Piirrustusohjelma

Tuomas Salminen,

597623,

Tietotekniikka,

2018,

24.4.2019

YLEISKUVAUS

Olen tehnyt piirrustusohjelman, jossa voi piirtää viivoja, neliöitä, ympyröitä ja soikioita eri väreillä ja paksuuksilla. Lisäksi ohjelmassa on undo- ja redo-mekanismi, sekä tallennus ja tiedoston avaus. Ohjelmassa en saanut toimimaan uuden tyhjän piirustuksen avaamista sekä tiedostosta avatun piirustuksen luomista uuteen ikkunaan. Nyt se korvaa olemassa olevan piirustuksen. Työ on toteutettu keskivaikealla asteella, uupumaan jäi äsken mainitut ominaisuudet. Vaativimmista toiminnoista on toteutettu lähes rajaton undo- ja redo mekanismi.

KÄYTTÖOHJE

Ohjelmassa piirtäminen tapahtuu hiirellä painamalla ja vetämällä, kuten tavallisessa piirrustusohjelmassa. Vasemmalla olevasta valikoista voi valita värin, kuvion muodon sekä viivan paksuuden. Ylhäällä valikosta voi tallentaa ja avata tiedostosta sekä käyttää undo ja redo-mekanismia. Vaihtoehtoisesti näppäimistöstä voi painaa z-nappia undo-toimintoon ja r-nappia redo-toimintoon. C-napista piirrustus tyhjentyy. Tyhjennystä ei voi perua undo-toiminnolla. Piirustuksen alhaalta löytyy käyttöohje.

