TAMPEREEN AMMATTIKORKEAKOLU
Tietokonetekniikan koulutusohjelma
Ohjelmistotekniikka

Tutkintotyö

Kai Fagerlund

# VEKTORIGRAFIIKAN KONVERTTERI

Työn ohjaaja Ohjelmistotekniikan lehtori Tony Torp
Työn teettäjä Tampereen ammattikorkeakoulu
Tampere 2006

TAMPEREEN AMMATTIKORKEAKOLU

Tietotekniikka
Ohjelmistotekniikka
Fagerlund, Kai
Vektorigrafiikan konvertteri
Tutkintotyö 36 sivua +1 liitesivu
Työn ohjaaja Tony Torp
Työn teettäjä Tampereen ammattikorkeakoulu
Toukokuu 2006
Hakusanat Symbian, Java, Series60, Vektorigrafiikka

## TIIVISTELMÄ

Työn tarkoitus oli tehdä ohjelma jota voitaisiin käyttää hyödyksi Symbian-ohjelmoinnin opetuksessa. Tämä sovellus generoi Symbiangrafiikka koodia sillä piirretyistä kuvioista. Valmis koodi voidaan siirtää sellaisenaan Symbian sovellukseen. Ohjelma tulisi säästämään opettajan ja opiskelijoiden työaikaa Symbian opinnoissa. Ohjelma auttaisi myös hahmottamaan grafiikan ohjelmointia.

Ohjelma toteutettiin Java-kielellä, koska Javan ja Symbian grafiikkakirjastot ovat lähes samanlaiset, niiden muuttaminen Javasta Symbianiin ei ole vaikeaa, muutamaa poikkeusta lukuunottamatta. Java mahdollistaa myös helpon kehitysympäristön graafisille sovelluksille ilman lisäkirjastoja sekä sisältää hyvän pohjan käyttöliittymien tekoon.

Työn tuloksena on valmis ohjelmisto, joka täyttää vaaditut tavoitteet ja mahdollistaa myös sen, että ohjelman runkoa voidaan täydentää tulevissa opinnäytetöissä.

TAMPERE POLYTECHNIC

Information technology
Software engineering
Fagerlund, Kai
Vectorgraphics converter
Engineering Thesis 36 pages, 1appendices
Thesis Supervisor Tony Torp
Commissioning Company Tampere Polytechnic
May 2006
Keywords Symbian, Java, Series60, Vector graphics

ABSTRACT

The main principle of the work was to design and create utility program to ease graphical programming in Symbian platform. This utility program is planned to be used in Symbian courses to save time for more essential parts of Symbian programming studies. Graphics programming itself is mainly calculating coordinates for various shapes. This it self is time consuming and somewhat pointless for programming studies.

The program is written in Java-language. This was chosen because of the similarities between Java's and Symbian's graphics libraries. Second Java has comprehensive library for creating user interfaces.

As the result of the work was a software, that filled requirements which were settled at the start of the project. This software's base can be used later in upcoming final thesis.

SISÄLLYSLUETTELO

# 1 JOHDANTO

SymbianPiirto on apuohjelma Symbian-kielen graafisten ohjelmien tekemiseen. Piirretystä kuvasta generoidaan Symbian-grafiikkakoodi, joka voidaan liittää suoraa Symbian-ohjelman lähdekoodiin. Ohjelmalla voidaan piirtää Symbian tuntemat peruskuviot, joita ovat viiva, neliö, ympyrä ja polygonit. Ensimmäinen kehitysversio ei sisällä pyöreä Kulmaista neliötä, kaaren, sektorin, tekstin piirtoa eikä kynän tyylin vaihtoa.

Symbianin grafiikan ohjelmointi tapahtuu koordinaateilla, kuten osittain Javassakin, mikäli ei käytetä bittikarttoja. Tästä johtuen grafiikan piirto on hidasta ja vaivalloista. Lopullista kuvaa on vaikea hahmottaa pelkästä koodista, ja kuvan katsominen välillä vaatii tekeillä olevan ohjelman kääntämisen. Tämä apuohjelma on tehty nopeuttamaan grafiikan piirtoa, ja se mahdollistaa myös monimutkaisempien kuvien piirtämisen.

Ohjelman tärkein ominaisuus on kuitenkin tulostaa piirretyn kuvan Symbian kielinen ohjelmakoodi. Tätä varten täytyi toteuttaa jokaiselle kuvioluokalle metodi, joka palauttaa tekstityyppisenä muuttujana vastaavanlaisen ohjelmarivin kuin mitä Javalla toteutettu kuvio vastaisi Symbianin ymmärtämässä muodossa.

Työn keskeisenä osana on Java sovellutuksen luomisen lisäksi perehtyminen kahden ohjelmointikielen grafiikan piirtoon niiden sisältämillä grafiikka-kirjastoilla. Koska ohjelma kääntää Javalla kirjoitettua grafiikkakoodia Symbianin vastaavaksi, oli oleellista verrata näiden kahden tyyppikohtaisia eroja. Perehdyin myös Symbianin, Series60:en ja Javan historiaan sekä näiden perusominaisuuksiin.

Kappale 2 käsittää Symbianin sekä S60 historian, niiden yleisimmät ominaisuudet sekä tutustutaan perus kuvioiden piirtoon sekä kuinka niiden ulkoasuun voidaan vaikuttaa. Javan osalla käydään nämä samat asiat kappaleessa 3. Ohjelman käyttöliittymä esitellään kappaleessa 4. Kappaleessa 4 on esillä myös puskuroitupiirto sekä värimallit. Toiminnallisuus ja toteutus kappale kertoo taas

ohjelman sisäisestä toiminnasta ja ohjelman luokista. Kappale 6:ssa käydään läpi työssä huomattuja havaintoja sekä kerrotaan työn tuloksista. Yhteenvedossa kerrotaan työnkulusta.

# 2 SYMBIAN

Symbian on mobiililaitteille suunniteltu käyttöjärjestelmä. Sitä kehittää vuonna 1998 perustettu Symbian Ltd., jonka omistajina ovat mm. Nokia, Ericsson, Motorola, Panasonic ja Psion. Perustajajäseniä näistä olivat Nokia, Psion ja Ericsson. Symbian on kehitetty Psionin valmistaman EPOC-käyttöjärjestelmän pohjalta. Symbianin sovellukset voidaan kirjoittaa eri kielillä, joista yleisimmät ovat C++ ja Java; muita vaihtoehtoja ovat Python, Perl, Simkin, Visual Basic ja OPL.

Symbian on rakenteeltaan lähellä pöytäkoneiden käyttöjärjestelmiä kuten Windows ja Linux. Samankaltaisuuksia näiden välillä ovat muun muassa ennakoiva moniajo, jota kutsutaan myös keskeyttäväksi moniajoksi (pre-emptive multitasking), säikeiden moniajo (multithreading) ja muistin suojaus (Memory protection).

Symbianin ohjelmoinnissa on huomioitava mobiililaitteisiin liittyvät rajoitukset; Muistin koko ei välttämättä ole suuri ja laitteet saattavat olla yhtäjaksoisesti päällä kuukausia. Tämän takia on tärkeää, että Symbian-ohjelmassa tulee huomioida seuraavanlaisia asioita: Muistin käytön tulee olla vähäistä, muisti vuodot ja niiden mahdollisuudet tulee pitää minimissään. Vaikka tämä on miltei itsestään selvää myös muidenkin kielien ja laitteiden ohjelmoinnissa.

Laitteiden massa muisteina toimii yleisesti flash-muistikortti, joten niidenkin koko on rajallisempi kuin esimerkiksi pöytäkoneissa. Tästä syystä turhaa tallettamista tai varastointia tulee välttää.

Symbianin ohjelmat ovat myös tapahtumapohjaisia, eli prosessori kytketään pois päältä silloin kun ohjelmaa ei käytetä. Tämä säästää laitteiden akkuja jotka, voivat kulua nopeasti loppuun, mikäli ohjelmia ajetaan turhaan. /2/,/9/

## 2.1 Series 60

Series 60 eli S60 on Symbianin pohjalta tehty sovellusalusta ja on tällä hetkellä maailman johtavin sovellusalusta älypuhelimille. S60:n ovat lisensoineet useat johtavat puhelinvalmistajat kuten Nokia, Siemens, Panasonic, korealaiset Samsung ja LG electronics sekä Suomessa vähemmän tunnetut valmistajat Sendo ja Lenovo.

Tyypillisiä ominaisuuksia S60-sarjan puhelimille on 176*208 pikselin näyttö. Normaalit puhelimen ominaisuudet ovat että puhelimella voidaan soittaa sekä vastaanottaa puheluita. Puhelimen mallista riippuen käytössä on joko Bluetooth tai infrapuna, joita voidaan käyttää esimerkiksi tiedostojen lähettämiseen toisiin matkapuhelimiin tai yhdistämään matkapuhelin tietokoneeseen. Internetin käyttö-mahdollisuus on WAP tuella, sekä uudemmissa puhelimissa on myös tuki XHTML:lle. Viestien lähettämisen sekä vastaan ottamisen mahdollisuus, viestien tuetut tyypit ovat SMS (Short Messaging Service) eli tavallinen tekstiviesti, MMS (Multimedia Messaging Service) jolla taas voidaan tekstiin lisätä sisältää ja ääntä. Mahdollisuus lukea sekä lähettää sähköpostia (eMail) matkapuhelimella. IM (Instant Message), jolla tarkoitetaan viestien lähetystä reaaliaikaisesti kuten keskusteluohjelmissa tietokoneella, tästä esimerkkinä Microsoftin Messenger-ohjelma. Useissa S60 sarjan puhelinmalleista löytyy myös digitaalikamera sekä erinäisiä soittimia musiikille, radiot ja nauhurit sekä kuvalle että äänelle. S60:stä on tällä hetkellä kaksi kehitysversiota, Series 60 2nd ja 3rd editiot.

