Mirnes Selimovic

Hyötysovelluksen kehitys pelimoottorin avulla

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Mediatekniikkan koulutusohjelma

Insinöörityö

18.4.2015

Metropolia

Tekijä Otsikko Sivumäärä Aika	Mirnes Selimovic Hyötysovelluksen kehitys pelimoottorin avulla 48 sivua 7.5.2015
Tutkinto	Insinööri (AMK)
Koulutusohjelma	Mediatekniikka
Suuntautumisvaihtoehto	Digitaalinen media
Ohjaajat	Toimitusjohtaja Erkki Heilakka Lehtori Antti Laiho

Insinöörityön tavoitteena oli tarkastella, mitä pelimoottori tarjoaa kehittäjälle, joka haluaisi luoda mobiililaitteelle hyötysovelluksia pelisovelluksien sijaan. Tarkoitus oli siis tutkia ja päätellä, mikä tekee hyvän mobiilisovelluksen ja mitä elementtejä pelimoottori tuo helpottaakseen sovelluksien luomista, testaamista ja päivittämistä. Lisäksi oli tavoitteena käydä läpi perussovellusrakenne ja se, mikä tekee sovelluksesta *mobiili*sovelluksen.

Tutkimusta tehtiin perehtymällä pelimoottorin toimintoihin, kehittämällä suomalaiselle peliyritykselle mobiilisovelluksia ja tutkimalla verkkomateriaaleja ja kirjallisuutta liittyen mobiilisovellusten kehittämiseen. Päätelmiä ja havaintoja tehtiin omakohtaisesti tutkimalla lukuisia suosittuja mobiilisovelluksia ja analysoimalla yrityksessä tehtyjen mobiilisovelluksien kehityksen vaiheita.

Insinöörityössä selvisi, että pelimoottori on hyvä sovellus alkaville kehittäjille tai pienyrityksille sen käyttäjäystävällisyyden ja tehokkuuden takia. Pelimoottori suorittaa vaikeat osat projektista, kuten fysiikan ja grafiikan laskemisen, automaattisesti ilman käyttäjän tarvetta tehdä mitään. Todettiin myös, että pelimoottorin muutamat ongelmat kohdistuvat pääasiallisesti suuryrityksiin, joilla on resursseja luoda oma räätälöity pelimoottori.

Lopputuloksena saatiin mobiilisovellus, joka on tehty pelimoottorilla noin puolessa vuodessa alusta loppuun. Mobiilisovellus julkaistaan toukokuun 2015 loppupuolella markkinoille. Tästä voi tehdä johtopäätöksen, että pelimoottori on hyvä sovellus hyötysovellusten tekemisessä ja sillä saa nopeasti tehtyä julkaistavan tuotteen jopa pienellä tiimillä.

Avainsanat	Mobiilisovellus, hyötysovellus, pelimoottori

Author Title Number of Pages Date	Mirnes Selimovic Development of Non-Game Applications With a Game Engine 48 pages 15 May 2015
Degree	Bachelor of Engineering
Degree Programme	Media Technology
Specialisation option	Digital Media
Instructors	Erkki Heilakka, CEO Antti Laiho, Senior Lecturer

The aim of this thesis was to consider and identify benefits a game engine offers to mobile application developers who wish to develop a non-game application for a mobile device. In other words, the objective was to research and deduce what makes a great mobile application and what elements Unity brings to make the development, testing and updating such an application faster and more user-friendly. In addition, there was an aim to examine the basic architecture of mobile applications and to find out what makes a mobile application *mobile*.

The research was conducted by familiarizing with a game engine, while working on mobile applications under a Finnish video game company, and also researching the internet resources and literature on mobile applications and the game engine. Own observations and conclusions were made based on the application developments at the company, and by analyzing and testing popular mobile applications.

In the end, it was deduced that a game engine is in fact a good application for starting developers or small firms, due to its user-friendliness and powerful capabilities. A game engine processes difficult aspects, like physics and graphics, automatically so that the developer can focus more on other parts of the project.

The end result of the thesis was a non-game application, which was created with a game engine within about 6 months. The application is to be launched at the end of May, 2015. This can be taken as a sign that a game engine provides a very solid and fast way for developers to make applications, even if the developer team is small.

Keywords	mobile application, non-game application, game engine

Sisällys

1 Johdanto

Insinöörityön aiheena on hyötysovellusten suunnittelu ja luonti Unity-pelimoottorilla. Pelimoottoriksi kutsutaan ohjelmistokehystä, joka on luotu pääasiallisesti videopelien tekoon ja sisältää tärkeitä ydinkomponentteja, joilla pelin grafiikka, ääni, logiikka, mahdollinen fysiikka ja koodi hallitaan ja luodaan. Pelimoottoreita ei ole yleisesti ottaen tarkoitettu hyötysovelluksien luomiseen, minkä takia sovelluskehittäjät saattavat kaihtaa pelimoottorin käyttöä omissa projekteissaan. Tässä opinnäytetyössä pyritään selvittämään onko Unity-pelimoottorista hyötysovelluksien tekemiseen vai pitäisikö hyötysovellukset tehdä ilman moottoria.

Insinöörityöraportissa keskitytään ensin neljään eri aihealueeseen: mitä hyötyä ja käyttömahdollisuuksia pelimoottorista on mobiilisovellusten luomisessa, mistä perusrakenteista mobiilisovellukset koostuvat, mikä tekee mobiilisovelluksista erilaisia tietokonesovelluksiin verrattuna ja mitä kehityksen eri vaiheita mobiilisovelluksissa on. Tämän lisäksi pohditaan, mitä suunnitteluun liittyviä asioita ja mitä osa-alueita mobiilisovellusten luominen vaatii.

Tämän jälkeen käydään esimerkkinä läpi Unityllä tehdyn mobiilisovelluksen suunnittelun ja toteutuksen eri vaiheet. Mobiilisovellus on kahden suomalaisyrityksen, Miivies Oy:n ja Futurecoden, yhteinen projekti. Mobiilisovellus on lapsille suunnattu videopostikorttisovellus, jonka suunnittelusta ja luomisesta vastasi Miivies Oy. Sovellus on luotu viiden hengen tiimissä, lukuun ottamatta Erkki Heilakkaa, Miivies Oy:n toimitusjohtajaa. Tiimin kaikki viisi työntekijää ovat opiskelleet Metropolia Ammattikorkeakoulussa. Kolme heistä on ollut 3D-animoinin ja -visualisoinnin opiskelijoita ja minun lisäkseni yksi muu mediatekniikan opiskelija. Kehitysprosessi kesti noin kuusi kuukautta.

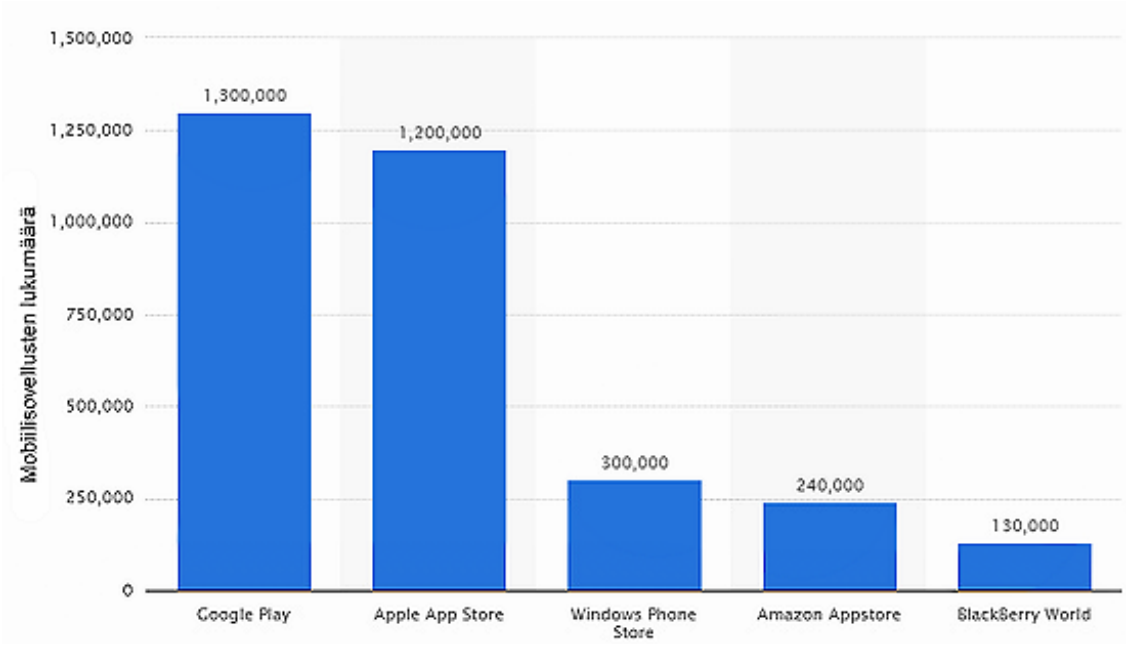
Tavoitteena on siis tarkastella ja todistaa, miten hyötysovelluksien luominen onnistuisi pelimoottorilla, mitkä eri elementit tekevät mobiilisovelluksesta *mobiilin*, mitä tarvitaan onnistuneen mobiilisovelluksen luomiseen, mitä työkaluja pelimoottorissa on ja tekeekö pelimoottori sovelluskehityksestä helpompaa ja joustavampaa.

2 Mobiilisovellus

Nykypäivänä moni kokee itsensä epämukavaksi ilman puhelintaan tai jonkinlaista mobiililaitetta. Puhelin tai taulutietokone on niin arkipäiväinen asia ihmisten elämässä, että on vaikea kuvitella, miten ennen pärjäsi ilman niitä. Mobiililaitteet mahdollistavat ihmissuhteiden ylläpidon, erilaisten jokapäiväisten asioiden tekemisen, kuten sähköpostin tarkastelun, ja lukuisien viihdyttävien asioiden katselun. Mobiililaite on kaikkien oma henkilökohtainen työ- ja viihdekeskus.

Vuonna 2014 puhelimien lukumäärä melkein ylitti ihmispopulaation määrän (noin 7 miljardia) nousemalla 6,8 miljardiin (1). Tilastojen mukaan maailman älypuhelimien lukumäärä tulee olemaan noin 2 miljardia vuoteen 2016 mennessä (2). Älypuhelimia on siis noin 30 % maapallon ihmispopulaatioon nähden, ja määrä kasvaa jatkuvasti. Muut mobiililaitteet, kuten taulutietokoneet, ovat myös nousussa. Ennustetaan, että vuoden 2015 loppuun mennessä taulutietokoneiden käyttäjiä on noin miljardi (3).

Mobiililaitteet ovat välttämätön osa ihmisten elämää ja niiden mukana myös erilaiset mobiilisovellukset. Mobiilisovellukseksi luokitellaan sovellus, joka on tehty mobiililaitteelle, kuten älypuhelimelle tai taulutietokoneelle. Tällaiset sovellukset ovat yleensä pieniä ja itsenäisiä (4). Mobiilisovellusten varsinainen vallankumous alkoi vuonna 2007 Applen iPhonen mukana (5). Nykypäivänä sovelluksia on yhteensä yli kolme miljoonaa, joista suurin osa myydään suuryhtiöiden, kuten Googlen, Applen ja Microsoftin, omissa digitaalisissa sovelluskaupoissa. Googlen omistama Google Play ja Applen omistama App Store myyvät sisällään suurimman osan kaikista mobiilisovelluksista, minkä voi havaita kuvasta 1. Tämä johtuu Googlen ja Applen mobiililaitteiden suuresta suosiosta. (6.)



Kuva 1. Mobiilisovellusten lukumäärä eri sisältöpalveluissa (7).

Älypuhelimet ja taulutietokoneet toivat maailmaan uuden markkina-alueen, jota ei ollut 10 vuotta sitten, ja tämä markkina-alue on kovassa kasvussa. Vuonna 2011 mobiilisovellusten latauksien määrä oli noin 37 miljardia, ja heti seuraavan vuoden lopussa se oli kasvanut 83 miljardiin. Latausten määrä oli siis kaksinkertaistunut yhdessä vuodessa. Tämän kasvun on ennustettu jatkuvan ja ylittävän 200 miljardia ladattua mobiilisovellusta vuoden 2017 loppuun mennessä. Tämä osoittaa, kuinka trendikäs markkina-alue mobiilisovellukset ovat. (8.)

Suuren kasvun takia yritykset ja jopa yksinyrittäjät kokevat halua luoda mobiilisovelluksia, sillä tuotteen luonti on suhteellisen yksinkertaista muihin markkina-alueisiin verrattuna. Mobiilisovelluksia on lukuisia, ja niitä on tehty moneen eri tarkoitukseen. Kaupat, kuten Google Play ja Apple App Store, ovat jakaneet sovellukset kategorioihin, joita on molemmissa noin 20. Niistä suosituimpiin kuuluvat pelit, liikeala, opetus, elämäntapa, viihde ja hyötytavara (9; 10). Vaikka sovelluksia on lukuisia, niiden suosio ei ole tasavertaisesti jakautunutta. Sovelluksen suosioon vaikuttaa sen kysyntä, hyöty, viihdyttävyys, helppokäyttöisyys ja käytettävyys sosiaalisessa mediassa. Sovellusten suosioon vaikuttaa myös mainostus ja yleinen trendi. Teknisesti mobiilisovelluksien luonti ei ole erityisen haastavaa, mutta jos tähtäimessä on suosio ja/tai tuottavuus, on kiinnitettävä erityistä huomiota sen kehittelyyn.

2.1 Hyvä mobiilisovellus

Sovellusta suunniteltaessa pitää muistaa, että käyttäjä kuluttaa hänelle kallisarvoisia resursseja, kuten aikaa, keskittymistä ja ajattelua, aina kun hän käyttää sovellusta. Tämän takia sovellus pitää suunnitella aina käyttäjää ajatellen. Suunnitteluvaiheessa on tärkeää kartoittaa sovelluksen tarkoitus ja kohdeyleisö. On myös hyvä kysyä kysymyksiä kuten ”mitä sovelluksella tehdään?” tai ”miksi juuri tällainen sovellus?”. (11, s. 17, 19.)

Yksi tapa vastata näihin kysymyksiin on miettiä, miksi käyttäjät käyttävät sovelluksia ylipäätään. Käyttäjiä on monia, ja kaikilla yksilöillä on omat syynsä käyttää haluamaansa sovellusta, mutta käyttäjien ajattelumallit voi kuitenkin rajata karkeasti kolmeen eri ajattelumalliin:

- ”Haluan tehdä jotain hyödyllistä.”
- ”Haluan olla yhteydessä muihin.”
- ”Minulla on tylsää.”