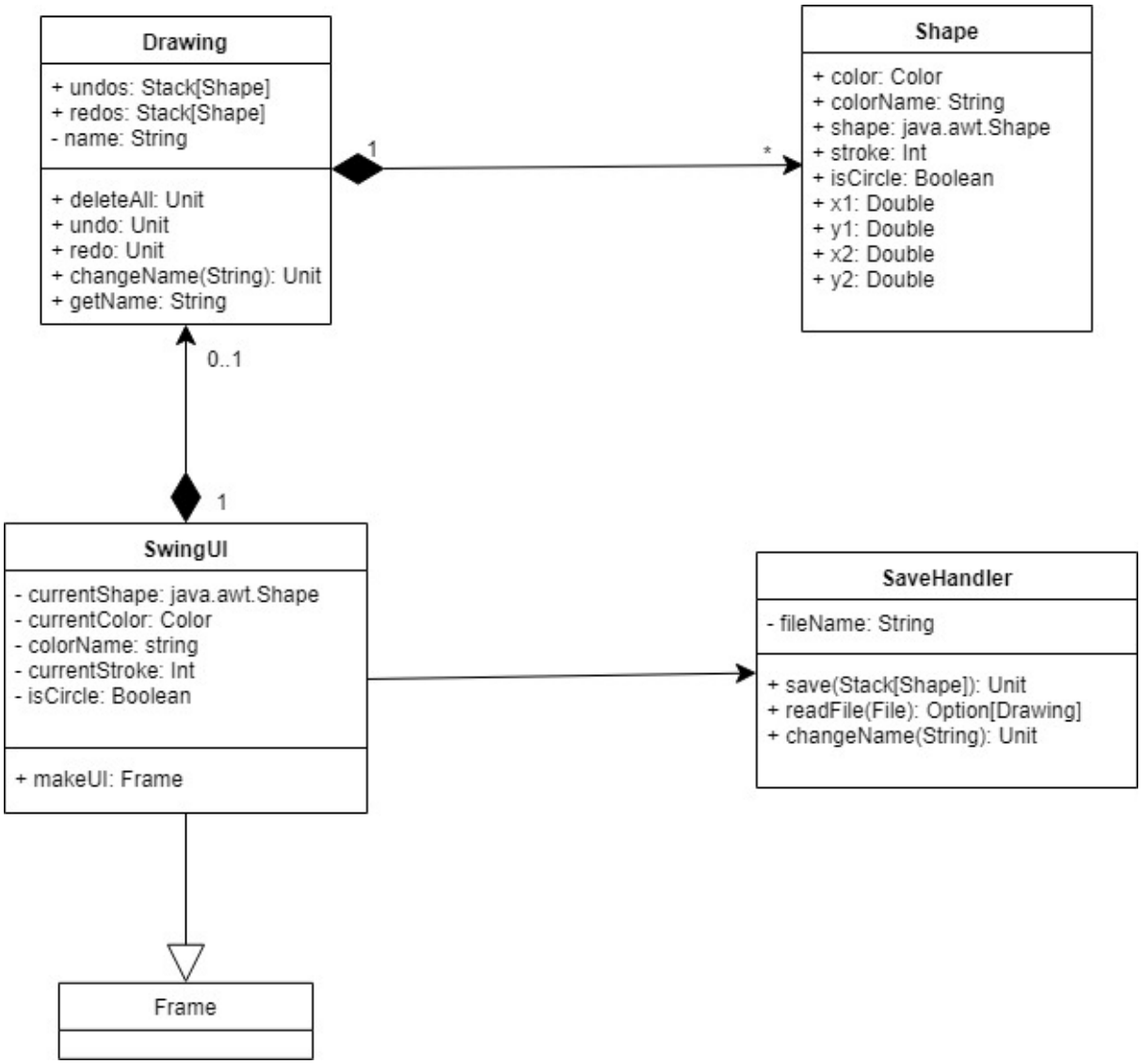
OHJELMAN RAKENNE

Ohjelmassa piirrustusta kuvaa luokka Drawing. Se pitää sisällään undo- ja redo-stackeissä Shape-luokan olioita, joista SwingUI-luokan olio pystyy luomaan grafiikat. Drawing luokassa itsessään on vähän toimintoja. Siinä undo-metodi poistaa päällimmäisen Shape-olion undo-pinosta ja lisää sen redo-pinoon. Redo-metodi toimii taas toisin päin. Shape-luokka pitää sisällään tietoa yksittäisistä kuvioista niiden piirtämistä sekä tallentamista varten.

SwingUI-luokka kuvaa graafista käyttöliittymää, jonka ProgramRunner-olio käynnistää. Yhtä hyvin voisin ajaa ohjelmaa muuttamalla SwingUI:n olioksi, mutta erotellulla yritin saada toimimaan uuden ikkunan avaamista piirrosta avatessa tiedostosta. SwingUIssa on muuttujissa tietoa valituista asetuksista, joiden mukaan kuvio piirretään. SwingUI:ssa

tapahtuu itse piirtäminen sen sisäisessä DrawPanel luokassa. Siinä paintComponent metodi piirtää grafiikan Graphics2D olion avulla. Kuviot piirretään java.awt.Shape aliluokkien oliolla, joita ovat mm. viiva, neliö ja soikio.

DrawPanelin ShapeTo-metodi luo kuvion aina hiirtä vetäessä ja päästäessä siitä irti. Metodi toimii eri tavoin halutusta kuvioista riippuen. Tallennuksia ja tiedoston avaamista hoitaa SaveHandler luokka. Siinä save-metodi tallentaa kuvan tiedostoon ja readFile luo tiedostosta Drawing-olion. Tein myös case class CorruptedFileExceptionin joillekkin tiedostoja luettaessa tapahtuville virheille.



Kaaviosta on jätetty jotain tietoja pois, erityisesti SwingUI-luokasta.

ALGORITMIT

Ohjelmassa ei monimutkaisia algoritmeja ole. Kuvioita luodessa piti ShapeTo-metodissa katsoa että neliöt, ympyrät ja soikiot piirtyvät oikein riippuen mihin suuntaan hiirtä vetää, koska ne piirtyivät vasemman yläkoordinaattien mukaan. Siinä vertasin hiiren koordinaattia alkukoordinaatteihin, ja sen mukaan annoin arvot kuvion yläkoordinaatille ja pituudelle sekä leveydelle.

