TAMPEREEN
AMMATTIKORKEAKOULU

MOLEKYYLINMALLINNUSOHJELMA

Marko Schmidt

Opinnäytetyö
Marraskuu 2015
Tietotekniikan koulutusohjelma
Ohjelmistotekniikka

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tietotekniikan koulutusohjelma
Ohjelmistotekniikka

SCHMIDT MARKO:
Molekyylinmallinnusohjelma

Opinnäytetyö 24 sivua
Joulukuu 2015

Työn tavoitteena on suunnitella ja toteuttaa käyttökelpoinen molekyylinmallinnus-ohjelma, jota voisi mahdollisesti hyödyntää koulujen kemianopetuksessa tavanomaisen oppimisen ohella. Projektin idea lähti tekijän omasta mielenkiinnosta aiheeseen, ja käyttötarpeen selvittäminen sekä mahdollinen käyttöönotto on tarkoitus aloittaa vasta työn päättymisen jälkeen.

Työn etenemiseen riitti teorian kannalta lukiossa opitut kemian sekä pitkän matematiikan taidot, joita jouduttiin kuitenkin kertaamaan internetlähteitä hyödyntäen. Ohjelma toteutettiin Unity-pelimoottorilla, joka on suunniteltu erilaisten ohjelmien helppoon tuottamiseen ja jonka käyttöönottokynnys on suhteellisen matala.

Työn ohjelmointipuolen hankaluudesta johtuen sen jää kesken, joten jatkokehitystä on tehtävä, ennen kuin ohjelman käyttöä voidaan harkita opetuksessa.

Asiasanat: unity, molekyyli, mallinnus

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
ICT Engineering
Software engineering

SCHMIDT MARKO:
Molecule modelling application

Bachelor's thesis 24 pages
December 2015

The aim of the thesis was to design and implement a 3D molecule modelling application that could be used in schools to help visualize molecules alongside with the traditional teaching methods. It remains to be seen whether the program will actually see any use, as the marketing side of the project was intentionally left out of the thesis work and will be conducted on a later date.

For this thesis it was enough to know the basics of molecular chemistry and maths learned in high school, even though some rehearsing was necessary. The software was made using the Unity-engine, which is designed for easy developing of software. Its deployment has also been made simple, which helped in choosing it for this work.

Due to the difficulties encountered during the development process the application was not made to the point originally planned. Further development is however planned for it and is needed before it can be considered to be used for teaching in schools.

Key words: unity, molecule, modelling

SISÄLLYS

LYHENTEET JA TERMIT

API	application programming interface, määritelmä ohjelmien välistä keskustelua varten
asset	Unityyn liitettävissä oleva lisäohjelma
atomimalli	tapa kuvata atomin rakennetta
debugger	virheenetsintätyökalu
frame	näytönkuva, joiden sarjasta peli muodostuu
GUI	graafinen käyttöliittymä
pelimoottori	ohjelmarunko
porttaus	ohjelman toimimaan saaminen eri käyttöjärjestelmillä
prefab	valmiiksi luotu peliobjekti, jota voidaan kopioida koodissa uusien objektien luomiseen
refaktorointi	koodin uudelleenjärjestely
scriptaus	ohjelmakoodin kirjoitus
shader	varjostin, grafiikan nopeaan manipulointiin erikoistuva ohjelma

1 JOHDANTO

Internetistä löytyy nykyään monenlaista digitaalista oppimismateriaalia, Youtube-videoista ilmaisiin ja maksullisiin nettikursseihin. Näitä hyödyntämällä yhä useammat ihmiset voivat opiskella joko omaksi ilokseen tai koulutusmielessä lähes mitä vain. Muun maailman digitalisoituessa yksi merkittävä osa-alue on kuitenkin jäänyt tämän murroksen ulkopuolelle. Kouluissa hyödynnetään tietotekniikkaa vain hyvin vähän, vaikka mahdollisuuksia sille onkin tarjolla useimmissa oppilaitoksissa. Tämä käy ilmi Accenturen sekä Helsingin opetusviraston tekemässä selvityksessä (Jakovuori, Rauhala, & Saastamoinen 2014, 11). Vaikka joitain projekteja oikeaan suuntaan onkin aloitettu Suomessa , niiden kehitys on ollut hidasta ja kehitys on jäänyt jälkeen monen muun maan taakse (OECD 2009). Tässä opinnäytetyössä on tavoitteena luoda kouluille digitaalista oppimateriaalia, jonka tarkoituksena on mahdollistaa oppilaita kokeilemaan kirjoista luettuja asioita ja sitä kautta edesauttaa oppimista.

Tässä opinäytetyössä kehitetty molekyylinmallinnusohjelma toimii pilottina isommalle ohjelmalle, jota on tarkoitus lähteä kehittämään, jos kiinnostusta sille löytyy riittävästi. Työn pääpainona on kuitenkin suunnitella ja kehittää itse ohjelmaa, ja markkinointiin siirrytään vasta myöhemmin, kun ohjelman kehitys on edennyt pidemmälle. Työssä tutustutaan myös hieman orgaaniseen kemiaan, joka on opinnäytetyössä tehdyn ohjelman aihealueena. Lisäksi opinnäytetyössä esitellään Unity-pelimoottoria, jolla ohjelma toteutettiin. Unitystä kerrotaan nopeasti mikä sen on, miten se toimii ja miksi se on valittu tässä projektissa käytettäväksi työkaluksi.

2 ORGAANINEN KEMIA

Orgaanisessa kemiassa tutkitaan erilaisia yhdisteitä, joiden pääelementtinä toimii hiiliatomi. Hiiliatomilla on neljä ulkoelektronia, joten se voi muodostaa enimmillään neljä eri sidosta toisten atomien kanssa. Hiili muodostaa eri alkuaineiden kanssa kovalenttisia sidoksia, jolloin muodostuu molekyylejä. Kovalenttisessa sidoksessa atomit jakavat elektroneja keskenään, mikä muodostaa todella vahvan ja vakaan sidoksen atomien välille. Jokainen hiiliatomi muodostaa joko neljä yksinkertaista sidosta, kaksi kaksoissidosta tai yhden kolmoissidoksen ja yhden yksinkertaisen sidoksen. Muodostuneet sidokset ovat vahvoja, joten ne eivät hajoa eivätkä siten voi reagoida muiden aineiden kanssa helposti. Hiiliatomit voivat myös muodostaa toistensa kanssa renkaita ja ketjuja, joiden koolla ei teoriassa ole maksimirajaa. Tämän takia erilaisia orgaanisia yhdisteitä on valtavan suuri määrä, ja suurin osa tunnetuista kemiallisista yhdisteistä onkin juuri orgaanisia yhdisteitä.

