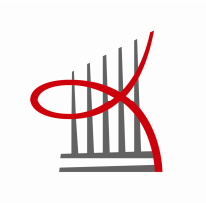


3D-sovellus Internet-ympäristöön

Case: Virtuaaliluokka

Ilkka Rönkä

Opinnäytetyö
Kesäkuu 2011
Tietojenkäsittelyn koulutusohjelma
Digimedia
Tampereen ammattikorkeakoulu

TAMPEREEN AMMATTIKORKEAKOULU
Tampere University of Applied Sciences

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tietojenkäsittelyn koulutusohjelma, Digimedia

Tekijä	Ilkka Rönkä
Työnnimi	3D-sovellus Internet-ympäristöön Case: Virtuaaliluokka
Sivumäärä	81 s.
Valmistumisvuosi	2011
Työntilaaja	Kasavuoren yläkoulu

Tämän opinnäytetyön tarkoitus on suunnitella ja toteuttaa sovellus Kauniaisissa toimivan Kasavuoren yläkoulun verkkosivuille, joka esittelee yläkoulun historian luokan pedagogista oppimisympäristöä ja toimintatapoja reaaliaikaisen kolmiulotteisen grafiikan, videon, kuvien ja äänen keinoin. Esityksen elementeistä pyritään muodostamaan helppokäyttöinen eheä verkkosovellus.

Opinnäytetyöni avulla pyritään löytämään uusia markkinointikanavia vapaaseen kilpailutilanteeseen, jossa työni tilaaja Kasavuoren yläkoulu kilpailee lisäoppilaista Espoon yläkoulujen kanssa. Kauniaisten väestömäärä ei riitä täyttämään kaikkia oppilaspaikkoja ja koulu tarvitsee lisäoppilaita taloudellisesti järkevän yläkoulu-toiminnan järjestämiseen. Työni avulla pystytään esittelemään elävästi opetustoimintaa ja oppimisympäristöjä sekä pedagogiikkaa.

Sovellus on toteutettu Unity3D-pelinkehitystyökalulla. Sovelluksen sisällön toteuttamiseen on käytetty Cinema4d-mallinnusohjelmaa ja HTML-kieltä sekä CSS-muotoilukieltä. Toiminnallisuuden ohjelmointiin selaimessa ja Unity3D-pelinkehittelytyökalussa käytettiin Javascript-ohjelmointikieltä.

Opinnäytetyöni raportin alkuosassa esittelen samankaltaisia verkkosivuilla käytettäviä tekniikoita, joilla sovellukseni olen toteuttanut, jonka jälkeen perehdyn tarkemmin käyttämieni työkalujen käyttöön ja ominaisuuksiin. Loppuosassa esittelen sovelluksen tekoon liittyviä toteutusvaiheita ja lopputulokset.

Opinnäytetyöni alkuperäinen suunnitelma oli luoda koko koulun käsittävä sovellus, jossa esiteltäisiin malliluokat oppiaineittain ja koulun julkiset tilat. Työni rajautui käsittämään vain yhden malliluokan esittelyn, koska koko koulun esittely muodostui työmäärällisesti liian suureksi. Onnistuin luomaan sovelluksesta demo-version, joka muodosti sovelluksen kehittämiselle hyvät lähtökohdat.

Asiasanat: virtuaaliympäristö, Unity3D, 3D-mallinnus, koulu

ABSTRACT

Tampere University of Applied Sciences
Degree Programme in Business Information Systems
Specialisation Option of Digital media
Ilkka Rönkä: 3D-application in the Internet environment Case : Virtual Classroom

Bachelor´s thesis
2011

Purpose of this thesis is to design and compile an application for an "Kasavuori" secondary school's website in Kauniainen, that would present in school's history course pedagogical learning environment and practices by real-time three-dimensional graphics, video, photos and audio. Our aim is to present all the information through an easy-to-use integrated web application.

This thesis aims to find new marketing means for free competition where the referred "Kasavuori" secondary school, which is involved to my project competes with secondary schools in Espoo in order to get additional students. As a consequence of Kauniainen's small population a portion of student spots remains uncovered. Thus acquisition of more studies is crucial for the establishment of school's financially reasonable performance. My work helps to interactively present the teaching methods and learning environment as well as pedagogics.

This application has been created with the help of Unity3D game's development tool. For this application's implementation has been used the Cinema4D modeling software, the HTML-language and CSS style sheet language. Javascript functional programming language was used to create application's functionality in the browser and in the Unity3D game's development tool the.

In the first part of my thesis's report will include an introduction of other similar to my application methods that are currently in use. As next I will present in details the potential applications and features of these tools. In the end, I will present application's phases of implementation, results and conclusions.

My thesis's original goal was to create an application that would cover the whole school and in which model classes for every teaching subject as well school's public spaces would be presented. My work was confined to include only one model class presentation as a whole school presentation would demand a tremendous workload. I managed to create a demo version of the application, which constituted a good starting point for application's further development.

Key Words: Virtual environment, Unit3, 3D-modelling, School

SISÄLTÖ

1 JOHDANTO

Opinnäytetyön raporttiosan tarkoituksena on toimia kuvauksena opinnäytetyöni käytännönosasta. Kirjallisessa osassa käyn läpi keskeiset vaiheet ja tekniikat liittyen käytännön työosuuden tekemiseen. Raportin alussa tutkin kolmiulotteisuutta verkkosivuilla, jonka jälkeen esitän käytännön teoriapohjan työssä käytetyistä tekniikoista. Raportin lopussa käyn läpi tekemäni sovelluksen työn syntyvaiheita.

Aihe opinnäytetyöhöni syntyi keskusteluista, miten kolmiulotteisen reaaliaikaisen ympäristön avulla voisi kuvata koulun ympäristöä, pedagogista toimintaympäristöä ja koulun tiloja. Keskustelujen seurauksena syntyi ajatus luoda virtuaalinen kouluympäristö, jonka kautta pystyisi visualisoimaan koululuokan toimintaa ja ympäristöä 3D-mallin, animaatioiden, videoiden ja tietoikkunoiden kautta. Työni rajautui yhden oppiaineen ja luokan toiminnan kuvaamiseksi, koska koko koulun toimintaa kuvaavan sovelluksen tekeminen näytti muodostuvan liian suureksi urakaksi. Tavoitteeksi muodostui aikaansaada helppokäyttöinen mainoksenomainen sovellus Websivulle, jonka kautta uusien oppilaiden vanhemmat ja uudet oppilaat voisivat tutustus oppiaineen opetukseen. Työni tekemiseen tarvitaan monia ohjelmia ja niiden ominaisuuksia, tämän takia olen rajannut työni kirjallisenosan sisällön kattamaan vain käytännön työn kannalta olennaiset asiat.

Opinnäytetyöni toimeksiantajana toimii Kauniaisten suomenkielisen koulutoimen yläkoulu, Kasavuoren koulu. Kauniaisten koulutoimenjohtaja Antti Rönkä kuvaa seuraavassa työnlähtökohtia. Koulussa käy espoolaisia oppilaita. Koululle on tärkeää saada Espoosta lisäoppilaita, koska Kauniaisten oma väestö ei riitä järkevään, taloudelliseen yläkoulun järjestämiseen ja ylläpitoon. Koulu on oppilaiden hankkimisessa vapaassa kilpailutilanteessa Espoon yläkoulujen kanssa. Mallintaminen palvelee koulun markkinointia, koska toimintaa ja oppimisympäristöjä sekä pedagogiikkaa voidaan esitellä koulun kotisivuilla elävästi sen avulla.

Koulu on tehnyt viime vuosikymmeninä paljon työtä toimintansa kehittämisessä. Sen pedagoginen perusta on tutkiva, yhteistoiminnallinen oppiminen. Koulu on myös hahmottanut ydinprosessinsa, joka on oppilaan kasvun ja oppimisen tukeminen. Tämä merkitsee perinteiseen oppiaineen sisältöjen oppimiseen perustuvan tavoitteiston huomattavaa laajenemista ja toiminnan painopisteen muuttumista. Kasavuoressa oppilas on vastuussa oppimisestaan ja opettaja toimii resurssina, joka antaa tarvittavan tuen oppimiselle. Toiminta on pitkälle ryhmätoimintaa, jossa oppimista tapahtuu myös toisilta oppilailta tai ulkomaisen yhteysprojektin tapauksessa yhteistyökoulujen oppilailta.

Koululle on järjestetty toiminnallisia oppimisympäristöjä, joista on hyvät nykyaikaiset yhteydet ulkomaailmaan. Koululla on myös hyvä tietotekninen infrastruktuuri, joka on oppimisympäristön keskeinen työkalu. Koneet on sijoitettu hajautetusti koko rakennukseen siten, että kussakin luokassa on 10 -20 konetta. Kokonaiskonemäärä on n. 400, joka tarkoittaa kone/ 2 oppilasta. Luokissa on myös sekä 100 Mb:n langallinen että langaton verkko.

Opetushallitus määrittelee perusopetuksen opetussuunnitelmaa seuraavasti. Oppiminen on seurausta oppilaan aktiivisesta, tavoitteellisesta toiminnasta, jossa hän aiempien tietorakenteiden pohjalla käsittelee ja tulkitsee opittavaa ainesta. Oppiminen riippuu aiemmin rakentuneesta tiedosta, motivaatiosta sekä oppimis- ja työskentelytavoista. Oppiminen on kaikissa muodoissa aktiivinen ja päämääräsuuntautunut, itsenäistä tai yhteistä ongelmanratkaisua sisältävä prosessi. Oppiminen on tilannesidonnaista, joten oppimisympäristön monipuolisuuteen on kiinnitettävä erityistä huomiota. (Perusopetuksen opetussuunnitelman perusteet 2004)

Koulujen tehtävänä on tarjota opetussuunnitelman toteutumiselle otolliset puitteet. Opinnäytetyössäni olevan malliluokan oppimisympäristön suunnitellussa on huomioitu perusopetuksen tavoitteet. Oppimisympäristön vaatimukset on määritelty seuraavanlaisesti. Oppimisympäristön tulee tukea oppilaan kasvua ja oppimista. Sen on oltava fyysisesti, psyykkisesti ja sosiaalisesti turvallinen ja tuettava oppilaan terveyttä. Tavoitteena on tukea oppilaan oppimismotivaatiota ja uteliaisuutta ja edistää hänen aktiivisuttaan,

itseohjautuvuuttaan ja luovuuttaan tarjoamalla kiinnostavia haasteita ja ongelmia. Oppimisympäristön tulee ohjata oppilasta asettamaan omia tavoitteitaan ja arvioimaan toimintaansa. Oppimisympäristön tulee tukea opettajan ja oppilaan välistä ja oppilaiden keskinäistä vuorovaikutusta. Sen tulee edistää vuoropuhelua ja ohjata oppilaita työskentelemään ryhmän jäsenenä. Tavoitteena on avoin, rohkaiseva, kiireetön ja myönteinen ilmapiiri. (Perusopetuksen opetussuunnitelman perusteet 2004)

Oppimisympäristö on sekä fyysinen (rakennukset, tilat, välineet, ympäröivä luonto, rakennettu ympäristö) että psyykkinen (oma tunne- ja motivaatiotilat, vuorovaikutukseen ja ihmissuhteisiin liittyvät tekijät). Kehittämäni sovelluksen tarkoitus on kuvata mainoksenomaisesti katsojalle miten kuvattu oppimisympäristö palvelee näitä tavoitteita. (Perusopetuksen opetussuunnitelman perusteet 2004)

Opinnäytetyöni otsikko rajautui kuvaamaan tekemäni sovelluksen elementeistä muodostuvaa kokonaisuutta. Käytin käytännöntyö osuudesta muodostuvan sovelluksen tekoon useita eri ohjelmia ja tekniikoita, joista kokonaisuus muodostui.

2 KOLMIULOTTEISUUS VERKKOSIVUILLA

3D-sisältö on tunnustettu laajalti seuraavaksi kehitysaskeleeksi digitaalisessa mediassa. 3D-ympäristöjen kehittämisen kustannusten aleneminen ja nykyiset menestystuotteet esim. Second life ja Google Earth, ovat luoneet uusia tapoja, miten ihmiset hahmottavat ja selaavat WWW-sivuja. Internetin on ennustettu siirtyvän tekstipohjaisesta esitystavasta visuaaliseksi 3D-maailmaksi.

Kolmiulotteinen sisältö mahdollistaa paremman interaktiivisuuden käyttäjän ja verkkosovelluksen välillä, koska käyttäjä voi tarkastella ja ohjata kohdetta monesta eri kuvakulmasta. Kolmiulotteisuuden synnyttämä sisältö on monimerkityksellistä, joka saattaa luoda katsojassa sekaannusta. Esitystapana kolmiulotteisuutta valittaessa kannattaakin miettiä visuaalisen sisällön merkitystä esityksessä. (Spagnuolo & Falcidieno 2009)

Kolmiulotteisuus verkkosivuilla avaa uusia mahdollisuuksia tiedon esittämiseen. Sen avulla pystytään paremmin visualisoimaan ja analysoimaan taloudellista, kaupallista ja tieteellistä tietoa, luomaan viihdyttävämpiä WWW-sivuja ja pelejä, tehostamaan opiskelua ja koulutusta, sekä esittelemään paremmin verkkokauppojen tuotteita.

2.1 3D-Selain tekniikoita

3D-selain on tavallisesti selainohjelmaan ladattava lisäosa, jonka avulla saadaan kolmiulotteista sisältöä näkymään verkkosivuilla. 3D-selain suorittaa kolmiulotteisen kuvan muodostukseen tarvittavat laskelmat ja pinnoitemateriaalin luontitoimenpiteet. Monet yritykset kehittävät samanaikaisesti omia 3D-selaimiaan. Osa selaimista on ilmaisia ja osa maksullisia. Pääsyy yhteisen 3D-standardin puuttumiseen on kilpailutilanne, jossa yksittäiset yritykset omalla tekniikallaan pyrkivät saavuttamaan dominoivan asemaa markkinoilla. 3D-selaimet eivät vielä ole lyöneet itseään läpi Internetin käyttäjien parissa. Kuitenkin monet organisaatiot ovat kiinnostuneita 3D-selainten mahdollisuuksista. (Leavitt 2006)

Aloja, joilla 3D:tä on hyödynnetty verkkosivuilla, ovat arkkitehtuuri, terveydenhuolto, IT, koulut, verkkokaupat, armeija, muotoilu, tieteellinen tutkimus ja teollinen tuotanto. Arkkitehtuurissa kolmiulotteisuuden avulla on luotu talojen visualisointeja verkkosivuille, joissa voi vapaasti liikkua ja tutustua rakennuksiin. Terveydenhuollossa on ihmiskehon anatomiaa ja sairauksia visualisoitu 3D-mallien avulla. Verkkokaupat ovat myös hyödyntäneet kolmiulotteisuutta tuotteidensa esittelyssä. (Leavitt 2006)

2.1.1 Flash-alusta

Adoben Flash-alustalle on kehitetty useita 3D-moottoreita ja grafiikkakirjastoja kolmiulotteisuuden esittämiseksi. Tekniikat toimivat selaimissa Adobe Flash-selainlaajennuksen avulla. Laajennus löytyy lähes jokaisesta selaimesta. Laajennus on ladattavissa ilmaiseksi Adoben kotisivuilta. Tunnettuja 3D-moottoreita ovat Papervision, Alternative3D, Sandy ja Away3d.

Papervision on pienen ydinryhmän kehittämä ja ylläpitämä avoimenlähdekoodin 3D-moottori Adoben Flash-alustalle. Moottori on julkaistu vuonna 2007. Suurin osa Papervisionin dokumentaatiosta löytyy erillaisista blogeista ja keskusteluryhmistä. Papervision on melko vakaa alusta, minkä johdosta sitä on käytetty monissa kaupallisissa projekteissa. (Kallonen 2010, 1)

Alternative3D on venäläisryhmän kehittämä 3D-moottori Flash-alustalle. Ensimmäinen versio alustasta on julkaistu vuonna 2007. Alustan käyttö on pääasiassa suuntautunut pelien tekoon. (Alternative 2010)

2.1.2 Editori-ohjelmat

Editori-ohjelmien avulla pystyy tuottamaan kolmiulotteista reaaliaikaista sisältöä sisältäviä sovelluksia. Ohjelmissa on yleensä kattavat ominaisuudet sovelluksien kehittämiseen. Verkkosivuilla sovellukset näkyvät ohjelmien omien 3D-selainlaajennuksien avulla. Selainlaajennukset ovat yleensä la-

dattavissa ohjelmien omilta kotisivuilta. Tunnettuja ohjelmia ovat Adobe Director, Quest3D ja Unity3D.

Quest3D on Hollantilaisen act3D-yrityksen julkaisema sisällöntuotanto-ohjelma, jonka avulla voi luoda reaaliaikaisia kolmiulotteisen esityksiä ja simulaatioita WWW-sivulle tai omaksi exe-sovellukseksi. Ensimmäinen versio ohjelmasta julkaistiin vuonna 2000. Uusin versio ohjelmasta on quest3D 4.0, joka on julkaistu vuonna 2008. 3D-mallit, kuvat ja äänet pitää luoda esityksiin erillisillä ohjelmilla. (Quest3D 2009)

Adobe Shockwave on vuonna 1995 Director-ohjelman tiedostoformaatti, joka mahdollistaa multimedia sisällön esittämisen verkkosivuilla. Formaatti suunniteltiin aluksi monenlaisen multimedia sisällön esittämiseen. Itsensä läpi formaatti on kuitenkin lyönyt erilaisten Internet-pelien alustana. (Adobe Shockwave 2010)

Unity3D on vuonna 2001 julkaistu pelienkehitystyökalu, joka mahdollistaa monipuolisen kolmiulotteisen sisällöntuottamisen tietokoneille, pelikonsoleille ja mobiililaitteille. Uusin versio ohjelmasta 3.2, joka on julkaistu vuonna 2011. Unitystä on kaksi versiota, Unity free ja Unity pro. Unity free version saa ladattua ilmaiseksi ohjelman omilta kotisivuilta. Unityn pro-versio on maksullinen, ja se sisältää joitain lisäominaisuuksia. (Unity 2011)

2.1.3 Kehityssuuntia

Ensimmäisiä kolmiulotteisten sovellusten esitys tekniikoita selaimissa oli Virtual Reality Modeling Language (VRML), joka julkaistiin vuonna 1994. Tekniikan avulla esitetään tekstimuotoista vektorigrafiikkaa tekstimuotoisen tiedoston avulla, jossa kuvataan kolmiulotteisien mallien ominaisuuksia esim. muodosta, pintamateriaalista, valaistuksesta, sijainnista suhteessa toisiinsa. VRLM-kielestä kehitettiin uusi versio vuonna 1997, jota käytettiin henkilökohtaisilla kotisivuilla ja kolmiulotteisissa Chat-sovelluksissa. (VRLM 2011)

VRLM-kielen pohjalle on kehittynyt X3D, joka on Web3D Consortium julkaisema ISO-standardisoima XML-pohjainen formaatti kolmiulotteisen grafiikan esittämiseen. X3D-tiedostossa kuvaillaan XML-formaattiin VRML97- tai binaari-syntaksin avulla kolmiulotteisen mallin ominaisuuksia, esim. 3D-mallien muotoa, sijaintia suhteessa toisiinsa, pintamateriaalia, valaistusta, heijastusta. WWW-sivulla näkyäkseen X3D tarvitsee erillisen 3D-selainohjelman, joka tukee tiedostomuotoa. X3D-tiedostomuodosta on povattu yleistä 3D-ohjelmien välistä rajapintaa. (What is X3d 2010)

Yksi kehityssuunta HTML5-kielen Canvas-elementti, jonka avulla pystyy piirtämään selaimeen kolmiulotteista grafiikkaa. Canvas-elementin pohjalle on kehitteillä WebGL-tekniikka, jossa Javascriptin avulla esitetään kolmiulotteista reaaliaikaista grafiikkaa Internetissä ilman erillistä selaimen lisäosaa. WebGL pohjautuu mobiililaitteissa käytettävään OpenGL ES 2.0 grafiikkastandardiin. (WebGL 2010)

Google yrittää myös omalta osaltaan O3D-tekniikan kehityksen myötä osallistua kolmiulotteisen Internetin kehitykseen. O3D on Googlen kehittämä kokeellinen avoimenlähdekoodin Javascript-rajapinta, jonka avulla voidaan esittää kolmiulotteista grafiikkaa verkkosivulla. WWW-sivuilla alusta toimii oman selaimeen ladattava lisäosan kautta. Google on ilmoittanut kehittävänsä O3D:tä toimimaan tulevaisuudessa omana Javascript-kirjastona WebGL rajapinnan päällä. (Ryan 2009)

2.2 3D-sovelluksia Internetissä

2.2.1 Finlandiapuisto

Finlandiapuisto on arkkitehti Petri Kokon Unity3D-alustalle suunnittelema virtuaaliympäristö, joka sijaitsee Helsingin kaupungin WWW-sivustolla. Virtuaalimallin avulla on tarkoitus visualisoida suunnitteilla olevaa Finlandiapuiston eteläosaa. Ympäristöä on tarkoitus kehittää laajemmaksi tulevaisuudessa. Ympäristössä liikutaan näppäimistön ja hiiren avulla sovelluk-

seen luotujen Avatar-hahmojen välityksellä, jotka valitaan useasta eri hahmo vaihtoehdosta. Alla olevassa kuvassa näkyy ilmakuva sovelluksen ympäristöstä (kuva 1). (Finlandiapuisto 2010a)

Kuva 1. Finlandiapuisto (Finlandiapuisto 2010b)

2.2.2 Ancient

Ancient on NewMediaStar-yrityksen suunnittelema futuristinen virtuaaliympäristö Alternativa Flash3D-alustalle. Ympäristöön on luotu oma pelimäinen äänimaisemansa, joka luo oman lisänsä tunnelmaan. Ympäristössä liikutaan hiiren ja näppäimistön avulla. Alla olevassa kuvassa näkyy sovelluksen näkymä (kuva 2).

Kuva 2. Sovellus näkymä (Ancient 2011)

2.2.3 Barcinski & Jeanjean

Barcinski & Jeanjean on kahden suunnittelijan itselleen luoma kolmiulotteinen portfolio, joka on toteutettu Papervision Flash3D-alustalle. Erikoisuutena on vaihtoehto käyttää 3D-laseja sivun katseluun. Kuvassa näkyy sovelluksen kolmiulotteinen käyttöliittymä (kuva 3).