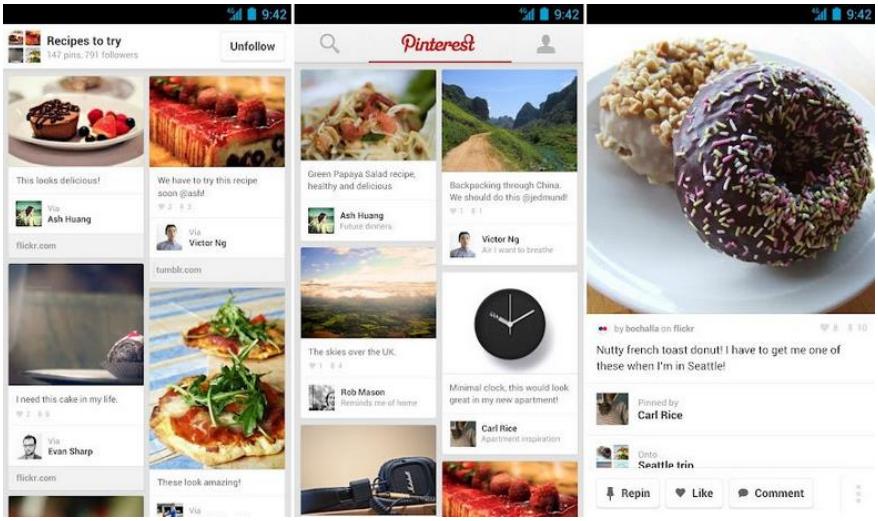
Kun pitää mielessä nämä kolme rajausta, on paljon helpompi suunnitella sovellus, joka voisi olla suosittu ja mahdollisesti jopa tuottava. (11, s. 32.)

Mobiililaitteiden yleistyessä aletaan korvata ennen käytettyjä laitteita, kuten kannettavia tietokoneita ja paperia, puhelimilla ja taulutietokoneilla pienien tehtävien suorittamiseen. Mobiililaitteet ovat pienempiä ja helpompia kantaa kuin kannettavat tietokoneet, ja koska ne ovat melkein poikkeuksetta aina mukana, ne ovat parempia kuin paperiarkit tai muistiot. On muistettava kuitenkin, että mobiililaitteet, varsinkin puhelimet, eivät nykyteknologialla kykene tietokoneiden tasolle, eivätkä ne ole verrattavissa tietokoneiden käyttömahdollisuuksiin ja suoritustehoihin.

Käyttäjät eivät siis pysty vielä tekemään mobiililaitteilla ihan samoja asioita kuin tietokoneella. Sovellukset, jotka on tehty tuottoisuutta varten, pitää suunnitella niin, että ne täyttävät mobiilikäyttäjän toiveet, mutta samalla pysyvät mahdollisimman kevyinä ja helppokäyttöisinä kosketusnäytöllisellä laitteella (11, s. 32). Esimerkki tällaisesta tuotteesta on kuvan 2 Evernote. Se on hyötysovellus joka pyrkii tarjoamaan käyttäjille nopean ja helppokäyttöisen tavan erilaisten muistiinpanojen tekemiseen. (12.)

Kuva 2. Evernote on hyötysovellus muistiinpanoja varten (13).

Tuotteliaisuuden lisäksi mobiililaitteita käytetään vahvasti sosiaalistumisessa. Nykypäivän sosiaalistuminen on melkein välttämätön osa internetkulttuuria. Sisällön jakaminen on suosittua nykyään, ja suuressa osassa sovelluksia on mahdollisuus jakaa jonkinlaista sisältöä muille ihmisille suoraan tai yhteiselle sosiaalisen median sivustolle. Ihmiset pitävät sisällön jakamisesta, ja monet sanovat sisällönjakamisen olevan heille viihdettä, itseilmaisua, sosiaalisien suhteiden ylläpitämistä, itsensä toteuttamista tai aatteen kannatusta (14). Sisältöä voi jakaa esimerkiksi kuvassa 3 näkyvän Pinterest-nimisen sosiaalisen median internetsivustolla.



Kuva 3. Pinterestin Android-sovellus (15).

Tällaisissa sovelluksissa on sisäänrakennettu *jaa*-nappi, mikä tekee jakamisesta helppoa ja nopeaa. *Jaa*-napin kaksi yleisintä kuvake näkyvät kuvan 4 vasemmalla

puolella. *Jaa*-napin alla ovat sisällön jakamismahdollisuudet, jotka sovellus tarjoaa. Kuvassa 4 oikealla puolella on esimerkki jakamismahdollisuuksista.

Kuva 4. Vasemmalla kaksi yleistä *jaa*-napin ikonia (16). Oikealla on *jaa*-valikko (17).

Mobiilisovellus voi tietenkin olla myös pelkkä viihdyttäjä, jonka tarkoitus on tarjota käyttäjälle mukavaa ajanvietettä. Suurin osa voittoa tuottavista mobiilisovelluksista kuuluu tähän kategoriaan. Tämän vuoksi viihdesovellusten, pääasiallisesti pelien, kehittäminen on trendikästä sekä uusissa että kokeneissa yrityksissä. Moni sovellus on oma kokonaisuutensa, joka viihdyttää tarjoamatta minkäänlaista sosiaalista tai tuotteliasta aspektia. Kaikkien sovellusten ei siis tarvitse antaa käyttäjille mahdollisuutta luoda asioita, mutta näiden aspektien lisäys ei tietenkään vie sovellukselta mitään. Kehittäjä tekee tietoisen päätöksen sovelluksen tarjoamista ominaisuuksista ja mitä hän haluaa kuluttajan voivan tehdä sovelluksella. Tällaisiin sovelluksiin kuuluvat muun muassa monet pelit ja musiikkisovellukset. Sovellukset Shazam ja SoundHound, jotka näkyvät kuvassa 5, on tehty musiikintunnistusta varten. Sovellukset kuuntelevat ympärillä soivaa musiikkia ja etsivät tietokannasta kappaleen tiedot. (18; 19.)

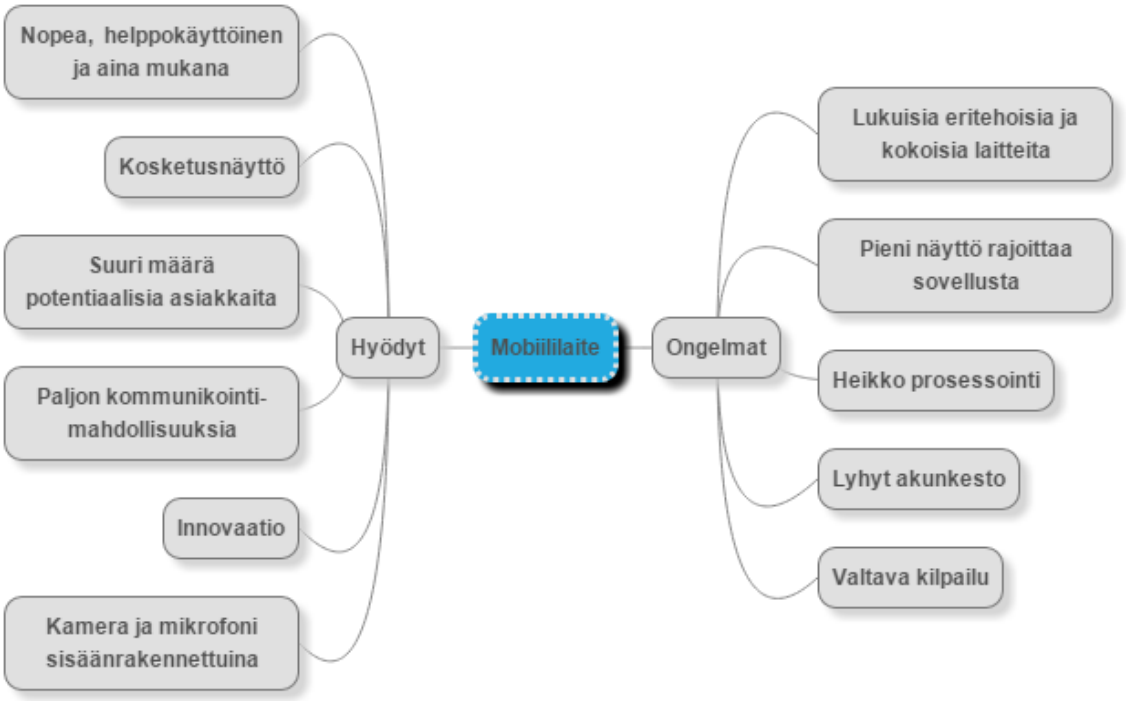
Kuva 5. Musiikintunnistussovellukset Shazam ja SoundHound (20).

Kehittäjä voi itse päättää, haluaako hän luoda sovelluksen, joka on oma kokonaisuutensa, vai haluaako hän sovelluksen olevan yhteydessä sosiaaliseen mediaan tai tarjoavan työkaluja, joilla voi luoda oikeaa sisältöä.

On hyvä muistaa, että mainitut kolme kategoriaa eivät ole toisiaan poissulkevia. Sovellus voi olla peli, joka tarjoaa sisällön luontia ja jakamista, tai sosiaalisen median verkosto, jolla voi tehdä jotain hyödyllistä tai viihdyttävää, tai mikä tahansa muu yhdistelmä. Niin kauan kuin sovellus tarjoaa käyttäjille jotakin, mitä he kaipaavat, uuden tavan nauttia vapaa-ajasta tai jonkin asian tekemisen helpommin kuin ennen, on sovellus hyvä ja on sillä potentiaalia tulla suosituksi.

2.2 Mobiilisovellus tietokonesovelluksen sijaan

On tärkeää tietää myös, miksi kehitettävä sovellus on *mobiili*sovellus. Mitä kaikkea mobiililaite tarjoaa, mitä muut laitteet, kuten mahdollisesti tietokone, eivät tarjoa? Mitä rajoituksia mobiililaite tuo ja miten tehdä sovellus, joka hyötyy mobiililaitteiden hyvistä puolista eikä kärsi laitteiden rajoituksista? Kuvassa 6 on havainnollistettu mobiililaitteiden ongelmat ja hyödyt.

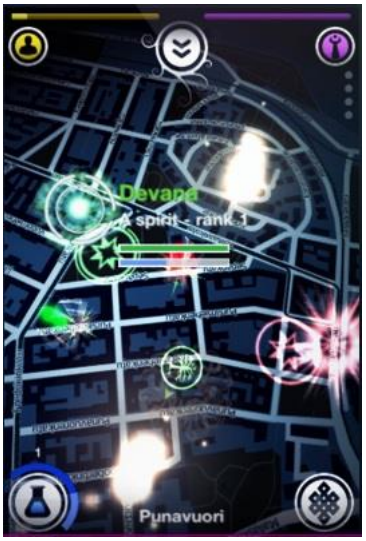


Kuva 6. Mobiililaite tarjoaa kehittäjälle paljon hyvää, mutta se samalla rajoittaa sovelluksen suuruutta.

Mobiililaitteet tarjoavat monia hyötyjä käyttäjälle. Kaikki mobiililaitteet, etenkin älypuhelimet, ovat pieniä, helposti kuljetettavia tietokoneita. Niillä on suuri potentiaali jokapäiväisessä elämässä, sillä ne tarjoavat monipuolisen alustan erilaisille sovelluksille. Suurin osa ihmisistä ei vietä koko päivää sisätiloissa tai kotona, joten on hyvä, että erilaiset hyödylliset sovellukset voivat kulkea helposti mukana. Jos haluaa hakea informaatiota internetistä, kirjoittaa nopeita muistiinpanoja, lähettää kuvia tai katsella videoita, se onnistuu helposti missä vain. Nämä ovat vain muutamia syitä, miksi mobiilisovellusten suosio jatkaa kasvamista ja miksi juuri nyt kannattaa panostaa mobiilisovellusten tekemiseen.

Nykypäivänä melkein kaikki älypuhelimet ovat kosketusnäytöllisiä. Kosketusnäyttöteknologia on kätevä mobiililaitteille, sillä se poistaa erillisten nappien vaatimukset ja tuo paljon joustavuutta sovellusten toimintaan. Kosketusnäytöllä on toinenkin hyvä puoli: se mahdollistaa suuren näytön. Kun ei ole tarvetta erillisille näppäimistöille, voi laitteen ruudun tehdä melkein laitteen kokoiseksi. Laitteissa on myös nykyään lähes poikkeuksetta sisäänrakennettu valokuvauskamera, videokamera, GPS-paikannin ja valo, joka toimii taskulamppuna. Näitä voi hyödyntää vapaasti sovelluksessa, ja ne mahdollistavat asioita, joita on hankala tai kömpelöä tehdä isommilla laitteilla.

Sosiaalisen median tarjoaman sisällön jakamisen tärkeys voi olla hyvä syy harkita jonkinlaisen sisällönluomisominaisuuden kehittämisen hyötysovellukseen (14). Kameran tai kosketusnäytön käyttö, esimerkiksi kuvien tai piirustuksien luomisessa, voi tuoda käyttäjille mukavan tavan luoda sisältöä, jota he haluavat jakaa muille. Mobiililaitteiden GPS-paikannin antaa myös erilaisia mahdollisuuksia sovelluskehityksessä. Suomalaisen yrityksen, Grey Arean, luoma peli Shadow Cities käyttää GPS-paikanninta kekseliäästi fantasiahenkisessä taistelussa. Kuvassa 7 voi nähdä pelissä esiintyvän karttapohjan, joka on oikean maailman kartta. (21.)



Kuva 7. Shadow Cities -fantasiapeli (22)

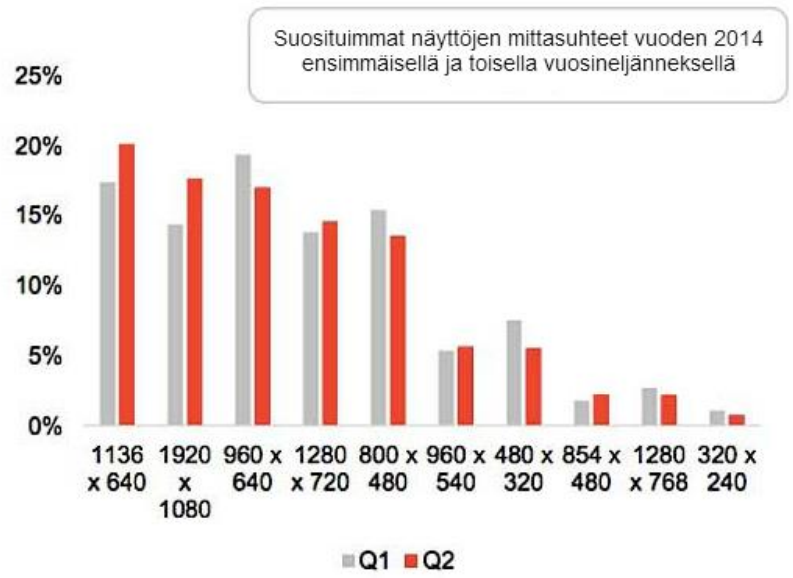
Kaikki edellä kuvatta vaikuttaa positiivisesti mobiililaitteiden suosioon. Mobiililaitteiden suosion kasvaessa mobiilisovellusten markkinat kasvavat myös. Nämä markkinat ovat valtavat, ja kun kaikkien laitteiden mukana tulee mobiilisovelluskauppa valmiiksi asennettuna puhelimeen, on omien sovellusten saaminen potentiaalisille asiakkaille helpompaa kuin minkään muu mediatuotteen.

Mobiilisovelluksen ei välttämättä pidä hyödyntää kaikkia mobiililaitteen tarjoamia mahdollisuuksia, mutta sen tulee ottaa huomioon kaikki laitteen rajoitukset. Puhelimia ja tabletteja on lukuisia eri malleja, ja kaikilla malleilla on eri tekninen suorituskyky. On hyvä ottaa huomioon, että eri laitteilla on eri tietokonelaitteistot, kuten prosessori ja näytönohjain, ja optimoida sovellus viemään mahdollisimman vähän tehoa näiltä komponenteilta.

