KARELIA-AMMATTIKORKEAKOULU
Tietojenkäsittelyn koulutusohjelma

Mika Åke

UNITYN JA BLENDERIN YHTEISKÄYTTÖ PELINTEOSSA

Opinnäytetyö
Joulukuu 2013

Karelia AMMATTIKORKEAKOULU

OPINNÄYTETYÖ
Joulukuu 2013
Tietojenkäsittelyn koulutusohjelma

Karjalankatu 3
80200 JOENSUU
+358 50 311 9160

Tekijä(t)

Mika Åke

Nimeke

Blenderin ja Unityn yhteiskäyttö pelinteossa

Tiivistelmä

Tämän opinnäytetyön tarkoituksena on selvittää pelinkehitysympäristö Unityn ja 3D-mallinnusohjelma Blenderin yhteiskäyttöä. Erityisesti tarkastellaan sitä, miten nämä kaksi ohjelmistoa soveltuvat pienen mittakaavan pelien toteuttamiseen. Unity ja Blender ovat vaihtoehto pienille yrityksille, koska niiden käyttöönotto ei vaadi suuria investointeja. Blender on täysin ilmainen ja Unitystäkin on olemassa ilmainen versio, jota voi käyttää kaupallisiin tarkoituksiin tietyin edellytyksin. Työssä verrataan Blenderiä kaupallisiin 3D-mallinnusohjelmiin sekä pohditaan, mitä tarpeita peliprojekteilla on 3D-mallien suhteen ja voiko Blender täyttää nämä tarpeet.

Lisäksi opinnäytetyössä kehitettiin autopeliä, johon tulevat mallit toteutettiin Blenderissä. Työssä tarkastellaan sitä, miten 3D-mallien päälle luodaan materiaalit ja tekstuurit sekä miten ne voidaan animoida Blenderissä. Työssä selvitetään, miten 3D-mallien siirto Blenderistä Unityyn onnistuu mahdollisimman hyvin ja mitä asioita on syytä huomioida, kun ohjelmistoja käytetään yhdessä.

Blender tarjoaa erinomaisen vaihtoehdon kalliille 3D-mallinnusohjelmille, kuten 3Ds Maxille ja Mayalle. Käyttäjän omat mieltymykset ovat suurin merkittävä tekijä sopivan ohjelmiston valinnassa. Yhteiskäyttö Unityn kanssa toimi hyvin. Suurimmat ongelmat oli Blenderissä toteutettujen animaatioiden tuomisessa Unityyn. Muilta osin Blender ja Unity sopivat hyvin pienen mittakaavan pelien tekemiseen.

Kieli

suomi

Sivuja 47
Liitteet 1

Liitesivumäärä 2

Asiasanat

Unity, Blender, 3D-mallinnus

Karelia
UNIVERSITY OF APPLIED SCIENCES

THESIS
December 2013
Degree Programme in Business Information Technology

Karjalankatu 3
FI 80200 JOENSUU
+358 50 311 9160

Author(s)

Mika Åke

Title

Blender and Unity in Game Development

Abstract

The purpose of this thesis is to find out how game development tool Unity and 3D-modelling software Blender work together. Especially this work aims to examine the suitability of those programs to create small-scale games. Unity and Blender are an option for small companies, because using them does not require large investments. Blender is completely free and there is a free version of Unity available, which can be used for commercial purposes under certain terms. In this study Blender software and commercial software are compared and the needs of game projects are deliberated to find out whether Blender meets those needs.

Along with this thesis there runs a development of racing game. 3D-models for that game are created in Blender. The thesis introduces ways to add materials and textures for 3D-models and how they can be animated in Blender. This work shows ways how importing operation goes well from Blender to Unity and what should be taken into considered, when these programs are used together.

Blender seems to offer an excellent option for expensive 3D-modelling softwares, like 3Ds Max and Maya. The users' own preferences are a huge factor when it comes to choose a suitable program for him/her or for the team. Blender and Unity work smoothly together, the largest problems are with animations that are created in Blender and then imported to Unity. Otherwise Blender and Unity are suitable for creating small-scale games.

Language	Pages 47
Finnish	Appendices 1
	Pages of Appendices 2

Keywords

Unity, Blender, 3D-modelling

Sisältö

Termit ja lyhenteet

3D-MALLI	Tietokoneella luotu kolmiulotteinen esitys olennosta, esineestä tai rakennuksesta.
LINUX	Vapaaseen ja avoimeen koodiin perustuva käyttöjärjestelmä, jonka ytimen kehitti alun perin suomalainen Linus Torvalds.
ANDROID	Mobiililaitteille suunniteltu käyttöjärjestelmä. Rakennettu Linux-ytimen päälle.
IOS	Applen kehittämä käyttöjärjestelmä, joka on käytössä Applen omissa tuotteissa.
FPS	Lyhenne sanoista First Person Shooter. Toimintapeli, jonka kuvakulma on pelihahmon näkökulmasta kuvattuna.
INDIE	Itsenäinen, yleensä pienellä budjetilla toimiva tekijä/tekijäryhmä. Ei toimi minkään suuren julkaisijan alla, vaan itsenäisesti, omana yksikkönään.
Exporter	Muunnin, joka muuntaa tietyssä ohjelmassa tallennetun tiedoston toiseen muotoon, jolloin se voidaan esimerkiksi avata toisessa ohjelmassa.
Importtaus	Sellaisen tiedostotyypin tuominen ohjelmaan, jota ohjelma ei suoraan tue. Importtauksen avulla tiedosto saadaan avattua ohjelmassa.
Sprite	Kaksiulotteinen, pikseleistä koostuva grafiikka. Yleisin etenkin vanhoissa peleissä, mutta nykyään käytetään paljon muun muassa mobiilipeleissä.
AAA-pelit	Korkealaatuiset, yleensä suurella budjetilla toteutetut pelit. Esimerkkeinä Grand Theft Auto -pelisarja sekä Forza Motorsport -pelit.
Steam	Valve Corporationin yrityksen kehittämä, tämän hetken merkittävin ja suurin digitaalinen videopelien jakelualusta.
Renderöidä	Tietokoneohjelman avulla tuotettu kuva. Hyödynnetään etenkin 3D-grafiikassa.
PNG	Lyhenne sanoista Portable Network Graphics. Häviötön bittikarttagrafiikan tallennusmuoto.

1 Johdanto

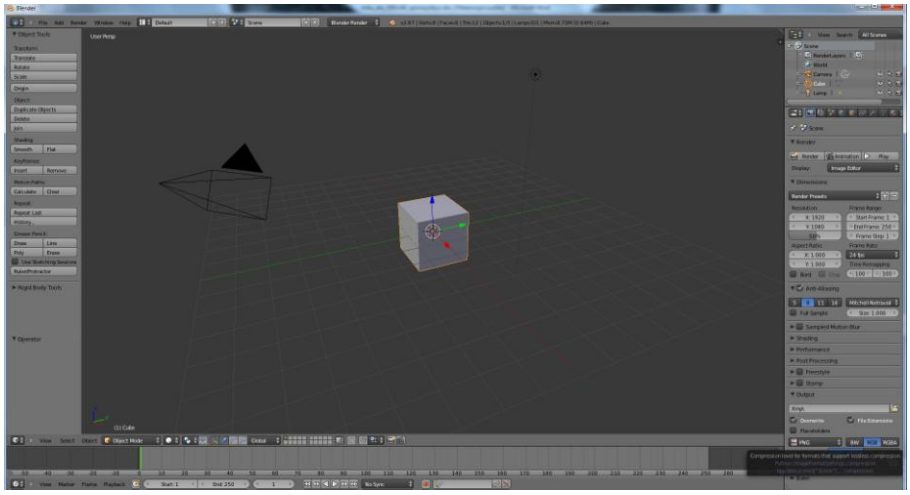
Pelien tekemiseen ei vaadita enää suuria investointeja, vaan tarjolla on lukuisia ilmaisia sovelluksia, joilla voi toteuttaa myös kaupallisiin tarkoituksiin tehtäviä pelejä. Tässä opinnäytetyössä otetaan tarkasteluun 3D-mallinnusohjelma Blender sekä pelinkehitysohjelma Unity. Blender on täysin ilmainen ohjelmisto, josta löytyy maksullisten kilpailijoiden ominaisuuksia. Unitystä puolestaan on tarjolla ilmainen versio, jolla voi tehdä pelejä monille eri alustoille, kuten Windowsille, Linuxille, Androidille, iOS:lle sekä monille muille. Tässä opinnäytetyössä keskitytään erityisesti selvittämään sitä, miten hyvin Blenderin ja Unityn yhteiskäyttö toimii ja miten nämä kaksi ohjelmistoa sopivat pienen mittakaavan pelin toteuttamiseen. Tavoitteena on selvittää, tarjoaako Unity ja etenkin Blender hyvän vaihtoehdon muille ohjelmistoille. Työssä käydään myös läpi niitä keinoja, joilla Blenderissä luodut 3D-mallit saadaan toimimaan oikein Unityssä.

Luvussa kaksi käydään läpi Blenderin ja Unityn historiaa sekä pohditaan sitä, miten näitä kahta ohjelmistoa on tähän mennessä hyödynnetty peleissä. Luvussa käydään myös läpi niitä tarpeita, joita peleillä on 3D-mallien suhteen ja sitä, mikä tarkoitus materiaaleilla ja tekstuureilla on 3D-malleissa.

Luku kolme käsittelee Wreck Garage -autopeliä ja luvussa käydään läpi myös materiaalien ja tekstuurien luominen 3D-malleille niin, että ne toimivat oikein Unityssä. Luvussa kerrotaan animaation toteuttamisesta 3D-mallille ja käydään läpi yksinkertainen animaatio, jossa vain liikutetaan 3D-mallin tiettyä osaa sekä ihmishahmolle luotava monimutkaisempi animaatio. Lopuksi käydään läpi 3D-mallin siirtäminen Unityssä luotuun peliprojektiin. Tässä luvussa käsitellään asetukset, jotka tulee ottaa huomioon importoinnissa sekä selvitetään miten tekstuurit ja animaatiot saadaan näkymään oikein. Neljännessä luvussa käydään läpi työn aikana saadut tulokset ja päätelmät.

2 Blender ja Unity 3D

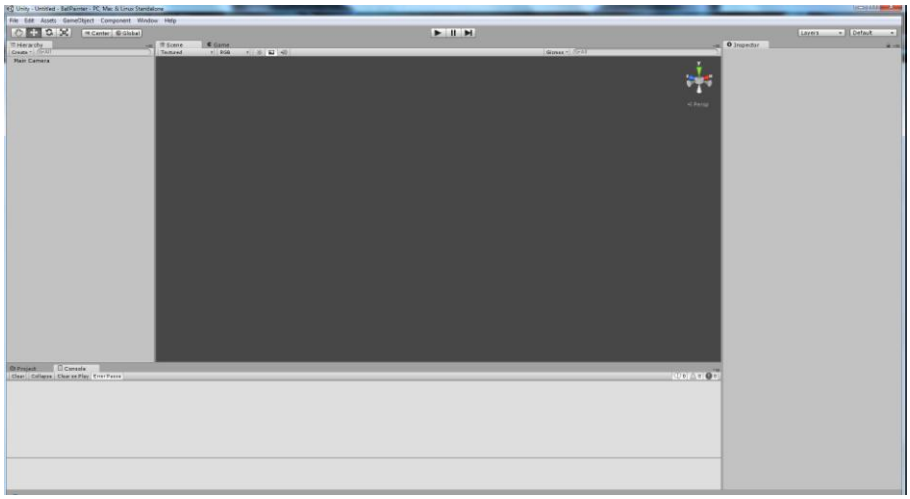
Blender (kuva 1) on 3D-mallien luomiseen suunniteltu ohjelmisto. Ohjelmaa levitetään GNU GLP -lisenssillä, joten se on täysin ilmainen ja vapaasti hyödynnettävissä myös kaupallisiin tarkoituksiin. Blenderin historia alkaa vuodesta 1988, jolloin hollantilainen Ton Roosendal kehitti sen ensimmäisen version yrityksensä käyttöön. Ohjelman kehittämistä jatkettiin myöhemmin ja vuonna 2002 perustettiin Blender Foundation -säätiö, jonka tarkoituksena on tukea Blenderin kehittämistä edelleen. (Blender.org 2013.) Tällä hetkellä Blender on menossa versiossa 2.69 ja siitä on ladattavissa omat versiot Windowsille (32- ja 64-bittinen), Linuxille (32- ja 64-bittinen) sekä Mac OS X:lle. Luvussa 2.1 tehdään tarkempaa vertailua Blenderin ja kaupallisten 3D-mallinnusohjelmien kesken ja tämän vertailun perusteella voidaan todeta, että Blender tarjoaa useita samoja ominaisuuksia, joita löytyy sen maksullisista kilpailijoista, kuten Autodesk-yrityksen Mayasta tai 3Ds Maxista.



Kuva 1. Blenderin aloitusnäkymä (versio 2.67b).

Unity3D tai pelkkä Unity, kuten sitä virallisesti kutsutaan, on Unity Technologies -yrityksen kehittämä ohjelma pelien tekemiseen (kuva 2). Unityllä voi tehdä pelejä ja sovelluksia lukuisille eri alustoille, kuten Windowsille, Linuxille, Androidille, iOS:lle sekä Windows Phonelle. Ohjelma soveltuu hyvin erityyppisten pelien tekemiseen, olivat ne sitten FPS-toimintapelejä, autopelejä tai vaikkapa 2D-

tasohyppelyitä. Unitystä löytyy myös kattavat mahdollisuudet toteuttaa 3D-mallien animaatioita. Unityn perusversio on lisänimeltään Free ja siitä on karsittu joitain ominaisuuksia maksulliseen Pro-versioon verrattuna, mutta yksinkertaisten pelien toteuttaminen onnistuu myös ilmaisella versiolla. Nimensä mukaisesti Pro-versio sisältää muun muassa kehittyneemmät tavat tuottaa animaatioita ja grafiikkaa. Se tulisi hankkia ilmaisversion tilalle, mikäli sitä käyttävän yrityksen tuotto on ollut vuodessa 100,000 dollaria (Unity Technologies 2013a). Tämän työn ohessa kehitettävä Wreck Garage -peli on toteutettu kokonaan Unity Free -versiolla. Monet pelialan yritykset hyödyntävät Unityä nykyään, ja esimerkkejä Unityllä luoduista peleistä ovat muun muassa Kerbal Space Program, Space Hulk ja suomalaisen Rovio-yrityksen Bad Piggies (Unity Technologies 2013b). Unityn kehittäjät myös päivittävät tiiviisti ohjelmistoa. Viime aikoina päivityksissä on keskitytty mobiililaitteiden tukemien ominaisuuksien lisäämiseen Unityyn, koska näille alustoille pelejä tehdään yhä enemmän ja enemmän.