Series 60 2nd editio julkaistiin vuonna 2003, ja sitä käytettiin ensimmäisen kerran Nokian 6600-puhelimessa. Uudistuksia ensimmäisestä versiosta ovat muun muassa parannettu käyttöliittymä, Java MIDP 2.0 tuki, nettiselain joka tukee HTML 4.01-, XHTML-, MP- ja WAP CSS-standardeja sekä muita vastaavanlaisia parannuksia ohjelmistoihin. 2nd editiota on laajennettu

jälkeenpäin myös kolmella täydennys paketilla. Ensimmäisessä paketissa lisättiin tuki EDGE-verkolle, ja selaimeen lisättiin tuki small screen HTML:lle. Toisessa paketissa tuli WCDMA- ja CDMA-verkko tuki, ja viimeisessä kolmannessa paketissa skaalattava käyttöliittymä ja mahdollisuus suurempiin näytön resoluutioihin.

Series 60 3rd editio on uusin versio S60-järjestelmästä, ja se sisältää kaikki toisen edition ja sen täydennyspakettien ominaisuudet. Pääpaino kolmannen kehitys-version kohdalla kuitenkin on parannetuissa multimediaominaisuuksissa ja parannetussa alustan arkkitehtuurissa.

Ensimmäinen S60:tä käyttänyt matkapuhelin oli Nokian 7650. Muita Nokian malleja ovat 3650,6600,7610 ja Nokian pelipuhelimet N-Gage sekä sen uudempi versio N-Gage QD. Muiden valmistajien malleja on esimerkiksi Siemensin SX1 ja Sendon X, näissä malleissa on myös erikoista se, että ne eivät käytä standardia näytön kokoa, vaan näissä se on kooltaan 176x220 pikseliä. /2/,/6/

## 2.2 Symbian sovelluskehys

Symbian OS:n sovelluskehyksen kaksi tärkeintä osaa ovat UIKON ja standardi EIKON. EIKON on kirjasto graafisten käyttöliittymien tekoon ja UIKON on käyttöliittymä ja ohjauskehys, joka on yleinen kaikille Symbian OS:ille. UIKON ja EIKON tarjoavat useita käyttöliittymä komponentteja sekä vastaavat ohjelman käynnistymisestä. Symbian perussovellus koostuu neljästä luokasta, jotka ovat seuraavissa alikappaleissa.

### 2.2.1 Application (sovellus)

Tämä olio luodaan aluksi sovelluksen käynnistyttyä, ja sen tehtävänä on vastata lopun koodin alustuksesta.

### 2.2.2 Document (dokumentti)

Luokan tehtävä on luoda oletusdokumentti silloin kun ohjelma käynnistetään ensimmäisen kerran. Tätä dokumenttia käytetään sovelluksen tilan tallentamiseen

ja lataamiseen, kun ohjelma suljetaan ja käynnistetään uudelleen. Tosin kaikkien ohjelmien ei tarvitse tallettaa tietoja käyttökertojen välillä, mutta dokumentti luokan tehtävänä on kuitenkin luoda ja alustaa käyttöliittymäluokka.

### 2.2.3 Aplication UI

Luokka on käyttöliittymä luokka, ja se vastaa ohjelmalle tulevista tapahtumista kuten näppäin- ja käyttöliittymän lähettämät komennot. Luokan tehtävänä on myös siirtää ohjelma taustalle, jos jokin muu ohjelma käynnistetään tai otetaan käyttöön. Tämä luokka luo myös luokan view (näkymä), tai useampia sellaisia jos sille on tarvetta.

### 2.2.4 View

Luokka hallitsee käyttöliittymänäkymää ja siten vastaa kaikesta, mitä käyttäjä näkee ruudulla. View-luokka on sinällään ainoa näistä luokista, jolla on tekemistä tämän työn kanssa, koska Symbian-Piirto-ohjelmaa käytetään vain grafiikka-koodin generointiin. Tähän luokkaan kuuluu metodi Draw(). Tämä metodi vastaa kaikesta grafiikasta, mikä ruudulle piirretään, ja se kutsutaan aina kun näytöllä tapahtuu jotakin, kuten avataan popup-valikko tai peliohjelmassa liikutetaan jotakin pelimerkkiä tai hahmoa. /9/

## 2. 3 Symbianin CWindowGC- luokka

Kaikki piirto tapahtuu luokan CWindowGC:n kautta, joka on periytetty luokasta CGraphicsContext. Se taas on abstrakti alustariippumaton rajapinta graafiseen sisältöön. CWindoGC-luokka sisältää piirtometodit esimerkiksi viivan, pisteen, neliön ja ympyrän piirtoon, myös bittikarttojen käyttö on mahdollista. Kynän koko, väri ja tyyli voidaan myös muuttaa. Grafiikka piirretään koordinaattipohjaisesti siten että 0,0 koordinaatti sijaitsee vasemmassa yläkulmassa ja päättyy oikeaan alakulmaan koordinaateissa 176, 146. Tämä edellä määritetty alue on siis se osa S60-sarjan puhelimesta, johonka grafiikkaa voidaan perussovellutuksessa piirtää, sillä osa ruudun koosta kuluu sovelluksen valikkoihin. Koko näytön koko on 176 kertaa 208 pikseliä. /2/

## 2.4 Esimerkkejä kuvioiden piirrosta

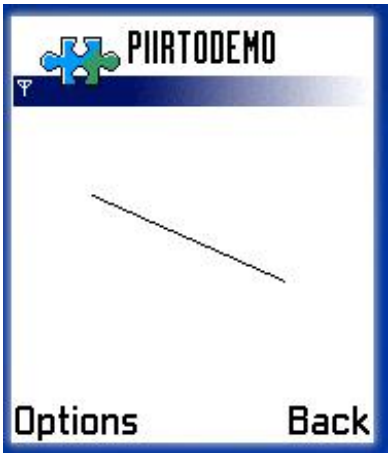
Seuraavaksi tarkastellaan hieman peruskuvioiden piirtämistä CWindowGC-luokan avulla. Symbian-sovelluksessa grafiikkakoodi sijoittuu sovelluskehyksen Wiew-luokasta löytyvän Draw-metodiin. Esimerkeissä ei ole näkyvissä muuta kuin ne ohjelmarivit, jotka tulevat edellä mainittuun kohtaan Symbian sovellusta. Kaksi ensimmäistä esimerkkiä käsittelee metodia normaalin suoran viivan piirtämiselle ja kuinka sen väriä ja paksuutta voidaan muuttaa. Kolmannessa ja neljännessä esimerkissä tarkastellaan nelikulmion piirtämisestä sekä sitä, kuinka kuvion täyttöväri toteutetaan.

### 2.4.1 Esimerkki viivan piirrosta

Piirretään musta viiva pisteestä 39,43 pisteeseen 133,85. Ensimmäinen rivi määrittää kynän värin, joka tässä tapauksessa oli musta. Väri määritetään RBG-arvona, josta kerrotaan lisää kappaleessa 3.1.4.1. Kynän väriä ei tarvitse alustaa joka kerta, vaan vain silloin kun väriä halutaan vaihtaa. Näin ollen määritetty väri säilyy kunnes se muutetaan. Toinenohjelma rivi määrittää kynän tyypin. Kynän tyyppi viivaa piirrettäessä on ESolidPen, joka on peruskiinteä yhtenäinen jälki.

Lopuksi näiden määritysten jälkeen piirretään itse viiva, joka siis tapahtui kahden koordinaattipisteen välillä: ensimmäinen (TPoint(39,43) on aloituspiste, ja viiva päättyy pisteeseen TPoint(133,85). Ensimmäinen suluissa oleva luku on x-koordinaatti ja jälkimmäinen on y-koordinaatti. Mikäli tämän jälkeen halutaan piirtää lisää tämän tyyppisiä viivoja, ei väriä tai kynän tyyppiä tarvitse määritellä enää uudestaan, vaan riittää että lisätään pelkkä DrawLine-metodi.

```
gc.SetPenColor(TRgb(0,0,0));
gc.SetPenStyle(CWindowGc::ESolidPen);
gc.DrawLine(TPoint(39,43),TPoint(133,85));
```