Laitteen oma suorituskyky vaikuttaa suoraan siihen, kuinka raskaita, graafisesti tai toiminnallisesti vaativia, sovelluksia laite pystyy prosessoimaan. Akun käyttö lisääntyy aina, kun komponenteilta vaaditaan enemmän työtä. Jotta mobiilisovellus saisi mahdollisimman paljon käyttäjiä, sen on hyvä toimia mahdollisimman monella eri laitteella, sillä markkinoilla on kymmeniä eri mobiililaitteita, ja suurin osa eroaa toisistaan. Sovelluksen saa kevyemmäksi rajoittamalla laitteelta vaadittavaa laskentatehoa ja muistin käyttöä. Jos sovelluksessa on esimerkiksi paljon 3D-grafiikkaa, voi sovellus vaatia laitteen näytönohjaimelta liikaa. Näytönohjain on komponentti, jonka tehtävä on piirtää näytölle kaikki esitettävä grafiikka, ja koska 3D-grafiikka vaatii runsaasti enemmän laskentatehoa kuin tavallinen 2D-kuva, voi 3D-grafiikka aiheuttaa hidasteluja ja tökkimistä laitteen näytöllä.

Sovelluksen kehittämiseen vaikuttavat myös mobiililaitteiden käyttämät käyttöjärjestelmät. Eri käyttöjärjestelmät ovat yhteensopivia eri tiedostoformaattien kanssa ja monesti vaativat jotain räätälöityä ohjelmointikoodia. Suosituimmat puhelinkäyttöjärjestelmät ovat Applen iOS ja Googlen Android. Sovellus, joka on tehty yhdelle, ei suoraan toimi toisella, vaan se pitää erikseen muuttaa yhteensopivaksi jokaiseen eri käyttöjärjestelmään. Myös käyttöjärjestelmien eri versioilla saattaa olla eri yhteensopivuudet; sovellus joka on tehty uudelle iOS- tai Android-versiolle, ei mahdollisesti toimi vanhalla. Kaikilla käyttäjillä ei ole aina uusimpia laitteita tai käyttöjärjestelmiä, joten on hyvä optimoida sovellus paria vanhempaa käyttöjärjestelmää silmällä pitäen.

Mobiililaitteiden näytön koko on myös rajoittava tekijä. Vaikka kosketusnäyttöteknologia mahdollistaa suuren näytön laitteen kokoon nähden, yleisesti ottaen mobiililaitteiden näyttö on suhteellisen pieni. Mobiililaitteita on lukuisia eri kokoja, ja kaikkien näyttöjen resoluutiot tai kuvasuhteet eivät ole samoja. Kuva 8 esittää vuoden 2014 suosituimmat näyttöjen mittasuhteet.



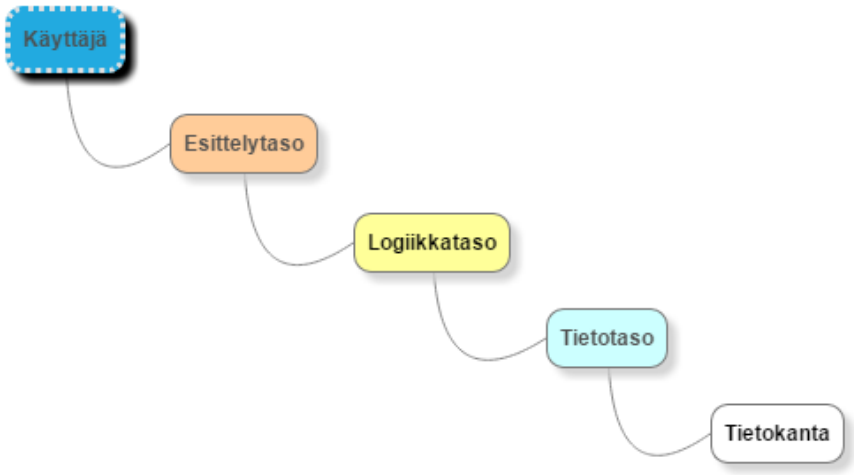
Kuva 8. Suosittuja mobiililaitteiden näyttöjen mittasuhteita vuodelta 2014 (23).

Sovelluksen pitäisi ideaalisesti sopia kaikkiin näyttökokoihin, jotta graafinen käyttöliittymä ja muut grafiikkaelementit eivät kärsisi. Väärä kuvasuhde tai resoluutio saattaa aiheuttaa vääristyneitä kuvia tai jopa poikkeamia grafiikan sijainnissa.

Kaikista näistä varteenotettavista asioista huolimatta mobiililaitteiden markkinat ovat erittäin menestyvät. Mikään edellä mainituista ongelmista ei ole niin suuri, ettei sovellusta voisi tehdä hyväksi ja mieluisaksi käyttää. Tämän takia mobiilimarkkinat ovatkin valtavat. Tämä valtavuus tuo mukanaan myös suurta kilpailua. Kaikki mobiilisovellusten kehittäjät kilpailevat suosiosta ja rahasta, ja kun sovelluksia on yli kolme miljoonaa, voi olla vaikeaa saada julkisuutta tai mainetta. Monesti valokeilassa olevat sovellukset ja kehittäjät ovat joko markkinoineet itseään vahvasti, saaneet apua jo valmiiksi tuottavilta ja menestyviltä yrityksiltä tai luoneet erittäin hyvän ja omaperäisen tuotteen. Sovellusta tehdessä on hyvä suunnitella jonkinlainen strategia julkisuuden saamiseksi. Jos kukaan ei tiedä tuotteen olemassaolosta, sille on vaikea saada käyttäjiä.

2.3 Sovelluksen perusrakenne

Mobiilisovelluksen toiminta voidaan jakaa kolmeen eri tasoon: esittely, logiikka ja tieto. Nämä tasot sijoittuvat käyttäjän ja tietokannan väliin, kuten kuvassa 9 on esitetty. Riippuen siitä, onko sovelluksen tarkoitus olla raskas tai kevyt asiakasohjelma, eri tasot tulee sijoittaa joko erilliselle palvelimelle tai suoraan paikallisesti mobiililaitteeseen (24, s. 8).

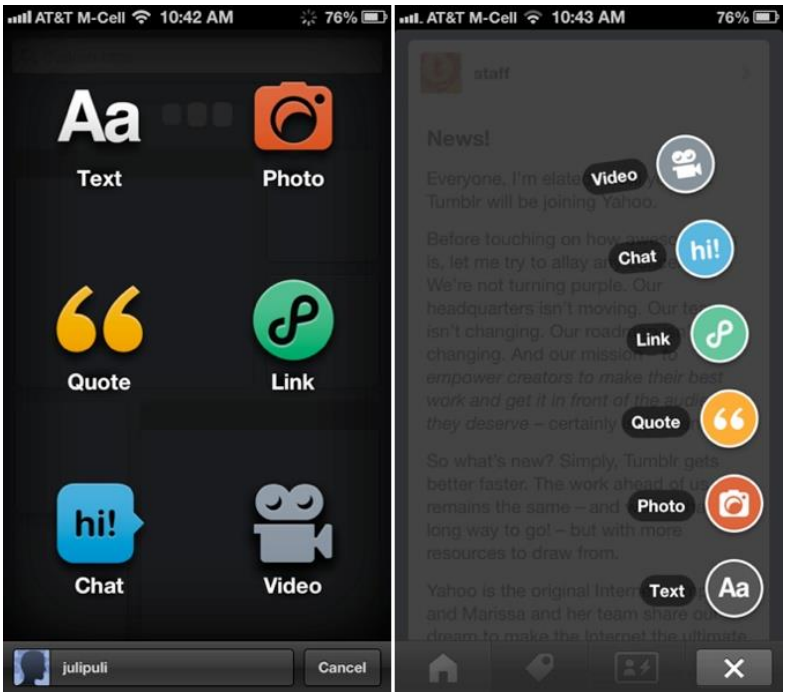


Kuva 9. Mobiilisovelluksen infrastruktuuri.

Esittelytaso

Sovelluksen esittelytaso koostuu kahdesta eri osasta: käyttöliittymästä ja käyttäjän toiminnan prosessoinnista. Yhdessä nämä kaksi osaa tarjoavat tavan, millä käyttäjä voi vuorovaikuttaa sovelluksen kanssa niin, että sovellus tekee käyttäjän haluamat toiminnot. Käyttöliittymäosuus esittää käyttäjälle tietoa ja grafiikkaa, jota käyttäjä voi hyödyntää sovelluksen käytössä, esimerkiksi nappi, jota käyttäjä voi painaa. Yksinkertaisuudessaan tämä tarkoittaa, että kaikki, mitä käyttäjä näkee ruudulla, kuten valikot, kuvat ja tekstit, on tämän osan esittämää. Kaikki vuorovaikutus, mikä käyttäjälle on suotu, on tehty tässä osassa. Ohjelmointikoodi, joka hallitsee napin painamisen tunnistamisen, on tämän osan sisällä. Sovelluksen toinen osa puolestaan kerää syötteet prosessoitavaksi ja kommunikoi logiikkatason kanssa. (24, s. 41.)

Vuonna 2013 sosiaalisen median verkkosivusto Tumblrin mobiilisovellus sai käyttöliittymäpäivityksen. Kuvassa 10 näkyy vasemmalla vanha ja oikealla uusi ulkoasu. Kaikki grafiikka ja teksti on esitetty sovelluksen esittelytasolla.



Kuva 10. Tumblrin vanha ja uusi ulkoasu (25).

Käyttöliittymä esittää käyttäjälle napin laitteen ruudulla, ja jos käyttäjä painaa nappia, syöteinformaatio lähetetään käsiteltäväksi prosessointiosaan. Toiminto voi olla siis mikä vain käyttäjän ja sovelluksen vuorovaikutus, kuten valikon avaaminen, tiedoston latauksen kutsu, kuvan suurennus tai jopa jonkin ääniraidan toisto. Menetelmä on aina sama: käyttäjä vuoro vaikuttaa esittelytason kanssa, ja esittelytaso tarjoaa syötetiedon logiikkatasolle. (24, s. 41.)

Logiikkataso

Logiikkataso on sovelluksen toinen taso, ja sen tehtävä on hallita informaatiota edellisen ja seuraavan tason välillä. Logiikkataso koordinoi sovelluksen toimintaa, prosessoi komentoja ja tekee loogisia päätöksiä. Taso koostuu neljästä osasta, joilla hallitaan kaikki sovelluksen tekemät toiminnot, jotka eivät liity käyttöliittymään. Ensimmäinen osa tehdään luomaan yksinkertaistettu rajapinta ulkoisten koodikirjastojen käyttämistä varten. Tämä osa ei ole aina tarpeellinen, mutta jos ulkoisia kirjastoja on paljon tai niiden hallinta on vaikeaa, osa yleensä luodaan. (24, s. 51.)

Toinen osuus logiikkatasoa ovat logiikkakomponentit. Näiden komponenttien tehtävä on tarjota erilaisia logiikkapalveluita, kuten sääntöjen prosessointi. Tällä voidaan toteuttaa

rakenneosa, joka mahdollistaa monen operaation tapahtumisen yhdessä transaktiokomponentissa. Jos vaikka sovellus kommunikoi ulkoisen palvelun kanssa ja ulkoinen palvelu tekee monta toimintoa, voidaan luoda transaktiokomponentti, joka hallitsee kaikki nuo toiminnot ja palauttaa ne yhtenä kokonaisuutena sovellukselle. (24, s. 51.)

Kolmanneksi tarvitaan logiikkaentiteettejä. Nämä entiteetit ovat eri tasojen välillä kulkevia osia. Osaa voi ajatella sovelluksen sisäiseksi tiedonvälittäjäksi. Kun esittelytasolla tapahtuu jokin vuorovaikutus, tämä osa hakee informaation ja tuo sen prosessoitavaksi logiikkatasolle. Jos logiikkatasolta pitää antaa jotain tietoa tietotasolle vaikka tallennettavaksi, tämä osa vie tiedon. Sovellus pitää siis suunnitella niin, että kaikki toiminnot ovat itsenäisiä ja niiden välinen kommunikaatio tapahtuu logiikkaentiteeteillä. Tämä mahdollistaa helpon ja yksinkertaisen ylläpidon ja koodinhallinnan. (24, s. 52.)

Viimeinen osa hallitsee pitkien prosessien työnkulun. Tämä osa varmistaa, että kaikki prosessit, jotka tarvitsevat organisointia, suoritetaan oikein. Tähän kuuluvat siis kaikki sovelluksen toiminnot, jotka ovat kauan kestäviä ja joissa on mahdollisesti monta vaihetta, kuten esimerkiksi tilauksien tekeminen. Jos käyttäjältä vaaditaan monta toimintoa, jotka ovat kytköksissä toisiinsa ja niiden on suoriuduttava oikeassa järjestyksessä, tämä osa hallitsee toimintojen kulun ja varmistaa, että järjestys ei mene sekaisin. (24, s. 52.)

Tietotaso

Sovelluksen alin taso on niin sanottu tietotaso. Tietotasolla tapahtuu kaikki sovelluksen tiedon tallennus, lataus ja säilytys. Tieto pidetään joko erillisellä palvelimella tai laitteen omassa muistissa. Kaikki tieto esittelytasolta tai logiikkatasolta, joka halutaan tallentaa, annetaan tietotasolle. Taso jakautuu kolmeen erilliseen osaan: osa, joka hallitsee tiedon yhteyslogiikkaa, toinen osa, joka koostuu tieto hyödykkeistä, ja viimeinen osa, joka hallitsee mahdollisen kommunikaation ulkoisten palvelujen kanssa. (24, s. 62.)

Ensimmäinen osa toimii tiedonsiirtologiikan hallitsijana ja supistajana. Tämä osa rakennetaan, jotta sovelluksen tiedon hakeminen paikallisesta tai ulkoisesta tietokannasta olisi helpompaa käsitellä ja ylläpitää. Tason suunnittelu ja toteuttaminen on syytä tehdä hyvin, sillä huonosti tehty tietokommunikaatio tietokannan ja sovelluksen välillä voi aiheuttaa vaikeuksia, jos sovellukseen tehdään myöhemmin muutoksia. Jos logiikka on puoles-

taan tehty hyvin ja se on irrallaan sovelluksen muista tasoista, eivät muiden tasojen muutokset vaikuta tietologiikkaan. Jos sovellus muuttuu paljon, on logiikka helppo muuttaa perässä. Kaikki osat on siis hyvä rakentaa erillisinä kokonaisuuksina. (24, s. 62.)

Jos haluaa tehdä sovelluksen, joka käyttää pelkästään ulkoista tietokantaa tai sovellusta, jolla on suuri määrä tallennettavaa tietoa, kannattaa sovellus luoda niin, että sovellus on vain käyttäjäpuolen sovellus. Tällä tarkoitetaan sitä, että sovelluksella ei olisi suoraa yhteyttä tietokantaan, vaan erillinen välikäsi luotaisiin kommunikoimaan mobiilisovelluksen ja tietokannan, välillä. Suora yhteys tietokantaan, jossa on mahdollisesti monen käyttäjän tiedot tallella, kuten esimerkiksi massiivinen monen pelaajan verkkopeli, olisi valtava tietoturvariski. (26.)

Tietotason toinen osa auttaa tiedon manipuloinnissa, käsittelyssä ja muuttamisessa. Siinä missä edellinen osa hallitsi tiedon lataamista, tallentamista ja yleistä kommunikointia tasojen välillä, tämä osa tietotasoa on suunniteltu toimimaan vain tietotason sisällä. Osa koostuu erikoiskirjastoista, jotka on luotu nimenomaan tiedon käsittelyä varten. (24, s. 62.)

