HAMK
HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

Ammattikorkeakoulututkinnon opinnäytetyö

Tieto- ja viestintätekniikan koulutus

Syksy 2022

Henri Toivonen

HAMK
HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

Tieto- ja viestintätekniikan koulutusohjelma
Tekijä Henri Toivonen
Työn nimi Mobiilipelin kehitys
Ohjaaja Antti Laakso

Tiivistelmä
Vuosi 2022

Opinnäytetyön tavoitteena on kehittää yksinkertainen mobiilipeli ja selvittää, miten peli saadaan kehitettyä mobiililaitteille. Tarkoituksena on tutkia pelikehityksen vaiheita, sekä selvittää millä pelimoottorilla, 3D-mallinnussovelluksilla ja muilla sovelluksilla voi kehittää mobiilipelejä. Pelinkehityksen vaiheita ovat esituotanto, tuotanto, testaus, julkaisu ja jälkituotanto.

Opinnäytetyön ensimmäisessä osiossa tutkittiin, mikä mobiilipeli on yleisesti ja mikä sen osuus on pelimarkkinoilla. Toisessa osiossa selvitettiin, mitä pelinkehitys on ja selvennettiin pelinkehityksen vaiheita ja sen osa-alueita. Kolmannessa osiossa tutkittiin, mikä on pelimoottori, sekä mitä työkaluja tarvittaisiin mobiilipelin kehittämisessä.

Työn neljäs osio käsittelee mobiilipelinkehityksen toteutusta. Käydään läpi mobiilipelistä tehtyä suunnitelmaa ja luodaan mobiilipeli valituilla työkaluilla. Tämä osio käsittelee peliprototyypin luontia, käyttöliittymän ja pelikenttien rakentamista, sekä ohjelmointia.

Viimeisessä osiossa kerrotaan omasta pelinkehityskokemuksesta, sen haastavuuksista ja onnistumisista. Pohditaan, miten pelinkehityksen eri vaiheet auttoivat mobiilipelin kehityksessä ja annetaan omat suositukset pelinkehityksen aloittamiseen.

Avainsanat Mobiilipelit, pelikehitys, peliohjelmointi
Sivut 44 sivua

HAMK
HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

Degree Programme in Information and Communication Technology

Abstract

Author Henri Toivonen
Year 2022
Subject Mobile game development
Supervisor Antti Laakso

The aim of the thesis is to develop a simple mobile game and find out how to develop the game for mobile devices. The aim is to examine the stages of game development and determine which game engine, 3D modeling applications and other applications can be used to develop mobile games. The stages of game development include pre-production, production, testing, publishing and post-production.

The first part of the thesis examined what a mobile game is in general and what is its share in the video game market. The second section explained what game development is and clarified the stages of game development and its areas. The third section examined what a game engine is, as well as what tools would be needed to develop a mobile game.

The fourth section of the thesis deals with the implementation of mobile game development. Additionally, the section presents the plan made for the mobile game and creates the mobile game with the selected tools. This section deals with creating a game prototype, building the user interface and game levels, and programming.

The last section talks about my own game development experience, its challenges and successes. I will reflect on how knowledge of the different stages of game development helped in the development of a mobile game as well as make recommendations in starting game development.

Keywords Game development, game programming, mobile games

Pages 44 pages

Sisällys

Kuvat

Liitteet

Sanasto

2D	Kaksiuloitteinen tietokonegrafiikka
3D	Kolmiuloitteinen tietokonegrafiikka
Android	Mobiilikäyttöjärjestelmä
C#	Microsoftin kehittämä olio-ohjelmointikieli
GDD	Game design document. Yksityiskohtainen pelin suunnitteludokumentti.
Genre	Pelin tyylilaji
Open Source	Avoin lähdekoodi
Pelimoottori	Videopelin ohjelmistokehys
Scene	Pelinäkymä, joka sisältää peliin rakennetut komponentit
Script	Koodia sisältävä tiedotsto
Unity Engine	Unity Technologies- yrityksen kehittämä pelimoottori

1 Johdanto

Tämän opinnäytetyön tavoitteena on saada kehitettyä mobiilipeli mobiililaitteille. Tutustutaan hieman mobiilipeleihin ja niiden osuuteen pelimarkkinoilla. Samalla selvitetään pelinkehityksen eri vaiheita ja millä sovelluksilla mobiilipeli voidaan kehittää.

Työn aluksi tutustutaan mobiilipeleihin ja tutkitaan niiden suosiota. Tutustutaan pelinkehityksen eri vaiheisiin ja pelinkehityksessä tarvittaviin työkaluihin. Itse työn toteutus aloitetaan tutustumalla työkaluihin, joilla mobiilipeliä kehitettiin, jonka jälkeen tarkastellaan mobiilipelin kehityksen kulkua.

Videopelit koostuvat säännöistä, haasteista, tavoitteista ja vuorovaikutuksista. Ennen videopelin kehittämistä, sinulla täytyy olla jonkinlainen suunnitelma videopelistä ja miten sellainen peli saadaan toteutettua. Kun lähdetään suunnittelemaan peli-ideaa, kannattaa kirjoittaa muistiin kaikki ideat, jotka voitaisiin rakentaa mobiilipeliin. Pelisuunnittelussa täytyy harkita useita pelin rakentamiseen liittyviä asioita: Millä työkaluilla videopeli tehdään, mikä on pelin genre, mille pelialustalle peli tehdään, tehdäänkö 2D- vai 3D-peli ja mikä on pelin kohdeyleisö. Mobiilipeli voidaan rakentaa pelimoottorilla ja pelimateriaalit voidaan tehdä itse tai ostaa internetistä lisensseineen. Kun on selvä idea videopelistä, joka halutaan kehittää, pitää päättää minkälaisella pelimoottorilla videopeli kehitetään. Suosittuja ilmaisia pelimoottoreita, jotka ovat yhteisön saatavilla ovat Unity Engine, Godot Engine ja Unreal Engine.

Jos halutaan kehittää 3D-videopelejä, tarvitaan 3D-mallinnussovellus 3D-mallien tekemiseen, kuten Blender tai Autodesk 3ds Max. Peleissä voidaan tarvita seuraavia asioita: 2D-kuvia, taidetta, 3D-malleja, musiikkia ja äänitteitä. Joskus omat taidonpuutteet tulevat esteeksi videopelien kehittämisessä, varsinkin jos niitä työstää yksin. Tässä tilanteessa internetistä voidaan ostaa materiaalia lisenssien kanssa omaan käyttöön, joita voi hyödyntää mobiilipelien kehityksessä.

Opinnäytetyön on tarkoitus vastata seuraaviin kysymyksiin: Mitä ovat mobiilipelit ja pelinkehitys? Mitä työkaluja tai sovelluksia pelinkehityksessä tarvitaan ja miten niitä käytetään?

2 Mobiilipeleistä yleisesti

Peli on kilpailun muoto tai sääntöjen mukaan pelattava urheilulaji. Pelejä pelataan useimmiten viihteen vuoksi, joskus myös saavutuksen tai palkinnon vuoksi. Pelejä voi pelata yksin, ryhmissä tai verkossa. Pelit sisältävät yleensä henkistä tai fyysistä stimulaatiota ja usein molempia. Monet pelit auttavat kehittämään käytännön taitoja ja toimivat harjoitusmuotona. (Kramer, 2000)

Videopelit ovat elektronisia pelejä, joita voidaan pelata elektronisilla laitteilla, kuten tietokoneella, pelikonsoleilla ja mobiililaitteilla. Mobiilipelit ovat pelejä, jotka on suunniteltu mobiililaitteille, kuten älypuhelimille ja peruspuhelimille.

2.1 Tunnettuja mobiilipelejä

Mobiilipelit ovat hyviä kasuaaleille. Kasuaalipelaajat ovat pelaajia, joilla ei ole paljon aikaa pelata tai he eivät pelaa kilpailuhenkisesti, vaan he yrittävät kuluttaa aikaa nopeasti ja pitää hauskaa. Kasuaalipelaajille kehitettyjä pelejä kutsutaan kasuaalipeleiksi, joissa on helpot ja yksinkertaiset pelimekaniikat, joita on helppo ymmärtää ja hallita. Kasuaalipeleissä eteneminen ei vaadi pitkiä peliaikoja, joten niitä voi pelata tauoilla. (Computer Hope, 2018) Esimerkki suositusta kasuaalipelistä on Candy Crush Saga -peli (kuva 1), jossa pelaajan täytyy saada tyhjennettyä rivejä ja sarakkeita, siirtämällä kolme tai enemmän samaa objektia vierekkäin. (King, 2022)

Kuva 1. Candy Crush Saga -peli (King, 2022).

Mobiilipeleissä on yleensä toistuva pelimekaniikka järjestelmä, jota kutsutaan ydinsilmukaksi, core loopiksi. Se on toimintojen sarja, jonka pelaaja toistaa jatkuvasti. Core loop luo ydinolemuksen sille, miksi pelaajat palaavat pelaamaan peliä uudestaan. Core loopit ovat yleensä lyhyitä ja helposti ymmärrettävissä. Esimerkiksi pelaajan täytyy suorittaa tehtäviä, jotta pelaaja ansaitsee rahaa, jolla hän voi ostaa rakennuksia tai pelitarvikkeita. (Wolstenholme, n.d.)

Clash of Clans (kuva 2) on Supercell Oy:n kehittämä reaaliaikainen strategiapeli mobiililaitteille, jossa pelaajan tarkoitus on rakentaa oma kylä. Pelaaja pystyy rakentamaan omaan kyläänsä puolustuksia ja rakennuksia, joilla pystyy kouluttamaan sotilaita. Pelaaja pystyy puolustamaan omaa kyläänsä muiden pelaajien hyökkäyksiltä tai hyökkäämään

toisten pelaajien kyliin. Pelaaja ansaitsee pelirahaa rakentamalla kultakaivoksia, tekemällä tehtäviä tai hyökkäämällä toisten pelaajien kyliin. (Katkoff, 2012)

Kuva 2. Clash of Clans (Supercell, 2022).

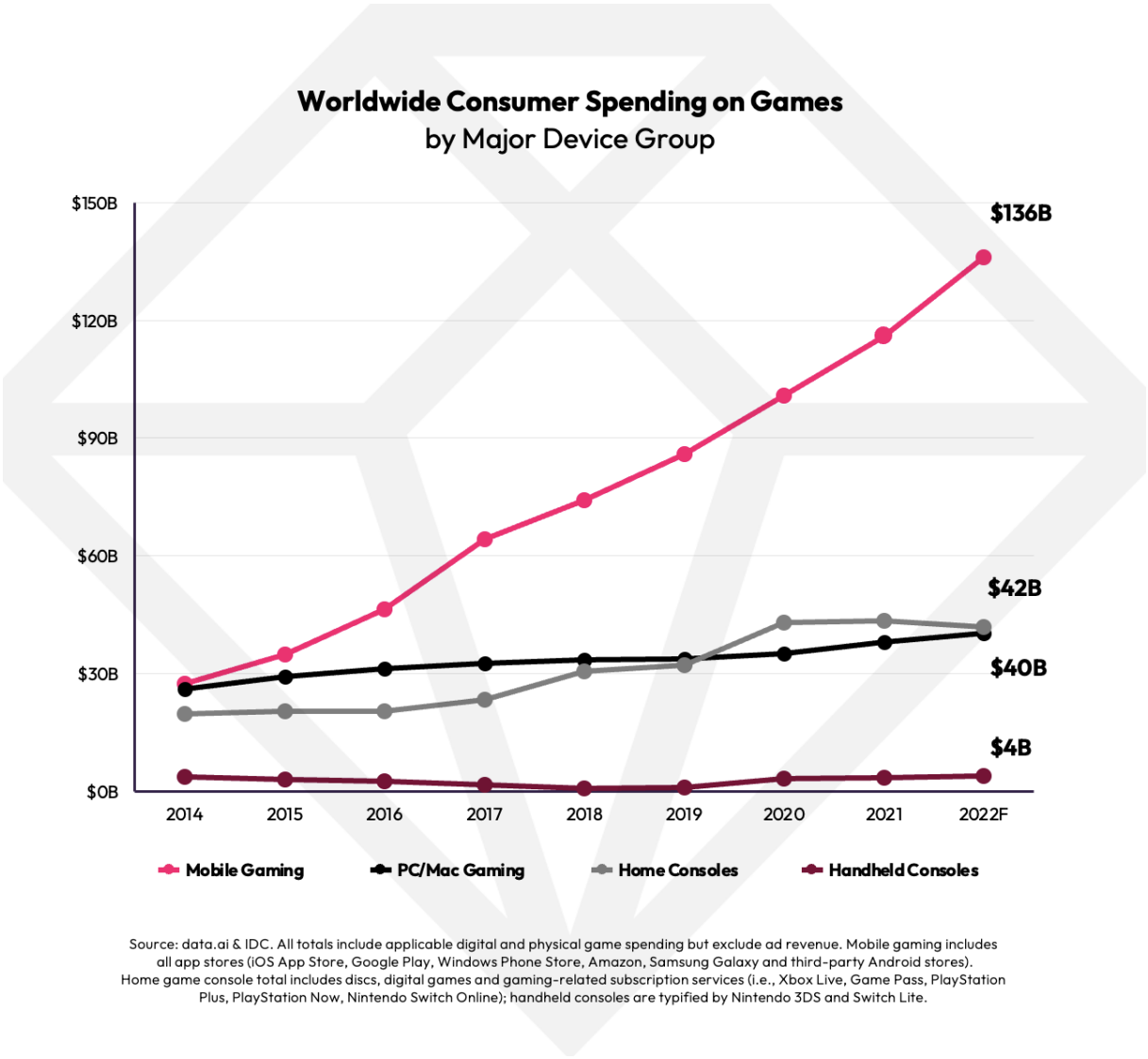
Kuvassa 3 näytetään Clash of Clans -pelin ydinsilmukka, jossa pelaajan täytyy kerätä kolikoita ja eliksiirejä, joilla pelaaja voi kouluttaa hahmoja tai rakentaa rakennuksia. Sen jälkeen pelaaja voi puolustaa omaa kyläänsä tai hyökätä toisen pelaajan kylään. (Wolstenholme, n.d.)

Kuva 3. Clash of Clans - Core loop (Wolstenholme, n.d.).

2.2 Mobiilipelien suosio

Mobiilipelit ovat nousseet suosioon pelimarkkinoilla. Data.ai:n ja IDC:n tekemän tutkimuksen mukaan, vuonna 2022 mobiilipelaaminen tulee ylittämään 60 % pelimarkkinaosuudesta (Kristianto, 2022). Kuvasta 4 näemme pelimarkkinoiden arvon neljän eri pelialustakategorian välillä.

