Totte Partanen

Tietokoneroolipelin prototyypin kehitysvaiheet

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Tietotekniikan koulutusohjelma

Insinöörityö

26.4.2016

Metropolia

Tekijä Otsikko Sivumäärä Aika	Totte Partanen Tietokoneroolipelin prototyypin kehitysvaiheet 42 sivua 7.4.2016
Tutkinto	Insinööri (AMK)
Koulutusohjelma	Tietotekniikka
Suuntautumisvaihtoehto	Ohjelmistotekniikka
Ohjaaja	Lehtori Antti Laiho

Insinöörityössä suunniteltiin ja toteutettiin tietokoneroolipelin prototyyppi, joka sisältää pelin pelattavuuden kannalta olennaisimmat pelimekaniikat. Lisäksi tutkittiin tietokoneroolipeleihin soveltuvia erilaisia pelimoottoreita ja pelinkehitystyökaluja. Peliprototyyppi kehitettiin Unity3D-pelimoottorilla käyttäen C#-ohjelmointikieltä.

Lopputyön pääpaino on tietokoneroolipelin peliprototyypin suunnittelussa ja sen pelimekaniikoiden toteutusratkaisuissa. Suunnitteluosio pitää sisällään peliprototyypin rakenteellisen suunnitelman ja käsittelee myös pelin pelaamisen kannalta olennaisia asioita. Peliprototyypin toteuttamiseen liittyvä osio on jaettu eri pelattavuuden kannalta tärkeisiin ominaisuuksiin, jotka on käsitelty tarkemmin jokaisen osion sisällä.

Kokonaisuudessaan peliprototyypin suunnittelu ja toteutus oli tekijälleen hyvin kehittävä prosessi joka opetti lisää ohjelmoinnista sekä pelikehityksestä. Peliprototyypin testaaminen olisi voinut olla aktiivisempaa sen käyttöliittymän ja pelidatan tallennusominaisuuksien suhteen. Peliprototyyppiä kehitetään eteenpäin lopputyön jälkeen ja alustavasti sen toteuttamista jatketaan harrasteprojektina.

Avainsanat	Tietokoneroolipeli, C#, Unity, Peliprototyyppi

Author Title Number of Pages Date	Totte Partanen Development stages of role-playing video game 42 pages 7 April 2016
Degree	Bachelor of Engineering
Degree Programme	Information Engineering
Specialisation option	Software Engineering
Instructor	Senior Lecturer Antti Laiho

Subject of the thesis was to plan and implement a prototype for a role-playing video game, which includes the most essential game mechanics for its playability. In addition, research for video game engines applicable for role-playing video game development was made. The development of the prototype was made with Unity3D game engine using C# for programming language.

Thesis' main focus is planning and implementing game mechanics for the prototype. Game prototype planning section includes design views and also deals with subjects about game playability. The section which includes the implementation of the prototype is divided in smaller sections. Each section focuses on one game element and its solutions for implementation.

As a whole, the development of the video game prototype was very resourceful for its maker. It taught more about game development and programming. There could have been more active testing cycles when implementing the user interface and a handler for the game data. The game prototype will be further developed in the future, starting first as a hobby project.

Keywords	Role-playing video game, C#, Unity, Game prototype

Sisällys

Lyhenteet ja käsitteet

C#	Microsoft-yhtiön .Net-konseptia varten kehittämä ohjelmointikieli.
Boolean	Ohjelmoinnissa käytettävä totuusarvo, jonka avulla voidaan tehdä erilaisia vertailuja.
HnS	Hack and Slash. Termi viittaa pelimuotoon, joka painottaa pelissä käytäviä taistelutilanteita.
NPC	No playable character. Videopelissä esiintyvä pelihahmo, jota pelaaja ei voi kontrolloida.
Skripti	Ohjelmointikielellä luotu kokonaisuus joka toimii omana komponenttinaan.
Indie-peli	Itsenäisesti kehitetty videopeli, joka ei ole rahoitettu ison pelikehitysyrityksen kautta.
Sprite	Tietokonepeleissä käytetty tietokonegrafiikan menetelmä, joka perustuu pikseligrafiikalla toteutettuihin kuvahahmoihin.
UI	User Interface. Käyttöliittymä, jonka kautta käyttäjä käyttää ohjelmistoa tai muuta tuotetta.
Komponentti	Peliobjektiin liitettävä osa, joka sisältää ohjelmoituja, toiminnallisuutta lisääviä ominaisuuksia.

1 Johdanto

Insinöörityön tarkoituksena on tietokoneroolipelin prototyypin suunnittelu ja toteutus Windows-käyttöjärjestelmälle. Tietokoneroolipeli on edelleen kehittyvä videopelien alatyyppi, joka on säilyttänyt paikkansa alati kasvavilla videopelimarkkinoilla. Pelikehityksen modernisoitumisen myötä yhä useammalle kehittäjälle on tarjolla päivä päivältä parempia työkaluja pelien suunnitteluun ja toteuttamiseen. Nykyään on tarjolla kaupallisia valmiita pelimoottoreita, joita pelikehittäjät voivat käyttää huoletta ilman suuria investointeja kehitystyökaluihin. Tämä näkyy indie-peleissä, joita julkaistaan vuosittain kymmeniä. Pienemmällä kokoonpanolla on mahdollista työskennellä myös hieman suurempien peliprojektien parissa, mikä aikaisemmin ei ollut mahdollista.

Tietokoneroolipeleillä on lukuisia eri lajityyppejä, jotka vaihtelevat pelattavuudeltaan ja sisällöltään. Ensimmäiset tietokoneroolipelit olivat poikkeuksetta kaksiulotteisia spritegrafiikkaa käyttäviä, yksinkertaisen näköisiä pelejä. Teknologian kehittyessä tietokoneroolipelit kehittyivät visuaalisesti ja uusia lajityyppejä syntyi erilaisten pelisuunnitteluratkaisujen tuloksena. Uusia ja vanhoja tietokoneroolipelejä kuitenkin yhdistää ydinajatus roolipelihengestä ja sen sisältämistä toimintaperiaatteista. Suosituimmat tietokoneroolipelien lajityypit ovat säilyttäneet paikkansa vahvasti, vaikka uusia lajityyppejä oletetaan syntyvän edelleen lisää.

Tarkoituksena oli kehittää peliprototyyppi, joka sisältää tietokoneroolipelin pelattavuuden kannalta kaikki olennaisimmat pelimekaniikat. Pelikehitys tehtiin Unity3D-pelikehitysympäristössä käyttäen C#-ohjelmointikieltä. Lopputyö painottaa pelikehitystä ja peliprototyypin toteutukseen liittyviä ohjelmointiratkaisuja. Peliprototyyppi noudattaa japanilaisen koulukunnan tietokoneroolipelien kaavaa pelattavuudeltaan sekä sisällöltään.

Työ koostuu viidestä osiosta. Ensimmäinen osio määrittelee tietokoneroolipelit ja kertoo hieman niiden alkuperästä sekä erilaisista alatyypeistä. Toisessa osiossa käydään läpi erilaisia tietokoneroolipelin kehitykseen soveltuvia kehitysympäristöjä ja pohdiskellaan niiden tarjoamaa potentiaalia prototyypin kehityksen kannalta. Seuraavat kaksi osiota keskittyvät itse peliprototyypin suunnitteluun ja toteutukseen. Lopuksi analysoidaan lopputyön tuloksia ja käydään läpi peliprototyypin jatkokehityksen kannalta olennaisia asioita.

2 Tietokoneroolipelien määrittely

2.1 Tietokoneroolipeli yleisesti

Tietokoneroolipeli (eng. Role Playing Game), lyhennettynä rpg, on yksi videopelien alatyypeistä. Kyseinen alatyyppi sisältää erilaisia pelejä. Kaikkia tähän alatyyppiin kuuluvia pelejä yhdistää se, että niihin on liitetty ominaisuuksia roolipeleistä. Roolipelit ovat vuorovaikutteisen kerronnan muoto, jossa osallistujat eläytyvät kuvitteellisten hahmojen rooleihin. Tietokoneroolipelin tyypillisiä elementtejä ovat

- tarina tai pääjuoni
- sivujuonet tai tehtävät
- kokemuspisteet
- hahmoluokat
- hahmonkehitys

Tietokoneroolipeleissä ohjataan yhtä tai useampaa hahmoa. Usein hahmot keräävät kokemuspisteitä pelin aikana ja sitä kautta kehittyvät voimakkaammiksi ja taitavammiksi. Kokemuspisteitä saa kukistamalla vihollisia ja myös mahdollisesti tekemällä erilaisia tehtäviä. (1.)

Tietokoneroolipelit poikkeavat varsinaisista roolipeleistä. Roolipelien idea useimmiten on, että pelaajat luovat hahmon ja eläytyvät sen persoonaan. Yksinpelattavissa tietokoneroolipeleissä pelaajalle esitellään ennalta määritellyt hahmot, joilla on valmis taustatarina. Poikkeuksia kuitenkin esiintyy esimerkiksi Fallout-pelisarjan peleistä, joissa pelaajan eläytyminen hahmoon on suuressa roolissa pelin etenemisen kannalta (1). Kuvassa 1 nähtävä Fallout 2 -peli perustuu kuvan mukaisiin avoimiin aluekokonaisuuksiin, jotka yhdessä muodostavat ison pelimaailman.

Kuva 1. Vuonna 1998 julkaistu Fallout 2 painottaa pelaajan omia valintoja.

2.2 Historia

Tietokoneroolipelien lajityyppi syntyi 70-luvun puolivälissä suurtietokoneille tehdyillä yksinkertaisilla peleillä, jotka ottivat vaikutteita kynällä ja paperilla pelattavista kuvitteellisista roolipeleistä, kuten Dungeons & Dragons. Vaikuttajina toimivat myös siihen aikaan suosiossa olleet muut seikkailupelit, sekä fantasiakirjallisuus. (2.)

Japanilaiset tietokoneroolipelit Dragon Quest (1986) sekä Final Fantasy (1987) keräsivät paljon suosiota osakseen, ja aiheuttivat tietokoneroolipelien jaon kahteen eri koulukuntaan.

Länsimainen ja Japanilainen koulukunta

Tietokoneroolipelit kategorioidaan usein länsimaiseen tai japanilaiseen koulukuntaan. Länsimaisen koulukunnan roolipelit antavat pelaajalle enemmän vapautta persoonallisen hahmon luomisessa ja kehittämisessä sekä pelimaailman tutkiskelussa. Japanilaistyyliset pelit ovat lineaarisempia ja etenevät usein valmiin tarinan mukana, jolloin ne sisältävät enemmän seikkailupelielementtejä. Japanilaisen koulukunnan tietokoneroolipe-

lejä usein yhdistää erillisessä pelitilassa käytävät, valikoiden kautta ohjattavat kamppailutilanteet, kun taas länsimaisen puolen pelit alkoivat pelien kehittyessä painottamaan enemmän reaaliaikaista, kuitenkin vuoropohjaista kamppailua. (3.)

2.3 Tietokoneroolipelien eri lajityypit

Tietokoneroolipeleille on syntynyt sen elinkaaren aikana lukuisia eli lajityyppejä ja uusia lajityyppejä on syytä odottaa lähitulevaisuudessa. Tietokoneroolipelien lajityypit poikkeavat aina jollain tavalla niiden tavanomaisesta rakenteesta.

Toiminnalliset tietokoneroolipelit

Tyypillisesti toiminnalliset tietokoneroolipelit (eng. Action Role-Playing Game), keskittyvät yhden pelihahmon ohjaamiseen reaaliajassa. Pelien pääpaino on vahvasti toiminnallista ja ne keskittyvät taisteluihin, jolloin esimerkiksi tarina saattaa olla sisällön kannalta pienemmässä roolissa (4). Taistelu on useimmiten HnS-tyylistä, josta kuvassa 2 on esimerkki. Pelaaja ohjaa pelihahmoa taisteluissa suoraan näppäimistöllä ja hiirellä tai peliohjaimella ilman erillisiä taisteluvalikoita.

Kuva 2. Toiminnalliset tietokoneroolipelit painottavat kamppailutilanteita. (Peli: Ys Origin)

Taktiset tietokoneroolipelit

Taktinen tietokoneroolipeli on lajityyppi, joka yhdistää tavanomaiseen asetelmaan elementtejä strategiapeleistä. Lajityyppi on suositumpi Japanissa kuin länsimaissa, joissa se tunnetaan nimellä simulaatioroolipeli (eng. Simulation Role Playing Game).Asetelma on yleensä normaalin tietokoneroolipelin tapainen, mutta taistelutilanteissa pelihahmojen sijoittelu tapahtuu ruutuihin jaetulla kartalla. Hyvin pelatut pelihahmojen sijoittelut ovat voiton kannalta ratkaisevia. (5.) Pelien taistelutilanteiden eteneminen muistuttaa etäisesti shakin pelaamista. Eri hahmoluokat liikkuvat ja taistelevat eri tavoin mikä lisää pelin strategista henkeä. Pelien taistelutilanteiden käyttöliittymät on rakennettu usein ruutupohjaisen liikkumisen ympärille, kuten 3 kuvasta käy ilmi.

Kuva 3. Taktisten tietokoneroolipelien avain on pelihahmojen paras mahdollinen sijoittelu pelikartalla. Peli: Disgaea: Hour of Darkness

Massiiviset monen pelaajan tietokoneroolipelit

Massiivinen monen pelaajan tietokoneroolipeli on suosittu tietokoneroolipelien lajityyppi, joka mahdollistaa monen pelaajan yhteispelaamisen internetyhteyden välityksellä. Lajityypin pelit yleensä sisältävät jaetun kuvitteellisen roolipelimaailman, johon pelin hankkimalla voi kirjautua sisään. (6.)

