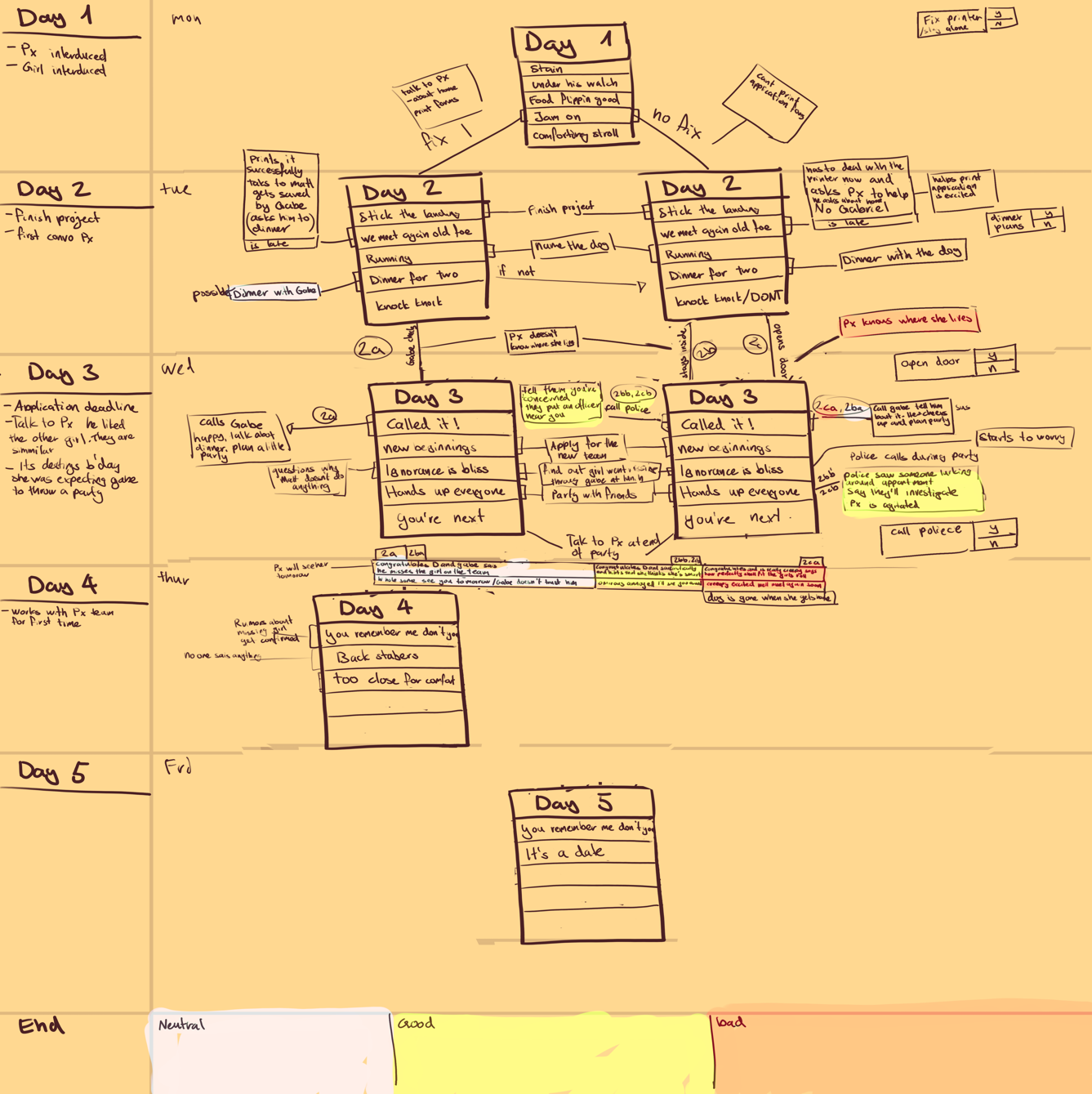
Destiny’s Reminder – Eine interaktive Geschichte

**Ein Projekt von:** Carole Meier

**Abteilung:** G4C

**Datum:** 31.8.23

**Schule:** Kantonsschule Wettingen (KSWE)

**Betreuende Lehrperson:** Oliver Probst

**Gegenleserin:** Sara Nyffenegger   
Zusammenfassung  
Hier kommt die Zusammenfassung.

Vorwort  
Ein eigenes Spiel wie dieses zu schreiben und gestalten war schon lange eine meiner Ambitionen. Seit meiner Kindheit befasse ich mich in meiner Freizeit gerne mit Videospielen und Geschichten jeglicher Art. Mich in Erzählungen zu vertiefen aber auch eigene zu kreieren war schon immer eine meiner Lieblingsbeschäftigungen. Neben dem Lesen und Spielen lasse ich aber auch gerne beim Malen meiner Kreativität freien Lauf. Gemalt habe ich immer schon gerne, doch eigene Geschichten zu entwickeln und zu illustrieren begann ich erst wirklich in den letzten Jahren. Während meiner Zeit and der Kantonsschule konnte ich dank dem breiten Freikursangebot bereits Erfahrungen im Bereich des Literarischen Schreibens sammeln und meine Leidenschaft Geschichten zu erzählen weiter ausleben. Doch für mich wurde schnell klar, dass ich meine Geschichten auch gerne visuell darstellen möchte. Durch mein Interesse an Informatik und Gamedesign entstand schlussendlich die Idee für diese Arbeit eine eigene interaktive Geschichte in Form eines Spiels zu programmieren.