TIEDOSTORAKENTEET

Ohjelmassa on Drawing-oliolla tallennettuna Shape-olioita stackeihin. Ne ovat joko undo-stackissa tai redo-stackissa. Aina kun ohjelmassa piirretään uusi kuvio, luodaan uusi Shape-olio joka lisätään undo-stackiin. Jos taas perutaan muutos undo-toiminnolla, poistuu Shape-olio undo-stackista ja siirtyy redo-stackiin. Redo-toimintoa käytettäessä siirtyy olio takaisin undo-stackiin. Jos piirretään uusi kuvio, tyhjentyy redo-stack.

Valitsin stack kokoelmatyypin joka on mutable, koska se toimii hyvin kun tarvitaan pääasiassa vain kokelman päälimmäistä elementtiä, kuten undo- ja redo-toiminnossa. Toisaalta ohjelma joutuu käymään undo-pinon kokonaan läpi ohjelman toisessa kohtaa, koska ohjelmassani täytyy piirtää kaikki vanhat kuviot uudestaan kun luodaan uusi kuvio, eli toiminnallisuus ei ole ideaalinen.

Piirrustusta tallennettaessa muutetaan kuvioiden ominaisuuksia tekstipätkiksi, joita voidaan tallentaa tekstitiedostoon. Tällöin käytän bufferia tekstipätkien tallentamiseen.

Käsittelen tiedostoa lukiessa myös mahdollisia virheitä ja loin oman virheluokan joka luodaan jos tiedostossa ei ole oikean mallista tekstiä.

Mitään monimutkaista tietorakennetta ei tarvinnut ohjelmaan rakentaa, joten melkein mikä tahansa kokelmatyyppi olisi käynyt haluttujen toimintojen aikaansaamiseksi.

TIEDOSTOT JA VERKOSSA OLEVA TIETO

Piirrustukset tallennetaan tekstitiedostoon, jossa joka rivillä on yhden kuvion tiedot. Yhden kuvion tiedot ovat muodossa “L,BLA,3,352.0,389.0,565.0,225.0”. Pilkulla erotetaan eri arvot joita tarvitaan kuvion uudelleenluomiseen.

Tiedot ovat järjestyksessä: kuviotyyppi (viiva, neliö jne), väri, paksuus, alkukoordinaatit ja loppukoordinaatit. Jos kuvio on muu kuin viiva, niin loppukoordinaattien sijaan kuvataan leveys ja pituus.

Kuvioita on “L”, eli viiva, “R” eli nelikulmio, “C” eli ympyrä ja “E” eli soikio. Värejä kuvataan värin kolmella ensimmäisellä kirjaimella ja paksuus on numero yhdestä kymmeneen. Loput ovat Double tyyppisiä numeroita.

TESTAUS

Ohjelmassa testaan vain tiedoston tallennusta ja lukua. Graafiset ominaisuudet tuli testattua suoraan näkemällä, että mikä toimii ja mikä ei piirtäessä. Testauksessa luon pari Shape-oliota ja lisään ne listaan, josta luon tallennetun tiedoston. Sitten luen tiedot readFile metodilla ja katson että luodut Shape-oliot vastaavat testin alussa luotuja olioita. Testissä oli se ongelma, että Shape-olioiden shape arvot eivät olleet yhtäsuuria, ilmeisesti sen takia että java.awt.Shape-muotoisia olioita ei voi verrata yhtäsuuruudella. Kokeilin siis testiä olioiden muilla arvoilla. Testit menivät läpi.

OHJELMAN TUNNETUT PUUTTEET JA VIAT

Suurin ongelma on että en onnistunut saamaan uuden tyhjän piirrustuksen luomista tai tiedostosta avaamista uudelle ikkunalle toimimaan. Voisin kuvitella että ominaisuuden saisi toimimaan, jos tekee käyttöliittymään toiminnon jossa voi luoda uusia ikkunoita jollain metodilla. Yritin jotai tämän tyyppistä, mutta en saanut toimimaan.