2.1.1 Hiilivedyt

Hiilivety sisältää nimensä mukaisesti vain hiili- ja vetyatomeja. Vedyllä on vain yksi elekroni, joten se voi muodostaa vain yksinkertaisia sidoksia muiden atomien kanssa. Yleisimmät hiilivedyt ovat rakenteensa ja suhteellisen pienen koonsa vuoksi kaasuja, mutta suurimmat niistä voivat esiintyä myös nesteinä. Yksinkertaisin hiilivety on nimeltään metaani, jossa yhteen hiiliatomiin on liittynyt neljä vetyatomia. Metaani on hiilidioksidin ohella merkittävä kasvihuoneilmiön aiheuttaja.

Fossiiliset polttoaineet ovat pääasiassa hiilivetyjä, ja niitä polttamalla saadaan runsaasti energiaa. Yleiskäyttöön tarkoitettu nestekaasusäiliö täytetään yleensä kahden eri hiilivedyn, propaanin ja butaanin, seoksella. Myös muovien pääraaka-aineina käytetään usein hiilivetyjä.

2.1.2 Muut hiiliyhdisteet

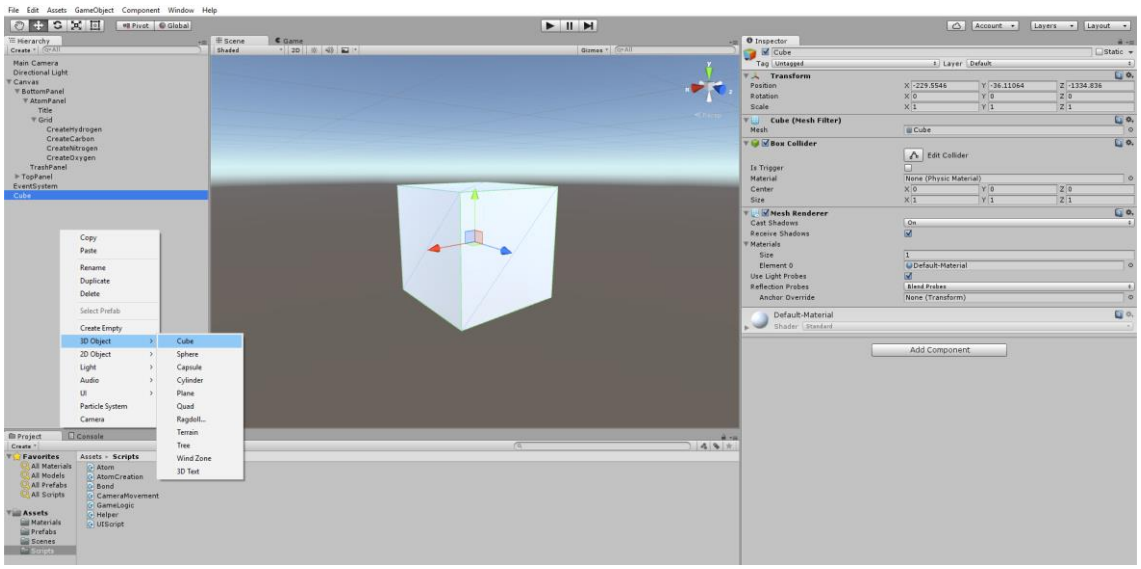
Hiili reagoi helposti myös muiden alkuaineiden kanssa, erityisesti hapen, typen, rikin, fosforin sekä eri halogeenien kanssa. Yksi tunnetuimmista hiiliyhdisteistä lienee hiilidioksidi, jossa hiiliatomiin on liittyneenä kaksi happiatomia. Hiilidioksidia syntyy mm. poltettaessa fossiilisia polttoaineita, minkä yleistyttyessä 1800-luvulta lähtien hiilidioksidimäärät ilmakehässä kasvoivat valtavasti (EPA 2015). Tänä päivänä tiedemiehet yrittävät kuumeisesti keksiä ratkaisuja hiilidioksidin sekä muiden kasvihuonekaasujen vähentämiseksi ilmakehässä ilmastonmuutoksen estämiseksi. Muita hiiliyhdisteitä ovat esimerkiksi erilaiset sokerit, vitamiinit sekä proteiinit, joilla kaikilla on tärkeä rooli elävien organismien toiminnassa. Koska hiilivetyjä on niin paljon ja ne ovat niin tärkeässä roolissa elämän muodostumiselle, niitä on mielenkiintoista tutkia, mistä johtuen niiden rakentaminen valittiin ensimmäiseksi ominaisuudeksi tähän ohjelmaan.

3 UNITY

Unity on Unity Technologiesin kehittämä pelimoottori, joka on kymmenen vuoden olemassaolonsa aikana kehittynyt yhdeksi suosituimmista ilmaisista pelinkehitysalustoista. Unityllä on helppoa tuottaa ohjelmia eri järjestelmille, sen tukien yli kahtakymmentä eri alustaa. Unity käyttää useita eri ohjelmistorajapintoja (API) eri alustoille tehtäviin ohjelmiin. Ohjelmointirajapinnat ovat määritelmiä, joiden mukaan eri ohjelmat keskustelevat keskenään. Macille tehtäviin ohjelmiin käytetään OpenGL-rajapintaa ja OpenGL ES:ää Android ja iOS –mobiilialustoille. Direct3D-API on käytössä Xbox 360 –alustalla, muilla konsoleilla on omat sovelluskohtaiset API:nsa. Windowsille ohjelmia tehtäessä ovat käytössä sekä Direct3D- että OpenGL-rajapinnat. Unityn saa ladattua heidän omilta nettisivuiltaan (unity3d.com 2015), josta löytää myös lisätietoa sen eri ohjelmaversioista.