Inhaltsverzeichnis

[Inhaltsverzeichnis 3](#_Toc146356405)

[1. Einleitung 1](#_Toc146356406)

[1.1 Themeneingrenzung 1](#_Toc146356407)

[1.2 Ziele 1](#_Toc146356408)

[1.3 Fragestellungen 1](#_Toc146356409)

[1.4 Methoden 1](#_Toc146356410)

[2. Theorie/Recherche 2](#_Toc146356411)

[2.1 GitHub 2](#_Toc146356412)

[2.2 Interaktive Geschichten und Visual-Novels 2](#_Toc146356413)

[2.3 Ren.py 3](#_Toc146356414)

[2.4 Welche Bild Elemente sind benötigt? 3](#_Toc146356415)

[3. Arbeitsprozess 4](#_Toc146356416)

[3.1 Konzipierung der Geschichte 4](#_Toc146356417)

[3.2 Planung der Handlungsstränge 5](#_Toc146356418)

[3.3 Schreibprozess des Skripts in Ren.py 6](#_Toc146356419)

[3.3.1 Definieren der Charaktere 6](#_Toc146356420)

[3.3.2 Schreiben der Szenen mittels Labels 7](#_Toc146356421)

[3.3.3 Einfügen Interaktiver Elemente 7](#_Toc146356422)

[3.4 Illustrationsprozess der Spielgrafiken 8](#_Toc146356423)

[3.4.1 Gestaltung der Charaktere und diverser Spielelemente 8](#_Toc146356424)

[3.4.2 Gestaltung der Hintergründe 8](#_Toc146356425)

[3.4.3 Gestaltung der Benutzeroberfläche und Icons 9](#_Toc146356426)

[3.5 Verknüpfen von Skript und Grafik 9](#_Toc146356427)

[3.5.1 Einfügen eigener Grafiken und Schriftarten 9](#_Toc146356428)

[3.5.2 Darstellen der Charakter und weiterer Spielelemente 9](#_Toc146356429)

[3.5.3 Aufrufen der Hintergründe 9](#_Toc146356430)

[3.5.4 Effekte und Übergänge 9](#_Toc146356431)

[3.5.5 Der Feinschliff 9](#_Toc146356432)

[4.Schluss 10](#_Toc146356433)

[4.1 Beantwortung der Fragestellung 10](#_Toc146356434)

[4.1.1 Wie schreibt man eine Interaktive Geschichte? 10](#_Toc146356435)

[4.1.2 Wie setzt man eine Geschichte in eine Spielbare form um/Welche Methoden gibt es? 10](#_Toc146356436)

[4.1.3 Was braucht es, um einen Visual Novell zu kreieren? 10](#_Toc146356437)

[4.1.4 Wie verstärken visuelle Elemente den Eindruck einer Geschichte? 10](#_Toc146356438)

[4.1.5 Wie kann ich meinen Code strukturiert notieren (Clean Code)? 10](#_Toc146356439)

[5. Reflexion 11](#_Toc146356440)

[5.1 Inhalt 11](#_Toc146356441)

[5.2 Methoden 11](#_Toc146356442)

[6. Quellenverzeichnis 11](#_Toc146356443)

[7. Anhang 12](#_Toc146356444)

# 1. Einleitung

## 1.1 Themeneingrenzung

Bla

## Ziele

Das Ziel dieser Arbeit war es, eine ansprechende Interaktive Geschichte zu gestalten und diese durch das Programmieren eines Videospiels in spielbarer Form umzusetzen. Die Geschichte sollte mehrere interessante Pfade und Enden besitzen, welche der Spieler/die Spielerin mit seinen/ihren Entscheidungen beeinflussen kann. Die Geschichte sollte hauptsächlich aus natürlich- und spannend gestaltetem Dialog der verschiedenen Charaktere bestehen und sollte durch passende Grafiken zu einem in sich abschließenden und visuell ansprechenden Spielerlebnis werden.

## 1.3 Fragestellungen

Eine interessante interaktive Geschichte in Form eines Videospiels erstellen zu können erforderte eine ausführliche Auseinandersetzung mit folgenden Fragen:

* Wie schreibt man eine Interaktive Geschichte?
* Wie setzt man eine Geschichte in eine Spielbare form um/Welche Methoden gibt es?
* Was braucht es, um einen Visual Novell zu kreieren?
* Wie verstärken visuelle Elemente den Eindruck einer Geschichte?
* Wie kann ich meinen Code strukturiert notieren (Clean Code)?

## 1.4 Methoden

Da sich die Umsetzung einer Geschichte in spielbarer Form aus einer Kombination vieler verschiedenen Bereichen zusammen setzt waren Recherche und Arbeit in den Bereichen Literatur, Kunst und Informatik nötig. Zur Recherche diente hauptsächlich das Internet. Bezogen wurden verschiedene Dokumentationen der genutzten Programme wie auch Anleitungsvideos zu spezifischen Problemen. Literarische Recherche war nur minimal nötig da bereits eine gute Grundlage und einiges an Erfahrung bezüglich literarischer Arbeit bestand. Als Vorbereitung wurden verschiedene Pläne und Mindmaps der möglichen Geschichte gezeichnet und schlussendlich auf ein Übersichtliches Flussdiagramm der Handlungen der Geschichte ausgearbeitet. Geschrieben wurde die Geschrieben anschließend direkt mit der Visual-Novel-Engine Ren’ Py. Letztlich wurden die nötigen Grafiken direkt Digital mit auf einem iPad in Procreate gemalt und genutzt. Als Hilfsmittel zur Planung und Illustration der Hintergründe diente Sims 4. Der Fortschritt der Arbeit wurde mittels GitHub festgehalten.