Toinen puute on se, että jos tyhjentää kuvan ei voi undo-metodilla palauttaa poistettua kuvaa. En kauheasti yrittänytkään tehdä tätä toimintoa, koska aika ei riittänyt.

Muita vikoja on mm se, että piirustuksen vasemmassa yläkulmassa näkyy pieni piste viivaa tai neliötä piirtäessä. En yhtään tiedä mistä se johtuu. Toinen on se että kun piirtää kuvion joka menee reunojen ulkopuolelle, tapahtuu välillä niin että tiedoston tallentaessa ja uudestaan avatessa siirtyy kuvio toiselle puolelle piirrosta. Pitäisi varmaan tehdä jokin ehto että jos kuvio on osittain ulkopuolella, pitää katsoa tarkemmin sen uudelleenrakentamista tiedostoa avatessa.

HEIKOIMMAT JA VAHVIMMAT KOHDAT

Heikoimmat kohdat ovat aikaisemmassa kohdassa mainittujen seikkojen lisäksi yleinen ohjelman toimintojen vajaavaisuus. Olisin mielelläni vielä halunnut lisätä kuvioiden siirto- ja muokkausmahdollisuuden.

Olisin myös halunnut saada ohjelman toimimaan siten, että uutta kuviota piirtäessä ei tarvitsisi piirtää kaikkia vanhoja kuvioita uudestaan. Tämä varmasti hidastaa ohjelman toimintaa.

Muuten onnistuin mielestäni hyvin kuvioiden käsittelyssä, jolloin niistä sai luotua helposti kuvan ja tallennettua tiedostoon, sekä poistettua undo-metodilla. Graafisen käyttöliittymän tekeminen vaati hieman opettelua, mutta minusta se näyttää ihan hyvältä ja toimiikin oikein.

POIKKEAMAT SUUNNITELMASTA, TOTEUTUNUT TYÖJÄRJESTYS JA AIKATAULU

Suunnitelmassani olisin tehnyt kuvioiden siirtelyn ja muokkauksen mahdolliseksi, mutta sitä en ehtinyt toteuttaa. Myös luokka-jako on hieman erilainen. Idea on sama, mutta en luonut itse luokkia erilaisille kuvioille vaan käytin valmiita java.awt.Shape luokan aliluokkia. Itse tekemäni Shape-luokka sitten talletti tiedon kuviosta. Suunnitelmani Piirros-luokka on aika sama kuin Drawing-luokkani, mutta jaoin suunnitelman kuviot-listan lopullisessa versiossa undo- ja redo-listaan.

Tekemäni työjärjestys sujui aikalailla samoin, mutta aikataulu meni pieleen puolessa välissä. En tehnyt projektia lähes kuukauteen toisen checkpointin jälkeen, ja tein projektin sitten loppuun parina viimeisenä päivänä. Tuntimäärät menivät aika lailla samoin aikataulun kanssa.

KOKONAISARVIO LOPPUTULOKSESTA

Olen tyytyväinen lopputulokseen ottaen huomioon että tein suurimman osan projektia parina viimeisenä iltana. Jälkeenpäin ajateltuna olisi pitänyt tehdä enemmän aikaa projektin tekoon. Olin myös lähellä kurssin keskeyttämistä kun deadline alkoi lähestyä, mutta päätin nyt kuitenkin tehdä parhaani, vaikka aika alkoi loppua. En ole tyytyväinen aikaisemmin mainittuihin tiedoston avaamisen ja uuden piirrustuksen luomisen puutteisiin. Nämä olisi saatu toimimaan jos olisin käyttänyt aikani paremmin.