Tietotason viimeinen osa luodaan, jotta kommunikointi ulkopuolisten palveluiden, jos sellaisia sovelluksella on, tapahtuisi helpommin. Tämä osa, jota kutsutaan palveluagentiksi, on hyvä luoda, jotta oma sovellus pysyy erillisenä muista palveluista.(24, s. 62.)

3 Sovelluksen tekeminen pelimoottorin avulla

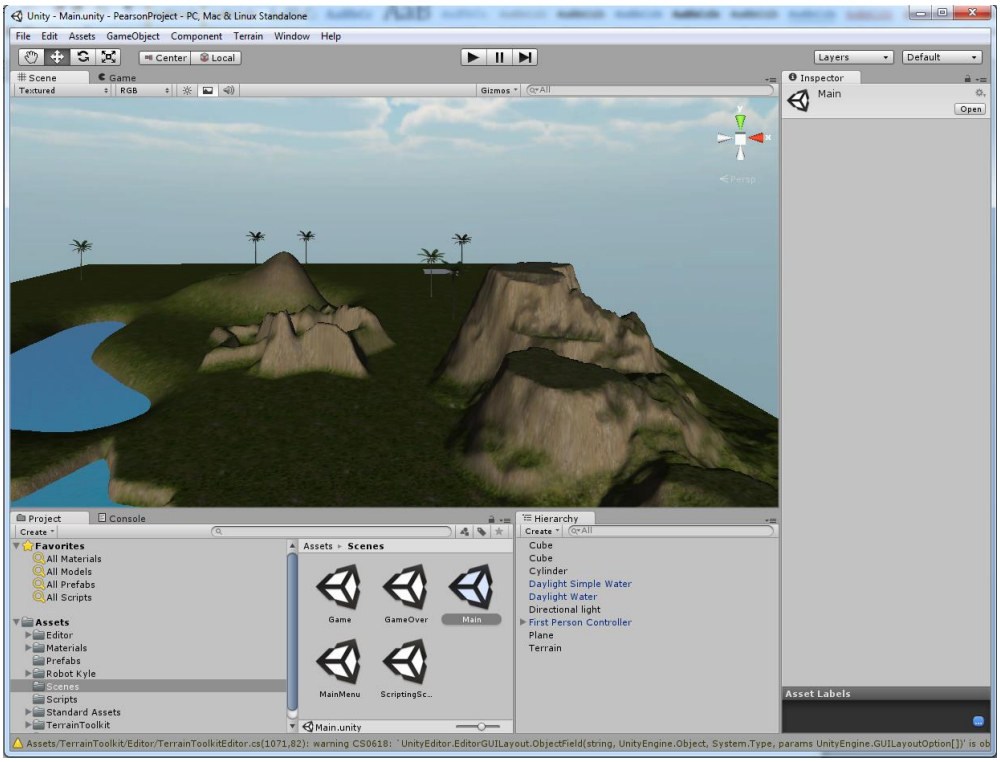
Pelimoottorit on sovellus, joka on tehty ja optimoitu puhtaasti pelejä ajatellen. Sen lukuisat työkalut ovat optimaalisia interaktiivisen ja viihdyttävän sovelluksen tekemiseen, ja sen suurin asiakaskunta rakentuu uusista pelinkehittäjistä tai pienistä yrityksistä. Pelimoottorin tarjoamat työkalut mahdollistavat helpon ja nopean pelien luomisen niille, joilla ei olisi resursseja tai taitoa luoda peliä kokonaan tyhjästä. Mutta vaikka pelimoottori on luotu pelejä varten, on olemassa kuitenkin lukuisia sovelluksia joita ei luokitella peleiksi, jotka on tehty pelimoottorilla. Unity3D-verkkosivun *Showcase*-osiossa on jopa oma kategoria tällaisille sovelluksille. (27.)

Tämä johtuu siitä, että sovellukset, joita ei voi luokitella peleiksi, voivat kuitenkin olla interaktiivisia ja viihdyttäviä. Tällaisten sovellusten luomiseen kannattaa käyttää pelimoottoria, sillä pelimoottorin valmiit komponentit ovat ammattilaisten tekemiä ja hoitavat vaikeat ohjelmointiasiat kehittäjän joutumatta menemään syvällisemmin niiden ohjelmointiin.

3.1 Pelimoottorin tarjoamat työkalut

Yksi pelimoottoreiden hyvistä puolista on, että se periaatteessa tekee puolet sovelluskehittäjän työstä. Pelimoottori tarjoaa laajan kokonaisuuden erilaisia työkaluja, joilla sovelluksia voi luoda. Yksinkertainen sovellus onnistuu helposti millä tahansa sovelluskehitysympäristöllä, mutta kun tavoitteena on monimutkainen ja graafisesti vaativa sovellus, sovelluskehittäjän pitää olla melkoisen kokenut.

Unity on suunniteltu käyttäjäystävälliseksi ja mahdollisimman helpoksi käyttää. Pelimoottori on visuaalinen ja mahdollistaa helposti 2D- ja 3D-esineiden liikuttamisen ja suorittamiseen. Unityn *Inspector*-ikkuna tarjoaa mahdollisuuden vaikuttaa ohjelmointikoodiin suoraan sovellusta testatessa. Tämä nopeuttaa esimerkiksi parametrien muuttamista ja testailua (28). Unity-pelimoottorin kuvaruutu on jaettu karkeasti kolmeen eri osaan: sovellusikkuna, joka näyttää kaikki 2D- ja 3D-esineet, projektinhallintaikkuna, joka mahdollistaa helpon tiedostojen hallinnan, ja *Inspector*-ikkuna. Kuvassa 11 on esitetty Unityn kehitysympäristö ja yksi sen ulkoasuista.

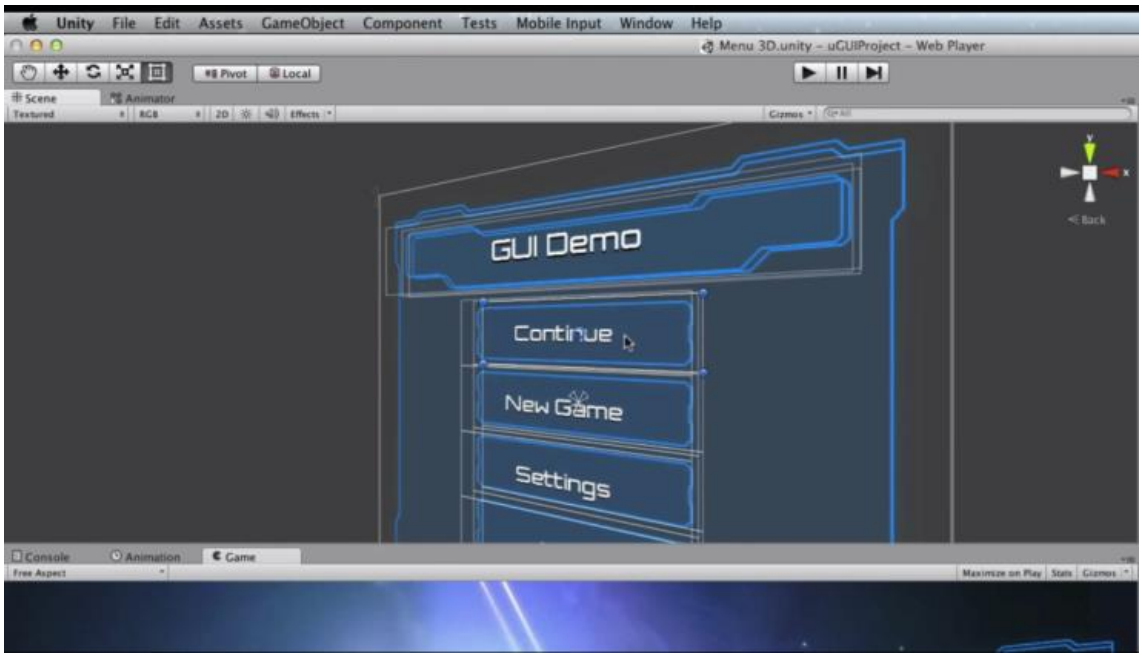


Kuva 11. Unity-pelimoottorin työasema (29).

Pelimoottorilla on lukuisia eri käyttäjä-syöteyhteensopivuuksia, sillä eri alustat käyttävät eri syötelaitteita. Jotkin pelit ja sovellukset toimivat hiirellä ja näppäimistöllä, toiset lukuisilla eri peliohjaimilla ja mobiilipelit kosketusnäytöillä. Tämän laajan vaativuuden takia Unity-pelimoottorissa on helppoa tehdä käyttöliittymä, joka toimii melkein millä vain syötemuodolla. (30.)

Unity tukee ulkoisia koodikirjastoja ja jopa rohkaisee niiden tekemistä. Unityllä on kauppa nimeltä Asset Store, jossa ihmiset voivat myydä omia luomuksiaan, jotka kuka tahansa voi ostaa ja käyttää omissa projekteissa. Kaupassa on kategoria ohjelmointikoodeille ja Unity-pelimoottorijatkeille. Täältä kehittäjät voivat löytää lukuisia kirjastoja, joita muut ovat tehneet. (31.)

Unity 4.6 -version mukana tullut käyttöliittymätyökalu mahdollistaa graafisien käyttöliittymäelementtien asettelun ja luomisen helposti ja intuitiivisesti 3D-tilassa. Kuten kuva 12 esittää, sillä voi luoda kiinnostavia ja omaperäisiä valikkoja, jotka poikkeavat tavallisista 2D-valikoista. (32.)



Kuva 12. Käyttöliittymän rakentaminen 3D-tilassa (33).

Nämä työkalut on tehty käyttäjäystävällisyyttä ajatellen. Kaikki peruselementit, jotka kuuluvat sovelluksen käyttöliittymään, kuten napit, valikot, kuvat ja animaatiot, ovat helposti luotavissa ilman erillistä ohjelmointia. Kaikki elementit on myös ennalta ohjelmoitu toimimaan kosketusnäytöllä. Nappien painaminen ja kuvien ja valikkojen vieritys onnistuu saman tien, kun elementti on asetettu kohdalleen ja tarvittavat toimintoparametrit on päätetty.

3.2 Grafiikka, fysiikka ja ääni

Yksi huomattavimmista eroista pelimoottoreiden ja muiden ohjelmistokehitysympäristöjen, kuten Windowsin .NET tai Applen iOS:n Cocoa Touchin välillä on se, että pelimoottorit panostavat enemmän grafiikan ja mahdollisen fysiikan laskentaan ja äänen manipulointiin 3D-tilassa kuin esimerkiksi erilaisten pienoisohjelmien luomiseen.

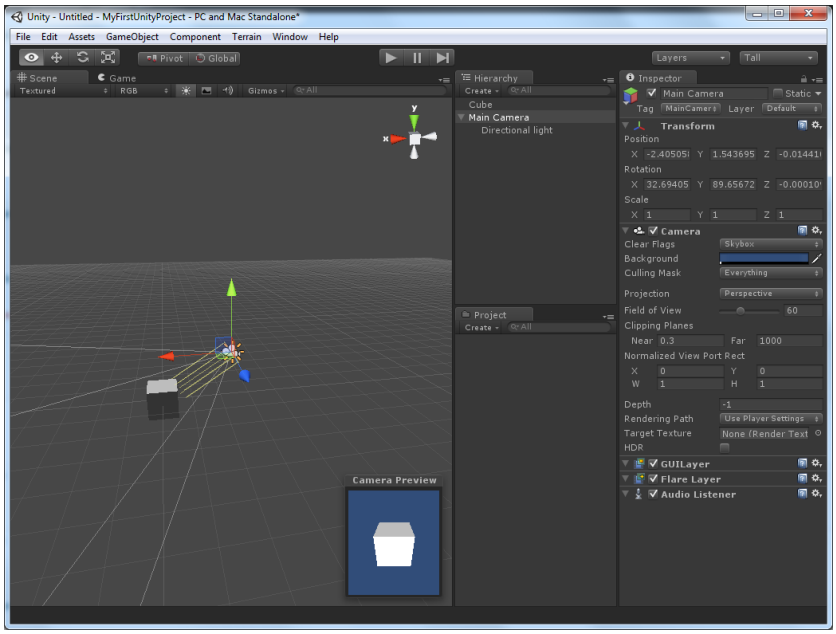
Grafiikan laskenta sekä 2D- ja 3D-grafiikka on tehty kehittäjälle valmiiksi Unityn omalla grafiikanpiirtomoottorilla. Unity tarjoaa valmiita valoja ja kameroita, jotka voi upottaa 3D-tilaan ja suunnata haluttuun kohtaan. Oletusarvoisesti Unity tarjoaa kameran, jonka näkökenttää voi muuttaa, sekä kameran kuvasuodattimen. Kehittäjä voi siis halutessaan päättää, mitä asioita kamera kuvaa ja mitä ei; kuvatut asiat näkyvät kuvaruudussa, kun

taas suodatetut asiat jätetään pois. Unityssä voi myös oletetusti luoda neljä eri valaistusvaihtoehtoa:

- suuntavalo
- valokeila
- pistevalo
- aluevalo (34).

Suuntavalo osuu 3D-tilassa joka kohtaan samasta suunnasta. Suuntavalolla voidaan tehdä esimerkiksi auringonvalo. Valokeila on valo, joka voidaan kohdistaa oikean valokeilan lailla haluttuun pisteeseen. Valokeilaa voidaan käyttää esimerkiksi katulamppujen tekemiseen. Pistevalo on valaistusmalli, joka valaisee halutun pisteen ympärille pallomaisesti. Tällä valolla voidaan tehdä esimerkiksi hehkulamppu. Viimeinen valo on aluevalo, ja sillä voidaan valaista valittu 3D- tai 2D-pinta halutusta suunnasta. Kameroissa ja valoissa on myös monta eri asetusta, kuten kameran leikkausetäisyys ja valon intensiteetti. (34; 35.)

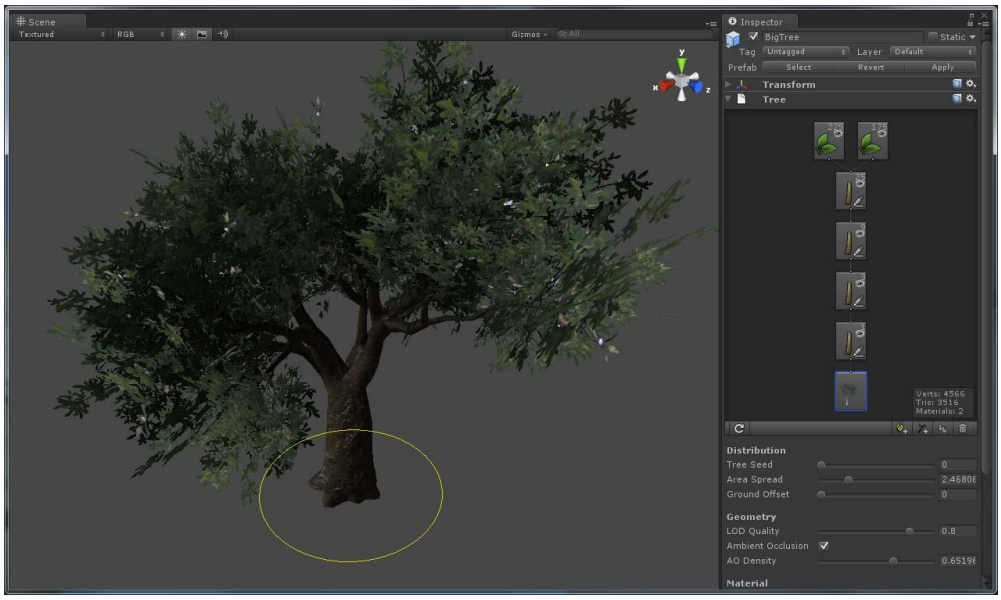
Kuvassa 13 näkyy tyhjään 3D-tilaan laitettu kamera, joka kuvaa harmaata kuutiota. Kuvan alalaidassa voi nähdä pienen neliön, joka näyttää, mitä kamera näkee. Tilaan on myös laitettu suuntavalo valaisemaan kuutiota viistosta. Kaikki esineet ovat käännettävissä ja siirrettävissä, ja niiden vaikutuksen 3D-tilaan näkee saman tien. Kameran ja valon käyttäytymistä voi muokata muuttamalla *Inspector*-ikkunan parametreja.



