Android-pelin kehitys ja julkaisu

Tatu Koski

2019 Laurea

Laurea-ammattikorkeakoulu
Laurea Leppävaara

Android-pelin kehitys ja julkaisu

Tatu Koski
Tietojenkäsittely
Opinnäytetyö
Huhtikuu, 2019

Laurea-ammattikorkeakoulu **Tiivistelmä**
Tietojenkäsittelyn koulutusala

Tatu Koski

Android-pelin kehitys ja julkaisu

Vuosi 2019 Sivumäärä 39

Tämän opinnäytetyön tarkoituksena oli kehittää ja julkaista peli Android-älypuhelimille Google Play-sovelluskaupassa. Tarkoituksena oli kirjoittaa samalla kattava tietopaketti pelinkehityksen avuksi aiheesta kiinnostuneille. Työssä keskitytään enemmän pelikehityksessä ohjeistukseen kuin lopputuloksena syntyvään prototyyppiin. Tavoitteena on kuitenkin luoda tuo prototyyppi valmiiksi jatkokehitystä varten. Aihe valittiin alaan kohdistuvan kiinnostuksen ja markkinatilanteen vuoksi. Opinnäytetyössä ei keskitytty pelikehityksen yksityiskohtiin ja itse ohjelmointiin, vaan enemmän pelinkehitysprosessin pääpiirteisiin. Työssä luotu ohjeistus voi toimia apuna aloitteleville kehittäjille heidän omissa pelinkehitysprojekteissaan.

Prototyyppi rakennettiin Unity3D-nimisellä alustariippumattomalla pelimoottorilla. Projektin koko huomioon ottaen peli rakennettiin alusta lähtien yksinkertaisena sekä idealtaan että toteutukseltaan. Työssä käytiin läpi pelikehityksen tärkeimmät osa-alueet ohjelmoinnista peligrafiikkaan ja audioon sekä mobiilipelin rakentamisesta sen julkaisemiseen. Lisäksi, työn alussa käytiin läpi kehitysprojektissa käytetyt lähteet ja työkalut sekä niiden asennuksiin ja käyttöönottoihin tarvittavat toimenpiteet. Projektin tietoperustana käytettiin Unity Technologiesin tarjoamaa manuaalia, jota hyödynnettiin pelinkehityksessä. Prototyyppinä syntynyttä peliä voi jatkokehittää ja laajentaa tulevaisuudessa.

Asiasanat: Android, mobiilisovellus, videopeli, unity

Laurea University of Applied Sciences **Abstract**
Degree Programme in Business Information Technology

Tatu Koski

Developing and Publishing an Android Game

Year 2019 Pages 39

The goal of this bachelor‘s thesis was to develop and publish an Android game to Google Play Store. The aim was to write an extensive report of the game development process for others interested in the subject. Rather than delving deep into details of programming, the goal of this thesis was to focus more on the game-development process as a whole. Lastly, a working prototype, ready for further improvements and expansions, was developed during this process. The subject was chosen for personal interest towards the industry, that has been growing exponentially for the past few years. This thesis provides a quick recapitulation about the basics of game development. inexperienced developers may use this in their own projects.

The protype was built with a cross-platform game engine called Unity3D. Considering the possible scope of the project, the game was intentionally designed elementary, both its idea and execution. The thesis covers the most important aspects of game-development from programming to graphics and audio design, as well as building and publishing the mobile package. Additionally, the focus was on the sources and tools, that were utilized throughout this project. They were reviewed early on, written about and along with guides to the necessary installations. The database used in the game-development-process solely leaned on the material from the manual Unity Technologies offer on their website. The thesis’ documentation consists of the various steps and orders throughout the development process from planning to developing, building and publishing. The prototype that was created as a result is ready for further development and expansion.

Keywords: Android, Mobile application, Videogame, Unity

Sisällys

1 Johdanto

Mobiilipelien suosio on ollut etenkin viime vuosien aikana räjähdysmäisessä kasvussa. Tämä ilmiö ei kuitenkaan ole uusi, sillä mobiilipeleihin kohdistuva kiinnostus räjähti heti ensimmäisten älypuhelinten ja Applen luoman App Storen ilmestyessä markkinoille vuonna 2008. Mobiilipelit eivät itsessään ole uusi keksintö, mutta älypuhelinten ja kosketusnäyttöjen tarjoamat uudet mahdollisuudet mullistivat mobiilipelimarkkinat. Uudet kauppapaikat, sekä alati kehittyvät mobiililaajakaistat, saivat pelien ja sovellusten jakelusta ennennäkemättömän helppoa.

Opinnäytetyön aihe valittiin juuri tämän mobiiilipelimarkkinoiden kasvun vuoksi. Android-alustan valintaan, näiden markkinoiden lisäksi, oli Androidille julkaisun suoraviivaisuus. Pienen mobiilipelin kehittäminen ei vaadi rahallista investointia, ja Play-kauppaan julkaisu on päälinjauksiltaan suoraviivaista. Lisäksi, mobiilipelit ovat luonnostaan yksinkertaisia ja pieniä sovelluksia alustan fyysisten rajoitusten vuoksi. Tämä sopii hyvin aloittelevalle ohjelmoijalle.

Työn tavoitteena on kehittää ja julkaista peliprototyyppi ja kirjoittaa aiheesta kattava tietopaketti, jota alasta kiinnostuneet voivat hyödyntää omissa projekteissaan. Työssä ei tulla keskittymään yksityiskohtiin tai itse pelin ohjelmointiin, vaan tarkoituksena on keskittyä pelikehitysprosessin kokonaiskuvaan, ja sisältöön. Työn tavoitteena on tutustua pelikehityksen perusteisiin valitulla pelimoottorilla ja rakentaa tällä valmis, yksinkertainen prototyyppi tulevaisuuden jatkokehitystä varten. Peli rakennetaan alustariippumattomalla Unity3D-pelimoottorilla ja julkaistaan Google Play-kaupassa Android-käyttöjärjestelmälle. Peli tulee olemaan prototyypin mukaisesti yksinkertainen sekä mekaniikoiltaan että ulkoasultaan, jotta projekti saadaan ajoissa valmiiksi. Työ perustuu iteratiiviseen pelikehitykseen. Tietopankkina käytetään Unity Technologies-tietokantaa.

Kehitystyöstä kirjoitettava raportti tulee sopimaan muille alasta kiinnostuneille ja kokemattomille henkilöille. Edeltävää kokemusta ei pelikehityksestä, Unity3D:llä tai ylipäätänsä muillakaan pelimoottoreilla ei ennen projektin aloitusta löytynyt, joten työ kirjoitetaan vastaavasti aloittelijan näkökulmasta. Ohjelmoinnin puolelta ei myöskään aiempaa kokemusta löydy perusteita kattavammin. Unity3d:llä käytettävä C# puolestaan täysin uusi ohjelmointikieli.

2 Unity

Unity on tanskalaisen Unity Technologies nimisen peliyhtion, vuonna 2005 julkiseen käyttöön julkaisema alustariippumaton pelimoottori, joka oli alunperin tarkoitettu vain macOS:lle käytettäväksi. Unity kuitenkin laajeni suosiostaan johtuen, ja tukeekin nykyään 27 alustaa. Alunperin yksinkertainen pelimoottori on vuosien mittaan laajentunut ja kehittynyt hienostuneemmaksi, ja tämä kehitys jatkuu tänäkin päivänä lähes viikoittain julkaistavimmilla

pienemmillä päivityksillä. Näiden hienosäätöjen lisäksi Unitystä julkaistaan vuosittain suurempia versiopäivityksiä, joiden myötä ohjelmisto, tai sen käyttö, saattaa muuttua hyvinkin radikaalisti. Näistä viimeisin on Unity 2019.1.0, joka julkaistiin 15. huhtikuuta, opinnäytetyön kirjoittamisen aikana. Unityä ei kuitenkaan päivitetty tähän uudempaan versioon, vaihdosta johtuvien lukuisten muutoksien vuoksi, vaan peli rakennettiin Unityn aiemmalla versiolla 2018.3.5.

Tänä päivänä Unitystä löytyy hyvinkin tärkeitä ominaisuuksia, kuten fysiikkamoottori, sekä 2D, että 3D:lle, animaatio, sekä hiukkasmoottori. Unityllä voi luoda sekä kaksi, että kolmiulotteisia pelejä. Ulottuvuutta vaihtamalla, Unity tarjoaa erilaisia ominaisuuksia, vaikkakin nämä muistuttavatkin enemmän tai vähemmän toisiaan. Tämä projekti rakennettiin kaksiulotteisessa avaruudessa. Unity on kaiken kattava pelimoottori/editori, jolla on periaatteessa mahdollista rakentaa kaikki ohjelmoinnista audioon ja animaatioon, erityisesti Unityn mukana tulevaa Asset Storea hyväksikäyttäen. Tämän projektin peli pyrittiin rakentamaan mahdollisimman pitkälle omia assetteja käyttäen. Poikkeuksena peliin ladattiin valmiit tekstifontit ja äänet, sekä Asset Storesta, että muista ulkoisista lähteistä.

Pelinkehitykseen suunniteltuja pelimoottoreita on tarjolla on kymmeniä, ellei jopa satoja vaihtoehtoja, riippuen kehittäjän tavoitteista. Lisäksi, moisen voi myös itse kehittää, mutta itseltäni ei löytynyt vuosikymmentä tuohon aikaa, saatikka tietotaitoa. Itse valitsin Unityn, jolla voi rakentaa pelejä käytännössä katsoen kaikille mahdollisille alustoille, joka on ilmainen harrastuskäytössä, ja johon löytyy loputtomasti sekä virallista, että käyttäjäyhteisön luomaa tukimateriaalia. Unity on myös yksi kattavimmista, sekä helpoiten ymmärrettävistä ohjelmistoista pelikehitykseen: se on pelimoottori, editori ja kauppapaikka yhdessä paketissa.

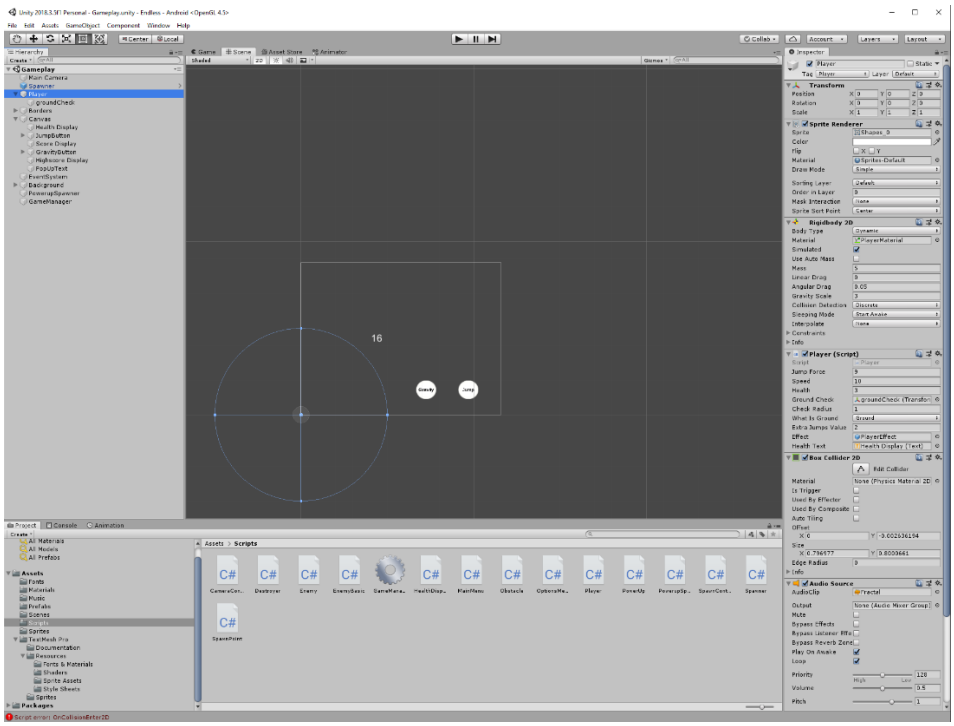
Yleinen harhaluulo on, ettei pelejä voi kehittää mikäli ohjelmoinnista ei ole kokemusta. Tämä ei pidä paikkaansa, sillä nykypäivänä internetistä löytyy, erityisesti Unityyn, järjettömästi informaatiota, esimerkkejä, sekä videoita, joiden avulla kuka vain pääsee alkuun. Vaikka aiempi koodauskokemus ei ole välttämätöntä yksinkertaisen pelin luomisessa, on sekä koodauksesta, että tutusta ohjelmointikielestä kuitenkin suunnattomasti hyötyä, sekä pelin laadun, että ajankäytön puolesta. Itse en ole ennen tätä projektia Unityä koskaan käyttänyt, eikä ohjelmointitaustaakaan löydy alkeita lukuunottamatta lainkaan.

Unity tarjoaa pelikehittäjien tavoitteista ja peliprojektien laajuudesta riippuen erilaisia lisenssejä 125e/kk asti. Ilmais- ja täysversion välillä on lukuisia eroja, joista merkittävimmät ovat Pro-version hienostuneemmat ominaisuudet, sekä mahdollisuus ryhmän sisäiseen versionhallintaan. Sekä ilmais-, että täysversiolla on mahdollista julkaista pelejä ilman rojaltteja Unitylle, mutta mikäli yrityksen liikevaihto ylittää 100 000 Yhdysvaltain dollarin rajan, on yrityksen hankittava ammattilaislisenssi. Tässä projektissa käytetään Unityn Personal-lisenssiä, joka on tarkoitettu harrastus ja non-profit käyttöön.

Unityn asennus ei muuta vaadi kuin rekisteröitymisen ja Unityn lataamisen. Sen jälkeen Unity on valmiina käytettäväksi. Jo pelkästään harrastuskäyttöön tarkoitettu, ilmainen, Unity Personal tarjoaa massiivisen määrän ominaisuuksia ja vaihtoehtoja kehittäjille, joka ensimmäisellä avauskerralla saattaa aiheuttaa aloittelijoissa epätoivoa. Tätä varten on suoiteltavaa turvautua aluksi Unityn tarjoamiin, virallisiin tutoriaaleihin, videoihin, sekä Unityn tarjoamaan User Manuaaliin. Kuitenkin, yksi Unityn vahvuuksista on mahdollisuus oppia kokeilemalla ja testaamalla eri asioita. Editorin lisäksi, pelin ja sen objektien ominaisuuksia pystyy jopa muokkaamaan reaaliajassa, pelin pyöriessä kehitysruudun rinnalla. Lisäksi, edellä mainittu Asset Store mahdollistaa paikkaamaan aukkoja omassa osaamisessa, oli se sitten ohjelmoinnissa tai artistisessa näkemyksessä.

