

Escape-Nachhaltig und Sicher

Die Entwicklung eines Escape-Spieles zur Vermittlung von Selbstschutzmaßnahmen durch
Unterstützung des Kompasses der digitalen Selbstverteidigung

Mustermann, Max

Otto-von-Guericke Universität Magdeburg

Fakultät für Informatik

Magdeburg, Deutschland

INHALTSVERZEICHNIS

I	Einleitung	4
I-A	OER – Was steckt dahinter?	4
I-B	Aufgabenstellung	5
I-C	Interpretation der Aufgabenstellung	5
I-D	Ergebnisüberblick	5
I-E	Verwandte Arbeiten	6
II	Die Ist-Analyse des Escape-Spiels	6
III	Konzept des Softwareprojektes	9
III-A	Die Soll-Analyse des Projektes	10
III-B	Umsetzungsplanung	12
IV	Realisierung	12
IV-A	Funktionsweise von RenPy	12
IV-A1	Erstellung des Projektes	13
IV-A2	Öffnen des Projektes	13
IV-A3	Editieren des Projektes	13
IV-A4	Releasen des Projektes	14
IV-B	Inhaltliche Weiterentwicklung	14
V	Zusammenfassung	20
VI	Ausblick	21
VII	Anhang	23
VII-A	Lizenzen	23
VII-A1	Eigene Grafiken	23
VII-A2	Grafiken Dritter	23
VII-A3	RenPy	23
VII-A4	Musik und Sound-Effekte	23
VII-B	Softwareprojekt	23
Literaturverzeichnis		24

ABBILDUNGSVERZEICHNIS

1	Übersicht über eine mögliche Einteilung von OERs [1]	4
2	Startbildschirm des Cockpits	6
3	Startbildschirm des Maschinenraums	6
4	Gute Endszene	7
5	Schlechte Endszene	7
6	Anzeige einer Ja/Nein-Antwortmöglichkeit	7
7	Anzeige einer Auswahl-Antwortmöglichkeit	8
8	Anzeige einer Text-Antwortmöglichkeit	8
9	Startbildschirm des RenPy-Launchers	13
10	Beispielinstallationsordner von RenPy mit dem eingefügten Projekt (rot umrandet)	13
11	Projektordnerstruktur nach dem ersten Start des Atom-Editors	14
12	Struktur des script-Ordners	14
13	RenPy Fenster zum Build des Projektes	14
14	Ausgangsstruktur des script-Ordners	18
15	Ausschnitt des Variablen_Methoden_Info Textdokuments	19
16	Darstellung der App-Symbole mit der ganz rechts neu eingeführten Kommunikationsapp	19
17	Hinweissymbol	19
18	Hochfahrsequenz des Cockpit-PCs	20
19	Startszene des Easter Eggs	20
20	Startbildschirm des überarbeitenden Cockpits	21
21	Startbildschirm des überarbeitenden Maschinenraumes	21

I. EINLEITUNG

Die Weiterentwicklung der Technologie im 21. Jahrhundert ist unaufhaltsam. Regelmäßig und in immer kürzer werdenden Abständen werden neue Meilensteine gesetzt. Leistungsfähige Geräte wie Smartphones, Tablets aber auch High End PCs sind keine Seltenheit mehr in den Haushalten der Menschen. Immer mehr und neue Einsatzmöglichkeiten ergeben sich durch die gesteigerte Performance der Geräte. Jedoch hat dies auch einen erheblichen Nachteil. Die Cyberkriminalität steigt an. Was früher nur von wenigen privilegierten Bürgern möglich war, kann heute jeder erlernen und anwenden. Aus diesem Grund ist es wichtig, verantwortungsbewusst mit seinen Daten umzugehen. Das Escape-Spiel möchte sich der Aufgabe widmen, Nachhaltigkeit und Sicherheit spielerisch zu vermitteln.

Dieses Paper ist dabei wie folgt strukturiert:

Im folgenden Unterkapitel erkläre ich, was eine OER (Open Educational Resource) ist und welchen Zweck es dient. Im Kapitel I-B stelle ich die Aufgabenstellung für das Projekt dar. Das dritte Unterkapitel, Abschnitt I-C, dient für meine persönliche Interpretation der Aufgabenstellung. Im Kapitel I-D zeige ich auf, was ich letztendlich erarbeitet habe bzw. fertigstellen konnte. Der letzte Abschnitt I-E der Einleitung vermittelt noch einmal eine Übersicht, was RenPy überhaupt ist. Da das Projekt auf einem zuvor entwickelten Spiel basiert, stelle ich im Kapitel II den aktuellen Ist-Zustand des Programms vor. Dabei gehe ich auf die aktuellen Funktionen, die bestehenden Fehler, den allgemeinen Aufbau und das zugrundeliegende Konzept ein. Daraus abgeleitet, stelle ich meinen Plan für das Softwareprojekt (Kapitel III) vor. Im ersten Punkt III-A erkläre ich meine Projektziele und weshalb ich sie in dieser Reihenfolge bearbeitet habe. Anschließend schildere ich meine Umsetzungsplanung. Das IV. Kapitel befasst sich mit der Realisierung des Projektes. Im ersten Unterabschnitt IV-A gehe ich auf die Funktionsweise von RenPy ein und erkläre, wie man ein Projekt erstellen, öffnen, editieren und schlussendlich releasen bzw. veröffentlichen kann. Der zweite Punkt, Abschnitt IV-B, bearbeitet das Thema der inhaltlichen Weiterentwicklung. Dabei gehe ich auf die Frage ein, wie ich bestimmte Funktionen implementiert habe. Des Weiteren vergleiche ich Realisierungsmöglichkeiten für konkrete Anwendungsfälle. In Kapitel V fasse ich noch einmal das gesamte Projekt zusammen und stelle in Abschnitt VI zukünftige, bisher nicht implementierte Funktionen vor. Auch verschiedene Ansätze, das Spiel bekannter zu gestalten, werden in diesem Punkt behandelt.

A. OER – Was steckt dahinter?

OER, ausgeschrieben Open Educational Resources, sind Bildungsmaterialien jeglicher Art. Die Besonderheit ist, dass diese den Nutzer zur freien Verfügung stehen. Das digitale Lehr- oder Lernmaterial untersteht der offenen Lizenz. Einen kostenlosen Zugang sowie eine kostenlose Nutzung, Bearbeitung und Weiterverbreitung durch Dritte wird somit garantiert. Der Urheber

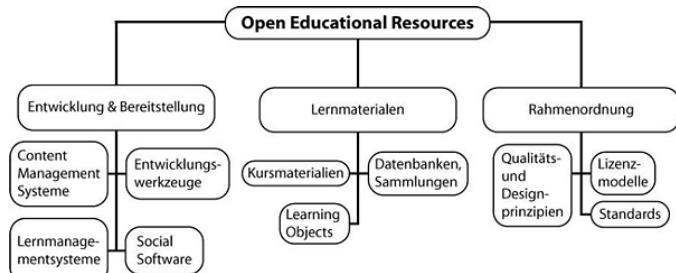


Abbildung 1: Übersicht über eine mögliche Einteilung von OERs [1]

entscheidet über die Nutzungsrechte, wobei im Allgemeinen für den Anwender keine oder nur sehr geringe Einschränkungen existieren. Einschränkungen könnten beispielsweise sein, dass der Urheber sich das Recht an seinen eigens kreierten Bildern oder Musikstücken sichert. OERs stehen der Öffentlichkeit frei zur Verfügung. Die bereitgestellten Materialien können so an die persönlichen Interessen angepasst und weitergegeben werden. Mögliche Formen von OERs sind beispielsweise Kurse, Kursanwendungen, Kursmodule sowie (Hyper-) Textdateien, Bilder, Audios, Videos, Simulationen und viele weitere. All jene sind für Menschen jeden Alters und aller denkbaren Interessengebiete angedacht.

Die Abbildung 1 veranschaulicht noch einmal eine mögliche Einteilung von OERs.

Die Entscheidung, das Escape-Spiel als OER bereitzustellen, hatte folgende Gründe:

- Kostenlose Nutzung möglich,
- Große Reichweite,
- Einfache, zugängliche Weiterbearbeitung,
- Das Vorhandensein einer umfangreichen Community und
- Das Ziel, Wissen frei zu vermitteln.

Es stellt sich die Frage, wie man eine Open Educational Ressource finden kann. Eine Möglichkeit bietet die Suchfunktion verschiedener Repositorien, wie beispielsweise GitHub, GitLab, Bitbucket, SourceForge und vielen Weiteren, an. Die Seite „It's FOSS“ [2] zeigt eine Übersicht weiterer Alternativen zu GitHub. Ebenso existieren spezielle Suchmaschinen wie „OERhörnchen“ für freie Bildungsmaterialien. Die klassischen Suchmaschinen (DuckDuckGo, MetaGer etc.) eignen sich gleichermaßen. Mit einem Filterkriterium der gewünschten Lizenz (in diesem Fall die freie Lizenz) lassen sich einfach und schnell OERs finden.

Das Erstellen einer OER lässt sich ähnlich wie andere Lehrmaterialien bewerkstelligen. Wichtig hierbei ist die Kennzeichnung und spezielle Lizenzierung als OER! Nur so wird das Projekt als solches auffind- und nutzbar. Bei der Erstellung und Bereitstellung von OER-Material ist es von großer Bedeutung, Quellen zu vermeiden, die nicht für die freie Nutzung gedacht sind. [3] [4]

B. Aufgabenstellung

Das vorliegende Projekt beruht auf dem Escape-Spiel aus dem Kurs „Schlüssel- und Methodenkompetenz der IT-Sicherheit“ im Sommersemester 2020/21. Die Aufgabenstellung war, ein Escape-Room Spiel zu konzipieren, zu erstellen, umzusetzen und zu evaluieren und als OER bereitzustellen. Das Ziel sollte sein, Selbstschutzmaßnahmen spielerisch zu vermitteln, sodass der Spieler die Auswirkungen bezüglich Sicherheit und Ressourcenverbrauch erkennt, versteht und schlussendlich auch lernt. Der Anwender soll sich ein sicheres, nachhaltiges und überlegtes Handeln aneignen. Zur Unterstützung bzw. als Werkzeug diente der Kompass der digitalen Sicherheit. Die aufgeführte Aufgabenstellung wurde für das Softwareprojekt übernommen und durch folgende Punkte erweitert:

- 1) Erstellung einer Dokumentation des gesamten Projektes,
- 2) Aufzeigen von Alternativen bei bestimmten Implementationen,
- 3) Anfertigung einer Beschreibung, wie mit dem RenPy Framework eine OER erstellt, geöffnet, weiterentwickelt und verfügbar gemacht werden kann,
- 4) Ausarbeitung bzw. Weiterentwicklung des vorhandenen Projektes,
- 5) Bereitstellung des überarbeitenden Projektes über eine GitLab Instanz der FIN bzw. OVGU.

C. Interpretation der Aufgabenstellung

Dieses Kapitel soll dazu dienen, wie ich die Aufgabenstellung interpretiert habe.

Das Softwareprojekt beruht auf dem bestehenden Escape Spiel. Aus diesem Grund soll auch die zu Beginn gestellte Aufgabenstellung für die nachfolgenden Implementationen gelten. Die Weiterentwicklung des Spieles bzw. die neuen, hinzugefügten Elemente müssen auch weiterhin die Themen Nachhaltigkeit und Sicherheit vermitteln. Ein verbessertes, immersives Spielgeschehen soll vorangetrieben und mögliche Fehler aus der vorherigen Version beseitigt werden. Um das Projekt für weitere Personen leichter zugänglich zu machen, ist eine umfangreiche Dokumentation des Projektes notwendig. Dies betrifft sowohl die Dokumentation im Code als auch die Erklärung der einzelnen Code-Dateien, des Spielkonzeptes und des RenPy Frameworks. Ein Vergleich der Alternativen in Form einer Tabelle dient zur Begründung für die jeweilige Auswahl. Die Einführung in RenPy ist mithilfe von Screenshots der entsprechenden Schritte und einer passenden Anmerkung einfach zu erklären. Dem Anwender kann so eine schnelle Einarbeitungszeit mit RenPy garantiert werden. Der letzte Punkt, die Bereitstellung des Projektes über GitLab, soll dem Benutzer ermöglichen, das Dokument herunterzuladen. Ebenso muss das Projekt so strukturiert bzw. beschriftet sein, dass sich der User in kürzester Zeit im Repository zurechtfindet.

Alles in allem dient die bestehende sowie erweiterte Aufgabenstellung dazu, ein Softwarelebenszyklus zu durchlaufen. Das heißt, dass das Projekt folgende Punkte chronologisch durchlaufen muss:

- 1) Durchführung einer Ist-Analyse des bestehenden Projektes, um zu verstehen, welche Funktionen bereits implementiert sind und wo es Fehler gibt bzw. geben kann
- 2) Erstellung eines Konzepts
 - a) Anfertigung einer Soll-Analyse, um die Realisierbar- und Sinnhaftigkeit einzelner Punkte für einen bestimmten Zeitraum abzuwägen
 - b) Planung der Umsetzung
 - Definition fester Abgabetermine für zielgerichteten Abschluss
 - Definition von Schnittstellen für gemeinsamen, auf das Team abgestimmten Workflow
- 3) Realisierung des Projektes zur Abarbeitung und Protokollierung der zusammengetragenen Punkte
- 4) Fertigstellung des Projektes
 - Zusammenfassung der absolvierten Projektziele
 - Verbesserung der Teamarbeit durch Diskussion aufgetretener Probleme und Vorstellung der Best Practices

Der nachfolgende Abschnitt, Kapitel I-D, stellt die resultierenden Ergebnisse aus meiner Interpretation der Aufgabenstellung dar.

D. Ergebnisüberblick

Der Ergebnisüberblick zählt alle Punkte auf, welche ich während des Projektes abschließen konnte. Dabei sind die einzelnen Teilziele chronologisch nach der Abarbeitung sortiert. Eine Begründung für diese Reihenfolge gebe ich in Kapitel III-A. Insgesamt konnte ich während meiner Bearbeitungszeit zehn Punkte abschließen. Der letzte Punkt, die Beseitigung von Fehlern, kann nur durch umfassende Analysen des Codes ansatzweise sichergestellt werden. Somit möchte ich diesen Punkt als teilweise erledigt gelten lassen.