Kuva 3. Kolmiulotteinen käyttöliittymä (Barcinski & Jeanjean 2011)

2.3 Virtuaalimaailmat

Internet on pullollaan erilaisia virtuaalimaailmoita kaiken ikäisille käyttäjille. Maailmat ovat kehittyneet tekstipohjaisista toteutuksista näyttäviksi visuaalisiksi taideteoksiksi. Virtuaalimaailmoista on kehittynyt oma rinnakkaistodellisuutensa, suurimmat yritykset ostavat mainostilaa, opiskelijat ympäri maailmaa saattavat kokoontua yhteiselle luennolle.

Wikipedian mukaan virtuaalimaailmalla tarkoitetaan tietokoneen luomaa virtuaalitodellisuutta, jossa käyttäjä voi liittyä jäseneksi tai pelaajaksi, ollakseen vuorovaikutuksessa muiden käyttäjien ja simulaatioiden kanssa. Käyttäjä luo kaksi tai kolmiulotteisella grafiikalla visualisoituun virtuaalimaailmaa oman Avatar-hahmon, jonka kautta hän seikkailee virtuaalimaa-

ilmassa. Internetissä toimivissa virtuaalimaailmoissa voi samanaikaisesti olla kymmeniätuhansia käyttäjiä. (Virtuaalitodellisuus 2010)

Nick Wilson määrittelee Tuija Aallon kirjoittamassa blogissa (Aalto 2007) tarkemmin virtuaalimaailman käsitettä. Hän ehdottaa sosiaalisen virtuaalimaailman olevan pelimäisen uppouttava, juoneton ja tavoitteeton yhteisöllinen viestin/väline, ja että keskeistä on läsnäolon tunne yhtaikaa muiden kanssa samassa paikassa "A Social Virtual World has game-like immersion and social media functionality without narrative driven goals. At its core is a sense of presence with others at the same time and place". (Interview with Nick Wilson of Metaversed.Wilsson.2007) Hän listaa nämä kriteerit täyttäviksi virtuaalimaailmoiksi seuraavat; Second Life, ActiveWorlds, Kaneva, vSide, Entropia, Ogoglio City, There.com, MTV's Worlds. Wilson erottelee myös toisistaan pelimäiset virtuaalimaailmat ja kokemusperäiset virtuaalimaailmat. Pelimäisillä virtuaalimaailmoilla hän tarkoittaa maailmoja, joissa on selkeä juoni ja tehtäväpohjainen toimintamalli. Kokemusperäisillä virtuaalimaailmoilla hän tarkoittaa maailmoja joissa toiminta syntyy läsnäolon tunteesta muiden kanssa. (Aalto 2007; Interview with Nick Wilson of Metaversed 2007)

2.3.1 Second Life

Second life on yksi tunnetuimmista kokemusperäisistä virtuaalimaailmoista. Maailman on julkaissut Linden lab vuonna 2003 ja tähän mennessä sillä on neljätoista miljoonaa käyttäjää. Maailmaan pääsee Internetissä ladattavan ilmaisen Viewer client ohjelman avulla. Ohjelmassa liikutaan oman Avatar-hahmon avulla. Hahmon välityksellä käyttäjä voi tavata muiden käyttäjien hahmoja ympäri maailmaa, osallistua erilaisiin tapahtumiin, seikkailla maailmaan luoduissa virtuaalitiloissa, ostaa virtuaalituotteita jne. Virtuaalimaailmalla on oma rahayksikkönsä Linden-dollari, joka on listattu oikeilla rahamarkkinoilla Internetissä olevissa valuuttapörsseissä. Tunnetut yhtiöt kuten Nokia, IBM, Reutesrs ovat löytäneet maailmasta uuden markkinointikanavan brändilleen perustamalla liiketiloja tai mainontaa maailmaan. Oppilaitokset ja yliopistot ovat myös löytäneet maailmasta mahdolli-

suuden omalle toiminnalleen perustamalla virtuaalikampuksia, joiden välityksellä on voinut osallistua esimerkiksi virtuaaliluennolle. Myös monella suurlähetystöllä on oma edustuksensa kuten Ruotsilla, Virolla, Malediivieilla. (Second Life 2011a)

Kuva 4. Second Live (Second Life 2011b)

2.3.2 World of Warcraft

World of Warcraft on Blizzard Etertainmentin vuonna 2004 julkaisema massiivinen monen pelaajan verkkoroolipeli. Peli on Internetissä löytyvistä pelimäisistä virtuaalimaailmoista suosituin. Pelillä on kaksitoista miljoonaa käyttäjää. Wow sijoittuu Azeroth-fantasiamaailmaan, johon on luotu oma tarinansa, jonka keskeltä pelaaja valitsee valinnoillaan roolinsa. Pelin alussa pelaaja luo itselleen hahmon erilaisista annetuista vaihtoehdoista. Valinnoilla on vaikutus hahmon kykyihin ja ulkomuotoon. Pelissä edetessä suoritetaan erilaisia tehtäviä, jonka kautta saadaan kokemuspisteitä ja palkintoja. Tarpeeksi kokemuspisteitä saatuaan hahmo etenee seuraavalle tasolle, jonka kautta aukeaa uusia kykyjä ja taitoja. Pelaaja voi liittoutua taistella toisia pelaajia vastaan. (World of Warcraft 2011)

3 3D-MALLINNUS

Tässä luvussa käyn läpi 3D-mallintamisen perustekniikoita ja Cinema4d-ohjelman eri ominaisuuksia liittyen 3D-mallien toteuttamiseen. Kolmiulotteisten mallien luonti muistuttaa hyvin paljon savenvalamista tai muovailuvahan muotoilua. Tietokoneohjelmistoilla 3D-mallit muodostuvat pisteistä, jotka yhdessä muodostavat mallin ääriviivat kolmiulotteisessa tilassa. Mallintaminen vaatii tekijältään monenlaisia taitoja. Huomioon on otettava 3D-objektien muodon lisäksi 3D-näyttämön asettelu, objektin tai tilan valaistus, objektien pintamateriaalien luominen, materiaalien ominaisuuksien luominen, animaatioiden luominen, jne. Kolmiulotteisten objektien ja ympäristöjen luontia varten tarvitaan erillisiä 3D-grafiikkaohjelmisto- ja. Ohjelmistoja on moniin eri tarkoituksiin elokuvateollisuuden käytöstä maisemasuunnitteluun. Tunnetuimpia julkaisuja ovat 3DMax, Cinema4d, Maya ja Blender.

3DMax ja Maya ovat Autodeskin julkaisemia ammattilaiskäyttöön tarkoitettuja 3D-mallinnusohjelmia. 3DMax on yksi laajimmin käytössä oleva ohjelma, jonka vahvuudet ovat mallinnus-ominaisuuksissa ja plugin-ominaisuuksissa. Maya on varsinkin elokuva ja peliteollisuuden käytössä oleva ohjelma. Maya ohjelmasta on saatavissa useita eri versioita eri ominaisuuksilla.(3Dmax ; Maya 2011)

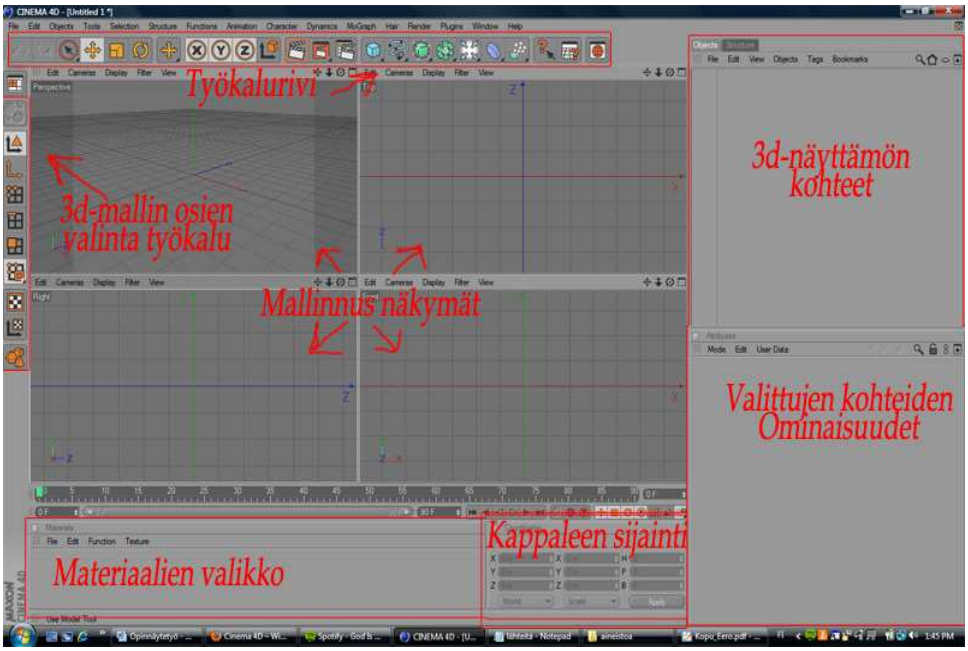
Blender on yksi suosituimmista täysin ilmaisista ammattilaistason 3D-mallinusohjelmista. Blender on mallinnusominaisuuksiltaan yhtä hyvä kuin kaupalliset ohjelmat. Ohjelmassa on myös oma pelimoottori. (Blender 2010)

3.1 Cinema4d

Käytin opinnäytetyössäni kolmiulotteiseen mallinnukseen Maxonin julkaisemaa Cinema4d-ohjelmaa. Ohjelman valintaan vaikuttivat omat käyttökokemukseni ja oppilaitoksellani oleva lisenssi ohjelmasta. Cinema4d-ohjelmasta löytyy monia versioita eri lisensseillä erilaisiin käyttötarkoituk-

siin. Tyypillisiä Cinema4d:llä tehtyjä kohteita ovat arkkitehtisuunnitelmien visualisoinnit, tuotteiden visualisointi, sekä animaatiot televisioon, elokuviin ja WWW-sivuille. Ohjelmassa löytyy kattavat mallintamis-, valaistus- ja animointi-ominaisuudet. Ohjelman itseopiskelua varten löytyy Internetistä monia yhteisöjä, jotka julkaisevat ohjeita ohjelman käyttöön. Yksi tunnetuimmista yhteisöistä on Cineversity.com.

Cinema4d:ssä käyttöliittymässä löytyy monta näkymään animointia, mallintamista ja teksturointia varten. Ohjelman mallintamisikkunaan voi asettaa kuvan neljästä eri kuvakulmasta tai vain yhden kuvakulman. Cinema4d-perusnäkymä sisältää työkalurivin, 3D-mallien osien valintatyökalut, mallinnusnäkymät, 3D-näyttämön kohteet, valittujen kohteiden ominaisuudet näkymän, materiaalivalikon ja kappaleen sijainti-kontrollit (kuva 5).



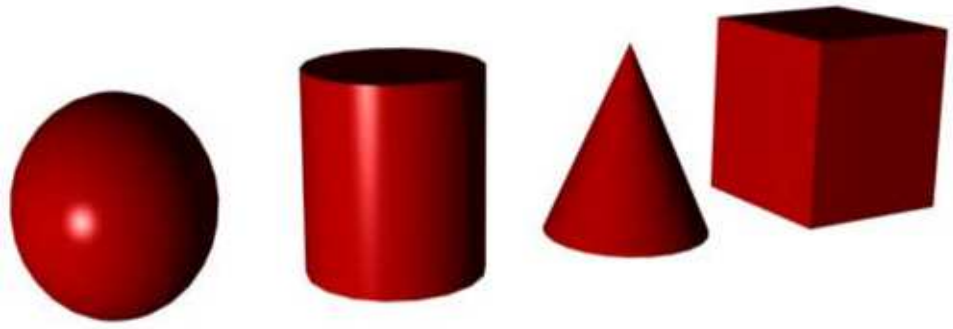
Kuva 5. *Standard view*

3.2 3D-mallien rakenne

Kolmiulotteinen mallinnus tapahtuu kolmiulotteisessa avaruudessa, jossa esineiden kohdat ja sijainti ilmoitetaan kolmen koordinaattiakselin X, Y ja Z avulla. Kappaleiden mittasuhteet esitetään reaalimaailman mittayksiköillä esimerkiksi, senttimetri, metri ja kilometri. Avaruus voi olla suuruudeltaan

rajaton millimetrin tarkoista mallinnuksista planeettojen kiertoratoja kuvaaviksi mallinnuksiksi. (Niemi 2011)

Kaikki kolmiulotteiset muodot saadaan aikaan neljän perusmuodon; laatikko, pallo, sylinteri ja kartio (Box, Ball, Disc ja Cone), avulla (kuva 6). Mallinnus-ohjelmat osaavat muodostaa perusmuodot eri kokoisina ja pituisina. Kolmiulotteisen mallinnuksen ajatuksena on muodostaa esine venyttämällä, pyörittämällä, leikkaamalla ja yhdistämällä näitä neljää muotoa. Työtapoja ja välineitä perusmuotojen käsittelyyn on loputtomasti. (Niemi 2011)

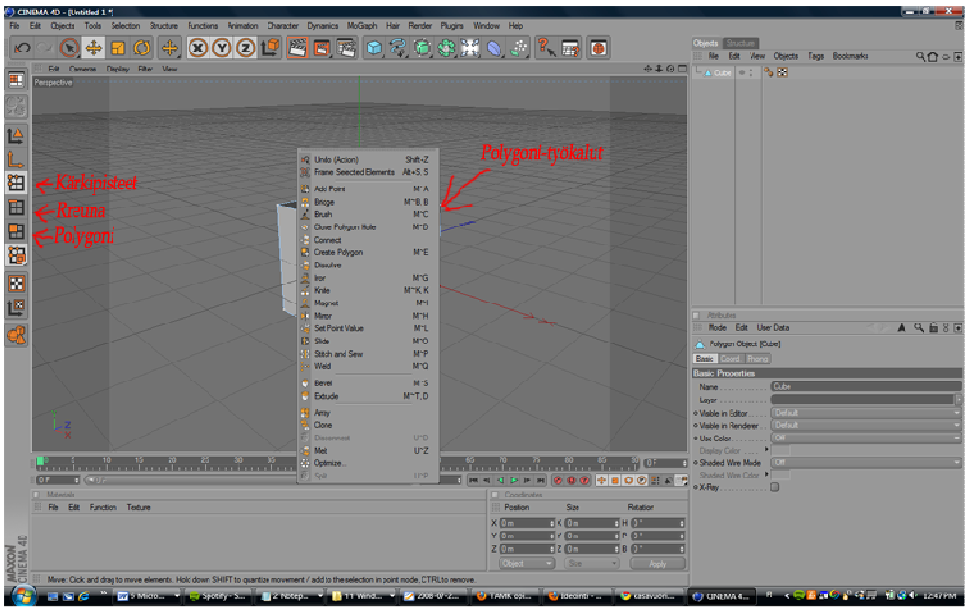


Kuva 6. Perusmuodot (Kämppi 2007)

3.2.1 Polygoni

Polygoneilla eli monikulmioilla kuvataan tietokonegrafiikassa kolmiulotteisia kappaleita, jotka muodostuvat kärkipisteiden verkosta, jotka määrittelevät kappaleen muodon. Yksinkertaisimmillaan polygoni muodostuu yhdestä pisteestä (Vector) tai kahdesta pisteestä (Particle). Kahden kärkipisteen yhdistävää viivaa sanotaan reunaksi (Edge). Tavallisesti polygoni on kolmen kärkipisteen muodostama kolmion. Monimutkaisempia muotoja voidaan muodostaa yhdistelemällä kärkipisteiden muodostamia kolmioita ja neliöitä halutun muodon aikaansaamiseksi. Yleensä polygoni-malli on näkyvä vain kappaleen sisäpuolelta, jolloin kappaleen sisäpinta jää näkymättömäksi. Tarvittaessa kappaleen molemmat puolet ovat mahdollista saada näkyviksi.

Polygonien valitsemiseksi Cinema4d:ssä löytyy valintatyökalut (Selection Tools), joiden avulla voi valita kärkipisteitä (Vector), reunoja (Edge) ja polygoneja (kuva 7). Cinema4d:stä löytyy rakennevalikosta (Structure Menu) useita Polygon-työkaluja eri efektien luomiseksi (kuva 7). Mallinnuksessa käyttämiäni työkaluja ovat puukko (Knife), polygonin luonti (Create polygon), pursutus (Extrude) ja yhdistäminen (Weld). Puukon avulla voi leikata pintoja pienemmiksi luoden uusia kärkipisteitä. Esimerkiksi neliön voi leikata keskeltä kahdeksi pienemmäksi neliöksi. Polygonin luonnin avulla voi luoda uusia polygoneja. Pursutus (Extrude) työkalulla voi pursuttaa mallista uusia pintoja. Yhdistämistyökalulla (Weld) avulla voi yhdistää kärkipisteitä, reunoja ja polygoneja.



Kuva 7. *Polygoni-työkalut*

Polygoni-mallinnuksessa käytettyjä tapoja ovat reunamallinnus ja laatikkomallinnus. Laatikkomallinnuksessa käytetään mallinnusohjelman Extrude inner ja Extrude-työkaluja. Extrude-inner-työkalulla pystytään pienentämään pintoja ja kulmia lisäten uusia pisteitä. Extrude-työkalulla pystytään pinnasta tai pinnoista luomaan uusia samankokoisia pintoja, joiden kärkipisteet yhdistyvät. Esimerkiksi neliöstä tulee laatikko. Reunamallinnuksessa yksittäisiä kärkipisteitä lisäämällä haetaan mallinnettavan mallin muotoa.

Low-poly-mallinnus on mallinnustekniikka, jossa pyritään mahdollisen vähillä kärkipisteillä saavuttamaan haluttu muoto. Tekniikkaa käytetään mallinnettaessa 3D-malleja, jotka on tarkoitettu sovelluksiin, joissa pyritään optimoimaan tietokoneen suorituskykyä. Tekniikkaa on käytetty tietokonepeleihin ja muihin sovelluksiin, jotka sisältävät reaaliaikaista kolmiulotteista grafiikkaa. (Polygon mesh 2011)

3.2.2 Nurbs

Non-uniform rational basis spline (Nurbs) on matemaattinen malli, jossa kappaleen pinnan muotoa muotoillaan matemaattisesti. Mallia on käytetty yleisesti CAD-suunnitelussa ja 3D-mallinnuksessa. Mallinnus-ohjelmien Nurbs-ominaisuuksien avulla pystytään kulmikkaiden kappaleiden pintoja pyöristämään kontrolli-parametreja (hallintakohta, käyrän järjestys ja solmuvektori) muuttamalla. Nurbs-käyrä lisää 3D-malliin uusia kärkipisteitä kehikoksi käyrän suuntaisesti pyöristäen mallinnettavan kappaleen muotoa. (Non-uniform rational B-spline 2011)

3.3 Cinema4d:n Nurbs-ominaisuudet

3.3.1 HyperNurbs

HyperNurbs on yksi tehokkaimista keinoista, joita 3D-muotoilija voi käyttää kolmiulotteisen mallin muotoa hakiessaan. Tekniikkaa on käytetty erityisesti hahmomallinnuksessa, mutta sitä voidaan käyttää lähes missä tahansa mallissa urheiluautoista huonekaluihin. HyperNurbs käyttää algoritmia jakamaan pinnan pienempiin osiin pyöristäen pinnan muotoa. Nurbsin kärkipisteiden ja reunojen punnitsemis-ominaisuuksien avulla saadaan aikaan vakaampia muotoja. (MAXON Manual 2010)

3.3.2 Extrude Object

Extrude-objektin avulla voidaan viivan (Spline) muoto pursuttaa paksummaksi kappaleeksi. Pursutus tapahtuu, kun Extrude-objektin lapsiobjektiksi lisätään piirretty viiva. Viivan avulla voidaan tehdä reikiä pursutettuun kappaleeseen esimerkiksi, jos viiva on piirretty ympyräksi ja sen sisällä on toinen ympyrä, niin sisempi ympyrä luo aukon muotoon pursutuksen yhteydessä (kuva 8). (MAXON Manual 2010)

Kuva 8. Extrude object (MAXON Manual 2010b)

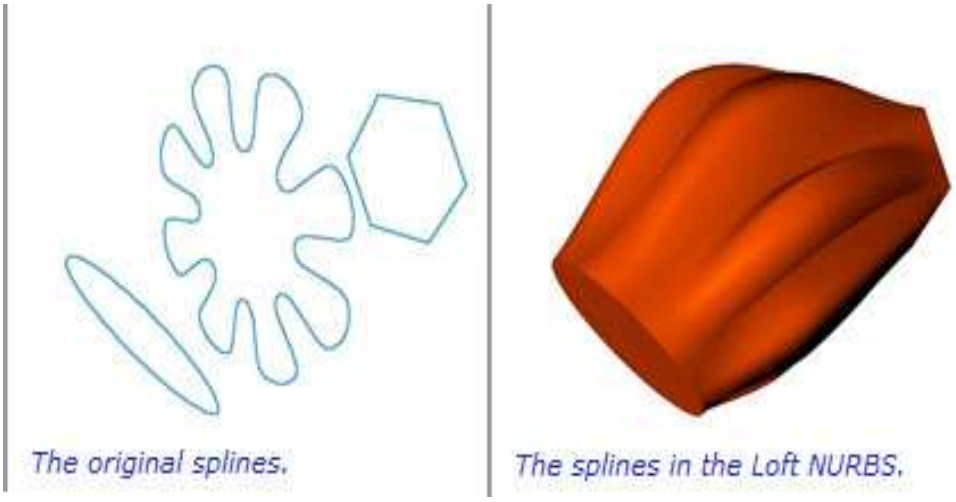
3.3.3 Lathe Object

Lathe-objekti luo viivan avulla piirretystä profiilista pyöreän muodon pyöräyttäen ylhäältäpäin katsottuna viivaa ympäri pitäen profiilin muodon samana. Muoto saadaan aikaiseksi, kun lisää viivan Lathe-objektin lapsi objektiksi (kuva 9). (MAXON Manual 2010)

Kuva 9. Lathe object (MAXON Manual 2010c)

3.3.4 Loft Object

Loft-objekti venyttää pinna useamman viivan (spline) muodon välille. Viivan muotojen järjestys Loft-objektin lapsiobjekteina määrää muotojen järjestyksen (kuva 10). (MAXON Manual 2010)



Kuva 10. Loft object (MAXON Manual 2010d)

3.3.5 Sweep Object

Sweep-objekti tarvitsee kaksi tai kolme viivaa (Spline). Ensimmäinen viiva määrää korkeuden, toinen käyrä määrittelee poikkileikkauksen, joka muodostaa luotavan muodon polun. Kolmannen vaihtoehtoisen viivan avulla voidaan määritellä kappaleen rata (kuva 11). (MAXON Manual 2010a)

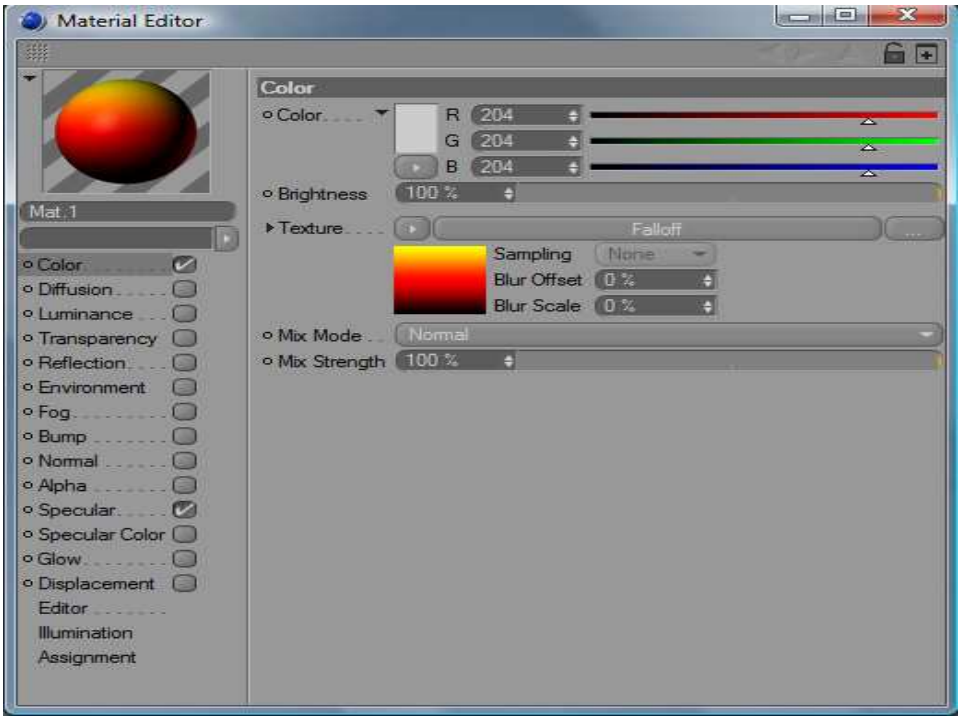
Kuva 11. Sweep object (MAXON Manual 2010e)

3.4 Materiaalit ja tekstuurit Cinema4d:ssä

3D-mallinnusohjelmissa mallinnetut mallit saadaan herätettyä henkiin pintamateriaalien ja tekstuurien avulla. Mallinnusohjelmilla voidaan jäljitellä reaalimaailmassa esiintyviä materiaaleja kuten kiveä, vettä, kangasta, puuta tai mitä tahansa reaalimaailman materiaalia. Mallinnusohjelmaan voidaan tuoda valokuva halutusta pinnoitteesta pintakuvioksi lisäämään realismia tai pinnan-ominaisuudet voidaan luoda mallinnusohjelmien materiaalivalikon perus-ominaisuuksien avulla. Yleisiä materiaalien perus-ominaisuuksia ovat väri (Color), diffuusio (Diffusion), valoisuus (Luminance), läpinäkyvyys (Transparency), heijastus (Reflection), ympäristö (Environment), sumu (Fog), syvyys (Bump), normaali (Normal), Alpha, Specular, Specular-Color, kiilto(Glow) ja Displacement.

