Käyttöliittymien ohjelmointi, Harjoitustyö  
TTOS0300

Asko Ropponen  
Alexander Andreev

Huhtikuu2017

Tieto- ja viestintätekniikka

Tekniikan ja liikenteen ala

Sisällys

[1 Github 2](#_Toc480915386)

[2 Rakennekaavio (UML) 2](#_Toc480915387)

[3 Luokat 3](#_Toc480915388)

[3.1 MainWindow 3](#_Toc480915389)

[3.2 Switcher & ISwitcahble 3](#_Toc480915390)

[3.3 Näkymät 4](#_Toc480915391)

[3.3.1 mainmenu 4](#_Toc480915392)

[3.3.2 highscores 4](#_Toc480915393)

[3.3.3 playfield 4](#_Toc480915394)

[3.4 Card & Deck 5](#_Toc480915395)

[3.5 newhighscore, RealHighScore &HighScoresHandler 5](#_Toc480915396)

[4 Työaikaraportti 6](#_Toc480915397)

[5 Kuvankaappaukset 7](#_Toc480915398)

[6 Keskeisimmät ongelmat työtä tehdessä 9](#_Toc480915399)

[7 Testaus 9](#_Toc480915400)

[8 Itsearviointi 10](#_Toc480915401)

[8.1 Asko 10](#_Toc480915402)

[8.2 Alex 10](#_Toc480915403)

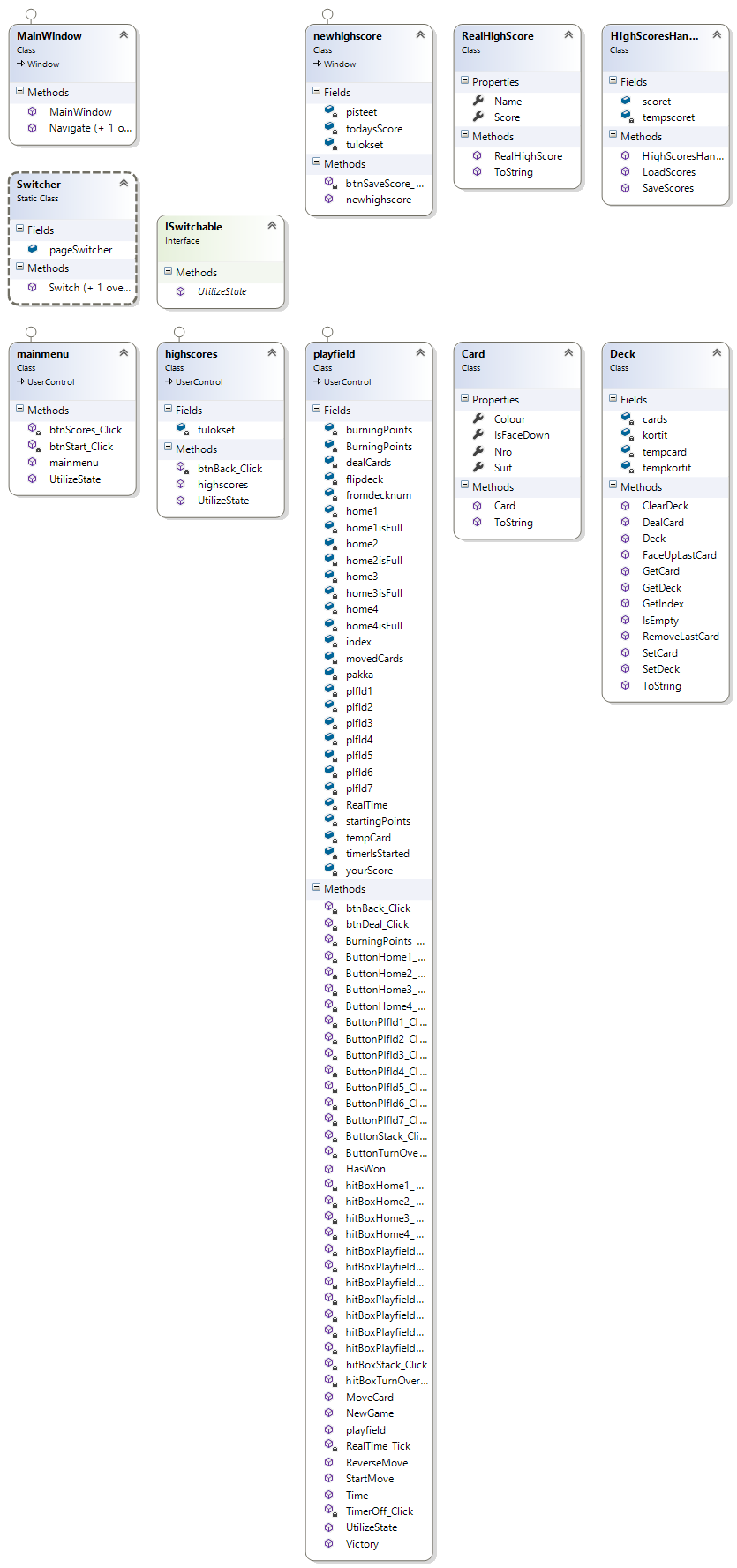
[9 Kurssipalaute 10](#_Toc480915404)

# Github

<https://github.com/Okxa/OLIO-SOLITAIRE> Suunnitteludokumentti on github-wikissä.