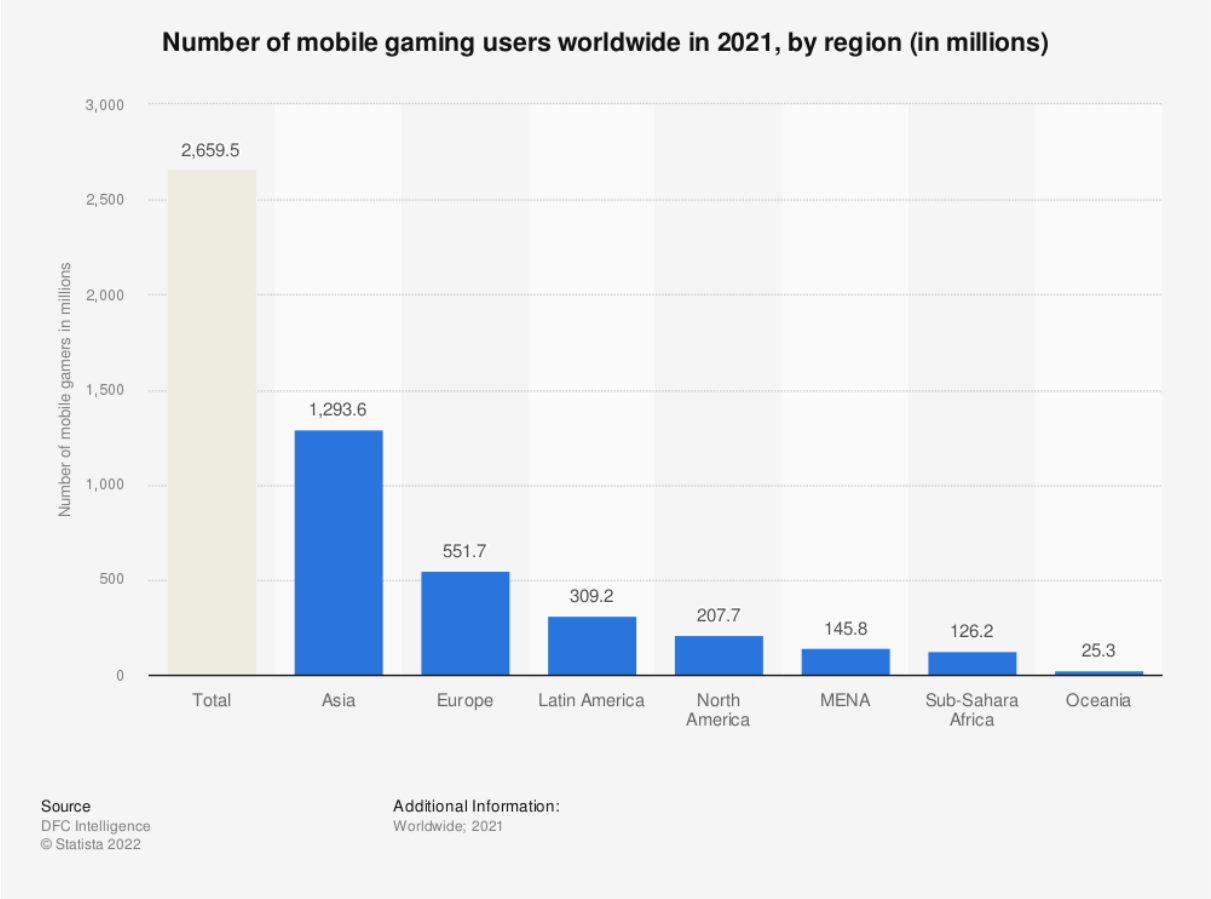
Kuva 4. Kuluttajien kulutukset peleihin maailmanlaajuisesti (Kristianto, 2022).



Mobiililaitteet ovat mahdollistaneet kuluttajille ja pelaajille helpon pääsyn videopeleihin. Mobiilipelaaminen on suosittua, koska mobiilipelien avulla voidaan kuluttaa aikaa ja rentoutua, sekä se sopii kiireellisten ihmisten elämään. Yksi suurimmista syistä mobiilipelaamisen suosioon on ilmaisten pelien saatavuus. Nämä ilmaiset pelit voidaan ladata ja pelata ilmaiseksi, mutta ne tarjoavat pelin sisäisiä ostoksia, mikrotransaktioita, jotka voivat sisältää lisäelämiä, tehosteita tai tasoja. Mikrotransaktiot ja pelin sisäiset mainokset tuottavat näiden ilmaisten pelien tuotot pelinkehittäjille. (Moroni, 2022)

Älypuhelimien suuri käyttäjämäärä vaikuttaa mobiilipelaamiseen. Yli kuusi miljardia aktiivista älypuhelimen käyttäjää maailmanlaajuisesti, joten kohdeyleisön tavoittaminen ei ole mahdotonta. (Gajsek, 2022) Kuvasta 5 näemme mobiilipelaajien määrän maailmanlaajuisesti. Aasian maissa on eniten mobiilipelaajia, joista suurin osa on Kiinasta.

Kuva 5. Mobiilipelaajien määrä maailmanlaajuisesti vuonna 2021, alueittain (Clement, 2021).



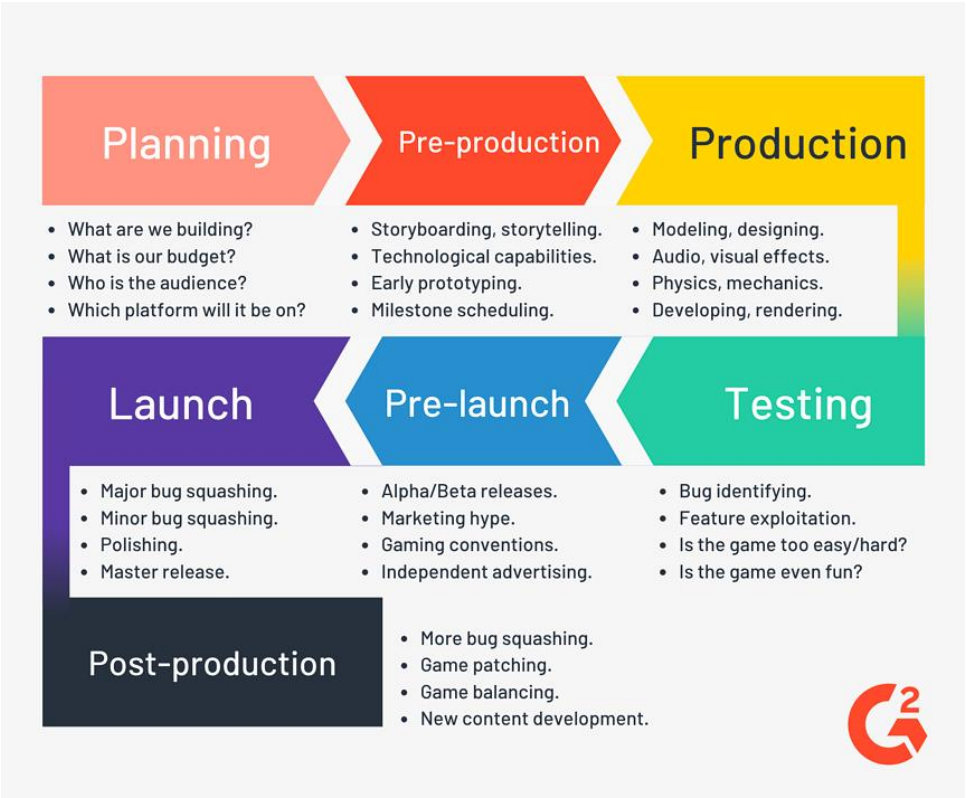
Mobiilipelejä kehitetään, koska niiden kehittäminen vaatii vähemmän työvoimaa ja ovat nopeampia testata. Mobiilipelien koko on huomattavasti pienempi verrattuna PC- ja konsolivideopeleihin ja niiden kehittämiseen kuluu vähemmän aikaa. Mobiilipelien kehittäminen on myös kannattavaa, koska niiden käyttäjämäärä ja pelimarkkinaosuus on suuri. (Gajsek, 2022)

3 Pelinkehityksen vaiheet ja osa-alueet

Pelinkehitys on pelin luomisen kokonaisuus, joka alkaa pelin konseptin luomisesta ja se päättyy analyysiin kuluttajan reaktiosta valmiiseen tuotteeseen ja havaittujen virheiden korjaamiseen.

Pelinkehitys koostuu tyypillisesti seuraavista vaiheista: Esituotanto, tuotanto, testaus, julkaisu ja jälkituotanto, mutta siihen voi kuulua myös muita vaiheita, kuten julkaisua edeltävä vaihe, pre-launch, jossa julkaistaan pelattava versio kuluttajille testattavaksi. (Kuva 6) Esituotanto vaiheen ensimmäinen askel on pelin suunnitteludokumentin (GDD) luominen. Pelin suunnitteludokumentti on ohjelmistosuunnitteludokumentti, joka toimii suunnitelmana rakennettavasta pelistä. Suunnitteludokumentti auttaa pelinkehittäjää ja pelinkehitystiimiä määrittelemään pelin laajuuden ja asettamaan projektin yleisen suunnan. (Tyler, 2022)

Kuva 6. Pelinkehityksen vaiheet (G2.com, Inc, 2019).



Esituotantovaiheessa tehdään pelisuunnitelma ja selvitetään, millä työkaluilla peli tullaan kehittämään. Esituotannossa selvitetään vastaukset seuraaviin kysymyksiin: Mille pelialustoille peli tehdään, mikä on pelin tyyppi, genre ja kohdeyleisö, mitä pelimekaniikkoja halutaan tehdä, 2D vai 3D-grafiikat, millä pelimoottorilla peli tehdään, millä sovelluksilla tehdään materiaalit. (Fisher, 2022)

Tuotantovaihe on työläin osuus pelinkehityksessä, joka on jaettu seuraaviin sisäisiin vaiheisiin: Prototyypin ja visuaalisen sisällön luominen, pelitason kehittäminen, äänisuunnittelu ja ohjelmointi. Tuotannossa kehitetään ensimmäisenä testattava prototyyppi pelistä, johon kehitetään pelin päämekaniikat. Sen avulla tarkistetaan pelimekaniikkojen ja käyttöliittymän toimivuus, pelattavuus ja käyttökokemus. 2D- ja 3D-Taiteilijat luovat pelihahmot, visuaaliset pelimateriaalit, erikoistehosteet ja käyttöliittymäelementit. Visuaalisen sisällön luominen vaatii luonnostelua, väritystä ja animointia. Pelitason kehittäjät luovat pelimaailman ja muut peliin sisältyvät pelattavat kentät. Äänisuunnittelussa luodaan peliin äänitteitä, musiikkia ja äänitehosteita, joiden avulla luodaan pelistä mukaansatempaava ja aidon tuntuinen. Peliäänien avulla annetaan pelaajille tieto vaaroista, tappioista ja voitoista. Ohjelmoijat kirjoittavat tuhansia koodirivejä yhdistääkseen kaikki pelin elementit ja käyttävät yleensä pelimoottoria tähän tarkoitukseen. Ohjelmoijat pyrkivät pitämään huolen siitä, että koodissa ei ole virheitä, jotka voisivat rikkoa pelikokemusta, jotta pelaajat saisivat positiivisen pelikokemuksen. (Game-Ace, 2021)

Testaus vaiheessa testataan pelin kunto ja toimivuus ennen pelin julkaisua. Testauksessa tehdään suuria päätöksiä. Pelimekaniikkoja tai pelinsisältöä saatetaan poistaa pelistä tai uusia ominaisuuksia on lisättävä pelattavuuden parantamiseksi. Pelin alpha-versiossa suurin osa pääominaisuuksista on lisätty ja peli on pelattavissa. Joitakin elementtejä voi vielä puuttua pelistä, mutta pelin säätimien ja toimintojen pitäisi toimia. Testaajat varmistavat, että pelin toiminnot toimivat ja raportoivat kaikista virheitä takaisin pelinkehittäjille. Pelin beetaversiossa, melkein kaikki sisältö ja resurssit on integroitu. Pelinkehittäjät keskittyvät tässä vaiheessa pelin optimointiin, jotta peli toimisi kaikilla käyttäjillä virheettömästi. Joitakin uusia toimintoja ja ominaisuuksia saatetaan lisätä beetaversion jälkeen. (Stefyn, 2022)

Kun pelin testaukset ovat onnistuneet ja peli koetaan valmiiksi, peli julkaistaan ja toimitetaan pelikauppoihin. Käyttäjät voivat sen jälkeen ostaa pelin tai ladata sen ilmaiseksi, riippuen siitä, mille pelialustoille peli on julkaistu ja millä ehdoilla.
Pelinkehitys ei pääty vielä pelin julkaisuun, vaan jopa julkaisun jälkeenkin peliä kehitetään ja parannellaan lisää, tätä kutsutaan jälkituotannoksi. Pelinkehittäjät saavat pelin julkaisun jälkeen palautetta käyttäjiltä pelissä olevista virheistä, ongelmista ja puutteista. Pelinkehittäjät korjaavat käyttäjien kohtaamia vikoja ja lisäävät peliin mahdollisesti myös käyttäjien toivomia lisäominaisuuksia. Korjausten jälkeen lähetetään päivitetty versio pelistä ja käyttäjät voivat ladata uudet päivitykset pelistä ilmaiseksi ja automaattisesti. (Mozolevskaya, 2021)

4 Pelinkehityksessä käytettävät työkalut

Pelinkehityksen esituotantovaiheessa tehdään selvitykset käytettävistä työkaluista. Pelinkehittäjälle tärkein työkalu on pelimoottori. Muita työkaluja, joita pelinkehittäjä tarvitsee ovat kuvankäsittelyohjelmat, 3D- mallinnusohjelmat ja äänityökalut.

4.1 Pelimoottorit ja muut työkalut

Varhaisia videopelejä varten, pelinkehittäjät ja pelisuunnitteluyritykset kehittivät omia pelimoottoreita, jotka oli suunniteltu erityisesti yhtä peliä varten. Nykyään pelinkehittäjille on tarjolla lukuisia pelimoottoreita, joita aloittelijat, pienet pelinkehittäjät ja yritykset voivat käyttää omien videopelien kehittämiseen.

Pelimoottori luo ohjelmistokehyksen videopelien rakentamiseen ja luomiseen. Ne tarjoavat useita työkaluja pelinkehittäjille, joilla luodaan pelimaailmat, pelihahmot, animaatiot ja muita videopelin rakentamiseen tarvittavia ominaisuuksia. Pelimoottorit vastaavat pelin grafiikkojen renderöimisestä, pelimekaniikoista, muistinhallinnasta ja pelifysiikasta, kuten peliobjektien törmäysten havaitsemisesta. (Interesting Engineering, 2016)

Pelimoottorit vaativat käyttäjältä ohjelmoinnin taitoja. Eri pelimoottorit käyttävät yhtä tai useampaa ohjelmointikieltä, joista yleisin on C++. Muita ohjelmointikieliä, joita pelimoottorit käyttävät ovat C, C# ja JavaScript. Muita hyödyllisiä ohjelmointikieliä ovat Python ja Java. (Rădulescu, 2020)

Pelimoottoria valittaessa täytyy ottaa huomioon, mille pelialustoille tai laitteille peli kehitetään, mitä ohjelmointikieltä pelimoottori käyttää. Samalla kannattaa selvittää, kuinka paljon dokumentaatiota tai ohjeita on tarjolla pelimoottorista. Mihin tarkoitukseen pelimoottori on suunniteltu, kuten 2D- vai 3D-pelien kehittämiseen.