Väri-ominaisuuden avulla pystytään luomaan mallille perusväri. *Diffuusion* avulla mallille voidaan luoda epäsäännöllinen rosoinen pintakuvio. *Valoisuuden* avulla mallille voidaan luoda itsevalaiseva ominaisuus, jonka kirkkautta voidaan säädellä. *Läpinäkyvyyden* avulla voidaan määritellään kappaleen läpinäkyvyysarvo. *Heijastuksen* avulla voidaan määritellä mallin pinnan heijastus eli peilaava vaikutus. *Ympäristö*-ominaisuudella voidaan simuloida kontrollitekstuurin avulla ympäristön heijastusta. *Sumu-*

ominaisuuden avulla mallille luodaan sumu-efekti. *Syvyyskartan* avulla voidaan pinnanmuodon syvyyttä säädellä muuttamatta geometrista muotoa. *Normaali* kanavan avulla matalaresoluutioisille malleille voidaan luoda korkearesoluutioinen ulkoasu. *Alpha*-kanavan avulla voidaan säädellä materiaalin läpinäkyvyyttä valon ja värien avulla. Tekstuurin valkoiset kohdat ovat läpinäkyviä. *Specular*-ominaisuuden avulla voi kappaleen pinnalle luoda korostettuja kohtia, missä valon määrä korostuu. *SpecularColor*-ominaisuuden avulla voidaan vaikuttaa Specular-heijastuksen väriin. *Kiilto*-ominaisuuden avulla voidaan mallista tehdä hohtava. *Displacement* käyttää normaali ja syvyyskartta (Bump) kanavoita muuttaen mallin geometrista pinnanmuotoa. Kuvassa näkymä Cinema4D materiaali editorista (kuva 12). (CINEMA 4D R11 Quickstart – Materials 2011)



Kuva 12. *Material editor*

3.4.1 Tekstuurien luonti

Teksturoinnilla tarkoitetaan kolmiulotteisen mallin päällystämistä bittikarttakuvalla eli tekstuurilla. Tekstuurit muodostuvat yleensä valokuvista, jotka esittävät oikeanmaailman kohteiden aitoja pintoja. Aitojen pintojen käyttäminen tekstuurin pohjana nopeuttaa työskentelyä, koska halutunlaisen pin-

nan-ominaisuuksien piirtäminen kuvankäsittelyohjelmalla veisi tekijältään paljon aikaa. Tekstuurien avulla pyritään luomaan realistisempia materiaaleja. Tekstuurit voivat olla monenlaista kuvaformaattia. Yleisimpänä kuvantallennusformaattina tekstuureissa käytetään JPEG-kuvaformaattia. Samassa tekstuurissa saattaa olla samanaikaisesti useita eri pintamateriaaleja.

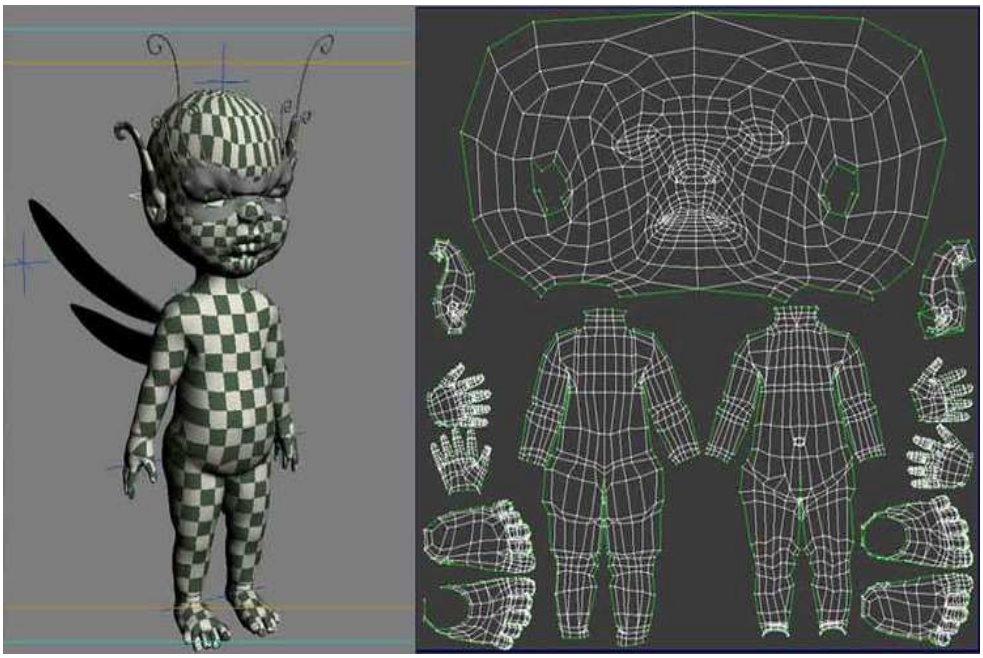
Toinen tapa tehdä tekstuureja on käyttää mallinnusohjelmien shaderi (Shader tai Procedural Texture) ominaisuuksia, tämä on edistyneempi tekstuurimuoto kuin tavalliset tekstuurikartat. Shader-tekstuurit tehdään matemaattisten kaavojen avulla. Tekstuurin tyypin etuna on läheltä katsottaessa tasainen pinta, kun taas tavalliset tekstuurit saattavat näyttää rakeisilta pikseleiden näkyessä. Pikselit tarkoittavat pisteitä, joiden avulla tietokone muodostaa kuvan ruudulle. Shader-tekstuurien avulla voidaan toteuttaa ainutlaatuisia pintoja, jota ei tavallisilla tekstuureilla kartoilla tai muilla metodeilla voida luoda. Shaderi-tekstuuri toimii paremmin vuorovaikutuksessa kolmiulotteisen näyttämän muiden ominaisuuksien kanssa kuin tavallinen tekstuuri. Tekstuuri ottaa huomioon muuttaen pintaansa näyttämän valon intensiteetin, suunnan, kameran sijainnin ja muiden näyttämön muuttujien vaihtelun. Shadereitä on kaksiulotteisia ja kolmiulotteisia. 2D-sheiderit levittävät muodostamansa kuvan 3D-objektin pinnalle tavallisten tekstuurien tapaan. 3D-sheiderit ovat riippumattomia 3D-objektin geometriasta ja projektio tavasta. (MAXON Manual 2010f)

3.4.2 UV-Mapping

UV-kartoituksella (UV-mapping) tarkoitetaan kolmiulotteisen polygonimallin päällystämistä kaksiulotteisella kuvalla, jonka yksityiskohdat voidaan määrittää tarkasti vastaamaan 3D-malllin osia. Käytettävää kuvaa kutsutaan UV-tekstuurikartaksi (UV-texturemap). Karttaan on tallennettu tieto kolmiulotteisenmallin eri osasista eli polygoneista, joita käytetään materiaalin sijoittelemiseen oikeille paikoilleen. Tekniikka mahdollistaa usean eri pinnoitteen käyttämistä samassa tekstuurissa, jotka voidaan tarkasti sijoittaa 3D-mallin pinnalle. UV-kartoitusta käytetään luotaessa yksityiskohtaisia

malleja, jotka sisältävät useita eri pintoja. Yleisimpinä käyttökohteina ovat erilaiset ihmishahmot ja eläinhahmot.

Kolmiulotteinen mallin pinta levitetään kaksiulotteiseen UV-karttatekstuuriin ohjelmien automaattisilla toiminnoilla tai luodaan manuaalisesti. Ohjelmien automaattisilla toiminnoilla ei aina saada halutunlaista lopputulosta ja tekstuurissa on päällekkäisyyksiä, sekä tekstuuri saattaa näkyä väärässä kohtaa mallin pinnalla. Yleensä joudutaankin muotoilemaan ja optimoimaan tekstuurin asentoa kaksiulotteisella pinnalla uudestaan. Mallinnusohjelmissa löytyy laaja kirjo ominaisuuksia miten 3D-mallin pinta voidaan levittää ja asemoida kaksiulotteisen tekstuurin pinnalle. Kuvassa esimerkki kolmiulotteisen hahmoa UV-kartasta, jossa näkyy miten hahmon pinnalta muodostettu kartta levittyy UV-karttatekstuuriin (kuva 13). (UV-Mapping 2011)



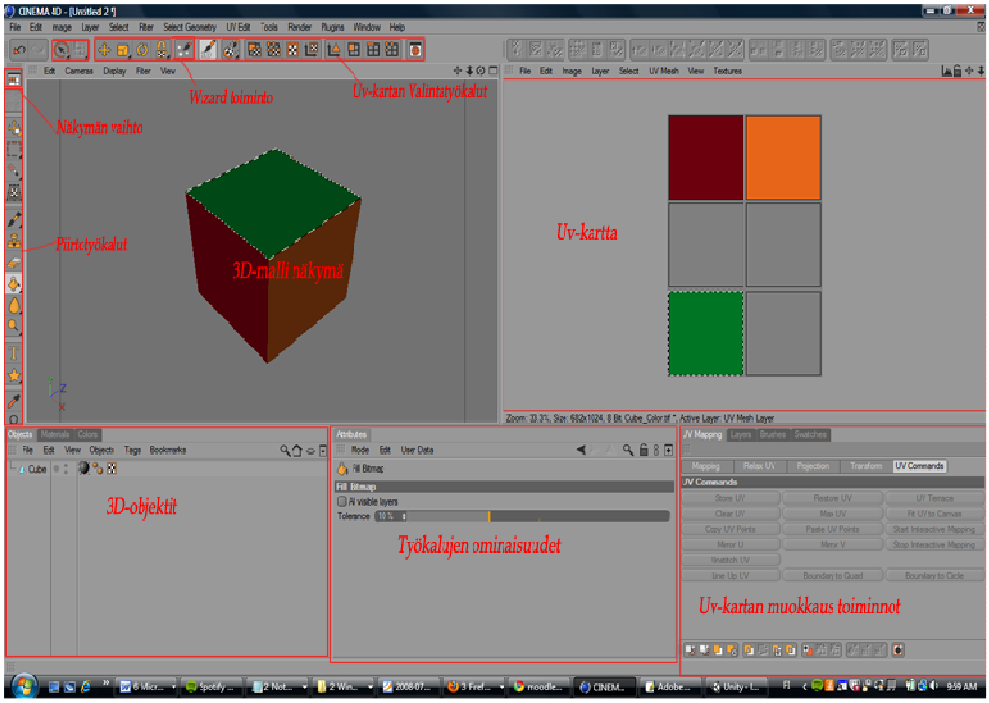
Kuva 13. UV-kartta (Sprites and billboarding 2011)

3.4.3 Cinema4d UV-editori

Cinema4d-ohjelmassa löytyy oma editori UV-karttojen tekoon, editorista löytyy monipuoliset ominaisuudet karttojen muokkaukseen. Editorinäkymästä tärkeimpinä käyttäminäni toimintoina löytyy velhotoiminto

(Wizard), UV-kartan valintatyökalut, piirtotyökalut, 3D-mallinäkymä, UV-kartta, 3D-objektit, Työkalujen-ominaisuudet, UV-kartan muokkaus toiminnot näkymät (kuva 14).

Wizard-toiminnon avulla voi automaattisesti luoda UV-karttan 3D-mallista. UV-kartan valintatyökaluilla voi valita kartasta osia, johon voi sijoittaa pintamateriaaleja. Piirtotyökaluilla pystyy kartan osia muokkaamaan. 3D-malli näkymässä näkyy 3D-malli UV-karttaan pintamateriaalin kanssa. UV-kartta näkymässä näkyy mallin osista muodostuva UV-kartta. 3D-Objektit näkymässä näkyy kaikki Scene:ssä olevat 3D-objektit. Työkalujen-ominaisuudet kohdasta voi asettaa eri työkalujen-ominaisuuksia. UV-kartan muokkaus toiminnot näkymästä löytyy työkaluja UV-kartan manuaaliseen muokkaamiseen.



Kuva 14. *UV-editori*

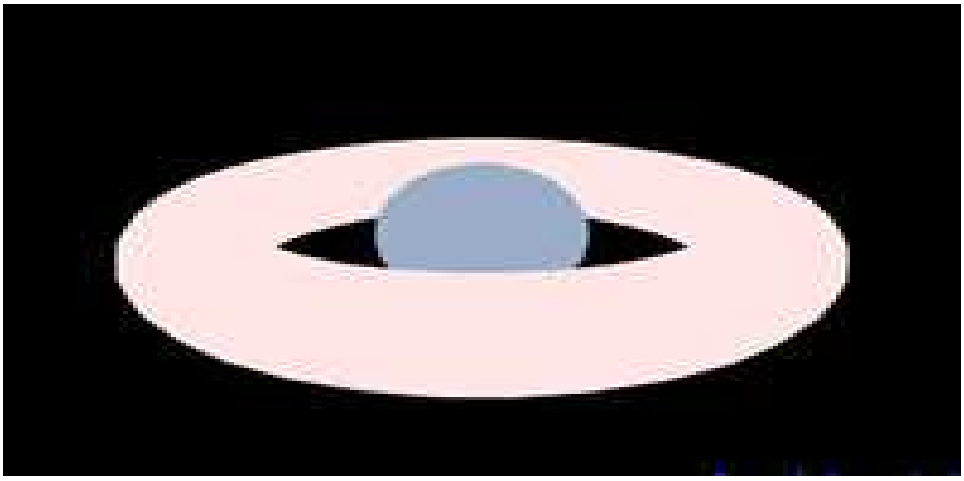
3.5 Valaistus

Valaistuksella on iso merkitys mallinnuksessa, koska valon avulla saadaan mallit näkyviksi ja eri pinnan-ominaisuudet toimimaan oikeanlaisina. Mallinnusohjelmissa valo käyttäytyy kolmiulotteisella näyttämöllä samankal-

taisesti kuin reaalimaailmassa luoden varjoja ja kiiltäviä pintoja riippuen pintojen materiaaleista. Mallinnusohjelmissa on monenlaisia valonlähteitä esimerkiksi kohdevaloja, itsevalaisevia materiaaleja, hohtavia materiaaleja, aurinkovalon kaltaisia valonlähteitä ja tiettyä aluetta valaisevia valoja. Valaistuksen ohella valonlähteistä syntyvät varjot ja heijastukset näyttelevät isoa roolia kolmiulotteisen näyttämön visuaalisessa ulkoasussa. Valon avulla voidaan jäljitellä monenlaisia tunnelmia auringonnoususta iltahämärään. Valon väriä vaihtelemalla voidaan luoda erilaisia vaikutelmia lämpimästä tai kylmästä valon lähteestä. Valon voimakkuutta voi säätää mallinnusohjelmissa löytyvän valonintensiteettiasetusten avulla. 3D-mallinnuksessa käytettävät yleiset valaisintyypit ovat yleisvalo, pistevalo, kohdevalo, loputon valo, viivavalo, näkyvä valo ja tasovalo.

3.5.1 Yleisvalo

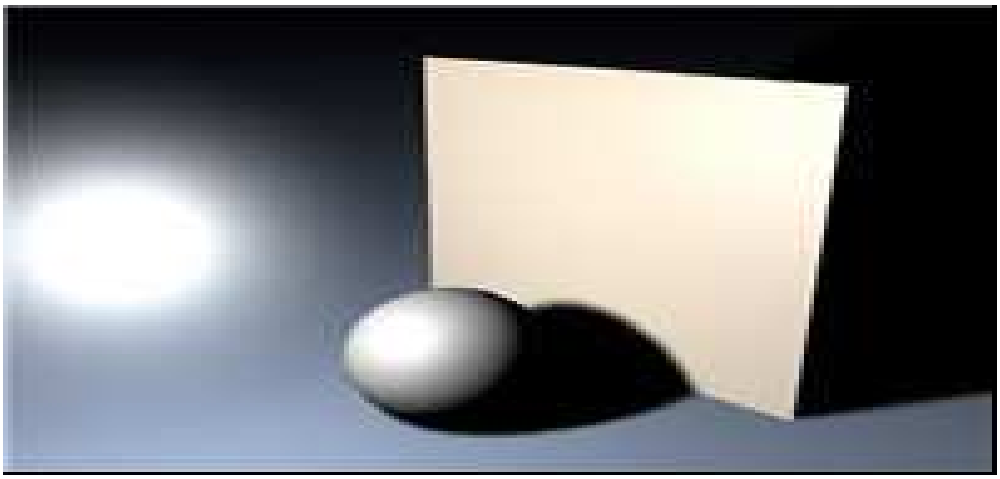
Yleisvalo (Ambient Light) on tasaista valoa, joka valaisee 3D-mallien kaikki pinnat valon tulokulmasta riippumatta (kuva 15). Useimmissa mallinnusohjelmissa oletusvalona käytetään yleisvaloa. Yleisvalo ei luo mallinpinnoista varjoja ja ainoastaan materiaalin väri vaikuttaa valon luonti laskelmiin. (MAXON Manual 2010)



Kuva15. Yleisvalo (MAXON Manual 2010g)

3.5.2 Pistevalo

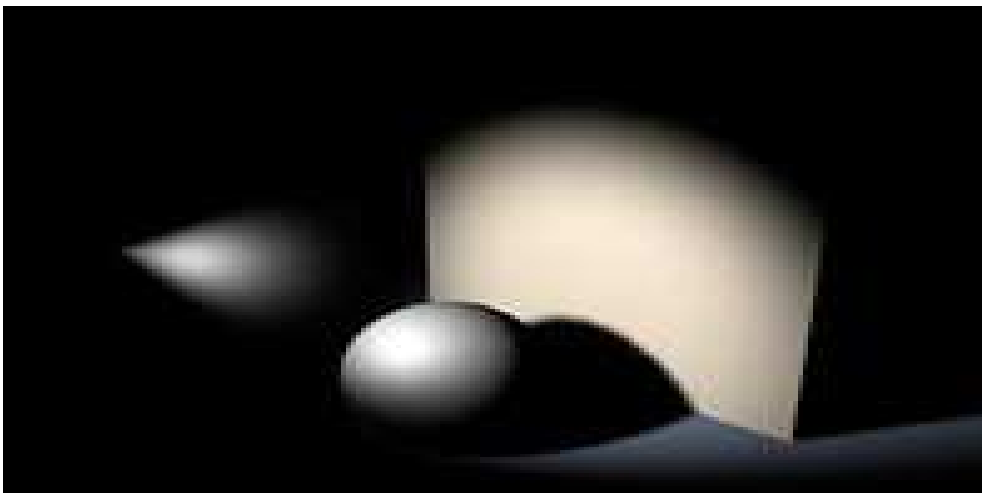
Pistevalo (Point Light/Omni Light) käyttää samankaltaisesti kuin hehkulamppu lähettäen valonsäteitä joka suuntaan (kuva 16). Jos pistevalo laitetaan keskelle kolmiulotteista näyttämöä, niin pistevalo valaisee tasaisesti näyttämöä. (MAXON Manual 2010)



Kuva 16. Pistevalo (MAXON Manual 2010h)