Pelit yhdistelevät eri tietokoneroolipelien lajityyppejä. Kaikkia pelejä kuitenkin yhdistää usein monipelaaminen. Pelaajat voivat muodostaa pelimaailmassa monen pelihahmon

ryhmiä ja työskennellä yhteisesti erilaisten tehtävien parissa. Pelimaailmaan luodaan oma persoonallinen roolipelihahmo, jonka ulkonäköön ja muihin ominaisuuksiin on usein mahdollista vaikuttaa. Usein pelit rohkaisevat pelaajia ryhmätyöskentelyyn tarjoamalla haastavia pelin sisäisiä tehtäviä tai taisteluita, jotka läpäistyään pelaaja palkitaan pelin sisäisillä esineillä.

Nykyään massiivinen monen pelaajan tietokoneroolipeli on lajityyppinä erittäin suosittu. kuvassa 4 esiintyvä World of Warcraft -tietokoneroolipeli oli suurenmoinen menestys, ja sitä pelaavat edelleen miljoonat ihmiset ympäri maailmaa. Kyseisen pelin räjähtävä myyntimenestys on rohkaissut pelikehittäjiä myöhemmin julkaisemaan monipelitilan joillekin tietokoneroolipeleille, jotka oli alun perin suunniteltu yksin pelattaviksi.

Kuva 4. World of Warcraft on erittäin suosittu massiivinen monen pelaajan tietokoneroolipeli

Modernit tietokoneroolipelien variaatiot

Tietokoneroolipelien sekä yleisesti videopelien teknisen kehityksen tuloksena on syntynyt myös lajityyppejä, joita on toisinaan hankalaa kategorisoida, sillä ne sisältävät elementtejä useammasta videopelien alatyypistä.

Varsinkin modernit räiskintäpelit sekoittavat tietokoneroolipeleille tyypillisiä ominaisuuksia pelien monipelitiloissa, joita pelataan ryhmässä muiden pelaajien kanssa internetin välityksellä. Kehittäjät ovat huomanneet, että kokemuspisteiden kautta avattavat pienet lisäominaisuudet tuovat mielenkiintoa ja enemmän motivaatiota monipelaamiseen. Tämän tyyppinen menettely ei jää ainoastaan räiskintäpeleihin sovellettavaksi, vaan lähes jokaiseen videopelilajityyppiin on tavalla tai toisella sovellettu roolipelityyppisiä elementtejä jonkin pelin muodossa. Kuvassa 5 näkyy, miten Borderlands-räiskintäpelin käyttöliittymä on hyvin tietokoneroolipelien kaltainen, mutta pelattavuudeltaan peli kuitenkin painottaa ensimmäisestä persoonasta kuvattua toiminnallista ammuskelua.

Kuva 5. Borderlands-räiskintäpeli yhdistelee tietokoneroolipelin elementtejä.

3 Tietokoneroolipeleihin soveltuvat pelimoottorit ja työkalut

Pelimoottori tai pelinkehitysympäristö on aloituspiste, josta peliä yleensä lähdetään kehittämään. Oman pelimoottorin suunnittelu ja kehittäminen vie paljon resursseja ja aikaa. Suuret pelistudiot kehittävät usein oman pelimoottorinsa, mutta varsinkin itsenäisten kehittäjätiimien puolella pelimoottoriongelma ratkaistaan usein käyttämällä valmista kaupallista pelimoottoria. On myös olemassa niin sanottuja pelin kehitykseen soveltuvia työkaluja (eng. Game Creation System), jotka ovat termistä huolimatta hyvin paljon pelimoottoreiden kaltaisia.

Tämä luku käsittelee erilaisia tietokoneroolipeleihin soveltuvia kaupallisia pelimoottoreita ja työkaluja. Samalla vertaillaan niiden soveltuvuutta lopputyöhön liittyvän peliprototyypin toteuttamiseen.

3.1 RPG Maker

RPG Maker on pelikehitystyökalu, joka on suunniteltu tietokoneroolipelien luomiseen. Työkalussa on valmiiksi rakennettu karttaeditori, oma yksinkertainen skriptipohjainen ohjelmointikieli sekä taisteluedi tori. Pelien kehitys rajoittuu 2-uloitteisiin tietokoneroolipeleihin. Kaikissa RPG Maker versioissa on mukana valmiita sprite-peligrafiikoita joita on mahdollista suoraan käyttää pelin sisällön luomisessa. (7.)

Vaikka RPG Maker onkin suunnattu lähinnä tietokoneroolipelien kehittämiseen, on sillä mahdollista luoda myös seikkailupelejä tai graafisia novelleja. Ensimmäinen konsoliversio RPG Maker pelimoottorista julkaistiin vuonna 1995 Super Famicom (SNES) pelikonsolille.

RPG Makerin käyttöliittymä on hyvin yksinkertainen, mutta se on samalla työkalun suurin ongelma. Uuden projektin luomisen jälkeen työkalu luo oletuksena valmiin aloitusruudun pelille. Aloitusruutu sisältää perustoimintoja, kuten uuden pelin aloittamismahdollisuuden ja peliasetusten vaihtamistoiminnon. Peli rakennetaan erilaisista ruutuihin jaetuista kartoista, mikä on hyvin tyypillinen piirre 80- ja 90-luvun konsoleille julkaistuissa tietokoneroolipeleissä. Kuvassa 6 on nähtävissä RPG Makerin editori sekä ajettu peli.

Omakohtaiseen pelisuunnitteluun ja kehittämiseen on hyvin vähän vaihtoehtoja tarjolla, mutta siitä huolimatta RPG Makerilla kehitettyjä indie-pelejä julkaistaan vuosittain niin paljon, että retro-tyyppinen tietokoneroolipelimalli on selvästi edelleen suosiossa. Pelikehitystyökalulla on nähty kehitettävän myös joitakin suosittuja seikkailupelejä kuten vuonna 2004 julkaistu Yume Nikki, joka nousi myöhemmin internetin kautta kulttimaineeseen. RPG Maker soveltuu käytettäväksi varsinkin nuoremmille pelikehityksestä kiinnostuneille henkilöille, sillä se ei vaadi käyttäjältään pitkäaikaista harjoittelua pelattavan pelin rakentamiseen.

RPG Maker ei sovellu lopputyöhön sisältyvän peliprototyypin toteuttamiseen, sillä kyseessä on ennemminkin valmis tietokoneroolipelirunko, johon käyttäjä luo pelisisältöä. Varsinaisen pelikehitykseen RPG Maker tarjoaa hyvin vähän.

Kuva 6. RPG-Makerin käyttöliittymä on yksinkertainen ja helposti opeteltavissa.

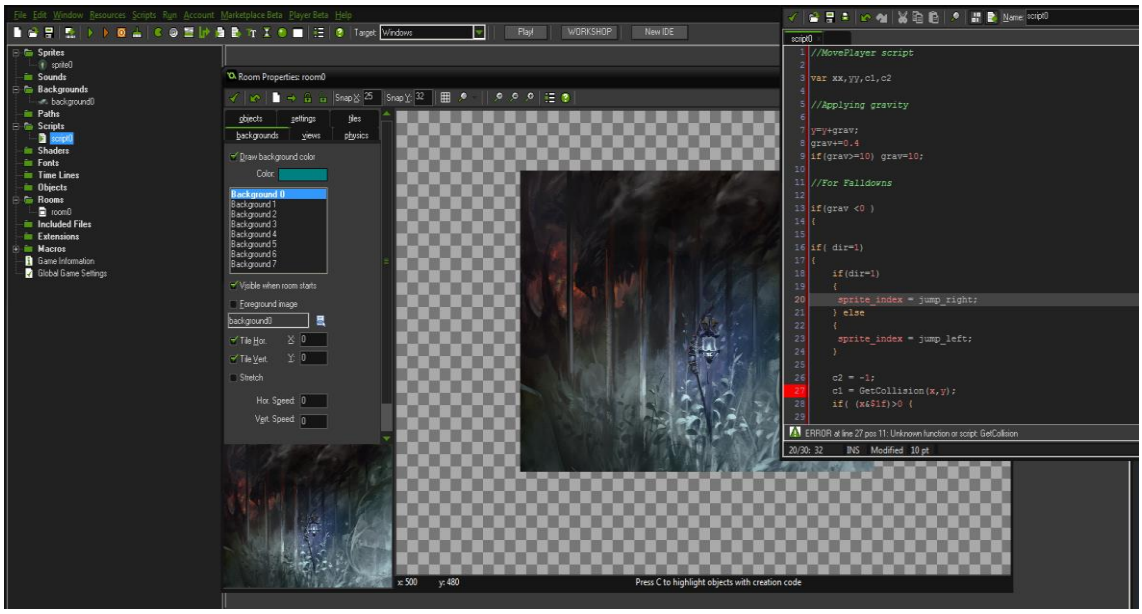
3.2 Game Maker Studio

Game Maker Studio on pelinkehitysympäristö, joka on luotu Delphi-ohjelmointikielellä. Se on suunniteltu helppoon ja nopeaan pelikehitykseen ilman monimutkaisen ohjelmointikielen opettelemista. Kehitysympäristö hyödyntää erilaisten valmiiden toimintojen raahaamista ja pudottamista peliobjektiin, jolloin luodaan haluttu toiminto tai pelielementti.

Kehitysympäristön sisään on rakennettu oma skriptaamiseen soveltuva ohjelmointikieli Game Maker Language (Lyhyesti GML). Ohjelmointikieli on tarkoitettu tukemaan valmiita peliobjekteihin upotettavia toimintoja ja se lisää lähestymistapoja eri suunnitelmien toteuttamiseen. (8.)

Pääasiallisesti Game Maker Studio käyttää kaksiulotteista grafiikkaa. Sen lisäksi tarjolla on tuki 3D-grafiikalle rajoitetusti. Kuten monessa muussakin kehitysympäristössä ja pelimoottorissa, Game Maker Studiossa on tyypillisiä pelikehitykseen liittyviä ominaisuuksia, kuten partikkeliefektit ja sisäänrakennettu mikseri.

Game Maker Studio on hyvän käyttöliittymänsä takia suosittu indie-pelien kehitysympäristö. Kehitysympäristön oletusvalikoima erilaisia valmiita pelimekaniikoita auttaa paljon vähemmän kokenutta pelikehittäjää tai harrastelijaa. Game Maker Studio soveltuu hyvin tietokoneroolipelien kehittämiseen, mutta pelisuunnittelun täytyy olla kaksiulotteista, jolloin peli käyttää oletuksena sprite-tyyppistä grafiikkaa. Kuvassa 7 näkyy Game Maker Studion perusnäkymä.



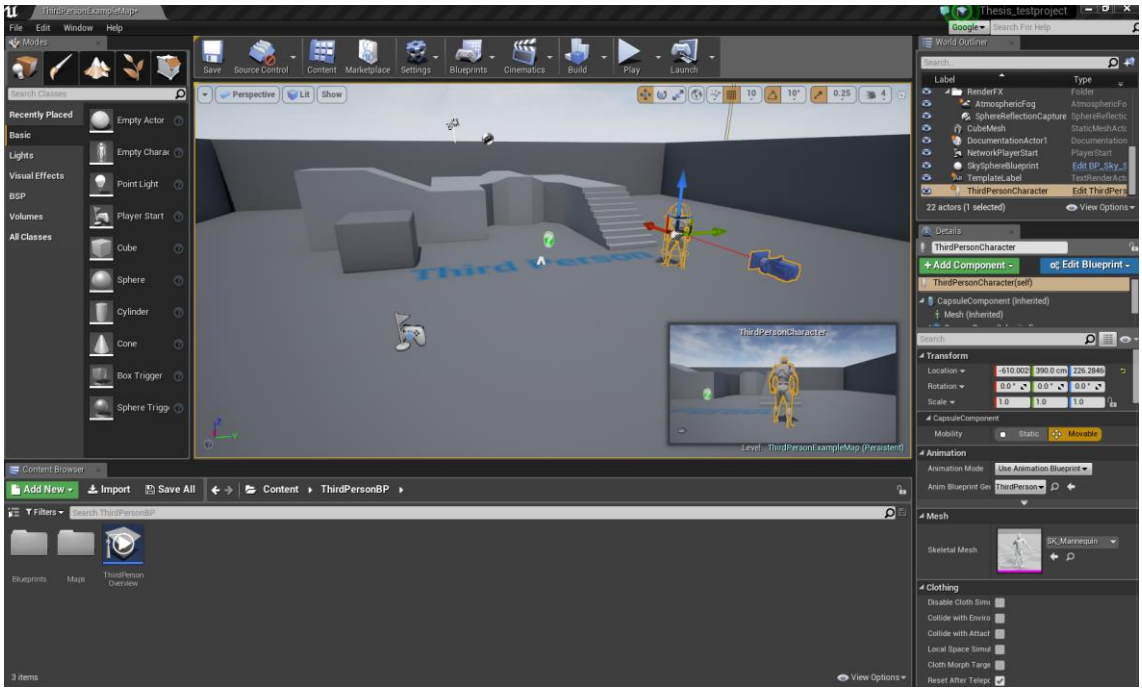
Kuva 7. Game Maker Studio sisältää 2D-pelien kehitykseen hyvät työkalut.

3.3 Unreal Engine

Unreal Engine kehitettiin alun perin Unreal-tietokonepeliä varten vuonna 1998 (9). Pelimoottori on ollut elinkaarensa aikana suosittu varsinkin ensimmäisen persoonan toiminta- ja räiskintäpeleissä, mutta sillä on myös kehitetty joitakin tietokoneroolipelejä, kuten suosittu Deus Ex -pelisarja sekä Lost Odyssey. Moottorista on julkaistu neljä eri versiota. Uusin versio on Unreal Engine 4.

Myös Unreal Enginellä on oma johdannainen UnrealScript-komentosarjakielensä jota käytetään pelin sisäisten tapahtumien luomiseen. Unreal Enginellä on mahdollista kehittää kaksi- sekä kolmiulotteisia pelejä. (9.)

Unreal Engine sisältää paljon erilaisia pelikehitykseen soveltuvia työkaluja ja toimintoja, ja sen käyttöliittymä on selkoperäinen. Kuvassa 8 voidaan nähdä, että pelimoottorin käyttöliittymä on suunniteltu lähinnä kolmiulotteisten pelien luomiseen. Pelimoottori on käyttäjälleen ilmainen ja se sisältää paljon esimerkkejä pelikehityksen tehostamista varten. Kolmiulotteisen tietokoneroolipelin kehitykseen Unreal Engine sopii hyvin.