2D-pelien kehittämiseen tarvitaan kuvankäsittelyohjelmia, joilla voidaan tehdä kuvamateriaalia videopeleihin ja 3D-pelien kehittämiseen tarvitaan 3D-mallinnusohjelma 3D-mallinnusten luomiseen. Hyviä kuvankäsittelyohjelmia ovat Adobe Photoshop, Adobe Illustrator ja GIMP. 3D-mallinnukseen hyviä 3D-mallinnus ohjelmia ovat Blender ja Autodesk 3ds Max.

Opinnäytetyössä mobiilipeli kehitettiin käyttämällä Unity pelimoottoria ja Blender-sovellusta 3D-mallinnuksiin. Projektin versiohallintajärjestelmänä käytettiin GitHub Desktop-sovellusta.

4.2 Unity Engine -pelimoottori

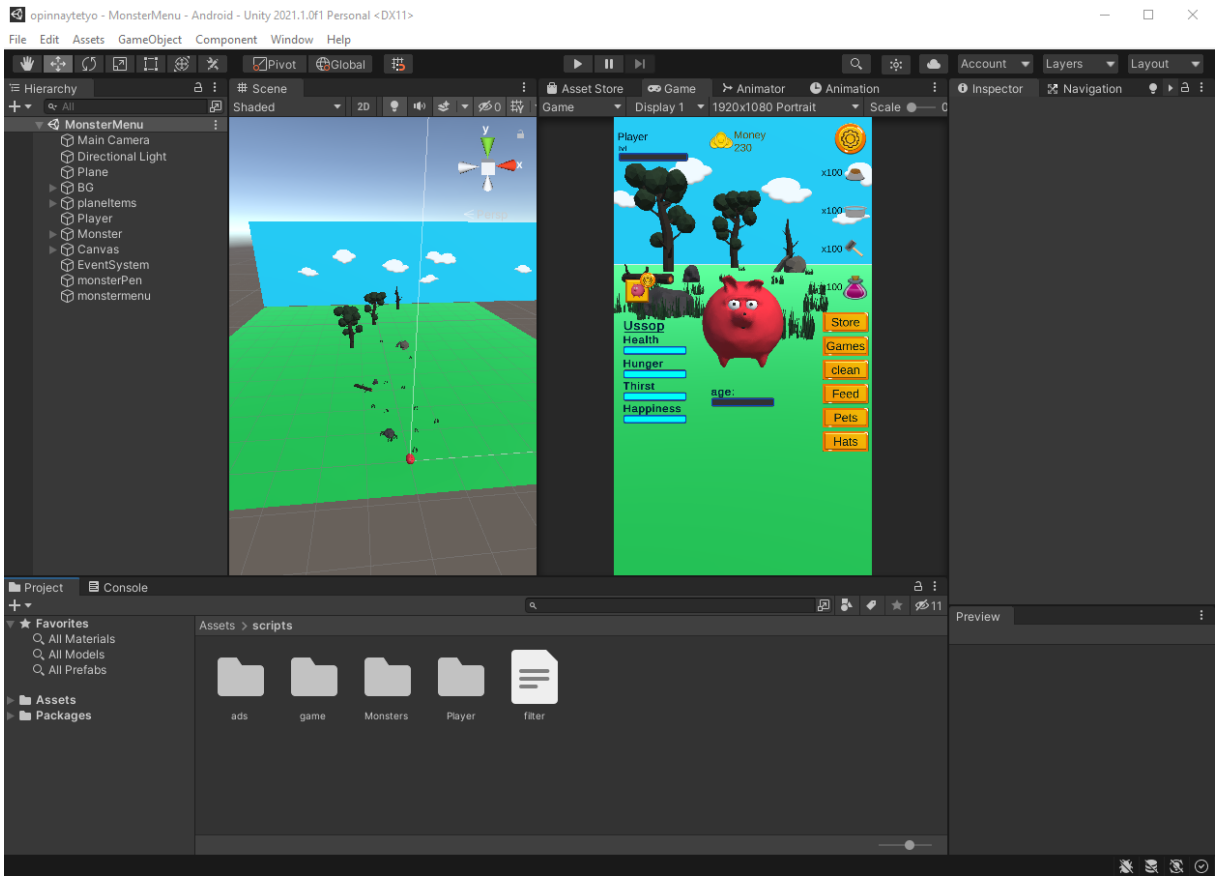
Tässä Opinnäytetyössä valitsin mobiilipelin kehittämiseen Unity pelimoottorin. Unity Engine on Unity Technologies -yrityksen kehittämä pelimoottori, jolla voi luoda 2D- ja 3D videopelejä useille eri alustoille. Unity engine on hyvä vaihtoehto aloittelijoille, koska se on yksinkertainen ja helppokäyttöinen pelimoottori, josta on tarjolla paljon ohjeita ja käyttäjille on tarjolla useita eri yhteisön tekemiä lisäosia. Tässä opinnäytetyössä käytettiin Unity Enginen 2021.1.0f1 versiota.

Unity engine on ladattavissa Windows-, Mac OS X -käyttöjärjestelmille ja Unity tukee virallisesti Linuxin Ubuntu ja CentOS käyttöjärjestelmiä. Unity Enginellä on mahdollista

kehittää pelejä useille eri pelialustoille, kuten tietokoneille, konsoleille, Android -ja iOS-laitteille ja monille muille alustoille. (Jead, 2022)

Unity Enginen editorin käyttöliittymä (kuva 7) sisältää useita eri ominaisuuksia ja työkaluja. Hierarchy -ikkuna näyttää jokaisen peliobjektin, joka on liitettynä sceneen. Scenet ovat komponentteja, joihin peli rakennetaan ja ne voidaan kytkeä toisiinsa. Jokainen scene voi sisältää oman peliympäristönsä, hahmonsa ja käyttöliittymänsä. (Unity Technologies, 2022a)

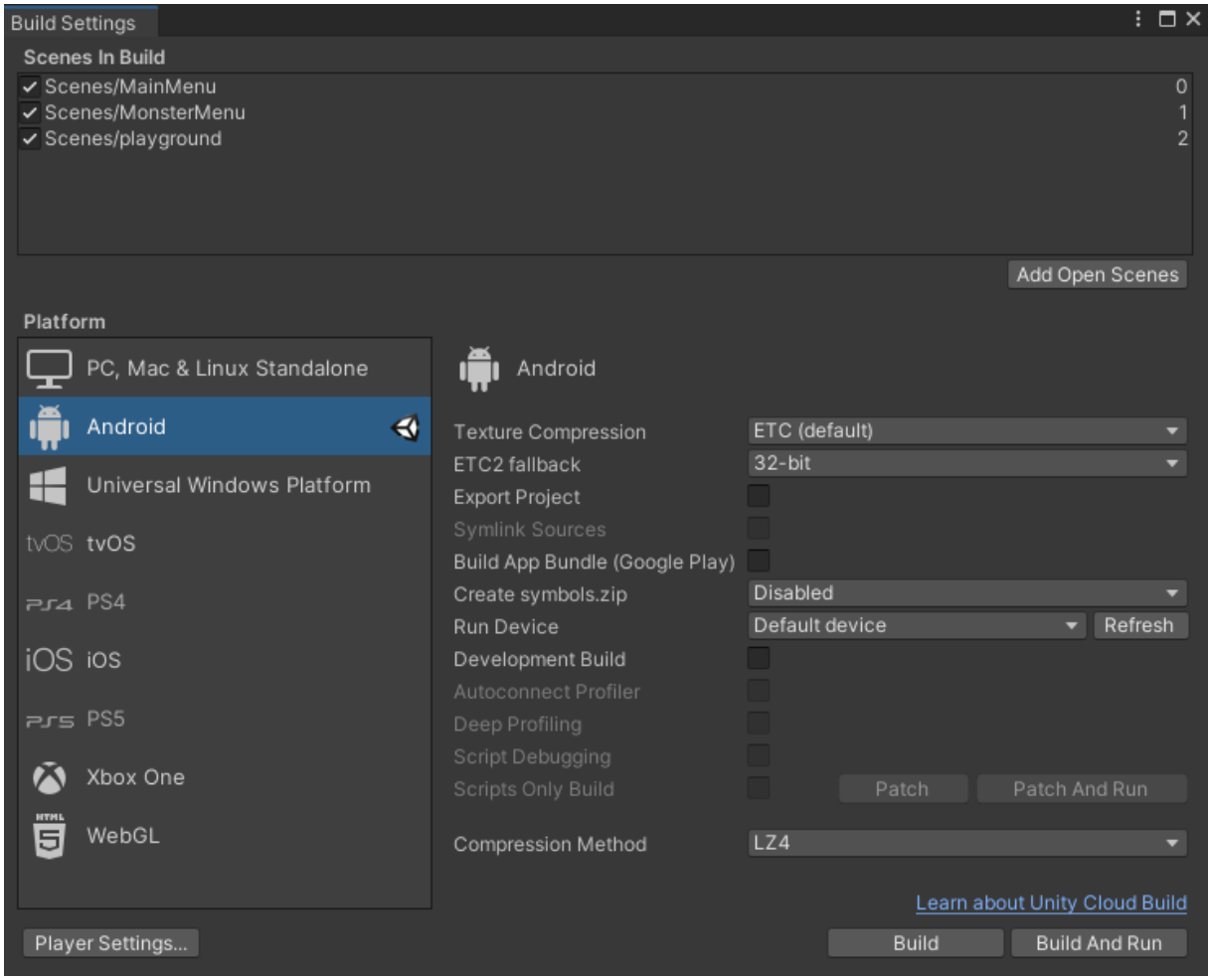
Kuva 7. Mobiilipeli Unity Enginen editorissa.



Game -ikkuna simuloi, miltä renderöity peli tulee näyttämään scenen kameroiden kautta. Inspector -ikkunan avulla voidaan tarkastella ja muokata valittujen peliobjektien asetuksia ja scriptejä. Project -ikkuna näyttää kaikki projektissa käytettävät resurssit, kuten scriptit, äänitteet, scenet, 2D ja 3D-mallinnukset. (Unity Technologies, 2022b)

Kuvassa 8 Unity Enginen Build Settings -valikko mahdollistaa peliprojektin rakentamisen valitulle pelialustalle. Scenes In Build- valikosta voidaan valita näkymät, joista rakennetaan pelikokonaisuus. Player Setting -valikosta voidaan tarkastella ja muokata pelin sisäisiä asetuksia.

Kuva 8. Unity Enginen Build Settings.

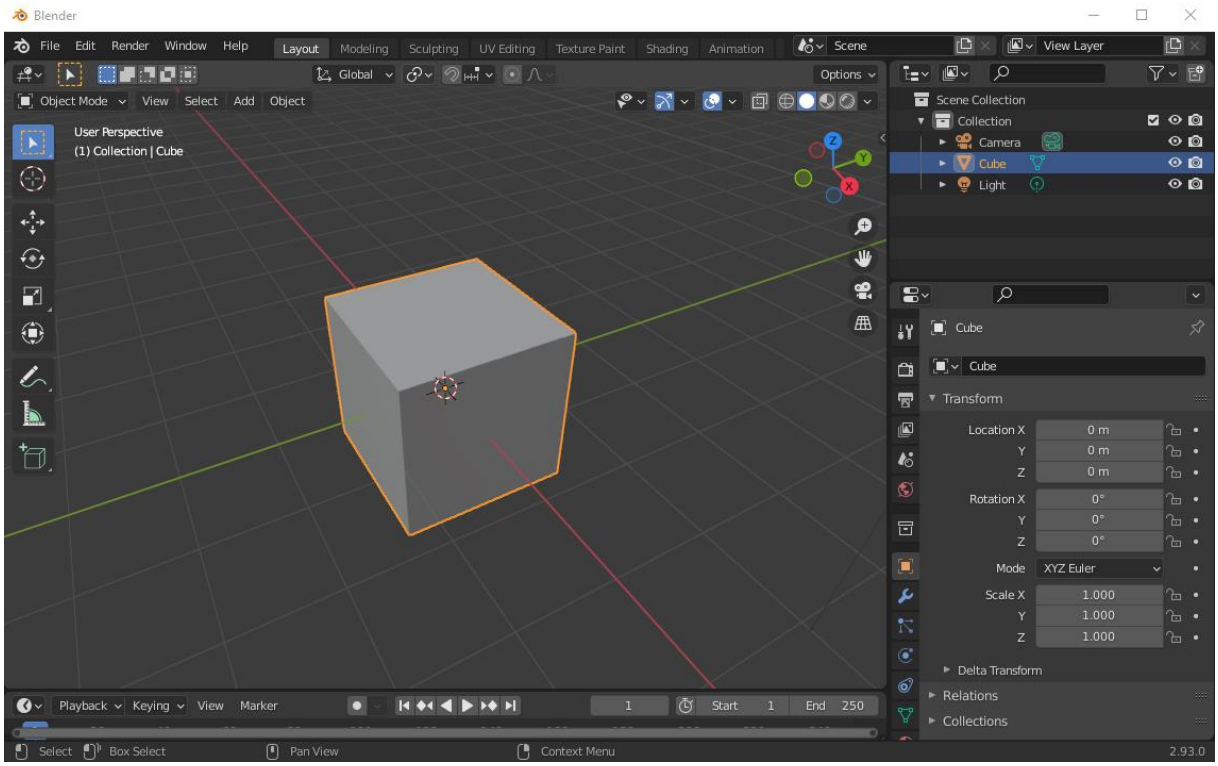


Unity Enginestä on tarjolla neljä eri lisensointivaihtoehtoa: Personal, Plus, Pro ja Enterprise. Personal lisenssiversio on ilmainen, jota voi käyttää kaikki, joiden tulot tai rahoitukset ovat alle 100 000 dollaria viimeisen 12 kuukauden aikana. (Unity Technologies, n.d.-a) Liitteessä 1 esitellään Unityn Lisensointivaihtoehdot tarkemmin.

4.3 Blender 3D-mallinnus ohjelma

Blender on ilmainen ja avoimen lähdekoodin 3D-mallinnustyökalu. (Kuva 9) Blender 3D ohjelman avulla pystytään suunnittelemaan ja mallintamaan videopeleihin pelihahmoja, esineitä ja pelikenttiä. Blenderillä pystytään myös tekemään animaatioita tehtyihin mallinnuksiin. Blender on ladattavissa Windows-, Mac OS X-, ja Linux -käyttöjärjestelmille. Blenderistä pystytään siirtämään 3D-mallinnukset, animaatiot ja muut tehdyt materiaalit suoraan Unity Engineen. (Blender Foundation, n.d.-a)

Kuva 9. Blender 3D-mallinnuksen ohjelmisto.