3.1 Unityn käyttö

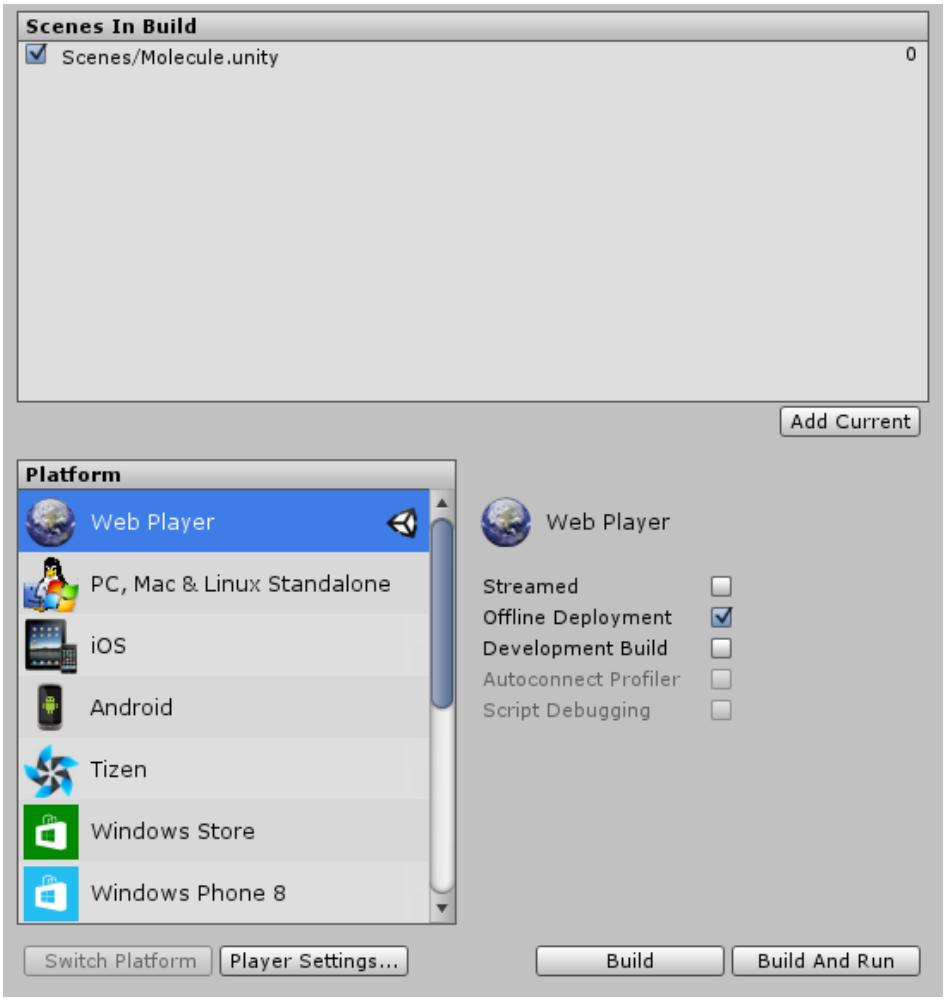
Kuvassa 1 on esitelty Unityn pääikkuna projektin alkuvaiheessa. Hierarchy-paneelista löytyy kaikki projektiin liitetyt peliobjektit. Uusien objektien luonti hoituu helpoiten painamalla oikeaa hiiren näppäintä Hierarchy-paneelin kohdalla, jolloin käyttäjälle aukeaa valikko kaikista Unityn valmiista objekteista.



KUVA 1. Unityn perusnäkymä

Objekteja voidaan muokata ja niihin voi lisätä erilaisia valmiiksi Unitystä löytyviä komponentteja Inspector-paneelin kautta.

Ohjelman ajonaikaista toimintaa voidaan testata suoraan editorista, ilman ohjelman rakentamista (eng. build), mikä nopeuttaa ohjelman luontiprosessia huomattavasti. Unityllä luodut ohjelmat saadaan rakennettua eri alustoilla toimivaksi helposti valitsemalla asetuksista haluttu alusta ja muutama asetus ja painamalla Build-nappia (kuva 2). Tämän jälkeen ohjelma löytyy halutusta tiedostopolusta omana kansionaan ja on valmis käytettäväksi



KUVA 2. Build-asetukset

Ohjelman porttaaminen eri alustoille on yksi suurimmista kompastuskivistä monille ohjelmistokehittäjille, koska siihen kuluu helposti paljon aikaa, eikä varsinkaan pienemmillä ohjelmistotaloilla ole riittävästi resursseja käytettäväksi pelien