# 2. Theorie/Recherche

## 2.1 GitHub

Schon von Beginn an war klar, dass in diesem Projekt selbst geschriebener Code involviert werden würde. Als Werkzeug zum Speichern und Teilen dieses Codes wurde deshalb von der Betreuenden Lehrperson empfohlen GitHub zu verwenden. «GitHub ist ein Onlinedienst zur Versionsverwaltung für Software-Entwicklungsprojekte» (<https://de.wikipedia.org/wiki/GitHub>). GitHub ist ein weitverbreitetes, kollaboratives Versionsverwaltung Tool, welches heutzutage von vielen Softwareentwicklern und Informatikern verwendet wird, um Code über das Internet zu speichern und mit anderen zu Teilen.

GitHub ist ein nützliches Hilfsmittel für jede Art Projekt, da es ermöglicht kontinuierlich neue Versionen von Codefiles und ganzen Projekten festzuhalten und archivartig abzuspeichern.

Mit einem GitHub Account kann reibungslos von verschiedenen Geräten und Personen and einem Projekt gearbeitet werden, dank ihres Cloud-Storage Systems.

Um die Änderungen eines Files oder eines Ordners aufzeichnen zu können, wird auf GitHub ein Repository generiert. Dieser agiert als Speichermedium aller Dateien, welche von GitHub für dieses spezifische Projekt erfasst werden sollen. Das Programm merkt, sobald eine Änderung innerhalb dieses Ordners vorgenommen wird und zeichnet diese auf. Werden diese Änderungen dann abgeschlossen, kann durch einen Commit-Befehl diese neue Version des Projekts wieder mit der Cloud synchronisiert werden. Die alte Version wird im Archiv abgespeichert und durch die Neue ersetzt. GitHub ist ein Nützliches Werkzeug für jegliche Arbeiten, welche mit Code zu tun haben, da es jede Änderung in seinen Speicher aufnimmt und chronologisch festhält. Falls ein Fehler entsteht oder Code aus älteren Versionen benötigt wird, können durch GitHub diese alten Versionen wiederhergestellt werden. Mit jedem Commit wird das Datum und die genaue Uhrzeit aufgezeichnet und der Nutzer/die Nutzerin wird aufgefordert eine kurze Beschreibung der Änderungen anzufügen. Dies erleichtert es später bestimmte Versionen des veränderten Codes wiederzufinden.

Für dieses Projekt bedeutete die Nutzung von GitHub, dass kein zusätzliches Journal geführt werden musste. Den Code und die weiteren Spieldateien fortlaufend in die Cloud hochzuladen, würde ein traditionelles Journal ersetzen, da jede Änderung bereits auf GitHub aufgezeichnet wird. Durch das Teilen des Projekt Repositorys mit der Betreuenden Lehrperson konnte so auch der Fortschritt der Arbeit dokumentiert werden.

## 2.2 Interaktive Geschichten und Visual-Novels

Interaktive Geschichten existieren in vielen Formen und für jederlei Altersgruppen. «Choose your own Adventure» Bücher wie zum Beispiel die 1000 Gefahren Serie des Ravensburger Verlags, sind zwar in Stil und Inhalt eher an Kinder gerichtet, zeigen jedoch erfolgreich wie auch in rein Text basierenden Medien ein spielerisches Leseerlebnis entstehen kann. Eine weitere Methode findet man auch in digitale Medien, wie Film und Videogames. «Bandersnatch» aus der Serie «Black Mirror» setzt die Idee einer selbst gewählten Geschichte erfolgreich in einem Film um. Inspiration für dieses Projekt wurde jedoch aus dem letzteren Bereich der digitalen Medien gezogen. Als Ziel der Arbeit wurde ein kurzer selbst illustrierter Visual-Novel gesetzt.

Visual-Novel:

Wie der Begriff bereits impliziert, besteht ein Visual-Novel aus einer Kombination von Text und Grafik, welche zusammen eine spielbare Geschichte bilden. Oft wird das Genre als Unterklasse der Videospiele gesehen, da sie ebenfalls eine Art digitaler, interaktiver Erzählungen sind. Seinen Ursprung nahm diese Spielart in Japan, wo sie auch heute noch sehr beliebt sind. Obwohl sie mittlerweile auch in anderen Ländern bekannt sind, ist der Einfluss Japanischer Kultur wie auch der von Manga und Anime oft noch gut erkennbar. Beispiele bekannter Visual-Novels wären die «Persona» Serie von ATLUS und SEGA oder «Doki Doki Literature Club» welches von einem kleineren, Indie-Studio namens Salvato programmiert wurde. Dank des simplen Aufbaus eignet sich ein Visual-Novel als Projekt für kleinere Studios oder Einzelpersonen. Ebenfalls eignen sie sich gut als Einstiegprojekt um sich mit dem Designen eines Spiels bekannt zu machen, da sie den Fokus auf eine zentrale Erzählung setzten und diese hauptsächlich durch sehr limitiertes dialogbasiertes Gameplay umsetzten. Im Gegensatz zu anderen Videospielen interagiert der Spieler/die Spielerin in einem Visual-Novel über eine direkte Ich-Perspektive mit der Geschichte. Auf Grund des sehr einseitigen Gameplay, welches primär aus Lesen besteht, werden Visual-Novels manchmal sogar von Videospielen getrennt.