Kuva 8. Unreal Enginen perusnäkymä kolmannen persoonan peliprojektissa.

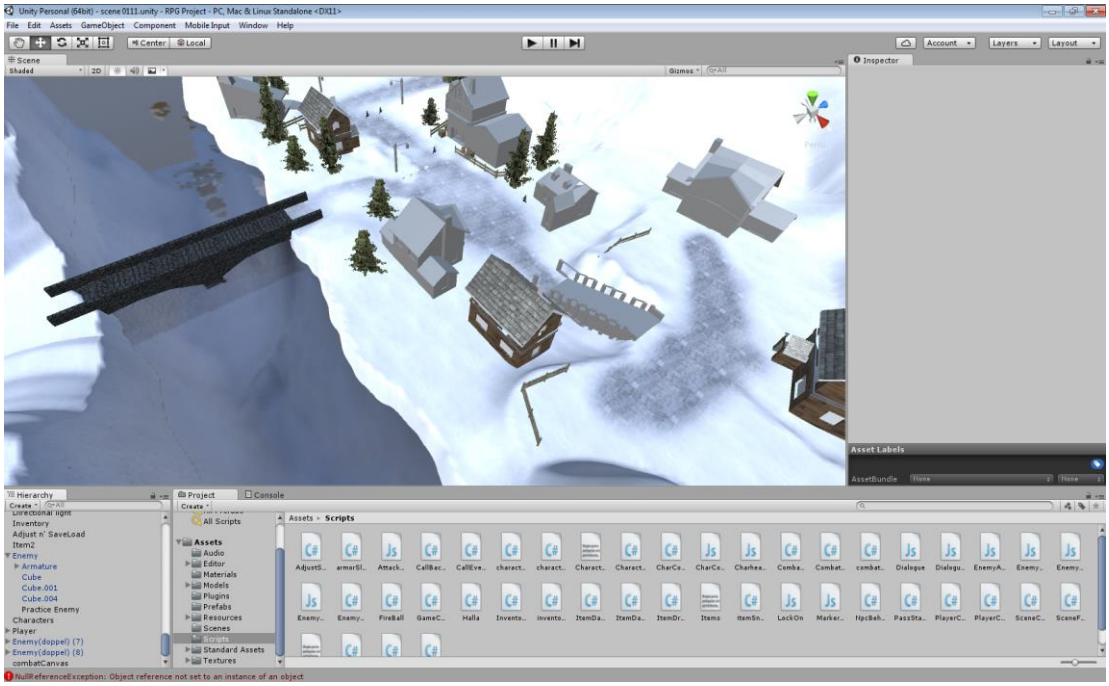
3.4 Unity

Unity on monialustainen pelimoottori, jolla on mahdollista kehittää kaksi- tai kolmiulotteisia selain-, konsoli- ja PC-pelejä. Sen kehittäjänä on toiminut vuodesta 2005 lähtien Unity Technologies. Nykyään Unity on yksi suosituimpia lisensioituja pelimoottoreita. Unitya on mahdollista käyttää usealla eri alustalla. Unity on ohjelmoitu C, C++ ja C# kielillä ja sillä on mahdollista kehittää pelejä käyttäen C#, Boo ja JavaScript-kieliä. Unitylla luota-

vien pelien rakenne koostuu peliobjekteista (GameObject), joiden sisällä on komponentteja**.** Komponentit määrittelevät peliobjektin toiminnan pelin sisällä, ja ne ovat verrattavissa ohjelmoituihin skripteihin (10;11;12).

Unityn käyttöliittymä on selkeä, ja se muistuttaakin hieman Unreal Enginen käyttöliittymää. Kuvassa 9 voidaan nähdä Unityn kehitysympäristö ja tietokoneroolipelin prototyypin avoin projekti. Uusimpien versiojulkaisujen myötä pelimoottorin mukana tulee oletuksena Microsoft Visual Studion kokeiluversio, joka saattaa vaikuttaa käyttäjäkohtaiselta pohjalta hieman ylimääräisen tuotteen tyrkyttämiseltä, sillä Visual Studio estää pääsyn ohjelmoitaviin skripteihin kokeiluajan päätyttyä. Ohjelmointipuolen toteutuksissa ongelma ei ole suuri. Visual Studion kautta avataan oletuksena kuitenkin esimerkiksi peliprojektin sisältämät 3D-mallit, joita käytetään peliobjekteina. Tämä häiritsee hieman pelikehitystä, sillä Visual Studio on suhteellisen raskas kehitysympäristö jo sellaisenaan.

Kuitenkin Unityn sisältämät selkoperäiset työkalut sekä tuki monelle ohjelmointikielelle tekevät siitä hyvin joustavan kehitysalustan mille tahansa pelille. Tietokoneroolipelikehitykseen Unity on hyvä valinta lajityyppiin katsomatta.



Kuva 9. Unity-pelimoottorin kehitysympäristö on helposti lähestyttävissä sen käyttöliittymän selkeyden vuoksi.

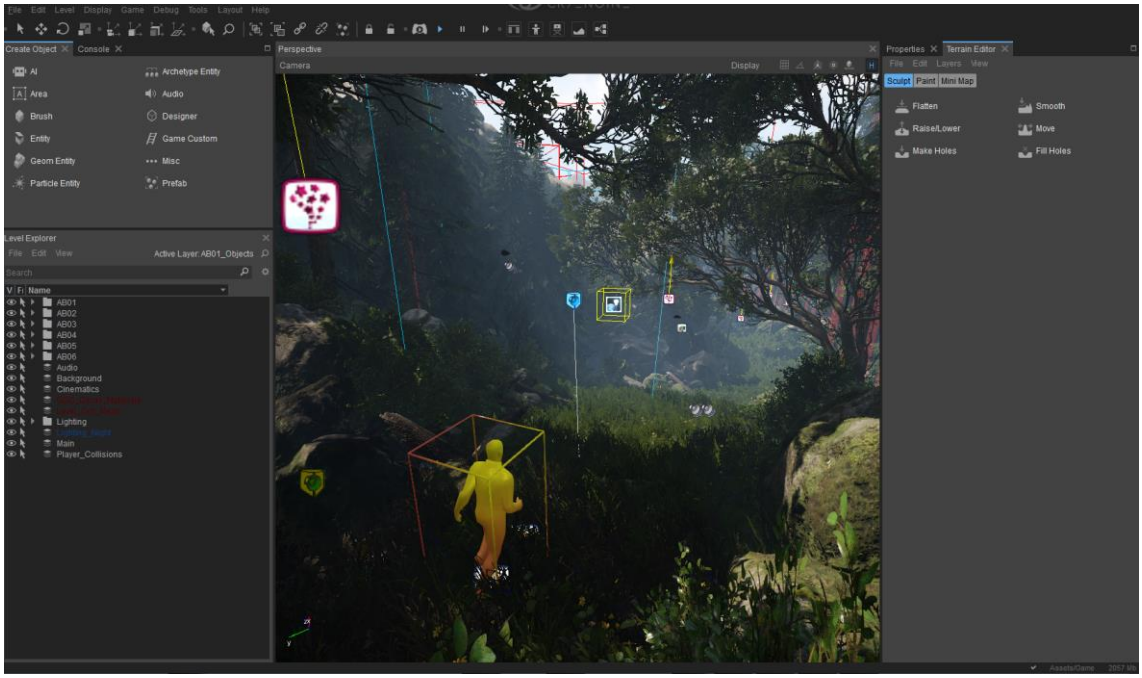
3.5 CryEngine

CryEngine on pelimoottori, jonka on kehittänyt saksalainen videopelikehittäjä Crytek. CryEnginea käytettiin ensimmäisen kerran vuonna 2004 julkaistussa Far Cry-pelissä. Nykyisin myös muut kehittäjäyrityksen Crytekin lisäksi käyttävät CryEnginea erilaisiin peliprojekteihin.

CryEnginella on mahdollista kehittää pelejä suurimmalle osalle moderneista kehitysalustoista. Pelimoottorilla on mahdollista ohjelmoida pelejä C++ ja C#-ohjelmointikielillä. Jos kehittäjällä ei ole tarkoitus kaupallistaa projektia, on CryEnginen lisenssi ilmainen. Kaupallisiin projekteihin on tarjolla eri tarpeisiin soveltuvia maksullisia käyttölisenssejä. CryEngine on suunniteltu kolmiulotteisten pelien kehitykseen. (13.)

CryEnginen kehitysympäristöstä löytyy muiden kaupallisten pelimoottorien tavoin laaja valikoima erilaisia sisäänrakennettuja ominaisuuksia ja työkaluja. Käyttöliittymäpuoli näyttää hyvin selkeältä, kuten kuvasta 10 voidaan nähdä. Sen omaksuminen vaatii kuitenkin hieman totuttelemista varsinkin, jos on aikaisemmin käyttänyt Unreal Enginea tai Unitya. CryEnginellä on siitä huolimatta erinomaisia kehityspuolen työkaluja. Esimerkiksi

pelien testaamiseen ja debuggaamiseen CryEnginestä löytyy muita kaupallisia pelimoottoreita laajemmat työkalut. Kolmiulotteisen tietokoneroolipelin kehittäminen CryEnginella olisi järkevää, jos projekti olisi suunniteltu kaupallistettavaksi jo ennen kehityksen aloittamista.



Kuva 10. CryEngine renderöi nykystandardien mukaista kaunista 3D-grafiikkaa.

4 Peliprototyypin suunnittelu

4.1 Pelin rakenne ja eteneminen

Tarkoituksena on luoda prototyyppi tietokoneroolipelille, joka noudattaa enemmän japanilaisen tietokoneroolipelin kaavaa. Peli painottaa lineaarisen tarinankerronnan mukaan etenevää pelaamista.

Pelin alussa pelaaja asettuu yhden ennalta määrätyn pelihahmon rooliin. Tämä pelihahmo toimii pelin päähenkilönä ja keskushahmona. Pelihahmoa ohjataan pelihahmolle määrättyihin kohteisiin, joissa suoritetaan pelin tarinaan sisältyviä erilaisia tehtäviä. Peli yhdistelee tarinankerrontaa, kamppailutilanteita sekä seikkailupelin elementtejä, esimerkiksi pulmanratkontaa.

Pelin edetessä vastaan tulee uusia pelihahmoja jotka liittyvät päähenkilöhahmon tueksi, jolloin muodostuu ryhmä pelihahmoja. Kaikilla pelihahmoilla on persoonalliset kyvyt ja vahvuudet. Pelihahmot täydentävät pelin tarinaa. Yleensä hahmoja kontrolloidaan vain taistelutilanteissa. Pelin edetessä pelihahmojen yhteismäärä ylittää ryhmäkoon maksimimäärän joka on kolme hahmoa. Tästä syystä pelaajan on valittava parhaaksi näkemänsä hahmot ryhmäänsä taistelutilanteita varten. Ryhmään kuuluvia pelihahmoja voi vaihdella pelialueiden tallennuspisteillä.

Pelimaailma

Peli jakautuu aluekokonaisuuksiin, joiden sisällä pelaaja suorittaa pelin tarinan edellyttämät tehtävät ennen kuin on mahdollista edetä tarinan mukana seuraavalle pelialueelle. Alueiden sisällä pelaaja voi toimia usein omien valintojen mukaan, mutta pelin tarinaan liittyvien osuuksien aikana peli asettaa pelaajalle tilanteet ja rajoitteet joiden sisällä on toimittava.

Yleisesti tietokoneroolipelit ovat rikkaita dialogin ja tarinankerronnan suhteen. Pelit sisältävät NPC-hahmoja joiden kanssa käydyt keskustelut täydentävät pelimaailmaa ja sen tarinaa. Ne vievät usein myös pelihahmoa eteenpäin erilaisten vihjeiden ja neuvojen avulla.

Peliin tulee vastaavanlaisesti samankaltainen asetelma jossa NPC-hahmojen kanssa käytävillä keskusteluilla on pelin etenemisen kannalta suuri rooli. Dialogit ovat usein ennalta määriteltyjä, mutta ne sisältävät välillä valintakysymyksiä, joihin vastataan valitsemalla jokin ennalta määritetty vastaus kahden tai kolmen vaihtoehdon joukosta. Vastausvaihtoehdot ovat usein kyllä- tai ei-tyyppisiä vastauksia, joilla on tarkoitus elävöittää dialogin sisältöä antamalla pelaajalle hieman lisää interaktiomahdollisuuksia.

4.2 Käyttöliittymä

Pelin käyttöliittymän kaksi pääelementtiä ovat taistelutila sekä normaali seikkailutila. Kaikki tietokoneroolipelille tyypilliset kamppailutilanteet käydään taistelutilassa. Seikkailutila on pelitila, jossa pelaaja on aina, kun kamppailutilanne ei ole käynnissä. Pelitilat poikkeavat toisistaan ohjattavuuden puolesta: taistelutila on täysin valikoilla ohjautuva ja seikkailutila keskittyy yhden pelihahmon vapaaseen ohjaukseen.

Seikkailutila

Seikkailutila on pelin käyttöliittymään kuuluva osa, jossa pelaaja viettää suurimman osan peliajasta. Tässä tilassa pelihahmo on vapaasti liikuteltavissa ja pelaaja pystyy olemaan vuorovaikutuksessa pelialueen interaktiivisiin peliobjekteihin, kuten NPC-hahmoihin ja pelin tallennuspisteisiin.

Aina kun pelaaja on seikkailutilassa, hänellä on pääsy pelin inventaariovalikkoon, jossa on mahdollista käyttää varastossa olevia esineitä, tarkastaa pelihahmojen ominaisuuksia ja vaihtaa pelihahmojen varusteita.

Seikkailutilan aikana tapahtuvat pelin tarinankulkuun liittyvät välianimaatiot, joiden aikana pelaaja menettää hetkellisesti hahmonsa ohjattavuuden. Välianimaatiot ovat dialogipainotteisia tilanteita, joissa peli on ohjelmoitu liikuttamaan tarvittavia peliobjekteja elokuvatyyppisen välikohtauksen luomiseksi.