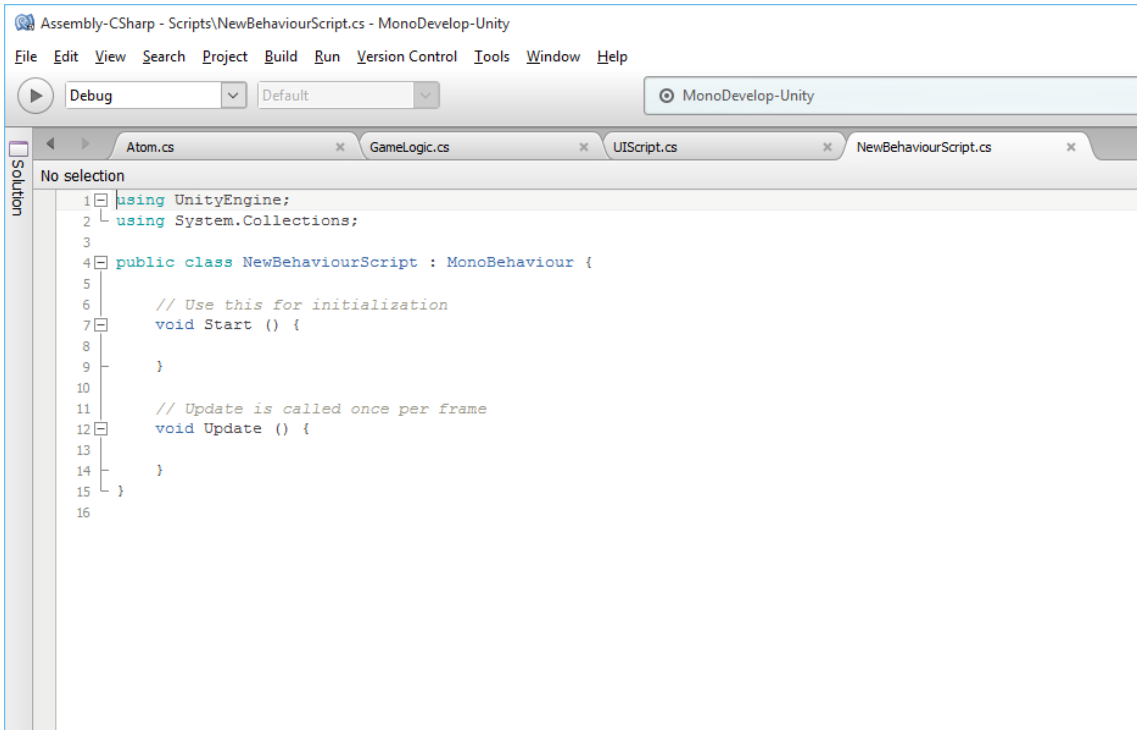
toimintakuntoon saamiseksi useille eri alustoille, kun pahimmassa tapauksessa voidaan joutua kirjoittamaan iso osa ohjelmakoodista kokonaan uudestaan (Mooney 1997). Unityn automaattinen porttausominaisuus sekä laaja alustatuki säästävät todella paljon aikaa, ja sitä kautta rahaa, ohjelmien kehityksessä.

3.2 MonoDevelop

Unityn asennuspaketin mukana tulee MonoDevelop-kehitysohjelma, jonka suunnittelussa Unity Technologies oli mukana järjestelmien yhteensopivuuden takaamiseksi. MonoDevelopia käytettiin tässä työssä ohjelmointiin sekä virheiden etsimiseen sen debugger-työkalulla.

Ohjelmoinnin voi hoitaa joko MonoDevelopilla, tai jos sen ulkoasu ei miellytä, ainakin Microsoftin Visual Studio (visualstudio.com 2015) toimii myös.

Uudesta Unityllä luodusta script-tiedostosta löytyy valmiiksi Start()- ja Update()-functiot (kuva 3). Start-funktio ajetaan vain kerran scriptin ajon alkaessa, joten siinä on hyvä esimerkiksi alustaa koodissa käytettävät muuttujat. Update-funktiota kutsutaan joka framella. Tällaisia funktioita on lukuisia muitakin eri tapahtumille, ja niitä voi etsiä Unityn dokumentaatiosta (Scripting API 2015), joka on hyvä tiedonlähde uudelle Unityn käyttäjälle.



KUVA 3. Näytönkuva MonoDevelopista

3.3 Ohjelmointikielet

Unity tarjoaa mahdollisuuden käyttää kolmea eri ohjelmointikieltä, joita ovat C#, UnityScript ja Boo.

C# (luetaan C sharp) on Microsoftin kehittämä ohjelmointikieli, jonka suunnittelun pääpainona oli erityyppisten ohjelmien helppo rakentaminen käyttäen .NET Frameworkia. .Net Framework on myös Microsoftin kehittämä ohjelmistokehys, joka tarjoaa valmiiksi rakennettuja, käyttövalmiita tietokoneohjelmien osia, mikä nopeuttaa ohjelmien kehitystyötä. Unityssä on paljon näitä ohjelmien osia, jotka on rakennettu nopeuttamaan pelien kehitystä (msdn.microsoft.com 2015).

UnityScript muistuttaa paljon JavaScriptiä, jota on muokattu Unityn kehittäjien toimesta toimimaan Unityn kanssa (UnityScript vs JavaScript 2015). JavaScriptiä käytetään enimmäkseen nettisivujen toiminnallisuuksien toteuttamiseen, vaikka sitä voidaan käyttää myös muissa sovelluksissa.

Bamboo, tai lyhemmin vai Boo, on avoimeen lähdekoodiin perustuva ohjelmointikieli, jonka pohjalla on Python-kieli. Sen alkuperäinen kehittäjä on Rodrigo Barreto de

Oliveira, joka omien sanojensa mukaan turhautui jo olemassaolevien kielien ongelmiin, ja päätti kehittää oman ohjelmointikielensä, jossa nämä ongelmat olisi korjattu toimimaan järkevämmin (Bamboo GitHub 2015).

Tähän projektiin valittiin ohjelmointikieleksi C# ensisijaisesti sen käyttömukavuuden takia. UnityScript tuntui välillä testattaessa huonosti tuetulta MonoDevelopissa. Boo oli projektia aloitettaessa täysin uusi kieli, ja vaikka siitä tietoja etsittäessä se alkoikin kuullostaa houkuttelevalta vaihtoehdolta, ei sen opettelu tuntunut järkevältä ajatukselta tätä projektia varten. Ohjelmointi aloitettiin käyttämällä UnityScriptiä, mutta projektin edetessä todettiin kokeilujen myötä, että C#:lla oli yksinkertaisempaa toteuttaa tiettyjä ohjelman ominaisuuksia, joten lopulta ohjelmointikieli vaihdettiin kokonaan C#-kieleen.

3.4 Miksi Unityllä?

Unitystä on olemassa kaksi eri versiota: Unity Personal sekä maksullinen Unity Pro. Personal-versio on ilmainen kaikille, joten pienelle budjetille se on erinomainen työkalu. Ilmaisuudestaan huolimatta se tarjoaa kaikki tarvittavat ominaisuudet pelinkehitykseen, ja maksullinen versio onkin hyödyllinen lähinnä isompiin projekteihin. Ilmaisella versiolla tehdyistä ohjelmista saaduista tuotoista maksetaan tekijänoikeuskorvauksia Unity Technologiesille vasta niiden ylittäessä $100 000, joten se kelpasi tähän tarkoitukseen varsin hyvin. Toinen vaihtoehto pelimoottorille oli Unreal Engine 4. Tämä kuitenkin hylättiin nopeasti, koska sen käyttöön kuuluu Unityä vähemmän ohjelmointia, jonka oli tarkoitus olla pääpainona tässä opinnäytetyössä. Unreal Enginessä ei periaatteessa tarvitse kirjoittaa riviäkään koodia pelien tekemiseen (Mayden 2014). Se on myös pääasiassa suunnattu peliohjelmointiin, kun taas Unityä mainostetaan monien erilaisten ohjelmistojen tuottamiseen.