## 2.3 Ren.py

Früh in der Eingrenzungsphase dieser Arbeit, fiel die Entscheidung die Visual-Novel Engine Ren’ Py zu nutzen, um schlussendlich das Spiel zu kreieren. Ren’ Py ist eine auf Python basierende Programmier-Engine, welche eine gute Grundlage bietet, um eigene Interaktive Geschichte in Form eines Videospiels zu erstellen. Über 4'000 spiele wurden bereits mit der Engine erstellt und sind zurzeit im Internet erhältlich (REFERENCE: <https://www.renpy.org> ). Spiele wie Doki Doki Literature Club, welches bereits zuvor erwähnt wurde, entstanden ebenfalls mit der Hilfe von Ren’ Py. Weshalb sie auch als Inspiration für mögliche Themen und Spielmechaniken, in Destiny’s Reminder dienten. Zusätzlich wurde schnell ersichtlich, dass die Kombination des Python-codes mit den bereits vordefinierten Befehlen, welche Ren’ Py seinen Nutzern/Nutzerinnen bietet, einen verständlichen und äusserst intuitiven Schreibprozess ermöglichen würde und eignete sich somit perfekt für dieses Projekt.

Ren’ Py stellt für jedes Projekt eine Vorlage zur Verfügung welche zu belieben verändert und auf das geplante Spiel angepasst werden kann. Es erlaubt eigene Bilder einzufügen und macht es einfach diese mit dem Code zu verknüpfen. Dafür erstellt das Programm für jedes Projekt einen Ordner, in welchen jegliche Spieldateien und Bilder abgelegt werden, um später vom Code genutzt zu werden. Dieser Ordner konnte sehr einfach zur Aufzeichnung des Fortschrittes und für Backups in einem GitHub Repository platziert werden, da dieser alle nötigen Dateien enthält, um später das Spiel starten zu können. Der Code, welcher schlussendlich das Spiel bildet, wird von Ren’ Py in vier verschiedene rpy-Dateien unterteilt, welche jeweils einen der Aspekte des Gamedesign Prozesses enthalten. Ren’ Py lässt seine Nutzer/Nutzerinnen die komplette Freiheit, wenn es um die Gestaltung der einzelnen Komponenten eines Spiels geht und kann sogar genutzt werden, um am Ende eine App-Version des eigenen Projektes zu generieren.

## 2.4 Welche Bild Elemente sind benötigt?

Wie bereits erwähnt sind die Bilder und Animationen ein essenzieller Teil eines Visual Novels. Das Darstellen von Bildelementen in Ren’ Py ist grundsätzlich keineswegs limitiert. Es können beliebig viele Grafiken und Animationen dargestellt werden und sogar die Benutzeroberfläche kann selbst angepasst werden.Welche Form die Visuellen Elemente für dieses spezifische Projekt nehmen mussten klärte sich erst nach einem genaueren Blick auf die Funktionen und Prozesse der Ren’ Py-Engine. Um eine genauere Idee zu erhalten welche Illustrationen für einen simplen Visual-Novel wie diesen benötigt werden wurden die Darstellungsweisen der Grafiken in 4 Klassen unterteilt.

Einfache Objekte:

Dies ist die einfachste der 4 Darstellungsweisen und wird benutzt um jegliche nicht interagierbare Objekte darzustellen. Um eine bestimmte Grafik so darzustellen, muss im Code lediglich ihr Name aufgerufen werden. Das bedeutet, dass ein beliebiges Bild, gezeigt werden kann. Man findet hier keine Limitationen bezüglich Form und Grösse. Charaktere wie auch andere einfache Objekte können so mit einer Vielzahl verschiedener darstellungs-Einstellungen, wie zum Beispiel Position, Grösse oder Transparent zu einem Beliebigen Zeitpunkt durch den show-Befehl aufgerufen und durch einen Hide-Befehl wieder versteckt werden. Diese Art des Darstellens limitiert nicht die Anzahl der sichtbaren Objekte. Und diese Können in beliebiger reihenfolge hintereinander stehen.

Benutzeroberfläche:

Diese Art von Grafik ist sehr ähnlich wie die der einfachen Objekte. Im Grunde unterscheiden sich die beiden gar nicht. Um das Aussehen von Elementen wie TextBoxen, Menus oder Icons anzupassen müssen nur die jeweiligen Grafiken im Projektordner ausgetauscht werden. Man kann so komplett selbst entscheiden wie zum Beispiel der Dialog im Spiel dargestellt wird und welche Form, Grösse etc. die TextBoxen und Menus haben. Diese Darstellungsweise wurden jedoch für dieses Projekt in eine separate Klasse eingeteilt, da sie nicht selbst im Code aufgerufen werden müssen und lediglich durch das ersetzten von vordefinierten Objekten erfolgten. Diese Bilder werden nicht durch das Skript, sondern durch einen separaten Teil des Codes, welcher unter GUI gespeichert ist, definiert und wurden um für dieses Spiel nur minimal angepasst, da der Fokus dieser Arbeit auf der Geschichte und dem Umsetzen des eigentlichen Spielerlebnis lag.

