Gewerbeschule Lörrach

Projektarbeit

C. Schmidt

Zeitraum  
23.11.2023 – 08.04.2025



**Dungeon Adventure  
Escape JOSCHUA**

Vorgelegt von:

Nico Brauer & Joschua Stammherr

[BrauerNic@mail.gws-loerrach.de](mailto:BrauerNic@mail.gws-loerrach.de)  
[StammherrJos@mail.gws-loerrach.de](mailto:StammherrJos@mail.gws-loerrach.de)

Abgabedatum: 08.04.2025

**Inhaltsverzeichnis**

1. Vorwort…………………………………………………………………………... 2

2. Konzeptionierung……………………………………………………………….. 3

2.1. Spieldetails…………………………………………………………………. 3

2.2. Aufgabenteilung……………………………………………………………. 3

2.3. Programmierung…………………………………………………………… 3

2.4. Grafikdesign………………………………………………………………... 3

3. Spielgeschehen…………………………………………………………………. 4

4. Komplikationen & Lösungen…………………………………………………… 4

4.1 Programmiertechnisch…………………………………………………….. 4

4.2 Graphisch…………………………………………………………………… 5

5. Fazit

6. Quellenverzeichnis

7. Anlagen

**Vorwort**

Die vorliegende Dokumentation entstand im Rahmen der Abschlussprüfung für die Fachhochschulreife am Berufskolleg für Informationstechnik der Gewerbeschule Lörrach.

Die gesamte Dokumentation wurde in der dritten Person verfasst, um es dem Leser zu vereinfachen, die Aufgaben und Perspektiven beider Teammitglieder zu verstehen.

Bevor die Entscheidung zwischen einer Cloud mittels eines Raspberry Pi und einem in Java programmierten Videospiel getroffen wurde, war es notwendig zwischen Einzel- oder Partnerarbeit zu wählen.  
Auf Grund unserer bisherigen Programmiererfahrung und der mangelnden Kenntnisse in Datenbank- und Serversystemen, entschieden wir uns für die Programmierung eines Spieles, entwickelt mit der objektorientierten Programmiersprache Java.  
Zu der Zusammenarbeit, anstatt der Einzelarbeit, entschieden wir uns, da wir uns einerseits sehr gut miteinander verstehen, aber andererseits auch gegenseitig zurechtweisen und unterordnen können. Somit bilden wir das perfekte Team.

Das Projekt ist ein 2D Dungeon Adventure, in dem der Spieler drei Rätsel lösen muss. Jedes dieser Rätsel findet in einem eigenen Raum statt, aus denen, durch Lösen des Rätsels, ein Signal im Hauptraum aktiviert wird. Die Grafiken dieses Spieles wurden per Pixel Art Technik selbst gezeichnet. Durch das Lösen aller Rätsel, kann das Spiel beendet werden.

**Konzeptionierung**

**Spieldetails**

Anfangs wurde sich ein bis zwei Mal pro Woche getroffen, um die Einzelheiten des Spieles auszuarbeiten. Vieles wurde aufgeschrieben und dann doch wieder verworfen, bis ein Spielkonzept zustande kam, welches beiden gefiel. Auch außerhalb der Treffen fand mittels WhatsApp ein Informationsaustausch statt, durch den Manche Ideen sofortig verworfen oder weiterentwickelt werden konnten, um sie bei den Treffen zu Ende zu führen. Als die Details fertig besprochen wurden, wurde die Aufgabenteilung festgelegt.

**Aufgabenteilung**

Wird sich für eine Partnerarbeit entschieden, ist es gerade bei der Entwicklung eines Spieles wichtig, eine sinnvolle Aufgabenteilung festzulegen, anstatt beständig an derselben Aufgabe zu arbeiten, um sich nicht gegenseitig zu behindern. Da sich schon in der Ideenfindung herausstellte, dass Joschua sehr kreativ und Nico eher programmierbegabt ist, wurde entschieden, dass sich Joschua um das Erstellen der Grafiken und sich Nico um das Programmieren kümmern wird.

**Programmierung**

Bevor der Programmierprozess begann, stand die Idee im Raum, mit  
UML-Klassendiagrammen zu starten, um strukturierter arbeiten zu können. Allerdings äußerte sich Nico dagegen, da er bezüglich der Attribute und Operationen eher spontan arbeite. Die genannten Klassendiagramme sind dennoch in den Anlagen vorzufinden.  
Um immer mit dem Fortschritt der letzten Sitzung vertraut zu sein, führte Nico ein Ergebnisprotokoll zu jeder Arbeit am Projekt, welches ebenfalls Beschwernisse und offene Fragen umfasst.

**Grafikdesign**

Joschua arbeitet mittels eines iPad und eines Apple Pencil. Er brachte sich selbst die Pixel Art Technik bei, um die Grafiken für das Spiel zu entwickeln. Parallel notierte sich auch Joschua auf welche Komplikationen er stieß.  
Um die Grafiken in das Spiel zu implementieren, sendete Joschua diese nach Fertigstellung per AirDrop (Apples Drahtlose Übertragungstechnik) an Nicos iPhone, welche Nico dann per Mail an sich selbst sendete, um sie Schlussendlich am Rechner in das Spiel zu implementieren.

