TAMPEREEN
AMMATTIKORKEAKOULU

MAINOSRATKAISUT MOBIILIPELEILLE ANDROID ALUSTALLA

Anssi Lappalainen

Elokuu 2016
Tietotekniikan koulutusohjelma
Ohjelmistotekniikka

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tietotekniikan koulutusohjelma
Ohjelmistotekniikka

LAPPALAINEN, ANSSI:
Mainosratkaisut mobiilipeleille Android alustalla

Opinnäytetyö 25 sivua, joista liitteitä 1 sivu
Elokuu 2016

Opinnäytetyön tarkoituksena oli löytää sopiva mainosratkaisu Polar Dashing mobiilipelille. Mainosratkaisun valinta suoritettiin tutkimalla Android alustalle tarjolla olevia mainosratkaisuita ja pisteyttämällä ne sovelluskehittäjän näkökulmasta olennaisissa vertailukategorioissa. Työ rajoittuu mainontaan ja ei kata ilmaispelien muita ansaintamalleja.

Työ kuvaa prosessia, jonka kautta kartoitettiin yleisimmät saatavilla olevat mainosratkaisut, jotka täyttävät asetetut vaatimukset. Karsinnan läpäisseet mainosratkaisut testattiin Android laitteilla ja pisteytettiin ohjelmistokehittäjille olennaisten ominaisuuksien perusteella. Mainosratkaisuiden sopimusten mukaisesti mainosratkaisuiden toiminnasta kertova sisältö on poistettu jälkeenpäin ja mainosratkaisuita vertaillaan yleistetysti paljastamatta tietoa, joka ei ole julkisesti saatavilla ilman käyttöoikeussopimuksien hyväksyntää.

Pisteytyksen tarkoitus on antaa kuva mainosratkaisuiden vahvuuksista ja heikkouksista sekä auttaa lukijaa priorisoimaan oman tutkimuksen kohteita sopivan mainosratkaisun etsinnässä. Valittu mainosratkaisu integroitiin peliin ja peli julkaistiin Google Play kaupassa.

Asiasanat: mobiilimarkkinointi, Android, Unity

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Degree Programme in ICT Engineering
Software Engineering

LAPPALAINEN, ANSSI:
Advertising Solutions for Mobile Games on Android Platform

Bachelor's thesis 25 pages, appendices 1 page
August 2016

The subject of this thesis was to find suitable advertising solution for Polar Dashing mobile game. Selection process was carried out by doing research on available advertising solutions and grading them in categories that are viewed as important from software developers point of view. Subject of this thesis is limited to advertising solutions and doesn't cover other commonly used monetization models in mobile games.

This thesis describes the process which was used to map out available advertising solutions that fulfil requirements set by the game. Advertising solutions that fulfilled the requirements were tested on Android devices and were given score according to how well they did in categories essential to software developers. In accordance with user agreements any information that might not be publicly available without accepting the user license agreements has been removed.

Purpose of these scores is to give an idea about strengths and weaknesses of these advertising solutions and help the reader to prioritize research of their own in finding a suitable advertising solution for their needs. Chosen advertising solution was integrated into the game before publishing it at Google Play store.

Key words: mobile marketing, Android, Unity

SISÄLLYS

ERITYISSANASTO

Pelimoottori	ohjelmistokehys, joka helpottaa ja nopeuttaa pelinkehittäjien työtä tarjoamalla yleishyödyllisiä työkaluja pelinkehitykseen.
Objektien ryhmittäminen	ohjelmointi tekniikka, jolla pyritään välttämään objektien turhaa luomista ja tuhoamista. Objektien tullessa tarpeettomiksi, ne siirretään takaisin käyttämättömien objektien ryhmään (object pool), josta ne voidaan tarvittaessa ottaa käyttöön nopeasti.
Google Play	Googlen tarjoama kauppa mobiilisovelluksille.
COPPA	Yhdysvalloissa lasten verkkoyksityisyyttä suojeleva laki (Children's Online Privacy Protection Act).
PEGI3	eurooppalainen ikäluokitus ryhmä, jonka omaava tuote sopii vähintään kolmevuotiaille lapsille.
ECMA	tietoliikenne standardeja kehittävä järjestö

1 JOHDANTO

Mobiilipelien yleisimmin käytetty liiketoimintamalli on julkaista peli ilmaiseksi (AppBrain 2016) ja pyrkiä saamaan tuloja joko mainoksilla tai myymällä peliin kuuluvaa sisältöä (Munir 2014). Tämän opinnäytetyön tavoite on vertailla markkinoilla olevia mainosratkaisuita ja löytää parhaiten sopiva mainosratkaisu Polar Dashing mobiilipelille. Pelin kehityksen lopussa se päätettiin julkaista ilmaiseksi Android alustalle Google Play kaupassa ja käyttää mainoksia tulolähteenä. Tämä opinnäytetyö keskittyy ainoastaan Android alustan mainosratkaisuihin ja muita alustoja tai pelin sisäisten tuotteiden myyntiä ei käsitellä lainkaan.

Aluksi kartoitettiin olemassa olevat mainosratkaisut ja määriteltiin kriteerit, jotka mainosratkaisun tulee täyttää. Kaikki kriteerit täyttävät mainosratkaisut pisteytettiin ja asetettiin järjestykseen sen perusteella, kuinka hyvin ne täyttivät asetetut vaatimukset. Jäljelle jäävistä vaihtoehdoista valittiin pelille parhaiten soveltuva mainosratkaisu ja integroitiin se peliin ennen pelin julkaisua.

Luvussa 2 käsitellään työn taustaa ja määritellään mainosratkaisulta vaaditut kriteerit, sekä asetetaan työn tavoitteet. Kolmas luku kuvaa työn toteutuksessa käytettyjä teknologioita. Neljäs luku kuvaa yleisesti mainosratkaisuiden toimintaperiaatetta ja viidennestä luvusta käy ilmi varsinaisen työn kulku, tulokset sekä loppuratkaisun integrointi. Lopuksi kuudennessa luvussa ajatuksia työn kulusta ja tuloksista sekä mietteitä jatkokehitysehdotuksista.

