TIE-20200
Ohjelmistojen suunnittelu
Syksy, 2014

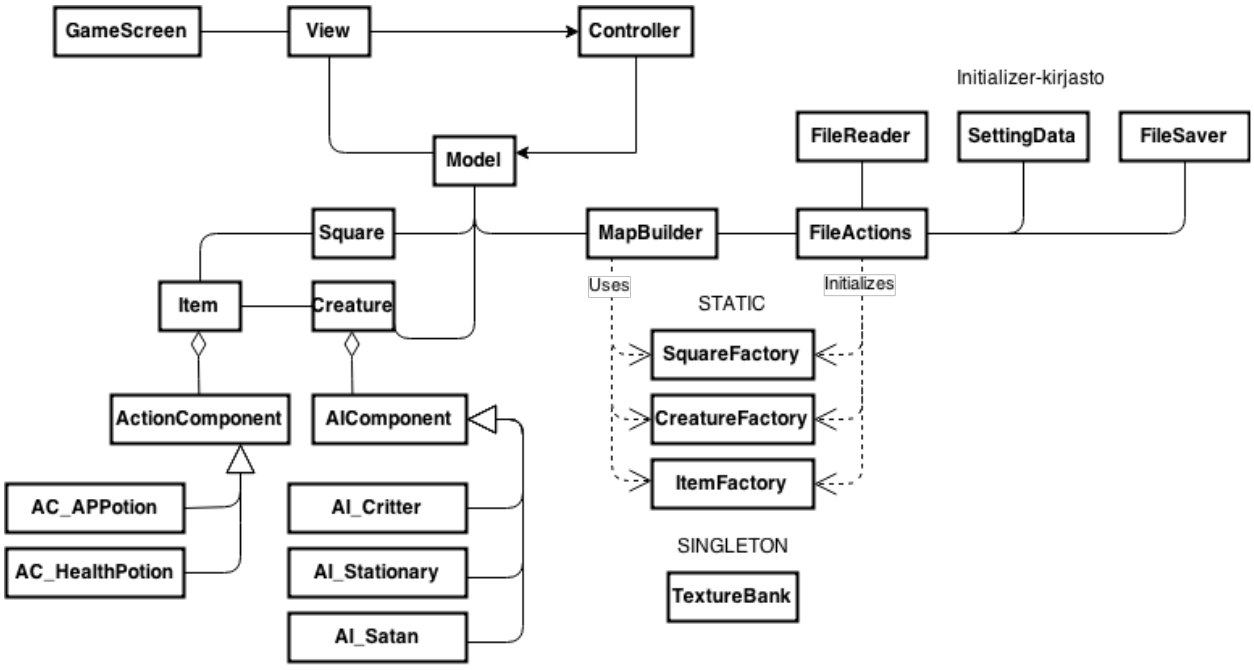
Harjoitustyödokumentaatio
Marjametsä - The Dark Knowledge
Ryhmä 25

# 1. Yleisesti toteutetusta ohjelmasta

Marjametsä - The Dark Knowledge (tästä eteenpäin vain MDK) on kaksiulotteinen peli, jossa pelaaja ohjaa Laura-nimistä sankaritarta vaarallisen metsän läpi tavoitteenaan löytää kadottamansa enemmän tai vähemmän demoninen reliikki. Matkalla Laura kerää mukaansa esineitä, taistelee metsän eläimiä ja olioita vastaan, sekä lopulta päihittää joko itse Pimeyden Ruhtinaan tai kipittää takaisin mökkiinsä piiloon.

MDKn ohjelmakoodi on kirjoitettu c++-kielellä käyttäen Qt Creatoria, jonka avulla myös käyttöliittymä on toteutettu. Toteutuksessa on myös kiinnitetty erityistä huomiota siihen, että peliä olisi mahdollista kehittää toiminnallisesti ja sisällöllisesti erillään, mikä on peliohjelmoinnissa de facto lähestymistapa.

# 2. Ohjelman rakenne




Kuva esittää yleispiirteittäin ohjelman sisäiset luokkasuhteet.

### 2.1 MVC

MDK:n vastuujako on suunniteltu MVC-mallin mukaisesti siten, että mallin rajapinnat toteuttaa MarjaModel, MarjaView ja MarjaController. Käyttöliittymäpuolesta vastaava MarjaView-luokka sisältää erikseen pelitilan piirtämisestä ja hiirisyötteiden vastaanottamisesta vastaavan GameScreen-luokan. MarjaView välittää vastaanotetut syötteet MarjaControllerille, ja piirtokäskyn tullessa se lukee MarjaModelista tarvittavat tiedot uuden tilan esittämiseen. MarjaController toimii lähinnä siltana MarjaModelin ja MarjaViewin välillä, minkä lisäksi se suorittaa muutamia syötteenkäsittelyitä. MarjaModel on MDK:n varsinainen pelitilaluokka, ja se osaa toimia controllerilta tulevien syötteiden perusteella, sekä käskyttää viewiä päivittääkseen näkymän.

Sen lisäksi että MarjaModel toteuttaa MVC-mallin Model-rajapinnan, toteuttaa se myös pelin logiikan. Varsinen GameLoop sijaitsee MarjaModelissa, minkä lisäksi siinä on tallessa pelikenttä, kentällä olevat oliot, sekä muutamia pelin tilaan liittyviä tietoja. MarjaModel tarjoaa pelin eri komponenteille useita apufunktioita, joiden avulla niiden on mahdollista muokata pelin tilaa.