Blenderissä on kolme erilaista työkalua renderöintiä varten: Cycles, Workbench ja Eevee. (Kuva 10) Cycles on fyysinen polun jäljitin tuotannollisiin renderöinteihin, joka tarjoaa realistisia tuloksia renderöityihin tuotoksiin. Cycles pystyy käyttämään renderöinnissä näytönohjaimen lisäksi myös prosessoria, joka nopeuttaa renderöintiaikoja. Workbench renderöintimoottori on optimoitu nopeaan renderöintiin, mallintamisen ja animaation

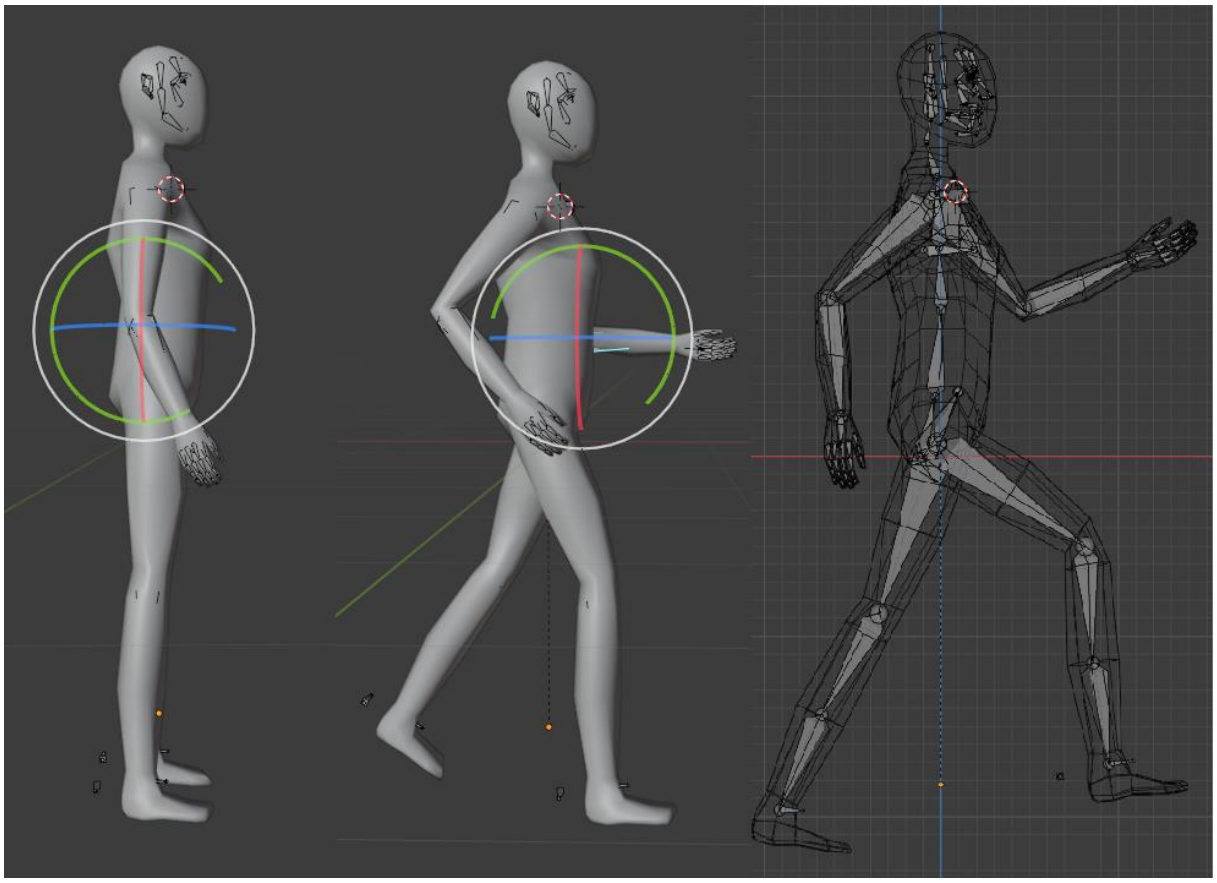
esikatselun aikana. Workbench ei ole tarkoitettu renderöimään lopullisia tuloksia, vaan sen ensisijainen tehtävä on renderöidä nopeasti mallinnus, kun sitä käsitellään. Eevee on reaaliaikainen renderöintimoottori, joka on keskittynyt nopeaan ja vuorovaikutteiseen renderöintiin. Eevee pystyy renderöimään mallinnuksia reaaliajassa Blenderin 3D-näkymässä ja se pystyy tuottamaan lopullisia renderöintejä korkealaatuisina. (Blender Foundation, 2022)

Kuva 10. Blenderin renderöinti työkalujen järjestys vasemmalta oikealle: Workbench, Cycles ja Eevee.

Blender tukee seuraavia 3D-tuotannon vaiheita: Mallintaminen, animointi, simulointi ja renderöinti ja monia muita. (Blender Foundation, n.d.-b)

Blender tarjoaa useita mallinnustyökaluja, joka tekee mallien luomisesta, muuntamisesta ja muokkaamisesta helppoa. Animointia varten, mallinnukselle voi lisätä virtuaalisen luurangon, tätä kutsutaan riggaukseksi, jonka avulla voidaan ohjata mallinnusta ohjailemalla luurankoa, esimerkiksi liikuttamaan käsiä ja jalkoja. (Kuva 11)

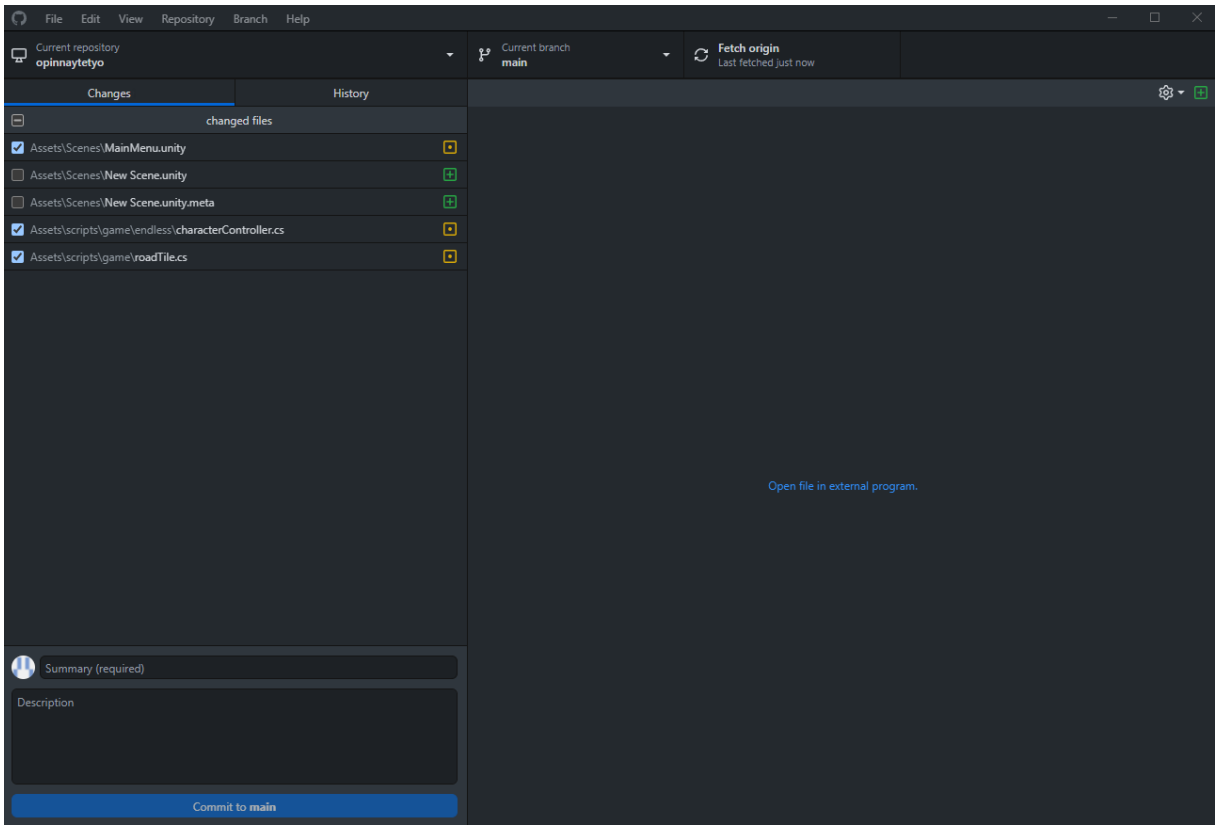
Kuva 11. Mallinnuksen ohjaaminen virtuaalisen luurangon avulla.



4.4 Git ja GitHub

Opinnäytetyön versionhallintajärjestelmänä on käytetty GitHub Desktop-sovellusta (kuva 12), joka on työpöytäohjelma, jonka avulla voidaan tallentaa ja ladata Git-projekteja. Git on Linus Torvaldsin luoma versiohallintajärjestelmä, jonka avulla voi säilyttää eri versioita omasta projektistaan, sekä jakaa niitä muiden projektiin osallistuneiden kanssa. GitHub Desktop mahdollistaa Gitin käytön graafisesti komentorivin sijasta. GitHubin kautta voidaan helposti ladata projekti omaan tai muiden projektin jäsenten käyttöön. GitHubin avulla voidaan myös palata takaisin aikaisempaan projektiversioon, jos nykyisessä versiossa on ilmennyt virheitä. (Chacon & Straub, 2014, s.14)

Kuva 12. GitHub Desktop.



GitHub Desktop- sovelluksella voidaan ladata GitHubista valmis projekti tai tallentaa projekti suoraan GitHub-palvelimille. Repository on kansio, joka sisältää kaikki projektin tiedostot ja kunkin tiedoston versiohistorian. GitHub Desktop näyttää kaikki projektin muutokset ”Changes”-ikkunassa, josta voi tarkastella muutoksia. Hyväksytyt projektin muutokset voidaan julkaista GitHubin palvelimille. History- ikkunasta voidaan tarkastella aikaisempia projektiversioita, joihin voidaan palata tarvittaessa takaisin.

5 Mobiilipelin kehittäminen

Opinnäytetyön mobiilipelin kehittäminen eteni alkuun hitaasti, koska mobiilipelin kehitys aloitettiin tutustumalla Godot-pelimoottoriin, josta käytettiin Godot Enginen 3.2.3 versiota. Godot on ilmainen ja avoimen lähdekoodin ohjelma. Godot on vuonna 2014 MIT- lisenssillä julkaistu 2D- ja 3D-pelimoottori (Linietsky, 2014), joka käyttää ohjelmointikielenä GDScript,

C# ja C++. GDScript on Godot Enginen oma ohjelmointikieli, joka vastaa Python ohjelmointikieltä. Godot Engine tukee Windows, MAC OS ja Linux tietokonejärjestelmiä. Godot on hyvin kevyt pelimoottori, joka mahdollisti sen nopean käynnistyksen ja toimintojen suorittamisen. Godotissa voi käyttää pelikehityksessä satoja sisäänrakennettuja pelikomponentteja, joita kutsutaan nodeiksi.

Lopulta päätettiin olla kehittämättä mobiilipeliä Godot Enginellä, koska Godot vaikutti liian monimutkaiselta ja tarjolla olevien yhteisön tekemien työkalujen määrä oli hyvin vähäinen. Godotissa piti tutustua Godotin käyttämiin nodeihin ja GDScripstiin, joista ei ollut ennen kokemusta ja joiden opetteluun meni liian paljon aikaa.

Mobiilipeliä aloitettiin kehittämään uudestaan käyttämällä Unity-pelimoottoria. Mobiilipelin kehittäminen aloitettiin rakentamalla aloituspohjan käyttämällä Unity Enginen tarjoamia työkaluja ja elementtejä. Aloituspohja on rakennettu ideoiden perusteella, jotka on suunniteltu etukäteen. Kun aloituspohja on saatu tehtyä, ohjelmoidaan toimintoja tehdyille käyttöliittymille ja pelin hahmoille. Seuraavaksi hahmotellaan pelimaailmaa tekemällä 3D-mallinnuksia ja graafisia muutoksia esineille, hahmoille ja käyttöliittymälle.

5.1 Pelisuunnitelma

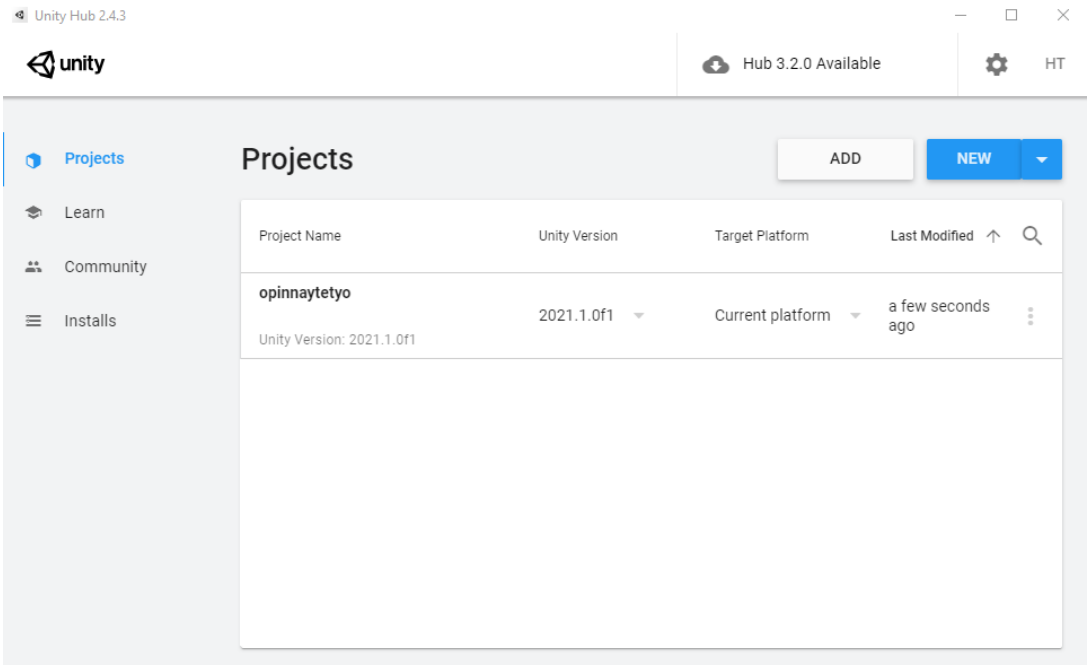
Opinnäytetyötä varten tehtiin suunnitelma mahdollisesta mobiilipelistä, joka kehitettäisiin mobiililaitteille. Tarkoituksena oli kehittää lapsille ja nuorille suunnattu ilmainen mobiilipeli. Ideana oli rakentaa Tamagotchin ja endless runner- genren tyylinen mobiilipeli. Tamagotchi on kädessä pidettävä virtuaalinen avaimenperälelu, jossa on kolme nappia. Tamagotchi on lemmikkipeli, jonka pääideana on pitää virtuaalista lemmikkiä elossa kasvattamalla ja hoitamalla sitä. Tamagotchi- laite pitää ääntä, jos lemmikki tarvitsee hoitoa, kuten ruokaa tai huomiota. Lemmikkien kasvattamisen ja hoitamisen lisäksi, mobiilipeliin haluttiin lisätä myös endless runner- tyylinen pelikenttä, jossa pelaajan valitsema lemmikki juoksee pitkää rataa pitkin keräten pisteitä ja vältellen esteitä.