Kuva 2. Unityn aloitusnäkymä (versio 4.2.2f1).

2.1 Blender vastaan kaupalliset 3D-mallinnusohjelmat

Yksiselitteistä vastausta kysymykseen, ovatko maksulliset ohjelmat parempia kuin Blender, on mielestäni vaikea antaa. Puolueettomia selvityksiä tästä on hankala löytää ja kaikkialla törmää siihen, miten ihmisten omat mieltymykset vaikuttavat asiaan. Jarmo Siira (2007) on pohtinut asiaa omassa opinnäytetyössään Avoimen lähdekoodin Blender vs. kaupallinen 3Ds Max, ja hän toteaakin työnsä yhteenvedossa, että 3D-mallinnusohjelmien vertailu on käytännössä mahdotonta ja siihen vaikuttavat hyvin vahvasti käyttäjien omat mieltymykset. Ominaisuuksien perusteella Blender on ottanut viime vuosina kiinni kaupallisia ohjelmia ja siihen on lisätty muun muassa kattavat animointityökalut. Mikäli verrataan 3D-mallinnusohjelmien ominaisuuslistoja, niin tällöin huomataan, että suuria eroja ei juuri ole (liite 1). Vertailun perusteella kaikki merkittävät ominaisuudet, jotka löytyvät kaupallisista ohjelmistoista, ovat mukana myös Blenderissä. Enää ei olekaan kysymys siitä, mihin kaikkeen Blender kykenee, vaan lähinnä siitä, minkä tyyppistä ohjelmistoa halutaan käyttää. Blenderin käyttöliittymä eroaa melko paljon monista kaupallisista 3D-mallinnusohjelmistoista. Tämä on merkittävä syy, miksi Blender koetaan hankalaksi käyttää. Blender hyödyntää hyvin paljon erilaisia näppäinyhdistelmiä ja komentoja. Monet toiminnot löytyvät myös painikkeina, mutta yleensä eri valikoiden alta. Blenderin hyödyntäminen tehokkaasti vaatiikin näiden näppäinkomentojen opettelemista ja jotkut voivat kokea tämän liian suurena kynnyksenä siirtyä esimerkiksi 3Ds Maxista Blenderiin. Osaavissa käsissä Blenderillä saa yhtä hyvää jälkeä aikaan kuin kaupallisilla 3D-mallinnusohjelmilla. Esimerkkejä tästä löytyy paljon Internetistä, kuten Blender Guru -sivustolta (http://www.blenderguru.com/winners-hall-of-fame/) sekä BlenderNation-sivulta (http://www.blendernation.com/). Blender on hyvä vaihtoehto myös freelancereiden työkaluksi. He ovat voineet kartuttaa osaamistaan ohjelmistosta riittävän hyvälle tasolle ja pystyvät tämän jälkeen tuottamaan sillä toimeksiantojen mukaisia töitä ilman, että joutuvat maksamaan ohjelmiston lisenssistä.

Käyttäjien kohtaamien ongelmatilanteiden ratkaisemissa kaupallisilla ohjelmilla on etumatka, koska kehittäjät tarjoavat teknistä tukea tuotteensa ostajille. Blenderin kehittäjillä ei luonnollisestikaan ole mahdollisuutta tähän, vaan ongelmatilanteissa on pyrittävä löytämään apua joko tutoriaalien kautta tai keskustelupalstoilta, esimerkiksi Blender Artists -palstalta. (http://blenderartists.org/forum). Neuvojen hakeminen näin ei ole yhtä nopeaa, kuin mitä se olisi teknisen tuen kautta. Sellaisissa peliprojekteissa, joissa on tiukka aikataulu, tällaiseen ei ole varaa. Pienemmät projektit pystyvät kestämään tällaista paremmin, etenkin jos aikataulu ei ole tiukka.

2.2 Blenderin ja Unityn hyödyntäminen peleissä

Unityä sekä Blenderiä on hyödynnetty jo paljon pelien tekemisessä. Sellaisia pelejä, jotka on toteutettu näiden molempien ohjelmien avulla, oli hankalampi löytää. Blenderin ja Unityn yhteiskäyttöä on hyödynnetty etenkin mobiilipeleissä. Eat Sheep (CG Cookie, 2012) ja Ginormous (Tunza Games, 2013) ovat molemmat iOS-laitteille tehtyjä pelejä, joiden 3D-mallit on tehty Blenderillä ja pelimoottorina käytetään Unityä. Molempien esimerkkien perusteella olen sitä mieltä, että Blenderillä on mahdollista toteuttaa persoonallista 3D-grafiikkaa ja Unity tarjoaa mahdollisuuden ottaa tämä grafiikka luovasti käyttöön peleissä.

Unityn ajatellaan olevan pelimoottori, joka soveltuu parhaiten pienen mittakaavan pelien tekemiseen. Unityn omilta sivuilta löytyy kattava lista sillä tehdyistä peleistä (http://unity3d.com/gallery/made-with-unity/game-list). Tätä listaa selaamalla voidaan huomata, että useimmat pelit ovat pieniä independent-tekijöiden toteutuksia ja yleensä suunnattu mobiililaitteille. Eräs syy tälle on se, että monet yleisesti käytössä olevat pelimoottorit on suunniteltu tarkasti tietyntyyppisiä pelejä varten. Esimerkiksi Unreal Engine ja Source Engine -pelimoottoreita hyödynnetään parhaiten 3D-toimintapeleissä, kuten Eidos Interactiven Batman: Arkham Asylumissa (2009) sekä Valve Corporationin Left 4 Dead 2:ssa (2009). Unity puolestaan on suunniteltu hyvin erilaisten pelien toteuttamiseen, eikä käyttäjien luovuutta ole haluttu rajoittaa vain tiettyihin pelityyppeihin. Vaikka useimmat Unityllä toteutetut pelit ovatkin

kolmiulotteisia, niin Unityn 4.3-versiossa mukaan liitettiin toimivat työkalut myös 2D-pelien tekemiseen. Aiemmin esimerkiksi sprite-grafiikan hyödyntämiseen Unityllä tehdyissä peleissä tarvittiin liitännäisiä. Uskon, että Unity tulee kasvattamaan entisestään suosiota pelintekijöiden joukossa tulevaisuudessa ja sillä tullaan toteuttamaan myös suurempia peliprojekteja. Unity Technologiesin mukaan Unityn käyttäjämäärä on tuplaantunut vuodesta 2012 (Unity Techologies 2013c). Tulossa on ainakin Mech-räiskintä Hawken (Meteor Entertainment) sekä Wasteland 2 (InXile Entertainment). Tällaisten suurien projektien myötä ja etenkin niiden onnistuessa Unity saa enemmän uskottavuutta ja pystyy osoittamaan, että myös se soveltuu niin sanottujen AAA-pelien tekemiseen.

Tulevaisuudessa peliala muuttuu entistä enemmän indie-pohjaiseksi. Pelien digitaaliset jakelualustat, kuten Steam, ovat mullistaneet alaa suuresti ja tarjonneet pienemmille kehittäjille mahdollisuuden saada pelinsä suuren yleisön tietoisuuteen. Enää siihen ei tarvita mittavaa markkinointia ja jakelukoneistoa. Pelialan yritykset ovat tulevaisuudessa nykyistä pienempiä, muutaman hengen yrityksiä. Tämä mahdollistaa suurempien riskien ottamisen pelisuunnittelussa ja tekijät pääsevät tehokkaammin hyödyntämään luovuuttaan, toisin kuin suuremmissa peliyhtiöissä, joissa keskitytään enemmän taattujen menestyspelien tekemiseen ja näiden jalostukseen jatko-osissa. Pienemmillä pelifirmoilla ei ole paljon rahaa käytössä, jolloin edullisemmat pelintekoon vaadittavat ohjelmistot tulevat tarpeeseen. Näitä ohjelmistoja edustavat nimenomaan Unity ja Blender.

2.3 Blenderin ja Unityn yhteiskäyttö tekniseltä kannalta

Unity tukee suoraan muutamien 3D-mallinnusohjelmien tiedostomuotoja, jolloin ne voidaan siirtää Unityyn. Blender on yksi näistä ohjelmista, mutta Unity ei kuitenkaan siirrä Blenderin blend-tiedostomuotoon tallennettua 3D-mallia suoraan, vaan hyödyntää siinä Blenderin omaa FBX-exportteria (Unity Technologies 2013d). Unity käytännössä avaa Blenderin taustalla ja suorittaa käännöksen FBX-muotoon. Tätä varten tietokoneelta, jolla halutaan tuoda Blenderissä luotuja 3D-malleja Uni-

tyyn, on löydyttävä myös Blender. Version tulee olla 2.60 tai uudempi, koska aiemmissa versioissa oli ongelmia FBX-exporterin kanssa (Unity Technologies 2013d).

Tällä hetkellä Blenderistä Unityyn importtauksessa siirtyvät 3D-mallista kaikki siihen liittyvät sijainnit, käännökset ja skaalaukset. Myös UV-karttoihin liittyvät tiedot sekä 3D-mallin animaatiot siirretään. Mikäli sellaiseen 3D-malliin, joka on jo siirretty Unityyn, tehdään muutoksia Blenderillä, siirtyvät muutokset tallentamisen jälkeen suoraan importoituun 3D-malliin, ilman uutta importtausta. (Unity Technologies 2013d.) Tarkemmin 3D-mallin siirtämistä Blenderistä Unityyn käsitellään luvussa 3.6.

2.4 3D-mallista peliobjektiksi

Blenderin ja Unityn yhteiskäyttö toimii yksinkertaistettuna niin, että aluksi luodaan 3D-malli Blenderissä, siihen lisätään tarvittavat materiaalit ja tekstuurit sekä halutaessa myös animaatiot. Tämän jälkeen siirrytään Unityyn, johon luodaan uusi peliprojekti. Toteutettu 3D-malli importoidaan Unityyn ja tarpeen mukaan muutetaan importointiasetuksia niin, että mallin tuominen onnistuu oikein. Siirto-operaatio on hyvin yksinkertainen, mutta mikäli tiettyjä vaiheita ei ole huomioitu jo Blenderissä, kohdataan ongelmia. Näitä ongelmakohtia pyrin selvittämään tulevissa luvuissa.

Lisäksi on syytä huomioida minkä tyyppinen toteutettava 3D-malli on. Peleissä käytettävät peliobjektit voidaan jakaa karkeasti kahteen ryhmään sen mukaan, mikä rooli niillä pelissä on: staattisiin ja dynaamisiin objekteihin. Staattisiin objekteihin kuuluvat sellaiset objektit, joihin pelaaja ei voi vaikuttaa ja joille ei ole tarvetta lisätä animaatioita. Tällaisia objekteja ovat muun muassa rakennukset ja dekoraatiot. Dynaamisia objekteja ovat objektit, jotka liikkuvat ja joihin pelaaja voi vaikuttaa. Esimerkkeinä dynaamisista objekteista on ajettavat autot, avattavat aarrearkut tai pelihahmot.

Blender ei rajoita käyttäjien luovuttaa sen suhteen, millaisia peliobjekteja sillä voidaan toteuttaa. Erityyppiset objektit vaativat kuitenkin tiettyjen asioiden huomioimista, ennen kuin ne viedään Blenderistä Unityyn. Näitä käydään tarkemmin läpi tulevissa luvuissa.

2.5 Materiaalien ja tekstuurien tarkoitus

Tässä työssä ei tarkastella varsinaista 3D-mallien luomista Blenderissä. Aihe on laaja ja siihen on saatavilla paljon ohjeistusta Internetissä sekä kirjallisuutena. Esimerkkejä hyvistä tutoriaaleista ja ohjeistuksista voi löytää Blenderin omilta sivuilta (http://www.blender.org/support/tutorials/), Wikibooksista (http://en.wikibooks.org/wiki/Blender_3D:_Noob_to_Pro) sekä Roland Hessin teoksesta Blender Foundations (Focal Press 2010). Näiden avulla sellainen henkilö, joka ei ole koskaan käyttänyt mitään 3D-mallinnusohjelmaa, pystyy opettelemaan käyttämään Blenderiä alusta pitäen.

Erityyppisistä objekteista otan tarkasteluun yhden jo luomistani 3D-malleista, joka siirretään kehittämääni Wreck Garage -peliin. Staattisista objekteista esimerkkinä toimii yksinkertainen jakoavain. Liikkuvista objekteista otin mukaan auton renkaan (kuva 4). Animoitavana objektina toimii auto ja tarkemmin auton konepelti (kuva 5). Hahmomallina on pelkistetty ihmishahmo (kuva 6). Tietyt vaiheet tapahtuvat samalla tavalla, riippumatta siitä, millainen objekti on kyseessä. Tästä syystä tässä työssä ei käydä tarkemmin läpi esimerkiksi materiaalien ja tekstuurien lisäämistä jokaiselle erityyppiselle objektille kerrallaan.

Kuva 4. Teksturoitu rengas.

Kuva 5. Teksturoitu auton 3D-malli.

Kuva 6. Ihmishahmon 3D-malli ilman tekstuureita.

Jokaiselle 3D-mallille lisätään tarvittavat materiaalit ja tekstuurit, että se saadaan näyttämään realistiselta. Materiaalien avulla voidaan määritellä, miten eri kohdat 3D-mallissa esimerkiksi imevät itseensä ja heijastavat takaisin valoa. Materiaalilla 3D-mallista voidaan myös tehdä läpinäkyvä. Tekstuureiden avulla puolestaan voidaan antaa pinnoille syvyyttä ja monipuolisuutta. Eri pintojen vaikutukset esimerkinä toimivaan jakoavaimeen huomaa kuvasta 7, jossa ei ole lainkaan materiaaleja tai tekstuureita. Kuva 8 näyttää saman 3D-mallin materiaalien kanssa ja kuvassa 9 jakoavaimeen on lisätty myös tekstuurit.

Kuva 7. Pinnoittamaton 3D-malli.

Kuva 8. 3D-malli materiaaleilla.

Kuva 9. 3D-malli tekstuureilla.

