Mobiilipelin luonti ja kohdeyleisöt

HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

Ammattikorkeakoulututkinnon opinnäytetyö

Hämeenlinnan korkeakoulukeskus, Tietojenkäsittely

Kevät, 2020

Tatu Hakala

HAMK
HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

Tietojenkäsittelyn koulutusohjelma
Hämeenlinnan korkeakoulukeskus

Tekijä Tatu Hakala **Vuosi** 2020

Työn nimi Mobiilipelin luonti ja kohdeyleisöjen tutkiminen

Työn ohjaaja/t Lauri Salminen

TIIVISTELMÄ

Opinnäytetyössä tutkittiin mobiilipelien kohdeyleisöjä ja sitä, miten valitulle kohderyhmälle luodaan ja suunnitellaan mahdollisimman sopiva peli. Opinnäytetyössä tehtiin mobiilipeli Android-puhelimille Unity-pelimoottorilla. Opinnäytetyön tavoitteena oli valmistaa peli, jossa kuljetetaan palloa puhelimen gyroskoopilla ja kuljetaan rataa pitkin.

Opinnäytetyön teoriaosuudessa kerrotaan 3D-mallinnuksesta ja videopelien kehitysprosessista. Teoriaosuudessa kerrotaan myös työkaluista, joita opinnäytetyön käytännön osuudessa käytettiin. Tämän lisäksi teoriaosuudessa tutkittiin Android-käyttöjärjestelmää ja sen soveltumista pelialustaksi sekä mobiilipelien kohderyhmiä.

Opinnäytetyön käytännön osuudessa tehtiin mobiilipeli Android puhelimille. Peli oli tarkoitus suunnitella mobiilipelien kohderyhmiä mielessä pitäen. Käytännön osuudessa käydään läpi mobiilipelin luontiprosessia, sen tuloksia ja ongelmia, joita käytännön osuudessa tulli vastaan. Käytännön osuudessa luotiin myös 3D-objekteja peliin Blender 3D-mallinnusohjelmalla. Käytännön osuuden lopussa testattiin valmistunutta peliä ja katsottiin kuinka hyvin se toimi erilaisilla puhelimilla.

Opinnäytetyön lopputuloksena syntyi mobiilipeli, jossa on kaksi pelattavaa hahmoa ja yksi kenttä. Peliä ohjataan puhelimen kiihtyvyysanturilla gyroskoopin sijaan. Pelin ulkoasua ja vaikeustasoa säädettiin, jotta se sopisi mobiilipelien kohdeyleisölle.

Avainsanat Gyroskooppi, Unity, Blender, 3D-mallinnus

Sivut 28

HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

Degree Programme in Business Information Technology
Hämeenlinna University Centre

Author Tatu Hakala **Year** 2020

Subject Mobile game development and researching the target audiences

Supervisors Lauri Salminen

ABSTRACT

The aim of this thesis was to examine the target audiences of mobile games and to design and create a mobile game for the most suitable target audience. The game that was developed in the thesis is an Android game made with Unity game engine. The goal of the thesis was to create a game that can be controlled with the mobile device's gyroscope.

This thesis contains information about 3D modeling and the process of video game development. The thesis also contains information about the tools that were used in the practical section of the thesis. This thesis also studies the target audiences of mobile games and how suitable the Android operating system is as a gaming platform.

In the practical section of the thesis, the goal was to create a mobile game for Android mobile phones. The game was intended to be designed with the target audiences in mind. The practical section of the thesis shows the game development process and the results and problems that was encountered in the project. Moreover, the 3D objects of the game were also created with Blender 3D modeling software. The end results of the game were tested with different mobile devices at the end of the practical section.

A result of the thesis was a simple mobile game with two playable characters and a single level. The game can be controlled with the mobile phone's accelerometer. The game's difficulty and aesthetics were adjusted to fit the mobile gaming target audiences.

Keywords Gyroscope, Unity, Blender, 3D modeling

Pages 28

SANASTO

Pullonkaula = Pullonkaula tarkoittaa osaa järjestelmässä, joka estää järjestelmää toimimaan parhaalla mahdollisella teholla.

Gyroskooppi = Gyroskooppi on laite, jolla voi mitata suuntaa ja kulmannopeutta.

Mikrosysteemi = Mikrosysteemi (englanniksi microelectromechanical systems) ovat erittäin pieniä komponentteja, joita voi esimerkiksi käyttää erilaisina antureina. Mikrosysteemit tunnetaan myös nimellä MEMS.

GameObject = GameObjectit ovat Unityssä käytettäviä objekteja, joita voi käyttää pelissä. GameObjectit, voivat olla esimerkiksi 3D ja 2D -objekteja, efektejä, Videoita, ääniä ja käyttöliittymän osia.

Beta-versio = Beta-versio on videopelin lähes valmis versio, jossa on vielä bugeja ja ongelmia suorituskyvyssä. Pelistudiot voivat joskus julkaista Beta-version pelistään rajatulle yleisölle, saadakseen palautetta pelistä.

Scene= Scenet ovat Unityssä olevia näkymiä, jotka sisältävät pelin valikot ja ympäristöt. Tulen kutsumaan Scenejä näkymiksi.

SISÄLLYS

1 JOHDANTO

Mobiililaitteiden suosio on ollut nousussa jo kauan ja nykyään melkein puolella maailman väestöstä on käytössä jonkinlainen älypuhelin. Tämän takia videopelien luominen mobiililaitteille on erittäin suosittua ja kannattavaa. (Bankmycell.com, 2020)

Tämä opinnäytetyö käsittelee Android-pelin luomista Unity-pelimoottorilla. Opinnäytetyössä tutkitaan minkälaiselle kohdeyleisölle peli kannattaa suunnata, ja mitä pelin suunnittelussa pitää ottaa huomioon, kun se suunnataan kyseiselle kohdeyleisölle. Teoriaosuudessa kerrotaan 3D-pelin luonnista yleisesti, sekä mitä työtehtäviä ja vaiheita siihen kuuluu. Teoriaosuudessa kerrotaan myös opinnäytetyön aikana käytettävistä työkaluista ja hieman niiden historiasta. Tämän lisäksi opinnäytetyössä kerrotaan pelimoottoreista ja kuinka hyvä pelialusta Android on.

Valitsin aiheen, koska käytin Unity-pelimoottoria hieman ICT-projektissa ja pidin sitä mielenkiintoisena aiheena. ICT-projektissa tehtävänä oli luoda AR-sovellus HAMK Smartin Vikke -hanketta varten. Projekti oli melko yksinkertainen tehdä MAXST AR SDK:n avulla ja siinä ei vaadittu juuri ollenkaan ohjelmointia. Tein AR-sovellukselle muutamia yksinkertaisia 3D-objekteja Unityllä, joka sai minut kiinnostumaan myös 3D-mallinnuksesta. Valitsin Androidin pelin alustaksi, koska sille on helpompi tehdä pelejä kuin iOS puhelimille. ICT-projektissa meidän oli tarkoitus luoda myös iOS-versio ja huomasin, että se oli paljon monimutkaisempaa, kuin Android version luonti. IOS-alustalle luontiin tarvitaan tietokone, jossa on macOS-käyttöjärjestelmä. Päätin luoda kuulapelin, koska kuulat eivät vaadi monimutkaisia liikeanimaatioita kuten esimerkiksi ihmishahmot.

Opinnäytetyön käytännön osuudessa tullaan käyttämään Blender 3d-mallinnusohjelmaa ja Unity-pelimoottoria.

Opinnäytetyön tutkimuskysymykset ovat seuraavat:
- Kuinka 3D-pelin luonti Android-puhelimelle onnistuu?
- Minkälaiselle kohdeyleisölle Android peli kannattaa luoda?
- Miten aloittelija voi luoda pelin Unityllä?

2 3D-PELIN LUONTI

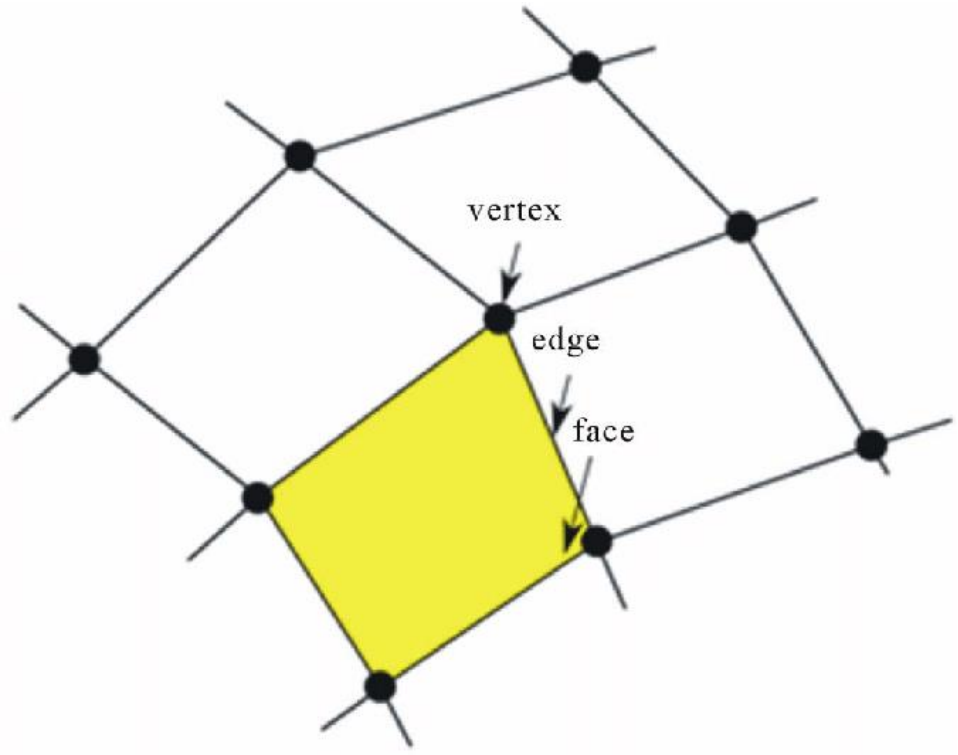
Tässä luvussa kerrotaan 3D-mallinnuksesta, pelinkehityksen vaiheista ja työtehtävistä sekä pelimoottoreista. Kappaleessa käsitellään myös Blender- ja Unity-työkaluja, jotka tulevat olemaan käytössä opinnäytetyön käytännön osuudessa sekä minkälainen Android on pelialustana.