OpenGL:ää kokeiltiin myös pikaisesti, mutta matalan tason ohjelmointirajapintana ohjelmistokehitys oli liian hidasta, että sen käyttö olisi ollut kannattavaa. OpenGL:stä myös puuttuu monia perustavanlaatuisia ominaisuuksia, kuten shaderit, muistinhallinta sekä tuki eri käyttöjärjestelmille. Niiden ohjelmoinissa olisi riittänyt opittavaa toiseen opinnäytetyöhön, joten se ohitettiin vain mielenkiintoisena sivuprojektina.

3.5 Asset Store

Asset Store on kokoelma sekä ilmaisia että maksullisia lisäohjelmia, joista osa on Unity Technologiesin ja osa sen käyttäjien tekemiä. Assetteja voi selata Unity Editoriin aukeavassa rajapinnassa (kuva 3) ja niitä voi ladata ja ottaa käyttöön suoraan omiin projekteihin. Unityn käyttäjät voivat myös julkaista ja myydä omia assettejaan Storessa.

KUVA 3. Asset Store

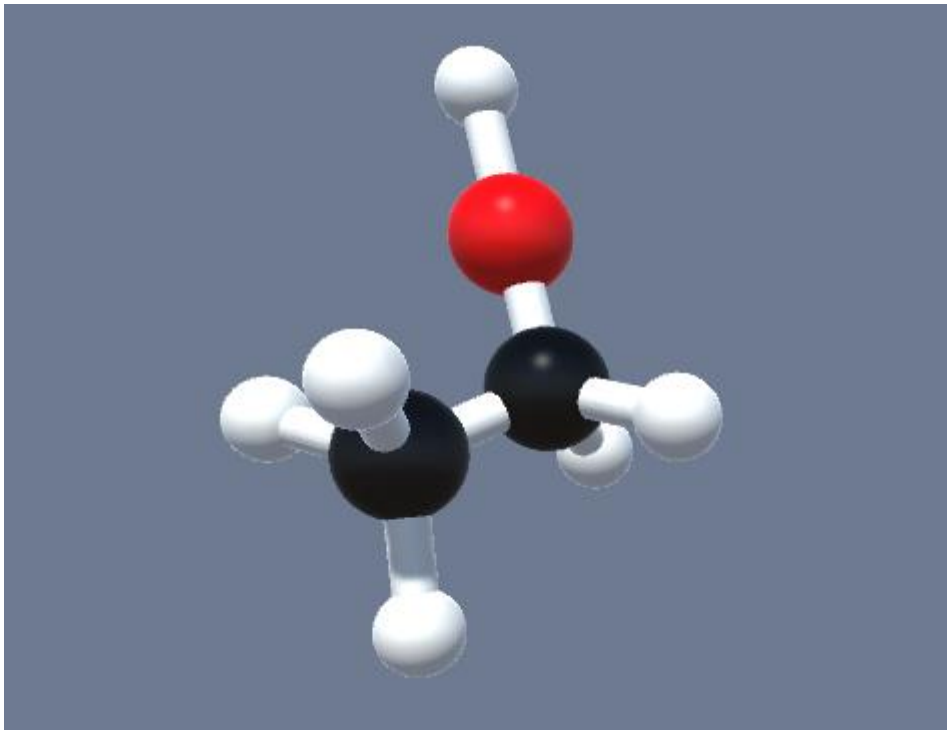
4 OHJELMAN KEHITYS

Minkä tahansa ohjelman kehityksen alkaessa on hyvä panostaa kunnolliseen suunnittelutyöhön ennen kuin ohjelmaa aletaan varsinaisesti tehdä. Hyvällä suunnittelulla säästetään työtunteja merkittävästi, vaikka siihen käytettävä aika usein ylittääkin itse ohjelman tekemiseen kuluvan ajan. Myös ajankäyttöä on hyvä suunnitella. Hyvä aikataulutus auttaa saamaan isonkin ohjelman ajoissa valmiiksi, kun sen kehitysvaiheet voidaan jakaa paremmin sulateltaviksi osiksi.

4.1 Suunnittelu

Vaikka ohjelmaa oli suunniteltu ainakin ajatustasolla jo useampi vuosi, alkoi varsinainen suunnittelutyö vasta keväällä 2015. Suunnittelua vaikeutti kokemuksen puute ohjelmointiympäristön käytössä, ja suunnitelmat muuttuivatkin muutamaan otteeseen ohjelman kehityksen edetessä.

Suurimpia ongelmia ohjelman suunnittelun kannalta oli elektronien ja erityisesti niiden välisten sidosten mallinnus sekä molekyylikaavan laskeminen. Tässä ohjelmaversiossa päädyttiin kuvaamaan elektroneja ns. tappimallilla, jossa elektronit ovat atomiytimestä lähteviä tappeja (kuva 4). Tämä tyyli muistuttaa monille tuttuja muovisia rakennussarjoja, joita nykyään käytetään monissa kouluissa molekyylien mallintamisessa.



KUVA 4. Ohjelmalla luotu etanoli-molekyyli

Tämä on varsin yksinkertainen tapa kuvata elektroneja atomiytimen ympärillä. Oikeassa elämässä elektronit eivät tietenkään toimi näin, vaan niiden paikkaa kuvataan todennäköisyyksillä sijaita tietyssä pisteessä atomiytimen ympärillä niin sanotulla elektronipilvellä (Kvanttimekaaninen atomimalli 2012). Elektronipilven mallintaminen ilman syvempää ymmärrystä asiasta todettiin kuitenkin liian vaikeaksi toteuttaa opinnäytetyölle varatussa ajassa.