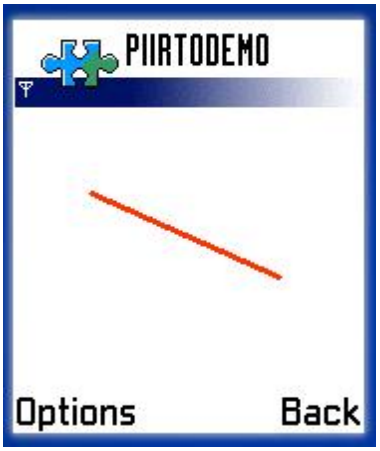



Kuva 1. Viivaesimerkki 1

### 2.4.2 Esimerkki värin ja kynän paksuuden muuttamisesta

Muutetaan edellisen esimerkin viivan väri punaiseksi ja tehdään viivasta paksumpi. Tässä esimerkissä ensimmäinen rivi määrittää jälleen kynän värin, joka nyt muutetaan punaiseksi määrittämällä RBG-arvon ensimmäinen luku 255:ksi, r on siis punainen väri, tässä esimerkissä on myös lisätty sinistä väriä ja näin saadaan aikaan punaisen eri sävy. Kynän tyyppi pysyy samana kuin esimerkissä 1. Viivan paksuus, joka nyt muutetaan kolmen pikselin levyiseksi SetPenSize-metodilla. Viimeisenä piirretään viiva, jolla on samat alku- ja loppu-koordinaatit kuin esimerkissä 1.

```
gc.SetPenColor(TRgb(255,51,0));
gc.SetPenStyle(CWindowGc::ESolidPen);
gc.SetPenSize(TSize(3,3));
gc.DrawLine(TPoint(39,43),TPoint(133,85));
```

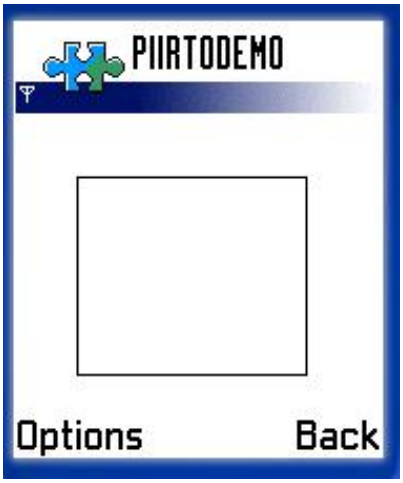



Kuva 2. Viivaesimerkki 2

### 2.4.3 Esimerkki neliön piirrosta

Piirretään tavallinen reunaviivallinen neliö ilman täyttöväriä. Kuvioiden reunaviiva piirretään kynällä eli asetetaan kynä halutun väriseksi, tässä esimerkissä mustaksi. Kun kuvioon ei haluta täyttöväriä, alustetaan siveltimen tyyppi ENullBrushiksi, tämä sivellintyyppi ei piirrä mitään, joten tällä tavoin estetään halutessa kuvion täyttöväri. Neliön piirto tapahtuu DrawRect-metodilla, jolle annetaan TRect-tyypin muuttuja. TRect typin muuttujalle annetaan neljä lukuarvoa, jotka määrittävät neliön rajat. Kaksi ensimmäistä lukua ovat vasemman yläkulman koordinaatit, josita kuvion piirto alkaa, ja kaksi jälkimmäistä ovat kuvion loppukoordinaatit eli vasemman alakulman piste.

```
gc.SetPenColor(TRgb(0,0,0));
gc.SetBrushStyle(CWindowGc::ENullBrush);
gc.DrawRect(TRect(30,30,140,125));
```




Kuva 3. Neliöesimerkki 1

### 2.4.4 Kuvion täyttövärin lisääminen

Muutetaan neliön tyyppi sellaiseksi, että lisätään edelliseen neliöön vihreä täyttöväri ja poistetaan kuvion reunaviiva. Määritetään siveltimen väri vihreäksi. koska kyseessä on nyt täyttöväri, käytetään tämän takia SetBrushColor-metodia SetPenColorin sijasta. Seuraavaksi poistetaan kuviosta reuna viiva asettamalla kynän tyyppi ENullPeniksi. Tämä tyyppi asettaa tyylin sellaiseksi, että kynä ei piirrä mitään. Reunaviiva voidaan poistaa myös astettamalla kynän väri samaksi kuin siveltimen väri. Jotta saadaan piirrettyä kuvio, joka on yhtenäisen värinen, asetetaan siveltimen tyyppi ESolidBrushiksi. Tämän tyyppinen sivellin mahdollistaa kuvion täyttövärin. Lopuksi piirretään itse kuvio, jolla on samat alku ja loppukoordinaatit kuin aikaisemmassa esimerkissä.

```
gc.SetBrushColor(TRgb(0,255,0));
gc.SetPenStyle(CWindowGc::ENullPen);
gc.SetBrushStyle(CWindowGc::ESolidBrush);
gc.DrawRect(TRect(30,30,140,125));
```


Kuva 4. Neliöesimerkki 2

# 3 JAVA

Java on Sun Microsystemsin kehittämä ohjelmointikieli. Sen kehittely alkoi vuoden 1990 paikkeilla, ja tarkoituksena oli alun perin tehdä ohjelmisto sulautettujen järjestelmien verkottamista varten. Tätä varten kehittyi ohjelmointikieli oak, joka huomattiin www:n yleistymisen myötä sopivaksi kyseiseen ympäristöön. Javasta tuli nopeasti suosittu, mikä johtui sen samankaltaisuudesta C++:aan mutta helpommasta kieliopista. Java on myös laitteistoriippumaton, joten sillä kirjoitettuja ohjelmia voidaan ajaa useilla eri alustoilla; tämä vauhditti myös kielen kasvua. Näitä alustoja ovat kämmentietokoneet, dynaamiset www-sivut, palvelinpuolen ohjelmistot sekä tietenkin matkapuhelimet. Java on oliopohjainen tulkittava kieli; tulkittavalla kielellä tarkoitettaan sellaista ohjelmointikieltä, jossa lähdekoodia ei käännettä suoraan konekielelle, vaan se käännetään tavukoodiksi, joka ajetaan virtuaalikoneen kautta.

Vuonna 1995 julkaistiin Javan ensimmäinen versio 1.0, ja samana vuonna Netscape julkaisi Java-tuellisen selaimen. Tästä vuoden päästä Microsoft julkaisi Internet Explorerin 3.0-version, jossa oli myös Java tuki. Javan yksi oleellinen ominaisuus on muistinhallintaan liittyvä muistikeräin, jota kutsutaan myös roskien keruuksi, tämä vapauttaa muistia sitä mukaa kuin jotain sinne talletettua ei enää tarvita. /1/,/8/

## 3.1 Javan grafiikan piirto

Kaikki Javan käyttöliittymäkomponenttien ja grafiikan piirtoon liittyvät luokat sisältyvät Java.AWT-pakkaukseen. Tämän luokkakoosteen sisällön avulla siis luodaan Java-sovelluksen käyttöliittymät sekä luodaan grafiikkaa tai kuvia. Grafiikan piirron kannalta näistä luokista tärkeimmät ovat Graphics, Graphics2D sekä Color.

Graphic-luokka on abstrakti perusluokka, jonka kautta kaikki grafiikallinen sisältö luodaan. Tämä sisältää kaiken graafisen sisällön piirtämisen joko suoraan käyttöliittymäkomponentteihin, kuten JPaneliin, tai sitten kuten tässäkin työssä piirretään ensin kuva puskuriin eli niin kutsuttuun off-screen imageen tämä tarkoittaa, että graafinen sisältö piirretään ensin muistiin ja sitten vasta tuodaan esille näyttöön. Tämä luokka sisältää samankaltaisia kuviometodeja kuin Symbianin CWindowGC-luokka, kuten viiva (drawLine), neliö (drawRect) ja ellipsi (drawOval).

Graphics-luokkaa laajennetaan Graphics2D-luokalla, tämän luokan avulla voidaan muuttaa ja lisätä efektejä Graphics-luokan metodeilla piirrettyihin kuvioihin, sekä se sisältää erilaisia metodeja kaksiulotteisten kuvioiden renderöintiin ja koordinaattimuunnoksiin. SymbianPiirto-ohjelman kannalta näistä tärkein oli kynän paksuutta muuttava SetStroke-metodi. Graphics2d-luokan avulla voidaan myös piirtää tekstiä drawString-metodin avulla, joka on ainoa draw-tyypin metodi jota ei ole Graphics luokassa. Luokan avulla voidaan lisätä kuvioihin myös täyttöväri, joka ei itsessään ole kovin hyödyllinen, sillä täyttövärillisille kuvioille on jo olemassa valmiit metodit Graphics-luokassa. Mutta jos halutaan käyttää vaikka esimerkkinä liukuväristä täyttöä, tehdään se erillisellä Graphics2D-luokan metodilla.

Color-luokan oliota käytetään tallettamaan värejä RBG-värimallin muodossa. Luokka sisältää yhteensä 13 valmista perusvärioliota, mutta sillä voidaan luoda myös uusia väriolioita itse valitsemalla RBG-arvoilla. Väriolioita voidaan käyttää muuttamaan komponentin pohjaväriä tai asettamaan piirrettävän kuvion väriä.

Tämä tapahtuu välittämällä olio parametrina jollekin väriin vaikuttavalle metodille, kuten kynän värin muuttava setColor tai komponentin taustavärin muuttava setBackground. Color-luokassa on myös metodeja, joilla voidaan palauttaa jonkin komponentit väri tai jokin väriolion RBG-komponentin lukuarvo.