**Spielgeschehen**

Klassisch für ein Dungeon Adventure, spielt dich das Geschehen in unterirdischen Höhlen ab. Der Spieler startet im Hauptraum, von dem er die Möglichkeit hat, jeden weiteren Raum zu betreten. Es existieren drei weitere Räume, welche jeweils ein Rätsel enthalten. In einem Raum hat der Spieler die Aufgabe zwei von vier Krügen auf zwei Druckplatten zu stellen. Sind beide Druckplatten aktiviert, ist das Rätsel gelöst. Ein weiterer Raum enthält die Aufgabe durch ein Labyrinth zu finden, während nur ein kleiner Radius um den Spieler sichtbar ist, um am Ende einen Schalter zu bedienen. Der letzte Raum fordert den Spieler dazu auf einen Ball in einen Kasten zu tragen. Allerdings wird ihm hierbei der Weg von einem  
NPC (=Non-Playable Character) erschwert, der sich mit jeder Spielerbewegung in eine zufällige Richtung verschiebt. Kommt der Spieler mit dem NPC in Berührung, wird der Raum zurückgesetzt. Der Spieler, der NPC und der Ball werden somit wieder auf ihre Startposition innerhalb des Raumes gesetzt. Der NPC wird mittels eines Bildes von Joschua dargestellt, woraus sich ebenfalls der Spieltitel ableitet. Jeder Raum kann ohne Voraussetzung betreten werden, was in einer flexiblen Rheinfolgenauswahl resultiert.  
Jedes abgeschlossene Rätsel lässt ein Signal im Hauptraum aufleuchten. Durch das Lösen aller Rätsel kann das Spiel beendet werden.

**Komplikationen & Lösungen**

**Programmiertechnisch**

Nico hat zwar gute Java Kenntnisse und stieß daher selten auf Syntaxbezogene Hürden, jedoch mangelt es ihm noch an reeller Projekterfahrung, weshalb es ihm nicht leichtfiel, ein funktionierendes Levelsystem zu entwerfen. Zuerst sollte bei Durchschreiten einer Tür eine Variable hochgezählt werden und je nach Wert das nächste Level geladen werden. Da das vorliegende Projekt allerdings nicht aus klassischen Leveln, sondern Räumen besteht, die immer wieder betreten und verlassen werden können, war eine andere Lösung von Nöten. Nach zwei Wöchigem überlegen und praktischem ausprobieren, fand Nico eine Lösung. Jede Tür hat einen eigenen Wert im Map-Array\*, welcher einem Raum zugewiesen ist. So wird, abhängig von der betretenen Tür, der entsprechende Raum geladen.

Weiter trat ein Problem mit dem NPC auf. Da eine zufällige Richtungsauswahl ebenfalls in einer nicht begehbaren Richtung resultieren konnte, zeigte der NPC gelegentlich keine Reaktion. Bevor Nico nach einer Lösung suchte, besprach er die Situation mit Joschua und sie überlegten, ob eine Lösung tatsächlich nötig sei. Nico und Joschua entschieden sich diesen Spielfehler zu beheben, also verschob Nico den Code zur Richtungsauswahl in eine Kopfgesteuerte Schleife, die endet sobald eine verfügbare Richtung festgelegt wurde.

Nun wurde im praktischen Programmierunterricht das Thema Vererbung vermittelt. Die Vererbung dient der Codereduzierung und Übersichtlichkeit, indem mehrere Klassen auf Attribute und Operationen der sogenannten Oberklasse zugreifen können. Ein Programm kann somit zeiteffizient fertiggestellt werden.

Jedoch befand sich das Projekt bereits in einem fortgeschrittenen Stadium, sodass die Vererbungstechnik, abgesehen von der Übersichtlichkeit, zunächst keinen direkten Nutzen aufgewiesen hätte. Doch aufgrund der verbesserten Übersichtlichkeit und des langfristigen Nutzens, entschied sich Nico, die Vererbung dennoch einzubinden. Hierzu wurde eine neue Oberklasse definiert, und die entsprechenden Unterklassen bestimmt. Etwa 140 Zeilen redundanter Code konnten entfernt werden, da ihre Funktionalität nun in der Oberklasse bereitgestellt wurde.

Auch war es Nico nicht umgehend begreiflich, wie die Dunkelheit des Darkrooms in das Spiel implementiert werden sollte. Hier schuf erneut der Austausch mit Joschua und eines weiteren Klassenkameraden Abhilfe. Ein schwarzes Bild, welches eine mittige Aussparung besitzt, wird über den Spieler gezeichnet und folgt jedem seiner Schritte. Das Bild muss hierfür größer als das Spielfenster sein und die Aussparung über dem Spieler liegen.

**Graphisch**

* Raster
* Programm abgestürzt, Fortschritt nicht gespeichert