Pelisuunnitelmana oli kehittää 3D- mobiilipeli, jonka pääideana on kasvattaa ja hoitaa omia lemmikkejä. Lemmikillä olisi elintasomittari, josta näkee lemmikin nälän-, janon- ja onnellisuustilan. Pelaajan täytyisi vahtia lemmikkinsä hyvinvointia ja kuntoa. Lemmikin hyvinvointia pelaaja pystyisi parantamaan antamalla lemmikille ruokaa ja vettä. Lemmikkejä varten kehitetään laaja alue, johon pelaaja pystyisi rakentamaan taloja, eläinkoppeja ja muita esineitä lemmikeille. Pelaajan täytyisi käydä lemmikin kanssa juoksuradalla, jotta lemmikin onnellisuus kasvaisi. Juoksuradalla pelaaja voi kerätä pelikolikoita, joilla pelaaja pystyy ostamaan kaupasta hyvinvointitarvikkeita ja hattuja lemmikillensä. Pelaajalla olisi pelin alussa vain yksi lemmikki, mutta pelaaja saisi lisää lemmikkejä saavutettuaan tietyn tason. Peli vaikeutuisi sitä mukaan, kun pelissä on lisää lemmikkejä, koska pelaaja joutuisi keskittymään useamman lemmikin hoitamiseen kerrallaan.

5.2 Projektin luonti Unityllä

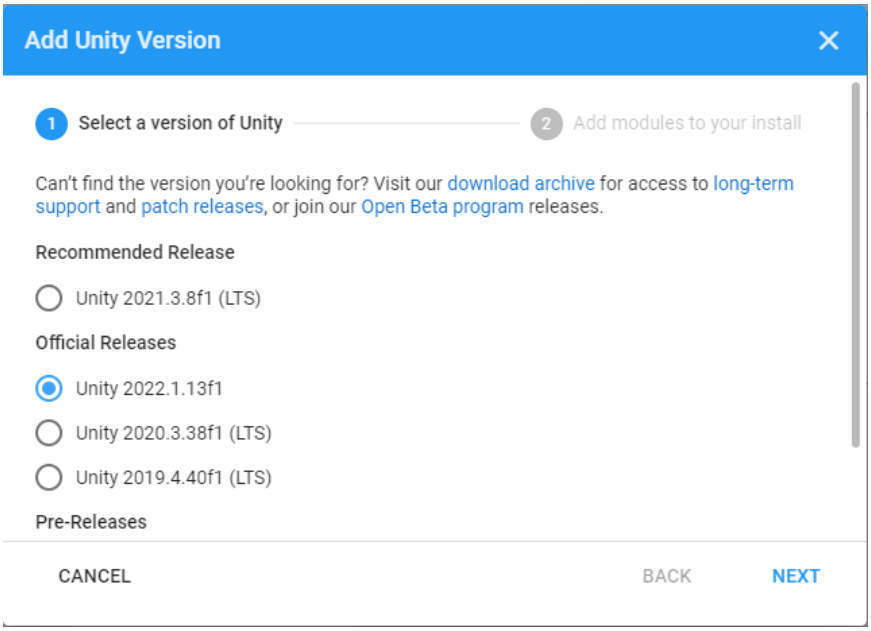
Unity projektin luominen on yksikertaista. Oman Unity-projektin luomiseen täytyy ladata Unity Hub-sovellus. (Kuva 13) Ennen projektin luontia, käydään lataamassa tai valitsemassa projektissa käytettävä Unity versio latausosiosta, jonka jälkeen ladataan tarvittavat moduulit projektiin. Tässä opinnäytetyössä käytettiin Unity Version 2021.1.0f1 ja ladattiin Android Build Support moduuli. Latausten jälkeen voidaan luoda uusi projekti, jossa annetaan projektille nimi, tallennussijainti ja projektin rakenne. Projektiin valittiin 3D rakenne, joka luo tyhjän 3D projektin, joka käyttää Unityn sisäänrakennettua renderöijää.

Kuva 13. Unity Hub.



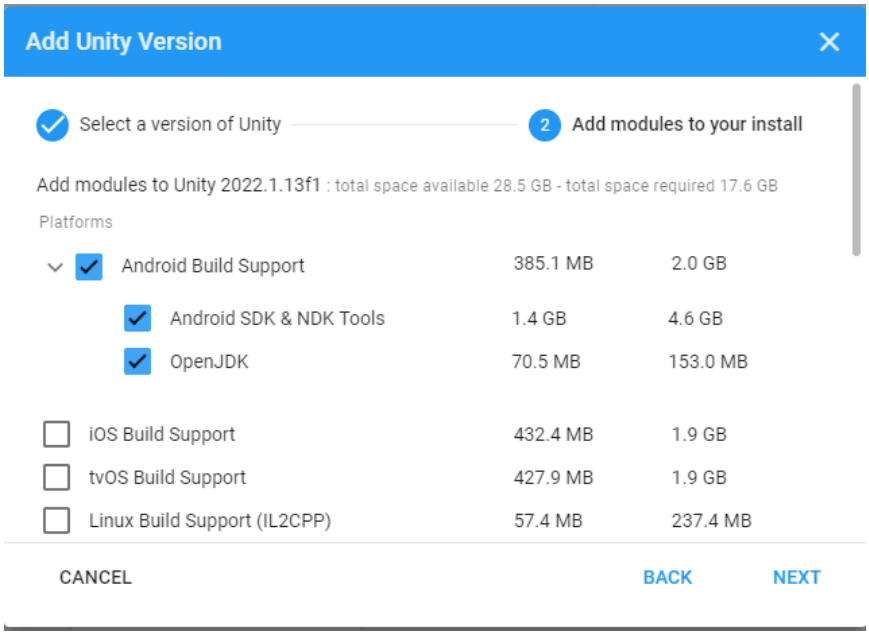
Kuvassa 14 valitaan Unity versio, joka halutaan ladata ja käyttää Unity -projektissa. Jos haluttavaa Unity versiota ei löydy listasta, voidaan käydä Unityn latausarkistosta lataamassa muu versio.

Kuva 14. Unity version valitseminen.



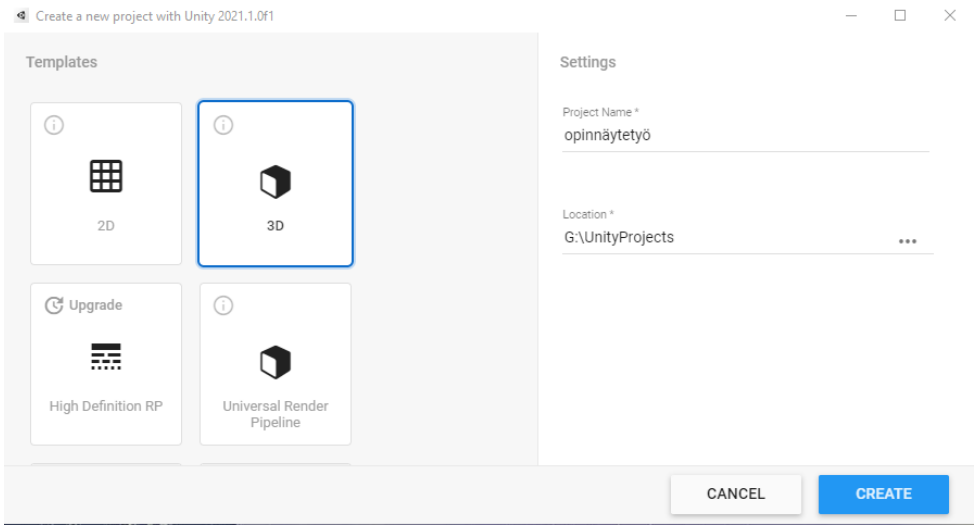
Kuvassa 15 on valittu Android tukimoduuli, jonka avulla saadaan kehitettyä peli mobiililaitteille. iOS tuen voi valita, jos peli halutaan myös kehittää iOS-laitteille.

Kuva 15. Asennettavat moduulit.



Kuvassa 16 valitaan projektissa käytettävä pohja, sekä annetaan projektille nimi ja asetetaan tallennuskansio.

Kuva 16. Uuden projektin luonti-ikkuna.



5.3 Pelipohjan rakentaminen

Mobiilipelin suunnitteluprosessin aikana oli tarkoitus rakentaa alkuun kolme erilaista sceneä eli pelinäkymää: Aloitusvalikon, lemmikkivalikko ja pelivalikko. Aloitusvalikossa pelaaja voi valita aloittavansa uuden pelin tai jatkaa siitä, mihin pelaaja jäi viime kerralla. Lemmikkivalikossa pelaaja pystyy tarkastamaan pelaajan edistymistä, lemmikkien hyvinvointia, ostamaan tuotteita ja pelaamaan tarjolla olevia pelejä. Pelivalikossa on pelejä, joissa pelaaja pystyy keräämään rahaa ja parantamaan lemmikkien kuntoa. Näiden suunnitelmien perusteella aloitettiin kehittämään projektiin yksinkertainen mallipohja jokaiselle pelinäkymälle, joihin kehitettiin objekteja, painikkeita ja toimintoja. Mallipohjan avulla voidaan tehdä ensimmäiset ja tärkeimmät ohjelmoinnit pelin toiminnoille. Mallipohjaa rakennetaan ja parannellaan jatkuvasti, kunnes saadaan lopputulokseksi valmis versio.

Aloitusvalikko on ensimmäinen pelinäkymä pelaajalle, kun käynnistetään mobiilipeli. Aloitusvalikon mallipohjaan rakennettiin alkuun neljä painiketta. Painikkeille ohjelmoitiin komennot, jotka avaavat toisen pelinäkymän tai sammuttaa sovelluksen. Asetukset-painike avaa ponnahdusikkunan asetuksille, jossa voi muokata ääniasetuksia. (Kuva 17.)

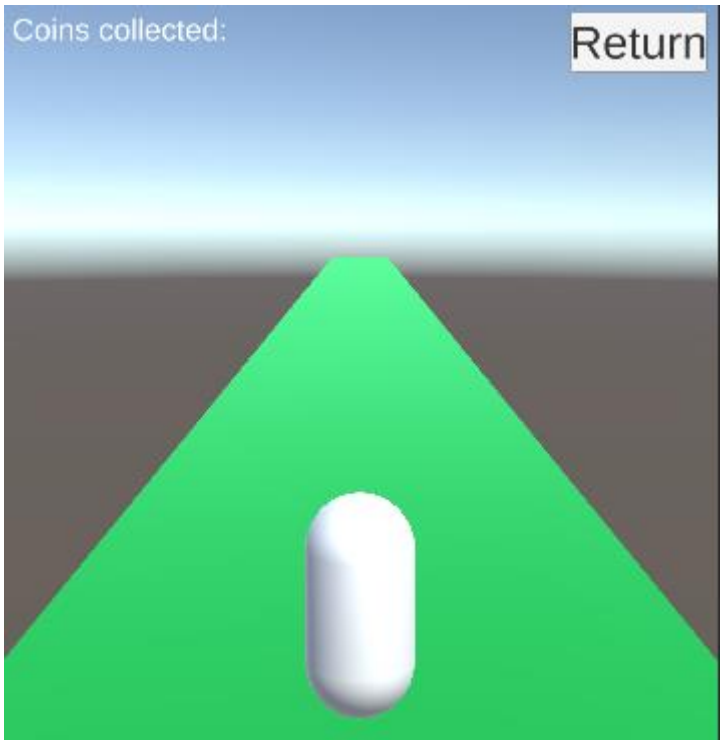
Kuva 17. Aloitusvalikon mallipohja.

Kuvassa 18 on lemmikkivalikon mallipohja, jonka tarkoitus on havainnollistaa, millainen näkymä halutaan kehittää lemmikkivalikolle. Käyttöliittymään on rakennettu painikkeita ja liukusäätimiä, joilla avataan ponnahdusikkunoita ja näytetään pelaajalle lemmikin hyvinvoinnin tila.

Kuva 18. Lemikkivalikon mallipohja.

Pelivalikon ensimmäiselle pelille rakennettiin pelattava hahmo, käveltävä tie ja automaattisesti päivittyvä teksti. Pelihahmolle ohjelmoitiin ensimmäiseksi automaattinen liikkuminen. Tielle tehtiin kolme kaistaa, jolla pelihahmo voi liikkua. Pelaaja pystyy siirtämään pelihahmoa eri kaistoille painamalla ruutua tai pyyhkäisemällä sormella vasemmalle tai oikealle. (Kuva 19)

Kuva 19. Pelivalikon mallipohja



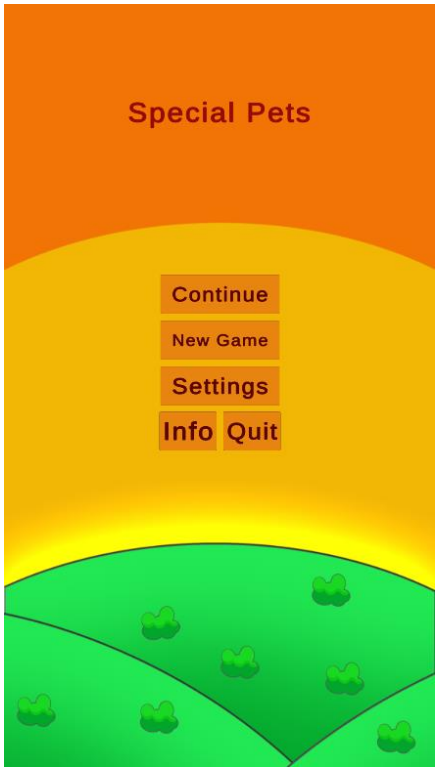
5.4 Käyttöliittymä

Käyttöliittymä on yksi tärkeimmistä videopelin osista, joka auttaa käyttäjää navigoimaan, löytämään tietoa ja saavuttamaan tavoitteita. Jokaisella pelinäkymällä on oma käyttöliittymänsä. Osa käyttöliittymän osioista on piilotettu, jotka pelaaja saa tarvittaessa näkyviin painamalla eri tarkoituksiin tarkoitettuja painikkeita. Unity käyttää käyttöliittymän ytimenä Canvas- peliobjektia. Canvas- peliobjekti sisältää kaikki käyttöliittymään sisältyvät elementit, kuten painikkeet, tekstit ja kuvat.

Kuvassa 20 on aloitusvalikon käyttöliittymä, jonka canvas-peliobjekti sisältää taustakuvan, viisi painiketta ja otsikkotekstin. Continue-painike on pelaajalle näkyvissä, jos peli on löytänyt laitteen tiedostoista pelitallenteen. New Game- painike aloittaa uuden pelin, mutta jos laitteesta on löytynyt pelitallennus, tulee ponnahdusikkuna (kuva 21), joka varmistaa

pelaajalta, että haluaako pelaaja varmasti aloittaa uuden pelin. ”Settings”-painike avaa ponnahdusikkunan asetuksille, jossa pelaaja voi muokata ääniasetuksia.

Kuva 20. Aloitusvalikon valmis käyttöliittymä



Kuva 21. Uuden pelin luominen.

