The HammasAppi

Sovelluksen tuottaminen ja lisääminen sovelluskauppaan

HAMK
HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

Ammattikorkeakoulun opinnäytetyö

Tieto- ja viestintätekniikka, insinööri (AMK)

Kevät 2022

Jarno Vallo

HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

Tieto- ja viestintätekniikka
Tekijä Jarno Vallo
Työn nimi The HammasAppi
Ohjaaja Toni Laitinen

Tiivistelmä
Vuosi 2022

TIIVISTELMÄ

Työn tavoitteena oli tuottaa Suomen Hammaslääkäriliiton viimeisimpien suositusten (Suomen Hammaslääkäriliitto, 2008) mukainen hampaiden harjausajastin. Kohderyhmänä ovat alle kouluikäiset lapset. Julkaisukelpoinen mobiiliappi toteutettiin iOS alustalle ja julkaistiin Applen sovelluskaupassa. Sovellus arpoo ruutuun erilaisia kuvitteellisia suun bakteereita yksikätisen rosvon tyyliin. Kyseessä ei kuitenkaan ole peli, joten häviötä tai voittoa ei ole. Ajatuksena oli tuottaa appi siten, että se ei kerää mitään käyttäjäinformaatiota eikä vaadi internetyhteyttä käyttämiseen ja näin ollen se suojaa käyttäjänsä identiteettitietoja. Appi toteutettiin käyttäen Unity-ohjelmointiympäristöä, jolloin samalla C#-kiellellä tuotettu koodi saadaan halutessa käännettyä Unityllä myös muille alustoille.

Tarkoituksena on saada nuori hampaiden harjaaja kiinnostumaan harjauksesta odottaessaan saisiko hän kolme samanlaista bakteeria tuloksena kolmen minuutin hampaiden harjauksesta. Lisäksi on saatavilla lisätietoa suun bakteereista. Informaatiosivulla on kuva ja lyhyt teksti kuudesta ihmisen suussa viihtyvästä bakteerista. Kuvat ovat fiktiivisiä piirroskuvia, mutta tekstit pohjautuvat faktaan ja ne on muokattu sovelluksen kohderyhmän ikätasolle sopiviksi.

Kehitystyön tuloksena valmistui harjausajastinsovellus, joka ohjelmoitiin Unityllä ja c#:lla sekä lähetettiin Applen App Store -sovelluskauppaan. Työn aikana havaittiin, että Unity soveltuu hyvin myös tavallisten sovellusten tekemiseen, ei vain pelien. Toinen havaittu asia oli, että ohjelman valmistelu sovelluskauppaa varten sekä itse sovelluskauppaan toimittaminen ei ole kovin käyttäjäystävällinen prosessi ja vaatii paljon tietoa eri vaiheista.

Työ toteutettiin RexiCode Oü:n toimeksiantona. Yritys toimitti sovellukseen kuvia ja äänet.

Avainsanat Ohjelmistotuotanto, ohjelmointiympäristöt, sovelluskauppa
Sivut 39 sivua ja liitteitä 4 sivua

HAMK
HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

Information and Communication Technology
Author Jarno Vallo
Subject The HammasAppi
Supervisors Toni Laitinen

Abstract
Year 2022

ABSTRACT

The goal of this thesis was to produce a dental timer app for mobile iOS platform that follows the latest recommendations of Finnish dental association (Suomen Hammaslääkäriliitto, 2008). The target group is children under school age. The publishable mobile app is implemented for iOS platform and published in Apples application store. The application raffles different fictional bacterial images in a slot machine style. Yet this application is not a game, so there is no win or lose. One of the main thoughts is to produce the application so that it does not collect any user information and does not need internet connection for using the application, so it also protects user identity. The application is implemented using Unity -programming environment which allows same code produced using c# language to be transferred to other platforms.

The goal is to get children interested in brushing their teeth while they wait to see if they get three different kinds of bacteria as a result of brushing their teeth for three minutes. There will also be additional information available about mouth bacteria. On the information page there is pictures and short text about bacteria that like to live in human mouth. Pictures are fictional images of bacteria, but short information is fact and they have been modified suitable for target groups age level.

As a result of this development work a brushing timer app was produced, planned, and programmed using Unity and C# and submitted to Apple application store. During this process it was found that Unity is suitable to produce also ordinary applications and not just games. Another matter I found out is that preparation for application store and submitting it to application store is not a very user-friendly process and it requires lot of knowledge about different steps.

The work was carried out as an assignment for RexiCode Oü. Graphics and sounds were supplied by them.

Keywords Softwaredevelopment, programmin platform, Application Store
Pages 39 pages and appendices 4 pages

Sisällys

Kuvat, taulukot ja kaavat

Käsitteistö

[SerializeField]	Private muuttujan näyttäminen tulkille
Android	Googlen mobiilikäyttöjärjestelmä
AppStore	iOS alustan sovelluskauppa
C#	Ohjelmointikieli
child -elementti	Hierarkisesti toisen elementin alla oleva elementti
CSR	Certificate Signing Request eli sertifikaatin allekirjoituspyyntö -tiedostotyyppi
Float	Liukulukumuuttuja
Hammasbakteeri	Fiktiivinen hammasbakteeria esittävä olio
HTML5	HTML -merkintäkielen versio 5
iOS	Applen mobiilikäyttöjärjestelmä
JPG	Kuvanpakkausformaatti
Metodi	Ohjelmointikielessä funktio
Monialustainen	Samalla ohjelmointiympäristöllä voidaan tehdä sovelluksia useille eri alustoille
Play-kauppa	Android alustan sovelluskauppa
PNG	Kuvanpakkausformaatti
Prefab	Valmispohja/malli elementille Unityssä
RC	Release canditate, julkaisuehdokas
Scene	Unityn itsenäinen ohjelmisto-osio
String	Merkkijonomuuttuja
Unity	Pelinsuunnittelu kehitysympäristö / pelimoottori
Visual Studio	Microsoftin tuottama ohjelmakehitysympäristö
Wav	Audiotiedostoformaatti
Yksikätinen rosvo	Slot machine -tyyppinen peliautomaatti
Bugi	Ohjelmointivirhe

Liitteet

Liite 1 Unityn tukemat laitealustat

Liite 2 Tietosuojalausekkeen lähdekoodi

Liite 3 KayttoScripti.cs ja loadinScr.cs lähdekoodi

Liite 4 Käytetyt graafiset kuvat

1 Johdanto

Opinnäytetyön tarkoituksena on tuottaa iOS-mobiilialustalle mobiilisovellus, jonka avulla pystyttäisiin pitämään yllä alle kouluikäisen lapsen (2-6v.) kiinnostusta hampaiden harjaukseen Suomen Hammaslääkäriliiton suosituksen mukaisesti (Suomen Hammaslääkäriliitto, 2008) kolme minuuttia per harjauskerta. Ajastinsovelluksen tarkoituksena on esittää visuaalinen arvonta. Arvonnassa kolmen saman kuvan suora saa esiin tähden. Tähti on kuva, joka esitetään ruudulla. Mikäli kolmen suoraa ei saavuteta, esitetään ruudulla Hienoa, hampaat pesty huolella! -kuva. Muita toiminteita ovat; äänet on/off, laskurin taukotoiminne ja pysäyttäminen sekä bakteerikuvaa painamalla avautuva lisätietosivu kuudesta suun bakteerista. Sovelluksessa käytetyt bakteeripiirrokset ovat fiktiivisiä, mutta lisätietosivun bakteeritietous on faktaa.

Valmistuva sovellus laitetaan ilmaiseksi kaikkien saataville Applen App Store -sovelluskauppaan. Sovelluksen sovelluskauppaan hyväksytysti toimittaminen on kokonaan oma osa-alueensa. Työssä käydään läpi kaikki tarvittavat vaiheet sovelluksen kauppaan hyväksytysti julkaistuksi saattamiseksi. Tähän sisältyvät muun muassa julkaisua varten tarvittavat sertifikaatit, julkaisuprofiilit, kuvaustekstit, arkistopaketit, kuvat ja muut sovelluksen julkaisemisen vaatimat asetukset niin Unityn kuin Xcoden puolelta sekä Apple App Store Connectin asetukset. Sovelluksen julkaisemista helpottaa ettei ohjelman haluta keräävän tietoa käyttäjistään tai ohjelman käytöstä ja sen on tarkoitus toimia latauksen jälkeen ilman internetyhteyttäkin.

1.1 Kilpailevat sovellukset

Applen sovelluskaupassa hakusanoilla ”hampaiden harjaus” tai ”harjausajastin” ei löydy yhtään kilpailevaa sovellusta. Oral B hakusanalla löytyy Disney Magic Timer by Oral-B sekä Oral-B Fun Zone sovellukset, jotka molemmat ovat englanninkielisiä ja kytkeytyvät vahvasti Oral-B hammasharjabrändiin ja hammasharjan rajapinnan kautta itse hammasharjaan. Englanninkielellä haettaessa hakusanalla ”teeth brush timer” löytyy useampia englanninkielisiä sovelluksia. The HammasAppi -sovellus on kuitenkin kohdennettu

Suomeen, joten periaatteessa se on ensimmäinen suomenkielinen kolmen minuutin harjausajastin.

1.2 Toimeksiantaja RexiCode Oü

RexiCode Oü, yrityksen logo kuvassa 1, on vuonna 2013 Viroon perustettu pieni ohjelmointialan yritys, joka on erikoistunut räätälöityjen ohjelmointiratkaisujen toteuttajaksi. Yritys tuottaa asiakkaan tarpeisiin suoraan sopivia räätälöityjä sovelluksia tarvittaville alustoille sopivilla ohjelmistokielillä. Yrityksen oma tuote on Postilukija-sovellus (RexiCode, 2016), joka on lanseerattu vuonna 2015. Postilukija-sovellus lukee yritykselle tulleita tilausposteja ja tuottaa tilaukset automaattisesti Erp-järjestelmään vähentäen näin asiakaspalvelun/tilauspalvelun manuaalista työtä tältä osin. RexiCode Oü tuottaa myös yritysten www-sivustoja tilauksesta.

Kuva 1. RexiCode logo (RexiCode Oü Graafinen ohjeisto, 2022)

2 Työkalut eli käytettävät ohjelmat

Tämän sovelluksen tuottamiseen käytetään Unity 5 2021.3.1f1 (Unity Technologies, n.d. -a) versiota sekä C# koodin tuottamiseen Microsoft Visual Studio for Mac 8.10.22 versiota. Kuvat on käsitelty Adobe Photoshop 2022 sovelluksella. XCode versio on 13.3.1. Ohjelmistot toki saavat jatkuvasti päivityksiä ja versiot voivat muuttua työn aikana.

2.1 Unity

Unity (Unity Technologies, n.d. -c) on monialustainen pelimoottori. Unityllä voidaan tuottaa sekä 2D- että 3D-pohjaisia pelejä. Peliohjelmointialustan termistä huolimatta Unity soveltuu hyvin myös tavallisten sovellusten tuottamiseen ja nimenomainen hyöty on sovellusten kääntämisen mahdollisuudesta useille eri alustoille. Tässä opinnäytetyössä testaan valmistuvaa sovellusta iOS:lla. Kuitenkin myöhemmin, työni ulkopuolella, olen suunnitellut kääntäväni sovelluksen myös Androidille. Käytössäni on maksullinen Unity Plus.

Aikaisemmin Unityllä saattoi käyttää Boo, Javascript sekä C# kieliä. Boo:n ja Javascriptin tuki on kuitenkin käytännössä päättynyt, joten C# on ainoa käytettävissä oleva kieli. Tämän vuoksi ohjelmointityö toteutetaan c#:lla.

Unity tukee useita alustoja, joille sovelluksen tai pelin voi kääntää. Yleisimmät kuten Windows, MacOS, iOS ja Android ovat ilmaisia, mutta osaan vaaditaan rekisteröityminen toisen alustan kehittäjäksi kuten Xbox ja toisille alustoille voidaan vaatia oma lisenssi alustan valmistajalta. Joidenkin alustojen lisenssit voivat maksaa ja olla hyvinkin kalliita, varsinkin julkaisua odottavien pelikonsolien lisenssit.

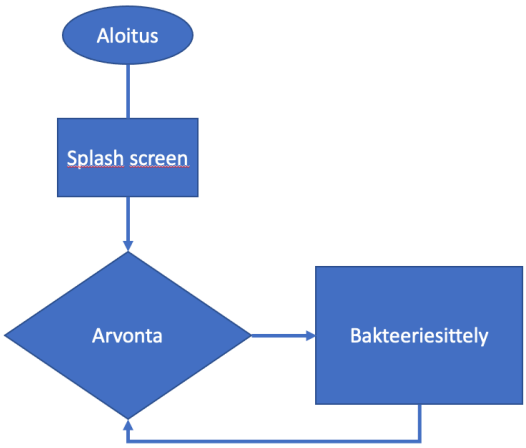
2.2 Microsoft Visual Studio Community

Microsoft Visual Studio Community on Microsoftin julkaisema ohjelmakehitysympäristö. Tässä projektissa käytettävä versio toimii saumattomasti Unityn kanssa yhteen. Kun Unityssä klikkaa .cs-scriptitiedostoa aukeaa valittu tiedosto Visual Studioon automaattisesti. Visual studiossa käytämme C# kieltä Unityn kanssa, mutta Visual Studio itsessään tukee muitakin kieliä kuten C++ ja F#. Community versio on kevyt ja maksuton. Käytän Visual Studio 2019 for Mac Community -versiota, koska Visual Studio 2022 for Mac on edelleen elinkaarensa RC versiossa macille ja mielestäni sen käyttäminen tuotantoon on toistaiseksi ennenaikaista.

3 Ohjelman rakenne

Mobiilisovellukset koostuvat Unityssä erillisistä tapahtumaosista (scene). Ohjelma voi omata lukuisan määrän scenejä ja niistä näytetään yleensä vain yhtä kerrallaan. Lisäksi Unity tukee niin iOS kuin Android latausanimaatioita, mutta tämän voi korvata yhdellä Unityn scenellä joka voi sisältää mitä tahansa materiaalia. Suunnittelin hammasappiin (nimeltään The HammasAppi) kolme rakenteellista sceneä. Nämä scenet ovat latausruutu, nimeltään latausRuutu, itse ajastinsivu nimeltään ajastinSivu ja suun bakteerien esittelysivu nimeltään popoSivu. Ohjelmassa on vain muutama scene ja tämän vuoksi toimintakaavio, joka on kuvattu kuvassa 2, on hyvin yksinkertainen.