3.5.3 Kohdevalo

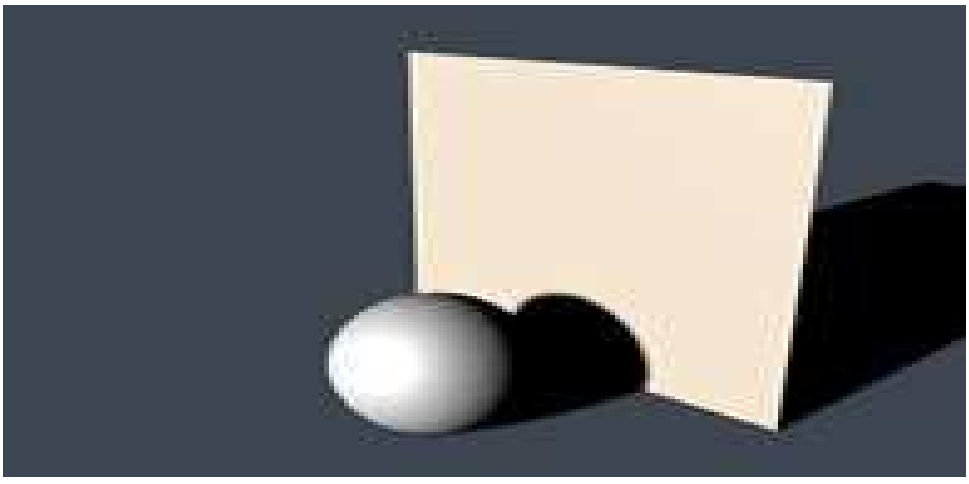
Kohdevalo (Spotlight) valaisee säteillään yhteen suuntaan oletusarvona kolmiulotteisen näyttämön Z-akseli (kuva 17). Kohdevaloa on helppo kohdistaa ja siirrellä eri kohteisiin. Kohdevalon valokeila voi olla pyöreän tai neliön ja valon säteiden näkyvyyden pituutta voi säädellä. (MAXON Manual 2010)



Kuva 17. Kohdevalo (MAXON Manual 2010i)

3.5.4 Loputonvalo

Loputon valo (Infinite Light) avulla voidaan muodostaa valoa, jonka säteet kulkevat samaan suuntaan loputtomasti (kuva 18). Valon tyypin avulla voidaan hyvin simuloida auringonvaloa. Valon säteiden suuntaa voidaan helposti muuttaa. (MAXON Manual 2010)

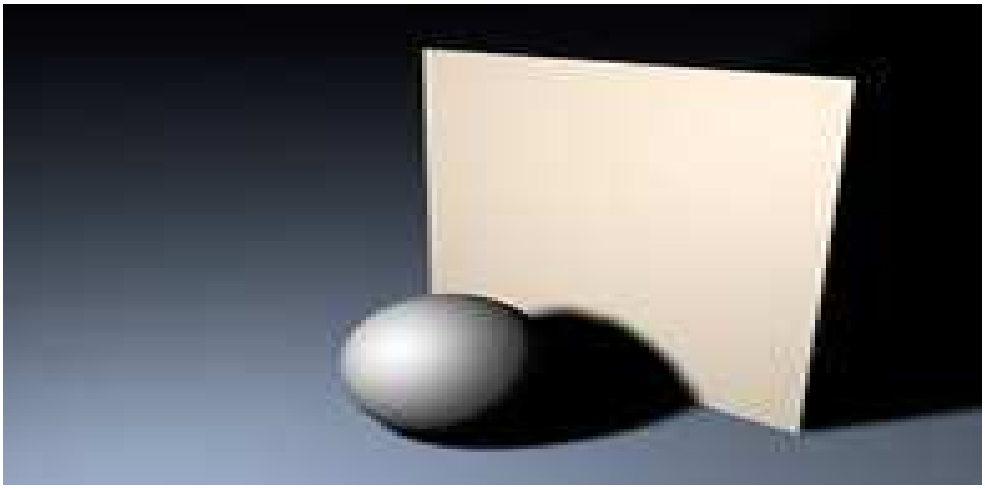


Kuva 18. Loputonvalo (MAXON Manual 2010j)

3.5.5 Tasovalo

Tasovalo (Area Light) lähettää valonsäteitä joka suuntaan koko neliön muotoisen pinta-alansa alueelta (kuva 19). Pinnan kokoa voidaan helposti

säädellä. Tietokoneen ruudun valo on hyvä esimerkki tasovalosta. (MAXON Manual 2010)



Kuva 19. Tasovalo (MAXON Manual 2010k)

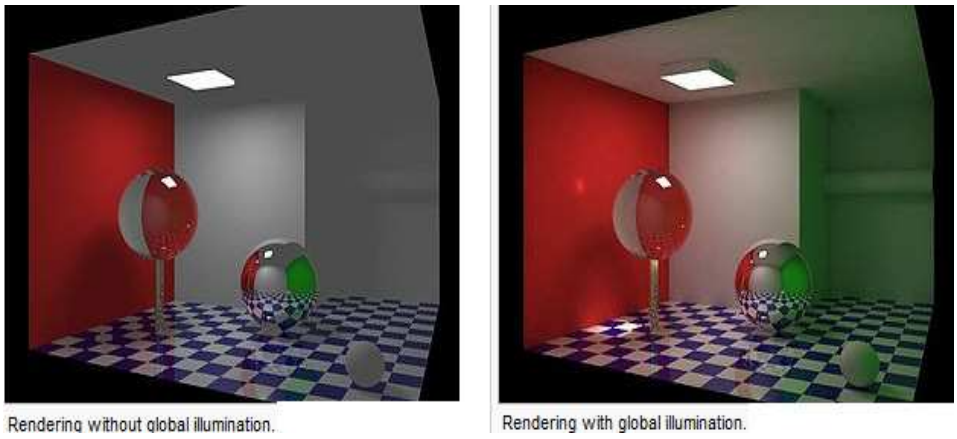
3.5.6 Näkyvä valo

Näkyvä valo (Visible Light) lähettää valonsäteitä, jotka luovat säteiden suuntaisen näkyvän valokeilan (kuva 20). Valon valonsäteet läpäisevät kaikki 3D-objektit. Näkyvää valoa käytetään monissa eri visuaalisissa efekteissä kuten savussa, pilvissä ja tulessa. (MAXON Manual 2010)

Kuva 20. Näkyvä valo (MAXON Manual 2010l)

3.5.7 Global Illumination

Globaalin valaistuksessa (Global Illumination) pyritään luomaan algoritmejen avulla realistisempi valaistus kolmiulotteiselle näyttämölle. Algoritmit ottavat huomioon valon, joka tulee suoraan valon lähteestä (suoravalaistus) ja valonsäteet mitkä heijastuvat muista pinnoista (epäsuoravalaistus). Näyttämöllä olevat kolmiulotteiset objektit vaikuttavat toisiinsa niistä syntyvien varjojen ja valonsäteiden heijastusten kautta. Objektien pintamateriaalien heijastus-ominaisuudet vaikuttavat heijastusten voimakkuuteen. Kuvassa on esimerkki globaalista valaistuksesta (kuva 21). (Global illumination 2011)



Kuva 21 Global illumination (Global Illumination 2011)

3.5.8 HDRI

High-Dynamic-Range Imaging on tekniikka, joka mahdollistaa suuremman valoisuusarvon tummimman ja valoisimman alueen välillä kuvassa, niin että yksityiskohdat säilyvät tarkkoina. HDRI-kirkkausaluetta tarkoittavaa dynamiikkaa mitataan aukkoina siten, että jokainen aukko kaksinkertaistaa kirkkauden. Kuvien muodostamiseen on kaksi tekotapaa; valotushaarukointi ja kahden samaa valotusta olevan kuvan yhdistäminen. Ensimmäinen soveltuu laajemman dynaamisenalueen tallentamiseen. HDRI-valaistusta käytetään kolmiulotteisissa sovelluksissa lisäämään realismia visuaaliseen ulkoasuun. (HDR 2011)

3.5.9 Lightmaps

Valaistuskartat (Lightmaps) ovat tekstuureja, joiden pintaan on tallennettu kohteen tai tilan valaistuksen luomat varjot ja heijastukset. Ennen valaistuskartan tekoa pitää suunnitella ja luoda kolmiulotteiselle näyttämölle valaistus. Valaistuksen ollessa oikeanlainen voidaan se tallentaa 3D-objektien valaistuskartta tekstuureihin. Jokaiselle mallille täytyy luoda oma tekstuurinsa kuvaamaan valon käyttäytymistä kohteessa. Valaistuskarttoja pystytään luomaan mallinnusohjelman automaattisella toiminnolla, joka tekee erillisen valaistuskartta tekstuurin.

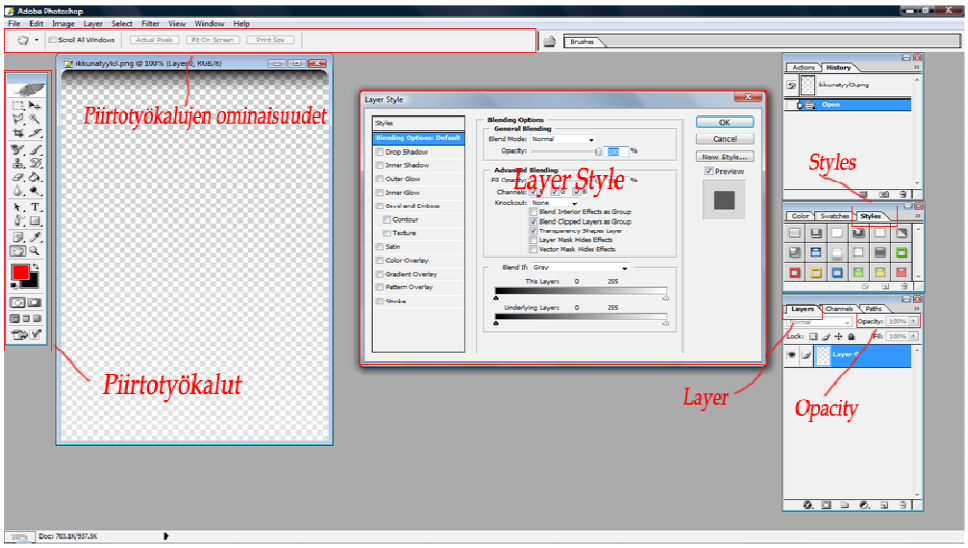
Valaistuskarttoja käytetään reaaliaikaisissa kolmiulotteisissa sovelluksissa kuten peleissä, joissa pyritään nopeuttamaan tietokoneen suorituskykyä. Valaistuskarttojen avulla vähennetään prosessorille lähetettäviä laskutehtäviä, koska tilan valaistuksessa syntyneet valot, varjot ja heijastukset ovat tallennettu 3D-objektien materiaaleihin, eikä tietokoneen tarvitse laskea valon lähteistä lähtevien valonsäteiden aiheuttamia muutoksia materiaaleissa reaaliaikaisesti. Cinema4d-ohjelmassa löytyy automaattinen Bake-toiminto Render-valikossa, jonka avulla helposti asetetusta valaistuksesta pystytään luomaan valaistuskartta tekstuurit. (Lightmaps 2011)

3.6 Photoshop

Käytin käytännön työn tekemisessä kuvankäsittelyyn Adoben julkaisemaa Photoshop-ohjelmaa. Ohjelmasta löytyy kattavat ominaisuudet ammattimaiseen kuvankäsittelyyn. Ohjelman sisältä löytyy paljon valmiita tyyliominaisuuksia. Tein ohjelmalla opinnäytetyön käytännön osaan valikossa tarvittavia painikekuvia ja tietoikkunapohjia. Käytin ohjelmaa hakiessani valikoille oikeanlaista tyyliä hahmottelemalla erilaisia valikko luonnostelmia. Käytin työssäni Photoshopin Styles, Opacity, Layer, Layer style ja piirtotyökalu-ominaisuuksia luodessani valikkojentyylejä (kuva 22).

Styles-valikosta voi valitulle kohteella asettaa Photoshopin tyylejä valmiista tyylivalikoimasta. Opacity-ominaisuudella voidaan prosentuaalisen

muuttujan avulla asettaa läpinäkyvyysarvo valitulle kohteelle. Layer-valikon avulla voi kuvaan lisätä uusia kerroksia, jotka näkyvät edellisen kerroksen päällä. Layer-style-valikosta voi asettaa eri tehosteita valitulle kohteelle. Piirtotyökalulla luodaan ja muokataan grafiikkaa ja kuvia.



Kuva 22. *Photoshop*

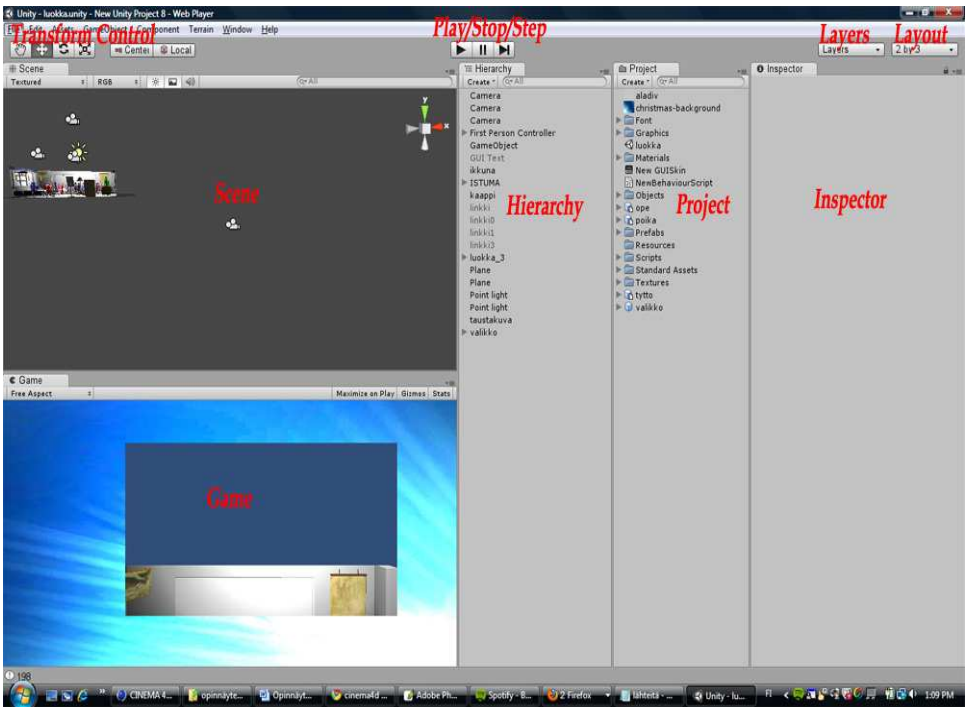
4 UNITY3D-PELINKEHITYSTYÖKALU

Opinnäytetyössä päädyin käyttämän Unity3D-pelimoottoria, koska se osoittautui ominaisuuksiltaan tehokkaaksi toteutusalustaksi kolmiulotteisen grafiikan esittämiseksi WWW-sivulla ja ohjelma on helppokäyttöinen. Tässä luvussa käyn läpi ohjelman perus-ominaisuuksia, jotka liittyvät valmiin sovelluksen tekemiseen.

Unity3D on monialustainen pelimoottori, joka mahdollistaa monipuolisen kolmiulotteisen ja kaksiulotteisen sisällöntuotannon peleistä erilaisiin kolmiulotteisiin visualisointeihin. Unityn avulla voi tehdä sovelluksia Windows-, Mac-, Wii-, Xbox360-, Playsstation3-, Ipad-, Iphone-, Android- ja Web-alustoille. Julkaiseminen eri alustoille tapahtuu helposti Unityn omalla ominaisuudella, joka mahdollistaa helpolla rasti ruutuun valinnalla alustan valitsemisen. Sama sovellus voidaan helposti kääntää kaikille alustoille. Unity-pelimoottorin avulla voi tehdä visuaalisesti näyttäviä pelejä ja sovelluksia. Moottori sisältää useita käyttövalmiita graafisia-efektejä ja tehokkaan valaistuksen hallinnan. Pelimoottorissa on helppo ohjelmointirajapinta, jonka avulla voidaan luoda toiminnallisuutta sovelluksiin. Moottori tarjoaa myös hyvän fysiikka mallinnuksen ominaisuudet. (Unity 2011)

4.1 Käyttöliittymä

Unity-editorin käyttöliittymä sisältää useita eri näkymiä, joiden avulla hallinnoidaan rakenteilla olevaa sovellusta. Käyttöliittymän näkymää on mahdollista muokata omanlaiseksi. Perusnäkymä sisältää näyttämön (Scene), kameran luoman pelinäkymän (Game), sovelluksen peliobjektien hierarkia-näkymän (Hierarchy), sovelluksen resurssien-näkymän (Project), objektien ominaisuuksien hallinnan-näkymän (Inspector), kerros-näkymän (Layer), käyttöliittymän taitto-ominaisuudet (Layout), näyttämön-hallintatyökalut (Transform tools) ja play/pause/step-painikkeet sovelluksen ajamiseen (kuva 23).



Kuva 23. *Perusnäkymä*

4.2 Scenen luominen

Näyttämö (Scene) on näkymä, joissa sovelluksen peliobjekteista (Gameobject) järjestetään suunniteltu sovellusympäristö. Objekteja voi liikutella ja sijoitella näyttämöllä hallintatyökalulla (Transform tool). Sovellus tai peli voi muodostua useista kerroksista, Scene edustaa yhtä tasoa. Scenen näkymää voi katsella akselien joka suunnalta. Näkymää on helppo vaihtaa Scenen vasemmassa yläreunassa olevan näkymän hallinnan avulla (Transform tool).

4.2.1 GameObject

Peliobjekteilla (GameObject) tarkoitetaan Unity-editorissa kaikkia peliin tuotavia tai luotavia osasia, joista sovellus tai peli muodostuu. Peliobjekteja on hyvin eri tyyppisiä. Erilaisia peliobjekteja ovat 3D-mallit, valot, kamerat, käyttöliittymä scriptit, äänet, 2D-kuvat, jne. Objekteja voi ajatella säilytystiloina, jonka sisällön eri osaset määrittelevät minkälainen peliobjekti on kyseessä. Objekteille voi lisätä uusia ominaisuuksia, joita voi kontrolloida

ominaisuuksien hallinta-näkymässä (Inspector). Jokaisella objektilla on automaattisesti sijainti (transform) ominaisuus, jonka avulla peliobjektia voi liikutella X, Y ja Z- akselilla kolmiulotteisella näyttämöllä. (GameObjects. 2010)

4.2.2 Import-tuonti

Suurin osa Unity-projekteissa käytettävistä peliobjekteista on tehty jollain muulla ohjelmalla. Ohjelmassa löytyy oma toimintonsa Import asset, jonka avulla ohjelman resurssienhallinta näkymään (Project) voidaan tuoda muilla ohjelmilla tehtyjä peliobjekteja. Unity-editoriin voidaan myös lisätä uusia objekteja tallentamalla ne projektikansioon, jolloin objektit automaattisesti näkyvät Unity-editorissa. Ohjelma luo automaattisesti uudelle projektille oman kansion. Unity-editori tukee monenlaisia tiedostomuotoja. Yleisempiä ohjelmaan tuotavia objekteja ovat 3D-mallit, animaatiot, tekstuuritiedostot ja äänitiedostot. (Importing asset 2010)

4.3 Ohjelmointi

Ohjelmointi Unityssä tapahtuu ohjelman oman editorin kautta. Editorista löytyy oma virheenkorjaustoiminto (Debug). Toiminnallisuuden luominen on mahdollista myös muilla editoreilla. Uusi tiedosto, johon ohjelmakoodi (New Behaviour Script) kirjoitetaan, luodaan Unityn-yläpudotusvalikosta Asset->create-> ja Javascript- tai C-ohjelmointikieli. Ohjelmoitu toiminnallisuus liitetään peliobjekteihin, joita halutaan kontrolloida tai tyhjään peliobjektiin. Ohjelmakoodi tarvitsee toimiakseen aina peliobjektin, johon se on liitetty. Scripti ja kontrolliparametrit näkyvät Inspector-näkymässä, joista arvoja voidaan muuttaa ohjelmanajon aikana reaaliaikaisesti. Toiminnallisuuden tekemiseen Unity:ssä löytyy laaja kirjo valmiita funktioita. Ohjelman kotisivuilta löytyy selkeä ohjelmointimanuaali, jossa opastetaan ohjelmointikäytäntöjä ja funktioiden käyttöä. Unity:ssä on mahdollista ohjelmoida Javascript-, Boo- ja C-ohjelmointikielillä. (Unity Script 2011)

4.3.1 Javascript

Päädyin käyttämään Javascript-kieltä, koska olen opiskellut aikaisemmin Java-kieltä, vaikka Javalla ja Javascriptllä on vähän yhtäläisyyksiä, niin oli se lähempänä omia käyttökokemuksiani. Javascript on Netscape Communications Corporationin kehittämä pääsääntöisesti Web-ympäristöön tarkoitettu oliopohjainen ohjelmointikieli, jota nykyiset selaimet laajaalaisesti tukevat. Javascriptin syntaksi perustuu väljästi C-ohjelmointikieleen. Javascript on scriptikieli, jonka koodi tulkitaan isäntäympäristön, kuten selaimen ohjelmatulkin avulla. Javascriptiä suoritettaessa koodi tulkitaan rivi kerrallaan ilman käännöstä omaksi ohjelmakseen. Nimensä mukaan voisi luulla Javascriptillä olevan yhteys Java-ohjelmointikieleen, mutta yhtäläisyyksiä on hyvin vähän kielien syntaksissa. Javascriptin vahvuus on käyttöjärjestelmä riippumattomuudessa, keveydessä, tehokkuudessa ja yksinkertaisessa vapaamuotoisessa syntaksissa. (JavaScript-kielen alkeet - osa 1 2010)

4.4 UnityGui

Gui tulee sanoista Graafinen käyttöliittymä (Graphical User Interface). Unity-editorissa käyttöliittymiä luodessa voidaan käyttää GuiText-objektia ja GuiTexture-objektia tai UnityGui-käyttöliittymäohjelmointia. GuiTextin avulla voi luoda millä fontilla tahansa tekstiä ruudun annettuihin koordinaatteihin. GuiTexturen avulla voi ruudulle lisätä 2D-kuvaan annettuihin koordinaatteihin. Kuvasta tai tekstistä on mahdollista tehdä painike, joiden avulla sovelluksen valikkorakenne on mahdollista toteuttaa.

UnityGui-ohjelmointi tapahtuu OnGui-funktion sisällä muutamia poikkeuksia lukuunottamatta. OnGui on funktio, joka päivittyy aina, kun tietokone piirtää kuvan tietokoneenruudulle. Jokaiselle Gui-kontrollille pitää määritellä koko ja sijainti ruudun ylälaidasta ja alalaidasta. Kontrolleista voi myös tehdä skaalautuvia sovelluksen ikkunaan nähden määrittelemällä halutun kontrollin prosentuaalinen koko suhteessa koko sovellusikkunaan. Käyttöliittymän eri osasille tarvitsee myös määritellä, mitä esimerkiksi ta-

pahtuu, jos painiketta painetaan. Kontrollien toiminnallisuutta vastaavat tapahtumat määritellään päälle pois menetelmällä esimerkiksi, jos tietyt ehdot täyttyvät, niin haluttu tapahtuma tapahtuu. Ohjelmointikielissä löytyy erilaisia operaattoreita ehtojen toteuttamiseen.

