OULUN AMMATTIKORKEAKOULU

Hanna Jurvelin

2D-mobiilipeligrafiikan optimointi

State of Matter -peliprojekti

2d-mobiilipeligrafiikan optimointi

State of Matter -peliprojekti

Hanna Jurvelin
Opinnäytetyö
Kevät 2016
Viestinnän tutkinto-ohjelma
Kuvallinen viestintä
Oulun ammattikorkeakoulu

TIIVISTELMÄ

Oulun ammattikorkeakoulu
Viestinnän tutkinto-ohjelma
Kuvallinen viestintä

Tekijä: Hanna Jurvelin
Opinnäytetyön nimi: 2D-mobiilipeligrafiikan optimointi
Työn ohjaaja: Tuukka Uusitalo
Työn valmistumislukukausi- ja vuosi: kevät 2016 Sivumäärä: 37+3

Optimointi tarkoittaa sovelluksen tiedostokoon pienentämistä vaihtelevilla keinoilla. Työssäni käyn läpi 2D-mobiilipeligrafiikan optimointikeinoja sekä teknisestä, että visuaalisesta puolesta. Työn tausta pohjautuu State of Matter -mobiilipeliprojektiin, jossa olin mukana graafikkona. Työn tavoite on selventää optimoinnin käsitettä ja selostaa 2D-grafiikan optimointiin liittyviä tekniikoita. Lisäksi kerron State of Matter -peligrafiikoiden suunnitteluprosessista.

Työssä on käytetty tietoperustana Internetistä löytyneitä lähteitä, ja erityisesti Unity-pelimoottorin nettisivuja. Koska käytimme State of Matter -peliprojektissa Unityä, työn pääpaino on peligrafiikan optimoimisessa Unityllä. Lisäksi lähetin haastattelukysymyksiä ammattilaismobiilipeligraafikoille, jotka kertoivat omia kokemuksiaan optimoinnista. Haastateltavat kertoivat omia mielipiteitään graafisen optimoinnin merkityksestä peliprojektissa, ja antoivat neuvoja sekä optimointiin että yleisesti mobiilipeligraafikon työstä kiinnostuneille.

Työssä käy ilmi, että mobiilipeliprojektissa optimointia tulisi suorittaa projektin jokaisessa vaiheessa, suunnittelusta testaukseen ja sovelluskauppaan asti. Optimointiin on olemassa useita erilaisia tapoja, mutta yhtä ainoaa oikeaa tapaa ei ole, sillä jokainen projekti vaatii omanlaisiaan teknisiä keinoja. Optimoinnin tekniset keinot muuttuvat nopeasti laitteiston parantuessa ja laajentuessa vuosi vuodelta.

Asiasanat: mobiilipelit, mobiilipeligrafiikka, graafinen suunnittelu, optimointi

ABSTRACT

Oulu University of Applied Sciences
Visual communication

Author: Hanna Jurvelin
Title of thesis: 2D-mobile game graphics optimization
Supervisor(s): Tuukka Uusitalo
Term and year when the thesis was submitted: Spring 2016 Number of pages: 37+3

Optimization means reducing the app size with various means. In my work I present ways to optimize 2D-mobile game graphics both from technical and visual point of view. The work is based on State of Matter -mobile game project, in which I worked as a graphic designer. The objective of the work is to clear the meaning of optimization and explain the ways to optimize 2D-graphics. I also describe a bit about the designing progress of State of Matter's graphics.

I used Internet sources as the knowledge basis of my work, and especially the websites of the Unity game engine. Because we used Unity in the State of Matter -project, the main focus of the work is mobile game graphics optimization in Unity. I also sent some questions to professional mobile game graphic designers, whom explained about their own experiences with optimization. The interviewees also told about their own opinions of the significance of graphic optimization in a game project. They also gave some advice about optimization, and tips to people who are interested in the work of a mobile game graphic designer.

In my work it transpires that optimization should be performed in all the stages of the game project, from designing to testing and even after uploading the game to app store. There are multiple techniques to optimization, but you can't find one true way to do it, since every project needs their own technical means. The technical ways of optimization change fast as the hardware improves and widen yearly.

Keywords: mobile games, mobile game graphics, graphic design, optimization

SISÄLLYS

1 JOHDANTO

Aloitin State of Matter -peliprojektissa työskentelemisen tammikuussa 2015. State of Matter on 2D-tasohyppelypeli mobiililaitteille, joka syntyi Game Lab -koulutuksesta. Sen alustava julkaisu (englanniksi soft launch) oli vuoden 2016 helmikuussa. Suunnittelin peliin monipuolisesti grafiikkaa hahmoista taustoihin, käyttöliittymään ja animaatioon. Koska tein pelistä produktion, oli loogista kirjoittaa myös opinnäytetyö, joka liittyy pelialaan. Valitsin aiheekseni mobiilipeligrafiikan optimoinnin, koska mielestäni tästä hyvin tärkeästä osasta peliprojektia ei ole kirjoitettu tarpeeksi.

Aioin kirjoittaa pelien optimoinnista sekä mobiili- että tablet-laitteisiin, mutta minua neuvottiin rajaamaan aihetta. Vedin siis rajan mobiililaitteiden optimointiin ja erityisesti älypuhelimien optimointiin. Päätin myös rajata aiheen 2D-grafiikaan ja olla ollenkaan käsittelemättä 3D-grafiikkaa. En ole perehtynyt aiheeseen, joten tarvittavaa tietopohjaa siihen minulla ei ollut. 3D-grafiikan tekeminen ja optimointi ovat aivan erilaisia kuin 2D-grafiikan, ja minulta olisi mennyt valtavasti aikaa, jos olisin alkanut perehtyä aiheeseen opinnäytetyön kirjoitusvaiheessa. Päätin myös jättää koodauksen tekniset optimoinnin keinot ja musiikin optimoinnin käsittelemättä samasta syystä.

Tärkein syy optimointiin on yksinkertaisesti mobiilipelin koon pienentäminen. Pelin täytyy olla pakattu mahdollisimman pieneksi, tinkimättä kuitenkaan pelattavuudesta, grafiikoista, elämyksellisyydestä ja pituudesta. Mobiilipelien kasvattaessa suosiotaan myös optimoinnin merkitys kasvaa. Kaikkien käyttäjäryhmien tulee olla mahdollista pelata niitä laitteistosta riippumatta, high-end-älypuhelimien käyttäjistä low-end-malleja käyttäviin. Erityisesti suuret Aasian ja Intian markkinat ovat usein low-end laitteiden käyttäjiä, mikä taas kasvattaa optimoinnin merkitystä pelien kehityksessä.

Suurin haaste opinnäytetyössäni on vakavasti otettavien lähteitten löytäminen. Mobiilipeligrafiikan optimoinnista ei ole kirjoitettu kovinkaan paljon. Infoa löytyy epämääräisiltä nettisivuilta ja keskustelupalstoilta, mutta vähemmän arvostetuilta nettisivuilta tai printtimediasta. Tieto myös vanhenee nopeasti mobiililaitteiden muuttuessa vuosi vuodelta. Aihe on kuitenkin tuore eikä sitä ole ainakaan liiaksi käsitelty. Vakavasti otettavien lähteiden vähyyden vuoksi lähetin kysymyksiä kolmelle ammattilaisille peligraafikoille, sillä heillä on suoran käden kokemusta pelien optimoinnista sekä piilotettua tietoa, mitä netistä ei löydy. He vastasivat kysymyksiin hyvin avoimesti ja antoivat neuvoja optimoinnin teknisestä näkökulmasta, sekä yleisiä vinkkejä alasta kiinnostuneille graafikoille.

Opinnäytetyöni on suunnattu erityisesti 2D-graafikoille, jotka suunnittelevat ja toteuttavat mobiilipelejä. Opinnäytetyöni hyödyttäisi ehdottomasti eniten, jos sen lukisi ennen projektin aloittamista, mutta optimointi ei ole myöhäistä edes projektin loppuvaiheessa. Tärkeintä on, että siihen kiinnittää huomiota jossain vaiheessa projektia, vaikka paras mahdollinen tulos syntyy, kun optimointia pohtii projektin jokaisessa vaiheessa. Usein ehkä ajatellaan, että optimointiin voi vaikuttaa vain ohjelmoinnin puolella, mutta myös graafisilla suunnittelijoilla ja artisteilla on useita eri tapoja osallistua tähän tärkeään prosessiin. Mobiilipeleissä mikään optimoinnin keino ei ole vähäpätöinen, sillä kaikki pelin kokoon vaikuttavat keinot kannattaa käyttää. Optimoinnilla on suora yhteys pelin menestymiseen.

2 OPTIMOINTI

2.1 Sovelluskauppojen rajoitukset

Applen App Store -kaupan sovellusten kokoraja on ollut sata megatavua jo hetken aikaa. Myös Google ilmoitti kaksinkertaistavansa Play-kaupan mobiilisovellusten koon viidestäkymmenestä megatavusta sataan. Lisäksi kehittäjillä on mahdollisuus laajentaa sovelluksiaan jopa kahdella gigatavulla lisämateriaalia. (David 2015, viitattu 22.10.2015). Tämä antaa sekä iOS- että Android-kehittäjille mahdollisuuden tehdä monimutkaisempia, laadukkaampia ja laajempia sovelluksia, mutta se ei silti tarkoita, että niiden koko kannattaisi kasvattaa maksimiin. Kehittäjien pitää ottaa huomioon laitteiston erot sekä se, että suurta peliä on vaikeampi ladata puhelimen omalla mobiilidatalla. Kaikki käyttäjät eivät käytä langatonta verkkoa, ja pelikoon suuruus voi vähentää käyttäjien määrää jopa puolella (Willoughby 2015, viitattu 22.10.2015). Willoughbyn mukaan poikkeuksena ovat premium -pelien ostajat, sillä erityisesti kalliita mobiilipelejä ostavat ovat valmiimpia lataamaan suuria pelejä. Siispä erityisesti ilmaiseksi ladattavan (englanniksi free-to-play) pelin koon tulisi olla mahdollisimman pieni. Miten siis tehdä esteettinen, hauska ja laaja mobiilipeli mahdollisimman pienessä koossa? Vastaus tähän on pelin optimoiminen niin graafisesti kuin ohjelmoinnin ja musiikin puolella.

2.2 Laitteiston monipuolisuus

Laitteisto vaihtelee todella paljon erityisesti Android-laitteilla. Android-älypuhelimien valmistajia on useita, koska Google antaa käyttöjärjestelmän oikeudet laitevalmistajille. Siispä Android-laitteita valmistetaan monelle eri käyttäjäryhmälle useissa eri hintaluokissa. Suurimmat Android laitevalmistajat ovat Samsung, Huawei, Xiaomi ja Lenovo. Samsung on kaikista suurin älypuhelimien valmistaja 21,4 prosentilla kaikista myydyistä laitteista. (IDC Research 2015, viitattu 26.10.2015). Samsung:in Galaxy S -sarja toimii yhtiön lippulaivamallistona. Yhtiöllä on ollut tapana valmistaa Galaxy S mallistostaan sekä high-end- että low-end-versio. Esimerkiksi Galaxy S5:ssa on 5,1- tuuman full HD-näyttö eli sen tarkkuus on 1920x1080 pikseliä ja 2,5 GHz neliydinsuoritin. Siitä halvempi versio on Galaxy S5 mini, jonka 4,5-tuuman näyttö on HD eli 720x1280 pikseliä ja 1,4 GHz neliydinsuoritin (Samsung 2015, viitattu 26.10.2015). Vuoden 2015 lippulaivamalli Galaxy S6, jonka näyttö on hurjat 2560x1440 pikselia ja sisällä prosesseja hoitaa 2,1GHz kahdeksanydinsuoritin, ei

ole tehostaan huolimatta menestynyt odotetusti, mutta IDC Research:in mukaan Samsung:in low-end mallit kasvattavat suosiotaan erityisesti Kaakkois-Aasiassa, Lähi-Idässä ja Afrikassa, jonne on odotettavissa myös mobiilipelimarkkinoiden hurjaa kasvua.

