Piirustusohjelman
Tekninen suunnitelma

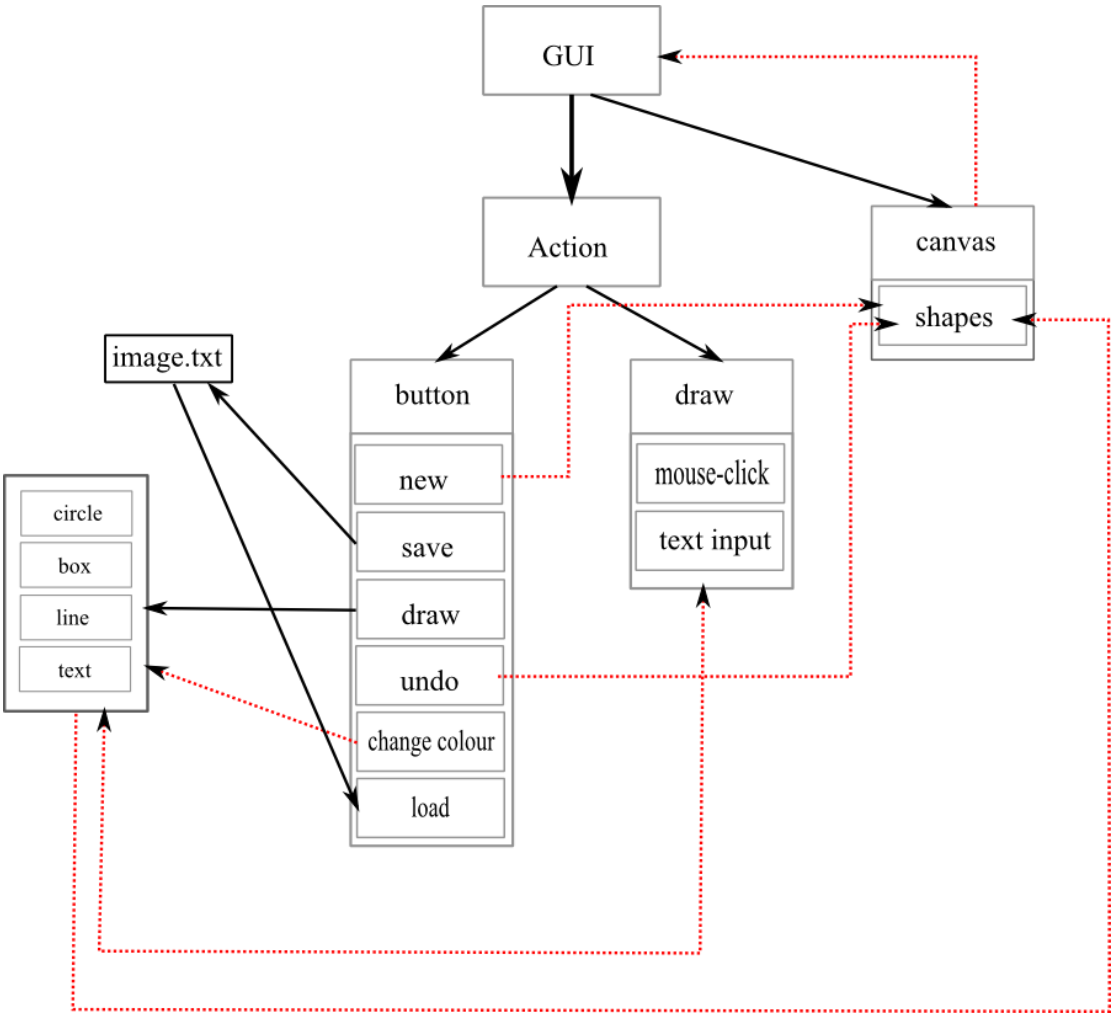
Kaisa Voutilainen

481629

AUT, 2.vuosikurssi

25.2.2016

1. Ohjelman rakennesuunnitelma



Kuva 1 Suunnitelma ohjelman rakenteesta ja vaikutussuhteista

Oheisessa kuvassa on kuvattu ohjelman alustava rakenne ja osien vaikutussuhteita. Tärkeimmät kolme luokkaa tulevat olemaan GUI, Action ja Canvas. GUI sisältää graafisen UI:n elementit ja koodin. Action käsittelee GUI:ssa suoritetut toiminnot, kuten nappuloiden painamiset, hiiren klikkaukset ja liikuttelun.