2 TYÖN TAUSTA

2.1 CosyCave

CosyCave on keväällä 2015 perustettu yritys. Yritykseen kuuluu kaksi henkilöä: Ohjelmoija Anssi Lappalainen ja graafikko Catarina Sarmento. Yrityksen pääasiallinen toimiala on ohjelmistosuunnittelu. Yrityksen ensimmäinen tuote, Polar Dashing mobiilipeli, julkaistiin heinäkuussa 2015 Google Play kaupassa.

2.2 Polar Dashing

Työ Polar Dashing pelin parissa alkoi keväällä 2014. Pelin tekoon osallistui kaksi Tampereen ammattikorkeakoulun opiskelijaa: tietotekniikan opiskelija Anssi Lappalainen ja median opiskelija Ana Catarina Fonseca. Projektin tavoite oli tehdä yksinkertainen peli Android alustalle ja julkaista se Google Play kaupassa ilmaiseksi. Pelissä on tarkoitus ohjata jäällä ja lumella liukuvaa pingviiniä väistellen esteitä ja keräten kaloja. Kerätyistä kaloista pelaaja saa pisteitä ja pelissä on erilaisia kaloja, jotka joko helpottavat tai vaikeuttavat peliä väliaikaisesti. Violetit pilaantuneet kalat nopeuttavat pingviinin kulkua, joten esteiden väistely vaikeutuu ja kultaiset kalat antavat yhden ylimääräisen yrityksen ennen pelin loppumista. Tavallisset siniset kalat ovat perus arvoltaan 10 pistettä ja mustekalat 100 pistettä. Pisteet moninkertaistuvat, mikäli pelaaja onnistuu keräämään kaikki normaalit kalat jättämättä väliin yhtään. Pistekerroin katoaa, jos yksikin kala jää keräämättä.

Peli loppuu pelaajan törmätessä esteeseen. Mikäli pelaajalla on jäljellä vähintään yksi kultainen kala, niiden kokonaismäärästä vähennetään yksi ja pelaaja voi jatkaa peliä seuraavaan törmäämiseen saakka. Tarkoituksena on saada mahdollisimman paljon pisteitä ja peli pitää kirjaa korkeimmasta saavutetusta pistemäärästä. Esteiden ja kalojen tyyppi ja sijainti ovat satunnaisia, joten jokaisella pelikerralla pelin kulku on erilainen edelliseen kertaan verrattuna.

KUVA 1. Kuvakaappaus Polar Dashing pelistä

Pingviinin liikkeitä ohjataan kosketusnäytön eleillä. Kosketusnäyttöä vasemmalle pyyhkäistessä pingviini liikkuu vasemmalle ja oikealle pyyhkäistessä oikealle. Pingviini menee eteenpäin vakionopeudella ja pingviinin sijainnille on kolme eri kaistaa; vasemmalla, oikealla tai keskellä. Peli luo satunnaisia elementtejä pingviinin edetessä ja poistaa elementit niiden poistuessa ruudulta.

2.3 Vaatimukset ja tavoitteet

Opinnäytetyön tavoitteena oli löytää Polar Dashing peliin sopiva mainosratkaisu ja integroida se peliin. Mainosratkaisun tulee täyttää vähintään seuraavat vaatimukset:

- tukee Android käyttöjärjestelmää
- mahdollistaa videomainoksien näyttämisen
- mahdollistaa pelaajan palkitsemisen videon katselun jälkeen
- täyttää COPPA määräykset.

Peli julkaistaan ainoastaan Android alustalle, joten muiden käyttöjärjestelmien tukea ei tarvittu. Videomainoksiin päädyttiin tutkien ja vertaillen suosittujen pelien mainostyyppejä. Suurin osa näistä peleistä on omaksunut käytännön, jossa pelaaja palkitaan videomainoksen katsomisen jälkeen. Myös tilastot ja ennusteet USA:n markkinoilta, joka on yksi suurimmista älypuhelinmarkkinoista, tukevat videomainosten valintaa. Vuonna 2015 mobiilimainonta kasvatti markkinaosuuttaan muihin mainosmedioihin nähden 25 %:sta 35 %:iin (IAB/PwC Internet Ad Revenue Report, 2015, 13).

Ennusteiden mukaan mainonnan painotuksen siirtyessä enemmän mobiililaitteille muilta osa-alueilta, mobiilivideomainosten odotetaan kasvattavan markkinaosuuttaan keskimäärin 73% kertyvän vuotuisen kasvuprosentin vauhtia (Hoelzel, 2014).

Näiden tietojen pohjalta päädyttiin käyttämään videomainoksia, joiden katsomisen jälkeen pelaaja palkitaan. Palkintona pelaajalle toimivat kultaiset kalat, jotka antavat uuden yritysmahdollisuuden pelaajalle, kun hän epäonnistuu väistämään estettä. Polar Dashing pelille haettiin Google Play kaupassa PEGI3 ikäluokitusta, jonka vuoksi mainosten tulee olla myös nuorille lapsille sopivia. Koska pelin tulee olla lapsille sopiva, tulee sen myös olla myös USA:n COPPA määräysten mukainen, joka mm. kieltää keräämästä mitään henkilötietoja alle 13-vuotiailta ilman vanhempien lupaa. Mainosratkaisut, jotka eivät täytä näitä edellytyksiä hylättiin karsinnassa välittömästi. Päätös jäljelle jäävien mainosratkaisuiden kesken tehtiin tarkastelemalla mainosratkaisuiden eroavaisuuksia seuraavissa asioissa:

- käyttöjärjestelmävaatimukset
- käyttöoikeusvaatimukset
- Internetyhteyden vaatimukset
- käyttäjätilin hallinnan yksinkertaisuus
- dokumentaation laatu
- API:n laatu
- statistiikka- ja analysointityökalujen monipuolisuus.