Applen iPhone-älypuhelimissa laitteiston vaihtelevuus ei ole niin suurta, sillä iOS-käyttöjärjestelmän oikeudet ovat vain Applella. iPhonet ovat pääsääntöisesti high-end-laitteita, lukuun ottamatta vuoden 2013 iPhone 5c:tä, joka vastasi prosessointiteholtaan vuoden 2012 standardeja. Näyttöjen resoluutiot kuitenkin vaihtuvat noin kahden vuoden sykleissä, joten vaihtelevuutta näistä löytyy vanhemman iPhone 4:n 960x640 pikselin resoluutiosta uusimpien iPhone 6:n 1134x750 pikselin näytöstä iPhone 6s:n 1920x1080 pikselin näyttöön (Unity 3D 2015a, viitattu 26.10.2015). Ero on melko suuri siihen nähden, että mobiilipelin pitää olla sujuvasti pelattava ja näyttää hyvältä riippumatta pelaako sitä iPhone 4:llä vai iPhone 6s:llä.

2.3 Erilaisia optimointeja

Mobiilipelien optimointi on laaja käsite. Opinnäytetyössäni käsittelemä optimointi tarkoittaa pääasiassa mobiilipelin kokoon vaikuttavia pääpainossa graafisia, mutta osittain myös teknisiä tapoja. Mobiilipelien optimoinnilla voidaan kuitenkin tarkoittaa muitakin konsepteja, kuten sovelluskauppojen sisäistä optimointia ja lokalisoitua optimointia.

2.3.1 Sovelluskaupan sisäinen optimointi

Kehittäjän pitää ottaa myös huomioon sovelluskauppojen sisäinen optimointi. Sovelluskauppojen sisäisellä optimoinnilla tarkoitetaan sitä, kuinka hyvin kyseinen sovellus, eli tässä tapauksessa mobiilipeli, löytyy sovelluskaupasta. Sen tavoitteena on saada peli mahdollisimman näkyväksi sovelluskaupassa sekä tehdä siitä helposti löydettävä. Tärkeintä sisäistä optimointia on pelin nimi ja hakusanat. Mobiilipelin nimi toimii hakusanana, eli jos sen vaihtaa myöhemmin, on sovellusta vaikeampi löytää. Nimeen voi myös lisätä hakusanan, mikä Gangulyn (2015, viitattu 23.10.2015) mukaan parantaa sen löydettävyyttä 10,3 prosenttia. Hakusanat ovat tärkein sisäisen optimoinnin tapa. Ne pitää miettiä erityisesti kohdeyleisön mukaan mutta myös pelin tärkeimpien ominaisuuksien perusteella sekä katsomalla kilpailevien mobiilipelien hakusanoja. Hakusanoja ei pidä keksiä viime hetkellä ennen pelin julkaisemista, vaan niitä pitää miettiä tarkkaan ja päivittää säännöllisesti julkaisunkin jälkeen.

Myös visuaaliset tekijät ovat tärkeitä sisäisessä optimoinnissa. Pelin ikoni toimii koko pelin edustajana, eli selatessa pelejä käyttäjä tekee valinnan ikonin perusteella. On siis tärkeää, että ikoni on tyylikäs, huomiota herättävä ja hauska, riippuen tietenkin myös kohdeyleisöstä. Pelin sovellussivun tulee jatkaa samalla visuaalisella linjalla kuin ikoni, bannerista kuvankaappauksiin ja videoon. Visuaalisia tekijöitä ei tule väheksyä tehdessä sovelluskaupan sisäistä optimointia.

Vähempiarvoisia sisäisen optimoinnin menetelmiä ovat latausten määrä sekä arvostelut ja arvosanat. Näitä kehittäjän on kuitenkin vaikea kontrolloida. Laadukkaiden sovellusten tekeminen on selkein tapa saada hyviä arvosteluja ja arvosanoja. Jos käyttäjä arvostaa peliä, antaa hän luultavasti sille hyvän arvosanan. Mainonnalla saa taas nostettua latausten määrää, mikä auttaa pelin tunnettuudessa. Mobiilipeleihin pätee dominoefekti: mitä enemmän latauksia peli saa, sitä enemmän sitä ladataan myös tulevaisuudessa. Mitä enemmän latauksia pelillä on, sitä enemmän myös arvosteluja ja arvosanoja. (Ganguly, 2015. Viitattu 23.10.2015.)

2.3.2 Lokaali optimointi

Globaalit kansainväliset markkinat houkuttelevat jokaista pelinkehittäjää. Varsinkin Aasian ja Intian kasvavat mobiilipelimarkkinat ovat lupaavia kultakaivoksia. Pelin suora vieminen paikalliseen sovelluskauppaan voi kuitenkin vähentää latauksien määrää. Siksi ennen pelin julkaisemista eri maissa, pitää se optimoida lokaalisti. OneSky:n (2014, viitattu 25.10.2015) mukaan lokalisaatiota voi kuitenkin toteuttaa pikkuhiljaa ennen koko pelin kääntämistä useille kielille. OneSky neuvoo aloittamaan lokalisaation sovelluskaupan kääntämisellä. Sovelluskauppojen kuvaukset ovat lyhyitä, joten niiden kääntäminen on nopea ja halpa tapa testata markkinoita. Kannattaa kuitenkin mainita pelin olevan eri kielellä kuin kuvaus sekaannusten välttämiseksi. OneSky myös suosittelee ammattilaisen palkkaamista kääntötyöhön ja muistuttaa hakusanojen kääntämisen tärkeydestä.

Vasta tämän jälkeen kannattaa tehdä päätös, kääntääkö koko peli eli kaikki pelin sisäiset tekstit paikalliselle kielelle. OneSky viittaa Distimon tutkimukseen, jonka mukaan iPhone-sovelluksen vetovoima kasvoi 128 prosenttia viikossa kääntämisen jälkeen verrattuna samaan aikaan ennen käännöksen tekemistä. Vaikka pelin kääntäminen voi tuntua suurelta urakalta, se luultavasti loppujen lopuksi palkitsee kehittäjän.

Kääntämisen lisäksi huomioon tulee ottaa kulttuurisaatio (Honeywood & Fung 2012, viitattu 25.10.2015). Lokalisaation ja kulttuurisaation ero on se, että kun lokalisaatio saa pelaajan ymmärtämään pelaamaansa peliä, kulttuurisaatio saa pelin sisällön tuntumaan mielekkäämmältä. Se myös estää sen, että pelaaja ei ymmärrä pelin osia tai jopa tuntee ne loukkaaviksi. Kulttuuriset eroavaisuudet voivat aiheuttaa hyvinkin vaarallisia väärinymmärryksiä, ja jopa estää pelin julkaisun maassa. Honeywood & Fung listaa tärkeimpiä asioita, joita kannattaa ottaa huomioon kulttuurisaatiossa: historia, uskonto ja uskomukset, etnisyys ja kulttuuriset konfliktit sekä geopoliittiset seikat. Heidän mukaansa kulttuurisaatio ei tarkoita koko pelin muuttamista, sillä pienillä muutoksilla voi estää suurta vahinkoa tapahtumasta.

3 OHJELMAT

3.1 Unity-pelimoottori

Unity-pelimoottori on markkinoiden johtava pelinkehitysohjelma. Sillä voi kehittää pelejä useille eri alustoille mobiilista tietokoneelle. Unityllä on 4,5 miljoonaa rekisteröitynyttä kehittäjää ja 600 miljoonaa pelaajaa (Unity 3D, viitattu 16.1.2016). Tärkeä syy suosioon on se, että ohjelma on ilmainen tiettyyn rajaan asti. Ilmaisella mobiilipelienkehitysversiolla voi kehittää ja myydä pelejä, kunhan yrityksen vuotuiset tulot jäävät alle 100 000 dollarin. Jos raja ylittyy, yrityksen on ostettava Unity Pro -versio, joka tarjoaa myös laajempia kehitysmahdollisuuksia (Helgason 2013, viitattu 16.1.2016).

Unityn monipuolinen käyttäjäkunta vaihtelee suurista yhtiöistä, kuten Cartoon Network ja Microsoft, pieniin indie-yrityksiin ja vasta-aloitteleviin kehittäjiin. Pelin kehittäminen Unityllä toimii raahaa ja pudota (englanniksi drag and drop) -tyylillä, jonka avulla on mahdollista rakentaa pelejä hyvinkin vähäisellä ohjelmoinnilla (Brodkin 2013, viitattu 16.1.2016). Ilmaisversio ja helppo käytettävyys ovat aiheuttaneett Unitylle ristiriitaisen maineen: vaikka sitä käytetään suuriin ja arvostettuihin peliprojekteihin, monet aloittelevat kehittäjät käyttävät sitä huonolaatuisten ja nopeasti tehtyjen pelien tekemiseen. Syy tähän on se että, pro-versiota käyttävien ei tarvitse näyttää Unityn logoa pelin aloitusruudussa (englanniksi splash screen), kun taas ilmaisversio vaatii sitä (Dale 2015, viitattu 16.1.2016).

Koska käytimme State of Matter -pelin tekemiseen Unity-pelimoottoria, jotkin esittelemäni optimointitekniikat voivat olla siihen sidonnaisia. Useimmat tekniikat ovat kuitenkin yleispäteviä pelien tekemisessä, ja joitakin tekniikoita voi käyttää soveltamalla niiden perusajatusta myös muiden pelimoottoreiden avulla tehtäviin 2D-mobiilipeleihin.

3.2 Adobe-ohjelmat

Käytin kaikkien peligrafiikoiden tekemiseen Adobe Illustrator -ohjelmaa. Adobe Illustrator on vektorigrafiikkaan erikoistunut piirto-ohjelma. Johnson (2011, viitattu 16.1.2016) listaa vektorigrafiikan ja Illustrator-ohjelman hyvät puolet klassisen rasterigrafiikan ja Photoshop-ohjelman yli: vektorigrafiikkaa voi skaalata huolehtimatta laadun huonontumisesta. Kuvan ääriviivoja voi muokata helposti

vektoripisteitä käyttäen. Ohjelmassa on näppäriä käyttäjää auttavia apukeinoja kuten rajaava suorakaide (englanniksi bounding box) ja älykkäät ohjausviivat, jotka voi kuitenkin sulkea halutessaan. Tasot (englanniksi layers) toimivat Johnsonin (2011, viitattu 16.1.2016) mukaan Illustratorissa paljon loogisemmin kuin Photoshopissa. Käyttäjä voi halutessaan tehdä kuvan yhdelle tasolle tai jakaa sen niin monelle tasolle kuin haluaa. Tason sisällä on alatasot, josta objektien hierarkiaa voi vaihtaa. Illustratorissa ei ole piirtoaluetta (englanniksi canvas) kuten Photoshopissa vaan piirtotauluja (englanniksi artboard), joita voi tehdä samaan dokumenttiin useita ja eri kokoisia. Mielestäni Illustratorin heikkous voi kuitenkin olla se, että työnjälki voi olla liian jäykkää ja jopa liian virheetöntä, varsinkin jos katsojan visuaalista silmää hivelee Photoshopin ja muiden rasterigrafiikkaohjelmien tekemä maalauksellinen ja orgaaninen tyyli.