Toiminnot jakautuvat kahteen osaan: nappuloiden painaminen ja käyttäjän inputit. Jokainen nappuloiden nimeämä toiminto on Button-luokassa oma metodinsa. New-metodi kysyy käyttäjältä, halutaanko varmasti poistaa nykyinen kuva ja luoda uusi ja poistaa sen jälkeen nykyisen kuvan ja luo uuden. Save- metodi tuo esiin ruudun, jossa käytääjä voi määritellä tiedoston nimen ja tallennussijainnin. Sen jälkeen kuva tallennetaan tekstitiedostoon. Load-puolestaan kysyy käyttäjältä, mikä tiedosto avataan ja lataa sen jälkeen ruudulle halutun ohjelmalla aiemmin luodun ja tallennetun tiedoston. Draw sisältää piirtovaihtoehdot, joista valitaan yksi kerrallaan ja sen perusteella ruudulle ilmestyy kuvioita piirrettäessä. Change colour vaikuttaa piirrettävien muotojen

ominaisuuksiin asettamalla mahdollisen uuden värin. Undo-metodi poistaa viimeisimmän kuvion, joka on piirretty. Poistettu objekti poistuu Canvas-luokan shapes-listalta.

Draw-luokan tehtävä on seurata, mitä piirustusalueella tapahtuu. Kun alueella (eli Canvasilla) klikataan jostain, mouse click –metodi seuraa, mihin hiirtä liikutetaan ja päivittää sen perusteella piirrettävän kuvion ominaisuuksia, eli tässä kuvion loppumiskoordinaattia. Tekstiä ”piirrettäessä” taas otetaan käyttäjän antama tekstisyöte näppäimistöltä ja päivitetään sitä ruudulle.

Canvas-luokka sisältää pääasiassa listan, johon listataan piirustusalustalle piirretyt kuviot järjestyksessä. Tätä listaa voidaan käyttää hyödyksi, kun halutaan peruuttaa edellinen toiminto. Luokan tärkeimmät metodit ovat kuvion lisääminen ja poistaminen listalta.

Kuvassa olevat punaiset katkoviivat ilmaisevat tärkeimmät vaikutukset toisiin luokkiin tai objekteihin. Draw-luokan havainnot ja värin vaihto vaikuttavat piirrettävien kuvioiden ominaisuuksiin, piirrettävät kuviot vaikuttavat piirtoalustaan ja piirtoalustan sisällön muuttuminen vaikuttaa GUI:hin. Myös uuden piirustuksen valitseminen tai undo vaikuttavat siihen, mitä piirtoalustalla on.

2. Käyttötapauskuvaus

Käyttötapaus: ajatuskartan tekeminen. Tapauksessa käyttäjä käynnistää piirustusohjelman ja haluaa piirtää yksinkertaisen ajatuskartan, jossa on ympyröitä, soikioita ja laatikoita, joiden sisällä on tekstiä ja joita yhdistetään viivoilla. Koska käyttöliittymä on graafinen, on käyttäjän helppo saavuttaa päämääränsä.

Kun ohjelma on käynnissä, klikkaa käyttäjä hiirellä nappulaa, jossa lukee esimerkiksi ympyrä. Tällöin button-luokasta osataan aktivoida ympyrän piirtäminen. Seuraavaksi käyttäjä klikkaa piirustusalueelta ja pitää hiiren painiketta pohjassa kunnes ympyrästä tulee sopivankokoinen. Tätä toimintoa seurataan draw-luokassa ja sieltä päivitetään ruudulla näkyvää kuviota. Seuraavaksi käyttäjä valitsee toisen värin seuraavaa kuviota varten klikkaamalla värivalikosta haluamansa värin. Värinvaihdon jälkeen käyttäjä klikkaa neliö-nappulaa ja alkaa piirtää neliötä kulmasta kulmaan. Värin vaihto ja muotojen piirtäminen toistuu niin monta kertaa kuin käyttäjä haluaa eri muotoja ja värejä käyttää.