2.1 3D-mallinnus

3D-mallinnus on digitaalisten kolmiulotteisten objektien luontia. 3D-mallinnuksessa polygoniverkko on kärkipisteistä, reunoista ja pinnoista koostuva 3d-objekti. Polygoniverkot muodostavat kaikkien 3D-hahmojen ja -objektien muodon. Suuret polygoniverkot luodaan pienistä toisiinsa yhdistetyistä pinnoista, kuten kolmioista tai suorakaiteista, jotka sopivat yhteen palapelin tavoin. Polygoniverkon jokainen kärkipiste sisältää X, Y ja Z suunta-arvot, joita muuttamalla kärkipisteiden sijaintia voi muuttaa. 3D-objektin yksityiskohtaisuus tarkentuu, mitä enemmän kärkipisteitä siinä on. Polygoniverkon jokainen pinta sisältää tietoa pinnasta, jota renderöintimoottori käyttää muun muassa objektin varjojen ja pintojen heijastuksen laskemiseen. Kuvassa 1 näkyy Polygoniverkko ja sen eri osat. (Petty, n.db)

3D-objektien luontiin on monia eri ohjelmia, kuten Blender, Maya ja 3ds Max. Kaikissa näissä ohjelmissa on tarvittavat työkalut 3D-objektien mallintamiseen, teksturointiin ja animointiin. Näitä ohjelmia on mahdollista käyttää esimerkiksi videopelien tai elokuvien 3D-objektien luomiseen. (Petty, n.db)

Polygoniverkkojen jokaisella pinnalla on etu- ja takaosa. Pintojen etuosia käytetään verkon pintakulman laskemiseen. Pintojen takaosien on tarkoitus olla piilossa. Jos pinnan takaosan tulee näkyviin jostakin syystä, seurauksena on yleensä epäonnistunut renderöinti. Esimerkkinä tästä voisi olla vanhat Nintendo 64 ja Playstation-pelit, joissa nähtiin tällaisia bugeja usein. Verkot sisältävät myös UV-koordinaatteja, joita käytetään tekstuurien näyttämiseen verkon pinnalla. (Petty, n.db)



Kuva 1. Polygoniverkko ja sen osat. (ResearchGate, 2012)

3D-objekteja käytetään esimerkiksi videopeleissä, elokuvissa ja arkkitehtuurissa. Elokuvissa 3D-mallinnusta käytetään erilaisten CGI-erikoistehosteiden luonnissa ja videopeleissä sitä käytetään objektien, hahmojen ja ympäristöjen luomiseen. Arkkitehdit ja insinöörit käyttävät 3D-mallinnusta luomaan kolmiulotteisen suunnitelman omasta työstään. 3D-mallinnusta käytetään myös 3D-tulostukseen, koska se vaatii 3D-mallin tulostamista varten. (Petty, n.da)

2.2 **Blender**

Blender on Blender Foundationin luoma avoimen lähdekoodin ilmainen 3D-mallinnusohjelma. Blenderillä on mahdollista luoda, renderöidä, simuloida ja sommitella 3D-objekteja. Blenderillä voi myös animoida 3D- ja 2D-objekteja sekä editoida videoita. Blender on julkaistu GNU General Public Lisenssillä, joka tarkoittaa, että sitä saa käyttää kuka tahansa, kaikkiin tarkoituksiin. (Blender Foundation, n.da)

Blenderin ensimmäinen versio julkaistiin vuonna 1995. Blender oli alun perin alankomaalaisen animaatiostudio NeoGeon uusi 3d-ohjelmisto. Vuonna 1998 NeoGeon perustaja Ton Roosendaal perusti uuden yrityksen nimeltä Not a Number, jotta hän voisi kehittää ja markkinoida Blenderiä. Not a Number sai kasvurahoitusta useilta sijoitusyhtiöiltä. Not a Numberin tavoitteena oli tehdä Blenderistä ilmainen työkalu 3D-sisällön tuottamiseen ja kaupallinen versio jakelua sekä julkaisemista varten. Vuoden 2002

alussa Not a Numberin sijoittajat päättivät lopettaa yrityksen kaikki toiminnat. Tämä tarkoitti, että Blenderin tukeminen loppui myös. (Blender Foundation, n.db)

Not a Numberin epäonnistumisen jälkeen, Ton Roosendaal perusti voittoa tavoittelemattoman Blender Foundationin toukokuussa 2002. Blender Foundationin tavoite oli jatkaa Blenderin kehittämistä avoimen lähdekoodin ohjelmistona. Heinäkuussa 2002 Ton onnistui saamaan Not a Numberin sijoittajat harkitsemaan tekemään Blenderistä avointa lähdekoodia. Ehtona oli, että jos ”Free Blender” rahankeräys kamppanja saisi kerättyä ainakin 100 000 Euroa, Sijoittajat suostuisivat tekemään Blenderistä avointa lähdekoodia. Monien yllätykseksi rahankeräys keräsi vaaditun 100 000 euron summan seitsemässä viikossa ja lokakuussa 2002 Blender julkaistiin ”GNU General Public Licensen” alla. (Blender Foundation, n.db)

2.3 **Pelinkehityksen työtehtävät**

Pelinkehitykseen ja suunnitteluun kuuluu monia eri vaiheita ja työtehtäviä. Tyypillisessä pelinkehitystiimissä on insinöörejä, artisteja, pelisuunnittelijoita, tuottajia ja muita hallinnon jäseniä sekä tukihenkilöstöä. (Gregory, 2019, s.5)

Insinöörien tehtävänä on suunnitella ja luoda pelin ohjelmistoa. Usein insinöörit jaetaan kahteen eri kategoriaan: ”Runtime programmer”, joka työskentelee pelimoottorin ja pelin itsensä parissa ja työkaluohjelmoija, joka työskentelee työkalujen parissa ja mahdollistaa muun ryhmän tehokkaan työskentelyn. Osa insinööreistä saattaa keskittyä vain yhteen pelin osa alueeseen, kuten tekoälyyn tai fysiikoihin koko uransa aikana. Osa insinööreistä on joka paikan osaajia, jotka ovat osana monissa projektin osuuksissa. Seniori-insinöörit voivat saada ylennyksen tekniseksi johtajaksi. (Gregory, 2019, s.5)

Artistien tehtävänä on luoda pelin visuaalinen ja audiovisuaalinen sisältö. Pelinkehitystiimeissä voi olla monenlaisia eri artisteja. Konseptiartisti luo piirroksia pelistä, joka antaa käsityksen muulle ryhmälle, miltä pelin olisi tarkoitus näyttää. 3D-mallintajat luovat 3D-objekteja peliin. 3D-mallintajat voivat luoda pelin esimerkiksi hahmoja, autoja ja vastaavia pelille tärkeitä objekteja, tai taustaobjekteja kuten puita tai vuoria. Tekstuuriartistit luovat kaksiulotteisia kuvia, jotka voi asettaa 3d objekteihin tekstuuriksi. Valaisuartistit asettavat valoefektit peliin. Animaattorit luovat liikkeet pelin hahmoille ja muille objekteille. Liikkeentallennusnäyttelijät luovat liikedataa, jota animaattorit voivat käyttää luomaan hahmoille animaatioita. Äänisuunnittelijat luovat ääniefektit ja musiikit peliin. Ääninäyttelijät näyttelevät pelin hahmoina. Joissakin peleissä on säveltäjiä, jotka luovat peliin alkuperäisen ääniraidan. (Gregory, 2019, s.6-7)

Pelisuunnittelijoiden tehtävänä on suunnitella pelin interaktiivinen osuus. Pelisuunnittelijat suunnittelevat muun muassa pelin tasoja, mihin viholliset syntyvät ja mistä pelaaja voi löytää tarpeellisia esineitä. Joissakin tapauksissa pelisuunnittelijat saattavat suunnitella pelin tarinaa, missä järjestyksessä pelin eri tasot etenevät ja mitä tavoitteita pelaajalla on. Peli-tiimi saattaa palkata tiimiin myös kirjoittajan, jonka tehtävä voi olla esimerkiksi kirjoittaa pelin tarinaa senioripelisuunnittelijan kanssa, tai pelihahmojen vuorosanojen kirjoittaminen. (Gregory, 2019, s.7)

Tuottajien työtehtävä vaihtelee riippuen pelistudiosta. Joissakin pelistudioissa tuottajan tehtävä on pitää huoli aikataulusta ja olla henkilöstöosaston johdossa. Toisissa pelistudioissa tuottajat toimivat vanhempina pelisuunnittelijoina. Pienemmissä pelistudioissa ei ole välttämättä tuottajia ollenkaan ja kaikki tiimin jäsenet ovat osana pelin kehityksessä. (Gregory, 2019, s.7-8)

Pelin julkaisun ja markkinoinnin hoitaa yleensä julkaisija, joista tunnettuja esimerkkejä ovat Electronic Arts tai Ubisoft. Monet pelistudiot eivät ole sidoksissa vain yhteen julkaisijaan, vaan myyvät pelinsä julkaisijalle, joka tarjoaa parhaan tarjouksen studiolle. Tästä esimerkki on From Software, joka on valmistanut pelejä Sony Interactive Entertainmentin, Bandai Namcon ja Activisionin alla. Toiset pelistudiot taas työskentelevät yksinoikeudella yhden julkaisijan kanssa, joko pitkäaikaisen julkaisusopimuksen kautta tai julkaisijan omistamana tytäryhtiönä. Naughty Dog on esimerkki tällaisesta yrityksestä, koska se on tunnettu lukuisista yksinoikeudella Playstationille suunnitelluista peleistä, kuten Uncharted, Crash Bandicoot ja The Last of Us pelisarjoista. (Gregory, 2019, s.8; From Software, 2020; Naughty Dog, 2020)

2.4 **Pelinkehityksen vaiheet**

Videopelien kehitys on monimutkainen prosessi, mutta sille on kehitetty erilaisia projektinkehitysmetodeja, jotka auttavat yrityksiä hallitsemaan projekteja kustannustehokkaasti. Pelinkehityksen vaiheita ovat suunnittelu, esituotanto, tuotanto, testaus, julkaisuun valmistautuminen, julkaisu ja jälkituotanto. (Pickell, 2019)

Suunnitteluvaiheessa syntyy idea videopelille. Devin Pickellin (2019) mukaan suunnitteluvaiheessa vastataan seuraaviin kysymyksiin:

- Minkälaista videopeliä olemme tekemässä?
- Tuleeko peli olemaan kaksi- vai kolmiulotteinen?
- Mitkä pelin tärkeimmät ominaisuudet?
- Mitä hahmoja pelissä tulee olemaan?
- Mikä on pelin tapahtumapaikka?
- Mikä on pelin kohdeyleisö?
- Mille alustalle pelin on tarkoitus julkaista?

