
                                                            <activity an-
droid:excludeFromRecents="false">
                                                                        <intent-filter>
                                                                                    <action
android:name="android.intent.action.MAIN"/>
                                                                                    <cate-
gory android:name="android.intent.category.LAUNCHER"/>
                                                                        </intent-filter>
                                                            </activity>
                                                </application>
                                    </manifest>
                        ]]>
                        </manifestAdditions>
            </android> -->
            <!-- End of the schema for adding the android specific tags in AndroidMan-
ifest.xml file -->

</application>
```

Silvelight-sovelluksen lähdekoodi luokittain

MainPage.xaml

```
<UserControl x:Class="Atlas.MainPage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:components="clr-namespace:Atlas.components"
    mc:Ignorable="d"
    d:DesignHeight="768" d:DesignWidth="1280">

    <Grid x:Name="LayoutRoot" Background="White">
        <components:kartta x:Name="mainMap"/>
    </Grid>
</UserControl>
```

MainPage.xaml.cs

```
using System.Windows.Controls;

namespace Atlas
{
    public partial class MainPage : UserControl
    {
        public MainPage()
        {
            InitializeComponent();
            mainMap.alusta();
        }
    }
}
```

kartta.xaml

```
<UserControl x:Class="Atlas.components.kartta"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:components="clr-namespace:Atlas.components"
    mc:Ignorable="d"
    d:DesignHeight="768" d:DesignWidth="1280">

    <Grid x:Name="LayoutRoot" Background="White">
        <Grid Name="worldMap" Visibility="Visible">
            <components:worldMap Visibility="Visible"/>
            <components:northEuropeMap2 x:Name="pohjoisEurooppa" MouseLeftBut-
tonUp="pohjoisEurooppa_MouseLeftButtonUp" Cursor="Hand"/>
            <components:westEuropeMap x:Name="lansiEurooppa"/>
            <Image Source="../kuvat/backbutton.png" Width="85" Height="78" x:Name="backButton"
Margin="798,562,397,128" Visibility="Collapsed" Cursor="Hand" />
        </Grid>
```

```
<Grid Name="closerMaps" Visibility="Collapsed">
            <components:northEuropeMapBig2 x:Name="pohjoisEurooppaIso" Visibility="Collapsed"/>
        </Grid>
        <components:vasenValikko x:Name="leftMenu" Width="278" Height="237" Mar-
gin="6,562,1024,6"  />
        <components:oikeaValikko x:Name="rightMenu" Width="242" Height="300" Mar-
gin="1032,462,6,6" />
        <components:sisaltoPopup x:Name="contentPopup" Width="924" Height="627" Visibil-
ity="Collapsed"/>

    </Grid>
</UserControl>
```

kartta.xaml.cs

```
using System;
using System.Collections.Generic;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Input;
using Atlas.luokat;

namespace Atlas.components
{
    public partial class kartta : UserControl
    {
        public kartta()
        {
            InitializeComponent();
            this.DataContext = this;
        }

        private string edellinenSivu = "maailma";
        private string nykyinenSivu = "maailma";

        private HaeTiedot tietoHaku = new HaeTiedot();
        public Dictionary<string, List<string>> sisaltoObjekti = new Dictionary<string,
List<string>>();

        public void alusta()
        {
            tietoHaku.LADATTU += new EventHandler(ladattu);
            leftMenu.KAAVIO += new EventHandler(naytaKaavio);
            leftMenu.TAULUKKO += new EventHandler(naytaTaulukko);
            rightMenu.ShowTrends += new EventHandler(showTrends);
            rightMenu.ShowProfits += new EventHandler(showProfits);
            contentPopup.LUKITSE += new EventHandler(lukitseTausta);
            contentPopup.VAPAUTA += new EventHandler(vapautaTausta);
            tietoHaku.ajaUpdate();
        }

        private void showProfits(object sender, EventArgs e)
        {
            if (worldMap.Visibility == Visibility.Visible)
            {
                pohjoisEurooppa.changeColor("green");
                lansiEurooppa.changeColor("orange");
            }
            else
            {
                //pohjoisEurooppaIso.changeColors("profits");
```