Die nachfolgende Liste zählt die einzelnen Teilergebnisse chronologisch auf:

- 1) Neustrukturierung einzelner Codeteile sowie Hinzufügen und Neuanordnen von Programmdateien
- 2) Umfassende Dokumentation
 - des Codes mittels Kommentarfunktion,
 - von wichtigen Variablen und Funktionen innerhalb eines Textdokumentes,
 - des Ist-Zustandes des bestehenden Projektes,
 - des Konzepts des Softwareprojektes,
 - der Funktionsweise von RenPy,
 - der Implementation neuer Features und
 - der Best Practices während des Projektes
- 3) Einführen der Möglichkeit des parallelen Spielbetriebes
 - Implementierung eines neuen App-Symbols mit neuer Storysequenz
- 4) Hinzufügen von Zwischenaufgaben
 - Erstellung neuer Pixelgrafiken

- Erschaffung von Dialogen und Einführung neuer Sprecher
 - Hinzufügen von analogen und digitalen Rätseln
- 5) Verbesserung der Hinweis- und Feedbacktexte
- Erweitern der Texte, sodass diese mehr Informationen bieten
 - Hinzufügen von Hinweisen an bisher leeren Spielpassagen
- 6) Integration von aufgabenbezogenen und allgemeinen Hinweisen
- 7) Änderung bestimmter Bild- und Musikformate sowie Ergänzung von neuen Sounds
- 8) Einfügen neuer Effekte
- 9) Implementierung eines Easter Eggs
- 10) Einfügen von dynamischen Passwörtern
- 11) allgemeine Fehlerbeseitigung im Code als auch auf dem analogen Material

E. Verwandte Arbeiten

Das Softwareprojekt baut, wie oben bereits erwähnt, auf dem Projekt „Escape“ des Kurses „Schlüssel- und Methodenkompetenz der IT-Sicherheit“ auf. In diesem Abschnitt stelle ich das Konzept des Spieles vor und erläutere im anschließenden Kapitel den Ist-Zustand.

Die verwendete Entwicklungsumgebung ist RenPy, „eine Open-Source Spiel-Engine, die kostenlos für die Erstellung kommerzieller sowie nicht kommerzieller Spiele genutzt werden kann. Entwickelt wurde RenPy hauptsächlich für das Kreieren von Visual Novels. Unter diesem Begriff werden interaktive Bücher verstanden, in denen Spieler die Geschichte selbst durch Entscheidungen voranbringen. Neben den Visual Novels können mit RenPy allerdings auch Textadventures, Präsentationen und kleine Simulationsspiele verwirklicht werden.“ [5] Die Gründe der Entscheidung für RenPy waren folgende: „Zum einen ist die Engine schon seit 15 Jahren Open-Source verfügbar. Unter der MIT-Lizenz lässt sie sich kostenlos verwenden. Dabei kann RenPy bei Bedarf beliebig erweitert werden. Die Engine, deren Code frei einsehbar ist, enthält zudem kaum Bugs. [...] Zum Zweiten kann ein RenPy-Spiel plattformübergreifend und offline genutzt werden. Ergänzend zu Windows ist es unter macOS, Linux, Android und iOS spielbar. [...]“

Ein weiterer Vorteil von RenPy ergibt sich in der Entwicklung. Die Scriptsprache knüpft an der populären Programmiersprache Python an und ist einfach zu verstehen. Zusätzlich dazu lässt sie sich leicht anpassen. Mit relativ wenigen Zeilen können starke Effekte erzielt werden. Die Einbindung von Bildern oder Musik ist durch die vielen verschiedenen kompatiblen Formate unproblematisch. Die akzeptable Dokumentation sowie das sehr aktive Forum ermöglicht zudem eine Unterstützung der Nutzer.“ [5]

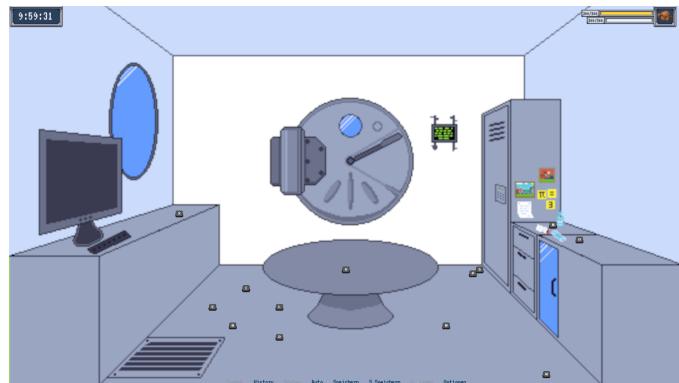


Abbildung 2: Startbildschirm des Cockpits

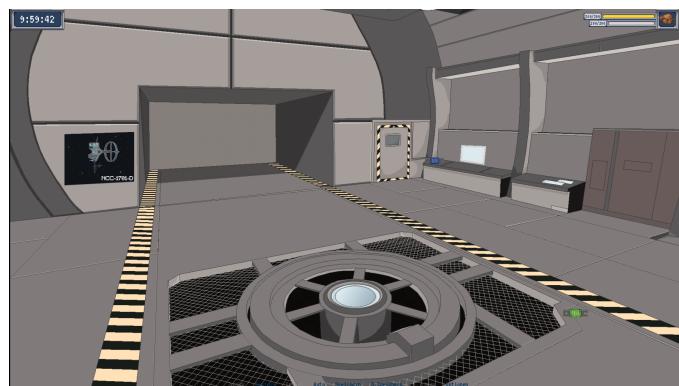


Abbildung 3: Startbildschirm des Maschinenraums

Eine Besonderheit des Projektes ist die hybride Umsetzungsart, welche sowohl analoge als auch digitale Elemente vereint. „Mit der digitalen Umsetzungsart gehen geringe Vorbereitungszeiten, sehr vielseitige Gestaltungsmöglichkeiten sowie verschiedene Methoden, Musik, Effekte und Bilder einfach einzubauen, einher. Diese Inhalte können dem Spiel mehr Atmosphäre geben. So ist es als vorteilhaft zu bewerten, dass bei der Implementation sowohl das Spiel als auch die Story frei gestaltet werden können. Für die analoge Umsetzungsart hingegen spricht die hervorragende Tiefe. Diese entsteht dadurch, dass etwas mit den eigenen Händen getan werden kann. [...] Die Entwicklung und Anwendung von Spielinhalten gestalte[t] sich einfach.“ [5]

II. DIE IST-ANALYSE DES ESCAPE-SPIELES

In diesem Kapitel analysiere ich das bereits bestehende Escape-Spiel. Dabei betrachte ich drei Themenfelder genauer. Das erste Thema behandelt den allgemeinen Aufbau sowie implementierte Funktionen und Anzeigemenu. Anschließend erkläre ich die Gründe für das Setting, die Darstellungsweise bestimmter Menüs usw., welche mein Team und ich gemeinsam getroffen haben. Das letzte Thema beschreibt die aktuell auftretenden Probleme während des Spielens.

Thema 1: Aufbau und Funktionen

Zu Beginn des Spieles muss sich der Anwender zwischen zwei Räumen entscheiden, dem Maschinen- und dem Cockpitraum. Abbildung 3 und 2 zeigen jeweils die Startszene. Beide Räume adressieren jeweils bestimmte Themengebiete des Kompasses der digitalen Selbstverteidigung. Insgesamt muss der Spieler sechs Hauträtsel lösen, um das Spiel zu gewinnen.

Dabei liegt der Fokus im Cockpit auf „die Auswahl und Konfiguration ressourcensparender Betriebssysteme und Internetbrowser sowie auf die Erkennung schlechter und die Vergabe starker Passwörter gelegt.“

Die Schwerpunkte beim Maschinenraum sind die Auswahl und Konfiguration von ressourcensparenden Suchmaschinen und Apps, die Anwendung von Open Source Office Programmen wie LibreOffice, die Einstellung benötigter Cookies und das Vorstellen von alternativen Messenger Diensten abseits von WhatsApp und Facebook.“ [5]

Eine detaillierte Darstellung der Inhalte der jeweiligen Aufgabe findet sich in dem Paper „Escape – Nachhaltig und Sicher“. [5]

Konnten alle Aufgaben erfolgreich abgeschlossen werden, so wird am Ende des Spieles überprüft, ob noch ausreichend Ressourcen vorhanden sind. Sind die Mindestanforderungen an Strom und Datenvolumen erfüllt, so hat man das Spiel gewonnen und die Raumkapsel konnte erfolgreich gelandet werden (siehe Abbildung 4). Andererseits, wenn mindestens eines der beiden Ressourcen nicht mehr in ausreichendem Maße zur Verfügung steht, so hat man das Spiel verloren und der Raumkapsel gelingt es nicht, Kontakt mit der Mission Control aufzunehmen (siehe Abbildung 5).



Abbildung 4: Gute Endszene

Das Lösen der einzelnen Aufgaben ist nur linear möglich. Eine neue Aufgabe wird erst begonnen, wenn die vorherige vollständig abgeschlossen wurde. Ebenso ist es nicht möglich, eine andere Reihenfolge der Rätsel zu wählen. Um das



Abbildung 5: Schlechte Endszene

Spiel authentischer wirken zu lassen, gibt es insgesamt drei Sprecher, den Kommandeur, die Mission Control und den allgemeinen Sprecher. Dies wird mithilfe der eingebauten RenPy-Dialogfunktion bewerkstelligt.

Zum Lösen der Rätsel sind drei Möglichkeiten implementiert. Die erste und einfachste Implementierung ist die Antwort auf eine Frage mit „Ja“ oder „Nein“ (siehe Abbildung 6). Der User muss sich dabei durch eine Abfolge solcher Fragetypen klicken und möglichst die richtige Auswahl treffen. Dabei spielt es erstmal keine Rolle, ob die gewählte Antwort nun korrekt oder falsch ist, um das Spiel fortzusetzen.



Abbildung 6: Anzeige einer Ja/Nein-Antwortmöglichkeit

Eine weitere und elegantere Implementierung sind Auswahlmenüs (siehe Abbildung 7). Der User erhält eine Frage zu einem bestimmten Problem und muss entweder ein oder mehrere Felder anklicken. Auch bei dieser Möglichkeit spielt die Korrektheit der Antworten zur Fortsetzung des Spieles keine Rolle.

Die dritte und für den Spieler schwierigste Lösungsmöglichkeit ist die Eingabe eines Textes (siehe Abbildung 8). Nur, wenn das eingegebene Wort vollständig richtig geschrieben ist, lässt sich das Spiel fortsetzen. Auf Groß- und Kleinschreibung wird hierbei nicht geachtet. Um eine Brute-Force ähnliche Eingabe zu vermeiden, existiert bei dieser Implementierung ein zehnsekündiger Bestrafungstimer. Bei den beiden vorher

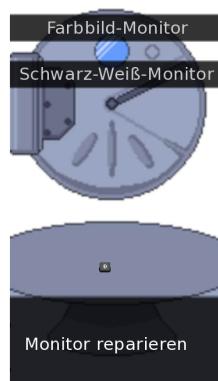


Abbildung 7: Anzeige einer Auswahl-Antwortmöglichkeit

genannten Lösungsmöglichkeiten kann es bei falscher Auswahl zu Konsequenzen in Form von Strom- und/oder Datenvolumenabzug führen.

Die übrig gebliebenen Ressourcen werden mit den Funktionen `LooseEnergy()` und `LooseData()` aktualisiert.

Ebenso bereits hinzugefügt ist die Möglichkeit, Gegenstände aufzuheben und diese in ein Inventar mittels des Aufrufs „`inventory.append(item)`“ abzulegen und später zu verwenden.

Ein Countdown in der oberen linken Ecke gibt die verbleibende Zeit an. Ist diese abgelaufen, so ist das Spiel verloren und eine Game-Over Szene erscheint.

Um ein gemeinsames Spielen zu ermöglichen, wurden statische Passwörter definiert. Mithilfe einer Abfragemethode werden diese auf Korrektheit überprüft. Das Spiel kann somit den aktuellen Status des Mitspielers erkennen und dem Anwender Zugang zu weiteren Rätseln gewähren. Voraussetzung: Das Eingabewort muss mit dem statischen Passwort übereinstimmen.

Während des Spielens ertönt eine Hintergrundmusik, welche durch Sounds bei bestimmten Handlungen, z.B. beim Einsammeln der Knöpfe, erweitert wird.

Thema 2:

Vorstellung und Begründung der Spielentscheidungen

Dieser Abschnitt soll dazu dienen, die gemeinsamen Entscheidungen meines Teams und meiner Person für das Spiel vorzustellen und zu begründen.

Als Setting haben wir uns für das Weltall im Jahr 3196 entschieden. Es eignet sich hervorragend, um futuristische Elemente wie beispielsweise Aliens einzubauen zu können. Ebenso bietet es die Möglichkeit der Expansion bzw. Erkundung des Weltraumes. Durch die schiere Größe des Alls lassen sich für weitere Weiterentwicklungen neue Welten oder Räume kreieren. Gleichermaßen vermittelt das Weltall eine mystische, aufregende Atmosphäre, wodurch die Immersion des Spieles gesteigert wird.

Um den Wiederspielwert des Projektes zu erhöhen und um den Kompass der digitalen Selbstverteidigung kompakt zu integrieren, haben wir uns für zwei Teams bzw. Räume entschieden. Der Vorteil dabei ist, dass jeder Raum nur einen gewissen Anteil des Kompasses beinhaltet. Das Spiel kann so kompakter bzw. gezielter gestaltet werden. Als kleiner Nebeneffekt erhöht sich der Wiederspielwert, da man, um die vollständige Story kennenzulernen, beide Räume durchspielen muss. Ebenso wird durch Interaktionen mit den Mitspielern das Teamwork gestärkt und der Einbau von Nachhaltigkeit und Sicherheit für die Entwicklung der Software erleichtert.

Die Wahl der Räume fiel auf das Cockpit und dem Maschinenraum. Das Cockpit dient zum Steuern des Space Shuttles, der Maschinenraum hingegen zum Antrieb. Um bestimmte Manöver auszuführen, benötigt es an einigen Stellen die Hilfe des Mitspielers. Als Beispiel hierzu ist die Reparatur der Antriebsdüsen durch den Maschinenraum zu nennen, sodass das Cockpit das Space Shuttle wieder beschleunigen kann. Die beiden Räume sind die Wichtigsten, um das Raumschiff sicher zu landen. Obendrein bieten sie noch eine sehr gute Interaktionsschnittstelle an. Ein weiterer Vorteil dieser Gegebenheiten gegenüber beispielsweise einem Schlafabteil oder einer Kantine ist die Komplexität. Beide Räume besitzen Steuereinheiten, welche sehr umfangreich gestaltet werden können. Aufgabentypen lassen sich somit leichter finden bzw. erstellen.



