15.5.2015

Projektityön dokumentti

Konsta Korhonen 425795

KJR: 2. vuosikurssi

1. **Yleiskuvaus**

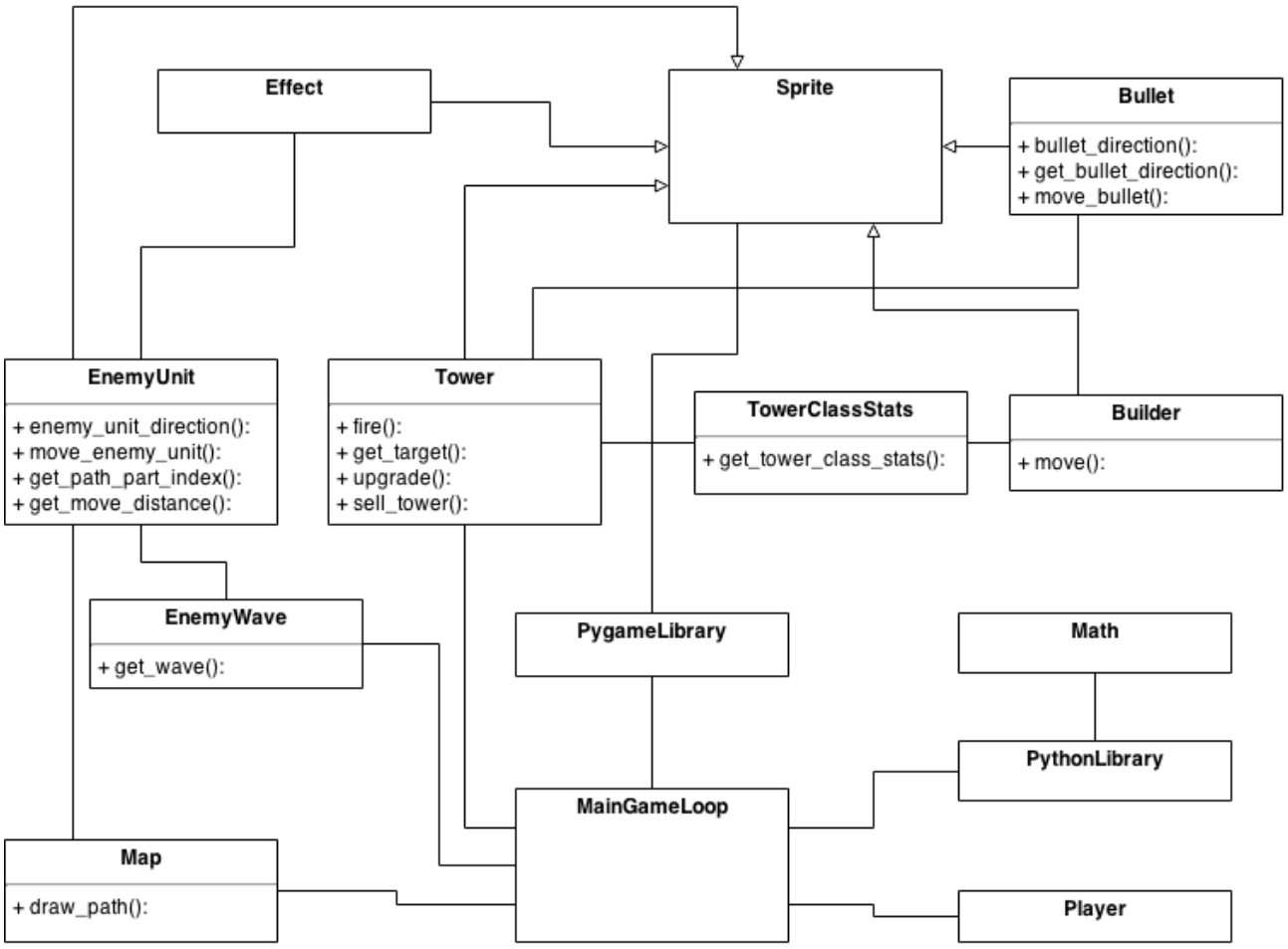
   Projekti on tornipuolustus-tyypin peli, joka sisältää kaksi tornityyppiä, kolme vihollistyyppiä, yhden kentän ja yhden pelimuodon, jossa puolustetaan 10 vihollisaallon verran. Pelin laajentaminen uusilla kentillä, vihollisilla, aalloilla ja torneilla olisi varsin helposti toteutettavissa. Pygamen käyttö projektissa helpotti projektia siinä määrin, että projektin vaikeustaso on lähempänä helppoa ja korkeintaan keskivaikea. Henkilökohtaisesti haastetta ja työtä oli reilusti tarpeeksi.