```
}
        }
        private void showTrends(object sender, EventArgs e)
        {
            if (worldMap.Visibility == Visibility.Visible)
            {
                pohjoisEurooppa.changeColor("red");
                lansiEurooppa.changeColor("yellow");
            }
            else
            {
                //pohjoisEurooppaIso.changeColors("trends");
            }
        }

        private void ladattu(object sender, EventArgs e)
        {
            sisaltoObjekti = tietoHaku.palautaObjekti();
        }

        protected void sivunVaihto(String sivu)
        {
            edellinenSivu = nykyinenSivu;
            nykyinenSivu = sivu;
            backButton.Visibility = Visibility.Visible;

            if (sivu == "maailma")
            {
                closerMaps.Visibility = Visibility.Collapsed;
                backButton.Visibility = Visibility.Collapsed;
                worldMap.Visibility = Visibility.Visible;
            }
            else
            {
                worldMap.Visibility = Visibility.Collapsed;
                //backButton.Visibility = Visibility.Visible;
                closerMaps.Visibility = Visibility.Visible;
                if (sivu == "northEurope")
                {
                    pohjoisEurooppaIso.Visibility = Visibility.Visible;
                }
            }
        }

        private void lukitseTausta(object sender, EventArgs e)
        {
            //grayBlur.blur();
        }

        private void vapautaTausta(object sender, EventArgs e)
        {
            //grayBlur.unBlur();
        }

        private void naytaTaulukko(object sender, EventArgs e)
        {
            contentPopup.Visibility = Visibility.Visible;
            List<Dictionary<string, string>> a = tietoHaku.palautaTiedot(nykyinenSivu);
            List<string> otsikot = new List<string>();
            otsikot = sisaltoObjekti["[ICP Top]"];
            contentPopup.naytaTaulukko(a, otsikot, sisaltoObjekti);
            contentPopup.vaihdaOtsikko(nykyinenSivu);
```

```
            contentPopup.vaihdaOtsikot(otsikot, sisaltoObjekti);
        }

        private void naytaKaavio(object sender, EventArgs e)
        {
            contentPopup.Visibility = Visibility.Visible;
            List<Dictionary<string, string>> a = tietoHaku.palautaTiedot(nykyinenSivu);
            contentPopup.naytaKaavio(a, sisaltoObjekti);
            contentPopup.vaihdaOtsikko(nykyinenSivu);
        }

        private void pohjoisEurooppa_MouseLeftButtonUp(object sender, MouseButtonEventArgs
e)
        {
            sivunVaihto("northEurope");
        }
    }
}
```

oikeaValikko.xaml

```
<UserControl x:Class="Atlas.components.oikeaValikko"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d"
    d:DesignHeight="300" d:DesignWidth="242">

    <Grid x:Name="LayoutRoot" Background="White">
        <Grid.RowDefinitions>
            <RowDefinition Height="Auto" />
            <RowDefinition Height="Auto" />
            <RowDefinition Height="Auto" />
            <RowDefinition Height="Auto" />
        </Grid.RowDefinitions>
        <!-- Näillä napeilla tehdään valintoja siitä mitä karttapohjalla näytetään -->
        <TextBlock Grid.Row="0" Name="asetus1" Height="35" Text="Show Trends" FontFami-
ly="verdanaNormi" MouseLeftButtonUp="clickHandler" FontSize="20" Cursor="Hand"/>
        <TextBlock Grid.Row="1" Name="asetus2" Height="35" Text="Show Profits" FontFami-
ly="verdanaNormi" MouseLeftButtonUp="clickHandler" FontSize="20" Cursor="Hand"/>
        <TextBlock Grid.Row="2" Name="asetus3" Height="35" Text="asetus3" FontFami-
ly="verdanaNormi" MouseLeftButtonUp="clickHandler" FontSize="20" Cursor="Hand"/>
        <TextBlock Grid.Row="3" Name="asetus4" Height="35" Text="asetus4" FontFami-
ly="verdanaNormi" MouseLeftButtonUp="clickHandler" FontSize="20" Cursor="Hand"/>
    </Grid>
</UserControl>
```

oikeaValikko.xaml.cs

