**Maturaarbeit**

**Titelblatt**

**Inhaltsverzeichnis**

**Abstract**

(Zielsetzung, Vorgehen, Weg zum Ziel)

**Vorwort**

(Grund für die Erstellung der Arbeit, persönliche Sicht auf die Arbeit, Adressat)

In dieser Maturaarbeit habe ich es geschafft zwei meiner Begeisterungen zu vereinen. Das eine ist der Rubik’s Cube, das andere ist die Informatik.

Der Würfel hat vor einigen Jahren an Weihnachten meine Begeisterung geweckt. Damals bekam mein jüngerer Bruder einen solchen geschenkt. Seine Begeisterung hielt sich in Grenzen. Als der Würfel verdreht war und ihm die Lösung versagt blieb, legte er den Würfel zur Seite.

Ich versuchte mich natürlich auch mit dem Würfel und drehte eifrig herum. Doch meine ersten Versuche scheiterten ebenfalls kläglich. Später versuchte ich im Internet nach Lösungsmethoden; und tatsächlich fand ich einige Tutorials. Beim Schreiben meiner Maturaarbeit kam mir diese Zeit immer wieder in den Sinn, weil ich damals schon ähnlich vorging, wie jetzt bei der Maturaarbeit.

Ich recherchierte auf unterschiedlichen Seiten bis ich den Würfel zum ersten Mal lösen konnte. Mit der Zeit lernte ich die Züge auswendig. Als ich den Würfel auf Zeit zu lösen versuchte, recherchierte ich nach weiteren Kombinationen, die es mir ermöglichten, den Würfel noch schneller zu lösen.

In meiner ersten Studienwoche im Gymnasium konnte ich andere meiner Klasse für den Würfel begeistern. Ich lernte ihnen mein Wissen. Inzwischen hat sich das „Würfelfieber“ wieder etwas abgeschwächt, doch viele erinnern sich immer noch an die Lösungswege.

Die Informatik begeisterte mich bereits früher. Ich hatte schon früh eine Leidenschaft für das Technische. In der sechsten Klasse wurde dies weiter gefördert, als unsere Klasse die einzige war, die eigene Computer im Schulzimmer hatte. Natürlich begann alles im kleinen Rahmen. Am Gymnasium wählte ich das Ergänzungsfach Informatik. Den Anfang fand ich sehr mühselig. Es ist ähnlich, wie wenn man eine neue Sprache lernt: Zu Beginn versteht man kaum etwas und erst nach einer Menge Aufwand kann man sich über kleine Erfolge freuen. Mit der Wahl des Ergänzungsfachs in Informatik machte ich einen grossen Schritt in diese Richtung. Dort lernte ich weitere Interessante Fakten zur Java-Programmierung.

Diese Arbeit soll dich dazu verleiten in die abstrakte Welt einzutauchen und dir zeigen, dass weder der Rubik’s Cube noch die Informatik abschreckende Themen sind. Im Gegenteil erscheint es mir sehr spannend, das Unbekannte zu entdecken und neues daraus zu lernen.

*„Lernen ist wie Rudern gegen den Strom. Sobald man aufhört, treibt man zurück“*

(Loazi, chinesischer Denker des 6. Jahrhunderts v. Chr.)

**Einleitung**

(Problemschilderung/Fragestellung, Zielsetzung, Ausgrenzungen, Lösungsweg)

Es ist ein Mysterium. Ob gross oder klein, jeder versucht sich damit. Es hat eine grosse Faszination, weil die sechs Farben immer andere Muster bilden und wenn man den Dreh einmal heraus hat, will man das Knobelspiel immer und immer wieder lösen. Es handelt sich um den Rubik’s Cube. Ein Spiel, das eine fast unbegrenzte Möglichkeit an Kombinationen hat und gleichzeitig immer ein Ganzes ist.

In meiner Maturaarbeit will ich das dreidimensionale Original auf einem Bildschirm darstelle. Dabei sollen die gleichen Manipulationen vorgenommen werden können, wie beim Original.