Hintergründe:

Wie auch die Elemente der Benutzeroberfläche, ähneln die Hintergründe den einfachen Objekten in vielen Aspekten. Sie werden gleich wie die Einfachen Objekte direkt durch das Skript aufgerufen, haben jedoch ein paar spezielle Eigenschaften. Dank der Weise, in welcher sie aufgerufen werden beeinflussen sie die anderen Grafiken welche zur Zeit vom Spiel dargestellt werden. Wenn ein Hintergrund durch den Befehl Scene aufgerufen wird, sorgt dies dafür, dass jegliche zuvor dargestellte Visuellen Objekte verschwinden. So kann durch einen Scene-Befehl sowohl ein neuer Hintergrund gesetzt werden wie auch direkt eine neue, leere Szene begonnen werden. Hintergründe haben ebenfalls eine Format Anforderung. Um korrekt vom Spiel dargestellt werden zu können müssen die Hintergrund Illustrationen bereits in derselben Grösse und Auflösung wie das Spiel gegeben werden.

Animationen und Videosequenzen:

Die letzte und speziellste Gruppe bilden die Animationen und Videosequenzen. Ren’ Py kann nämlich nicht nur Bilder, sondern auch ganze Videodateien darstellen. Dies geschieht jedoch über einen separaten komplizierteren Befehl, welcher in diesem Projekt nicht angewendet wurde.

# 3. Arbeitsprozess

## 3.1 Konzipierung der Geschichte

Handlung:

Bevor mit dem eigentlichen Schreibprozess begonnen werden konnte, war eine packende Idee nötig, die die Handlungen des Spiels später tragen würde. Diese Idee kam in Form einer bereits begonnenen Geschichte, welche das Kern-Konzept der Täglichen Achievements einführte. Während dem Verlauf des Spiels würde der Hauptcharakter Destiny auf einem für andere unsichtbaren Textfeld erfahren, was am kommenden Tag geschehen würde. Die Geschichte verfolgt über 5 Tagen wie der Hauptcharakter lernt wie mit diesen Prophezeiungen umzugehen ist und erfährt, was sie verursacht. Die klare 5 Tage Struktur eignete sich gut für ein Spiel dieser Art, da die Kapitel artigen Etappen dem Spieler/der Spielerin genügend eine klare Struktur mit Abwechslung bieten und gleichzeitig auch genutzt werden konnten, um einen klassischen Spannungsbogen aufzubauen. Die Tage wurden, wie in einem klassischen 5 Akt Drama unterteilt, um jeweils die Exposition, die Steigende Handlung, den Höhepunkt, die Fallende Handlung und die Lösung des Konflikts einem der Tage zuteilen zu können.

Um den Umfang des Projekts etwas zu limitieren wurde der Fokus der Geschichte auf 5 Hauptcharaktere gesetzt, von denen eine die Spielbare Figur und somit den Spieler/die Spielerin repräsentiert. Im Laufe der Geschichte erfährt die Hauptfigur Destiny von einer Entführung einer der anderen Charaktere, Kai, und versucht dieser Entführung auf den Grund zu gehen. Im Verlaufe des Spiels soll sich dann der Spieler/die Spielerin, basierend auf Hinweisen in den 5 Tagen, entscheiden, welche der restlichen 3 Figuren in der Entführung involviert waren und wie Destiny der Entführten Person helfen kann. Das Ziel soll sein die entführte Person zu retten und den Täter zu schnappen. Die Geschichte sollte zu beginn sehr fröhlich und nichts ahnend sein doch sich dann weiter und weiter verwicklen bis zum Teil sogar eine düstere Atmosphäre entsteht.

Charaktere:

Da ein erheblicher Teil der Geschichte auf den Gesprächen zwischen Destiny und diesen 3 Charakteren beruht, wurde großen Wert darauf gelegt jeden einzigartig zu Gestalten und ihnen eine eigene Persönlichkeit zu verleihen. Destiny’s und die Persönlichkeit der entführten Figur wurden absichtlich eher vage gehalten um Freiraum zu lassen, sich besser in sie hinein versetzten zu können. Die Entscheidungen sollten sich natürlich anfühlen sowohl für den Spieler/die Spielerin wie auch für Destiny als Figur in der Geschichte. Einige ihrer Charakterzüge und Eigenarten wurden deshalb dennoch eingefügt, um ihr eine eigene Persönlichkeit zu geben und ihre Interaktionen mit den anderen Figuren zu leiten.

Destiny ist eine eher schüchterne, aber talentierte junge Frau. Manchmal etwas unordentlich und unorganisiert. Diese manchmal unvorteilhaften Charaktereigenschaften zu zeigen, diente dazu sie dem Spieler/der Spielerin sympathischer zu machen und gleichzeitig ihre Unbeholfenheit in einer neuen Situation zu Illustrieren. Während dem Verlauf der Geschichte erfährt man, dass Destiny vor etwa einem Jahr einen neuen Lebensabschnitt bei einer neuen Firma und einem neuen Wohnort begonnen hatte.