3.3 Tiedostomuodoista

PNG eli Portable Network Graphics on rasteroidun grafiikan tiedostomuoto, joka tehtiin korvaamaan GIF-tiedostomuoto. PNG käyttää häviötöntä pakkausta, eli kuvadataa ei katoa tallentaessa tai katsoessa tiedostoa (Bither, 2010, viitattu 18.1.2016). Vaikka JPEG-muoto on Internetissä käytetyin tiedostomuoto pienen tiedostokoon vuoksi, se ei kuitenkaan taltioi alpha-kanavaa eli läpinäkyvyyttä. PNG yhdistää läpinäkyvyyden ja hyvän kuvanlaadun tehokkaaseen tiedon pakkaukseen.

JPEG on paras tiedostomuoto valokuville pienen tiedostokoon takia, mutta mobiilipeleissä PNG on parempi valinta juuri läpinäkyvyyden vuoksi. Sprite-objekteissa taustan on pakko olla läpinäkyvä. Vaikka myös GIF taltioi alpha-kanavan, se tekee sen paljon huonommin kuin PNG, eikä taltioi puoliksi läpinäkyviä objekteja (Gordon 2011, viitattu 18.1.2016). Kuvio 1 näyttää tästä käytännön esimerkin, jossa GIF ei toista puoliksi läpinäkyvää gradienttia läpinäkyvällä pohjalla. Vektorimuotoisia objekteja muuntaessa rasteritiedostoiksi PNG on paras valinta, vaikka ongelmia voi tulla tehdessä pikselin tarkkoja reunoja.

KUVIO 1. Vasemmalla GIF-tiedosto ja oikealla PNG.

4 2D-MOBIILIPELIGRAFIIKAN OPTIMOINTITEKNIIKOITA

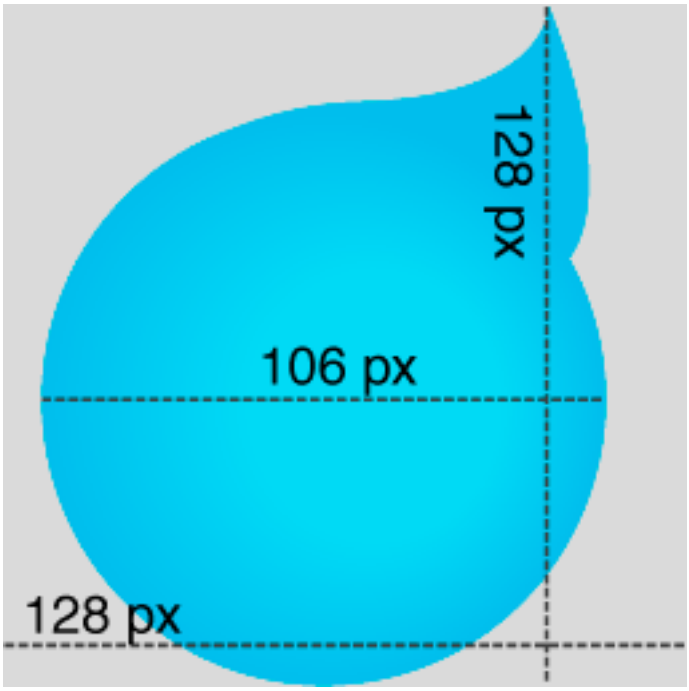
4.1 Sanastoa

2D-grafiikka (engl. 2-dimensional)	Tarkoittaa sekä kaksiulotteista kuvaa, että sen tuottamiseen tarkoitettua tekniikkaa. 2D-grafiikka sisältää kaksi ulottuvuutta, eli pituuden ja leveyden.
Sprite	Kuva, jolla ei ole taustaa.
Spritesheet	Usean spriten sisältävä kuva, jossa ei ole taustaa.
Free-to-play	Peli, jonka voi ladata ja aloittaa ilmaiseksi. Saattaa sisältää mikromaksuja.
Premium	Peli, jonka lataaminen maksaa.
Partikkeli-efekti	Pienistä kuvista koostuva liikkuva elementti, jota kontrolloidaan partikkelisysteemin avulla pelimoottorissa. Esimerkiksi savupilvi, jossa yksi savuhaituva sprite muodostaa suuremman pilven.
Low-end	Tuotesarjansa halvimpaan päähän kuuluva tuote.
High-end	Tuotesarjansa kalleimpaan päähän kuuluva tuote.

4.2 Power of two –sääntö

Yksi tärkeimmistä optimointitekniikoista on nimeltään kahden sääntö (englanniksi power of two). Sääntöä käytetään sekä mobiili- että PC-peleissä, sillä kaikki tietokoneet, konsolit, älypuhelimet ja pelimoottorit käyttävät tätä samaa tekniikkaa prosessoidessaan tietoa. Koska dataa ei prosessoida kerralla vaan osissa, on koneen helpompi prosessoida kahdella jaollisia osia. Kaiken graafisen materiaalin, mitä viedään peliin, tulee siis olla kooltaan kahdella jaettavissa. Kuvan ei välttämättä tarvitse olla neliö, eli sivut voivat olla eri pituiset, kunhan molemmat sivut ovat kahdella jaollisia.

Jos tuotu data ei ole kahdella jaollinen, pelimoottori ”korjaa” koon aiheuttaen turhaa prosessointitehon kuluttamista ja kuvan laadun huonontumista (Katsbits 2015, viitattu 15.10.2015). Vaikka tuodun graafisen elementin oikea koko ei olisikaan jaettavissa kahdella, pitää se silti viedä peliin ”power of two” -säännön mukaisessa koossa. Esimerkkinä tästä on kuvio 2, jossa hahmon koko on 128x106 pikseliä ja spriten koko on 128x128 pikseliä.



KUVIO 2. Havainnointi hahmon koosta suhteessa spriten kokoon.

Liian suurien objektien teko ei ole suositeltavaa, sillä ne kasvattavat pelin kokoa. Graafikon on mietittävä yhdessä ohjelmoijan kanssa objektien koko ennen niiden viemistä Unityyn. Helppo tapa selvittää objektin koko on miettiä, minkäkokoinen se on suhteessa muihin objekteihin. Esimerkiksi State of Matterin päähahmot ovat pituudeltaan 128 pikseliä, joten kaikki muut objektit voi miettiä suhteessa hahmon kokoon. Myös spriten kokoa täytyy miettiä. Pientä objektia ei kannata laittaa suurelle spritelle, vaan miettiä mahdollisimman pieni koko, sillä läpinäkyvä pintakin vie tilaa.

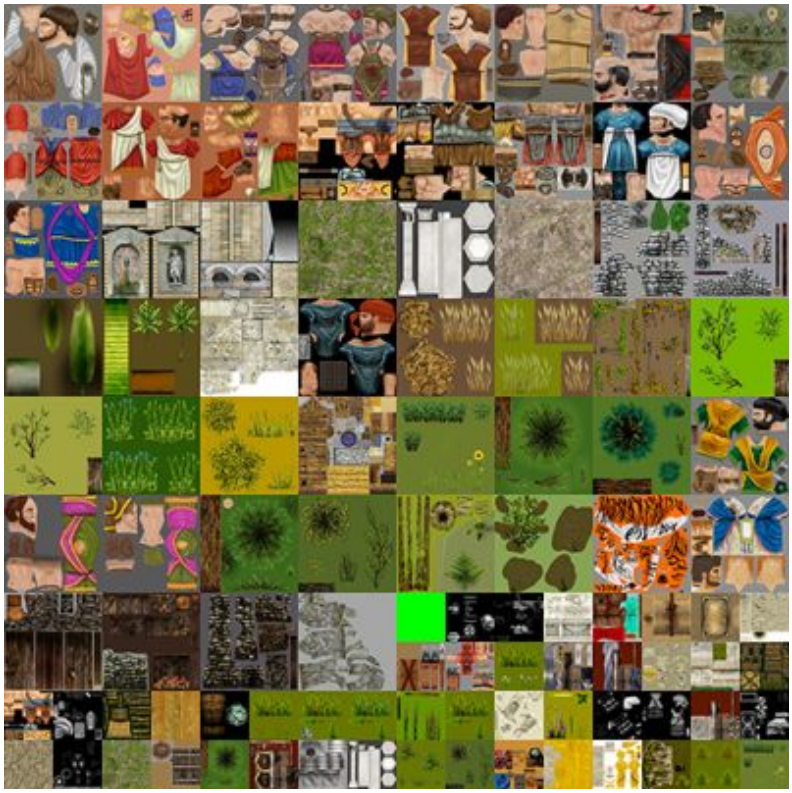
Objektin kokoa ei kannata muuttaa Unity-pelimoottorin sisällä, vaan viedä se pelimoottoriin siinä koossa kuin sitä pelissä käytetään. Skaalaaminen piirto-ohjelmassa on parempi tapa sekä kuvan laadun suhteen, että optimoinnin takia, sillä vääränkokoiset objektit voivat helposti kertyä ja viedä

turhaa tilaa. Varsinkin vektoripohjaisissa ohjelmissa, missä vektorigrafiikkaa voi skaalata loputtomiin ilman laadun huonontumista, on erittäin helppoa muuttaa kuvan kokoa. Jos työstää esimerkiksi Adobe Photoshop -ohjelmalla rasterigrafiikkaa, kannattaa työskennellä liian suuren kuvan kanssa ja skaalata se alaspäin todelliseen kokoon ennen Unityyn viemistä. Rasterigrafiikkaa voi skaalata alaspäin mutta ylöspäin skaalaaminen ei ole suositeltavaa laadun heikkenemisen vuoksi.

4.3 Tekstuuriatlas

Jokainen piirrettävä objekti pelimoottorissa aiheuttaa prosessin nimeltä piirtokutsu (englanniksi drawcall). Piirtokutsu on vaihe, jossa pelimoottori kutsuu objektin ja sen tekstuurit piirrettäväksi grafiikan rajapinnasta yksittäisen kuvaruudun aikana. Piirtokutsut ovat paljon prosessointitehoja vieviä vaiheita ja niiden vähentäminen parantaa pelin optimointia (Unity 3D 2015c, viitattu 19.10.2015).

Yksi tapa vähentää piirtokutsuja on laittaa kaikki samantyyppiset kuvat samaan atlakseen. Atlas eli pelimoottorin kuvakartta on yksi iso kuva, joka sisältää useita pienempiä kuvia. Atlaksen etuna on se, että niiden käyttö aiheuttaa vain yhden piirtokutsun. Atlaksia voi olla useampia erityyppisille grafiikoille ja tekstuureille. Vaikka atlaksien käyttö vähentää muistin kuormitusta, liian usean suureen atlaksen tekeminen vie muistia ja sekoittaa myös pelintekijää. Esimerkkinä tekstuuriatlaksesta on alla oleva kuvio 3.



KUVIO 3. Esimerkki tekstuuriatlaksesta.

4.4 Grafiikoiden uudelleenkäyttö

Yksi helpoimmista tavoista optimoida mobiilipeliä on grafiikoiden uusiokäyttö (Unity 3D 2015d, viitattu 18.10.2015). Sekä artistin että ohjelmoinnin kannalta on helpompaa, jos samantyyppisiä grafiikoita voi kierrättää uudelleen pelin sisällä eikä tehdä uutta samantyyppistä elementtiä. Kierrätys kannattaa pitää mielessä jo peliä suunnitellessa. Jos grafiikkaa ei voi käyttää uudelleen, voi pohtia kuinka tärkeä elementti on pelin kannalta. Toissijaisista, vain esteettisesti tärkeistä elementeistä kannattanee luopua, jos uudelleen käytettävyyttä ei ole ja elementin tarve pelissä ei ole suuri. Toisaalta uudelleenkäytettävyyttä ei tule pitää ensisijaisena prioriteettina, jos sen eteen joutuu uhraamaan pelin kannalta tärkeitä ja mielenkiintoisia elementtejä. Esimerkiksi klassisessa 2D-tasohyppely pelissä käytettävät tiilimäiset maapalaset (englanniksi tileset) kannattaa suunnitella mahdollisimman ekologisesti niin, että samaa maapalaserää voi käyttää läpi koko pelin tai ainakin useammassa kentässä. Saumattoman tiilimäisen maapalaserän luomisen hyöty on se, että pelikentät voi suoraan koota näitä pieniä maapaloja käyttäen luoden hauskaa ja mielenkiintoista pelattavaa vähällä vaivalla. Myös kenttien muuttaminen on paljon helpompaa tällä menetelmällä.