Kuva 2. Ohjelman toimintakaavio



3.1 LatausRuutu

Sovelluksesta tehdään kolmen scenen appi. Sovellusta käynnistettäessä ladataan ja näytetään ensin latausRuutu, johon sijoitetaan ohjelmaa kuvaileva taustakuva sekä RexiCode-logo. Tätä sivua näytetään kolmen sekunnin ajan. LatausRuutu ei sisällä toiminteita, eikä sen näyttämistä voi nopeuttaa napauttamalla ruutua. Kun kolme sekuntia on kulunut latausRuutu poistetaan muistista ja tilalle ladataan ajastinSivu. Näillä Splash screeneiksi kutsutuilla latausruuduilla on oikeasti olemassa myös tehtävänsä. Latausruutujen

aikana usein ladataan seuraava scene käyttövalmiiksi, jolloin väistytään pelkän ladataan-palkin näyttämiseltä. Valmistuva sovellus on kuitenkin niin pieni, ettei scenen ennakkolataamiseen ole tässä työssä tarvetta.

3.2 AjastinSivu

Ajastinsivulla on visuaalisesti neljä pääelementtiä, joilla kaikilla on oma tarkoituksensa. Elementit asetellaan Käyttöliittymässä käyttäjäystävälliseen tapaan. Unityn koodissa elementit ovat canvas tyyppiä, joille annetaan sijaintia määrittävät asetukset elementin Rect Transform -osiossa.

3.2.1 Äänivalinta

Ensimmäisenä oikeassa yläreunassa on äänet päällä -logo. Tästä logosta napauttamalla logo muuttuu äänet poissa -logoksi ja ohjelmaan valittu ääni häviää. Ohjelma ei soita taustamusiikkia. Kun ajastin käynnistetään arvottavat hammasbakteerit alkavat pitää yksikätinen rosvo -pelin tyyppistä ääntä. Äänet on mahdollista laittaa päälle ja pois kuvan 3 kuvakkeista.

Kuva 3. Ääninappi-kuvakkeet on/off

3.2.2 Bingoalue

Tapahtuma-alue, jossa arvonta tapahtuu, on nimeltään bingoalue. Kun ohjelma käynnistetään on tällä alueella aloituskuva, kuva 4, joka viestii ohjelman tarkoitusta.

Aloituskuvassa lukee: Avuksi lasten hampaiden hoitoon. Kun ohjelma käynnistetään, eli aloitetaan hampaiden harjaus, tämä alue vaihtuu itse arvonnaksi. Aloituskuva piilotetaan ja tilalle tulee arvontaelementti, kuva 5, jossa yksikätisen rosvon tapaan bakteerien kuvat pyörivät ympyrää. Minuutin välein yksi kuvista pysähtyy, kunnes kolme minuuttia on kulunut ja arvonta on suoritettu loppuun. Lopuksi esitetään jompikumpi kuvan 6 tuloskuvista. Vaihtoehtoisia tuloskuvia on kaksi; tähti, jonka saa kolmesta samanlaisesta bakteerista tai ympyräkuva, jossa kannustetaan huolelliseen hampaiden hoitoon.

Kuva 4 Aloituskuva

Kuva 5 Arvontaelementti

Kuva 6. Kolmen samanlaisen bakteerin tähti tai huolellisen pesun merkki

3.2.3 Ajastinkenttä

Ajastintekstikenttä, joka on esitetty kuvassa 7, on Unityssä nimeltään laskuriAlue. Se on muotoiltu teemaan sopivin värein ja fonttina on Peachykeen-fontti. Kun ajastin käynnistetään tätä kenttää päivitetään Unityn ajastimella sekunnin välein. Ajan loppuessa tässä kentässä lukee 0:00. Kenttä myös nollautuu takaisin 3:00 kenttään painamalla lopeta- tai aloita-nappeja. Tämä kenttä kertoo paljonko harjausaikaa on jäljellä.

Kuva 7. Ajastinkenttä

3.2.4 Käyttönapit

KayttoNapit alue, kuvassa 8, asemoituu ajastinSivun alareunaan ja sisältää kolme nappia; aloita-nappi aloittaa laskurin ja arvonnan, lopeta-nappi lopettaa arvonnan ja nollaa laskurin sekä tauko-nappi, jos hampaiden harjauksen joutuu keskeyttämään. Tauko-napin teksti vaihtaa väriä tauolla. Harjauksen voi myös lopettaa kesken tauon. Aloituksessa muut napit ovat harmaita ja kun laskennan käynnistää muuttuvat lopeta- ja tauko-napit värillisiksi. Värien vaihtuminen kertoo käyttäjälle nappien olevan valmiita käytettäviksi. Laskurin käynnistyttyä aloita-nappi muuttuu harmaaksi, sillä kesken olevaa laskentaa ei voi aloittaa uudelleen.

Kuva 8. Ajastimen hallintanapit aloituksessa

3.3 PopoSivu eli hammasbakteerien esittelysivu

Tällä sivulla voi sovelluksen käyttäjä tutustua tarkemmin muutamaan suun bakteeriin. Hammasbakteerien kuvia voidaan katsella isompina ja lapsi voi lukea itse tai aikuisen avustuksella pienen asiatekstin jokaisesta hammasbakteerista. Tälle sivulle pääsee painamalla vasemman yläreunan bakteerikuvaa. Bakteerikuva-nappi ei ole käytettävissä, kun arvonta on käynnissä.

Kuvassa 9 ovat hammasbakteerit, joita sovelluksessa käytetään. Kaikki hammasbakteerit ovat fiktiivisiä ulkomuodoltaan, mutta niiden nimet ja kuvaukset ovat todellisuutta vastaavia.

Kuva 9. Bakteerit

Kariesbakteeri

Mutans-Streptokokkibakteeri

Parodontiitti

Suusieni

Laktobasilli

Veillonella

4 Tietosuoja

Tietosuojasivu on tärkeä osa kaikkia ohjelmia. Ohjelmiston käyttäjän tulee tietää, mihin ja miten ohjelmisto käyttää keräämäänsä dataa. Tämä ohjelma ei kuitenkaan kerää dataa hampaiden harjauksen kestoa enempää ja tämäkin data nollaantuu kun pesun lopettaa. Dataa ei siis lähetetä mihinkään, ei käsitellä mitenkään, eikä sitä edes tallenneta käyttäjän laitteelle. Koska dataa ei kerätä ja käsitellä helpottuu apin julkaisu App Storen sääntöjen mukaan sekä myöskin GDPR:n noudattamisesta tulee helppoa. Yleisimmät sovelluskaupat kuten Applen AppStore sekä Googlen Play -kauppa vaativat kuitenkin netissä olevan tietoturvasivun (tietosuojalausekkeen) englanniksi nähtäville sovelluksen kotisivuilla ja linkki tälle sivulle tulee asettaa sovelluskauppaan appia julkaistaessa.

4.1 GDPR

EU:n yleinen tietosuoja-asetus, 2016/679 (engl. General Data Protection Regulation, GDPR) on Euroopan parlamentin, Euroopan unionin neuvoston ja Euroopan komission yhteinen pyrkimys yhtenäistää tietosuojaa koskeva lainsäädäntö kaikkien Euroopan unionin jäsenmaiden kesken. EU:n sisäisen säätelyn lisäksi se koskee myös tahoja, jotka tallentavat EU:ssa asuvien henkilöiden henkilötietoja Euroopan unionin ulkopuolelle. Yleisen tietosuoja-asetuksen tarkoituksena on ensisijaisesti vahvistaa EU:ssa asuvien henkilöiden oikeuksia omiin henkilötietoihinsa sekä yksinkertaistaa sääntely-ympäristöä niin, että sekä EU:n sisäinen että kansainvälinen liiketoiminta helpottuu (Council of the European Union, 2015).

Ohjelmistokehityksessä tämä asetus, jonka Suomikin on ratifioinut, asettaa rajoituksia sille mitä dataa saa kerätä ja säilyttää sekä missä dataa saa säilyttää. Tärkeimmät huomioitavat asiat ovat, ettei mitään turhaa yksilön tunnistavaa dataa saa kerätä ja data tulee säilyttää EU-alueella. Myöskin tämä tuo mahdollisuuden datan keräyksen kohteelle vaatia ja saada nähtäväksi sekä poistettavaksi häntä koskevan datan. Tämä toki on Suomen lainsäädännön mukaan ollut mahdollista jo aiemminkin, mutta yleisesti säännöstö on silti kiristynyt.

GDPR:n noudattamista Suomessa valvoo tietosuojavaltuutetun toimisto sekä Euroopan unionin tietosuojavaltuutettu. Nämä tahot valvovat, että sääntöjä noudatetaan ja henkilötietoja käsitellään asianmukaisesti. Euroopan tietosuojavaltuutettu suorittaa myös tutkimuksia ja tietosuojavaltuutetun toimisto ottaa vastaan kanteluita GDPR:ää loukkaavista menettelyistä. GDPR toi lisäksi tiedon kerääjille velvollisuuden ilmoittaa tietoturvaloukkauksissa 72 tunnin kuluessa valvontaviranomaiselle.

4.2 Tietosuojalauseke

Koska tämä ohjelmisto ei kerää dataa on tietosuojalauseke lyhyt. Kokeilin aikaisemmin julkaisemieni appien kanssa pelkästään kohdilla 1 ja 4, mutta muutkin kohdat ovat pakollisia sovelluskauppojen mukaan, vaikka rekisteriä ei edes ole käytännössä olemassa. Tietosuojalausekesivu on tehty html5-kielellä. Tietosuojalausekesivun lähdekoodi on liitteessä 2 ja itse sivu on julkaistu rexicode.fi-palvelimella (RexiCode, 2022).

1. The controller
Name:
Y-Business id:
Address:
Zip code:
Post office:
Phone number:
Email address:

2. Person in charge of registry matters
Company:
Name:
Phone number:
Email address:

3. Data protection Officer
Yritys:
Nimi:
Puhelinnumero:
Sähköpostiosoite:

4. Purpose of the register
No data is collected

5 Unity Asset Store

Unity Asset Store on sovelluskauppa Unitylle. Tarjolla ei ole valmiita ohjelmia vaan valmiita moduleita noin 60 000 kappaletta, joita voi käyttää omassa sovelluksessa. Hinnat alkavat 0.00€:sta. Koska kaikkea ei tarvitse tehdä itse uudelleen, käytän projektissani valmista modulia arvonnassa. Tämän modulin nimi on Simple Scroll Snap ja sen on kirjoittanut Daniel Lochner. Moduli on ilmainen. Simple Scroll Snap:ia kuvaillaan Unity Asset Storessa seuraavasti.

” "Simple Scroll-Snap" is an elegantly designed, intuitive solution that allows for elements within a ScrollView to be snapped to, offering a wide range of customization options.” (Lochner, 2022)

Teksti vapaasti suomennettuna. ” "Yksinkertainen vieritys-napsahdus" on elegantisti muotoiltu, intuitiivinen ratkaisu, joka sallii rullausnäkymässä olevien elementtien kiinnittyä, tarjoten laajan valikoiman muokkausoptioita.”

6 Koodi

Unity käyttää C# kieltä, joka voi ohjata eri elementtejä sekä suorittaa kaikkea mahdollista mitä ohjelmointikoodin halutaan suorittavan. Kirjoitan useita metodeita, jotka ohjaavat toimintaa eri sceneissä. Metodeista kerron lisää kohdassa 6.3.

6.1 LatausRuutu

LatausRuudulle tein oman C# scripti tiedoston nimeltään loadinScr.cs. Käytän usein projekteissani sekaisin suomen- ja englanninkieltä, sillä se luo persoonallisen tavan käsitellä tietoa sekä koodata, mutta samalla tekee muille koodien toiminnan selvittämisen vaikeammaksi. Tämän vuoksi käytän tapaani lähinnä vain omissa projekteissani, joita käsittelen itsenäisesti. Käytän tässä public tyyppisen muuttujan sijaan [SerializeField] komentoa muuttujalle, joka tekee siitä tulkille näkyvän joka puolella koodia.

LatausRuutu scenen halutaan näkyvän kolme sekuntia, joten aluksi tehdään float tyyppinen muuttuja nimeltä delayBeforeLoading, jolle annetaan arvoksi haluttu aika eli 3f. F tarkoittaa Unityssä ”yksi aste sekunnissa” joka kerrotaan Delta-ajalla. 1f on näin ollen yksi sekunti ja 3f on kolme sekuntia.

Seuraavaksi halutaan kertoa ohjelmalle mikä scene ladataan kun kolme sekuntia on kulunut ja sitä varten tehdään merkkijonotyyppinen muuttuja, johon kirjoitetaan ladattavan scenen nimi Unityn puolella scriptin tiedoissa inspectorissa. Lisäksi tarvitaan float tyyppinen muuttuja, johon tallennetaan jo odotettu aika.

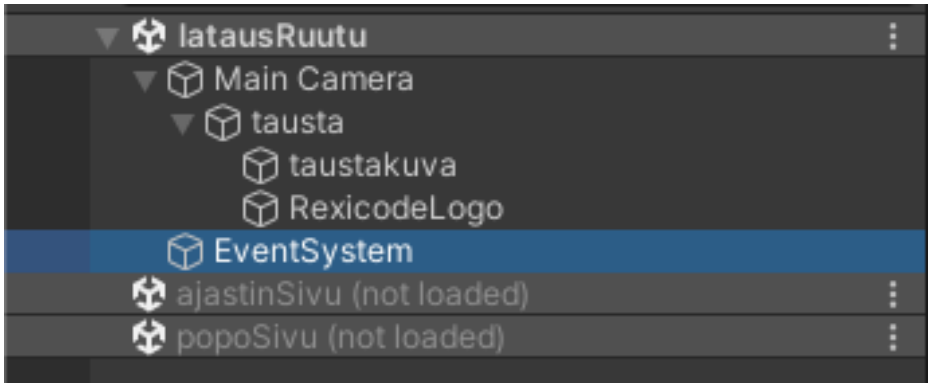
Unityn omassa update()-metodissa lisätään timeElapsed-muuttujaan Time.deltaTime-komennolla järjestelmästä saatu aika. Time.deltaTime kertoo kuluneen ajan edellisen kehyksen suorittamisen ja nykyisen kehyksen välillä.

Tämän jälkeen verrataan timeElapsed ja delayBeforeLoading aikoja keskenään. Mikäli kulunut aika on isompi kuin 3f, suoritetaan SceneManagerilla uuden scenen lataus. Uudemmissa Unity versioissa Unity poistaa itse muistista muut scenet kuin juuri ladatun, eli ne joita ei käytetä, joten niistä ei tarvitse huolehtia enempää. Joissakin sovelluksissa voi olla tarpeen käyttää useampaa sceneä yhtä aikaa, vaikka vain yhtä näytettäisiinkin ja tämänkin toteuttamiseen löytyy keinot tarvittaessa.