Kuva 13. 3D-tila, jossa on kamera, suuntavalo ja kuutio (36).

Unity tukee myös Shadereita eli varjostimia. Shadereilla voidaan luoda monimutkaisia ja hienoja pintoja eri 3D-esineille. Pelimoottorissa voi myös laittaa kaikille esineille materiaaleja ja tekstuureja suoraan vetämällä ne esineeseen (37; 38). Tämä tekee grafiikan luomisesta nopeaa ja intuitiivista. Tarjolla on myös kokonainen kasviston luomistyökalu, jolla voi luoda omia puita tai pensaita. Kaikki nämä mahdollistavat vaativankin grafiikan tekemisen ilman kehittäjän omaa tarvetta ohjelmoida mitään erillisiä grafiikkaa.

Unityssä on myös sisäinen puidenluontikomponentti, joka on esitetty kuvassa 14 (39). Sillä voi luoda erilaisia puita tarvitsematta käyttää erillistä 3D-mallinnusohjelmaa. Komponentti voi siis korvata 3D-puiden mallintamisen kokonaan, tai se voi tarjota nopean tavan tehdä tilapäisiä puita testaamista varten.



Kuva 14. Unityn puiden luonti (40).

Pelimoottori tarjoaa myös 2D- ja 3D-fysiikkamoottorin. Niillä voi luoda törmäilyä ja painovoimaa. Fysiikka on hyödyllinen työkalu peleissä, kuten esimerkiksi Rovion Angry Birdsissä, jossa lintuja heitetään erilaisiin rakennelmiin, jotta ne sortuisivat. Angry Birdsin linnut ja tasojen rakennelmat tottelevat 2D-fysiikkamoottoria nimeltä Box2D (41). Ilman tätä osaa ei pelissä olisi sen tärkeintä ominaisuutta. Fysiikkaa voi myös käyttää sovelluksissa, jotka eivät ole pelejä, esimerkiksi erilaisissa opetus sovelluksissa, joissa käyttäjälle opetetaan fysiikan vuorovaikutusta.

Pelimoottori mahdollistaa myös paikallisen äänen 3D-tilassa. Unity laskee äänen ja kameran välisen eron ja jäljittää oikean äänen käyttäytymistä kuulijaan nähden. Tämä mahdollistaa helpon äänen manipuloinnin sovelluksessa.

Sovellus voi olla kokonaan 3D-tilassa tehty ja sisältää kameran pyörittelyä, tavaroiden ja valikkojen liikutusta sekä paikallista ääntä. Kaikki, mikä on tavanomaista peleissä, on mahdollista luoda sovellukseen.

Kolmiulotteisen tilan käyttö mahdollistaa sovelluksen viemisen pois perinteisestä kaksiulotteisesta sovellusmallista ja sen viemiseen 3D-tilaan. Tämä voi olla immersioivaa ja visuaalisesti mukavaa käyttäjälle. Tätä tekniikkaa on käytetty muun muassa Decktools

Inspire-hyötysovelluksessa. Kuvassa 15 näkyvä Unityllä tehty terassinsuunnitteluhyötysovellus mahdollistaa visuaalisesti miellyttävän kehittämiskokemuksen tietokoneella tai mobiililaitteella. (42.)

Kuva 15. Decktools Inspire -hyötysovellus (42).

3.3 Monet eri alustat

Unityn tukemiin alustoihin kuuluvat muun muassa Windows, Mac OS, iOS, Android ja jopa suosituimmat konsolit; kehittäjä voi siis helposti tehdä sovelluksen monelle eri alustalle, kuten kuvasta 16 näkee. Tämä joustavuus tekee pelimoottorista mainion työkalun kehittäjälle, joka haluaa tarjota sovelluksensa mahdollisimman monelle käyttäjälle. Unity mahdollistaa sen, että kun sovellus on valmis, sen voi rakentaa mille vain alustalle. Eri alustat saattavat vaatia joitakin erityisiä toimenpiteitä tai ohjelmointimenetelmiä, mutta yleisesti ottaen alustalle rakentaminen tai alustan vaihtaminen on yksinkertaista. Unityn valikossa on vaihtoehtoja, mille alustalle haluaa sovelluksen rakentaa. (43.)

Kuva 16. Unityn tukemat alustat (43).

3.4 Tiedon tallennus

Unityssä on valmiiksi luotu tallennus- ja latausvaihtoehtoja. Oman sovelluksen sisäisiä tiedoston tallennuksia ja latauksia on helppoa tehdä vähäisellä ohjelmoinnilla. Unityn yksinkertaisin tiedostontallennusmenetelmä on *PlayerPrefs*, johon voi tallentaa yksinkertaista ja ei suuressa prioriteetissa olevaa tietoa (44). Siihen voi esimerkiksi tallentaa käyttäjän valikon asetukset, kuten mahdolliset värimuutokset tai tekstikoot eli tietoa, jonka katoaminen ei aiheuttaisi suuria vaikeuksia käyttäjälle ja jonka voisi tarvittaessa korvata nopeasti.

Unityllä voi myös sarjallistaa sovellustietoa eri tiedostomuotoihin (45). Näin pystyy tekemään esimerkiksi sovelluksia, jotka antavat käyttäjän tallentaa tekemiään luomuksia, kuten tekstiä tekstinkirjoitussovelluksessa, erilliseen tiedostoon. Tiedoston voi tallentaa laitteen muistiin tai erikseen ulkoiselle palvelimelle.

3.5 Pelimoottorin ongelmat

Pelimoottorit eivät tietenkään ole vailla ongelmia. Unityn ongelmat kohdistuvat pääasiallisesti käyttäjille, jotka ovat kokeneita ja haluavat enemmän hallintaa omaan työnkulkuun. Unityn merkittävimmät haittapuolet ovat

- tuntemattoman koodin suuri määrä
- lopullisten tiedostojen mahdollisesti suuri koko

- maksullisuus.

Pelimoottorit ovat hyvä valinta aloitteleville mobiilisovellusten tai pelien kehittäjille. Jos kehittäjä toimii yksin tai pienessä yrityksessä, on valmis pelimoottori ideaali. Isossa yrityksessä, jossa on taitoa ja työvoimaa, voi olla parempi luoda oma yrityksensisäinen pelimoottori, joka on räätälöity juuri yrityksen tarpeiden mukaisesti. Pelimoottorien käyttäjät on aina rajattu pelimoottorin sallimiin toimintoihin ja pelimoottorin mukana tuleviin työkaluihin. Koska pelimoottori on valmiiksi luotu ympäristö, jonka syvimpiin ohjelmakoodeihin ei pääse käsiksi, ei pelimoottorilla voi tehdä ihan kaikkea.

Pelimoottorin tehtävä on olla työkalu nopean ja helpon sovelluskehittämisen apuna. Jos käyttäjä ei ole kokenut tai hänellä ei ole aikaa tehdä kaikkea alusta asti, pelimoottorin tarjoama apu varmistaa, että hänellä on kuitenkin mahdollisuus luoda hyviä sovelluksia.

Pelimoottoreiden tehtävä on olla mahdollisimman monipuolisia, mutta jos monipuolisuutta ei kaipaa, käyttäjä saa paljon turhaa ohjelmistoa. Tämä voi vaikuttaa lopullisen tiedoston kokoon. Kokeneiden kehittäjien, jotka haluavat optimoida sovelluksensa vain omien tarpeiden mukaisesti, kannattaa luoda sovellus puhtaalta pöydältä tai tehdä oma pelimoottori. Esimerkiksi japanilainen peliyritys Square Enix on luonut ja käyttää yrityksen omiin tarpeisiin räätälöityä Crystal Tools -nimistä pelimoottoria. Kuvassa 17 näkyy pelimoottorin logo. Pelimoottorilla on tehty tähän asti viisi peliä, joihin kuuluvat muun muassa Final Fantasy XII, Dragon Quest X ja Lightning Returns: Final Fantasy XIII (46).

Kuva 17. Square Enixin yrityksen sisäinen pelimoottori Crystal Tools (46).

Viimeinen huomattava ongelma pelimoottoreissa on niiden maksullisuus. Pelimoottorit, jotka on tehty kuluttajien käyttöön, ovat monesti maksullisia, jos niillä haluaa tehdä tuot-

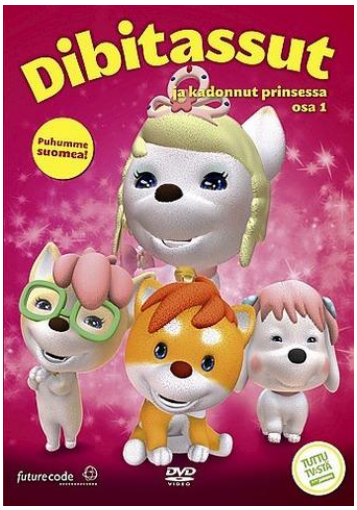
tavia tuotteita tai käyttää pelimoottorin kaikkia ominaisuuksia. Unity tarjoaa käyttäjille ilmaisen version, mutta ilmaisversiossa ei ole kaikkia pelimoottorin tarjoamia ominaisuuksia, kuten veden heijastus ja aluevalo. Jos nämä ominaisuudet halutaan, on ostettava Unityn maksullinen sovellus. (47.)

Maksullisuus ei tietenkään ole välttämättä huono puoli. Maksullisuus takaa yleensä sen, että pelimoottoria parannetaan ja ylläpidetään tulevaisuudessa. Tämä tuo käyttäjälle turvallisuuden tunnetta ja varmuutta, että maksullisella pelimoottorilla voi tehdä pelejä ja niitä voi myös jatkossa päivittää ja ylläpitää samalla pelimoottorilla.

4 Videopostikorttisovellus

Insinöörityössä tarkastellaan esimerkkinä Miivies Oy:ssä työstettyä mobiilisovellusta nimeltä Dibitales Tervehdys. Sovelluksen työstö aloitettiin ennen opinnäytetyön tekemistä, joten se ei suoranaisesti seuraa luvussa 2.3 mainittuja sovelluksen infrastruktuureja.

Projektin toteutukseen kuului alustava suunnittelu sekä 3D-mallien mallinnus ja sovelluksen toteutus Unitylla Android-käyttöjärjestelmälle. Tarkoitus oli saada noin kuuden kuukauden ajassa valmis sovellus, joka menisi myyntiin pääasiallisesti Aasian markkinoille. Alkuperäinen tavoite oli luoda yksinkertainen ja kevyt sovellus, jolla voisi lähettää 3D-videopostikortteja. Sovelluksen tärkeimmät vaatimukset olivat, että se olisi pieni, yksinkertainen ja hauska. Sovellus mahdollistaisi pienen animaation lähettämisen, missä lähettäjä itse ääninäyttelisi 3D-animoidun hahmon puheen. Sovelluksen hahmot ja maailma perustuisivat suomalais-kiinalaiseen animaatiosarjaan nimeltä Dibitassut (kuva 18). Sovelluksen kehittäjäryhmään kuului kolme graafikkoa ja kaksi ohjelmoijaa. Itse toimin toisena ohjelmoijana. Projektin tiimijohtajana toimi Noora Heiskanen ja työtä valvoi Miivies Oy:n omistaja Erkki Heilakka. Futurecode Oy:n puolelta projektissa olivat mukana yrityksen toimitusjohtaja Jim Solatie ja yrityksen luova johtaja Pia Solatie. Suunnittelu aloitettiin samalla viikolla, kun tieto projektista oli ilmoitettu tiimille.



Kuva 18. Futurecode Oy:n ja kiinalaisen animaatiostudion, Bluearcin, tuottama animaatiosarja Dibitassut.

4.1 Dibitassut-animaatiosarja

Dibitassut on animaatiosarja jonka ovat luoneet Futurecode Oy ja Bluearc. Animaatiosarja on lapsille suunnattu hyvän mielen seikkailuanimaatio. Tarinan on kirjoittanut kirjailija Tuija Lehtinen, ja se kertoo pienten koirien elämästä niiden omalla planeetalla. Tarinan teema on lähitulevaisuuden avaruusseikkailu, ja sen pääosana on shiba-rotuinen koira nimeltä Viki ja sen kaverit Emmi ja Mimi. Animaatiosarja on voittanut Huldavientipalkinnon, ja se on kerännyt yli 50 miljoonaa katsojaa. Aasiassa ohjelmaa on esitetty yli 50 kanavalla. Kuvassa 19 näkyvät sarjan kolme hahmoa, Viki, Sissi ja Emmi. (48.)

Kuva 19. Sarjan päähenkilö Viki ja sen vasemmalla puolella prinsessa Sissi ja oikealla puolella Emmi (48).

Viki on sarjan päähenkilö. Se on toimelias ja kesytön poikakoira. Sillä on isä, joka on avaruusaluksen ohjaaja, äiti, joka on kirjastonhoitaja, ja isoveli joka on viemäritarkastaja. Emmi on toinen Vikin kavereista. Se on älykäs ja tunteellinen. Se on kuninkaan tytär, mutta ei pidä siitä, että sitä kohdellaan kuin aatelista. Ryhmän kolmas kaveri on nimeltään Mimi. Sen on kiltti ja herttainen tyttökoira.

Sarjan tarina alkaa, kun Emmin sisko, prinsessa Sissi, siepataan ja kaverusten pitää lähteä pelastamaan sitä. Sarjassa esiintyy myös monia muita hahmoja, kuten Vikin veli Roni, varjot, Kaktusplaneetan kissat ja kuukaudet. Erilaiset hahmot tuovat värikkyyttä ja mielenkiintoa 26 jakson animaatiosarjaan. Sovellukseen päätettiin ottaa mukaan 12 hahmoa. Hahmot valittiin roolin tärkeyden mukaan alkuperäisessä animaatiosarjassa.

4.2 Kohderyhmä

Suunnittelun alussa käytiin läpi tuotteen potentiaalinen asiakaskohderyhmä. Koska sovelluksen tulisi olla Dibitassut-henkinen ja sijoittua animaatiosarjan maailmaan, on kaikista potentiaalisin ryhmä animaatiosarjan nykyiset katsojat. Ohjelmaa on esitetty Aasiassa 28 eri maassa, ja sen suosio on yltänyt 50 miljoonaan katsojaan. Sovellus tehtäisiin siis lapsille, jotka ovat sarjan kohdeyleisö. Suunniteltaessa päätettiin myös, että sovelluksen pitäisi kuitenkin olla sen verran yksinkertainen, että jopa lapset, jotka eivät ole

katsoneet tai kuulleet alkuperäisestä sarjasta, voivat nauttia videopostikorttien tekemisestä ja lähettämisestä ja täten myös kiinnostua alkuperäisestä animaatiosarjasta.

