# Informationen zu Projektteilnehmern

### Projektteilnehmer

* Hendrik Janert  
  E-Mail-Adresse: [hendrik.janert@haw-hamburg.de](mailto:hendrik.janert@haw-hamburg.de)
* Tobias Schweisfurth  
  E-Mail-Adresse: [tobias.schweisfurth@gmail.com](mailto:tobias.schweisfurth@gmail.com)

### Projektleiter

Tobias Schweisfurth

# Produktvision - Ziel

*Upside Down* ist ein Spiel, in welchem man auf Zeit basierend geschickt durch ein Labyrinth navigieren soll. Besonders steht dabei die Veränderung der Steuerung je nach Blickwinkel im Vordergrund. Das Spiel soll zuerst für den PC programmiert werden.

Die Welt ist minimalistisch gehalten und wird mit zweidimensionalen Figuren arbeiten. Die Hauptfigur ist dabei ein Quadrat.

Die Hauptfigur bereist während des Spiels eines Planeten, welcher anders als man ihn gewöhnlich kennt, nicht kreisrund, sondern viereckig ist. Die Oberfläche des Planeten ist von unterschiedlichen Hindernissen geprägt. Der Hauptcharakter versucht dabei den Planeten mehrmals zu umrunden, wobei sich die Landschaft dabei immer wieder ändert. Neue Hindernisse erscheinen und die Bewegung des Charakters verändert sich äquivalent zur Gravitationsrichtung zum Mittelpunkt des Planeten.\*[[1]](#footnote-1)  
Das Spiel ist an sich endlos. Stirbt der Charakter oder startet der Spieler das Spiel neu, so beginnt er erneut vom Anfang.

Das Spiel soll auf dem Desktop laufen, möglicherweise aber auch auf Android. Für die Steuerung werden Tastatur und Maus benutzt.

# Umsetzung

Die Programmierung findet in Java statt. Dafür wird das Framework LibGDX benutzt. Zur Versionskontrolle wird Git benutzt. Zur Aufzeichnung der Anforderungen wird das Liste-Programm Trello ([www.trello.com](http://www.trello.com)) benutzt.

# Merkmal

Das größte Merkmal des Spiels ist die Einzigartigkeit der Anpassung der Steuerung relativ zur Oberfläche. Kurz gesagt, bedeutet das, dass die Steuerung des Charakters über die Pfeiltasten oder WASD-Tasten sich verändert, wenn die Oberfläche nicht mehr horizontal ist, sondern vertikal und dann abhängig davon, in welche Richtung sich der Charakter bewegen muss. Man kann sich hier ein Rechteck vorstellen mit vier Seiten: Um dieses Rechteck zu bereisen, läuft man den Umfang des Rechtecks ab und somit 4 verschiedene Seiten. Und jede Seite hat ihre individuelle Steuerung. Grundsätzlich bewegt sich der Charakter im Uhrzeigersinn um den Planeten, später könnten aber auch noch Alternativen eingebaut werden. Zu diesen Alternativen zählen:

* Bewegung des Charakters gegen den Uhrzeigersinn um den Planeten
* Veränderung der Gravitation des Planeten (größere/kleinere Sprünge möglich)
* Invertierung der Fläche, auf der man läuft. (Nicht mehr nur außen um das Rechteck herum, sondern innerhalb des Rechtecks)
* Mögliche Gegner
* Mögliche Rätsel über das Betätigen von Schaltern.

# Design

Das Design soll schlicht gehalten werden. Es handelt sich um ein Spiel, welches in 2D funktioniert. Demnach wird mit den einfachsten geometrischen Formen gearbeitet, die man gut über Ränder und ihre individuellen Farben unterscheiden kann. Landschaften sollen entweder landschaftstypische Farben erhalten (grün für Gras, braun für Berge und Landschaften) oder aber extra spezielle Design erhalten, die an das Außerirdische erinnern (blaues Gras, komplett mit Metall ummantelte Oberfläche wie in SciFi).  
Alternativ erscheinen die Welten in einem eher abstrakteren Aussehen, mit einfachen Formen und knalligen Farben.

Der Hauptcharakter ist ein Quadrat – möglicherweise mit einem Gesicht. Die Farbe sollte sich grundsätzlich immer stark von der Landschaft abheben, sodass der Spieler jederzeit den Überblick hat. Im Sprung sollte sich das Quadrat drehen, je nachdem wie schnell man war, bevor man abgesprungen ist.