Gabriel, Matthew und Simon, die 3 Figuren welche Destiny in den 5 Tagen antrifft, erfüllen jeweils ein Klischee eines potenziellen Antagonisten. Der Trope des „Besten Freund“, der „Tyrann“, und des „Wolf im Schafspelz“ können alle erfolgreich genutzt werden, um sowohl den wahren Antagonisten zu verkörpern aber auch eine Falsche Fährte zu legen. Sie auf diese sehr oberflächlichen und bekannten Klischees zu reduzieren, machte es einfacher starke und klar motivierte Persönlichkeiten zu kreieren. Ausserdem helfen diese Stereotypen ebenfalls dem Spieler/der Spielerin bereits früh die verschiedenen Charaktere anzuordnen und zu verstehen.

Gabriel ist der „Beste Freund“. Er ist sehr fürsorglich und nett zu Destiny. Er versucht oft das Beste aus einer Situation zu machen und dient so auch als „Comicrelief“ in einigen Szenen. Er und Destiny sind bereits befreundend zum Zeitpunkt zu welchem die Geschichte beginn und verbringt so natürlich am meisten Zeit mit ihr. Je nach Route welche der Spieler/die Spielerin einschlägt hilft Gabriel den Täter zu schnappen, kann jedoch auch verdächtigt und von Destiny genauer verfolgt werden.

Matthew füllt die Rolle des „Tyrann“. Als Destiny’s Chef und Leiter der Abteilung passt seine eher sehr kalte und egoistische Art zu ihm. Er nutzt oft seine Autorität um andere einzuschüchtern und eigene Vorteile zu erziehlen. Er ist ernster als die beiden anderen und zeigt offen, dass er Destiny nicht leiden kann. Seine Nazistische Art kann Verdacht erwecken, doch letztendlich hat er nichts mit der Entführung zu tun.

Simon ist der wahre Täter und ist ein „Wolf im Schafspelz“. Er ist zu beginn sehr freundlich zu Destiny und hilft ihr

## 3.2 Planung der Handlungsstränge

Als die Charaktere und die grobe Handlung des Spiels entschieden waren, galt es die genaueren Details und Verzweigungen der Geschichte festzulegen. Dabei war zu beachten, dass, je stärker eine Entscheidung auf die Handlung wirkte würde oder je häufiger man solche treffen könnte desto persönlicher und somit spannender würde schlussendlich das Spielerlebnis auch sein. Gleichzeitig aber jedoch bei jeder neuen Verzweigung erneuter Schreibaufwand und womöglich neuen Bedarf an Illustrationen entstehen würde. Folglich wurde Wert darauf gelegt die Anzahl und die Konsequenzen der Entscheidungen in einem umsetzbaren Rahmen zu halten und diese gut zu planen, um später Fehler zu vermeiden. Mit einem genauen Plan konnte, während dem Schreiben des Skript-codes der Überblick behalten werden.

Um die Handlung zu planen, eigneten sich erst Mindmaps und später Flussdiagramme, welche die verschiedenen Entscheidungen und die darauffolgenden Pfade klar visuell darstellen konnten. Ein solches wurde auf der Website(!) generiert. Abbildung(!) zeigt dieses detaillierte Flussdiagramm. Es enthält alle größeren und auch einige der nicht so relevanten Entscheidungen. Der Spieler/Die Spielerin kann zu (!) Zeitpunkten, während dem Verlauf des Spiels zwischen zwei oder mehr Optionen wählen und so den weiteren Verlauf der Geschichte beeinflussen. Wie zuvor genannt, gibt es auch mehrere kleinere Entscheidungen, welche keinen oder minimen Einfluss auf den allgemeinen Verlauf der Geschichte nehmen. Diese sind zwar eher insignifikant, ermöglichen jedoch ein individuelleres und interaktiveres Spielerlebnis, ohne den Zeitaufwand relevant zu erhöhen.

Im Ganzen besitzt die Geschichte 11 verschiedene Wege das Spiel zu beenden und vier haupt Endsituationen, welche so erreicht werden können.

Je nach Entscheidung, welche gefällt wurde, trifft ein sehr schlechtes, schlechtes, gutes oder sehr gutes Ende ein.

Das Ziel des Spielers/der Spielerin sollte sein zu einem möglichst guten Ende zu gelangen. Diese Eingrenzung auf 4 Haupt Situationen half, eine fokussiertere Geschichte zu erzählen und minimierte wiederum den nötigen Schreibaufwand. Die 4 Situationen unterscheiden sich in 2 Aspekten voneinander. Einerseits beeinflussen die Entscheidungen, ob die Entführte Person gefunden wird, andererseits bestimmen sie, ob der Täter gefangen wird.

## 3.3 Schreibprozess des Skripts in Ren.py

### 3.3.1 Definieren der Charaktere

Um mit dem Skript zu beginnen, mussten als aller erstes der Charaktere und die Darstellungsweise ihres Textes definiert werden. Um im Dialog zwischen den verschiedenen Figuren zu erkennen wer gerade spricht, werden in einem Visual-Novel meist für jeden Charakter eigene Grafiken, TextBoxen und Text Formatierungen verwendet. Die Formatierung, welche in diesem Schritt zu definieren waren, stellen später im Spiel den Namen wie auch den Gesprochenen Text des Charakters dar. Um mehr Persönlichkeit ausdrücken zu können, können diese in Ren’ Py für jeden Charakter individuell angepasst werden. Unteranderem kann man so die Größe, Schriftart, Farbe, Transparenz und noch viele weitere Optionen von Namen und Text festlegen. Eine weitere Möglichkeit ist es ein TextBox-Design zu definieren, welches angezeigt wird, wenn dieser Charakter spricht. Das Speichern all dieser Einstellungen erfolgt durch das Erstellen einer Variabel, welche später im Code aufgerufen werden kann, um zu entschieden, wie der nachfolgende Text dargestellt wird.