Vielä pitää lisätä alkuun using.UnityEngine.SceneManagement; rivi, joka kertoo tulkille että tämä koodi ladattaessa otetaan käyttöön Unityn Scene management moduli, joka osaa ladata halutun scenen käyttöön. Koko loadinScr.cs koodi on nähtävillä liitteessä 3.

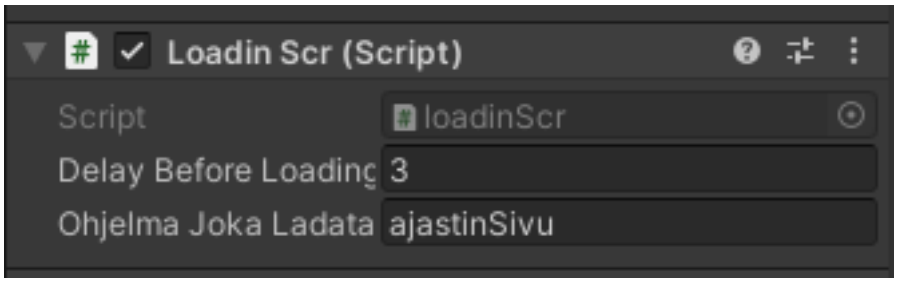
Nyt koodi lataa kolmen sekunnin päästä aloitusruudun jälkeen itse pääscenen eli ajastinSivun. Koodi pitää vielä kiinnittää elementtiin. LatausRuutu scenen rakenne selviää kuvasta 10. Kuvassa on canvas nimeltään tausta, johon on lisätty image elementit taustakuva ja RexicodeLogo. Muut scenet eivät ole ladattuna.

Kuva 10. LatausRuutu-scenen rakenne



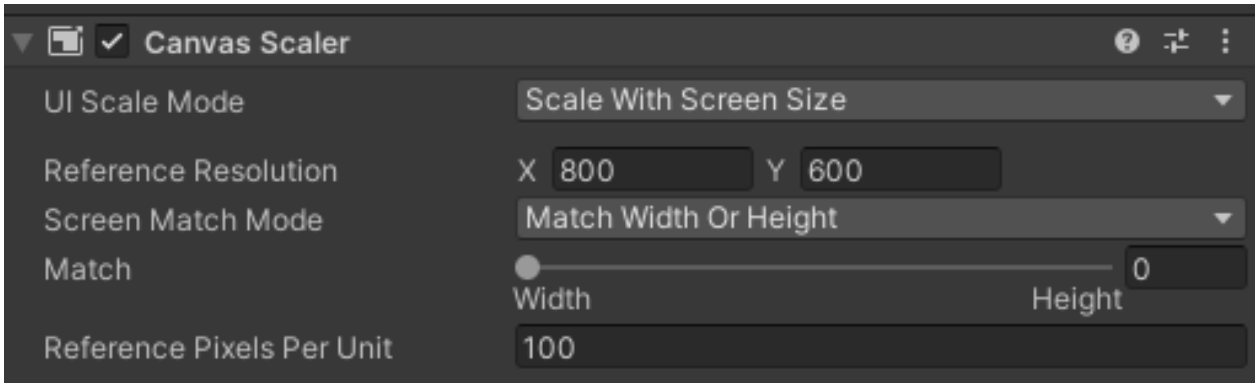
Taustaelementille lisätään script komponentti, johon loadinScr scripti vedetään drag & drop tekniikalla Asset kansiosta. Scriptille voi määrittää tässä kohdassa kauanko tätä sceneä näytetään, jonka jälkeen ladataan ajastinSivu. Myös ladattavan scenen nimen voi laittaa Ohjelma joka ladataan -kenttään kuten kuvassa 11. Mikäli seuraavaan versioon haluttaisiinkin ohjelman käynnistyvän suoraan bakteerisivulle, tähän kirjoitettaisiin popoSivu ja käännettäisiin ohjelma uudelleen.

Kuva 11. LoadinScr scripti tausta elementin Script komponentissa



Taustaelementille määritellään tausta canvaksen, canvas scaler osiossa ja UI scale modeksi ”Scale with Screen size”, kuva 12. Tämä saa taustakuvan skaalautumaan automaattisesti ruudun koon mukaan. Käytän tätä samaa asetusta kaikkien tausta canvasten kanssa jokaisessa scenessä.

Kuva 12. Tausta canvaksen canvas scaler elementin asetukset



6.2 AjastinSivu

AjastinSivu on koko ohjelman tärkein ruutu. Siinä kiteytyvät kaikki toiminnot samassa scenessä. Siinä ovat ajastin, ääninappi, käyttönapit, nappi bakteerisivulle sekä arvontarullat. Tähän sceneen lisään kaksi canvas elementtiä. Toisen nimeksi asetan pyorijat ja toisen Tausta. Pyorijat sisältää vain Simple Scroll Snapin arvontarullat. Tausta sisältää taustakuvan sekä muut elementit scenessä. Kummatkin canvakset asetetaan canvas scaler -komponenteissa ”Scale with Screen size” tilaan, jotta nämäkin säätyisivät automaattisesti ruudun koon muuttuessa eri laitteilla.

6.2.1 Tausta canvas

Tausta canvakseen lisätään ensin Taustakuva-niminen imagemoduli, jolle asetetaan taustakuvaksi hammastausta.png-niminen taustakuvatiedosto, joka on kuvassa 13.

Kuva 13. hammastausta.png-taustakuva

Tausta canvakseen lisätään seuraavat canvakset child elementeiksi; ylaNapit, kayttoNapit, laskuriAlue ja bingoAlue. Näillä saadaan jaoteltua erilaiset toiminteet omiin alueisiinsa ajastinsivulla ja niiden työstäminen toimintaan on selkeämpää. Samalla saadaan asetettua nämä elementit pitämään paikkansa suhteessa ruudun reunaan tai keskikohtaan, kun skaalausta tehdään. Tällä saavutetaan se, että käytettäessä sovellusta suhteessa erikokoisilla näytöillä ohjelman perusvisuaalinen ulkoasu säilyy hyvinkin samannäköisenä. Mikäli käyttäisiin kiinteitä mittoja jotkin elementit voisivat päätyä kokonaan tai osittain ruudun ulkopuolelle. Kiinteiden mittojenkin käytöllä on kuitenkin paikkansa ja niitäkin voidaan asettaa toimimaan erilaisilla ruuduilla siten että ne näkyvät kokonaan, vaikka ruudun koko vaihtuisikin. Silloin useimmiten lasketaan elementin paikka ruudulla ruudun koosta ja näin

usein joudutaan tekemään myöskin lisättäessä dynaamisesti elementtejä ruudulle.

Tausta canvakseen lisätään myös tarvittaville äänille omat Audio Source -tyyppiset modulit. Yhteensä neljä kappaletta, nimiltään plot1-3 ja clap. Näihin jokaiseen määritetään äänitiedosto. Äänet ovat mp3-tyyppisiä ja nimeltään plot.mp3 sekä clap.mp3. Kun arvontaa ajetaan jokainen rulla soittaa omaan nopeuteensa täsmätyllä sekvenssillä plot-ääntä. Unity osaa soittaa vain yhden äänilähteen kerrallaan, joten kolmelle rullalle tarvitaan kolme äänilähdettä. Tällä myöskin saadaan aikaiseksi autenttisemman kuuloinen ääni, kun äänet muuttuvat rullien nopeuden mukaan ja vähenevät kun rullat pysähtyvät. Kuvassa 14 näkyy Tausta canvaksen lopullinen rakenne.

Kuva 14. Tausta canvaksen rakenne

6.2.2 AaniNappi canvas

AaniNappi canvaksen kooksi määritin 180px * 180px ja se sijoitetaan scenen oikeaan yläreunaan. Vaikka elementit asetellaan ruudulle dynaamisesti ruudun koon mukaan säätyvästi, nappien koot asetaan kiinteiksi. Kuitenkin niiden parent element, tässä tapauksessa Tausta canvas, säätyy ruudun koon mukaan. Rect Transform määrityksessä valinta laitetaan top right -kohtaan, jotta tulkki tietää sijoittaa aaniNappi canvaksen aina vastaavaan paikkaan näytöllä. AaniNappi:n alle sijoitetaan child elementiksi Button elementti, jonka teksti poistetaan ja lisätään Button elementille Image komponentti. Tähän

image komponenttiin laitetaan kuvan 3 vasemmanpuoleinen kuvake indikoimaan sitä, että äänet ovat päällä. Tällä rakenteella saadaan kuva toimimaan nappina.

6.2.3 KayttoNapit canvas

KayttoNapit canvas asetetaan Rect Transform komponentissa bottom-stretch asettelulle, joka venyttää alueen aina ruudun leveyden mukaan suhteessa asetettuun leveyteen. KayttoNapit canvaksen sisään laitetaan kolme imageobjektia; aloita, lopeta ja tauko.

Aloita imagelle asetetaan Rect Transform komponentissa bottom-left asetus, joka pitää sen ruudun muuttuessa vasemmassa alanurkassa. Kuvaksi valitaan kuvassa 8 vasemman reunimmaisena oleva aloita-nappi kuvake, joka on vihreänä. Tämä kertoo käyttäjälle, että napista lähtee käyntiin laskuri. Muut napit tässä canvaksessa tulevat olemaan harmaita aloituksessa. Kun laskenta käynnistetään napin sprite kuva vaihdetaan harmaaksi ja napin interactable asetus laitetaan koodista arvoon false. Aloita-napin button elementin onClick() määritykseen lisätään Tausta canvas, joka tuo kayttoScriptin public metodit valittaviksi. Valitaan OnSpin()-metodi, joka käynnistää ajastimen ja rullat. OnSpin() metodin toiminta on selvitetty tarkemmin työn kohdassa 6.3.3.

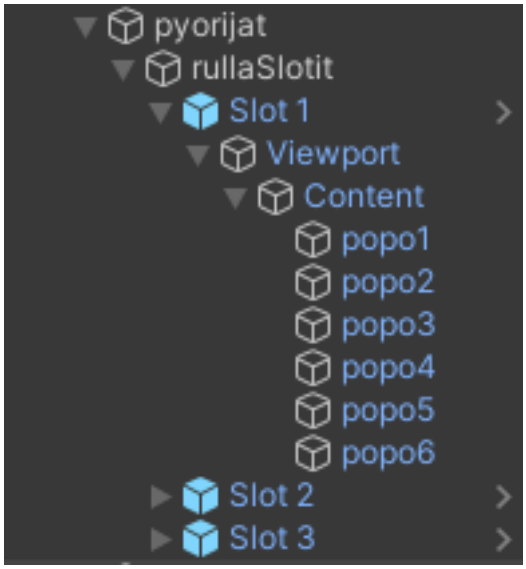
Lopeta imagelle asetetaan center-middle -asetus, joka keskittää kuvan kayttoNapit alueen keskelle. Kuvaksi laitetaan kuvassa 8 keskellä oleva harmaa lopeta-nappi kuvake. Lopeta-nappi kuvakkeelle annetaan OnClick() määritykseen OnStop()-metodi, joka pysäyttää laskurin ja resetoi kaiken uutta aloitusta varten. Tämän metodin tarkempi toimintatapa on selvitetty kohdassa 6.3.4.

Lopuksi laitetaan vielä kolmas image paikoilleen ja Rect Transformissa bottom-right -sijainniksi. Kuvaksi valitaan kuvan 8 oikean reunimmainen kuvake, joka on harmaa tauko -nappi. Tämän napin osalta kaikki toimii samoin kuin aiempienkin, mutta käytetään OnPause()-metodia. Metodin toimintapa on selvitetty kohdassa 6.3.5.

6.2.4 Pyorijat canvas

Pyorijat canvaksen, kuvassa 15, alle lisätään toinen canvas ja sille annetaan nimeksi rullaSlotit. Se on kuitenkin vain tavallinen canvas, jolle asetetaan sijainti Rect Transformissa. Tämän alle laitetaan kolme valmista prefab objektia ja niiden nimet ovat valmiina Slot1, Slot2 ja Slot3. Jokaisella Slot prefabilla on komponenttina Simple Scroll-Snap komponentti. Slot objektilla on valmiiksi child objektina viewPort-Unity objekti ja sillä taas child objektina content-unity objekti. Content sisältää kaikki ne objektit, joita kyseisessä slotissa arvotaan. Tässä tapauksessa laitetaan sinne kuusi image objektia nimettynä popo1-popo6, joista jokaisessa on yhden tietyn bakteerin kuva. Sama toistetaan Slot2 ja Slot3 komponenteille.

Kuva 15. Pyorijat canvaksen rakenne



Itse arvonnan saa käyntiin suorittamalla OnSpin() metodin. Tätä voidaan kutsua vaikka napilla, mutta ensin on laitettava muutakin koodia ylimmän tason canvakselle, jotta scripti saadaan käyttöön. Lisätään siis tausta canvakselle kayttoScripti.cs tiedosto. Tämä lisättiin tausta canvaksen scripti komponentiksi. Tässä vaiheessa ajetaan pyöriviä slotteja käyttäen Simple Scroll Snapin mukana tullutta SlotMachine.cs esimerkkikoodia, mutta myöhemmin koodi siivotaan julkaisua varten ja poistetaan ylimääräinen nappi sekä siirretään metodit

SlotMachine.cs:stä kayttoScripti.cs:ään. Näin SlotMachine.cs jää kokonaan pois käytöstä lopullisesta versiosta.

Tekemisen aikana lisättiin tausta canvakselle ylimääräinen nappi, jolla ajettiin SlotMachine.cs-scriptiä. SlotMachine.cs on Simple Scroll-Snapin mukana tuleva tiedosto. Myöhemmin nappi poistettiin, kun päästiin siihen vaiheeseen että kaikki käyttönapit olivat paikoillaan ja oltiin tyytyväisiä siihen miten slotit arpoivat bakteereja ja nopeudet ja ajastukset olivat kohdallaan. Edellisen ja seuraavan bakteerin kuva jätettiin näkyviin yläreunasta, jotta lopputulos näyttäisi hieman yksikätinen rosvo -peliltä.

6.3 Metodit

Käyn lyhyesti läpi kaikkien metodien toiminta-ajatuksen. Metodit löytyvät liitteestä 3. Selitän jokaisen metodin rakenteesta olennaisia asioita, selittämättä kuitenkaan koodin kirjoitussemantiikkaa.