Taistelutila

Pelin seikkailutilan kautta siirrytään taistelutilaan aina kun vihollinen on tarpeeksi lähellä pelaajaa. Taistelutilassa pelihahmoa ei ole mahdollista enää kontrolloida vapaasti, vaan ohjattavuus siirtyy vuoropohjaiseen, valikoiden kautta käytävään kamppailusessioon.

Tässä tilassa jokaiselle pelihahmolle ja viholliselle on asetettu oma ajastin. Kun ajastin saavuttaa tietyn aikamääreen, on pelihahmolla mahdollisuus tehdä erilaisia toimintoja, jotka riippuvat pelaajasta. Pelihahmo voi hyökätä vihollista vastaan tai käyttää mahdollisesti esineitä tai muita pelihahmolle asetettuja kykyjä. Ajastin on riippuvainen pelihahmon nopeusattribuutista joka vaihtelee pelihahmojen välillä. Attribuutit tarkoittavat kokemuspisteistä saatavia taitopisteitä. Esimerkiksi metsästäjä-hahmoluokka on oletuksena nopeampi toimimaan kuin velho, mutta nopeusattribuuttia nostamalla velho voi päästä metsästäjän tasolle nopeudessa.

Taistelutila päättyy, kun kaikki viholliset on kukistettu. Jos taas pelaajan kaikki kamppailevat pelihahmot kukistetaan, peli päättyy ja pelaajan on aloitettava peli viimeksi tallentamastaan pelitilanteesta.

4.3 Hahmonkehitys

Hahmonkehitys on pelin etenemisen kannalta tärkein elementti. Jokaisella pelihahmolla on oma kokemustaso, jota on mahdollista kasvattaa kokemuspisteiden avulla. Kokemuspisteitä pelihahmot saavat voitettuaan taistelutilanteita, ja pisteet määräytyvät vihollisten määrän ja niille asetetun kokemuspistemäärän perusteella. Tarinan edetessä pelaaja kohtaa entistä vaarallisempia vihollisia, jotka antavat enemmän kokemuspisteitä mutta ovat myös isompi uhka kamppailutilanteissa.

Aina kokemustason kasvaessa pelihahmo voimistuu. Pelihahmo saa enemmän sille asetettuja taito- ja voimapisteitä, jolloin hahmo pärjää paremmin kamppailutilanteissa. Aina saavuttaessaan tietyn kokemustason, pelihahmo oppii myös uusia taitoja, joita voi käyttää taistelu- ja seikkailutilassa riippuen, minkälainen taito on kyseessä.

Jokaisella pelihahmolla on omat varusteet, jotka vaikuttavat myös hahmojen ominaisuuksiin. Esimerkiksi jos pelaaja varustaa pelihahmon aseella, joka on sen antamien

hyökkäyspisteiden kannalta parempi kuin vanha ase, nousevat pelihahmon hyökkäyspisteet. Parempi haarniska toimii samalla tavalla, mutta antaa lisää suojapisteitä hyökkäyspisteiden sijaan.

4.4 Pelin koukuttavuus ja houkuttelevuus

Tietokoneroolipeleillä on lajityypistä riippuen useita ominaisuuksia, jotka pitävät pelaajan otteessaan loppuun asti. Länsimaiset tietokoneroolipelit tarjoavat enemmän vapautta, jota osa pelaajista haluaa yli kaiken. Japanilaiset pelit taas painottavat lineaarisuutta ja tarinankerrontaa, joista osa pelaajakunnasta on kiinnostunut. Molemmilla koulukunnilla ja sen lajityypeillä on kuitenkin usein samoja ominaisuuksia, jotka saattavat jopa koukuttaa pelaajaa jatkamaan pelin pelaamista.

Progressiivisuus

Pelin edetessä pelihahmot voimistuvat ja oppivat uusia taitoja. Tämä pelielementti tarjoaa uusia ominaisuuksia kamppailutilanteisiin ja tuo saavutuksen tunteen, kun pelihahmot kehittyvät ja pystyvät kukistamaan entistä voimakkaampia vastustajia. Myös pelihahmojen varustaminen erilaisilla aseilla ja haarniskoilla tuo lisää monipuolisuutta hahmojen muokkaukseen.

Kun peli etenee tarinan mukana, saa pelaaja käyttöönsä enemmän pelihahmoja joita voi käyttää kamppailutilanteissa. Uudet persoonalliset, kehittyvät pelihahmot pitävät kamppailutilanteet ja tarinan vaihtelevampina ja vähentävät samojen asioiden toistoa. Isompi määrä pelihahmoja auttaa pelaajaa löytämään helpommin persoonan, johon samaistua. (17.)

Pelissä on oma sisäinen rahayksikkö, jota useimmiten saa palkinnoksi voitettuaan kamppailutilanteen vihollisia vastaan. Rahalla on mahdollista ostaa parempia varusteita, jotka vaikuttavat suoraan pelihahmojen taistelukykyyn hyökkäys- ja suojapisteiden kautta.

Tämä tekijä motivoi pelaajaa osallistumaan useammin kamppailutilanteisiin, vaikka siihen ei välttämättä olisikaan tarvetta.

Roolipelaaminen ja tutkiskelu

Kun kyseessä on japanilaisen tietokoneroolipelin tyyppinen kokonaisuus, niin suoraa roolipelaamista ja avointa seikkailua on tarjolla rajoitetusti. Kuitenkin pelaajalla on paljon vapautta tutkiskelun ja myös päätösten suhteen. Monen asian kanssa ei ole pakko olla suoraan vuorovaikutuksessa, mikä tuo vapauden tunnetta.

Pelimaailma sisältää erilaisia kätkettyjä esineitä, mikä rohkaisee tutkiskelemaan enemmän. Seikkailevan pelaajan palkitseminen aarteella motivoi pelaajaa tutkimaan enemmän. Se myös edesauttaa eläytymään pelimaailmaan.

Pelin läpäiseminen

Jos pelaaja kokee pelin mielenkiintoiseksi, haluaa hän yleensä läpäistä pelin mikä tuo myös eräänlaisen saavutuksen tunteen. Pelaajaa saattaa mahdollisesti kiinnostaa miten pelin sisältämä tarina päättyy tai miten pitkälle pelin sisältämiä hahmoja on mahdollista kehittää. (18.)

5 Peliprototyypin toteutus

Peliprototyyppiin liittyvät 3D-mallit ja ympäristö on luotu peliprototyyppiin nopeasti, eikä niiden visuaaliseen ulkoasuun ole panostettu prototyypin kehityksen aikana. Kaikki 3D-mallit on luotu Blender-mallinnusohjelmalla. Osa malleista sisältää tekstuureja, jotka ovat peräisin avoimista lähteistä, jotka on tarkoitettu vapaaseen käyttöön. Peliympäristöön kuuluu animoituja 3D-malleja, joista mainitaan esimerkiksi ohjaukseen liittyvissä ratkai-

suissa, mutta ne eivät olennaisesti kuulu peliprototyypin pelimekaniikkaan. Pelin testaamisen kannalta on kuitenkin olennaista, että jonkinlaista sisältöä löytyy pelistä edes vähän. Grafiikkaan ja visuaaliseen ilmeeseen keskitytään myöhemmin, mutta lopputyöhön kuuluva prototyyppi ei sitä tee.

5.1 Ohjaus

Pelissä ohjataan yhtä tai useampaa pelihahmoa riippuen siitä mikä pelitila pelisession sisällä on käynnissä. Pelin ohjattavuus vaihtelee eri pelitilojen välillä. Taistelutilassa käytetty valikkopohjainen ohjaus on hyvin tyypillinen ratkaisu Japanilaisen koulukunnan tietokoneroolipeleille. Kameran ohjaus on erikseen määritelty, sillä se on vapaasti käytettävissä pelitilasta riippumatta.

Konsoliohjaintuki

Pelin prototyyppi tukee Playstation 3-konsolin ohjainta ja kehitys sekä testaus on toteutettu pääosin käyttäen konsoliohjainta. Pelin käyttöliittymä on ajateltu enemmän konsolipohjaiselle pelille sopivaksi ja konsoliohjaimella pelaaminen tuntuukin hieman luontevammalta. Suunnitelmana on luoda mahdollisimman hyvä ohjattavuus myös näppäimistö ja hiiri -yhdistelmälle.

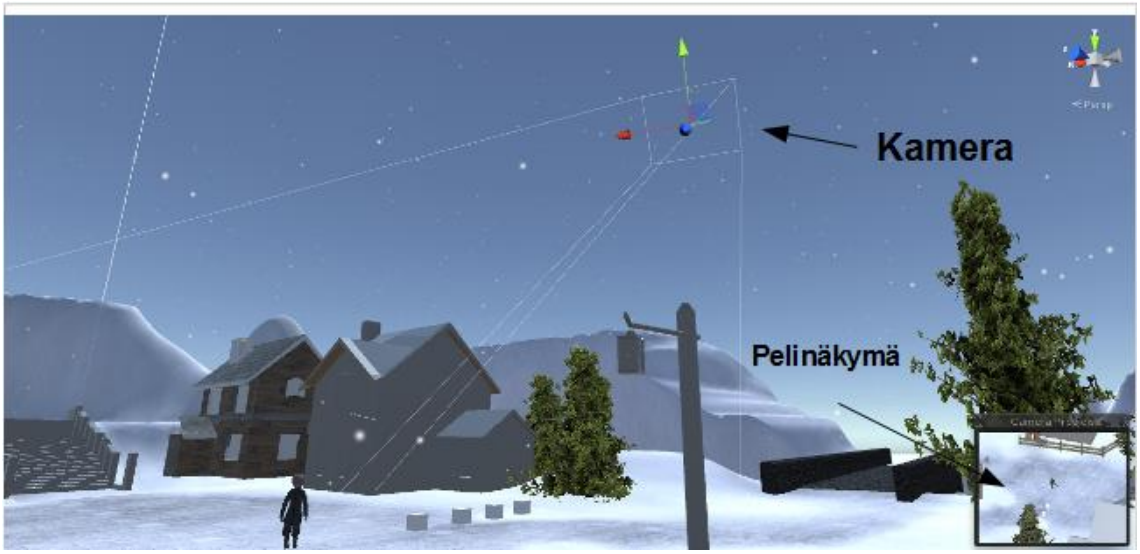
Unity tunnistaa automaattisesti suurimman osan uusimmista konsoliohjaimista. Tästä syystä painikkeiden kartoittaminen Unity-ympäristössä onnistuu yleensä nopeasti.

Kameran ohjaus

Pelin kamera seuraa seikkailutilan aikana ohjattavaa pelihahmoa. Pelaaja kontrolloi tarpeidensa mukaan kameran kuvakulmaa, joka säilyttää aina saman kiintopisteen jonka ympäri se pyörii. Kamera on ainut kontrolloitava peliobjekti, jonka ohjattavuus ei muutu pelin edetessä ollenkaan. Kameraa on mahdollista pyörittää pelihahmon ympäri molempiin suuntiin. Kamerakulman säätely tuo ylimääräistä interaktiivisuutta pelaajalle.

Kameran ohjaus tapahtuu sen omalla PlayerCamera–skriptillä, joka on sijoitettu suoraan Kamera-peliobjektiin. Skriptissä kameralle luodaan uusi transform-komponentti, joka

määritellään pelihahmoksi. Tämän toimenpiteen avulla pelihahmosta saadaan kiintopiste, jonka ympärillä kamera pyörii akselin tavoin ja jota kamera myös seuraa pelihahmon liikkuessa. Kuvassa 12 voidaan nähdä kameran sijoitus kolmannen persoonan kuvakulman saavuttamiseksi.



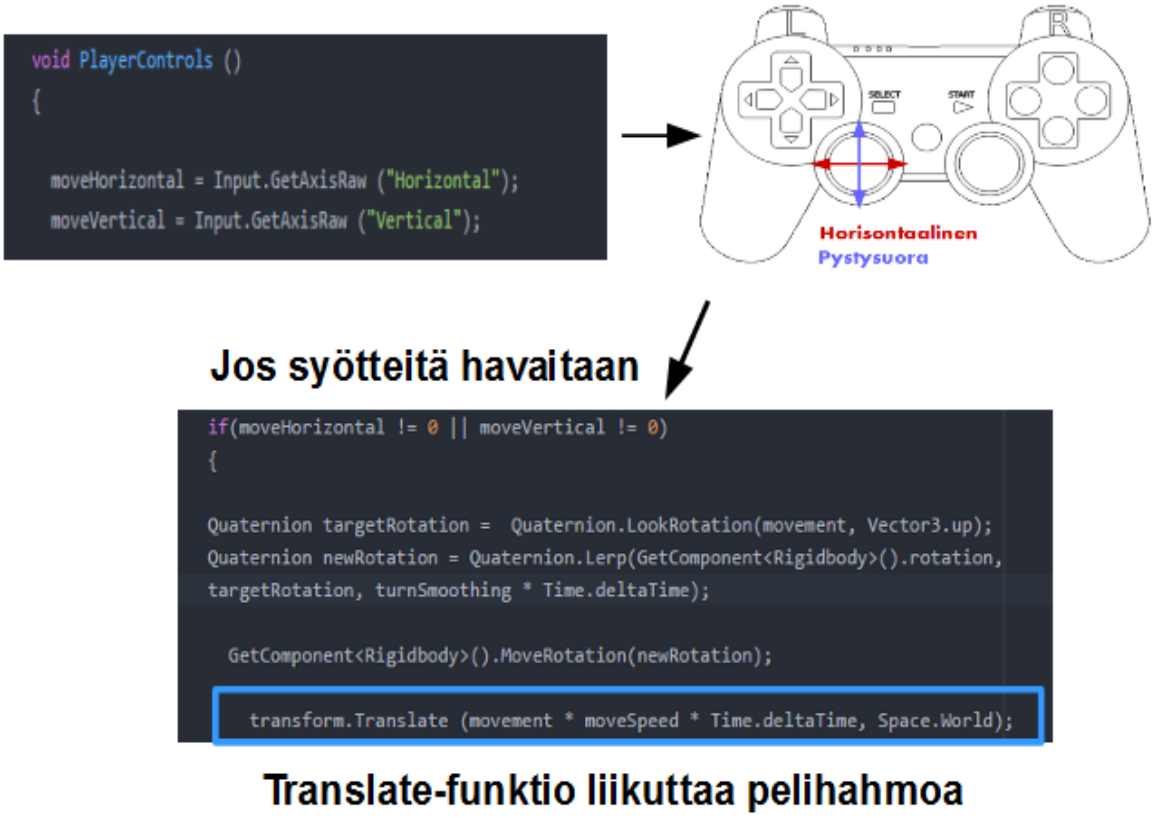
Kuva 11. Pelin kamera on sijoitettu näyttämään pelitilanteet kolmannesta persoonasta.