UnityGui-käyttöliittymäohjelmoinnissa löytyy useanlaisia Gui-kontrolleja (Gui control), joiden avulla sovelluksen tai pelin valikkorakenne voidaan toteuttaa. Erilaisia Gui-kontrolleja ovat painike (Button), työkalurivi (Toolbar), etiketti (Label), tekstikenttä (Textfields), tekstialue (TextArea), vaihtopainike (Toggle), valintataulukko (Selection Grid), vaakasuuntainen vierityspalkki (Horizantal Scrollbar), pystysuuntainen vierityspalkki (Vertical Scrollbar), vieritysnäkymä (ScrollView) sekä ikkuna (Window).

4.4.1 Label

Etiketti (Label) on vuorovaikutukseton kontrolli, jonka avulla on hyvä esittää teksti tietoa (kuva 24). Kontrollille asetetaan Rect-arvot, joiden avulla asetetaan kontrollin sijainti ruudulla ja koko. Rect-arvoja käytetään kaikkien Gui-kontrollejen yhteydessä.

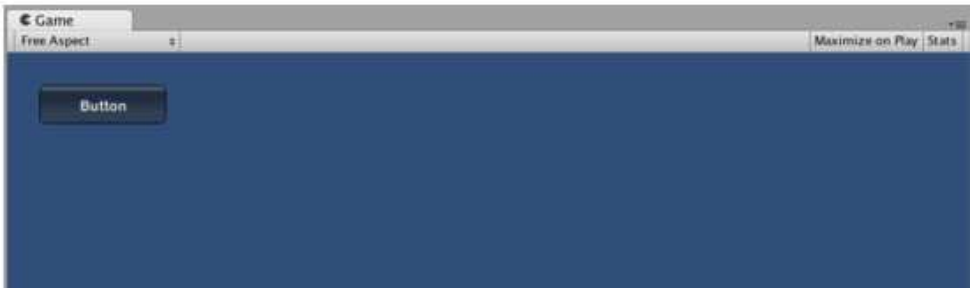
```
/* GUI.Label example */

function OnGUI () {
      GUI.Label (Rect (25, 25, 100, 30), "Label");
}
```

Kuva 24. Label (Controls 2011a)

4.4.2 Button

UnityGui palauttaa arvon tosi (True), kun painiketta painetaan. Jos halutaan suorittaa ohjelmakoodi, kun painiketta painetaan, niin koodi sijoitetaan painikkeen jos (If)-lausekkeeseen, joka toteutuu painiketta painettaessa (kuva 25).

```
/* GUI.Button example */

function OnGUI () {
      if (GUI.Button (Rect (25, 25, 100, 30), "Button")) {
            // This code is executed when the Button is clicked
      }
}
```

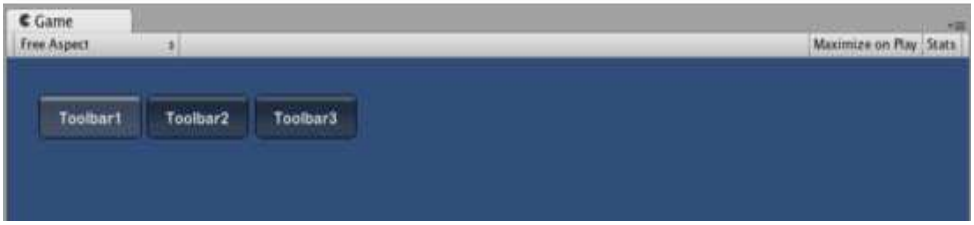


Kuva 25. Button (Controls 2011b)

4.4.3 Toolbar

Työkalurivi (Toolbar) on rivi painikkeita, joista yksi kerrallaan on aktiivinen, niin pitkään kuin toista painiketta painetaan (kuva 26). Painikkeiden määrän työkalurivillä voi määrätä numerolla. Aktiivinen painike saadaan selville kokonaisluku (Integer) arvon mukaan.

```
/* GUI.Toolbar example */

var toolbarInt = 0;
var toolbarStrings : String[] = ["Toolbar1", "Toolbar2", "Toolbar3"];

function OnGUI () {
      toolbarInt = GUI.Toolbar (Rect (25, 25, 250, 30), toolbarInt, toolbarStrings);
}
```



Kuva 26. Toolbar (Controls 2011c)

4.4.4 Selection Grid

Valintataulukko (Selection grid) on monirivinen työkalurivitaulukko (kuva 27). Vain yksi painike voi olla aktiivinen kerrallaan. Aktiivinen painike saadaan selville kokonaisluvun (Integer) avulla. Painikkeiden määrä saadaan asettamalla elementtien määrä valintataulukkoon.

```
/* GUI.SelectionGrid example */

var selectionGridInt : int = 0;
var selectionStrings : String[] = ["Grid 1", "Grid 2", "Grid 3", "Grid 4"];

function OnGUI () {
      selectionGridInt = GUI.SelectionGrid (Rect (25, 25, 100, 30), selectionGridInt, selectionStrings, 2);

}
```

Kuva 27. Selection grid (Controls 2011c)

4.4.5 Textfield

Tekstikenttä (Textfield) on vuorovaikutteinen yksirivinen kirjainten (String) esityskenttä (kuva 28). Tekstikenttä palauttaa kirjainjoukon, joka on syötetty tekstikenttään.

```
/* GUI.TextField example */

var textFieldString = "text field";

function OnGUI () {
      textFieldString = GUI.TextField (Rect (25, 25, 100, 30), textFieldString);
}
```

Kuva 28. Textfield (Controls 2011d)

4.4.6 Textarea

Tekstialue (Textarea) on vuorovaikutteinen monirivinen tietokenttä kirjain tietotyyppiselle (String) tietosisällölle (kuva 29). Tekstialue palauttaa sen sisälle syötetyn kirjainjoukon.

```
/* GUI.TextArea example */

var textAreaString = "text area";

function OnGUI () {
      textAreaString = GUI.TextArea (Rect (25, 25, 100, 30), textAreaString);
}
```

Kuva 29. Textarea (Controls 2011e)

4.4.7 Toggle

Vaihtopainike (Toggle) luo valintaruudun vaihtoehtoina päällä/pois (kuva 30). Valintapainikkeen ruudun tilaa saa vaihdettua Toggle-painiketta painamalla.

```
/* GUI.Toggle example */

var toggleBool = true;

function OnGUI () {
      toggleBool = GUI.Toggle (Rect (25, 25, 100, 30), toggleBool, "Toggle");
}
```

Kuva 30. Toggle (Controls 2011f)

4.4.8 ScrollView

Vieritysnäkymän avulla voidaan vierittää vieritysnäkymää sivuttain, ylös ja alas (kuva 31). Vieritysnäkymä tarvitsee kaksi kokoarvoa (Rects), ensimmäinen määrittelee, kuinka suuri vieritysalue on ja toinen määrittelee vierityspalkkien sijainnin.

```
/* ScrollView example */

var scrollViewVector : Vector2 = Vector2.zero;
var innerText : String = "I am inside the ScrollView";

function OnGUI () {
      // Begin the ScrollView
      scrollViewVector = GUI.BeginScrollView (Rect (25, 25, 100, 100), scrollViewVector, Rect (0, 0, 400, 400));

      // Put something inside the ScrollView
      innerText = GUI.TextArea (Rect (0, 0, 400, 400), innerText);

      // End the ScrollView
      GUI.EndScrollView();
}
```

Kuva 31. Scrollview (Controls 2011g)

4.4.9 Window

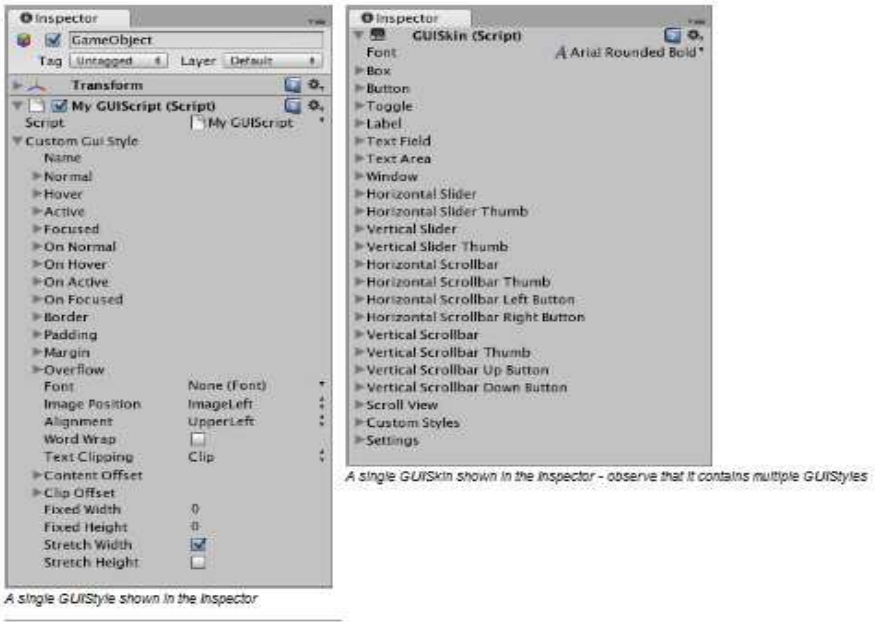
Ikkuna (Window) on liikuteltava säiliö muille kotrollielementeille. Ikkuna on ainut UnityGui-kontrolli, joka tarvitsee oman funktion Ongui-funktion hakasulkeiden ulkopuolelle (kuva 32). Ikkunalle annetaan oma Id-numero ja funktio, joiden avulla ikkuna on mahdollista suorittaa.

```
/* Window example */

var windowRect : Rect = Rect (20, 20, 120, 50);

function OnGUI () {
      windowRect = GUI.Window (0, windowRect, WindowFunction, "My Window");
}

function WindowFunction (windowID : int) {
      // Draw any Controls inside the window here
}
```

Kuva 32. Window (Controls 2011h)

4.4.10 GuiStyles

Gui-tyyleillä (GuiStyles) voi muuttaa valikoiden visuaalista ulkonäköä oman sovelluksen tai pelin tarpeisiin. Tyylejä pystyy muotoilla tyylivalikon avulla tyylien-ominaisuuksia muuttamalla (kuva 33). Jos mitään Gui-tyyliä ei ole käytössä, käyttää UnityGui-ohjelman omaa oletustyyliä. Jos käytössä on monta Gui-tyyliä, voidaan kaikki ne koota Gui-kokoelmaan (Guiskin) (kuva 33). Gui-tyylit on tehty jäljittelemään Web-sivuilla käytettävää CSS (Cascading Style Sheet) muotoilukieltä. (Customization 2011)

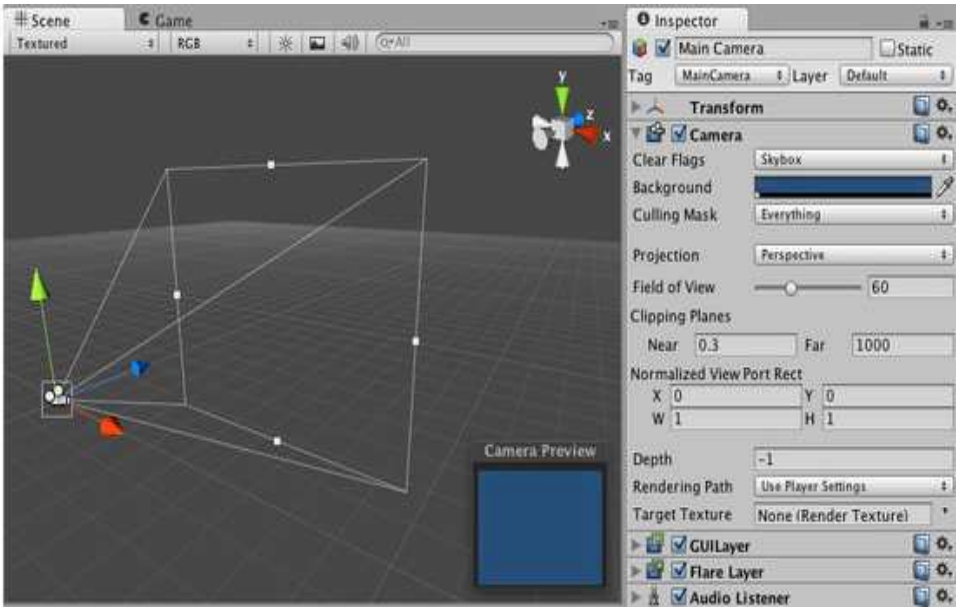


Kuva 33. *Guistyles*

4.4.11 Kamerat

Jokaisessa Scenesssä on vähintään yksi kamera, jonka avulla luodaan Unity-editorissa pelin tai sovelluksennäkynä. Scenessä voi myös samanaikaisesti käyttää useampia kameroita, jotka voi sijoittaa kuvaamaan eri kohteita jakamalla Game-näkymän pienenpiin ikkunoihin, joissa eri kameroiden kuva näkyy. Kamerat voidaan määritellä eri tasoille syvyysarvon (Depth) avulla määritellään kameroiden järjestyksen, eli sen, mikä kamera on etualalla ja mikä taka-alalla. Kameroita voi ohjata ohjelmakoodien avulla ja kameran-ominaisuuksien avulla. Kameroille voi luoda lähestulkoon minkälaista toiminnallisuutta tahansa, joka on mahdollista toteuttaa ohjelman puitteissa. Kamerat voi laittaa kuvaamaan eri kuvakulmaa, jota voi vaihtaa esimerkiksi painikkeenpainalluksella tai kamerat voi animoida kulkemaan jotain haluttua liikerataa pitkin. Erilaisia kameran-ominaisuuksia ovat ikkunan näkymän valinta (Clear Flag), kuvan taustavärin valinta (Background), objektien tasojen hallinta (Culling Mask), kameran perspektiivin valinta (Projektion), kameran näyttöalueen koko (Size), kameran näyttöalueen leveys (Field of view), kameran näyttöalueen ulottuvuus (Clipping planes), kameran ikkunan sijainti (Normalized view port rect), eri kameroiden

päällekkäisyyksien hallinta (Depth) ja kuvanpiirto metodin valinta (Rendering Path) (kuva 34). (Camera 2011)



Kuva 34. *Kamera*

4.4.12 FirstPersonControl

Käytin työssäni hahmon hallintakontrollia (FirstPersonControl), jonka avulla pystytään luomaan Sceneen liikuteltava kamera. Kameraa voi liikutella nuolinäppäimien ja hiiren avulla näyttämöllä. Kameran edustalle voi luoda myös Avatar-hahmon, jonka välityksellä voi liikkua kolmiulotteisessa tilassa. Kontrolli on Unity-editoriin luotu valmis ominaisuus, jonka voi lisätä haluamalleen peliobjektille.

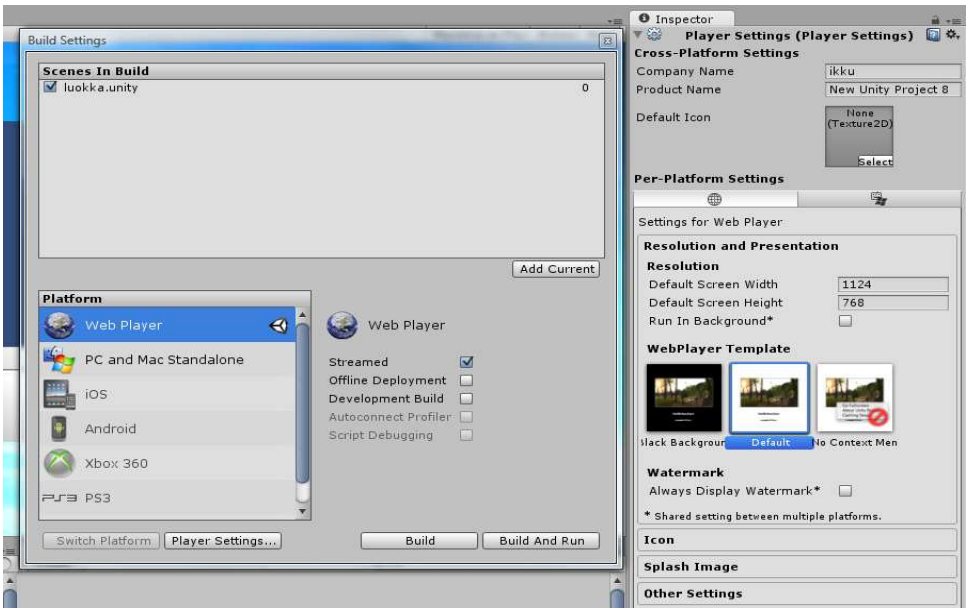
4.4.13 Äänet

Jokaisessa Scenessä on oma äänten kuuntelija (Audio Listener), joka toimii mikrofonin kaltaisesti toistaen äänen Sceneen lisätyistä äänenlähteistä. Äänilähteitä voi olla useita eri puolella kolmiulotteista näyttämöä. Äänen toistossa voi käyttää kaksiulotteista tai kolmiulotteista muotoa. Kaksiulotteinen ääni kuuluu tasaisena kolmiulotteisen näyttämön sijainnista riippumatta, kun taas kolmiulotteinen ääni kuluu vain äänenlähteen läheisyydessä kol-

miulotteisella näyttämöllä. Äänet lisätään Sceneen liittämällä ääni peliobjektin-ominaisuuksiin. (Sound 2010)

4.4.14 Julkaiseminen

Valmiin sovelluksen julkaiseminen (Publish) tapahtuu UnityEditorissa helposti. Julkaisemisasetuksiin (Build Setting) tulee valita sovelluksen tasot eli Scenet ja sovelluksen julkaisualusta. Jokaiselle julkaisualustalle löytyy lisää asetuksia sovelluksen-ominaisuuksien säätelyyn. Julkaistavan sovelluksen-ominaisuuksien säätämiseen löytyy oma toistimen asetukset-valikko (Player setting), josta voi säätää sovellusikkunan resoluutiota, asettaa latausikonin kuva, asettaa meta-tietoja ja säätää WebPlayer-sivun tyyliä. Build-napin avulla Unity tekee sovelluksesta valmiin julkaisualustallaan suoritettavan itsenäisen sovelluksen. Unity free-lisensillä on mahdollista julkaista vain Web-player-, Windows- ja Mac/Os X-alustoille. Kuvassa Unity3D-editorin julkaisunäkymä (kuva 35).



Kuva 35. *Publish*

4.5 WebPlayer

Webplayerille julkaistaessa tekee Unity-sovelluksesta HTML-sivun ja Unity WebPlayer-tiedoston, johon on sijoitettu tieto toistettavasta sovelluksesta. Unity-soitin on sijoitettu WWW-sivulle CSS (Cascading Style Sheet) muotoilukielen ja HTML-kielen avulla. Unity-soittimen voi myös liittää omalle Web-sivulle. Soitinikkunan kokoa voidaan myös muuttaa kattamaan koko tietokoneennäytön ruudun tai annetun pikselimäärän verran. Unityn soitinikkunassa näkyvä sovellus ja Web-sivu voivat kommunikoida keskenään Javascriptin avulla. Unityssä löytyy kommunikointia varten omia funktioita.

(Cascading style sheet) on WWW-dokumenteille kehitetty tyyliohjeiden laji, joka yhdistyy yhdeksi säännöstöksi. CSS:n rinnalla voidaan käyttää muitakin muotoilukieliä. CSS:ssä yleinen esitystapa on laatikkomalli (Boxmodel), jossa jokainen muotoilukielen elementti mielletään laatikoksi, joka sijoitetaan ympäröivän elementin sisään muiden saman tason elementtien rinnalle. Lähestulkoon kaikki nykyiset selaimissa löytyy CSS-tuki.
(CSS 2011)

HTML (Hypertext markup language) avoimesti standardoitu kuvaus kieli, jonka avulla voidaan kuvata hypertekstiä. HTML-kieltä käytetään pääsääntöisesti Web-sivujen kuvaamiseen. Kieli toimii oman syntaksin mukaisten elementtien avulla, joiden avulla voidaan esitettävä sisältö sijoittaa ja asetella HTML-sivulle. HTML-kielen alkuperäinen tarkoitus oli kuvata WWW-sivun rakennetta ja ulkoasua. Uusin versio kielestä on HTML5. (HTML 2011)

Käytin opinnäytetyöni käytännön osassa videoiden ja laaja-alaisten tekstien esittämiseen Unityn3D-selaimen päälle avautuvaa Iframe-selainikkunaa. Ifame on HTML-kielen elementti, jonka sisään voi avata uuden WWW-sivun, jolle voi asettaa leveyden(Width), korkeuden(Height) ja avattavan sivun osoitteen (Src). Iframe-ikkuna avautuu HTML-sivulla siinä kohdassa HTML-koodia, johon elementti on asetettu.

4.6 UnityObject

UnityObjektin avulla saa Unityn 3D-selaimen sijoitettua HTML-sivulle vähäisellä ohjelmointikoodi määrällä. Alla olevassa kuvassa head-osassa UnityObjectjavascript-tiedosto lataa 3D-selaimen div-osaan UnityPlayer. div on CSS-muotoilukielessä käytettävä tilanvaraaja. Kuvassa body-osassa UnityPlayerin div on sijoitettu HTML-sivulle (Kuva 36). Body-osan sisään HTML-kielessä tulee HTML-sivulle tulevat sivulla näkyvät elementit. Kuvassa näkyy myös kohta, jonka avulla voi muuttaa Unityn 3D-selaimen ikkunan kokoa (kuva 36).

Kuva 36. UnityObject

4.6.1 Unity-soittimen ja selaimen kommunikaatio

HTML-sivun ja Unityn WebPlayer-soittimen välisen kommunikaation avulla pystytään luomaa vuorovaikutusta Web-sivun ja Unity-sovelluksen välille. HTML-sivulle voidaan esimerkiksi luoda valikko, jonka avulla voi kontrolloida WebPlayerin sisältöä. Kommunikaatio voi tapahtua kahteen suuntaan, joko WebPlayer kutsuu HTML-sivun funktiota tai HTML-sivu kutsuu WebPlayer-funktiota. Kuvassa näkyvät funktiot, joilla voi kutsua joko selaimen Javascript-funktioita tai Unityn Javascript-funktioita (kuva38).

Kuva 38. *Communication*

5 VIRTUAALILUOKKA

Opinnäytetyökokonaisuuteni tavoitteena oli suunnitella ja luoda mainoksenomainen sovellus verkkosivuille, jonka avulla esittelen toimeksiantajani Kauniaisissa toimivan Kasavuoren yläkoulun historian luokan pedagogista toimintatapaa ja oppimisympäristöä Unity3D-ohjelmalla luodun virtuaalisen kolmiulotteisen luokkatilan, videoiden, äänen ja tietoikkunoiden kautta. Työn tavoitteena on löytää uusia markkinointi keinoja, joiden avulla työn kohteena oleva koulu voi esitellä toimintaansa. Esittelyssä käytettävien palasten avulla pyritään luomaa eheänä kokonaisuutena toimiva helppokäyttöinen sovellus. Tekemäni sovelluksen muodostama esittelyn kohderyhmänä toimivat uusien koululaisten vanhemmat ja uudet koululaiset.