Da ich mich nicht mit Matrizen auskenne, habe ich mich entschieden eine Bibliothek zu verwenden, die mir die Rechenaufgaben abnimmt. Ich habe mich entschieden die Java 3D Bibliothek von Oracle zu verwenden.

**Geschichte des Rubik’s Cube**

Die Geschichte des Rubik’s Cube beginnt bei einem Dozenten an der Fakultät für Innenarchitektur. Er unterrichtete an der Universität für angewandte Künste in Budapest. Von Natur interessierten ihn Geometrie und dreidimensionale Formen. Seinen Studenten brachte er viele Modelle aus Holz, Pappe, Papier oder Plastik mit, um ihnen seine Ideen zu erklären.

Später stellte er sich selbst die Aufgabe ein dreidimensionales Objekt zu erfinden. Er wollte ein Knobelspiel erfinden, das viele Kombinationen zulässt und immer ein Ganzes ist. Bei seinen ersten Überlegungen schien ihm dies unmöglich, dass sein Modell zusammenhält und gleichzeitig in drei Achsen verdreht werden kann. Seine ersten Versuche einen Würfel mit Gummibändern zusammenzuhalten scheiterten. Nach längeren Überlegenen kam er auf die Idee, dass die innere Struktur rund sein musste, damit sich die einzelnen Bestandteile des Würfels in alle drei Richtungen verdrehen liessen. Er entschied sich die sechs verschiedenen Flächen durch Farben voneinander zu unterscheiden.

Im Jahre 1974 kam die erste Patentanmeldung für seinen Prototyp.

Der ungarische Spielzeughersteller „Politechnika“ nahm sich der Aufgabe an, den Würfel als Massenprodukt herzustellen. Wegen der Komplexität im inneren des Würfels und der schlechten inneren wirtschaftlichen Lage stellte sich dies als schwierig heraus. Erst 1979 konnten die ersten Würfel in Spielzeugläden gekauft werden.

Zwei im Westen lebende Ungaren, erreichten, dass der Würfel auch auf der anderen Seite des Eisernen Vorhang zum Massenprodukt wurde. Es handelte sich dabei um Dr. Laczi Tibor und Tom Kremer.

Dr. Tibor entdeckte den Würfel auf einer Geschäftsreise. Er erkannte sofort das Potenzial des Würfels. Er nahm den Würfel mit und erhoffte sich auf der Nürnberger Spielzeugmesse 1979 einen interessierten Spielzeughersteller zu finden. Obwohl sein Erfolg ausblieb, traf er Tom Kremer, der ihn von da an unterstützte. Die beiden Männer waren es, die den Würfel auch auf der anderen Seite des Eisernen Vorhangs Bekanntheit verschafften.

Dr. Laczi ging zurück nach Ungarn, um die Produktion anzutreiben und Kremer machte eine Welt-Tour, um Spielzeughersteller für sein Produkt zu überzeugen. Bei den grossen Firmen löste der Würfel kein grosses Interesse aus, da die Produktion zu schwierig und zu teurer erschien.

Kremer gelang es nach vielen Fehlversuchen mit der Firma „Ideal Toy Corporation“ eine Bestellung von einer Million Exemplaren auszuhandeln.

1980 hatte der Würfel auf mehreren westlichen Spielzeugmessen Erfolg. Bei der Lieferung der Würfel gab es Probleme, da die westlichen Qualitätsansprüche sehr hoch waren.