4.3 Ideointi ja suunnittelu

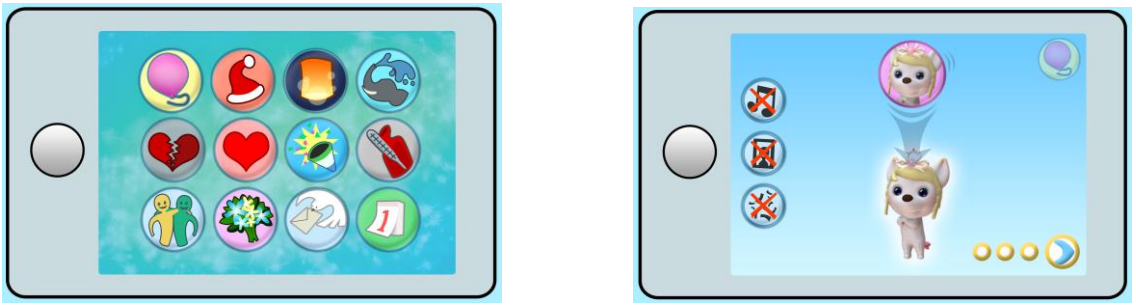
Sovellusta suunniteltaessa kehitystiimi kävi läpi yleisen tunnelman kriteerejä. Nämä kriteerit loisivat selkärangan sovellukselle, jonka ympärille muut osat rakennettaisiin. Sovellukselle määriteltiin kolme alustavaa tavoitetta, jotta sovelluksen ilme ja tunnelma sopisi alkuperäiseen konseptiin:

- valoisa ja pirteä ulkoasu
- helppokäyttöinen ja nopea
- hauska ja monipuolinen postikorttivalinta.

Aluksi grafiikkatiimi päätti sovelluksen yleisen visuaalisen tyylin. Visuaalinen tyyli olisi sekoitus 2D- ja 3D-grafiikkaa. Jotta sovellus olisi intuitiivinen ja helppokäyttöinen, se päätettiin tehdä kokonaan ilman tekstiä, pääsivun logoa lukuun ottamatta.

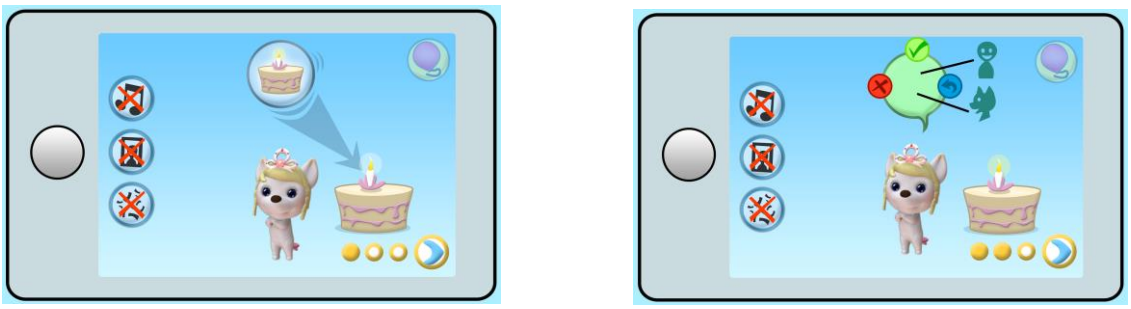
Seuraavaksi suunniteltiin yleinen sovelluksen kulku. Ideoinnin aikana tuli päätetyksi, että videopostikortin luominen olisi monivaiheinen prosessi, jossa jokainen vaihe pyytää käyttäjää valitsemaan yhden toiminnon kerrallaan.

Aluksi päätettäisiin teema, kuten syntymäpäivä tai joulukortti. Kaikilla teemoilla tulisi olemaan oma 3D-ympäristö, musiikki, tavarat ja efektit. Toiseksi valittaisiin hahmo, jonka halutaan esiintyvän kortilla. Hahmovalintoja olisi 12, ja kaikki olisivat Dibitassut-animaatiosarjan hahmoja. Kuvassa 20 näkyvät konseptit teeman ja hahmon valitsemiselle.



Kuva 20. Vasemmalla on teeman ja oikealla on hahmon valitsemisen konseptikuva.

Kolmanneksi voisi valita 3D-tilaan erilaisia tavaroita, jotka tulevat valitun teeman mukana, kuten lahjat, kakku tai vesipyssy. Neljänneksi äänitettäisiin hahmolle puhe. Puhetta voisi kuunnella ja halutessaan uudelleen nauhoittaa. Kuvassa 21 on esitetty konseptit tavaroiden valitsemiselle ja puheen lisäykselle.



Kuva 21. Vasemmalla on tavaroiden valitsemisen ja oikealla on puheen lisäyksen konseptikuva.

Viidenneksi hahmolle päätettäisiin hauska animaatio. Animaatio olisi esimerkiksi tanssiminen tai hyppiminen. Kuudenneksi voisi päättää teemaan sopivan efektin ja taustamusiikin päälle tai pois laittamisen. Kuva 22 esittää animaation valitsemisen ja efektien ja taustamusiikin valitsemisen konseptikuvat.

Kuva 22. Vasemmalla on animaation valitsemisen ja oikealla on efektien ja taustamusiikin valitsemisen konseptikuva.

Lopuksi video prosessoitaisiin videotiedostoksi, ja sen jälkeen sen voisi lähettää kaverille esimerkiksi sähköpostitse tai ladata sosiaalisen median sivustolle.

4.4 Toteutus

Alustavan suunnittelun jälkeen tuotetta alettiin toteuttaa tehdyn konseptin perusteella. Mobiilisovelluksen toteutus jaettiin karkeasti kolmeen eri kehitysvaiheeseen. Nämä vaiheet olivat visuaalinen ulkoasu, puheen äänitys ja videon prosessointi sekä lopullisen videon lähetys. Vaiheita 1 ja 2 aletiin tehdä heti, ja vaihe 3 jätettiin tehtäväksi, kun vaihe 2 olisi valmis.

Visuaalinen ulkoasu

Sovelluksen ja valmiiden videopostikorttien visuaalisen ulkoasun rakentaminen jakautui pääasiallisesti viiteen eri luokkaan:

- käyttöliittymä
- postikorttien teemat
- hahmot
- esineet
- efektit.

Unityn uuden 4.6-päivityksen mukana tulleen käyttöliittymä työkalun avulla oli mahdollista luoda monia eri käyttöliittymäasetteluja nopeasti ja helposti. Alkuperäinen suunnitelma sovelluksen kululle oli tehdä niin sanottu Wizard-käyttöliittymä. Tällä tarkoitetaan moniaskelista käyttöliittymäsuunnitelmaa, jossa käyttäjä tekee moniaskelisen prosessin askel kerrallaan, pääsemättä takaisin edelliseen askeleeseen aloittamatta koko prosessia alusta (49). Eri vaiheet suunniteltaisiin visuaalisesti helpoiksi ymmärtää, jotta takaisinmenemiselle ei olisi tarvetta. Valintoja tehtäisiin siis toistensa jatkeiksi, ja seuraava valinta aina rakentuisi edellisen päälle. Tämän tarkoitus oli tuota suoraviivaisuutta ja yksinkertaisuutta, jotta lasten ei tarvitsisi ajatella monia asioita samaan aikaan.

Alkuperäinen teeman valitseminen oli suunniteltu tulevan koko ruudulle, missä kaikki 12 tiimin valitsemaa teemaa olisivat ruudukossa. Tämän asettelun havainnollistaa kuva 23. Teemaa valittaessa taustalle tulisi näkyviin valittu teema ruudukon taakse.

Kuva 23. 12 alkuperäistä teemaa.

Ensimmäisiä teemoja, jotka sovellukselle suunniteltiin ja joita alettiin työstää, oli yhteensä 12:

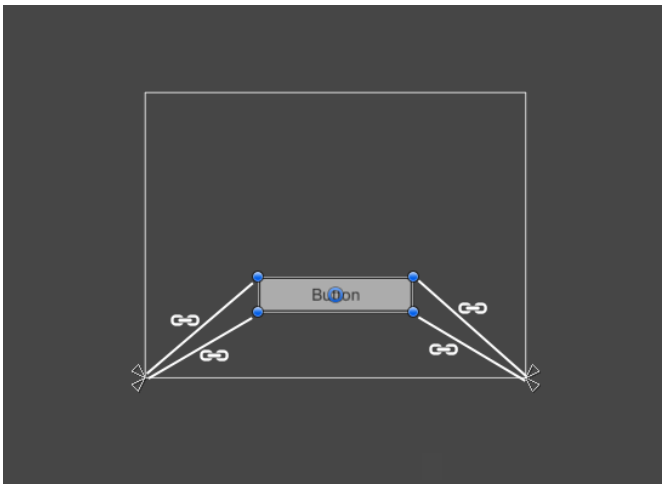
- syntymäpäiväkutsu
- uudet tapahtumat
- kahvilakutsu
- sympatiapuisto
- ”parane pian” -viesti
- rakkauskirje
- tervehdysposti
- kesälomakortti
- syystervehdys
- juhlakutsu
- uudenvuoden lyhtyjuhla
- joulukortti.

Tarkoitus oli siis luoda nappeja, joita koskettamalla teema ilmestyisi ruudulle. Jos teeman valitsemisen jälkeen koskettaisi toista teemanappia, edellinen teema vaihtuisi uudeksi teemaksi. Tällaisen valikon tekeminen Unityllä oli yksinkertaista. Unityn käyttöliittymätyökalu mahdollisti nopean yhteyden valmiin 3D-mallin ja napin välille.

Käyttöliittymätyökalu tarjoaa lukuisia eri valmiita käyttöliittymäkomponentteja, jotka voi suoraan upottaa 3D-tilaan. Upotetut komponentit ilmestyvät päällimmäiseksi, kun sovellusta käyttää. Komponentit on jaettu visuaalisiin komponentteihin ja interaktiivisiin komponentteihin. Visuaalisien komponenttien alla ovat tarvittavat tarvikkeet kuvien, tekstien ja 2D-animaatioiden tekemiseen. Niillä voi esimerkiksi tehdä kuvakkeita tai taustakuvia, jotka eivät sisällä mitään toiminnollisuuksia. Kuvat voi myös animoida liikkumaan, jos vaikka haluaisi tehdä eri vaiheiden välisen latauskuvakkeen.

Interaktiivisien komponenttien alla on työkaluja, joilla voi tehdä nappeja, valikoita, liikutettavia alueita, monivalintaruutuja ja tekstinsyöttöalueita. Niillä saa monia eri käyttöliittymäasetteluja tehtyä pelkästään raahaamalla halutut komponentit 3D-tilaan. 3D-tila mahdollistaa syvyysakselin käytön, jolla voi haluttaessa tehdä 3D-valikkoja. Komponentteja voi myös animoida erilaisilla tavoilla. Niiden kokoa, paikkaa ja väriä voidaan vaihtaa; ne voidaan laittaa myös pyörimään. Komponenteille voi myös asettaa oman taustan tai reunuksen.

Kaikki komponentit voi myös ankkuroida haluttuun kohtaan ruudulla, kuten kuvassa 24 on esitetty. Tällä voi estää mahdollisien kuvakokomuutoksien vaikuttamisen komponenttien asemaan kuvassa. Kun komponentit on asetettu haluttuihin kohtiin, niille valitaan referenssipiste jostain ruudulta, useimmiten lähin kuvaruudun kulma. Jos komponentti ankkuroidaan eri puolelta eri kohtaan ruutua, voidaan tehdä napista kokoa muuttava. Tällaisessa tapauksessa jos näytön kuvasuhde muuttuu, nappi venyy ja supistuu samassa suhteessa.



Kuva 24. Nappi on ankkuroitu alareunoihin, jotta sen koko ei muuttuisi (50).

Käyttöliittymää luotaessa sovelluksen napit asetettiin 3D-tilassa haluttuihin kohtiin, ja ulkoisella ohjelmalla tehty 3D-mallinnettu teematila tuotiin Unityyn. 3D-teematilan linkitys valittuun nappiin onnistui helposti. Kaikissa Unityn nappikomponenteissa on valmiiksi komento, jolla voi kutsua halutun luokan haluttua funktiota. Tämän lisäksi Unityssä on valmiiksi funktio, jolla voi saada esineitä ilmestymään 3D-tilassa. Tämä funktio on nimeltään *Instantiate()*. (51.)

Instantiate-komento mahdollistaa minkä tahansa esineen luomisen haluttuun koordinaatiopisteeseen. Funktio ottaa sisään kolme parametria: esineen, joka halutaan tuoda esille, sijainnin ja rotaation. Kuva 25 esittää Unityn dokumentaatiosivulla olevan esimerkin esineiden luonnista. Esimerkissä luodaan esine nimeltä *projectile* kohtaan *transform.position,* ja sille annetaan rotaatioksi *transform.rotation*. (51.)

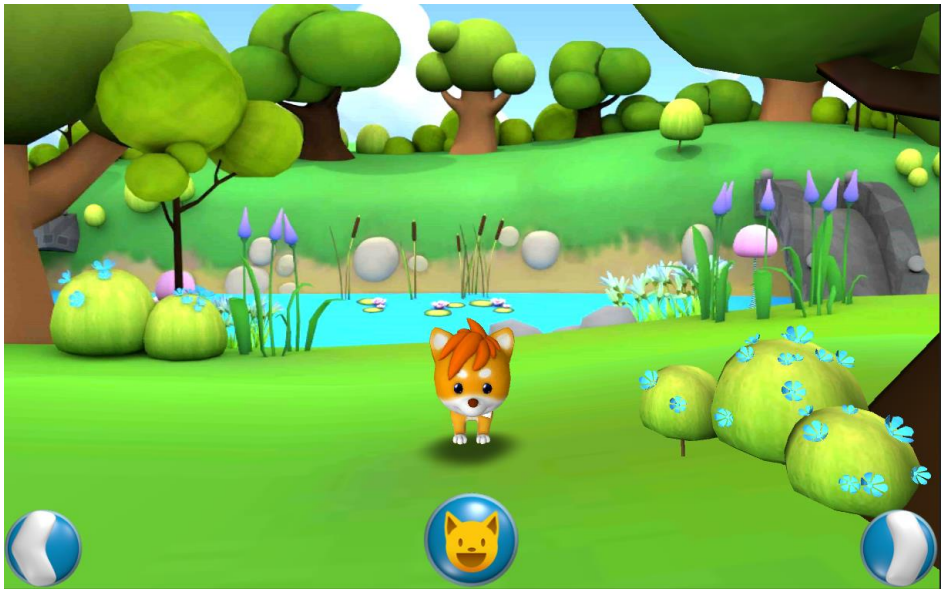
```
if (Input.GetButtonDown("Fire1")) {
        // Instantiate the projectile at the position and rotation of this transform
        var clone : Rigidbody;
        clone = Instantiate(projectile, transform.position, transform.rotation);

        // Give the cloned object an initial velocity along the current
        // object's Z axis
        clone.velocity = transform.TransformDirection (Vector3.forward * 10);
}
```

Kuva 25. Esimerkki *Instantiate()*-funktiosta Unityn dokumentaatiosivulla (51).

Kun yksinkertainen koodi oli tehty ja napin linkitys haluttuun luokkaan oli valmis, sovellusta pystyi testaamaan ensimmäisen kerran.