2.1 Unity Editor

Unityn käyttöliittymä on nimeltään Unity Editori. Tämä editori koostuu ikkunoista sekä niiden välilehdistä, joita käyttäjät voivat vapaasti siirrellä, muokata, lisäillä ja poistaa tarpeidensa mukaan. Juurikin tämä modulaarisuus on yksi Unityn suurimmista eduista, muihin vastaaviin pelimoottoreihin verrattuna.



Kuva 1: Unity Editorin päänäkymä

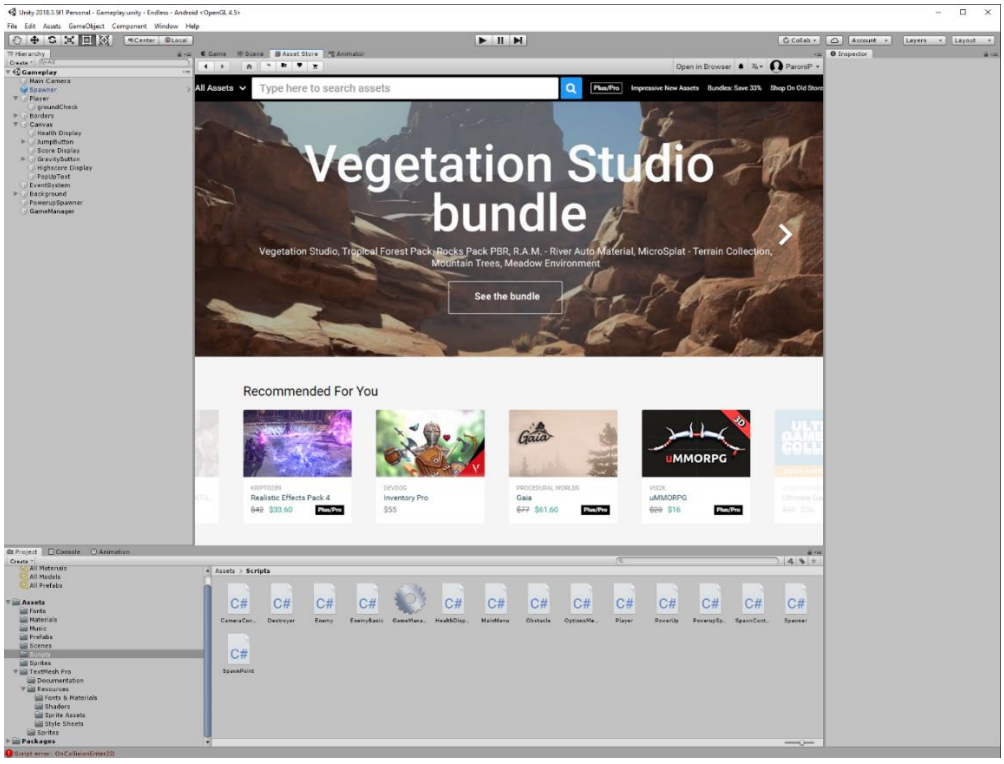
Ensikertalaiselle editori saattaa alkuun näyttää pelottavan monimutkaiselta, mutta totuus on, että Unityn käyttöliittymä on lyhyen totuttelun jälkeen käyttäjäystävällinen ja selkeä. Samalla se on kuitenkin tarpeeksi kattava kokeneempienkien kehittäjien isompiin

projekteihin. Drag & Drop- periaatteella toimiva editori on looginen, ja pelkästään Unity Technologiesin tarjoamilla tutoriaalivideoilla pääsee pitkälle.

Kehittäjille editorin tärkeimpiin osiin kuuluu Inspector, jossa peliobjektien ominaisuuksia lisätään/muokataan/poistetaan, sekä ’Scene’ ja ’Play’ tilat, joissa vastaavasti kehitetään ja testataan peliä. Peliobjektien lisääminen tapahtuu Scene-ikkunaan nämä, joko raahaamalla, tai ’Create’-napista ’Hierarchy’-ikkunassa. Projektissa käytettävät assetit puolestaan tallennetaan Editor-ruudun alaosassa sijaitsevaan Assets-kansioon.

2.2 Unity Asset Store

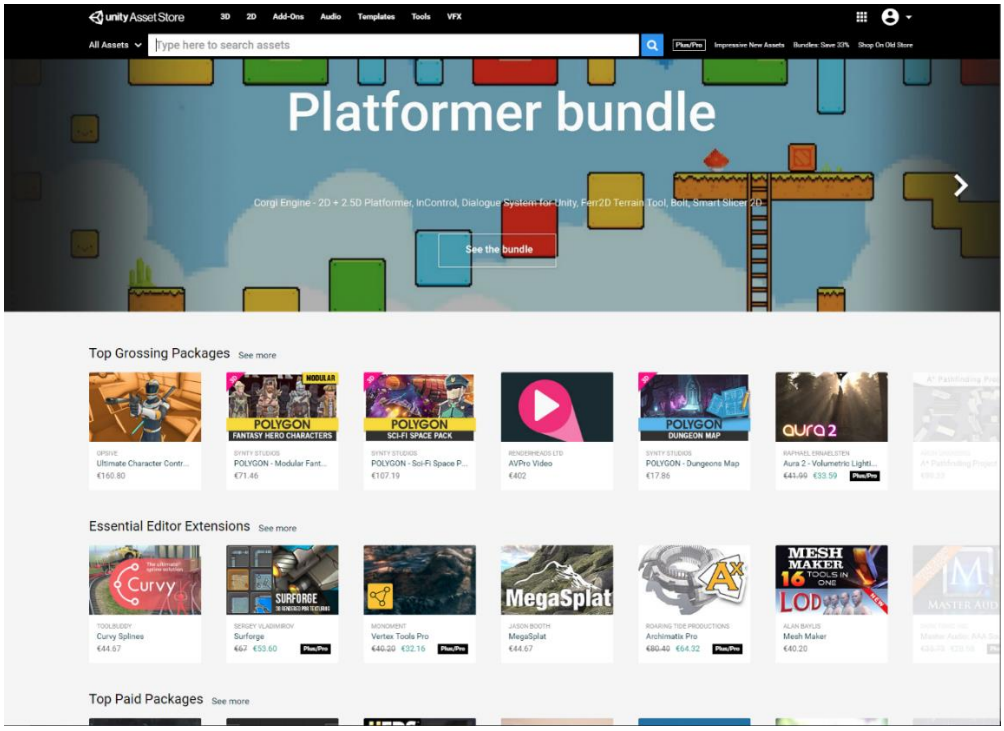
Unity Asset Store on Unity Technologiesin julkaisema ja Unityn mukana tuleva resurssikauppapaikka, jonne kehittäjät voivat halutessaan julkaista kanssakehittäjien käyttöön erilaisia peliobjekteja, joko ilmaiseksi tai maksua vastaan. Vaihtoehtoisesti, Unity Asset Storea voi käyttää myös selaimella Unity Technologiesin sivuilta (https://assetstore.unity.com/).



Kuva 2: Unity Asset Storen etusivu kirjoitushetkellä

Asset Store muistuttaa sekä ulkoisesti, että toiminnallisuudeltaan muita digitaalisia kauppapaikkoja, kuten esimerkiksi Google Play Storea ja iosin App Storea, mutta valmiiden sovellusten sijaan, Asset Store tarjoaa työkaluja näiden sovellusten kehittämiseen. Kauppapaikalta löytyy tällä hetkellä kymmeniä tuhansia erilaisia resursseja aina grafiikoista ja äänistä valmiisiin koodipätkiin ja jopa Unityyn liitettäviin lisäosiin. Tässä projektissa Asset

Storen käyttö pyrittiin pitämään minimaalisena. Valitut, valmiit resurssit, joita pelinkehityksessä käytettiin olivat asioita, joihin osaamista tai aikaa ei löytynyt, kuten pelissä käytettävät tekstifontit ja äänet.

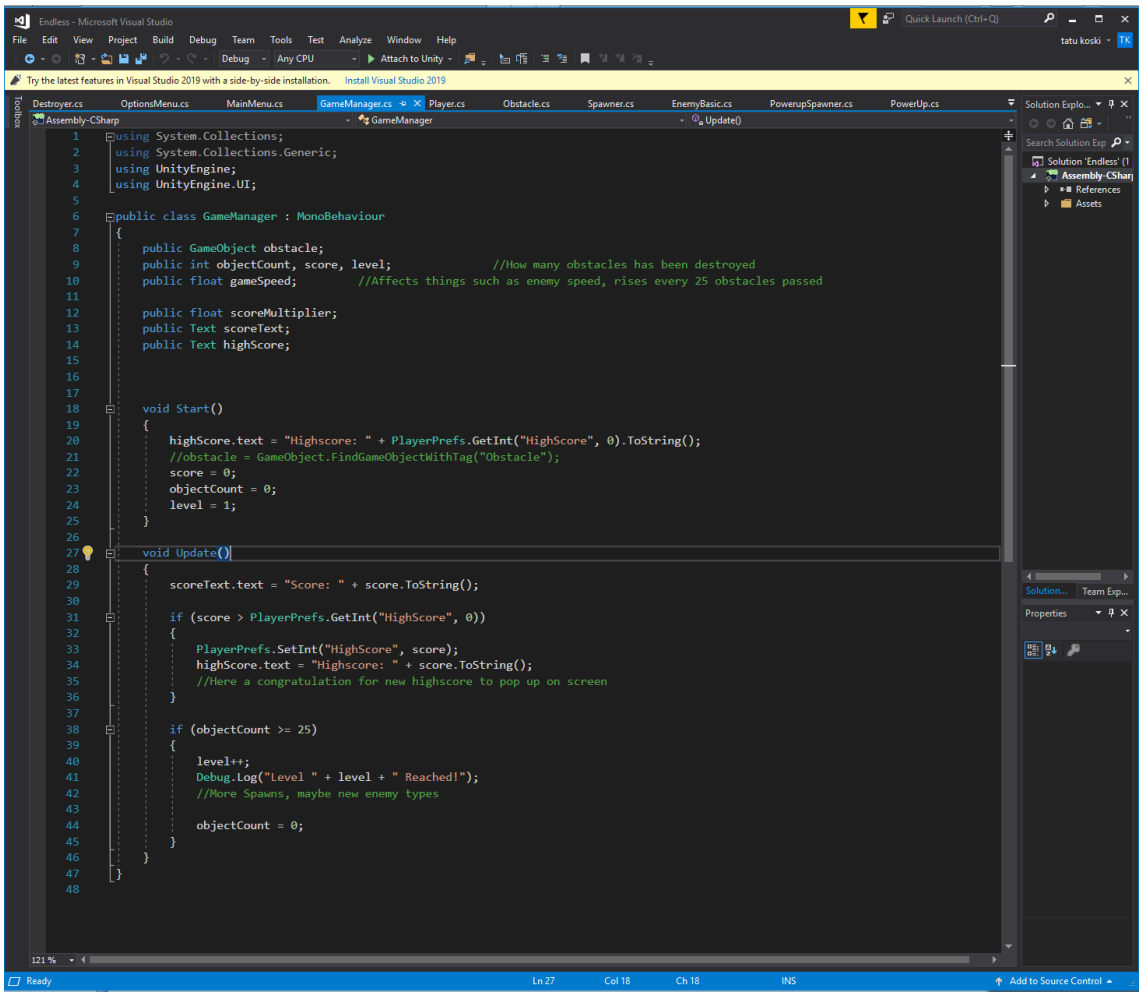


Kuva 3: Asset Store vaihtoehtoisesti selaimella

Ulkoisten assettien liittyminen Unity Asset Storen ulkopuolelta on tietenkin mahdollista, ja suotavaa. Unity3D vuonna 2019 tukee kaikkia yleisimpiä tiedostomuotoja, sekä liudan muita, pienempiä ja harvinaisempiakin. Ulkoisten assettien lisäys onnistuu joko tiedoston yksinkertaisesti raahaamalla Unityn Asset-kansioon, tai valitsemalla Editorista ’Import new asset.’

2.3 Ohjelmakoodi

Ohjelmointi Unityllä, ja muilla nykyajan pelimoottoreilla, noudattaa samoja sääntöjä kuin mitä tahansa ohjelmaa kirjoittaessa. Ohjelmointipätkät rakentuvat attribuuteista ja funktiosta, joita niille määrätyt peliobjektit noudattavat. Suurimpana erona pelimoottoreiden ja muiden sovellusten välillä on, ettei itse pelin pyörimistä tarvitse itse ohjelmoida, sillä tämä osa koodista on sisällytetty jo itse pelimoottoriin.



