## 1. Henkilötiedot

Pienoisrautatie

Niko Hakanen 222914
TFM -10
6.5.2016

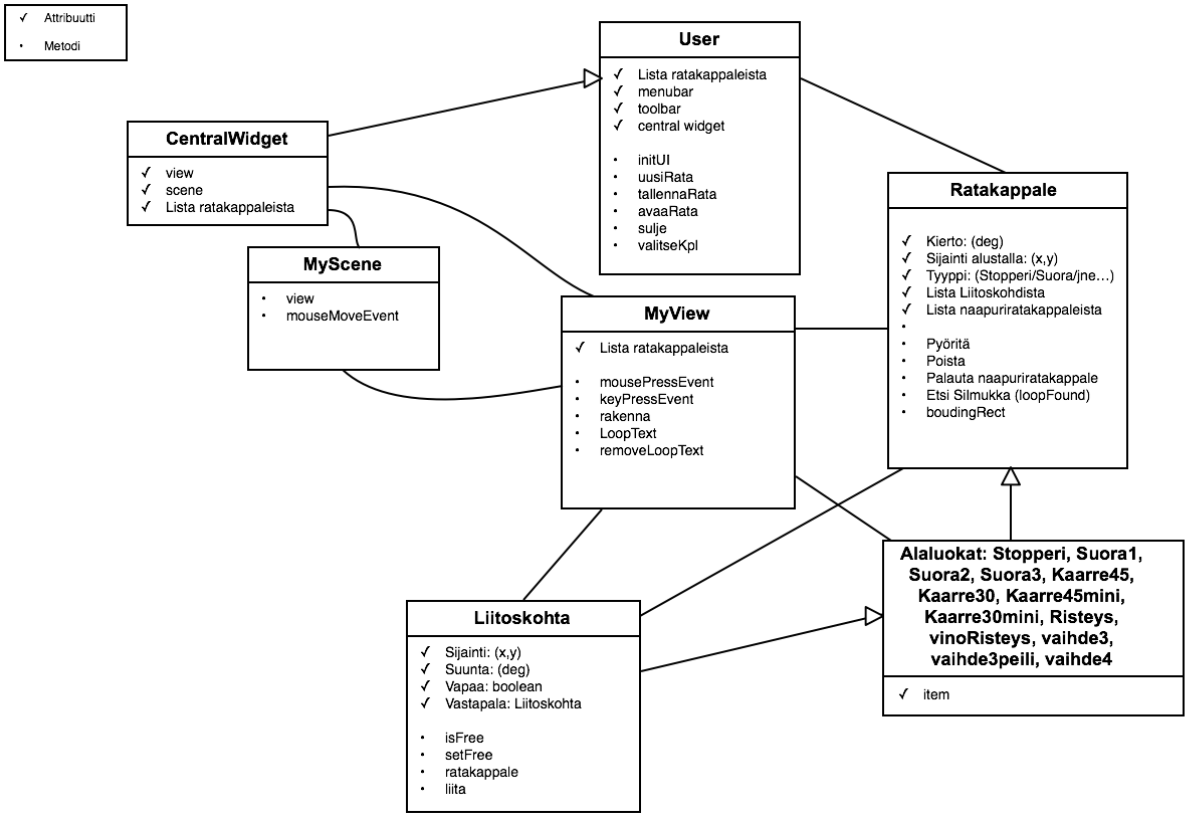
## 2. Yleiskuvaus

Luotiin ohjelma, jossa voi graafisella käyttöliittymällä suunnitella pienoisrautateitä. Pienoisrautatiet koostuvat standardikokoisista paloista, joita voidaan liittää toisiinsa. Paloja on montaa eri tyyppiä mm. eri mittaisia suoria kappaleita, eri kaarevuussäteellä olevia 30 tai 45 asteen kaarteita, vaihteita, stoppereita ja risteyksiä. Ohjelman huomaa jos palan lisääminen muodostaa silmukan. Ohjelmassa on mahdollisuus tallentaa rata ja myöhemmin ladata ja jatkaa suunnittelua. Olen toteuttanut ohjelman **vaativalla** vaikeusasteella.