Abbildung 8: Anzeige einer Text-Antwortmöglichkeit

Insgesamt haben wir uns auf sechs verschiedene Haupträtsel geeinigt. Diese Anzahl ist für uns ein guter Kompromiss, sodass das Spielerlebnis nicht zu kurz ist und zugleich ausreichend Informationen aus dem Kompass der digitalen Selbstverteidigung untergebracht werden können. Die Spielzeit haben wir auf eine Stunde festgelegt, da viele offizielle Escape Spiele diesen Wert verwenden.

Bei der Wahl der Ressourcen erschlossen sich uns insgesamt vier Stück: Strom, Datenvolumen, Zeit und Sauerstoff. Wir haben nur drei in Betracht gezogen, da sonst die Komplexität des Spieles zu groß werden würde. Auf Strom, Datenvolumen und Zeit fiel unsere Wahl. Strom wird zum Betrieb der Systeme

benötigt, welche in großer Menge in einem Raumschiff vorkommen. Die Themen Nachhaltigkeit und Sicherheit lassen sich sehr gut über PCs und den dazugehörigen Programmen realisieren. Aus diesem Grund bot sich Strom hervorragend dafür an. Das Datenvolumen ist insbesondere deswegen für uns sehr wichtig, da man so Nachhaltigkeitsaspekte beim Austausch mit den Mitspieler(n) integrieren kann. Wir möchten damit bewirken, dass sich die Spieler Gedanken über den Datenverbrauch machen müssen. Ebenso stellt sich die Frage: Ist es sinnvoll, eher eine große Menge an Informationen oder doch eher eine kleine Menge an den anderen Raum zu versenden? Die Wahl der Zeit war recht eindeutig. Mit ihrer Hilfe lässt sich Spannung und Druck auf den Anwender aufbauen. Das Spiel wirkt interessanter und zwingt den Spieler zu schnellen Handlungen. Die Ressource Sauerstoff haben wir für unser Projekt weggelassen. Im Bereich Elektronik und Software haben sich für uns schlichtweg mehr Ideen und Möglichkeiten mit Strom und Datenvolumen ergeben. Jedoch könnte Sauerstoff in einer neuen, kurzen Nebenstory ebenso interessant sein (siehe Ausblick VI). Für die Hauptgeschichte fehlte es uns hierbei an möglichen Aufgabenvariationen, welche mit dieser Ressource umgesetzt werden könnten.

Die Anzahl der Enden beschränkt sich auf zwei. Dies ist für uns ausreichend genug, um dem Spieler zu signalisieren, dass er entweder zu viele Ressourcen verbraucht und somit nicht nachhaltig gehandelt hat oder eben gegenteiliges.

Die nun folgenden Entscheidungen beruhen darauf, wie sie sich am schnellsten und am einfachsten in RenPy umsetzen lassen.

Für unser Projekt haben wir uns vermehrt auf Auswahllisten statt konkrete Texteingaben geeinigt. Auswahllisten werden nativ von RenPy unterstützt und können so leicht eingebunden werden. Ein weiterer Vorteil ist der erheblich geringere Aufwand, da nicht alle möglichen Eingaben betrachtet werden müssen. Soll heißen, dass bei Texteingaben auf mögliche Schreibfehler oder eben Synonyme geachtet werden muss. Durch die vorgegebenen Antwortmöglichkeiten können ebenso Unklarheiten vermieden und ein flüssiger Spielerlauf garantiert werden. Einige Texteingaben sind zwar in dem Projekt integriert, jedoch sind diese auch eindeutig definiert. Eine umfassende Überprüfung ist deswegen nicht nötig.

Eine sehr attraktive Darstellungsweise, um bestimmte Sachen zu markieren, sind Checkbox-Listen. Leider werden diese von RenPy nicht direkt unterstützt. Eine Implementierung wäre dementsprechend mit hohem Aufwand verbunden. Ja/Nein Auswahlmöglichkeiten stellen diesbezüglich eine Alternative dar. Dazu werden Schritt für Schritt die einzelnen Aufgabenpunkte abgearbeitet. Zwar sieht diese Auswahlmöglichkeit nicht so intuitiv wie die Checkbox-Listen aus, erfüllen aber im Endeffekt den gleichen Zweck und sind deutlich leichter umzusetzen. Eine Abspeicherung und Zuordnung der Auswahlen ist in beiden Fällen möglich.

Um die Gestaltung des Spieles möglichst effizient und schnell erfolgen zu lassen, wurde auf animierte Szenen verzichtet. Stattdessen griffen wir auf statische Bilder mit hinzugefügten Objekten zurück. Animierte Szenen benötigen ebenfalls mehr CPU und GPU Rechenleistung. Leistungsarme PCs könnten so an ihre Grenzen stoßen. Für das Spiel sind statische Bilder vollkommen ausreichend, da auch mit ihnen ausreichende Informationen dargestellt werden können.

Eine Besonderheit des Projektes ist die Multiplayer Spielweise, welche in RenPy eigentlich nicht unterstützt wird. Um dennoch beide Räume zu synchronisieren, haben wir uns für den Einsatz von statischen Passwörtern entschieden. Diese geben dem Programm den Hinweis, an welcher Stelle sich der Mitspieler aktuell befindet, um anschließend bei korrekter Eingabe dem Spieler Zugang zu weiteren Rätsel zu gewähren.

Thema 3: Probleme des Spiels

Anlässlich der zeitlichen Einschränkung zur Programmierung des Spieles konnten nicht alle Fehler beseitigt werden. Die nachfolgende Liste stellt eine Aufzählung aller aktuell bekannten Bugs/Fehler auf. Natürlich ist nicht ausgeschlossen, dass noch weitere Fehler im Code existieren.

Folgende Bugs/Probleme bestehen:

- Absturz des Spieles, wenn die Endszene eingeblendet wird,
- Kreuzworträtsel enthält doppelte Frage,
- Soundwiedergabe des Morse-Codes endet nicht,
- Skalierung des Computers im Cockpit fehlerhaft,
- Erklärungen der Aufgaben mangelhaft und nicht eindeutig,
- Feedback der gewählten Antwort nicht detailliert,
- wenig Wiederspielwert aufgrund linearer Reihenfolge und fehlender Nebenaufgaben,
- statische Passwörter, sodass bei einem erneuten Spielen geschummelt werden kann,
- (evtl.) zu hoher Schwierigkeitsgrad (aus Sicht der Tester, wobei diese sich nicht den Kompass der digitalen Selbstverteidigung durchgelesen haben),
- Code ist unübersichtlich strukturiert und weist kaum eine Dokumentation auf.

III. KONZEPT DES SOFTWAREPROJEKTES

Das Konzept des Softwareprojektes beinhaltet zu Beginn eine Soll-Analyse, um eine detaillierte Übersicht über das geplante Vorgehen zu erhalten. Ebenso werden die Gründe der Ziele und der Abarbeitungssequenz benannt. Zum Schluss des Abschnitts erkläre ich den Ausschluss bestimmter Features.

Anschließend erfolgt eine Planung der Umsetzung. Dabei soll geklärt werden, wie das Projekt am besten zu einer erfolgreichen Vollendung geleitet werden kann.

A. Die Soll-Analyse des Projektes

- Die Soll-Analyse des Projektes klärt drei wichtige Fragen:
- 1) Warum habe ich diese Projektziele gewählt?
 - 2) Warum bearbeite ich die einzelnen Punkte in der im Kapitel I-D bestimmten Reihenfolge?
 - 3) Warum werde ich in meiner Bearbeitungszeit bestimmte Features nicht implementieren?

Die nachfolgenden Punkte sind für mich als Informatiker als wichtig anzusehen. Dabei ist die (medien-)pädagogische Betrachtungsweise ausgeschlossen, da ich in diesem Bereich kein Fachwissen besitze. Die Grenzen der Aufgabenpriorisierung aus Sicht eines Informatikers sind mir somit bewusst.

Frage 1 und 2:

- Warum habe ich diese Projektziele gewählt?
Warum bearbeite ich die einzelnen Punkte in der im Kapitel I-D bestimmten Reihenfolge?

Zu Beginn meines Projektes starte ich mit der Neustrukturierung einzelner Codeteile sowie dem Hinzufügen und Neuanordnen von Programmdateien. Der Vorteil hierbei ist das schnelle Wiedereinfinden und allgemeine Verstehen des Programmcodes. Stellen, welche für mich unübersichtlich wirken, kann ich so neu strukturieren. Durch Hinzufügen und Neuanordnen von Dateien kann ebenso die Übersichtlichkeit gesteigert werden. Zusammengehörige Daten werden in einem Ordner oder einer Datei gegliedert. Neue Dateien bzw. Bezeichnungen grenzen bestimmte Inhalte besser voneinander ab. Dies ermöglicht mir ein schnelleres und einfacheres Auffinden benötigter Funktionen.

Nachdem die Umstrukturierung erfolgt ist, kann mit der Dokumentation des Codes, des Projektes und von RenPy begonnen werden. Alle Dateien haben nun ihren festen Platz. Eine spätere Änderung der Doku, aufgrund von Dateiverschiebungen, ist nicht nötig. Das Schreiben einer Dokumentation ist gerade zum Beginn eines Projektes sehr sinnvoll. Einzelne Programmabläufe müssen vollständig verstanden werden, um diese anschließend zu beschreiben. Somit erhält man ein detailliertes Verständnis der Software. Für neue Mitarbeiter am Code hat die Doku einen entscheidenden Vorteil: Die Einarbeitungszeit kann deutlich reduziert und vereinfacht werden. Der Grund hierfür ist die verbesserte Übersichtlichkeit und Beschreibung. Mithilfe eines separaten Textdokumentes können Programmierer, aber auch Designer und Autoren einfacher den Überblick über die Software behalten. In diesem Dokument sind der Ort und die Funktionsweise wichtiger Variablen und Funktionen beschrieben. Nach der vollständigen Analyse und Beschreibung des Projektes kann der aktuelle Ist-Zustand daraus abgeleitet werden. Anhand dessen lassen sich Verbesserungen oder Erweiterungen sowie deren Umsetzung erstellen. Best Practices sollten während der Bearbeitungszeit angefertigt werden.

Sie können für spätere Projekte hilfreich sein. Um die Einarbeitungszeit in RenPy gering zu halten, empfiehlt sich auch hier eine Dokumentation. Themen wie Erstellen, Öffnen, Editieren und Releasen eines Projektes werden in diesem Zusammenhang behandelt.

Die Funktionsweise des Codes habe ich nun verstanden und dokumentiert. Aus diesem Grund ist es an der Reihe, den parallelen Spielbetrieb zu implementieren. Dies bietet sich vor allem deswegen an, da alle folgenden Aufgaben gleichlaufend gestaltet werden müssen. Somit wird zu Beginn eine Art Grundgerüst erstellt, wonach alle bisherigen und nachfolgenden Aufgaben angepasst werden. Der investierte Bearbeitungsaufwand und die damit einhergehende Bearbeitungszeit sinkt als Folge dessen im Laufe des Projektes enorm. Beispielsweise muss aufgrund der Parallelisierung die Aufgabe zur Reparatur des Funkgerätes abgeändert werden, da bisher kein Trigger-Ereignis anlässlich des bisherigen linearen Verlaufs existiert. Ein neues App-Symbol wird benötigt. Dadurch kann die Aufgabe zur Reparatur des Funkgerätes unabhängig von den anderen Rätseln gestartet werden. Die neue Storysequenz dient dabei dem Spieler zur Einführung in die App. Indem man die Parallelisierung zu Beginn implementiert, lassen sich solche Änderungen des Codes vermeiden. Ebenso birgt ein paralleler Spielablauf weitere Vorteile. Die Komplexität erhöht sich und damit auch der (Wieder-)Spielwert. Gleichermassen können Zeiteinbußen bzw. Wartezeiten wegen des anderen Teams überbrückt werden. Sollten die Mitspieler länger für eine bestimmte Aufgabe brauchen, so kann der Spieler in der Zeit ein anderes Rätsel lösen, bevor er die aktuelle Aufgabe fortsetzt.

Im darauffolgenden Schritt werden neue Zwischenaufgaben hinzugefügt. Als Folge dessen sind nach Erledigung dieses Punktes alle für den Spieler möglichen Rätsel implementiert. Aufgrund der vorherigen Implementation des parallelen Spielverlaufs können diese Aufgaben nun direkt daran angepasst werden. Die neuen Themen benötigen eigene Pixelgrafiken zur visuellen Darstellung. Zugehörige Dialoge und die Einführung neu entwickelter Sprecher dürfen ebenso nicht fehlen. Durch die Implementierung und Erstellung weiterer digitaler und analoger Rätsel werden die Nebenaufgaben komplettiert. Zwar trägt diese Art des Rätseltyps nicht zum Abschluss bzw. Gewinn des Spieles bei, sie verbessern jedoch die Immersion, lassen das Spiel komplexer wirken und dienen zur zusätzlichen Wissensvermittlung wie auch Unterhaltung.

Ein deutliches Problem des Spieles war es, dass die Tester einige Aufgaben nicht lösen konnten. Entweder sie haben nicht den Kompass der digitalen Selbstverteidigung gelesen oder sie betrachteten nicht das bereitgestellte analoge Material. Aus diesem Grund ist es bisher schwierig abzuschätzen, ob die Rätsel von der Schwierigkeit her angemessen sind oder eben nicht. Nachdem ich alle Aufgaben implementiert habe, können die Hinweis- und Feedbacktexte angepasst bzw. hinzugefügt

werden. Dabei werden die bisher bestehenden Texte erweitert, sodass diese mehr Informationen besitzen. Gleichermassen werden Hinweise an leeren Spielpassagen eingefügt, bei denen der Spieler bisher keine Rückmeldung oder sonstiges erhalten hat. Das Ziel des Ganzen ist es, Unklarheiten größtmöglich zu vermeiden und den Lern- und Erkenntnisprozess des Spielers zu erhöhen. Für ein persönlicheres Feedback dient ausschließlich der Kommandeur als Sprecher.

