Bericht zur 05 Transformationen

Marcus Bätz, Andreas Kiauka, Robert Dziuba

11. Januar 2016

Inhaltsverzeichnis

1 Aufgabenstellung

Es soll eine 4x4 Matrix eine Klasse für die Transformation und ein Scenegraph implementiert werden.

1.1 4x4Matrix

Es sollte eine 4x4 Matrix implementiert werden die als Funktion eine Multiplikation mit einem Vector3 Objekt, einem Point3 Objekt und einer anderen 4x4Matrix sowie einer Funktion zur Rückgabe der transponierten Matrix.

1.2 Transformation

Transformation enthält eine Matrix und die inverse dieser Matrix. Der öffentliche Konstruktor initialisiert die Matrix mit der Einheitsmatrix. Der Konstruktor der Matrix der beide Attribute als Parameter entgegen nimmt ist privat. Des weiteren besitzt die Klasse Methoden zur Skalierung, Translation und Rotation entlang aller 3 Achsen.

1.3 Scenegraph

Die von Geometrie abgeleitete Klasse Node bekommt über den Konstruktor ein Transform Objekt und eine liste an Geometrien

2 Lösungsstrategien

Zur besseren Bearbeitung der einzelnen Klassen, haben wir zu Beginn von allen Klassen gemäß des Klassen-Diagramms Dummy-Klassen erzeugt. Damit konnten die Klassen ohne Compiler-Fehler geschrieben werden. Wir haben sie gleichmäßig verteilt und auf unseren eigenen Git-Branche bearbeitet. Als alle Klassen fertig waren wurden sie auf den master gemerged und abschließend alle noch auftretenden Fehler beseitigt.

3 Implementierung

3.1 Matrix4x4

Die abstrakte Klasse "Matrix4x4“ erhält 16 Parameter. und 4 Funktionen.

```
1          public Vector3 mul(final Vector3 v) {
2          if (v == null) throw new IllegalArgumentException(
               "Null as parameter");
3          final double x = this.m11 * v.x + this.m21 * v.y +
               this.m31 * v.z;
4          final double y = this.m12 * v.x + this.m22 * v.y +
               this.m32 * v.z;
5          final double z = this.m13 * v.x + this.m23 * v.y +
               this.m33 * v.z;
6
7          return new Vector3(x, y, z);
8          }
```

3.2 Transfom

Das Transform Objekt enthält 2 Attribute, nämlich 2 4x4Matrizen sowie 7 Funktionen. 4 Funktionen für Rotation Skalierung und Translation und 2 Funktionen um die Oberflächen normale sowie den Strahl der Kamera zu transformieren.

```
1  v
2  }
```

3.3 Node

Node dient als einheitliche Behälter für alle von Geometry abgeleiteten Klassen, also auch Node. Dadurch lassen sich beliebige Gruppierungen und Transformationen erreichen.

```
1     public Hit hit(final Ray r) {
2     final Ray tr = t.mul(r);
3
4     Hit hit = null;
5
6     for (final Geometry g : geos) {
7     final Hit h = g.hit(tr);
8     if (hit == null || (h != null && h.t < hit.t)) hit = h;
```

```
9       }
10
11      return (hit == null) ? hit : new Hit(hit.t, t.mul(hit.n)
            ,r,hit.geo,hit.texCoord);
12      }
```

4 Probleme bei der Bearbeitung

Während der Bearbeitung sind mehrere Probleme aufgetreten.

1. Die Attribute der Mat4x4 waren mehrfach falsch gesetzt und führten zu falschen Ergebnissen.

2. Es dauerte einige Zeit bis wir den Aufbau der Transform Klasse verstanden hatten.