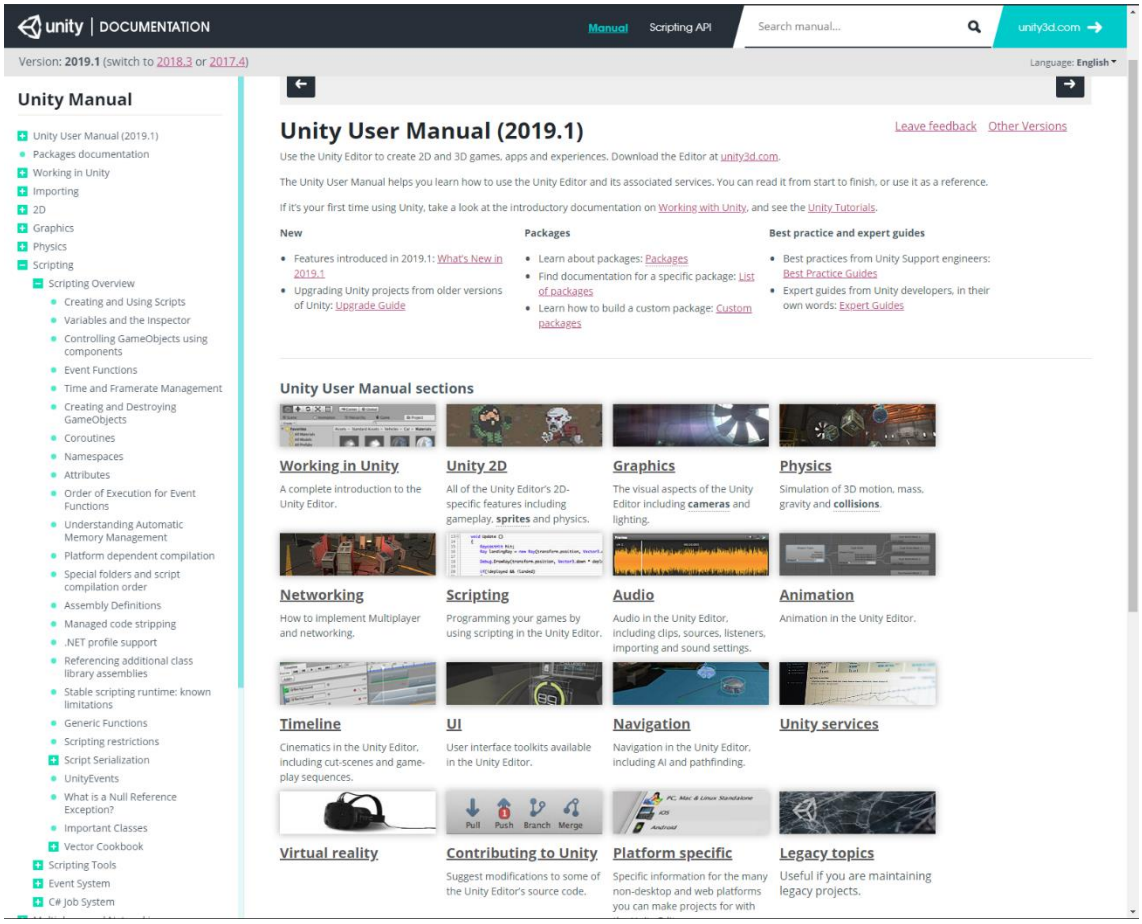
Kuva 4: Visual Studio 2017, kuvassa GameManager-niminen scripti

Unity tuki aikaisempina vuosina koodikielinä nykyisen C#:n lisäksi Boo:ta, sekä JavaScriptiin pohjautuvaa UnityScriptiä. Näistä kuitenkin ainoastaan C# on nykyisin Unityn virallisesti tukema ohjelmointikieli. Unityn mukana tulee tänä päivänä Microsoft Visual Studio 2017-ohjelmointiympäristö, jonka ominaisuuksiin lukeutuu monien modernien ohjelmointiympäristöjen tavoin ominaisuuksia, kuten automaattinen täyttö ja syntaksin väritys. Unityn mukana tulevan Visual Studion käyttö on suositeltavaa, sillä se on optimoitu juuri tähän käyttöön, mutta käyttäjät voivat halutessaan käyttää mitä tahansa muuta editoria.

Unityssä, jokaisella pelissä käytettävällä objektilla ja komponentilla tulee olla kirjoitettuna niille skripti. Ilman koodia, ei pelin osat luonnollisestikaan mitään itsestään tee. Unityssä valmiiden komponenttien ohjelmakoodi on avoin, vapaasti muokattavissa ja laajennettavissa kehittäjien tarpeiden mukaan.

2.4 Unity User Manual

Suurin etu, etenkin aloitteleville kehittäjille, jonka Unity tarjoaa, on Unity Technologiesin tarjoama Unity User Manual. Tämä manuaali löytyy Unityn kotisivuilta Learn-välilehden alta (https://docs.unity3d.com/Manual/index.html), ja kattaa kaiken mahdollisen tiedon, mitä Unitystä on syytä tietää. Nämä sivut pitävät sisällään kaiken yleistiedosta Unityn eri osista, aina käytännön esimerkkeihin ja koodipätkiin asti. Tämä dokumentaatio myös päivittyy jatkuvasti Unityn mukana, eikä projektin aikana kertaakaan törmätty ongelmaan, johon ei vastausta mainituilta sivuilta löytynyt. Tämä dokumentaatio toimi toiminnallisen opinnäytetyön ainoana lähteenä ja siihen turvauduttiinkin pelikehityksen jokaisessa vaiheessa.



Kuva 5: Unity Manuaalin etusivu kirjoitushetkellä

User manual on jaettu kategorioihin ja alakategorioihin, aina yksittäisiin funktioihin asti. Projektin aikana tärkeimmäksi tietopankiksi osoittautui Unityn fysiikkamoottorista kertovat kappaleet. Pelimoottorien fysiikkamallinnukset toimivat korkean luokan matematiikan perusteiden pohjalta jo pelkästään 2D-avaruudessakin, joten manuaalin hyväksikäyttäminen on suositeltavaa.

Unityn perusteiden ja osien lisäksi, User manual sisältää vielä kategoriat versiopäivityksissä tapahtuneisiin muutoksiin sekä sanaston. Näistä ensimmäistä on hyvä tarkkailla erityisesti, jos Unityn version haluaa keskeneräisen projektin aikana päivittää, sillä muutokset saattavat olla hyvinkin laajoja. Sanasto on puolestaan hyvin yksiselitteinen. Aloittelevalle, etenkin Unityä käyttävälle, ei hyödyllisempää sivua löydy. Manuaalin sanasto sisältää lyhyen selityksen jokaisesta Unityn termistä.

2.5 Unity Remote

Viimeisimpänä lisätyökaluna käytettiin Unity Remotea. Remote on vastaavasti Unity Technologiesin julkaisema mobiilisovellus, ladattavissa Play-kaupasta. Unity Remotea käytetään pelin testaamiseen kohdelaitteella reaaliajassa.

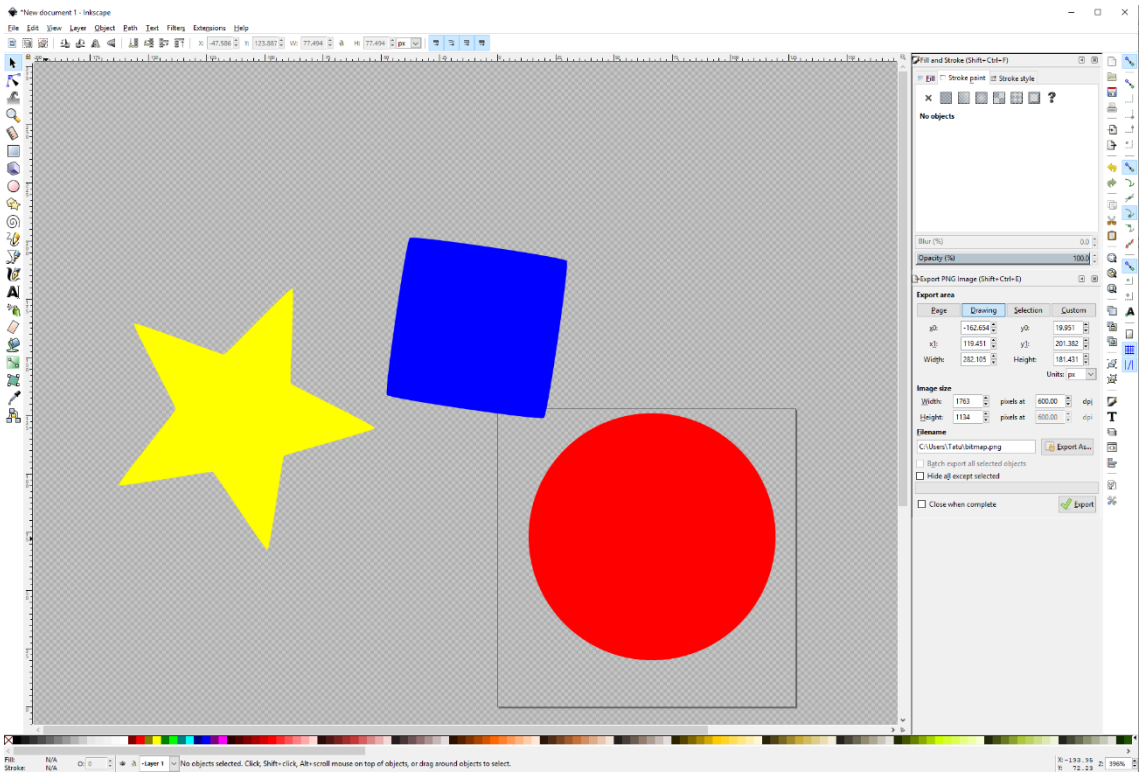
Mobiililaitteen ollessa liitettynä tietokoneeseen, Unity Remote piirtää pelikuvan liitetyn laitteen ruudulle, josta peliä voidaan nyt myös ohjata. Vaikka Remote ei anna todenmukaista kuvaa pelin tehokkuudesta, on sen käyttö äärimmäisen suotavaa, sillä ilman sovelluksen käyttöä, peli joudutaan fyysisesti rakentamaan ja siirtämään kohdelaitteelle jokaista testikertaa varten. Rakentaminen on prosessina suhteellisen nopea proseduuri, mutta vie turhaa aikaa itse kehitystyöltä.

Unity Remoten käyttöönotossa on useampi askel. Tämän asennuksesta ja asetuksista lisää kappaleessa 5.1.4. Lisäksi, ennen kuin Remotesta on hyötyä, tulle pelin kontrollit olla luotuna mobiilialustalle. Tämä tarkoittaa joko väliaikaisten virtuaalinäppäinten tai kosketuskontrollien luomista.

3 Muut työkalut ja audio

Vaikka Unitylla olisi periaatteessa mahdollista hoitaa pelikehityksen jokainen osa-alue pelimoottorin sisäistä kauppapaikkaa hyödyntäen, pelin audiovisuaalinen puoli hoidettiin ulkoisilla ohjelmilla ja lähteillä. Tekstifonti, sekä osa pelin äänistä ladattiin Unityn Asset Storesta, kun taas grafiikat piirrettiin ulkoisia ohjelmia käyttämällä. Kehitystyön osia ulkoistamisella säästettiin aikaa, joka pystyttiin käyttämään pelin tärkeämpiin ja kiinnostavampiin puoliin.

Pelin grafiikat piirrettiin Inkscape-nimisellä, ilmaisella vektorigrafiikkaeditorilla. Vektorigrafiikalla tarkoitetaan tietokonegrafiikkaa, joka perustuu koordinaatistoon sidottuihin objekteihin, kuten viivoihin, ellipseihin ja monikulmioihin. Itse piirtäminen tapahtuu valmiilla työkaluilla ja kuviot esitetään sekä koordinaatein, että matemaattisin funktioin. Käytännössä tämä tarkoittaa, että piirretyiden kuvion kokoa ja muotoa voi muokata ilman kuvanlaadun menetystä. Kuvioilla piirtäminen on myös huomattavasti helpompaa, joten itsenäisen pelikehittäjän ei tarvitse olla taiteilija luodakseen peleihinsä taidetta.



Kuva 6: Inkscape-piirrustusohjelma

Pelin äänet saatiin kolmansilta osapuolilta hyödyntäen sekä Unityn omaa Asset Storea, sekä ilmaisia ääniä tarjoavaa sivustoa, https://freesound.org/. Tähän päädyttiin oman osaamisen puutteiden sekä projektin aikarajoitusten vuoksi. Pelin audiovisuaalinen puoli pidettiin minimaalisina, ja on todennäköisesti ensimmäinen osa-alue jatkokehitystä suunniteltaessa.

Taustamusiikin lisäksi pelaajahahmo asetettiin soittamaan erilaisia ääniä eri peliobjekteihin osuttaessa. Unityssä pelin äänet soitetaan Audio Manager- nimisellä komponentilla, jota ohjataan muiden komponenttien tavoin ohjelmakoodin sisällä. Kuten mihin tahansa muukin pelin kehityksessä, on audion toistamiseen lukuisia keinoja, mutta tässsä projektissä päädyttiin liittämään tuo Audio Manager pelaajahahmoon.

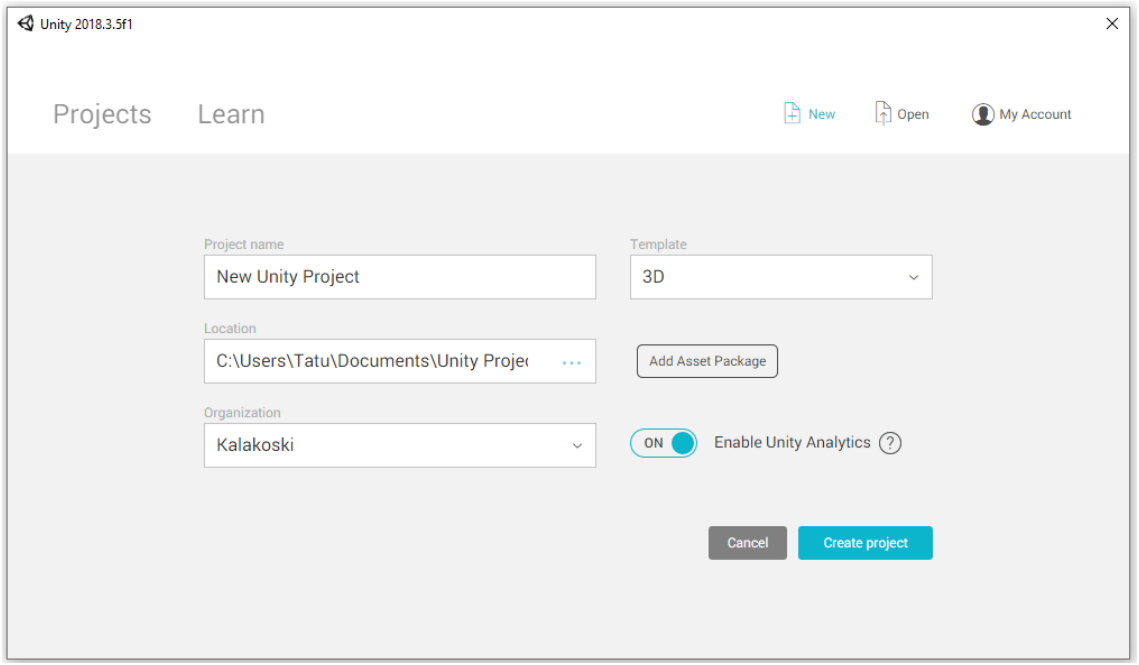
Ulkoisilla ohjelmilla luodut tiedostot saa sisällytettyä Unityn projektiin, joko raahaamalle ne Assets-kansioon, tai suoraan editorista Assets → Import New Asset. Tänä päivänä Unity tukee laajalti lähes kaikki tiedostomuoja, mutta nuo on hyvä tarkistaa ennen assettien luomista, mikäli ollaan käyttämässä näistä harvinaisempia.

4 Pelikehityksen vaiheet

Ennen kuin pelin kehitys voidaan aloittaa, on edessä muutamia askeleita. Unity3D tulee asentaa kehitystyössä käytettävälle tietokoneelle. Tämän lisäksi Unity itsessään tarvitsee tiettyjä työkaluja ja tietokantoja, jotta pelien rakennus onnistuu Android-laitteille.

