Otto-von-Guericke-Universität Magdeburg
Fakultät für Informatik
Acagamics e.V.

Technical Design Document

Veil of Death

Autoren:
Christoph Dollase
(Team leader)

Lars Wagner

Alexander Heck

8. September 2017

Betreuer:

Gerd Schmidt
Stefan Schwarz
Acagamics e.V.

Prof. Dr.-Ing. Holger Theisel
Institut für Simulation und Graphik

Inhaltsverzeichnis

1 Einleitung **4**
- 1.1 Spielprinzip . . . 4
- 1.2 Zielsetzung . . . 4
- 1.3 Projektumfeld . . . 4

2 Projektplanung (in Arbeit - Chris) **5**
- 2.1 Projektphasen (in Arbeit - Chris) . . . 5
- 2.2 Ressourcenplanung (in Arbeit - Chris) . . . 5
- 2.3 Entwicklungsprozess (in Arbeit - Chris) . . . 5

3 Das Spiel **6**
- 3.1 Hauptmenü . . . 6
  - 3.1.1 Spiel Starten . . . 6
  - 3.1.2 Einstellungen . . . 6
  - 3.1.3 Statistiken . . . 6
  - 3.1.4 Credits . . . 6
  - 3.1.5 Sonstiges . . . 6
- 3.2 Spielwelt . . . 6
  - 3.2.1 Levelgenerator (in Arbeit - Chris) . . . 6
  - 3.2.2 Player . . . 6
  - 3.2.3 Fallen (in Arbeit - Lars) . . . 6
  - 3.2.4 Feedback . . . 6
  - 3.2.5 Assets . . . 6

4 Technische Umsetzung **7**
- 4.1 Codestruktur . . . 7
  - 4.1.1 GameStates . . . 7
  - 4.1.2 Globale Klassen . . . 7
- 4.2 InGame . . . 7
  - 4.2.1 Movement . . . 7
  - 4.2.2 Collision (QA - Lars) . . . 7
  - 4.2.3 Sontiges . . . 9
- 4.3 GUI . . . 9
  - 4.3.1 Panel System (in Arbeit - Chris) . . . 9
- 4.4 Rendering . . . 9
  - 4.4.1 Aufbau . . . 9
  - 4.4.2 Weltkoordinaten . . . 9
  - 4.4.3 Animation (QA - Lars) . . . 9
  - 4.4.4 Partikel Effekte . . . 10
- 4.5 Statistik . . . 10

5 Projektverlauf **11**
5.1 Prototypen 11
5.1.1 Prototyp 1 - Alexander Heck und Robert Jendersie 11
5.1.2 Prototyp 2 - Christoph Dollase und ??? 11
5.1.3 Prototyp 3 - Lars Wagner und Mattis Hagen (QA - Lars) 11
5.2 Meilensteine 12
5.2.1 MS I 12
5.2.2 MS II 12
5.2.3 MS III 12
5.2.4 MS IV 12
5.2.5 Abgabe 12

6 Fazit **13**
6.1 Soll - Ist - Vergleich 13
6.2 Lessons Learned 13
6.3 Ausblick 13

1 Einleitung

1.1 Spielprinzip

1.2 Zielsetzung

1.3 Projektumfeld

2 Projektplanung (in Arbeit - Chris)

2.1 Projektphasen (in Arbeit - Chris)

2.2 Ressourcenplanung (in Arbeit - Chris)

2.3 Entwicklungsprozess (in Arbeit - Chris)

3 Das Spiel

3.1 Hauptmenü

3.1.1 Spiel Starten

3.1.2 Einstellungen

3.1.3 Statistiken

3.1.4 Credits

3.1.5 Sonstiges

3.2 Spielwelt

3.2.1 Levelgenerator (in Arbeit - Chris)

3.2.2 Player

3.2.3 Fallen (in Arbeit - Lars)

3.2.4 Feedback

3.2.5 Assets

4 Technische Umsetzung

4.1 Codestruktur

- siehe TDD

4.1.1 GameStates

- alle mal kurz anreißen

4.1.2 Globale Klassen

- GameConstants
- Gamemanager (siehe TDD)

4.2 InGame

4.2.1 Movement

- Lane prinzip
- Jump-Funktion (Parabel anhand von Geschwindigkeit)

4.2.2 Collision (QA - Lars)

In unserem Spiel haben wir zwei unterschiedliche Kollisionmethoden verwendet. Zum Einen Arbeiten wir mit Gridpositions für statische Objekte und mit den Axis-Alligned-Bounding-Boxes für bewegbare Objekte.