```
using System;
using System.Windows.Controls;
using System.Windows.Input;

namespace Atlas.components
{
    public partial class oikeaValikko : UserControl
    {
        public oikeaValikko()
        {
```

```
            InitializeComponent();
        }

        public event EventHandler ShowTrends;
        public event EventHandler ShowProfits;

        protected void clickHandler(object sender, MouseButtonEventArgs e)
        {
            var blokki = (TextBlock)sender;
            switch (blokki.Name)
            {
                case "asetus1":
                    if (this.ShowTrends != null)
                        this.ShowTrends(new object(), new EventArgs());
                    break;
                case "asetus2":
                    if (this.ShowProfits != null)
                        this.ShowProfits(new object(), new EventArgs());
                    break;
            }
        }
    }
}
```

vasenValikko.xaml

```
<UserControl x:Class="Atlas.components.vasenValikko"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d"
    d:DesignHeight="208" d:DesignWidth="241">

    <Grid x:Name="LayoutRoot" Background="White">
        <Grid.RowDefinitions>
            <RowDefinition Height="Auto" />
            <RowDefinition Height="Auto" />
            <RowDefinition Height="Auto" />
        </Grid.RowDefinitions>
        <TextBlock Name="taulukkoButton" Grid.Row="0" Text="Show data chart" MouseLeftBut-
tonUp="taulukkoButton_clickHandler" FontFamily="verdanaNormi" FontSize="20" Cursor="Hand"/>
        <TextBlock Name="kaavioButton" Grid.Row="1" Text="Show graphical chart" MouseLeft-
ButtonUp="kaavioButton_clickHandler" FontFamily="verdanaNormi" FontSize="20" Cursor="Hand"/>
        <TextBlock Name="tietojaButton" Grid.Row="2" Text="Show other stuff" MouseLeftBut-
tonUp="tietojaButton_clickHandler" FontFamily="verdanaNormi" FontSize="20" Cursor="Hand"/>

    </Grid>
</UserControl>
```

vasenValikko.xaml.cs

```
using System;
using System.Windows.Controls;
using System.Windows.Input;

namespace Atlas.components
{
    public partial class vasenValikko : UserControl
    {
        public vasenValikko()
        {
            InitializeComponent();
        }
        public event EventHandler TAULUKKO;
        public event EventHandler KAAVIO;

        protected void taulukkoButton_clickHandler(object sender, MouseButtonEventArgs e)
        {
            if (this.TAULUKKO != null)
                this.TAULUKKO(new object(), new EventArgs());
        }

        protected void kaavioButton_clickHandler(object sender, MouseButtonEventArgs e)
        {
            if (this.KAAVIO != null)
                this.KAAVIO(new object(), new EventArgs());
        }
    }
}
```

sisaltoPopup.xaml

```
<UserControl x:Class="Atlas.components.sisaltoPopup"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:sdk="http://schemas.microsoft.com/winfx/2006/xaml/presentation/sdk"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:toolkit="http://schemas.microsoft.com/winfx/2006/xaml/presentation/toolkit"
    mc:Ignorable="d"
    d:DesignHeight="627" d:DesignWidth="924">

    <Grid x:Name="LayoutRoot" Background="White">
        <Grid Name="tietoIkkuna" Visibility="Visible" Height="594" Width="923">
            <Grid.RowDefinitions>
                <RowDefinition Height="Auto" />
                <RowDefinition Height="Auto" />
                <RowDefinition Height="Auto" />
                <RowDefinition Height="Auto" />
                <RowDefinition Height="Auto" />
                <RowDefinition Height="Auto" />
                <RowDefinition Height="Auto" />
            </Grid.RowDefinitions>
            <Grid.ColumnDefinitions>
                <ColumnDefinition Width="Auto"/>
                <ColumnDefinition Width="Auto"/>
                <ColumnDefinition Width="Auto"/>
                <ColumnDefinition Width="Auto"/>
```