Käyttöjärjestelmävaatimuksilla tarkoitetaan Android käyttöjärjestelmän minimiversiovaatimusta. Mitä pienempi käyttöjärjestelmävaatimus on, sitä useammalla laitteella ohjelmaa voidaan käyttää. Pyrkimys on saada ohjelma toimimaan Android versiolla 2.3.3, joka kattaisi 99,9% markkinoilla olevista laitteista. Ehdoton maksimi Android versiolle on Android 4.1, joka Googlen tilastojen mukaan kattaisi 94,7% markkinoilla olevista laitteista. (Android Developer Dashboard 2016)

Tietoturvasta huolissaan olevat käyttäjät saattavat olla epäileväisiä tarpeettomien käyttöoikeuksien suhteen. Mitä vähemmän oikeuksia käyttöjärjestelmältä ohjelma tarvitsee, sen mutkattomampaa ohjelman käyttöönotto on. Mainosratkaisulla ei pitäisi missään vaiheessa olla tarvetta esim. tehdä puheluita, käsitellä kontaktitietoja tai päästä käsiksi puheluhistoriaan.

Mobiiliyhteyksien maksimisiirtonopeudet ovat lähivuosien aikana nousseet huomattavasti, mutta tarjolla on edelleen myös hitaita yhteyksiä. Mobiiliyhteyksien nopeudet vaihtelevat suuresti ja monta minuuttia kestävät videomainokset saattavat viedä useita kymmeniä megatavuja tilaa. Mainosratkaisun, joka lataa videoita internetin kautta, täytyy sopeutua erilaisiin yhteysnopeuksiin. Mitä hitaammalla Internet-yhteydellä tullaan toimeen, sitä parempi käyttäjäkokemus voidaan tarjota hitaimpien yhteyksien käyttäjille.

Käyttäjätilin asetusten vaihtaminen tulee olla mahdollisimman helppoa ja halutut asetukset pitäisi löytyä helposti hallintapaneelista. Tilin luonti pitäisi olla yksinkertaista ja nopeaa, jotta kynnys kokeilla mainosratkaisua olisi mahdollisimman matala. Ratkaisu, jonka ominaisuudet ja toimivuus on helppo testata, antaa ohjelmoijalle positiivisen ensivaikutelman.

Mainosratkaisua integroidessa usein herää kysymyksiä mainosratkaisun toiminnan yksityiskohdista. Tässä auttavat koodiesimerkit, kattavasti dokumentoitu API sekä selkeät esimerkki projektit. Kaikki nämä seikat on huomioitu vertaillessa ratkaisuiden dokumentaation laatua.

API:n laatu on usein hyvin subjektiivinen käsite, joten on tärkeää pyrkiä määrittämään tarkasti mitä sillä kussakin tapauksessa tarkoitetaan. Tässä tapauksessa laatu voidaan jakaa kahteen tarpeen osa-alueeseen: käytön yksinkertaisuus ja toimintojen monipuolisuus siihen verrattuna. Mitä enemmän toimintoja yksinkertaiseen API:in voidaan sisällyttää, sen parempi API on laadultaan.

Mainospalvelun toimintaa voidaan parantaa ainoastaan, mikäli on kerätty tilastotietoa sen nykyisestä toiminnasta. Hyvä mainospalvelu tarjoaa käyttäjilleen tilastotietoa, jonka perusteella käyttäjä voi säätää mainosten asetuksia parempien ansaintamahdollisuuksien toivossa. Statistiikka- ja analysointityökalut ovat tärkeitä tämän tiedon hankinnassa.

3 KEHITYSYMPÄRISTÖ JA TYÖKALUT

3.1 Android

Android on Android Inc. yhtiön kehittämä käyttöjärjestelmä, jonka Google osti vuonna 2005 (Thomas 2010). Google I/O tapahtumassa (2008) julkaistun tiedon mukaan Android-käyttöjärjestelmän perustana on Linux ja sen ydintoiminnot on ohjelmoitu C-kielellä. Käyttöliittymä- ja ohjelmistokehittäjille tarjotut toiminnot käyttävät Java-kieltä. (Brady 2008). Android on pääasiassa tarkoitettu mobiililaitteille, mutta käyttöjärjestelmää on käytetty myös muissa tarkoituksissa kuten esim. autoissa, televisioissa, pelikonsoleissa ja stereojärjestelmissä (Dennis 2014).

3.2 Mono

Mono-projektin verkkosivulla kuvataan Monon olevan avoimen lähdekoodin versio Microsoftin .NET ohjelmistokomponenttikirjastosta ja perustuu ECMA:n määrittelemään C# ja CLR standardiin (Mono Project 2016). Unityn käyttämä Mono kirjaston versio vastaa .NET 2.0 toiminnallisuutta, johon on lisätty joitakin .NET version 3.5 ominaisuuksia (Hauwert 2014).

3.3 Unity

Unity on Unity Technologiesin (n.d.) kehittämä pelimoottori. Unity on ohjelmoitu C++ kielellä ja pelikehittäjät voivat käyttää C# tai UnityScript -kieliä pelien ohjelmointiin. C# kieli tulkataan käyttäen muokattua Mono 2.0 kehitysympäristöä ja UnityScript pohjautuu löyhästi Javascript -ohjelmointikieleen (Unity Technologies 2016).

Unity on vakiinnuttanut paikkansa mobiilipelimarkkinoilla ja on erittäin suosittu pienten pelinkehitysyritysten keskuudessa helppokäyttöisyytensä ansiosta. Unity valittiin pelimoottoriksi aikaisempien kokemusten perusteella, koska uuden ohjelmointiympäristön opetteleminen veisi aikaa ja viivästyttäisi pelin julkaisua. Vision Mobilen tekemän tutki-

muksen mukaan (2014), Unity pelimoottoria käyttää 47 % pelin kehittäjistä ja 29 % pelinkehittäjistä käyttää sitä ensisijaisena pelinkehitysalustana. Lähimpänä kilpailijana mobiilipelien kehityksessä mainitaan itse kehitetyt pelimoottorit. Tilastoissa Unity:n lähin kilpailija julkisesti saatavilla olevista pelimoottoreista on Cocos2D, jota käyttää ensisijaisena pelimoottorina noin 9 % pelinkehittäjistä. (Vision Mobile 2014).