4.5 Valot ja varjot

Pelin valot ja varjokohdat on mahdollista tuottaa Unity-pelimoottorin sisällä. Optimoinnin kannalta on kuitenkin parempi, jos valot saa piirrettyä itse kuviin. Valokohtien piirtäminen grafiikkoihin on erittäin suotavaa, sillä se säästää hyvin paljon prosessointitehoa. Varsinkin 2D-grafiikassa artistin on helppo määrittää ja piirtää valot sekä varjot, sillä erityisesti sarjakuvamaisessa tyylissä valotus saa olla kaukana realistisesta.

Sama ongelma pätee myös muihin Unityn sisäisiin efekteihin. Esimerkiksi kimallus, sumu tai salamointi on paljon parempi toteuttaa piirrettyinä animaatioina kuin Unityn efekteinä. Upeat valot, varjostukset ja efektit ovat kustannuksiltaan liian kalliita kevyille mobiilipeleille. Taitava artisti saa ne kuitenkin sisällytettyä grafiikkoihin niin, ettei pelaaja edes huomaa niiden poissaoloa. (Unity 3D 2015b, viitattu 18.10.2015.)

4.6 Animaatio

Useimmat mobiilipelit vaativat ainakin jonkin verran animointia. Animointi itsessään on jo aivan oma taidemuotonsa, joka vaatii omanlaistaan ammattitaitoa. Jos pienessä peliyrityksessä ammattimaisen animaattorin palkkaamiseen ei ole varaa, kannattaa peliä suunnitellessa miettiä, miten paljon animointia peli kaipaa. Mielenkiintoisen ja hauskan mobiilipelin voi tehdä myös hyvin vähällä animoimisella. Esimerkiksi State of Matter -mobiilipelissä robottiviholisilla on jalkojen tilalla renkaat tai telaketjut siksi, että ne olisi paljon helpompi animoida.

Animaation tekemiseen on olemassa monenmoisia eri ohjelmia. Useimmissa 3D-mallinnusohjelmissa on lisäksi myös mahdollisuus tehdä animaatiota, esimerkiksi Blender ja 3DS Max-ohjelmissa. 2D-artistit voivat käyttää esimerkiksi Adoben Flash -animointiohjelmaa, joka kommunikoi hyvin Adoben muiden piirto-ohjelmien kanssa, tai käyttää jotakin lukuisista Internetistä löytyvistä ilmaisohjelmista. Itse olen kuitenkin käyttänyt mobiilipelien 2D-animaatioon Unityn omaa animaatiotyökalua. Unityn animaatiotyökalu ei ehkä ole paras mahdollinen monimutkaisten animaatioiden tekemiseen, mutta mobiilipeleihin vaadittaviin yksinkertaisiin animaatioihin se on vallan mainio työkalu. Etuna tässä menetelmässä on se, että animaatiot tehdään yhden spritesheetin pohjalta, joka leikataan ja kootaan Unityssä. Kuvio 4 on esimerkki spritesheetistä, jossa jokainen kuvio on sa-

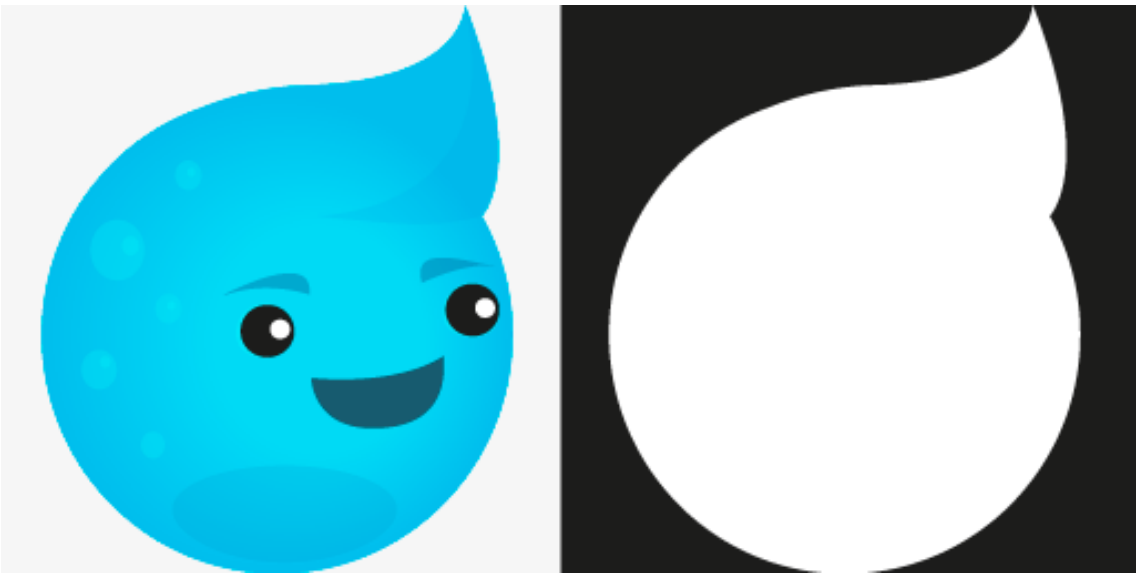
malla spritesheetillä mahdollisimman lähekkäin mutta silti niin erillään, että ne saa leikattua animoitaviksi objekteiksi. Samasta spritesheetistä työstettiin robotin kävely-, hyökkäys- ja kuolinanimaatiot. Jos animaatioita työstää spritesheet-menetelmään pohjautuvalla animointiohjelmalla, Unityyn pitää tuoda pelkästään yhteen animaatioon useita spritesheetejä, mikä kasvattaa pelin kokoa ja vie prosessointitehoa. Unityn sisäisellä animaatiotyökalulla kaikki animaatiot käyttävät säästeliäästi samaa spritesheetiä, ja samasta spritesheetistä voi tehdä kaikki hahmolle tarvittavat animaatiot.

KUVIO 4. Animointiin tarkoitettu spritesheet.

4.7 Värit ja läpinäkyvyys

Tietokonegrafiikassa on neljä värikanavaa: kolme RGB-väriä ja alphakanava eli läpinäkyvyys. Alphakanavalla ei periaatteessa ole pikseleitä, vaan se muokkaa alla olevia pikseleitä ja vaikuttaa niiden läpinäkyvyysasteeseen. Alphakanava hallitsee sekä läpinäkyvyyttä, että osittaista läpinäkyvyyttä. Mobiilipeligrafiikassa yleisesti käytetty sprite on kuvaobjekti, joka koostuu näkyvästä ja läpinäkyvästä osasta. (Smith 1995, viitattu 10.1.2015.) Kuvio 5 näyttää alpha-kanavan paikan spritesheetissä.

Koska alphakanava täyttää pikselit useammin kuin kerran, vie sen käyttö enemmän prosessointitehoa. Erityisesti osittaisen läpinäkyvyyden käyttö on ongelma suurissa objekteissa, jotka täyttävät koko ruudun, erityisesti jos läpinäkyviä objekteja on kerroksittain. Päällekkäinen osittainen läpinäkyvyys ja suuret objektit heikentävät kuvataajuutta. Parasta on siis käyttää osittaista läpinäkyvyyttä säästeliäästi ja välttää kerrostamista. (Robot Invader 2014, viitattu 10.1.2015.)



KUVIO 5. Vasemmalla tavallinen sprite, oikealla alpha-kanava mustalla.

5 OPTIMOINTI AMMATTILAISPELIGRAAFIKOIDEN NÄKÖKULMASTA

Haastattelin sähköpostin avulla kolmea oululaista ammattipeligraafikkoa. Tuomas Soukka on työskennellyt Fingersoftilla peligraafikkona vuodesta 2013. Hän on ollut mukana muun muassa Hill Climb Racing- ja I Hate Fish -pelien kehityksessä. Hänen työnkuvaansa kuuluu grafiikan tekeminen tuotannon jokaiseen osa-alueeseen, eli esimerkiksi konseptointi, peligrafiikka, animaatio, videotuotanto, markkinointimateriaali, asiakaskontaktit ja osallistuminen pelisuunnitteluun. Antti Miettinen on vuonna 2013 perustetun Bad Crane -mobiilipeliyrityksen päägraafikko. Hänellä on vuosien kokemus pelialasta ja mobiiliohjelmista. Miettinen on ollut tekemässä Bad Cranen julkaisemia Zapresso ja RGB Express mobiilipelejä. Hän vastaa pelien visuaalisesta puolesta ja lisäksi auttaa myös pelien kokonaisuuden suunnittelussa, toteutuksessa ja testauksessa. Marko Sellman on vuonna 2013 perustetun Meizi Games:in Art Director, mutta pelialalla hän on ollut jo vuodesta 2007. Aluksi hän teki web-selaimella pelattavia Flash-pelejä, mutta nykyisin pääasiassa mobiilipelejä. Hänellä on kokemusta graafisen suunnittelun lisäksi ohjelmoinnista. Sellman on toiminut graafisena suunnittelijana muun muassa Hertan Maailma, Trivian, Hamsterscape ja Freak Circus Racing projekteissa. Hänen työnkuvaansa kuuluu visuaalisen suunnittelun lisäksi peligrafiikan tekninen toteutus.

Kaikki haastateltavat kertovat käyttävänsä grafiikantuotantoon Adoben ohjelmia, erityisesti Photoshop- ja Illustrator-ohjelmia. Soukka käyttää pääasiallisesti Photoshopia, ja kertoo sen vahvuuden olevan ohjelman monipuolisuudessa. Hänen mukaan lähes kaikki 2D-grafiikka kääntyy ohjelmalla, ja on tehokasta, että kaikki pelin graafiset elementit ovat yhteensopivia toistensa kanssa. Photoshopilla voi tuottaa grafiikkaa aina konseptoinnista lopulliseen grafiikkaan ja markkinointimateriaaliin asti, Soukka kertoo. Miettinen käyttää tasapuolisesti Illustratoria ja Photoshopia. Pääsyyksi hän mainitsee sen, että ne ovat käytännössä alan standardeja ja hän on käyttänyt niitä niin kauan, että komennot tulevat luonnostaan. Myös Sellman korostaa Adoben ohjelmien olevan alalla normi. Hänen aloittaessaan pelialalla Adobe ohjelmistot olivat yleisesti käytössä, ja sekä ohjelmistotuki että yhteensopivuus on toimivaa.

Pelimoottoreista Soukan tiimi käyttää Unityä prototypointiin ja Cocos2D -ohjelmaa syvällisempiin tuotantoihin. Unityn vahvuus on Soukan mukaan se, että sitä voi käyttää ilman koodaustaustaa. Unityn hyvistä puolista hän listaa sen helppouden ja nopeuden. Hänen mukaansa Unityn käyttö vähentää huomattavasti projektin kehitysaikaa, kun graafikot voivat itsenäisesti tuottaa ja muokata

sisältöä. Cocos2D on taas haastavampi ympäristö graafikolle, vaikka se tarjoaa Cocos Studio -ympäristön, jossa on visuaalinen editori hieman Unityn tapaan. Vaikka editori on kuitenkin vielä paljon suppeampi ja vaikeaselkoisempi, sen edut ovat suorituskyvyssä ja muokattavuudessa, selostaa Soukka. Miettisen tiimi käyttää pelintekoon Corona SDK -pelimoottoria, jolla heidän kaikki pelinsä on toteutettu. Sellman kertoo ennen käyttäneensä Flashia ja nykyään Unityä. Sellman sanoo käyttävänsä Unityä samoista syistä kuin käyttäessään Adoben ohjelmistoja, sillä ne ovat alalla yleisesti käytössä, ohjelmistotuki ja yhteensopivuus ovat toimivaa sekä erityisesti pelikehityksessä Flash oli sen ajan johtavin ja monipuolisin web-teknologia.