Jo ensimmäisissä testailuissa huomattiin, että ruudullinen teemojen asettelu ei olisi hyvä vaihtoehto. Teemat tulivat koko ruudulle ja piilottivat takana olevan taustan. Ruudullinen asettelu vaihdettiin alareunassa olevaan, vasemmalle ja oikealle liukuvaan valikkoon. Valikon asettelu näkyy kuvissa 26 ja 27. Wizard-käyttöliittymän ominainen eteneminen ilman mahdollisuutta tulla takaisin poistettiin myös. Tilalle laitettiin napit, joilla voi siirtyä eteen- ja taaksepäin.

Kuva 26. Aloitusikkuna. Ensimmäinen asia, minkä käyttäjä näkee sovelluksen sisällä.



Kuva 27. Viki sympatiapuistossa. Alareunassa hahmonvalitsemisnappi ja nuolinapit, joilla voi siirtyä eteen- ja taaksepäin.

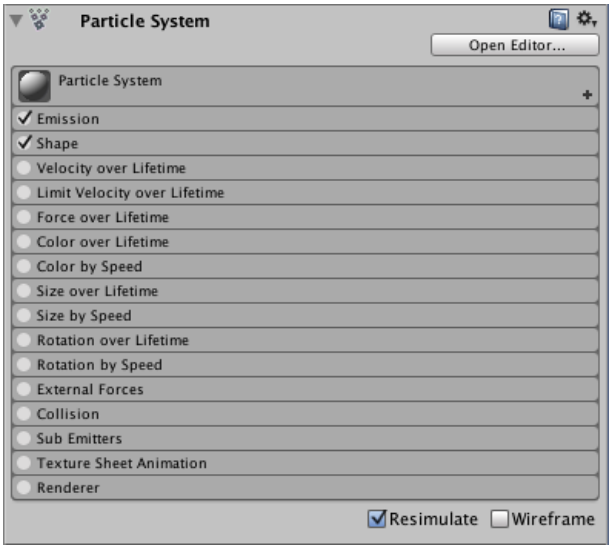
Käyttöliittymän ja teematilojen lisäksi sovellukseen asetettiin hahmoja ja teematavaroita. Sekä hahmojen että tavaroiden valitseminen ja ilmestyminen luotiin samalla tavalla kuin teematilojen. Koirien 3D-hahmomallit saatiin Bluearcilta, ja esineet mallinnettiin Suomessa. Yhteen videopostikorttiin sai kerrallaan vain yhden koiran. Koira ilmestyi keskelle 3D-tilaa, kun hahmonappia koski. Jos nappia koski uudelleen, koira vaihtui.

Koiran valitsemisen jälkeen pystyi valitsemaan erilaisia esineitä koiran ympärille. Esineet olivat aina teemaan sopivia, kuten kakku syntymäpäiväkutsuilla tai poro joulukortissa. Esineiden paikat oli valmiiksi määritetty, ja niitä oli joka teematilassa oma määrä. Kaikkiin paikkoihin sai laitettua minkä tahansa kyseisen teematilan tavaran.

Kohta, johon esineen voi luoda, on merkitty pomppivalla tähdellä. Tähti laitettiin pomppimaan, jotta se herättäisi käyttäjän huomiota ja uteliaisuutta. Käyttäjän haluttiin ajattelevan ”mitäköhän tapahtuu jos kosken tähteä”. Tämä toisi intuitiivisuutta ja poistaisi ylimääräisen tekstin tarpeen. Kun tähteä koskettaa, se vaihtuu esineeksi. Esineen voi vaihtaa toiseksi koskettamalla ilmestynyttä esinettä. Kuvasta 28 voi nähdä sinisen tähden ja muutamia tavaroita.

Kuva 28. Viki kesälomakortissa. Altaan ympärille voi laittaa vesipyssyn, snorkkelin ja uimalasit tai uimarenkaan.

Videopostikortin viimeinen visuaaliseen ulkoasuun liittyvä osio oli partikkeliefektien tekeminen. Unity sisältää oman partikkeliefektien luomistyökalun, jota käytettiin kaikkiin sovelluksen partikkeleihin. Tällä työkalulla pystyy tekemään lukuisia eri efektejä sovelluksiin tai peleihin, ja se tarjoaa eri tapoja manipuloida partikkeliefektejä. Efektien useita eri ominaisuuksia pystyy muokkaamaan helposti suoraan kuvassa 29 esitetystä Unityn partikkeliefektiikkunasta. Näihin ominaisuuksiin kuuluvat muun muassa väri, tiheys, nopeus, määrä, muoto, kuva ja kesto. (52.)



Kuva 29. Unityn partikkeliefekti ikkuna (57).

Efektit suunniteltiin teemakohtaisiksi, esimerkiksi joulukorttiin laitettiin lunta, syntymäpäiväkutsuille ilmapalloja ja kesälomakorttiin suihkulähteet. Kuvassa 30 on esitetty Mimi rakkauskirjeessä sekä kortin teemakohtaiset sydänefektit kohoamassa taivaalle. Sovelluksen viimeisessä askeleessa käyttäjä päättäisi, haluaako hän laittaa efektit päälle vai ei. Sovellus ei tarjoa käyttäjälle mitään muita valintoja efektien suhteen.

Kuva 30. Mimi rakkauskirjeessä.

Äänen nauhoitus, huulisynkronointi ja videon prosessointi

Mobiilisovelluksen toinen kehitysvaihe oli käyttäjän puheen nauhoitus ja videon prosessointi. Sovelluksen lopussa video prosessoitaisiin videotiedostoksi, jossa hahmo sanoisi äänitetyn viestin. Viestin aikana hahmo liikuttaisi suuta puheen tahtiin, animoituisi valitulla animaatiolla ja taustalla olisi valitut efektit sekä hiljaa soiva taustamusiikki. Erkki Heilakka oli edellisissä projekteissa luonut äänennauhoituksen ja huulisynkronoinnin Unityllä. Valmiit osat tuotiin suoraan nykyiseen projektiin, ja niitä käytettiin sellaisinaan. Unityn mahdollistaa helpon tiedostojen siirron projektien välillä. (53.)

Videon prosessointia varten käytettiin *Intel Inde Media for Mobile* -nimistä liitännäistä (plugin). Se mahdollistaa videotiedoston luonnin reaaliajassa Android-mobiililaittella. Unityn liitännäisten tuki mahdollisti helpon integroinnin videon prosessoinnille. Liitännäinen vaatii Unityn version 4.3.0 toimiakseen ja Andoid-sovelluskehitysympäristön (54). Liitännäiselle lähetettiin erillisestä luokasta tiedot, milloin videon prosessointi aloitetaan ja milloin se lopetetaan. Tämän lisäksi sille kerrotaan, millä nimellä animaatio tallennetaan. Liitännäisen sisäisiä koodeja muokkaamalla pystyi valitsemaan tallennuslokaatioon ja tarvittavat prosessointiparametrit, kuten kuvataajuuden.

4.5 Videopostikortin lähetys

Mobiilisovelluksen viimeinen vaihe oli valmiin videon lähettäminen. Suunnittelussa päätettiin, että video pitäisi pystyä jakamaan sähköpostitse. Unityssä voi käyttää Unityn funktiota *Application.OpenURL()* sähköpostien lähettämiseen, mutta se ei tarjoa tiedostojen lähettämisen suoraan (///). Komennolla voi avata sähköpostin, ja sieltä voi erikseen valita liitetiedoston, mutta tämä oli suunnittelutiimin mielestä ylimääräinen askel käyttäjälle ja se pitäisi jättää pois. Tämän takia päätettiin käyttää Androidin oman kirjaston tarjoamaa *android.content.intent.action_send*-komentoa. Tämä komento sallii liitetiedoston lisäämisen lähetettävään viestiin. Kun sähköpostilähetys oli tehty, huomattiin, että tämä toiminto mahdollisti Facebookissa ja Viberissä videon jakamisen.

Tavoitteena oli tehdä jakaminen mahdolliseksi myös Aasiassa suosituksi tulleella Line-mobiilisovelluksella. Line ei kuitenkaan tukenut videoiden lähettämistä samalla menetelmällä kuin sähköposti, Facebook tai Viber. Tämän takia kokeiltiin Linen omaa liitännäistä, jonka piti tarjota sovellukselle tiedostojen jakamismahdollisuus. Linen käyttäminen osoittautui hankalammaksi kuin alun perin uskottiin, pääasiallisesti sen rajoittuneen dokumentaation takia, eikä Line-lähetystä saatu toimimaan.

4.6 Testaus ja tulevaisuus

Sovellusta testattiin jatkuvasti kehityksen yhteydessä, ja sen ulkoasua, toiminallisuutta ja yleistä toimivuutta on jatkuvasti paranneltu. Viiden hengen tiimi pystyi kätevästi testaamaan ja muuttamaan ideoita pienen kokonsa ansiosta. Kaikki muutokset, jotka haluttiin tehdä, pystyi heti esittämään kaikille osapuolille, ja päätökset tehtiin sujuvasti. Unityn visuaalisuus ja helppokäyttöisyys mahdollistivat myös nopeaa ja vaivatonta iterointia eri versioiden välillä. Ulkopuolista testausta sovellukselle ei ole suoritettu.

Ensimmäinen ongelma, joka huomattiin sovelluskehityksessä, oli käyttöliittymänn alkuperäinen asettelu. Alkuperäinen ajatus oli asetella kaikki 12 valittua teemaa ruudukoksi, jonka takana vaihtuisi kuva, mutta ruudukko oli ikävästi 3D-tilan tiellä. Ruudukkoa yritettiin sommitella aluksi niin, että puolet kuvakkeista siirrettäisiin hieman lähemmäksi molempia kuvaruudun puolia, kuten kuvassa 31 on esitetty. Tällä toivottiin saatavan aikaiseksi hieman lisää tilaa keskelle ruutua, jotta 3D-tilan näkisi ja kaikki teemanapit olisivat edelleen käyttäjällä esillä.

Kuva 31. Kahteen ryhmään jaettu ruudukko.

Tätä testattaessa huomattiin, että ruudukko ei piilottaisi taustaa liikaa, kuvakkeita pitäisi pienentää noin puoleen alkuperäisestä koosta. Mobiililaitteiden rajallisen näyttökoon takia tämä oli tiimin mielestä huono vaihtoehto, joten asettelu päätettiin muuttaa.

Käyttöliittymän asettelu päätettiin siirtää kokonaan alareunaan, missä muut käyttöliittymäelementit, kuten hahmon ja animaation vaihtaminen, sijaitsivat alun perin. Alareunaan siirretty teeman valitseminen on esitetty kuvassa 32. Alareunaan tehtiin liukuva valikko eli valikko, jota pystyisi sormella liikuttamaan oikealle ja vasemmalle ja johon kaikki 12 teemaa asetettiin. Valikko päätettiin myös animoida sovelluksen käynnistyessä niin, että se liukuisi lopusta alkuun. Tämän tarkoitus oli näyttää käyttäjälle, että vaikka ruudulla näkyy kerralla vain 12 kuvaketta, on valikossa vielä muitakin piilossa.

Kuva 32. Teemanapit alareunassa.

Toiseksi huomattiin, että tavallinen valaistus ei ollut tarpeeksi viehättävä. Tämän takia päätettiin käyttää shadereita, joilla voitiin luoda keinotekoinen valaistus ruohoon ja pensaisiin. Eron vanhan ja uuden välillä voi havaita kuvasta 33. Shaderit olivat tuntematon aihe kaikille tiiminjäsenille, joten niiden käyttö oli hieman hankalaa alussa.

Kuva 33. Alkuruutu ennen shadereiden käyttöä ja sen jälkeen.

Testailussa huomattiin myös, että kun ääntä yritti äänittää koiralle, sovelluksen taustamusiikki ei sammunut. Tämä aiheutti sen, että mikrofoni kuuli äänitettävän puheen ja laitteesta tulevan taustamusiikin. Taustamusiikki tuli siis kaksi kertaa videon ääniraidalle. Tämä ongelma oli helppo korjata sammuttamalla musiikki äänittämisen ajaksi.

Videon prosessointi videotiedostoksi toi myös pulmia tiimille. Videon prosessoinnissa ei alun perin esiintynyt äänitettyä ääntä, sillä video ja ääni olivat erillisissä tiedostoissa. Jotta prosessoitu video olisi lähettämiskelpoinen, piti ääni yhdistää videoon prosessointivaiheessa. *Intel Inde Media for Mobile -liitännäisen* sisällä oli funktioita, joilla ääntä voi yhdistää videoon, joten ongelma korjattiin käyttämällä näitä funktioita.

Videon jakamisvaiheessa huomattiin ongelma, jonka takia alkuperäinen *Application.OpenURL()*-sähköpostin lähetys ei toiminut. Toiminnolla ei pystynyt lähettämään liitetiedostoja. Tämän takia lähetysmenetelmä vaihdettiin Androidin omaan *android.content.intent.action_send*-komentoon. Androidin komento mahdollisti myös jakamisen muutamalla sosiaalisen median sivuston omalla sovelluksella.

Kun ensimmäinen versio sovelluksesta oli valmis, huomattiin, että sovelluksen tiedostokoko oli noin 200 megatavua. Tiedoston suuren koon takia Google Play -kauppa ei hyväksynyt sovellusta ilman sen jakamista kahteen erilliseen tiedostoon. Tämä hoituu jakamalla sovellus yhdeksi *apk*-tiedostoksi ja yhdeksi *obb*-tiedostoksi (55). Tähän tarvittiin Googlen tarjoama liitännäinen *Google Play OBB Downloader* (56). Rajallisen dokumentaation takia ohjelmointitiimi ei saanut liitännäistä yhdistettyä muiden liitännäisien kanssa, minkä takia sitä ei saatu toimimaan oikein. Tämä aiheutti kaatumisia ja erilaisia virheilmoituksia sovelluksessa, kun sitä yritti suorittaa mobiililaitteella.

Suuren tiedoston ongelma hoitui kuitenkin, kun kaikki sovelluksen grafiikka ja äänitiedostot pakattiin. Lopullinen tiedostokoko alitti 50 megatavua, joka on Google Playn raja yksittäisien sovellusten lataamiselle kauppaan.

Tulevaisuus

Sovelluksen tulevaisuus riippuu sen menestyksestä. Sovellus julkaistaan vuoden 2015 toukokuun loppupuolella. Tulevaisuudessa sovelluksen ulkoasua halutaan hioa ja parannella. Sovelluksen suosio ulkomailla ja Suomessa vaikuttaa tuleviin päivityksiin. Sovellukseen halutaan lisätä tulevaisuudessa seuraavia asioita:

- juhlapyhiin liittyviä teemakortteja, kuten halloween ja pääsiäinen
- kouluun liittyviä teemakortteja, kuten koulunaloitus ja luokanvaihto
- lisää hahmoja sarjasta
- uutta taustamusiikkia ja efektejä
- lisää teemakohtaisia esineitä
- äänenmuunnin.

Tekniset ratkaisut ja tulevaisuuden parannukset

Insinöörityössä esitettyä sovelluksen infrastruktuuria ei noudatettu sovelluksen tekemisessä suoranaisesti, sillä sovellus aloitettiin ennen asiaan perehtymistä, jolloin tutkimustyötä sovellusten rakenteesta ei ollut vielä tehty. Sovelluksen alussa tähdättiin siihen, että kaikki osat olisi luotu erillisiksi kokonaisuuksiksi, mutta tiimin kokemattomuuden takia sovelluksen tekninen toteutus kärsii hieman epäjärjestelmällisyydestä. Sovelluksen eri komponentit olisi voinut tehdä paremmin omiksi kokonaisuuksiksi noudattaen kolmen tason jakoa.