3.4 Visual Studio 2013 ja 2015

Visual Studio on Microsoftin kehittämä ohjelmankehitysympäristö, joka sisältää koodi editorin ja debuggerin (Visual Studio Products 2016). UnityVS liitännäisen ansiosta Visual Studio sopii hyvin ohjelmointiympäristöksi Unity pelimoottorille (Visual Studio Tools for Unity 2016). Visual Studio valittiin ohjelmointiympäristöksi, koska Unityn tarjoama vaihtoehto, MonoDevelop, on aikaisempien kokemusten perusteella epävakaampi ja rajoitetumpi vaihtoehto suhteessa Visual Studioon. Unityn versiosta 5.1.0 lähtien UnityVS liitännäinen on ollut mahdollista asentaa Unity -pelimoottorin asennuksen yhteydessä, mikäli ohjelmointiin halutaan käyttää Visual Studio ympäristöä (SyntaxTree, n.d.). Kaikki ohjelmointi suoritettiin Visual Studion Community versioita 2013 ja 2015 käyttäen.

3.5 Git

Git on versionhallintatyökalu, jonka keskeinen idea on versionhallinnan hajautus (n.d.). Perinteisen keskitetyn versionhallintapalvelimen sijaan jokaisella käyttäjällä on oma versio kehitteillä olevasta ohjelmistosta ja kehitys tapahtuu jakamalla ja yhdistelemällä koodia ohjelmistokehittäjien kesken (Git, n.d.). Tässä projektissa Git toimi lähinnä henkilökohtaisena varmuuskopiointityökaluna, koska projektissa oli vain yksi ohjelmoija. Git palvelun isäntänä toimi BitBucket -palvelun ilmaisversio. Git valittiin versionhallintatyökaluksi aikaisempien kokemusten perusteella. Git on ilmainen, vakaa ja helppokäyttöinen. Mikäli tiedostoja poistetaan vahingossa tai ne vaurioituvat, ne on helppo palauttaa entiselleen versionhallinnan kautta.

3.6 Android Studio

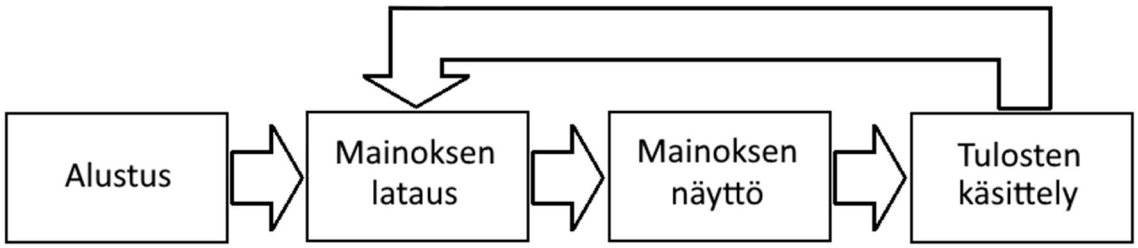
Ducrohet, Norbye ja Chou (2013) esittelivät Android Studion ensimmäistä kertaa Andoid kehittäjien blogissa. Android Studio tarjoaa työkaluja ohjelmistokehittäjille, jotka haluavat kehittää ohjelmia Android alustalle. Android Studio ohjelmistopaketti sisältää Android SDK koodikirjastot sekä ohjelmointiympäristön, joka on rakennettu IntelliJ IDEA -ohjelmointiympäristön päälle (Ducrohet, Norbye & Chou, 2013). Android SDK koodikirjastot sisältävät yleishyödyllisiä pienohjelmia, jotka auttavat Android ohjelmistokehityksessä, kuten laitelokien lukuohjelman, Android emulaattorin sekä ohjelman lisäohjelmistopakettien ja -työkalujen asentamiseen (Android SDK tools).

4 MAINOSPALVELUIDEN TOIMINTAPERIAATE

Suurin osa mainospalveluista toimii saman periaatteen pohjalta. Mainospalvelun käyttäjä, joka haluaa tarjota mainoksia mobiilisovelluksen yhteydessä, luo itselleen käyttäjätilin. Käyttäjätilin avulla mainospalvelu tunnistaa mainosten näyttäjän ja määriteltyjen asetusten kautta tietää hänen tarpeensa mainospalvelun suhteen. Yleensä käyttäjätilin hallintapaneelista saa luotua asetukset erikseen jokaiselle sovellukselle, joka näyttää tarjottuja mainoksia. Näin sovelluskehittäjä voi tarjota erityyppisiä mainoksia eri sovelluksissa. Kun asetukset on määritelty, sovelluskehittäjälle annetaan tunnuskoodi, jonka avulla hän voi käyttää sovelluskohtaisia asetuksia.

Kun käyttäjätili on luotu ja kaikki hallintapaneelin asetukset ovat kunnossa, tarvitaan mainospalvelun tuottajalta kirjasto, joka kommunikoi sovelluksen ja mainospalvelun välillä. Mainospalveluntarjoajien kirjastojen laatu on vaihtelevaa, mutta ne kaikki toimivat noudattaen samaa periaatetta. Mainospalvelun käyttäjä lähettää ensin komennon, joka luo kommunikaatio yhteyden mainoksia tarjoavalle palvelimelle. Kun yhteys on muodostettu, voidaan palvelimelle tehdä mainospyyntö. Sovelluskohtaisten asetusten tunnuskoodi syötetään joko yhteyttä muodostaessa tai mainoksia pyytäessä riippuen mainosten tarjoajan toteutuksesta. Kun palvelimelta pyydetään mainosta onnistuneesti, mainoksen lataus laitteelle alkaa.

Mainoksen latausprosessin tarkkailuun käytetään kahta eri menetelmää. Näistä yleisempi ja käyttäjäystävällisempi tapa on callback-funktion käyttö. Tässä tapauksessa ennen mainoksen latausta määritellään funktio, joka suoritetaan, kun mainoksen lataus on valmis. Funktiossa voidaan määritellä toimenpiteet, jotka käydään läpi kun mainoksen lataus on suoritettu loppuun. Toinen tapa tarkkailla mainoksen latauksen valmiutta on ns. polling -menetelmä. Tässä vaihtoehtoisessa menetelmässä tehdään tietyin väliajoin kyselyjä mainoksen tilasta. Vastauksena saadaan tieto, onko mainos valmis näytettäväksi. Tämän tiedon pohjalta voidaan tehdä kaikki samat asiat, kuin callback-funktio tekniikalla.