Itse kuvioiden piirto tapahtuu välittämällä Javan paint-metodille Graphics-olio, jonka kautta piirrettään kuviot. Paint-metodin sisään kirjoitetaan haluttu grafiikka-koodi. /1/

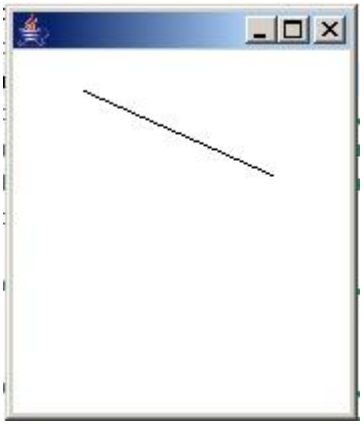
## 3.2 Esimerkkejä kuvioiden piirrosta

Piirretään samat kuviot kuin kappaleessa 2.2.3, jossa käsiteltiin kuvioiden piirtoa Symbianin CWindowGC-luokalla hyödyntämällä Javan Graphics-, Graphics2D- ja Color-luokkia. Näitä esimerkkejä varten kirjoitettiin lyhyt Java-sovellus, joka sisältää pelkästään samankokoisen piirtopinnan kuin SymbianPiirto ohjelmassa. Esimerkkien yhteydessä esitetään vain ne koodirivit, joilla itse piirto tapahtuu paint-metodissa.

### 3.2.1 Esimerkki viivan piirrosta

Piirretään samanlainen viiva kuin aikaisemmassa esimerkissä. Piirretään musta viiva pisteestä 39,43 pisteeseen 133,85. Tästä huomataan, että Javalla tapahtuva piirto on paljon lyhyempi, sillä mustaa väriä ei tarvitse alustaa koska se on vakio-piirtoväri. Sama koskee myös kynän tyyppiä, jota ei tarvitse erikseen määrittää; viiva on vakiona yhden pikselin levyinen.

```
g.drawLine(39,43,133,85);
```



Kuva 5. Viivaesimerkki Javalla

### 3.2.2 Värin ja kynän tyypin vaihtaminen

Kuten Symbianin esimerkissä, muutetaan kynänväri punaiseksi ja tehdään viivasta paksumpi. Koska kynän tyypin ja koon muuttamiseen tarvittavat metodit sijaitsevat Graphics2D-luokassa, ja paint-metodille on määritetty parametriksi Graphics-olio, tulee ensimmäisenä tehdä tyyppimuutos Graphics2D-olioksi. Tyyppimuunnoksen jälkeen voidaan käyttää tämän luokan metodeja. Javassa piirtoväri muutetaan setColor-metodilla, tälle annetaan parametrinä Color-tyypin olio. Tässä esimerkissä luodaan uusi väriolio jolle annetaan samat RBG-arvot kuin aikaisemmassa Symbian esimerkissä (luku 2.4.2). kuten Symbianissa, tämä piirtoväri säilyy niin kauan kunnes se vaihdetaan. Seuraavaksi metodilla setStroke vaikutetaan kynän jäljen paksuuteen, ja kuten värin kanssa, on tämäkin niin kauan voimassa kunnes se vaihdetaan. Lopuksi piirretään itse viiva.

```
Graphics2D g2 = (Graphics2D) g;
g2.setColor(new Color(255,51,0));
g2.setStroke(new BasicStroke(3));
g2.drawLine(39,43,133,85);
```

Kuva 6. Viivaesimerkki 2 Javalla

### 3.2.3 Esimerkki neliön piirrosta

Neliön piirtäminen tapahtuu Javassa drawRect-metodilla. Parametreiksi sille annetaan vasemman ylänurkan koordinaatti, josta kuvion piirto alkaa. Esimerkissä tämä piste on 30,30, luvuista ensimmäinen on x-akselin koordinaatti ja toinen y-akselin. Kaksi viimeistä lukua ovat kuvion leveys ja korkeus,
Symbianissa kuviolle annetaan myös loppupiste koordinatteina, ja tämä on merkittävä ero Javan ja Symbianin grafiikan piirrossa. Näistä eroista kerrotaan tarkemmin kappaleessa 6.1 Eroja Symbianin ja Javan piirtometodeista ja grafiikan piirrosta. Koska halutaan piirtää sama kuvio kuin esimerkissä 2.4.3, täytyy matemaattisesti laskea kuvion korkeus ja leveys, jotta saadaan kuvio samanlaiseksi. Vähennetään kuvion oikean alanurkan koordinaateista, mihin kuvio päättyy, kuvion alkupisteen koordinaatit. Tällä tavalla saadaan korkeus ja leveys vastaavanlaisiksi. Vaihtoehtoisesti voidaan kuvio piirtää myös Graphics2D draw-metodilla, tällä päästään samaan lopputulokseen; tosin nyt tarvitsee lisätä edellisessä esimerkissä ollut tyyppimuunnos.

```
g.drawRect(30,30,140-30,125-30);
```

Tai vaihtoehtoisesti:

```
Graphics2D g2 = (Graphics2D) g;
g2.draw(new Rectangle(30,30,140-30,125-30));
```

Kuva 7 Neliöesimerkki Javalla

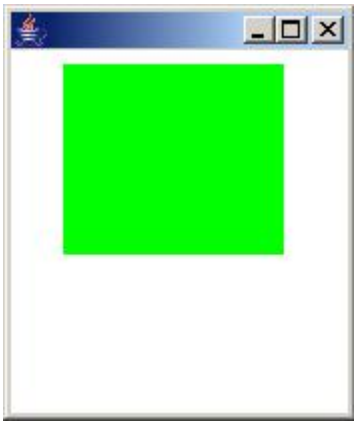
### 3.2.4 Kuvion täyttövärin lisääminen

Täyttövärin lisääminen kuvioon on huomattavasti helpompaa kuin Symbianissa, sillä Graphics-luokka sisältää täyttövärillisille kuvioille omat metodinsa. Muutetaan edellistä esimerkkiä siten, että muutetaan väri vihreäksi ja vaihdetaan drawRect-metodi fillRect-metodiin, jotka ovat muuten toiminnaltaan identtiset pois lukien täyttöväri. Täyttöväri voidaan toteuttaa myös Graphics2D-luokan fill-metodilla.

```
g.setColor(new Color(0,255,0));
g.fillRect(30,30,140-30,125-30);
```

Tai vaihtoehtoisesti:

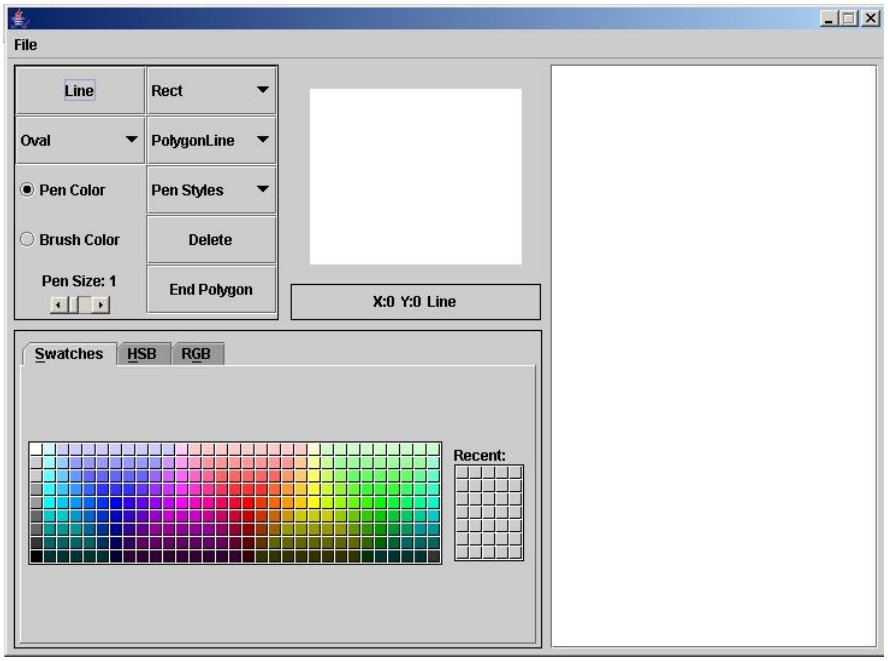
```
g2.setColor(new Color(0,255,0));
Rectangle r = new Rectangle(30,30,140-30,125-30);
g2.draw(r);
g2.fill(r);
```



Kuva 8 Neliöesimerkki 2 Javalla

# 4 OHJELMAN KÄYTTÖLIITTYMÄ

Käyttöliittymä on tehty hyödyntämällä Java.AWT-pakkauksen sisältämiä käyttöliittymäkomponentteja. Komponentit on sijoitettu frameen siten, että jokaiselle käyttöliittymän osalle tehtiin ensin oma paneeli (JPanel) ja nämä paneelit asetettiin frameen. Paneeleille annettiin rajat setBounds-metodeilla, joten käytössä ei ole vakioasemoijia. Paneeli, johon työkaluvalikot, pen- ja brush color-valintanapit sekä pen size-liukuri asetettiin, on ainoa poikkeus, tässä paneelissa on käytössä gridLayout-asemoija.




Kuva 9 Käyttöliittymä

Käyttöliittymän keskeisin osa on vasemmasta yläkulmasta alkava paneeli, joka sisältää alasvetovalikot kuviotyökalun valitsemiseksi sekä oman napin viiva-työkalulle. Alasvetovalikot on ryhmitelty kuviotyyppien mukaan, jotka ovat neliöt, ympyrät ja polygonit. Tässä paneelissa on myös kaksi valintanappia (radiobutton), joilla määrätään, ollaanko vaihtamassa kynän tai siveltimen väriä. Nämä napit on ryhmitetty siten että vain toinen voi olla valittuna.