   Alun perin suunnitelmana oli tehdä ruudukkomallinen tornipuolustus, jossa yhteen ruutuun voi rakentaa yhden tornin, mutta päädyin vapaampaan rakentamiseen. Muista tornipuolustuksista pelin erottavan idean työstämiseen ei jäänyt tarpeeksi aikaa, joten peli on tylsän tavallinen tällä hetkellä.

2. **Käyttöohje**

   Ohjelma käynnistetään ajamalla MainGameLoop-moduuli. Pelin pelaaminen on yksinkertaista, ainakin jos on joskus pelannut tornipuolustus-tyypin peliä. Hiiriohjauksella valitaan halutut toiminnot GUI:sta, nappuloissa lukee tiedot siitä, mitä ne tekevät. Tornit ovat HUD:in keskivaiheilla, josta ne voidaan klikkaamalla ja raahaamalla viedä kentälle ja rakentaa klikkaamalla. Rakentamisen jälkeen torneja voi valita klikkaamalla, jolloin HUD:ssa näkyy tornin tiedot ja ampumisetäisyyttä kuvaava ympyrä tornin ympärillä.

3. Ohjelman rakenne



Luokista ja metodeista on lyhyet kuvaukset lähdekoodissa.

4. Algoritmit

Ammusten hakeutuminen kohteeseen:

Ammus luodaan tornin koordinaatteihin ja lisätään sprite-grouppeihin. Joka framella käydään läpi groupin sisältämät ammukset ja haetaan niille suunta, joka selvitetään laskemalla ammuksen ja kohteen koordinaattien välinen vektori ja sen suuntainen yksikkövektori. Ammusta liikutetaan yksikkövektorin suuntaan ammuksen nopeudesta vauhdista määrä.

Tornin kohteen valinta:

Jos vihollisia on kentällä, aloitetaan kohteen haku. Etäisyyttä jokaiseen viholliseen mitataan Pythagoraan lauseen avulla $l = \sqrt{(x_t - x_k)^2 + (y_t - y_k)^2}$ ja jos vihollinen on tornin ampumaetäisyydellä, lisätään se listaan. Sitten käydään läpi lista ja valitaan kohteeksi listan pisimmälle liikkunut vihollinen.

Vihollisten liikkuminen reittiä pitkin:

Jos ei olla reitin lopussa, haetaan ensin liikkumissuunta, joka määräytyy reitin koordinaattien perusteella. Haetaan reitin koordinaattien indeksit ja liikutetaan vihollista reittiä pitkin liikkumissuuntaan vihollisen nopeuden mukaan. Jos ollaan lähellä reitin kulmaa, eli lähempänä tai yhtä lähellä kuin vihollinen liikkuu framen aikana, siirretään vihollinen seuraavaan polun koordinaattiin. Tämä estää vihollisen liikkumisen polun yli, kun vihollisen nopeus on suurempi kuin yksi pikseli yhdessä framessa.

Reitin piirtäminen:

Luodaan Pygamen Rect-luokan oliot (suorakulmaista aluetta kuvaavia) jokaisen reitin koordinaatin välille ja täytetään ne mustalla värillä. Luodaan Rect-oliot myös reitin koordinaatteihin eli reitin kulmiin ja täytetään ne mustalla värillä. Tehdään sama mutta vähän kapeammilla Rect-olioilla ja valkoisella värillä, jolloin saadaan näkyviin mustareunuksinen ja keskeltä valkoinen reitti.

Vihollisten luonti aalloissa:

Aallot on määritelty listassa, jonka alkiot ovat listoja, joiden alkiot ovat listoja, joiden ensimmäinen alkio määrittää vihollistyypin ja toinen alkio sen määrän. Kun next wave –nappulaa painetaan, aloitetaan vihollisten luonti, jos aaltoja on jäljellä. Luodaan vihollistyyppiä määrän mukaan tietyllä intervallilla ja siirrytään seuraavaan vihollistyyppiin.

5. **Tietorakenteet**

Pythonin valmiit tietorakenteet, pääasiassa listat.

6. **Tiedostot**

Ohjelma vaatii toimiakseen Pygamen uusimman version asennuksen. Muita tiedostoja ovat png-muotoiset kuvatiedostot, joita tarvitaan GUI:n ja muiden grafiikoiden piirtämiseen. Torni-, vihollis- ja ammustyyppien sekä kenttien tiedot ovat osa lähdekoodia, mutta pienillä muokkauksilla ne voitaisiin lukea tekstitiedostoista.