Seuraavaksi käyttäjä haluaa lisätä muotoihin tekstiä. Käyttäjä klikkaa teksti-nappulasta ja heti sen jälkeen piirustusalueelta siitä kohtaa, mihin haluaa tekstin. Koska ollaan

painettu nappulaa ”text”, tietää ohjelma asettaa tekstikentän seuraavan klikkauksen kohdalle. Kun ruudulle ilmestyy kursori, käyttäjä kirjoittaa haluamansa tekstin, joka tallennetaan objektina canvas-luokan shapes-listaan.

Tietysti käyttäjä haluaa myös yhdistellä muotoja toisiinsa viivoilla. Tämän tehdäkseen käyttäjä valitsee viivatyökalun ja alkaa piirtää klikkaamalla viivan aloituspisteestä ja raahaamalla viivan loppupisteeseensä. Viivan piirtäminen toimii samalla periaatteella kuin muotojen tai tekstin piirtäminen. Button-luokan draw-metodi luo halutun objektin ja se saa Draw-luokan mouse click –metodin seuraamat arvot. Tämän jälkeen luotu objekti lisätään Canvas-luokan shapes-listalle.

Lopuksi käyttäjä haluaa tallentaa työnsä. Tämä tapahtuu klikkaamalla tallennusnappulaa. Tällöin ilmestyy uusi ruutu, jossa käyttäjä voi valita, mihin tiedosto tallennetaan ja millä nimellä. Ohjelma muuntaa ruudulla näkyvän kuvan tekstitiedostomuotoon ja tallentaa sen .txt-tiedostoksi. Kun tallennus on suoritettu loppuun ok-nappulaa painamalla, voi käyttäjä joko painaa uusi-nappulaa, joka luo uuden piirustuspohjan tai sulkea ohjelman painamalla ruudun sulkemisnappulaa oikeasta ylänurkasta.

3. Algoritmit

Piirustusohjelma ei sisällä erityisiä algoritmeja, kuten tekoälyt tai ratkaisuohjelmat, mutta ohjelma vaatii jonkinlaisen tavan suorittaa esimerkiksi peruutustoiminto (undo), kuvan muuttaminen tekstimuotoon ja takaisin.

Peruutustoiminnon aion toteuttaa oletettavasti tallentamalla kaikki objektit järjestyksessä listaan, josta niitä voidaan sitten poistaa viimeisimmästä ensimmäiseen. Tämä mahdollistaa rajattoman undon taaksepäin, mutta ei eteenpäin (tyyliin crl+z toimii ja crl+y ei toimi).

Kuvan muuntamisen tekstimuotoon ja takaisin kuvaksi aion toteuttaa listaamalla tekstitiedostoon jokaisen pikselin värin heksamuodossa. Kuvan saa rakennettua takaisin helposti tekemällä prosessin toiseen suuntaan. Tämä tapa toteuttaa tallennus ja lataaminen vaikutti alustavasti ainakin yksinkertaiselta idealta, jota voi kehittää pidemmälle tai parantaa jos se toiminta ei ole mieluisa.

4. Tietorakenteet

Oleellisin tietorakenne projektissani on lista, joka sisältää kaikki piirustuksessa olevat kuviot järjestyksessä. Lista on mielestäni hyvä valinta, koska kuvioiden määrää ei voida tietää etukäteen, joten rakenteen täytyy olla joustava. Siihen pitää pystyä lisäämään objekteja niin paljon kuin haluaa ja siitä pitää myös pystyä poistamaan viimeisin objekti jos siihen on tarvetta.