Haastateltavat ovat yhtä mieltä graafisen optimoinnin tärkeydestä ja graafikon suuresta vaikutuksesta optimointiin. Mobiilipelien suorituskyvyn kannalta on tärkeää, että kaikki graafiset elementit ovat optimoituja, korostaa Soukka. Tämä vaatii hänen mukaansa tarkkuutta ja suunnitelmallisuutta assettien tuotannossa alusta loppuun saakka. Soukka listaa suorituskykyyn vaikuttavia tekijöitä: tekstuurien koot, 3D-elementtien monimutkaisuus, partikkeli-efektien määrä ja uniikkien tekstuurien määrä. Miettisen historia optimoinnin parissa alkoi demoscenejen parissa aikana, jolloin koneet olivat tehottomia ja kaikki piti optimoida hyvin tarkasti. Silloin hän pikselöi kuvia demopartyjen kisoihin ja teki grafiikkaa eri demoihin. Tämän takia hän korostaa optimoinnin merkitystä myös nykypäivänä ja haluaa, että heidän pelinsä toimivat myös hieman vanhemmilla laitteilla. Tietenkin merkitys vähenee, mitä tehokkaammaksi mobiililaitteet tulevat, Miettinen tuumaa. Myös Sellman korostaa graafikon vaikutusta pelin optimointiin. Hänen kertoo grafiikan olevan usein se tekijä, joka syö mobiililaitteiden resursseja kaikkein eniten. Graafikko voi omillaan toimillaan vaikuttaa pelin suorituskykyyn positiivisesti, jopa niin, ettei pelin ulkoasu kärsi, vaikka tekstuureissa olisikin vähemmän pikseleitä ja 3D-mallit olisivat yksinkertaisempia. Graafikon taiteellinen silmä on monesti avainasemassa näissä tilanteissa, korostaa Sellman. Grafiikalla voi jopa joskus korvata raskasta koodia, Sellman neuvoo, esimerkiksi paljon laskentatehoa vaativat fysikkamallinnukset voidaan toteuttaa grafiikan avulla ”huijaamalla”. Esimerkkinä hän mainitsee auton renkaat, jotka saadaan pyörimään kevyemmin animaation avulla kuin, että koodin fysiikkalaskennat pyörittäisivät niitä.

Haastateltavat painottavat projektin eri vaiheita, joka on heidän mukaansa optimoinnin kannalta tärkein. Kaikki kuitenkin korostavat erityisesti projektin loppuvaiheen tapahtumia. Soukan mukaan tärkeintä optimoinnissa on valmiiden grafiikoiden tuotanto ja pelimoottoriin tuominen. Hänen mielestään huolellinen suunnittelu on tärkeää, mutta vasta kun peliä voidaan testata, nähdään sen suorituskyky. Miettinen taas optimoi grafiikkaa kaikissa projektin vaiheissa, mutta painottaa erityi-

sesti lopetusvaihetta. Silloin hän käy kaiken läpi ja optimoi sieltä, missä sitä vielä tarvitaan, esimerkiksi yhdistelemällä spritejä samoihin kuviin ja koettaen näin vähentää muistinkäyttöä ja latausaikoja. Sellman taas korostaa jokaisen vaiheen merkitystä optimointiin, vaikka korostaakin erityisesti testaamisen merkitystä. Hänen mukaansa joka vaiheessa voidaan vaikuttaa optimointiin ratkaisevastikin, mutta testauksissa voidaan löytää pelistä ongelmakohtia, joita taas optimoinnilla voidaan mahdollisesti korjata. Testauksia kuuluisi myös järjestää projektin aikana useita, ja jopa suunnitelmia ja konsepteja voidaan testauttaa, hän neuvoo. Optimoinnin huomioimisesta voi Sellmanin mukaan olla jopa haittaa suunnitteluvaiheessa, sillä se voi rajoittaa luovaa ajattelua liikaakin. Tosin tämä ei hänen mukaansa tarkoita sitä, että optimointia ei saisi harrastaa suunnitteluvaiheessa, sillä jo varhaisessa vaiheessa aloitettu optimointi voi vähentää kalliita työtunteja projektin loppuvaiheesta. Valitettavan monesti optimointia aletaan tekemään vasta projektin loppuvaiheessa, jolloin se voi pahimmassa tapauksessa olla liian myöhäistä, minkä seurauksena koko projekti voidaan joutua tekemään jopa alusta asti uudestaan, Sellman kuvailee.

Optimoinnin teknisiä keinoja haastateltavat nostavat esiin useita. Soukka listaa hänen mukaansa parhaita menetelmiä: tekstuurien koon optimointi, atlas-tekstuurit, tekstuurien fiksu uudelleenkäyttö mahdollisimman usein ja modulaariset tekstuurit, josta esimerkiksi hän kertoo käyttöliittymän painikkeet jotka koostuvat niin sanotusta 9-piece järjestelmästä, jossa painike koostuu reunapaloista ja täyttöväristä. Hän on myös huomannut, että yleensä mobiilipeleihin päätyy hieman liian suuria tekstuureja, ja monesti lopputulos näyttää samalta laitteen ruudulla, vaikka tekstuureja pienentäisi hyvinkin rajusti. Juuri tästä syystä on tärkeää testata grafiikkaa itse mobiililaitteilla tietokoneen näytön sijaan, Soukka korostaa. Hän kertoo tekstuurien optimoinnissa olevan monta vaihetta mutta itse tekstuurin koko on isossa roolissa paketin koon pienentämisessä ja siinä, miten kauan tekstuureja joudutaan lataamaan pelin muistiin. Optimointiin vaikuttavat Soukan mukaan myös se, että käytetäänkö yhtä tekstuuria fiksusti useassa eri paikassa, käytetäänkö atlas-tekstuureja ja miten monimutkainen itse tekstuuri on. Soukka myös korostaa animaatioiden tuottamista bone-animaationa, jossa animoitavan objektin sisälle luodaan hierarkisesti toimiva luurankomainen runko, perinteisen "sprite-sheet" menetelmän sijaan.

Miettinen taas korostaa "power of two" -kokoja ja objektien koon määrittämisen tärkeyttä. Hän ei käytä liian isoja kuvia tilanteissa missä niitä ei välttämättä tarvita (esim. nopea liike). Lisäksi hän poistaa kuvien metatiedon eli tiedoston liitännäistiedon ja mahdollisesti tarpeettomat kanavat kuten läpinäkyvyyden. 8-bittiskuvissa hän kertoo käyttävänsä vain tarpeellisen värimäärän. Lisäksi hän pitää ylipäätään grafiikan mahdollisimman simppelinä ja helposti hahmotettavana.

Sellman antaa enemmänkin yleisiä neuvoja kuin yksityiskohtaisia menetelmiä, sillä hänen mukaansa joskus jossain toisessa projektissa toimivat optimointimenetelmät eivät toimikaan muissa peliprojekteissa. Ylitse muiden keinojen hän korostaa testausta, sillä sen avulla voidaan löytää optimoinnin kannalta parhaimmat mahdolliset menetelmät ja pahimmat ongelmakohdat, joita voidaan yrittää korjata uusilla optimointimenetelmillä. Toisena hyvänä keinona Sellman pitää grafiikoiden kierrättämistä, eli toisin sanoen pienien palasien käyttämistä uudestaan jonkin isomman luomiseen. Toteutuskeinoina hän listaa maailmojen/kenttien rakentamisen tilesettien avulla, 3D-malleissa materiaalien kierrättämisen useimmissa malleissa, UV-mapin avulla tekstuurien yhdistämisen samaan atlakseen, ja spritejen yhdistämisen samaan spritesheetiin. Sellmanin mukaan kierrättämisellä usein tavoitellaan vähempää muistin käyttöä, mutta esimerkiksi Unityssä spritesheetien ja atlaksien käyttö vähentää myös prosesseja, jotka vaativat näytönohjaimelta laskentatehoja.

Sekä Sellman että Soukka korostavat graafikkojen ja ohjelmoijien hyvän yhteistyön vaikutusta optimointiin. Joskus vastaan voi tulla tilanteita, joissa grafiikan toteuttaminen koodilla on mobiililaitteiden resursseja säästävämpi vaihtoehto, toteaa Sellman. Soukankin mukaan kaiken grafiikan optimointi vaatii graafikon ja ohjelmoijan yhteistyötä, sillä suurin osa optimointia tapahtuu koodissa. Vaikka graafikon tehtävä on toimittaa mahdollisimman optimoidut assetit peliprojektiin, pelimoottorilla työskentely on huomattavasti tehokkaampaa, jos graafikko itse pystyy luomaan peliympäristöjen graafisen puolen, Soukka kertoo. Jokaisen pelifirman kannattaisi sopeuttaa graafikot osaksi pelimoottorin käyttämistä siinä määrin, että he pystyisivät itse asettelemaan grafiikat peliin niin pitkälle kuin suinkin mahdollista, sillä kun graafikko näkee reaaliajassa, miltä grafiikat näyttävät, niitä voi optimoimaan nopeammin, mikä säästää ohjelmoijien aikaa ja tuottaa myös parhaannäköisen lopputuloksen, hän korostaa.

Haastateltavat antavat kukin neuvoja pelialasta kiinnostuneille graafikoille. Sen lisäksi, että kannattaa opiskella itsenäisesti tehokasta grafiikantuotantoa, on elintärkeää tehdä yhteistyötä muidenkin kuin toisten graafikkojen kanssa, Soukka vinkkaa. Hän kertoo ohjelmoijien tietävän usein optimoinnista enemmän ja tiiviin yhteistyön tuottavan molemmille osapuolille uutta osaamista. Kannattaa myös tutustua pelimoottoreihin, Soukka opastaa, esimerkiksi Unityyn ja Unreal Engineen, sillä ne ovat ilmaisia käyttää ja niissä voi itsenäisesti opiskella, miten peli rakennetaan varsinkin grafiikan osalta. Soukka kannustaa kokeilemaan kaikkia grafiikantuotannon osa-alueita, sillä jokainen todennäköisesti erikoistuu johonkin suppeampaan tehtävään, mutta hänen mukaansa on hyödyksi

tuntea tuotannon kaikki vaiheet. Miettisen neuvot keskittyvät grafiikkaan. Hänen mukaansa on tärkeää, että kuvien ja spritejen resoluutio pysyy suhteessa toisiinsa suurin piirtein samana koko pelissä. Liian usein huomaa pelejä, joissa osa elementeistä on selvästi tarkempia kuin toiset, hän kertoo. Sellman antaa kolme vinkkiä. Ensimmäiseksi hän korostaa ohjelmoijien ja graafikkojen yhteistyön merkitystä. Toisena vinkkinä hän painottaa testailun ja prototyyppien tekemisen merkitystä, ja neuvoo kokeilemaan uusia menetelmiä ja tutoriaalien pohjalta tekemistä sekä koodaamista. Kolmanneksi hän suosittelee kokeilemaan ja osallistumaan erilaisiin pikaprojekteihin ja -tapahtumiin, sillä hänen mielestään ne ovat hyvin opettavaisia: niissä on mahdollisuus tutustua uusiin ihmisiin ja toteutusmenetelmiin, ne eivät yleensä vie paljoakaan aikaa, ja ehkä tärkeimpänä, niissä on mahdollisuus kokea onnistumisen sekä epäonnistumisen tunteita, minkä takia erilaiset tapahtumat ovat hyviä keinoja kartuttaa kokemusta.