```
                <ColumnDefinition Width="Auto"/>
                <ColumnDefinition Width="Auto"/>
                <ColumnDefinition Width="Auto"/>

            </Grid.ColumnDefinitions>
            <TextBlock Height="45" Grid.Row="0" Grid.Column="0" Text="Metso"
Name="tietoOtsikko" FontSize="38" FontFamily="Calibri" FontWeight="bold" Fore-
ground="#005D55"/>
            <TextBlock Height="45" Text="Otsikko 2" Name="otsikko2" FontSize="24" Font-
Weight="normal" FontStyle="normal" TextDecorations="none" Margin="316,7,147,204"
Grid.RowSpan="2" />
            <TextBlock Height="45" Grid.Row="4" Text="Otsikko 3" Name="otsikko3" Font-
Size="24" Margin="318,1,445,291" />

            <TextBox Grid.Row="1" Width="415" Height="180" Name="tekstiAlue1" Mar-
gin="316,17,147,15" />
            <TextBox Grid.Row="4" Width="415" Height="199" Name="tekstiAlue2" Mar-
gin="318,4,145,89" />

            <Image Grid.Row="1" Width="270" Height="212" Name="tietoKuva" Mar-
gin="0,0,608,0" />

            <Slider Grid.Row="2" Name="alkuSlider" ValueChanged="Slider1_ValueChanged"/>
            <Slider Grid.Row="3" Name="loppuSlider" Value-
Changed="Slider1_ValueChanged"/>
            <toolkit:Chart Grid.Row="1" Name="theChart">
            </toolkit:Chart>
        </Grid>
        <Grid Name="valinnat" Height="67" Width="878" Margin="0,505,46,55">
            <Grid.RowDefinitions>
                <RowDefinition Height="Auto" />
                <RowDefinition Height="Auto" />
                <RowDefinition Height="Auto" />
            </Grid.RowDefinitions>
            <Grid.ColumnDefinitions>
                <ColumnDefinition Width="Auto" />
                <ColumnDefinition Width="Auto" />
                <ColumnDefinition Width="Auto" />
                <ColumnDefinition Width="Auto" />
            </Grid.ColumnDefinitions>
            <Image Grid.Column="0" Grid.Row="0" Name="netSalesNuoli"
Source="../kuvat/valintanuoli.png" Width="30" Height="23" Cursor="Hand" />
            <TextBlock Grid.Column="1" Grid.Row="0" Text="Net Sales"
Name="netSalesValinta" FontSize="30" FontFamily="Calibri" FontWeight="normal" Fore-
ground="#005D55" Cursor="Hand" />
            <Image Grid.Column="0" Grid.Row="1" Name="newOrdersNuoli"
Source="../kuvat/valintanuoli.png" Width="30" Height="23" Cursor="Hand" />
            <TextBlock Grid.Column="1" Grid.Row="1" Text="New Orders"
Name="newOrdersValinta" FontSize="30" FontFamily="Calibri" FontWeight="normal" Fore-
ground="#005D55" Cursor="Hand" />
        </Grid>
    </Grid>
</UserControl>
```

sisaltoPopup.xaml.cs