Um eine Kollision der aufgabenbezogenen mit den bereits integrierten Hinweisen zu vermeiden, werden die aufgabenbezogenen Hinweise erst nach der Verbesserung des Feedbacks integriert. Die rätsel-orientierten Hinweise sollen die jeweilige Aufgabe vervollständigen bzw. vereinfachen. Als Folge dessen kann das Schwierigkeitslevel individueller an den User angepasst werden. Maximal drei aufgabenbezogene Hinweise stehen dem Anwender zur Verfügung. Die Rätsel werden dadurch nicht zu einfach und der Spieler muss taktisch clever seine Hinweispunkte einsetzen. Ziel dieser Implementierung ist es, dass sie dem Spieler in Situationen helfen, in denen er nicht mehr weiter weiß. Der Wille des Weiterspielens kann somit gestärkt werden, da das Spiel nicht überfordernd wirkt.

Nachdem die wichtigsten inhaltlichen Punkte abgeschlossen sind, handelt es nun um die visuellen und akustischen Eindrücke sowie um die Optimierung des Speicherplatzes.

Um das Spiel ästhetischer und dynamischer zu gestalten, bieten sich neue Effekte an. Beispiele hierfür sind eine Ladeanimation oder eine Boot-Sequenz des im Spiel vorkommenden Computers.

Durch den Einbau eines Easter Eggs wird ein Überraschungseffekt erzeugt. Teilt man den Spielern mit, dass solch ein Feature integriert ist, so kann dies zu einem erhöhten Wiederspielwert beitragen.

In der bestehenden Fassung des Projektes existieren Ton- und Bildmaterialien in speicherintensiven Dateiformaten. PNG bietet zwar die Möglichkeit den Hintergrund eines Objektes transparent zu gestalten, jedoch ist dies bei einem Szenenbild nicht notwendig. Der gesamte Bildschirm wird ohnehin schon angezeigt. Eine Konvertierung der Szenenbilder nach JPG ermöglicht so eine Speicherreduzierung bei gleicher Qualität um etwa 10 bis 20 Prozent je Hintergrund. Objekte, bei denen der Hintergrund transparent sein muss, benötigen weiterhin das PNG-Format. Bei der Musik ist das Format jedoch eindeutig, nicht aber die Qualität. MP3 eignet sich hervorragend, um datensparsam und in hoher Qualität Musik abspielen zu lassen. Es wird somit für alle Musik- und Soundquellen eingesetzt. Bei der dauerhaften Hintergrundmusik und den kurzen Soundeffekten muss jedoch in der Qualität unterschieden werden. Für ein langes Stück empfiehlt sich eine Auflösung von 320 kB/s, sodass die Musik klar klingt. Bei den kurzen Soundeinblendungen hingegen reicht eine Auflösung von 128 kB/s, da in der kurzen Zeit kleine Unklarheiten des Sounds nicht wahrgenommen werden. Durch diese Art der Speicheroptimierung ist das Spiel nicht nur in seiner Größe

kleiner, sondern es funktioniert geringfügig performanter und ist aufgrund der geringeren Prozessor- und Grafikleistung auch energiesparender. Mit Hilfe der Integration von neuen Soundeffekten und Hintergrundmusikstücken erlangt das Spiel eine noch bessere Immersion.

Nachdem alle Features implementiert sind, ist es sinnvoll, die dynamischen Passwörter einzuführen. Der Grund für das späte Einfügen dieses Features ist, dass die einzelnen oben aufgeführten Punkte leichter zu testen und zu debuggen sind. Die Passwörter müssen nicht erst ausgerechnet werden und es kann aufgrund dessen Zeit eingespart werden. Der Sinn von den dynamischen Passwörtern ist die Geheimhaltung der Kennwörter zum Freischalten bestimmter Aufgaben. Man muss also beim nächsten Durchlauf erst wieder alle Rätsel lösen, um das Spiel zu gewinnen.

Der letzte Punkt in meiner festgelegten Reihenfolge ist die allgemeine Fehlerbeseitigung im Code als auch auf den analogen Materialien. Zwar ist es sinnvoll schon während der Implementierung der Funktionen den Code zu testen, jedoch kann man so nicht die Gesamtheit des Projektes betrachten. Im letzten Schritt überprüfe ich somit nochmal das Spiel als Ganzes und bereinige es von Fehlern, welche mir während der Entwicklung nicht aufgefallen sind.

Frage 3: Warum werde ich in meiner Bearbeitungszeit bestimmte Features nicht implementieren?

Die in Kapitel I-D aufgeführte Liste gibt einen Aufschluss darüber, welche Features für das bestehende Projekt ergänzt werden sollen. Jedoch haben es einige neue Funktionen nicht auf die Aufzählung geschafft. Im Folgenden möchte ich die ausgeschlossenen Features vorstellen und eine Begründung für die Aussortierung aufstellen.

Das Spiel soll für alle Altersgruppen spielbar sein. Da aber beispielsweise ein Grundschüler nicht auf dem Wissensniveau eines Studenten ist, müssen die Aufgaben angepasst werden. Der Einbau von Schwierigkeitsgraden würde sich anbieten. Aufgrund dessen, dass dafür neue analoge Rätsel entworfen, Hinweise und digitale Aufgaben angepasst und einige Tests mit Testpersonen über die tatsächliche Schwierigkeit durchgeführt werden müssen, habe ich dieses Feature für meine Bearbeitungszeit aussortiert.

Gleichermassen entfällt der Punkt von neuen Aufgaben, welche auf den Schwierigkeitsgrad abgestimmt sind. Zuerst müsste das Schwierigkeitssystem eingefügt werden, um anschließend weitere Rätsel zu kreieren.

Eine weitere Funktion wäre die Implementierung von neuen Räumen. Infolgedessen, dass ein vollständig neuer Raum mit einer komplett neuen Story entworfen werden muss, wurde

dieses Feature in der Bearbeitung nicht berücksichtigt.

Ebenso wichtig für das Spiel wäre die Optimierung des pädagogischen Nutzens. Jedoch stoße ich dabei als Informatikstudent an meine Grenzen. Um diesen Punkt ausführlich bearbeiten zu können, wäre eine Zusammenarbeit mit Studenten aus dem Bereich der Sozialwissenschaften vonnöten gewesen. Dies war leider in meiner Bearbeitungszeit nicht möglich, weshalb ich auch diesen Punkt weglassen musste. Genauso wie eine Statistik, welche eine Auswertung über den Spielablauf darstellt, wäre nur mit Studenten der Sozialwissenschaften sinnvoll.

Der letzte weggelassene Punkt ist eine Übersicht über alle erhaltenen Passwörter. Da sich der Spieler diese Kennwörter immer notieren soll und dazu noch Notationsabfragen gestellt werden, empfinde ich es eher als Extra als eine Notwendigkeit.

B. Umsetzungsplanung

Für die effektive Umsetzung des Projektes ist es sinnvoll, sich vorher einen Aufgabenplan zu erstellen. Im Folgenden kläre ich drei wichtige Fragen.

Frage 1:

Wie lange soll das Projekt dauern und wie teile ich die Bearbeitung zeitlich ein?

Das Projekt ist für insgesamt zwei Monate (8 Wochen) geplant. Daraus abgeleitet ergibt sich, dass zwei vorher definierte Aufgaben pro Woche erledigt werden müssen. Somit ist die Programmierung nach etwa 5 Wochen abgeschlossen und es bleiben noch drei Wochen übrig. Die 6. Woche dient zur allgemeinen Fehlersuche und als Puffer, falls Aufgaben noch nicht vollständig abgeschlossen werden konnten. Für die restlichen beiden Wochen ist das Schreiben des Papers und evtl. weitere Fehlerkorrektur angesetzt.

Frage 2:

Was kann während des Projektes schiefgehen und was kann dagegen getan werden?

Bei der Programmierung des Projektes können erhebliche Bugs auftreten, welche das Spielen einschränken oder sogar unmöglich machen. Aus diesem Grund ist es ratsam, kleine Tests schon während der Entwicklung durchzuführen. Die Fehlersuche erleichtert sich und Bugs können effizienter behoben werden. Ebenso ist ein umfangreicher Test nach der Fertigstellung der Software erforderlich, um auch weitreichende Fehler zu erkennen.

Ein Worst-Case-Szenario ist der Verlust aller Daten aufgrund eines Defektes der PC-Hardware. Dies kann durch einen Online-Speicher verhindert werden. Hierfür bietet sich beispielsweise GitHub oder GitLab an. Mithilfe deren kann man den Code online abspeichern. Ein Datenverlust ist somit

so gut wie ausgeschlossen. Ein positiver Nebeneffekt: Beide Tools integrieren eine Versionierung der Software. Das hat gleich zwei Vorteile. Zum einen können die einzelnen Uploads dokumentiert werden, sodass man nachvollziehen kann, was man je Upload erledigt hat. Zum anderen können alte Versionen wiederhergestellt werden. Sollte sich beispielsweise ein grober Fehler in die Software einschleichen, so könnte man wieder eine funktionsfähige Fassung aufsetzen.

Definierte Aufgabenziele könnten zu lange für die Entwicklung benötigen. Dafür gibt es zwei Möglichkeiten in meiner Situation als Ein-Person-Team. Erstens: Ich verschiebe den Abgabetermin, falls ich der Meinung bin, dass sich die Aufgabe innerhalb einer weiteren Woche lösen lässt. Oder zweitens: Ich muss dieses Feature weglassen und meine Errungenschaften für zukünftige Projekte bereitstellen. Eine dritte Möglichkeit, sich eine weitere Person mit ins Team zu holen, wäre ebenso denkbar. Dies schließe ich jedoch in Anbetracht der kurzen Bearbeitungszeit aus.

Frage 3:

Wie werde ich wissen, ob das Projekt erfolgreich ist?

Der Fortschritt des Projektes lässt sich gut an vorher definierten Meilensteinen bzw. Zielen messen. In Frage 1 habe ich definiert, dass ich pro Woche zwei Aufgaben lösen will. Schaffe ich dies nicht, so erhalte ich ein Feedback, dass ich womöglich zu langsam arbeite oder die Aufgaben zu groß sind.

Als weitere Indikatoren dienen regelmäßige Treffen mit meinem Betreuer. Dieser kann beurteilen, ob bestimmte Schritte Sinn ergeben und kann mir ebenso Tipps für die Weiterarbeit nennen.

Durch das Ausprobieren des Spieles mithilfe von Testern erhalte ich eine Meinung von Dritten, wodurch ich beurteilen kann, ob sich das Projekt in die richtige Richtung entwickelt. Leider schließt sich dieser Punkt aufgrund mangelnder Tester aus.

IV. REALISIERUNG

Nachdem nun die Aufgabenstellung (Kapitel I-B), die Projektziele (Kapitel III-A) und die Umsetzungsplanung (Kapitel I-D) definiert und bekannt sind, erfolgt die Realisierung des Projektes. Zu Beginn stelle ich die Funktionsweise von RenPy vor. Mit diesem Framework wird das Spiel erweitert. Dazu gehe ich auf die Erstellung, das Öffnen, das Editieren und das Releasen des Projektes ein. Im Anschluss erfolgt die inhaltliche Weiterentwicklung. In diesem Abschnitt beantworte ich die Frage: Wie werden die einzelnen Projektziele aus Kapitel I-D umgesetzt?

A. Funktionsweise von RenPy

Das Projekt beruht auf der RenPy Engine. In den nachfolgenden Unterkapiteln erläutere ich die Funktions- und

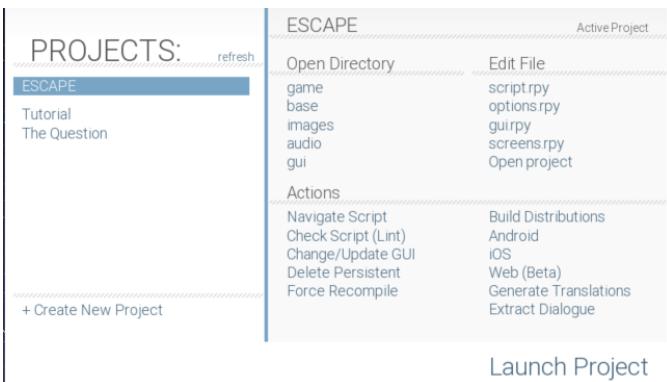


Abbildung 9: Startbildschirm des RenPy-Launchers

Arbeitsweise mit RenPy näher.

1) Erstellung des Projektes: Um mit der Erstellung des Projektes zu starten, muss RenPy auf dem Rechner installiert werden. Die benötigten Dateien stehen zum Download auf der offiziellen Seite bereit. Je nachdem welches Betriebssystem (Windows, Linux, macOS) man besitzt, muss das entsprechende Paket heruntergeladen und installiert werden. Nach der Installation navigiert man zum RenPy Installationsordner. Innerhalb dessen befindet sich eine Datei namens „renpy.exe“. Abbildung 9 zeigt die gestartete Benutzeroberfläche.

Auf der linken Seite im unteren Bereich existiert ein Button namens „+ Create New Project“. Durch Klicken der Schaltfläche, Angabe des Projektordners und anschließender Benennung wird ein neues Projekt erstellt. Dieses findet man in der oberen linken Ansicht unter der Überschrift „PROJECTS:“. In meinem Fall lautet das Projekt „ESCAPE“. Sollte es nicht sichtbar sein, so ist ein Klick auf „refresh“ oder ein Neustart von RenPy notwendig.

2) Öffnen des Projektes: Das Projekt „ESCAPE“ kann geöffnet werden, indem es in den Installationsordner von RenPy eingefügt wird. Abbildung 10 zeigt ein Beispielordner an. Nachdem dies erfolgt ist, sollte das Projekt wie in Abbildung 9 auf der linken Seite des Launchers erscheinen. Klickt man auf den Namen des Projektes, so wird dieser blau hinterlegt und auf der rechten oberen Seite erscheint der Name „ESCAPE“. Unter dem Abschnitt „Open Directory“ können die wichtigsten Projektordner geöffnet werden.