6 STATE OF MATTER -MOBIILIPELI

Aloitin State of Matter -mobiilipelin parissa työskentelyn tammikuussa 2015. State of Matter on pulmanratkaisu tasohyppelypeli, jonka pelimekaniikka perustuu hahmon olomuotojen vaihtamiseen. Pelistä oli olemassa jo demo, mutta pelisuunnittelijan vaihtuessa peli suunniteltiin uusiksi, ja myös grafiikat haluttiin uusia kokonaan. En ollut ennen suunnitellut mobiilipelejä, joten kokonaisen pelin grafiikoiden osittainen suunnittelu oli aivan uusi aluevaltaus minulle. Suunnittelin grafiikoita vanhoihin grafiikkoihin ja uuteen game design -dokumenttiin perustuen. Tässä kappaleessa on tietoa sekä State of Matterin peligrafiikan synnystä että käyttämistäni graafisen optimoinnin keinoista. Useimmat käyttämäni tekniikat löytyvät tarkemmin edellisestä kappaleesta, mutta tässä kappaleessa niistä on syvempiä esimerkkejä ja uudenlaisia käyttömahdollisuuksia.

6.1 Hahmot

Aloitin pelin grafiikoiden uudistamisen hahmoista. Pelin kolme päähahmoa ovat veden kolme eri olomuotoa: Whalter- nestemäinen, Whisper- höyry ja Phrost- jää. Hahmojen vanha design kaipasi hiomista ja persoonallisuutta. Suunnittelemani hahmojen ulkoasut löytyvät kuviosta 6. Aloitin nestemäisestä Whalterista, joka on pelin ensimmäinen pelattava hahmo eli pelin perushahmo. Suunnittelin hahmosta useita eri versioita ennen kuin olin tyytyväinen. Whalter, joka oli ennen pelkkä pyöreä pallo, sai komean pystyotsatukan tuomaan luonnetta. Vartalo pysyi pyöreänä mutta läpinäkyvämmän oloisena sekä kuplilla höystettynä tuomaan vesimäistä tunnelmaa hahmoon. Kasvot olivat yksinkertaiset: kaksi pyöreää silmää, kulmakarvat ja suuri hymyilevä suu. Väreiksi valitsin vahvan vetisen sinisen sävyjä.

Whalterin jälkeen suunnittelin Whisperin. Whisperin eli höyryhahmon vanha ulkonäkö oli hyvin pilvimäinen ja hahmon persoona oli kolmikosta söpöin ja tyttömäisin. Halusin säilyttää pilvimäisen ulkonäön mutta aivan pilven muotoinen vartalo ei mielestäni toiminut. Hahmolle muotoutui hahmottelun jälkeen pyöreä päälaki ja pilvimäinen helma. Päätä koristaa kiemurainen etutukka ja muuten samantyyppiset kasvot kuin Whalterilla, mutta suu on hieman suloisempi, ja sen muoto perustuu : 3 -hymiöön. Värimaailma on sävyiltään valkoista ja vaaleansinistä.

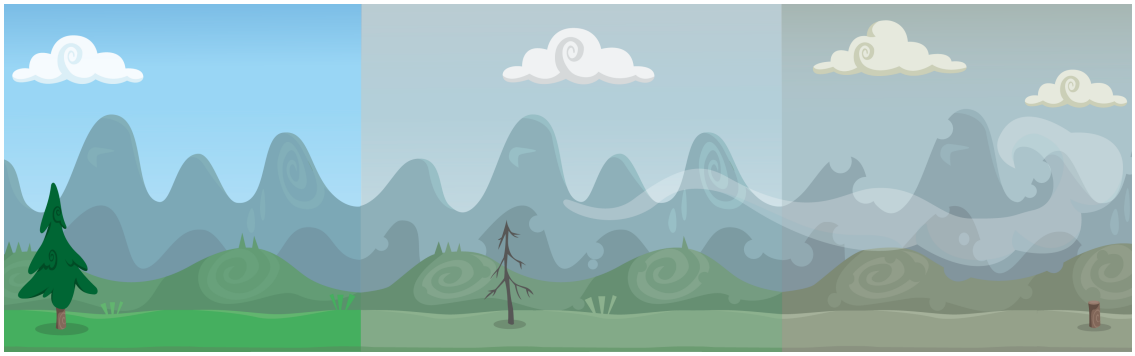
Viimeisenä suunnittelin Phrostin. Phrost oli jo aiemmin jääpalan muotoinen, ja myötäilin muodolla vanhaa designia. Vartaloon tuli kuitenkin lisää läpinäkyvyyttä ja kolmiulotteisuutta. Koska kahdella aiemmalla hahmolla oli veikeät hiukset, yritin suunnitella myös Phrostille hiustyyliä. Harmikseni Phrostin irokeesit eivät kuitenkaan näyttäneet hyvältä, joten hahmo sai vahvat kulmakarvat ja suuren arven korostamaan hahmon kovuutta ja vahvuutta. Suusta muotoutui kapea mutta silti hymyilevä. Väritys Phrostilla on valkoinen ja harmaan sinertävä.

Hahmoille ei tehty animaatiota. Hahmojen uniikit liikkumistyylit tehtiin ohjelmoinnilla Unityssä. Whalter liikkuu elastisesti pomppien, Whisper leijuu ja Phrost liukuu nopeasti tasaisella pinnalla, mutta ei voi hypätä. Myöhemmin Whisperille lisättiin partikkeliefektillä höyryä sen ruumiin ja lattiatason väliin, ja sekä Whisperin että Phrostin kasvot vaihtavat ilmettä tietyissä tilanteissa. Peliprojektin loppuvaiheessa hahmot saivat myös power up -efektiversiot, jossa ne saippuaa syödessään muuttuvat hetkellisesti vaaleanpunaiseksi ja ovat tuhoutumattomia.

KUVIO 6. State of Matterin hahmot. Vasemmalta oikealle: Whisper, Whalter ja Phrost.

6.2 Taustat

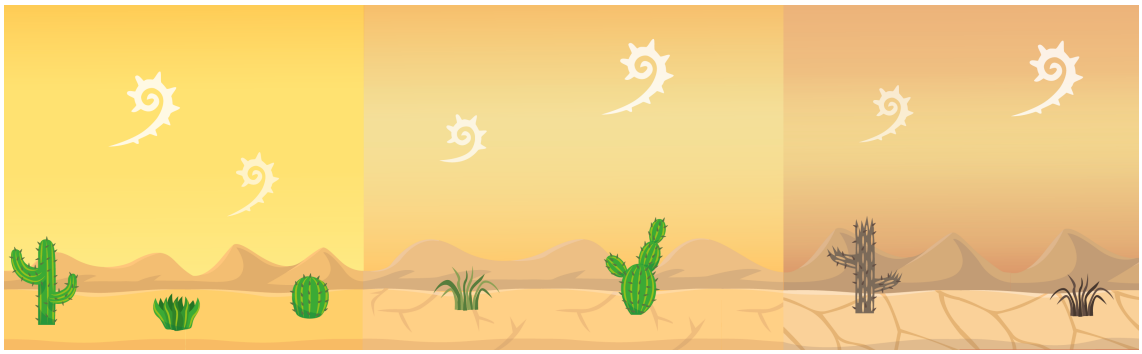
Pelissä on tällä hetkellä kolme erilaista planeettaa. Jokaisessa planeetassa on kolme eri tasoa: normaali, saastunut ja tuhoutunut. Taustojen avulla piti tuoda esille luonnon saastuminen ja tuhoutuminen. Aloitin suunnittelun ensimmäisestä planeetasta, joka oli hyvin perinteinen avoin ruohomaisema. Hain ideoita muista peleistä, ja erityisesti minua inspiroivat Angry Birdsien yksinkertaiset mutta veikeät taustat. Maisemalle suunnittelin ruohoa, mäkeä ja vuoria käyttäen pehmeitä muotoja ja kiemuroita. Saastuneemmat versiot muotoituivat normaalin version pohjalta värejä vaihtaen ja lisäten muutamia yksityiskohta, kuten puunrankoja ja usvaa. Kuviosta 7 näkee ensimmäisen planeetan taustat ja niiden erot.



KUVIO 7. Ensimmäisen planeetan taustat.

Aavikkoplaneetta on pelin toinen maailma. Planeetta on hehkuvan kuuma, ja pelaajan täytyy välillä vilvoitella varjossa tai vedessä selviytyäkseen. Tämän täytyi tulla designissa ilmi. Suunnittelin planeetalle melko yksinkertaisen taustan, jossa on hiekkaa ja dyynejä keltaisen sävyissä. Pilviä on vähän, mutta ne jatkavat ensimmäisellä planeetalla toistuvaa kiemuratyyliä. Kasvistoa on melko runsaasti ja lisäksi suunnittelin useita planeetan pelimekaniikassa tärkeänä osana olevia kiviä. Mietin mitä aavikolle tapahtuu ilmastonmuutoksen seurauksena suunnitellessani saastuneita taustoja. Ensimmäisellä planeetalla luonnon vehreys katoaa ja maisema muuttuu harmaaksi ja usvaiseksi, kun taas aavikolla lämpötila nousee ja viimeisetkin elämän merkit häipyvät. Aavikon ensimmäisellä

tasolla olevat kaktukset ja muut sitkeät kasvit ruskistuvat ja surkastuvat. Maa halkeilee ja värimaisema muuttuu keltaisesta punertavaksi, kuten kuviosta 8 voi huomata.

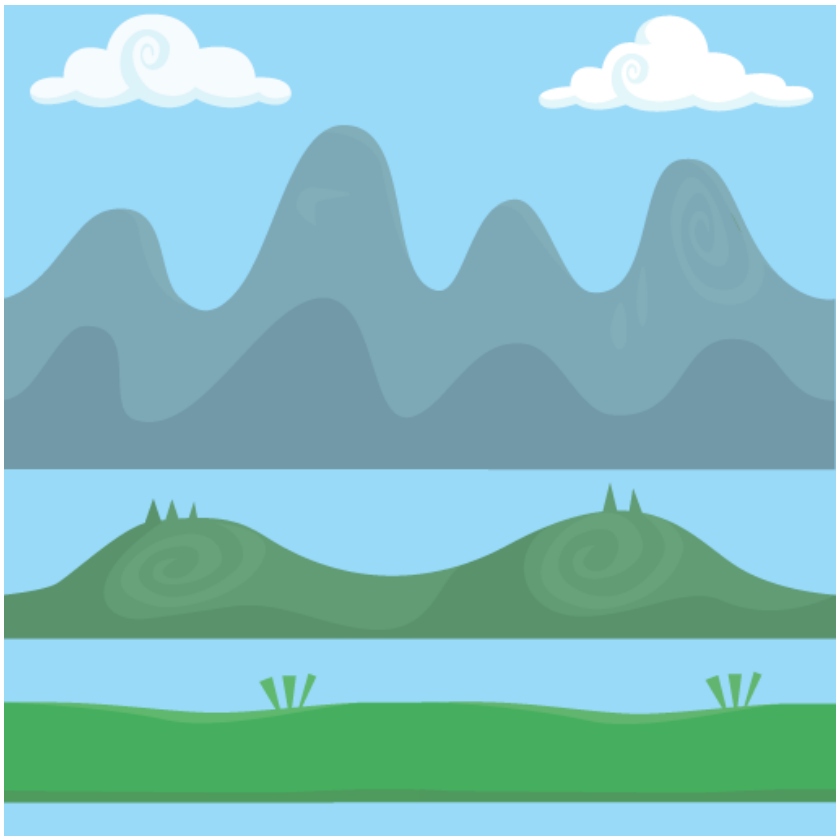


KUVIO 8. Aavikkoplaneetan taustat.