KUVIO 1. Mainosten näyttö prosessi

Mainoskampanjoista maksetut palkkiot perustuvat ohjelman käyttäjän vuorovaikutukseen mainoksen kassa. Mainoksen katselu tapahtumasta tarvitaan palautetta, jotta pelaaja voitaisiin palkita mainoksen katsomisen jälkeen asian mukaisesti. Ohjelman kehittäjä, joka näyttää mainoksen pelaajalle, saa korvauksen perustuen mainoksen näkyvyyteen ja käyttäjän vuorovaikutukseen mainoksen kanssa. Mainosten tarjoaja määrää kriteerit, joiden perusteella korvauksen määrä lasketaan. Yleinen sääntö kuitenkin on, mitä enemmän mainoksen näyttäminen toi arvoa mainostajalle, sitä paremmin mainoksen näyttänyt sovelluskehittäjä palkitaan. Lähes poikkeuksetta on mahdotonta arvioida ansaintamahdollisuuksia, koska mainostulot ovat riippuvaisia monesta eri tekijästä, joita mainosten tarjoajat eivät avoimesti kerro. Ainoa konkreettinen tapa vertailla mainosratkaisuiden soveltuvuutta kyseessä olevalle pelille on ottaa käyttöön vertailtavat mainosratkaisut ja verrata saatuja tuloksia keskenään.

5 TUTKIMUS JA TYÖ

5.1 Tutkimus

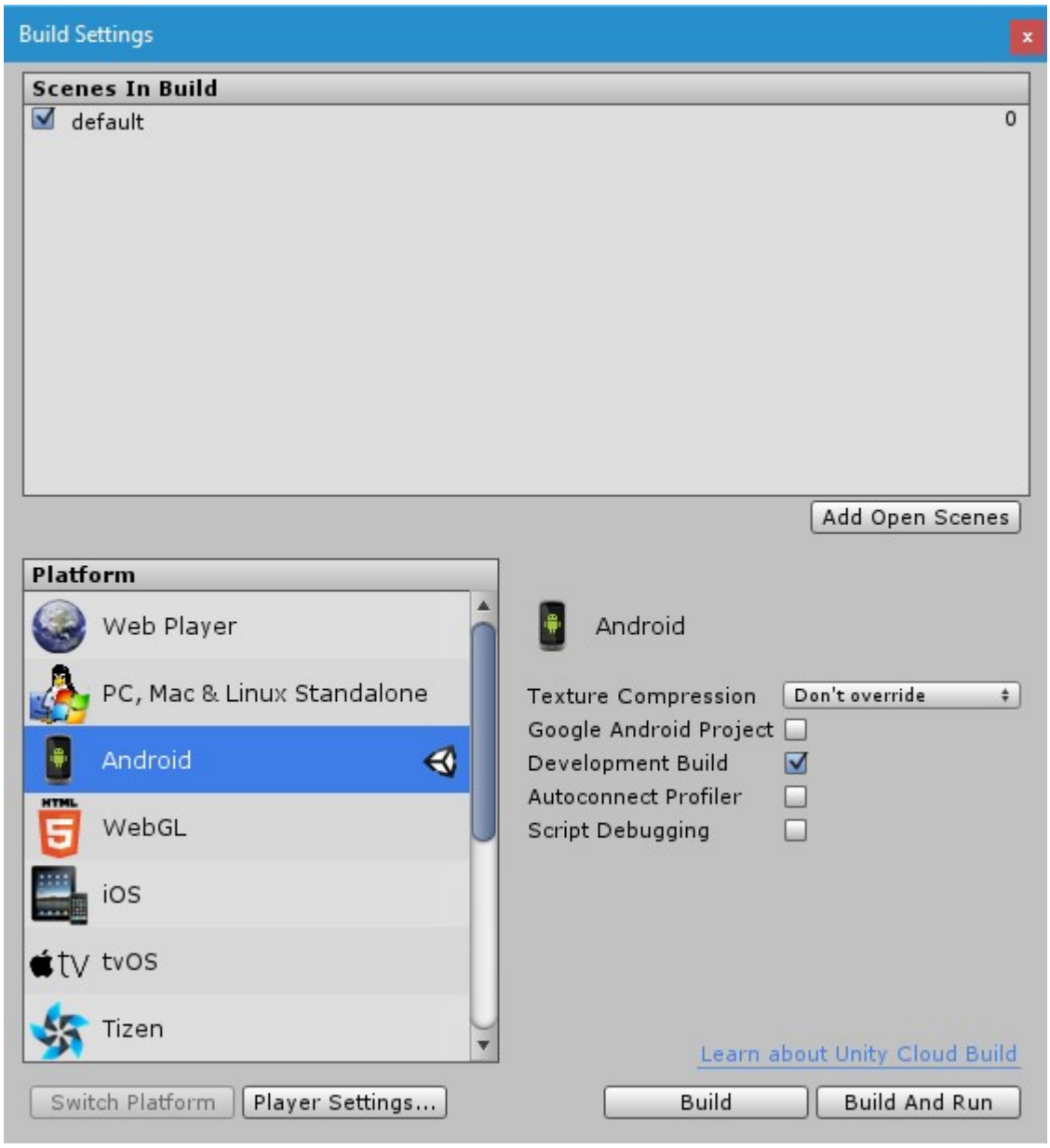
Ennen mainosratkaisuiden testausta tehtiin tutkimustyötä julkisesti saatavilla olevien tietojen pohjalta. Tässä työvaiheessa karsittiin mainosratkaisut, jotka eivät täyttäneet vaatimuksia tai joissa havaittiin muita olennaisia ongelmia ensisilmäyksellä. Tutkimusvaiheen jälkeen luotiin käyttäjätilit palveluihin, jotka alustavasti täyttivät ennalta asetetut vaatimukset. Tilin luonnin yhteydessä annetaan mainospalvelun tarjoajalle tietoja palvelun käyttäjästä, kuten nimi ja yhteystiedot. Käyttäjätilin luonti edellyttää palveluntarjoajan käyttöehtojen hyväksymistä.