4.2 Rakenne

Unityllä ohjelmistokehitys on helposti jaettavissa graafisen käyttöliittymän eli GUI:n kehitykseen sekä scriptaukseen. Käyttöliittymän luonti on intuitiivista Unityn drag&drop-valikoilla, eikä sen tekemiseen tarvitse ohjelmointia lainkaan. Koodi jaettiin eri tiedostoihin käyttötarkoituksen mukaan GUI-toimintoihin, näppäinkomentoihin sekä itse ohjelmalogiikkaan.

GUI-toimintoja ovat atomien luonti sekä tuhoaminen ja molekyylikaavan laskeminen näytöllä näkyvistä atomeista. Myöhemmissä versioissa ohjelmaan on tarkoitus lisätä molekyylien tietojen esittäminen jossain muodossa.

Näppäinkomennoilla liikutetaan kameraa näytöllä, jotta atomia voitaisiin katsoa eri kulmista. Tähän ei ole käytetty kovin paljoa kehitysaikaa, koska se on loppujen lopuksi varsin yksinkertaista tehdä eikä vaikuta itse ohjelman toimintaan kovin merkittävällä tavalla.

Atomiytimet luodaan Unitystä valmiiksi löytyvillä palloilla ja elektronit lieriöillä. Näitä yhdistellään koodin puolella, jolloin niistä saadaan eri atomeja kuvaavat mallit. Atomeja varten luotiin Atom-niminen luokka, joka sisältää kaiken niihin liittyvän tiedon. Luokka liitetään jokaisen eri atomin prefabiin, jonka nimen perusteella kaikki parametrit syötetään atomille (kuva 5). Prefabit ovat Unityllä valmiiksi tehtyjä peliobjekteja, joista luodaan kopio, kun halutaan luoda uusia peliobjekteja Instantiate-funktiolla.

```
/* Initialize variables and create the right atom depending on the prefab name */
private void Start() {
    Initialize();
    if (gameObject.name == "Carbon")
        Create("Carbon", "C", 8, 4, 4, 6, 12.01f, 14, 2, 2.55f);
    else if (gameObject.name == "Nitrogen")
        Create("Nitrogen", "N", 8, 5, 3, 7, 14.01f, 15, 2, 3.0f);
    else if (gameObject.name == "Oxygen")
        Create("Oxygen", "O", 8, 6, 2, 8, 16.0f, 16, 2, 3.5f);
    else if (gameObject.name == "Hydrogen")
        Create("Hydrogen", "H", 2, 1, 1, 1, 1.008f, 1, 1, 2.1f);

    //Add bonds to the created atom
    CreateBonds();
}
```

KUVA 5. Atomin tietojen alustus

Kuvassa 6 tulee hyvin esille, kuinka paljon grafiikkaohjelmointiin liittyviä asioita Unity hoitaa itse. Vektori- tai matriisilaskentaa ei tarvitse itse ohjelmoida tai edes osata kunnolla, riittää kun tietää mitä eri matematiikkafunktioita on olemassa ja mihin tarkoitukseen niistä kutakin käytetään. Esimerkiksi matriisilaskennasta tutut rotaatio ja translaatio (kappaleen liikuttaminen koordinaatistossa) ovat hyvin yksinkertaisia käyttää Unityssä. Tässä ohjelmassa vaikein käytetty funktio oli luultavasti ristitulo, jonka käyttö oli lukioajoilta kerennyt unohtua, minkä seurauksena noin kolmen rivin kirjoitukseen kului aikaa pitkälle toista viikkoa.

```
//Attaches the atoms
private void Attach (List<Transform> bonds) {
    //Make sure there's at least one pair
    if (bonds.Count >= 2) {
        //Go through all of the pairs
        for (int i=0; i<bonds.Count; i+=2) {
            Transform selection = bonds[i];
            Transform target = bonds[i+1];

            //Get the direction vectors for the bonds
            Vector3 dir1 = selection.position - selection.parent.position;
            Vector3 dir2 = target.position - target.parent.position;

            //Calculate the axis of rotation and angle and rotate the atom bonds to face each other
            Vector3 axis = Vector3.Cross(dir1, -dir2);
            float angle = Vector3.Angle(dir1, -dir2);
            if (axis == Vector3.zero) {
                axis = Vector3.Cross (selection.position, selection.parent.position);
            }
            selection.parent.Rotate(axis,angle,Space.World);

            //Move the selection atom to the target and align it
            selection.parent.position = target.position;
            Vector3 dir = (target.position-selection.position).normalized;
            selection.parent.position += dir*2*0.5f;
```

KUVA 6. Osa koodista atomien liittämiseksi

4.3 Ohjelman toiminta

Ohjelmaa käytetään vetämällä ikkunan alalaidasta löytyvästä valikosta atomeja näytölle. Atomit liittyvät toisiinsa, kun ne tuodaan riittävän lähelle toista atomia, kunhan sillä on vapaita elektroneja liitokseen. Liitostyyppiä voi vaihtaa valintaruutujen avulla.

