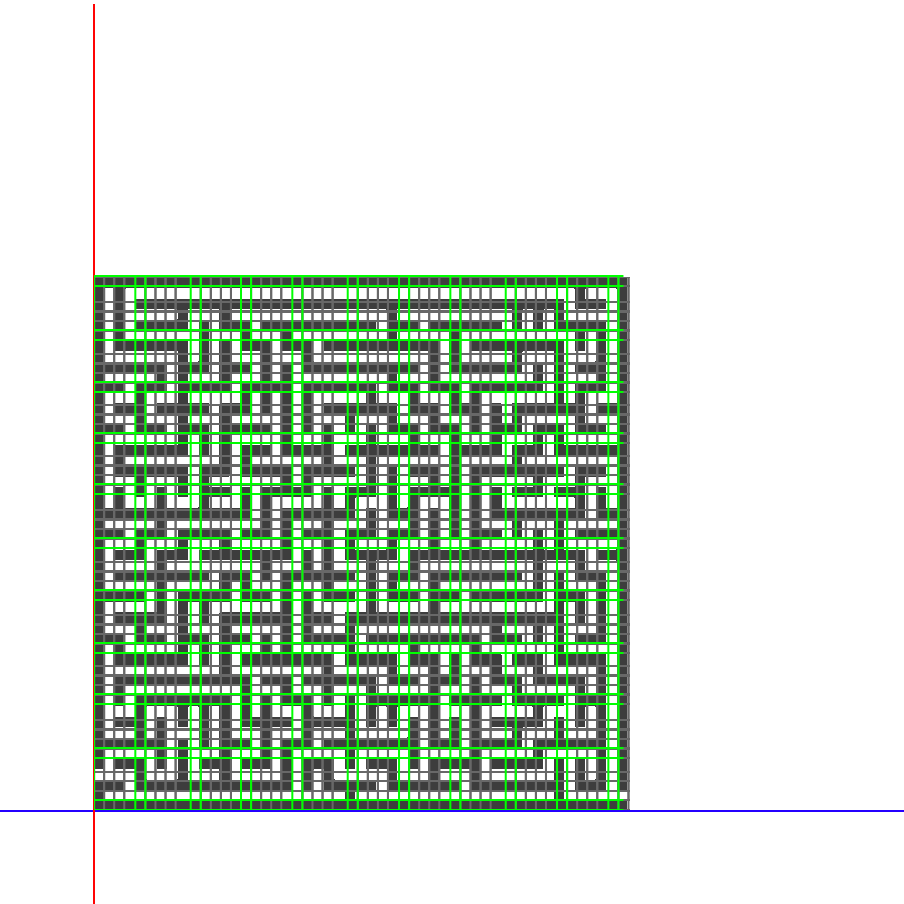
# Dokumentation „aMAZEing Project“

## Implementierung der Features

### Erstellung der Labyrinthkarte

Das Labyrinth wird nicht zufällig durch einen Algorithmus erzeugt, sondern „von Hand“ anhand eines bestehenden Labyrinth-Bildes. Dazu wird das Bild in Quadrate unterteilt, bei denen zwischen unterschiedlichen Grundttypen von Gangsegmenten unterschieden wird.



(Beispiel: Map 2)

Die Standard-Gangsegmente sind (Benennung bezüglich der 2D-Karte):

* Gang (oben/unten, rechts/links)
  + Wände an zwei Seiten
* Sackgasse (oben, unten, rechts, links)
  + Wände an drei Seiten
* T-Kreuzung (oben, unten, rechts, links)
  + Wand an einer Seite
* Kreuzung
  + keine Wände
* Kurve (rechts/oben, rechts/unten, links/oben, links/unten)
  + Wände an zwei aneinandergrenzenden Seiten
* Wand
  + Decke (nur von oben sichtbar)

Die Informationen werden mithilfe einer Enum in einem 2-dimensionalen Array gespeichert und können von einer eigenen Klasse ausgelesen werden, die dann diese in Vertex-Positionen umsetzt. Die Gangbreite beträgt 5 Einheiten.

Wir haben uns für diese Variante statt einer vollständig automatisierten entschieden, da so Jacqueline Klein trotz Krankheit etwas zum Spiel beitragen konnte, das dann nicht vom Rest der Gruppe erledigt werden musste.

Leider war es uns aufgrund von Zeitmangel nicht möglich, die Karten gründlich auf Fehlerlosigkeit zu untersuchen und ihre Schaffbarkeit zu testen. Diese Überprüfung musste zugunsten von wichtigeren Features (z.B. die Kollisionserkennung) hintenanstehen.

### Spieler und Kamera

Der Spieler wird über die Pfeiltasten bewegt. Die Bewegung entspricht seinem Forward- / Backward- / Left- / Right-Vektoren. Die Tülle der Teekanne weist dabei immer nach vorne.

Die Kamera befindet sich ursprünglich 2 Einheiten unter (y-Richtung) und 3 Einheiten hinter (z-Richtung) dem Spieler und kann mit den Tasten WASD um den Spieler herum gedreht werden. Dabei ist ihr Blick stets auf die Teekanne gerichtet. Sie bewegt sich bei Betätigung der Pfeiltasten ebenfalls genauso weit wie der Spieler, um diesem durch das Labyrinth zu folgen. Bei den rechts/links-Drehungen (A/D) wird auch der Spieler gedreht, um zu gewährleisten, dass die Vorwärtsbewegung mit Pfeiltaste weiterhin in Richtung der Spitze der Teekanne geht. Dies soll die Orientierung erleichtern.

Leider ist dieses Feature nicht 100%ig zuverlässig. Bei sehr viel Drehen ergibt sich mit der Zeit ein Offset. Außerdem sollte man sich nicht versuchen, über die Kanne zu drehen.

Die Kamera bekommt dafür das Spieler-Objekt bei Initialisierung übergeben, damit sie bequem auf dessen Position etc. zugreifen und sich danach richten kann.

### Kompass

Der Kompass ist ein blauer Pfeil, der sich genau unter dem Spieler auf dem Boden bewegt. Er zweigt immer in Richtung des Vector3.Forward, also „nach Norden“. Er soll die Orientierung im Labyrinth erleichtern, die aufgrund der vielen Richtungswechsel und Drehungen etwas schwierig werden kann.

### Hint

Der Hint ist ein grüner Pfeil, der über dem Kompass schwebt. Er bewegt sich ebenfalls direkt unter dem Spieler (etwas höher als der Kompass).

Er kennt das Zielfeld der Labyrinthkarte und weist mit seinem Forward-Vektor immer in diese Richtung. Dies dient auch der Orientierung, ist aber nur bedingt eine Hilfe, da sich zwischen dem Spieler und dem Ziel für gewöhnlich viele Wände und Sackgassen befinden.

### Timer (nicht fertig implementiert)

Geplant war die Implementierung eines Timers, der die Zeit misst, die der Spieler braucht, um das Labyrinth hinter sich zu bringen. Dazu wurde eine Timer-Klasse erzeugt, die sich in der Update-Methode die TotalGameTime (in Sekunden) speichert, um diese anzuzeigen.

Leider scheint sich die spriteBatch.DrawString()-Methode nicht mit den anderen Draw-Methoden der anderen Objekte (Labyrinth, Spieler, etc.) zu vertragen und sorgt für eine vollkommen unsinnige grafische Darstellung. Dieses Problem konnte trotz Versuchen mit mehreren verschiedenen Effekten nicht gelöst wurden und der Timer wurde daher auskommentiert.