```
using System;
using System.Collections.Generic;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Controls.DataVisualization.Charting;
using System.Windows.Data;
using Atlas.luokat;

namespace Atlas.components
{
    public partial class sisaltoPopup : UserControl
    {
        public sisaltoPopup()
        {
            InitializeComponent();
            this.DataContext = this;
        }

        private List<Dictionary<string, string>> dataSave = new List<Dictionary<string,
string>>();
        private Dictionary<string, List<string>> sisObjektiSave = new Diction-
ary<string, List<string>>();

        private int nykalkuPiste = 0;
        private int nykloppuPiste = 20;
        private int alkuPiste;
        private int loppuPiste;

        private List<kaavioSisalto> kaavioArray;
        private HaeTiedot tietoHaku = new HaeTiedot();

        public event EventHandler LUKITSE;
        public event EventHandler VAPAUTA;

        public void naytaTaulukko(List<Dictionary<string, string>> data, List<string>
otsikot, Dictionary<string, List<string>> sisObjekti)
        {
            //täysin dynaaminen taulukkototeutus ei onnistu käyttäen Silverlightin
datagridin omia ominaisuuksia
        }

        public void vaihdaOtsikot(List<string> a, Dictionary<string, List<string>> o)
        {
            List<string> YTD = o["YTD"];
            for (int i = 0; i < a.Count - 1; i++)
            {
                DataGridTextColumn uusiColumn = new DataGridTextColumn();
                uusiColumn.Header = a[i] + " " + YTD[i];
                uusiColumn.Binding = new Binding(a[i] + YTD[i]);
                //taulukko.Columns.Add(uusiColumn);
            }
        }

        public void naytaKaavio(List<Dictionary<string, string>> data, Diction-
ary<string, List<string>> sisObjekti)
        {
            dataSave = data;
            sisObjektiSave = sisObjekti;

            for (int i = 0; i < theChart.Series.Count; i++)
            {
```

```
             theChart.Series.RemoveAt(i);
            }

            List<string> YTD = sisObjekti["YTD"];
            List<string> ICP = sisObjekti["[ICP Top]"];
            List<string> MCT = sisObjekti["MCT000"];

            alkuPiste = 2;
            loppuPiste = data[0].Count - 5;
            alkuSlider.Minimum = alkuPiste;
            alkuSlider.Maximum = loppuPiste;
            loppuSlider.Minimum = alkuPiste;
            loppuSlider.Maximum = loppuPiste;

            bool textIsNumeric = true;
            String parseString = "";

            nykalkuPiste = Convert.ToInt32(alkuSlider.Value);
            nykloppuPiste = Convert.ToInt32(loppuSlider.Value);

            for (int i = 0; i < data.Count - 1; i++)
            {
                kaavioArray = new List<kaavioSisalto>();
                LineSeries newSeries = new LineSeries();
                newSeries.IndependentValueBinding = new Binding("Nimi");
                newSeries.DependentValueBinding = new Binding("Arvo");
                newSeries.SetBinding(LineSeries.ItemsSourceProperty, new Bind-
ing("Series" + i));

                int alkuLaskuri = 0;

                foreach (KeyValuePair<string, string> pair in data[i])
                {
                    if (alkuLaskuri >= nykalkuPiste && alkuLaskuri <= nykloppuPiste -
4)
                    {
                        textIsNumeric = true;
                        try
                        {
                            parseString = pair.Value;
                            parseString = parseString.Replace(" ", String.Empty);
                            int.Parse(parseString);
                        }
                        catch
                        {
                            textIsNumeric = false;
                        }

                        if (textIsNumeric == true)
                        {
                            kaavioArray.Add(new kaavioSisalto() { Nimi = pair.Key,
Arvo = Convert.ToInt32(parseString) });
                        }
                    }
                    alkuLaskuri = alkuLaskuri + 1;
                }
                newSeries.ItemsSource = kaavioArray;
                theChart.Series.Add(newSeries);
            }
        }
```

```
private void Slider1_ValueChanged(object sender, RoutedPropertyChangedEven-
tArgs<double> e)
        {
            nykalkuPiste = Convert.ToInt32(alkuSlider.Value);
            nykloppuPiste = Convert.ToInt32(loppuSlider.Value);
            naytaKaavio(dataSave, sisObjektiSave);
        }
    }
}
```

HaeTiedot.cs