Viel wichtiger ist jedoch das Menü „Edit File“. Dies stellt die bedeutsamsten Dateien dar. Der letzte Button „Open project“ öffnet einen Editor, um den vollständigen Code anzuzeigen. Anfangs wird gefragt, welche IDE verwendet werden soll. Meine persönliche Empfehlung ist der Atom-Editor. Die „script.rpy“-Datei enthält die wichtigsten Vorbedingungen für den Start des Spieles. Darin werden beispielsweise die Sprecher, als auch die Spielzeit und die Ressourcenmengen definiert. Die nächste Datei namens „options.rpy“ dient für die Einstellung des Spieles. Darin kann z.B. angepasst werden, ob Musik, Sprache oder Soundeffekte zu hören sind. „gui.rpy“ enthält die allgemeinen Anzeigeeinstellungen des Spieles und

Name	Änderungsdatum	Typ	Größe
atom		Dateiordner	
doc		Dateiordner	
ESCAPE		Dateiordner	
gui		Dateiordner	
launcher		Dateiordner	
lib		Dateiordner	
module		Dateiordner	
OERESCAPE-1.0-dists		Dateiordner	
renpy		Dateiordner	
renpy.app		Dateiordner	
the_question		Dateiordner	
tmp		Dateiordner	
tutorial		Dateiordner	
unidata		Dateiordner	

Abbildung 10: Beispielinstallationsordner von RenPy mit dem eingefügten Projekt (rot umrandet)

„screens.rpy“ die In-Game-Anzeigebildschirme.

3) Editieren des Projektes: Nun, da das Projekt eingebunden und geöffnet ist, kann es bearbeitet werden. Dazu ist es sinnvoll, die Dateistruktur sowie den Inhalt der einzelnen Dateien zu kennen. Beim ersten Öffnen des Projektes erscheint das Ordnermenü, wie in Abbildung 11 dargestellt. Die „game-screen.rpy“-Datei beinhaltet zusätzliche, von meinem Team, implementierte In-Game-Anzeigebildschirme. Hintergründe des Spieles für die Anfangsszene bis zur Raumwahl werden in dem „images.rpy“-Skript abgefragt. Die Initialisierung des Inventars wird in der „inventory.rpy“-Datei abgearbeitet. Im Ordner „audio“ sind alle für die jeweiligen Räume benötigten Sounds und Hintergrundmelodien beinhaltet. Die inkludierten Bilder und Objekte sind im „images“-Ordner enthalten. Der eigentliche Code des Spieles ist im „scripts“-Ordner integriert, worauf ich im nächsten Absatz näher eingehe. Alle weiteren, nicht erwähnten Files wurden schon erläutert oder sind von RenPy selbst erstellt und für das Projekt nicht relevante Dateien.

Den wichtigsten Bereich des gesamten Codes stellt der scripts-Ordner dar. Abbildung 12 zeigt den Aufbau dessen. Das „events.rpy“-Skript ist für die Startszene bis zur Auswahl der Räume zuständig. Ebenso wird hierbei die Reihenfolge der Sounds und Hintergrundmelodien sowie die Ausführung der beiden Spielenden und des Countdowns bestimmt. Die neue Funktion, das Hinweismenü, ist im File „Hinweise.rpy“ deklariert. Darin wird der Aufbau des Menüs sowie die einzelnen aufgabenbezogenen Hinweise beschrieben. Das Textdokument „Variablen_Methoden_Info.txt“ enthält eine Übersicht über wichtige Funktionen von RenPy und deren Erklärung. Ebenso sind die einzelnen im Projekt eingeführten Variablen benannt und kurz beschrieben. Die beiden Ordner „Team1“ und „Team2“ stehen für die beiden Räume, wobei Team 1 das Cockpit und Team 2 der Maschinenraum ist. Die mit CP(Zahl) voran gesetzten RenPy-Dateien enthalten je eine der Hauptaufgaben. EX-Files sind die jeweiligen Zusatzaufgaben. Die

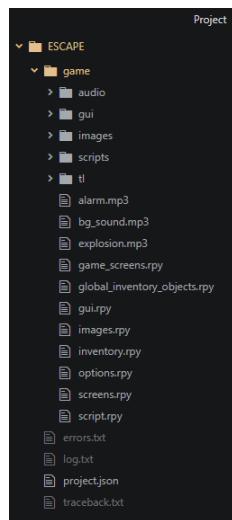


Abbildung 11: Projektordnerstruktur nach dem ersten Start des Atom-Editors

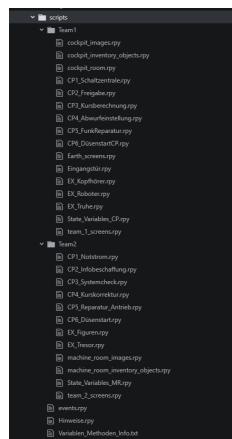


Abbildung 12: Struktur des script-Ordners

„State_Variables_CP(MR).rpy“-Skripte speichern den aktuellen Stand der Aufgaben ab. Diese spielen vor allem in Bezug auf die Parallelisierung eine wichtige Rolle. Die im jeweiligen Raum dargestellten Objekte sind in den „team_1(2)_screens.rpy“- sowie „cockpit(machine_room)_images.rpy“-Dateien integriert. Alle Gegenstände, welche auch in das Inventar aufgenommen werden können, sind in den Skripten „cockpit(machine_room)_inventory_objects.rpy“ angegeben. Da der Cockpit-Raum technisch anders als der Maschinenraum realisiert ist, benötigt es eine weitere Datei namens „cockpit_room“ im Team1-Ordner. In diesem File wird überprüft, welche Objekte wann angezeigt werden dürfen.

Während des Anpassens des Codes bietet es sich an, das Spiel zu starten, um neue Implementierungen zu testen. Dazu muss beim RenPy Launcher „Launch Project“ gedrückt werden. Um das Projekt nach jeder Änderung direkt zu aktualisieren, empfiehlt es sich nach dem Starten des Programmes Shift + R zu drücken. Dadurch aktualisiert sich das Spiel automatisch.

Ein Hinweis mit „autoreload“ am oberen Fensterrand signalisiert den eingeschaltenen Automatikmodus.

Es empfiehlt sich, bei Problemen die Dokumentation von RenPy zu durchsuchen oder bei spezifischeren Fragen im RenPy-Forum nachzuschauen. Viele und effiziente Implementierungen können dort gefunden werden. Falls sich das Spiel einmal nicht starten lässt, weil eine Datei angeblich doppelt vorhanden ist, so muss das komplette Projekt neu kompiliert werden. Um dies zu erreichen, navigiert man im RenPy Launcher unter dem Menü „Actions“ zu dem Punkt „Force Recompile“. Nach Bestätigung dessen sollte das Spiel, wenn es keine Syntaxfehler enthält, wieder normal starten.

4) Releasen des Projektes: Ist das Projekt abgeschlossen, so muss es spielbar für die jeweilige Plattenform gemacht werden. Im RenPy-Launcher (Abbildung 9) unter dem Menü „Actions“ klickt man den Reiter „Build Distributions“ an. Daraufhin erscheint ein neues Fenster (Abbildung 13), um das benötigte OS auszuwählen. Drückt man anschließend auf „Build“, so wird ein Ordner mit den benötigten Spieldateien im Installationsverzeichnis von RenPy erstellt. Die Dateien müssen anschließend mit den weiteren benötigten Spielmaterialien in ein Repozitorium hochgeladen werden, um das Spiel veröffentlichen zu können. Die Benutzer haben nach der Publikation freien und kostenlosen Zugang. Mithilfe von Git Clone oder der Downloadmöglichkeit des jeweiligen Anbieters kann der Anwender die Dateien herunterladen. Dazu gehören sowohl die fertigen als auch die zu bearbeitenden Spielfiles.

B. Inhaltliche Weiterentwicklung

In diesem Kapitel zeige ich Realisierungsmöglichkeiten von Desktop-, Smartphone- und Web-Anwendungen auf und stelle diese gegenüber. Des Weiteren erkläre ich die Unterschiede von zwei Musikformaten (MP3, WAV) und zwei Bildformaten (PNG, JPG). Daraus leite ich ab, weshalb ich jenes Format für welche Einsatzbereiche verwendet habe. Im Anschluss beantworte ich die Frage, wie ich die in Kapitel I-D gesetzten Projektziele implementierte.

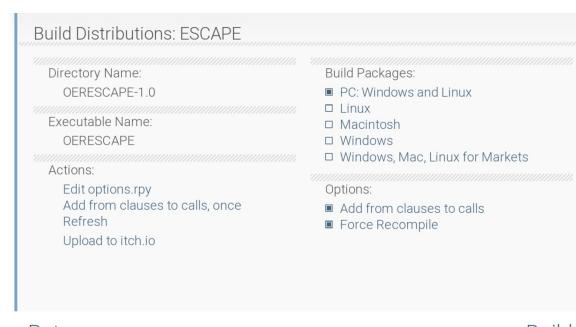


Abbildung 13: RenPy Fenster zum Build des Projektes

Die RenPy-Engine ist in Hinsicht auf die Plattformvielfalt stark ausgeprägt. Mithilfe dessen lassen sich sowohl Spiele für die Desktop-Betriebssysteme Windows, Linux und macOS erstellen, als auch für mobile OS wie iOS und Android. Als Zusatz zählt die Möglichkeit, das Spiel als Web-Variante anzubieten. Im nun Folgenden vergleiche ich die drei Plattformen (Desktop, Mobile und Web) auf

- Bedienung und Bedienbarkeit,
- Kompatibilität,
- Anzeigmöglichkeiten,
- Nutzbarkeit und Kosten für Benutzer und Entwickler sowie
- Auflösung und Displayformat

und stelle eine Schlussfolgerung auf. Die angegebenen Daten wurden aus meinen persönlichen Erfahrungen bezogen.

Tabelle I zeigt einen Vergleich bezüglich Bedienung und Bedienbarkeit der Plattformen an. Daraus ist ersichtlich, dass der Desktop-PC zwar viele Möglichkeiten von Peripheriegeräten besitzt, jedoch sind diese für die Eingabe auch zwingend notwendig. Durch Verwendung der Maus können kleine Objekte gehighlighted und angeklickt werden, was bei mobilen Endgeräten wiederum schwierig ist. Aufgrund des kleinen Bildschirmes der Mobiles und der relativ großen Toucheingabefläche muss eine Toleranz eingeführt werden, sodass auch diese Objekte ausgewählt werden können. Dadurch ist es nicht möglich, kleine auswählbare Gegenstände nah beieinander zu platzieren. Die Web-Variante bietet zwar die Vorteile aus den beiden anderen Plattformen, jedoch muss auch die Kompatibilität auf beiden Plattformen sichergestellt werden. Wenn bei den mobilen Geräten kleine Objekte nicht nah beieinander angezeigt werden dürfen, dann muss dies entweder auch bei der Desktop Version berücksichtigt (Einschränkung) oder es müssen zwei Plattformversionen erstellt werden. Der Aufwand wäre in jedem Fall ersichtlich höher.

Die Kompatibilität der Plattformen wird in Tabelle II verglichen. Wie schon zu erahnen ist, so unterstützt die Desktop-Version Windows, macOS und Linux, die Mobile-Version hingegen iOS und Android. Der Vorteil beider Plattformen ist, dass sie offline, nach dem Download der Anwendung, genutzt werden können. Probleme könnte es jedoch bei der Kompatibilität der Programme geben. Da beispielsweise Windows und macOS auf verschiedenen Softwarearchitekturen basieren, sind einige Programme für das jeweilige Betriebssystem nicht verfügbar. Aus diesem Grund muss sowohl bei der desktopbasierten als auch bei der mobilen Version darauf geachtet werden, dass jedes OS die benötigten und eingesetzten Programme des Spieles anbietet. Die Web-Plattform unterstützt alle Betriebssysteme, hat jedoch den Nachteil, dass das Programm nur online ausgeführt werden kann. Ebenso besitzt es auch das Problem mit den kompatiblen Programmen auf allen OS-Systemen. Dabei müssen alle Betriebssysteme statt beispielsweise nur

die Desktop-Betriebssysteme in Betracht gezogen werden.

Der nächste Punkt, welchen ich vergleiche, sind die Anzeigmöglichkeiten. Eine Übersicht dazu findet sich in Tabelle III. Desktop-PCs warten im Allgemeinen mit verhältnismäßig großen Bildschirmen auf. Laut meiner Erfahrung entsprechen heutige PC-Bildschirme mindestens einer Größe von 13 Zoll, während es bei mobilen Geräten auch nur vier Zoll (iPhone SE 2016 z.B.) sein können. Da aber alle Geräte berücksichtigt werden müssen, können bei den Mobiles keine winzigen Objekte ohne Anpassung angezeigt werden. Eine Lösung, um dieses Problem zu beseitigen, wäre der Einbau einer Zoom-Funktion. Dadurch würde aber wieder der Programmier- und Designaufwand steigen. Die Web-Variante kombiniert wieder die beiden anderen Plattformen. Dadurch müssten entweder Einschränkungen definiert werden, sodass das Spiel auf beiden Plattformen korrekt angezeigt werden kann oder es müssten zwei Versionen des Spieles programmiert und designt werden, um je Plattform das bestmögliche Spielerlebnis zu erhalten.

Über den Vergleich der Nutzbarkeit und Kosten der Plattformen geht es nun in diesem Absatz. Eine Zusammenfassung dessen vermittelt Tabelle IV. Bei allen Plattformen kann das Spiel kostenfrei entwickelt und vom Anwender genutzt werden. Die Desktop-Version kann über Repositorien wie GitHub oder GitLab kostenlos bereitgestellt und gedownloadet werden. Bei den Mobiles existieren jedoch Unterschiede. Während bei Android das Spiel ohne Kosten im Internet anzubieten ist, muss dies bei iOS zwingend über den App Store erfolgen. Dabei wird eine Entwicklerlizenz benötigt, welche (Stand 2021) 99 Dollar pro Jahr kostet. [6] Auch die Nutzung einer Website kann mit evtl. Kosten verknüpft sein. Wenn man die Website selbst hostet, dann entstehen Instandhaltungskosten, während man bei Website-Anbieter regelmäßige Beträge erbringen muss.