Käyttäjätilin asetukset ovat palveluntarjoajakohtaisia. Yleisesti ottaen palvelussa luodaan sovelluskohtainen tunnus, jonka avulla määritellään asetukset kyseissä sovelluksessa käytetyille mainoksille. Näiden asetusten kautta usein määritellään mm. mainosten tyyppi, sijainti, kohderyhmät ja mainosten tyyppi. Kun asetukset oli muokattu haluttuun kokoonpanoon, luotiin Unity projekti mainosten testatusta varten. Unity projektia luodessa Android alustalle on otettava huomioon seuraavat seikat:

- testaus tapahtuu Android laitteella, ei Unity editorissa kuten yleensä
- useat mainosratkaisut luottavat Unity:n alustakohtaisiin määrityksiin
- kääntäjä tarvitsee kelvollisen ”bundle identifier” -arvon projektin asetuksiin
- AndroidManifest.xml -tiedoston pitää olla ratkaisun vaatimusten mukainen.

Normaalisti Unity -peliä kehitettäessä, virheviestit ohjautuvat Unityn konsoli-ikkunaan. Android laitteella peliä testatessa virheviestien lukemiseen täytyy käyttää Android alustan kehitystyökaluja. Virheviestit voidaan lukea Android laitteelta käyttäen ”adb logcat” -käskyä komentokehotteessa. Käsky näyttää kaikki yhdistetyn Android laitteen lokitiedot, joten puskurista löytyvästä tiedosta on hyvä suodattaa pois kaikki ylimääräinen tieto, josta ei ole hyötyä ohjelmaa kehittäessä. ”adb logcat -s Unity” -komento näyttää kaikki Unity -ympäristöön liittyvät viestit ja ”adb logcat -c” -komento tyhjentää loki puskurin. Puskuri on hyvä tyhjentää aina kokeilujen välillä, että edellisen käynnistyskerran virheitä ei vahingossa sekoiteta uusien virheiden kanssa ohjelmakoodin muuttuessa.

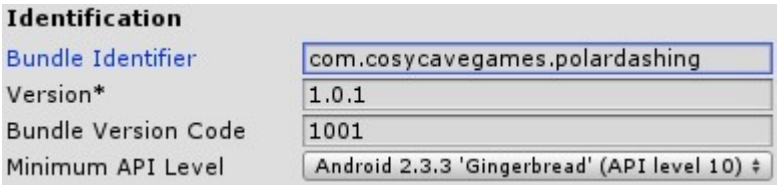
Mainosratkaisut tukevat yleensä useita eri käyttöjärjestelmiä, joista Android on vain yksi monesta. Mainosratkaisut tunnistavat käyttöjärjestelmän Unity -projektin asetusten perusteella. Julkaisualusta määritellään käyttämään Android alustaa *Build Settings* > *Platform* -valikosta.



KUVA 2. Android käyttöjärjestelmän valinta alustaksi

Android ohjelmaa ei voida kääntää, ennen kuin sille on määritelty paketti tunniste (bundle identifier). Android paketti tunniste on muotoa *pääte.domain.ohjelma*, joka Polar Dashing pelillä on *com.cosycavegames.polardashing*. Pakettitunniste on jokaiselle Android sovellukselle ainutlaatuinen ja Google Play palvelu tunnistaa sovelluksen tämän

tunnisteen perusteella. paketti tunniste voidaan määritellä *Edit* > *Project Settings* > *Player* -valikosta *Android* > *Identification* -alaotsikon alta.



KUVA 3. Android pakettitunnisteen määrittely

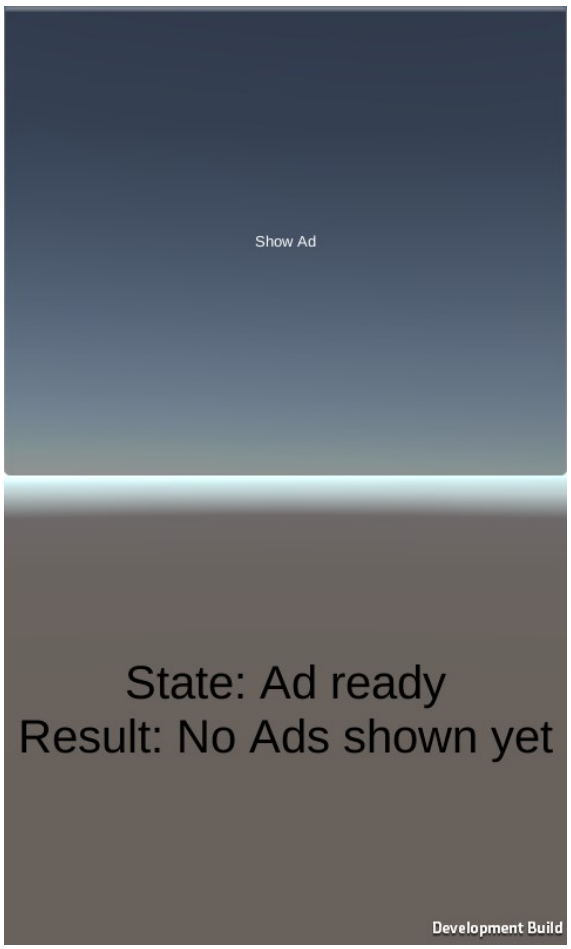
Android ohjelman kääntäminen edellyttää Java Development Kit (JDK) asentamista. Kaikissa testeissä ja loppuratkaisun integrointiin käytettiin viimeisintä JDK versiota (1.8.0.60), joka julkaistiin kesäkuussa 2015.

Testattavia mainosratkaisuita on useita, joten aluksi luotiin Unity projekti pohja, jonka pohjalta kehitettiin sovellus jokaisen mainosratkaisun testaukseen. Projekti pohja sisältää kaikki tarvittavat projekti asetukset sekä pohjustavaa ohjelmakoodia, joka tulee olemaan osa jokaista testiä. Saman pohjakoodin käyttäminen kaikille ratkaisuille vähentää tekijöiden määrää, jotka vaikuttavat mainosratkaisun toimintaan vertailua tehdessä. Pohjakoodin käyttäminen nopeuttaa myös kehitysprosessia, koska samaa koodia ei tarvitse erikseen kirjoittaa useaan kertaan eri ratkaisuille.