(Quelle: <http://www.rubiks-zauberwuerfel.de/home.html?geschichte.html>)

**Theoretischer Teil**

**Aufbau**

**Der Scene Graph**

Der Scene Graph ist das zentrale Grundelement von Java 3D. Es handelt sich um eine hierarchische Baumstruktur. Alle Objekte, die verwendet werden, werden in die Baumstruktur eingesetzt. Das Universe bildet die Wurzel des Scene Graph. Es bildet den dreidimensionalen Raum, in welchem sämtliche Objekte eingesetzt werden.  
Darunter folgt das Local. Es stellt den Ursprung im Raum dar. Mit ihm wird ein Koordinatensystem geschaffen. Unterhalb des Local verzweigt sich der Baum in zwei unterschiedliche Branch Graph -Objekte. Die beiden Subgrafen haben unterschiedliche Aufgaben. Der Content Branch Graph beinhaltet sämtlichen Inhalt einer Applikation. Das heisst Geometiren, Erscheinungen, Bewegungen und Beleuchtungen. Der View Branch Graph beinhaltet die Ansicht. Das heisst die Kameraposition und die Kameraausrichtung.

Da das Universe, Locale und der View Branch Graph in jedem Programm gleich aufgebaut werden, gibt es ein Hilfsobjekt, das Simple Universe. Es erstellt diese Objekte und fügt die Standartwerde ein. Ich habe das Simple Universe in meinem Programm verwendet und mich deshalb nicht tiefgehend mit dem View Branch Graph auseinandergesetzt.

Im Content Branch Graph gibt es weiter Node und Node Component Objekte. Die meisten gehören zu den Node’s.

Ein Node ist ein Knotenpunkt im Scene Graph. Wenn er am Ende eines Astes steht und somit keine Kind-Objekte hat, nennt man ihn Leaf Node. Hat es unter dem Node Kind-Objekte, so gehört er zu den Group Node Typen.

In meinem Projekt waren bei den Group Node’s die Transform Group und die Branch Group von Bedeutung. Es gibt jedoch noch andere (siehe Abbildung x).

Ein Branch Graf wird zum Leben erweckt, wenn er in ein Locale eingesetzt wird. Alle Kind-Objekte des Branch Graf werden auch lebendig. Um Parametewr von lebenden Objekten zu verändern müssen die Capabilities vor dem Einhauchen des Lebens gesetzt werden.

Bevor ein Branch graf in ein Local eingesetz wird, sollte er compiled werden. Dadurch wird er effiezeinter.

BranchGrafen sind die einzigen, die Kinder eines Locals sein dürfen.

Canvas3D Class

The Canvas3D class is derived from the Canvas class of the Abstract Windowing Toolkit (AWT). At

least one Canvas3D object must be referenced in the viewing branch graph of the scene graph3.

Transform3D ermöglichen Translationen und Rotation von Objekten in Java 3D. Sie werden nur mit Transformations Gruppen verwemdet. Die Transform3D ist kein kein SceneGrapf. Er spezifiert eine transformation der TransformGroup.

Transform3D enthält drei wichtige Methoden, die die Richtung und die Anzahl Grad (in Radianten) einer Drehung angeben. rotX(), rotY(), rotZ()

Durch das Multiplizieren von zwei Transform3D können mehrer Drehungen zusammengefügt werden. Dadurch kann ein Objekt in einem Schritt um mehrere unterschuiedliche Achsen gedreht werden.

TransformGroup beinhaltet Transformationen, die sie von Transform3d erhält. Es ist die einzige Klasse, die die Orientierung, Ort und die Grösse von Objekten verändern kann.

**Arc**

Zwischen den Datenelementen im Scene Graph gibt es Beziehungen. Diese heissen Arc. Man unterscheidet auch hier zwischen zwei unterschiedlichen Beziehungen.

Ein Eltern-Kind Beziehung („Parent-Child link“) kann bei einem Group Node auftreten. Dabei kann es mehrere Kind-Objekte geben jedoch immer genau ein Eltern-Objekt.

Eine Referenz („referenz“) verbindet einen Node Component –Objekt mit einem Szenengraf-Node.

Node Componenten beschreiben die Gestaltungs- und das Aussehens-Attribute.

Ein Node bildet im Szenengraf eine Wurzel. Von ihr führt zu jedem Node Leaf genau ein Weg. Deshalb sind auch Kreise in einem Szenengraf nicht erlaubt. Der Weg wird als Szenengraf-Weg bezeichnet.