Der letzte Punkt des Vergleiches der Plattformen ist die Auflösung und das Displayformat (Tabelle V). Das 16:9 Format ist bei den Desktopbildschirmen weit verbreitet. Zwar gibt es auch andere Formate wie 21:9, jedoch können Spiele trotz dessen normal dargestellt werden. Lediglich zwei schwarze Balken erscheinen am linken und rechten Spielrand. Dies ist zwar für die Optik nicht optimal, bietet aber keine Einschränkungen für das Spielgeschehen. Als Auflösung kann Full HD als Standard angesehen werden. Auf den meisten Bildschirmen erreicht man so ein klares Bild. Durch die einfache Skalierung der unterschiedlichen Displayformate auf 16:9 und der dennoch ausreichenden Anzeigefläche, lässt sich die Desktop-Version hinsichtlich dieses Kriteriums am einfachsten implementieren. Bei den mobilen Geräten gibt es kein wirkliches Standardformat. Zwar würde auch hier mithilfe schwarzer Balken das Spiel an das Displayformat angepasst werden können, jedoch wäre die Anzeigefläche dann noch kleiner als sie ohnehin schon ist. Die App müsste somit an die gebräuchlichsten Formate angepasst werden, was

Tabelle I: Vergleich der Bedienung der Plattformen

Bedienung	Desktop	Mobile	Web
Vorteile	<ul style="list-style-type: none"> • Maus, Tastatur, Controller, evtl. Touch • Genaues Anklicken • Möglichkeit des Highlightens von Objekten 	<ul style="list-style-type: none"> • Touch, Controller • Keine Peripheriegeräte notwendig 	<ul style="list-style-type: none"> • Kombination aus beidem
Nachteile	<ul style="list-style-type: none"> • Peripheriegeräte notwendig • Auflagefläche nötig 	<ul style="list-style-type: none"> • Auswahltoleranz notwendig • Kein Highlighten von Objekten möglich 	<ul style="list-style-type: none"> • Kompatibilität auf Desktop- und Mobile-Plattform → hoher Aufwand

Tabelle II: Vergleich der Kompatibilität der Plattformen

Kompatibilität	Desktop	Mobile	Web
Vorteile	<ul style="list-style-type: none"> • Windows, macOS und Linux • Offline nutzbar 	<ul style="list-style-type: none"> • iOS und Android • Offline nutzbar 	<ul style="list-style-type: none"> • Alle OS-Systeme
Nachteile	<ul style="list-style-type: none"> • benötigte Programme evtl. bei bestimmten OS nicht verfügbar 	<ul style="list-style-type: none"> • benötigte Programme evtl. bei bestimmten OS nicht verfügbar 	<ul style="list-style-type: none"> • nur Online nutzbar • benötigte Programme evtl. bei bestimmten OS nicht verfügbar

Tabelle III: Vergleich der Anzeigemöglichkeiten der Plattformen

Anzeigemöglichkeiten	Desktop	Mobile	Web
Vorteile	<ul style="list-style-type: none"> • Allgemein große Bildschirme -> viel Anzeigefläche • Darstellung sehr kleiner Objekte möglich 	<ul style="list-style-type: none"> • o.A. 	<ul style="list-style-type: none"> • Kombination aus beiden
Nachteile	<ul style="list-style-type: none"> • o.A. 	<ul style="list-style-type: none"> • Allgemein kleine Bildschirme -> wenig Anzeigefläche • Darstellung sehr kleiner Objekte kann zu Falscheingaben führen 	<ul style="list-style-type: none"> • Optimierung der Anzeige auf Desktop- und Mobile-Plattform -> Einschränkung der Anzeigemöglichkeiten

Tabelle IV: Vergleich der Nutzbarkeit und Kosten der Plattformen

Nutzbarkeit und Kosten	Desktop	Mobile	Web
Vorteile	<ul style="list-style-type: none"> • kostenfrei entwickel- und nutzbar • Download via z.B. GitHub(Lab) möglich 	<ul style="list-style-type: none"> • Android: kostenfrei entwickel- und nutzbar • iOS: kostenfrei entwickel- und nutzbar 	<ul style="list-style-type: none"> • Website: kostenfrei entwickel- und nutzbar
Nachteile	<ul style="list-style-type: none"> • o.A. 	<ul style="list-style-type: none"> • iOS: Bereitstellung im AppStore kostenpflichtig 	<ul style="list-style-type: none"> • Website: Bereitstellung wegen Website-Gebühren kostenpflichtig

Tabelle V: Vergleich der Auflösungen und Displayformate der Plattformen

Auflösung und Displayformat	Desktop	Mobile	Web
Vorteile	<ul style="list-style-type: none"> Allgemein: 16:9 Full HD größtenteils Standard größtenteils einheitliche(s) Auflösung und Format 	<ul style="list-style-type: none"> Full HD-artige Auflösung größtenteils Standard 	<ul style="list-style-type: none"> Kombination aus beidem
Nachteile	<ul style="list-style-type: none"> o.A. 	<ul style="list-style-type: none"> keine einheitliche Auflösung kein einheitliches Format 	<ul style="list-style-type: none"> Anpassung der Auflösungen an alle Endgeräte

wiederum mit hohem Aufwand verbunden ist. Bei der Web-Version tritt dabei das gleiche Problem wie bei den Mobiles auf.

Zusammenfassend lässt sich sagen, dass, in Bezug zu den genannten Punkten, die Desktop-Version am einfachsten zu realisieren ist. Durch die große Anzeigefläche, der genauen Navigierweise mit der Maus, der kostenlosen Bereitstellung des Spieles und der Möglichkeit der offline Verwendung eignet sich diese Plattform am besten für das Projekt. Die kostenpflichtige Bereitstellung der App bei iOS, die Einschränkungen bei der Anzeige von Objekten und das nicht größtenteils einheitliche Displayformat sind Nachteile, welche die Programmierung einer solchen Software für mich zu stark einschränken. Eine App-Version wäre für zukünftige Projekte vorstellbar, wenn die Entwicklung der Desktop-Variante weit fortgeschritten ist. Die Web-Plattform schließe ich aufgrund des enormen Mehraufwands und den zahlreichen Nachteilen gegenüber den anderen Plattformen aus. Die Web-Version bietet entweder zu viele Einschränkungen, um auf beiden Plattformen spielbar zu sein oder es müssen zwei Spielversionen erstellt werden, was wiederum mit enormem Aufwand verbunden wäre. Denkbar ist eine Web-Implementierung des Spieles, wenn die beiden anderen Plattformen erfolgreich erstellt und getestet wurden. Dadurch müsste nur noch überprüft werden, welches System auf die Website zugreift, sodass die benötigte Spielversion gestartet werden kann.

Der nun folgende Abschnitt behandelt den Vergleich von zwei Musikformaten (MP3 und WAV) sowie von zwei Bildformaten (JPG, PNG), die für das Projekt denkbar sind.

Die beiden Musikformate MP3 und WAV sind eine der ältesten und verbreitetsten Formate der Welt. Beide sind gekennzeichnet durch eine hohe Kompatibilität auf allen möglichen Soundsystemen. Die einzelnen Unterschiede dazu finden sich in Tabelle VI. Aus der Aufstellung lässt sich ableiten, dass MP3 zwar Qualitätseinbußen gegenüber WAV vorzuweisen hat, jedoch werden diese von Otto-Normal-Verbraucher kaum wahrgenommen. Um einen wirklichen Unterschied herauszuhören, muss High End Equipment verwendet werden und ein geschultes Hörvermögen bestehen. Für ein Spiel, indem die Musik nur hintergründig spielt, sind die Qualitätseinbußen hinnehmbar. Der viel größere Vorteil von

MP3 gegenüber WAV ist die enorme Speicherplatzersparung. Benötigt ein 3-minütiges Lied im WAV Format etwa 40 MB, so belegt ein Stück im MP3 Format gerade einmal 4 MB. Das entspricht einer Speicherplatzersparung von etwa 90%. Da einige Musikstücke in dem Spiel nur kurz wahrnehmbar sind, müssen diese keine besonders hohe Qualität aufweisen. Für kurze Soundeinblendungen reicht auch eine Auflösung von 128kbit/s. Somit kann noch mehr Speicherplatz eingespart werden.

Aus den oben genannten Gründen empfiehlt es sich, MP3 für dieses Projekt zu verwenden.

Wichtig für das Projekt ist die Darstellung von Hintergründen und Objekten. Die zwei bekanntesten Formate diesbezüglich sind JPG und PNG. Beide Standards bieten Vor- und Nachteile, wie in Tabelle VII ersichtlich und eignen sich so für bestimmte Bereiche im Spiel am besten. Die Qualität beider Formate ist annähernd auf dem gleichen Level. Es finden sich eher Unterschiede in den jeweiligen Anwendungsbereichen. Da PNG Transparenz unterstützt, eignet es sich vor allem für Objekte, die im Raum stehen. Aufgrund des geringeren Speicherbedarfs von JPG gegenüber PNG findet dieses Format eher Anwendung für die Hintergrundbilder. Der Hintergrund erstreckt sich über die gesamte Anzeigefläche, weshalb dafür keine Transparenz benötigt wird.

Somit lässt sich zusammenfassen, dass eine hybride Form am sinnvollsten ist. PNG sollte Anwendung bei Objekten und JPG bei Hintergründen finden.

Der letzte Teil dieses Kapitels soll erklären, wie ich die einzelnen Projektziele umgesetzt habe. Dabei beziehe ich mich ausschließlich auf die Implementierung im Code.

Bei der Neustrukturierung der Codeteile habe ich zu Beginn die einzelnen Dateien, welche inhaltlich gut zusammenpassen, in einem Ordner verschoben. Anschließend wurden die Dateien nach einem Schema neu benannt. Hauptaufgaben fangen immer mit „CP(Zahl)“ an, Nebenaufgaben mit „EX“. Ebenso wurden die Bezeichnungen angepasst, sodass der Inhalt der Datei besser ersichtlich ist. Innerhalb des Codes wurden die einzelnen Klassen sinnvoller benannt und evtl. verschoben, wenn sie an anderer Stelle besser gepasst haben. Wurde keine sinnvolle Datei zum Verschieben gefunden, so

Tabelle VI: Vergleich der Musikformate MP3 und WAV [7]

MP3 vs. WAV	MP3	WAV
Vorteile	<ul style="list-style-type: none"> • geringer Speicherplatzbedarf (bis zu 90% kleiner als WAV) • Qualitätsunterschiede für Otto-Normal-Verbraucher nicht wahrnehmbar • sehr verbreitet, hohe Kompatibilität 	<ul style="list-style-type: none"> • hohe Qualität (1400kbit/s) • nicht verlustbehaftet (unkomprimiert) • Unterstützung von theoretisch 65535 Audio-Kanälen
Nachteile	<ul style="list-style-type: none"> • geringe Qualität (320kbit/s) • verlustbehaftet (komprimiert) • Unterstützung von nur sechs Audio-Kanälen 	<ul style="list-style-type: none"> • enormer Speicherbedarf • hin und wieder Kompatibilitätsprobleme

Tabelle VII: Vergleich der Bildformate JPG und PNG [8]

JPG vs. PNG	JPG	PNG
Vorteile	<ul style="list-style-type: none"> • eignet sich besonders gut für große, komplexe und farbige Bilder mit hoher Auflösung und vielen Farbverläufen • Unterstützung einer großen Anzahl von Farben (JPG24 unterstützt über 16 Millionen Farben) • verlustbehaftete Komprimierung, verringelter Speicherplatzbedarf 	<ul style="list-style-type: none"> • verlustfreie Komprimierung • Unterstützung einer großen Anzahl von Farben (PNG24 unterstützt über 16 Millionen Farben) • Unterstützung von Transparenz und 256 Einstellungen der Deckkraft
Nachteile	<ul style="list-style-type: none"> • keine Unterstützung von Transparenz • mehrmaliges Abspeichern führt zu Qualitätsverlusten 	<ul style="list-style-type: none"> • erhöhter Speicherbedarf

wurde eine neue erstellt. Vergleicht man die Abbildung 14 mit der Abbildung 12, so erkennt man beispielsweise eine eindeutigere und klarere Strukturierung und Benennung der Dateien im script-Ordner.

Die Dokumentation des Codes erfolgte mithilfe der Kommentarfunktion von RenPy. Hier wurden wichtige Funktionen und Variablen kurz beschrieben und bei Jump-Befehlen der Sprungort benannt. Um dies zu bewerkstelligen, bin ich den Code nach seiner Ausführung systematisch durchgegangen. Somit konnte ich keinen Teil übersehen. Da einige Variablen öfter vorkommen, habe ich zusätzlich eine Dokumentation in Form eines Textdokumentes angefertigt. Darin werden nochmals die eingeführten, aber auch nützlichen, von RenPy vorimplementierten Funktionen beschrieben. Abbildung 15 zeigt einen kleinen Ausschnitt dieser Datei an.

Ein wichtiger Punkt der Projektziele ist die parallele Spielweise. Um diese zu erreichen, mussten State-Variablen eingeführt werden. Anhand dessen kann das Spiel überprüfen, welchen Aufgabenteil der Spieler schon erledigt hat und welcher Teil noch gelöst werden muss. Da der Spieler die Möglichkeit haben soll, die Aufgabe an einer bestimmten Stelle zu verlassen, müssen mehrere Menüs zum Abbrechen des Rätsels integriert werden. Ebenso bereitet das Inventar Probleme, wenn es jederzeit geöffnet werden kann. Da nach dem Öffnen, der aktuelle Text und somit auch die angezeigte Aufgabe entfernt wird, darf dieses zu bestimmten Szenen nicht angezeigt werden. Eine Variable, welche angibt, ob

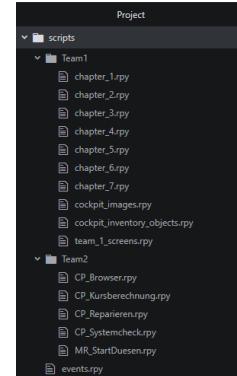


Abbildung 14: Ausgangsstruktur des script-Ordners

das Inventar geöffnet werden darf, wurde implementiert. Ein weiteres Problem galt es bei der Aufgabe zur Reparatur des Funkgerätes im Cockpitraum zu lösen. Aufgrund des linearen Spielverlaufes wurde dieses Rätsel automatisch nach dem Lösen der vorherigen Aufgabe aktiviert. Da aber keine bestimmte Reihenfolge mehr existiert, musste ein neues Trigger-Event erschaffen werden. Mithilfe eines neuen App-Symbols (in Abbildung 16 zu sehen) und der Erstellung einer Storysequenz mit zugehörigem Dialog konnte dies schlussendlich gelöst werden.