Luokkajako toimii minusta ihan hyvin, mutta pitäisi tarkemmin katsoa että mitkä tiedot kuuluisi millekkin luokille parhaiten. Esimerkiksi SwingUI-luokassa en ehkä haluaisi säilyttää tietoa piirrettävästä kuviosta muuttujissa. Olisin myös halunnut saada uuden kuvion luomisen tapahtumaan ilman että täytyisi piirtää kaikki vanhat kuviot uudestaan. Scala swingin repaint() metodi poisti vanhat kuviot jostain syystä jollen piirtänyt niitä uudestaan. Yritin ratkaista tätä ongelmaa, mutta aikaa ei ollut riittävästi.

Tulevaisuudessa haluaisin eniten saada uuden piirrustuksen luomisen ja tiedostosta avaamisen uuteen ikkunaan toimimaan. Sitten haluaisin tehdä kuvioiden jälkeenpäin muokkauksen ja ryhmittämisen mahdolliseksi. Tämä saattaa tosin vaatia suurempaa muutosta ohjelman toimintoihin. Myös kuvan tyhjentämisen kumoaminen olisi kiva saada ohjelmaan. Enempää en ehkä alkaisi ohjelmaan tekemään tällä hetkellä.

Jos aloittaisin ohjelman uudestaan alusta, ensinnäkin käyttäisin aikani paremmin. Itse ohjelman rakentaisin tukemaan kuvioiden ryhmittämistä ja muokkausta jollain sitä vastaavalla luokalla. Jälkikäteen katsottuna kun tiedän paremmin miten käyttöliittymä rakennetaan ja miten grafiikkaa saa näkymään, olisi voinut keskittyä enemmän itse ohjelman toiminnallisuuksiin.

VIITTEET

Ohjeita käyttöliittymään:

https://plus.cs.hut.fi/studio_2/k2019/k18/osa02/

http://otfried.org/scala/gui.html

https://www.scala-lang.org/api/2.9.1/scala/swing/SimpleSwingApplication.html (swing dokumentaatio)

Ohjeita tiedostoihin:

https://alvinalexander.com/scala/how-to-open-read-text-files-in-scala-cookbook-examples

https://www.scala-lang.org/api/2.9.3/scala/swing/FileChooser.html

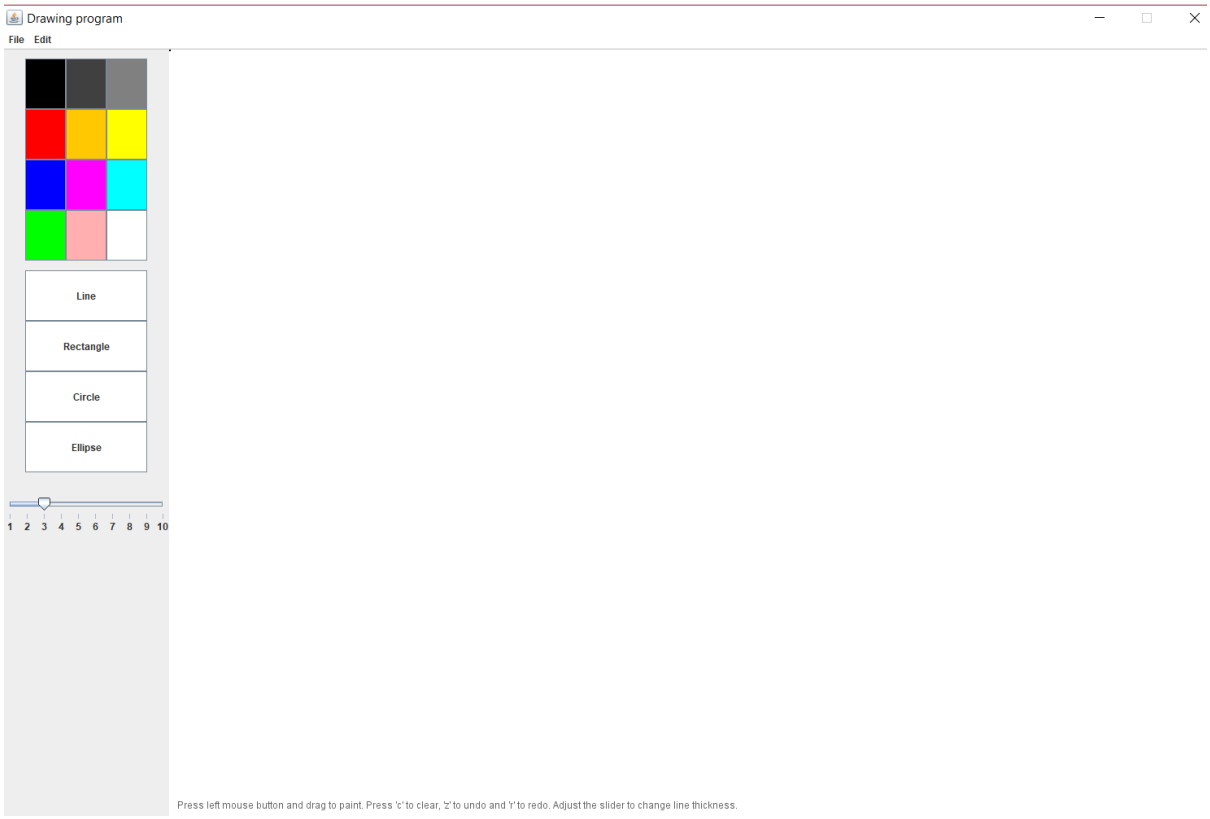
https://stackoverflow.com/questions/23627894/scala-class-running-a-file-chooser