In Abbildung (!) ist ein Beispiel zu sehen in welchem der Charakter Gabriel definiert wurde. Als erstes wurde wie bereits erwähnt eine Variabel gewählt, welche die Formatierungseinstellungen dieses spezifischen Charakters speichert. Durch das Abkürzen des Namens „Gabriel“ zu „g“ konnte der Code später effizienter geschrieben werden, blieb jedoch weiterhin gut verständlich.

Der erste Wert, der auf das Definieren des Charakters folgt, speichert den Namen. Dieser muss als String eingegeben werden und wird auf der TextBox angezeigt, wenn Gabriel spricht.

Mit Hilfe der nachfolgenden „Who“ und „What“ Tags wurden dann die Formatierungen, welche für den Text dieses Charakters zutreffen festgelegt. „Who“ Aussagen beziehen sich auf das Aussehen des Namens, während „What“ Aussagen das Aussehen des gesprochenen Textes verändern. Da Ren’ Py bereits einige Grundeinstellungen besitzt, welche bei nicht definierten Attributen eines Charakters verwendet werden, war das Ändern der Größe und des TextBox-Designs bei diesem Charakter nicht nötig.

A black background with colorful text

Description automatically generated

Abbildung(!)

A white rectangular object with black border

Description automatically generated

Abbildung(!)

Die Abbildung(!) zeigt wie ein Text durch den nun definierten Charakter „Gabriel“ dargestellt wird.

### 3.3.2 Schreiben der Szenen mittels Labels

Um das Navigieren des Codes und das Wechseln zwischen Szenen einfacher zu machen, arbeitet Ren’ Py mit einer eigenen „Label“ Funktion. Wie in Abbildung(!) zu sehen ist wird jeder neue beliebige Textabschnitt von einem Label Statement initiiert. Der Name dieses Labels kann frei gewählt werden und hilft danach diesen Teil des Codes wieder aufzurufen. Labels helfen das Skript zu strukturieren, sind jedoch auch nützlich, um von einer Szene zur nächsten zu springen.

Durch den Befehl „Jump“ kann zwischen den Labels beliebig hin und her gesprungen werden. Labels sind grundsätzlich nicht nötig um eine Geschichte mit Hilfe der Ren’ Py Engine zu schreiben, werden jedoch zu einem der wichtigsten Hilfsmittel, sobald interaktive Elemente eingebaut werden. So kann zwischen den verschiedenen Handlungs-Strängen gewechselt und wieder zu gemeinsamen Szene gesprungen werden, ohne den Text mehrfach im Skript vorhanden haben zu müssen.

A screenshot of a computer

Description automatically generated

Abbildung(!)

Innerhalb dieser Labels konnten nun die verschiedenen Szenen wie in einem gewöhnlichen Skript geschrieben werden. Jede neue Zeile enthält den Text der zu einem Zeitpunkt auf dem Bildschirm angezeigt wird und per Klick wird im Spiel dann zu nächsten gewechselt.

### 3.3.3 Einfügen Interaktiver Elemente

Menus:

Um wie zuvor genannt die Wahl einer Handlung zu ermöglichen, wurden Menus in das Skript eingebaut. Diese funktionieren grundsätzlich sehr ähnlich wie Labels, jedoch mit mehreren von dem Spieler/der Spielerin zu wählenden Optionen.

Durch das Aufrufen eines Menus wird im Spiel eine Auswahl von anklickbaren Optionen angezeigt. Diese können in beliebiger Menge und mit zuvor definiertem Text angezeigt werden. Um eine Option eines Menus zu generieren, wird dem Code ein String gegeben, welcher den Text enthält, der für diese Option angezeigt werden soll. Wird nun eine dieser Optionen angeklickt, springt das Programm zum Code, welcher unter dieser Option steht und springt nach dessen Ausführung, falls keine andere Anweisung gegeben wurde, wieder zum Code außerhalb des Menus. Auf Abbildungen(!) sind Beispiele eines Menus-Befehls und dessen generiertes Interface zu sehen.

Durch solche Menus können direkt kurze Texte ausgegeben werden oder mit Hilfe von Labels zu ganz neuen Handlungsstränge gesprungen werden. In Destiny’s Reminder wurden so alle der in Abbildung (!) aufgezeichneten Interaktiven Stationen in das Spiel eingebaut.

Spieler-input:

Eine etwas kompliziertere weise den Spieler/die Spielerin in das Spielerlebnis mit einzubeziehen kam in der Form des Benennens einer Katze in einem der Abschnitte. Dies konnte mittels Python befehlen getan werden. Mit einem Input Statement kann das Spiel nach einem Text fragen und diesen in einer variabel speichern. Wie in Abbildung (!) zu sehen ist wurde dieser in Destiny’s Reminder in die Variabel „Catname“ untergebracht. Später im Spiel kann dann zum Beispiel bei der Definition des Namens der Katze auf diesen Text zugegriffen werden.