6.3.1 AvaaPopoSivu() ja avaaAjastinSivu()

AvaaPopoSivu() metodissa käytetään Unityn SceneManager luokkaa. Tätä metodia kutsumalla ladataan bakteerien esittelysivu. AvaaAjastinSivu() metodissa käytetään samoin Unityn SceneManager luokkaa ja tätä metodia kutsumalla ladataan pääsivu eli ajastinSivu.

6.3.2 Update()

Tämä metodi on yksi Unityn vakiometodeista. Se ajetaan automaattisesti jokaisessa kehyksessä, jos MonoBehaviour perusluokka on käytössä. Update on MonoBehaviourin funktio. Tässä tiedostossa käytetään update89 metodia audiosekvenssin laskemiseen, äänien soittamiseen, slottien hidastamiseen, jäljellä olevan harjausajan laskemiseen, tuloksen tarkastamiseen sekä slottien ja arvontatuloksen häivyttämiseen/näyttämiseen.

6.3.3 OnSpin()

OnSpin() metodi käynnistää itse laskennan. Sitä kutsutaan painamalla aloita-nappia. Se vie bingoAlueen piiloon kutsumalla ShowbingoAlue() metodia ja tuo slotit näkyville kutsumalla HidePyorijat() metodia. Tässä asetetaan käyttönapit siten että aloita on lukittu, lopeta ja tauko ovat käytettävissä. Tämä tapahtuu kutsumalla asetaKayttoNapit(false,false,3) metodia. asetaKayttoNapit metodille kerrotaan, että aloita-nappi on false, lopeta-nappi on false ja tauko-nappi on 3, mikä tarkoittaa falsea asetaKayttoNapit() metodissa. Jäljellä oleva harjausaika asetetaan 180 sekuntiin ja aikanäyttöön, joka kertoo käyttäjälle jäljellä olevan harjausajan, laitetaan ”3:00”. LaskuriOn boolean arvo laitetaan true:ksi. Tämä kertoo update() metodille että aikaa lasketaan. Slotit 1-3 saavat hidastuvuusarvoksi 1, mikä tarkoittaa ettei pyöriminen hidastu lainkaan. Inertia eli hitaus saa arvoksi true, eli inertia on käytössä. Lopuksi vielä jokaiselle slotille annetaan vauhtia foreach-loopissa. RandonNumber float muuttujaan arvotaan luku 400-800 väliltä. Se antaa pientä vaihtelua eri slottien pyörimisnopeuksiin. AddVelocity funktiolla annetaan slotille nopeutta 5000 lisättynä randomNumber määrällä.

6.3.4 OnStop()

OnStop() metodia kutsutaan painamalla lopeta-nappia. Lopeta-napilla voidaan laskenta päättää kesken. Tällöin ei harjaustuloksesta puhuta mitään vaan ohjelma palautetaan alkutilaansa. Aluksi kutsutaan metodeja ShowbingoAlue() ja HidePyorijat(), jotta saadaan aloituskuva näkyviin ja slotit piiloon häivyttämällä. LaskuriOn boolean muuttuja asetetaan false tilaan, jotta laskenta päättyy jo jäljellä olevan ajan näyttöön asetataan ”3:00” seuraavaa laskentaa odottamaan. Jokainen slotti laitetaan Addvelocityllä nopeuteen nolla eli AddVelocity.zero. Lopuksi asetetaan aloituskuvaksi kuva 4.

6.3.5 OnPause()

OnPause() metodia kutsutaan painamalla tauko-nappia kun nappi on aktiivinen. Tauko-napilla on kolme olomuotoa; harmaa = ei käytössä, värikäs valkoisella tekstillä = käytettävissä ja värikäs keltaisella tekstillä = tauko on aktiivinen. Kun tauko-nappia painetaan

metodissa katsotaan ensin onko harjaus tauolla. Mikäli jäljellä olevan ajan näytössä lukee ”tauko”, on laskenta tauolla. Tällöin metodi ajattelee, että tauko päättyi ja laskentaa jatketaan. Se asettaa slottien 1-3 inertian päälle ja jokaiselle slotille SetVelocityllä tallennetun nopeuden. Lisäksi laskuriOn asetetaan true arvoon, jolloin update() metodissa suoritetaan taas laskentaa. Käyttönapit asetetaan arvoihin false, true, 1. Tällöin aloita-nappi on harmaana eli inaktiivisena, lopeta-nappi on aktiivinen eli suorituksen voi halutessaan lopettaa ja tauko-nappi on valkoisella tekstillä eli käytettävissä uutta taukoa varten.

Mikäli jäljellä olevan ajan näytössä on muuta kuin tauko, metodi olettaa käyttäjän haluvan pitää tauon ja se asetta laskuriOn arvoon false, kutsuu metodia slottienNopeus(), joka tallentaa muuttujiin kunkin slotin nopeuden, asettaa jokaisen slotin 1-3 velocityn eli nopeuden arvoon zero vectorin2 mukaisesti, eli pystysuunnassa. Lisäksi kaikkien slottien inertia asetataan false arvoon, mikä estää joskus Unity elementeillä näkyvää nykimistä, kun inertia on päällä mutta nopeus nolla. Tämä näkyvä nykiminen on sellaista paikallaan tärinää. Jäljellä olevan ajan näyttöön laitetaan ”Tauko” ja käyttönapit kutsumalla metodia asetaKayttoNapit() arvoihin false, true, 2, eli aloita-nappi on harmaa, lopeta-nappi on käytettävissä ja tauko-nappi keltaisella värikkäällä kuvalla.

6.3.6 NimiEsiin(int paneelinNumero)

Tämä metodi katsoo jokaisen slotin näkyvän paneelin nimen talteen valinta1-3 muuttujiin. Tämä tapahtuu kutsumalla SimpleScrollSnap modulin slotsia[x] viittaamalla pinon järjestyslukuun 0-2 ja kutsumalla halutun paneelin CurrentPanel metodia.

6.3.7 NaytaArvonta(string kuva)

NaytaArvonta(string kuva) metodi vaihtaa bingoalueelle halutun kuvan kuvastringissa tulevan tekstin mukaan. Vaihtoehdot ovat bingo, perus ja aloita. Bingo on tähti, jonka saa kun arvonnassa osuu kohdalle kolme samanlaista bakteeria. Perus on kuva, jossa kerrotaan että hampaat on pesty huolella. Kyseisen kuvan saa kolmen minuutin harjauksesta, vaikka bakteerit olisivat toisistaan eriäviä. Aloituskuva on kuva, jonka näkee ohjelmaa käynnistettäessä. Bingokuva gameObjectille on Unityn puolella liitetty aloituskuva image

komponentti. NaytaArvonta vaihtaa spriteä tälle elementille kuva stringin mukaan. kayttoScripti.cs:n alussa on kolme public Sprite kuvaa, joille on Unityn puolella laitettu kuvat bingovoitto ja peruspeli kuvassa 6 ja aloita kuvassa 4.

6.3.8 AsetaKayttoNapit(bool nappi1, bool nappi2, int nappi3)

Tämä metodi säätää alareunan kolmea käyttönappia, jotka näkyvät kuvassa 8. Napeilla ohjataan sovelluksen toimintaa. Metodi odottaa saavansa aina kaksi bool arvoa sekä yhden int arvon. Nappi1 ohjaa aloitusnappia, nappi2 ohjaa lopetusnappia ja nappi3 taukonappia. Napit, jotka laitetaan interactable on true eli toimiviksi tai otetaan pois sen mukaan onko arvo true vai false sekä kuva vaihdetaan falsella harmaaksi ja truella värikkääksi. Nappi3 on poikkeus, koska sillä on kolme olomuotoa. Tämän vuoksi sitä ohjataan numeroilla 1-3. Nappi3 voi olla true sekä keltaisella tekstillä että valkoisella tekstillä, riippuen siitä ollaanko tauolla vai ei.

6.3.9 ShowPyorijat() ja HidePyorijat()

Tällä metodilla ohjataan slotit näkyville tai piiloon häivyttäen. Kun pyorijatFadeIn on true, update() metodilla ajetaan deltaTimen avulla slotit häivyttäen näkyville tai kun pyorijatFadeOut on true niin update() metodilla ajetaan deltaTimen avulla slotit häivyttäen pois näkyviltä.

6.3.10 ShowbingoAlue() ja HidebingoAlue()

Tällä metodilla ohjataan bingoAluetta, jossa on lopputulos/aloituskuva, näkyville tai piiloon häivyttäen. Kun bingoalueFadeIn on true niin update() metodi ajaa deltaTimen avulla bingoAlue häivyttäen näkyville tai kun bingoalueFadeOut on true niin update() metodi ajaa deltaTimen avulla bingoAlueen häivyttäen pois näkyvistä.

6.3.11 SoitaPlot1(), SoitaPlot2(), SoitaPlot3() ja SoitaClap()

Näillä metodeilla soitetaan haluttu ääni metodia kutsuttaessa. Kaikki kolme SoitaPlotX() metodia soittavat saman äänen, mutta jokaista metodia kutsutaan erikseen. SoitaPlot1 on slotille yksi ja muut vastaavasti kaksi ja kolme. Tällä tavoin ääniä voidaan myöskin erikseen ohjata slottien pyörimisnopeuden mukaan ja äänisekvenssiä voidaan vaihtaa pyörimisnopeuden mukaan. SoitaClap() metodi soittaa Clap-äänen, joka on taputusääni. Kun laskuri pääsee nollaan soitetaan taputusääni samalla kun bingoAlueen kuva vaihtuu.

6.3.12 AanetOnOff()

Tämä metodi ohjaa äänet päälle tai pois päältä muuttamalla bool muuttujan aanetPaalla joko true tai false arvoon. Lisäksi samalla vaihdetaan kuvasprite kuvassa 3 olevaan aaniNappi komponenttiin, joka kertoo käyttäjälle ovatko äänet päällä vai poissa. Update() metodissa on if-lauseke, joka joko soittaa äänet tai jättää soittamatta aanetPaalla muuttujan arvon mukaisesti.

6.3.13 PyoriikoKolmasRulla()

Tämä metodi katsoo kolmannen slotin pyörimisnopeuden. Mikäli nopeus on suurempi kuin 300, bool muuttuja kolmasRullaSpeed on true tai nopeus on alle 300 niin muuttujaan laitetaan false. Tämän tiedon mukaan Update() metodissa ohjataan laskennan lopettamista. Kun kolmas rulla on pysähtynyt tarkistetaan arvotut kuvat, tuliko täysosumaa vai ei ja samalla tehdään lopettamistoimenpiteet.

6.3.14 SlottienNopeus()

Tämä metodi tallentaa kutsuttaessa slottien 1-3 nopeudet jokaisen slotin omaan muuttujaan, esimerkiksi slotin1Nopeus muuttujaan. Tietoa käytetään muun muassa update() metodissa ohjattaessa ääniä tai tuloslaskentaa.

6.3.15 Update() (loadinScr.cs tiedosto)

Tämä metodi ajaa pääscenen (ajastinSivu) latauksen määritellyn sekuntimäärän (3) päästä käyttäen Unityn SceneManager luokkaa.

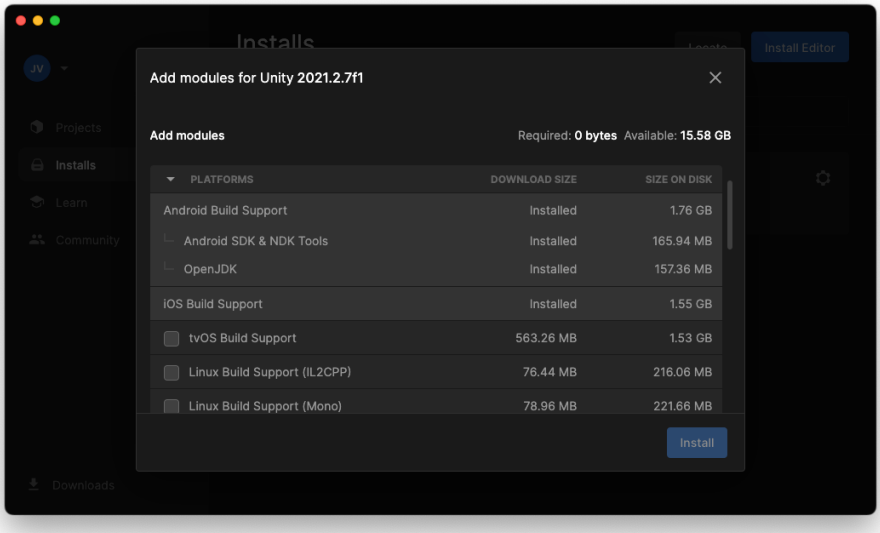
7 Julkaisu

Valmis appi julkaistaan Applen App Storessa. Julkaisua varten sovelluksesta pitää ottaa App Storeen soveltuvat kuvaruutukaappaukset sekä kirjoittaa tarvittavat kuvaustekstit. App Storella on omat vaatimuksensa tekstien minimimitoille sekä kuvien koolle. Ohjelmasta suositellaan myös tekemään lyhyt esittelyvideo.

7.1 Apple Developer Sertifikaatti ja App ID

Julkaistaessa mille tahansa alustalle Unityssä pitää ensin asentaa kyseisen alustan moduli, ellei sitä ole jo valmiiksi asennettuna. Yleensä sen alustan moduli asentuu aina Unityn mukana, mille alustalle Unityn asentaa. Tässä tapauksessa iOS Build Support on tarvittava moduli. Modulin asentaminen tapahtuu Unity Hubissa, installs kohdassa. Kuvassa 16 näkyy, että asennettuna on tarvittavat modulit sekä Androidille että iOSille (Unity Technologies, 2020 -b).

Kuva 16. Unity Hub, modulien hallinta



Seuraavaksi pitää tehdä ja asentaa Apple Developor portaalissa sertifikaatit paikoilleen. Tämä vaatii rekisteröitymisen Apple Developeriksi jollakin Apple ID -tunnuksella. Samalla pitää suorittaa Apple Developer maksu. En käsittele tämän enempää Apple developeriksi rekisteröitymistä. Prosessin voi tehdä https://developer.apple.com/ -osoitteessa, jossa myös tapahtuu kirjautuminen Apple Developer portaaliin. Apple Developer portaalissa, kohdassa Certificates, lisätään uudet sertifikaatit. Uuden sertifikaatin lisääminen tapahtuu plus-merkistä klikkaamalla.