Kuvasta 22 pelaaja näkee oman käyttäjänsä ja lemmikkinsä tiedot. Käyttöliittymä päivittää reaaliajassa lemmikin terveydentilan ja näyttää pelaajan käytössä olevien tarvikkeiden ja

rahan määrän. Painikkeista avautuu lisää käyttöliittymäelementtejä tai käynnistää ohjelmoidun komennon. Käyttöliittymä päivittyy reaaliajassa, jotta lemmikkien hyvinvointia pystyy seuraamaan tarkasti. Pelaajan rahatilanne kuluu sitä mukaan, kun pelaaja ostaa kaupasta tavaroita ja tavaroiden määrä käyttöliittymässä nousee tai laskee riippuen niiden käytöstä. Pelaaja pystyy vaihtamaan lemmikkiä painamalla ”Pets”- nappulaa, joka tuo näkyviin ponnahdusikkunan, jossa on kaikki pelaajan omistuksessa olevat lemmikit. Pelaajan vaihdettuaan lemmikkiä, käyttöliittymä näyttää valitun lemmikin tiedot. Pelaaja pystyy puhdistamaan lemmikkinsä painamalla ”clean”-painiketta ja ruokkimaan lemmikkiä painamalla ”Feed”-painiketta. Jokaiselle lemmikille pelaaja pystyy valitsemaan oman hattunsa painamalla ”Hats”-painiketta.

Kuva 22. Kuva lemmikkivalikon valmiista käyttöliittymästä.

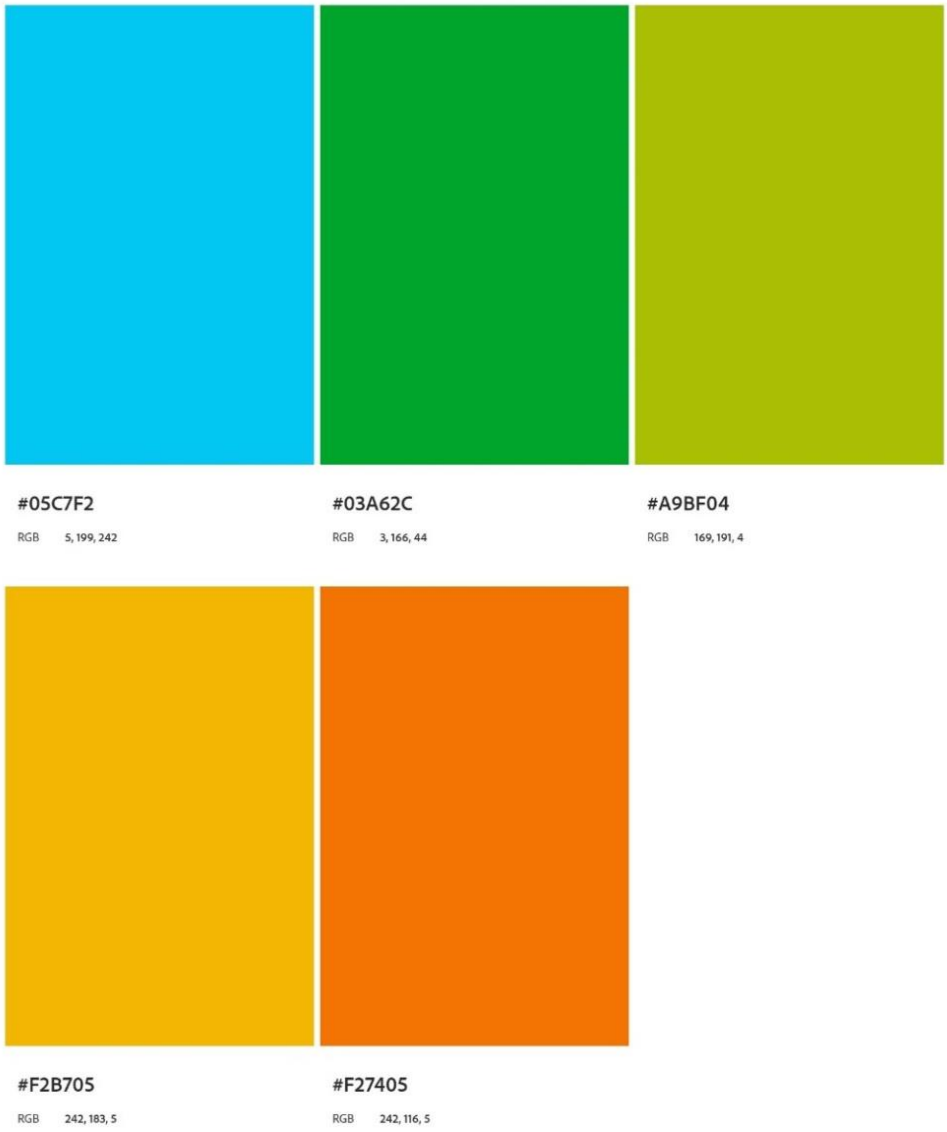
”Games”-ponnahdusikkuna (Kuva 23) näyttää kaikki pelattavat pelit, joissa pelaaja voi kerätä rahaa ja parantaa lemmikkinsä kuntoa. Painamalla ponnahdusikkunan ”Play”-painiketta, peli siirtyy pelivalikkoon, jossa pelaaja pystyy valitsemaan lemmikin, jolla suorittaa valittu peli.

Kuva 23. Painikkeesta avautuva pelit-ponnahdusikkuna.

5.5 Grafiikat

Grafiikkojen värimaailman valitsemisessa käytettiin hyödyksi Adobe Color-sivustoa ja tutkittiin väritrendejä. Suunnitelmana oli löytää väritrendi, joka olisi lapsille ja nuorille sopiva. Opinnäytetyössä käytettiin hyödyksi kuvan 24 teemaa, jolla luotiin värit käyttöliittymän painikkeille ja taustakuville. Samalla käytettiin hyödyksi Adobe Colorin helppokäyttötyökalua (kuva 25), jonka avulla löydettiin mobiilipelin teksteille sopiva väri, joka määräytyi taustakuvan värin perusteella. Sijoittamalla värikoodi Tekstin väri- osiolle ja Taustaväri-osiolle, Adobe Color ilmoittaa käyttäjälle, jos värit sopivat hyvin yhteen.

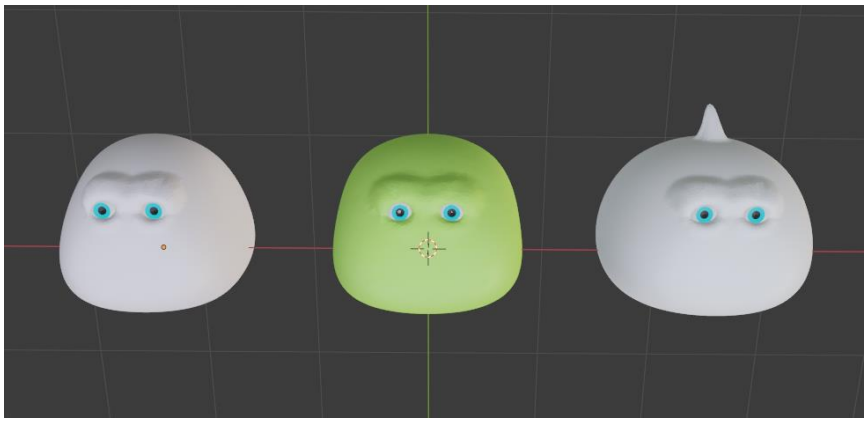
Kuva 24. Flor Cebollan luoma Adobe Color teema.



Kuva 25. Adobe Color Helppokäyttötyökalu

Blender 3D mallinnussovelluksella tehtiin kaikki 3D-mallinnukset. Kuvissa 11, 26, 27 ja 29 on esimerkkejä luoduista 3D-malleista. Blenderistä saa tuotua helposti Unity Engineen 3D-mallinnukset. Valmis 3D-mallinnus voidaan viedä Blenderistä Unity projektiin .fbx-muodossa. Tallenteen mukana tulee 3D-mallinnus ja animaatiot. Unity Engine tukee .fbx ja .obj 3D-tiedostomuotoja. Unity Engine pystyy myös lukemaan seuraavia tiedostomuotoja; .dae ja .dxf. (Unity Technologies, 2022)

Kuva 26. Blenderillä luotuja 3D pelihahmoja.



Kuva 27. Blenderillä luotuja peliobjekteja.

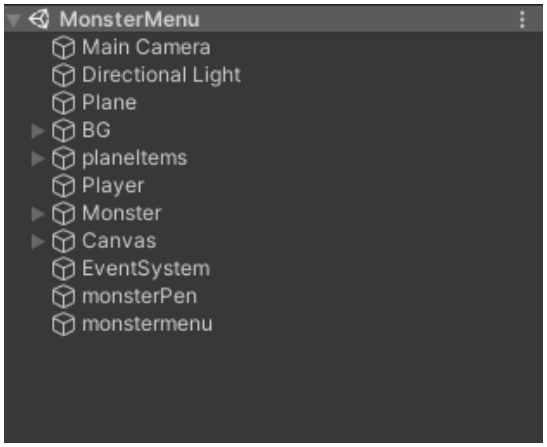
Hahmojen ja esineiden kuvakkeet tehtiin Blenderillä ja Adobe Photoshopilla. Blenderissä otettiin kuva esineistä ja pelihahmoista Blenderin sisäänrakennetulla kameralla, jonka jälkeen kuvia muokattiin Adobe Photoshopilla ja siirrettiin .png-tiedostomuodossa Unity Engineen. 2D-kuvia hyödynnettiin kaikissa painikkeissa ja taustakuvissa.

5.6 Pelikentät

Pelin ensimmäinen pelattava pelikenttä on endles runner- tyylinen peli, jossa pelihahmo juoksee niin pitkään, kunnes se törmää esteeseen tai pääsee maaliin saakka. Pelikenttä on rakennettu käyttäen peliobjekteja, joihin on liitetty ohjelmistokoodia ja muita komponentteja. Kuvassa 28 on pelikentän käytössä olevat peliobjektit, jotka luovat pelimaailman ja sen toiminnot. Jokaisella peliobjektilla on oma merkityksensä, komponentit ja ominaisuudet. Main Camera, eli pääkamera kaappaa kuvaa pelistä ja näyttää pelimaailman pelaajalle. Jotkin peliobjektit sisältävät pelkästään ohjelmistokoodia ja ne eivät

ole näkyvissä pelaajalle. Niiden tehtävä on ainoastaan suorittaa toimintoja ja sisältää pelille tärkeitä tietoja.

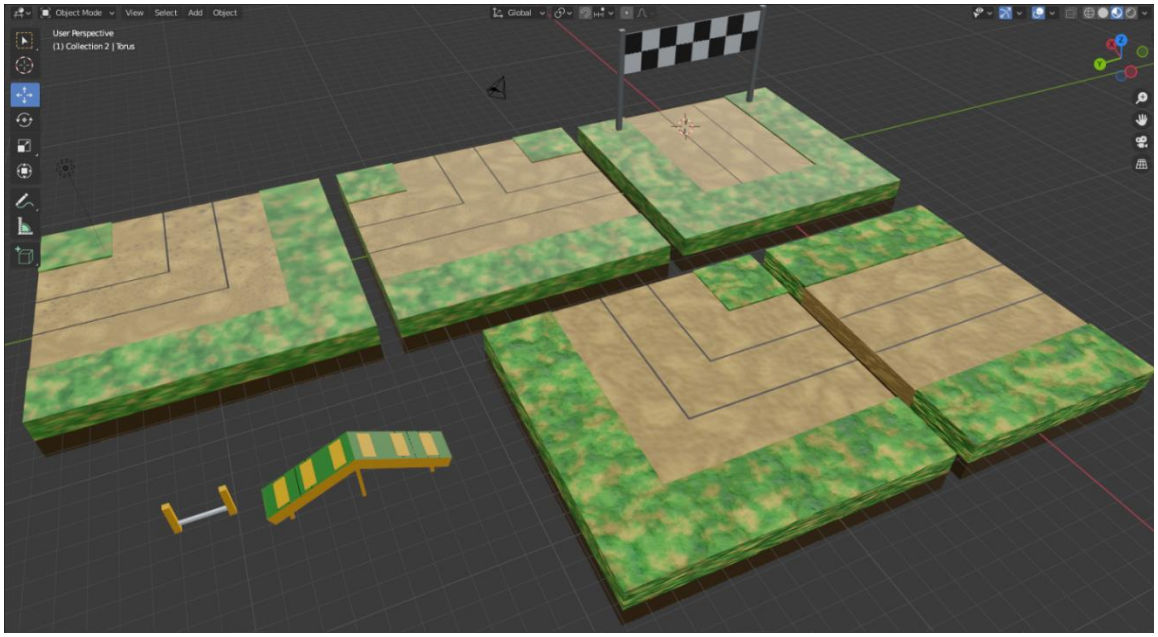
Kuva 28. Pelikentän peliobjektit.



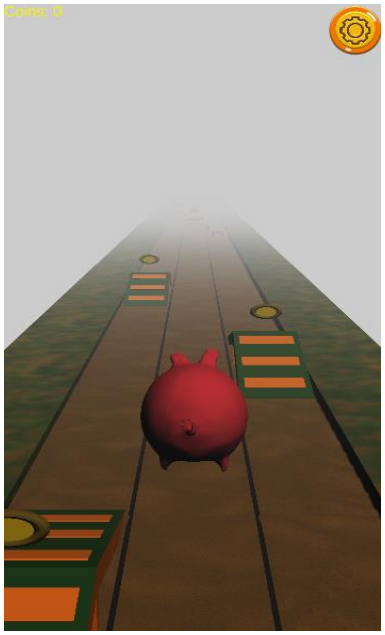
Pelikenttä muuttaa suuntaansa satunnaisesti ja juoksuradalla on kolme kaistaa, jossa pelihahmo voi liikkua. (Kuva 30) Pelaaja pystyy ainoastaan vaihtamaan pelihahmon kaistaa. Kenttä on ohjelmoitu poistamaan vanhat ja pelaajan ylittämät tiet, jotta ne eivät aiheuta suorituskykyongelmia pelattavalla laitteella. Peli lisää uusia teitä sitä mukaan, kun niitä poistetaan, jotta pelissä on aina tietty määrä teitä. Kaistoilla tulee vastaan satunnaisesti esteitä ja kolikoita. Tavoitteena pelaajan on väistettävä esteitä ja päästävä maaliin saakka. Lemmikin törmätessä esteeseen, pelaaja joutuu aloittamaan pelikentän uudestaan tai palaamaan lemmikkivalikkoon. Matkan varrella pelaaja voi kerätä kolikoita. Tämä pelimuoto on pelaajan tärkein keino kerätä rahaa. Pelaajan päästyään maaliin, pelaajalle näytetään

kerättyjen rahojen määrä, jotka siirtyvät pelaajan käytettäväksi. Pelikentän esineet ja maasto on tehty käyttäen Blender 3D-mallinnuksia. (Kuva 29)

Kuva 29. Blenderillä tehdyt 3D-mallinnukset endless runner pelikenttään.



Kuva 30. Unityssä rakennettu endles runner pelikenttä.