# Rakennekaavio (UML)

Verrattuna suunniteltuihin rakennekaavioihin toteutettuja oli enemmän ja niissä on enemmän tavaraa. Card –luokalla on kuitenkin samat maa, numero, väri ja naamapuoli ominaisuudet kuten suunniteltu, mutta highscore –luokka on toteutettu hieman eri tavalla. Pisteiden tallennus ja lataus ovat omassa luokassaan.



# Luokat

## MainWindow

Pääikkuna, tässä tapauksessa sillä ei ole paljonkaan tarkoitusta, sillä on metodi Navigate, jolla voidaan vaihtaa näkymiä.

## Switcher & ISwitcahble

Swicher –luokka ja ISwitchable –rajapinta ovat käytössä näkimien vaihdossa. Kunkin näkymän täytyy implementoida ISwitchable –rajapinta. Kutsumalla Switcher –luokan Switch –metodia, voidaan vaihtaa näkymää.

## Näkymät

Pelissä on monta näkymää. Näkymät ladataan, käyttäen Switcher luokkaa ja ISwitchable –rajapintaa.

### mainmenu

Pelin päävalikko. Tästä päästään katsomaan tuloksia ja pelaaman peliä.

### highscores

Tulokset. Näyttää Top 10 parasta tulosta. Tulokset ladataan tekstitiedostosta käyttäen HighScoresHandler –luokkaa.

### playfield

Itse peli. Luokassa on Deck olioiden ilmentymiä jokaiselle pelikentän pakalle, eli 13 yhteensä. Lisäksi yksi ilmentymä juuri siirrettäville korteille. Luokassa on myös dealcard -kokoelma jonka avulla jaetaan kortit pelikentällä oleviin 7 pakkaan.

Korttien siirto on toteutettu StartMove, MoveCard ja ReverseMove metodeilla. Klikatessa korttia StartMove siirtää Korti(n/t) käyttämällä Deck –olion GetCard tai GetDeck -metodia siirrettyjen korttien olioon (kayttämällä Deck –olion SetCard tai SetDeck –metodia). Klikatessa seuraavaa korttia (tai tyhjää paikkaa) MoveCard yrittää laittaa kortin siihen. Jos se ei onnistu, MoveCard kutsuu ReverseMove –metodia joka palauttaa kortin alkuperäiseen paikkaan.

Pisteenlasku on myös toteutettu tässä luokassa. Toteutus: meillä on maksimi pistemäärä, josta joka sekunti miinustetaan pisteitä. Uusi pistemäärä tallennetaan highscore -muuttujaan.

Pelin voittaminen tapahtuu laittamalla oikean yläkulman neljään pakkaan kuhunkin päällimmäiseksi kuningas –kortti. Peli seuraa tätä neljällä boolean –muuttujalla. Peli tarkistaa pisteenlaskun yhteydessä kutsumalla metodia HasWon, joka tarkistaa nämä muuttujat. Jos peli on voitettu kutsutaan, Victory -metodia. Tämä pysäyttää pisteenlaskun ja kutsuu pisteentallennus metodia.

## Card & Deck

Card on luokka joka kuvaa korttia. Kortilla on ominaisuudet maa, numero, väri ja onko kortti päällispuoli ylöspäin.

Deck -luokka kuvaa pakkaa. Siinä on kokoelma Card –olioita. Lisäksi luokassa on joitain kokoelmia ja listoja, joita käytetään korttien siirrossa. Korttien otossa ja laitossa käytetään GetCard, SetCard ja GetDeck ja SetDeck –metodeja. Näihin voi välittää kokonaisluvun joka kuvaa pakan tyyppiä, jonka mukaan siirtoehdot toimivat. Eli ylärivin pakoissa alin kortti on ässä jne.

Deck –luokassa on myös metodeja, jotta voi tarkistaa onko pakka tyhjä, saada kortin indeksin kokoelmassa, tyhjentää pakan, poistaa viimeisen kortin pakassa ja kääntää pakan viimeisen kortin päällepäin.

## newhighscore, RealHighScore &HighScoresHandler

Kun pelin voittaa, tehdään uusi newhigscore –ikkuna. Tälle ikkunalle välitetään pelaajan pisteet. Ikkuna kysyy pelaajan nimen ja pelaajan painaessa Save Score –painiketta, kutsutaan HighScoresHandler –luokan SaveScores –metodia.

RealHighScore on luokka joka kuvaa tulosta. Tuloksessa on pelaajan nimi ja pistemäärä.

HighScoresHandler –luokka hallitsee tulosten latauksen ja tallennuksen. Luokassa on ObservableCollection, jossa on RealHighScore –olioita. Tähän ladataan tekstitiedostosta tulokset. Samoin kokoelmasta tallennetaan kukin tulos tekstitiedostoon.