Opinnäytetyöni käytännön osuuden sisältö muodostui HTML-sivusta, joka pitää sisällään 3D-näyttämön, 2D-valikon, video-ikkunoita ja tekstiikkunoita. 3D-näyttämön avulla esitettiin virtuaalinen 3D-malli esiteltävän luokan tilasta, oheislaitteista ja huonekaluista. 2D-valikkojen avulla hallinnoidaan sovelluksen sisältöä. Valikot näkyvät 3D-sisällön päällä. Videoikkunat sisältävät havainnoivaa video-materiaalia esiteltävistä kohteista. Tietoikkunat sisältävät tekstitietoa esiteltävistä kohteista. Työn eri elementit muodostavat yhdessä eheänä kokonaisuutena toimivan sovelluksen.

5.1 Suunnittelu

Työtä suunniteltaessa oli otettava huomioon omat resurssit, kuten ajankäyttö, osaaminen, tiedontarpeet, saatavilla olevat ohjelmat. Alun perin työn oli tarkoitus olla laajempi kokonaisuus, mutta työn vaativuus ja tarvittavat työtunnit rajoittivat työn rajauksen pienemmäksi kuin alkuperäinen suunnitelma. Työn suunnittelu ja ideointi lähti keskustellen vuorovaikutuksessa yhteistyökumppanin kanssa hahmotellen työn ominaisuuksia, tavoitteita ja tarkoitusta. Työn teko vaati myös oikeanlaisten työvälineitä, jotka mahdollistavat kyseisen suunnitelman toteuttamisen. Huomioon otettavia tekijöitä työvälineiden valinnassa oli oma osaaminen, ohjelmien-ominaisuudet ja eri ohjelmistojen lisenssit. Itse virtuaalitilaa suunnitellessa täytyi miettiä, kuin-

ka yksityiskohtaisesti luokkaympäristö mallinnettaisiin, koska sovelluksen on tarkoitus olla tarpeeksi kevyt toimiakseen Web-sivulla. Liian yksityiskohtaiset kohteet oli jätettävä pois. Luokan yksityiskohtien esittämiseen oli löydettävä oikea rajaus, kuitenkin tarvittavat yksityiskohdat huomioiden.

5.1.1 Mallit

3D-mallinuksesta vei paljon aikaa ja sen takia mallinnettava ympäristö piti suunnitella tarkoin. Mallinnus tapahtui digivalokuvista silmämääräisen mittakaavan mukaan. Työn mallinnustarkkuuden tarkoitus ei ollut saavuttaa täysin autenttista mallia oikeasta malliluokasta. Mallien tekoon piti löytää mahdollisemman yksinkertainen toteutustapa ja sen takia mallinnukseen varattavaan aikaan tuli huomioida tiedonhaku ja uusien menetelmien omaksuminen. Esityksen sisältö määritteli pitkälle sen, mitä asioita piti korostaa ja mallintaa tarkasti.

5.1.2 Toiminnallisuus

Työhön täytyi miettiä ja suunnitella keinoja, joilla halutut asiat voisi esittää. Sain oppilaitokselta aineistoa ja rungon luokan toiminnan esittelyyn, jonka ympärille oli tavoitteena suunnitella ja rakentaa esitys 3D-mallia, ääntä, tekstiä ja videota hyödyntäen. Oppilaitos tuottaa itse tietosisällön, esitykseeni eli videot tekstit. Esityksen sisältö määrittelee melko pitkälle myös sen, minkälaista toiminnallisuutta ja keinoja haluttujen asioiden esittelyyn tarvitaan. Erilaisia toiminnallisuuden keinoja olivat esimerkiksi kameran kuvakulmanvaihtelut, animaatiot, tietoikkunan avaaminen.

5.1.3 Käyttöliittymä

Työn käyttöliittymän ja valikkojen avulla voi helposti pilata hyvän sovelluksen, jos käytettävyys on heikko epäselvien ja vaikeakäyttöisten valikkojen takia. Tavoitteena oli luoda helppokäyttöinen käyttöliittymä mahdolli-

simman vähillä toiminnoilla. Käyttöliittymän suunniteluun vaikutti pitkälti esityksen sisältö, eli se, mitä toimintoja esitys piti sisällään. Valikon tekotapaan oli useita vaihtoehtoja. Erilaisia valikon tekotapoja oli kolmiulotteinen valikko, kuvista tehtävä valikko tai ohjelmoimalla tehtävä valikko. Työhöni käytin UnityGui-käyttöliittymäohjelmointia. Käyttöliittymän suunnittelua varten tein Photoshop-kuvankäsittelyohjelmalla luonnoksia käyttöliittymän ulkonäöstä. Photoshopin avulla suunnittelin haluamani painike- ja ikkuna-tyylit. Käyttöliittymän esitystapana päädyin käyttämään ikoni-kuvakkeita niiden selkeän esitys tavan vuoksi.

5.1.4 Ulkoasu

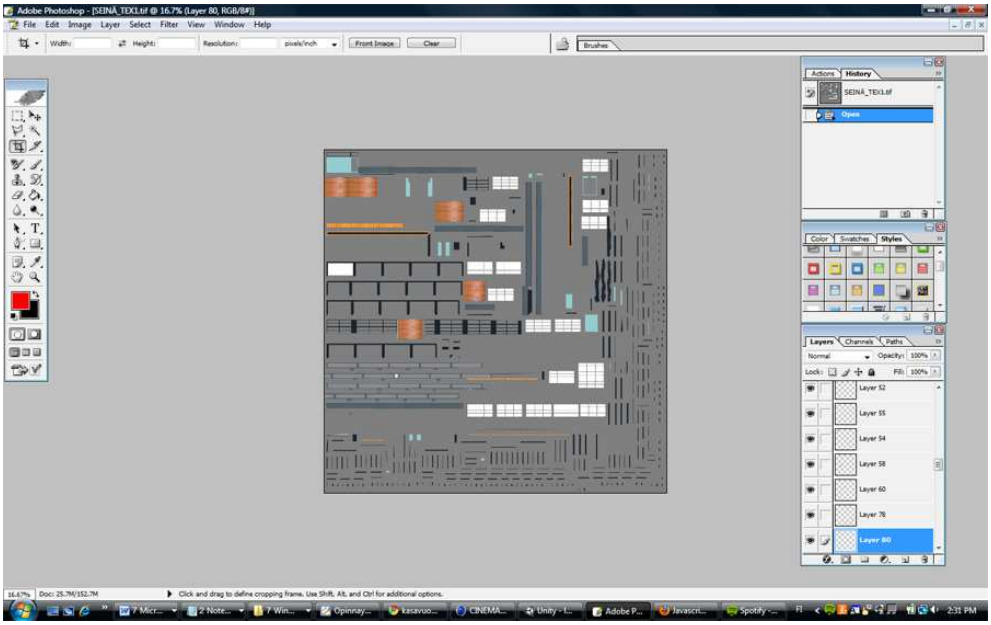
Työni ulkoasua miettiessä tuli huomioida kohde koulun kotisivujen visuaalinen ilme, koska valmistuessa tekemäni sovellus liitetään koulun verkkosivuille. Visuaalisenilmeen luonnissa ja värivalinnoissa tuli huomioida selkeä ulkoasun aikaansaaminen. Silmiinpistäviä väriyhdistelmiä pyrin välttämään. Myös työn asettelussa pyrin löytämään selkeän yhdistelmän, jotta asia kokonaisuudet pystyi erottamaan toisistaan. Kirjasinasu valinnoissa pyrin hakemaan yhtenäistä linjaa koulun kotisivun fontti valintojen kanssa.

5.2 Toteutus

5.2.1 Valokuvaus

Työtä varten täytyi ottaa digikuvia luokasta, johon virtuaaliluokka pohjautuu. Kuvia piti miettiä ja suunnitella etukäteen, koska kuvien perusteella mallinnettiin virtuaaliympäristön kohteet ja kuvista otettiin myös pintamateriaalit mallinnettuihin 3D-malleihin. Kuvien kokoa piti myös suunnitella ja muokata kuvankäsittelyohjelmalla huomioon ottaen tiedostokoko ja kuvanlaatu.

Materiaalin luontia varten otettavat kuvat piti kuvata tarkasti, miettien miten kuvan saisi aseteltua malliin selkeästi. Materiaaleiksi tarkoitetut kuvat käsittelin Photoshop-kuvankäsittelyohjelmalla. Kuvien kokoa ja muotoa piti monien materiaalien kohdalla muuttaa. UV-karttoja tehdessä piti materiaaleiksi tarkoitetuista digikuvista leikata UV-karttaan sopiva pala ja asetella pala UV-kartan ääriviivojen sisälle oikeaan kohtaan. Yhteen karttaan käytin palasia useammasta digikuvasta. Kuvassa Photoshop-ohjelmaan viety UV-kartta (kuva 39).



Kuva 39. *UV-kartta*

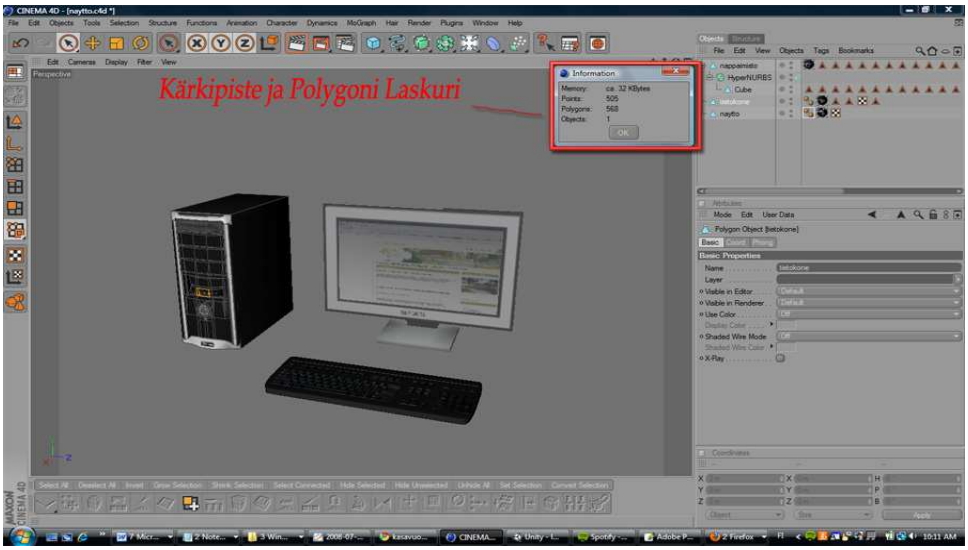
5.2.2 Mallit

Mallinnusta valmistellessani piti tutustua mallinnettavaan ympäristöön ja valokuvattava valitut kohteet. Mallinnukseen käytin Cinema4d-ohjelmaa. Mallinnuksessa yritin saavuttaa mallinnettavat muodot mahdollisimman vähiä kärkipisteitä käyttäen noudattaen Low-poly-mallinnuksen ideaa. Mallinnukseen käytin Cinema4d:n Polygon-työkaluja, joiden avulla malli voidaan rakentaa joko aloittamalla perus 3D-ojekteja muokkaamalla tai kärkipisteitä yksi kerrallaan lisäämällä halutun muodon saavuttamiseksi.

Ensimmäisenä mallinsin luokkahuoneen seinät ja lattian valitsemieni yksityiskohtien kanssa. Luokkahuoneen seinien mallintamisen aloitin perus-

objektista laatikko (Box), josta poistin polygonit päältä ja alta, jolloin laatikon sisus muodosti seinät. Luokkahuone mallin seiniin piti leikata ikkunan ja liitutaulujen kohdat. Leikkaaminen tapahtui Puukko (Knife) polygonityökalulla. Ikkunat muodostin pursuttamalla leikattua kohtaa sisäänpäin, jonka jälkeen erotin ikkuna ruudun omaksi 3D-objektiksi. Liitutaulujen muodon tein myös pursuttamalla leikattua kohtaa ulospäin ja muodostaen halutut kulmat pintaan.

Luokan sisustuksena olevat esineet kuten pulpetit, tuolit, lamput, tietokoneet ja kartat mallinsin yksitellen valokuvista silmämääräisiä mittasuhteita käyttäen. Malleja, joita oli luokassa paljon, tarvitsi mallintaa vain kerran ja sitten monistaa mallista tarvittava määrä objekteja. Yksityiskohtaisiin kohteisiin täytyi käyttää enemmän kärkipisteitä tarvittavien muotojen aikaansaamiseksi. Malleja tehdessäni tarkkailin kuitenkin jatkuvasti käytettyjen pisteiden määrän laskuria (kuva 40). Mallintamistapana käytin laatikkomalli ja reunamallinnus yhdistelmää, jossa leikkaamalla pintoja tai reunoja pienemmäksi ja sitten pursuttamalla saadaan aikaiseksi haluttu muoto. Joitain pyöreitä muotoja saavuttaakseni, minun oli käytettävä Nurbs-ominaisuuksia. Mallin ollessa valmis, kävin mallin läpi yhdistämistyökalulla (Weld) poistaen muodon kannalta merkityksettömät ja ylimääräiset pisteet. Lopuksi täytyi myös tarkistaa, että mallien pintojen sivut (Normals) ovat oikein päin, koska mallit eivät näy oikein Untity3D-editorissa, jos pinnat ovat väärinpäin.

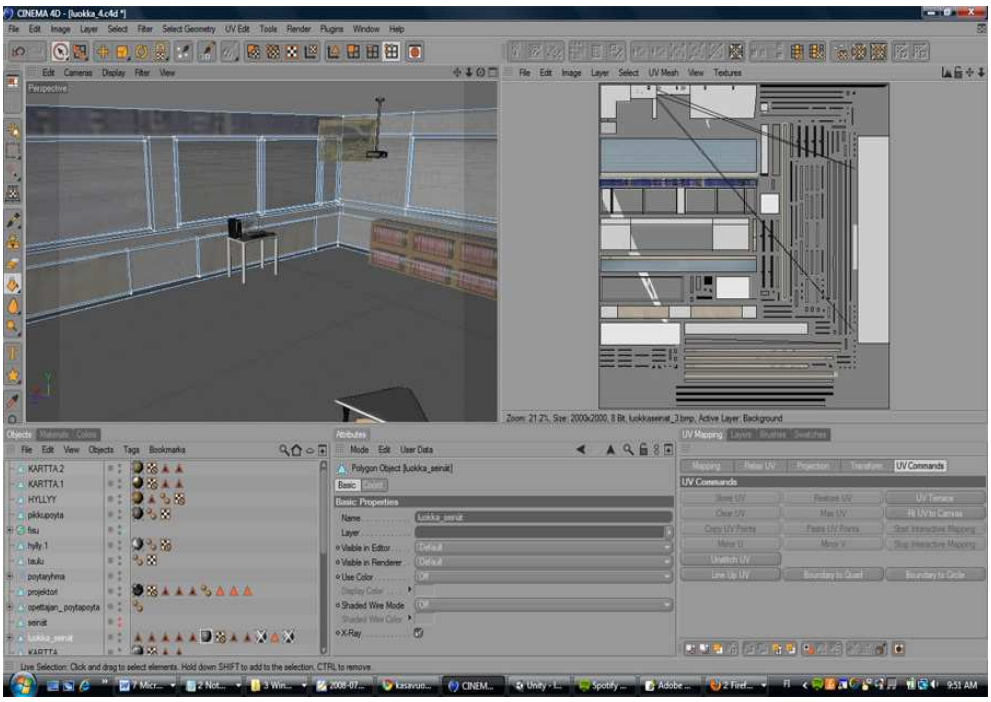


Kuva 40. Polygon-laskuri

5.2.3 Materiaalit

Jokaiselle 3D-mallille oli luotava oma materiaali, jonka avulla kappaleen pinnan-ominaisuudet voitiin esittää. Pintakuviot materiaaleihin suurimmaksi osaksi otin digivalokuvista. Käytin jossain kohteissa yksittäistä väriä, jolle määrittelin pinnan muita ominaisuuksia. Materiaalien luonnissa käytin apuna Cinema4d:n UV-mapping-ominaisuuksia, joiden avulla kappaleen pinnoista saatiin aikaiseksi materiaaleja varten kartta, jonka polygonipalasiin voitiin sijoittaa oikeanlaiset pintamateriaalit. Tein myös valaistuskartan luokkahuone mallin seinien ja lattian materiaaleille.

Kolmiulotteisen mallien UV-karttoja tehdessäni käytin apuna Photoshop-kuvankäsittelyohjelmaa, jossa leikkasin ja asetin Cinema4d:ssä luotuun mallin UV-kartta kuvan ääriviivoihin oikeanlaiset pintamateriaalit. Käytin Photoshop-ohjelmaa Cinema4d-editorin sijaan, koska Photoshopin käyttöliittymä on minulle tutumpi käyttää. Tein kaikkiin malleihini UV-kartan. Kartan teko tapahtui samalla tavalla kaikille 3D-malleille. Ensiksi täytyi luoda UV-kartta Cinema4d:hen UV-editorin Wizard-toiminnon avulla, mikä luo automaattisesti UV-kartan mallista. Mallin karttaa voi joutua muokkaamaan manuaalisesti UV-wrap toiminnoilla, koska kartta ei välttämättä näy oikein. Jouduinkin muutaman 3D-mallin UV-karttaa muuttamaan hieman. Kartan luonnin jälkeen loin näkyvät ääriviivat kartan polygonien reunoihin, jotta näkisin kartan Photoshop-ohjelmassa, missä sijoitin pintamateriaalit mallin UV-kartalle. UV-kartan ollessa valmis Photoshopissa toin UV-kartan kuvan takaisin Cinema4d-ohjelmaa ja sijoitin kartan oikealle 3D-mallille, jonka jälkeen materiaalin pintakuviointi näkyi mallin pinnalla (kuva 41).



Kuva 41. *Seinien UV-kartta*

Seinille ja lattialle luodun valaistuskartan tein Cinema4d:ssä Render-valikon alla sijaitsevan automaattisen toiminnon (Bake Object) avulla, mikä tallentaa valaistuskartta materiaaleihin kolmiulotteiselle näyttämöille asetettujen valonlähteiden luoman valon. Ennen valaistuskartan tekoa piti luoda näyttämölle valaistus. Valaistuksen teossa yritin huomioida luokan valonlähteitä, kuten ikkunoista tuleva auringonvalo ja lampuista syntyvä valo.

5.2.4 Vienti/Export

Cinema4d:n kolmiulotteisesta näyttämöstä luodun tiedoston voi tallentaa usealle eri formaatille. Itse tarvitsin omasta Scenestäni formaatin, joka näkyy oikein Unity3D-ohjelmassa. Cinema4d-ohjelman ja Unity3D-editorin välillä on ollut ongelmia; kolmiulotteiset mallit eivät ole kääntyneet oikein ohjelmasta toiseen. Kokeilin eri tiedostomuotoja ja päädyin käyttämään Autodeskin .3ds-tiedostomuotoa viedessäni malleja ohjelmasta toiseen. .3ds vaikutti kokeilujen perusteella vakaimmalta tiedostomuodolta. Vienti toiminto löytyi file-valikon alta, josta export-valinnan jälkeen valitaan tiedostomuoto, jolle Scene käännetään.

5.2.5 Unity-editori

Uuden projektin luodessa Unity-editori luo oman projektikansion projektille. Projektikansioon sijoitetaan kaikki sovelluksessa tarvittavat palaset. Sovelluksen teko aloitettiin Unityssä projektin luomisella ja peliobjektien tuonnilla ohjelmaan. Ohjelmassa on oma tuonti (Import asset), jonka avulla voi tuoda kuvia, 3D-malleja, ääntä, tekstiä ja ohjelmoitikoodeja (Script) ohjelmaan.

Ensimmäisenä toin Cinema4d:llä luomani kolmiulotteiset mallit ohjelmaan ja asettelin ne pelinäkymään (Scene). Seuraavaksi toin ohjelmaan 3D-malleille tekemäni UV-kartta materiaalit. Osa materiaaleista tuli mukana 3D-mallien tuonti operaatiossa. Puuttuvat materiaalit täytyi tuoda yksitellen ohjelmaan ja asettaa oikealle paikoille 3D-mallille objektin-ominaisuuksien hallintanäkymästä (Inspector).

Seuraavaksi pääsin lavastamaan kolmiulotteista näyttämöä Unityn tuomillani 3D-malleilla. Olin tehnyt malleja useampaan eri tiedostoon ja Unityssä mallien koko piti skaalata toisiinsa sopiviksi. Mallien kokoa muutin näyttämönhallintatyökalun avulla (Transform tool). Mallejen lisääminen näyttämölle tapahtui lisäämällä Unityn Project-näkymän peliobjektit Hierarchy-näkymään. Näyttämölle lisäämisen jälkeen, piti peliobjekteista tehdä oikeankokoisia ja peliobjektit piti liikutella oikeisiin kohtiin näyttämöllä. Peliobjektien muokkaus tapahtui Transform-työkaluilla.

5.2.6 Näkymät

Seuraavassa vaiheessa asettelin kolme kamera kerrosta näkymään (Scene), jotka ovat käyttöliittymää, kolmiulotteinen malleja ja taustakuvaa varten (kuva 42). Päällimmäisin kamera kerrosnäkymä on käyttöliittymän kontrollielementtejä varten. Kerros näkyy muiden kameroiden päällä, koska sille on määritelty suurin syvyysarvo (Depth). Keskimmäinen kameranäkymä täyttää vain osan ruudusta ja sen sisällä esitetään kolmiulotteista mallia kuvaavat kamerat. Muut keskimmäisen kerroksen näkymät esittävät luokkaa

eri kuvakulmista. Kolmas kamera kerros kuvaa tasoa (Plane), jonka materiaaliksi näyttämöllä näkyvä taustakuva on asetettu. Taustakuva jää muiden kerrosten taakse.