Pelin suunnittelu voi olla yksi pelinkehityksen vaikeimmista vaiheista, koska siinä syntyvä idea tulee olemaan pelin selkäranka. Seuraavaksi suunnitellaan soveltuvuusselvitys. Soveltuvuusselvityksessä otetaan kaikki ideat ja tarkistetaan, kuinka ne soveltuvat pelistudion luotaviksi. Soveltuvuusselvityksessä vastataan kysymyksiin, kuinka paljon pelin valmistaminen tulee maksamaan, onko meillä tarpeeksi tietotaitoa pelin luomiseen ja Kuinka ison tiimin pelinkehittäminen vaatii. (Pickell, 2019)

Esituotantovaiheessa analysoidaan suunnitteluvaiheessa syntyneet ideat ja mietitään miten niitä voisi käyttää hyödyksi. Tässä vaiheessa pelinkehitystiimit tekevät yhteistyötä. Tiimien yhteistyö voi olla muun muassa esimerkiksi kirjoittajien ja projektipäälliköiden tarinankerronnan kehitys, tai insinöörien ja kirjoittajien pelin teknisten rajoitteiden selvitys. Artistit ja suunnittelijat tekevät myös tiimityötä, kun he suunnittelevat pelin visuaalista puolta ja varmistavat, että se on yhdenmukainen. Ohjelmoijat ja insinöörit määrittelevät, miten pelin fysiikat ja pelimekaniikat toimivat. Tämän lisäksi projektipäälliköt keskustelevat kaikkien tiimien kanssa siitä, kuinka hauskaa peliä on pelata. Pelin vaikeutta on vaikea arvioida vielä tässä vaiheessa pelinkehitysprosessia. Tässä vaiheessa on tyypillistä luoda prototyyppejä pelihahmoista, peliympäristöistä ja muista pelin osista. Prototyypit auttavat kuvaamaan, miltä peli näyttää, tuntuu ja miten pelin osat toimivat keskenään. (Pickell, 2019)

Tuotantovaihe on yksi videopelinkehitysprosessin haastavimmista vaiheista. Suurin osa pelinkehityksen resursseista ja ajasta kuluu tuotantovaiheessa. Tuotantovaiheessa luodaan pelin hahmot ja ympäristöt siten, että ne sopivat pelin tarinaan. Tässä vaiheessa luodaan peliin äänimaailma, joka sisältää ääniä erilaisille pelissä tapahtuville asioille. Ohjelmoijat kirjoittavat tuhansia rivejä koodeja tämän vaiheen aikana, jotta pelin sisällöstä tulisi mahdollisimman eloisaa. Projektipäälliköt pitävät huolen tiimien aikataulusta tämän vaiheen aikana asettamalla tiimeille tavoitteita ja sprinttipalavereita. Nämä tehtävät voivat viedä erittäin kauan ennen kuin ne saavuttavat halutun lopputuloksen ja projektin aikana suunnitelmiin tulee todennäköisesti muutoksia, jotka pidentävät vaihetta vielä enemmän. (Pickell, 2019)

Testausvaiheessa testataan pelin jokainen ominaisuus ja mekaniikka sekä tarkistetaan mitä korjattavaa pelissä vielä on. Pelintestaajat tarkistavat tässä vaiheessa, mitä bugeja pelissä on, voiko pelin ominaisuuksia käyttää väärin, onko pelihahmolla mahdollista jäädä jumiin tai mennä objektien läpi. Osa pelisuunnittelijoista keskittyy enemmän pelin hauskuuteen testaamalla pelin vaikeutta ja pelaamalla pelin kokonaan läpi nähdäkseen kuinka tyydyttävä peli on kokonaisuutena. Testaus- ja tuotantovaihe eivät toimi lineaarisesti sen sijaan vaihe vaihtelee testauksen ja tuotannon välillä, kunnes pelin bugit on korjattu ja peli on tarpeeksi hauska testaajien mielestä. (Pickell, 2019)

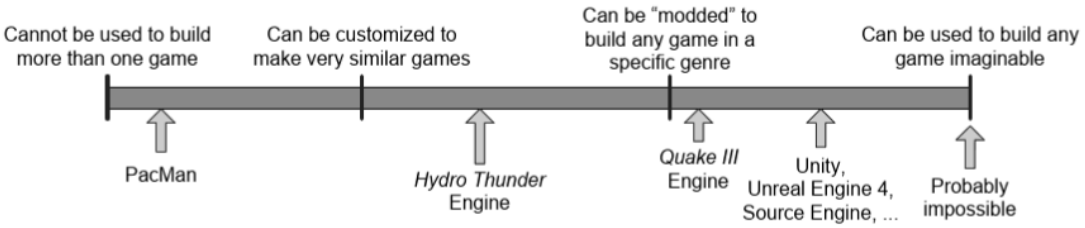
Julkaisuun valmistautuminen on vaihe, jossa peliä aletaan markkinoida ja peliä näytetään yleisölle ensimmäistä kertaa. Julkaisuun valmistautumisen aikana pelistudiot usein antavat Beta-versioita pelistä esimerkiksi peliyhteisön vaikuttajille, kuten Youtube- tai Twitch-sisällönluojille, jotta he pelaisivat peliä omalle yleisölleen. Isompien pelistudioiden julkaisijat voivat haluta pelistä mainosvideon, joka on sekoitus tietokoneanimaatiota ja videopeliä. Julkaisija voi joskus myös varata paikan videopelimessuilta, jossa messuilla olevat ihmiset saavat pelata peliä ennen pelin julkaisua. (Pickell, 2019)

Kun peli on lähes valmis, pelinkehitys siirtyy julkaisuvaiheeseen, jossa viimeisimmätkin bugit yritetään saada korjattua ja peliin tehdään vielä viime hetken parannuksia. Kun bugit on korjattu ja peliä on hiottu tarpeeksi, peli julkaistaan ja pistetään myyntiin. Pelin julkaisun jälkeen pelistudio siirtyy jälkituotantovaiheeseen, jossa korjataan pelin bugeja, joita tiimi ei huomannut testausvaiheessa. Pelistudiot saavat yleensä bugiraportteja pelaajilta joko bugiraporttien tai nettifoorumien kautta. Toinen yleinen tehtävä jälkituotannossa on julkaista peliin lisäsisältöä, joko maksullisena tai ilmaiseksi. Pelin lisäsisältö lisää pelin uudelleenpeluuarvoa, jonka takia nykypäivän pelistudiot useasti päättävät julkaista lisäsisältöä peleihinsä. (Pickell, 2019)

2.5 **Pelimoottori**

Pelimoottorit ovat ohjelmistoja, joissa on pelin kehitykseen olennaisia ominaisuuksia. Pelimoottori tekee pelien kehittämisestä nopeaa ja tehokasta. Pelimoottoriin voi tuoda eri alustoilta sisältöä, kuten esimerkiksi 3d objekteja 3D-mallinnusohjelma Blenderistä. Pelimoottorilla peliin voi lisätä helposti erilaisia peleissä olennaisia asioita kuten valotehosteita tai fysiikoita objekteille. (Unity Technologies, n.dc)

Pelimoottori terminä syntyi 90-luvulla, kun id Softwaren Doom julkaistiin. Doom oli luotu siten, että pelin eri ohjelmistokomponentit (kuten esimerkiksi audio, törmäyksen tunnistus ja 3D grafiikat) sekä pelin taide resurssit, pelikartat ja pelin säännöt oli määritelty eri osa alueisiin. Muut pelien tekijät näkivät id Softwaren innovaation hyödyt ja alkoivat lisensoida pelejä ja uudelleenjulkaisemaan vanhoja pelejä luomalla uutta taidetta, karttoja ja sääntöjä. Kuva 2 näyttää, miten eri pelimoottorit eroavat muokattavuudeltaan ja uudelleenkäytettävyydeltä. (Gregory, 2019, s.11)



Kuva 2. Pelimoottorien muokattavuus ja uudelleenkäytettävyys. (Gregory, 2019)

Pelimoottoreita on monia ja osa niistä on vapaasti kenen tahansa käytettävissä joissakin tapauksissa jopa ilmaiseksi(ei-kaupallisesti). Unity-pelimoottori ja Unreal Engine ovat esimerkiksi ilmaisia Ei-kaupalliseen käyttöön, mutta esimerkiksi Electronics Artsin Frostbite-moottoria ei voi käyttää kukaan EA:n ulkopuolella. (Epic Games, n.d)

2.6 **Unity**

Unity on pelimoottori, jota voi käyttää ilmaiseksi ei-kaupalliseen käyttöön. Kaupalliseen käyttöön Unity maksaa yksityishenkilöille 15 Euroa kuukaudessa ja yrityksille, joko 40 tai 150 euroa kuukaudessa. Unityllä voi tehdä pelejä monille eri alustoille, kuten Androidille, iOS:ille, PC:lle ja konsoleille. Unityllä voi tehdä 2D sekä 3D pelejä ja sillä voi tehdä myös VR/AR pelejä. Unity editorin saa ladattua Windows, Mac OS X, Ubuntu ja CentOS alustoille. (Unity Technologies, n.db; Unity technologies, n.de)

Unity Technologiesin perustivat David Helgason, Joachim Ante ja Nicholas Francis. Unityn ensimmäinen version julkaistiin vuonna 2005 Ja se julkaistiin Windows tietokoneille sekä internet selaimille. Unityn suosio lähti nousuun vuonna 2008, kun Apple julkaisi mobiilisovellus markkinapaikka App Storen. Unity oli ensimmäinen iPhonea tukeva pelimoottori, joka sai monet mobiilipelikehittäjät kiinnostumaan Unitystä. (Brodkin, 2013.)

Unityn viimeisin version opinnäytetyön kirjoittamisen aikana on versio 2019.2 ja se julkaistiin syksyllä 2019. Tässä versiossa tuli parannuksia ja uusia ominaisuuksia muun muassa ohjelmointi- ja suunnittelutyökaluihin. (Unity technologies, 2019)

Taulukko 1. Osa Unity 2019.2 version uusista ominaisuuksista. (Unity technologies, 2019.)

Kategoria	**Ominaisuuden nimi**	**Selitys**
Suunnittelu	ProBuilder 4	ProBuilder on 3D mallinnus ja tason suunnittelutyökalu.
Suunnittelu	Polybrush	Polybrush on työkalu, jolla käyttäjä voi helposti muotoilla 3D-objekteja ”brush”-työkalulla.
Ohjelmointi	iOS ja android parannuksia	iOS ja Android versioiden kirkkautta voi nyt säätää ja käyttöliittymään voi lisätä paikan puhelimien kameralovelle/reiälle.
Ohjelmointi	PhysX Cloth Solver	PhysX 4.x tuo parempilaatuiset kangasfysiikat Unityyn. PhysX 4.x parantaa myös kankaiden suorituskykyä.
Ohjelmointi	Intel® VTune™ Amplifier support	Tämä ominaisuus auttaa käyttäjää löytämään suorituskyvyn pullonkaulat. VTunen avulla käyttäjä voi analysoida koodia, eristää ongelmat ja optimoida suorituskykyä moderneille prosessoreille.
Grafiikat	Shadow Layers	Tämä ominaisuus mahdollistaa varjojen lisäämisen peli objekteille, vaikka objekti ei vastaanota valoa. Tämä mahdollistaa esimerkiksi varjojen värin muuttamisen.