End polygon-painiketta taas käytetään silloin kun polygonin kulmien koordinaatit on annettu ja kuvion piirto halutaan lopettaa. Kun tätä painiketta painetaan, piirtyy silloin määritetty polygoni piirtopinnalle sen värisenä ja viivan paksuuden mukaan, mitkä on valittu.

Delete-painike poistaa aina viimeiseksi piirretyn kuvion. Kynän piirtoviivan paksuutta voidaan muuttaa alhaalla sijaitsevasta liuku säätimestä (scrollbar). Aina kuvion piirtämisen loputtua tarkistetaan, onko kynän kokoa muutettu kuvioiden välissä, ja generoidaan tarvittava Symbian-koodi koon määrittämiseksi sen muuttuessa.

Kuvassa näkyy myös valikko kynän tyyleille, joita tähän ohjelman kehitysversioon ei vielä tehty, mutta tämä valikko lisättiin jo valmiiksi käyttöliittymään. Valikko ei itsessään sisällä minkäänlaista toiminnallisuutta.

Keskellä käyttöliittymää, piirtopinnan alapuolella näkyy laatikko jossa esimerkki kuvassa on teksti X:0 Y:0 Line. Tässä laatikossa on tiedot siitä missä koordinaateissa piirtopintaa hiiren kursori on, sekä sillä hetkellä valittuna oleva piirtotyökalu.

## 4.1 Piirtopinta

Piirtopinta sijaitsee keskellä käyttöliittymää. Pinnan koko on 176 kertaa 145 pikseliä, vaikka series60 sarjan puhelimien näytön koko on 176 kertaa 208 pikseliä. Tämä aikaisemmin todettu koko on siis se, mikä on käytettävissä

varsinaiseen piirtoon, koska tavallisissa EIKON-sovellukset vaativat ylhäältä ja alhaalta osan ruudun koosta käyttöjärjestelmän omiin valikoihin.

## 4.2 Piirtopinnan toteutus

Piirtopinta toteutettiin normaalilla Javan JPanelilla, joka siis on perus käyttöliittymäkomponentti, johon voidaan sijoittaa käyttöliittymäkomponentteja tai, kuten tässä tapauksessa, piirtää grafiikkaa.

Tähän kyseiseen piirtopaneeliin liitettiin MouseListener tapahtumakuuntelija joka, mahdollistaa paneelin käyttämisen hiirellä. Kun hiiren vasen nappi painetaan paneelin kohdalla pohjaan, alkaa kuvion piirto joka voidaan muokata halutunlaiseksi raahaamalla kuvion loppukoordinaatti toivottuun pisteeseen. Ohjelma näyttää tämän ajan kuvion haamukuvan, jonka avulla voidaan hahmottaa kuvio, jota halutaan piirtää. Hiiren napin vapauttamalla piirtyy kuvio piirtopinnalle, ja tämän jälkeen voidaan piirtää seuraava kuvio. Hiiren vapautuksen yhteydessä tarkistetaan, onko piirtoväriä/värejä vaihdettu. Tapauksessa, jossa väriä on vaihdettu, tulostuu vaadittava Symbian-koodi tekstikenttään. Hiiren näppäimen vapauttamisen yhteydessä talletetaan luotu kuvio-olio niille varattuun taulukkoon.

### 4.2.1 Puskuroitu piirto

Puskuroitu (buffered) piirto mahdollistaa grafiikan piirron ja esittämisen reaaliajassa siten että kuva ei välky. Tästä on hyötyä esimerkiksi animaatioissa tai tässä työssä niin kutsutun haamu-kuvion piirrossa. Tämä on toteutettu siten että kuva piirretään ensin puskuriin muistiin ja sieltä kuva kopioidaan näytölle, ja näin tehdään aina kun kuvassa tapahtuu muutos.

Puskuroidussa piirrossa on tosin haittapuolensa kuten että se kuormittaa prosessoria, lisää muistin käyttöä, kuormittaa laiteväylää (system bus), ja ohjelmoija saattaa huomaamattaan tehdä tehotonta koodia, koska ei ole täyttä varmuutta, piirretäänkö halutut asiat todellakin vain kerran.

Vilkkumattomalle grafiikalle on olemassa toisenlaisiakin ratkaisuja. Tässä työssä oli aluksi käytössä tapa, joka käytti hyödyksi Javasta löytyvää setXORmodea. Tästä tosin oli haittaa, koska silloin täytyi piirtää aina kaksi kuviota ensin haluttu kuvio ja kuvion muuttuessa täytyi se peittää käänteisvärillä. Toinen syy tämän menetelmän hylkäämiseen oli, että Double Buffered-piirto on yleisemmin käytetty, mikä taas helpottaa jatkokehittäjien työtä.

### 4.3 Tulostusalue

Valmis Symbian-koodi tulostetaan oikeassa reunassa olevalle tekstikentälle. Teksti kenttää voidaan vierittää alas ja ylös kuten Windowsin tai muiden graafisten käyttöjärjestelmien ikkunoita, jos kaikki ikkunassa tai paneelissa olevat asiat eivät mahdu kerralla kuvaan. Tekstikentästä voidaan kopioida teksti normaaliin tapaan ja siten siirtää haluttuun paikkaan kuten tämän ohjelman kanssa Symbian-lähdekoodiin.

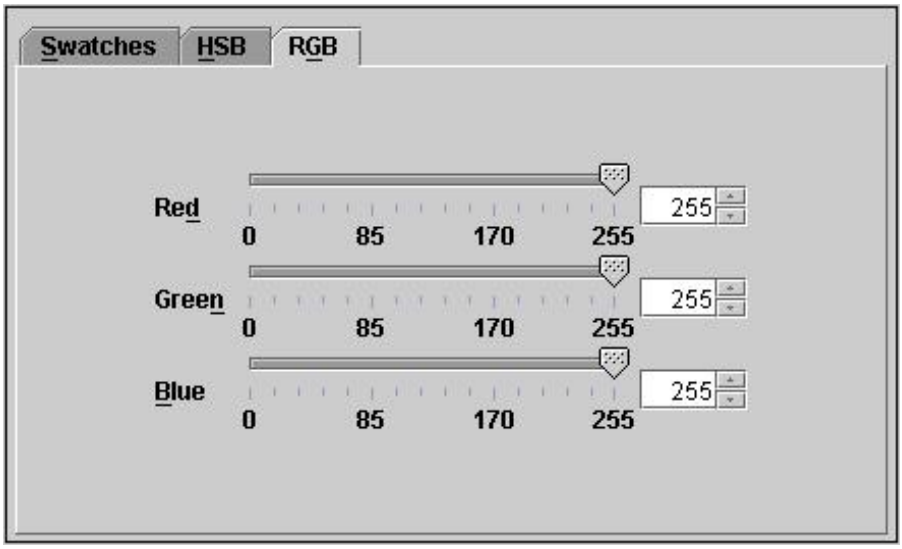
### 4.4 Värinvalintapaneeli ja värimallit

Värit voidaan valita alakulmassa olevalla värinvalintapaneelilla. Painikkeet joilla valitaan kynä tai sivellin, määrää siis sen kumman värin tämä paneeli vaihtaa. Värinvalintapaneelissa on kolme välilehteä, joissa jokaisessa on erilainen värinvalintamalli, oletuksena tässä on käyttöliittymän kuvassakin näkyvä Swatches-tyyppi jossa, on valmiina 279 valmista värisävyä. Tästä paneelista valitut värit tallentuvat myös recent-kohtaan, josta nämä värit voidaan valita uudestaan. Kahden muun välilehden kuvat näkyvät kuvissa 10 ja 11. Värioliot käsitellään kuitenkin aina RBG-muodossa, joten muilla värimallilla luodut värit muutetaan lopulta tähän muotoon.

Väripaneelia on muutettu siten, että oletusarvoisesti siihen kuuluu myös esikatselupaneeli, jolla voidaan esimerkiksi katsoa etukäteen sopiiko valittu taustaväri tekstin väriin. Tämän ohjelman kannalta tätä esikatselua ei vielä tarvittu, joten se on poistettu käytöstä.