5. Aikataulu

Työjärjestys:

1. Alustava GUI, jotta ominaisuuksia päästään testaamaan jo koodauksen ohessa. Aikaa kuluu noin 20h, tähän on laskettu mukaan PyQt5:een tutustuminen.
2. Canvas-luokan toteutus, jotta tulevat ominaisuudet pystytään päivittämään GUI:hin. Ajankäyttö: 4h.
3. Piirto-ominaisuus: muodon ja värin valitseminen, käyttäjän syötteen lukeminen ja käsittely. Ajankäyttö: 40h. Sisältää paljon manuaalista testausta.
4. Canvasiin vaikuttavat loput nappulat: new ja undo. Aikaa kuluu: 4h.
5. Tallennus ja tiedoston lataaminen, ajan käyttö: 20h. Voi mennä enemmänkin aikaa jos suunniteltu systeemi ei toimikaan ja pitää keksiä uusi.
6. GUI:n viimeistely ja hienosäätö. Aikaa kuluu: 8h.
7. Testausta toimivuudesta tähän asti, koska projektin pitäisi olla suunnitelman mukainen. Ajan käyttö: 10h.
8. Vaativat työn ominaisuuksien lisääminen jos ehtii.

6. Yksikkötestaussuunnitelma

Aion toteuttaa manuaalista testausta koko projektin ohjelmoinnin läpi. Uusia ominaisuuksia on hyvä testata mahdollisimman pian, jotta tietää, miten ne oikeasti toimivat. Koska ohjelman on graafinen ja seuraa käyttäjän syötteitä, on sille vaikea keksiä yksikkötestejä.

Ohjelman piirto-ominaisuus on piirustusohjelman tärkein osio, joten sen testaamiseen pitää panostaa. Testaus toteutetaan manuaalisesti piirtämällä ja katsomalla, tekeekö ohjelma sen, mitä sen odotetaan tekevän. Myös virhetilanteita testataan klikkailemalla kohdista, joista ei kuuluisi klikata, esim. piirrettäessä piirtoalueen ulkopuolelta.

Kuvan tallennusta ja lataamistakin testataan huolella piirtämällä kuva ja tallentamalla se ja lataamalla uudestaan. Jos tallennetun kuvan jättää piirtoalustalle ja lataa sitten saman kuvan tiedostosta, pystytään virheet huomaamaan helposti kun kuva vaihtuu.

Undo on metodi, jota voi testata yksikkötestauksella. Sitä on hyvä testata ääritapauksissa, kuten siinä jos yritetään käyttää peruutustoimintoa tyhjään piirustukseen tai juuri avattuun kuvaan. Tällaisissa tapauksissa metodin tulisi palauttaa ilmoitus, että toimintoa ei voida suorittaa eikä esimerkiksi yrittää poistaa olematonta objektia. Yksikkötestauksen voi tehdä Pythonin omilla testausvälineellä, esim. käyttämällä assertEqual()-metodilla.

7. Kirjallisuusviitteet ja linkit

Olen käyttänyt:

http://takluyver.github.io/posts/so-you-want-to-write-a-desktop-app-in-python.html

> Jonkinlainen tutoriaali GUI toolkitin valintaa varten. Tämän perusteella päädyin valitsemaan PyQt:n.

https://grader.cs.hut.fi/static/y2/_static/html/python3-installation/Python3asennusohjeet.html

> Ohjeet PyQt5:n asentamista varten. Noudatin tätä ohjetta asentaessani ohjelmiston valmiiksi.

Aion käyttää:

http://zetcode.com/gui/pyqt5/

> Kattava tutoriaali PyQt5:n perusominaisuuksista. Sisältää paljon piirustusohjelman kannalta oleellisia vinkkejä ja koodiesimerkkejä. Aion hyödyntää tätä lähdettä kun aloitan ohjelmoinnin.

http://pyqt.sourceforge.net/Docs/PyQt5/index.html

> PyQt:n luokat. Näistä on varmasti hyötyä, kun perehdyn lähemmin siihen, miten toteutan eri luokat ja ominaisuudet.

Näiden lisäksi tulen varmasti etsimään apua ongelmiin google.fi –hakupalvelun avulla.