## 3. Käyttöohje

Ohjelma käynnistyy User-luokasta, jolloin avautuu ikkuna Pienoisrautatie. Ikkunan vasemmassa laidassa on ratakappalevalikko, josta käyttäjä voi klikkaamalla valita haluamansa erilaisista ratakappaleista. Valitun kappaleen voi asettaa valkoiselle alustalle haluamaansa kohtaan klikkaamalla. Valittuja kappaleita voi pyörittää haluamaansa kulmaan näppäimistön oikea ja vasen nuolinäppäimillä. Del-näppäimellä valitun kappaleen voi poistaa. Kappaleita voi liittää toisiinsa viemällä valitun kappaleen liitoskohdan (vaaleansininen alue) lähelle alustalla olevan kappaleen liitoskohtaa, jolloin valittu kappale asettuu automaattisesti sopivaan kohtaan ja suuntaan. Jos kappaletta ei voi asettaa kyseiselle paikalle, se muuttuu punaiseksi. Jos kappaleet muodostavat silmukan, ikkunaan ilmestyy hetkeksi teksti ”Muodostui silmukka”. Tiedosto-valikossa käyttäjällä on vaihtoehdot ”Uusi”, ”Avaa”, ”Tallenna” ja ”Sulje”. ”Uusi” poistaa nykyisen radan täysin ja jättää käyttäjälle tyhjän alustan. ”Avaa” poistaa nykyisen radan ja rakentaa käyttäjän valitseman aiemmin tallennetun radan. ”Tallenna” tallentaa avoinna olevan radan. ”Sulje” sulkee ohjelman.

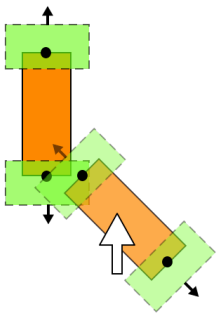
## 4. Ohjelman rakenne




PyQt4:n valmiit luokat määrittivät luokkarakenteen melko vahvasti. Pääikkuna perii valmiin luokan QMainWindow, joka on valmis pohja ikkunaohjelmille. Pääikkunaan taas voidaan asettaa varsinainen ohjelma/widgetti joka on tässä tapauksessa CentralWidget. CentralWidget on vain puolestaan pohja varsinaisille kappeleiden käsittely- ja piirtoluokille MyScene ja MyView jotka perivät PyQt4 valmiit luokat QGraphicsScene ja QGraphicsView. Ratakappaleen alaluokat ovat QGraphicsItem:itä, joita voidaan tallentaa MySceneen. Liitoskohta on radankappaleiden alaluokka, sillä yhteen radankappaleeseen on liittynyt aina yksi tai useampi liitoskohta.

Tärkeimpiä metodeita on Ratakappale-luokan loopFound, joka tarkastaa muodostuuko kappaleista silmukkaa. Lisäksi tärkeä on MyView-luokan mousePressEvent, joka käsittelee, mitä tehdä käyttäjän klikatessa hiirtä ja MyScene-luokan mouseMoveEvent, joka toteuttaa asioita hiiren liikkuessa. Myös Liitoskohta-luokan liita-metodi on olennainen, sillä se liittää radankappaleet toisiinsa.

## 5. Algoritmit



Oikealla olevassa kuvassa on havainnollistettu, miten ohjelma liittää radankappaleita yhteen. Todellisuudessa ikkunassa näkyy ainoastaan radankappaleet (oranssilla), liitoskohdat (vihreällä) ja hiiri. Ohjelma tunnistaa, että liitettävän kappaleen liitoskohta on alustalla olevan ratakappaleen vapaan liitoskohdan päällä. Liitoskohdilla on suunnat (mustat nuolet) ja ohjelma pyörittää

liitettävää kappaletta siten, että liitoskohtien suunnat ovat vastakkaiset. Tämän jälkeen se asettaa liitoskohdat vastakkain ja käyttäjän klikkauksella asettaa liitettävän kappaleen alustalle.

Merkittävin ongelma ohjelman toteutuksessa on, miten ratakappaleet liittyvät toisiinsa. Vaihtoehtoinen rakennustapa olisi ollut, että rata rakentuu valitsemalla aina valikosta seuraava kappale edellisen liitoskohdan jatkoksi, jolloin ohjelma asettaa automaattisesti palan kohdalleen. Kappaleita ei saisi asettaa vapaasti alustalle paitsi ensimmäisen palan. Päädyin kuitenkin käyttämään ”drag and drop” tyyppistä menetelmää, koska se on intuitiivisempi ja muistuttaa enemmän pienoisrautatien rakentamista todellisuudessa.