### 4.4.1 RGB-värimalli

RGB värimallia käytetään sekä Javan että Symbianin värien määritykseen. Yleisimmin tunnettu käyttö on televisioissa sekä monitoreissa. Kirjainyhdistelmä on lyhenne sanoista red green ja blue eli punainen vihreä ja sininen. Nämä kolme pääväriä yhdistämällä saadaan aikaan erilaiset värit. Numerollisesti esitettynä väri on muotoa 0,0,0, joka tässä tapauksessa on musta. Arvojen tulee olla välillä 0-255, jossa siis 0,0,0 tarkoittaa sitä että kyseinen väri on musta ja 255,255,255 on valkoinen. Esimerkkinä keltainen väri saadaan jättämällä sininen väri arvoon nolla ja antamalla punaiselle ja vihreälle sävylle arvo 255. RBG-värimallissa voidaan esittää yhteensä 16 miljoonaa erilaista väriä. Kuvassa 10 on kaikki värikomponentit asetettu 255:ksi joten väri on täysin valkoinen. /7/

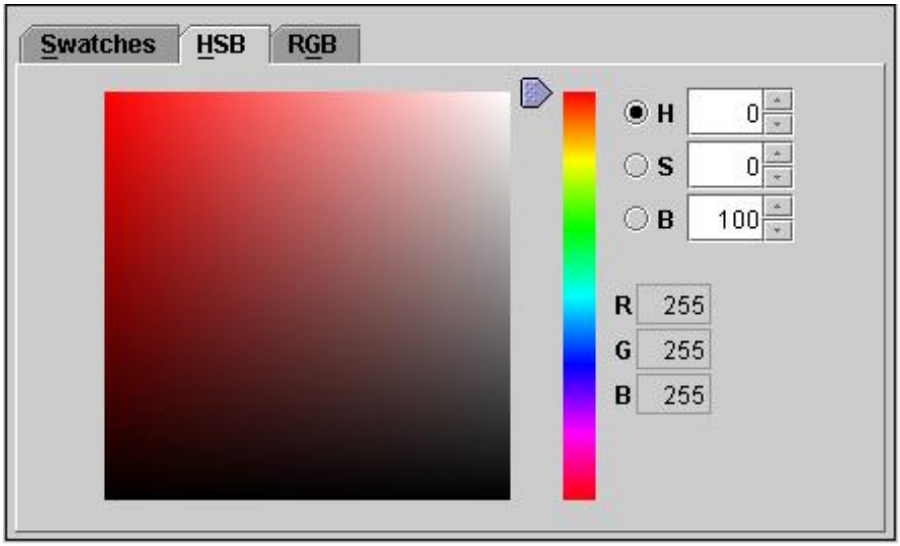



Kuva 10. RBG-värivalintavalikko

### 4.4.2 HSB-värimalli

HSB on matemaattinen tapa ilmoittaa väri, tässä menetelmässä väri hajotetaan kolmeen komponenttiin: H hue eli vivahde, tämä on itse puhdas väri. Huen arvo määritellään väriympyrän mukaan, ja asteikko on välillä 0:sta 360 asteeseen. Jokainen kulma-aste vastaa eri väriä näistä esimerkkinä punainen on joko 0 tai 360, keltainen sijaitsee 60 asteen kohdalla ja vihreä 120. S on saturation eli kylläisyys, joka määrittää, kuinka ”puhdas” väri on. Tämä arvo annetaan

prosentteina siten, että 0 prosenttia on valkoinen ja 100 on siis tämä puhdas väri. B on viimeinen arvo ja se tarkoittaa kirkkautta. Kirkkauden arvo on myös prosentti-luku ja mitä suurempi se on, sitä kirkkaampi on väri. HSB-värimallissa on näkyvissä myös kyseisen värin RBG-arvo. /7/




Kuva 11. HSB- värimalli

## 5 TOIMINNALISUUS JA TOTEUTUS

Ohjelma koostuu kolmesta luokasta, sekä yhden näistä shape-luokan aliluokista. Luokat ovat SymbianPiirto joka on pääluokka; Luokka Draw vastaa haamukuvion piirrosta, ja Shape-luokasta periytetään kuvio-luokat jotka ovat: Line, Square, fillSquare, outlinedFillSquare, Oval, fillOval, outlinedFillOval, polygonLine, PolygonClass ja fillpolygon.

Ohjelman luokkakaavio löytyy liitteestä 1. Luokkakaavion luokissa ei ole esitettynä läheskään kaikkia ohjelman luokkien muuttujia, sillä kaavio olisi paisunut liian suureksi. Shape-luokan alaluokkien laatikot on jätetty tyhjiksi, sillä niiden metodit ja attribuutit ovat identtiset itse Shape-luokan kanssa.

Ohjelman toimintoja ohjataan, kuten Javalla toteutetuissa käyttöliittymissä yleensä, tapahtumakuuntelijoiden kautta. Näistä kuuntelijoista käytössä ovat

Actionlistener, MouseListener, MouseMotionListener, AdjustmentListener ja ChangeListener.

ActionListener vastaa nappien ja alasvetovalikoiden toiminnasta. Näistä komponenteista lähteneet herätteet muuttavat sellaisia ohjelman attribuutteja, jotka kertovat esimerkiksi, mikä kuviotyyppi on valittuna. Poikkeuksena ovat delete- ja endpolygon-painikkeet. Delete-painikkeen herätteestä poistetaan oliotaulukosta viimeisin olio ja lasketaan ohjelman kirjaa pitävistä laskureiden arvoa yhdellä. Näitä laskureita on kuvioiden lukumäärälle sekä sille, kuinka mones samanlainen kuvio on piirretty peräkkäin. Kuvioiden lukumäärän tietäminen on tärkeää, koska tämän laskurin avulla käsitellään taulukkoa, johon kuvio-oliot on talletettu. Näihin komponentteihin, joita tällä kuuntelijalla tarkkaillaan, kuuluu myös valintanappiryhmä, jolla valitaan, ollaanko vaihtamassa kynän vai siveltimen väriä.

MouseListener kuuntelee hiirtä. Tämän kuuntelijan alle kuuluu viisi eri kuuntelija metodia joista käytössä tämän ohjelman kanssa on kolme: mouseCliked, tarkistaa onko hiirellä klikattu jotakin. MousePressed, tarkkailee painetaanko hiiren näpäin pohjaan. MouseReleased, saa herätteen kun hiiren näppäin vapautetaan. Kaksi muuta metodia, jotka eivät ole käytössä ovat mouseEntered ja mouseExited. Kuten nimetkin kertovat näiden herätteet tulevat silloin kuin hiiren kursori siirtyy jollekin alueelle tai siltä pois.

Ohjelmassa mouseListener kuuntelija on liitetty piirtopintaan. Kun hiiren näppäintä painetaan piirtopinnalla otetaan talteen sen pisteen koordinaatit. Talletetut koordinaatit välitetään draw-luokalle, jotta tämä luokka tietää mistä pisteestä kuvion piirto alkaa. Hiiren näppäimen vapauttaminen taas lopettaa kuvion piirron ja tämän herätteen jälkeen luodaan juuri piirretty kuvio olio. Molemman herätteen yhteydessä otetaan siis talteen joko alku tai loppu koordinaatit ja ne välitetään kuvio-olion rakentajalle. Hiiren klikkauksia kuunnellaan kahdessa tapauksessa, nämä ovat polygonin piirron yhteydessä ja muiden työkalujen ollessa käytössä piirretään pelkkä piste.

MouseMotionListener on toinen hiireen liittyvä kuuntelija. Tähän kuuntelijaan kuuluu kaksi metodia mouseDragged ja mouseMoved. MouseDragged heräte tapahtuu silloin kun hiirtä ”raahataan” jollakin komponentilla. Ensin siis hiiren näppäin painetaan pohjaan jonka jälkeen hiirtä liikutetaan. mouseMoved on samankaltainen kuin mouselistenerin mouseEntered ja saa siis herätteen silloin kun hiiren kursori liikkuu jollekin alueelle mutta näppäimiä ei ole painettu. Ohjelmassa on käytössä mouseDragged jonka avulla draw luokka osaa piirtää ja muuttaa piirrettävää kuvaa reaaliaikaisesti. Kuuntelija on yhdistetty piirtopintaan. Aina hiirtä ”raahatessa” päivittyy sen hetkinen kursorin koordinaatti loppu pisteelle varattuihin x ja y muuttujiin. Tämän herätteen alla on myös tarkistus jotta osataan käyttää oikeanlaista draw-luokan piirto metodia, jotta piirretään valittua työkalua vastaava kuvio.

AdjustmentListener kuuntelijaa käytetään tutkimaan onko jokin säädettävän komponentin arvo muuttunut. Ohjelmassa näitä komponentteja on vain yksi jolla vaihdetaan kynän paksuutta. Tämän herätteen alla vaihdetaan kynän paksuutta vastaava muuttuja samaksi miksi se on säädetty liukukytkimellä.

ChangeListenerin tehtävä on tarkkailla onko jokin sille määritetty komponentti vaihtanut tilaansa. SymbianPiirto-ohjelmassa tätä käytetään JcolorChooser-komponentin kanssa. Silloin kun paneelista valitaan uusi väri, vaihdetaan ohjelman jompikumpi väri muuttuja samanlaiseksi. Tähän vaikuttaa siis se onko valittu vaihdettavaksi kynän vai siveltimen väri.

Jokaisella kuviotyypillä on oma tunniste numero. Tämä Numero on tyypiltään tavallinen integer-luku. Työkalua vaihdettaessa annetaan muuttujan arvoksi valitun kuviotyypin tunniste jotta ohjelma tietää minkälaista kuviota ollaan piirtämässä. Tieto tästä välitetään kuvion piirron aikana Draw-luokalle jotta käytetään oikeaa piirto-metodia. Piirron päätyttyä luodaan oikeanlainen kuvio-olio tunnisteen mukaa.

Aina kuvion piirron alkaessa tarkistetaan onko piirtoväriä vaihdettu, jotta tämä värinvaihto näkyy myös generoituvassa Symbian koodissa. Tarkistus tehdään

pääohjelmassa, mutta metodi värin tulostukseen sijaitsee kuvio alaluokissa. Kynän paksuuden muutos tarkistetaan samalla tavalla, mutta tulostus tapahtuu suoraan pääohjelmassa.