3 Wreck Garage

Wreck Garage -autopelin perusidea pohjautuu California Dreamsin julkaisemiin Street Rod 1 (1989) ja Street Rod 2 (1990) -peleihin. Näissä peleissä pelaaja ostaa itselleen auton ja sen jälkeen hän voi alkaa parannella sitä omassa autotallissaan. Hän voi vaihtaa siihen osia, aina moottoria myöten tai maalata autonsa erivärisek-si. Sen jälkeen pelaaja voi lähteä autollaan kaupungille ja haastaa toisia kuljettajia autokilpailuun. Näiden kilpailujen panoksena on raha tai kuljettajan oma auto. Saadulla rahalla voi jatkaa autonsa parantelua tai ostaa kokonaan uuden. Wreck Garagessakin (kuva 10) pelaaja hankkii halvalla auton ja sen jälkeen hän voi ostaa

siihen osia, joko käytettynä tai uutena. Tämä peli nojaa kuitenkin vahvasti rap-pioromantiikkaan ja pelissä ajetaan vanhoilla, ruosteisilla autoilla ja pelaajan tukikohtana toimiva korjaamokin on pelkkä laudoista kasattu rakennus. Jos peliä pitäisi verrata jo olemassa oleviin peleihin, niin tällöin sopivin yhdistelmä olisi Street Rod (1 ja 2) sekä Flatout-pelisarjan pelit (Bugbear Entertainment, 2004 - 2007).

Kuva 10. Kuvakaappaus Wreck Garage -autopelistä.

3.1 Pelin toteutus

Wreck Garage on 3D-peli. Kaikki pelissä esiintyvät autot, rakennukset ja esineet on toteutettu aitoina 3D-malleina. 3D-mallien toteuttamiseen on käytetty pelkästään Blenderiä. Lisäksi malleihin on lisätty tekstuurit Blenderissä. Tämän jälkeen 3D-mallit on siirretty Unityyn, jossa varsinainen pelinkehitys tehdään. Erilaisia 3D-malleja pelissä tulee olemaan satoja, joista osa on hyvin yksinkertaisia, kuten vaikkapa työkaluja, ja osa taas monimutkaisia, kuten pelissä olevat autot. Usein autopeleissä oleviin autoihin ei ole tehty kovin tarkkaa vahingonmallinnusta ja tällöin saadaan vapauksia siihen, miten tarkasti autot pitää näihin peleihin mallintaa.

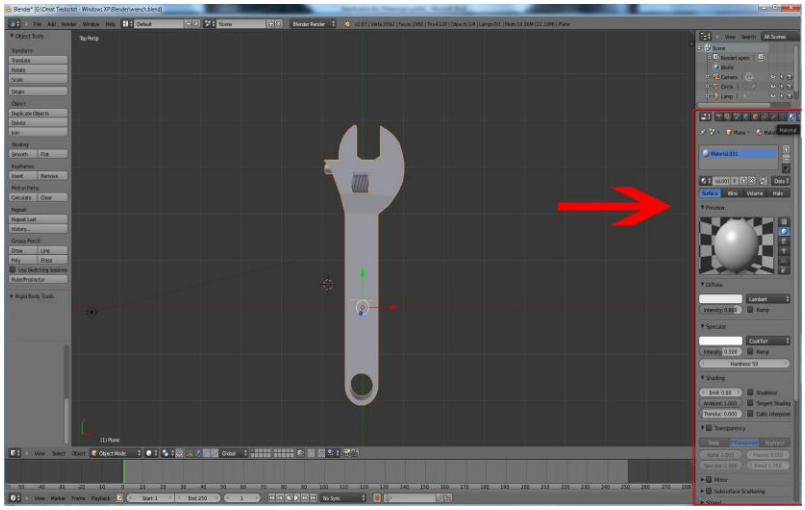
Wreck Garagessa autot pystytään särkemään ja purkamaan, jolloin tarvitaan enemmän työtä niiden mallintamiseen, kun esimerkiksi moottorin sisätilat tulee mallintaa 3D-malliin.

Opinnäytetyön aikana syntynyt peli on varhainen versio, johon on sisällytetty osa grafiikoista ja tärkeimmät toiminnot, jotka liittyvät auton korjaamiseen, purkamiseen ja paranteluun korjaamolla. Tästä versiosta ei löydy vielä esimerkiksi lainkaan kilpailuja muita kuljettajia vastaan tai useita erilaisia autoja, joita pelaaja voi itselleen ostaa. Versio tarjoaa kuitenkin hyvän pohjan tämän opinnäytetyön tekemiselle ja tukee sitä omalta osaltaan. Pystyin toteuttamaan eritasoisia 3D-malleja Blenderissä, teksturoimaan nämä ja siirtämään ne sitten Unityyn. Osalle 3D-malleista toteutin myös tarvittavat animaatiot.

Wreck Garage -pelin 3D-grafiikat koostuvat erityyppisistä objekteista. Jako voidaan tehdä karkeasti staattisiin ja dynaamisiin objekteihin. Teen kuitenkin tarkemman luokittelun erityyppisiin dynaamisiin objekteihin selventääkseni niihin vaadittavia asetuksia Blenderissä. Jaan dynaamiset objektit vielä erikseen liikkuviin, animoitaviin ja orgaanisiin objekteihin. Staattisia objekteja ovat esimerkiksi työkalut ja rakennukset. Liikkuviksi objekteiksi voidaan laskea autojen renkaat sekä tietyt rikkoutuvat 3D-mallit, kuten puuaidat. Animoitavia objekteja ovat erityisesti tietyt auton osat. Orgaanisia objekteja ovat ihmiset sekä eläimet, kuten linnut. Vaikka opinnäytetyöni käsitteleekin vain yhtä peliä ja siihen tulevia 3D-malleja, pyrin antamaan yleiskuvan niistä asetuksista ja huomioista, joita on otettava huomioon erityyppisten objektien kohdalla. Pelistä riippuen käytössä on kuitenkin aina sama jako erityyppisten objektien kesken.

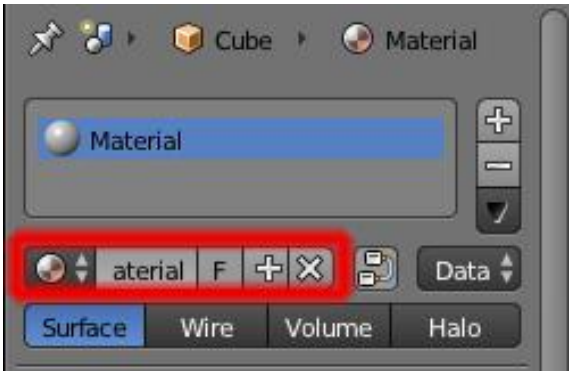
3.2 Materiaalien lisääminen

Materiaalien valinta 3D-malliin tapahtuu oman valikon kautta perusnäkymän oikeasta reunasta (kuva 11). Materiaaleja on mahdollista lisätä vain tiettyihin kohtiin 3D-mallia. Esimerkiksi jakoavaimeen riittää yksi materiaali, koska 3D-malli on melko yksinkertainen. Toisaalta taas auton 3D-malli tarvitsee useita erilaisia materiaaleja esimerkiksi lasipinnoille, renkaille sekä korin eri osille. Toisin sanoen voidaan todeta, että mitä monimutkaisempi objekti, sitä enemmän tarvitaan erilaisia materiaaleja. Materiaalit kannattaa nimetä, koska nämä nimet siirtyvät 3D-mallin mukana Unityyn. Siirrettäessä objektia Unityyn Unity tarkistaa, onko objektiin määritelty materiaaleja. Mikäli on, sen jälkeen luodaan määritellyn niminen materiaali Unityn projektikansioon. Mikäli määritellyn materiaalin nimi jo löytyy projektikansiosta, Unity käyttää samaa nimeä myös siirretyssä objektissa. Sellaiset pinnat, joiden käyttäjä haluaa näyttävän samalta, autopelissä esimerkiksi auton renkaat, voi määritellä samalla nimellä kaikkiin luotaviin auto-objekteihin. Jos käyttäjä haluaa myöhemmin muuttaa esimerkiksi kaikkien renkaiden väriä, hän voi muokata vain Unityn materiaalitiedostoa ja tämän jälkeen muutokset päivittyvät kaikkiin renkaisiin, jotka käyttävät muokattua tiedostoa. Kuitenkin omien kokemusteni perusteella jo pelkästään projektin selkeyden kannalta materiaalit kannattaa aina nimetä, muuten ollaan pian siinä tilanteessa, että Unityn projektikansiossa on materiaaleja, jotka on nimetty vain oletuksena toimivilla nimillä kuten “Material 1” ja ”Material 2”.



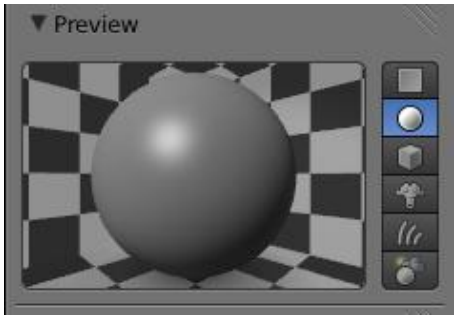
Kuva 11. Materiaalien valintaikkuna Blenderissä.

Materiaaleja voi lisätä Blenderissä valitsemalla ensin halutut kohdat 3D-mallista ja siirtymällä sitten materiaalien valintaikkunaan. Kuvassa 12 on laatikko, johon listataan eri materiaalit. Materiaaleja voi lisätä klikkaamalla Plus-painiketta ja niitä voi poistaa Miinus-painikkeen avulla. Materiaalien nimeäminen tapahtuu materiaalien listauslaatikon alapuolella olevaan tekstikenttään kirjoittamalla. Tekstikenttä on ympyröity punaisella kuvassa 12.



Kuva 12. Materiaalien listaus- sekä nimeämislaatikko.

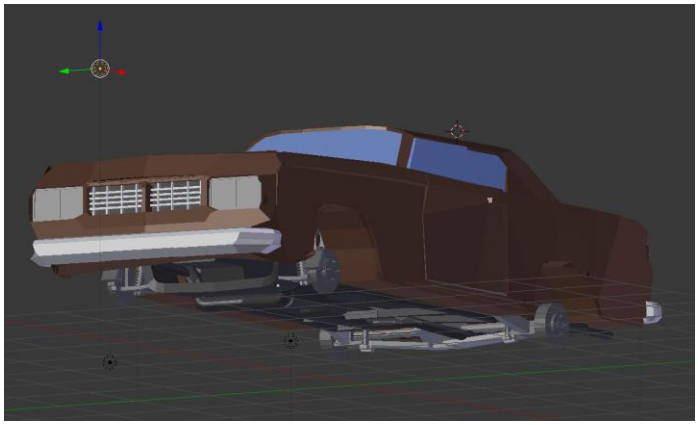
Preview-ikkuna näyttää valitun materiaalin asetukset reaaliajassa käyttäjälle (kuva 13). Esikatseluikkunan näkymä toteutetaan Blenderissä sen omalla renderöijällä, joten se vastaa lopullista renderöinnin tulosta eikä vääristymiä sen suhteen voi tulla (Hess 2010, 168). 3D-mallin muodon perusteella kannattaa valita sopiva muoto esikatseluikkunassa näkyvälle objektille. Tämä helpottaa selvittämään sen, miltä materiaali näyttää muokattavan 3D-mallin päällä.



Kuva 13. Materiaalien esikatseluikkuna.

Preview-ikkunan jälkeen voidaan valita **Diffuse-**arvot (kuva 14). Ne määrittävät sen, miten 3D-malliin tulevaa valoa käsitellään. Blender tarjoaa viisi erilaista tapaa sille, miten se laskee materiaalille tulevan valon. Näitä tapoja kutsutaan nimellä shader eli varjostin. **Oren-Nayer** sopii monille ei-kiiltäville pinnoille, **Lambert** kiiltäville metalleille ja muoveille, **Minnaert** tekstiileille ja kankaille, **Toon**-varjostimen avulla voi antaa 3D-malleille sarjakuvamaisen ilmeen sekä **Fresnel**-varjostimella pystyy vaikuttamaan esimerkiksi siihen, miten läpinäkyviä pintoja käsitellään ja näytetään (Blender Wiki 2013a). **Intensity-**säätimen avulla määritellään puolestaan se, miten paljon materiaali heijastaa valoa takaisin. Esimerkiksi jakoavaimelle annoin arvon 0.350, joka tarkoittaa sitä, että 35 % materiaaliin tulevasta valosta heijastetaan takaisin. Diffuse-kohdasta valitaan myös materiaalille ”perusväri”. Tämä väri toimii pohjana materiaalille ja se myös näytetään 3D-mallissa. Itse olen tehnyt niin, että olen valinnut perusväriksi sellaisen värin, joka on lähimpänä 3D-mallin lopullista väritystä. Väri tulee myös näkyviin teksturoinnissa käytettävään UV-karttaan, joten erilaisten värien avulla voidaan helpottaa selventämään UV-karttaa. Tietyn tekstuurin vaativat osat voidaan värittää esimerkiksi samalla värillä. Kuvassa 15 näkyy automalli, jonka materiaaleille olen määrittelyt eri värit sen mukaan, minkä tyyppisiä tekstuureita tulen siihen myöhemmin lisäämään. Värin valinta tapahtuu painamalle hiiren vasemmalla painikkeella väriä ja valitsemalla sopiva väri. Diffuse-paneelin alta löytyy myös **Roughness-**säädin. Tämän avulla voidaan säätää pinnan pientä epätasaisuutta. Jos vaikkapa halutaan puun kuori näyttämään hieman karhealta, niin se onnistuu tämän säätimen kautta.

Kuva 14. Materiaalien Diffuse -arvojen määritysvalikko.

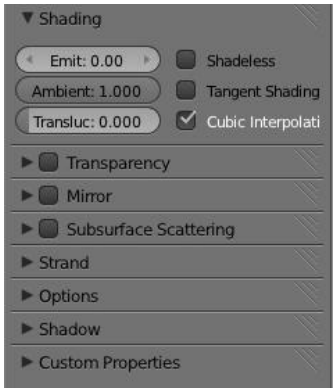


Kuva 15. Automallin tietyt osa-alueet on määritelty eri väreillä.

Seuraava paneeli on **Specular** eli peilimäinen (kuva 16). Asetukset vaikuttavat materiaalin tapaan heijastaa kirkkaimpia kohtia ja toisin kuin Diffuse-varjostus, Specular-varjostus on riippuvainen katsomiskulmasta, josta objektia tarkastellaan (Blender Wiki 2013b). Tämän paneelin alta löytyy useita samoja toimintoja, kuin jotka ovat Diffuse-paneelissakin. Värin valinta Specular-paneelista vaikuttaa siihen, minkä väristä valoa materiaali heijastaa takaisin. Hess (2010, 170) mainitsee, että yleensä heijastavan värin tulee olla valkoinen, mutta mikäli kyseessä on värillinen metalli, valitun värin on vastattava sitä väriä, joka on asetettu Diffuse-arvolle. Myös Specular-valikon kautta täytyy valita sopiva varjostin. **Blinn** sopii useille erilaisille pinnoille, **WardIso** toimii parhaiten metalli- ja muovipintojen kanssa, **Phong** sekä **CookTorr** puolestaan kiiltävissä metallipinnoissa. **Toon**-varjostin vastaa Diffuse-paneelin varjostinta ja toimii parhaiten, kun tavoitellaan sarjakuvamaista tyyliä (Blender Wiki 2013b). **Hardness-**säätimellä määritetään se, miten paljon pinnan kirkkaimpia kohtia heijastetaan. Yleensä tähän kannattaa valita melko maltillinen arvo, ettei materiaalista tule liian kirkas.