4.1 Unityn asennus ja uuden projektin luonti

Unity on ladattavissa Unity Technologiesin sivuilta, https://unity.com/, jonka asentaminen noudattaa samaa kaavaa kuin mikä tahansa muu nykyaikainen tietokonesovellus. Unityn käyttö edellyttää rekisteröitymistä, jonka voi suorittaa ennen tai jälkeen Unityn asennuksen. Unityä asentaessa, asennusohjelma tarjoaa suoraan vaihtoehtoisia lisäpaketteja ja alustalaajennuksia, joita käyttäjät voivat valita tarpeidensa mukaan. Tässä projektissa tarvittiin mobiilitukea, joten valitsin android-lisäpaketin. Valintojen jälkeen Unity asentuu koneelle ja on valmiina käytettäväksi.



Kuva 7: Unity Launch- ruutu

Uuden Unity-projektin luominen tapahtuu Projects-välilehdeltä, jolloin Unity avaa ikkunan, jossa valitaan muutamia asetuksia projektia kohden. Esimerkkeinä tärkeimmistä asetuksista ovat projektin rakentaminen joko 2D tai 3D-avaruuteen, projektin nimi, sekä siihen mahdollisesti ladattava Asset Store-materiaali. Kun asetukset on määritetty ja projekti nimetty, Unity rakentaa projektin, jonka jälkeen ruudulle avautuu Unity Editor.

4.2 Android SDK Tools- asennus

Jotta Unityllä on mahdollista rakentaa pelejä android-laitteille, kehitystyössä käytettävältä tietokoneelta tulee löytyä mobiililaitteiden kehitystyön komponentit. Andoridilla tämä tarkoittaa Android SDK:ta. Lisäksi, työasemalta tulee löytyä joko Java JDK tai JRE.

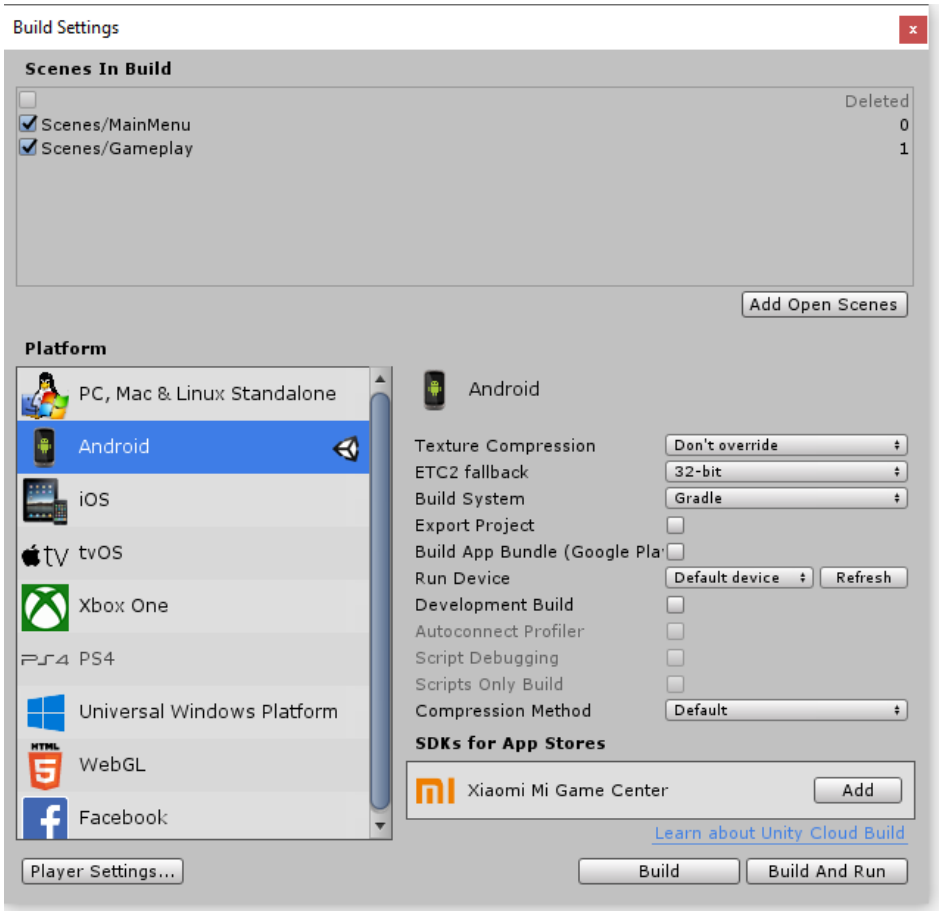
Android SDK on ladattavissa manuaalisesti, jolloin asennus tapahtuu komentoriviä käyttäen. Vaihtoehtoisesti, sekä SDK että Javan komponentit tulevat automaattisesti Android Studio-

nimisen ohjelman mukana. Tässä projektissa päädyttiin asentamaan Android Studio, mutta molemmat keinot ajavat saman asian.

Mainittuja komponentteja asentaessa tärkeintä on painaa muistiin Android SDK:n paikallinen asennussijainti. Asennuksen polku tulee määrittää vielä Unityn sisällä Edit → Preferences → External Tools → Android. Tämän voi myös jättää myöhemmiksi, vasta kun pelistä on ensimmäistä kertaa rakentamassa apk-muotoista pakettia.

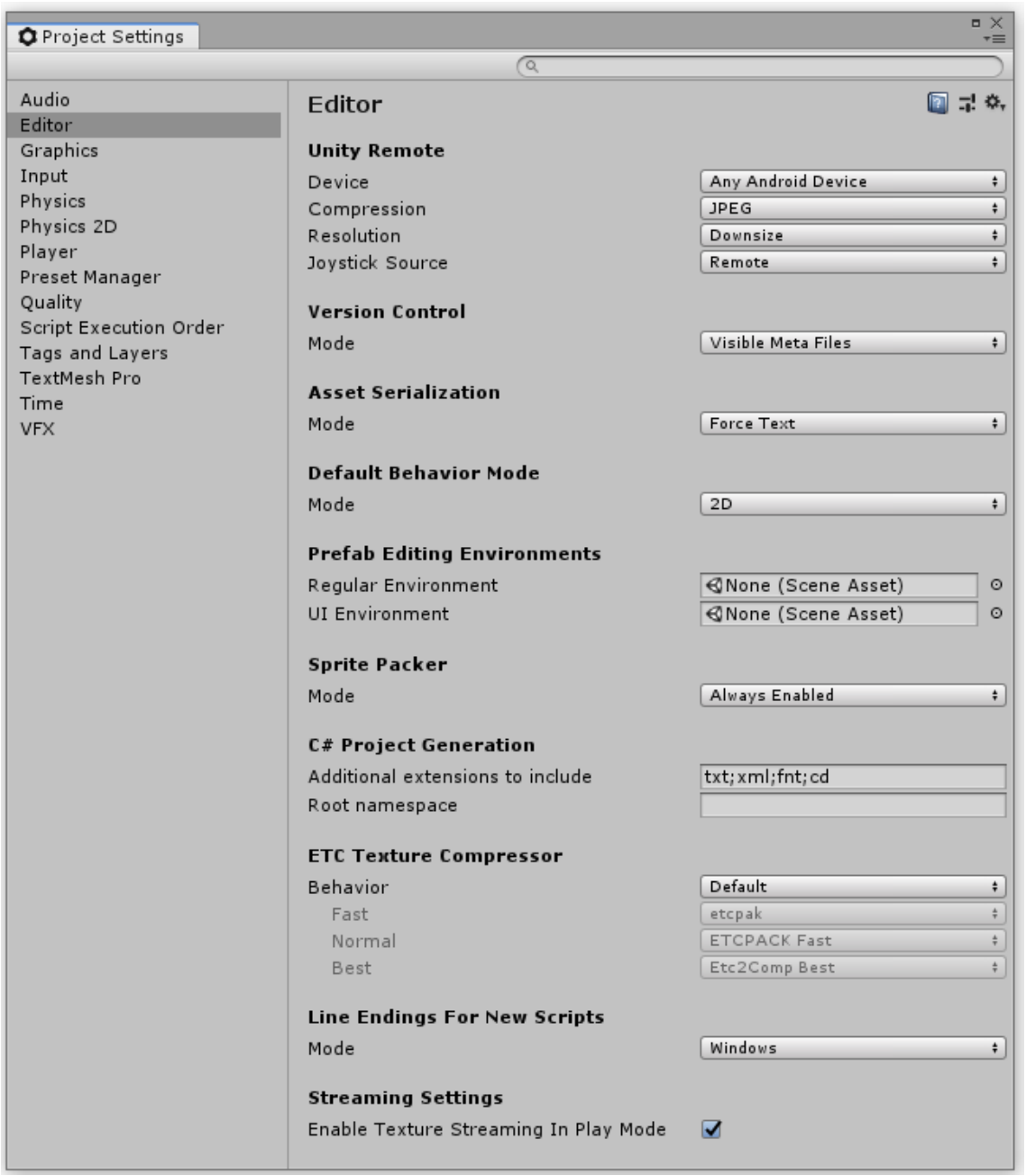
4.3 Mobiiliprojektin asetukset

Ennen itse pelikehityksen aloittamista, kehittäjän on hyvä vielä vilkaista Unityn asetuksia. Mobiilialustalle rakentaessa on tiettyjä asetuksia, jotka on pakollista määrittää. Ensimmäiseksi, on vaihdettava rakennettavan projektin kohdealusta. Tämä tapahtuu File → Build Settings → Platform. Oletuksena, Unity3D rakentaa pelit PC, Mac & Linux Standalone. Kohdealustan saa vaihdettua allaolevalta listalta, joka juurikin koostuu valituista lisäosista Unityn asennuksen yhteydessä. Tässä projektissa valitaan Android ja Switch Platform. Build Settings – ruudulta on valittavissa myös myös muita tärkeitä asetuksia, joihin palataan myöhemmin.



Kuva 8: Unity Build Settings

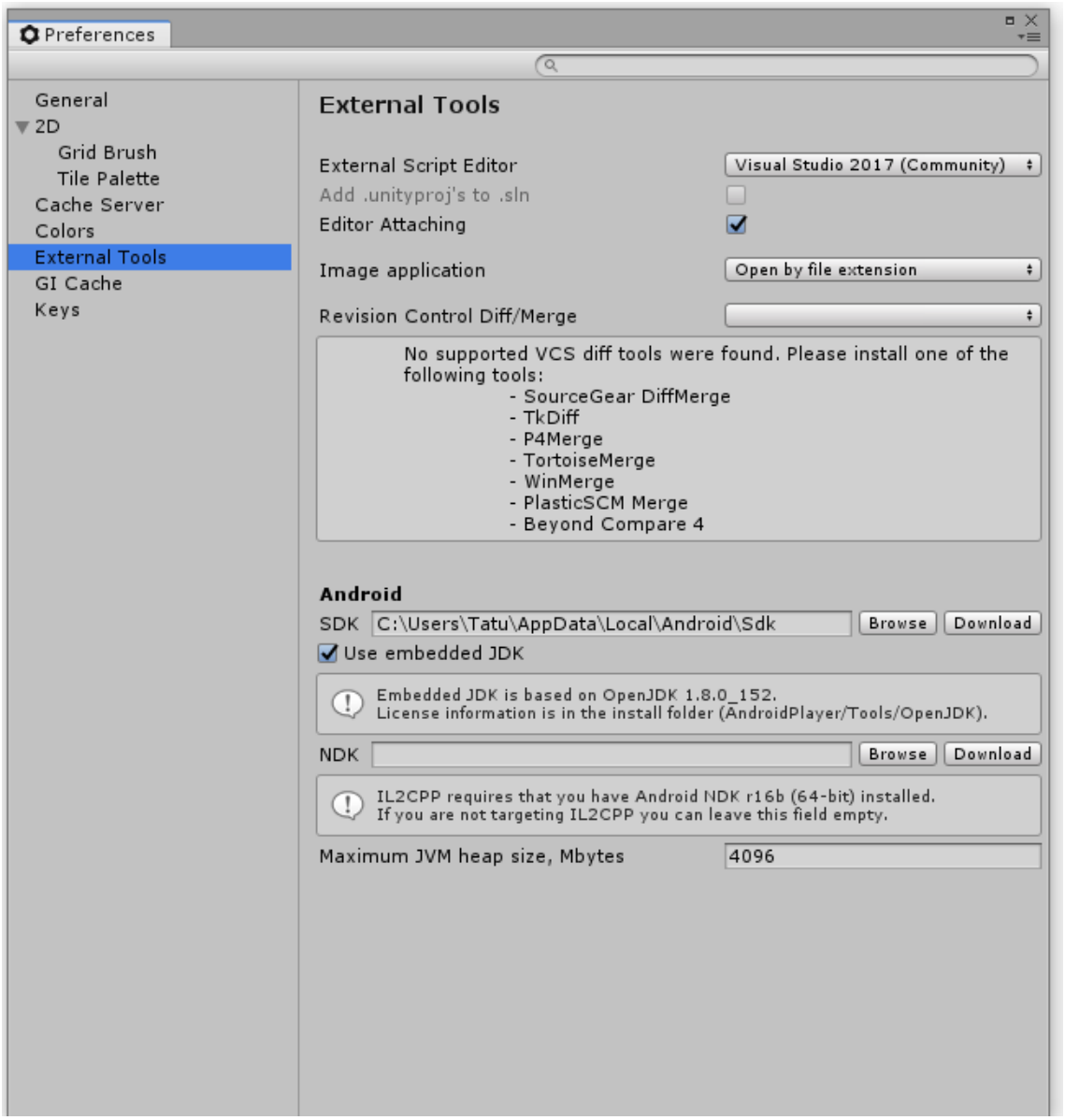
Seuraavaksi siirrytään Edit → Project Settings, josta löytyvät projektikohtaiset asetukset, jotka ovat vapaasti kehittäjän tarpeiden mukaan muokattavissa. Mobiilipelin kehityksen kannalta oleelliset asetukset löytyvät Edit-välilehden alta, josta tulee varmistaa, että Unity Remoten laitteeksi on valittu 'Any Android Device.' Project Settings → Player-välilehti puolestaan tarjoaa välttämättömiä asetuksia pelin julkaisuun liittyen, joihin palataan myöhemmin.