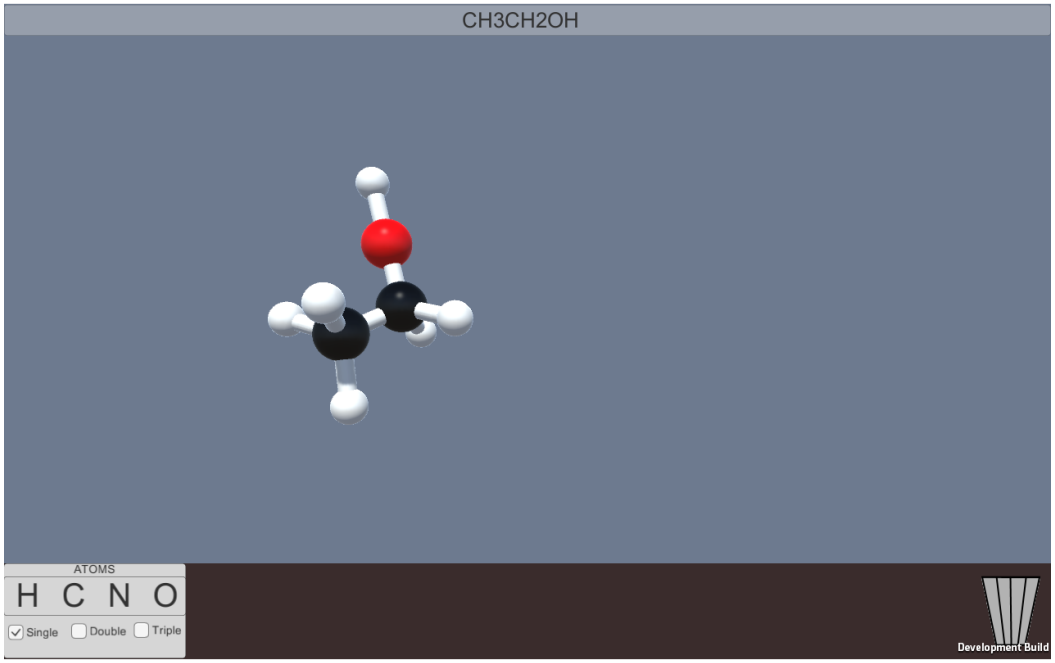
Ohjelman yläreunassa näytetään syntyneen molekyylin molekyylikaava. Kaavan määrittäminen osoittautui odotettua hankalammaksi, ja sitä osaa koodista joudutaankin myöhemmin muokkaamaan, mahdollisesti rankallakin kädellä. Kuvassa 7 osa koodista, joka hoitaa kaavan muodostamisen. Siinä aikaisemmin määritettyä kaavan nimeä supistetaan lähemmäs lopullista muotoa, koodin kommenteissa on esimerkki funktion käytöstä.

```
/*
 * Compress the generated formula to a more familiar form. This needs to
 * be further modified to get the final form using FindFunctionalGroups()
 * Example:
 * CompressForumula("CHHOHCHHH") === "CH2OHCH3"
 */
private string CompressFormula(string text) {
    string ret = "";    //compressed text
    int count = 1;      //counter for consecutive letters
    if (text.Length > 0) {
        //Go through each character in text
        foreach(char ch in text) {
            /* if ret has something in it and the character is not the
             * same as the last one was we add the letter to the 'ret' string
             */
            if (ret.Length > 0 && ch != ret.Last()) {
                string add = count > 1 ? ""+count+ch : ""+ch;
                ret = ret.Insert(ret.Length, add);
                count = 1;
            }
            /* if we are at the first character of the text
             * we simply add the character to the 'ret' string
             */
            else if (ret.Length == 0) {
                ret = ret.Insert(0, ""+ch);
            }
            /* if the character is the same as last one was
             * we increase the counter for that letter and move
             * on to the next
             */
            else if (ch == ret.Last()) {
                count++;
            }
        }
        // handle the last letter in the 'ret'
        string end = count > 1 ? ""+count : "";
        ret = ret.Insert(ret.Length, end);
    }
    return ret;
}
```

KUVA 7. Molekyylikaavan määrittämistä

Atomit voidaan poistaa ohjelmasta pudottamalla ne oikeassa alalaidassa olevaan roskakoriin. Peliobjektien poistamisessa täytyy pitää huolta siitä, että kaikki viittaukset poistettuun objektiin poistetaan ennen itse objektin tuhoamista, sillä Unity ei sitä itse osaa hoitaa.

Kuten kuvasta 8 käy varsin hyvin ilmi, ohjelman toiminnallisuus nykyisessä vaiheessa on vielä hyvin rajallista, ja tyhjää tilaa on paljon. Tila tulee kuitenkin tulevaisuudessa mitä luultavimmin käytettyä, kun ohjelmaan lisätään lisäominaisuuksia. Myös ulkoasu pitänee vaihtaa, jotta kaikki ominaisuudet saadaan mahtumaan näytölle järkevästi.



KUVA 8. Näytönkuva ohjelman toiminnoista.

4.4 Testaus

Ohjelmaa testattiin jatkuvasti kehityksen aikana. Testauksessa oli alusta asti mukana TTY:n opiskelija Timo Paldanius, jonka ankaran silmän alla yksikään kivi ei jäänyt kääntämättä. Myös muutamia muita henkilöitä osallistui ajoittain ohjelman testiajoon. Testauksen ohella he myös toivat uuden käyttäjän näkökulman ohjelman käyttöön, sillä heillä ei ollut aiempaa käsitystä siitä, miten tai mihin ohjelmaa tulisi käyttää.

Alkutestaus suoritettiin joko paperille manuaalisesti, tai kun mahdollista, käyttäen Unityn omaa debugger-työkalua logiikka- ja käännösvirheiden etsimiseen ja korjaamiseen. Kun ohjelma ei enää päällisin puolin ilmoita virheistä, alkaa ohjelman ajonaikainen testaus, jossa tarkistetaan, että kaikki ominaisuudet toimivat niin kuin niiden pitäisi. Siinä pyritään kokeilemaan ohjelman toimintaa kaikilla mahdollisilla tavoilla, jotta voidaan varmistua siitä, että ohjelma ei mene rikki, vaikka kokematon käyttäjä käyttäisi sitä väärin. Tätähän ei saisi tietenkään tapahtua, vaan pitää varmistaa, että ohjelman käyttö onnistuu ilman siihen perehtymistä. Testaus on usein koko ohjelmistokehityksen turhauttavin osuus, ja mikä vain apu otettiin mielellään vastaan, kun sitä tarjottiin.

5 JATKOKEHITYS

Tätä ohjelmaa ei ollut koskaan tarkoitus saada täysin valmiiksi opinnäytetyötä varten, vaan sen kehitystä on tarkoitus jatkaa myöhemmin. Siksi ohjelman käyttöominaisuudet jäivät vähäisiksi eikä sen käyttö vaikuta vielä nykyisessä muodossaan erityisen hyödylliseltä. Tämä ongelma on tarkoitus korjata olemalla yhteydessä jonkin koulun opettajiin, joilla toivottavasti olisi ideoita, mitä ominaisuuksia ohjelmaan voisi lisätä.