Kuva 16. Materiaalien Specular-arvojen määritysvalikko.

Materiaalin alta löytyy myös useita muita paneeleja (kuva 17), mutta etenkään yksinkertaisten 3D-mallien luomisessa niitä ei tarvita. Mainitsemisen arvoinen paneeli on kuitenkin **Transparency**, jonka avulla voidaan tehdä materiaalista läpinäkyvä. Tätä tarvitaan esimerkiksi lasimateriaalien kanssa. Hyödynsin tätä asetusta automallin lasipinnoissa. Hess (2010, 170) mainitsee Shading-paneelin alta löytyvästä asetuksesta nimeltään **Cubic Interpolation**, joka tulisi olla hänen mielestään päällä jokaisessa materiaalissa. Asetus ei ole oletuksena päällä. Aktivoimalla tämä asetus saadaan materiaalissa olevat siirtymät tummista alueista valoisiin näyttämään luonnollisemmilta. Omien kokeilujen perusteella olen huomannut tämän neuvon toimivaksi ja tällöin esimerkiksi varjot toistuvat tarkemmin.



Kuva 17. Muut materiaalien määritysvalikot.

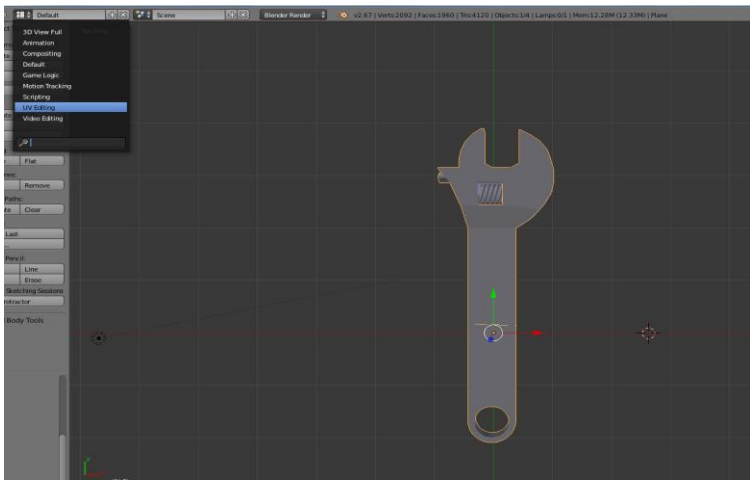
Vaikka materiaalien asetuksia voidaankin säätää hyvin tarkasti, niin on syytä muistaa se, mihin käyttöön muokattava 3D-malli on tulossa. Mikäli kyseessä on vaikka hyvin pieni esine, jolla on pieni rooli varsinaisessa pelissä, esimerkiksi jokin dekoraatio, on turha käyttää aikaa materiaalien tarkkaan säätämiseen. Materiaalien väriin ja heijastavuuteen voidaan vaikuttaa myös Unityssä sen jälkeen, kun 3D-malli on siirretty sinne. Omien kokemusten perusteella Blenderissä kannattaa asettaa materiaalin asetukset jonkinlaisiin ”perusarvoihin”, joita voidaan sitten tarpeen mukaan muokata Unityn puolella, kun nähdään miltä ne näyttävät varsinaisessa pelissä.

3.3 Tekstuurien lisääminen

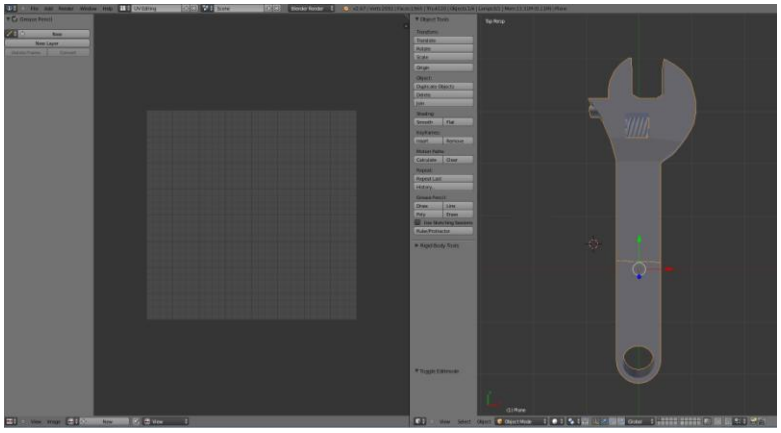
Lisäämällä materiaalit 3D-mallin pintoihin voidaan vaikuttaa siihen, miten eri pinnat ottavat vastaan valoa ja heijastavat sitä takaisin, mutta jotta pinnat saadaan näyttämään luonnollisemmilta, tarvitaan tekstuureita. Huonot tekstuurit voivat saada hyvin tehdyn 3D-mallin näyttämään epäonnistuneelta. Blender tarjoaa kaksi erilaista tapaa lisätä tekstuureita. Ensimmäinen tapa on proseduraalinen ja toisessa tekstuurit lisätään kuvien avulla. Proseduraalisessa tavassa Blenderin oma renderöijä tuottaa tekstuurit matemaattisten kaavojen avulla (Hess 2010, 171). Blender tarjoaa muutamia valmiita proseduraalisia tekstuurityylejä, mutta sellaisen voi myös toteuttaa itse. Tämä tapa vaatii kuitenkin vaivannäköä, että lopputuloksesta tulisi hyvännäköinen. Vieläkin tärkeämpi huomioitava asia on se, että jos ollaan viemässä valmista 3D-mallia Unityyn, proseduraalinen teksturointi ei tule kysymykseen, koska Unity ei ymmärrä sitä, eikä sitä saada vietyä oikein ohjelmasta toiseen. Unity vaatii, että 3D-mallin teksturointi on toteutettu kuvien avulla, tarkemmin sanottuna UV-mappauksella. UV-mappauksen avulla 3D-mallin tekstuureista luodaan kokonaan oma kuvatiedosto, josta Unity poimii siihen tuodun 3D-mallin tekstuurit. Blender tarjoaa kyllä mahdollisuuden muuntaa proseduraalisesti luodut tekstuurit UV-mapping-muotoon, mutta tämä on aikaa vievää ja mielestäni helpointa ja järkevintä onkin toteuttaa tekstuurit suoraan Unityn tukemaan muotoon.

3.3.1 UV-mappaus

Yksinkertaistettuna UV-mappauksessa kolmiulotteinen malli levitetään yhdeksi tai useammaksi kaksiulotteiseksi pinnaksi. Tämän jälkeen voidaan pintoihin lisätä sopivat tekstuurit, jotka sitten tulevat näkymään 3D-mallin päällä. (Blender-tutoriaali 2008.) Blenderissä on kokonaan oma näkymä UV-mappauksen tekemistä varten. Tähän näkymään pääsee joko vasemman reunan ylävalikon kautta (kuva 18) tai käyttämällä näppäinyhdistelmää Ctrl + nuolinäppäin oikealle tai vasemmalla, jolloin siirrytään eri näkymien välillä. Blender avaa nyt uuden näkymän perustilana toimivan 3D-näkymän vasemmalle puolelle (kuva 19).



Kuva 18. UV-näkymään siirtyminen perusnäkymästä.



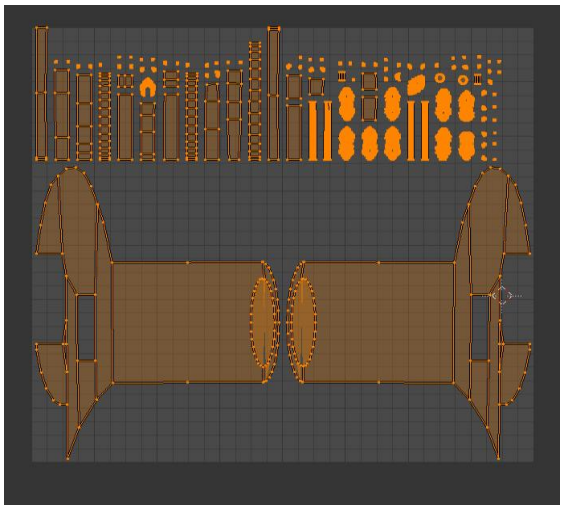
Kuva 19. UV-näkymä avattuna perusnäkymän rinnalle.

Seuraavaksi on kerrottava ohjelmalle se, miten UV-mappaus halutaan tehdä eli toisin sanoen on saatava tehtyä 3D-mallista 2D-kuva. Tähän tarvitaan **Unwrap**-työkalua eli vapaasti suomennettuna 3D-mallin pinnat on avattava kuin lahjapakkauksessa. Blender tarjoaa tähän muutamia erilaisia tapoja. Käyttäjä voi joko itse määritellä saumat (seams), joiden kohdalta mallin saa aukaista tai sitten voi käyttää erilaisia valmiita keinoja. Itse olen käyttänyt paljon Blenderin **Smart UV Project** -työkalua. Tämän avulla ohjelma itse määrittelee sopivat saumat ja avaa kuvan niiden perusteella. Toiminto sopii parhaiten yksinkertaisiin objekteihin sekä arkkitehtuurisiin malleihin, monimutkaisempien mallien kohdalla on syytä määritellä saumat manuaalisesti, jotta lopputuloksesta tulisi luonnollisemman näköinen

(Blender Wiki 2013c). Käyn seuraavaksi läpi tavat lisätä jakoavaimelle tekstuurit UV-mappauksen avulla käyttäen Smart UV Project -työkalua sekä auton renkaalle määrittelemälle saumat manuaalisesti. Sen jälkeen siirryn varsinaiseen tekstuurien lisäämiseen tähän UV-karttakuvaan.

3.3.2 UV-kartan tekeminen Smart UV Project -työkalulla

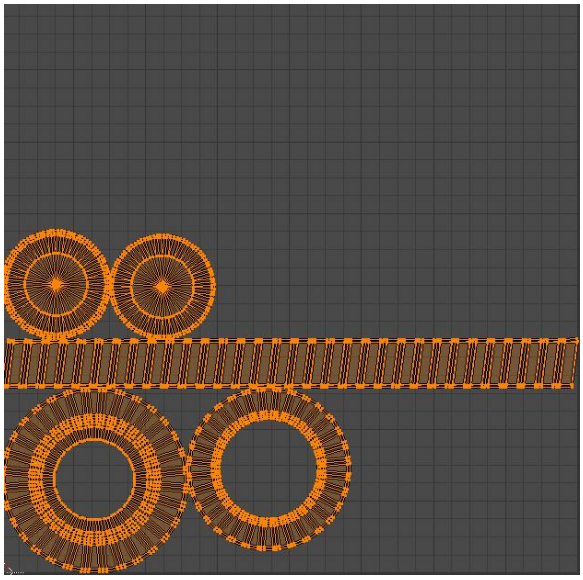
Aluksi siirrytään 3D-näkymässä **Edit-**tilaan, että päästään valitsemaan halutut pinnat UV-mappausta varten. Mikäli kyseessä on monimutkainen 3D-malli, voitaisiin eri kohdat mallista tehdä erillisten UV-karttojen avulla. Esimerkkinä toimiva jakoavain on kuitenkin sen verran yksinkertainen, että siihen riittää yksi UV-kartta. Valitsin kaikki objektin pinnat painamalla A-näppäintä ja sen jälkeen voin avata Unwrap-valikon. Tämä tapahtuu joko painamalla 3D-näkymässä U-näppäintä tai sitten valitsemalla Unwrap-painikkeen 3D-näkymän vasemmassa reunassa olevasta valikosta **UV Mapping-**kohdan alapuolelta. Avautuvasta valikosta valitaan Smart UV Project. Tämän jälkeen määritellään tarkemmat asetukset. **Angel limit** tarkoittaa sitä, millä tavoin eri pinnat ryhmitellään UV-karttaan. Suuri arvo tekee enemmän ryhmiä, mutta vähentää vääristymiä ja pienempi arvo toimii puolestaan päinvastoin. Jakoavaimessa annoin arvon olla oletuksena, joka on 66. **Island margin -**arvon avulla määritellään se, miten lähellä eri ryhmät ovat toisiaan UV-kartassa. Selvyyden vuoksi suosittelen kasvattamaan tätä arvoa oletuksesta, joka on 0. Tällöin lopputuloksena on selkeämpi UV-kartta, kuin jos eri ryhmät olisivat aivan kiinni toisissaan. Jakoavaimen kohdalla käytin arvoa 0.50. Viimeinen arvo on **Area weight**. Tämän avulla voidaan määritellä eri kohtien mittasuhteita ja tällä on vaikutusta etenkin silloin, jos 3D-mallista on suuria yhtenäisiä pintoja. Jakoavaimeen asetin arvoksi 1.00. Nämä kaikki kolme arvoa vaikuttavat toisin sanoen siihen, miten selkeä lopullinen UV-kartta on. Näitä arvoja kannattaa muuttaakin lähinnä silloin, jos tuloksena syntyvä UV-kartta vaikuttaa sekavalta. Kun kaikki kolme arvoa on valittu, voidaan **ok-**painiketta painamalla luoda UV-kartta (kuva 20).



Kuva 20. Smart Project UV-työkalulla luotu UV-kartta.

3.3.3 UV-kartan tekeminen Unwrap-työkalulla

Aluksi siirrytään Edit-työtilaan ja valitaan ne kohdat, joihin halutaan tehdä saumat UV-kartassa. Saumat kannattaa laittaa sellaisiin kohtiin, joissa niitä ei voi nähdä. Tällaisia kohtia ovat esimerkiksi reunat tai objektin alapuolelle jäävät osat. Valitaan 3D-mallista kohdat, joihin halutaan tehdä saumat ja merkitään ne **Mark seems** -työkalulla, joka löytyy työkalupalkin UV Mapping-kohdan alta. Auton renkaan kohdalla tein niin, että jaoin saumojen avulla renkaan osiin sen mukaan, millainen tekstuuri mihinkin tulee: Renkaan pintoihin oma tekstuuri, auton sisä- ja ulkopinnoille oma tekstuuri sekä vanteisiin oma tekstuuri. Sen jälkeen, kun ollaan tyytyväisiä valittuihin saumoihin, tehdään UV-kartta Unwrap-työkalulla. Kuva 21 näyttää auton renkaasta syntyneen UV-kartan. Kaksi ylintä rengasta ovat vanteen tekstuureita varten, keskellä oleva vinoista suorakulmioista koostuva kohta on renkaan urapinnan tekstuuria varten ja alhaalla olevat kaksi ympyrää ovat renkaat ulko- ja sisäpintojen kumitekstuureita varten. Mikäli UV-kartasta tulee sekava, voi kokeilla vaihtaa saumojen paikkaa poistamalla ensin vanhat saumat **Clear Seam** -toiminnon avulla. Sopivat saumojen paikat vaihtelevat hyvin paljon mallin mukaan, joten kokeilun ja kokemuksen kautta oppii parhaiten.