Pelattavissa pelimuodoissa pelaaja pystyy valitsemaan lemmikin, jolla pelaaja haluaa pelata. Pelaajan päästyään maaliin, pelaaja ja lemmikki saavat tasopisteitä, sekä lemmikki saa nostettua onnellisuusmittaria.

5.7 Ohjelmointi

Unity pelimoottorilla on mahdollista ohjelmoida toimintoja käyttäen C#-ohjelmointikieltä tai javascriptiä. Tässä opinnäytetyössä on käytetty C#-ohjelmointikielen kirjoittamiseen Microsoft Visual Studio 2017 -ohjelmistoa. Jos Visual Studio on liitettynä Unityyn, niin Visual Studio käyttää samalla Unityn omaa virheentarkistusta, joka nopeuttaa virheiden löytämistä ja korjaamista.

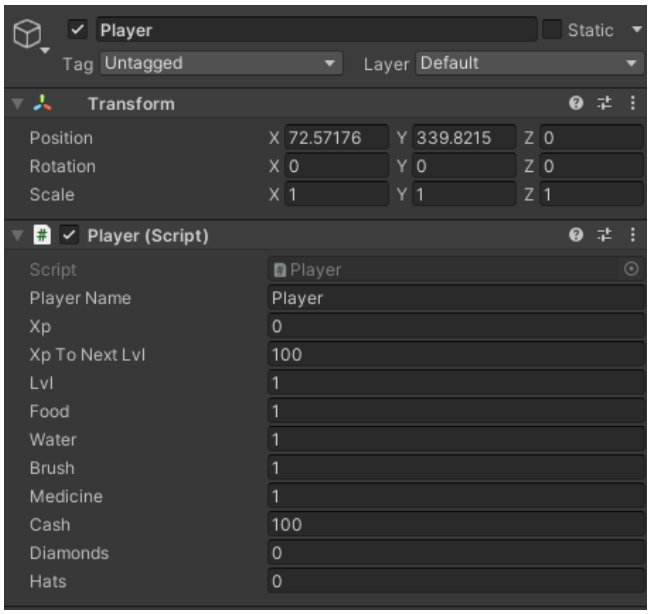
Tässä opinnäytetyössä ohjelmoitiin ensimmäisenä aloitusvalikko, josta pystyy siirtymään muihin valikkoihin ja muokkaamaan pelin asetuksia painamalla niille tarkoitettuja painikkeita. Lemmikkivalikossa ohjelmoitiin pelaajaa, sekä lemmikkejä koskevia tietoja. Peli on ohjelmoitu vähentämään lemmikkien hyvinvointia ajan myötä ja pelaaja pystyy nostamaan lemmikkien hyvinvointia painamalla ohjelmoituja painikkeita. Pelivalikossa ohjelmointi keskittyy pelikentän luomiseen ja pelihahmon liikkeeseen, sekä pelihahmoon kohdistuvasta toiminnasta.

Unity käyttää objekteja, joihin voidaan liittää komponentteja, joiden avulla objekteja voidaan kontrolloida. Unity sallii käyttäjän luoda omia komponentteja käyttämällä sriptejä. Scripti on komentosarja, joka kertoo peliobjektille, kuinka käyttäytyä. Scriptien avulla voidaan suorittaa komentoja, luoda tekoäly hahmoille tai automatisoida pelin käyttäytymistä, muokata komponenttien ominaisuuksia ja käytöstä, sekä vastata pelaajan toimintoihin.

Kuvassa 31 on Pelaaja peliobjekti, jossa on jokaiselle objektille yleinen Transform-komponentti, joka määrää pelikentässä objektin sijainnin, kiertosuunnan ja objektin mitat. Peliobjektiin on liitettynä myös pelaajalle tärkeä Player-scripti, joka käsittelee kaikkia pelaajalle tärkeitä tietoja. Kaikki peliobjektit, joihin vaikuttaa pelin fysiikat, tarvitsevat

collider- tai rigidbody- komponentin, jotta ne eivät putoa maasta lävitse. Collider-komponentit käsittelevät törmäyksiä ja osumia toisiin peliobjekteihin, joilla on collider-komponentti liitettynä. Collidereiden avulla voidaan ohjelmoida tapahtumia, jotka tapahtuvat silloin kun collider törmää johonkin. Rigidbody- komponentti mahdollistaa peliobjektin käytön ja liikkumisen, kun niihin kohdistuu fysiikkaa, kuten painovoimaa ja vääntömomenttia.

Kuva 31. Peliobjekti komponentilla ja scriptillä.



Kuvassa 32 on lemmikkivalikon käytössä oleva koodi, jonka avulla näytetään pelaajalle lemmikin tiedot. Tiedot siirtyvät lemmikkivalikossa oleville liukusäätimille, joista näkee missä kunnossa lemmikki on.

Kuva 32. Lemmikkivalikon toiminnot.

```
void Start()
{
    gm = FindObjectOfType<GameManager>();
    currentMonster = FindObjectOfType<Monster>();
    scene = SceneManager.GetActiveScene().name;
    petName.text = currentMonster.petName;
    changeButtonImage();
}

// Update is called once per frame
void Update()
{
    if (scene == "PetMenu")
    {
        curHP.value = currentMonster.Health;
        curHap.value = currentMonster.Happiness;
        curHun.value = currentMonster.Hunger;
        curThir.value = currentMonster.Thirst;
        curXp.value = currentMonster.xp;
        age.text = "Age: " + currentMonster.lvl;
        petName.text = currentMonster.petName;
        curHP.maxValue = currentMonster.MaxHealth;
        curHap.maxValue = currentMonster.MaxHappiness;
        curThir.maxValue = currentMonster.MaxThirst;
        curHun.maxValue = currentMonster.MaxHunger;
        curXp.maxValue = currentMonster.xpToNextLvl;


    }
}
```

Kuvan 33 ohjelmointikoodi mahdollistaa lemmikin vaihtamisen lemmikkivalikossa. Pelaaja painaa lemmikkivalikossa ”Pets”-painiketta, joka avaa ponnahdusikkunan. Pelaaja valitsee ponnahdusikkunasta lemmikin, jonka tietoja halutaan tarkastella.

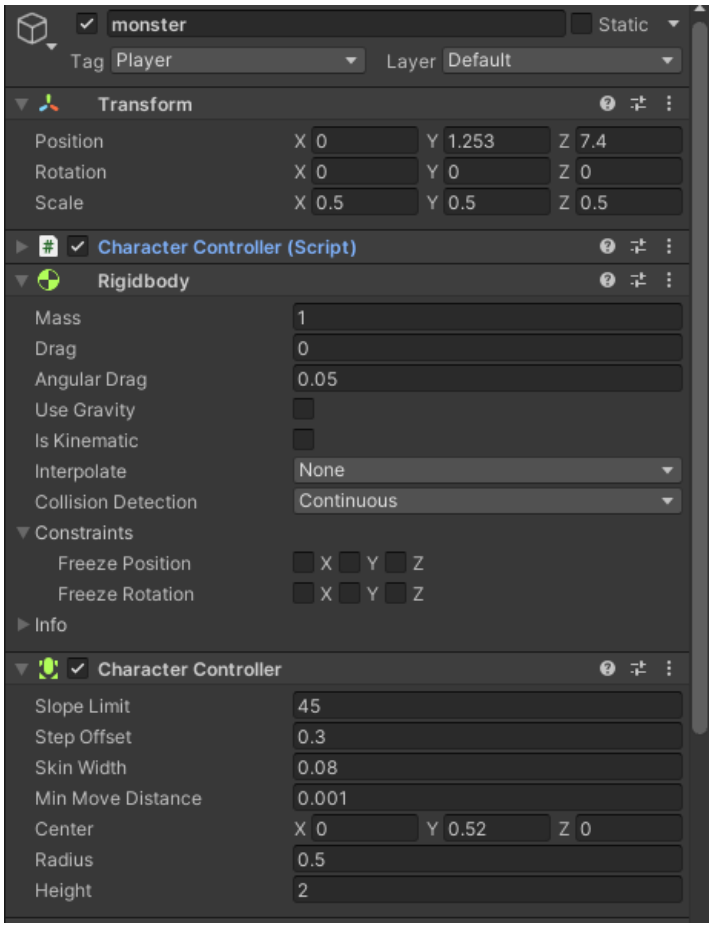
Kuva 33. Ohjelmointikoodi lemmikin vaihtamiselle lemmikkivalikossa.

```
public void changeCurMonster(int monster)
{
    gm.SaveMonster();
    Destroy(currentMonster.gameObject);
    var newMonster = Instantiate(monsters[monster], new Vector3(0, 0, 0), monsters[monster].transform.rotation, parent);
    newMonster.transform.localScale = new Vector3(1, 1, 1);
    currentMonster = newMonster.GetComponent<Monster>();
    petName.text = currentMonster.petName;
    changeButtonImage();
    FindObjectOfType<hatmanager>().updatePlacement();
    gm.LoadMonster();
    FindObjectOfType<hatmanager>().curMonsterHat(currentMonster.hatnum);
}
```

Pelihahmoilla on kuvassa 34 käytössä Character Controller -komponentti ja scripti, jotka käsittelevät hahmon liikettä ja toimintaa. Character Controller -komponentin avulla saatiin luotua hahmoille tasainen kävely, mutta sen toimintaa muokattiin scriptin avulla. (Kuva 35)

Character Controller -scriptin avulla muokattiin pelihahmoille liikkumiseen liittyviä ominaisuuksia, kuten jatkuvaa suoraan liikkumista, hahmon kääntyminen mutkissa automaattisesti ja kaistan vaihtamista pelaajan toimesta.

Kuva 34. Liikkuvan pelihahmon komponentit.



Kuva 35. Pelihahmolle ohjelmoitu automaattinen liikkuminen.

```
z += Time.fixedDeltaTime * moveSpeed;
x = direction.x;
move = new Vector3(x, velocity.y, z);
cc.Move(move);
velocity.y += gravity * Time.fixedDeltaTime;
cc.Move(velocity * Time.fixedDeltaTime);
Quaternion target = Quaternion.Euler(0f, 0f, 0f);
transform.rotation = Quaternion.Slerp(transform.rotation, target, Time.deltaTime * 5f);
```

Pelikentälle tehtiin toiminto, joka lisää teitä sitä mukaan, kuinka pitkälle pelaaja on edennyt. Pelikentälle ohjelmoitiin myös toiminto, joka poistaa pelaajan ylittämät tiet, jotta ne eivät ylikuormittaisi mobiililaitetta tai peliä. Pelikenttään ohjelmoitiin satunnaisesti ilmestyviä esteitä tielle, joita pelaajan täytyy väistää. Esteiden lisäksi ohjelmoitiin kolikoiden satunnainen ilmestyminen, joita pelaaja pystyy keräämään.

Peliin on ohjelmoitu käyttäjää varten pelin tallennus- ja latausominaisuus, joka mahdollistaa käyttäjän edistyksen tallentamisen laitteen tietoihin. (Kuva 36) Käyttäjä pystyy latausominaisuuden ansiosta jatkamaan peliä siitä, mihin käyttäjä keskeytti pelin viimeksi. Tallennusominaisuus tallentaa kaikki pelihahmon ja lemmikkien tiedot. Latausominaisuus hakee laitteesta pelitallenteen, joka syöttää tallennuksen tiedot takaisin pelihahmolle ja lemmikeille.

Kuva 36. Pelin Tallennuksen ja tallennuksen lataus koodi.

```
public static void SavePlayer(Player player)
{
    BinaryFormatter formatter = new BinaryFormatter();
    string path = Application.persistentDataPath + "/player.data";
    FileStream stream = new FileStream(path, FileMode.Create);
    Debug.Log(Application.persistentDataPath);
    gameData data = new gameData(player);
    formatter.Serialize(stream, data);
    stream.Close();
}
public static gameData loadPlayer()
{
    string path = Application.persistentDataPath + "/player.data";
    if (File.Exists(path))
    {
        BinaryFormatter formatter = new BinaryFormatter();
        FileStream stream = new FileStream(path, FileMode.Open);

        gameData data = formatter.Deserialize(stream) as gameData;
        stream.Close();
        return data;

    }
    else
    {
        return null;
    }
}
```

Kun peli käynnistää aloitusvalikon, suorittaa se ensimmäisenä komennon, joka etsii laitteesta mahdollisen pelitallennuksen. Jos pelitallennus löytyy, pelaaja voi jatkaa peliä siitä, mihin hän on jäänyt viimeksi. Jos pelitallennus ei löydy, pelaaja pystyy tekemään uuden pelin. Kuvassa 37 esitellään aloitusvalikon komentoja.

Kuva 37. Aloitusvalikon painikkeiden komennot.

```
public void findSave()
{
    string path = Application.persistentDataPath + "/player.data";
    if (File.Exists(path))
    {
        saveExist = true;
        button[0].interactable = true;
    }
    else
    {
        saveExist = false;
        button[0].interactable = false;

    }
}
public void continueGame()
{
    FindObjectOfType<GameManager>().newGame = false;
    SceneManager.LoadScene("MonsterMenu");
}
public void newGame()
{
    if (!saveExist)
    {
        FindObjectOfType<GameManager>().newGame = true;
        FindObjectOfType<GameManager>().deleteSaveData();
        SceneManager.LoadScene("MonsterMenu");
    }
    else
    {
        popUP.SetActive(true);
    }
}
public void quit()
{
    Application.Quit();
}
```

Peli tallentaa automaattisesti pelaajan edistyksen, jotta pelaajan ei tarvitsisi omatoimisesti etsiä tallennuspainiketta.

5.8 Materiaalit

Suurin osa materiaaleista tässä opinnäytetyössä, kuten kaikki 3D-mallinnukset ja osa graafisista kuvista on itsetekemiä. Jotkin materiaalit, kuten kaikki musiikit, äänitteet ja osa graafisista kuvista ovat oman taidonpuutteen takia jouduttu ostamaan lisensseineen, jotta niitä saataisiin käyttää pelissä.

Kaikki musiikit, äänitteet ja osa graafisista kuvista tässä opinnäytetyössä on ostettu lisensseineen Humble Bundle tarjouksista. Ostettuja Humble Bundle tarjouksia olivat: Humble Big Royalty-Free Music Bundle, Royalty-Free RPG Game Development Assets, Find

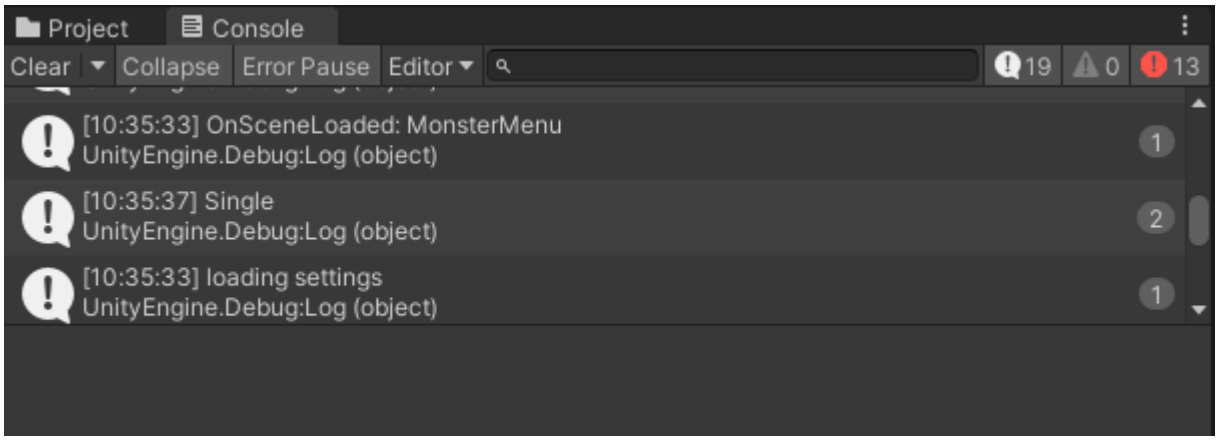
the Cure ja Game Dev GUI Goodness. Tarjoukset sisältivät materiaaleja, kuten 2D- kuvia, äänitteitä ja musiikkia useilta eri artisteilta ja taiteilijoilta.