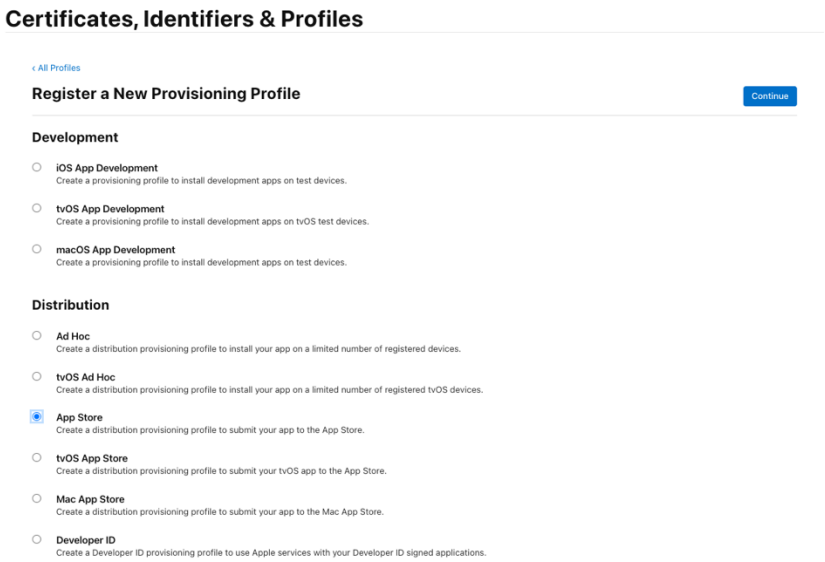
Sertifikaatin tyypiksi valitaan iOS Distribution. Erilaisia vaihtoehtoja on runsaasti erilaisiin tarkoituksiin. esimerkiksi testausta varten iOS Distribution (App Store And Ad Hoc), jolla saa 100 asennuskertaa App Storesta lähinnä pienimuotoista levitystä ja beta-testausta varten. Tähän työhön valitsin kuitenkin suoraan sellaisen sertifikaatin, jolla voidaan levittää valmista sovellusta rajattomasti. Uutta sertifikaattia varten tarvitaan sertifikaatin pyyntötiedosto, jonka luomisen käsittelen kohdassa 7.2. Tämä tiedosto tarvitaan Apple Developer portaaliin. Mikäli valinnat ovat jotenkin ristiriidassa tai esimerkiksi sertifikaattitiedoston muoto on väärä, sivusto ei anna prosessin jatkua. Seuraavassa vaiheessa valmis sertifikaatti voidaan ladata ja ottaa käyttöön ohjelmassa.

Ohjelma tarvitsee oman Apple ID:n. Se on vähän kuin henkilön Apple ID, mutta ohjelmaa varten eikä sillä voi kirjautua erilaisiin Applen toimintoihin. Apple Store tunnistaa tehdyn ohjelman ID:n avulla. Apple Developer portaalissa valitaan Identifiers välilehti, jossa tehdään tälle ohjelmalle oma AppleID. Plussasta lisätään uusi ja valitaan valikosta App IDs. Tämän jälkeen valitaan App ja jatketaan.

Ohjelmalle luodaan Bundle Identifier ja valitaan mahdolliset lisäpalvelut, joita sovellus voisi pyytää laitteelta tai käyttöjärjestelmältä käyttöönsä. Koska sovellus itsessään on melko yksinkertainen ei tästä kohdasta tarvita mitään lisävalintoja. Jatketaan ja valitaan Register. Ohjelmalle on nyt luotu App ID. Tällä tunnisteella App Store tunnistaa sovelluksen ja sitä käytetään niin Unityssä kuin Xcodessakin.

Seuraava vaihe on luoda ohjelman julkaisuprofiili Apple Developerin kohdassa Profiles. Profiilin tyypiksi valitaan App Store, kuva 17. Tämän jälkeen siirrytään seuraavaan vaiheeseen ja valitaan aiemmin luomamme App ID alasvetovalikosta. Jatketaan ja valitaan sertifikaatti listasta, jossa näkyy viimeksi tekemämme sertifikaatti. Seuraavaksi luodaan profiili ja nimetään se. Kirjoitetaan nimi kenttään ja valitaan Generate.

Kuva 17. Julkaisuprofiilin luominen, valitaan jakelutapa

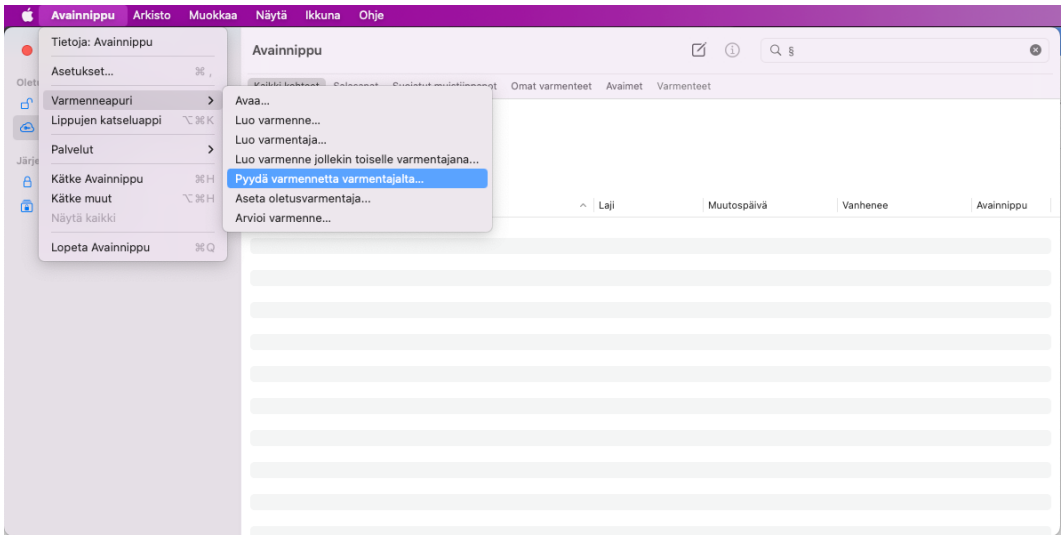


Generate-napin jälkeen tulevalta sivulta ladataan valmis sertifikaatti ohjelmaa varten. Nyt on tuotettu Sertifikaattipyyntö, sertifikaatti ja APP ID. Seuraavaksi siirrytään tekemään julkaisua varten muutokset Unityyn kohdassa 7.2.

7.2 CSR tiedoston generointi

Seuraavaksi tarvitaan .csr-tiedosto, joka on digitaalinen sertifikaattipyyntö. Sertifikaattipyynnön luomiseen on tarjolla useita eri keinoja, joista osa kohdistuu suoraan eri tarkoituksiin. Mikäli tämä sovellus tehtäisiin Microsoft Windowsilla käytettäisiin luultavasti Certreq ohjelmaa komentokehoitteella. Koska sovellus tehdään Macillä tapahtuu tämä sertifikaattipyynnön luominen avainnippu-sovelluksessa. Kuvassa 18 on kuvallinen ohje sertifikaattipyynnön luomiseen. Valitse Avainnippu >> Varmenneapuri >> Pyydä varmennetta varmentajalta.

Kuva 18. Varmenneapuri Avainnippu-sovelluksessa

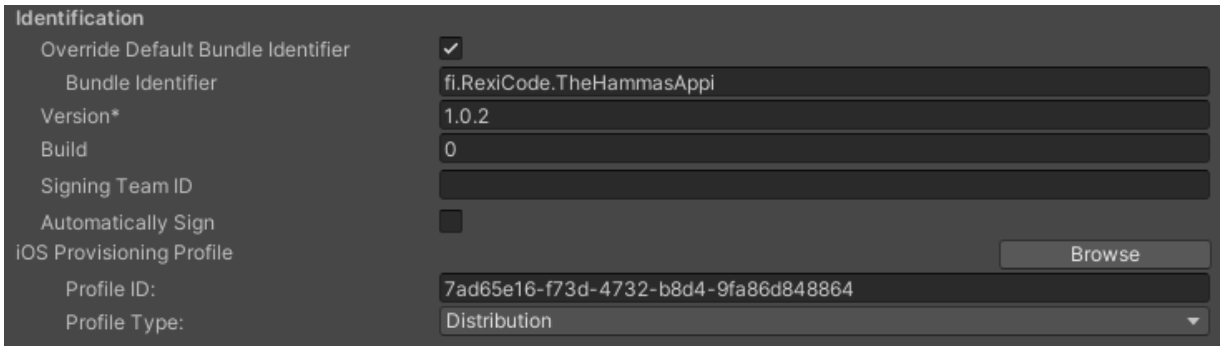


Avautuvaan ikkunaan määritellään sama sähköpostiosoite, jolla on kirjauduttu Apple Developer portaaliin, yleinen nimi sekä valitaan sertifikaattipyynnön tallennus levylle. CA sähköposti -kohdan voi jättää tyhjäksi. Lopuksi tiedosto tallennetaan levylle ja tiedosto syötetään Apple Developer portaalissa sille varattuun kenttään.

7.3 Sovelluksen julkaisemisen valmistelu Unityssä

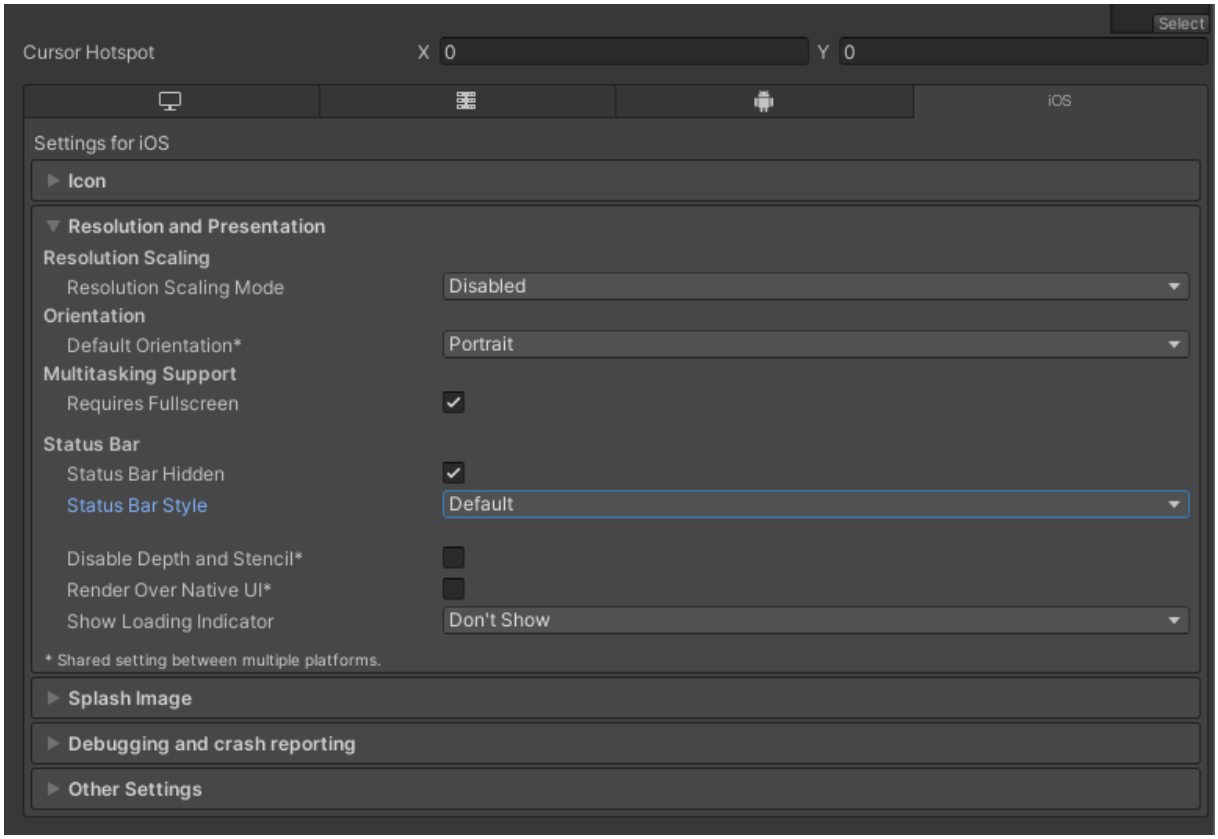
Unityssä laitetaan paikoilleen juuri tehdyt asetukset, jotta julkaisu voidaan tehdä. Unityn File valikosta valitaan Build Settings. Minulla on jo tuossa koldassa valmiina iOS, sillä olen alunperinkin tehnyt tätä sovellusta iOS-profiilin päälle. Player settings kohdasta täytetään Company Name ja Product Name sekä laitetaan versiointinumero. Myös kuvake pitää olla ennen julkaisua ja ne määritellään tässä kohdassa.

Kuva 19. Player settings kohdan perustiedot



Seuraavaksi Other Settings kohta laajennetaan ja valitaan Bundle Identifier Identification kohdan alta, kuten kuvassa 19, ja tähän kenttään kirjoitetaan sama Bundle Indetificator kuin Apple Developerissa luotuun App ID:n laitettiin, kuva 17. Versionumero kannattaa valita huolella ennen julkaisua. Itse olen julkaissut ensimmäisen version 1.0.0 -versiona. Ainakin aiemmin jos Play-kauppa vaati jotain muutoksia ja uudelleen arviointiin lähetyksen piti version kasvaa muutosten välissä vaikka itse Appi ei ollut päässyt julkiseksikaan vielä. Tämä oli kehno ratkaisu, sillä se kasvatti softan versionumeroa jo ennen julkaisua. Lisäksi valitsin, että julkaisen ohjelman vain puhelimelle, sillä koen iPadin käytön harjausajastimena hankalaksi. Valitsen siis iPhone only asetuksen ja minimi käyttöjärjestelmäksi (iOS) 11. Kuvassa 20 lukitsen puhelimen suunnan pystyyn, sillä vaakatason tuesta ei olisi lisäarvoa sovellukselle. Tässä vaiheessa valitaan Build ja Unity kääntää ohjelman Xcodelle ja avaa sen Xcodeen.