Ohjelma pitää myös kirjaa siitä kuinka mones samanlainen kuvio on piirretty ja välittää tämän tiedon kuvioluokalle, tämän tarkoitus johtuu siitä että Symbianin vaativat kynän ja siveltimen tyypit tulostuvat koodin vain kerran eikä joka kerta kun kuvio piirretään, tämä säästää turhia koodirivejä. Työkalua vaihdettaessa nollataan tämä muuttuja. Näin ollen ohjelma tietää että seuraava kuvio on uuden tyylinen, ja kuvion Symbian koodia palauttaessa osaa aliluokka palauttaa myös tarvittavat rivit kynän ja siveltimen muutosta varten. Tämä tarkistus sisältyy jokaiseen Shape-luokan alaluokkaan ja sijaitsee Print-metodissa.

## 5.1 SymbianPiirto-luokka

Pääluokkana toimii SymbianPiirto-luokka joka vastaa käyttöliittymän luonnista, tapahtumien käsittelystä, valmiiksi piirrettyjen kuvioiden Symbian-koodin tulostuksesta tekstikenttään. Tapahtuman käsittelyyn sisältyy normaalien käyttöliittymäkomponenttien toiminta, kuten nappien (JButton) painamisesta tulevat herätteet sekä piirtopinnalle piirtäminen. Luokka sisältää myös pääohjelman ja se luo muut luokat ohjelman käynnistyksen yhteydessä.

Luokka sisältää useita muuttujia. Useita näistä käytetään kuitenkin vain tiedon hetkelliseen tallettamiseen, nämä tiedot eivät välity muille luokille. Tämän tyyppisiä muuttujia käytetään etupäässä vertailujen tekoon, kuten siinä, onko väriä vaihdettu kuvioiden piirron välillä. Oleellisimmat muuttujat ovat ne, jotka välitetään ohjelman muille luokille kuten hiiren koordinaatit, kynän paksuus ja valitut värit. Tärkein muuttuja on taulukko, johon piirretyt kuvio-oliot talletetaan.

Pääluokalla on yksi oma metodi, picture, jolla oliotaulukkoon tallennetut oliot piirretään uudestaan, mikäli kuvaa tulee päivittää. Picture-metodi käsittelee kuvio-oliotaulukon läpi siten, että kuviot piirretään uudelle bufferedImagelle joka

näytetään sen jälkeen piirtopinalla. Tässä yhteydessä tulostetaan myös tekstimuotoinen Symbian-koodi tekstikenttään.

## 5.2 Draw-luokka

Draw-luokka vastaa haamukuvion piirrosta silloin, kun uutta kuviota piirretään piirtopinnalle. Draw-luokka luodaan heti käynnistyksen yhteydessä. Luokassa on jokaiselle kuviolle oma metodi, joka piirtää ja palauttaa SymbianPiirto-luokalle bufferedImagen. Tämä piirretään sitten piirtopinnalle. Draw-luokalle välitetään myös sillä hetkellä jo näkyvissä oleva kuva bufferedImagena. Tämä mahdollistaa bufferoidun piirron siten, että jo valmiiksi piirretyt kuviot eivät katoa. Tämän vaiheen väliin jättäminen johtaisi siihen, että uudelleen tehty bufferedImage ei sisältäisi kuin viimeisenä piirretyn kuvion. Kuvion piirron alussa välitetään kynän ominaisuudet sekä väri Draw-luokalle.

Jokaiselle piirtometodille välitetään alkupisteen lisäksi sen pisteen koordinaatit, johon hiiri on liikutettu. Draw-luokan alla piirretään siis kuva, tämä ratkaisu mahdollistaa kuvioiden renderoinnin Graphics2D-luokan avulla, ja tätä kautta pystytään vaikuttamaan esimerkiksi viivan paksuuteen.

Luokan tärkeimmät metodit ovat edellä mainitut bufferedImage-tyyppiset piirtometodit ja setBuffer-metodi, joka asettaa piirtopinnalle piirretyn kuvan puskurin taustakuvaksi.

## 5.3 Shape-luokka

Shape-luokka on abstrakti luokka, josta periytetään kaikki muut kuviot. Se ei itsessään sisällä muuta kuin oletusrakentajan ja metodien esittelyt. Metodien toteutukset ovat aliluokissa, ja niiden toiminta vaihtelee kuvion tyypistä riippuen. Jokaisella aliluokalla on kuitenkin metodit itse piirtoa varten sekä käännetyn Symbian koodin palautusta varten.

### 5.4 Shape-luokan alaluokat

Line-olio on tavallinen suora viiva, josta voidaan muuttaa paksuutta ja väriä. Luokalle välitetään tiedot, minkä paksuinen kynä on valittu, mikä oli valittu väri ja kuinka mones line tyypin viiva on piirretty, sekä tietenkin viivan alku- ja loppukoordinaatit. Shape-luokan rakentajan takia luokalle välitetään myös tieto siveltimen väristä- vaikka itse Line luokka ei sitä tarvitsekaan- sillä Drawline metodiin ei vaikuta täyttöväri.

Oval-, FillOval-, OutLinedFillOval-, Square-, fillSquare- ja outlinedFillSquare-luokat ovat lähes samanlaiset, merkittävin ero näiden välillä näkyy piirto-metodissa, jossa vaihtuu ainoastaan piirrettävän kuvion tyyppi. Piirtometodissa tehdään myös tarkastukset, mihin suuntaan kuviota lähdetään ohjelmassa piirtämään, jotta kuvio piirtyy oikein sitä piirrettäessä. Käytännössä joudutaan siis matemaattisesti laskemaan koordinaattimuutos, jotta kuvio piirtyy kursorilla määritettyyn paikkaan, alkaen siitä mistä kuvion piirto alkoi ja loppuen pisteeseen, missä hiiri vapautettiin piirtopinnalla.

Kappaleessa 4 KÄYTTÖLIITTYMÄ kerrottiin polygonien piirron eroista muihin kuvioihin nähden. Merkittävin ero on siinä, että polygoneja piirrettäessä annetaan kuviolle useita koordinaattipisteitä, joiden kautta kuvion ääriviivat piirtyvät. Näin ollen välitetään polygonLine, fillPolygon ja polygonClass aliluokille taulukko, johon määritetyt pisteet on tallennettu. Tämä tapahtuu vasta endpolygon-painikkeen painamisen jälkeen.

### 5.5 Testaus

Työssä ei ollut käytössä testaussuunnitelmaa eikä mitään erikoisempia testejä ajettu. Ohjelman toiminnallisuutta tarkasteltiin ja testattiin siis koko ajan työn edetessä. Poikkeuksien käsittelyjä sisältyy muutamaan kohtaan, talletukseen ja siihen, ettei kuvio-olioiden määrä kasva taulukkoa suuremmaksi, johon edellä mainitut oliot talletetaan.

# 6 TULOKSET

Työn tarkoituksena oli suunnitella ja toteuttaa Java-sovellus, jolla voidaan piirtää piirtopinnalle Symbian CWindowGC:n peruskuviot ja tulostaa näitä kuvioita vastaava Symbian-koodi. Ohjelmalla tuli myös olla helppo selkeä käyttöliittymä. Nämä tavoitteet saavutettiin, tosin joitakin ominaisuuksia jouduttiin karsimaan projektin etenemisen yhteydessä. Ohjelman runkoa on myös tarkoitus käyttää myöhemmissä opinnäytetöissä, joten näitä karsittuja ominaisuuksia on mahdollista lisätä myöhemmin. Karsituista ominaisuuksista lisää kappaleessa 6.2 Jatkokehitys.

## 6.1 Eroja Symbianin ja Javan piirtometodeissa ja grafiikan piirrossa

Työssä huomattu merkittävin ero koskee neliöiden ja ovaalien piirtoa. Symbianissa neliön ja ovaalin piirtometodille annetaan alku- ja loppu-koordinaatit, alkaen vasemmasta yläkulmasta ja päättyen oikeaan alakulmaan, kun taas Javassa annetaan vain alkupiste koordinaattina ja sen jälkeen kuvion pituus ja korkeus. Ohjelman piti kuitenkin piirtää kuviot kuten Symbianissa joten, näille kuviolle täytyi laskea oikea pituus ja korkeus, joka vastasi vastaavanlaista kuviota koordinaattipohjaisesti piirrettynä.

Tästä johtuen ohjelmaan tuli tehdä tarkistukset, koska ohjelma mahdollistaa myös piirron vasemmalta oikealle ja alhaalta ylöspäin, sekä näiden yhdistelmät, jotta kuvio piirtyy itse ohjelmassa oikein. Tämä vaikutti myös Symbian-koodin generointiin. Koska vasemman yläkulman koordinaatin on oltava aloituspiste ja oikean alakulman lopetus, täytyi siis varmistaa että nämä pisteet menivät oikein päin generoidussa koodissa.

Toinen ero on kuvioiden täytössä: Javassa täytetylle neliölle, ovaalille ja polygonille löytyy omat metodinsa, Symbianissa näiden kuvioiden täyttö taas tapahtuu muuttamalla siveltimen tyyppiä. Symbian mahdollistaa myös kuvioiden reunaviivojen piirron, tämänlaisia kuvioita ei Javan grafiikkakirjastosta löydy

suoraan, joten ne täytyi piirtää yhdistämällä kaksi kuviota yhdeksi. Tämä toteutettiin siten, että ensin piirrettiin kuvio, joka oli täytetty, ja sen päälle piirrettiin samankokoinen kuvio, jota ei ollut täytetty, ja tämä kuvio toimi reunaviivallisen kuvion ääriviivoina.