### 2.2 Peliluokat

Varsinainen peli rakentuu kolmen eri luokan yhteistoiminnan varaan. Nämä luokat ovat Creature (kaikki pelin viholliset, sankari, pääpahis), Item (kaikki pelin tavarat), ja Square (kaikki pelin ruudut), joista jokaiselle on olemassa oma tehdasluokka. Olennaista on huomata, että creatureita, squareja ja itemeitä ei erotella luokkatasolla, vaan esimerkiksi sekä orava että pääpahis ovat täysin sama luokka, joiden eroava toiminnallisuus toteutetaan komponenteilla.
Pelikentän rakentaminen ja siten pelin alustaminen tapahtuu MapBuilder-luokalla, joka manipuloi suoraan MarjaModelin sisältöä. MapBuilderin tehtävänä on myös alustaa tehtaat, varmistaa luetun datan oikeellisuus, sekä toimia tallennus- ja latausoperaatioiden käsittelijänä.

### 2.3 Tiedostonkäsittely

Tiedostojen tallentaminen ja lukeminen suoritetaan Initializer-kirjaston avulla, joka hieman muokattu versio valmiiksi saadusta tiedostonlukukirjastosta. Paitsi että kirjasto tukee pelitilan lataamista ja tallentamista, voidaan sen avulla lukea erilliset alustustiedostot, joissa määritetään eri squaret, itemit ja creaturet.

## 3. Ajoaikainen käytös

1. Mainissa luodaan MVC-yhteydet
2. Alustetaan tehtaat ja mapbuilder lukemalla creaturejen, itemien, squarejen ja templatejen alustustiedostot
3. Mapbuilder rakentaa talon sisäpuolen, asettaa sen modelin kentäksi

4. Pelisilmukka alkaa, pelaaja näkee käyttöliittymän ja talon sisäpuolen.
5. Pelaaja voi poimia talosta yhden esineen. Kun hän kävelee ovelle, mapbuilder rakentaa pääkentän ja varsinainen peli alkaa.
6. Pelaaja voi tehdä asioita, kunnes häneltä loppuu toimintapisteet. Kun toimintapisteet loppuvat, annetaan metsän eläimille vuoro liikkua. Jos pelaaja kuolee, peli loppuu.
    a. Pelaaja voi liikkua tyhjiin ruutuihin
    b. Pelaaja voi poimia esineitä kävelemällä niiden päälle
    c. Pelaaja voi hyökätä vihollisten kimppuun yrittämällä kävellä niiden ruutihin
    d. Pelaaja voi aktivoida esineitä
    e. Pelaaja voi pudottaa esineitä
    f. Pelaaja voi tutkia ympäristöään
    g. Pelaaja voi lopettaa vuoronsa ennenaikaisesti
7. Pelaaja poimii reliikin, ja pääpahis syntyy.
8. Peli päättyy kun pelaaja joko päihittää pääpahiksen, palaa takaisin talolleen, tai kuolee.

# 4. Pelin laajentaminen

Pelin laajentamisesta on pyritty tekemään mahdollisimman yksinkertaista, jotta ohjelmointitaidottomat gameplay designerit voisivat testailla asioita ihan itsenäisesti. Sekä creatureiden, itemien, squarejen että templatejen lisääminen onnistuu vain lisäämällä yksi rivi alustustiedostoon, paitsi jos halutaan toteuttaa uutta toiminnallisuutta, milloin joudutaan ohjelmoimaan uusi komponentti, ja rekisteröidä se oikeaan factoryyn.

## 4.1 Esimerkki uuden Creaturen lisäämisestä.

Gameplay designer uskoo, että helposti lahdattavien viherpiipertäjien lisääminen peliin toisi uskomatonta uudelleenpeluuarvoa. Hänellä sattuu olemaan sopiva grafiikka ja ajatus creaturen attribuuteista, eli ainoastaan lisääminen on jäljellä.

1. Lisätään grafiikka resursseihin sopivalla nimellä, esim “vinii.png”
2. Muokataan resources/initfiles/creature.txt-tiedostoa lisäämällä seuraava rivi: 1-viherpii-critter-nasty hippie-2-0-5-5-3-rabbit,squirrel-:/graphics/vinii.png
3. Uusi creature on nyt lisätty, ja se voi ilmestyä pelissä tier 1 alueille.

Vastaavalla tavalla voidaan lisätä myös tavaroita, squareja ja templateja. Jokaisen alustustiedoston alussa on rivi, joka kertoo oikean syntaksin lisäämiselle.

## 4.2 Esimerkki uuden AI-komponentin lisäämisestä.

Designerin mielestä viherpiipertäjän käyttäytyminen on critter-AI:n takia liian epärealistinen. Hän pyytää ohjelmoijaa tekemään uuden “viherpiipertaja”-AI:n, joka vastaisi odotuksia paremmin. Ohjelmoija ryhtyy toimeen.

1. Luodaan uusi AI_Viherpiipertaja-luokka, joka periytyy AIComponent-luokasta.

2. Luodaan toteutus virtuaaliselle AIAction-funktiolle, missä toteutetaan varsinainen tekoäly.
3. Muokataan CreatureFactoryn verifyTemplateIntegrity- ja makeFromTemplate-funktioita siten, että ne yhdistävät “viherpiipertaja”-AItunnisteen AI_Viherpiipertaja-luokkaan.
4. Uusi AI on nyt lisätty, ja tunnistetta “viherpiipertaja” voidaan nyt käyttää creatureja määritettäessä.