```
using System;
using System.Collections.Generic;
using System.IO;

namespace Atlas.luokat
{
    public class HaeTiedot
    {
        public event EventHandler LADATTU;
        public Dictionary<string, List<string>> csvObjekti = new Dictionary<string,
List<string>>();

        public HaeTiedot()
        {}

        public List<Dictionary<string, string>> palautaTiedot(String s)
        {
            List<Dictionary<string, string>> sisaltoArray = new
List<Dictionary<string, string>>();
            List<string> kategoriat = new List<string>();

            switch (s)
            {
                case "northEurope":
                    kategoriat = new List<string> { "Northern Europe", "SD Finland",
"SD Scandinavia" };
                    break;
                case "europe":
                    kategoriat = new List<string> { "Northern Europe", "SD Finland",
"SD Scandinavia", "Great Britain", "DE", "PL", "BE", "NL", "Southern Europe", "SD
France", "Iberica", "Central Europe" };
                    break;
                case "asia":
                    kategoriat = new List<string> { "Asia Pacific MCT", "IN", "CN",
"Australasia", "Oceania", "Turkey", "Middle East", "Russia and CIS" };
                    break;
                case "africa":
                    kategoriat = new List<string> { "Western North Africa", "Eastern
North Africa", "Southern Africa", "Africa West", "Africa Central" };
                    break;
                case "northAmerica":
                    kategoriat = new List<string> { "North America MCT", "SD USA",
"US", "CA", "Central America" };
                    break;
                case "southAmerica":
                    kategoriat = new List<string> { "South America MCT", "BR", "South
Cone", "Andean Area", "North West", "South West" };
                    break;
```

```
                case "maailma":
                    kategoriat = new List<string> { "EMEA", "Americas", "Asia Pacific
MCT" };
                    break;
                default:
                    kategoriat = new List<string> { "nollaArpa" };
                    break;
            }

            List<string> otsikot = new List<string>();
            otsikot = csvObjekti["[ICP Top]"];
            List<string> YTD = csvObjekti["YTD"];

            for (int i = 0; i <= kategoriat.Count-1; i++)
            {
                Dictionary<string, string> sisalto = new Dictionary<string,
string>();

                for (int j = 0; j <= csvObjekti[kategoriat[i]].Count-1; j++ )
                {
                    List<string> testiLista = new List<string>();
                    testiLista = csvObjekti[kategoriat[i]];
                    if (sisalto.ContainsKey(otsikot[j] + YTD[j]))
                    {}
                    else
                    {
                        sisalto.Add(otsikot[j] + YTD[j], testiLista[j]);
                    }
                }
                sisaltoArray.Add(sisalto);
            }
            return sisaltoArray;
        }

        public Dictionary<string, List<string>> palautaObjekti()
        {
            return csvObjekti;
        }

        public void ajaUpdate()
        {
            string ladattuData;
            string fileLoc = "G:/Softat/Silverlight/Atlas/Atlas/data/taulukko.csv";

            using (StreamReader sr = new StreamReader(fileLoc))
            {
                ladattuData = sr.ReadToEnd();
            }
            parseMyData(ladattuData);
        }

        private void parseMyData(string data)
        {
            List<string> rows = new List<string>(data.Split('\n'));
            int i = 0;

            foreach (String arvo in rows)
            {
                List<string> rivi = new List<string>(arvo.Split(';'));
                List<string> solut = new List<string>();

                foreach (String solu in rivi)
                {
```

```
                    solut.Add(solu);
                }

                String trimstring = solut[0];

                trimstring = trimstring.Trim();

                if (trimstring != "")
                {
                    csvObjekti.Add(trimstring, solut);
                }
                i++;
            }
            if (this.LADATTU != null)
                this.LADATTU(new object(), new EventArgs());
        }
    }
}
```

kaavioSisalto.cs

```
namespace Atlas.luokat
{
    public class kaavioSisalto
    {
        public string Nimi { get; set; }
        public int Arvo { get; set; }
    }
}
```