Silmukoiden muodostuminen tarkistetaan aina liitettäessä uusi radankappale. Radankappaleen naapurikappaleet palautetaan rekursiivisesti perintää käyttäen, ja jos jossain vaiheessa palautettu kappale on kappale itse, niin silmukka on muodostunut.

## 6. Tietorakenteet

Kaikki tieto on ohjelmassa valmiina luokissa esim. Radankappale-luokilla ohjeet kappaleiden piirtämiseen. Dynaamisia rakenteita ohjelmassa ovat mm. User-luokan lista radankappaleista ja Ratakappale-luokan listat liitoskohdista ja naapurikappaleista. Kun rata tallennetaan, ohjelma kirjoittaa .txt -pohjaisen tietorakenteen ja tallentaa sen samaan kansioon ohjelman kanssa. Tekstitiedostoon tallennetaan radankappaleiden tyyppi, sijainti ja kierto (Tyyppi x y kierto) esim. 0 401.456049 370.566728 60. Kun kappaleita on useita, tekstitiedosto voisi näyttää tältä 3 45 600 0 7 180.505999 604.405125 45 0 500 400 90. Tässä luvut on erotettu välilyönnillä mutta todellisuudessa niiden välillä on rivinvaihto. Kun tallennettu rata avataan, ohjelma asettaa radankappaleet alustalle samassa järjestyksessä, kuin ne on alun perin tallennettu, tekstitiedoston tietojen perusteella. Tiedostoa avattaessa ohjelma suorittaa kappaleiden yhteen liittämiset.

## 7. Tiedostot

Kuten edellisessä kappaleessa on kerrottu, ohjelma avaa ja tallentaa .txt –pohjaisia tiedostoja, joissa yksi radankappale on esitetty neljällä luvulla (tyyppi: int, x: float, y: float, kierto: float) jotka on erotettu rivinvaihdolla. Tyyppi on luku välillä 0-12, joka kertoo ohjelmalle, minkä tyyppinen radankappale on kolmestatoista vaihtoehdosta. Tiedostoja on erittäin helppo luoda asettamalla ohjelmalla kappaleita radalle ja tallentamalla radan.

## 8. Testaus

Suurin osa testauksesta oli ohjelman graafisen käyttöliittymän kokeilua käytännössä. Kun ratoja voitiin luoda ja tallentaa, pystyttiin kirjoittamaan varsinaisia testejä, jotka testaavat toimivatko aiemmin rakennetut radat, kun ohjelmaa muutetaan.

## 9. Ohjelman tunnetut puutteet ja viat

Joskus ratakappaleen voi liittää toiseen, vaikka se näkyy punaisena. Tällöin kappale jää punaiseksi alustalle. Jos kuitenkin hiirellä korjaa hieman kappaleen paikkaa, niin punainen väri poistuu. Tämän voisi korjata säätämällä liitoskohdan kokoa ja paikkaa. Joskus

automaattiset liittämisehdotukset ovat hieman hassuja, eli kappaletta ei voi asettaa kohtaan, mihin ohjelma yrittää sen kääntää. Tämäkin yleensä korjaantuu asettamalla kappaleen ”oikeassa suunnassa” eli pyörittämällä kappaleen haluttuun suuntaan nuolinäppäimillä. Lisäksi itse ohjelmakoodin tehokkuutta voisi parantaa huimasti. Siellä on todella paljon toistoa, erityisesti radankappaleiden alaluokissa. Esim. kaikki kaarteet voisi piirtää yhdellä luokalla eri input-parametreilla. Tämä johtuu siitä, että halusin käyttää aikaani mahdollisimman paljon ohjelman toiminnallisuuden lisäämiseen koodin siistiyden sijasta.

**10. 3 parasta ja 3 heikointa kohtaa**

Mielestäni yksi ohjelman parhaita ominaisuuksia on sen käytettävyys eli kappaleita on todella helppo ja nopea napata valikosta ja liittää toisiinsa. Automaattiset liittämisehdotukset toimivat yleensä varsin hyvin. Lisäksi alusta on käytännössä loputon, eli ikkunaa voi venyttää ja kappaleita asettaa kaikkialle, jopa näkyvissä olevan alustan ulkopuolelle. Suunnittelu ei siis voi tyssätä ”seinään”.