Kuva 21. Unwrap-työkalulla luotu UV-kartta auton renkaasta.

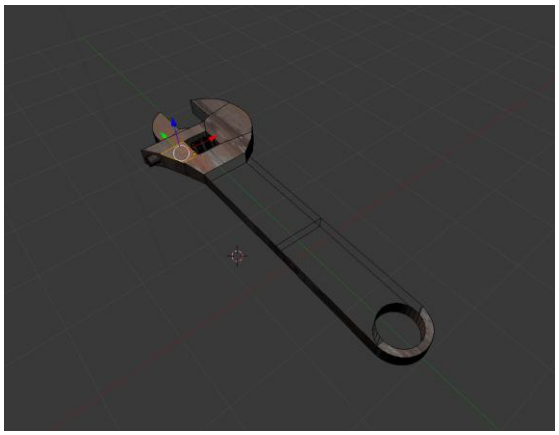
Blender tarjoaa myös muita tapoja toteuttaa UV-kartan 3D-mallista. Unwrap-valikosta löytyy vaihtoehdot tuottaa kuva ympyrän, neliön tai lieriön muotoisista objekteista. 3D-mallin muoto on suurin tekijä, kun mietitään miten UV-kartta toteutetaan. Wreck Garage -pelin 3D-malleja teksturoidessa olen selvinnyt hyvin pelkästään käyttämällä Smart UV Project- sekä Unwrap-työkaluja. Tärkeintä on löytää keinot, joilla pystyy toteuttamaan selkeitä UV-karttoja, joiden päälle on helppo rakentaa tekstuurit kuvankäsittelyohjelmassa. Suuremmissa pelialan yrityksissä on erikseen henkilöt, jotka toteuttavat tekstuurit 3D-mallien päälle, mutta pienemmissä yrityksissä tähän ei yleensä ole mahdollisuutta. Monesti 3D-mallintaja toimii myös tekstuurien tekijänä. Selkeiden UV-karttojen avulla työn tekeminen tehostuu ja nopeutuu. Tästä syystä on tarpeen miettiä miten UV-kartat on rakennettu.

3.3.4 UV-kartan muuntaminen kuvatiedostoksi ja tekstuurin lisääminen

UV-näkymässä oleva UV-kartta muutetaan kuvatiedostoksi valitsemalla alareunasta **UVs**-painike ja sieltä **Export UV Layout**. Tämän jälkeen annetaan tiedostolle nimi ja tallennetaan se haluttuun paikkaan. Tiedosto tallennetaan oletuksena PNG-muodossa, mutta myös vaihtoehtoisia tallennusmuotoja on tarjolla. PNG toimii kui-

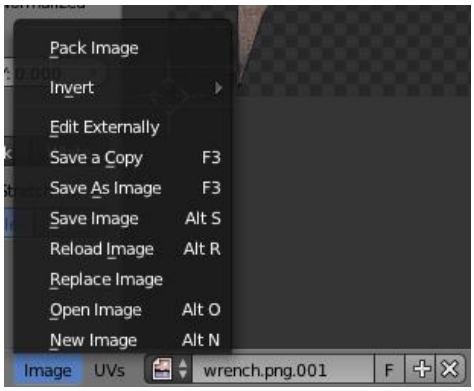
tenkin hyvin myös Unityn kanssa, joten sitä ei ole syytä muuttaa. Tämän jälkeen siirrytään kuvankäsittelyohjelmaan ja lisätään luotuun kuvatiedostoon halutut tekstuurit. Sen jälkeen, kun kuvaan on lisätty halutut tekstuurit, se täytyy ottaa käyttöön Blenderissä. UV-näkymän alareunasta valitaan **Image-**painike ja **Open image**. Etsitään muokattu UV-kuvatiedosto ja avataan se.

Tämän jälkeen 3D-mallin tekstuurit on aktivoitu ja Blender osaa yhdistää kuvatiedoston 3D-malliin. Tekstuureita voidaan tarkastella 3D-näkymässä, kun valitaan **Viewport shading** -asetuksista **Texture**. Joskus 3D-mallissa olevat facet eli pinnat saattavat osoittaa väärään suuntaan eli 3D-mallin sisäpuolelle. Tekstuurit luodaan aina pintojen toiselle puolelle, toinen puoli jää tyhjäksi. Tästä johtuen tällaisiin pintoihin lisätty tekstuuri on läpinäkyvä toiselta puolelta katsottuna. Kuva 22 näyttää esimerkin tällaisesta tapauksessa jakoavaimen kohdalla. Tällöin pinta on käännettävä. Tämä voidaan tehdä kahdella tavalla. Joko väärinpäin kääntynyt pinta valitaan hiirellä ja painetaan W-näppäintä, joka avaa **Specials**-valikon. Tästä valikosta valitaan toiminto **Flip Normals**. Toinen vaihtoehto on valita koko objekti ja painaa näppäinyhdistelmää Ctrl+N. Tämä toiminto laskee uudelleen valitut pinnat ja kääntää ne osoittamaan ulospäin (Blender Wiki 2013d). Nyt tekstuurit näkyy oikein ja 3D-malli on valmis vietäväksi Unityyn. Mikäli kyseessä on sellainen pinta, joka on hyvin ohut, esimerkiksi paperi tai muu hyvin kapea esine, voidaan kopioida pinta (**Duplicate**) ja kääntää kopioidun pinta osoittamaan toiseen suuntaan. Tämän avulla tekstuuri saadaan näkymään molemmilla puolilla pintaa.



Kuva 22. Yksi jakoavaimen pinnoista osoittaa väärään suuntaan.

Mikäli valittu tekstuuri halutaan korvata tai poistaa kokonaan, se onnistuu UV-näkymässä (kuva 23). Aluksi 3D-näkymässä siirrytään Edit-työtilaan ja valitaan ne kohdat 3D-mallista, joista tekstuurit halutaan korvata tai poistaa. Poistaminen tapahtuu painamalla kuvatiedoston nimen oikealla puolella olevaa valkoista rastia. Tällöin Blender poistaa 3D-malliin linkitetyn kuvan. Valitun tekstuuritiedoston korvaaminen tapahtuu painamalla Image-painiketta ja valitsemalla joko **Replace Image**- (korvaa kuvatiedoston), **Reload Image**- (lataa uudestaan valitun kuvatiedoston) tai Open Image (avaa kuvan) -toiminnon.

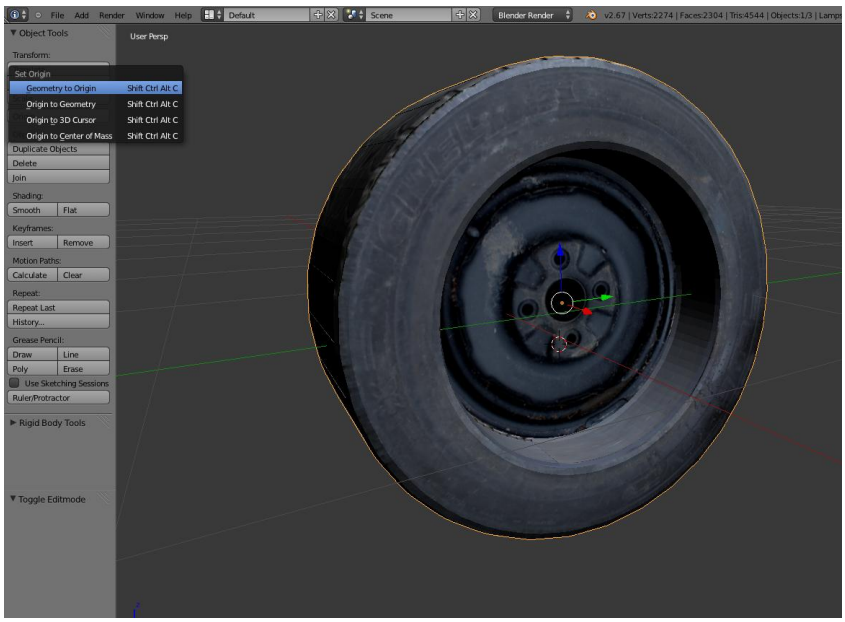


Kuva 23. UV-kuvatiedoston poistaminen ja korvaaminen.

3.4 Objektin keskipisteen määrittäminen

Mikäli ollaan luomassa Blenderissä 3D-mallia, jonka on tarkoitus liikkua tai esimerkiksi pyöriä, on otettava huomioon se, että objektin pivot point eli nollapiste oli määritelty oikein. Nollapiste kertoo Unitylle, missä objektin keskikohta sijaitsee. Jos kysymyksessä on vaikkapa rengas, jota on tarkoitus pyörittää keskipisteensä ympäri ja keskipiste on määritelty väärin, niin tästä seuraa luonnollisesti ongelmia. Keskipiste voidaan kyllä määritellä uudestaan Unityssä luomalla tyhjä objekti, siirtämällä tämä renkaan keskikohtaan ja määrittelemällä varsinaisen renkaan tämän tyhjän objektin lapsiobjektiksi (child). Tämä vaatii kuitenkin ylimääräistä työtä, joten nollapiste kannattaa määritellä oikein jo Blenderissä. Nollapisteen pystyy selvittämään Blenderissä, kun valitsee Object-tilan ja valitsee objektin hiiren oikealla painikkeella. Tällöin x-y-z-akseleita esittävä kursori siirtyy objektin nollapisteeseen.

Kursorin sijainnista voi nyt päätellä, onko keskipiste määritelty oikein. Kuvassa 24 on auton rengas, jonka nollapiste on määritelty oikein renkaan keskelle. Mikäli keskipiste on väärässä paikassa, se voidaan muuttaa valitsemalla Object Tools-valikon alta **Origin**-toiminto. Avautuvasta valikosta valitaan **Geometry to Origin**. Tämä siirtää nollapisteen objektin keskelle. Vaihtoehtoisesti nollapiste voidaan määritellä 3D-kursorin avulla. 3D-kursori sijoitetaan haluttuun kohtaan perusnäkymässä hiiren vasemmalla painikkeella ja sen jälkeen valitaan Origin-valikosta **Origin to 3D-cursor**. Jälkimmäistä käytetään silloin, kun nollapiste halutaan määrittää johonkin muuhun kohtaa kuin objektin keskelle.



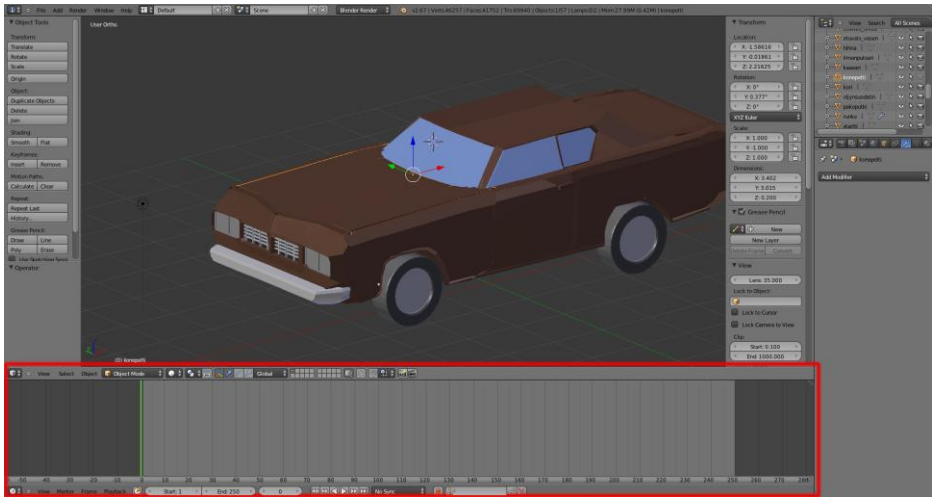
Kuva 24. Auton renkaan nollapiste ja sen määrittäminen Origin-valikon kautta.

3.5 Animointi Blenderin työkaluilla

Unity tukee myös Blenderissä toteutettuja 3D-mallien animaatioita. Blenderissä luotavat animaatiot voi jakaa kahteen ryhmään: yksinkertaisiin animaatioihin, joissa pelkästään liikutetaan, skaalataan tai käännetään objekteja tai niiden osia sekä riggauksen (engl. rigging), avulla toteutettuihin animaatioihin. Mikäli ollaan esimerkiksi toteuttamassa hahmoanimaatiota, niin tällöin hyödynnetään riggausta, jossa

pelihahmolle määritellään luuranko, jota liikuttelemalla sille voidaan luoda erilaisia animaatioita. Huomioitavaa on myös se, että mikäli objektille halutaan luoda useita erilaisia animaatioita, käytetään riggausta. Käyn seuraavaksi läpi yksinkertaisen animaation luomisen, joka avaa auton konepellin sekä toteutan ihmishahmolle animaation riggauksen avulla. Molemmat animaatiot voidaan siirtää 3D-mallin mukana myös Unityyn hyödynnettäväksi pelissä.

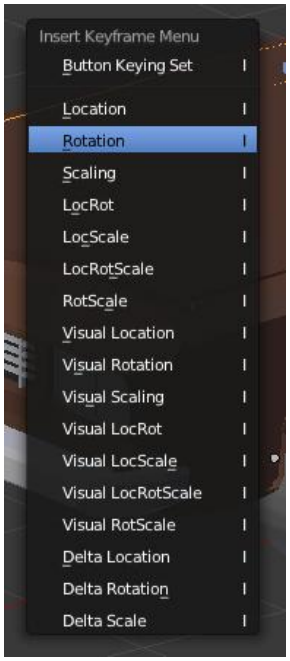
Yksinkertaiset animaatiot voidaan toteuttaa suoraan Blenderin perusnäkymässä. Perusnäkymän alareunasta löytyy **Timeline** eli aikajana-ikkuna, jota hyödynnetään animaatioiden luomisessa (kuva 25). Oletuksena Blender käyttää 24 ruudun (frames) animointia sekuntia kohden. Toisin sanoen 24 ruutua aikajanalla vastaa yhden sekunnin mittaista animaatiota. Aikajanan voi muuttaa myös näyttämään ajan sekunteina viemällä hiiren aikajanan päälle ja painamalla näppäinyhdistelmää **Ctrl + T**. (Hess 2010, 46 - 47.) Animaatiot toteutetaan ”keyframejen” avulla. Keyframe voidaan ymmärtää Blenderin keinona tallentaa objektista tietoja, kuten esimerkiksi objektin suunta, koko ja paikka. Kun lisätään kaksi keyframea eri kohtiin aikajanaa ja muutetaan toiseen keyframeen esimerkiksi objektin sijaintia, niin tällöin Blender toteuttaa animaation keyframejen välille. Se siirtää objektin sijainnin aloituspaikasta lopetuspaikkaan. Käyttäjän ei siis tarvitse itse siirtää objektia jokainen ruutu kerrallaan, vaan hänen tarvitsee vain määritellä nämä alku- ja loppupisteet eri animaatioille.