Kuva 20. Orientation valinnan lukitseminen pystyyn

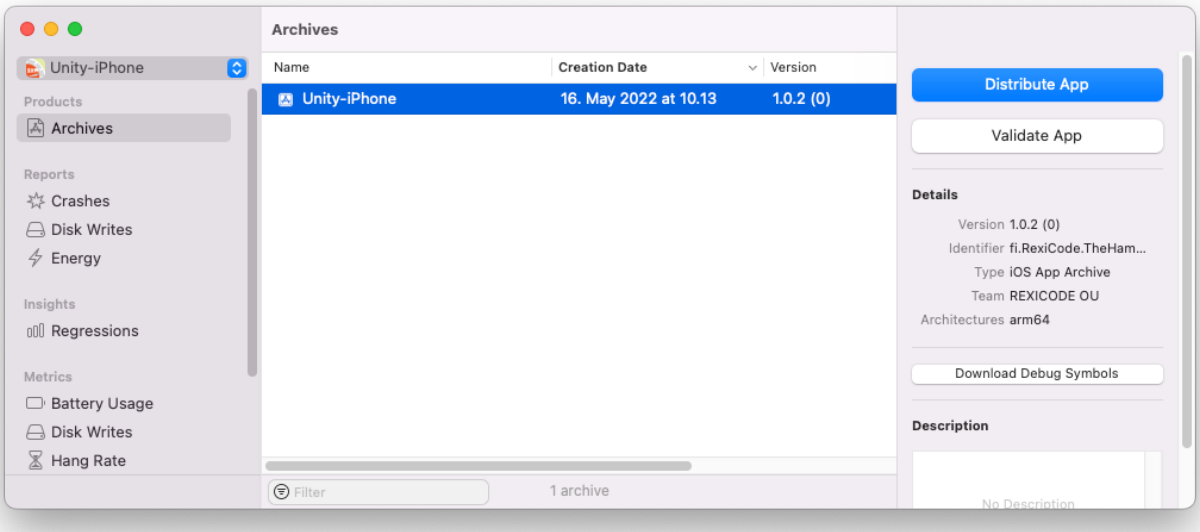


Lopuksi ajetaan Unityssä Build Settings välilehdeltä Build ja annetaan kansio, johon julkaistava versio rakennetaan. Seuraavaksi siirrytään Xcoden puolelle.

7.4 Sovelluksen julkaisun valmistelu Xcodessa

Kun kaikki on laitettu paikoilleen Unityn puolella, laitetaan seuraavaksi kaikki kuntoon Xcoden puolella. Unityssä ajoimme Buildin, joka käänsi Unity projektin Xcodea varten. Tämän käännöksen voi nyt avata suoraan Xcodeen. Xcoden puolella ohjelmasta tehdään arkistoitu versio, joka lopuksi lähetetään Applen AppStoreen. Ohjelmasta pitää ensin tehdä paketti sopivaan muotoon App Storea varten. Ajamme siis Xcodessa Products valikosta Archive ja tämän jälkeen odotellaan kun ohjelma tarkastetaan ja käännetään arkistomuotoon. Kun käännös on valmis arkistoikkunaan ilmestyy arkistoitu ohjelma. Seuraavaksi valitaan arkistoikkunasta Distribute App, kuva 21, joka lataa arkistomuotoisen ohjelman AppStoreen vietäväksi, tarkemmin Applen App Store Connect (Apple Inc, n.d.) -sivustolle.

Kuva 21. Xcode Archives - Distribute App



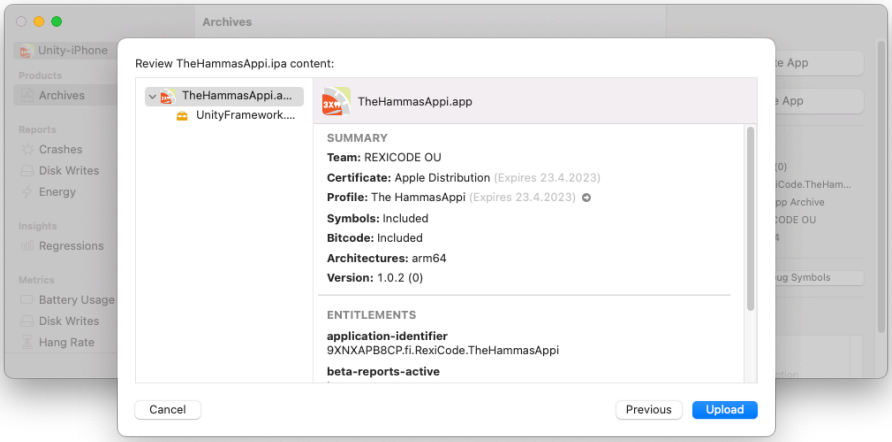
Distribute App aiheuttaa sarjan kysymyksiä, kuten millainen jakelu on kyseessä. Vaihtoehtoina ovat tavallinen App Store, Ad Hoc, Enterprise tai Development. Ad Hoc on tiettyihin laitteisiin osoitettu jakelu. Julkaisusta saa linkin, jonka kautta ohjelman voi asentaa maksimissaan sataan laitteeseen. Enterprice on jakelu, joka on suunnattu yrityksen sisäiseen käyttöön. Development taas on jakelu oman tiimin kesken, jolloin kesken olevan ohjelman saa testaukseen tuotantotiimin kesken. Koska julkaisemme ohjelmaa vapaaseen käyttöön, valitsemme App Store Connect.

Seuraava vaihe on valita lähetetäänkö tämä ohjelma nimenomaan App Store Connectiin vai ainoastaan allekirjoitetaan ja viedään paketti levynkulmalla odottamaan jatkokäyttöä. Valitaan App Store Connectiin. Tässäkin on luonnollisesti useampia vaihtoehtoja. Valitaan kaikki vaihtoehdot, sillä annamme App Storen kääntää sovelluksen kaupan puolella, jolloin saadaan viimeisimmät päivitykset mukaan ohjelmaan automaattisesti. Seuraava valinta koskee Ohjelman symboleja eli bitcode symbolikarttaa. Tämä vie mukana App Storeen dSYM tiedostot, jolloin mahdolliset kaupan kautta tulevat kaatumisraportit kohdistuvat suoraan siihen koodiin joka on Xcodessa. Tämä helpottaa mahdollista virheenetsintää. Valitsen tämän vaihtoehdon, vaikka koodi onkin tehty Unityssä c#:lla ja koodi on Unityn kääntämää. Se voi kuitenkin auttaa löytämään mahdollisen massiivisen virheen tutkimalla Xcoden koodia

ja siitä vertaamalla vastaavaan toiminteeseen Unityssä. Vielä kolmantena valintana on antaa App Storen hallita versiointinumeroita. Koska julkaisuversioni on 1.0.2 niin mahdollinen bugikorjattu versio olisi automaattisesti 1.0.3.

Seuraava vaihe onkin jo valita profiilit, joiden pitää täsmätä aiemmin Apple Developerissa luotujen profiilien kanssa. Profiileja tosin tarjotaan suoraan oletuksena, koska laitoimme ne paikoilleen Unityssä ja Xcodessa. Seuraavaksi Xcoden App Store Connectiin vienti tarkastaa automaattisesti avaimet, koodin, paketin ja kaikki mitä olemme valinneet ja antaa yhteenvedon siitä mitä ollaan julkaisemassa App Storeen, kuva 22.

Kuva 22. Xcodessa yhteenveto ennen ohjelman latausta App Storeen



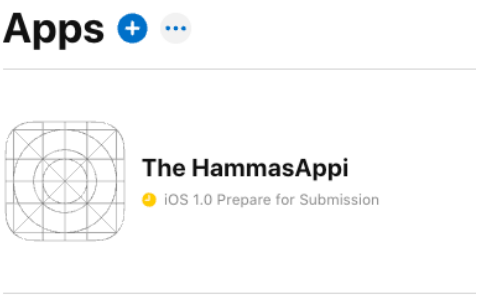
Tämän jälkeen hyväksytään kaikki mitä olemme asettaneet kohdalleen ja annetaan ohjelman aloittaa itse App Storeen lataus. Kuvassa 22 näkyvän Upload napin painamisen jälkeen Xcode tekee vielä lisää tarkastuksia ja käytännössä lataa paketin App Store Connectiin. Kun Xcode on saanut kaikki tarkastukset ja lataukset suoritettua tulee lopulta ruudulle ilmoitus latauksen valmistumisesta. Seuraava vaihe on kertoa App Storelle kaikki tarvittava ohjelmasta, lisätä esittelytekstit ja kuvat sekä suorittaa ohjelman lähetys arvioitavaksi.

7.5 Sovelluksen julkaisun valmistelu ja julkaisu App Store Connectissa

Ohjelma on nyt siirretty App Storeen Connectiin, mutta sitä ei vielä ole julkaistu. Ennen lopullista julkaisua kaupassa ohjelma tarvitsee muun muassa kuvat, hinnat, esittelytekstit sekä Applen hyväksynnän julkaisuun.

Kirjaudutaan App Store Connect -portaaliin osoitteessa https://appstoreconnect.apple.com/apps ja siellä Apps välilehdeltä löytyvät kaikki ohjelmat, jotka tällä kyseisellä tunnuksella on julkaistu tai odottavat julkaisua. Koska ohjelma siirrettiin App Store Connectiin juuri äsken se näkyy vielä harmaana ilman kuvaketta, kuva 23, odottamassa lisätietoja ja lopullista julkaisua.

Kuva 23. App Store Connect -ohjelmajulkaisun viimeistely alkaa



Saadaksemme ohjelman hyväksyttyä ja julkaistua sovelluskaupassa teemme vielä asetteluita ja kerromme App Storelle mitä olemme julkaisemassa. Koska teimme ohjelman vain iPhonelle ja pystyasentoon meidän ei tarvitse laittaa ohjelmasta kuin muutama kuva parille erikokoiselle ruudulle suunnattuna. App Store Connect pyytää kuvia 6.5” ja 5.5” näyttöjen iPhoneille. Koska testilaitteeni on 6.5” näytöllä varustettu iPhone saan kuvaruutukaappaukset otettua helposti suoraan puhelimella. Otan kuvaruutukaappaukset Splash Screenistä, alkuruudusta ja kesken olevasta arvonnasta. Nämä kuvat voidaan laittaa suoraan sellaisenaan App Store Connectiin. Vaikka sivuilla sanotaan, että kuvat muun kokoisille ruuduille tehdään näistä kuvista automaattisesti, huomaan myöhemmin tarkastukseen lähetettäessä että sivusto herjaa puuttuvista 5.5” näytön kuvista. Minulla ei

ole 5.5” näytöllä varustettua iPhonea, joten päädyn venyttämään kuvat sopivan kokoisiksi Adobe Photoshopilla. 6.5” näytöllä varustettu puhelimeni otti suoraan 1284px x 2778px kokoisia kuvia, mutta 5.5” näytön kuvien pitäisi olla kooltaan 1242px x 2208px. Photoshopilla muutetut kuvat kelpasivat App Store Connect sivustolle suoraan.

Ohjelma tarvitsee sovelluskauppaan myös sopivan kuvailevat tekstit. Näihin kuuluvat muun muassa promootioteksti, kuvaus ja lisäotsikko. Otsikoksi tulee ohjelman nimi, joka määräytyi jo Unityn asetuksissa. Ohjelman nimi on siis ”The HammasAppi”. Promootiotekstinä käytän lausetta ”3 minuutin kestoinen The HammasAppi -harjausajastin avuksi lasten hampaiden harjaukseen.”.

Itse kuvaukseksi riittää lyhyehkö teksti ohjelman sisällöstä. ”Ohjelmassa on kolmen minuutin ajastin ja piirrettyjä fiktiivisiä bakteereita. Lisäksi lyhyesti tietoa oikeista suun bakteereista. Ohjelma arpoo kolmen minuutin aikana kolme bakteeria yksikätisen rosvon tyyliin, tarkoituksena on saada kolme samaa bakteeria. Saatko sinä kolmen bakteerin suoran?”. En halua viitata liikaa arvontaan, sillä App Storessa ollaan tarkkoja siitä ettei ohjelma ole uhkapeliluokassa. Tässä ohjelmassa ei voi hävitä eikä asettaa panosta, joten ollaan jo pitkälti turvallisilla vesillä uhkapeliasetusten suhteen.

Ohjelman löytämistä App Storesta auttavat hakusanat ja päädyin seuraaviin sanoihin; hampaiden harjaus, harjausajastin, lasten harjausajastin, suun terveys, hammashygienia. Tälle sivulle kirjataan myös tukisivuston URL eli mistä ohjelman käyttöön voi saada apua. Tähän kenttään laitoin https://rexicode.fi, kuten myös vapaaehtoiseen markkinointisivun asetukseen. Versioksi laitoin 1.0.2 ja copyright kenttään 2022 RexiCode. Sivulla on muitakin mahdollisia konfiguroitavia asetuksia, esimerkiksi iMessage ja Apple Watch asetukset. Sovellus ei käytä näitä, joten jätän ne tyhjäksi. Tällä sivulla voi myös määrittää ohjelman julkaisun. Itse valitsin automaattisen julkaisun. Sitten kun ohjelma hyväksytään Applen päässä ilmestyy se suoraan kauppaan. Muita vaihtoehtoja olisivat olleet ajastettu julkaisu tai manuaalinen julkaisu.

Koska kyseessä on kuitenkin sovelluksen julkaisu eivät yhden sivun tiedot vielä riitä ohjelman lähettämiseksi sovelluskauppaan. Seuraavaksi pitää täyttää tiedot App Information sivulla.

Tällä sivulla voi määrittää sovelluksen julkaisunimen, mutta mikäli se eroaa kovasti siitä mitä ohjelma kertoo olevansa muun muassa Splash Screenissä, ei Apple suostu julkaisemaan sovellusta. Annan siis nimeksi The HammasAppi, alaotsikoksi ”3 minuutin ajastin”. Alaotsikko on sen verran lyhyt kenttä, ettei siihen mahtunut harjausajastin -tekstiä.

Sivulla on toki muutakin tietoa, kuten BundeID ja ohjelman Apple ID. Pääasiallisen kielen asetan arvoon Finnish. Ohjelmalle myös valitaan täällä luokat. Mielestäni sovellus kuuluu kategoriaan Health & Fitness sekä Lifestyle. Muitakin soveltuvia luokkia on olemassa kuten suoranainen medical luokka, mutta siinä julkaisuvaatimukset ovat tiukemmat. Tämä on hyvä asia sen kannalta, jos sovellusta käyttäisi suoranaisena lääketieteellisenä sovelluksena, mutta lapsille suunnattuna harjausajastimena hieman liioiteltua.

Lisäksi tälläkin sivulla on pienimuotoinen kysely koskien kolmannen osapuolen näytettävästä sisällöstä. Tässä sovelluksessa käytin SimpleScrollSnappi modulia, joka on kolmannen osapuolen tuottama mutta se ei ota yhteyksiä sovelluksen ulkopuolelle eikä lataa muualta näytettävää sisältöä sovellukseen. Näin ollen valintani tässä kohdassa on, ettei kolmannen osapuolen sisältöä näytetä.