Lisäksi eräs ominaisuus syntyi ohjelmaan ikään kuin vahingossa. Jos valikosta klikkailee kappaleita useamman kerran, näitä voi kerätä ”pinon” hiiren mukaan. Tähän olisi toisaalta ollut hyvä lisätä laskuri.

Kaikki ohjelman heikkoudet johtuvat oikeastaan siitä, että en ehtinyt kehittää ohjelmaa enempää. Olisin halunnut ”katalogiin” kuvat kappaleista, kun nyt niistä on vain nimet. Lisäksi kappaleiden muotoa suhteessa toisiinsa ei ole mietitty lainkaan eli esim. voiko risteyksestä tehdä helposti silmukan pelkillä kaarteilla siten, että se päätyy takaisin samaan risteykseen. Kappaleet jäivät sen kokoisiksi, kun ne aluksi luotiin, vaikka niiden kokoa ja muotoa olisi helppo muuttaa. Lisäksi vinoristeyksessä ja vaihteissa sivuavat haarat ovat ohuempia, koska nämä oli huomattavasti nopeampi piirtää kuin laskea kahden päällekkäisen samanleveisten suorien leikkauskohtia.

**11. Poikkeamat suunnitelmasta**

Ainakin luokkarakenne meni heti PyQt4 valmiisiin luokkiin tutustuttaessa heti uusiksi. Mikäli ei halua tehdä kaikkea itse, luokkareferenssikirjasto antaa melko tiukat raamit ohjelman luokille. Kappale valikko jäi kokonaan toteuttamatta, koska koin sen melko turhaksi, koska kappaleita on paljon nopeampi ja helpompi pyörittää ja poistaa näppäimistöllä. Lisäksi edellä mainittu kuvallinen katalogi jäi tekemättä. Ajankäyttöarvio ei osunut kovin oikeaan pääasiassa siitä syystä, että aloitin projektin varsinaisen toteuttamisen paljon myöhemmin muiden koulutöiden vuoksi. Toteutusjärjestys oli jotakuinkin suunnitelman mukainen.

**12. Toteutunut työjärjestys ja aikataulu**

Viikot 9-13: Kierros 5 tehtävät, GitLab testailua, PyQt4 ja muita koulutöitä.
Viikko 14: Kappaleiden alustalle piirtämisen testailua.
Viikko 15: Kappaleiden pyörittäminen ja alustalle lisääminen.
Viikko 16: Kaikki edellinen uusiksi, koska päätin toteuttaa QGraphicsItemien avulla.
Viikko 17: Kappaleita voi valita, siirtää ja pyörittää.
Viikko 18: Silmukoiden tunnistus, ratojen tallennus ja lataus.

## 13. Arvio lopputuloksesta

Mielestäni ohjelmani oli kokonaisuudessaan hyvin onnistunut, varsinkin kun en ole ikinä aiemmin graafisia käyttöliittymiä tehnyt. Ohjelmani tekee kaiken, mitä Pienoisrautatieohjelman dokumenttisivu vaatii eli siinä on erilaisia ratakappaleita, ohjelma tunnistaa silmukat ja ratoja voi tallentaa ja avata. Olen edellä maininnut vikoja ja asioita, joita olisin halunnut vielä toteuttaa. Hyvä puoli ohjelmakoodissa on se, että sitä on erittäin helppo laajentaa. Periaatteessa radankappaleet voivat olla millaisia vain, ohjelma ei välitä niiden muodosta tai liitoskohtien määrästä. Radankappaleita olisi helppo lisätä vaikka heti kymmeniä erilaisia lisää.

## 14. Viitteet


- PyQt 4 perusteet: http://zetcode.com/gui/pyqt4/
- Kurssimateriaali: https://plus.cs.hut.fi/y2/2016/
- The Python Standard Library: https://docs.python.org/3/library/
- CSE-A1111 Ohjelmoinnin peruskurssi Y1 Opetusmoniste syksy 2014
- PyQt Class Reference: http://pyqt.sourceforge.net/Docs/PyQt4/classes.html
- Mark Summerfield - Rapid GUI Programming with Python and Qt. Pearson Education, 2008.


## 15. Liite – Esimerkkiruutukaappaus ohjelmasta