Unityn uusin LTS eli “Long Term Support” versio 2018.4.15f1 julkaistiin kymmenes tammikuuta 2020. Unityn LTS-versiot ovat Unityn versioita, joita tuetaan pitkään. Jokaista Unityn LTS-versiota tuetaan kaksi vuotta ja ne sopivat hyvin käyttäjille, jotka haluavat käyttää yhtä Unityn versiota pi-

demmän aikaa. LTS-versioissa ei ole uusimpia Unityn ominaisuuksia ja parannuksia, mutta niissä pyritään korjaamaan Unityn isommat ongelmat, kuten ohjelman kaatuminen. (Unity technologies, n.dd)

2.7 **Android pelialustana**

Android on Googlen omistama käyttöjärjestelmä, jota käytetään monissa eri laitealustoissa. Androidia käytetään eniten puhelinten ja tablettien käyttöjärjestelmänä, mutta sitä voi käyttää myös esimerkiksi televisioissa ja älykelloissa. Androidilla on yli 70 prosentin markkinaosuus maailman mobiilikäyttöjärjestelmistä, kun taas iOS:in markkinaosuus on yli 20 prosenttia. Android on selkeästi maailman käytetyin mobiilikäyttöjärjestelmä, jonka takia se on suosittu, myös ohjelmoijien keskuudessa. (Statcounter, 2019; Roy, 2016)

Nykypäivän puhelimissa on erilaisia mikrosysteemikomponentteja, joiden avulla matkapuhelimilla voi mitata muun muassa puhelimen suuntaa ja suuntautumista. Gyroskooppi on yksi puhelimissa usein käytetyistä mikrokomponenteista. Gyroskooppia käytetään objektien rotaation mittaamiseen. Alun perin gyroskoopit toimivat pyörimällä kuten hyrrä. Tämä mahdollisti gyroskoopin pitämään oikean suuntautumisen. Modernit gyroskoopit toimivat värähtelytekniikalla, joka tekee niistä kestävämpiä vähäisemmän kitkan takia. Tämä mahdollistaa sen, että gyroskoopeista voidaan tehdä mikrosysteemikomponentteja. Tämän ansiosta lähes kaikissa nykypuhelimissa on gyroskooppi puhelimen rotaation mittaamiseen. Puhelimen gyroskooppia voidaan käyttää peleissä, tästä esimerkkinä on muun muassa Pokémon Go, joka käyttää gyroskooppia ja muita sensoreita havaitakseen pelaajan sijainnin maailmassa. (Swing, 2017, s 4-5)

Android 10 on viimeisin Android-versio opinnäytetyön kirjoittamisen aikana. Android 10 julkaistiin kolmas syyskuuta 2019. Android 10 tuli saataville ensimmäisenä osaan Googlen, OnePlussan ja Essentialin valmistamista puhelimista ja muiden valmistajien puhelimiin myöhemmin julkaisun jälkeen. Android 10 uusia ominaisuuksia ovat muun muassa tuki taittuvanäyttöisille puhelimille, tuki 5G-verkoille, tumma teema, joka säästää puhelimen akkua ja on ideaalinen hämärissä olosuhteissa. Android 10:ssä on myös tietoturvaan liittyviä päivityksiä, kuten puhelimen muistin salaus ja tietoturvapäivitysten siirtyminen Google Play Storeen, mikä tarkoittaa, että puhelimien valmistajien sijaan päivitykset julkaisee Google. Tämä tarkoittaa, että tietoturvapäivitykset tulevat saataville kaikille Android 10-puhelimille samaan aikaan. (Google, 2019; Android Developers, 2020; Android Authority, 2020)

Android on yksi maailman suurimmista pelialustoista, koska Android-puhelimiin on valmiiksi asennettuna Google Play -sovelluskauppa. Google Play -kaupan bruttotulot olivat vuonna 2019 yli 29 miljardia dollaria. Vuonna 2016 bruttotulot olivat 15 miljardia ja 2018 24 miljardia. Google

Play kauppa on siis kasvava sovellusalusta. Vuonna 2016 Google Playn tuloista 90% tuli videopeleistä. (Statista, 2020; Android Authority 2017)

2.8 **C#**

C# on Microsoftin kehittämä ohjelmointikieli, jonka ensimmäinen versio julkaistiin vuonna 2002. C# on helppokäyttöinen ohjelmointikieli, joka on osa Microsoftin .NET-ohjelmistokomponenttikirjastoa. C# on oliopohjainen ohjelmointikieli, joten sen syntaksi on helppo oppia, jos käyttäjällä on jo kokemusta olio-ohjelmoinnista. (Ky, J, 2013, s1; Microsoft, 2017)

C#-ohjelmointikieltä käytetään Unityn ohjelmointikielenä. C# koodeilla voidaan muun muassa ohjata ja hallita pelin objektien käyttäytymistä. Ohjelmointi Unityssä poikkeaa puhtaasta ohjelmoinnista, koska Unityä käyttäessä ei tarvitse luoda käynnistyvää ohjelmaa itse, koska Unity tekee sen automaattisesti. Unity suorittaa koodeja silmukassa, joka tarkoittaa, että koodit suoritetaan jatkuvasti ohjelman päällä ollessa. Mitä korkeampi ruudunpäivitys tarkoittaa että, Unity suorittaa koodin useammin ja tämä tarkoittaa esimerkiksi parempaa ohjattavuutta pelissä. Unity-pelimoottorissa koodit pitää asettaa peliobjekteihin, jotta Unity osaa suorittaa ne. Unity-koodeissa on kolme pääosaa: luokat, funktiot ja muuttujat. (Unity Technologies, n.da)

Muuttujia käytetään Unityn koodeissa työkaluina. Muuttuja voi olla esimerkiksi peliobjekti tai peliobjektin komponentti. Muuttujilla on useita eri näkyvyystyyppejä, mutta yleisimmin näkyvyystyyppinä käytetään joko julkista tai yksityistä näkyvyyttä. Julkista muuttujaa voi muokata Unityn puolella, mutta yksityistä muuttujaa voi hallita vain sen koodin ja luokan sisällä missä se itse on. Yksityiset muuttujat tekevät koodin virheenkorjauksesta helpompaa, koska muuttujan arvo voi tulla vain yhdestä tietystä luokasta. Julkinen muuttuja on hyödyllinen silloin, kun pelin objektien on kommunikoitava muiden objektien kanssa. Muuttujan tyyppi on tärkeä osa muuttujaa. Muuttujan tyyppi voi olla esimerkiksi numero tai tekstiä. Muuttujan tyyppi voi olla myös esimerkiksi peliobjektin komponentti, jonka avulla komponenttia voi hallita koodin avulla. (Unity Technologies, n.da)

Koodit käsittelevät muuttujia funktioiden avulla. C#:ssa on monia funktioita, jotka toimivat automaattisesti Unityn sisällä. Awake-funktio aktivoituu, kun peliobjekti syntyy. Start-funktio aktivoituu, pelin ensimmäisellä ruudulla. Update-funktio aktivoituu joka ruudulla. Jos ruudunpäivitys on esimerkiksi 60, Update-funktio aktivoituu 60 kertaa sekunnissa. FixedUpdate-funktio on hyödyllinen pelin fysiikoiden luonnissa. LateUpdate-funktio on samanlainen kuin Update-funktio, mutta funktio aktivoituu ruudun lopussa. (Unity Technologies, n.da)

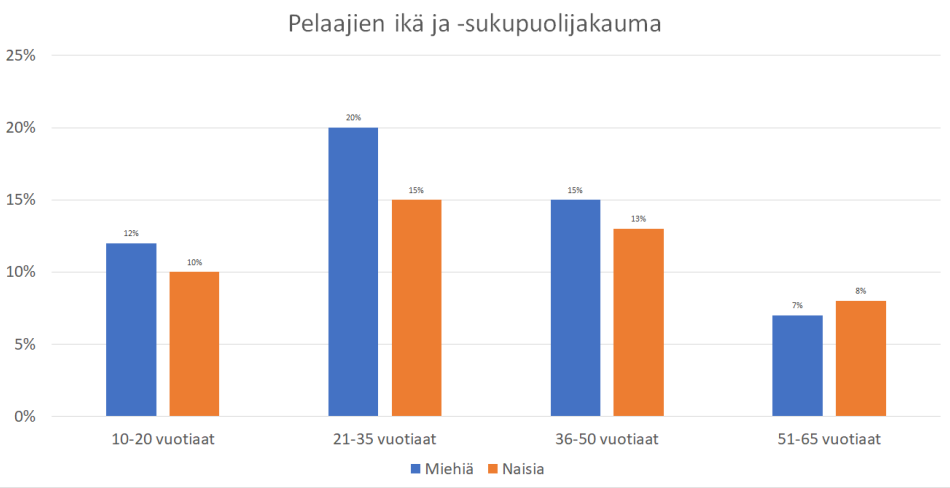
Luokat ovat funktioiden ja muuttujien kokoelmia. Luokan on oltava samanniminen kuin se kooditiedosto, jonka sisällä se on. Jotta luokka toimisi peliobjektin kanssa, sen on oltava peräisin Monobehaviour luokasta. (Unity Technologies, n.da)

3 KOHDERYHMÄT

3.1 Mobiilipelien kohderyhmät

Mobiililaitteet ovat suurin alusta videopeleille. Niiden osuus videopeliteollisuuden tuotoista oli 51% vuonna 2018. Vuosien 2014-2018 mobiilipuhelimen päivittäisen käytön keskivertoaika nousi 152 minuutista 215 minuuttiin. Puhelimien keskiverto käyttöajan oletetaan nousevan noin 234 minuuttiin vuoteen 2021 mennessä. Mobiilipeleillä on suuri vaikutus mobiililaitteiden suosion kasvuun. Pohjois-Amerikassa mobiilipeliteollisuus oli arvoltaan 9,8 miljardia dollaria vuonna 2018 ja Aasiassa mobiilipeliteollisuuden tulot olivat vuonna 2018 41,5 miljardia dollaria. (Gough, 2019b)

Vuonna 2012 Candy Crush Saga oli yksi maailman suurimmista mobiilipeleistä. Vuonna 2018 Candy Crush ja sen lukuisat jatko-osat keräsivät yli miljardi dollaria pelaajien ostamien mikromaksujen avulla. Toinen mobiilipeli menestys viime vuosilta on Pokémon Go, joka saa kuukausittain yli 10-miljoonaa uutta pelaajaa Yhdysvalloissa ja se tuottaa miljoonia dollareita päivässä. (Gough, 2019b)



Kuva 3. Videopelien käyttäjäryhmien ikä ja -sukupuolijakauma vuonna 2017. Tiedot on saatu Statistan “Distribution of video gamers worldwide in 2017, by age group and gender” kaaviosta. (Statista, 2019)

Kuten kuvasta 3 näkee videopelit ovat suosituimpia nuorten aikuisten keskuudessa. Pelaajien sukupuolijakauma on 54% miehiä ja 46% naisia. Vaikka sukupuolijakauma on melko tasan miesten ja naisten välillä, sukupuolten edustajat pelaavat silti erilaisia pelejä. Esimerkiksi PlayerUnknown’s Battlegrounds peliä pelasivat pääasiassa miehet ja naiset suosivat enemmän

pelejä kuten esimerkiksi Candy Crush Sagaa. WePC:n keräämän statistiikan mukaan miesten motivaatio videopelien pelaamiseen on kilpailu ja naisten motivaatio syntyy pelin läpäisystä. 51-65-vuotiaat pelaavat videopelejä vähiten. Kaavio luotiin kyselyn perusteella, johon vastasi 10-65-vuotiaita pelaajia kolmestatoista eri maasta. Vastaajien määrää ei kerrottu. (Gough, 2019a; WePC, 2020)

Yksi videopelisuunnittelun vaikeimmista asioista on vaikeustason säätäminen sopivaksi. Vaikeustason olisi hyvä olla tarpeeksi vaikea, jotta pelaaja saisi pelistä sopivasti haastetta, mutta liian vaikea peli voi saada pelaajan turhautumaan ja lopettamaan pelin. Videopelin vaikeustasoon ei ole selkeää määritelmää, joten vaikeustason suunnittelu on subjektiivinen prosessi. Vaikeustason suunnittelussa pelisuunnittelijat luovat sarjan haasteita ja muuttavat sen jälkeen pelin parametrejä vastaamaan pelin haluttua vaikeustasoa. Jotta pelin vaikeustasosta saataisiin oikeanlainen, pelisuunnittelijat yleensä tekevät paljon pelin testausta. Pelisuunnittelijoiden tekemä testaus on yleensä erittäin paljon aikaa vaativaa työtä, koska pelisuunnittelijoiden on usein vaikea arvioida vaikeustasoa peliin, jota he ovat olleet mukana tekemässä tuntikausia. (Aponte, Levieux, Natkin, 2011, s.1)

Videopelin vaikeustaso riippuu paljon pelaajan taidoista tai kyvystä suorittaa edessä oleva haaste. Videopelien vaikeustason on yleensä hyvä alkaa helposta ja hiljalleen nousta vaikeammaksi, jotta jopa uudempienkin pelaajien taidot kasvavat jokaisen haasteen jälkeen ja he saavat iloa haasteen päihityksestä. Vaikeustasosta on myös mahdollista tehdä skaalautuva, mikä tarkoitta, että peli tunnistaa kuinka hyvin pelaaja pelaa peliä ja osaa joko helpottaa tai vaikeuttaa peliä. (Aponte, 2011, s.4)

4 TUTKIMUSMENETELMÄT

Opinnäytetyön käytännönosuus tulee olemaan toiminnallinen tuotekehityshanke, koska opinnäytetyössä kehitetään ja suunnitellaan uusi videopeli Android-alustalle. Käytännönosuudessa tutkitaan, kuinka helppoa Unityllä on luoda videopeli Android-puhelimelle.