Ohjaus seikkailutilan aikana

Tässä pelitilassa ohjataan yhtä pelihahmoa sille annetun CharController-skriptin avulla. Skripti tarkistaa ensin tekeekö pelaaja pystysuoraan tai horisontaaliseen liikkumiseen liittyviä syötteitä ohjauspainikkeiden kautta. Kun syöte havaitaan, pelihahmoa liikutetaan haluttuun suuntaan Translate-funktion avulla, joka käyttää oikean liikkumissuunnan määrittelemiseen pelihahmon suuntavektoreita. Kyseisen funktion sisällä määritellään myös pelihahmon liikkumisnopeus, jossa nopeutta määrittelee pelin sisäisen kellon nopeus kerrottuna halutulla kerroinarvolla.

CharController-skripti vastaa myös pelihahmon liikkumisen yhteyteen kuuluvista animaatioista. Pelihahmolle asetetun normaalin liikkumisvauhdin lisäksi pelihahmo kykenee juoksemaan ja hyppäämään seikkailutilan aikana. Juokseminen tapahtuu painamalla juoksuun määrättyä painiketta normaalin pelihahmon liikuttamisen aikana, jolloin CharController-skripti nostaa Translate-funktion liikkumisnopeuden kerrointa ja vaihtaa normaalin kävelyanimaation juoksuanimaatioon. Hyppääminen käyttää hyväksi pelihahmon

sisään määriteltyä Rigidbody-komponenttia, joka asettaa painovoiman peliobjektille. Rigidbody-komponentilla on valmiiksi määritelty AddForce-funktio, jolla on mahdollista määritellä suuntavektoreihin hetkellinen tarvittava liikkumisvoima. Tässä tapauksessa pelimaailmassa ylöspäin osoittava vektori on kerrottu hyppyvoimaa määrittelevällä muuttujalla, jolloin hyppynappia painamalla pelihahmo nousee ilmaan ja tekee hyppyyn liittyvän animaation. Pelihahmolle kuuluu myös Rigidbody-komponenttiin kuuluva peliobjektin painosta vastaava muuttuja, jolla voidaan säädellä kuinka nopeasti pelihahmo palaa takaisin maahan. Kuvassa 13 on selvennetty pelihahmon kontrollointiin liittyvää toimintaperiaatetta käyttäen Playstation 3-ohjainta.



Kuva 12. Pelihahmon ohjaamisen toimintaperiaate Playstation-3 ohjaimella.

Skripti pysyy aktiivisena myös esimerkiksi taistelutilan aikana, mutta normaaliin ohjattavuuteen liittyvät toiminnot on estetty niin kauan, kunnes peli palaa takaisin seikkailutilaan. Normaalin seikkailutilaan kuuluvan toiminnallisuuden aiheuttaa PlayerControls-funktio CharController-skriptin sisällä, jota kutsutaan skriptin Update-funktion kautta. Update-funktiota kutsutaan jokaisen ruudunpäivityksen aikana, kun peli on käynnissä.

Tämä mahdollistaa kontrollointiin liittyvän responsiivisuuden. Funktion kutsu estetään tarvittaessa boolean-arvolla toisen skriptin kautta.

Seikkailutilaan kuuluu myös inventaariovalikko, jonka ohjattavuus sekä toiminta on käsitelty kokonaisuudessaan kohdassa 5.3. Kuvassa 14 voidaan nähdä seikkailutilassa kontrolloitava pelihahmo kolmannesta persoonasta.

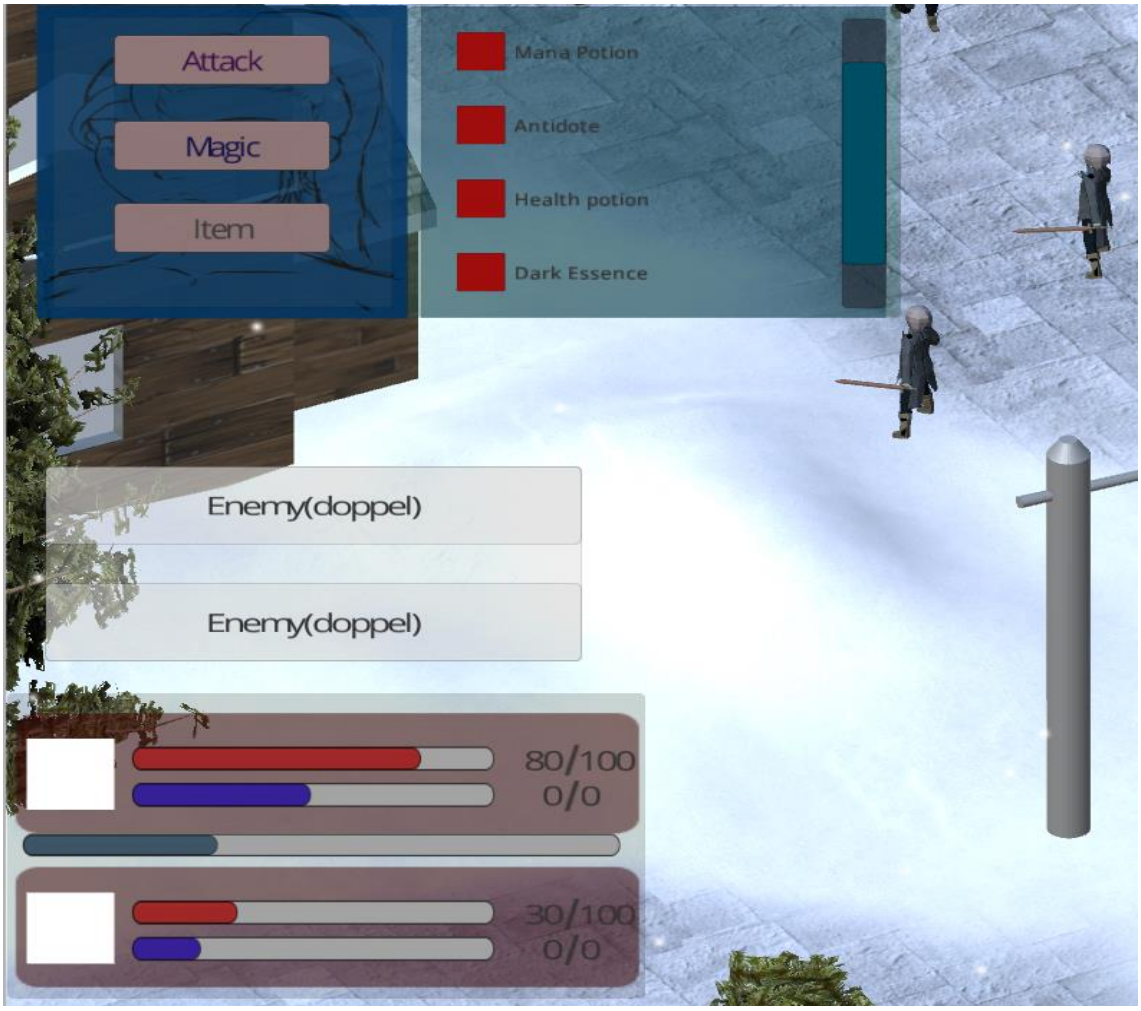
Kuva 13. Seikkailutila on taistelutilan tavoin kuvattu kolmannesta persoonasta.

Ohjaus taistelutilan aikana

Taistelutilan aikana pelihahmoa ei voi ohjata enää vapaasti, vaan kaikki liikuttelu käydään taistelutilaan kuuluvilla valikoilla, jotka sisältävät pelihahmoille tarkoitettuja komentoja. Taistelut ovat vuoropohjaista kamppailua vihollisia vastaan, joten hahmojen interaktiot vihollisten kanssa ovat käytännössä vain näennäistä liikkumista, joka värittää taistelukokonaisuuksia.

Kun pelihahmo on tarpeeksi lähellä vihollisia, kutsutaan pelihahmon engageCombat-funktiota, joka estää seikkailutilaan liittyvän vapaan liikkumisen ja inventaariovalikon käytön. Taisteluvalikoiden ohjaaminen toimii Unityn Event Controller -komponentilla, joka tunnistaa käyttöliittymävalikoiden rakenteet ja lisää niihin ohjattavuuden System Input

Handler -komponentilla. Kuvassa 15 näkyy taistelutilan käyttöliittymään kuuluvia valikoita. Peliprototyypissä ei varsinaisesti ole panostettu käyttöliittymän visuaaliseen ulkonäköön, vaan hyvän toiminnallisuuden saavuttamiseen.



Kuva 14. Taistelutilan käyttöliittymä ja sen sisältämät valikot. Items-painike, joka on valittu, näyttää esinelistan.

5.2 JSON-tiedostomuoto, paikalliset tietokannat ja pelin inventaariovalikko

JSON-tiedostomuoto ja paikalliset tietokannat

Tietokoneroolipelit sisältävät paljon tietoa pelatun pelisession sisällä. Jokaisella pelihahmolla on esimerkiksi paljon erilaisia ominaisuuksia, jotka vaikuttavat pelissä käytyihin

taistelutilanteisiin. Myös erilaisia esineitä ja taitoja on yleensä suuri määrä, ja pelin kehitysvaiheessa niihin liittyvä suunnittelu on tuskin viety loppuun asti.

Varsinkin esineitä ja pelihahmojen taitoja lisätään peliin kehitysvaiheessa testimielessä. Tämän takia tietokantatyyppinen toteutustapa, johon on mahdollista lisätä nopeasti uutta sisältöä, on hyvin suositeltava ratkaisu, kun puhutaan pelidatan käsittelystä.

Peliprototyyppi käyttää JSON-tiedostomuotoa isoihin esine- ja taitolistauksiin. JSON on yksinkertainen avoimen standardin tiedostomuoto tiedonvälitykseen (14). JSON-tiedostojen avulla luodaan oliopohjaiset pelitietokannat, joista on helppo hakea tarvittava tieto esimerkiksi pelaajan löytämälle esineelle. JSON-tukeen liittyvä ohjelmointikirjasto on lisätty projektiin erikseen, sillä Unity ei tue sitä oletuksena. Kuvassa 11 näkyy JSON-tiedoston perusrakenne, joka sisältää tiedot yhdelle pelin sisäiselle esineelle. Jokaisella esineellä on vastaavanlainen rakenne vuoron perään JSON-tiedoston sisällä.

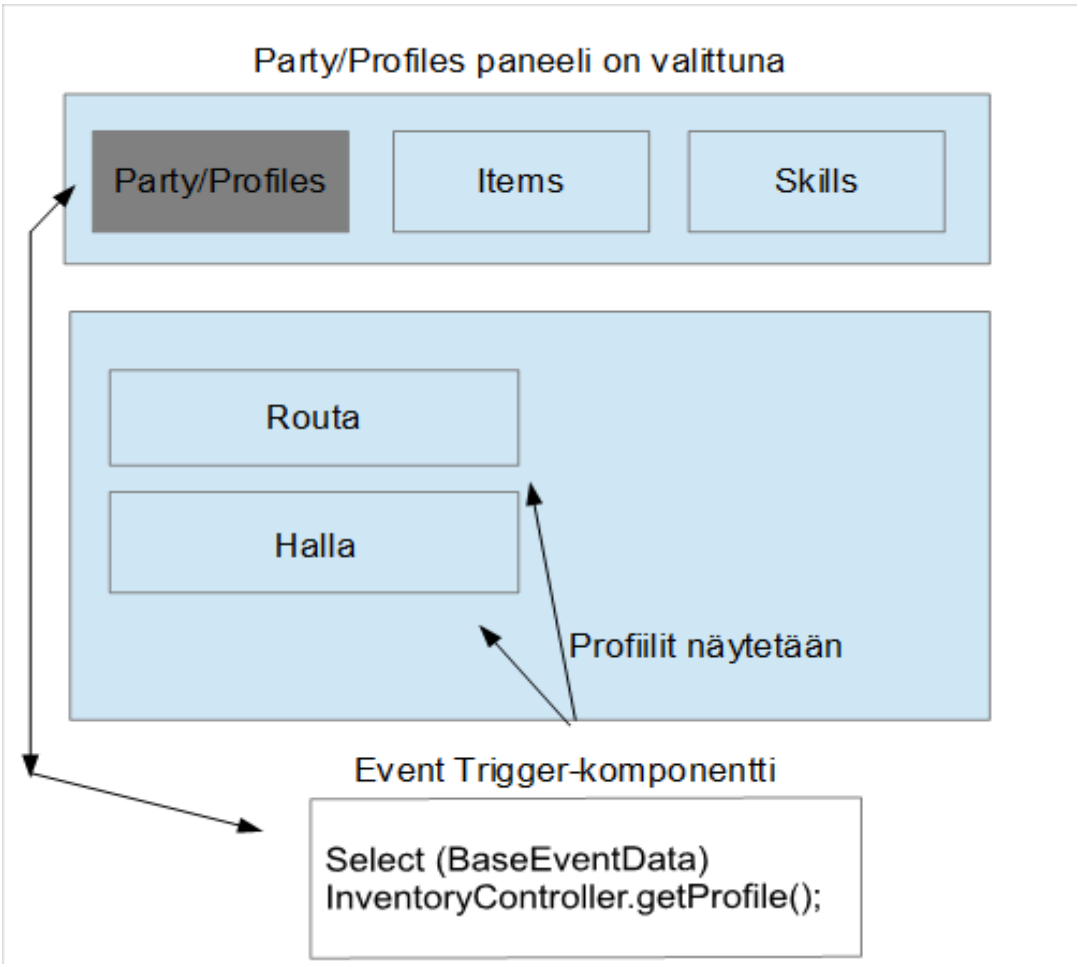
```
[
  {
    "id": 0,
    "title":"Mana Potion",
    "amount": 1,
    "value": 0,
    "stats": {
      "attack":0,
      "defense":0,
      "health": 0,
      "mana": 35
    },
    "description": "Restores Mana",
    "stackable": true,
    "equippable": false,
    "weapon": false,
    "armor": false,
    "usable": true,
    "slug": "manapotion"
  }
]
```

Kuva 15. JSON-tiedoston perusrakenne.