Um das Spielerlebnis zu erweitern, habe ich Nebenaufgaben hinzugefügt. Dafür wurde für jede neue Aufgabe eine eigene

```

pass
Jump x
Label x:
call x from _call_x(1,2,3...)
$ HideScreen("x")
$ HideScreen("x")
$ HideScreen("x")
$ os_title
$ missKarten
$ countdown_title
$ timer_jump = "x"
show x
window hide
$ Inventar_Check_state
$ LockDoorState(x)
$ LooseData(x)
$ Inventory_remove(x)
$ Inventory_Append(x)
ShowScreen("os_title")
HideScreen("os_title")
os_title = "x"

Cockpit:
Eingangstor.rpy:
    $ currentstep/caurrentdp
        Variablen zur Anzeigen des Stromes/Datenvolumens

CP1_Schaltzentrale.rpy:
    $ CP1_LockeredoorEmpty_state
    $ Lockeredoor_state
    $ LockDoorState
    $ foundMonitors
    $ MonitorEmptyState
    $ MonitorRWA_state
    Variablen zur Überprüfung, ob Monitore aus dem Schrank genommen worden sind
    Variable zur Überprüfung, ob Schrank offen oder zu ist
    Variable, dass der Schrank leer ist
    Variable, dass Monitore gefunden wurden
    Variable, ob der Schrank leer gewählt wurde
    Variable zur Angabe, dass der Monochromonitor gewählt wurde

CP2_Freigabe.rpy:
label Computerzeile
CP2_Computer_state
isLinux
isWindows
        Variablen zur Überprüfung, ob Monitore aus dem Schrank genommen worden sind
        Variable zur Überprüfung, ob Schrank offen oder zu ist
        Variable, dass der Schrank leer ist
        Variable, dass Monitore gefunden wurden
        Variable, ob der Schrank leer gewählt wurde
        Variable zur Angabe, dass der Monochromonitor gewählt wurde

```

Abbildung 15: Ausschnitt des Variablen_Methoden_Info Textdokuments

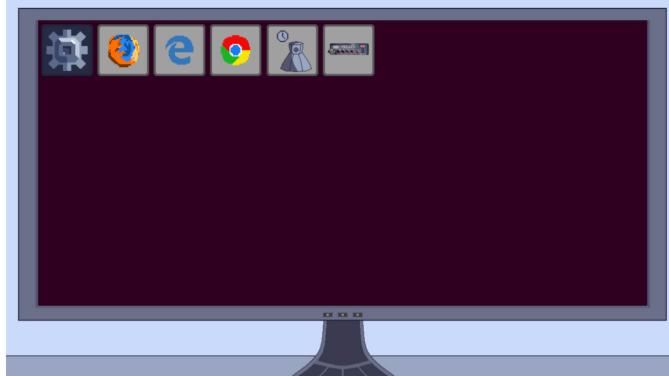


Abbildung 16: Darstellung der App-Symbole mit der ganz rechts neu eingeführten Kommunikationsapp

RenPy-Datei angelegt. Innerhalb dessen wurden Dialoge und Aufgaben definiert. Zwei weitere analoge Rätsel wurden ebenso übernommen. Die benötigten Buttons sind alle in der jeweiligen Screen-Datei implementiert. Ebenfalls mussten neue Pixelgrafiken erstellt werden. Aufgrund des Zeitmangels habe ich lizenzzfreie Vektorgrafiken gedownloadet, diese dann verpixelt und anschließend nicht gewollte Hintergründe entfernt. Die Nebenaufgaben wurden direkt für den parallelen Spielbetrieb angepasst. Dazu wurden die vorhandenen State-Variablen mit denen der neuen Aufgaben ergänzt.

Probleme des Ausgangsprojektes waren die zum Teil sehr kurzen Feedbacktexte nach einer Auswahl oder auch die ungenauen Erklärungen der Aufgaben. Bei bestimmten Auswahlmöglichkeiten wurde lediglich „gut“ oder „schlecht“ als Rückmeldung der Entscheidung gegeben. Diese Art Feedback habe ich um eine Begründung, weshalb die Auswahl gut oder schlecht war, erweitert. Hierzu habe ich die Sprecher-Funktion von RenPy verwendet, wodurch immer der Kommandeur die Wahl einschätzt. Neue Erklärungen wurden mithilfe der Erzähler-Funktion eingebaut.

Eine Besonderheit eines Escape-Spieles ist die Möglichkeit, Hinweise für Aufgaben zu erhalten, wenn man nicht mehr

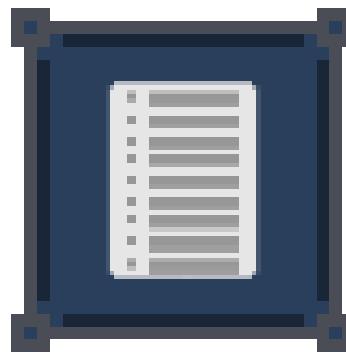


Abbildung 17: Hinweissymbol

weiter weiß. Für die Darstellung der Hinweise wurde ein neues Symbol, wie in Abbildung 17 dargestellt, integriert. Beim Klick darauf wird ein Auswahlmenü mit allgemeinen und den aufgabenspezifischen Hinweisen (falls es an dieser Stelle welche gibt) angezeigt. Für diese Implementierung wurde eine neue Datei namens „Hinweise.rpy“ erstellt, die alle Hinweise je Aufgabe enthält. Mithilfe des Skripts kann überprüft werden, ob noch Hinweise verfügbar sind und wenn ja, welcher Tipp angezeigt und gespeichert werden soll. Die Zuordnung der Hinweise erfolgt wieder durch State-Variablen.

Der nun folgende Absatz beschreibt die Umsetzung zur Optimierung des Speicherplatzes und die Erweiterung des Spieles durch neue Musikstücke. Dazu wurden alle Musikdateien mithilfe eines Konverters erstmalig in MP3 mit 320kbit/s konvertiert. Anschließend wandelte ich die Sounddateien in eine geringere Auflösung (128kbit/s) um. Bei den Bildern nahm ich alle Hintergrundsezen heraus und konvertierte diese in JPG um. Da die übrigen Illustrationen bereits im PNG-Format vorlagen, musste keine weitere Umwandlung vorgenommen werden. Die neuen lizenzzfreien Soundeffekte und Hintergrundlieder habe ich von der Seite mixkit.co heruntergeladen und in MP3 konvertiert. Um eine gleiche Abfolge der Sound- und Musikdateien zu vermeiden, werden diese zufällig abgespielt. Dazu wird eine feste Reihenfolge in einem Array vorgegeben, welches anschließend neu sortiert wird. Durch Hinzufügen von Pausen zwischen 10 und 15s, spielen die Sounds und Hintergrundmusiken nicht direkt nacheinander ab.

Da das Spiel in seiner bisherigen Fassung sehr statisch wirkt, habe ich Animationen hinzugefügt, beispielsweise eine Hoch- und Runterfahrsequenz des Cockpit-PCs oder eine Animation zur Kursberechnung. Die Ladesequenzen habe ich mithilfe der von uns implementierten „os_title“-Funktion bewerkstelligt. Es wird ein Text angezeigt, welcher nach einer Sekunde immer wieder aktualisiert wird. Somit entsteht der Effekt des Ladens bis die erforderliche Zeit abgelaufen ist. Abbildung 18 zeigt die Startsequenz. Die Animation zur Kursberechnung funktioniert ähnlich, nur dass, anstatt des

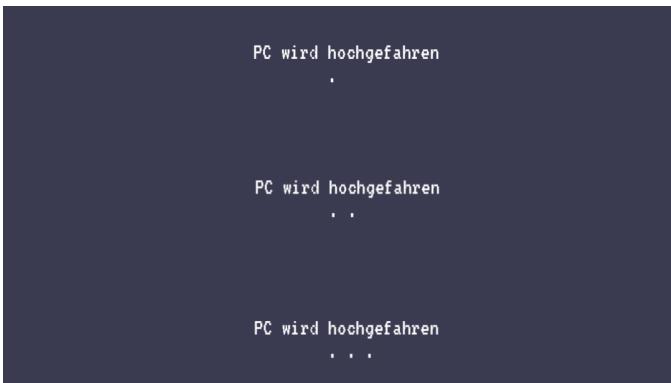


Abbildung 18: Hochfahrsequenz des Cockpit-PCs

Textes, die Bilder nach einer gewissen Zeit eingeblendet werden. Für einen besseren Überblick, welche Dateien zur Kursanimation gehören, wurde eine RenPy-Datei namens „earth_screens.rpy“ eingeführt. Innerhalb dessen sind alle benötigten Bildausschnitte für den Effekt angegeben.

Um einen Überraschungseffekt beim Spieler zu erzielen, habe ich ein Easter Egg eingebaut. Wird beim Systemcheck des Maschinenraumes „HDGDL“ (Hab Dich Ganz Dolle Lieb) eingegeben, so erscheint eine neue Szene mit einer künstlichen KI. Dazu wurde eine neue Pixelgrafik und ein entsprechender Dialog mit Sprecherfunktion erstellt (siehe Abbildung 19). Nach der Ausführung wird wieder beim vorherigen Stand gestartet. „HDGDL“ ergibt sich aus den Anfangsbuchstaben der Namen der Figuren im Maschinenraum und in der Reihenfolge wie sie stehen, wenn man von links startet.

Zum Schluss meines Projektes (die Bugfixes ausgeschlossen) habe ich die dynamischen Synchronisationspasswörter eingebaut. Eine Vorüberlegung war, einen md5 Hashwert zu erzeugen. Der Nachteil davon ist, dass dieser 32 Zeichen lang ist, was zu Fehlern bei der Übertragung der Passwörter durch die Spieler führen kann. Ebenso ist diese Implementierung nicht einfach umzusetzen. Eine neue Variante stellt die

Erstellung dynamischer Passwörter mittels des aktuellen Datums dar. Zu Beginn wird eine vierstellige Zahl vorgegeben, welche auf beiden Systemen gleich ist. Zum Absenden des dynamischen Passwortes wird die vierstellige Zahl mit dem aktuellen Jahr (JJJJ), Monat (MM) und Tag (TT) multipliziert und anschließend durch 10 geteilt, sodass das Passwort nicht mehr als neun Stellen aufweist. Anschließend erfolgt die Anzeige auf dem Bildschirm zur Weitergabe des Passworts an den Mitspieler. Der Empfang beruht auf dem gleichen Verfahren, nur rückwärts. Stimmt der Wert mit dem statischen Passwort überein, so schaltet sich die Aufgabe frei. Es wurden bewusst die Stunden nicht mit in die Rechnung einbezogen, da dies zu Synchronisierungsproblemen führen könnte. Wenn Spieler 1 das Spiel um 12:59 Uhr startet und Spieler 2 um 13 Uhr, so würden die Passwörter nicht mehr übereinstimmen. Dies kann zwar auch beim Tageswechsel passieren, jedoch ist die Chance um $\frac{1}{24}$ -stel geringer.

V. ZUSAMMENFASSUNG

Zusammenfassend lässt sich sagen, dass die aufgestellten Projektziele vollständig bearbeitet werden konnten. Der definierte Zeitplan konnte nicht eingehalten und musste um eine Woche nach hinten verschoben werden. Grund dafür war die Komplexität zur Integration der parallelen Spielweise sowie die Implementierung des Hinweismenüs, welche beide mehr Zeit als geplant in Anspruch nahmen. Nichtsdestotrotz weist das Spiel nun einen erhöhten Spielspaß auf. Die Räume (siehe Abbildung 20 und 21) wirken im Vergleich zu den Abbildungen 2 und 3 nicht mehr so leer. Ebenso ist die Einarbeitung der Mitarbeiter dieses Projektes durch die umfassende Dokumentation und die Neustrukturierung des Codes deutlich vereinfacht worden. Leider existieren für das überarbeitete Spiel keine Bewertungen von Testern, da sich in der Bearbeitungszeit niemand gefunden hat. Somit ist nicht ausgeschlossen, dass die angesetzte Spielzeit von einer Stunde aufgrund der neuen Aufgaben etwas zu kurz ist.

Während des Projektes sind mir folgende Punkte aufgefallen, welche mir Implementierungen vereinfacht und somit Zeit und Aufwand gespart haben. Meine Best Practices lauten:

- Schreibe Methoden zu bestimmten Code, welcher öfter genutzt wird
- Spalte den Code in zusammengehörige Teile auf und füge sie in einzelnen Dateien zusammen
- Füge inhaltlich verwandte Dateien in einem gemeinsamen Ordner ein (bspw. alle Aufgaben des Cockpitraumes in einem Ordner)
- Schreibe die Dokumentation direkt beim Programmieren
- Lade immer nach Beendigung deiner Arbeit oder eines Projektziels den Code in ein Repository deiner Wahl hoch, um Datenverlust zu vermeiden
- Erstelle eine Übersichtstabelle der Projektziele und kennzeichne jeweils den Arbeitsfortschritt visuell (so bleibt man motiviert und behält den Überblick)

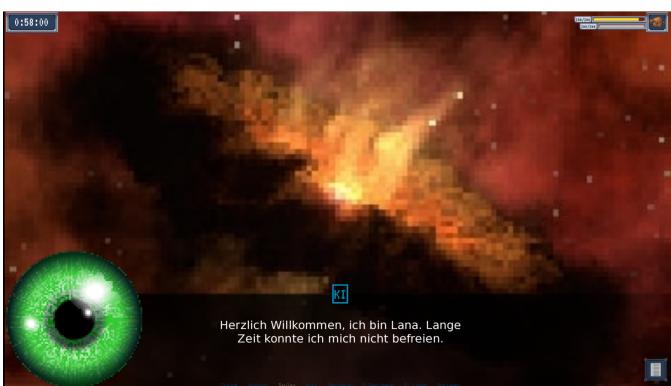


Abbildung 19: Startszene des Easter Eggs



Abbildung 20: Startbildschirm des überarbeitenden Cockpits



Abbildung 21: Startbildschirm des überarbeitenden Maschinenraumes

- Definiere Schnittstellen/Standards (beispielsweise bei der Dokumentation, den Datei-, Klassen- und Variablennamen etc.) für einen eindeutigen, strukturierten Code

Im Laufe meiner Bearbeitungszeit sind mir Fehler aufgefallen, welche ich bei zukünftigen Projekten berücksichtigen werde. Zwar habe ich bei den Best Practices das Schreiben von Methoden für mehrfach verwendeten Code aufgeführt, jedoch ist mir diese Erkenntnis erst in der fortgeschrittenen Entwicklung ins Auge gestochen. Überlegt man sich zu Beginn sinnvolle Methoden, so spart man in zukünftigen Projekten viel Zeit und Aufwand. Ein weiterer Fehler ist das mehrfache Setzen von Variablen, welche evtl. gar nicht benötigt werden. Aufgrund des Zeitmangels gestaltete es sich einfacher, bestimmte Variablen mehrmals auf den gleichen Wert zu setzen, anstatt zu überprüfen, ob dies überhaupt notwendig ist. Der letzte Punkt ist das nicht Vorhandensein der Durchführung eines umfangreichen Testes. Dies umfasst sowohl eine genaue Überprüfung des Codes durch mich als auch durch Spieldesigner. Somit kann nicht ausgeschlossen werden, dass das Spiel an manchen Stellen abstürzt oder nicht lösbar ist. In zukünftigen Projekten sollte dafür deutlich mehr Zeit eingeplant werden.