Käytännönosuudessa tarkoituksena on selvittää, kuinka 3D-pelin luonti onnistuu Android puhelimelle ja kuinka paljon tehoa peli vaatii. Työssä selvitetään myös, miten pelin luonti sujuu aloittelijalta Unity-pelimoottorilla. Kolmantena tavoitteena työssä on tutkia mobiilipelien kohdeyleisöä ja selvittää minkälainen peli niille kannattaa suunnitella.

Ensimmäistä kysymystä tutkitaan siten, että teen pelin Unitylle ja katson, miten pelin luonti onnistui. Raportoin opinnäytetyöhön, mikä pelissä onnistui ja missä asioissa tuli ongelmia. Toiseen kysymykseen tulee vastaus sen jälkeen, kun peli on valmistunut, koska minulla ei ole paljoa kokemusta Unity-pelimoottorista. Kolmanteen kysymykseen vastaus tulee esille osittain teoriaosuudessa ja osittain käytännönosuudesta. Teoriaosuudessa tutkin mobiilipelien kohdeyleisöjä sekä pelin vaikeustason säätämistä ja käytännönosuudessa yritän soveltaa näitä tietoja oman pelin tekemisessä.

Tutkimusaineistona työssä käytettiin muun muassa videopelien kehitykseen liittyviä E-kirjoja sekä verkkosivustoja. Myös opinnäytetyössä käytettyjen työkalujen omat internet sivut toimivat työssä lähteinä. Lähteinä on pyritty käyttämään mahdollisimman tuoreita ja luotettavia lähteitä, mutta joistakin aiheista, kuten Unityn historiasta ei löytynyt tuoretta lähdettä.

Käytännön osuuteen pyrittiin kirjoittamaan työssä edistyneet kohdat ja ongelmat aina päivän päätteeksi. Käytännön osuudessa ei kirjoiteta jokaisesta koodista mitä projektin aikana syntyi, koska se tekisi työstä liian tylsää ja yksitoikkoista luettavaa. Käytännön osuudessa kuvia on käytetty melko paljon. Kuvia käytetään pääosin 3D-objektien näyttämiseen lukijalle.

5 KUULA RACER

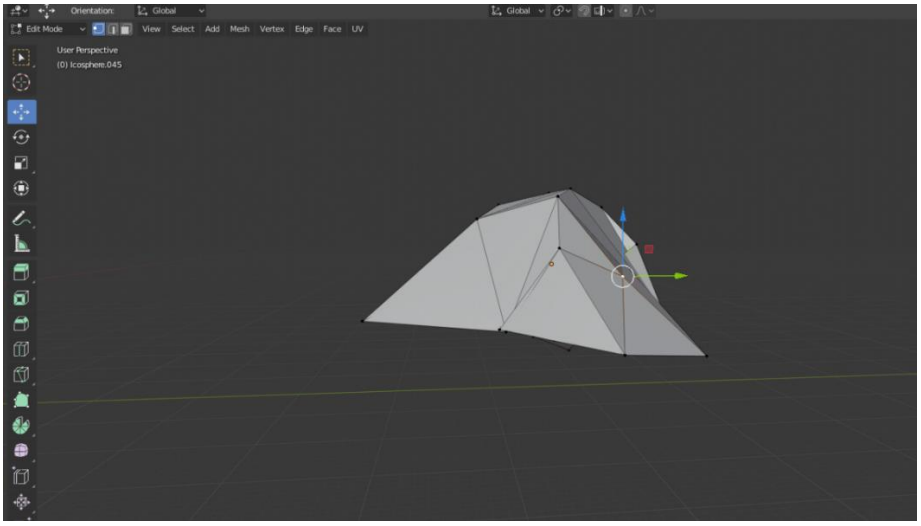
Opinnäytetyön käytännönosuudessa luodaan Android-peli Unity-pelimoottorilla. Pelissä on tarkoituksena ohjata kuulaa rataa pitkin ja saada mahdollisimman nopea aika. Kuulaa ohjataan puhelimen gyroskooppi ominaisuudella ja kuulalla voi hypätä klikkaamalla puhelimen näyttöä. Pelissä vapaaehtoisena tavoitteena pelaaja voi yrittää kerätä kaikki kolikot radalta. Taulukossa 2 näkyy puhelimet, joilla pelin testaus tulee tapahtumaan. Testattavana on siis viisi puhelinta, joiden tehokkuus ja ikä vaihtelevat. Testipuhelimet tulevat antamaan hyvän kuvan siitä, kuinka peli tulee toimimaan uudemmilla ja vanhemmilla puhelimilla.

Taulukko 2. Puhelimet, joilla peliä tullaan testaamaan.

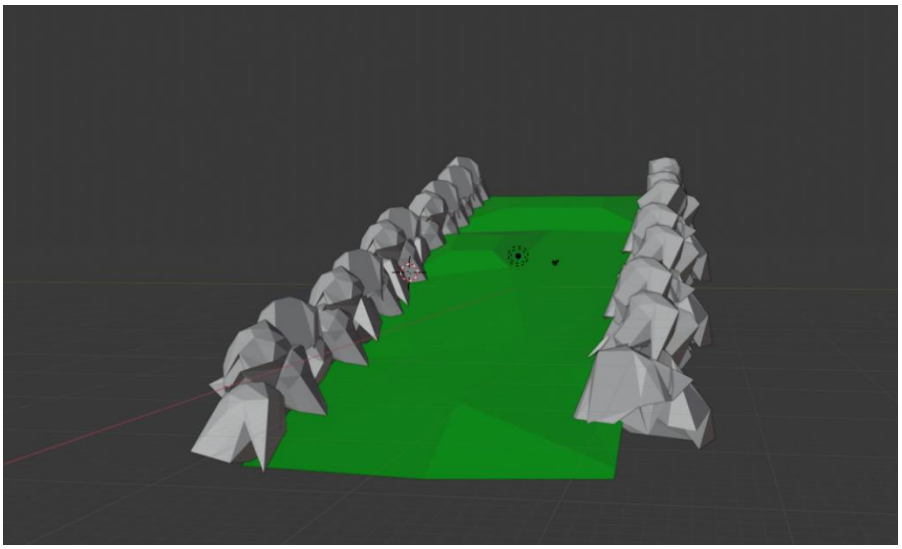
Puhelin	Ram-muisti	Järjestelmäpiiri	Android-versio
Asus Zenfone 3	4Gt	Qualcomm Snapdragon 625	Android 8 Oreo
OnePlus 7 Pro	8Gt	Qualcomn Snapdragon 855	Android 10
Samsung Galaxy J5	1,5Gt	Qualcomm Snapdragon 410	Android 7 Nougat
Samsung Galaxy S3	1Gt	Exynos 4412	Android 4.4 KitKat
Xiaomi Mi 9T	6Gt	Qualcomm Snapdragon 730	Android 10

5.1 3D-objektit

3D-objektien luonti tehtiin Blender 3D-mallinnusohjelmalla. Ensimmäiseksi luotiin 3D-objekti, joka toimii osana pelin rataa. Objekti on nurmikkoinen polku, jossa on kalliota reunoilla, jotta kuula ei voi poiketa polulta. Objektin luominen alkoi vaakasuoran tason luonnista, joka tulee toimimaan objektin nurmikkona. Tavallisen nurmikon lisäksi, peliin luotiin nurmikosta töyssyisen version lisäämällä objektiin lisää kulmia ”Insert Faces” -työkalulla ja nostamalla osaa kulmista hieman ylös. Sen jälkeen luotiin kiviä muokkaamalla ”Ico sphere” objektia näyttämään enemmän kiven muotoiselta kuten kuvassa 4 näkyy. Kiviobjektien luomisen jälkeen kivet aseteltiin nurmikon reunoille, jotta siitä syntyisi rata, jota voi käyttää pelissä. Kuvassa 5 näkyy valmistunut rata.



Kuva 4. Kiven luominen Blenderissä.

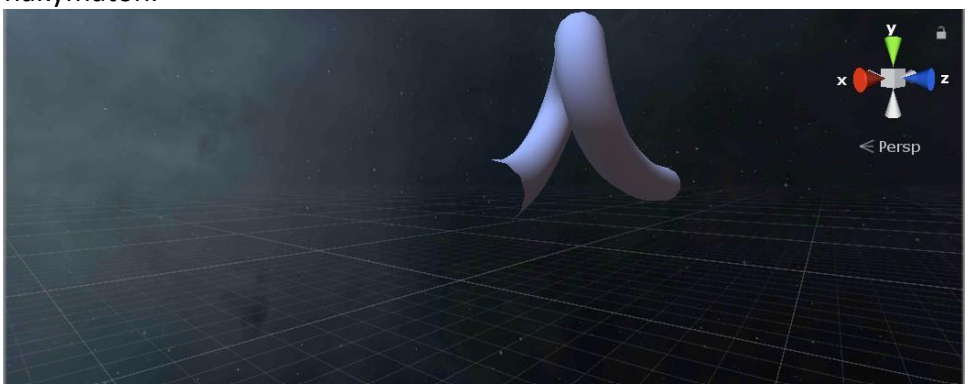


Kuva 5. Mäki, jossa on töyssyjä. Luotu Blenderillä.

Seuraavaksi peliin luotiin kolikko-objekti Blenderillä. Kolikosta suunniteltiin salmiakin muotoisen keltaisen objekti, jonka keskellä on dollarimerkki. Kuten kuvassa 6 näkyy, kolikko-objektin luonti oli melko simppeliä, koska se on vain litistetty kuutio, jonka kylkeen lisättiin tekstityökalulla dollarimerkki. Kolikosta tehtiin hieman metallisemman näköisen Unityn puolella, jotta siitä tulisi kultaisen näköinen.