Kuva 25. Blenderin Timeline-ikkuna.

3.5.1 Yksinkertaisen animaation lisääminen 3D-mallille

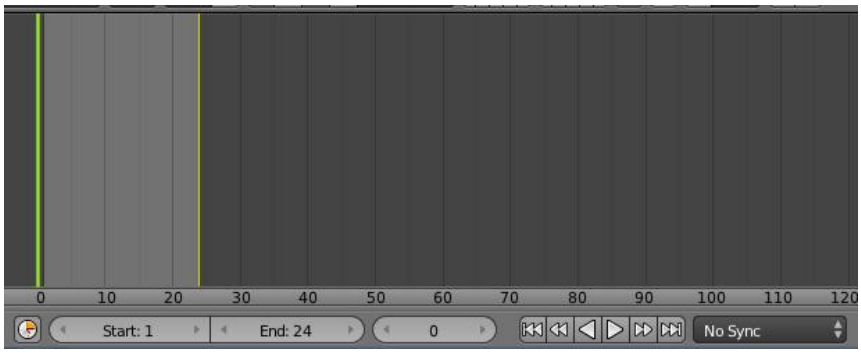
Animaation luominen perusnäkymässä aloitetaan valitsemalla animoitava objekti ja siirtämällä hiiren kursori 3D-näkymän päälle. Tämän jälkeen painetaan I-näppäintä, joka avaa **Keyframe Menun** (kuva 26). Tästä menusta valitaan se, mitä tietoja tallennetaan luotavaan keyframeen. Animaatioon, joka avaa konepellin, tarvitaan vain tieto siitä, mihin suuntaan objekti eli konepelti on käännettynä. Tässä tapauksessa valitaan valikosta **Rotation**-toiminto. Blender lisää uuden keyframen aikajanalle kohtaan 0. Keyframet näytetään aikajanalla keltaisina pystyviivoina. Keyframe Menussa tehtävä valinta riippuu siitä, miten objektia tullaan liikuttamaan. LocRot (sijainti, käännös), LocScale (sijainti, skaalaus), LocRotScale (sijainti, käännös, skaalaus), RotScale (käännös, skaalaus) -valintojen avulla voidaan tallentaa kerralla useita erityyppisiä muutoksia.



Kuva 26. Keyframe Menu avattuna.

Seuraavaksi on määriteltävä animaation lopetuspiste. Tässä vaiheessa täytyy miettiä, kuinka pitkään animaation haluaa kestävän. Haluan konepellin avaamisen kestävän noin sekunnin, joten klikkaan aikajanalta kohtaa 24 eli yksi sekunti. Seuraava keyframe lisätään tähän valittuun kohtaan. Siirrytään 3D-näkymään ja siirre-

tään animoitava objekti siihen asentoon, johon sen halutaan jäävän animaation lopussa. Kun halutut muutokset objektille on tehty, avataan jälleen Keyframe-menu ja tehdään valinta sen mukaan, mitä muutoksia objektille tehtiin. Uusi keyframe ilmestyy aikajanalle ja animaatio on valmis. Tämän jälkeen animaation toimivuutta voi testata aikajanan alapuolella olevien painikkeiden avulla
(kuva 27). Start- ja End-painikkeiden avulla voidaan määritellä animaation aloitus- ja lopetusruudut. Valikossa on myös kelaus- ja toistopainikkeet, joilla animaatiota voidaan esikatsella 3D-näkymässä.



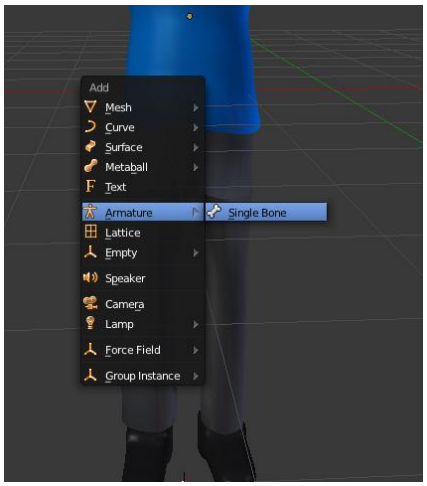
Kuva 27. Aikajanan toistopainikkeet animaatioita varten.

Yksinkertaisia animaatioita voi hyödyntää parhaiten sellaisiin objekteihin, joilla on melko pieni merkitys varsinaisessa pelissä tai joiden animaation ei tarvitse olla kovinkaan tarkkaa. Ennen kuin valitsee objektilleen sopivan animointitavan, tulisi miettiä objektin tarkoitusta. Riittääkö objektille yksi animaatio vai tarvitseeko se useita? Mikäli kyseessä on vaikkapa jonkinlainen dekoraatio-objekti, joka tekee jotain liikettä, kuten generaattori, sen toteuttamiseen voidaan käyttää yksinkertaista animaatiota. Tämä nopeuttaa animointiprosessia hieman, kun ei tarvitsee erikseen määritellä objektilla riggauksessa tarvittavia luita ja luurankoa, joita käsitellään seuraavassa luvussa. Yksinkertaisten animaatioiden tuominen Unityyn on myös helpompaa.

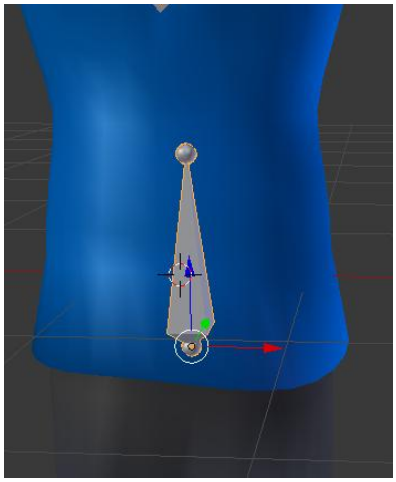
3.5.2 Hahmoanimaation lisääminen

Monimutkaisemmat animaatiot, kuten hahmomallien liikkeet, vaativat tarkemman tavan toteuttaa ja määritellä objektien liikkeitä. Tätä varten tarvitaan riggausta. Aluksi objektille luodaan luuranko, 3D-malli yhdistetään tähän luurankoon (skinning) ja tämän jälkeen luurankoa liikuttamalla 3D-mallille voidaan lisätä erilaisia animaatioita. Blenderissä luurankoa kutsutaan Armatureksi ja se koostuu useista luista (Bones). (Hess 2010, 247.)

Armaturen rakentaminen aloitetaan lisäämällä ensimmäinen luu. Tämä tapahtuu Object Modessa ja painamalla näppäinyhdistelmää Shift + A. Avautuvasta valikosta valitaan Armature ja sen jälkeen Single Bone (kuva 28). Tämä luo luun, josta Armaturea voidaan lähteä rakentamaan (kuva 29).



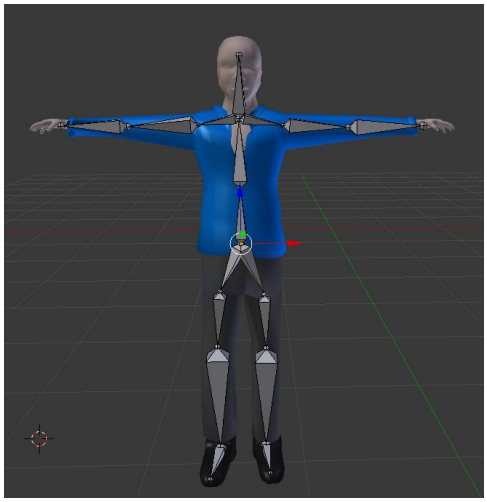
Kuva 28. Luun lisääminen 3D-mallille.



Kuva 29. Armaturen ensimmäinen luu.

Tämän jälkeen siirrytään Edit Modeen ja luita pystytään lisäämään, siirtämään ja skaalaamaan samalla tavalla, kuin mikäli muokattaisiin varsinaista 3D-mallia. Luiden liitoskohdat rakennetaan niin, että ne vastaavat hahmossa niitä kohtia, joissa on niveliä. Toteutettavan pelin vaatimuksista riippuu, miten tarkkaan luuranko tulee rakentaa. Onko esimerkiksi tarvetta luoda luut jokaiselle sormelle erikseen vai riittääkö vain yksi luu kättä varten? Tämä riippuu pitkälle pelin tarpeista. Wreck Gara-

ge -pelin ihmishahmolle riittää yksinkertaisempi Armature, jossa määritelty vain suurimmat luut, joiden hahmoa voidaan animoida (kuva 30).

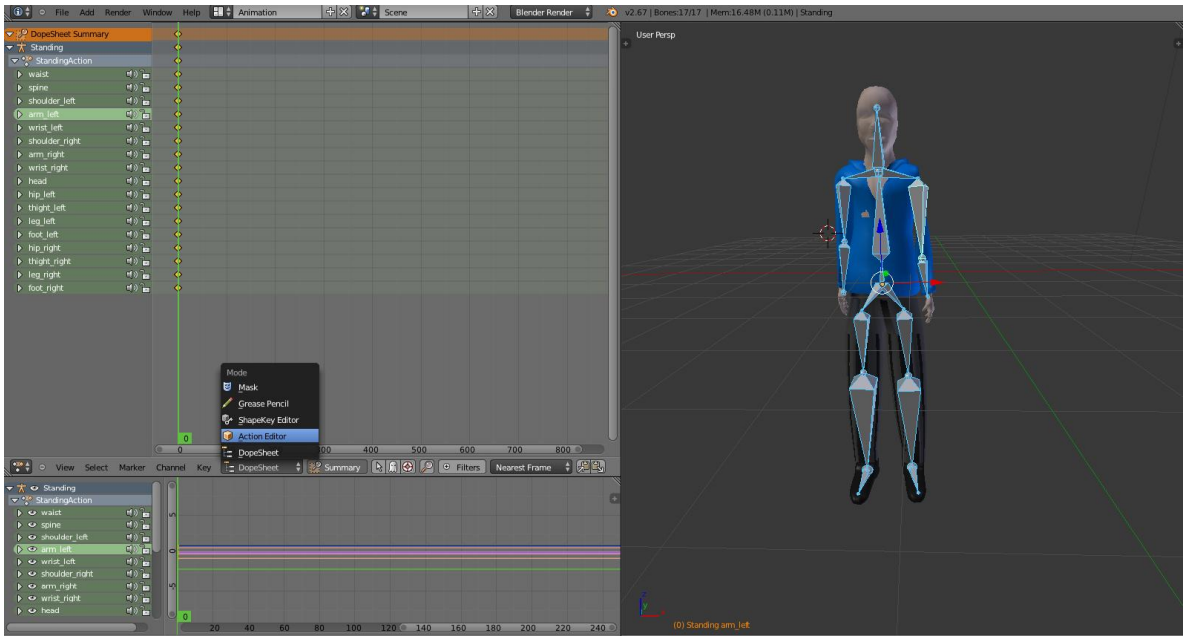


Kuva 30. Hahmomallille rakennettu Armature.

Tämän jälkeen 3D-malli on yhdistettävä Armatureen. Siirrytään Object Modeen ja valitaan ensin 3D-malli sekä sen jälkeen painetaan Shift-painike pohjaan ja valitaan Armature. Painamalla näppäinyhdistelmää Ctrl + P avautuu valikko, josta valitaan **Armature Deform: With Automatic Weights**. Tämän jälkeen voidaan siirtyä Pose Modeen. Mikäli Armaturen luita nyt liikutetaan, liikkuu myös varsinainen 3D-malli niiden mukana. Toteutan seuraavaksi hahmolle animaation, jossa hahmo seisoo paikoillaan. Tätä animaatiota käytetään silloin, kun hahmo ei tee pelissä mitään.

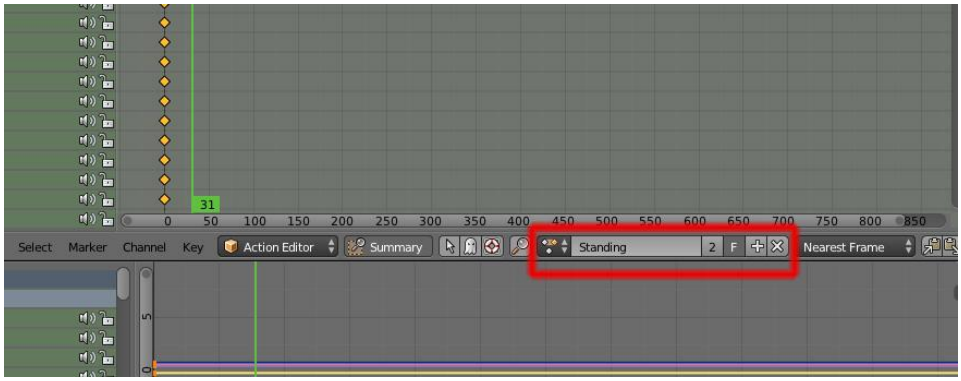
Animaatioiden teko Armaturen avulla noudattaa samaa kaavaa, jota käytettiin yksinkertaisten animaatioiden teossa. Erona on, että tällä kertaa liikutetaan luita, ei varsinaisia objekteja. Rigatun objektin animointiin on Blenderissä oma näkymänsä, joka löytyy näkymävalikosta **Animation**-nimellä. Tämä näkymä on vielä jaettu erilaisiin moodeihin (mode), joista Unityyn vietävien peliobjektien kohdalla tarvitaan kahta: **DopeSheetia** sekä **Action Editoria**. Moodia voi vaihtaa oletuksena olevan DopeSheet-näkymän alareunassa olevasta painikkeesta (kuva 31). DopeSheet-näkymä esittää kaikki 3D-mallille asetetut keyframet. Keyframejen asettaminen ja

animointi tapahtuu liikuttamalla luita perusnäkymän Pose Mode -tilassa. Tämän jälkeen luodaan uudet keyframet I-näppäintä painamalla ja tekemällä sopiva valinta sen mukaan, miten luuta on liikutettu ja mitä tietoja keyframeen halutaan tallentaa. Hahmon seisonta-animaatioita varten liikutin hahmon käsien luut niin, että hahmot kädet osoittaa alaspäin. Tämän jälkeen tallensin keyframen LocRotScale-asetuksella.