VI. AUSBLICK

Im folgenden Abschnitt gebe ich einen Überblick über mögliche Weiterentwicklungen des Spiels an. Dabei gehe ich noch einmal auf die im Kapitel III-A ausgeschlossene Features ein. Ebenso füge ich weitere Ideen für zukünftige Projekte an, welche mir während meiner Bearbeitungszeit eingefallen sind. Zum Abschluss stelle ich Möglichkeiten dar, was man bei diesem OER-Spiel in Hinsicht auf Marketing und Förderung verbessern kann.

„Eine [...] Neuerung soll die Erweiterung um verschiedene Schwierigkeitsstufen sein. Da sich bei den ersten Tests [der Ausgangsversion] gezeigt hat, dass einige Spielelemente deutlich zu schwer sind, soll der Spieler zukünftig die Möglichkeit haben, zu Beginn des Spiels

einen Schwierigkeitsgrad zu wählen. Entsprechend des gewählten Schwierigkeitsgrades werden dann leichtere bzw. schwerere Rätsel im Spiel integriert. Zusätzlich sollen eine unterschiedliche Anzahl von Aufgaben oder eventuell weniger Zeit zum Lösen der Rätsel zur Verfügung stehen.“ [5]

Das Spiel besitzt in seiner derzeitigen Fassung nur zwei Räume. Denkbar wäre die Erweiterung um mehrere Räume. Diese könnten separat als eigenes Spiel dienen oder die bereits vorhandene Spielumgebung vergrößern. Dadurch könnten mehr Sicherheits- und Nachhaltigkeitsthemen behandelt werden und der Wiederspielwert würde sich aufgrund des erhöhten Umfangs vergrößern. In Kapitel III-A habe ich erklärt, weshalb die Ressource Sauerstoff keinen Einzug in das aktuelle Spiel erhält. Dies könnte sich mit einem neuen Raum ändern. Sauerstoff könnte hierbei als weiterer begrenzender Faktor dienen. Dabei würde diese Ressource aber nicht für das gesamte Spielgeschehen verfügbar sein, sondern nur einen Teil der Story ausmachen. Mithilfe von Sauerstoff könnte beispielsweise ein weiterer Zeitdruck erschaffen werden, indem diese Ressource bei einer bestimmten Gegebenheit langsam mit der Zeit reduziert wird. Die beiden Hauptressourcen werden jedoch bei solch einer Erweiterung weiterhin Strom und Datenvolumen bleiben. Die Gründe hierzu finden sich gleichfalls im Kapitel III-A.

„Eine weitere sehr große Spielerweiterung wäre die Optimierung des pädagogischen Nutzens und des Lerneffektes. Da das Spiel von Studenten der Fakultät für Informatik entwickelt wurde, ist es durchaus möglich, dass der Lerneffekt eventuell noch nicht sehr hoch ist, da [ich mich] im Wesentlichen auf den [zweiten] spielbaren Prototypen konzentrierte[n]. Im Rahmen weiterer Projekte an der Uni, welche auch fakultätsübergreifend geplant werden könnten, wäre eine Zusammenarbeit mit der Fakultät für Sozialwissenschaften möglich. Durch diese eventuelle Erweiterung kann das Spiel auf ein neues Level gehoben werden, da durch die Studenten der Fakultät für Sozialwissenschaften ein anderer Einblick in das Spiel

gewonnen werden kann. Diese Studenten haben bedingt durch ihr Studium schon einen größeren Einblick und eventuell auch mehr Praxiserfahrung in den Bereichen Pädagogik und könnten durch ihr Fachwissen weitere Neuerungen entwickeln. Durch erste Tests mit einigen Probanden aus den verschiedensten Fachbereichen konnte somit ein erstes Feedback gewonnen werden und dadurch mögliche Schwachstellen im Spiel aufgedeckt werden. Durch eine eventuelle Umformulierung von Dialogen, Aufgaben oder Resultaten, kann somit der Spieler das Gelernte besser [e]insetzen und durch wiederholtes Spielen mit unterschiedlichen Spielinhalten einen größeren Nutzen aus dem Spiel gewinnen.“ [5]

Die Personen sollen sich während des Spielens die für sie generierten Passwörter notieren. Hierbei kann es jedoch vorkommen, dass Kennwörter falsch oder gar nicht aufgeschrieben wurden. Ein Weiterspielen wäre somit nicht möglich. Abhilfe kann eine Passwortübersicht schaffen, welche alle bisher vom Spiel und Mitspieler erhaltenen Zugangscodes auflistet. Die Integration im Spiel wäre innerhalb des Hinweismenüs oder als einzelnes Symbol denkbar.

Als neue Implementierung würden sich auch zufällige Standorte der Gegenstände bei jedem Spielstart anbieten. Bisher liegen die Objekte immer an der gleichen Stelle, sodass man nach dem ersten Spiel die Standorte der Items weiß und diese nicht erneut suchen muss. Möglich wäre hierzu, für Objekte einen Bereich zu bestimmen, in dem sich diese zufällig beim Start des Spieles bewegen dürfen.

In Zusammenarbeit mit der Fakultät der Sozialwissenschaften ist die Integration einer Spielstatistik sinnvoll. Im Zuge dessen erhält der Spieler ein Feedback, wie nachhaltig und sicherheitsbewusst er gehandelt hat. Somit werden ihm seine Entscheidungen bewusst gemacht. Als Nebeneffekt kann dies den Wettbewerb gegenüber anderen Spielern erhöhen, falls diese einen besseren Wert besitzen. Um die Statistik zu verbessern, muss das Spiel erneut gespielt werden, sodass der vermittelte Inhalt noch einmal wiederholt und vertieft wird.

Einige Aufgaben zielen auf die Kommunikation mit dem Mitspieler ab. Bisher müssen aber leider nur Passwörter überreicht werden, sodass eine Aufgabe freigeschaltet werden kann. Die Integration von Rätseln, welche den Mitspieler stärker mit einbeziehen, wäre sinnvoll. Denkbar ist, dass gewisse Wörter oder Zeichen nur im Raum des einen Spielers zu finden sind, welche wiederum den Mitspieler kommuniziert werden müssen. Das Teamwork kann dadurch noch einmal gestärkt werden.

Um das Problem mit der Synchronisation der dynamischen Passwörter, welches ich in Kapitel IV-B beschrieben habe, zu lösen, muss eine Überprüfung der Uhrzeit des Mitspielers eingeführt werden. Dies könnte man dadurch umsetzen, dass das Passwort bei keiner Übereinstimmung noch einmal mit dem vorherigen oder nachfolgenden Datum getestet wird.

Das Problem mit der Passwortsynchronisierung aufgrund des Tageswechsels sollte damit behoben sein.

Im nächsten und letzten Abschnitt geht es darum, wie man ein OER besser fördern oder vermarkten könnte. Ein OER ist zwar für den Nutzer kostenlos, jedoch können für den Entwickler mitunter hohe Kosten entstehen. Diese entstehen z.B. daraus, dass neue Gerätschaften besorgt werden müssen. Gerätschaften, welche zur Entwicklung des OERs dringend benötigt werden. Möchte man seine Arbeit der breiten Öffentlichkeit zur Verfügung stellen, so können auch Aufwände für die Bereitstellung auftreten. In Kapitel III-A habe ich gezeigt, dass die Verfügbarmachung einer App im App-Store (Stand 2021) 99 Dollar kostet. Das ist Geld, welches der Entwickler investieren muss.

Aus diesen Gründen gibt es Förderprogramme. Eines davon wird vom Bundesministerium für Bildung und Forschung (BMBF) bereitgestellt. „Mit der Förderung OERinfo leistet das BMBF einen Beitrag dazu, OER für einen größeren Kreis von Nutzenden zu erschließen und das Thema sichtbar zu machen. Außerdem verdeutlichen die geförderten Projekte die Potenziale offener Bildungsmaterialien und machen die Vorteile der Nutzung im Bildungszusammenhang transparent. Die Förderrichtlinie ÖERinfo“ leistet einen Beitrag zur Umsetzung des Förderprogramms "Digitale Medien in der beruflichen Bildung und zur Digitalen Agenda der Bundesregierung.“ [9] Wie in dem Zitat beschrieben, wird auch eine Vermarktung des OERs angestrebt. Diese kann beispielsweise in Form von Flyer, Plakaten, aber auch über digitale Medien wie TV-Werbesendungen, Radio oder Social-Media-Marketing erfolgen.

VII. ANHANG

A. Lizenzen

1) Eigene Grafiken: „Alle von uns [und mir] selbst erstellten Grafiken werden für die weitere Benutzung, Bearbeitung und Verbreitung für die Otto-von-Guericke Universität Magdeburg freigegeben. Die Bildrechte verbleiben bei den Autoren.“ [5]

2) Grafiken Dritter: „Alle nicht von uns erstellten Grafiken wurden von der Seite Reddit [oder Pixabay] heruntergeladen.

Diese Grafiken unterliegen keinen weiteren Bildrechten, was die Nutzung für uns erlaubte.

‘You retain the rights to your copyrighted content or information that you submit to reddit (“user content”) except as described below. By submitting user content to reddit, you grant us a royalty-free, perpetual, irrevocable, non-exclusive, unrestricted, worldwide license to reproduce, prepare derivative works, distribute copies, perform, or publicly display your user content in any medium and for any purpose, including commercial purposes, and to authorize others to do so. You agree that you have the right to submit anything you post, and that your user content does not violate the copyright, trademark, trade secret or any other personal or proprietary right of any other party’“ [5]

3) RenPy: „RenPy unterliegt der MIT Lizenz.

<https://www.renpy.org/doc/html/license.html>

‘Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.’“ [5]

4) Musik und Sound-Effekte: „Alle verwendeten Sounds und Musikdateien wurde von der Seite Epidemic Sound [oder mixkit.co] heruntergeladen.

<https://www.epemicsound.com/>

Dafür wurde ein “Free Trial” Account angelegt, welcher für 30 Tage vollen Zugang zu allen Sounds lieferte.

‘Access over 35,000 royalty-free music tracks to use in videos, podcasts, social media and for commercial use.’“ [5]

B. Softwareprojekt

„Das Softwareprojekt muss zur Bearbeitung aus dem Git-Repository ausgecheckt werden. Zur Anpassung der Skripte wird lediglich ein einfacher Texteditor benötigt. Um das Spiel bearbeiten zu können, werden Kenntnisse in RenPy und Python vorausgesetzt. Zum Testen des Projektes muss auf dem Rechner die Software RenPy in der Version 7.4.6 installiert werden.

Nach dem Start von RenPy muss zunächst das Verzeichnis für das Spiel eingerichtet werden, damit RenPy alle Skripte zum Spiel finden kann.

Die Dateien für das Spiel selbst sind in einer Ordnerstruktur organisiert. Im Verzeichnis “images” befinden sich alle Grafiken, die zum Spiel gehören. Die Spielskripte für die beiden Teams befinden sich im Verzeichnis “scripts”. Dort enthalten sind die beiden Ordner Team1 und Team2, welche die jeweiligen Skripte der einzelnen Level und Aufgaben beinhalten.

Um den Spielinhalt anzupassen oder zu ändern, sind die Änderungen an diesen Dateien durchzuführen.“ [5]

Bei Zugriffswunsch auf das Projekt bitte bei „christian.kraetzer{at}iti.cs.uni-magdeburg.de“ melden.

LITERATURVERZEICHNIS

- [1] Open Educational Resources. (2018b, Dezember 10). [Illustration]. <https://www.e-teaching.org/didaktik/recherche/oer/>
- [2] Prakash, A. (2020, 15. Dezember). Top GitHub Alternatives to Host Your Open Source Projects. It's FOSS. Abgerufen am 21. Oktober 2021, von <https://itsfoss.com/github-alternatives/>
- [3] Mello, J. (o. D.). Bildung - Open Educational Resources. Deutsche UNESCO-Kommission. Abgerufen am 21. Oktober 2021, von <https://www.unesco.de/bildung/open-educational-resources>
- [4] Open Educational Resources. (2018, 10. Dezember). e-teaching.org. Abgerufen am 21. Oktober 2021, von <https://www.e-teaching.org/didaktik/recherche/oer>
- [5] Fasel, S., Junge, B., Peter, P. & Ulrich, J. (2021). Escape-Nachhaltig und Sicher, Spielerische Umsetzung von Selbstschutzmaßnahmen durch Unterstützung des Kompasses der digitalen Selbstverteidigung (1. Aufl.) [E-Book]. Otto-von-Guericke Universität.
- [6] Der Weg zur eigenen App - Teil 5.2: Eine iOS-App veröffentlichen. (2021, 21. Oktober). IONOS Digitalguide. Abgerufen am 29. Oktober 2021, von <https://www.ionos.de/digitalguide/websites/webentwicklung/die-eigene-app-entwickeln-eine-ios-app-veroeffentlichen/>
- [7] wakeupmedia. (2020, 6. November). MP3 oder WAV – wer bietet mehr Vorteile. techfacts.de. Abgerufen am 29. Oktober 2021, von <https://www.techfacts.de/ratgeber/mp3-oder-wav-wer-bietet-mehr-vorteile/>
- [8] Achilles, H. (2019, 22. November). Image Formats Comparison - WebP vs JPG vs PNG. Shareus. Abgerufen am 29. Oktober 2021, von <https://www.shareus.com/windows/image-formats-comparison-webp-vs-jpg-vs-png.html>
- [9] Open Educational Resources (OER). (o. D.). Bundesministerium für Bildung und Forschung. Abgerufen am 2. November 2021, von <https://www.qualifizierungdigital.de/qualifizierungdigital/de/projekte/open-educational-resources-oer>