https://plus.cs.hut.fi/studio_2/k2019/k15/osa02/ (exceptions)

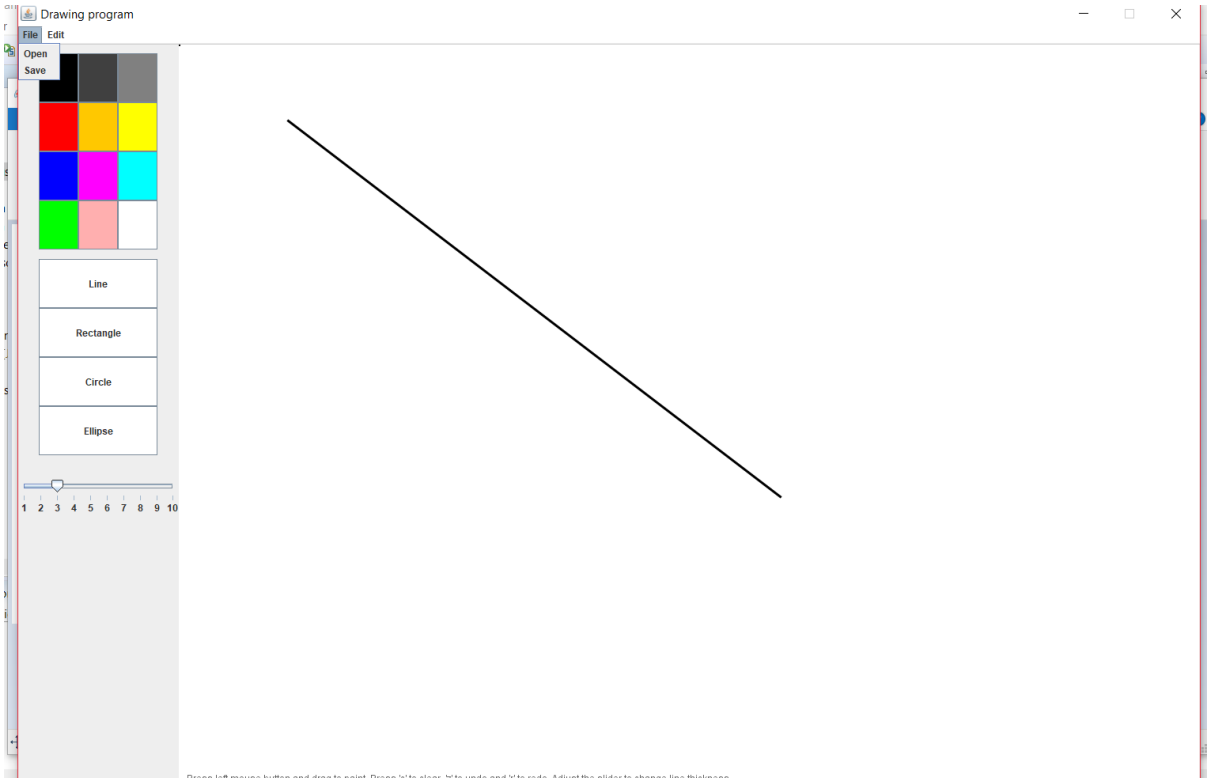
Ohjeita testaukseen: https://plus.cs.hut.fi/studio_2/k2019/k16/osa02/