Kuva 9: Unity Project Settings

Viimeiset, ja mobiilipelikehityksen kannalta erittäin tärkeät, jo ennalta mainitut asetukset löytyvät Edit → Preferences → External Tools. Jos Android SDK:ta ei ole vielä asennettu,

tulee se nyt viimeistään tehdä, ja asettaa asennussijainti Android-kohdan alta. External Tools- välilehden alta onnistuu myös ohjelmointiympäristön vaihto. Mainittakoon vielä, että Unity3D tarjoaa edellämainittujen asetusten lisäksi satoja vaihtoehtoisia asetuksia, sekä kehitysympäristöön, että rakennettaviin projekteihin. Tämän projektin yhteydessä nämä jätettiin kuitenkin enimmäkseenn koskemattomiksi.

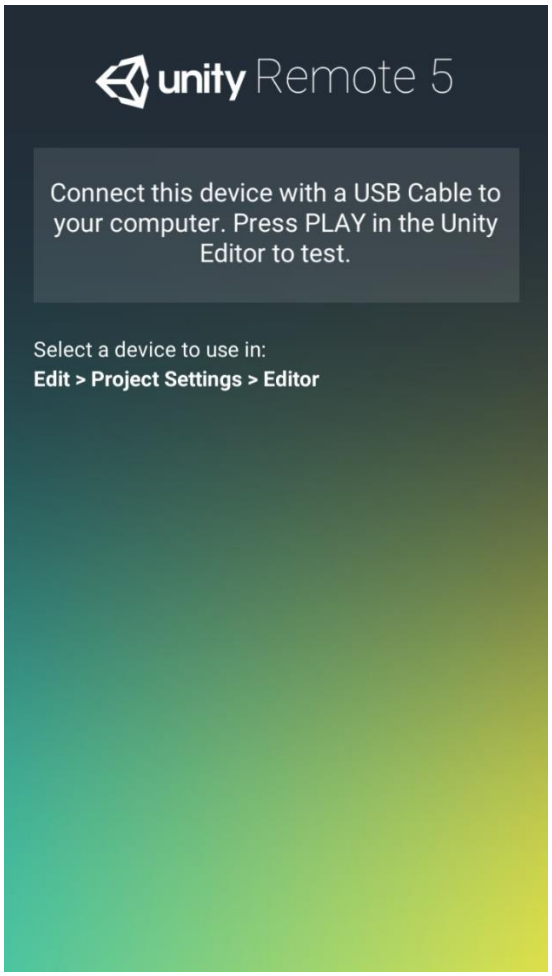


Kuva 10: Unity External Tools ja Android SDK

4.4 Unity Remoten asennus

Siinä missä yllämainitut toimenpiteet ovat mobiilikehityksen kannalta välttämättömiä, on Unity Remote täysin valinnainen lisätyökalu. Mikäli opinnäytetyön askeleita on tähän asti noudatettu, ei Remoten käyttöönotossa ole enää jäljellä kuin sen asennus mobiililaitteelle,

sekä kyseisen laitteen asetuksien tarkastaminen. Unity Remote on ladattavissa, kuten mikä tahansa muu mobiilisovellus, Play-kaupasta.



Kuva 11: Unity Remoten perusnäkymä ennen pelikuvaa

Kun Unity Remote on ladattu, siirrytään puhelimen asetuksiin. Mobiilisovelluksia kehittäessä, joudutaan niitä välttämättä testaamaan kohdelaitteilla. Tätä varten näiden laitteiden kehittäjäasetukset tulee ottaa käyttöön. Tämä tapahtuu useimmilla android-laitteilla näpäyttämällä useaan kertaan Koontiversio-nimistä kohtaa puhelimen asetuksista. Olemassaolevien laitteidan määrän vuoksi, Koontiversion sijainti asetuksissa vaihtelee hieman. Esimerkkinä Samsung J5: Asetukset → Tietoja puhelimesta → Ohjelmiston tiedot → Koontiversio.

Kun puhelimen kehittäjäasetukset on otettu käyttöön, ilmestyy puhelimen asetuksiin uusi alavalikko nimeltään Sovelluskehittäjien asetukset. Täältä tulee vielä ottaa käyttöön asetus ’USB-virheenkorjaus.’ Tämän jälkeen puhelin on valmiina kehitystyöhön ja testaukseen. Käynnistetään Unity Remote- sovellus puhelimella, sitten yhdistetään laite USB-kaapelilla kehityksessä käytettävään tietokoneeseen. Seuraavan kerran Unityn käynnistäessä Remoten

pitäisi olla oikeaoppisesti yhdistettynä. Tämän voi varmistaa yksinkertaisesti Play- nappia painamalla, jolloin Unity lähettää pelikuvan samanaikaisesti myös puhelimen näytölle.

Unity Remote on tunnettu epävakaasta toiminnastaan ja asennuksen hankaluudesta, joten kyseisein ohjelman käyttöönotossa voidaan olettaa hankaluuksia, jopa ohjeita seuraamalla. Mikäli kuvaa ei saada siirtymään puhelimelle, on syytä ensin varmistaa puhelimen olevan ajantasalla ja ajureiden olevan kunnossa. Muita mahdollisia syitä kuvan puuttumiseen on huono USB-yhteys, joka syntyy heikosti liitetystä, tai heiluvasta liitännästä. Myös ohjelmien, Unityn ja Remoten, käynnistysjärjestys voi vaikuttaa ohjelman toimimattomuuteen. Itselläni parhaiten on toiminut seuraava järjestys: Unity Remote päälle → USB kiinni → Unity päälle.

4.5 Peli-idea

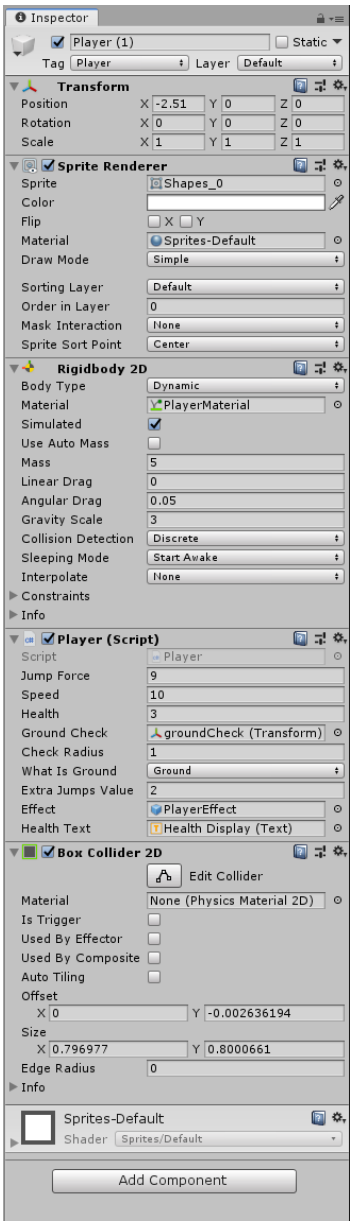
Jokaisen sovelluksen kehitys lähtee ideasta. Pelikehityksessä tällä tarkoitetaan pelin selkärangan rakentavaa, keskeistä pelimekaniikkaa; mikä on pelaajan tavoite, ja miten hän sen saavuttaa. Pelin perusmekaniikan jälkeen ideaa lähdetään laajentamaan; siihen yhdistetään ominaisuuksia, vihollisia, tasoja ja muuttujia niin kauan kunnes pelistä on saatu tarpeeksi mielenkiintoinen, jotta pelaajat jaksavat siihen tarttua tunti toisensa perään. Tätä osa-aluetta pelikehityksessä kutsutaan Game Designiksi, eli pelisuunnitteluksi.

Pelisuunnittelussa tarvitaan mielikuvitusta, sekä kykyä realisoida tämä näkymään ja toimimaan ruudulla. Esimerkkinä tämän projektin peli: Endless, jossa pelaajan tulee kerätä mahdollisimman paljon pisteitä. Tämä tapahtuu yksinkertaisesti vain selviytymällä. Pelaaja sijoitetaan loppumattomaan putkeen, jossa hänen eteensä luodaan alati kiihtyvään tahtiin vihollisia, joita hänen tulee väistellä. Pelaajan kontrollit haluttiin pitää yksinkertaisena, ja loppuen lopuksi päädyttiin kahteen toiminnallisuuteen: pelaaja voi hypätä kosketuksella ja kääntää painovoiman ylösalaisin pyyhkäisemällä. Tämä on pelin ’core’ mekaniikka.

Kun pelin alustava idea on keksitty ja rakennettu näytölle, voi projektin lisäkehitys alkaa. Peliin luodaan lisää muuttujia, jotka lisäävät tai mahdollisesti helpottavat pelaajan kokemusta; lisää vihollistyyppejä, tasoja, bonuksia. Peli-idean laajennus jatkuu läpi kehitystyön, ja jopa pelin julkaisun jälkeen kirjoitettavilla päivityksillä.

4.6 Ohjelmointi

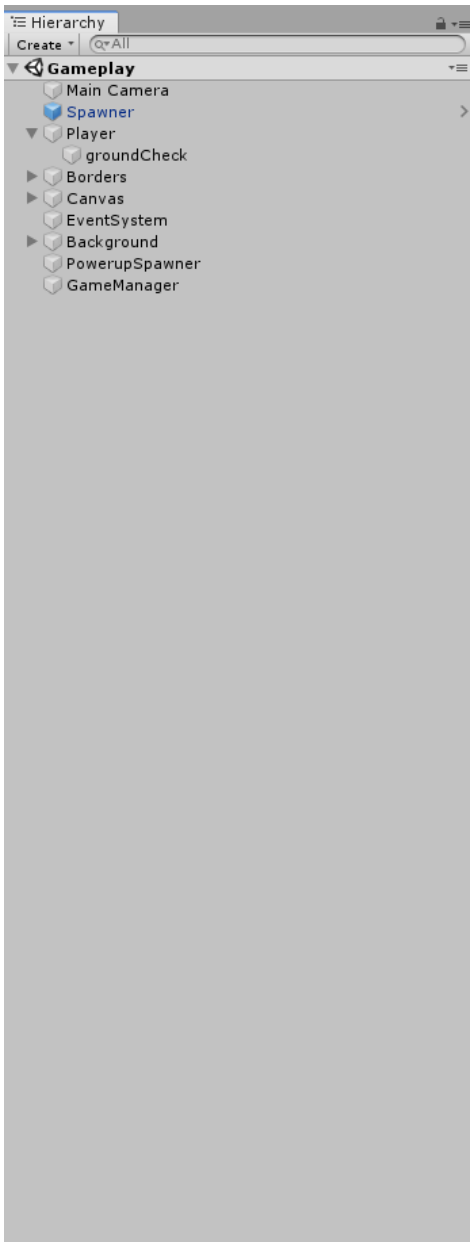
Aivan kuten muutkin sovellukset, pelit ovat loppuen lopuksi vain koodipätkää, ja ohjelmointiin kuluukin helposti suurin osa pelikehityksestä, mikäli yksityiskohtaista peligrafiikkaa ei tarvitse piirtää tai tarinoita kirjoitella. Unityllä, ohjelmointi tapahtuu edellämainitulla Microsoft Visual Studio 2017:lla, mutta ohjelmointiympäristön voi vaihtaa mieluisakseen: Edit → Preferences → External Tools. Ohjelmointikielenä toimii C#.



Kuva 12: Inspector-näkymä Player-objektille

Unity on kohtauksiin (Scenes) perustuva pelimoottori, mikä tarkoittaa, että pelit rakentuvat erilaisille kohtauksille, joihin peliobjektit sijoitetaan. Se, mitä nämä kohtaukset käytännössä tarkoittavat, ovat paljolti kehittäjän itse määriteltävissä, mutta useimmiten nämä voidaan käsittää pelin tasoina ja erilaisina tauko- ja valikkotiloina.

Peliobjektit ovat puolestaan näille näyttämöille sijoitettavia kohteita, jotka muodostavat pelin sisällön. Unityssä peliobjekti on laaja, ja mahdollisesti abstraktikin, käsite, ja voi tarkoittaa kaikkea täysin tyhjästä objektista aina valoihin ja kameroihin asti. Näitä objekteja voidaan muokata ja niihin voidaan sisällyttää muita objekteja ja komponentteja lähes rajoituksetta Unity Editorin Inspector-näkymää käyttäen.



Kuva 13: Unity Hierarchy-näkymä Gameplay-kohtaukselle

Tämän projektin peli pidettiin tarkoituksenmukaisesti yksinkertaisena. Näyttämöitä on pelissä kaksi: päävalikko, jolla on alivalikkona pelin asetusten muokkaamiseen oma Options-valikko, sekä pelin taso. Lukuisten eri tasojen sijaan päädyttiin luomaan vain yksi, alati vaikeutuva taso, teknisen osaamisen ja aikarajoituksen vuoksi. Pelin päävalikossa on kolme vaihtoehtoa: ’Play’, ’Options’ ja ’Quit’. Ensimmäinen käynnistää pelin, toisesta päästään tarkastelemaan pelaajan piste-ennätystä, sekä säätämään pelin äänenvoimakkuutta, ja viimeinen lopettaa pelin, palauttaen käyttäjän mobiililaitteen aloitusnäkymään.

4.6.1 Komponentit

Unityssä jokainen peliobjekti rakentuu Unityssä yhdistelemällä lukuisia ominaisuuksia ja osia yhteen. Näitä kutsutaan komponenteiksi, joiden lisääminen objekteihin tapahtuu ’Add Component’ napilla Inspector-valikosta. Vastaavasti, valmiiden objektien komponentteja voidaan vapaasti poistaa tai korvata kehittäjän toiveiden mukaan.

Valmiita komponentteja löytyy Unitystä sadoittain, ja niistä jokaista voidaan muokata lähes rajattomasti käyttötarkoitukseen sopivaksi. Komponenttien olemassa olevia ominaisuuksia muokataan Inspector-valikosta, ja valittavissa olevat asetukset määritellään komponenttien scripteissä. Valmiiden, Unitystä löytyvien komponenttien lisäksi, kehittäjän on täysin mahdollista luoda omia komponenttejaan tyhjästä, mikä toimiikin edellä mainitun Asset Storen kantavana voimana.

Valmis peliobjekti voi esimerkiksi sisältää Sprite Render-nimisen komponentin, joka piirtää tälle valitun grafiikan näytölle, Scriptin, eli ohjelmakoodin, joka määrittää objektin käyttäytymistä, sekä Box Colliderin, joka tarkkailee objektin osumisen toiseen peliobjektiin. Edellämainittujen komponenttien lisäksi, projektin pelissä hyödynnetteein RigidBody2D-komponenttia, joka vastaa Unityn kaksiulotteisesta fysiikkamallinnuksesta.