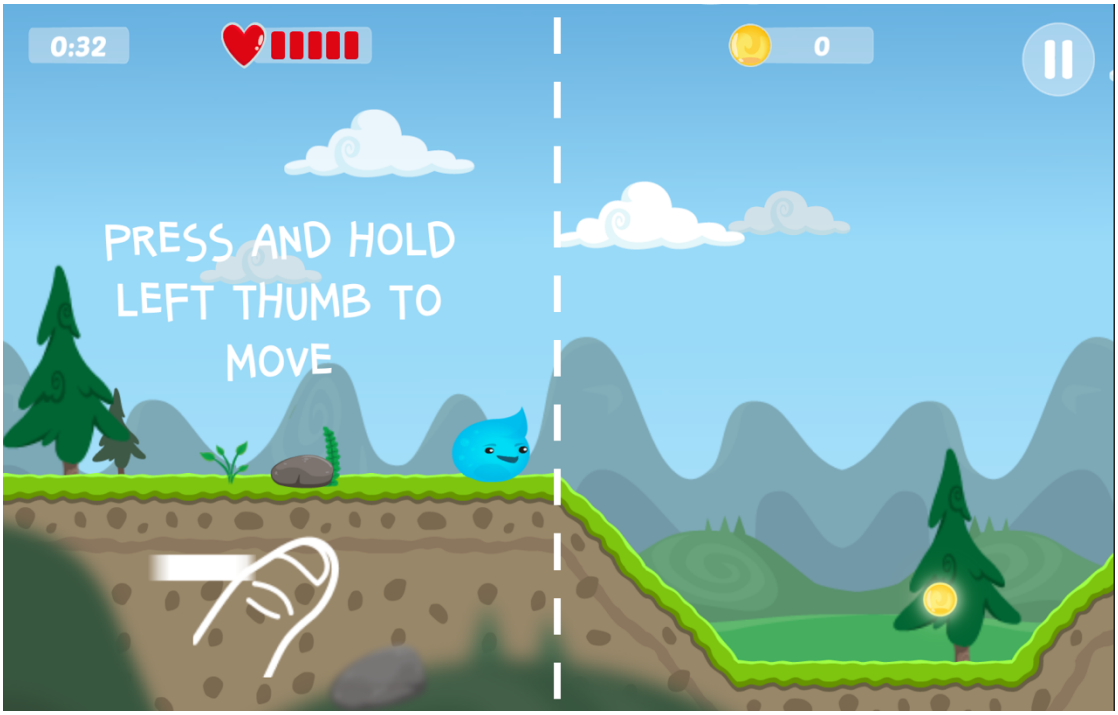
Kolmannen planeetan teema on suo. Planeetan pelimekaniikan mukaisesti suoplaneetta on kokonaan pimeä hohtavilla yksityiskohdilla, kuten tulikärpäsillä. Vaikka taustasta ei näy paljoa, halusin suon olevan vehreä ja kostea neonvihreillä yksityiskohdilla. Suunnittelin planeetalle vetisyyttä ja paljon kasvustoa, lumpeita, sieniä ja kaislaa. Uloin tausta tuotti hieman päänvaivaa, sillä en halunnut taustalle avointa taivasta kuten aiemmilla planeetoilla oli. Sen sijaan tein taustalle puunrunkojen ja liaanien siluetteja luoden illuusiota eksoottisesta sademetsästä. Kuten aiempia planeettoja suunnitellessa, halusin saasteversioihin realistisen kuvan siitä, mitä suomaisemalle tapahtuu pelin päävihollisen Boss L. Evilborgin saasteiden seurauksena. Vehreä suo kuivuu muuttuen ruskeaksi ja liejuiseksi. Kasvusto vaihtuu hiekkaan ja kiveen, kuten kuviosta 9 näkee.

KUVIO 9. Suoplaneetan taustat.

Taustojen tekninen puoli on mielenkiintoinen. Koska pelissä kuljetaan koko ajan eteenpäin, taustan on oltava hyvin pitkä. Yksittäinen, pitkä ja suuri kuva veisi liikaa prosessointitehoa ja muistia. Taustat onkin rakennettu erillisistä palasista. Esimerkiksi ensimmäisen planeetan taustapalaset ovat taivas, vuoripala, mäkipala ja ruohopala, kuten kuviosta 10 huomaa. Näiden osien päällä on tilesetistä kootut pelattavat maaosat ja uloimpana on vielä sumennettu etualaosa. Kuvio 11 on pelikuvaa, josta näkyy taustamaiseman rakenne. Tämä palapelimainen tekniikka on siis malliesimerkki jo aiemmin mainitsemastani optimoinnin tekniikasta eli grafiikoiden uudelleenkäytöstä. Yksinkertaiselta ja ehkä jopa ilmiselvältä tuntuva idea säästää nerokkaasti mobiilipelin rajallisia resursseja.



KUVIO 10. Esimerkki taustapalasista.



KUVIO 11. Pelikuvaa.

6.3 Viholliset

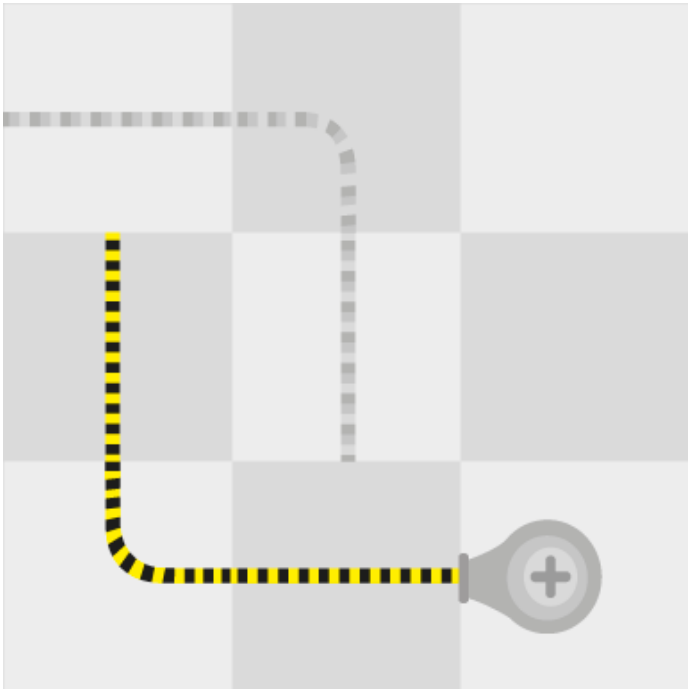
Pelissä on kaikkiaan kahdeksan erilaista vihollisrobottia, joilla kaikilla on eri käyttäytyminen, aseet ja tuhoutumismekaniikka. Näiden lisäksi on Boss L. Evilborg, pelin päävastus ja ainoa ihmismäinen hahmo. Suunnittelin kaikki robotit pelisuunnittelijan ohjeisiin perustuen. Pelaajan täytyi pystyä yhdistämään robotit luonnon tuhoutumiseen. Pelisuunnittelija halusi roboteista ruosteisia ja steampunkmaisia. Hyödynsin suunnittelussa steampunk-elementtejä muun muassa nahkahattujen ja suojalasien muodossa likatahroja unohtamatta. Yhdistelin robottien ulkonäköön myös eläinmäisiä piirteitä. Boss L. Evilborg mukailee robottien ulkomuotoa suojalaseilla ja steampunkmaisella vanhanaikaisella kaksiosaisella puvulla. Päätä koristaa villi tukka, viikset ja tietenkin silinterihattu, josta törröttää pakoputki. Kuviossa 12 näkyvät kaikki kahdeksan robottia ja Boss L. Evilborg.

Kaikilla roboteilla on useita reaktio-animaatioita, joten opettelin yrityksen ja erehdyksen kautta Unityn animointityökalua niitä animoidessa. Suunnittelin robottien ulkonäön niin, että ne olisi mahdollisimman helppo animoida. Kaikki robotit on tuotu Unityyn spritesheetteinä. Animoinnin hyödyn Unityssä kerroin kappaleessa neljä.

KUVIO 12. Vihollisrobotit, Boss L. Evilborg keskellä.

6.4 Muut objektit

Koska State of Matter on tasohyppelyn lisäksi myös osittain pulmanratkaisupeli, tarvittiin lisäksi myös esteitä ja muita objekteja. Näitä olivat esimerkiksi putket, turbiinit, johdot ja männät. Osa näistä esineistä oli staattisia ja aina saman kokoisia, joten ne vietiin Unityyn spritesheetissä. Objektit, jotka vaativat animaatiota, vietiin Unityyn omassa spritesheetissä, jossa jokainen leikattava osa on erillään. Tekniikka on siis sama kuin missä tahansa muussakin animoitavassa kohteessa. Joidenkin objektien, kuten johtojen ja mäntien koko kuitenkin vaihtelee tilanteesta riippuen, joten ne vietiin Unityyn tilesettinä. Esimerkiksi johdon pituutta pystyi helposti lisäämään kopioimalla keskipalasta. Kuviossa 13 näkyy johtotilesetin palaset. Objektien tileset-tekniikkaa käyttäen objekteja ei tarvitse viedä Unityyn useissa eri ko'oissa.



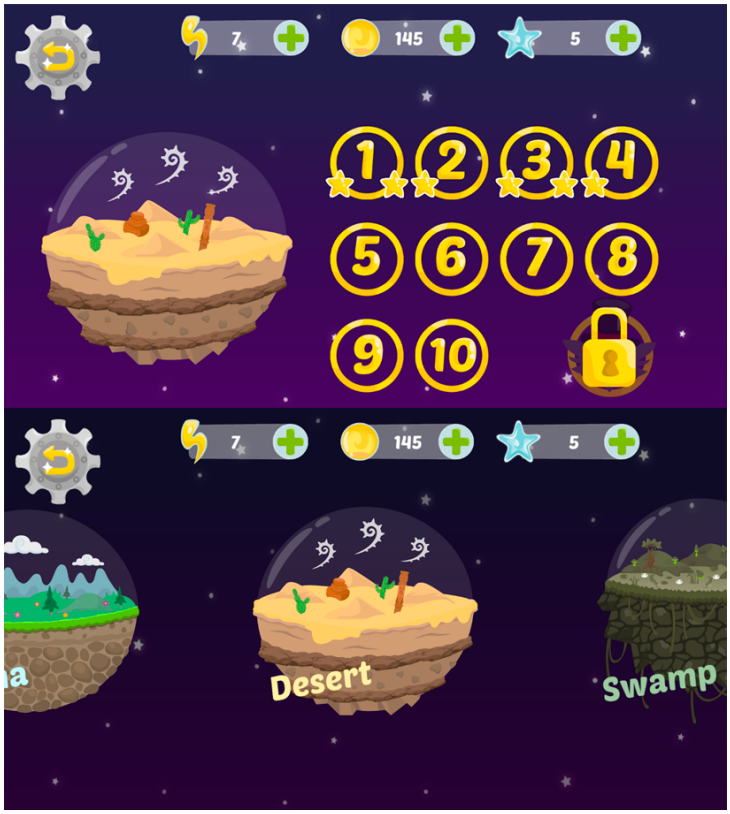
KUVIO 13. Johtotileset.