Inventaariovalikko

Pelin inventaariovalikko on toteutettu Unityn omilla UI-peliobjekteilla ja sen hallittavuutta on täydennetty ohjelmoiduilla skripteillä. Inventaariovalikko sisältää kaikki pelihahmoihin liittyvät tiedot, kuten ansaitut kokemuspisteet ja käytössä olevat varusteet. Valikko listaa myös nykyisen pelisession varastoidut esineet, jotka ovat käytettävissä aina taistelutilan tai välikohtauksen ulkopuolella. Jos pelihahmoilla on taistelutilan ulkopuolella käytettävissä olevia taitoja, esimerkiksi parannustaikoja, on myös niitä mahdollista käyttää inventaariovalikon kautta. Näin ollen valikko toimii eräänlaisena hallintapaneelina pelihahmoille seikkailutilan aikana.

Unity käyttää UI-elementeille pohjana Canvas-peliobjektia, johon kaikki inventaariovalikkoon liittyvät paneelit on sijoitettu. Kun valikko avataan, näkymä avautuu ensimmäiseen alavalikkoon, joka sisältää nykyisessä ryhmässä olevat pelihahmot. Kaikki muut paneelit on piilotettu InventoryController-skriptin avulla. Skripti toimii aktiivisten paneelien Event Trigger -komponentin kanssa, joka kutsuu tarvittavan skriptin funktiota ja näyttää tai piilottaa tarvittavat alavalikot. Kuvassa 16 on selvennetty Event Trigger-komponentin toimintaa ja havainnollistettu miten se näyttää ja tarvittaessa kätkee tarvittavia alavalikoita.



Kuva 16. Inventaariovalikon alapaneelit tuodaan näkyviin Event Trigger -komponentin avulla.

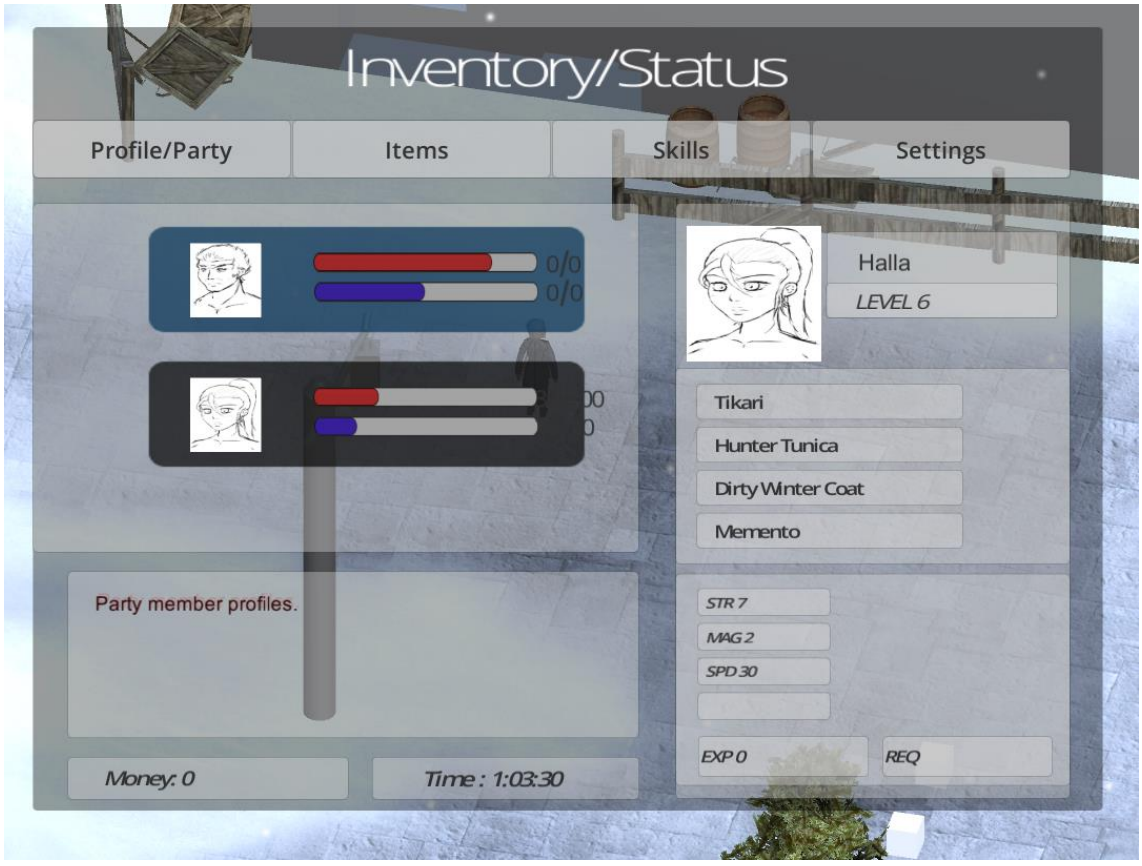
Paneelien välinen ohjaus toimii myös UI-komponenttien avulla. Kun Canvaksen sisään asetetaan jollekkin UI-peliobjektille Event Controller-komponentti, alustetaan kokonaisuuteen kuuluvat paneelit kontrollointia varten. Jokaiseen paneeliin, johon halutaan ohjaustoiminto, on lisättävä myös Standalone Input Module-komponentti, joka vastaanottaa peliobjektille saapuvat navigaatioon liittyvät syötteet.

Inventaariovalikon sisältämät esineet, taidot ja pelihahmoprofiilit luodaan pelisession käynnistyessä. Sisällön luontiin käytetään joko edellisen pelisession tallennustiedostoa tai uuden pelin aloitukseen liittyvää pelin alustusfunktiota. Jokaiselle pelidataa sisältävälle valikolle on luotu oma datan käsittelykomponentti, joka on sijoitettu näkymättömään peliobjektiin pelikohtauksen sisällä. Toimintaperiaatteeltaan kaikki datan esittämiseen ja

luontiin liittyvät komponentit ovat hyvin samankaltaisia. Alavalikot ovat toisistaan poikkeavia, joten saman pelikomponentin käyttö ei tässä tapauksessa onnistu.

Party/Profile-valikko

Ensimmäinen alavalikko on Party/Profile-valikko, johon on listattu ryhmässä olevat pelihahmot. Pelihahmojen painikkeista on mahdollista tarkastella jokaisen hahmon ominaisuuksia ja sen hetkistä varustusta. Listattua pelihahmoa painamalla siirrytään tarkastelemaan hahmon varusteita. Hahmoilla on neljä eri paikkaa erilaisille päälle puettaville varusteille. Valittua pelihahmon päällä olevaa varustetta painamalla on mahdollista vaihtaa varustusta, mikäli esinelistalta löytyy mahdollisia vaihdettavia varusteita. Kun pelaaja valitsee varusteen vaihtoa varten, lisää valitun esineen sisään sijoitettu SlotCall-skripti sen hahmon varustelistalle. Samalla pelisession esinelistalta poistetaan valittu, päälle puettu varuste ja esinelistalle myös lisätään se varuste, joka oli aikaisemmin hahmon päälle puettuna. Varusteet listataan valintaa varten varustepaikkakohtaisesti. Esimerkiksi jos pelaaja haluaa vaihtaa pelihahmon käytössä olevan aseen, näyttää Items-valikko ainoastaan käyttöön otettavat aseet valintatilanteen aikana. Kuvassa 17 nähdään inventaariovalikko ja pelihahmoprofiilit.



Kuva 17. Inventaariovalikon perusnäkymä. Halla-pelihahmon profiili on valittuna.

Pelihahmojen olennaisimmat tiedot ladataan info-paneeliin, joka on aina näkyvissä. Tietojen esittämisestä vastaa InventoryController-skriptiin kuuluva showCharacter-funktio, joka näyttää sen pelihahmon tiedot, jota pelaaja osoittaa Party-valikossa. Pelihahmokohtaiset tiedot funktio noutaa characters-taulukosta, joka sisältää pelihahmojen tiedot sisältävät character-oliot.

Items-valikko

Toinen alavalikko listaa pelaajan käytettävissä olevat esineet. Pelisession sisältämien esineiden tiedot on tallennettu items-taulukkomuuttujaan, joka sisältää item-olioita. Olioihin on säilötty kaikki esineisiin liittyvä olennainen tieto, kuten esineen nimi, seliteteksti ja mahdolliset esineen kautta saatavat hyödyt, esimerkiksi elinpisteet.

Pelisession käynnistyessä Inventory-skripti luo aina uudet peliobjektit esineille items-taulukon perusteella. Jos pelaaja saa uuden esineen, luodaan pelisession aikana taulukkoon uusi esineen tiedot sisältämä olio ja Inventory-skripti luo esineelle uuden peliobjektin Items-paneelin sisään.

Vaikka Items-alavalikko listaakin kaikki varastoidut esineet, on sen kautta mahdollista alustavasti käyttää vain pelihahmojen attribuuttien edistämiseen käytettäviä esineitä, kuten elinpisteiden palauttamiseen tarkoitettuja esineitä. Jokainen esine-peliobjekti sisältää SlotCall-skripti, joka validoi onko esine käytettävissä vai ei. Validaatio tehdään kutsumalla esineen ItemData-skriptiä, johon on esineen luontivaiheessa säilötty tieto esineen käyttöominaisuuksista. Jos esineen voi käyttää suoraan pelihahmoa varten, siirtää InventoryController-skripti näkymän Party-alavalikkoon, jolloin on mahdollista valita pelihahmo, johon haluaa esineen käyttää.

Skills-valikko

Kolmas alavalikko sisältää pelihahmojen henkilökohtaiset taidot, joita on mahdollista käyttää taistelutilan ulkopuolella. Listattavat taidot ovat parantavia, esimerkiksi haavoittuneen pelihahmon elinpisteitä palauttavia tukitoimintoja. Pelihahmojen taitoihin liittyvä

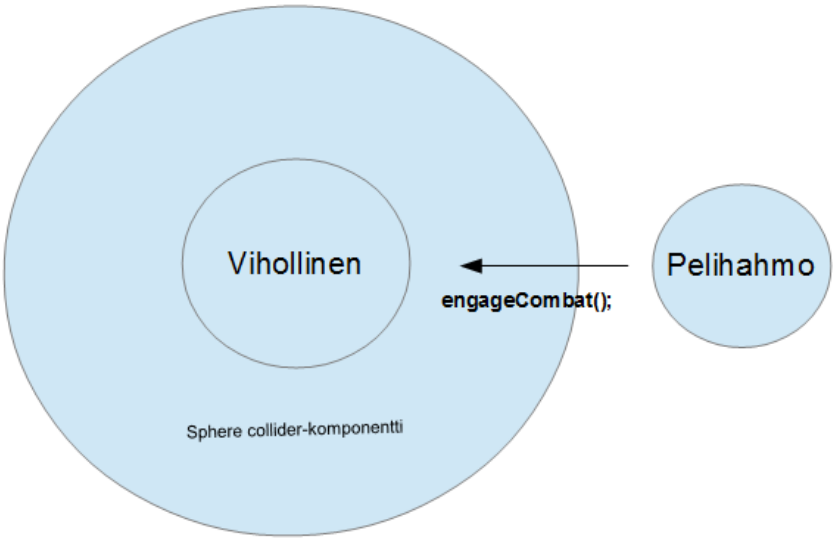
tietojen luominen sekä hakeminen on toteutettu noudattaen esineisiin liittyvää toteutustapaa. Taidoilla on oma JSON-tiedostosta muodostuva tietokantansa, jonka avulla rakennetaan jokaiselle taidolle oma olio. Olio sisältää tarvittavat tiedot taidon toimintaan liittyen taistelutilassa ja sen ulkopuolella. Boolean-arvo validoi taitojen listauksen Skills-valikkoon.

Eri taidot ovat käytettävissä ainoastaan kun pelihahmojen ryhmään kuuluu kyseisen taidon omaava pelihahmo.