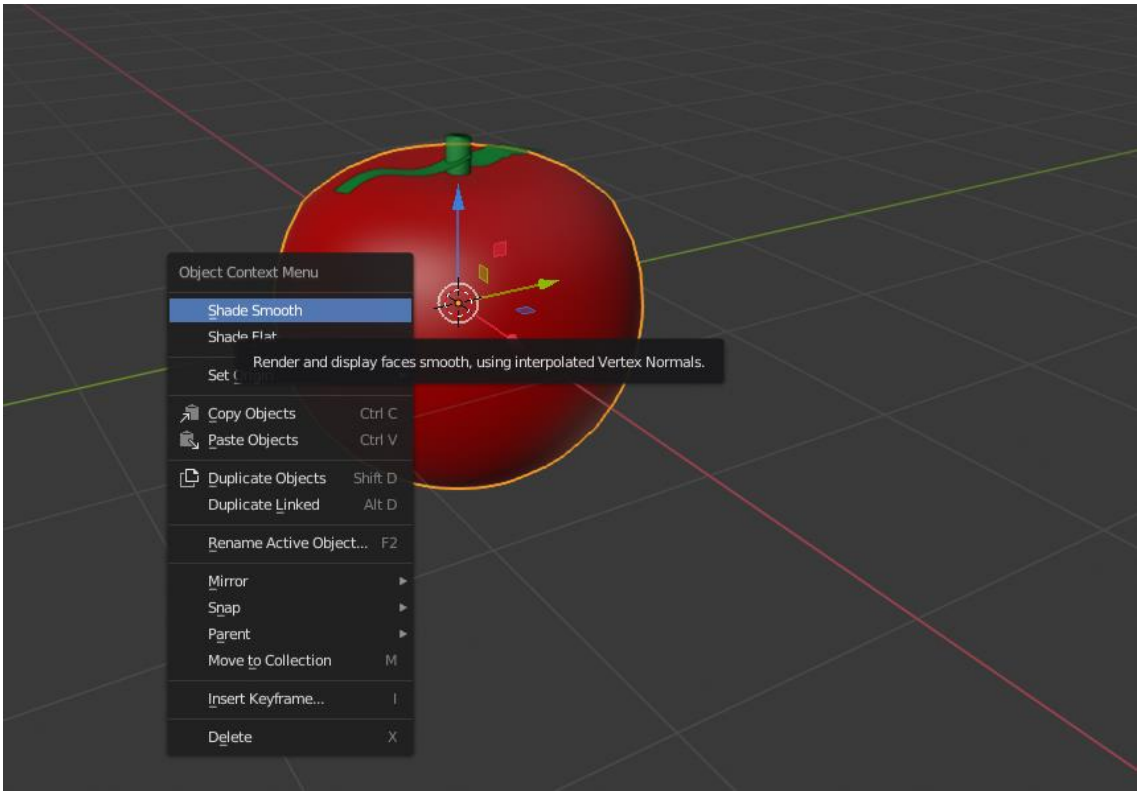
Kuva 6. Blenderillä luoty kolikko.

Kuten kuvassa 7 näkyy, Blenderin kanssa tuli myös ongelmia. Peliin luotiin Blenderillä putki luomalla NurbsPath-objekti ja asettamalla sen “Bevel depth” asetus 0,5 metriin. Putki näytti siltä kuin sen pitäisikin Blenderissä, mutta kun sen siirsi Unityn puolelle putken sisäosa oli syystä tai toisesta näkymätön.



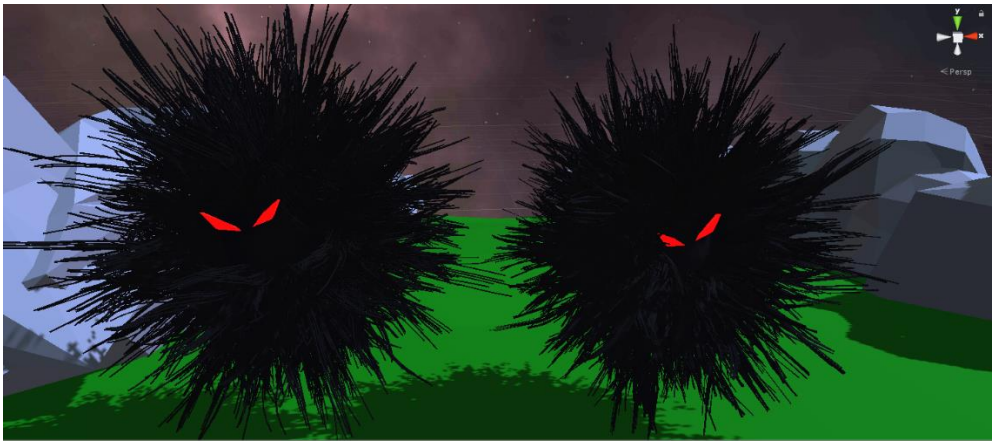
Kuva 7. Blenderissä luotu putki, kun se on siirretty Unityyn.

Peliin luotiin Blenderillä toinen pelattava hahmo. Toisena pelihahmona toimii tomaatti. Tomaatin luonti aloitettiin luomalla ensin pallo ja litistämällä sen kärki. Sen jälkeen pallon päälle lisättiin kaksi BezierCurve-objektia ja sylinteri, joista muodostui tomaatin lehdet. Tämän jälkeen objektin kulmia pehmennettiin käyttämällä objektin ”Shade smooth” asetusta. Kuten kuvassa 8 näkyy, tomaatin luonti onnistui melko hyvin.



Kuva 8. Tomaatti 3D-objekti ja Shade Smooth -työkalu.

Pelissä on tarkoitus olla ainakin yksi vihollinen, joka vahingoittaa pelaajaa, kun se on kosketuksessa häneen. Vihollisesta suunniteltiin musta karvapallo, jolla on punaiset silmät ja suu. Hahmon luonti aloitettiin luomalla pallo ja lisäämällä siihen hiuspartikkeleita. Partikkeliasetuksia säädettiin, jotta hiuksista tulisi luonnollisemman näköiset. ”Brownian” -asetuksen avulla muutettiin hiusten suoruutta. ”Children” -asetuksilla säädettiin, kuinka paljon yhdellä hiuksella on lapsia, eli toisin sanoen, kuinka tuuhea hiuksista tulee. Kun objekti siirrettiin Unityn puolelle ensimmäistä kertaa, objektin hiukset eivät tulleet näkyviin. Ongelma ratkesi, kun objekti muunnettiin Blenderissä "Curve" muotoon, asetettiin "Fill Deformed" -asetus päälle ja lisättiin objektin "Bevel Depth" -asetusta hieman. Tämän jälkeen objekti muutettiin "Mesh" muotoon, jonka jälkeen se toimi myös Unityn puolella. Unityllä luotiin uusi materiaali, joka on punainen ja siinä on Emission efekti. Tämä materiaali asetettiin hirviön silmiin, joka teki niistä erittäin punaiset. Kun hirviötä testattiin puhelimella, ilmeni että peli pyöri todella huonosti. Hirviötä pitäisi siis muokata vähemmän tehoa vaativaksi. Hirviön vietiin takaisin Blenderiin ja sitä yritettiin säätää vähemmän suorituskykyjä vaativaksi. Objektin geometrian yksinkertaistaminen onnistui "Decimate Geometry" -työkalulla. Työkalun käytön jälkeen objektin koko putosi 180 megatavusta 16 megatavuun. Kuvassa 9 näkyy alkuperäinen ja uusi versio möröstä. Kuten kuvasta näkee, objektit ovat lähes samannäköisiä, koska mörön musta väri piilottaa paljon objektien yksityiskohdista.



Kuva 9. Alkuperäinen (vasemmalla) ja uusi (oikealla) mörkö vierekkäin Unityssä.

Peliin luotiin Blenderillä myös kaatunut puu ja kanto sille. Puu toimii siltana pelissä olevalle joelle, jonka käyttäjän on tarkoitus ylittää. Joen pohjassa on vesilelu, joka luotiin myös käyttäen Blenderiä. Itse joki luotiin käyttämällä Blenderin NurbsPath objekteja. Ensimmäisen tason lopussa on viemäri, johon pelaajan on tarkoitus hypätä suorittaakseen ensimmäisen tason. Tämä objekti luotiin käyttämällä Boolean-työkalua ja sylinteriä pelin rata objektille. Sylinteri asetettiin siihen kohtaan, mihin viemärin oli tarkoitus tulla ja Boolean-työkalun Difference-ominaisuudella tasoon leikattiin reikä sylinterin paikalle. Ensimmäisen kentän päädyssä on torni, joka luotiin Blenderillä. Tornin sisällä on tasoja, joita pitkin pelaaja voi kiivetä tornin huipulle ja napata ylimääräisen kolikon. Tornin oven ja ikkunan luotiin samalla tavalla kuin viemärin aukko, eli Boolean-ominaisuutta käyttämällä.

5.2 **Pelin luonti Unityssä**

Pelin luomisessa käytettiin Unity LTS -versiota 2018.4. Pelin luonti lähti liikkeelle pallon ohjattavuudesta ja sen fysiikoiden määrittelystä. Pallon ohjattavuus saatiin toimimaan skriptillä, jossa määritellään hyppy ja ohjaus, joka toimii puhelimen kiihtyvyysanturin avulla. Pelin ohjaukseen yritettiin aluksi käyttää puhelimen gyroskooppia, mutta se ei toiminut yhtä hyvin kuin kiihtyvyysanturi. Gyroskooppi-ohjauksessa pallo ei liikkunut yhtä luonnollisen näköisesti, koska gyroskooppisyötettä ei saatu asetettua vector3 arvoon. Gyroskooppiohjauksessa käytettiin ”transform.translate” -komentoa, joka ei ole hyvä ohjaukseen, koska sillä tavalla pallon liike näyttää leijuvalta ja usein pallo ei reagoinut pelin muihin objekteihin realistisesti. Kiihtyvyysanturin syöte saatiin asetettua Vector3-arvolle, joka teki pallon liikuttamisesta paljon paremman näköistä. Vector3:a voi käyttää ”Rigidbody.AddForce” -arvona, joka lisää voimaa objektiin liikuttaen sitä. Hyppyyn määrittelin rajoitteen, joka tarkistaa, että pallo ei ole ilmassa silloin kun se yrittää hypätä. Kuvassa 10 näkyy koodi, jossa on pallon ohjattavuus ja hyppääminen. Pelin kameraan lisättiin koodi, jonka avulla kamera osaa seurata kuulaa. Tämän skriptin luonti oli helppoa aluksi, mutta kun peliin

lisättiin hahmon valinnan, kameranohjaus skriptille piti määrittää oikea objekti näkymän alussa, joka osoittautui odotettua vaikeammaksi. Skripti saatiin toimimaan lisäämällä siihen GameObject-muuttujan, jolle lisätään käyttäjän valitsema objekti. Tämän jälkeen target attribuutille annetaan GameObjectin eli pelattavana olevan hahmon “Transform” komponentti. Target attribuutin avulla kamera osoittaa sinne suuntaan missä pelihahmo on.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

0 references
public class Movement : MonoBehaviour
{
    0 references
    float speed = 0.05f;
    0 references
    private float nextpress = 0f;
    0 references
    private float delay = 5f;
    4 references
    bool isGrounded;
    3 references
    Rigidbody rb;
    1 reference
    bool isFlat = true;
    0 references
    private Quaternion rot;
    // Start is called before the first frame update
    0 references
    void Start()
    {
        isGrounded = true;
        rb = GetComponent<Rigidbody>();