4.6.2 Peliobjektit ja peliobjektiprototyypit

Unityssä peliobjektit rakennetaan käytännössä kahdessa paikassa: ohjelmointiympäristössä koodaamalla sekä Unity Editorin Inspectorissa, jossa niiden ominaisuuksia ja osia hallitaan. Mainittakoon kuitenkin, että kaiken oleellisen luominen pelkästään ohjelmointiympäristöä käyttämällä on mahdollista, jokseenkin työlästä. Unity Editorin voi käsittää graafisena käyttöliittymänä (GUI, Graphical User Interface), kun taas Microsoft Visual Studion kaltainen ohjelmointiympäristö on tekstipohjainen käyttöliittymä.

Jokaisen pelin tärkein objekti on pelaajan ohjattavissa oleva hahmo. Tässä pelissä pelihahmoksi valittiin valkoinen kuutio, jota liikuttamalla pyritään välttämään vahinkoa. Aluksi, Unityn Scene-maisemaan luodaan pelaajahahmo luomalla uusi tyhjä objekti, Empty Object ’Inspector’-välilehdeltä. Objektia klikkaamalla, joko Hierarchy-välilehdessä tai Scene-näkymässä, se avautuu Inspector-välilehdelle, jossa tämän hallinta tapahtuu. Add Component-nappia painamalla avautuu lista kaikista Unityn tarjoamista komponenteista, jotka objektille on mahdollista lisätä. Komponenteista merkittävimmät, tässä projektissa käytettävät, ovat Script, Sprite Renderer, RigidBody2D, sekä Box/Circle Collider 2D. Nämä vastaavat peliobjektin käyttäytymisestä, grafiikasta, fysiikkamallinnuksesta, sekä kosketuskontakteista.

Muita pelin kannalta elintärkeitä objekteja ovat erilaiset viholliset, jotka voidaan tämän projektin tapauksessa rakentaa samalla pohjalle kuin pelaaja. Tämän perustan muokkaaminen tapahtuu lisäämällä ja poistamalla komponentteja, sekä niitä muokkaamalla. Uusia

vihollistyyppejä, sekä aivan uusia luokkia kuten esineitä tai boosteja, voidaan luoda esimerkiksi kirjoittamalla niistä pois pelaajan kontrollit ja lisäämällä niiden tilalle uudet, mutta jättämällä muut komponentit liki alkuperäisiksi. Todellisuudessa, suurissa AAA-luokan peleissä, erilaisia luokkia, ominaisuuksia, olioita on järjetön määrä, joiden luomiseen tarvitaan monisatapäistä henkilöstöä, mutta pientä mobiilipeliä luodessa, on periaatteessa mahdollista rakentaa lähes jokainen pelissä käytettävä objekti muutaman peruspohjan päälle. Tämä yksinkertaistaa ja nopeuttaa pelin ohjelmointiosuutta. Näitä peruspohjia, eli prototyyppejä kutsutaan Unityssä prefabeiksi. Prefab luodaan raahaamalla Unity Editorin Hierarchy-näkymästä haluttu objekti Asset-kansioon, useimmiten vielä erikseen luotuun Prefabs-kansioon.

Liikkuvien peliobjektien lisäksi, peli tarvitsee rajat, joiden sisällä kaikki tapahtuu. Tämän voi hoitaa lukemattoman monella tapaa, kuten rajoittamalla peliobjektien liikkuvuutta tiettyjen koordinaattien sisään, mutta itse päädyin luomaan pelin seinät kiinteillä peliobjekteilla, joihin iskeytyessä pelaajahahmo pysähtyy, ja viholliset kimpoavat takaisin. Rajojen luonti erillisinä peliobjekteina mahdollistaa myös niiden muokkaamisen pelin edetessä.

4.6.3 Kontrollit

Suurin ero mobiilipelien kontrolleissa muihin alustoihin verrattuna ovat kosketuksiin ja kallistuksiin perustuva ohjaus. Siinä missä mobiililaitteiden kiihtyvyysantureiden (accelerometer) käyttämiseen jouduttiin aikoinaan kehittämään aivan uudenlaiset funktiot ja periaatteet, kosketukseen perustuva ohjaus puolestaan noudattavat suurimmaksi osaksi tavallisia, hiirellä tapahtuvia ohjausliikkeitä, kuten klikkausta sekä raahausta. Tässä projektissa ei käytetty kiihtyvyysantureita.

Projektissa luodun peliprototyypin ohjaus perustuu sekä näpäytyksiin (tap), että pyyhkäyksiin (swipe). Pelin kontrollit pyrittiin pitämään projektin skaalan huomioiden yksinkertaisina: kosketus saa pelaajahahmon hyppäämään ja ylös- ja alaspain suuntautuvat pyyhkäiset puolestaan kääntävät pelin painovoiman vastaavasti ylös- tai alaspäin. Perusohjausten lisäksi, Unityssä on valmiiksi määritelty kosketusten ominaisuuksia, joita hyödyntämällä pelin kontrolleja saadaan säädettyä. Unity tunnistaa samanaikaiseti tapahtuvat useat kosketukset, kosketusten sijainnin mobiililaitteen näytöillä, että kosketusten eri vaiheet, kuten alku (began), liikutettu (moved) ja loppu (ended).

Pelin valikoissa navigoiminen tapahtuu UI-elementtejä koskettamalla. UI-elementit ovat sidottu skaalautumaan käytettävän näytön koon perusteella, ja tunnistamaan kosketukset tiettyjen pisteiden sisällä koordinaatistossa. Kuten monet muutkin asiat, valikoiden toimivuus on kehittäjän määriteltävissä.

4.6.4 Käyttöliittymä

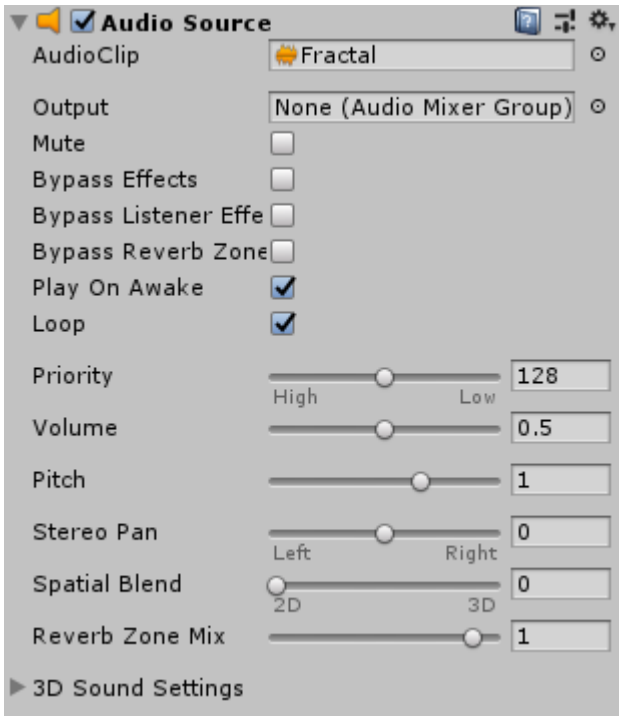
Käyttöliittymällä (User Interface, UI), peleistä puhuttaessa, tarkoitetaan pelin ”päälle” piirrettäviä elementtejä, jotka usein tarjoavat kriittistä tietoa pelistä, kuten pelaajan piste- ja elinvoimatilanteesta. Käyttöliittymä pitää sisällään abstrakteja objekteja, kuten tekstielementtejä ja numeroita, sekä mahdollisesti erilaisia liukuvalikoita ja nappuloita. Myös pelien valikot rakennetaan lähes poikkeuksetta vastaavalle, erilliselle tasolle, aivan kuten niissä navigointikin. Unityssä, kaikki käyttöliittymät ja usein myös pelin HUD, eli Heads-Up-Display (kirjaimellisesti heijastusnäyttö) rakennetaan Canvas-nimisen objektin päälle. Canvas-objekti itsessään toimii samoin periiaattein, kuin mikä tahansa muukin peliobjekti. Sekä sitä itseään, että sen kompponentteja voidaan muokata, laajentaa ja poistaa tavalliseen malliin. Suurimpana erona on Canvaksen sijainti pelinäytöllä verrattuna muihin objekteihin, sillä se piirretään ikään kuin muun pelin päälle.

Projektin pelissä user interface voidaan jakaa kahteen osa-alueeseen: pelin valikoihin, joissa ei ole muita objekteja kuin UI-elementtejä, kuten tekstiä, nappuloita ja liuteltavia valikoita, sekä itse pelin HUD:iin, johon luodaan pelaajan elinvoima, sekä senhetkiset pisteet, että pelaajan piste-ennätys. Pelin valikot koostuvat teksti, näppäin, slider-objekteista, jotka ovat sijoitettu Canvas-objektille. Päävalikon lisäksi pelistä löytyy ’Options’ valikko, jossa voidaan muuttaa erilaisia pelin asetuksia, kuten äänenvoimakkuutta.

User Interface-elementtejä rakentaessa on tärkeää varmistaa niiden skaalautuvuus erikokoisille näytöille, jotta käytettävät teksit eivät veny muodottomaksi tai jää kokonaan ruudun ulkopuolelle. Suoraviivaisin ratkaisu on valita Canvas-elementin Inspector valikosta Canvas Scaler-komponentin alta UI Scale Mode ja asettaa se skaalautumaan laitteiden näytön koon perusteella (Scale With Screen Size). Samalla komponentilla voidaan myös asettaa referenssiresoluutio pelille. Toki Unity tarjoaa näiden lisäksi runsaasti erilaisia säätöjä ja hienosäätöjä pelin resoluution ja skaalauksen suhteen, mutta niihin paneutuminen ei ollut tämän projektin työssä tarpeellista.

4.6.5 Audio

Viimeinen laajempi osa-alue, johon projektissa keskityttiin oli pelin audio. Pelin äänet ladattiin sekä Unityn Asset Storesta, sekä ulkoiselta nettisivulta. Audiotiedostojen, kuten kaikkien muidenkin komponenttien, käyttämiseen löytyy tusinoittain erilaisia vaihtoehtoja, mutta projektin pelin skaalan mukaisesti tämä pidettiin mahdollisimman yksinkertaisena.



Kuva 14: Audio Source -komponentti Pelaaja-objektille

Tässä tapauksessa audiotiedostojen hallinta tapahtuu Audio Source-nimisellä, Pelaaja-objektiin liitetyllä komponentilla, jota hallitaan vastaavasti Pelaajan scriptillä. Pelin musiikit asetettiin looppaamaan, eli jatkumaan ikuisesti niin kauan kuin pelaajan hahmo on hengissä ja loppumaan kun peli hävitään. Erilaiset kontakti-äänet puolestaan asetettiin soitettavaksi aina kun pelaajan hahmon Box Collider havaitsee osuman toisen peliobjektin kanssa.

Audio Source-komponetti tarjoaa muutamia mielenkiintoisia asetuksia 3D-avaruudessa toimiville peleille, kuten erilaiset etäisyyteen perustuvat kaiku- ja vaimennusasetukset. Projektin pelissä vaihtoehtoisiin lisäaasetuksiin ei kuitenkaan koskettu Loop- ja Play On Awake- asetuksia lukuunottamatta.

4.7 Pelin viimeistely & testaus

Pelien testaus toimii kuten minkä tahansa muun sovelluksen testaus, eli se yritetään rikkoa. Mitä enemmän tähän voi aikaa ja resursseja sijoittaa, sitä valmiimpi tuotos saadaan aikaan. Optimaalisesti, pelille löytyy jo kehitysvaiheessa useampi testaaja prosessin jokaisessa vaiheessa. Myös julkaisun jälkeiset, pelin ensimmäisten versioiden pelaajat voidaan laskea eräänlaisiksi testaajiksi, joskin tehtyjä muutoksia kutsutaan tuolloin päivityksiksi. Se, miten testaus hyödyttää peliä ja sen kehittäjiä, on täysin tilannekohtainen. Esimerkkinä projektin peli, jonka kontrollit saatiin huomattavan paljon responsiivisimmiksi verrattuna lähtötilanteeseen.

Pelin viimeistelyllä tarkoitetaan erinäisten osien hiontaa, joka perustuukin lähinnä testauksesta saatuihin tuloksiin. Viimeistely tapahtuu kehitysprosessin aikana pienempinä korjauksina ja hienosäätöinä, sekä prosessin lopulla, kun peli on saatu ’valmiiksi’, eli kun kaikki siihen tähän mennessä sisällytetyt osa-alueet toimivat kuten niiden sopii olettaa; eli pelin voi aloittaa, sitä pystyy pelaamaan, ja sen pystyy lopettamaan.

Eniten aikaavievä osio näistä loppuprosessin viimeistelyistä on optimointi, jolla tarkoittaa pelin erinäisten osien kehitystä suoraviivaisimmiksi ja tehokkaammiksi. Eräs Unityllä käytettävä työkalu tähän on Unityn Profiler-ikkuna, joka tarkkailee ja tallentaa pelin teknistä puolta, sekä sen eri osa-alueita, pelin pyöriessä kohdelaitteella. Optimoinnin yhteydessä pelin ohjelmoijat palaavat kirjoittamansa koodin ääreen etsien mahdollisia parannuskohteita. Tällä voidaan tarkoittaa esimerkiksi tyhjien koodipätkien poistamista, tai monimutkaisten ja epätehokkaiden ehtolauseiden korvaamista ja suoraviivaistamista. Vastaavasti, pelin visuaalisen puolen muodostavia grafiikoita on hyvä tarkkailla. Ylisuuret tiedostot, etenkin kun niitä piirretään useampi samanaikaisesti näytölle, vaativat laitteelta sekä tehoja, että tallennustilaa. Tämä on mobiilisovelluksissa erityisen tärkeää, ottaen huomioon mobiililaitteiden fyysiset rajoitukset.

Optimoinnin ja pienten korjausten lisäksi pelin viimeistelyyn voidaan sisällyttää mielenkiintoisempiakin asioita, kuten erilaisten efektien lisäämistä peliin. Projektin peliin lisättiin loppuvaiheella kevyt partikkelisysteemi, jolla luotiin illuusio vauhdista, puhaltamalla pieniä partikkeleita pelaajahahmoa vastaan. Tällä pienellä muutoksella saatiin staattisesta mustasta taustasta elävämpi. Vastaavasti, vihollistyypit ja poimittavat bonukset laitettiin ’räjähtämään’ kappaleiksi, pelaajan näihin osuessa.

