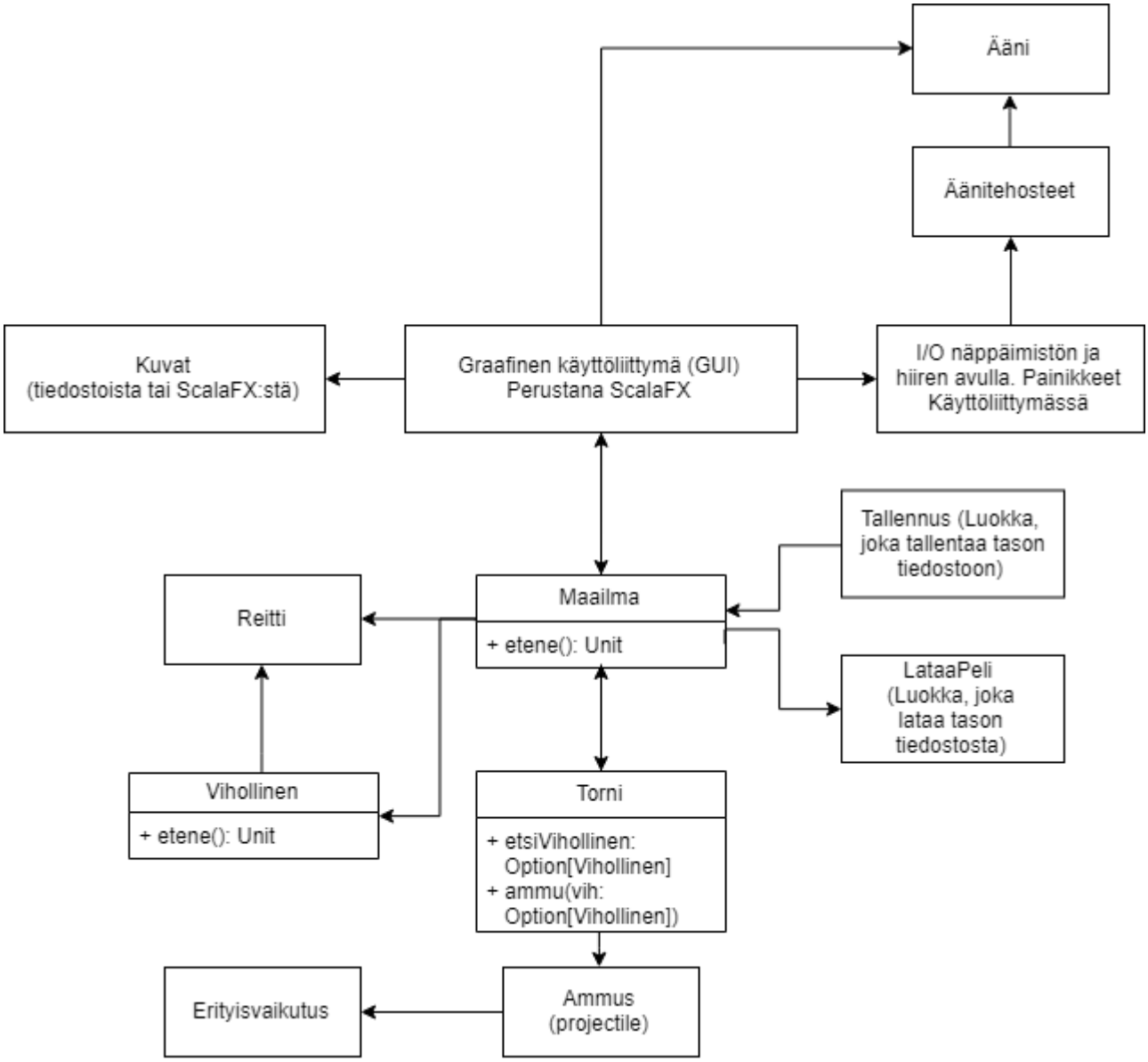
Anton Pirhonen, 653318, Tietotekniikka, 1. vuoden opiskelija (2018), 10.2.2019