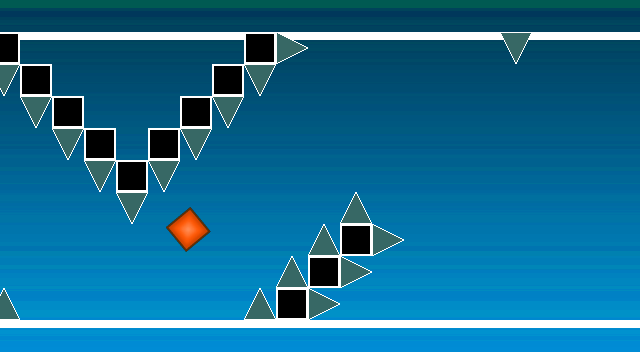


Abbildung Aussehen dargestellt anhand "The Impossible Game"

Einfache geometrische Formen wie Dreiecke können dem Spieler leicht symbolisieren, dass er hieran zerplatzt bzw. sich hier ein Hindernis befindet, wohingegen Quadrate und Rechtecke ein Hindernis symbolisieren, welches nicht gefährlich ist, aber umgangen werden muss.

# Zeitplan

**Phase 0:** Die Projektbearbeitung soll mit der Einarbeitung in die gewünschten Frameworks beginnen. Dabei sollen bereits bestehende Projekte ausprobiert und daran experimentiert werden.

**Phase 1:** Danach liegt der Fokus auf den drei grundsätzlich wichtigsten Punkten: Steuerung des Charakters, Steuerung der Kamera und deren Verknüpfung aufgrund ihrer Abhängigkeit. Mit diesen Meilensteinen soll der Grundbaustein für die Weiterarbeit gesetzt sein, da ab diesem Punkt das Level- und Game-Design im Vordergrund steht. Gerade da es sich hier um die Basis des Spiels handelt, sollte hierauf auch am meisten Arbeitszeit investiert werden.

**Phase 2:** Nachdem man diese Grundbasis gesetzt hat, soll das Leveldesign im Vordergrund stehen. Ab hier rückt der Einsatz von Texturen und Playtesting in den Vordergrund, um verschiedene Design-Entscheidungen zu überarbeiten. Gerade weil diese Arbeit so viel Testing und kleinere Anpassungen benötigt, sollten hier eine große Zahl an Personenstunden angenommen werden.

**Phase 3:** Somit steht zwar das eigentliche Spiel, jedoch müssen noch alle Dinge für den Nutzer gemäß angepasst werden. Dafür soll ein Interface im Spiel und die Übergange von Startbildschirm zu Level-Auswahl zu eigentlichem Spiel erstellt werden.

**Phase 3.5:** Zwar findet Bugfixing über die ganze Projektarbeit hinweg statt, soll aber am Ende noch einmal gesondert in den Fokus geraten.

Wenn man von einer Projektbearbeitungszeit von 3 Monaten ausgeht, lassen sich die Phasen ungefähr wie folgt einschätzen: Phase 0 - 2 Wochen, Phase 1 – 3 Wochen, Phase 2 – 4 bis 5 Wochen, Phase 3 – 2 bis 3 Wochen.

## Aufwandseinschätzung – Wichtigste Teilaufgaben

Die Einarbeitungszeit in die verschiedenen Frameworks wird auf ungefähr 20 Personenstunden geschätzt.

|  |  |
| --- | --- |
| Teilaufgaben | Aufwand in Personenstunden |
| Steuerung und Veränderung der Steuerung des Charakters | 20 |
| Steuerung der Kamera | 10 |
| Verknüpfung der Steuerung des Charakters mit Steuerung der Kamera in der Spielwelt | 20 |
| Erstellen von Hindernissen in der Spielwelt nach System | 15 |
| Erstellen der Level bzw. Ablauf eines Levels | 40 |
| Erstellen eines HUD | 10 |
| Übergänge einzelner Gamestates – Zusammenführung des Spiels | 5 |
| Bugfixing | 20 |

# Initialisierte Versionskontrolle

Der Projektplan und zum Projekt gehöriger Quellcode findet sich in einem Repository auf github.com unter folgendem Link: https://github.com/Arcraith/ProjektB

1. \* Genauere Erklärung und Absatz Merkmal [↑](#footnote-ref-1)