Idea miten piirtäminen onnistuu: https://github.com/ingoem/scala-swing/blob/master/scala/swing/test/LinePainting.scala

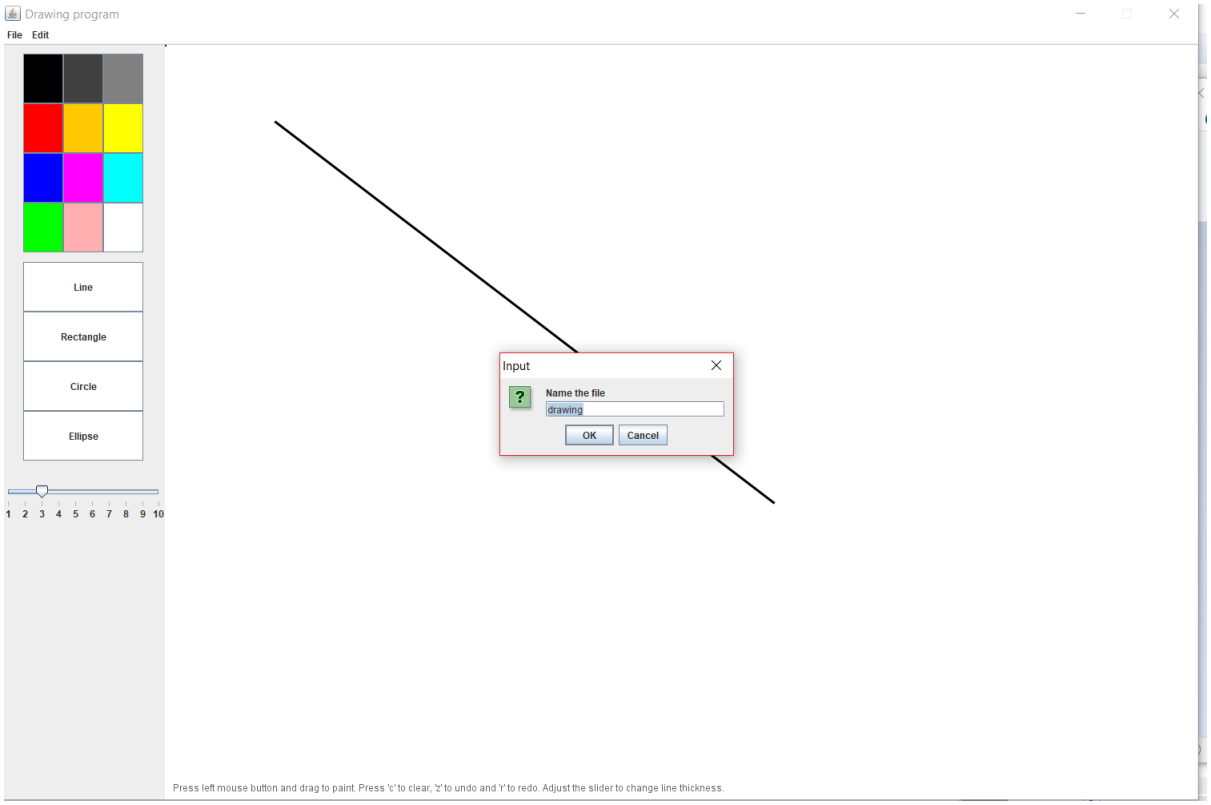
KÄYTTÖESIMERKKI:



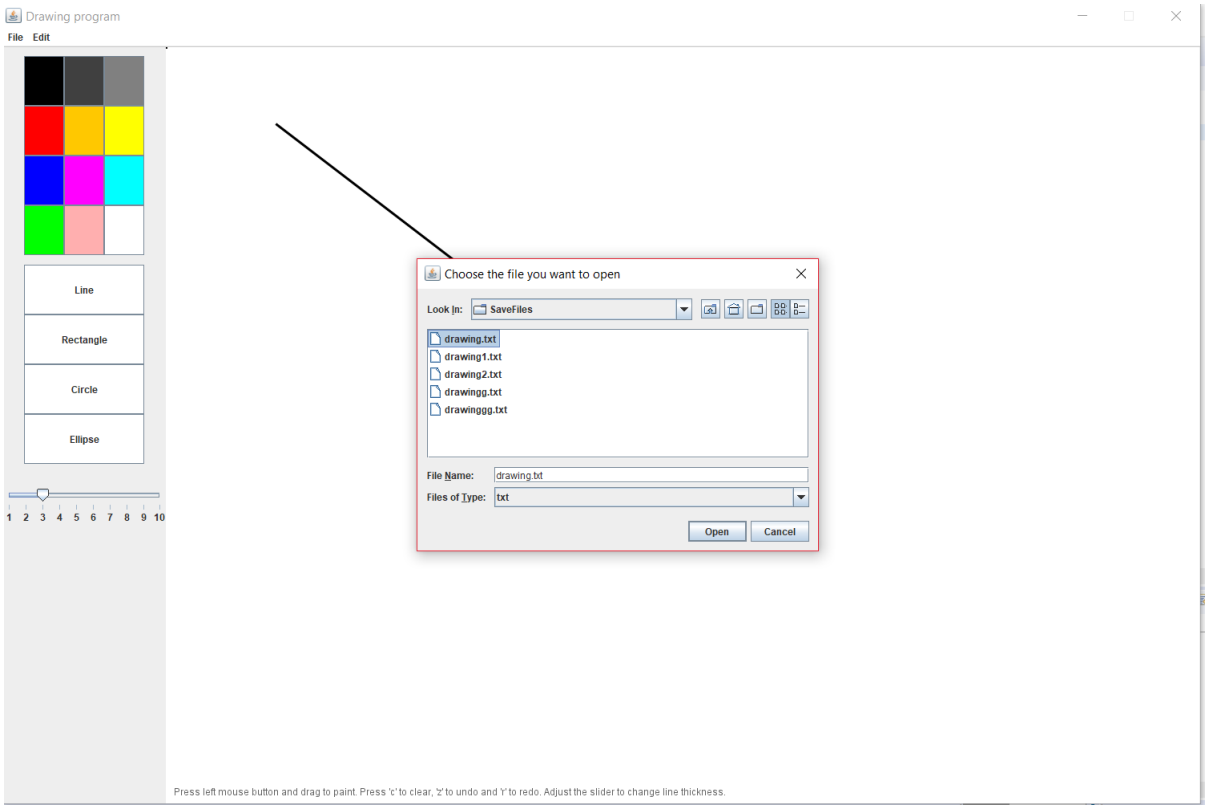
Alussa avautuva näkymä, vasemmalta voi valita värin jne.



Piirrettyä jotain vasemmalta yläkulmasta voi tallettaa kuvion



Avautuvassa ikkunassa voi nimetä tiedoston, se muotoutuu automaattisesti .txt tiedostoon



Open valintaa painaessa avautuu SaveFiles kansio, jonne kuvat ovat tallentuneet. Vain .txt tiedostoja voi avata.