# Työaikaraportti

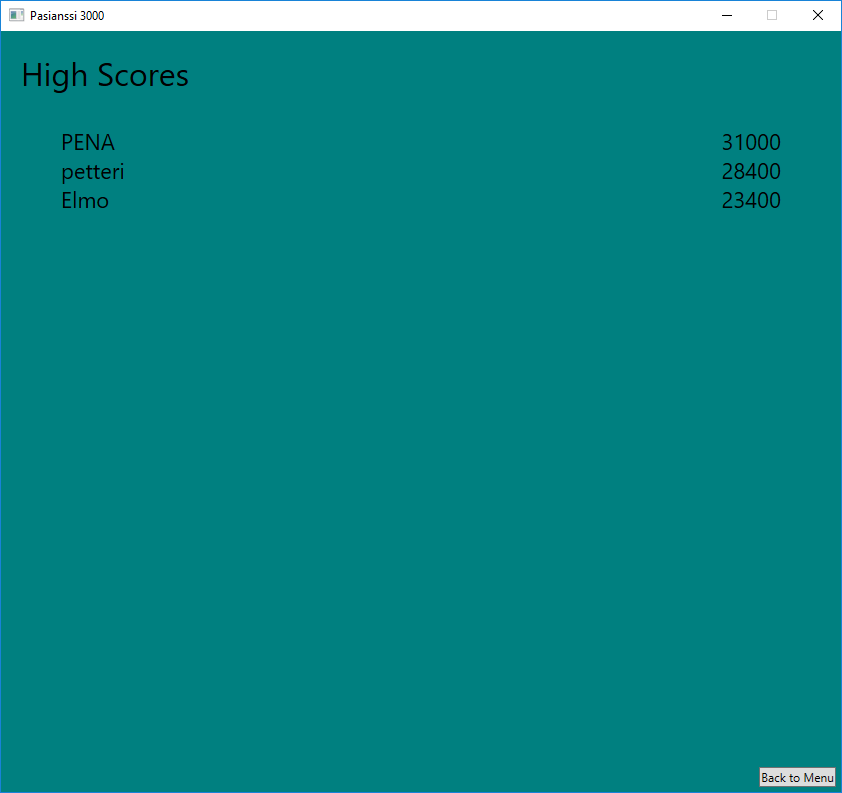
Teimme työtä tunnilleilla yhdessä noin 40h per nokka joista n. 6h suunnittelua. Lisäksi teimme työtä omalla ajalla 20-30h per tekijä.

|  |  |  |
| --- | --- | --- |
| MItä | Asko | Alexander |
| Suunnittelu | 6h | 6h |
| Käyttöliittymän pohjat | 8h | 3h |
| Pelilogiikka | 55h | 25h |
| Pisteytys | 0h | 8h |
| Pisteen tallennus txt -tiedostoon | 4h | 2h |
| Pisteentallennus tietokantaan | 0h | 15h |
| Dokumentaatio | 2h | 1h |
| Grafiikat | 4h | 0h |

# Kuvankaappaukset



Kuva Päävalikko. Päävalikosta pääsee Huipputulosnäkymään ja itse peliin



Kuva Huipputulokset. Huipputuloksissa näkyy parhaiden pelaajien tulokset



Kuva Pelikenttä. Kortteja siirrellään klikkaamalla yhtä kohtaa ja sitten toista

# Keskeisimmät ongelmat työtä tehdessä

Huolellista suunnittelusta huolimatta tuli eteen odottamattomia ongelmia joita piti ratkaista lennosta. Pelin logiikan kanssa painiminen vei aikansa, joten lopputulos ei ole aivan ideaali olio-ohjelmointi ideaalin kannalta. Pakkaluokan olisi ehkä voinut tehdä fiksumminkin. Google avusti.

Tietokantaan tallennuksen kanssa oli liikaa ongelmia. Ensiksi olen tehnyt virheen WPF:n kanssa, sen jälkeen en saanut edes koodia toimimaan joten joutunut pyytämään apua opettajalta, ratkaisu oli tosi simppeli, mutta en ole itse huomannut siitä.

# Testaus

Testasimme tehdessä ominaisuuksia. Mitään automaattitestejä emme toteuttaneet. Lisäksi pelasimme peliä jonkin aikaa jotta voimme varmistaa sen toimivan.

# Itsearviointi

## Asko

4.5455555555…/5 Vaikka jotkin asiat ovat kovin sotkuisesti toteutettuja (pelin logiikka), onnistui projekti kuitenkin hyvin. Ja toteutuksessa on myös siistejä osioita, kuten pisteiden tallennus.

## Alex

2.5-3/5. Suurin ja hankalin osa pelilogiikan suunnittelusta ja toteuttamista on Askon tekemä. Olin auttamassa häntä tunneilla sekä olen toteuttunut highscore logiikan, tietokannan, tallennus ja scorien haku tietokannasta, sekä ensimmäisen version scoren .txt tiedostoon tallentamisesta. Mielestäni osuuteni tästä työstä onnistui aika hyvin, paitsi jotkut yksinerikertaiset asiat ovat vieneet liikaa aikaa.

# Kurssipalaute

Ei mitään valitettavaa.