Humboldt Schule Bad Homburg

Q1 Informatik GK Scheuring

**Dokumentation zum Textadventure**

Angefertigt von: Maša Paskov, Fabian Ruttmann

Datum: 06.01.2022

Kurs: INFO 021 Scheuring

**Inhalt**

1. a) Beschreibung der Spielsituation

b) des Spiels

c) der Erweiterungen

2. Raumübersicht

3. Screenshots

4. Klassendiagramm

**1.**

**a)** Der Spieler beginnt das Abenteuer in einem Raum mit einem Kamin und einem Weihnachtsbaum. Die Spielerfigur ist eine Helferin des Weihnachtsmanns und bereitet die Geschenke für die Kinder in aller Welt vor. Sie ist eine magische Weihnachtspuppe namens Carmen. Nach einem Monolog im ersten Raum erfährt der Spieler von einem faulen Elfen, der sich seltsam verhält und seine Arbeit nicht ordnungsgemäß ausführt. Sie beschwert sich über seine angeblich erfundene Verschwörung über einen grünen Mann, der Weihnachten zerstören wolle.

Carmen beschließt, die Arbeit der Elfen zu überprüfen, und der Spieler wird angewiesen, den nächsten Raum zu betreten. Nachdem der Spieler den Raum jedoch betritt, sieht er, dass er bis auf einen einsamen Elfen, der in der Mitte des Raumes steht, leer ist. Wenn man sich dem Elfen nähert, wird ein Gespräch ausgelöst und Carmen erfährt, dass ein grüner Mann die Geschenke gestohlen und die Elfen hypnotisiert hat, die sich daraufhin gegen ihn wandten. Daraufhin begibt sich Carmen auf ein Abenteuer kämpft gegen die Elfen.

Wenn der Spieler den letzten Raum erreicht, kommt es zu einem Bosskampf, und nachdem er das grüne Monster besiegt hat, stellt sich heraus, dass es die ganze Zeit der Weihnachtsmann war, nachdem der Boss seine Maske abgenommen hat. Er plante diesen Angriff auf Weihnachten, weil er sich von den Kindern nicht geschätzt fühlte und das Gefühl hatte, dass Weihnachten seine Bedeutung verloren hatte. Für den Weihnachtsmann geht es an Weihnachten um Liebe und Wertschätzung füreinander und nicht nur um Geschenke. Nach seiner Niederlage erkennt der Weihnachtsmann, dass es bessere Wege gibt, Weihnachten zu retten, und verspricht, mit Carmen zusammenzuarbeiten, um die Menschen daran zu erinnern, wofür Weihnachten wirklich steht.

**b)** Das Spiel besteht aus einigen zentralen Komponenten, von denen einige dem Spieler erst ab einem bestimmten Punkt in der Geschichte zugänglich sind. Von Anfang an kann der Spieler gehen und springen. Der Spieler kann nur einmal springen, und bis die Spielfigur nach dem Sprung den Boden berührt, ist es unmöglich, erneut zu springen.

Im Laufe des Spiels kann der Spieler der Handlung folgen, indem er den Text liest, der während bestimmter zeitlich festgelegter Spielabschnitte angezeigt wird, die entweder von selbst nach dem Betreten eines Raums oder dem Starten des Spiels auftreten oder beim Sprechen mit anderen Charakteren aktiviert werden können.

Nach dem Erreichen des zweiten Raums steht dem Spieler eine neue Methode zur Verfügung: das Angreifen. Sie kann freigeschaltet werden, nachdem man die Zuckerstangenwaffe eingesammelt hat, die dem Inventar des Spielers dauerhaft eine Waffe hinzufügt und die grafischen Texturen der Spielfigur verändert. Um einen Feind angreifen zu können, muss der Spieler sich nah genug in seiner Nähe befinden und seinen Angriff perfekt zeitlich anpassen.

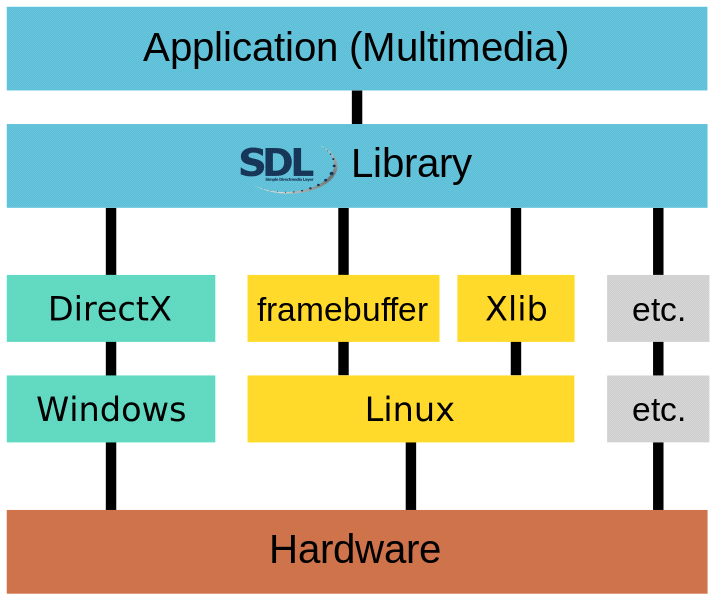
Es gibt 2 verschiedene Arten von Gegnern, die jeweils ihre eigenen Angriffsmuster und Verhaltensweisen haben:

1) Die Elfen sind die erste Art von Gegnern, auf die man treffen kann. Sie verfügen über ein Gegnererkennungssystem, das den Spieler in der Nähe erkennen kann, was sie dazu bewegt, auf den Spieler zuzugehen und anzugreifen. Obwohl sie nur angreifen können, wenn der Spieler in ihrer Nähe ist, haben sie einen größeren Angriffsbereich als der Spieler. Um dies richtig auszugleichen, sind die Elfen nicht in der Lage zu springen, so dass man ihren Angriffen mit zeitlich angepassten Sprüngen ausweichen muss.

2) Der zweite Gegner, auf den der Spieler treffen kann, ist der Endgegner, der im letzten Raum des Spiels zu finden ist. Im Gegensatz zu den Elfen verfügt er über eine Pistole, die ihn mit einer Fernkampfmechanik ausstattet.

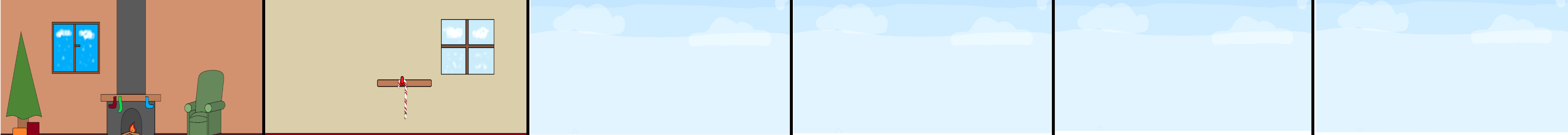
**c)** SDL2:

SLD2 ist eine Bibliothek zur Entwicklung leistungsfähiger graphischer Anwendungen für viele verschiedene Plattformen (z.B. Windows oder Linux). Intern übersetzt SDL2 Befehle in die plattformspezifischen Befehle. SDL ist in C geschrieben und funktioniert ursprünglich mit C++. Es gibt Verknüpfungen für verschiedene Sprachen, einschließlich C#, Python und Pascal.



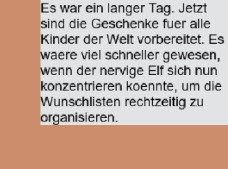
<https://www.freepascal-meets-sdl.net/chapter-1-introduction/>

Da das Grafik-Framework von Lazarus nicht mit dem Fokus auf Spielentwicklung entwickelt wurde, haben wir uns für SDL entschieden, das speziell für die Spielentwicklung und schnelles Rendering entwickelt wurde, welches die Implementierung von Bewegungsanimationen mit verschiedenen Einzelbildern, die einen Bewegungsablauf ergeben erleichtert hat. Darüber hinaus war es besonders nützlich, wenn es darum ging, neue Räumen- und Gegnertexturen schnell zu rendern.

**2.** Die ersten beiden Räume befinden sich in einem Haus und dienen als Einführung in das Spiel. Die anderen vier Räume befinden sich draußen im winterlichen Nordpol, wo der Spieler gegen andere Gegner kämpfen muss.

**3.**

****Der Hauptcharakter im Sprung.

**** Text-Widget in der oberen rechten Ecke zum Erklären zusätzlicher Informationen und zum Implementieren eines einfachen Dialogs

Zusätzlich: Video in Anhang A.

**4.**

Siehe Anhang B.

**Anmerkung**

Das Spiel ist noch nicht fertig. Es ist kein fertiges Kampfsystem implementiert, und es fehlen Gegner und tatsächliche Interaktionen. Allerdings sind die Klassen und Methoden für diese Funktionalitäten bereits zum Teil ausgearbeitet, und das Spiel lässt sich aufgrund des modularen Aufbaus der Klassen, der, so weit wie möglich verallgemeinert, eine gute Grundlage für weitere Gegnerklassen und das Einführen neuer Features bietet, sehr gut erweitern. Besonders der konsistente Aufbau der einzelnen Komponenten oder Widgets (z.B. *TRoomDisplay* oder *TPlayerDisplay*) mit jeweiligen *draw()*-Methoden zum Rendern der jeweiligen Instanz, während die Darstellungslogik in der Klasse selbst abgetrennt ist, und die meist konsequente Trennung von Datenobjekt (z.B. *TRoom*) und Datendarstellung (z.B. *TRoomDisplay*), sorgen dafür, dass neue Features komfortabel implementiert werden können.

Allerdings haben wir uns dafür entschieden, mit dem Ziel des Erkenntnisgewinns und der Freude am kreativen Prozess sowohl ein vollständig neues Grafik-Framework zu verwenden, als auch eine vollständige Bewegungslogik selbst zu implementieren sowie alle Texturen selbst zu gestalten. Daher blieb keine Zeit mehr für ein Implementieren aller vorgesehenen Features. Dafür bitten wir um Verständnis.

Das Git-Verwaltungssystem wurde für die Organisation benutzt.