Eroavaisuus on myös siinä, että Javassa grafiikan piirtoon liittyviä luokkia on useita, kuten kappaleessa 3.2 kerrotaan. Symbianissa nämä kaikki vastaavat asiat on koostettu CWindowGC-luokan alle.

## 6.2 Jatkokehitys

Jatkokehityksen ensimmäisessä vaiheessa voisi lisätä kynän tyypin vaihdon mahdollisuuden, Symbianin tukemia erilaisia kynätyyppejä on kuusi jotka ovat näkyvillä alla olevassa taulukossa.

| | |
|---|---|
| ESolidPen | peruskynä joka piirtää yhtenäisen viivan |
| EDottedPen | piirtää pisteviivan |
| EDashedPen | piirtää katkoviivan |
| EDottDashPen | tyylissä on pisteitä ja katkoviivoja vuorotellen |
| EDotDotDashPen | vuorotteleva viiva ja pistepari |
| ENullPen | ei piirrä mitään |

Taulukko 1 Kynäntyypit

Toiseen kehitysversioon tulisi myös lisätä seuraavat kuviot: DrawArc, joka piirtää osan ellipsin kehästä. DrawPie, jolla piirretään sektori ellipsistä. DrawRoundRect on neliö jonka kulmat on pyöristetty. DrawText ja DrawTextVertical, joita käytetään tekstin tulostamiseen piirtopinnalle. DrawTextVertical tulostaa tekstin pystysuoraan ylhäältä alaspäin.

Ohjelman olisi hyvä jatkossa mahdollistaa myös tuki bittikartoille siten, että piirretty kuva olisi mahdollista tallettaa suoraan bittikarttana, tai että ohjelmaan voitaisiin ladata valmis bittikartta.

Mahdollisuus toteuttaa valmiita animaatioita olisi myös graafisten sovelluksen tekoa helpottava ja nopeuttava toiminto. Kuviolle annettaisiin aloituspiste ja piste, johon sen tulisi siirtyä. Tämän pohjalta ohjelma generoisi valmiin koodin ajastimille ja muille animaation tarvitsemille asioille, kuten tarvitseeko kuvion muuttaa muotoaan tai väriään animaation aikana.

Myöhemmässä vaiheessa voitaisiin myös liittää ohjelmaan joko jokin valmis 3D-grafiikkakirjasto tai tehdä siihen kokonaan oma. Tämä mahdollistaisi 3D-kuvioiden käytön sekä niiden käyttäytymisen myös kolmannessa ulottuvuudessa. Käyttöliittymästä olisi voinut tehdä kompaktimman. Nyt käyttöliittymä on toteutettu Javan peruskäyttöliittymäkomponenteilla niiden oletusmuodoissa. Myöhemmin ominaisuuksia lisättäessä voidaan törmätä tilan puutteeseen, joten työkalujen näppäimiä on syytä pienentää.

# 7. YHTEENVETO

Työn lopputuloksena oli valmis ohjelmisto, joka täytti sille asetetut vaatimukset. Sisällöllisesti työ osoittautui kiinnostavaksi sen sisältäessä monipuolisia osa-alueita, kuten käyttöliittymän suunnittelu ja sen toteuttaminen. Itseäni on aina kiinnostanut graafinen ohjelmointi, joten tarkempi tutustuminen työhön liittyneiden ohjelmointiympäristöjen grafiikan piirtoon oli mielenkiintoista. Uutena asiana olivat myös Symbianin ja Javan taustat ja historia, joihin en aikaisemmin ollut tutustunut. Tämä osa alue tosin tuli mukaan työhön vasta tämän raportin kirjoittamisvaiheessa.

Suurin osio työtä oli itse piirto-ohjelman suunnittelu ja toteuttaminen. Siihen sisältyi useita osa-alueita, kuten käyttöliittymän suunnittelu ja sen toteuttaminen. Yksi valintakriteeri siihen, että ohjelma kirjoitettiin Javalla, oli että se sisältää helppokäyttöiset luokkakirjastot valmiille käyttöliittymäkomponenteille. Näin ollen pystyttiin nopeasti ja helposti toteuttamaan käyttöliittymä ja keskittymään ohjelman toiminnallisiin osiin.

Seuraava työvaihe oli toteuttaa piirtopinta. Tätä varten valittiin menetelmäksi puskuroitu piirtomenetelmä, joka mahdollisti kuvioiden samanaikaisen esittämisen näytöllä; näin uusi kuvio ei pyyhkinyt pois edellistä piirrettyä kuviota.

Lisäksi toteutustavan avulla voitiin säilyttää kukin kuvio omana olionaan, ja tätä kautta niiden muokkaaminen on mahdollista jälkeenpäin. Piirtopinnan vaativin osa-alue oli reaaliaikaisen apukuvan piirtämisen toteutus, joka on vastaavanlainen kuin yleisesti tunnetuissa kuvankäsittelyohjelmissa. Tämä tarkoittaa sitä, että kuviota piirrettään, näyttää ohjelma mallin siitä mitä ollaan tekemässä, eli kuvio muuttuu sitä mukaa kuin hiirtä liikutetaan piirtopinnalla.

Pelkän kuvion tulostuksen toteutus ei ollut vaikeaa, vaan ongelmaksi tässä osiossa osoittautui se, että vältyttiin ylimääräisien ohjelmarivien tulostukselta. Tämä johtui siitä, että ohjelman tuli antaa myös tarvittavat tulostukset värin, kynän tyypin ja mallin muutoksiin. Ratkaisuksi toteutettiin tulostusmetodeihin tarkistuksien lisääminen, onko edellä mainittuja muutoksia tapahtunut, joten pääohjelman tuli aina välittää kuvioluokalle sen hetken valitut piirtotyökalun ominaisuudet.

Työn edessä huomasin, kuinka helposti ohjelmat voivat paisua työmäärältään liian suuriksi, varsinkin kun kyseessä on yhden henkilön toteuttama projekti. Tämän takia ohjelmasta jouduttiin karsimaan joitakin ominaisuuksia, jolla ei sinällään ollut merkitystä. Yhtenä työn tarkoituksena oli, että ohjelman runkoa käytettäisiin tulevissa opinnäytetöissä. Karsittuja ominaisuuksia voidaan siis lisätä myöhemmin ohjelman laajentuessa.

Haasteellinen osa työstä oli juuri toteuttaa laajempi ohjelmistokokonaisuus kokonaan itse. Huomasin, kuinka paljon asioita onkaan huomioitava, ja kuinka paljon aikaa ja virheitä säästyy panostamalla suunnitteluun enemmän. Näin ollen kuvioluokkien määrää olisi voinut laskea joko käyttämällä hyödyksi graphics2D-luokan renderointiominaisuuksia tai yhdistämällä samantyyliset kuviot yhteen luokkaan ja tekekmällä tarkistukset paint-metodin alle siitä, minkälainen kuvio halutaan piirtää.

Käyttöliittymän olisi voinut toteuttaa tekemällä se jollakin tähän tarkoitukseen tehdyllä editorilla tai apuohjelmalla. Kuitenkin päädyin tekemään sen suoraan ohjelmoimalla, koska tarkoitukseen sopivan editorin löytäminen ja sen käytön opettelu olisi vienyt yhtä paljon aikaa kuin menetelmä, jolla käyttöliittymä on nyt toteutettu.

Toisaalta projektin toteuttaminen kokonaan itse mahdollisti sen, että pääsin tekemään kaikki osa-alueet siten kuin ne halusin toteuttaa. Opintojen kannalta oli myös hyvä nähdä ja tehdä kaikkia projektin osa-alueita, kuten suunnittelua, käyttöliittymän tekoa sekä tietenkin itse ohjelmointia.

Ohjelma kirjoitettiin Eclipse-editorilla, joka on The Eclipse Foundationin julkaisema ilmainen vapaa lähdekoodiprojekti. Ohjelman kotisivut ovat osoitteessa www.eclipse.org.

# LÄHDELUETTELO

## Sähköiset lähteet

1. Java 2 Platform Standard Edition 5.0 API Specification. [www-sivu]. [viitattu 15.5.2006] Saatavissa: http://java.sun.com/j2se/1.5.0/docs/api/
2. Symbian OS. [www-sivu]. [viitattu 15.5.2006] Saatavissa : http://www.symbian.com/
3. Tutkintotyöohje.[sähköinen dokumentti]. [viitattu 15.5.2006] Saatavissa: https://intra.tpu.fi/sivut/tm/data/index.htm
4. S60 Platform SDK for Symbian OS, for C++.[sähköinen dokumentti]. [viitattu 15.5.2006] Saatavissa: http://www.forum.nokia.com/main/1,,034-4,00.html
5. Forum Nokia. [www-sivu]. [viitattu 15.5.2006] Saatavissa: http://www.forum.nokia.com/main.html
6. Wikipedia. [www-sivu]. [viitattu 15.5.2006] Saatavissa: www.wikipedia.org
7. Värimallit ja värimäärät. [www-sivu]. [viitattu 15.5.2006] Saatavissa: http://www.internetix.ofw.fi/atk-tuki/opinnot/skannaus/osa2.htm
8. Java 1 ja 2 kurssin luentokalvot 2003, Tony Torp. [Tamk:n sisäinen dokumentti]
9. Symbiani OS-ohjelmointi kurssin luentokalvot 2004, Tony Torp. [Tamk:n sisäinen dokumentti]

LIITE 1 LUOKKAKAAVIO