Tällä hetkellä jatkokehitys tarkoittaa ensisijaisesti systeemien saamista sellaiseen kuntoon, että virheilmoituksia ei enää tulisi ja kaikki perusominaisuudet toimisivat niin kuin ne on suunniteltu. Koodia on luultavasti myös refaktoroitava, jotta mahdollisimman suurta osaa siitä voisi käyttää muissa ohjelmaan jatkossa tulevissa moduuleissa. Näin ohjelman yleisilme ja perustoiminta saadaan pysymään yhdenmukaisena ja sen käytettävyys paranee.

Joitain lisäominaisuuksia on kuitenkin jo suunnitelmissa, mistä voidaan kiittää innokkaiden ystävien ahkeraa testausta. Ohjelman voisi pelillistää, esimerkiksi antamalla käyttäjälle valmiiksi rakennetun molekyylin, jonka molekyylikaava pitäisi syöttää ohjelmalle. Tämä toimisi tietysti myös päinvastoin. Myös atomien ja molekyylien tiedot täytyy saada näkymään näytöllä jollain tapaa, esimerkiksi esittämällä ne taulukossa. Tämän pitäisi olla melko yksinkertainen asia lisätä, kun on päätetty mitkä asiat ovat tärkeitä olla esillä. Nämä ovat vain esimerkkejä siitä, mitä kaikkea ohjelmaan voisi lisätä. Tämä molekyylinmallinnusohjelma toimii kuitenkin vain pilottiohjelmana isommalle ohjelmakokonaisuudelle, joka sisältää myös uusia ohjelmakokonaisuuksia omine moduuleineen. Lopullinen versio ohjelmasta tulee siten olemaan paljon suurempi ja voi hyödyttää muitakin kuin pelkkiä kemian opiskelijoita.

6 POHDINTA

Tässä opinnäytetyössä tarkoituksena oli tutustua Unity-pelimoottorin toimintaan, ja toteuttaa sillä runko 3D-molekyylinmallinnusohjelmaa varten. Ohjelmaa on jatkossa tarkoitus käyttää kouluissa opetustarkoituksessa, joten sen jatkokehityksessä olisi hyvä löytää halukkaita testikäyttäjiä jatkuvaa palautteen saamista varten.

Suurimmaksi kompastuskiveksi projektin varrella osoittautui kokemattomuus laajojen ohjelmistojen suunnittelussa, varsinkin uuden kehitysympäristön kanssa. Jatkossa tämänlaajuisten projektien alkaessa olisikin hyvä tutustua paremmin kehitystyökalujen toimintaan ja kuluttaa enemmän aikaa suunnittelutyöhön. Näin säästyisi loppupeleissä runsaasti aikaa, kun ohjelmaa ei tarvitse kirjoittaa moneen kertaan uudestaan ongelmiin törmättäessä tai kun huomataan, että jokin suunniteltu osio ei toimikaan jostain syystä.

Tehdyistä virheistä kuitenkin otettiin opiksi, ja tulevaisuudessa niihin ainakin toivottavasti törmätään harvemmin, parhaassa tapauksessa niihin ei enää tarvitsisi törmätä uudestaan. Unity todettiin varsin toimivaksi ohjelmaksi tämänkaltaiseen ohjelmistoprojektiin, joten sitä uskaltaa käyttää jatkossakin, ja suositellaan muillekin peliohjelmoinnista kiinnostuneille varteenotettavana vaihtoehtona.

LÄHTEET

GitHub. 2015. The Boo Programming Language. Luettu 4.12.2015. https://github.com/bamboo/boo

Jakovuori, R., Rauhala, M. & Saastamoinen, T. 2014. Digitalisaation kynnyksellä: Kohti tulevaisuuden lukiota. PDF-dokumentti. https://www.accenture.com/fi-en/insight-threshold-digitalization-high-school-future.aspx

Kvanttimekaaninen atomimalli. Youtube 2012. Katsottu 28.11.2015. https://www.youtube.com/watch?v=k_Mezl9nJho

Learner. 2015. Life Science: Organic molecules. WWW-sivu. Luettu 24.11.2015. https://www.learner.org/courses/essential/life/session1/closer2.html

Mayden, A. 2014. Digital Tutors. Blogi. Luettu 19.11.2015. http://blog.digitaltutors.com/unreal-engine-4-vs-unity-game-engine-best

Meyer, B. & Kristensen M. Ei julkaisutietoja. Digitalising Schools – the Challenge of Building Educational Environments for the Future. PDF-dokumentti. Luettu 6.11.2015. http://conference.pixel-online.net/edu_future/common/download/Paper_pdf/ENT33-Meyer.pdf

Michigan State University. 2013. The Shape of Molecules. WWW-sivu. Päivitetty 5.5.2013. Luettu 6.11.2015. https://www2.chemistry.msu.edu/faculty/reusch/VirtTxtJml/intro3.htm

Microsoft Developer Network. 2015. C#. WWW-sivu. Luettu 4.12.2015. https://msdn.microsoft.com/en-us/library/kx37x362.aspx

Mooney, J. 1997. Bringing Portability to the Software Process. PDF-dokumentti. Luettu 30.11.2015. http://www.cs.wvu.edu/~jdm/research/portability/reports/TR_97-1.pdf

OECD. 2009. OECD Study on Digital Learning Resources as Systemic Innovation. PDF-dokumentti. Luettu 1.12.2015. http://www.oecd.org/edu/ceri/42033180.pdf

Stack Overflow. 2015. UnityScript vs JavaScript. WWW-sivu. Luettu 4.12.2015. http://stackoverflow.com/questions/29194823/unityscript-vs-javascript

United States Environmental Protection Agency. 2015. Climate Change Science Overview. WWW-sivu. Päivitetty 4.11.2015. Luettu 24.11.2015. http://www3.epa.gov/climatechange/science/overview.html

Unity Technologies. 2015. Unity – Game Engine. WWW-sivu. Luettu 4.12.2015. http://unity3d.com/

Unity Technologies. 2015. Unity Scripting API. WWW-sivu. Luettu 30.11.2015.

http://docs.unity3d.com/ScriptReference/

Visual Studio. 2015. Visual Studio – Microsoft Developer Tools. WWW-sivu. Luettu 4.12.2015.
https://www.visualstudio.com/