Ohjelman rakennesuunnitelma



Ohjelman sisäistä mallia kuvaa luokka Maailma. Se kokoaa yksittäiset luokat yhteen ja antaa käyttöliittymälle toimintaohjeita. Reitti-luokka mallintaa reittiä, jota pitkin viholliset kulkevat. Vihollisten on oltava tietoisia reitistä, sillä ne kysyvät siltä seuraavaa sijaintiaan. Vihollinen-luokka etenee reittiä pitkin etene()-metodillaan. Tornilla on ampuma-alue. Se etsii ampuma-alueensa sisällä olevat viholliset, valitsee niistä pisimmälle edenneen ja kutsuu Ammu-metodia. Metodi lähettää vihollista vahingoittavan ammuksen kohti vihollista. Ammuksilla voi olla erityisvaikutusta, mutta ne ovat lisäominaisuus, jonka toteutan jos aikaa on ylimääräistä.

Peli etenee kutsumalla Maailma-luokan etene()-metodia. Tällöin jokaiselle viholliselle kutsuteen etene()-metodia. Sen jälkeen jokaiselle tornille kutsutaan Ammu(etsiVihollinen) mikäli on kulunut riittävästi aikaa siitä kun se ampui viimeksi.

Peliin kuuluu myös ominaisuus tallentaa ja ladata tallennettu peli tiedostosta. Molempia kuvaa oma luokkansa. Myös graafisen käyttöliittymän uusi peli-vaihtoehto hyödyntää LataaPeli-luokkaa, mutta tällöin se lataa alkuvaiheessa olevan pelin. Lataa peli luo maailmaan oliota kuten tulevat vihollisaallot ja pelaajan rakentamat tornit mikäli tallennustiedostossa sellaisia on.

Graafiseen käyttöliittymään kuuluu ikkuna, johon pelin malli, eli Maailma-luokka, piirretään. Graafinen käyttöliittymän perustana käytän ScalaFX-kirjastoa. Käyttöliittymässä on ylhäällä valikko, josta voi valita uuden pelin aloittamisen tai vanhan pelin lataamisen. Kun pelin on ladannut, valikosta voi myös lisätä torneja peliin sekä tallentaa/käynnistää uuden pelin. Pelissä olevat grafiikat rakennan joko kirjastosta muodostamistani tai netistä ladatuista kuvista.

Käyttäjän kokemaan rajapintaan liittyy myös äänet. Torneilla on ampumiseen liittyvä ääniefekti. Toinen ääniefekti aktivoituu, kun ammus osuu viholliseen. Lisään peliin taustamusiikin, jota soitan omalla säikeellä tai Futurella, jottei musiikin soittaminen keskeytä peliä.

Käyttötapauskuvaus

Pelaaja käynnistää sovelluksen. Pelaaja on aloitusruudussa. Ylhäällä olevasta valikosta hän valitsee kohdan uusi peli. Tällöin käyttöliittymä muuttuu ensimmäistä tasoa kuvaavaksi maailmaksi. Taustalla luokka LataaPeli lataa tiedostosta tasoon liittyvät tiedot kuten vihollisten reitin ja vihollisaaltojen ominaisuudet. Näistä tiedoista se muodostaa pelin maailmaa kuvaavan Maailma-olion. Pelaaja voi nyt asettaa uusia torneja maailmaan ylävalikosta. Kun hän asettaa tornin, niin maailma rekisteröi tornit-kokoelmaansa uuden tornin. Toimenpide toistetaan jokaiselle pelaajan asettamalle uudelle tornille. Kun pelaaja kutsuu näytössä olevan painikkeen avulla seuraavan vihollisaallon, niin Maailman viholliset- kokoelmasta luodaan viholliset säännöllisin väliajoin kulkemaan reittiä pitkin. Viholliset kysyvät Reitti- luokalta seuraavaa sijaintiaan ja etenevät tasaista vauhtia pelin tickien myötä. Kun vihollisaalto on aktiivinen tornit etsivät ympäristöstään vihollisia etsiVihollinen-metodilla ja ampuvat, mikäli vihollinen alueelta löytyi. Ampuessa torni luo ammuksen, jonka lentoradan se laskee kulkemaan suoraa vihollista päin. Ammus lentää ja mikäli se liikkuu tarpeeksi lähelle vihollista, niin vihollisen elämäpisteet vähenevät. Tätä toistetaan kunnes viholliset ovat joko kuolleet tai päässeet reitin loppuun. Jos vihollinen pääsee reitin loppuun se katoaa ja maailma-luokassa sijaitsevat elämäpisteet vähenevät. Kun taso on ohi, niin sama toimenpide toistetaan seuraavalle tasolle kun pelaaja sen haluaa kutsua.