5.3 Taistelujärjestelmä

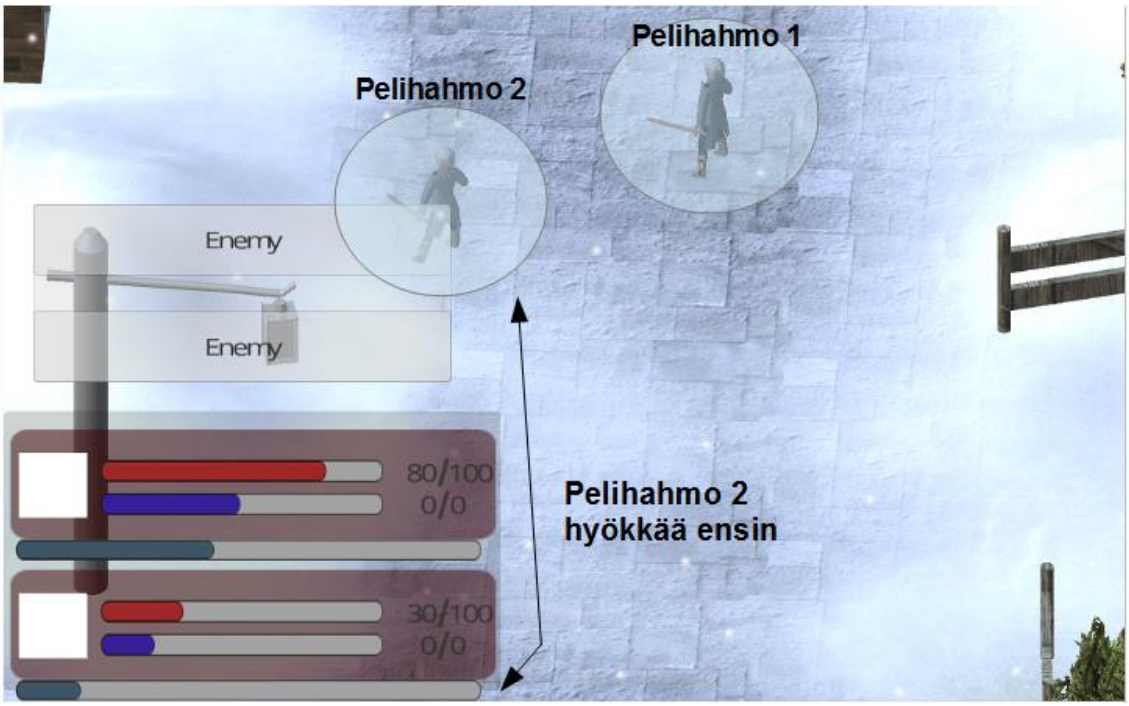
Taistelujärjestelmä vastaa pelin taistelutilan toiminnasta. Taistelutila on vuoropohjainen pelitila, jossa pelaajan ryhmässä olevat pelihahmot ottavat yhteen erilaisten vihollisten kanssa.

Taistelutila käynnistyy, kun pelaaja ohjaa pelihahmonsa tarpeeksi lähelle vihollista. Jokaisella vihollisella on oma Sphere Collider -komponentti, joka muodostaa pallon muotoisen tunnistusalueen pelaajan kontrolloimaa pelihahmoa varten. Kuvassa 18 on selvennetty Sphere Collider -komponentin ja pelaajan kontrolloiman pelihahmon välistä vuorovaikutusta. Vihollisten lukumäärä vaihtelee yhdestä neljään, ja taistelun alkaessa ne ryhmittyvät pelialueelle aina saman etäisyyden päähän pelihahmoista. Taistelun alkaessa pelaaja menettää suoran ohjattavuuden pelihahmoonsa, joka jää paikalleen odottamaan taistelun etenemistä. Samalla kaikki pelaajan ryhmässä olevat muut pelihahmot aktivoidaan ja ryhmitetään sen pelihahmon viereen, jota pelaaja ohjasi seikkailutilan aikana.



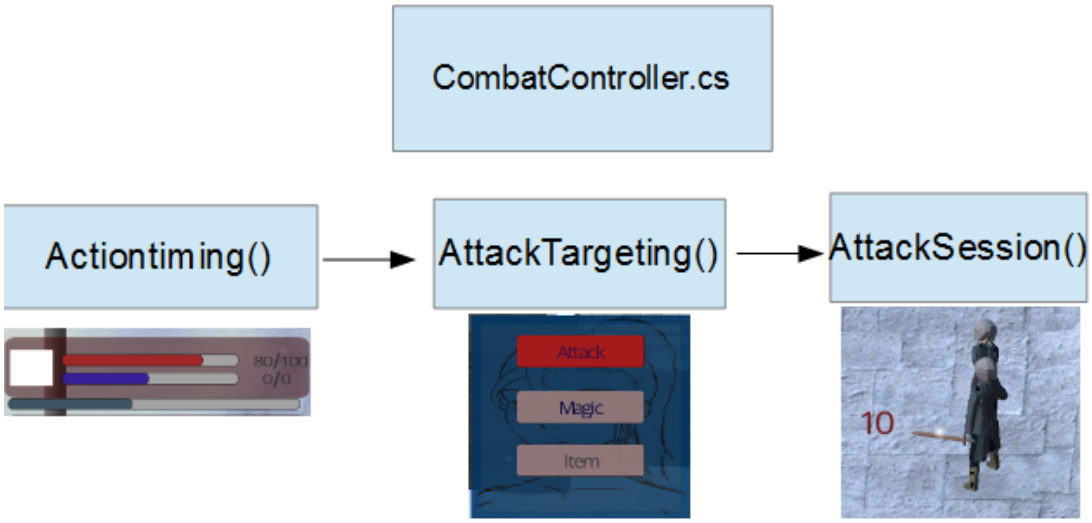
Kuva 18. Taistelutilanteet alkavat pelaajan liikkuessa tarpeeksi lähelle vihollisen tunnistusaluetta.

Heti taistelutilan aktivoituessa käynnistyy jokaiselle viholliselle ja pelihahmolle asetettu toimintavuoroa mittaava ajastin. Kun ajastin tyhjenee, on pelihahmolla tai vihollisella mahdollisuus tehdä erilaisia toimintoja, kuten hyökätä, käyttää opittuja taitoja tai esineitä. Se taistelun osanottaja, jonka ajastin täyttyy ensimmäisenä, on oikeutettu toimimaan ennen muita osanottajia. Ajastimet oikeastaan järjestävät toimintajärjestelmän taistelun osanottajille riippuen hahmon nopeusattribuutista. Kuvassa 19 voidaan nähdä esimerkki kahden pelihahmon hyökkäysjärjestyksestä. Oletuksena pelaaja näkee vain oman hahmoryhmänsä ajastimet, mutta pelihahmoihin liittyvillä taidoilla on mahdollista myös nähdä vastapuolen ajastimet, jolloin taktikointiin tulisi hieman enemmän mahdollisuuksia. Kun toimintavuoroa mittaavat ajastimet perustuvat pelihahmon tai vihollisen nopeuteen, voisi pelissä olla myös nopeutta lisääviä taitoja tai esineitä, jotka 66vaikuttaisivat vahvasti joihinkin taistelutilanteisiin.



Kuva 19. Käynnissä oleva taistelutilanne ja vuoroja indikoivat ajastimet.

Jokaisen pelihahmon vuorolla näkyviin tulee taisteluvalikko, josta pelaaja valitsee parhaaksi näkemänsä toiminnon siinä tilanteessa. Valikko on toteutettu käyttäen Unityn UI-peliobjekteja ja niihin liittyviä komponentteja, jotka toimivat yhdessä CombatController-skriptin kanssa. Riippuen pelihahmojen ja vihollisten lukumäärästä, CombatController-skripti listaa taistelun osanottajat ja kutsuu niille määriteltyjä funktioita, joilla saadaan aikaan tarvittavat hyökkäysanimaatiot ja elinpisteiden vähentäminen. Hyökättäessä vihollista kohti CombatController-skriptin tekee myös pelihahmon yksinkertaiset liikkeet vihollista kohti. Lähitaistelutilanteessa pelihahmo siirretään vihollisen lähelle, tehdään tarvittavat hyökkäykseen liittyvät animaatiot ja elinpisteiden vähennys viholliselta, jonka jälkeen hahmo siirretään takaisin alkuperäiseen paikkaan johon se oli sijoitettu taistelun alkaessa. Kuvassa 20 voidaan nähdä viholliseen kohdistetun hyökkäyksen sisältämät funktiot CombatController-skriptin sisällä.



Kuva 20. CombatController-skriptin toimintaperiaate, kun pelaaja hyökkää pelihahmolla.

Normaali hyökkäys on passiivisin toiminto taistelujärjestelmän pelimekaniikoiden kannalta. Se perustuu pelihahmon vahvuuspisteisiin sekä käytössä olevaan varustukseen, jolloin tehdyt vahinkopisteet on helppo laskea ja vähentää vihollisen elinpisteistä. Erilaiset taidot listataan pelihahmojen henkilökohtaisista tiedoista, jotka on määritelty erillisessä JSON-tiedostossa. Taistelutilanteissa on mahdollista valita vain niitä esineitä, joita voi käyttää suoraan parantamaan pelihahmon elinpisteitä tai muita taistelutilanteeseen liittyviä attribuutteja.

Useimmin käytetyt toimintavaihtoehdot taisteluvalikon kautta liittyvät hyökkäämiseen tai taitojen käyttöön, koska usein paras ratkaisu on eliminoida viholliset mahdollisimman nopeasti. Kuitenkin monen pelihahmon ryhmä pystyy helpommin esimerkiksi tukemaan pahasti haavoittuneita pelihahmoja uhraamalla vuoronsa parannustaidon tai esineen käyttöön.

5.4 Pelihahmojen kokemuspisteet ja kehittyminen

Pelihahmoihin liittyvät kokemuspisteet ja niiden kautta saavutettu kokemustaso on pelin etenemisen kannalta tärkeimpiä tietoja, ja ne tallennetaan pelisessioon seuraavaa pelikertaa varten. Pelihahmojen kehitys vaikuttaa ainoastaan taistelutilanteisiin, mikä on yksi pelin pääelementeistä. Taisteluiden kautta voimakkaammaksi kehittynyt pelihahmo pärjää paremmin taisteluissa, joten peli on suunniteltu sisältämään taistelutilanteita runsaasti ja pelin edetessä aluekohtaiset viholliset on suunniteltu usein vastaamaan pelihahmojen kokemustasoa ja lukumäärää.

Pelissä eteneminen riippuu pitkälti pelihahmoille saavutetuista kokemustasoista, sillä vastaan tulee jatkuvasti voimakkaampia ja haastavampia vihollisia, jolloin pelihahmojen on myös kehityttävä taisteluvoittojen saavuttamiseksi.

Jokaisen voitetun taistelun lopussa pelihahmot palkitaan kokemuspisteillä. Kokemuspisteet määräytyvät alustavasti vastustajan voimakkuuden mukaan. Pisteet lasketaan ja lisätään jokaisen pelihahmon omaan Character-olioon, jotka on listattu taulukkomuuttujaan. Laskennasta vastaa CombatController-skripti, joka kutsuu aina vihollisen kukistamisen jälkeen sille määriteltyä funktiota, joka palauttaa määritellyn kokemuspistemäärän ennen kuin vihollisen peliobjekti tuhotaan. Kun saavutetaan tietty kokemuspistemäärä, pelihahmon kokemustaso nousee yhdellä kokemustasoyksiköllä, jolloin hahmo saa lisää elinpisteitä ja muita attribuutteja. Seuraavaan kokemustasoon tarvittava määrä on aina ensimmäinen vaadittu kokemustaso kerrottuna kahdella mikä kerrotaan nykyisellä kokemustasolla. Näin ollen vaaditut kokemuspisterajat kasvavat lineaarisesti aina korkeammiksi kun uusi kokemustaso saavutetaan.

Pelihahmoille on määritelty alustavasti kolme eri attribuuttia, jotka vaikuttavat hahmon suorituskykyyn taisteluissa: voimapisteet, nopeuspisteet sekä taikapisteet. Voimapisteet nostavat pelihahmon tekemiä vahinkopisteitä ja mahdollisesti vaikuttavat pelihahmojen kykyyn käyttää tietynlaisia varusteita. Nopeuspisteet määrittelevät, kuinka nopeasti peli-

hahmo kykenee toimimaan taistelutilanteissa. Se vaikuttaa myös hahmon todennäköisyyteen väistää hyökkäys. Taikapisteet vaikuttavat alustavasti pelihahmojen kaikkiin taitoihin, vaikka nimi viittaakin suoraan taikoihin. Pisteet vaikuttavat siihen, kuinka paljon vahinkopisteitä taidot ja taiat tekevät taistelutilanteissa.

5.5 Dialogijärjestelmä ja pelimaailman vuorovaikutus

Dialogijärjestelmä

Dialogijärjestelmä on ominaisuus, joka näyttää peliin kuuluvat ennalta määritellyt dialogitekstit pelaajalle. Dialogijärjestelmä on yksinkertaisesta rakenteestaan huolimatta roolipelielementti, joka tuo lisäeloa pelimaailmaan NPC-hahmojen kanssa käytyjen keskustelujen kautta.

Dialogijärjestelmä pääosin laukaisee erilaisia keskusteluja NPC-hahmojen kanssa. Suurimalla osalla NPC-hahmoista on näkymätön tunnistusalue, joka määrittelee milloin on mahdollista suorittaa hahmolle määritelty keskustelu. Kun pelihahmo on tunnistusalueen sisällä, on keskustelu mahdollista käydä toimintapainiketta painamalla.

Jokaisessa keskustelussa NPC -hahmolla on etukäteen määritelty lauseista muodostuva keskustelukokonaisuus. Lauseet tuodaan näkyviin dialogipaneeliin, joka tulee esiin keskustelun alkaessa. Kuvassa 21 voidaan nähdä dialogipaneeli, sen sisältämä dialogilause ja NPC-hahmon nimi paneelin vasemmassa yläkulmassa. Lauseet tulostetaan kirjain kerrallaan dialogipaneeliin, jolloin saadaan rullaavan tekstin efekti, jota on yleisesti käytetty erilaisissa dialogipainotteisissa peleissä.

Kuva 21. Dialogipaneeli ja sen sisältämä lause. Dialogit rakentuvat useasta kuvassa näkyvästä lauseesta.

Pelimaailman vuorovaikutus

Pelimekaniikat toimivat vuorovaikutuksessa pelaajan kanssa. Kun pelaaja suorittaa toiminnon, peli muuttaa tilaansa siihen määriteltyjen pelimekaniikkojen pohjalta. Pelimekaniikkoja ei kuitenkaan ole tarkkaan määritelty ja niiden luonteesta on erilaisia näkemyksiä. (15.)

Edellä mainittu lainaus tuo todennäköisesti hieman selkeyttä luvun sisältöön ja siihen mitä se tarkoittaa tietokoneroolipeliprototyypin pelattavuuden ja toteuttamisen suhteen.