Kuva 42. *Näkymät*

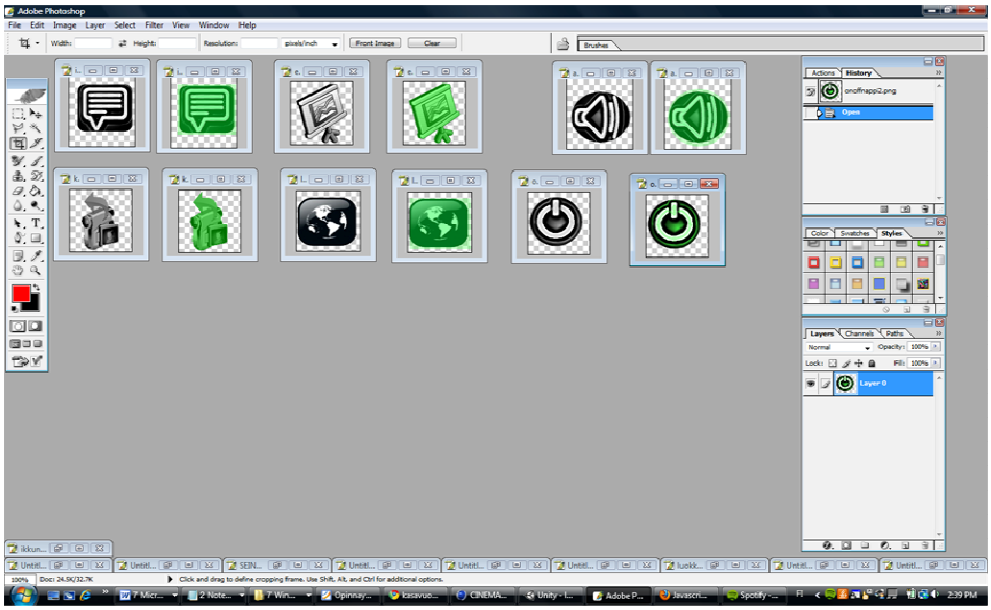
Keskimmäisen kerroksen kolmiulotteisessa näkymään on aseteltu useampi kamera, joista yksi vuorollaan on käytössä. Kamerat menevät päälle ja pois käyttöliittymästä tehtyjen valintojen perusteella. Yhdessä valinnassa voi liikkua virtuaaliluokassa nuolinäppäimillä ja hiirellä liikuteltavan kameran (FirstPersonControl) avulla. Muutin kontrollin Javascript-koodia niin, että hiirellä liikuttelu tapahtuu vain nappia painamalla ja hiirtä liikuttamalla, joka tuntui selkeämmältä tavalta käyttää sovellusta vapaassa liikkumisnäkymässä. Muissa valinnoissa kamera on asetettu kuvaamaan tietystä kuvakulmasta asetettua kohdetta.

5.2.7 Käyttöliittymän luonti

Käyttöliittymän luonnissa täytyi hahmottaa sovelluksessa tarvittavat toiminnot, joita tarvittiin valittujen kohteiden esittämiseen. Toiminnoilla tarkoitan sitä, milloin jokin tietoikkuna aukeaa, milloin video lähtee pyöri-

mään tai ääni soitetaan. Rakenteen, asettelun ja kontrollielementtien tyylien ollessa valmiita pääsin ohjelmoimaan itse käyttöliittymää.

Ensimmäisessä vaiheessa käyttöliittymää tehdessäni tein Photoshopilla valikossa tarvittavia ikonikuvakkeita ja ikkunatyylejä. Ikoneista tein kahdenlaisia eri versioita, joista ensimmäisessä ikoni on normaali tilassa ja toisessa ikoni on valittuna. Ikonin pohjakuvia otin verkosta ilmaisista Ikonikokoelmista, joita muokkasin kuvankäsittelyohjelmalla omanlaisekseni. Valittu ikoni eroaa normaalista korostevärin välityksellä. Ikkunatyylin loin Photoshopin valmiin tyyli (Styles) ja läpinäkyvyys (Opacity) ominaisuuksien avulla. Kuvassa 43 on sovellukseen luotuja ikoni-painikkeita (kuva 43).



kuva 43. *Ikonit*

Seuraavaksi vaiheessa pääsin ohjelmoimaa käyttöliittymän toiminnallisuutta, jonka toteutin UnityGui-ohjelmoinnilla OnGui-funktion sisälle. Ensimmäisenä ohjelmoin käyttöliittymään ikkunat, joiden sisälle muut kontrollielementit tulevat. Päävalikkoikkunan sisälle sijoitin Ikonipainikkeet, joiden avulla aukeaa alavalikkoikkunaan lisää toimintoja. Alavalikon ikonipainikkeista aukeaa tekstiä, tietoikkunoita ja videoita. Valikkojen asettelun ollessa oikeilla kohdillaan, muutin valikkojen tyyliasetuksia Guistyle-valinnoilla. Guistyle-tyyliasetuksista sijoitin ikonikuvat painikkeille, muu-

tin ikkunapohjat Photoshopissa luomiini omiin ikkuna tyyleihin. Muutin myös kirjasintyyppiä ja kokoa oman tyylini mukaiseksi.

Valikkojen visuaalisenilmeen ja asettelun ollessa valmiin ryhdyin luomaan valikon painikkeiden if-lauseidein tapahtumia, eli tapahtumia, joita tapahtuu nappia painettaessa. Päävalikon ikonipainikkeet avaavat oman ikkunan, joka ilmestyy ruudun alareunaa ja pitää sisällään valittavaan aiheeseen liittyviä ikoni-painikkeita. Seuraavaksi loin painikkeiden tapahtumille sisällön. Osa painikkeista liikuttelee kameraa ja avaa tietoikkunoita. Osa taas kutsuu selaimen Javascript-funktioita, jotka avaavat tietoikkunoita, jotka taas pitävät sisällään video ja teksti sisältöä. Sovelluksessani käyttämäni käyttöliittymän Javascript-ohjelmakoodi löytyy opinnäytetyöni liitteistä kohdasta UnityGui-ohjelmakoodi sivulla 70.

Päävalikko pitää sisällään ohje-painikkeen, vapaatutkiminen näkymäpainikkeen ja ohjatunkierroksen-painikkeen. Valikkoon on suunnitteilla lisää ominaisuuksia. Ohje-painikkeen takaa löytyy tietoikkuna, opastetaan sovelluksen käyttöön. Vapaatutkiminen-painikkeen takaa löytyy näkymä, jossa voi vapaasti tutkia malliluokkaa. Näkymässä liikutaan hiiren ja nuolinäppäinten avulla luokassa ja luokan sisällä on info-kuvakkeita, jotka avaavat tietoikkunoita ja videoita valituista kohteista. Ohjatussa kierroksessa tutustutaan toimintakertomuksen kautta luokan oheislaiteisiin ja pedagogiseen toimintaympäristöön.

5.2.8 Webplayer

Tein sovelluksestani verkkosivuilla toimivan julkaisun Unity-editorin julkaisu (Publish)-toiminnon avulla valitsemalla alustaksi WebPlayerin. Seuraavaksi muokkasin Unityn 3D-selaimen kattamaan koko ruudunnäytön. Käytin kokoa muokatessa prosentuaalista kokoa ruutuun nähden, jolloin resoluutiosta riippumatta Unityn 3D-selain kattaa koko näytön leveyden.

Seuraavaksi liitin sovelluksen HTML-sivulle erilliset Javascript-tiedostot, joita 3D-selain kutsuu funktioiden välityksellä ja Javascript-tiedostot, jotka

kutsuvat sivun toimintoja tai 3D-selainta. Unityssä painiketta painettaessa kutsutaan HTML-sivulle liitettyjä Javascript-funktioita, jotka avaavat uuden oman näköiseksi muokatun Iframe-ikkunan. Unityssä luotujen painikkeiden if-lauseisiin on ohjelmoitu Unityn 3D-selaimen pysäytys-toiminto uuden ikkunan auetessa, kun avattu ikkuna suljetaan, ikkunaa luotu sulje-painike kutsuu Unityn 3D-selainta ja käynnistää sen uudestaan sulkien iframe-ikkunan.

Videon näyttäminen sovelluksessa muodostui aluksi ongelmalliseksi, koska en keksinyt tapaa, jolla saisin Unity3D-selaimen päälle näkymään HTML-kielellä toteutettua ikkunaa. Ohjelman omaa käyttöliittymäohjelmointia ei voinut käyttää, koska käytössäni oli Unity3D Free-versio, jossa ei videon näyttämiselle ole mahdollisuutta.

Videota, laajempia tekstiosuuksia ja kuvia sisältävät ikkunat toteutin selaimessa aukeavan uuden Iframe-ikkunan avulla. Uusien ikkunoiden muotoiluun käytin HTML- ja CSS-muotoilukieliä. Iframe-ikkuna avautuu HTML-sivulle luomaani div-osioon, jonka Unityssä olevien painikkeiden avulla aktivoin tai suljen. Painikkeet kutsuvat Unityn omalla funktiolla Application.OpenUrl() selaimeen Javascriptillä luomiani funktioita WindowOpen() ja Suljeikkuna(). Kutsu pitää sisällään kaksi muuttujaa vaihe ja layer. Vaihe-muuttuja lähettää valitun osoitteen tunnisteen, joka tulee Iframe-ikkunan sisälle. Layer-muuttuja aktivoi valitun div osan HTML-sivulla. Käytössäni oli kaksi div osaa layer ja layer2. Layer on kuva ja teksti tiedon esittämistä varten. Layer2 on videoiden esittämistä varten. Luomani funktiot näkyvät liitteissä sivulla 82.

6 LOPPUTULOS

Lähdin tekemään opinnäytetyötäni mielessäni pelkkä idea, jonka toteuttamiseen minun täytyi löytää työkalut, jotka mahdollistivat idean toteuttamisen. Osa valitsemistani työn tekemiseen tarvittavista työkaluista olikin minulle jo ennestään tuttuja. Unity3D-pelinketystyökalu, jolla suurimman osan työstäni tein oli aivan uusi ohjelma minulle ja jouduin opettelemaan sen käyttöä alkeista lähtien. Olin aikaisemmin tehnyt hieman samankaltaisia projekteja eri alustoilla. Verkkosivut työn toimintaympäristönä loi uusia haasteita ja ongelmia, koska verkkosivuilla toimivien sovelluksien toiminnassa täytyy huomioida tiedostokoko, etteivät latausajat kasvaisi liian suureksi. Käytin opinnäytetyöni raporttiosassa paljon kuvia, koska mielestäni kuvilla pystyy havainnollistamaan esittämiäni asioita selkeämmin.

Opinnäytetyöraportin tavoite toteutui mielestäni hyvin, koska raportin tarkoitus oli toimia käytännöntyön menetelmien kuvauksena. Raportin alussa halusin myös hieman tutkia muita tekniikoita, joita on käytössä verkkosivuilla esitettävän kolmiulotteisuuden esittämiseksi. Opinnäytetyöni käytännön työosuus ei aivan päässyt sille asetettuihin tavoitteisiinsa, koska työn tekemiseen tarvittiin oletettua enemmän työtunteja. Työtä tehdessäni jouduin usein ongelmatilanteeseen, mihin minun täytyi löytää ratkaisu päästäkseni eteenpäin. Tiedonhakuun ja uusienmenetelmien omaksumiseen kului myös yllättävän paljon aikaa. Suurimpina ongelmina työtä tehdessäni muodostui löytää oikeanlaiset tapa esittää video ja teksti tietoa sovelluksessa. Kokeilin monia eri vaihtoehtoja ennen kuin löysin toimivan ratkaisun. Sain tehtyä tekemäni sovelluksen demo-vaiheeseen (kuva 44). Demon avulla sain havainnoitua sovelluksen lopullista ideaa. En saanut mielestäni hyödynnettyä kolmiulotteisuuden tuomia mahdollisuuksia tarpeeksi hyvin sovelluksen demossa.

Kuva 44. *näkymä ylhäältä*

Onnistuin mielestäni luomaan sovellukseen selkeän ja helppokäyttöisen käyttöliittymän, jonka pohjalle on helppo kehittää ja luoda uusia ominaisuuksia. Myös sovelluksen visuaalinen ulkoasu ja mallintamalla luomat kolmiulotteiset mallit onnistuivat mielestäni hyvin. Vaikka sovelluksen luomiseen tarvitsi paljon eri ohjelmia ja menetelmiä, niin mielestäni kokonaisuus pysyi hyvin kasassa. Sain mielestäni luotua sovelluksen demoversioon hyvän pohjan pedagogisen toimintaympäristön kuvaamiseen. Sain myös itse paljon kokemusta ja taitoja opinnäytetyö prosessin kautta.

Tarkoituksenani on jatkaa työn ominaisuuksien kehittämistä ja saattaa loppuun aloittamani projekti. Työssä on vielä monta osa-aluetta, joita tarvitsee suunnitella ja kehittää. Demo-versiossa en ollut vielä hyödyntänyt kolmiulotteisen virtuaaliluokan animointi-mahdollisuuksia. Myös muiden alustojen ja tekotapojen yhdistäminen sovellukseen luo uusia mahdollisuuksia. Tarkoituksenani on myös luoda HTML-sivut käyttäen PHP-ohjelmointikieltä, johon tiedon tallentamiseen käytän joko XML-kieltä tai tietokantoja. Myöhemmässä vaiheessa tarkoituksenani on luoda koko koulun kattava esittely koulun kotisivuille. Esittelyyn tulee mukaan jokaisesta oppiaineesta malliluokka, jonka kautta esitellään luokantoimintaa. Tarkoituksenani on myös luoda malli koko koulu rakennuksesta, jonka sisällä voi vapaasti liikkua.

7 LÄHTEET

Opinnäytetyöt:

Kallonen, Matti. 2010. Papervision3D:n käyttö verkkosivuilla. Mediatekniikka. Hämeen ammattikorkeakoulu. Opinnäytetyö

Sähköiset lähteet:

Aalto.T. 2007 [www-sivu].[Luettu 01.02.2010]. Saatavissa: http://tuhatnaa.net/luettele_10_virtuaalimaailmaa_maarittele_virtuaalimaa ilma.

Adobe Shockwave. 2010.[www-sivu]. [Luettu 01.02.2011]. Saatavissa:http://en.wikipedia.org/wiki/Adobe_Shockwave.

Alternative. 2010. [www-sivu].[Luettu 01.02.2011]. Saatavissa: http://alternativaplatform.com/en/about.html.

Blender. 2010. [www -sivu].[Luettu25.02.2011]. Saatavissa: http://fi.wikipedia.org/wiki/Blender_%28ohjelma%29.

Camera. 2010 [www-sivu].[Luettu25.02.2011]. Saatavissa: http://unity3d.com/support/documentation/Manual/Cameras.html.

CINEMA 4D R11 Quickstart – Materials.2011 [www-sivu]. [Luettu 01.02.2011]. Saatavissa: http://http.maxon.net/pub /r11/doc/quickstart_us.pdf.

CSS [www-sivu].[Luettu25.02.2011]. Saatavissa: http://fi.wikipedia.org/wiki/Cascading_Style_Sheets.

Custamization.2011 [www-sivu].[Luettu 01.02.2011]. Saatavissa: http://unity3d.com/support/documentation/Components/gui-Customiza- tion.html.

Finlandiapuisto 2010a. [www-sivu].[Luettu 01.02.2010]. Saatavissa: http://www.finlandiapuisto.fi/pages/fi/puistoa-rakennetaan /virtuaa-limalli.php.

GameObjects. 2010. [www -sivu].[Luettu25.02.2011]. Saatavissa: http://unity3d.com/support/documentation/Manual/GameObjects.htm.l

Global illumination. 2011 [www-sivu].[Luettu 01.02.2011]. Saatavissa:http://en.wikipedia.org/wiki/Global_illumination.

HDR. 2011 [www-sivu].[Luettu 01.02.2011]. Saatavissa:

http://fi.wikipedia.org/wiki/HDR.

HTML [www-sivu].[Luettu25.02.2011]. Saatavissa: http://fi.wikipedia.org/wiki/HTML.

Importing asset. 2010. [www-sivu].[Luettu25.02.2011]. Saatavissa: http://unity3d.com/support/documentation/Manual/Importing%20Assets.html.

Interview with Nick Wilson of Metaversed .2007 [www-sivu]. [Luettu 01.02.2010]. Saatavissa: http://www.usablemarkets.com/2007/11/11/interview-with-nick-wilson-of-metaversed/.

JavaScript-kielen alkeet - osa 1 2010 [www-sivu]. [Luettu 25.02.2011]. Saatavissa: http://weppipakki.com/js/opas/alkeet1.htm.

Lightmaps [www-sivu].[Luettu 01.02.2011]. Saatavissa: http://en.wikipedia.org/wiki/Lightmap.

Maya. 2011. [www -sivu].[Luettu25.02.2011]. Saatavissa: http://fi.wikipedia.org/wiki/Maya_%28tietokoneohjelma%29.

Spagnuolo Michela & Falcidieno Bianca. 2009.[www-sivu].[Luettu 01.02.2011]. Saatavissa: http://ieeexplore.ieee.org.elib.tamk.fi/stamp/stamp.jsp?tp=&arnumber=4804828.

Leavitt Neal. 2006. [www-sivu].[Luettu 01.02.2011]. Saatavissa: http://ieeexplore.ieee.org.elib.tamk.fi/stamp/stamp.jsp?tp=&arnumber=1703304.

MAXON Manual 2010a.[HTML Help System 11.530] [viitattu 28.2.2011]. Saatavissa: Cinema4d-ojelmasta:Manual/CINEMA 4D/Material manager/Shader.

Niemi T.2011 [www-sivu].[Luettu 01.02.2011]. Saatavissa: http://cc.joensuu.fi/~tniemi/3d/3.html

Non-uniform rational B-spline.2011 [www-sivu].[Luettu 01.02.2011]. Saatavissa:http://en.wikipedia.org/wiki/Nurbs.

Polygon mesh.2011 [www-sivu].[Luettu 01.02.2011]. Saatavissa: http://en.wikipedia.org/wiki/Polygon_mesh.

Perusopetuksen opetussuunnitelman perusteet. 2004 [www-sivu].[Luettu 01.02.2011]. Saatavissa: http://www02.oph.fi/ops/perusopetus/pops_web.pdf.

Quest3d. 2009. [www-sivu].[Luettu 01.02.2011]. Saatavissa: http://quest3d.com/index.php?id=208.

Ryan Paul. 2009. [www-sivu].[Luettu 01.02.2011]. Saatavissa:

http://arstechnica.com/software/news/2009/04/google-releases-3d-graphics-plugin-for-browsers.ars.

Second Live [www-sivu].[Luettu 01.02.2011]. Saatavissa: http://fi.wikipedia.org/wiki/Second_Life.

Sound. 2011[www-sivu].[Luettu25.02.2011]. Saatavissa: http://unity3d.com/support/documentation/Manual/Sound.html.

Unity.2011 [www-sivu].[Luettu 01.02.2011]. Saatavissa: http://unity3d.com/company/fast-facts.

Unity Script [www-sivu].[Luettu25.02.2011]. Saatavissa: http://unity3d.com/support/documentation/Manual/Scripting.htm.

Virtuaalitodellisuus.2010 [www-sivu].[Luettu 01.02.2010]. Saatavissa:http://fi.wikipedia.org/wiki/Virtuaalitodellisuus#Virtuaalimaailmat

VRLM.2011 [www-sivu].[Luettu 01.02.2011]. Saatavissa: http://en.wikipedia.org/wiki/VRML.

WebGL.2009 [www-sivu].[Luettu 01.02.2010]. Saatavissa: http://code.google.com/p/webhierarkia/wiki/WebGL.

What is X3D? [www-sivu].[Luettu 01.02.2010]. Saatavissa: http://www.web3d.org/about/overview/.

World of Warcraft.2011 [www-sivu].[Luettu 01.02.2011]. Saatavissa: http://fi.wikipedia.org/wiki/World_of_Warcraft#Pelin_eteneminen.

3DMax.2011[www -sivu].[Luettu25.02.2011]. Saatavissa: http://fi.wikipedia.org/wiki/3ds_Max.

Kuvalähteet:

Ancient.2011. [www-sivu].[Luettu 01.02.2011]. Saatavissa: http://alternativaplatform.com/swf/demos/ancient/.

Barcinski & Jeanjean.2011.[www-sivu].[Luettu 01.02.2011]. Saatavissa: http://www.barcinski-jeanjean.com/.

Cameras.2011 [www-sivu].[Luettu 01.02.2011]. Saatavissa: http://unity3d.com/support/documentation/Manual/Cameras.html.

Controls.2011 [www-sivu].[Luettu 01.02.2011]. Saatavissa: http://unity3d.com/support/documentation/Components/gui-Controls.html.

Custamization.2011.[www-sivu].[Luettu 01.02.2011]. Saatavissa: http://unity3d.com/support/documentation/Components/gui-Customiztion.html.

Finlandiapuisto.2011b.[www-sivu].[Luettu 01.02.2011].Saatavissa: http://www.finlandiapuisto.fi/media/video/video.html.

Global Illumination[www-sivu].[Luettu 01.02.2011].Saatavissa: http://en.wikipedia.org/wiki/Global_illumination.

Kämppi, Toni. 2007.Malinnus ja teksturointi virtuaalisovellukseen Me diatekniikka. Lahden ammattikorkeakoulu. Opinnäytetyö

MAXON Manual 2010b. [HTML Help System 11.530] [viitattu 28.2.2011]. Saatavissa Cinema4d-ojelmasta:Manual/CINEMA 4D/Material manager/Objects/NURBS/Extrude.

MAXON Manual 2010c.[HTML Help System 11.530] [viitattu 28.2.2011]. Saatavissa Cinema4d-ojelmasta: Manual/CINEMA 4D/ Objects/NURBS/Lathe.

MAXON Manual 2010d.[HTML Help System 11.530] [viitattu 28.2.2011]. Saatavissa Cinema4d-ojelmasta:Manual/CINEMA 4D/ Objects/NURBS/Loft.

MAXON Manual 2010e.[HTML Help System 11.530] [viitattu 28.2.2011]. Saatavissa Cinema4d-ojelmasta:Manual/CINEMA 4D/ Objects/NURBS/Sweep.

SecondLive. [www-sivu].[Luettu 01.02.2011]. Saatavissa: http://secondlife.com/destination/the-secret-garden.

Sprites and billboarding.2011[www-sivu].[Luettu 01.02.2011]. Saatavissa: http://goanna.cs.rmit.edu.au/~gl/teaching/ Interactive3D/2009 /lecture9.html.

Unity Web Player and browser communication.2011[www-sivu]. [Luettu 01.02.2011]. Saatavissa:http://unity3d.com/ support/ documentation/Manual/ Unity%20Web%20 Player% 20and%-20browser%20communication.html.