Tämänhetkisessä ratkaisussa vain esittelytaso on oma kokonaisuutensa ja kaikki sovelluksen käyttöliittymäelementit ovat helposti muutettavissa. Sovelluksen logiikka- ja tietotaso puolestaan sekoittuvat hieman keskenään. Tämän takia yksittäiset muutokset logiikka- tai tietotasolle saattavat vaatia muutoksia molempiin tasoihin. Nykyinen rakenne haittaa mahdollisien päivityksien tekemistä nopeasti ja itsenäisesti. Tulevaisuudessa tämän voisi korjata, ja sovelluksen rakenteen voisi luoda uudelleen.

5 Yhteenveto

Insinöörityön tavoitteena oli käydä läpi mobiilisovelluksen tekeminen pelimoottorilla ja tarkastella, millä tavalla mobiilisovelluksen luominen onnistuu ohjelmalla, joka on pääasiallisesta tehty pelejä varten. Työssä perehdyttiin mobiilisovellusten suosioon ja yleistymiseen, siihen, miten ne eroavat tavallisista mobiilisovelluksista ja mikä niiden rakenne on. Huomattiin, että sovellus koostuu pääasiallisesti kolmesta tasosta, jotka esiintyvät käyttäjän ja tietokannan välillä. Kaikki tasot olisi hyvä tehdä erillään toisistaan helpomman ja nopeamman ylläpidon ja iteroinnin takia.

Tämän jälkeen tutkittiin pelimoottorin tarjoamia työkaluja, joilla kehittäjä voisi luoda hyötysovelluksia mahdollisimman helposti ja nopeasti yksin tai yrityksessä. Todettiin pelimoottorin visuaalisuus ja helppokäyttöisyys 3D-tilan mahdollistamalla esineiden kontrollilla, lukuisien eri alustojen tuki sekä monia komponentteja, kuten partikkelientekokomponentti ja puidenluomiskomponentti, joilla voi tehdä grafiikkaa pelimoottorin sisällä. Pelimoottorin ongelmiksi havaittiin tuntemattoman koodin suuri määrä, lopullisien tiedostojen mahdollisesti suuri koko ja maksullisuus. Maksullisuutta lukuun ottamatta ongelmat kohdistuvat pääasiallisesti suuriin kehitystiimeihin tai yrityksiin. Yrityksen, jolla on resursseja luoda oma pelimoottori, kannattaisi tehdä omiin tarpeisiinsa räätälöity pelimoottori, jotta se säästyisi turhalta koodilta, jota on vaikea muokata ja saattaa aiheuttaa odottamattomia tiedostokokoja. Maksullisuus takaa, että pelimoottori on ylläpidetty ja päivitetty, mutta se saattaa olla ongelma niille, joilla ei ole tarpeeksi rahaa sijoittaa pelimoottoriin.

Työssä käsiteltiin esimerkkinä Miivies Oy:ssä tehty Dibitales Tervehdys -videopostikorttimobiilisovelluksen kehityksen eri vaiheet. Huomattiin että pelimoottori mahdollisti nopean iteroinnin vaivattomasti. Tämä mahdollisti käyttöliittymän muuttamisen kesken projektin heti, kun huomattiin alkuperäisen ajatuksen olevan huono. Unity-pelimoottorin käyttäjäystävällisyyden ja grafiikkamoottorin takia saatiin myös shadereita laitettua projektiin, vaikka ne olivat kehitystiimille uusi konsepti. Sovellus valmistui noin kuudessa kuukaudessa kuuden hengen ryhmällä, mistä voidaan tehdä johtopäätös, että pelimoottorin käyttö mahdollistaa pienen yrityksen osallistumisen mobiilimarkkinoille.

Mobiilisovellusten suosio on suuri ja kasvaa edelleen. Markkinat ovat kovat, ja uusia sovelluksia tulee koko ajan lisää. Pelimoottori on tehty nopeiden ja helppojen pelien tekoa varten niille, joilla ei ole taitoa tai resursseja tehdä peliä alusta asti yksin. Mutta pelimoottori osoittautui sovellukseksi, jota voi käyttää myös hyötysovelluksien tekemisessä. Unity tarjoaa kehittäjälle monia työkaluja, joilla voi päästä potentiaalisesti suurten mobiilisovellusten markkinoiden aallonharjalle.

Lähteet

1. Fernholz, Tim. 2014. More people around the world have cell phones than ever had land-lines. Verkkodokumentti. <http://qz.com/179897/more-people-around-the-world-have-cell-phones-than-ever-had-land-lines/>. Päivitetty 25.2.2014. Luettu 9.4.2015.

2. 2 Billion Customers Worldwide to Get Smart (phones) by 2016. 2014. Verkkodokumentti. eMarketer. <http://www.emarketer.com/Article/2-Billion-Consumers-Worldwide-Smartphones-by-2016/1011694> Päivitetty 11.12.2014. Luettu 9.4.2015.

3. Tablet Users to Surpass 1 Billion Worldwide in 2015. 2015. Verkkodokumentti. eMarketer. <http://www.emarketer.com/Article/Tablet-Users-Surpass-1-Billion-Worldwide-2015/1011806>. Päivitetty 8.2.2015. Luettu 9.4.2015.

4. Janssen, Cory. Mobile Application. Techopedia. <http://www.techopedia.com/definition/2953/mobile-application-mobile-app>. Luettu 15.4.2015

5. Pawan. 2013. Verkkodokumentti. i2Mag. <http://i2mag.com/history-of-smartphone-mobile-application-development/>. Päivitetty 7.10.2013. Luettu 9.4.2015.

6. Number of apps available in leading app stores as of July 2014. 2014. Verkkodokumentti. Statista. <http://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/>. Luettu 9.4.15.

7. Statica. 2014. Verkkodokumentti. <http://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/> Luettu 9.4.2015.

8. Mobile Application Futures 2013–2017. Verkkodokumentti. Portio Research. <http://www.portioresearch.com/en/mobile-industry-reports/mobile-industry-research-reports/mobile-applications-futures-2013–2017.aspx> Luettu 9.4.2015.

9. Most popular Apple App Store categories in March 2015, by share of available apps. Verkkodokumentti. Statista. <http://www.statista.com/statistics/270291/popular-categories-in-the-app-store/>. Luettu 9.4.2015.

10. Most popular Google Play categories. Verkkodokumentti. Appbrain. <http://www.appbrain.com/stats/android-market-app-categories>. Luettu 9.4.2015.

11. Clark, Josh. 2010. Tapworthy. O'Reilly Media.

12. Evernote. Verkkodokumentti. Evernote. <https://evernote.com/intl/fi/>. Luettu 9.4.2015.

13. Evernote. 2008. Verkkodokumentti. Mergy. <http://mergy.org/2012/05/use-the-evernote-clipper-with-safari-on-ipad-and-iphone/>. Luettu 7.5.2015

14. Moon, Garret. 2014. Why people share. Verkkodokumentti. http://coschedule.com/blog/why-people-share/ Päivitetty 2.10.2014. Luettu 17.4.2015.

15. Pinterest. 2009. Verkkodokumentti. Agrinnet. <http://cdn.yjc.ir/files/fa/news/1391/10/9/669184_848.jpg>. Luettu 7.5.2015

16. Share icons. Verkkodokumentti. Google kuvahaku. <https://www.google.fi/search?q=share+button&espv=2&biw=917&bih=785&source=lnms&tbm=isch&sa=X&ei=wNwvVdyJHMmUsAHPsYLYDw&ved=0CAYQ_AUoAQ#tbm=isch&q=share+icon> Luettu 15.4.2015.

17. Share file link via. 2012. Verkkodokumentti. Guidingtech. < http://www.guidingtech.com/assets/postimages/2012/12/cloudsend-to-share-files-on-Android-1.png> Luettu 7.5.2015.

18. Shazam. Verkkodokumentti. Shazam. <http://www.shazam.com/> Luettu 15.4.2015.

19. Soundhound. Verkkodokumentti. Soundhound. <http://www.soundhound.com/> Luettu 15.4.2015.

20. Shazam, Soundhoun. 1999, 2009. Verkkodokumentti. Somosiphone. <http://somosiphone.com/wp-content/uploads/shazam_soundhound.jpg>. Luettu 15.4.2015.

21. Shadow Cities. Verkkodokumentti. Grey Area. <http://fi.wikipedia.org/wiki/Shadow_Cities> Luettu 15.4.2015.

22. Shadow Cities. 2010. Verkkodokumentti. Gigaom. <https://gigaom.com/2011/02/24/419-grey-area-gets-2-6-million-to-build-out-location-aware-gaming-business/>. Luettu 7.5.2015.

23. Nanji, Ayaz. 2014. Mobile Trends: Most Popular Phones, Screen Sizes, and Resolutions.Verkkodokumentti. <http://www.marketingprofs.com/charts/2014/25740/mobile-trends-most-popular-phones-screen-sizes-and-resolutions>Päivitetty 6.8.2014 Luettu 15.4.2015.

24. Meier, J.D., Homer, A., Hill, D., Taylor, J., Bansode, P., Wall, L., Boucher, R., Bogawat, A. 2008. Mobile Architecture Guide. Verkkodokumentti. <http://robtiffany.com/wp-content/uploads/2012/08/Mobile_Architecture_Guide_v1.1.pdf>. Luettu 10.4.2015.

25. Clover, Juli, 2013. Tumblr for ios gets redesigned user interface after yahoo acquisition. Verkkodokumentti. <http://www.mac-rumors.com/2013/05/20/tumblr-for-ios-gets-redesigned-user-interface-after-yahoo-aquisition/> Päivitetty 20.5.2013. Luettu 15.4.2015.

26. Pitt, Ben. 2010. How can I connect Unity to an SQL database in order to implement an MMO? Verkkodokumentti. <http://answers.unity3d.com/questions/15694/how-can-i-connect-unity-to-an-sql-database-in-orde.html>. Päivitetty 21.4.2010. Luettu 15.4.2015.

27. Showcase. Verkkodokumentti. Unity. <http://unity3d.com/showcase/gallery/non-games>. Luettu 15.4.2015.

28. Inspector. Verkkodokumentti. Unity. <http://docs.unity3d.com/Manual/Inspector.html> Luettu 15.4.2015.

29. Unity view. 2013. Verkkodokumentti. Informit. < http://ptgmedia.pearsoncmg.com/images/art_geig_whyuseunity/elementLinks/Geig1_fig1.png> Luettu 7.5.2015.

30. Input. Verkkodokumentti. Unity. <http://docs.unity3d.com/Manual/Input.html> Luettu 15.4.2015.

31. Asset Store. Verkkodokumentti. Unity. <https://www.assetstore.unity3d.com/en/> Luettu 15.4.2015.

32. UI. Verkkodokumentti. Unity. <http://docs.unity3d.com/Manual/UISystem.html> Luetu 15.4.2015.

33. Unity UI. 2014. Verkkodokumentti. Brunchnews. < https://tctechcrunch2011.files.wordpress.com/2014/11/gui.png?w=680> Luettu 7.5.2015.

34. Light. Verkkodokumentti. Unity. <http://docs.unity3d.com/Manual/class-Light.html> Luettu 15.4.2015.

35. Camera. Verkkodokumentti. Unity. <http://docs.unity3d.com/Manual/class-Camera.html> Luettu 15.4.2015.

36. Unity camera and light. 2012. Verkkodokumentti. 3dgep. < http://3dgep.com/wp-content/uploads/2012/07/Unity-Directional-Light-Aligned-to-Main-Camera-397x300.png> Luettu. 7.5.2015.

37. Material. Verkkodokumentti. Unity. <http://docs.unity3d.com/Manual/class-Material.html> Luettu 15.4.2015.

38. Textures. Verkkodokumentti. Unity. <http://docs.unity3d.com/Manual/terrain-Textures.html> Luettu 15.4.2015.

39. Tree Creator. Verkkodokumentti. Unity. <http://docs.unity3d.com/Manual/class-Tree.html> Luettu 15.4.2015.

40. Tree creator. 2013. Verkkodokumentti. 3dgep. < http://3dgep.com/wp-content/uploads/2012/11/Unity-Tree-Creator.jpg> Luettu 7.5.2015.

41. Matthew, Humphries. 2011. Box2D creator asks Rovio for Angry Birds credit at GDC. Verkkodokumentti. <http://www.geek.com/games/box2d-creator-asks-rovio-for-angry-birds-credit-at-gdc-1321779/> Päivitetty 2.3.2011. Luettu 15.4.2015.

42. Dectools Inspire. Verkkodokumentti. Decktools Inspire. <http://www.decktoolsinspire.com/> Luettu 15.4.2015.

43. Multiplatform. Verkkodokumentti. Unity. <http://unity3d.com/unity/multiplatform> Luettu 15.4.2015.

44. PlayerPrefs. Verkkodokumentti. Unity. <http://docs.unity3d.com/ScriptReference/PlayerPrefs.html> Luettu 15.4.2015.

45. Serialization. Verkkodokumentti. Unity. <http://docs.unity3d.com/Manual/script-Serialization.html> Luettu 15.4.2015.

46. Crystal Tools. Verkkodokumentti. Game watch< http://game.watch.impress.co.jp/docs/20080225/gdc_cry.htm> Luettu 7.5.2015.

47. Unity Pro. Verkkodokumentti. Unity. <http://unity3d.com/get-unity> Luettu 15.4.2015.

48. Vientipalkinto Dibitassut-animaatiolle. 2014. Verkkodokumentti. Helsingin Sanomat. <http://www.hs.fi/kulttuuri/a1389852063607> Päivitetty 17.1.2014. Luettu 15.4.2015.

49. Wizard UI. Verkkodokumentti. Microsoft. < https://msdn.microsoft.com/en-us/library/windows/desktop/bb246451%28v=vs.85%29.aspx> Luettu 7.5.2015.

50. Unity UI button. 2014. Verkkodokumentti. Xclouder. < http://xclouder.com/images/164d81f4bdb29635f2db9aff5e4141104e005c60.gif> Luettu 7.5.2015.

51. Instantiate. Verkkodokumentti. Unity. <http://docs.unity3d.com/ScriptReference/Object.Instantiate.html> Luettu 15.4.2015.

52. Particle System. Verkkodokumentti. Unity. <http://docs.unity3d.com/Manual/ParticleSystems.html> Luettu 15.4.2015.

53. How do I reuse assets between projects? Verkkodokumentti. Unity. <http://docs.unity3d.com/Manual/HOWTO-exportpackage.html> Luettu 15.4.2015.

54. Aleshkov, Ilya. 2014. Intel® INDE Media for Mobile Tutorials - Advanced Video Capturing for Unity3d* Applications on Android. Verkkodokumentti. Unity. <https://software.intel.com/en-us/articles/intel-inde-media-pack-for-android-tutorials-advanced-video-capturing-for-unity3d> Päivitetty 17.4.2014. Luettu 15.4.2015.

55. Support for Split Application Binary (.OBB). Verkkodokumentti. Unity.< http://docs.unity3d.com/Manual/android-OBBsupport.html> Luettu 18.4.2015.

56. Google Play OBB Downloader. Verkkodokumentti. Unity. < https://www.assetstore.unity3d.com/en/#!/content/3189> Luettu 18.4.2015.