Valitsen oletuslisenssiehdoksi Applen standardiehdon. En näe syytä kirjoitella ohjelmalle omaa lisenssiehtoa, koska sovellus on ilmainen ja kaikkien vapaasti käytettävissä eikä se kerää mitään tietoa mihinkään.

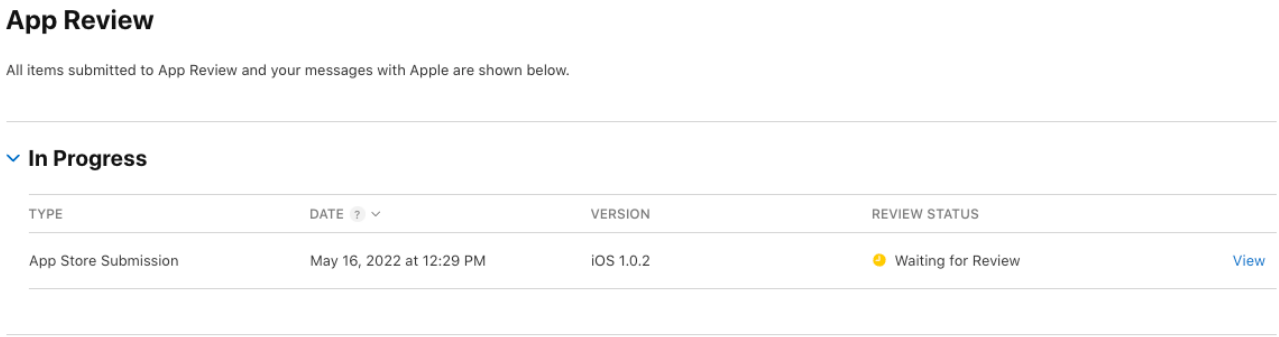
Vielä yksi asetus on tällä sivulla tärkeä, nimittäin ikä. Tämäkin vaatii pienimuotoisen kyselyn täyttämisen, jossa kysytään muun muassa onko ohjelmassa väkivaltaa tai uhkapelejä missään muodossa. Mikäli yksikään valinta tulee edes keskitason valikkoon, hyppää ikäsuositus välittömästi 12+ luokkaan, jota pidän itse hyvänä asiana. Ohjelmani on kuitenkin lapsiturvallinen ja valitsen kaikista kysymyksistä alimman kategorian joka on ei lainkaan. Lopputuloksena sivu arvioi ohjelmani ikäluokkaan 4+ sopivaksi. Tämä sopii hyvin alkuperäiseen suunnitelmaani.

Seuraavaksi siirrytään Pricing and Availability sivulle. Täällä valinnat ovat jo hieman helpompia. Valitsen ohjelman hinnaksi EUR 0.0 (Free) ja Availability kohdassa valitsen ainoaksi sovelluskaupaksi Finland. Maita olisi kaikkiaan tarjolla 175, mutta koska sovelluksen kielenä on suomi julkaisen ohjelman tässä yhteydessä vain ja ainoastaan Suomen sovelluskaupassa. Vielä lopuksi tällä sivulla voi tehdä valintoja sen mukaan onko sovellus julkinen kaikille sovelluskaupassa vai erikoistarjous esimerkiksi Apple School Managerin kautta tai massoittain ostettaessa. Ohjelman voi määrittää myös yksityiseksi siten, ettei sitä tarjota Apple School managerin kautta tai Apple Business Managerin kautta. Oma sovellukseni on kaikkien vapaasti käytettävissä, joten valitsen ”Public”.

Seuraava sivu on App Privacy. Täällä asetan Privacy Policy URL kenttään linkin, josta löytyy kohdassa 5.2 tehty tietosuojalauseke. Saman linkin laitoin User Privacy Choices URL kohtaan. Vaikka App Store Connect sanoo tämän kohdan olevan vapaaehtoinen ei sivun validaattori anna jättää kenttää tyhjäksi. Sivulla on vielä Product Page Preview kohta, jossa otan muutaman kohdan kyselyn koskien datankeräystä. Lopputuloksena ohjelma ei kerää mitään tietoja, kuten aiemmin määrittelin ohjelman tavoitteeksi työn alkupuolella.

Sivuja olisi vielä paljonkin koskien muun muassa Apin sisäisiä ostoja, tilauksia, tarjouskampanjoita, ohjelman sisäisiä tapahtumia, mutta koska sovelluksessani ei ole mitään näistä käytössä en täytä näille sivuille mitään tietoja. On aika palata sivulle App Information, josta kaupan tietojen täyttäminen alkoi. Sivun oikeassa yläkulmassa on nappi ”Submit for Review”. Klikkaan napista ja ohjelma lähetetään Applen arviointiin, kuva 24. Mikäli arviointi menee odotetusti, sovellus julkaistaan automaattisesti sovelluskaupassa aiemmin tekemieni valintojen mukaan.

Kuva 24. The HammasAppi Applen arvioitavana



7.6 Valmis ohjelma App Storessa

Kun App Store review on hyväksytty tulee siitä sähköpostilla ilmoitus. Tämän jälkeen menee noin vuorokausi kunnes appi alkaa näkyä App Storessa. Alla olevasta QR-koodista, kuva 25, pääsee suoraan tehdyn ohjelman sivulle App Storeen ja sen voi ladata ilmaiseksi iOS-käyttöjärjestelmän puhelimille.

Kuva 25. QR-linkki Applen App Storeen, The HammasAppi sovellukseen

8 Yhteenveto

Tuotettu sovellus on julkaistu Applen sovelluskaupassa. Se toimii suunniteltujen speksien mukaan ja on luokiteltu sovelluskaupassa 4+ vuotiaille sopivaksi. Se on GDPR ynteensopiva ja tietoturvallinen sekä toimii ilman internetyhteyttä. Ohjelma itsessään on pieni, mutta pienenkin sovelluksen tuottaminen vaatii samat työvaiheet kuin isomman sovelluksen tuottaminen. Rajasin sovellukseni kokoa, jotta sen toteutusprosessi onnistuu yhdeltä ihmiseltä ja kustannukset pysyvät alhaisina. Jos projekti laajenee tarvitaan pian lisää osaajia ja kustannukset voivat nousta helposti sovelluksen tuottamisen esteeksi varsinkin pienemmillä tilaajilla.

Sovelluskaupassa julkaisu oli yllättävän monimutkainen prosessi, jota hankaloitti entisestään kiristetty tietoturva ja GDPR vaatimukset. Tietoturvan ja yksilötietojen suojaamiseen ei voi panostaa koskaan liikaa ja pienessäkin työssä näitä asioita joutui pohtimaan syvemmin. Mielestäni tietoturvavaatimusten kiristyminen on kuitenkin positiivinen asia.

Mikään sovellus ei ole täydellinen eikä loppuun asti hiottu heti julkaisun jälkeen ja siksi bugikorjauksien lisäksi on hyvä jatkokehittää sovellusta. Oman sovellukseni osalta on noussut ajatuksiin muutamia mahdollisia jatkokehitysajatuksia. Esimerkiksi äänten tilalle voisi tulla kolme minuuttia pitkä luettu satu. Satuja voisi olla erilaisia ja ne arvottaisiin satunnaisesti. Saduissa voisivat seikkailla kuvista tutut bakteerit. Toinen mahdollinen jatkokehitysajatus on yhteydet kehittyneisiin älyhammasharjoihin. Hammasharjoista saisi harjaustiedon, liian lujaa painamisen tiedon ja joistakin voisi saada jopa harjattavan hampaan tiedon. Näitä tietoja voisi käyttää lasten kannalta hauskasti, esimerkiksi lyhytanimaatioihin. Kun saat yhden alueen hampaista harjattua, niin näet animaatiossa kuinka bakteerit juoksevat karkuun suustasi.

The HammasAppi -harjoitustyö osoitti, että pelien valmistamiseen tarkoitettua ohjelmointiympäristöä voidaan käyttää auttavasti myös sovelluksien tuottamiseen. Sovellusta ohjelmoidessani tein huomion, että aikaa kului paljon eri elementtien toiminnan testaukseen ja bugien etsintään.

Lähteet

Apple Inc. (n.d.). *App Store Connect*. Applen appien julkaisujärjestelmä. https://appstoreconnect.apple.com/

Suomen Hammaslääkäriliitto (5.6.2008). *Suunhoito-opas yläkouluille*. https://www.hammaslaakariliitto.fi/sites/default/files/mediafiles/kuvat/suunterveys/suunhoito-opas_ylakouluille.pdf

Lochner, D. (3.3.2022). Simple Scroll-Snap plugin Unitylle, Unity Asset Store. https://assetstore.unity.com/packages/tools/gui/simple-scroll-snap-140884#description

RexiCode Oü (20.4.2022). *TheHammasajastin-tietosuojalauseke*. https://rexicode.fi/TheHammasajastin-tietosuojalauseke.html

RexiCode Oü (1.10.2016). *Postilukija, Future@MailService -reader*. Automaattinen tilausten käsittelijä. https://rexicode.fi/index.php?sivu=fmsr

RexiCode Oü Graafinen ohjeisto (1.2.2022). *RexiCode Oü, yrityksen logo*.

Council of the European Union (11.6.2015). *An official website of the European Union*, Open Data. https://data.consilium.europa.eu/doc/document/ST-9565-2015-INIT/en/pdf

Unity Technologies. (n.d. -a). *Unity Technologies*. https://unity.com/

Unity Technologies. (17.6.2020 -b). *Publishing for iOS*. https://learn.unity.com/tutorial/publishing-for-ios#

Unity Technologies. (n.d. -c). *Unity Technologies Multiplatform*. https://unity.com/solutions/multiplatform

Liite 1: Unityn tukemat laitealustat (Unity Technologies, 2022)

Mobiilialustat

iOS

Android (Android TV)

tvOS

Työpöytäalustat

Windows (Universal Windows Platform)

Mac

Linux

Selainalustat

WebGL

Konsolialustat

PlayStation (PS4, PS5)

Xbox (Xbox One, Xbox Series X/S)

Nintendo Switch

Stadia

Virtuaali / lisätyn todellisuuden alustat

Oculus

PlayStation VR

Google's ARCore

Apple's ARKit

Windows Mixed Reality (HoloLens)

Magic Leap

Via Unity XR SDK

Steam VR

Google Cardboard

Aiemmin tuetut alustat

Wii

Wii U

PlayStation 3

Xbox 360

Tizen

PlayStation Vita

3DS

BlackBerry 10

Windows Phone 8

Samsung Smart TV

Gear VR

Daydream

Vuforia

Facebook Gameroom

Liite 2: Tietosuojalausekkeen lähdekoodi

```
<!Doctype html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Tietosuojalauseke 3 minuutin Hammasajastin</title>
</head>
<style>
div {
font-size:10;
}
</style>
<body>
<div>
<b>1. The controller </b></p>
Name: RexiCode Oü</p>
Y-Business id: 2589049-2</p>
Address: Linjatie 26</p>
Zip code: 04500</p>
Post office: Järvenpää</p>
Phone number: +358 (0)45 164 0007</p>
Email address: info@rexicode.fi</p><br>
<b>2. Person in charge of registry matters</b></p>
Company: RexiCode Oü</p>
Name: Jarno Vallo </p>
Phone number: +358 (0)45 164 0007</p>
Email address: info@rexicode.fi</p><br>
<b>3. Data protection Officer</b></p>
Yritys: RexiCode Oü</p>
Nimi: Aaro Kallioinen</p>
Puhelinnumero: +358 (0)45 164 0007</p>
```