Kuva 31. Blenderin Animation-näkymän moodin valinta.

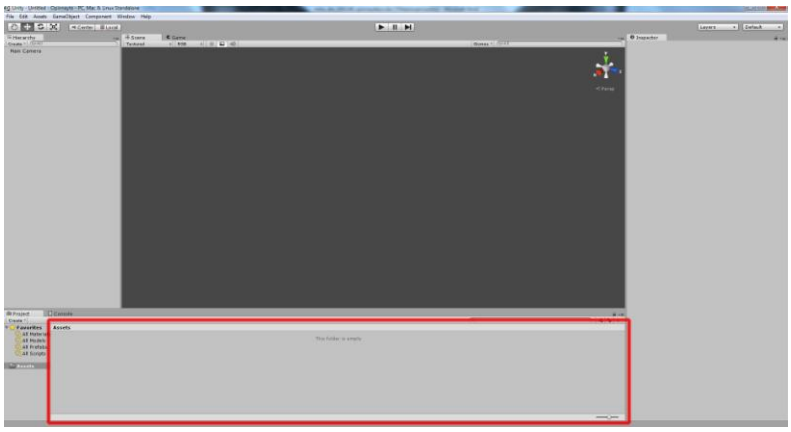
Seuraavaksi on luotava uusi toiminto animaatiota varten. Näiden toimintojen (actions) avulla Unity ymmärtää, että kyseessä on erilliset animaatiot. Siirrytään Action Editor -näkymään. Näkymä muistuttaa hyvin pitkälle DopeSheet-näkymää, mutta erona on, että tässä näkymässä näytetään vain ne keyframet, jotka koskevat perusnäkymässä valittuna olevaa objektia. Toiminnon nimeäminen tapahtuu samasta palkista, jossa on näkymänvalinta-painike (kuva 32). Tähän kirjoitettu nimi tulee näkyviin Unityssä omana animaationa. Toisin sanoen Unity ymmärtää Blenderissä määritellyt toiminnot omina animaatioinaan. Sen jälkeen, kun nimi on annettu, painetaan tekstilaatikon oikealla puolella olevaa F-painiketta. Tämän painikkeen avulla valitun objektin keyframet linkitetään tähän toimintoon. Tämän jälkeen 3D-malli voidaan tallentaa ja siirtää Unityyn.



Kuva 32. Animaation nimeäminen Blenderin Action Editor -modessa.

3.6 3d-mallin siirtäminen Unityyn

Kaikki Unityssä luodut projektit tekevät oman **Assets**-tiedostokansion varsinaisen projektikansion alle (kuva 33). Tähän Assets-kansioon kerätään kaikki kyseiseen projektiin liittyvät resurssit, kuten 3D-mallit. (Unity Technologies 2013e.) Assets-kansion sisällä on yleensä vielä muita alakansioita erityyppisille tiedostoille, jotta projektinhallinta olisi helpompaa ja selkeämpää. Tarpeen mukaan uusia alakansioita kannattaa luoda itse ja järjestellä erilaiset resurssit niihin. Aina kun projektiin importoidaan uusi resurssi, tämä resurssi siirtyy automaattisesti Assets-kansioon, ellei käyttäjä siirrä sitä itse eri kansioon.

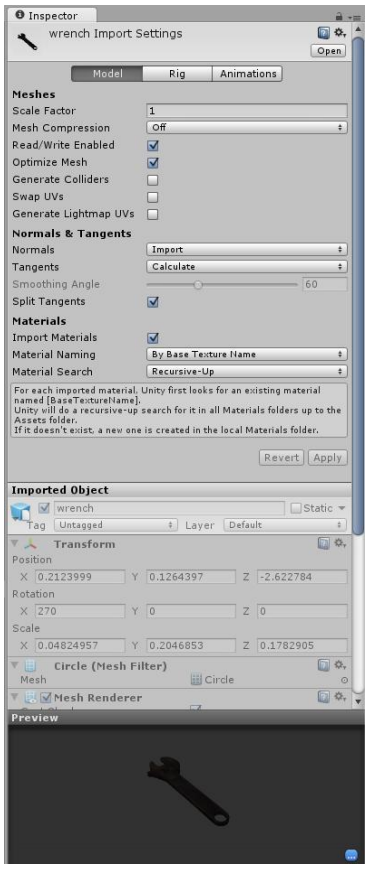


Kuva 33. Unityn Assets-kansion näkymä.

Importtaus voidaan tehdä kahdella eri tavalla. Aluksi käynnistetään Unity ja haluttu projekti. Tämän jälkeen siirrytään pois Unitystä ja avataan tiedostokansio, missä siirrettävä tiedosto sijaitsee. Tiedostoon tartutaan hiiren vasemmalla painikkeella ja se raahataan tehtäväpalkissa näkyvän Unity-painikkeen päälle, tällöin Unity aktivoituu ja käyttäjä voi pudottaa tiedoston joko Assets-kansioon tai johonkin sen alapuoliseen kansioon. Toinen tapa on tehdä importtaus suoraan Unitystä. Se onnistuu joko valitsemalla ylhäältä valikkopalkista **Assets** ja **Import New Asset**… Tämän jälkeen etsitään haluttu tiedosto, valitaan se ja klikataan **Import**-painiketta.
Import New Asset… -toiminto löytyy myös, kun painaa oikeaa hiirennappia Assets-näkymän päällä. Molemmat tavat toimivat samalla tavalla ja lopputuloksena on Unityyn importoitu resurssi, joka voidaan ottaa käyttöön varsinaisessa projektissa.

3.6.1 Import-asetuksien muuttaminen

Tarvittaessa jokaisen importoivan resurssin asetuksia voidaan muuttaa Unitystä. Tämä tapahtuu sen jälkeen, kun resurssi on ensin siirretty projektin Assets-kansioon. Resurssi valitaan vasemmalla hiiren napilla ja tämän jälkeen Inspector-ikkunaan avautuu **Import Settings**-valikko (kuva 34). 3D-mallien kohdalla valikko on jaettu kolmeen välilehteen: **Model**, **Rig** ja **Animations**. Kaksi jälkimmäistä kuuluvat 3D-malliin liitettyjen animaatioiden toimintaan ja ne käsitellään seuraavissa luvuissa.



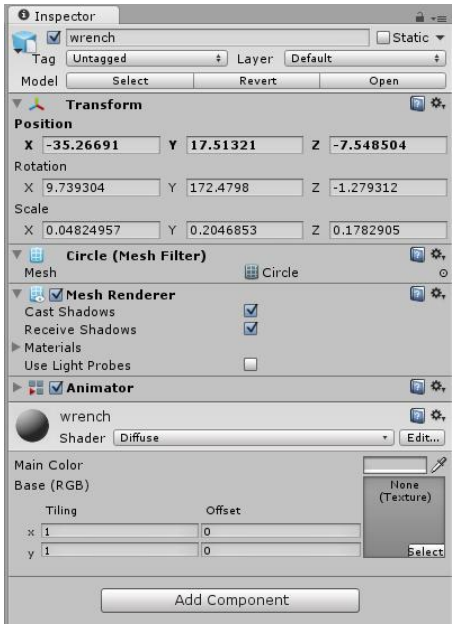
Kuva 34. Unityn Import Settings-valikko.

Varsinaiset 3D-mallin asetukset löytyvät ensimmäiseltä, Model-välilehdeltä. Kaikista asetuksista löytyy kattavat selitykset Unityn omilta sivuilta (http://docs.unity3d.com/Documentation/Components/FBXImporter-Model.html), joten tässä ei käydä läpi tarkemmin kaikkia kohtia. Muutama on kuitenkin hyvä mainita, mikäli tuodaan 3D-mallia Blenderistä. **Scale Factor** -asetus vaikuttaa siihen, minkä kokoisena 3D-malli tuodaan Unityyn. Oletuksena valittuna on 1, joka tarkoittaa objektin alkuperäistä kokoa lähdetiedostossa. **Materials**-tekstin alta löytyvät asetukset vaikuttavat siihen, miten 3D-malliin lisättyjä materiaaleja käsitellään. Mikäli materiaaleja ei haluta tuoda ollenkaan 3D-mallin mukana, voidaan poistaa täppä **Import Materials** -valinnasta. Oletuksena tämä on aina päällä. Material Naming -asetuksen muuttaminen puolestaan vaikuttaa siihen, miten materiaalit nimetään. Oletuksena valittuna on **By Base Texture Name**, jolloin etsitään objektiin liitettyjä tekstuureita ja ellei niitä löydy, niin materiaalit nimetään löytyvien materiaalien mukaan. Muut vaihtoehdot ovat materiaalien nimeäminen objektin materiaalien mukaan (**From Model's Material**) sekä mallin nimen sekä siinä olevi-

en materiaalien nimen yhdistäminen (**Model Name + Model's Material**). (Unity Technologies 2013f.) Mikäli 3D-malliin määriteltyjä materiaaleja halutaan käyttää, ne kannattaa nimetä selkeästi jo Blenderissä, jolloin niitä on helpompi käsitellä myös Unityssä. Kaikki tehdyt muutokset Import Settings -valikossa täytyy hyväksyä **Apply**-painikkeen avulla, muuten ne eivät päivity importoituun resurssiin.

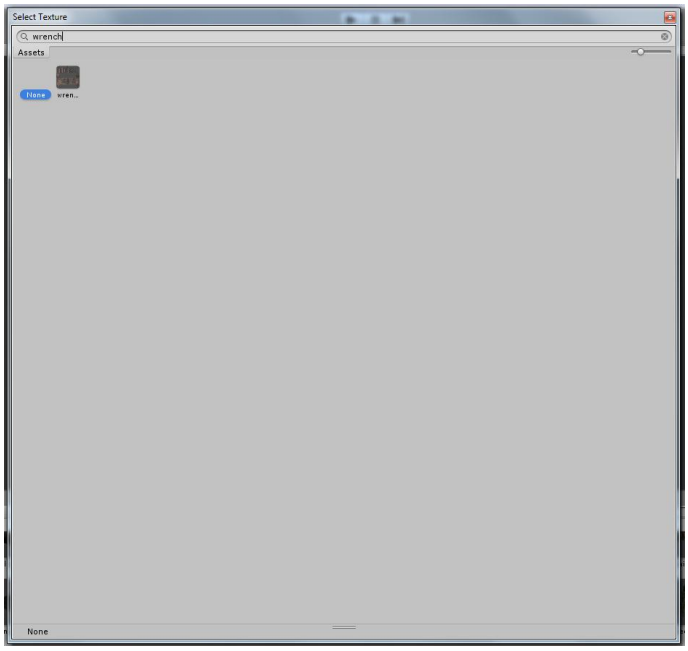
3.6.2 Tekstuurien toimivuus

3D-mallin tekstuurit eivät siirry suoraan mallin mukana Unityyn, vaan ne on tuotava erikseen. Tekstuurien importointi tapahtuu samalla tavalla kuin muiden tuotavien resurssien. Tekstuuritiedostot voidaan ottaa käyttöön, kun ne löytyvät projektin Assets-kansiosta. Järkevää ja projektinhallinnan kannalta selkeämpää on kuitenkin tehdä tekstuureille oma alikansio Assets-kansion alle ja suuremmissa projekteissa vielä omat alikansiot erityyppisille tekstuureille. Ellei Unity lisää tekstuuria automaattisesti 3D-mallille se täytyy tehdä manuaalisesti. Tämä tapahtuu valitsemalla ensin 3D-mallin Assets-kansiosta ja klikkaamalla sitä vasemmalla hiiren painikkeella. Resurssin tiedot avautuu nyt Inspector-näkymään (kuva 35). Alimpana ovat tekstuureihin liittyvät asetukset.



Kuva 35. Resurssin asetukset Inspector-ikkunassa.

Mikäli 3D-malliin oli määritelty useita materiaaleja, jokaiselle voitaisiin määritellä tässä oma tekstuuritiedosto. Jakoavaimelle määriteltiin kuitenkin vain yksi materiaali. Tekstuuri voidaan lisätä joko raahaamalla se suoraan Assets-ikkunasta harmaan tekstuurilaatikon päälle. Vaihtoehtoinen tapa on painaa tekstuurilaatikossa olevaa **Select**-painiketta, jolloin pääsee valitsemaan suoraan kaikista projektin tekstuureista (kuva 36). Mikäli tekstuureita on paljon, voidaan hakutuloksia rajata kirjoittamalla etsittävän tekstuurin nimi yläreunassa olevaan tekstilaatikkoon. Haluttua tekstuuria klikataan kaksi kertaa, jolloin valinta hyväksytään ja tekstuuri otetaan käyttöön 3D-mallissa.

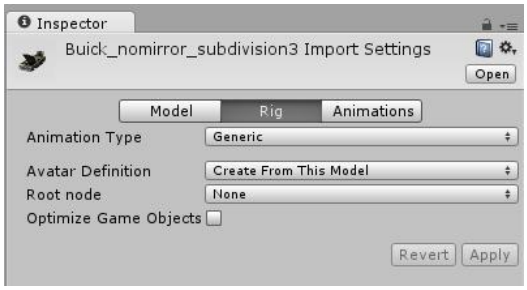


Kuva 36. Tekstuurien valintaikkuna.