7. **Testaus**

Testaus suoritettiin ohjelman ajamisella ja tarkastelemalla, toimiiko ohjelma halutulla tavalla. Print-komennot koodin vaiheissa auttoivat tarkastelemaan, mitkä koodin rivit käydään läpi ja muun tiedon, kuten koordinaattien hahmottamisessa. Loppuvaiheessa pelin toimivuutta testattiin pelaamalla peliä ja tarkkailemalla mahdollisien bugien esiintymistä ja käyttöliittymän toimivuutta. Suunnitelmassa mainitsin että osia ohjelmasta voisi yksikkötestata mutta en päätynyt tekemään näin varsinaisessa projektissa.

8. **Ohjelman tunnetut puutteet ja viat**

Isoimpia puutteita on sisällön puute, peli kaipaisi lisää torni- ja vihollistyyppejä sekä kenttiä, jotka voisi valita kenttävalikosta. Näiden ominaisuuksien lisääminen ei olisi vaikeaa ohjelman rakenteen laajennettavuuden ansiosta. Myös äänet ja musiikki on helppo toteuttaa pygamen avulla ja lisätä projektiin.

Kun viimeinen vihollisaalto on tuhottu, peli siirtyy game over -screeniin. Tämä johtuu aikataulun kiireellisyydestä, oikeasti game over -screenin sijaan siirryttäisiin victory-screeniin, josta voi edelleen siirtyä päävalikkoon tai lopettaa pelin. Tämä on helposti korjattavissa, mutta toisaalta peli ohi -viesti kuvaa myös voittoa, sillä tässäkin tilanteessa peli on päättynyt.

Ohjelman rakennetta olisi voinut toteuttaa joissain osissa enemmän luokkapohjaisena tai funktioilla, esimerkiksi monet MainGameLoopin rakenteista. Eri screenit olisi voinut toteuttaa omilla luokillaan selkeyden lisäämiseksi. Käyttöliittymän piirtoa olisi myös voinut toteuttaa luokilla. MainGameLoopin sisällön pitäisi olla pääasiassa main-funktiota, mutta pygamen käynnistymisessä oli tällä rakenteella ongelmia jossain vaiheessa projektia ja myöhemmin en muistanut tehdä sitä uusiksi.

9. **3 parasta ja 2 heikointa kohtaa**

Hyviä kohtia:

-Reitin toteutus eli piirto ja vihollisten reittiä pitkin kulkeminen

-Käyttöliittymä, mielestäni käyttö selkeää

-Laajennettavuus, sisältöä helppo lisätä

Heikompia kohtia:

-MainGameLoop, rakenteen voisi tehdä selkeämmäksi

-ToverClassStats, tiedot voisi tallentaa listoihin ja rakenne voisi muutenkin olla parempi

10. **Poikkeamat suunnitelmasta**

Lopullinen peli oli usealla tavalla erilainen kuin suunnittelin, esimerkiksi ruudukkomallin sijaan päädyin vapaaseen tornien rakentamiseen. Aikataulu meni suunniteltua myöhemmäksi, ajankäyttöarvio meni alakanttiin.

11. **Toteutunut työjärjestys ja aikataulu**

22.4. Projektin aloitus, Pygamen käytön harjoittelua, näytölle spritejen piirtäminen ja niiden liikuttaminen

28.4. – 9.5. Ohjelman perusrakenteen suunnittelua ja koodausta, ammusten ohjaus, vihollisten liikkuminen, tornien kohteen valinta

9.5. – 13.5. Rakenteen tarkennus, vihollisten luominen aaltoina, reitin piirtäminen

13.5. – 15.5. Käyttöliittymän koodausta: screenit ja tekstit, tornien valinta ja toiminnot, viimeistelyä, kommenttien lisäämistä, yleistä sähläystä kiireessä

Suunnitelma oli hyvin suurpiirteisesti laadittu, ainakin aikataulu poikkesi suunnitellusta sen myöhäisellä aloituksella ja työn painottumisella loppupäähän, mikä oli kuitenkin odotettavaa.

12. Arvio lopputuloksesta

Pääasiassa olen tyytyväinen projektiin ja siihen että sain sen ylipäätään tehtyä näin pitkälle. Pelin sisältö jäi vähäiseksi, mutta laajennettavuus on yksi sen vahvoista puolista, joten sisällön lisääminen onnistuisi varsin helposti.

13. Viitteet

1. Pygamen kotisivut. Saatavissa: www.pygame.org
2. Program Arcade Games With Python And Pygame –sivusto. Saatavissa: www.programarcadegames.com
3. The Python Game Book –sivusto. Saatavissa: www.thepythongamebook.com
4. Python Standard Library

14. Liitteet

Lähdekoodi GitLabissa.