**Praktischer Teil**

**Vorgehen**

1. Da in meiner Matura-Arbeit der Rubik’s Cube eine zentrale Rolle spielt habe ich mich zu Beginn mit dem Aufbau dieses Würfels beschäftigt. Meine erste Erkenntnis war, dass der Würfel aus unterschiedlichen Steinen besteht. Es gibt insgesamt drei Steintypen. Dabei hat jeder Typ andere Eigenschaften.

Der Mittelstein befindet sich in der Mitte einer Fläche. Er hat nur eine farbige Oberfläche. Die sechs Mittelsteine sind miteinander verbunden und halten den Würfel zusammen. Daher können die Mittelsteine beim Drehen der Flächen in ihrer Position nicht verändert werden. Beim meinem Würfel heisst das konkret, dass der weisse Mittelstein immer gegenüber des gelben, der rote gegenüber des orangen, der blaue gegenüber des grünen Mittelsteines stehen wird, egal wie oft der Würfel gedreht wird. Insgesamt gibt es sechs Mittelsteine.

[Foto des offenen Cubes]

Der Kantenstein ist bereits etwas spannender. Er befindet sich zwischen zwei Mittelsteinen und hat zwei farbige Oberflächen. Dieser Steintyp ist wesentlich freier, da er seine Position im Würfel verändern kann. Er kann mit jedem der elf anderen Kantenstein vertauscht werden.

Der Eckstein liegt, wie der Name schon sagt, in den Ecken des Würfels. Er hat drei farbige Oberflächen. Die Ecksteine können ihre Position untereinander vertauschen. Insgesamt gibt es acht Ecksteine.

Für mein Programm habe ich mir gedacht, dass ich aus dieser Erkenntnis eine Klasse „Stein“ bilde, von der ich danach drei Klassen ableite. Diese Klassen sollen die oben genannten Steintypen darstellen. Den Rubik’s Cube würde ich aus diesen drei Klassen zusammensetzen, indem ich mehrere Objekte erstelle.

Für die Vaterklasse benötigte ich zur Erstellung des Würfels noch zwei Methoden. Eine Methode sollte die Stein-Objekte an ihre richtige Position verschieben und die zweite sollte die Stein-Objekte an einer bestimmten Position ausrichten.

1. Der zweite wichtiger Punkt war die Herausforderung der dritten Dimension. Bei der Zielfindung habe ich mich lange schwer getan, weil ich nicht wusste, wie ich dieses Problem angehen sollte. Ich hatte ein Buch, welche das Problem durch Matrizen löst. Da mir dieses Thema den Rahmen meiner Arbeit zu sprengen drohte, habe ich den Tipp meiner Betreuungsperson beachtet und mich dafür entschieden mit der Java 3D-Bibliotheke zu arbeiten.  
     
   Die Benutzung der Bibliothek benötigte einen grösseren Zeitaufwand, weil es sich um die erste Bibliothek handelt, die ich in grossen Umfang benutzte. Im Internet habe ich eine umfassende Dokumentation über die Klasse gefunden, die ich als Selbststudium las. Dabei habe ich folgendes System entwickelt: Nach den Kapiteln habe ich versucht das gelesenen in Beispielprogrammen umzusetzen. Zu Beginn habe ich Beispiele im Internet gesucht, direkt abgeschrieben und versucht zu interpretieren, was die einzelnen Zeilen genau aussagen. In einem späteren Zeitpunkt habe ich die Beispielprogramme meinem Projekt angepasst. Mit diesem Vorgehen habe ich mich langsam an mein tatsächliches Ziel angenähert.  
     
   Während dem Selbststudium habe ich gemerkt, dass es sich nicht auszahlt die drei Steintypen zu vererben. Weshalb ich alle Steine aus einer Klasse bildete.

Aufbau

Mein Programm besteht aus fünf Klassen. Die Klasse Stone wird nur am Anfang aufgerufen und zeichnet die

**Ergebnisse/Diskussion**

**Literaturverzeichnis**

**Anhang**

**Selbstständigkeitserklärung**