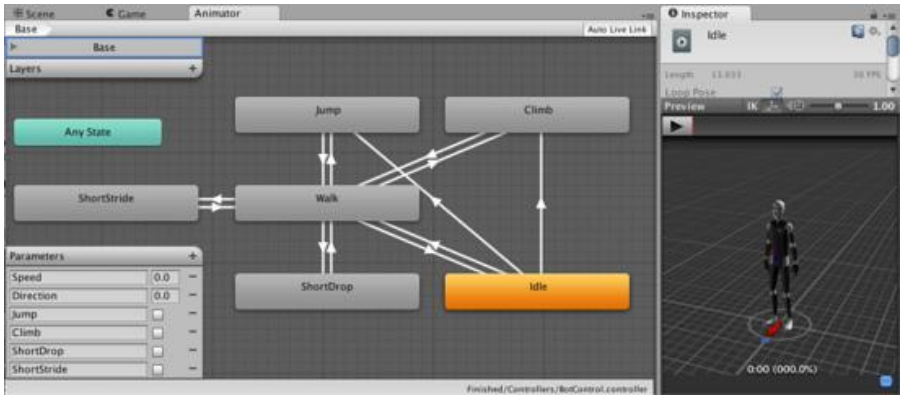
3.6.3 Animaatioiden toimivuus

Animoituja objekteja Unityyn importoitaessa on tarkastettava Import Settings -valikon kautta Rig ja Animation-välilehtien asetukset. Rig-välilehdeltä (kuva 37) määritellään minkätyyppistä ”runkoa” animointiin käytetään. Oletuksena on valittuna **Generic**-asetus. Omien kokemusteni mukaan Generic-asetus toimii huonosti

Blenderistä tuotujen animaatiota sisältävien objektien kanssa. Olenkin käyttänyt itse **Legacy**-asetusta. Legacy on vanhemmissa Unityn versioissa käytössä ollut animaatiojärjestelmä. Valittavana on myös Humanoid-asetus. Tätä tulisi käyttää ihmismäisten objektien (kaksi jalkaa, kaksi kättä ja pää) kohdalla, mikäli halutaan hyödyntää Unity 4.0 versiossa käyttöön otettua Mecanim-animaatiojärjestelmää (Unity Technologies 2013g). Mecanimin avulla voidaan rakentaa animaatioita ja luoda animaatioiden välisiä suhteita erityisen Visual Programming Tool -työkalun avulla (kuva 38) (Unity Technologies 2013h). Mecanim-järjestelmää voi hyödyntää parhaiten, mikäli toteutetaan animaatiot vasta Unityn puolella. Tällöin Blenderissä määritellään pelkästään armature ja luut 3D-mallille, joita hyödyntämällä toteutetaan animaatiot Unityssä Mecanim-järjestelmän avulla. Tässä opinnäytetyössä en käy tarkemmin läpi animaatioiden toteuttamista Unityssä Mecanim-järjestelmällä. Aiheeseen liittyen löytyy kattavasti tutoriaaleja Internetistä, esimerkiksi Unityn omilta sivuilta osoitteesta http://docs.unity3d.com/Documentation/Manual/MecanimAnimationSystem.html.

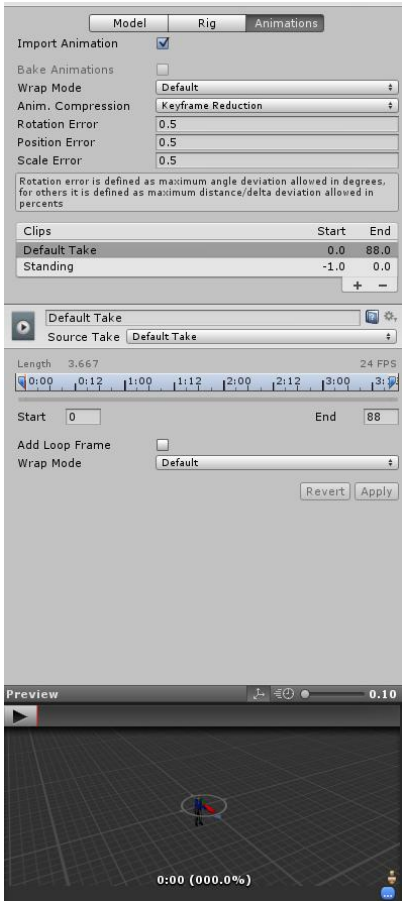


Kuva 37. Import Settings -valikon Rig-välilehti.



Kuva 38. Unityn Visual Programming Tool -työkalu (Unity Technologies 2013).

Animations -välilehdeltä (kuva 39) pystytään esikatselemaan 3D-mallissa olevia animaatioita ja testata niiden toimivuus. Animaatiot voidaan jättää kokonaan importoimatta poistamalla täppä **Import Animations** -valinnasta. Clips-kohdan alta löytyy kaikki 3D-mallissa olevat animaatiot eli Blenderissä määritellyt toiminnot. Animaatioita voi testata alareunan preview-näkymässä Toisto-painiketta painamalla. Wrap Mode -valinnasta voi määritellä sen, miten animaatiota haluaa toistettavan. Vaihtoehtoina on, että animaatio toistetaan kerran (Once), jatkuvasti (Loop), alusta loppuun ja sen jälkeen lopusta alkuun (Ping Pong) sekä alusta loppuun ja jäädään toistamaan viimeistä ruutua (Clamp Forever) (Unity Technologies 2013i).



Kuva 39. Import Settings -valikon Animations-välilehti.

4 Yhteenveto

Tämän työn tarkoituksena oli selvittää, miten hyvin ilmainen 3D-mallinnusohjelma Blender soveltuu yhteiskäyttöön Unity-pelimoottorin kanssa. Unityn kehittäjät ovat ottaneet huomioon sen, että osa käyttäjistä käyttää Blenderiä ohjelmistonsa kanssa ja he ovatkin lisänneet Unityyn tuen Blenderistä tuotaville 3D-malleille. Unityä myös käytetään pelimoottorina yhä enemmän pelialan yrityksissä, mutta Blenderiä puolestaan hyvin vähän. Kaupalliset ohjelmistot, kuten 3Ds Max ja Maya, ovat olleet markkinoilla kauemmin ja vakiinnuttaneet asemansa. Blender tulee perästä, mutta ominaisuuksien perusteella on jo ottanut kiinni kaupallisia ohjelmistoja. Merkittävä osa kaupallisten ohjelmistojen ominaisuuksista on mukana jo Blenderissä. Käyttöliittymä ja virallisen tuen puuttuminen ovatkin käytännössä ainoat tekijät, joissa Blender eroaa niistä. Rajallisella budjetilla työskenteleville pelialan yrityksille Blender on hyvä vaihtoehto. Blender sopii myös freelancereille, joilla ei välttämättä ole tarvetta tai halua hankkia kallista ohjelmistoa työhön, jota he tekevät epäsäännöllisesti. Blender taipuu kattavasti erityyppisten peliobjektien toteuttamiseen. Opinnäytetyön ohessa toteuttamaani Wreck Garage -peliin pystyin rakentamaan erityyppisiä objekteja Blenderillä, pelin tarpeen mukaan. Staattiset, liikkuvat, animoitavat ja orgaaniset objektit onnistuivat. Suurimmat haasteet tulivat eteen animoinnissa. Unity vaihtoi vanhan Legacy-animaatiojärjestelmän uuteen Mecanim-järjestelmään versiossa 4.0. Mecanim-järjestelmä ei ole vielä kehittynyt yhteensopivuudessa riittävän hyvälle tasolle Blenderin kanssa. Järkevintä onkin tällä hetkellä käyttää peliobjektien importoinnissa Unityyn vanhaa Legacy-järjestelmää. Omien kokeilujen perusteella se toimii erinomaisesti.

Työn ohessa toteuttamani Wreck Garage -autopeli on edennyt kohti ensimmäistä 0.1-versiota. Tähän mennessä olen käyttänyt aikaa etenkin 3D-mallien luomiseen, teksturointiin ja niiden siirtämiseen Unityyn. Olen myös jonkin verran ohjelmoinut pelin toimivuutta. Pelin kehittäminen jatkuu varmasti tulevaisuudessa. Tämä opinnäytetyö on toiminut opettavana kokemuksena minulle, koska ennen tätä minulla oli hyvin vähän tietoa Blenderistä ja Unitystä. Työstä saatujen tulosten perusteella

aion jatkaa työskentelyä näillä ohjelmistoilla. Molemmat ovat osoittautuneet toimiviksi ratkaisuiksi erilaisten 3D-mallien saamiseksi peliin. Tulosten perusteella uskonkin, että Blender ja Unity ovat yhdessä hyvä vaihtoehto pelejä tekeville ihmisille toteuttaa visioitaan ilman mittavia kustannuksia.

Lähteet

Blender.org. 2013. Tutorials. http://www.blender.org/support/tutorials/. 19.11.2013.

BlenderNation. 2012. Eat Sheep - iOS Game built with Blender and Unity 3D. http://www.blendernation.com/2012/08/13/eat-sheep-ios-game-built-with-blender-and-unity-3d/. 20.11.2013.

BlenderWiki. 2013a. Diffuse Shaders. http://wiki.blender.org/index.php/Doc:2.6/Manual/Materials/Properties/Diffuse_Shaders. 22.9.2013.

Blender Wiki. 2013b. Specular Shaders. http://wiki.blender.org/index.php/Doc:2.6/Manual/Materials/Properties/Specular_Shaders. 22.9.2013.

BlenderWiki. 2013c. Unwrapping a mesh. http://wiki.blender.org/index.php/Doc:2.6/Manual/Textures/Mapping/UV/Unwrapping. 30.9.2013.

BlenderWiki. 2013d. Face editing. http://wiki.blender.org/index.php/Doc:2.6/Manual/Modeling/Meshes/Editing/Faces. 13.11.2013.

Blender-tutoriaali. 2008. UV-mappaus. http://opendimension.org/blender/pages/tekstuurit/uv-mappaus.php?lang=EN. 30.9.2013

Hess, R. 2010. Blender Foundations. USA: Focal Press.

iTunes. 2013. Ginormous: https://itunes.apple.com/au/app/ginormous/id684172085?mt=8. 20.11.2013.

Marketwire. 2012. Unity 4.0 launches. http://www.marketwired.com/press-release/unity-40-launches-1726144.htm. 12.11.2013.

Siira, J. 2007. Avoimen lähdekoodin Blender vs. kaupallinen 3Ds Max. Lahden ammattikorkeakoulu. https://publications.theseus.fi/bitstream/handle/10024/11563/2008-03-19-04.pdf?sequence=1. 18.10.2013.

The CG Society. 2013. Comparison of 3d tools. http://wiki.cgsociety.org/index.php/Comparison_of_3d_tools. 26.11.2013.

Unity Technologies. 2013a. Licence Comparison. http://unity3d.com/unity/licenses. 25.11.2013.

Unity Technologies. 2013b. Game list. http://unity3d.com/gallery/made-with-unity/game-list. 14.11.2013.

Unity Technologies. 2013c. Unity Technologies Doubles Community to Two Million Developers. http://unity3d.com/company/public-relations/news/unity-technologies-doubles-community-two-million-developers. 29.11.2013.

Unity Technologies. 2013d. Importing objects from Blender. http://docs.unity3d.com/Documentation/Manual/HOWTO-ImportObjectBlender.html. 12.10.2013.

Unity Technologies. 2013e. Importing Assets. http://docs.unity3d.com/Documentation/Manual/ImportingAssets.html. 13.11.2013.

Unity Technologies. 2013f. Models. http://docs.unity3d.com/Documentation/Components/FBXImporter-Model.html. 15.11.2013.

Unity Technologies. 2013g. FBX Importer, Rig options. http://docs.unity3d.com/Documentation/Components/FBXImporter-Rig.html. 11.11.2013.

Unity Technologies. 2013h. Mecanim Animation System. http://docs.unity3d.com/Documentation/Manual/MecanimAnimationSystem.html. 22.11.2013.

Unity Technologies. 2013i. FBX Importer - Animations tab. http://docs.unity3d.com/Documentation/Components/FBXImporter-Animations.html. 15.11.2013.

WikiBooks. 2013. Blender 3D: Noob to Pro. http://en.wikibooks.org/wiki/Blender_3D:_Noob_to_Pro. 19.11.2013.

3D-mallinnusohjelmien ominaisuuksien vertailutaulukko

Taulukkoon 1 on otettu mukaan Blender sekä neljä maksullista 3D-mallinnusohjelmaa. Mukaan on otettu keskeisiä ominaisuuksia ja esitetty, mitkä ominaisuudet ovat mukana missäkin ohjelmassa (CG Society 2013).

Taulukko 1. 3D-mallinnusohjelmien ominaisuuksien vertailutaulukko (CG Society 2013).

	Blender	3Ds Max	Maya	Softimage / Softimage XSI	Lightwave 3D
Tuetut käyttöjärjes-telmät	Windows, Mac OSX, Linux,	Windows	Windows, Mac OSX, Linux	Windows, Linux	Windows, Mac OSX
Hinta	Ilmainen	3900 €	3900 €	3300 €	740 €
Mallintamistyökalut	Blender	3Ds Max	Maya	Softimage / Softimage XSI	Lightwave 3D
NURBS	Kyllä	Pluginin avulla	Kyllä	Kyllä	Ei
Patch	Ei	Kyllä	Kyllä	Ei	Ei
SubD	Kyllä	Kyllä	Kyllä	Kyllä	Kyllä
Polygon	Kyllä	Kyllä	Kyllä	Kyllä	Kyllä
Smooths 3gons	Kyllä	Kyllä	Kyllä	Kyllä	Kyllä
Smooth *n*gons	Kyllä	Kyllä	Kyllä	Kyllä	Kyllä
UV Unwrapping	Blender	3Ds Max	Maya	Softimage / Softimage XSI	Lightwave 3D
Basic	Kyllä	Kyllä	Kyllä	Kyllä	Kyllä
Pelt	Ei	Kyllä	Pluginin avulla	Pluginin avulla	Pluginin avulla
LSCM	Kyllä	Pluginin avulla	Pluginin avulla	Ei	Ei
ABF++	Kyllä	Ei	Ei	Ei	Ei
Multi UV Sets	Kyllä	Kyllä	Kyllä	Kyllä	Kyllä
Subsurfed UV	Kyllä	Kyllä	Ei tietoa	Kyllä	Kyllä
Teksturointi ja maalaus	Blender	3Ds Max	Maya	Softimage / Softimage XSI	Lightwave 3D
Node Bases Textu-ring	Kyllä	Kyllä	Kyllä	Kyllä	Kyllä

2D paint	Kyllä	Pluginin avulla	Kyllä	Kyllä	Ei
3D paint	Kyllä	Pluginin avulla	Kyllä	Ei	Ei
Vertex paint	Kyllä	Kyllä	Kyllä	Kyllä	Kyllä
Animointi ja simulointi	Blender	3Ds Max	Maya	Softimage / Softimage XSI	Lightwave 3D
Yksinkertainen (simple)	Kyllä	Kyllä	Kyllä	Kyllä	Kyllä
Hahmo (character)	Kyllä	Kyllä	Kyllä	Kyllä	Kyllä
Partikkelit (particle)	Kyllä	Kyllä	Kyllä	Kyllä	Kyllä
Fysiikat (physics)	Kyllä	Kyllä	Kyllä	Kyllä	Kyllä
Softbody	Kyllä	Kyllä	Kyllä	Kyllä	Kyllä
Vaatteet (cloth)	Kyllä	Kyllä	Kyllä	Kyllä	Kyllä
Hiukset (hair)	Kyllä	Kyllä	Kyllä	Kyllä	Pluginin avulla
Neste (fluid)	Kyllä	Pluginin avulla	Kyllä	Kyllä	Kyllä
Savu (smoke)	Kyllä	Kyllä	Kyllä	Kyllä	Pluginin avulla
Tuli (flame)	Kyllä	Pluginin avulla	Kyllä	Ei	Pluginin avulla
Yleisö (crowd)	Pluginin avulla	Kyllä	Pluginin avulla	Kyllä	Ei