LISTAUS 1. Mainosratkaisuiden testauksen pohjakoodi

```
using UnityEngine;

public class AdTest : MonoBehaviour
{
    private string resultText = "Result: " + "No ads shown yet";
    private string adStateText = "State: " + "Ad not ready";
    GUIStyle centeredTextStyle;
    bool ready = false;

    void Start()
    {
        centeredTextStyle = new GUIStyle("label");
        centeredTextStyle.alignment = TextAnchor.MiddleCenter;
        centeredTextStyle.fontSize = 40;

        // Ad solution initialization here
    }

    void OnGUI()
    {
        if (GUI.Button(new Rect(0, 0,
            Screen.width, Screen.height / 2), "Show Ad"))
        {
            ShowRewardedAd();
        }
        GUI.color = Color.black;
        GUI.Label(new Rect(0, Screen.height / 2, Screen.width,
            Screen.height / 2), adStateText + "\n" + resultText,
            centeredTextStyle);
    }

    public void ShowRewardedAd()
    {
        if (ready)
        {
            // Code to show ad here
        }
        else
            resultText = "Result: " + "Interstitial failed to load";
    }
}
```

Ennen varsinaisten mainosratkaisuiden testausta varmistetiin pohjakoodin toiminta kääntämällä ohjelma ja lataamalla se Android laitteelle. Pohja koodi sellaisenaan luo käyttöliittymän, joka sisältää yhden napin ja kaksi tekstiselitettä, joita käytetään näyttämään tietoa ohjelman tilasta (Kuva 4).



KUVA 4. Käyttöliittymä mainosratkaisuiden testaukseen

Osa mainosratkaisuista vaatii muutoksia ohjelman käyttöoikeuksiin toimiakseen. Oletusarvoisesti Unity vaatii oikeuden käyttää Internet yhteyttä ja luvan tiedon tallentamiseen laitteelle. Lisäoikeuksien antaminen ohjelmalle tehdään AndroidManifest.xml tiedoston kautta. Unity -projektissa AndroidManifest.xml luodaan projektin ”Assets/Plugins/Android” -alikansioon. Mikäli ohjelmaa käännettäessä Unity havaitsee AndroidManifest.xml tiedoston tässä alikansiossa, Unity käyttää tätä tiedostoa ohjelmaa kääntäessä normaalin automaattisesti luodun asetustiedoston sijaan.

Suurin osa mainosratkaisuiden käyttöehtosopimuksista kieltää julkistamasta mitään mainosratkaisuun tai sen toimintaan liittyvää tietoa, joka ei ole jo muuten julkisesti saatavilla. Mainosratkaisuille annettiin pisteet erikseen jokaisessa vertailukategoriassa, jotta ratkaisuita voitaisiin vertailla keskenään julkistamatta yksityiskohtia niiden toiminnasta. Pisteytyksen tarkoitus on antaa lukijalle kuva mainosratkaisuiden vahvuuksista ja heikkouksista sekä lähtökohta oman tutkimuksen suorittamiseen.

Mainosratkaisuiden pisteytys suoritettiin kohdassa 2.3 mainittujen perusteiden pohjalta, mutta yksityiskohdat, jotka käsittelevät mainosratkaisun toimintaa on jätetty tarkoituksella pois, jotta käyttöoikeussopimuksia ei rikottaisi. Tasapuolisen vertailun saavuttamiseksi myös ne mainosratkaisut, jotka sallisivat tiedon jakamisen kolmansille osapuolille, käsitellään samoja rajoituksia noudattaen. Jokaiselle mainosratkaisulle on annettu erikseen pisteet (taulukko 1) jokaisessa kategoriassa 1-5: 1 = huono, 2 = välttävä, 3 = keskiverto, 4 = hyvä ja 5 = erinomainen.

TAULUKKO 1: Mainosratkaisuiden pisteytys

	Käyttöjärjestelmän vaatimukset	Käyttöoikeus vaatimukset	Internetyhteyden vaatimukset	Käyttäjätilin hallinta	Dokumentaation laatu	APIn laatu	Statistiikka- ja analysointityökalut	Keskiarvo
AdMob	4	4	4	3	4	3	5	3,86
Unity Ads	4	4	3	4	4	5	3	3,86
AdColony	4	4	4	3	2	4	4	3,57
StartApp	?	3	4	3	3	2	4	3,17
HeyZap	4	4	2	4	4	3	4	3,57
AdBuddiz	5	5	4	3	2	3	2	3,43
RevMob	5	4	4	2	3	2	3	3,29

Vertailun perusteella sopivin mainosratkaisu oli Unity Ads. Unity Ads täyttää kaikki asetetut vaatimukset ja on yksi vertailussa parhaiten pisteytetyistä mainosratkaisuista. Unity Technologies on kehittänyt sekä Unity pelimoottorin, että Unity Ads mainosratkaisun, joten mainosten integrointi sovellukseen on hyvin suoraviivaista ja myös päivitykset sujuvat oletettavasti suoraviivaisemmin verrattuna muihin ratkaisuihin. Unity Ads ei tarvitse erillistä liitännäistä toimiakseen, vaan se on integroitu suoraan Unity editoriin.

5.2 Valitun ratkaisun integrointi

Mainoksien katsominen on mahdollista sekä pelin jälkeen, että pelin päävalikossa. Käyttöliittymän painike, jonka kautta mainoksia voi katsoa, näytetään ainoastaan, jos pelaajalla ei ole yhtään kultaista kalaa ja pelaaja on pelannut peliä vähintään kerran käynnistämisen jälkeen ja edellisen mainoksen katselusta on kulunut vähintään 20 sekuntia ja mainoksen lataus palvelimelta on suoritettu. Käyttöliittymän Earn-painiketta ei näytetä ollenkaan, mikäli yksikin näistä ehdoista jää täyttämättä.

KUVA 5: Earn-painike pelin päävalikossa

Käyttöliittymäelementtien asettelu muuttaa muotoaan sen mukaisesti, näytetäänkö Earn-painiketta valikossa. Kun pelaaja painaa nappia hänelle näytetään mainos. Pelaaja palkitaan mainoksen katsomisesta kultaisella kalalla, mikäli mainos katsottiin loppuun saakka. Loppuratkaisun lähdekoodi on listattu liitteessä 1.