    }
    // Update is called once per frame
    0 references
    void Update()
    {
        //säädä kääntymisen voimakkuutta
        Vector3 tilt = Input.acceleration * 3;
        if(isFlat){

            tilt = Quaternion.Euler(135,0,0) * tilt;
        Transform Component.transform
        The Transform attached to this GameObject.
        transform.Translate(Input.GetAxis("Horizontal")*Time.deltaTime,0f,0f);

        if(Input.touchCount > 0 && isGrounded == true){

                rb.AddForce(Vector3.up * data.Hyppyvoima);
                isGrounded = false;
        }
    }

    0 references
    void OnCollisionEnter(){
        //Kun pallo tulee kosketuksiin jonkin esineen kanssa isgrounded arvosta tulee true
        isGrounded = true;
    }

}
```

Kuva 10. Koodi, joka luo pallon ohjattavuuden.

Ensimmäinen ongelma pelin luonnissa tuli vastaan, kun Unityä yritettiin päivittää uusimpaan versioon. Aluksi suunnitelmana oli päivittää Unity 2019.2-versioon, koska siinä on ominaisuuksia, joita olisi voinut hyödyntää tässä opinnäytetyössä, kuten taulukosta 1 tulee ilmi. Opinnäytetyön kirjoittamisen aikana julkaistiin uusi Unity-versio 2019.3, joten Unity päivitettiinkin siihen versioon. Projektista luotiin aluksi varmuuskopio, jonka jäl-

keen projekti päivitettiin 2019.3-versioon. Päivityksen jälkeen Unity-projektissa oli paljon virheitä, jotka liittyivät pelin Text mesh pro -komponenttiin. Internetistä etsittiin korjausta ongelmaan, mutta ongelmaan ei löytynyt toimivaa ratkaisua. Unity-version vaihtaminen kesken projektin osoittautui huonoksi ideaksi, joten Unity päätettiin palauttaa versioon, joka oli alun perin käytössä.

Pelin valikot tehtiin luomalla ”Canvas” GameObjectin ja lisäämällä siihen painikkeet. Alkuvalikossa on kaksi painiketta: ”Start game” ja ”Select character”. ”Start game” -painikkeella peli aloitetaan ja ”Select character” -painikkella käyttäjä pääsee valitsemaan pelihahmon. Pelihahmon valinta tallennetaan data.cs tiedostoon, jota voi käyttää kaikissa pelin näkymissä. Data.cs tiedostossa on tallennettuna myös pelaajan elämät ja kerätyt kolikot.

Pelin törmäyksen tunnistukset luotiin käyttämällä erilaisia ”Collider”-komponentteja. Esimerkiksi palloon lisättiin ”Sphere Collider” -komponentti, josta yritettiin tehdä mahdollisimman samankokoisen kuin pallo, jotta pallo olisi vuorovaikutuksissa peliympäristön kanssa mahdollisimman tarkasti. ”Mesh Collider” -komponentilla voi lisätä törmäyksen tunnistuksen tarkasti objektin ympärille. Tämä on hyödyllinen objekteille, jotka ovat monimutkaisempia kuten esimerkiksi pallo ja kuutio. Pelin radassa käytettiin ”Mesh Collider” -komponentteja, koska se on epätasainen ja siinä on jonkin verran yksityiskohtia.

Pelin ensimmäisestä tasosta tehtiin melko yksinkertaisen ja lyhyt. Ensimmäisen tason tarkoitus on opettaa pelaajalle pelin kontrollit ja säännöt. Pelin tasoista voi saada maksimissa kolme tähteä. Ensimmäisen tähden saa siitä, kun läpäisee tason. Toisen tähden saa siitä, kun läpäisee tason tarpeeksi nopeasti. Kolmannen tähden saa siitä, kun kerää tasosta kaikki kolikot. Ensimmäisessä tasossa on kolme kolikkoa. Ensimmäinen kolikko on heti tason alussa, jotta pelaaja näkee, että hänen tarkoituksensa on kerätä näitä objekteja. Toinen kolikko on hieman vaikeampi kerätä, koska se on joessa vesilelun päällä. Tämä kolikko on melko piilossa ja todennäköisesti pelaaja ei edes huomaa kolikkoa ensimmäisellä pelikerralla. Tämä opettaa pelaajalle, että osa kolikoista voi olla piilossa. Kolmas kolikko on helppo löytää, mutta sen kerääminen on haaste. Kolikko on tason lopussa olevan tornin sisällä ja pelaajan on tarkoitus kiivetä tornin huipulle. Kuvassa 11 näkyy ensimmäisen kentän alkuosa.

Kuva 11. Pelin ensimmäisen tason alku.

Animaatioiden luonti Unityllä oli yllättävän helppoa. Animaatiot luotiin avaamalla animaatioikkuna ja luomalla objektille uusi animaatio. Kun animaatio komponentti on luotu, sille piti valita, miten objektia halutaan animoida. Vaihtoehtoina on muun muassa objektin rotaation, koon ja sijainnin animointi. Loin kolikolle animaation, jossa se pyörii paikallaan. Loin myös monsterille animaation, jotta se liikkuisi edes takaisin radalla ollakseen este pelaajalle. Vaikein animaatio, joka luotiin peliin oli joen virtaus, jossa valkoiset viivat kulkevat jokea pitkin. Viivojen tarkoitus oli näyttää siltä, että joen vesi liikkuisi. Tämä animaatio oli hieman hankalampi luoda verrattuna muihin luotuihin animaatioihin, koska virtausviivoja oli melko paljon ja niiden piti kääntyä ja liikkua joen mukana. Viivojen animaatioiden piti myös alkaa eri aikoihin, jotta ne eivät kulkisi rivissä, vaan niiden järjestys näyttäisi hieman satunnaiselta.

5.3 **Pelin optimointi kohdeyleisölle**

Kuten kuvasta 2 näkee videopelejä pelaavat monen ikäiset miehet ja naiset. Tämän takia pelistä yritettiin suunnitella sellainen, että se sopii mahdollisimman laajalle yleisölle. Tämän vuoksi pelin vaikeustason ei kannata olla kovinkaan hankala, jotta esimerkiksi lapset ja vanhukset voisivat nauttia pelistä. Pelissä on tarkoituksena olla 2-3 kenttää. Ensimmäinen kenttä tulee olemaan helppo kenttä, jossa pelaajalle opetetaan pelin mekaniikat ja säännöt. Seuraava kenttä tulee olemaan hieman vaikeampi, ja kolmas kenttä tulee olemaan pelin vaikein kenttä. Täten pelin vaikeustaso tulee nousemaan hiljalleen, opettaen pelaajan pelaamaan peliä ilman että pelaaja turhautuu peliin.

Pelin värimaailmasta on tarkoitus tulla melko kirkas, jotta se herättäisi pelaajan huomion. Pelin ulkoasusta ei haluttu kuitenkaan tehdä liian lapsellista, koska se ei välttämättä vetoaisi vanhempiin pelaajiin. Pelin ympäristöstä päätettiin tehdä tekstuuriton, joka tekee ympäristöstä melko yksinkertaisen ja piirrosmaisen näköisen. Pelin tasoista on tarkoitus tulla melko lyhyitä, koska mobiilipelejä pelataan usein esimerkiksi bussimatkalla, joten pelisessiot eivät kestä usein kovinkaan pitkään.

5.4 **Testaus**

Pelin ensimmäisellä testauskierroksella selvisi, että peli oli liian vaikea. Olin itse tottunut pelin kontrolleihin, koska olen työskennellyt pelin kanssa jo viikkoja. Kentän lopussa oleva torni osoittautui aivan liian vaikeaksi pelaajille, jotka kokeilivat peliä ensimmäistä kertaa. Päätin tehdä tornin hyppy alustoista leveämpiä, jotta niihin olisi helpompi osua ja pysyä niiden päällä. Pelin kentän maali, eli viemäri johon pelaajan on tarkoitus hypätä, ei ollut tarpeeksi selvästi merkattu. Kaikki pelaajat väistivät viemärin, menivät torniin ja kysyivät mitä heidän pitäisi tehdä. Pelaajalle pitää siis viestiä selkeämmin, että viemäriin kuuluu hypätä ja, että se ei ole yksi pelin esteistä. Pelin suorituskyky oli ensimmäisellä testikierroksella hyvä. Yksikään pelaaja ei löytänyt joessa olevaa kolikkoa vaan he hyppäsivät joen yli. Päätin korjata ongelman vaihtamalla kamerakulmaa sellaiseksi, että pelaaja näkee kolikon, kun pelihahmo on joen kohdalla. Pelin kuvataajuus oli molemmilla testatuilla puhelimilla noin 60 kuvaa sekunnissa. Peliä testattiin OnePlus 7 Pro ja Xiaomi Mi 9T -puhelimilla. Peliä yritettiin testata myös Samsung Galaxy S3 puhelimella, mutta pelin testausversiossa ei ollut asennettuna Android KitKatin SDK-alustaa, joten testausta piti siirtää myöhemmäksi. Ensimmäisellä testauskierroksella peliä oli testaamassa kolme tietojenkäsittelyn opiskelijaa.

Testauksen toisella kierroksella testattiin Asus Zenfone 3, Galaxy S3 ja Samsung Galaxy J5 -puhelimilla. Galaxy J5 puhelimessa on Qualcomm Snapdragon 410 -järjestelmäpiiri ja 1,5 gigatavua RAM-muistia. Galaxy J5:ssä on Android 7 -käyttöjärjestelmä. Peli käynnistyi Asus-puhelimella, mutta pelin ruudunpäivitys oli vain noin 20 ruutua sekunnissa. Tällä kertaa olin asentanut Android KitKat SDK-työkalut, mutta peli ei siltikään käynnistynyt Samsung Galaxy S3 -puhelimella. Peli ei käynnistynyt myöskään Galaxy J5 -puhelimella. Kamerakulman muuttaminen joen kohdalla oli hyvä muutos, koska tällä kertaa testaajat huomasivat joessa olevan kolikon ja yrittivät kerätä sen. Vaikka viemäristä yritettiin tehdä selkeämpi maali, pelaajat silti eivät ymmärtäneet, että sinne on tarkoitus hypätä tason lopussa.

Taulukko 3. Pelin toimivuus eri puhelimilla.

Puhelin	**Pelin toimivuus**	**Kuvataajuus**	**Ohjauksen toimivuus**
Xiaomi Mi 9T	Toimii	60	Toimii kuten pitääkin
OnePlus 7 Pro	Toimii	60	Toimii kuten pitääkin
Asus Zenfone 3	Toimii	20	Ohjaus toimii melko huonosti, joka todennäköisesti johtuu huonosta ruudunpäivityksestä.
Samsung Galaxy S3	Peli ei käynnisty ollenkaan	-	-
Samsung Galaxy J5	Peli ei käynnisty ollenkaan	-	-

6 YHTEENVETO JA JOHTOPÄÄTÖKSET

Peli onnistui melko hyvin ja siihen valmistui yksi valmis taso. Ensimmäisestä tasosta tuli melko lyhyt, mutta kolikoiden kerääminen antaa tasolle hieman kestoa. Pallon ohjaus toimii melko hyvin ja pelin suorituskyky on hyvä ainakin uudemmilla puhelimilla. Kuten taulukosta 2 näkee, pelin ruudunpäivitys ei ole halutulla tasolla, vanhemmilla puhelimilla.