8 LIITEET

UnityGui-ohjelmakoodi:

```
//Globaalit muuttujat

//Ikkunoiden päälle/pois muuttujat

static var vipu = false;
static var vipu1 = false;
static var vipu2 = false;
static var vipu3 = false;
static var vipu4 = true;
static var vipu5 = true;
static var ohje = false;

//Muutttuja näkymän asettamiselle

static var nakyma;

//Selaimelle lähtettävä muuttuja, joka sulkee valitun Div-osion

static var tyylit;

//Selaimelle lähetettävä muuttuja, joka lähettää avattavan WWW-osoitteen

static var vaihe_kuva;

//Selaimelle lähetettävä muuttuja, joka lähettää avattavan WWW-osoitteen

static var vaihe_video;

//Selaimelle lähetettävä muuttuja, joka lähettää avattavan WWW-osoitteen

static var vaihe_teksti;


// muuttuja merkkijonon tallentamiseen

static var tyyli = "', 'layer";

//apu-muuttujat merkkijonojen tallentamiseen

static var apu;
static var apu2;


//Kamera muuttujat

var camera1 : Camera;
var camera2 : Camera;
var camera3 : Camera;
var camera4 : Camera;
var camera5 : Camera;
var camera6 : Camera;

//Kameroiden aloitus tilan asetus

camera1.enabled = true;
camera2.enabled = false;
camera3.enabled = false;
camera4.enabled = false;
camera5.enabled = false;
camera6.enabled = false;
```

```
//Tekstuuri muuttujien asettelu

var controlTexture : Texture2D;
var logo : Texture2D;

//Teksti muuttujien asettelu

var vaihe_1 : TextAsset;
var vaihe_2 : TextAsset;
var vaihe_3 : TextAsset;
var vaihe_4 : TextAsset;
var vaihe_5 : TextAsset;
var vaihe_6 : TextAsset;
var ohje1 : TextAsset;
var kaikki1 : TextAsset;

// Teksti osioiden aktivointi muuttuja

static var vaihe1 = false;
static var vaihe2 = false;
static var vaihe3 = false;
static var vaihe4 = false;
static var vaihe5 = false;
static var vaihe6 = false;
static var vaihe7 = false;
static var vaihe8 = false;
static var kaikki = false;

// Käyttöliittymän ikkunoiden koko asetukset

var windowRect : Rect = Rect (0, 123, 350, 400);
var windowRect1 : Rect = Rect (0, 123, 350, 400);
var windowRect2 : Rect = Rect (0, 123, 350, 400);
var windowRect4 : Rect = Rect (0, 650, 2000, 150);
var windowRect5 : Rect = Rect (0, 123, 350, 300);
var windowRect6 : Rect = Rect (0, 123, 350, 300);

//Painikkeiden, tekstin ja ikunoiden tyyli-muuttujat

public var myStyle :GUIStyle;
public var menuStyle :GUIStyle;
public var tekstiStyle :GUIStyle;
public var buttonStyle :GUIStyle;
public var buttonStyle1 :GUIStyle;
public var tekstibutton :GUIStyle;
public var closebutton :GUIStyle;
public var closebutton2 :GUIStyle;
public var buttonesittely :GUIStyle;
public var buttonvapaa :GUIStyle;
public var buttonohje :GUIStyle;
public var buttonkamera :GUIStyle;
public var buttonvaihe1 :GUIStyle;
public var buttonvaihe2 :GUIStyle;
public var buttonvaihe3 :GUIStyle;
public var buttonvaihe4 :GUIStyle;
public var buttonvaihe5 :GUIStyle;
public var buttonvaihe6 :GUIStyle;
public var buttonvaihe7 :GUIStyle;

//Ikkunan skaalautuvuus muuttujat

var resolution_ratio : float;
var actual_height : float;
var native_height : float = 768;

//On-Gui funktio

function OnGUI() {
```

```
//Ikkunan skaalautumisen asetus

        actual_height = Screen.height;
        resolution_ratio = actual_height / native_height;

        GUI.matrix = Matrix4x4.TRS (Vector3(0, 0, 0), Quaternion.identity, Vector3
(resolution_ratio, resolution_ratio, 1));

//Logon koko ja sijainti asetukset

GUI.Label (Rect (0, 0, 500, 130),logo);

//Ikkunan avaamis-funktiot

if (vipu ==true) {
windowRect = GUI.Window (0, windowRect, WindowFunction, 'nakyma',myStyle);
}

if (vipu1 ==true) {
windowRect1 = GUI.Window (1, windowRect1, WindowFunction1, "My Win-
dow",myStyle);
}
if (vipu2 ==true) {
windowRect2 = GUI.Window (2, windowRect2, WindowFunction2, "My Win-
dow",myStyle);
}
if (vipu3 ==true) {
windowRect4 = GUI.Window (4,Rect (0, 650, 2000, 150), WindowFunction4,
"",menuStyle);
}
if (vipu4 ==true) {
windowRect5 = GUI.Window (5,windowRect5, WindowFunction5, "Valikko",myStyle);
}
if (kaikki ==true) {
windowRect6 = GUI.Window (6,Rect (0, 650, 2000, 150), WindowFunction6,
"",menuStyle);
}
if (ohje ==true) {
windowRect = GUI.Window (0, windowRect, WindowFunction, 'Ohje',myStyle);
}
}

//Sivu ikkuna, jossa teksti-muuttujat näkyvät

function WindowFunction (windowID : int)  {

//Sulje-painike

if (GUI.Button(Rect(350,5,40,30),"Close",closebutton))
{

//

if (kaikki ==true || ohje == true)
{

//Ikkunoiden päälle/pois asettaminen

vipu=false;
vaihe1=false;
vaihe2=false;
vaihe5=false;
vaihe4=false;
vaihe5=false;
vaihe6=false;
vaihe7=false;
kaikki= false;
ohje=false;
vipu4=true;
```

```
//Kameran asettaminen

camera1.enabled = true;
camera2.enabled = false;
camera3.enabled = false;
camera4.enabled = false;
}
else
{
        //Näkymä muuttujan toiminnot tiloissa 1 ja 2

        switch(nakyma)
        {
        case 1:
                kaikki = false;
                vaihe1 = false;
                vaihe2 = false;
                vaihe3 = false;
                vaihe4 = false;
                vaihe5 = false;
                vaihe6 = false;
                vipu=false;
                vipu1=false;
                vipu2=true;
                vipu3=false;
                vipu4=false;
                camera1.enabled = false;
                camera2.enabled = false;
                camera3.enabled = true;
                camera4.enabled = false;
                camera5.enabled = false;
                camera6.enabled = false;
                Application.OpenURL (tyylit);
                print (tyylit);
        break;
        case 2:
        kaikki = true;
                vipu=false;
                vipu1=false;
                vipu2=false;
                vipu3=false;
                vipu4=false;
                vaihe1 = false;
                vaihe2 = false;
                vaihe3 = false;
                vaihe4 = false;
                vaihe5 = false;
                vaihe6 = false;
                vaihe7 = false;
                camera1.enabled = true;
                camera2.enabled = false;
                camera3.enabled = false;
                camera4.enabled = false;
                camera5.enabled = false;
                camera6.enabled = false;
                Application.OpenURL (tyylit);
                print (tyylit);
 break;
}
}
}

//Teksti muuttujien aktinointi If-lauseet

if(vaihe1==true)
{
GUI.Label (Rect (50, 80, 300, 250), vaihe_1.text,tekstiStyle);
}
if(vaihe2==true)
{
```

```
GUI.Label (Rect (50, 80, 300, 250), vaihe_2.text,tekstiStyle);
}
if(vaihe3==true)
{
GUI.Label (Rect (50, 80, 300, 250), vaihe_3.text,tekstiStyle);
}
if(vaihe4==true)
{
GUI.Label (Rect (50, 80, 300, 250), vaihe_4.text,tekstiStyle);
}
if(vaihe5==true)
{
GUI.Label (Rect (50, 80, 300, 250), vaihe_5.text,tekstiStyle);
}
if(vaihe6==true)
{
GUI.Label (Rect (50, 80, 300, 250), vaihe_5.text,tekstiStyle);
}
if(vaihe7==true)
{
GUI.Label (Rect (50, 80, 300, 250), vaihe_5.text,tekstiStyle);
}
if(vaihe8==true)
{
GUI.Label (Rect (50, 80, 300, 250), vaihe_5.text,tekstiStyle);
}
if (ohje ==true) {
GUI.Label (Rect (50, 80, 300, 250), ohje1.text,tekstiStyle);
}
if (kaikki ==true) {
GUI.Label (Rect (50, 80, 300, 250), kaikki1.text,tekstiStyle);
}
}

//Vapaaliikkuminen ikkuna

function WindowFunction2 (windowID : int)  {
if (GUI.Button(Rect(350,5,40,30),"Close",closebutton))
{

vipu2=false;
vipu4=true;
camera1.enabled = false;
camera2.enabled = false;
camera3.enabled = true;
camera4.enabled = false;
}
GUI.Box (Rect (100, 100, 200, 200),controlTexture);
}

//Esitys: valitun kohteen alavalikko-ikkuna (vaihe1-6.)

function WindowFunction4 (windowID : int)  {

if (GUI.Button(Rect(Screen.width-200,0,40,30),"Close",closebutton2))
{

//Sulje-painikkeen sulkutoiminnot nakyma-muuttujan arvoilla

switch(nakyma)
{
case 1:
        kaikki = false;
        vaihe1 = false;
        vaihe2 = false;
        vaihe3 = false;
        vaihe4 = false;
        vaihe5 = false;
        vaihe6 = false;
        vipu=false;
```

```
            vipu1=false;
            vipu2=true;
            vipu3=false;
            vipu4=false;


            camera1.enabled = false;
            camera2.enabled = false;
            camera3.enabled = true;
            camera4.enabled = false;
            camera5.enabled = false;
            camera6.enabled = false;

            //Selaimen suljeikkuna funktion kutsu muuttujan tyylit arvolla

            Application.OpenURL (tyylit);
            print (tyylit);
  break;
case 2:
  kaikki = true;
            vipu=false;
            vipu1=false;
            vipu2=false;
            vipu3=false;
            vipu4=false;
            vaihe1 = false;
            vaihe2 = false;
            vaihe3 = false;
            vaihe4 = false;
            vaihe5 = false;
            vaihe6 = false;
            vaihe7 = false;
            camera1.enabled = true;
            camera2.enabled = false;
            camera3.enabled = false;
            camera4.enabled = false;
            camera5.enabled = false;
            camera6.enabled = false;

            //Selaimen suljeikkuna funktion kutsu muuttujan tyylit arvolla

            Application.OpenURL (tyylit);
            print (tyylit);
            break;
            }
}

            //Esitys: valitun kohteen video-, kuva-ja teksti-painikeet


            if (GUI.Button(Rect(Screen.width/2-150,10,100,60),"Kuva",buttonStyle))
            {

                        //Apu-muuttujan merkkijonon asettaminen

                        apu = 'javasc
            ript:openWindow('+"'"+''+vaihe_kuva+''+tyyli+''+"'"+');';

                        //Selaimen suljeikkuna funktion kutsu muuttujan tyylit arvolla

                        Application.OpenURL (tyylit);
                        print (tyylit);

                        //Tyylit muuttujan arvon vaihtaminen

                        tyylit='javascript:suljeikkuna('+"'"+'layer'+"'"+');';

                        //Selaimen OpenWindow funktion kutsu muuttujan apu arvolla

                        Application.OpenURL (apu);
```

```
                print (apu);
        }
        if (GUI.Button(Rect(Screen.width/2,10,100,60),"Video",buttonStyle1))
        {
                //Apu2-muuttujan merkkijonon sisälle tuleva tyyli1 muuttuja

                var tyyli1 = "', 'layer2";


                /* Selaimen funktion OpenWindow taso asetettu layer2 + vaihe_teksi
                muuttujan arvo
                */

                var
        apu2='javascript:openWindow('+"'"+''+vaihe_video+''+tyyli1+''+"'"+');';
                print (apu2);

                //Selaimen suljeikkuna funktion kutsu muuttujan tyylit arvolla

                Application.OpenURL (tyylit);
                tyylit='javascript:suljeikkuna('+"'"+'layer2'+"'"+');';
                print (tyylit);

                //Selaimen OpenWindow funktion kutsu muuttujan apu2 arvolla

                Application.OpenURL (apu2);

        }
        if (GUI.Button(Rect(Screen.width/2+150,10,100,60),"Teksti",tekstibutton))
        {


                //Apu-muuttujan merkkijonon asettaminen

                apu = 'javasc
        ript:openWindow('+"'"+''+vaihe_teksti+''+tyyli+''+"'"+');';


                //Selaimen suljeikkuna funktion kutsu muuttujan tyylit arvolla

                Application.OpenURL (tyylit);
                print (tyylit);
                tyylit='javascript:suljeikkuna('+"'"+'layer'+"'"+');';

                //Selaimen OpenWindow funktion kutsu muuttujan apu arvolla

                Application.OpenURL (apu);
                print (apu);
        }


}

//Esitys-painikkeen kaikki-alavalikko

function WindowFunction6 (windowID : int)  {

        // sulje-painike

        if (GUI.Button(Rect(Screen.width-200,0,40,30),"Close",closebutton2))
        {
                kaikki = false;
                vipu=false;
                vipu1=false;
                vipu2=false;
                vipu3=false;
                vipu4=true;
                camera1.enabled = true;
                camera2.enabled = false;
                camera3.enabled = false;
```

```
                camera4.enabled = false;
                camera5.enabled = false;
                camera6.enabled = false;
        }

        //Osio vaihe1: Aiheen käsittely painike
        if (GUI.Button(Rect(Screen.width/2-500,20,100,60),"Vaihe 1",buttonvaihe1))
        {
                vaihe_kuva="vaihe1_kuva";
                vaihe_video="vaihe1_video";
                vaihe_teksti="vaihe1_teksti";
                kaikki = false;
                vaihe1 = true;
                vaihe2 = false;
                vaihe3 = false;
                vaihe4 = false;
                vaihe5 = false;
                vaihe6 = false;
                vipu4=false;
                vipu3 = true;
                vipu1 = false;
                vipu2 = false;
                camera1.enabled = false;
                camera2.enabled = false;
                camera3.enabled = false;
                camera4.enabled = true;
                camera5.enabled = false;
                camera6.enabled = false;

        //Tyylit muuttujan arvo asetettu ”layer”

                tyylit='javascript:suljeikkuna('+"'"+'layer'+"'"+');';

        //Apu-muuttujan merkkijonon asettaminen

        apu = 'javascript:openWindow('+"'"+''+vaihe_kuva+''+tyyli+''+"'"+');';

        //Selaimen OpenWindow funktion kutsu muuttujan apu arvolla

                Application.OpenURL (apu);

}

        //Osio vaihe2: Tiedon etsintä ja  tuottaminen painike

        if (GUI.Button(Rect(Screen.width/2-350,20,100,60),"Vaihe 2",buttonvaihe2))
                {
                vaihe_kuva="vaihe2_kuva";
                vaihe_video="vaihe2_video";
                vaihe_teksti="vaihe2_teksti";
                kaikki = false;
                vaihe1 = false;
                vaihe2 = true;
                vaihe3 = false;
                vaihe4 = false;
                vaihe5 = false;
                vaihe6 = false;
                vipu4=false;
                vipu3 = true;
                vipu1 = false;
                vipu2 = false;
                camera1.enabled = false;
                camera2.enabled = true;
                camera3.enabled = false;
                camera4.enabled = false;
                camera5.enabled = false;
                camera6.enabled = false;
                tyylit='javascript:suljeikkuna('+"'"+'layer'+"'"+');';
        apu = 'javascript:openWindow('+"'"+''+vaihe_kuva+''+tyyli+''+"'"+');';
                Application.OpenURL (apu);
```

```
		}


	//Osio vaihe3: pohjustus projektorilla painike

	if (GUI.Button(Rect(Screen.width/2-200,20,100,60),"Vaihe 3",buttonvaihe3))
	{
		vaihe_kuva="vaihe3_kuva";
		vaihe_video="vaihe3_video";
		vaihe_teksti="vaihe3_teksti";
		kaikki = false;
		vaihe1 = false;
		vaihe2 = false;
		vaihe3 = true;
		vaihe4 = false;
		vaihe5 = false;
		vaihe6 = false;
		vipu4=false;
		vipu = false;
		vipu3 = true;
		vipu1 = false;
		vipu2 = false;
		camera1.enabled = false;
		camera2.enabled = true;
		camera3.enabled = false;
		camera4.enabled = false;
		camera5.enabled = true;
		camera6.enabled = false;
		tyylit='javascript:suljeikkuna('+"'"+'layer'+"'"+');';
	apu = 'javascript:openWindow('+"'"+''+vaihe_kuva+''+tyyli+''+"'"+');';
		Application.OpenURL (apu);
	}

	//Osio vaihe4: aiheen käsittely painike

	if (GUI.Button(Rect(Screen.width/2-50,20,100,60),"Vaihe 4",buttonvaihe4))
	{
		vaihe_kuva="vaihe4_kuva";
		vaihe_video="vaihe4_video";
		vaihe_teksti="vaihe4_teksti";
		kaikki = false;
		vaihe1 = false;
		vaihe2 = false;
		vaihe3 = false;
		vaihe4 = true;
		vaihe5 = false;
		vaihe6 = false;
		vipu4=false;
		vipu = false;
		vipu3 = true;
		vipu1 = false;
		vipu2 = false;
		camera1.enabled = false;
		camera2.enabled = false;
		camera3.enabled = false;
		camera4.enabled = true;
		camera5.enabled = false;
		camera6.enabled = false;
		tyylit='javascript:suljeikkuna('+"'"+'layer'+"'"+');';
		apu =
'javascript:openWindow('+"'"+''+vaihe_kuva+''+tyyli+''+"'"+');';
		Application.OpenURL (apu);
		}

	//Osio vaihe5: Aiheen taustoitus painike

	if (GUI.Button(Rect(Screen.width/2+100,20,100,60),"Vaihe 5",buttonvaihe5))
	{
		vaihe_kuva="vaihe5_kuva";
		vaihe_video="vaihe5_video";
```

```
                vaihe_teksti="vaihe5_teksti";
                kaikki = false;
                vaihe1 = false;
                vaihe2 = false;
                vaihe3 = false;
                vaihe4 = false;
                vaihe5 = true;
                vaihe6 = false;
                vipu4=false;
                vipu = false;
                vipu3 = true;
                vipu1 = false;
                vipu2 = false;
                camera1.enabled = false;
                camera2.enabled = true;
                camera3.enabled = false;
                camera4.enabled = false;
                camera5.enabled = false;
                camera6.enabled = false;
                tyylit='javascript:suljeikkuna('+"'"+'layer'+"'"+');';
        apu = 'javascript:openWindow('+"'"+''+vaihe_kuva+''+tyyli+''+"'"+');';
                Application.OpenURL (apu);
        }

        //Osio vaihe6: Aiheen virittäminen painike

        if (GUI.Button(Rect(Screen.width/2+250,20,100,60),"Vaihe 6",buttonvaihe6))
{
                vaihe_kuva="vaihe6_kuva";
                vaihe_video="vaihe6_video";
                vaihe_teksti="vaihe6_teksti";
                kaikki = false;
                vaihe1 = false;
                vaihe2 = false;
                vaihe3 = false;
                vaihe4 = false;
                vaihe5 = false;
                vaihe6 = true;
                vipu4=false;
                vipu = false;
                vipu3 = true;
                vipu1 = false;
                vipu2 = false;
                camera1.enabled = false;
                camera2.enabled = false;
                camera3.enabled = false;
                camera4.enabled = true;
                camera5.enabled = false;
                camera6.enabled = false;
                tyylit='javascript:suljeikkuna('+"'"+'layer'+"'"+');';
        apu = 'javascript:openWindow('+"'"+''+vaihe_kuva+''+tyyli+''+"'"+');';
                Application.OpenURL (apu);
        }
}

//Aloitus-valikon ikkuna

function WindowFunction5 (windowID : int)  {

// vapaaliikkuminen-painike

if (GUI.Button(Rect(150,60,100,60),"Vapaaliikkuminen",buttonvapaa))
{
        nakyma=1;
        vipu4=false;
        vipu = false;
        vipu1 = false;
        vipu2 = true;
        camera1.enabled = false;
        camera2.enabled = false;
```

```
            camera3.enabled = true;
            camera4.enabled = false;
            camera5.enabled = false;
            camera6.enabled = false;
}

//Esittely-painike

if (GUI.Button(Rect(10,60,100,60),"Esittely",buttonesittely))
{
nakyma=2;
kaikki =true;
vipu4 =false;
}

//Ohje-painike

if (GUI.Button(Rect(290,60,100,60),"Ohje",buttonohje))
{
ohje=true;
vipu4=false;
}
}
```

Selaimen Javascript-funktiot OpenWindow() ja Suljeikkuna():

```
//Muuttuja Unityn Application.OpenUrl() osoite kutsulle

var newwindow;

//Muuttuja Unityn Application.OpenUrl() kutsulle Div osion näyttämiseksi

var ikkuna;

//Muuttuja Unityn Application.OpenUrl() kutsulle Div osion sulkemiseksi

var ikkuna1;

//Testi-muuttuja

var joku;

//Taulu Html-sivun nimille

var videot = new Array("vaihe1_video", "vaihe2_video", "vaihe3_video", "vai-
he4_video", "vaihe5_video", "vaihe6_video", "vaihe7_video","vaihe1_kuva", "vai-
he2_kuva", "vaihe3_kuva", "vaihe4_kuva", "vaihe5_kuva", "vaihe6_kuva", "vai-
he7_kuva","vaihe1_teksti", "vaihe2_teksti", "vaihe3_teksti", "vaihe4_teksti", "vai-
he5_teksti", "vaihe6_teksti", "vaihe7_teksti");



// ikkunan avaaminen

function openWindow(obj,obj2)
            {

            //Unitystä tuleva osoite

            newwindow = obj;

            //Unitystä tuleva Div nimi

            ikkuna = obj2;
            //alert(newwindow)
            //alert(ikkuna)
```

```
        //For-lause HTML-sivut taulun käsittelemiseksi

        for (i = 0; i < videot.length; i++)
                        {
                        if(videot[i] == newwindow)
                                {
                                                //alert(i)
                                                joku = i;

                        //Div osion aktivointi

                        ikkuna = document.getElementById(ikkuna);
                        ikkuna.style.visibility = (ikkuna.style.visibility == 'visible')
        ?       'hidden' : 'visible';
                                //Funktio sivun avaamiselle Iframe ikkunaan
                                iframe();
                                }
                        }
                //alert(videot[joku])
        }


//ikkunan sulkeminen

function suljeikkuna(obj)
        {
        //alert(newwindow)
        ikkuna1=obj;

        for (i = 0; i < videot.length; i++)
                        {
                        if(videot[i] == newwindow)
                                {
                                                //alert(i)
                                                joku = i;

                        //Div osion sulkeminen

                        ikkuna1 = document.getElementById(ikkuna1);
                        ikkuna1.style.visibility = (ikkuna1.style.visibility == 'visible')
        ?       'hidden' : 'visible';

                                }
                        }
        }
```