Gridpositions eignen sich hervorragend für statische Objekte in Spielen, daher

werden die Kollisionen von Münzen und Slowtraps von ihnen verwaltet. Hier für werden die Positionen von unseren kollidirbaren Objekten abgespeichert. Die Kollision wird dann mit Hilfe von if-Bedingungen berechnet.

Axis-Alligned-Bounding-Boxes berechnen für uns die Kollisionen mit beweglichen Obejkten, so dass jedes Objekt seine eigene Bounding Box hat. Hier für wird um das Objekt eine Hülle gespannt. In unserem Spiel haben wir die jeweiligen Maximalwerte der Modelle in Betrachtung der Achsen gewählt. Die Kollision entsteht wenn die Minimalwerte der Achsen von Körper A kleiner als die Maximalwerte von Körper B sind. Sowie die Maximalwerte der Achsen von Körper A größer sind als die Minimalwerte der Achsen von Körper B. Anders gesagt wenn die erweiterte konvexe Hülle des einen Körpers, die des Anderen schneidet.

4.2.3 Sontiges

4.3 GUI

4.3.1 Panel System (in Arbeit - Chris)

4.4 Rendering

4.4.1 Aufbau

4.4.2 Weltkoordinaten

4.4.3 Animation (QA - Lars)

Die Animation der Modelle erfolgt unter den Gebrauch einer Bibliothek. Hierfür benötigt man eine .x-Datei. Des Weiteren wird eine Textdatei benötigt, in der festgehalten ist, wie das Modell animiert werden soll. In dieser Textdatei kann man auf das jeweilige Modell bezogen so viele Methoden (Animationparts) erstellen wie man möchte. Man kann entscheiden, ob die Animation wiederholt werden soll oder einfach gespielt wird. Außerdem wird hier das Start- und Endframe der Animation festgelegt. Auch die Animationsdauer ist eine weitere Einstellungsoption. Nach dem man die jeweiligen Einstellungen für seine Methoden festgelegt hat, kann man diese im späteren Code unter ihrem angegebenen Namen aufrufen. Bevor man diese jedoch aufrufen kann, muss die Textdatei gelesen werden. Es gilt weiterhin zu beachten, dass nur eine Animations gleichzeitig abgespielt werden kann. Darunter ist zu verstehen, dass es nicht möglich ist die Animation in mehrere Schichten aufzuteilen, sodass es möglich ist mehrere Animationen abzuspielen aber nur Eine sichtbar zu zeichnen, während die Anderen im Hintergrund weiterlaufen. Diese Option würde den Übergang zwi-

schen unterschiedlichen Animationen vereinfachen, da man die Zweite einfach auf transparent stellen würde.

4.4.4 Partikel Effekte

4.5 Statistik

5 Projektverlauf

5.1 Prototypen

5.1.1 Prototyp 1 - Alexander Heck und Robert Jendersie

5.1.2 Prototyp 2 - Christoph Dollase und ???

5.1.3 Prototyp 3 - Lars Wagner und Mattis Hagen (QA - Lars)

Unser Prototyp war ein Spaceshooter indem man ein Raumschiff steuern konnte, welches Meteoriten ausweichen musste. Es ist auch möglich gewesen die Meteoriten abzuschießen. Für den Abschuss der Meteoriten hat man Punkte gekriegt. Das Spiel endet sobald das Raumschiff keine Trefferpunkte mehr hat, welche man verliert wenn man mit den Meteoriten kollidiert. Ziel des Spiels ist es solange wie möglich zu überleben und seinen Highscore zu maximieren.

Das Spiel basiert auf einem Grid-System, da die Meteoriten nur in einem Grid aus festgelegten Punkten spawnen können. Die Kamera ist starr fixiert und lässt sich nur zu Debugzwecken bewegen. Die Meteoriten rotieren um eine zufällig gewählte Achse während sie sich dem Spieler nähern. Mit der Zeit nimmt die Geschwindigkeit der Meteoriten zu. Unsere Laserstrahlen sind Punkte die wir dann als separate von einander getrennte Linien zeichnen lassen. Auch wurde der Spieler über die GUI über seinen Score, seine verbliebenen Trefferpunkte informiert. Die Asteroiden werden durch den MeteorHandler verwaltet. Dadurch sind nie mehr als 20 Meteoriten auf dem Bildschirm, sobald ein Meteor hinter der Kamera despawned, spawned sofort ein Neuer.

5.2 Meilensteine

5.2.1 MS I

5.2.2 MS II

5.2.3 MS III

5.2.4 MS IV

5.2.5 Abgabe

6 Fazit

6.1 Soll - Ist - Vergleich

6.2 Lessons Learned

6.3 Ausblick