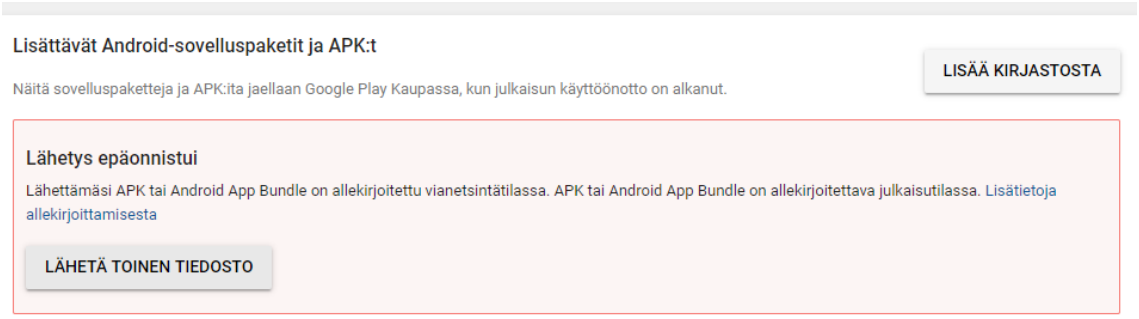
4.8 Pelin rakennus androidille

Jotta peli saadaan julkaistua Play-kaupassa, tulee siitä rakentaa .apk-tiedosto, joka toimii pelin pakettina. Lisäksi, Play-kauppa edellyttää allekirjoitusavaimia sovelluksille, kehittäjän henkilöllisyyden varmentamiseen. Unityllä tämä vaatii pieniä muutoksia pelin asetuksissa.

4.8.1 Apk-tiedoston rakennus

Pelien rakentaminen Unityllä noudattaa tuttua kaavaa; erilaisia asetuksia ja hienosäätöjä löytyy pelottavan paljon. Tässä projektissa keskitytään kuitenkin vain oleellisimpiin seikkoihin. Pelin rakentaminen Unityllä vaatii seuraavia tietoja, jotka hoidetaan ’Player Settings’-ikkunassa. Peli vaatii ensinnäkin Bundle Identifier:in, joka muodostuu pelin sekä kehittäjän tai studion nimestä. Bundle Identifier toimii paketin väliaikaisena nimenä, ja sen määritys tapahtuu Edit→Project Settings→Player Settings → Identification → Package Name. Bundle Identifier noudattaa nimeämiskäytäntöä com.yourCompanyName.yourGameName. Nämä voi tässä kohtaa valita, ja mahdollisesti muuttaa, mikäli alkuperäinen pelin tai kehittäjän yrityksen nimet halutaan muuttaa. Bundle identifier:illä on muutamia vaatimuksia.

Ensinnäkin, tulee sen olla ainutlaatuinen. Sitä ei myöskään Play-kaupan julkaisun jälkeen voi muuttaa.



Kuva 15: Debug-tilassa allekirjoitetun appk-tiedoston latausyritys

Androidille sovelluksen, tässsä tapauksessa pelin, rakentamisella tarkoitetaan .apk-muotoisen pakentin luomista. Kun Unityssä on saatu halutut ja tarvittava Build-asetukset määriteltyä, tapahtuu se Build-nappia painamalla Build Settings-valikosta. Tämä rakentaa pelin, jonka sijainnin tietokoneella saa valita aukeavasta valikosta. Tämä on hyvä sijoittaa pelin juurikansion sisälle luotavaan Versions-alakansioon. Juurikansiolla tarkoitetaa sijaintia, missä muutkin pelitiedostot sijaitsevat omissa alakansioissaan. Tähän alikansioon on myös hyvä käytäntö tallentaa kaikki pelin eri versiot. Mikäli peliä kehittäessä tapahtuu vakavampaa virhettä, saadaan näin vanhempi, aiemmin toiminut versio rakennettua. Mikäli Unity Remotea ei ole käytetty, pelin testaaminen onnistuu tässä samalla, liittämällä mobiililaitteen USB-kaapelilla koneeseen ennen rakennuksen aloittamista. Tämä kopio syntyvän paketin myös kohdelaitteella, josta se voidaan käynnistää kuten muutkin sovellukset.

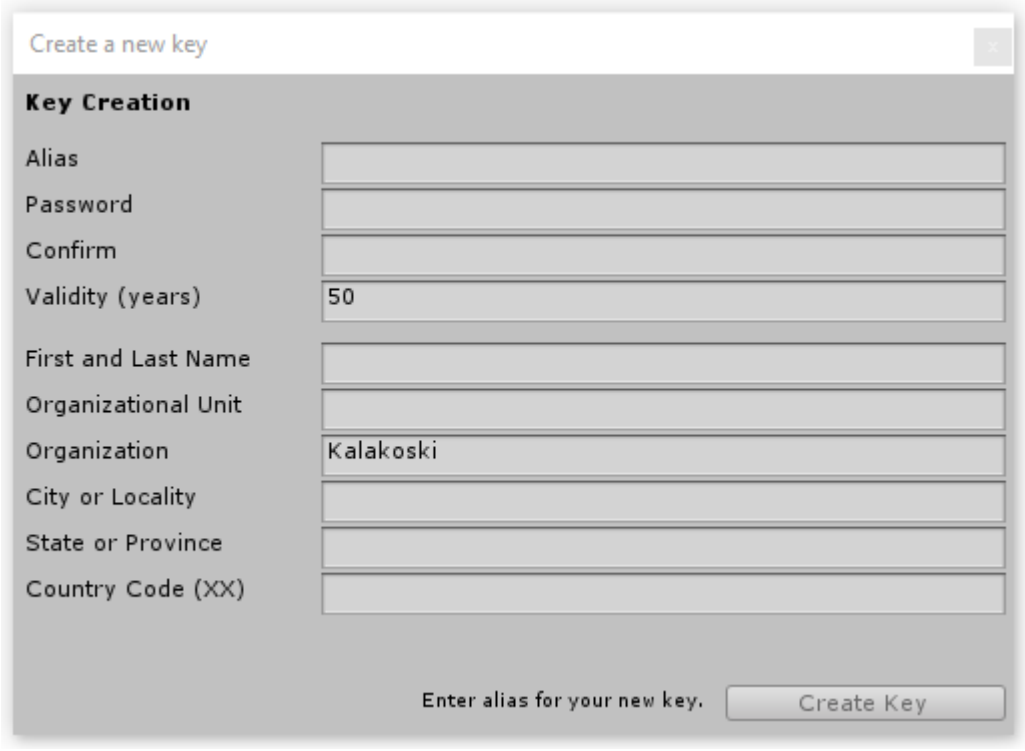
4.8.2 Allekirjoitusavaimet ja niiden luonti

Lyhyesti, allekirjoitus, sekä salausavaimilla tarkoitetaan kryptattuja teksitiedostoja, joilla sovelluskehittäjän henkilöllisyys varmennetaan sovellusta ladattaessa kauppapaikoille, kuten Play-kauppaan. Allekirjoitusavaimet ovat ainutlaatuisia ja niiden avulla kauppapaikat voivat varmentaa sovelluksen tulevan virallisista lähteistä, rekisteröidyiltä kehittäjiltä. Google Consoleen ei pysty lataamaan allekirjoittamattomia apk-tiedostoja. Unityn tapauksessa virheilmoituksena saadaan maininta vianetsintätilassa (debug) allekirjoitettu apk- tai Android App Bundle.

Kuva 16: Unityn Publishing Settings

Allekirjoituksen muutos tapahtuu Unityllä seuraavasti: Build Settings → Player Settings →Publishing Settings. Keystore-otsikon alta valitaan ’Create new keystore’, minkä jälkeen etsitään kyseinen debug-muotoinen keystore-tiedosto, jonka Unity luo automaattisesti

käyttäjän kansioon C\Users\KÄYTTÄJÄ\.android, nimellä debug.keystore. Ennen tiedoston valintaa, tuo kannattaa kopioida ja uudelleennimetä vaikka projektin nimellä.



Kuva 17: Unity Key Creation

Kun keystore-tiedosto on valittu, siirrytään Key-otsikon alle. Alias-pudotusvalikosta valitaan kyseinen tiedosto, jonka määrittämiseen avautuvasta ikkunasta valitaan aliakselle nimi, sekä salasana. Muut tiedot ovat vaihtoehtoisia. Kun avain on saatu luotua, tallennetaan lopullinen Unity-projekti, jonka jälkeen rakennetaan apk-tiedosto koneelle ja siirrytään pelin julkaisuun

5 Pelin julkaisu Play-kaupassa

Pelin julkaisu Google Play-kaupassa edellyttää muutamia askeleita, jotka käydään nyt kronologisessa järjestyksessä. Asiat ja osa-alueet, jotka tehdään ovat seuraavanlaiset: Google-tilin luonti ja sovelluskehittäjäksi listautuminen, apk-tiedoston allekirjoitusavaimen luonti, sekä apk:n lataaminen Googlen kehittäjäkonsolille, sovelluksen tietojen, kuten nimen, kuvauksen ja kuvien täyttö, ikärajoituksen arvioiminen, sovelluksen hinnan ja mahdollisten pelinsisäisten ostojen määrittäminen.

5.1 Google käyttäjätili ja android-kehittäjäksi rekisteröityminen

Muiden sovellusten tapaan, jotta peli saadaan julkaistua Play-kaupassa, kehittäjä tarvitsee luonnollisesto google-tilin. Google-tili löytyneekin valmiiksi, mutta seuraava askel on rekisteröityä Android-kehittäjäksi Google Play:n kotisivujen rekisteröitymis-osiossa,

https://play.google.com/apps/publish/signup/#EnterDetailsPlace. Rekisteröityminen noudattaa jotakuinkin yleistä kaavaa, mutta tietojen täydentämisen lisäksi Google perii kertaluontoisen noin 20 euron maksun kehittäjälisenssiä vastaan.

Rekisteröitymisen jälkeen kehittäjän käyttöön aukeaa Google Play Developer Console, jossa kaikkien sovellusten hallinta ja lataus tapahtuu. Kehittäjäkonsolista löytyy myös työkaluja ja tilastoja omien tuotteiden seurantaa, sekä omien sovelluksien arvostelut.

Google Developer Consoleen ei ehditty projektin aikana tutustumaan yksittäisen pelin lataamista pidemmälle. Huomattavaa on, mikäli useampaa peliä tai sovellusta on suunnitelmissa julkaista, omien nettisivujen luominen on suositeltavaa. Samoin on Googlen kauppapaikan sääntöjen ja julkaisemisen käytäntöihin tarkempi perehtyminen, erityisesti jos luotujen pelien halutaan tuottaa rahaa.

5.2 Pelin lataaminen Play-kauppaan

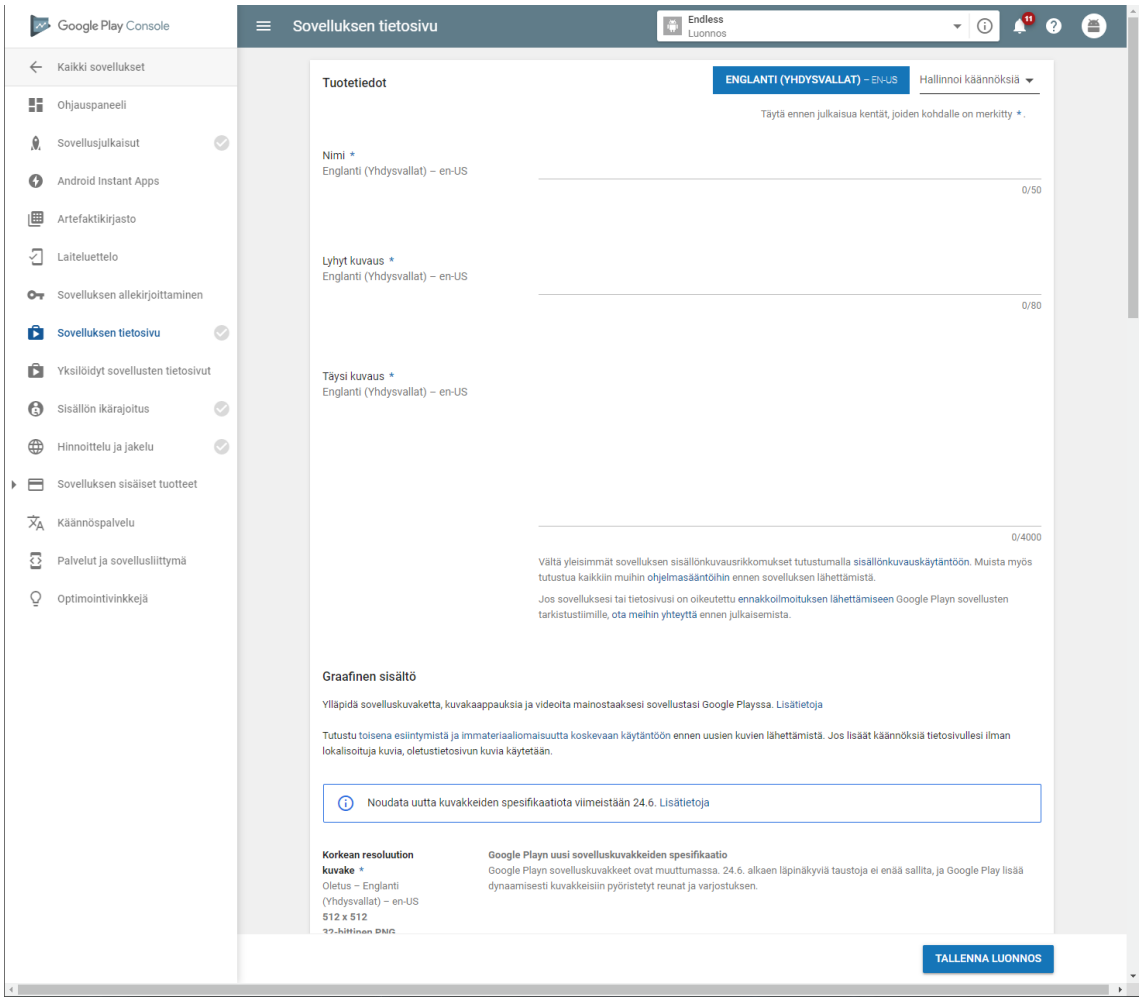
Sovelluksin lisääminen Googlen kehittäjäkonsolille tapahtuu ’Kaikki sovellukset’-sivun napilla ’Luo Sovellus’. Ensin valitaan sovelluksen oletuskieli, sekä sovellukselle nimi. Tässä kohtaa Google Play Console-ikkuna selaimen vasemmalla reunalla muuttuu yleiskatsauksesta sovelluskohtaiseksi. Konsolin käyttöliittymä yleiskatsauksen ja yksittäisten tuotteiden välillä on tehty saumattomaksi.