6.5 Käyttöliittymä

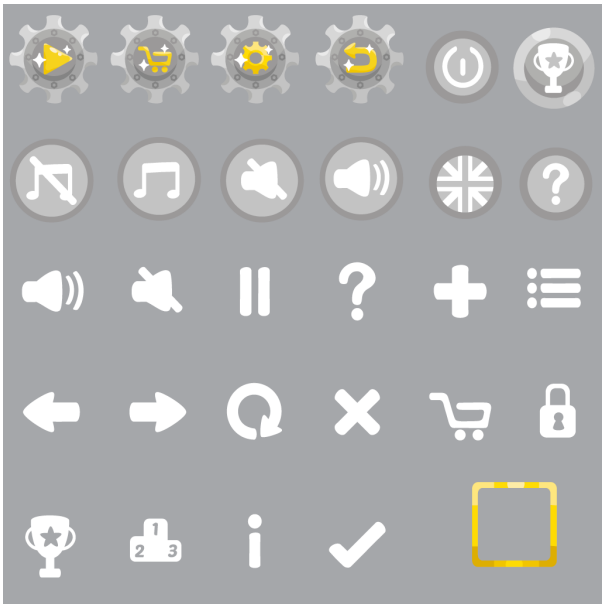
Käyttöliittymäsuunnittelu oli viimeisimpiä haasteitani peliprojektin parissa. Koska kukaan tiimistämme ei ollut suuremmin perehtynyt siihen ja pelisuunnittelijamme oli palannut kotimaahansa Espanjaan, koitui käyttöliittymän suunnittelu ja toteutus tehtäväkseni. Käyttöliittymää suunnitellessa yritin keskittyä mahdollisimman yksinkertaiseen rakenteeseen, koska mielestäni tärkeintä on saada rakenteesta mahdollisimman käyttäjäystävällinen. Suunnittelin käyttöliittymän ulkomuodon kahdesti, sillä ensimmäisellä kerralla en ollut tyytyväinen lopputulokseen. Käyttöliittymän ulkonäkö on hyvin värikäs ja sarjakuvamainen, mikä kuvastaa pelin muutakin graafista ilmettä. Kuvio 14 on esimerkki käyttöliittymästä.

Käyttöliittymän ikonit kannattaa sijoittaa samalle spritesheetille. Jos materiaalia on paljon ja spritesheetejä on useampia, kannattaa samantyyppiset ikonit sijoittaa samaan spritesheetiin. Kun ikonit ovat aina tietyssä kohtaa spritesheetissä, muutoksia tehdessä uudella versiolla voi suoraan korvata vanhan version, kunhan sen nimeää samaksi kuin vanha versio. Käyttöliittymän painikkeissa olevat mahdolliset reunat on helpoin tehdä asettamalla reunuksen kulmat ja pienen palan keskiosaa yhdeksi ikoniksi. Näin keskiosaa voi lisätä leikkaamalla ikonin Unityssä ja reunuksen kokoa

voi suurentaa laadun heikkenemättä. Pienen reunusikonin voi lisätä spritesheetiin ja näin säästää tilaa, kuten esimerkkikuviossa 15.



KUVIO 14. State of Matterin käyttöliittymää.



KUVIO 15. Ikonispritesheet, oikeassa reunassa reunaikoni

7 POHDINTA

Opinnäytetyöni tavoite oli miettiä optimoinnin merkitystä peliprojektissa sekä etsiä siihen parhaiten sopivia yleispäteviä menetelmiä. Avasin optimoinnin käsitettä sekä termin teknisestä näkökulmasta että käsitteen monimuotoisuudesta. Teknisiä tapoja löytyi useita, ja varmasti erilaisia tapoja on olemassa vielä useampia. Jokaisella tiimillä on oma tapansa toteuttaa optimointia, mutta kaikkein tärkeintä on huomata projektin optimoinnin mahdolliset puutteet ja ongelmakohdat tekniikasta riippumatta. Optimointi ei ole yksinkertainen ja suoraviivainen prosessi, vaan se vaatii huomiota jokaiselta projektiin osallistuvalta tekijältä, ja se pitää ottaa huomioon projektin jokaisessa vaiheessa.

Mielestä onnistuin työssäni hyvin. Esittelin optimointia eri näkökulmista omien rajoitteideni mukaisesti. Näkökulmaani kavensivat oma työnkuvani, hieman suppea kokemukseni pelialalta ja aiheen rajaus pääasiassa 2D-grafiikan optimointiin. Optimoinnin teknisemmät seikat kuuluvat ohjelmoijille, joten päätin suosiolla olla käymättä niitä läpi. Aiheen läpikäyminen oli opettavainen kokemus, ja pystyn varmasti hyödyntämään oppimaani tietotaitoa työssäni. State of Matter oli projektina hyvin opettavainen ja antoi alkusysäyksen myös tuleville projekteille. Käymällä läpi yleisiä teknillisiä tapoja sekä kertomalla omasta prosessistani haluan antaa lukijalle sekä teknistä tietoa että kokemusperäistä näkökulmaa optimointiin. Toivon työstäni olevan hyötyä kaikille kiinnostuneille pelialan graafikoille. Vaikka lähteeni eivät ehkä täytä kaikkia kriteerejä, mielestäni asiantuntijoilta saavani haastattelut antavat työlleni luotettavaa tietopohjaa.

Tärkeimmät käytännön toimintaohjeet optimointiin ovat optimoinnin huomioiminen koko peliprosessin ajan suunnittelusta tekovaiheeseen, testaukseen ja sovelluskauppaan asti. Optimointia ei tule unohtaa pelin ollessa valmis, vaan muistettava myös sovelluskaupan optimointi, lokalisaatio ja kulttuurisaatio. Asiantuntijoidenkin korostama hyvä yhteistyö koko tiimin välillä on erittäin tärkeää, ja erityisesti graafikoiden ja ohjelmoijien selkeä kommunikointi. Asioiden kysyminen ja tiedon jakaminen helpottavat työskentelyä. Pienetkin optimoinnin tekniset keinot auttavat kokonaisprosessissa, mutta tietenkin huomio pitää kiinnittää myös kokonaiskuvaan.

Optimointi on monimuotoinen ja nopeasti muuttuva aihe, mikä mahdollistaa aiheen lisätutkinnan myös tulevaisuudessa. Käsite muuttuu yhdessä mobiilimarkkinoiden ja älylaitteiden kehityksen kanssa. Tulevaisuuden Aasian, Intian ja Afrikan mobiilimarkkinoiden ottaessa lisää jalansijaa kansainvälisiin markkinoihin optimoinnin merkitys varmasti korostuu, sillä käyttäjät ovat pääasiallisesti

low-end mallien haltijoita. Vaikka elämmekin yhä globaalimmassa yhteiskunnassa, myös lokalisaation ja kulttuurisaation merkitys korostuu kulttuurien yhteen kohtaamisessa. Mobiilipelimarkkinoiden kasvaessa alan merkitys korostuu varmasti entisestään, ja yhä useammat löytävät tiensä alalle. Optimoinnista löytyy tutkittavaa mobiililaitteiden ja 2D-grafiikan lisäksi 3D-graafikoille, ohjemoijille, pelimuusikoille ja tablet-laitteisiin perehtyville. Toivon, että muutkin kiinnostuneet löytävät tämän mielenkiintoisen ja tärkeän aiheen, eivätkä kavahda teeman teknistä puolta.

LÄHTEET

Bither, Bill 2010. Benefits of the PNG Image Format. Viitattu 18.1.2015, <http://content.atala-soft.com/h/i/68054041-benefits-of-the-png-image-format>.

Brodkin, Jon 2013. How Unity3D Became a Game-Development Beast. Viitattu 16.1.2016, <http://insights.dice.com/2013/06/03/how-unity3d-become-a-game-development-beast>.

Dale, Laura 2015. Unity - does indie gaming's biggest engine have an image problem? Viitattu 16.1.2016, <http://www.theguardian.com/technology/2015/jul/06/unity-indie-gamings-biggest-engine-john-riccitiello>.

David, Eric 2015. Google doubles the size limit for Android APKs on the Play Store to 100MB. Viitattu 22.10.2015, <http://siliconangle.com/blog/2015/09/28/google-doubles-the-size-limit-for-android-apks-on-the-play-store-to-100mb>.

Ganguly, Robi 2015. App Store Optimization – A Crucial Piece of the Mobile App Marketing Puzzle. Viitattu 23.10.2015, <https://blog.kissmetrics.com/app-store-optimization/>.

Gordon, Jen 2011. Exporting Graphics for Mobile Apps: PNG or JPEG? Viitattu 18.1.2016, <http://code.tutsplus.com/tutorials/exporting-graphics-for-mobile-apps-png-or-jpeg--mobile-5154>.

Helgason, David 2013. Putting the Power of Unity in the Hands of Every Mobile Developer. Viitattu 16.1.2015, <http://blogs.unity3d.com/2013/05/21/putting-the-power-of-unity-in-the-hands-of-every-mobile-developer>.

Honeywood, Richard & Fung Jon 2012. Best Practices for Game Localization. Viitattu 25.10.2015, <http://englobe.com/wp-content/uploads/2012/05/Best-Practices-for-Game-Localization-v21.pdf>.

IDC Research 2015. Smartphone Vendor Market Share, 2015 Q2. Viitattu 26.10.2015, <http://www.idc.com/prodserv/smartphone-market-share.jsp>.

Johnson, Joshua 2011. Adobe Illustrator 101: 10 Things You Should Know About Ai. Viitattu 16.1.2015, <http://designshack.net/articles/software/adobe-illustrator-101-10-things-you-should-know-about-ai>.

Katsbits 2015. Make Better Textures For Games, 'Power Of Two' & Proper Image Dimensions. Viitattu 15.10.2015, <http://www.katsbits.com/tutorials/textures/make-better-textures-correct-size-and-power-of-two.php>.

OneSky 2014. The Beginner's Guide to Mobile Localization. E-kirja, OneSky publication.

Robot Invader 2014. Performance Optimization for Mobile Devices. <http://robotin-vader.com/blog/?p=438>.

Samsung 2015. Compare Cell Phones. Viitattu 26.10.2015, <http://www.samsung.com/us/compare/#category/N0000002/products/SM-G800RZKAUSC,SM-G900TRKATMB,SM-G920RZKAUSC>.

Smith, Alvy Ray 1995. Image Compositing Fundamentals: Technical Memo 4. Viitattu 10.1.2015, <http://www.cs.princeton.edu/courses/archive/fall00/cs426/papers/smith95a.pdf>.

Unity 3d, Company Facts. Viitattu 16.1.2016, <http://unity3d.com/public-relations>.

Unity 3d 2015a. iOS Hardware Guide. Viitattu 26.10.2015, <http://docs.unity3d.com/Manual/iphone-Hardware.html>.

Unity 3d 2015b. Mobile Developer Checklist: Optimization. Viitattu 18.10.2015, <http://docs.unity3d.com/Manual/MobileOptimisation.html>.

Unity 3d 2015c. Optimizing Graphics Performance: Draw Call Batching. Viitattu 19.10.2015, <http://docs.unity3d.com/Manual/DrawCallBatching.html>.

Unity 3d 2015d. Practical Guide to Optimization for Mobiles: Graphics Methods. Viitattu 18.10.2015, <http://docs.unity3d.com/Manual/MobileOptimizationGraphicsMethods.html>.

Willoughby, Scott 2015. Size matters: How file size impacts installs for mobile games. Viitattu 22.10.2015, <https://playfab.com/blog/size-matters-how-file-size-impacts-installs-mobile-games>.

KUVALÄHTEET

Kuvio 1. Ivanov, Ivan-Assen 2006. Viitattu 26.10.2015, <http://www.gamasutra.com/view/feature/130940/practical_texture_atlases.php>.