Opinnäytetyössä tuli vastaus tutkimuskysymyksiin. Opinnäytetyön tutkimuskysymykset olivat: ”Kuinka 3D-pelin luonti Android-puhelimelle onnistuu?”, ”Minkälaiselle kohdeyleisölle Android-peli kannattaa luoda?” ja ” Miten Unity-pelin luonti onnistuu aloittelijalta?”. 3D-pelin luonti onnistui Android alustalle hyvin ja pelin siirtäminen Android puhelimelle oli helppoa, koska siitä on tehty mahdollisimman helppoa Unity-pelimoottorilla. Pelin ohjaaminen Android-puhelimen kiihtyvyysanturilla oli melko helppo toteuttaa, koska Unityllä oli paljon dokumentointia Android alustasta ja pelin luonnista sille. Android pelejä pelaa monen ikäiset miehet ja naiset. Vaikka suurin osa pelaajista on nuoria aikuisia, päätin suunnitella pelistä sopivan mahdollisimman monelle ikäryhmälle. Pelin luonti sujui yllättävän hyvin vaikka minulla ei ollut juurikaan kokemusta aiheesta. Projektin alussa käytin melko paljon Unityyn liittyviä ohjevideoita, mutta projektin loppupuolella asiat sujuivat nopeammin enkä joutunut käyttämään juuri ollenkaan ohjeita, vaan sovelsin Unityn omaa dokumentointia pelin luomisessa.

Pelissä on ainakin yksi tiedossa oleva ongelma. Joskus pallolla on mahdollista hyppiä seinää vasten monta kertaa peräkkäin. Tämä mahdollistaa pallon pääsyn paikkoihin, joihin sen ei kuuluisi päästä ja huijaamisen pelin torniosuudessa. Ongelma johtuu siitä, että pelin hyppiminen estetään, jos pelin hahmo ei ole kosketuksessa minkään objektin kanssa. Tämä tarkoittaa sitä, pallo voi hypätä, kun se koskettaa mitä tahansa objektia, kuten seinää. Pelin luonnin aikataulutuksessa tuli kiire, joten en ehtinyt luoda peliin ääniä tai musiikkia. Pelin tähdet toimivat hyvin, mutta niiden keräämisestä ei viestitä pelaajalle ollenkaan. Pelin pelaajalle ei ole minkäänlaista tietoa siitä, miten kaikki kolme tähteä olisi tarkoitus saada. Pelaajalle olisi hyvä kertoa esimerkiksi pelin alussa, miten eri tähdet saadaan.

3D-objektien luonti sujui odotettua paremmin. Erityisesti Hirviön ja puun luonti onnistui mielestäni hyvin. Ensimmäisen kentän ulkoasu on hyvä, kunhan pelaaja pysyy pelialueella, eikä käytä bugia karatakseen pelialueelta. Aluksi objektien luonti oli erittäin hidasta ja vaikeaa. Vaikeutta lisäsi se, että suuri osa Blenderiin liittyvistä ohjevideoista oli tehty vanhemmalla Blender-versiolla ja videoissa käytetyt painikkeet ja työkalut eivät olleet siellä missä ne näkyivät videolla. Blenderin käytöstä tuli sujuvampaa ja nopeampaa projektin edetessä.

Opin opinnäytetyön aikana paljon tietoa pelin kehityksestä ja 3D-mallinnuksesta. Opin luomaan yksinkertaisia 3D-malleja Blender-ohjelmalla.

Opinnäytetyön aikana opin paljon Unity-pelimoottorin käytöstä ja pelinkehityksestä. Ennen opinnäytetyön alkua ohjelmointitaitoni olivat melko ruosteessa. Opinnäytetyö kuitenkin toi taidot takaisin mieleen ja ohjelmointi C#-ohjelmointikielellä oli mielekästä. Opin aikataulun suunnittelun tärkeyden, koska projektiin jäi tekemättä asioita, joiden olisin halunnut olevan pelissä. Tämä ongelma ei olisi välttämättä olemassa, jos olisin suunnitellut projektin etenemistä etukäteen enemmän.

7 JATKOKEHITYS

Suunnittelin jatkokehityksenä tehdä peliin vielä ainakin kaksi uutta tasoa ja korjata pelin bugit. Pelissä ei ole myöskään vielä yhtään ääniä tai musiikkia, joka on myös yksi asia, jonka haluaisin tehdä peliin opinnäytetyön jälkeen. Pelin toisesta tasosta suunnittelin viemäriä, koska se olisi melko helppo tehdä suljetun tilan takia. Viemäritaso ei vaadi paljoa monimutkaisia 3D-objekteja kuten puita tai kiviä. Yhtenä jatkokehitystavoitteena on luoda kolmas hahmo peliin, joka toimisi ehkä hieman eri tavalla kuin kaksi ensimmäistä hahmoa. Hahmolla voisi olla esimerkiksi erikoisominaisuus, jolla se voi hypätä korkeammalle kuin muut hahmot, ja täten avata esimerkiksi oikotien mahdollisessa toisessa tasossa.

Jos pelistä joskus tulee tarpeeksi laaja ja laadukas, on mahdollista, että julkaisen sen Google Play Store -sovelluskauppaan. En tule todennäköisesti tekemään pelistä iOS-versiota, koska minulla ei ole tarvittavia laitteita tähän tarkoitukseen.

Pelissä on paljon bugeja ja ongelmia, jotka aion korjata opinnäytetyön valmistumisen jälkeen. Suurin ongelma pelissä on äänien ja musiikin puuttuminen. Tulen todennäköisesti käyttämään ilmaisiä ääniefektejä ja musiikkia pelin äänimaailman luonnissa. Suurin bugi pelissä on seinää vasten hyppiminen. Suunnitelmana oli yrittää korjata tämä bugi mahdollisimman nopeasti.

LÄHTEET

Android Authority. (2020) *When should you expect to receive Android 10?* Viitattu 3.2.2020
https://www.androidauthority.com/android-10-update-1023882/

Android Developers. (2020) *Android 10 for Developers.* Viitattu 3.2.2020
https://developer.android.com/about/versions/10/highlights

Aponte, M-V., Levieux, G. & Natkin, S. (2011) *Measuring the level of difficulty in single player video games.* Viitattu 12.2.2020
https://www.sciencedirect.com/science/article/pii/S1875952111000231

Avisekhar, R. (2016) *The Android Game Developer's Handbook.* Viitattu 31.1.2020
https://hamk.finna.fi/Record/vanaicat.128946

Bankmycell. (2020) *How many smartphones are in the world?* Viitattu 10.3.2020
https://www.bankmycell.com/blog/how-many-phones-are-in-the-world

Blender Foundation. (n.da) *About.* Viitattu 23.1.2020
https://www.blender.org/about/

Blender Foundation. (n.db) *History.* Viitattu 23.1.2020
https://www.blender.org/foundation/history/

Brodkin, J. (2013) *How Unity3D Became a Game-Development Beast*. Viitattu 17.1.2020
https://insights.dice.com/2013/06/03/how-unity3d-become-a-game-development-beast/

Epic Games. (n.d) *FREQUENTLY ASKED QUESTIONS (FAQ)* Viitattu 20.1.2020
https://www.unrealengine.com/en-US/faq?active=release

From Software, Inc. (2020) *History.* Viitattu 28.1.2020
https://www.fromsoftware.jp/ww/company_history.html

Google. (2019) *Android 10.* Viitattu 3.2.2020
https://www.android.com/android-10/

Gough, C. (2019a) *Distribution of video gamers worldwide in 2017, by age group and gender.* Viitattu 30.1.2020
https://www.statista.com/statistics/722259/world-gamers-by-age-and-gender/

Gough, C. (2019b) *Mobile gaming – Statistics & Facts.* Viitattu 29.1.2020 https://www.statista.com/topics/1906/mobile-gaming/

Hindy, J. (2017) *2016 recap: 90% of Google Play's revenue came from games (and more fun stats!)* Viitattu https://www.androidauthority.com/2016-recap-90-percent-google-play-revenue-gaming-fun-stats-743626/

Jason Gregory. (2019) *Game Engine Architecture.* Viitattu 27.1.2020 https://hamk.finna.fi/Record/vanaicat.133453

Ky, J. (2013) C#: *A Beginner's Tutorial.* Viitattu 3.3.2020 https://hamk.finna.fi/Record/nelli19.3370000000000961

Microsoft. (2017) *The history of C#.* Viitattu 3.3.2020 https://docs.microsoft.com/en-us/dotnet/csharp/whats-new/csharp-version-history

Naughty Dog, LLC. (2020) *About.* Viitattu 29.1.2020 https://www.naughtydog.com/company

Petty, J. (n.da) *What is 3D Modeling & What's It Used For?* Viivattu 20.2.2020 https://conceptartempire.com/what-is-3d-modeling/

Petty, J. (n.db) What is a Polygon Mesh? Viitattu 9.3.2020 https://conceptartempire.com/polygon-mesh/

Pickell, D. (2019) *The 7 Stages of Game Development.* Viitattu 7.2.2020 https://learn.g2.com/stages-of-game-development

ResearchGate. (2012) Mesh elements: vertex, edge and face. Viitattu 10.3.2020 https://www.researchgate.net/figure/Mesh-elements-vertex-edge-and-face_fig3_272666473

Statcounter. (2019) *Mobile Operating System Market Share Worldwide.* Viitattu 31.1.2020 https://gs.statcounter.com/os-market-share/mobile/worldwide

Statista. (2020) *Worldwide gross app revenue of Google Play from 2016 to 2019.* Viitattu 21.2.2020 https://www.statista.com/statistics/444476/google-play-annual-revenue/

Swing, O. (2017) *Using Gyroscope Technology to Implement a Leaning Technique for Game Interaction.* Viitattu xx.xx.2020 http://www.diva-portal.org/smash/get/diva2:1144867/FULLTEXT02

Unity Technologies. (n.da) *Coding in C# in Unity for beginners.* Viitattu 3.3.2020
https://unity3d.com/learning-c-sharp-in-unity-for-beginners

Unity Technologies. (n.db) *Download Unity*. Viitattu 17.1.2020
https://unity3d.com/get-unity/download

Unity Technologies. (n.dc) *Game engines – how do they work?* Viitattu 17.1.2020
https://unity3d.com/what-is-a-game-engine

Unity technologies. (n.dd) Long Term Support. Viitattu 17.3.2020
https://unity3d.com/unity/qa/lts-releases

Unity Technologies. (n.de) *Plans and pricing.* Viitattu 17.1.2020
https://store.unity.com/

Unity Technologies. (2019) *Unity 2019.2.* Viitattu 21.1.2020
https://unity.com/releases/2019-2?_ga=2.256300363.1594364899.1579595684-47065188.1579251536

Vu, A. (2018) *Game Design: A Different Approach to Difficulty.* Viitattu 12.2.2020
https://www.gamedev.net/tutorials/game-design/game-design-and-theory/game-design-a-different-approach-to-difficulty-r4992/

WePC. (2020) *2020 Video Game Industry Statistics, Trends & Data.*
https://www.wepc.com/news/video-game-statistics/#gamers-demographic