6 YHTEENVETO

Työn tavoite oli löytää sopiva mainosratkaisu Polar Dashing -mobiilipelille Android alustalla. Tavoite saavutettiin ja peli julkaistiin Google Play kaupassa käyttäen Unity Ads mainosratkaisua. Jokaisessa mainosratkaisussa havaittiin pieniä puutteita ja parantamisen varaa tietyissä rajatapauksissa, mutta käyttäjätilastojen mukaan Unity Ads mainosratkaisu on toiminut ongelmitta vuoden ajan. Puoli vuotta pelin julkaisemisen jälkeen pelillä oli ainoastaan 10-20 yhtäaikaista aktiivista pelaajaa, joten vertailuja mainosratkaisuiden ansaintamahdollisuuksien kesken ei pystytty tekemään. Unity Ads kehittäjien mukaan, keskimäärin mainoksilla voi ansaitaan noin $6/1000 katselua, mutta tarvittavaa pelaajamäärää tämän tiedon tarkastamiseksi ei koskaan saavutettu.

Alun perin opinnäytetyön tarkoituksena oli yksityiskohtaisesti tuoda esiin jokaisen testatun mainosratkaisun heikkoudet ja vahvuudet. Tästä lähestymistavasta jouduttiin kuitenkin luopumaan, koska useat mainosratkaisuiden käyttöehtosopimukset kieltävät julkistamasta mitään sellaista tietoa mainosratkaisuiden toiminnasta, joka ei ole julkisesti saatavilla ennen sopimuksen hyväksymistä. Tästä johtuen yksityiskohtaisen analyysin sijaan tyydyttiin jokainen karsinnan läpäissyt mainosratkaisu pisteyttämään ohjelmistokehittäjälle olennaisissa kategorioissa. Tämä ei ole ideaalinen lähestymistapa, mutta toivottavasti tämän lähestymistavan avulla lukijalle muodostuu karkea kuva mainosratkaisuiden vahvuuksista ja heikkouksista.

Jatkokehityksen kannalta kiinnostava tutkimuksen kohde voisi olla useiden eri mainosratkaisuiden käyttäminen rinnakkain. Näin voisi olla mahdollista välttää tiettyyn mainosratkaisuun sidonnaiset ongelmat, jotka voivat johtua esim. sijainnista tai yhteyden laadusta. Mikäli yksi mainosratkaisuista ei jostain syystä toimi, pyrittäisiin mainokset näyttämään käyttäen toista mainosratkaisua.

LÄHTEET

Android Developers, About Android: Dashboards. Luettu 25.2.2016. https://developer.android.com/about/dashboards/index.html

Android SDK Tools. n.d. Luettu 17.8.2016 https://developer.android.com/studio/command-line/index.html

Android Studio: An IDE built for Android. 2014. Luettu 17.8.2016. http://android-developers.blogspot.fi/2013/05/android-studio-ide-built-for-android.html

AppBrain. 2016. Free vs. paid Android apps. Luettu 17.8.2016. http://info.localytics.com/blog/app-monetization-6-bankable-business-models-that-help-mobile-apps-make-money/

Munir, A. 2014. App Monetization: 6 Bankable Business Models That Help Mobile Apps Make Money. Luettu 17.8.2016. http://info.localytics.com/blog/app-monetization-6-bankable-business-models-that-help-mobile-apps-make-money/

Brady, P. 2008. YouTube. Google I/O 2008 - Anatomy and Physiology of an Android. Katsottu 15.8.2016. https://www.youtube.com/watch?v=G-36noTCaiA

Dennis, R. 2014. Not Just For Phones And Tablets: What Other Devices Run Android? Luettu 15.8.2016. http://www.makeuseof.com/tag/phones-tablets-devices-run-android/

Git. n.d. Luettu 17.8.2016. https://git-scm.com/about/

Hauwert, R. 2014. The future of scripting in Unity. Luettu 17.8.2016. https://blogs.unity3d.com/2014/05/20/the-future-of-scripting-in-unity/

Hoelzel, M. 2014. Business Insider. Mobile Video Advertising Is Taking Off, As Ad Buyers Pile Billions Of Dollars Into Small-Screen Ads. Luettu 25.4.2016. http://www.businessinsider.com/mobile-video-is-the-growth-area-2014-10

Interactive Advertising Bureau & PricewaterhouseCoopers LLP. 2015. IAB internet advertising revenue report 2014 full year results April 2015. Luettu 25.4.2016. http://www.iab.com/wp-content/uploads/2015/05/IAB_Internet_Advertising_Revenue_FY_2014.pdf

Mono Project. n.d. Luettu 17.8.2016. http://www.mono-project.com/

SyntaxTree. n.d. Luettu 17.8.2016. Native support for VSTU 2.1 in Unity 5.2. http://unityvs.com/documentation/native-support/

Thomas, O. 2010. VentureBeat. Google exec: Android was “best deal ever”. Luettu 12.8.2016 http://venturebeat.com/2010/10/27/google-exec-android-was-best-deal-ever/

Unity Technologies. n.d. Luettu 17.8.2016. http://unity3d.com/unity/editor/

Vision Mobile. 2014. State of the Developer Nation Q3 2014. Luettu 16.7.2016 http://www.visionmobile.com/product/developer-economics-q3-2014/

Visual Studio. n.d. Luettu 17.8.2016. https://www.visualstudio.com

Visual Studio Products. Luettu 17.8.2016. Overview of Visual Studio 2015 Products. https://www.visualstudio.com/vs-2015-product-editions/

LIITTEET

Liite 1. Unity Ads loppuratkaisun lähdekoodi

```
using System;
using UnityEngine;
using UnityEngine.Advertisements;

public class EarnFishButton : MonoBehaviour {
    void Start() {
        if (GameManager.Instance)
            gameObject.SetActive(GameManager.Instance.CanEarnFish());
    }

    public void EarnFish()
    {
        Advertisement.Show(null, new ShowOptions {
            pause = true,
            resultCallback = result => {
                // Only condition that matters is did they finish
                // watching the ad
                if (result == ShowResult.Finished && GameManager.Instance) {
                    GameManager.Instance.AddGoldenFish();
                    GameManager.Instance.LastFishEarned = DateTime.Now;
                }
            }
        });
        if (GameManager.Instance)
            gameObject.SetActive(GameManager.Instance.CanEarnFish());
    }
}
```