5.9 Testaus

Pelin testaaminen on tärkeimpiä toimintoja, joita täytyy tehdä joka kerta, kun peliin tehdään uusia toimintoja tai korjataan virheitä. Pelitestauksia voidaan tehdä Unityn sisäänrakennetulla testausohjelmalla tai rakentamalla peli pelialustalle testattavaksi. Pelitestaus onnistuu Unity Enginen sisäänrakennetulla pelitestaukseen tarkoitetulla pelitestausikkunalla, jonka avulla peliä voi testata ilman, että se täytyisi rakentaa mobiililaitteille. Tällä testaustavalla pystytään korjaamaan selkeät ohjelmointivirheet ja muut ongelmat, mutta se ei pysty näyttämään kaikkia ongelmia. Rakentamalla Unityllä mobiililaitteille testattava versio pelistä, voimme tarkastella tarkemmin, mitä vikoja pelissä on. Pelifysiikkaan ja käyttöliittymään liittyvät ongelmat huomataan helpommin varsinaisesta mobiilipelin testiversiosta, joita Unityn sisäänrakennettu testiversio ei kykene näyttämään.

Kuvassa 38 testaajalle on tullut Unityn konsoliin ilmoituksia testauksesta, joka ilmoittaa kaikista mahdollisista virheistä ja peliin tehdyistä toiminnoista. Virheilmoituksista saa helposti selville, missä koodissa ja koodirivillä virhe ilmeni. Debug-viestit ilmoittavat, mitkä kooditoiminnot onnistuivat tai saatiin suoritettua.

Kuva 38. Debug-viestit ja virheilmoitukset Unityn konsolissa.



6 Pohdinta

Opinnäytetyön tavoitteena ollut mobiilipelin luominen onnistui. Mobiilipelin kehitys eteni alkuun hitaasti, koska mobiilipelin kehitys aloitettiin tutustumalla Godot- pelimoottoriin. Opinnäytetyössä ei lopulta päädytty kehittämään mobiilipeliä Godot Enginellä, koska Godot vaikutti liian monimutkaiselta ja tarjolla olevien yhteisön tekemien työkalujen määrä oli hyvin vähäinen. Godotissa piti tutustua Godotin käyttämiin nodeihin ja GDScripstiin, joista ei ollut ennen kokemusta ja joiden opetteluun meni liian paljon aikaa. Unity Enginen käytöstä löytyy enemmän opastusta ja ohjeita, verrattuna nuoreen Godot Engineen. Koska suunnitelmana oli kehittää 3D-mobiilipeli, haluttiin valita oikea pelimoottori sen kehittämiseen. Unity Engine tukee paremmin 3D-pelien kehittämistä, kuin Godot Engine, mutta se on hyvä vaihtoehto 2D-pelien kehittämiseen.

Myöhemmin aloitettiin projekti kokonaan uudestaan Unity Enginellä ja aloitettiin rakentamalla peli selkeästi ja yksinkertaisesti. Opinnäytetyössä käytetyt sovellukset ja työkalut, kuten Unity Engine, Blender ja Adobe Photoshop olivat ennestään tuttuja ja niiden käytöstä oli osaamista. 3D-mallit saatiin tehtyä Blenderin avulla ja itse pelin luominen onnistui helposti Unityllä. Osa materiaaleista, kuten äänitteet, musiikit ja osa graafisista kuvista jouduttiin oman taidonpuutteen takia ostamaan netistä lisensseineen. Opinnäytetyöprojekti eteni suurimmalta osin tehdyn suunnitelman mukaan. Joitain

ominaisuuksia poistettiin projektista, koska osa niistä aiheuttivat virheitä tai ne ei toiminut halutulla tavalla. Projektiin myös lisättiin uusia ominaisuuksia, jotka vaikuttivat hyviltä ideoilta.

Työn tuloksena saatiin visuaalisesti, ominaisuuksiltaan ja toimivuudeltaan valmis mobiilipeli. Projektista puuttuu vielä uusia pelikenttiä ja hahmoja, jotka tullaan lisäämään myöhemmin osaksi pelin kokonaisuutta. Mobiilipeliä on vielä mahdollista jatkaa tulevaisuudessa lisäämällä lisäominaisuuksia ja korjailemalla käyttäjien kohtaamia ongelmia ja virheitä.

Mobiilipeli on tehty toistaiseksi Android-mobiilijärjestelmille, mutta se on myös mahdollista kääntää iOS-käyttöjärjestelmille. Silloin täytyisi tehdä pelin ohjelmointiin muokkauksia ja julkaisuprosessi Applen App Storeen on erilainen.

Pelinkehitys on pitkä prosessi, jos peliä kehitetään itsenäisesti. Pelinkehityksen esituotannossa tehtiin mobiilipelistä pelisuunnitelma ja tutkittiin työkaluja, joista valittiin sopivat työkalut mobiilipelin kehittämiseen. Tuotantovaiheessa tehtiin prototyyppi pelistä, johon tehtiin pelin päämekaniikat. Prototyyppiä paranneltiin jatkuvasti ja lisättiin uusia ominaisuuksia. Testausvaiheessa tehtiin testejä pelin toimivuudelle ja korjattiin testauksessa ilmenneitä ongelmia. Kun mobiilipeli julkaistaan, alkaa viimeinen vaihe, jälkituotanto, jossa voidaan tehdä korjauksia ja parannuksia mobiilipeliin käyttäjien palautteiden perusteella. Seuraamalla pelinkehityksen vaiheita pystyttiin tekemään tarkka suunnitelma mobiilipelistä ja aloittamaan mobiilipelin kehitys. Pelisuunnitelman avulla pystyttiin helpottamaan mobiilipelin kehitystä.

Pelinkehittäjäksi voi ruveta kuka tahansa, sillä pelinkehityksessä ei tarvitse osata heti alkuun ohjelmointia tai 3D-mallintamista. Aloittelijoille löytyy internetistä paljon ohjeita ja avustuksia pelinkehitykseen. Pelinkehittäjien käytettävissä on useita ilmaisia pelimoottoreita ja muita työkaluja.

Ennen pelinkehitystä, suosittelen luomaan ensimmäisenä pelisuunnitelman kehitettävästä pelistä ja tutkia, mitä työkaluja kannattaisi käyttää. Suosittelen seuraamaan myös

pelinkehityksen vaiheita, varsinkin tuotantovaihetta, sillä se käsittelee itse pelin rakentamista ja sen kulkua.

Lähteet

Bandai Namco. (n.d.). *Tamagotchi*. Haettu 11.5.2022 osoitteesta https://tamagotchi.com/product/tamagotchi-pix/

Blender Foundation. (10.5.2022). *Rendering Introduction*. https://docs.blender.org/manual/en/latest/render/introduction.html

Blender Foundation. (n.d.-a). *Blender About*. Haettu 9.5.2022 osoitteesta https://www.blender.org/about/

Blender Foundation. (n.d.-b). *Blender Features*. Haettu 9.5.2022 osoitteesta https://www.blender.org/features/

Chacon, S. & Straub, B. (2014). *Getting Started. Pro Git, 10–25.* https://git-scm.com/book/en/v2

Clement, J. (7.9.2021). *Number of mobile gaming users worldwide in 2021, by region* [Kuva]. *https://www.statista.com/statistics/512112/number-mobile-gamers-world-by-region/*

Computer Hope. (13.11.2018). *Casual Gaming.* https://www.computerhope.com/jargon/c/casual-gaming.htm

Fisher, C. (1.9.2022). *Stages of Game Development – From Idea To Success.* https://www.gbhbl.com/stages-of-game-development-from-idea-to-success/

Flavell, L. (30.12.2010). *Beginning Blender: Open Source 3D Modeling, Animation, and Game Design.*

Gajsek, D. (26.5.2022). *The Growing Opportunities of Mobile Game Development.* https://circuitstream.com/blog/the-growing-opportunities-of-mobile-game-development/

Game-Ace. (10.12.2021). *Five Key Game Development Stages: A Look Behind The Scenes*. https://game-ace.com/blog/game-development-stages/

GitHub, Inc. (n.d.). *About repositories.* Haettu 10.5.2022 osoitteesta https://docs.github.com/en/repositories/creating-and-managing-repositories/about-repositories

Godot. (n.d.). *Features*. Haettu 23.8.2022 osoitteesta https://godotengine.org/features

Humble Bundle, Inc. (n.d.). *Humble About*. Haettu 10.5.2022 osoitteesta https://www.humblebundle.com/about

Interesting Engineering. (2.11.2016). *How Do Game Engines Work?* https://interestingengineering.com/innovation/how-game-engines-work

Jead. (9.5.2022). *What platforms are supported by Unity?*. https://support.unity.com/hc/en-us/articles/206336795-What-platforms-are-supported-by-Unity-

Katkoff, M. (16.9.2012). *Clash of Clans – the Winning Formula.* https://www.deconstructoroffun.com/blog//2012/09/clash-of-clans-winning-formula.html

King. (12.9.2022). Candy Crush Saga- peli [Kuva]. https://play.google.com/store/apps/details?id=com.king.candycrushsaga&hl=fi&gl=FI

Kramer, W. (2000). What is a Game? http://www.thegamesjournal.com/articles/WhatIsaGame.shtml

Kristianto, D. (25.5.2022). *Kuluttajien kulutukset peleihin maailmanlaajuisesti* [kuva]. *2022 Gaming Spotlight: Mobile Extends Lead Over PC and Console as Gaming Market Hits $222 Billion.* https://www.data.ai/en/insights/mobile-gaming/2022-gaming-spotlight-report/

Lever, N. (2022). *Introduction to the Universal Render Pipeline for advanced Unity creators.* https://resources.unity.com/games/introduction-universal-render-pipeline-for-advanced-unity-creators

Linietsky, J. (14.1.2014). *First public release! Godot news.* https://godotengine.org/article/first-public-release

Moroni, A. (1.7.2022). *Why Mobile Games Are Becoming Increasingly Popular in 2022.* https://www.hardcoredroid.com/why-mobile-games-are-becoming-increasingly-popular-in-2022/

Mozolevskaya, V. (8.2.2021). *6 Key Stages of Game Development: From Concept To Standing Ovation.* *https://kevurugames.com/blog/6-key-stages-of-game-development-from-concept-to-standing-ovation/*

Pickell, D. (15.10.2019). Pelinkehityksen vaiheet [kuva]. The 7 Stages of Game Development. https://www.g2.com/articles/stages-of-game-development

Rădulescu, R. (18.8.2020). *Choosing The Right Game Engine.* https://www.gdquest.com/tutorial/getting-started/learn-to/choosing-a-game-engine/

Stefyn, N. (9.5.2022). *How video games are made: the game development process*. https://www.cgspectrum.com/blog/game-development-process

Supercell. (10.8.2022). *Clash of Clans* [Kuva]. https://play.google.com/store/apps/details?id=com.supercell.clashofclans&hl=fi&gl=US

SYBO Games. (15.9.2022). *Subway Surfers* [Kuva]. https://play.google.com/store/apps/details?id=com.kiloo.subwaysurf&hl=fi&gl=DE

Tyler, D. (25.8.2022). *How to Build Your First Game Design Document.* https://www.gamedesigning.org/learn/game-design-document/

Unity Technologies. (n.d.-a). *Compare Unity plans.* Haettu 9.5.2022 osoitteesta https://store.unity.com/compare-plans

Unity Technologies. (19.8.2022). Model file formats (Version 2021.3). https://docs.unity3d.com/2021.3/Documentation/Manual/3D-formats.html

Unity Technologies. (2021). *Optimize Your Mobile Game Performance.* *https://resources.unity.com/games/unity-e-book-optimize-your-mobile-game-performance*

Unity Technologies. (10.8.2022a). *Scenes* (Version: 2021.3). https://docs.unity3d.com/Manual/CreatingScenes.html

Unity Technologies. (10.8.2022b). *Unity's interface* (Version 2021.3). https://docs.unity3d.com/Manual/UsingTheEditor.html

Wolstenholme, K. (n.d.). *Clash of Clans - Core loop* [Kuva]. *What is a Core Loop in a Mobile Game?* Haettu 23.9.2022 osoitteesta https://risinghighacademy.com/what-is-a-core-loop-in-a-mobile-game/

Liite 1: Unity Lisensointivaihtoehdot. (Unity Technologies, n.d.-a)

	Personal	**Plus**	**Pro**	**Enterprise**
Hinta	Ilmainen	$399/vuosi per käyttäjä	$1800/vuosi per käyttäjä	$4000/kuukausi per 20 käyttäjää
Taloudellinen kelpoisuus	Tulot tai rahoitus ovat alle 100 000 dollaria viimeisen 12kk aikana.	Tulot tai rahoitus ovat alle 200 000 dollaria viimeisen 12kk aikana.	Tulot tai rahoitus ovat yli 200 000 dollaria viimeisen 12kk aikana.	Vähintään 20 käyttäjää. Jos Tulot tai rahoitus ovat yli 200 000 dollaria viimeisen 12kk aikana.
Aloitusnäytön muokkaaminen	Ei	Kyllä	Kyllä	Kyllä
Luo ja ota käyttöön suljetuilla alustoilla	Ei	Ei	Kyllä	Kyllä
Pääsy lähdekoodin	Ei	Ei	Kyllä, lisämaksuilla	Kyllä, lisämaksuilla
Pilvidiagnostiikka	Ei	Kyllä	Kyllä	Kyllä
Ydinanalyysi	Kyllä	Kyllä	kyllä	Kyllä
Tekninen tuki	Ei	Ei	Kyllä	Kyllä
Customer Success Manager- palvelu	Ei	Ei	Ei	Kyllä

	Personal	**Plus**	**Pro**	**Enterprise**
Esisijainen pääsy Unity Success Advisors-palveluun	Ei	Ei	Kyllä	Kyllä
Ensisijainen jono asiakaspalveluun	Ei	Ei	Kyllä	Kyllä
Unity mainokset	Kyllä	Kyllä	Kyllä	Kyllä
Sovelluksen sisäiset ostokset-laajennus	Kyllä	Kyllä	Kyllä	Kyllä