Algoritmit

Pelissä käytettävät algoritmit ovat suhteellisen yksinkertaisia. Tilanne, jossa tarvitaan kokoelmasta etsimistä ja perusaritmetiikkaa monipuolisempia laskuja, on kun torni lähettää ammuksen vihollista kohti. Tällöinkin sen pitää laskea vektori kahden pisteen välillä, joka onnistuu yksinkertaista trigonometriaa hyödyntämällä

Tietorakenteet

Kokoelmia tarvitaan mm. vihollisten, tornien ja ammusten mallintamiseen pelissä. Kokoelmien on oltava muuttuvatilaisia, sillä alkioiden määrä muuttuu tilanteesta riippuen jatkuvasti.

Aikataulu

18.2-3.3	• ScalaFX:ään tutustuminen • Pelin malliin vaadittavien luokkien alkeellisten versioiden rakentaminen • Käyttöliittymän rakentamisen aloittaminen
4.-17.3	• Käyttöliittymän kehittäminen • Mallin luokkien kehittäminen • Luokkien testaus
18.-31.3	• Käyttöliittymän kehittäminen • Luokkien kehittäminen toimiviksi • LataaPeli-luokan kehittäminen • Uusien aliluokkien kehittäminen piirreluokille Vihollinen ja Torni • Luokkien toiminnan testaus
1.-14.4	• TallennaPeli-luokan kehittäminen • Grafiikoiden ja äänien suunnittelu • Uusien aliluokkien kehittäminen piirreluokille Vihollinen ja Torni • Luokkien toiminnan testaus: Pelin pitäisi olla pelattava ja toimintakunnossa
15.-23.4	• Pelin viimeistely grafiikoiden, äänien ja toiminnan osalta

Yksikkötestaussuunnitelma

Pelin testaamisen apuna on graafinen käyttöliittymä, jonka avulla on helppo kerätä palautetta luokkien yhteistoiminnasta. Aluksi on tärkeää testata, että Viholliset osaavat kulkea Reittiä pitkin. Toimivuuden voi testata, kun viholliset ja reitti visualisoidaan käyttöliittymään. Sen jälkeen testataan, että reitin loppuun päässyt vihollinen vähentää elämäpisteitä. Kun viholliset ja reitti on saatu toimimaan, lisätään tornit. Aluksi on hyvä testata, että tornit havaitsevat vihollisen alueellaan. Tällöin torni voi esim. tulostaa konsolille viestin, joka kertoo, että se havaitsee vihollisen. Kun tiedetään, että torni havaitsee vihollisen testataan, että se osaa lähettää ammuksen kohti vihollista. Lopuksi testataan, että ammus vahingoittaa vihollista.

Kun nämä toiminnot on testattu peli on toimintakunnossa. Tämän jälkeen testataan uusia luokkia ja ominaisuuksia niitä lisättäessä.

Kirjallisuusviitteet ja linkit

Mahdollisia grafiikoita peliin: opengameart.org

ScalaFX: http://www.scalafx.org/

Open Source- musiikkia: https://freemusicarchive.org

Scala-API: https://www.scala-lang.org/api/current/