Sähköpostiosoite: info@rexicode.fi</p>

4. Purpose of the register </p>

No data is collected</p>

</p>

</body>

</html>

Liite 3: kayttoScripti.cs ja loadinScr.cs lähdekoodi

```
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.SceneManagement;
using TMPro;
using Random = UnityEngine.Random;
namespace DanielLochner.Assets.SimpleScrollSnap {
  public class kayttoScripti: MonoBehaviour {
    public Button popoRuutu;
    public float timeLeft = 180;
    [SerializeField] private string ohjelmaJokaLadataan;
    [SerializeField] private string paaSivu;
    [SerializeField] private CanvasGroup pyorijat;
    [SerializeField] private CanvasGroup bingoAlue;
    [SerializeField] protected SimpleScrollSnap[] slots;
    public TMP_Text aikaaJaljella;
    public ScrollRect slotti1, slotti2, slotti3;
    public GameObject Contentti;
    public int valinta1, valinta2, valinta3;
    public GameObject rullat;
    public GameObject palkintoKuva;
    bool laskuriOn = false;
    public Sprite bingovoitto, peruspeli, soundOn, soundOff, aloita, aloitaNappiSininen,
aloitaNappiHarmaa, lopetaNappiSininen;
    public Sprite lopetaNappiHarmaa, taukoNappiSininen, taukoNappiKeltainen,
taukoNappiHarmaa;
    public GameObject bingoKuva;
    public Vector2 slotin1Nopeus, slotin2Nopeus, slotin3Nopeus;
    public Button aloitaNappi, lopetaNappi, taukoNappi, aaniNappi;
    public GameObject bakteeriNappi;
    bool aanetPaalla = true;
    bool kolmasRullaSpeed = false;
    public AudioSource plot1, plot2, plot3, clap;
```

```
public double aanisekvenssi1 = 0.3;
public double aanisekvenssi2 = 0.3;
public double aanisekvenssi3 = 0.3;
private bool pyorijatFadeIn = false;
private bool pyorijatFadeOut = false;
private bool bingoalueFadeIn = false;
private bool bingoalueFadeOut = false;
private string sekunnit;
// Tämä metodi avaa bakteereita kertovan sivun
public void avaaPopoSivu() {
 SceneManager.LoadScene(ohjelmaJokaLadataan);
}
//Tämä metodi avaa ajastinsivun
public void avaaAjastinSivu() {
 SceneManager.LoadScene(paaSivu);
}
// Unityn vakio metodi
void Update() {
 // laskuri toiminteet
 if (laskuriOn == true) {
  if (aanetPaalla == true) {
   slottienNopeus();
   //Otetaan aikabase aanisekvenssin muutokselle
   aanisekvenssi1 -= Time.deltaTime;
   aanisekvenssi2 -= Time.deltaTime;
   aanisekvenssi3 -= Time.deltaTime;
   if (aanisekvenssi1 <= 0) {
    if (slotin1Nopeus.y > 500) {
     SoitaPlot1();
     //Jos slot1 nopeus on vakio, soitetaan normaalilla äänisekvenssillä
     if (slotin1Nopeus.y >= 4000) {
      aanisekvenssi1 = 0.3;
     }
     //Jos slot2 nopeus on alle 500, lasketaan sekvenssi sopivaksi pyörimisnopeuteen
```

```
      else {
        aanisekvenssi1 = ((5000 - slotin1Nopeus.y) / 8000);
      }
    }
  }
  if (aanisekvenssi2 <= 0) {
    if (slotin2Nopeus.y > 500) {
      SoitaPlot2();
      if (slotin2Nopeus.y >= 4000) {
        aanisekvenssi2 = 0.3;
      } else {
        aanisekvenssi2 = ((5000 - slotin2Nopeus.y) / 8000);
      }
    }
  }
  aanisekvenssi3 -= Time.deltaTime;
  if (aanisekvenssi3 <= 0) {
    if (slotin3Nopeus.y > 500) {
      SoitaPlot3();
      if (slotin3Nopeus.y >= 4000) {
        aanisekvenssi3 = 0.3;
      } else {
        aanisekvenssi3 = ((5000 - slotin3Nopeus.y) / 8000);
      }
    }
  }
}
if (timeLeft > 0) {
  //Mikäli aikaa on jäljellä, lasketaan jäljellä oleva aika näytettäväksi käyttäjälle
  timeLeft -= Time.deltaTime;
  float minutes = Mathf.Floor(timeLeft / 60);
  float seconds = Mathf.RoundToInt(timeLeft % 60);
  // Lisätään etunolla jos seskunteja on alle 10
  if (seconds < 10) {
```

```
        sekunnit = "0" + seconds.ToString();
       } else {
        sekunnit = seconds.ToString();
       }
       aikaaJaljella.text = minutes.ToString() + ":" + sekunnit;
      }
      if (timeLeft < 120) {
       //Kun aikaa on kulunut 1 minuutti, hidastetaan slottia yksiä ja otetaan valittu paneeli esiin muuttujaan
       slotti1.decelerationRate = 0.75 f;
       nimiEsiin(0);
      }
      if (timeLeft < 60) {
       //Kun aikaa on kulunut 2 minuuttia, hidastetaan slottia yksiä ja otetaan valittu paneeli esiin muuttujaan
       slotti2.decelerationRate = 0.75 f;
       nimiEsiin(1);
      }
      if (timeLeft < 9) {
       //Kun aikaa on kulunut lähes 3 minuuttia, hidastetaan slottia yksi ja otetaan valittu paneeli esiin muuttujaan
       slotti3.decelerationRate = 0.75 f;
       nimiEsiin(2);
      }
      if (timeLeft < 0) {
       pyoriikoKolmasRulla();
       // Aika loppui, laitetaan aloitusaika valmiiksi ja katsotaan tuliko "bingo"
       aikaaJaljella.text = "3:00";
       if (kolmasRullaSpeed == false) {
        if (valinta1 == valinta2) {
         if (valinta2 == valinta3) {
          //Laskuri pois päältä
          laskuriOn = false;
          //Slotit piiloon
          HidePyorijat();
```

```
        //Logoalue esiin
        ShowbingoAlue();
        //Logoaalueeseen jacpot kuva
        naytaArvonta("bingo");
        SoitaClap();
       }
      }
      if (valinta1 != valinta2 || valinta1 != valinta3) {
       //Laskuri pois päältä
       laskuriOn = false;
       //Slotit piiloon
       HidePyorijat();
       //Bingoalue esiin
       ShowbingoAlue();
       //Tavallinen harjaus valmis kuvake esiin
       naytaArvonta("perus");
       SoitaClap();
      }
    }
  }
}
//Pyörijäiden Häivytys
if (pyorijatFadeIn) {
  if (pyorijat.alpha < 1) {
    pyorijat.alpha += Time.deltaTime;
    if (pyorijat.alpha >= 1) {
      pyorijatFadeIn = false;
      asetaKayttoNapit(false, true, 1);
    }
  }
}
if (pyorijatFadeOut) {
  if (pyorijat.alpha <= 1) {
    pyorijat.alpha -= Time.deltaTime;
```

```
      if (pyorijat.alpha <= 0) {
        pyorijatFadeOut = false;
      }
    }
  }
  //Bingoalue Häivytys
  if (bingoalueFadeIn) {
    if (bingoAlue.alpha < 1) {
      bingoAlue.alpha += Time.deltaTime;
      if (bingoAlue.alpha >= 1) {
        bingoalueFadeIn = false;
        asetaKayttoNapit(true, false, 3);
      }
    }
  }
  if (bingoalueFadeOut) {
    if (bingoAlue.alpha <= 1) {
      bingoAlue.alpha -= Time.deltaTime;
      if (bingoAlue.alpha <= 0) {
        bingoalueFadeOut = false;
      }
    }
  }
}
//Tämä metodi käynnistää laskurin ja bakteeriarvonnan
public void OnSpin() {
  //Asetataan äänisekvenssit lähtötasolle
  aanisekvenssi1 = 0.3;
  aanisekvenssi2 = 0.3;
  aanisekvenssi3 = 0.3;
  //Piilotetaan logo kuva
  HidebingoAlue();
  //Näytetään slotit
  ShowPyorijat();
```

```
  //Laitetaan muuttujaan että kolmasrulla pyörii
  kolmasRullaSpeed = true;
  asetaKayttoNapit(false, false, 3);
  //Asetetaan aika alkuun
  timeLeft = 180;
  //Nollataan näytettävä aika
  aikaaJaljella.text = "3:00";
  //Aika laskenta käyntiin
  laskuriOn = true;
  //Asetetaan slottien hidastusvauhdiksi 1
  slotti1.GetComponent < ScrollRect > ().decelerationRate = 1;
  slotti2.GetComponent < ScrollRect > ().decelerationRate = 1;
  slotti3.GetComponent < ScrollRect > ().decelerationRate = 1;
  slotti1.GetComponent < ScrollRect > ().inertia = true;
  slotti2.GetComponent < ScrollRect > ().inertia = true;
  slotti3.GetComponent < ScrollRect > ().inertia = true;
  // Slotit vauhtiin
  foreach(SimpleScrollSnap slot in slots) {
    float randomNumber = Random.Range(400, 800);
    slot.AddVelocity((5000 + randomNumber) * Vector2.up);
  }
 }
 //Tämä metodi pysäyttää bakteeeriarvonnan ja laskurin
 public void OnStop() {
  //Näytetään logokuva
  ShowbingoAlue();
  //Piilotetaan slotit
  HidePyorijat();
  //Asetetaan käyttönapit oikein
  asetaKayttoNapit(false, false, 3);
  //pysäytetään laskuri
  laskuriOn = false;
  //Asetetaan näytettävä aika kolmeen minuuttiin
  aikaaJaljella.text = "3:00";
```

```
  //Varmisteaan että kaikki slotit on seis
  foreach(SimpleScrollSnap slot in slots) {
    slot.AddVelocity(Vector2.zero);
  }
  //Näytetään aloituslogo
  naytaArvonta("aloita");
}
//Tämä metodi laittaa laskurin tauolle tai jatkaa laskentaa tauolta
public void OnPause() {
  //Jos ollaan tauolla, niin jatketaa
  if (aikaaJaljella.text.Equals("Tauko")) {
    //Sloteille nopeudeksi tallennettu nopeus
    slotti1.GetComponent < ScrollRect > ().inertia = true;
    slotti2.GetComponent < ScrollRect > ().inertia = true;
    slotti3.GetComponent < ScrollRect > ().inertia = true;
    if (slotin1Nopeus.y >= 500) {
      slots[0].SetVelocity(slotin1Nopeus);
    }
    if (slotin2Nopeus.y >= 500) {
      slots[1].SetVelocity(slotin2Nopeus);
    }
    if (slotin3Nopeus.y >= 500) {
      slots[2].SetVelocity(slotin3Nopeus);
    }
    //Laskuri käyntiin jatkaen kohdasta mihin jäätiin
    laskuriOn = true;
    //Asetetaan käyttönapit oikein
    asetaKayttoNapit(false, true, 1);
  }
  //Aloitetaan tauko
  else {
    //Laskuri seis
    laskuriOn = false;
    //Otetaan slottien nopeus talteen
```

```
    slottienNopeus();
    //Asetataan kaikki slotit on seis
    slotti1.velocity = Vector2.zero;
    slotti2.velocity = Vector2.zero;
    slotti3.velocity = Vector2.zero;
    //Asetaan kaikkien Slottien inertia seis
    slotti1.GetComponent < ScrollRect > ().inertia = false;
    slotti2.GetComponent < ScrollRect > ().inertia = false;
    slotti3.GetComponent < ScrollRect > ().inertia = false;
    //Ilmoitetaan käyttäjälle että ollaan tauolla
    aikaaJaljella.text = "Tauko";
    //Asetetaan käyttönapit oikein
    asetaKayttoNapit(false, true, 2);
  }
}
// Tämä metodi ottaa talteen paneelit jotka on arvonnassa valittu
public void nimiEsiin(int paneelinNumero) {
  foreach(SimpleScrollSnap Paneeli in slots) {
    // Jokaisen arvotun paneelin nimi talteen omaan muuttujaan
    valinta1 = slots[0].CurrentPanel;
    valinta2 = slots[1].CurrentPanel;
    valinta3 = slots[2].CurrentPanel;
  }
}
//Tämä metodi näyttää joko arvottavat slotit tai bingoalueen logoin oikealle kuvalla
public void naytaArvonta(string kuva) {
  // Asetetaan kuvaksi päävoitto -kuva
  if (kuva.Equals("bingo")) {
    bingoKuva.GetComponent < Image > ().sprite = bingovoitto;
  }
  //Asetetaan kuvaksi tavallinen harjauskuva
  else if (kuva.Equals("perus")) {
    bingoKuva.GetComponent < Image > ().sprite = peruspeli;
  }
```

```
  //Asetetaan kuvaksi aloituskuva
  else if (kuva.Equals("aloita")) {
    bingoKuva.GetComponent < Image > ().sprite = aloita;
  }
}
// Metodi säätää käyttönappien statusta ja asettaa niihin tarvittavat kuvat valintojen mukaan
public void asetaKayttoNapit(bool nappi1, bool nappi2, int nappi3)
{
  //Aloitusnappi
  if (nappi1 == true) {
    //Asetetaan aktiiviseksi ja värikäs kuva
    aloitaNappi.interactable = true;
    aloitaNappi.GetComponent < Image > ().sprite = aloitaNappiSininen;
    //Asetetaan bakteerisivulla ohjaava nappi pois käyttöön ja näkyville
    bakteeriNappi.SetActive(true);
  } else {
    //Asetaan nappi tilaan, ettei sitä voi käyttää ja harmaaksi
    aloitaNappi.interactable = false;
    aloitaNappi.GetComponent < Image > ().sprite = aloitaNappiHarmaa;
    //Asetetaan bakteerisivulla ohjaava nappi pois käytöstä ja näkyvistä
    bakteeriNappi.SetActive(false);
  }
  // Lopetusnappi
  if (nappi2 == true) {
    //Asetetaan aktiiviseksi ja värikäs kuva
    lopetaNappi.interactable = true;
    lopetaNappi.GetComponent < Image > ().sprite = lopetaNappiSininen;
  } else {
    //Asetaan nappi tilaan, ettei sitä voi käyttää ja kuva harmaaksi
    lopetaNappi.interactable = false;
    lopetaNappi.GetComponent < Image > ().sprite = lopetaNappiHarmaa;
  }
  // Taukonappi
  if (nappi3 == 1) {
```

```
    //Asetetaan aktiiviseksi ja värikäs kuva
    taukoNappi.interactable = true;
    taukoNappi.GetComponent < Image > ().sprite = taukoNappiSininen;
  } else if (nappi3 == 2) {
    //Asetetaan taukonappiin keltaisen tekstin kuva, mikä kertoo että nappi on käytössä
    taukoNappi.interactable = true;
    taukoNappi.GetComponent < Image > ().sprite = taukoNappiKeltainen;
  } else if (nappi3 == 3) {
    //Asetaan nappi tilaan, ettei sitä voi käyttää ja kuva harmaaksi
    taukoNappi.interactable = false;
    taukoNappi.GetComponent < Image > ().sprite = taukoNappiHarmaa;
  }
}
// Tämä metodi ohjaa häivytyksen kautta bingoalueen ja slottien näyttöä
void ShowPyorijat()
{
  pyorijatFadeIn = true;
}
void HidePyorijat() {
  pyorijatFadeOut = true;
}
void ShowbingoAlue() {
  bingoalueFadeIn = true;
}
void HidebingoAlue() {
  bingoalueFadeOut = true;
}
// Tämä metodi soittaa PLOT-äänen
void SoitaPlot1() {
  plot1.Play();
}
void SoitaPlot2() {
  plot2.Play();
}
```

```
  void SoitaPlot3() {
    plot3.Play();
  }
  // Tämä metodi soittaa Clap-äänen
  void SoitaClap() {
    clap.Play();
  }
  //Tämä metodi säätää äänet pois päältä / päälle ja vaihtaa äänikuvakkeen sen mukaan sopivaksi
  public void aanetOnOff() {
    if (aanetPaalla == true) {
      aanetPaalla = false;
      aaniNappi.GetComponent < Image > ().sprite = soundOff;
    } else {
      aanetPaalla = true;
      aaniNappi.GetComponent < Image > ().sprite = soundOn;
    }
  }
  // Tämä metodi kertoo ajastukselle, pyöriikö kolmas rulla vielä
  void pyoriikoKolmasRulla() {
    if (slotti3.velocity.y > 300) {
      kolmasRullaSpeed = true;
    } else {
      kolmasRullaSpeed = false;
    }
  }
  void slottienNopeus() {
    slotin1Nopeus = slotti1.velocity;
    slotin2Nopeus = slotti2.velocity;
    slotin3Nopeus = slotti3.velocity;
  }
 }
}
```

LoadinScr.cs

```
using UnityEngine;
using UnityEngine.SceneManagement;
public class loadinScr: MonoBehaviour {
  [SerializeField]
  private float delayBeforeLoading = 3 f;
  [SerializeField]
  private string ohjelmaJokaLadataan;
  private float timeElapsed;
  // Update is called once per frame
  void Update() {
    timeElapsed += Time.deltaTime;
    if (timeElapsed > delayBeforeLoading) {
      SceneManager.LoadScene(ohjelmaJokaLadataan);
    }
  }
}
```

Liite 4: Käytetty grafiikka


HIENOA,
HAMPAAT PESTY
HUOLELLA!


3
minuutin
HARJAUS-
AJASTIN


REX
CODE