Peliä tai sovellusta ladatessa Play-kauppaan, harmailla Hyväksytty-merkeillä merkityt välilehdet pakollisia määrittää. Näillä välilehdillä asetetaan tärkeitä asetuksia sovellukseen ja sen jakeluun liittyen. Pakollisiin välilehtiin kuuluvat sovellisjulkaisut, sovelluksen tietosivu, sisällön ikärajoitus, sekä hinnoittelu ja jakelu.

Näiden välilehtien asetukset on mahdollista tarkistaa missä järjestyksessä tahansa. Tässä projektissa tuo tehtiin seuraavasti: sovelluksen tietosivu, sovellusjulkaisut, sisällön ikärajoitus ja viimeisenä hinnoittelu ja jakelu. Kun pakolliset asetukset on määritetty, voidaan mahdollisia lisäasetuksia asettaa, tai siirtyä suoraan sovelluksen julkaisuun.

5.2.1 Sovelluksen tietosivu

Sovelluksen tietosivu avautuu automaattisesti ensimmäisenä uutta sovellusta luodessa. Avautuneeseen lomakkeeseen täytetään erilaisia tietoja pelistä, sekä sen kehittäjästä; kuten kuvaukset pelistä, sen nimi, sekä kehittäjän sähköposti. Lisäksi sovellukselle annetaan ikonikuvan lisäksi vähintään kaksi kuvakaappausta sekä tulee sille valita luokka ja alakategoria.



Kuva 18: Sovelluksen tietosivu

Pelin kuvauksia on kaksi: lyhyt ja täysi kuvaus. Lyhyt kuvaus sisältää maksimissaan 80 merkkiä ja näkyy suoraan pelin etusivulla Play-kaupassa. Täysi kuvaus puolestaan löytyy pelin sivulta Lisätietoja-kohdan alta. Täyteen kuvaukseen on hyvä laajentaa lyhyen kuvauksen tiivistelmä, sekä kertoa mahdollisia lisätietoja. Täysi kuvaus antaa kirjoittaa 4000 merkkiin asti.

Valinnaisiin asetuksiin kuuluvat esimerkiksi kehittäjän kotisivut, puhelinnumero, sekä mahdolliset lisäkuvat ja esittelyvideot. Lisäksi, sovelluksen tietosivun yläkulmasta voidaan määrittää sovelluksen mahdolliset käännökset. Tämä ei kata pelin sisäistä sisältöä, vaan Play-kaupassa käytettävät tietosivut ja kuvaukset. Lomakkeen lopussa olevasta Tietosuojakäytännöstä ei tarvitse tässä kohtaa välittää.

5.2.2 Sovellusjulkaisut

Sovellusjulkaisut-välilehdeltä tapahtuvat apk-pakettien alkuperäiset sekä mahdolliset, tulevaisuuden versiopäivitysten lataukset. ’Tuotanto’- otsikon alta valitaan ’Hallinnoi’ ja

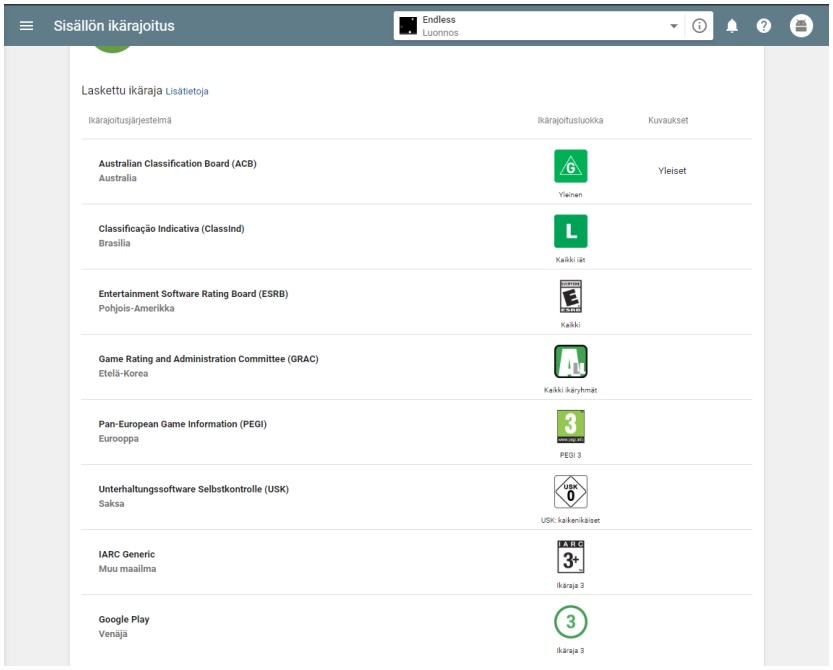
täältä kohdasta ’Luo julkaisu’ päästään sovellus lataamaan konsoliin. Seuraavalta sivulta tulee sallia kohta ’Anna Googlen luoda sovelluksen allekirjoitusavain ja ylläpitää sitä (suositus).’

Ensimmäistä sovellusta luodessa, luetaan ja hyväksytään Googlen oikeus- ja vastuuvapauslomake, joka on syytä lukea läpi mikäli aikoo tulevaisuudessa julkaista lisää sisöltöä Play-kauppaan, etenkin jos mainitulla sisällöllä haluaa tienata elantonsa. Lyhykäisyydessään mainittu oikeus- ja vastuuvapauslomake sisältää normaalin lakipykälätekstin lisäksi lähinnä ammattilaisille suunnattua tietoa. Näihin lukeutuu Googlem 30% rojaltti pelin myynneistä, sekä mainontaan ja tiedonkeruuseen liittyviä sääntöjä.

Mikäli Unityn peli on allekirjoitettuna oikeaoppisesti kohdan 5.6.2 ohjeiden mukaan, tulee tuo luotu apk-tiedosto valita ’Selaa Tiedostoja’-kohdasta. Tämän lisäksi täytetään sivulla olevat kohdat ’Julkaisun nimi’ ja ’Mitä uutta tässä julkaisussa on?’ Tähän kohtaan kirjoitettava nimi ei näy Play-kaupassa, vaan ainoastaan kehittäjäkonsolissa, joten tuoksi on hyvä asettaa pelin tämänhetkinen versionumero. Julkaisun tiedoiksi puolestaan täytetään tehdyt muutokset, mikäli kyseessä on sovelluksen päivityspaketti. Jos ollaan lataamassa ensimmäistä kertaa, voi tähän kirjoittaa pelin kuvauksen.

5.2.3 Sisällön ikärajoitus

Kun pelin apk-paketti on saatu hyväksytysti ladattua Googlen konsolille, siirrytään seuraavaksi kohtaan ’Sisällön ikärajoitus.’ Täällä pelille määritellään ikäluokitus yksinkertaisen lomakkeen avulla. Lomake koostuu kysymyksistä ja Kyllä/Ei-vastauksista, kuten ’Sisältääkö peli väkivaltaa.’ Kun lomake on täytetty, saa se IARC-sertifikaattiin perustuvan ikärajoituksen.



Kuva 19: Sisällön ikärajoitus ja saatu luokitus

Mobiiliympäristössä ikärajoitukset eroavat hieman verrattuna fyysisten tuotteiden myyntiin. Erityisesti, jos peli on maksuton, ei sen lataamiseen tarvita muuta kuin verkkoyhteys. Tämän vuoksi ikärajoituksen noudattaminen on enimmäkseen alaikäisten huoltajien vastuulla. Mikäli mobiililaitteella on alaikäinen henkilö, olisi kyseiselle laitteelle hyvä asettaa lapsilukko.

Samat pelisännöt pätevät myös seuraavan luvun alla olevaan hinnoitteluun. Erityisesti pelinsisäisillä ostoksilla voidaan saada luotua nopeastikin jopa tuhansien eurojen lasku. Vaikkakin nämä ovat enimmäkseen huoltajien vastuulla, tulisi pelin tietosivulla tai kuvauksessa mainita vielä erikseen mahdollisista sopimattomuuksista ja lisämaksuista.

5.2.4 Hinnoittelu ja jakelu

Viimeinen tehtävä ennen sovelluksen lähettämistä julkaistavaksi tapahtuu Hinnoittelu- ja Jakelu-välilehdeltä. Näillä välilehdillä valitaan julkaisumaiden lisäksi pelin maksullisuus sekä mahdolliset pelin sisäiset ostokset. Huomioitavaa on, että kun Maksullinen/Ilmainen on valittu, ei sitä voi enää julkaisun jälkeen muuttaa.

Sivun lopussa on lisää valinnaisia vaihtoehtoja ja määrityksiä, joista pakolliset valinnat on merkitty tähdillä. Valintoihin ja asetuksiin kuuluu asioita, kuten onko peli lapsille suunnattu tai sisältääkö se mainoksia. Lupa-otsikon alta tulee hyväksyä sekä sisältösäännöt että Yhdysvaltain vientilait.

Kun kaikki 6.2-otsikon alta suoritettavat vaiheet on saatu tehtyä, voi pelin lähettää hyväksyttäväksi. Kun kaikki on kunnossa peli ilmestyy Play-Kauppaan onnistuneen arvioinnin jälkeen. Mikäli valmis peli ei täytä Googlen vaatimuksia, peli ei ilmesty Play-kauppaan tarjolle, vaan kehittäjä saa sähköpostiinsa ilmoituksen, mitä tulee vielä korjata.

6 Yhteenveto

Valmiin pelin rakennus ei ollut tässä työssä ensisijaisena tavoitteena. Tavoitteena oli kehittää Android-pelin luomista, ohjelmointia ja hankkia kokemusperäistä tietoa ja kattavampaa kuvaa pelinkehitysprosessista. Kehitystyön lopputuloksena syntyi toimiva prototyyppi yksinkertaisesta mobiilipelistä, jota voidaan helposti lähteä jatkokehittämään ja laajentamaan. Lisäksi, täysin uuden Unityllä tapahtuvan pelikehitysprojektin aloittaminen on jatkossa helpompaa odotusten, vaatimusten ja mahdollisuuksien selkeydyttyä tämän työn pohjalta.

Opinnäytetyöraportti muodostaa kattavan yleiskuvauksen Android-pelin kehitysprosessista, jota aloitteleva pelikehittäjä voi hyödyntää ensimmäisissä projekteissaan. On kuitenkin huomattava, että empiirinen kokemus osoittautui merkittäväksi eduksi peliä kehittäessä. Virheistä oppii ja Unityllä peliä kehittäessä nämä virheet avautuvat silmien eteen suoraan ja ilman viiveitä. Tämän opinnäytetyön lukemalla voi alasta kiinnostunut saada perusteet,

työkalut, sekä autenttisen kuvauksen edessä olevasta työurakasta, mutta todellinen oppiminen tapahtuu itse tekemällä.

Tämän raportin lisäksi Unitystä löytyy myös kattavasti tietoa ja tutoriaaleja Unity Technologiesin omilta sivuilta, jotka toimivatkin tämän projektin tietokantana. Näitä sivuja ei voi suositella liikaa. Jokainen projektin aikana kohdattu ongelma oli ratkaistavissa Unityn manuaaliin perehtymällä ja sen sisältämää tietoa soveltamalla tarvittavaan muotoon. Lisäksi, Unity on tällä hetkellä kuuma sana pelikehityspiireissä, joten saatavilla olevaa tietoa on saatavissa rajattomasti lukuisista lähteistä, keskustelupalstoista ja videoista.

Tämän työn keskeisimmäksi osioksi muodostui pelin julkaisu Play-kauppaan. Julkaisuvaiheen prosessi osoittautui ennakko-odotuksia monimutkaisemmaksi ja muodostikin valtaosan projektista saadusta oppimisesta. Tässä raportissa julkaisuprosessin vaatimukset ja työvaiheet käytiin läpi pääpiirteissään. Tavoitteena oli antaa alasta kiinnostuneille oikeaoppinen kuva tuosta prosessista, jotta mahdolliset ongelmatilanteet julkaisuun liittyen voidaan välttää.

Lähteet

Sähköiset

Unity Public Relations. https://unity3d.com/public-relations. Luettu Huhtikuu, 2019.

Unity Company. https://unity3d.com/company. Luettu Huhtikuu, 2019.

Unity Learn. https://unity.com/learn. Luettu Huhtikuu, 2019.

Unity User Manual. https://docs.unity3d.com/Manual/index.html. Luettu Huhtikuu, 2019.

Kuviot

Liitteet

Liite 1: Sanastoa

Android	Googlen omistama mobiilikäyttöjärjestelmä.
APK-tiedosto	Android Application Package, eli Androidille rakennettavan sovelluksen tiedostomuoto.

Asset	Unityn valitsema termi erilaisille pelissä käytettäville resursseille, kuten grafiikat, äänet ja peliobjektit.
Asset Store	Unityn ylläpitämä kauppapaikka lukuisille, julkaistuille resursseille, joita kehittäjät voivat jakaa, joko ilmaiseksi, tai maksua vastaan.
Build	Projektista paketoitu jakelutiedosto, androidilla muotoa .apk.
Collider	Unityn valitsema nimitys peliobjektien kosketukset rekisteröiville komponenteille.
Game Object	Kaikki pelissä käytettävät objektit.
Google Play Store	Googlen omistaman Android-käyttöjärjestelmän kauppapaikka.
Inspector	Unity Editorin osa, jossa määritetään ja lisätään valitulle peliobjektille asetuksia ja ominaisuuksia.
Loop	Pelin kulku, tila, jonka aikana peli suoriutuu.
Microsoft Visual Studio	Unityn mukana tuleva ohjelmointiympäristö.
Pelimoottori	Ohjelmistokehitysalusta videopelelille. Sisältävät nykyisin usein kaiken tarvittavan pelikehitykseen.
Prefab	Valmis peliobjektiprototyyppi uudelleenkäyttöä varten. Esimerkiksi vihollistyyppi.
Rigidbody2D	Unityssä käytettävä komponentti, joka vastaa peliobjektien fysiikkamallinnuksesta.
SDK	Software Development Kit. Ohjelmistojen kehitystarvikkeet yhdessä paketissa, tämän projektin aikana juurikin Androidin SDK.
Scene	Kohtaus, Unityn nimitys tiloille, joihin peliobjektit sijoitetaan. Esimerkiksi pelin päävalikko, sekä itse peli.

Script	Ohjelmakoodi, joka määrittelee peliin luotujen objektien toimintaa. Esimerkiksi pelaajan liikkumista.
Sprite	2D-mallinen kuva.
Unity Editor	Unityn käyttöliittymä.
Unity Remote	Unityn kehittämä mobiilisovellus pelin nopeaan testaamiseen kohdelaitteella.