Pelimaailman keskeisin vuorovaikutus pelaajaan syntyy pelihahmojen kehityksen kautta. Pelihahmojen kehitys näkyy taistelutilassa, jossa pelihahmot saattavat pärjätä paremmin korkeamman kokemustason tai pelihahmolle hankitun paremman varustuksen avulla. Kokemustason kautta on myös mahdollista oppia uusia taitoja, joita pelaaja voi käyttää taisteluiden aikana. Tämän kaltaiset täydentävät pelimekaniikat säilyttävät mielenkiinnon taisteluiden parissa ja viestivät samalla pelaajalle, että taistelut ovat hyödyllisiä pelihahmojen kehittymisen takia.

5.6 Pelin tallennus, lataaminen ja testaus

Pelin tallentaminen ja lataaminen

Pelissä ei ole useimpien nykypelien tavoin automaattista tallennusjärjestelmää. Tämän kaltainen menettely on tyypillistä japanilaisen koulukunnan tietokoneroolipeleille, joissa pelin eteneminen ei ole jaettu lyhyisiin pelitaso-kokonaisuuksiin. Tämän sijaan isompi aluekokonaisuus tutkittavine sijainteineen ja tehtävineen tarjoaa pelaajalle paluupisteen seikkailuista, josta seuraavalla pelikerralla on hyvä jatkaa peliä. Tyypillisiä tämän kaltaisia pisteitä ovat esimerkiksi kylät ja kaupungit, joissa peliä jatkaessa on mahdollista suorittaa pelinkululle tyypillisiä toimenpiteitä, kuten uusien tehtävien vastaanottamista ja tarvittavien esineiden hankintaa pelin esinekaupoista.

Pelin tallentamisesta vastaa siis pelaaja itse, mutta tallentaminen vaatii tallennuspisteen. Prototyypissä on yksi tallennuspiste, joka sijaitsee peliin sijoitetussa kylässä. Ohjaamalla pelihahmon tallennuksesta vastaavan peliobjektin läheisyyteen ja painamalla toimintanappia, aukeaa pelaajalle mahdollisuus tallentaa pelisessio.

Pelille on luotu GameControl-skripti, joka vastaa pelitietojen tallentamisesta ja lataamisesta. Pelidatalle luodaan esine- ja hahmotietokannan tyyppisesti PlayerData-olio, johon viedään tallennusta varten kaikki tarvittavat tiedot. Olion alustamisen jälkeen käytetään pelidatan luontiin ja tallentamiseen sille luotua funktiota GameControl-skriptin sisällä. Kuvassa 21 voidaan nähdä tallentamisesta vastaava funktio ja sen rakenne. Samanlainen funktio vastaa pelitietojen lataamisesta, kun pelaaja haluaa taas jatkaa pelaamista seuraavalla pelikerralla.

```
33    public void Save()
34    {
35      BinaryFormatter bf = new BinaryFormatter();
36      FileStream file = File.Create(Application.persistentDataPath + "/playerInfo.dat");
37      Debug.Log("Saving Game.");
38      PlayerData data = new PlayerData();
39      data.health = health;
40
41      data.experience = experience;
42
43      bf.Serialize(file, data);
44      file.Close();
45
46    }
```

Kuva 22. Pelin tallennukseen liittyvä Save-funktio.

Testaus

Peliprototyyppiin liittyvä testaaminen on tehty peliprototyypin kehityksen yhteydessä. Uudet ominaisuudet sekä niiden toteutuksiin tehdyt muutokset lähes poikkeuksetta testataan nopeasti käyttäen sopivia testipeliobjekteja.

Testauksen hyödyllisin työkalu on Unityn Debug -luokka, joka sisältää ohjelmointiin liittyviä, virheiden etsimiseen ja poistamiseen soveltuvia kutsuttavia funktiota. Käytetyin Debug-luokan funktio on Log, joka yksinkertaisesti kirjaa halutun muuttujan arvot Unityn sisäiseen konsoliin.

6 Prototyypin kokonaisuus ja jatkokehitys

6.1 Tulokset

Lopputyön tarkoituksena oli oppia pelikehitystä, siihen liittyvää suunnittelua ja ohjelmointia Unity-pelimoottorille. Tässä onnistuttiin suunnitelmien mukaisesti. Aikaisemmin Unitylle oli tehty pelikehitystä käyttäen pääasiassa pelimoottorin kehitysympäristöön kuuluvaa JavaScript-painotteista Unityscript-kieltä. Peliprototyyppi toteutettiin kokonaan käyttäen C#-kieltä. Kyseisen kielen käyttöön tuli paljon kehitystä lopputyöprosessin aikana,

ja motivaatio ohjelmointitaitojen edelleen kehittämiseen on suuri. Tietokoneroolipelikehitys sisältää paljon erilaista pelidatan hallintaa ja sen esittämiseen liittyvää suunnittelua, joka oli prototyypin ohjelmointiin liittyvissä seikoissa tärkein meriitti.

Unity-pelimoottorin käyttöön liittyvää osaamista tuli paljon lisää prototyypin eri kehitysvaiheiden aikana. Varsinkin Unityn kehitysympäristön omiin käyttöliittymäpohjaisiin kehitystyökaluihin tuli laaja katsaus, ja niiden soveltaminen jatkossa onnistuu hyvin.

Peliprototyyppi vastasi suunnitelman mukaista kokonaiskuvaa, mutta pelin tallennukseen ja pelisession uudelleenlataamiseen liittyvää testausta olisi voinut olla enemmän lopputyön aikana. Niihin liittyvät seikat jäävät pelin jatkokehityspuolelle. Peliprototyyppiin liittyvistä pelimekaniikoista parhaiten toteutuivat taistelujärjestelmään ja inventaariovalikkoon liittyvät suunnittelu- ja toteutusratkaisut. Kyseiset pelimekaniikat veivät myös eniten toteutukseen käytettyä aikaa ja antoivat enemmän haastetta verrattuna muihin mekaniikoihin. Näiden kahden peliprototyyppiominaisuuden kehittämiseen meni myös muihin ominaisuuksiin verrattuna enemmän aikaa, sillä niiden parissa on otettava huomioon suurin osa tietokoneroolipelin sisältämästä pelidatasta, kuten pelihahmojen kokemuspisteet ja muut peliin vaikuttavat pelihahmokohtaiset attribuutit.

6.2 Analyysi

Tietokoneroolipelin toteutus prototyyppinä lopputyötä varten on hyvin laaja aihealue. Se ei kuitenkaan tullut yllätyksenä aihetta valittaessa, vaan tarkoituksena oli nimenomaan tarttua kokonaisuuteen, joka olisi aikaisempia pieniä peliprojekteja isompi. Prototyypin kehityksen aikana keskityttiin ainoastaan pelimekaniikoiden suunnitteluun ja toteuttamiseen. Prototyypin tekeminen vaati kuitenkin hyvin paljon työtä, ja vastaavanlaisen prototyypin kehittäminen lopputyötarpeisiin soveltuu paremmin kahdelle henkilölle.

6.3 Tulevaisuus ja jatkokehitys

Toteutettu peliprototyyppi viedään lopputyön jälkeen jatkokehitykseen, jota on tarkoitus tehdä harrasteprojektina. Pelimekaanikoiden kannalta pelin peruselementit alkavat olemaan kunnossa, mutta lopputyöprototyyppiin ei sisällytetty kuin pelattavuuden kannalta tärkeimmät ominaisuudet.

Pelin jatkokehityksessä tehdään ensin toteutukset kaikille tarvittaville pelin keskuselementeille, jotka puuttuvat prototyypistä. Tällaisia ominaisuuksia ovat esimerkiksi pelin sisäisen valuutan ja kaupankäynnin toteuttaminen pelialueilla. Suunnitteilla on toteuttaa myös taistelujärjestelmään lisää kamppailutilanteita syventäviä ominaisuuksia. Kun peliprototyypin runko on valmis puuttuvien ominaisuuksien kannalta, aletaan työskennellä varsinaisen pelisisällön parissa. Sisältö tarkoittaa esimerkiksi pelialueiden vihollishahmojen suunnittelua ja mallintamista.

Pelin jatkokehitykselle ei ole arvioitu tarkkaa aikaa, mutta rehellisesti voidaan puhua vuosista projektin keston suhteen. Pelikehitystä on mahdollista nopeuttaa luomalla pelinkehitysryhmä, jossa eri pelinkehitykseen liittyviä tehtäviä on mahdollista jakaa ja työskennellä sitä kautta tehostetusti.

7 Yhteenveto

Insinöörityössä annettiin lyhyt kuvaus tietokoneroolipeleistä ja niihin soveltuvista pelimoottoreista. Työn pääpaino oli suunnitella ja toteuttaa tietokoneroolipeliin liittyviä pelattavuuden kannalta olennaisia pelimekaniikoita, jotka tekevät videopelistä tietokoneroolipelin.

Tietokoneroolipelit ovat videopelien alatyyppinä hyvin laaja käsite. Lopputyössä käytiin läpi tähän alatyyppiin kuuluvat olennaisimmat asiat, jolloin lukija saa hyvän peruskäsityksen tietokoneroolipeleistä. Tietokoneroolipeleille on useita eri lajityyppejä, joista käytiin läpi suosituimmat. Modernit videopelit yhdistelevät hyvin usein tietokoneroolipeleille olennaisia roolipelielementtejä, sillä niillä saadaan lisättyä varsinkin monipelaamiseen enemmän mielenkiintoa. Nykyään on tarjolla tietokoneroolipelien kehittämiseen lukuisia eri kaupallisia pelimoottoreita, joiden tarjoamat työkalut ovat laadukkaita ja usein helppokäyttöisiä. Pelisuunnitelmasta ja pelimekaniikoiden laajuudesta riippuen tarjolla on lähes jokaiselle jotain.

Peliprototyypin suunnitelma vastaa pitkälti toteutettua peliprototyyppiä. Peli vastaa japanilaisen koulukunnan tietokoneroolipeliä pelattavuudeltaan, ja se sekoittaa vanhaa ja modernia tietokoneroolipelisuunnittelua. Tietokoneroolipeleillä on niiden pelisuunnitteluun liittyviä ominaisuuksia, jotka saavat pelaajan kiinnostumaan ja palaamaan peliin takaisin.

Toteutettava peliprototyyppi tehtiin Unity3D-pelimoottorilla käyttäen C#-ohjelmointikieltä. Peliprototyypin kolmiulotteiset mallit oli toteutettu ennen pelimekaniikoiden suunnittelua ja ne palvelivat lähinnä testiympäristön värittämiseen soveltuvana materiaalina. Kaikki tarvittavat pelin paikalliset tietokannat toteutettiin käyttämällä JSON-tiedostomuotoa ja siihen liittyvää kirjastoa. Pelimekaanisiin toteutuksiin meni hieman odotettua enemmän aikaa. Eniten aikaa meni inventaariovalikon ja taistelujärjestelmän suunnitteluun ja toteuttamiseen, jotka olivat myös peliprototyypin haastavimmat kehitysosiot.

Tietokoneroolipelin prototyyppi oli onnistunut, mutta varsinkin pelidatan tallentamiseen ja lataamiseen sekä käyttöliittymäpuolen ratkaisuihin olisi voinut käyttää enemmän aikaa pelitestauksen puolella. Prototyypin toteuttaminen oli kehittävä prosessi, joka toi lisää kokemusta pelikehitykseen ja C#-ohjelmointiin.

Lähteet

1 The Adventure/RPG Computer Game Museum Of Finland

< http://www.springbringer.com/web/fantasya/> 3.2.2016. Luettu 22.3.2016

2 The History of Computer Role-Playing Games Part 1: The Early Years (1980-1983). Verkkojulkaisu

<http://www.gamasutra.com/view/feature/3623/the_history_of_computer_.php> 23.2.2007. Luettu 10.1.2016

3 Kawaisa!: A Naïve Glace at Western and Eastern RPGs. Verkkojulkaisu

<http://armchairarcade.com/neo/node/733> Luettu 12.1.2016

4 Hack and Slash: Whas Makes a Good Action RPG? Verkkojulkaisu

<http://www.1up.com/do/blogEntry?bId=9030743> 18.5.2010. Luettu 20.1.2016

5 Tactical Role-Playing Game. Verkkojulkaisu

<http://www.worldlibrary.org/articles/tactical_role-playing_game> 2016. Luettu 20.1.2016

6 Massively Multiplayer Online Game. Verkkotietojärjestelmä

<https://www.techopedia.com/definition/27054/massively-multiplayer-online-game-mmog> 2016.Luettu 20.1.2016

7 RPG Maker - Faqs

<https://www.rpgmakerweb.com/support/products/faqs> 2016. Luettu 22.2.2016

8 GameMaker Docs

<http://docs.yoyogames.com/> 2015. Luettu 22.2.2016

9 History of Unreal Part 1. Verkkojulkaisu

<https://www.beyondunreal.com/articles/history-of-unreal-part-1/?page=2> 7.3.2016. Luettu 22.3.2016

10 Unity - Editor

<http://unity3d.com/unity/editor> 2016. Luettu 12.3.2016

11 Unity – Multiplatform

<http://unity3d.com/unity/multiplatform/> 2016. Luettu 12.3.2016

12 Unity – Component

<http://docs.unity3d.com/ScriptReference/Component.html> 2016. Luettu 7.1.2016

13 CryEngine Docs

<http://docs.cryengine.com/display/SDKDOC1/Home> 29.6.2015. Luettu 22.3.2016

14 Introducing JSON

<http://www.json.org/> 21.3.2016. Luettu 22.3.2016

15 Defining Game Mechanics

<http://gamestudies.org/0802/articles/sicart?viewType=Print&viewClass=Print> 19.12.2015. Luettu 22.3.2016

16 C# Language Specification. Kirja

<http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-334.pdf 3.2.2015. Luettu 22.3.2016

17 Why are role-laying games so popular?. Verkkojulkaisu

<http://speeli.com/articles/view/Why-are-role-playing-games-(RPG)-so-popular%3F> Luettu 24.2.2016

18 What makes a video game addictive?

<http://www.video-game-addiction.org/what-makes-games-addictive.html> Luettu 24.2.2016