## 3.4 Illustrationsprozess der Spielgrafiken

Mit dem Plan der Handlung und den Anforderungen der 4 Bildklassen, welche während der Recherche Phase definiert wurden, konnten die nötigen Illustrationen nun passend zur Geschichte angefertigt werden.

### 3.4.1 Gestaltung der Charaktere und diverser Spielelemente

Charaktere:

Dank der dialogbasierten Spielweise eines Visual-Novels, sind es die Charaktere, welche die Geschichte zum Leben erwecken. Die Geschichte am Mittelpunkt dieser Arbeit, handelt von fünf sehr unterschiedlichen Personen. Um diesen fünf nicht nur durch ihre Sprechweisen, sondern auch durch ihr Aussehen voneinander unterscheiden zu können wurden für jeden der 5 Charaktere eigene Illustrationen angefertigt. Um die Emotionen der Figuren besser darzustellen und ein abwechslungsreicheres Spielerlebnis zu bieten werden in Visual-Novels oft mehrere verschiedene Illustrationen oder Animationen eines Charakters erstellt. Je nach Gefühl, welches gerade in einer bestimmten Aussage hervortritt, kann dies dann durch eine passende Grafik unterstrichen werden. Welche Emotion schlussendlich öfter gezeigt wird kann von Figur zu Figur variieren und somit helfen deren Persönlichkeit zu verstärken. Mit 4-5 Illustrationen pro Charakter konnten die wichtigsten Gefühlslagen und Gesichtsausdrücke abgedeckt werden.

Jede der 5 Figuren erhielt eine zentrale Farbe. Diese Farbe wurde sowohl für die Illustrationen wie auch das Design des Textes verwendet.

Weitere Spielelemente:

Natürlich Objekte wie zum Beispiel ein Handy oder ein Computer Bildschirm mussten also ebenfalls

### 3.4.2 Gestaltung der Hintergründe

Die Gestaltung der Hintergründe erfolgte in zwei Schritten. Da für die Geschichte relativ viele Hintergründe benötigt wurden, basieren alle Illustrationen auf Bildern welche entweder aus dem Internet heruntergeladen oder selbst gemacht wurden. Diese Art der Gestaltung vereinfachte das Zeichnen glaubhafter und räumlich wirkenden Szenen und minimierte den Zeitaufwand, da von bereits existierenden Strukturen ausgegangen werden konnte und nicht alle Räume von Hand gemalt werden mussten.

Für allgemeinere Orte, welche nur eine Illustration pro Raum beanspruchten, wie zum Beispiel Destiny’s Büro oder die Strasse, wurden Bilder aus dem Internet als Vorlage verwendet. Um jedoch individuellere Orte wie beispielsweise Destiny’s Wohnung von unterschiedlichen Blickwinkeln zeigen zu können, diente Sims4 als nützliches Hilfsmittel um die Räume zu bauen und die verschiedenen Ausschnitte zu fotografieren. In Abbildung (!!) ist das Model Destiny’s Wohnung wie auch der fertige Hintergrund zu sehen.

A cartoon of a kitchen

Description automatically generated

Abbildung(!)

Die schlussendlichen Illustrationen über die gesammelten Bilder erfolgten in Procreate.

Die Konturen der Aufnahmen wurden nachgezogen und allenfalls mit weiteren Objekten oder Änderungen ergänzt. Um ein einheitliches Aussehen für alle Hintergründe zu haben, wurde für das Färben der Bilder das Werkzeug „Verlaufsumsetzung“ verwendet. Dieses wird genutzt, um jedem hell-dunkel Wert neu auch eine bestimmte Farbe zu verleihen. Dies ermöglichte einen schnelleren Malprozess, da für die eher aufwändigeren Hintergründe nun nur noch auf die Helligkeit eines bestimmten Objekts geachtet werden musste. Zu den Illustrationen in Destiny’s Reminder wurden für die verschiedenen Orte eigene Verläufe erstellt und angewendet. In Abbildung(!) sind die einzelnen Verlaufsumsetzungen mit ihren zugehörigen Illustrationen zu sehen. A screenshot of a video game

Description automatically generated

Um mögliche Unstimmigkeiten zwischen Hintergrund und Charakter zu vermeiden, wurden alle Szenen in einem sehr weichen Stil gestaltet. Es wurden hauptsächlich blasse Farben und schwache Kontraste verwendet. Die Farben einer Jeweiligen Umgebung wurden gewählt, um eine bestimmte Stimmung zu erzielen. Für Destiny’s Wohnung hieß das ein ruhiges Grün, für das Büro ein kaltes Blau und für die restlichen Orte ein blasses Rot. Die Hintergründe für die Szenen, welche nur mit der Entführung zu tun haben, wurden schwarz-weiß gelassen, um eine bedrückendere und leblosere Atmosphäre zu kreieren.

### 3.4.3 Gestaltung der Benutzeroberfläche und Icons

## 3.5 Verknüpfen von Skript und Grafik

### 3.5.1 Einfügen eigener Grafiken und Schriftarten

Wie bereits erwähnt erstellt Ren’ Py für jedes neue Projekt einen eigenen Ordner. Um eigene Illustrationen oder Fotos im Spiel verwenden zu können müssen diese lediglich sinnvoll benennt und im dafür vorgesehenen Ordner abgelegt werden.

### 3.5.2 Darstellen der Charakter und weiterer Spielelemente

d

### 3.5.3 Aufrufen der Hintergründe

Um

### 3.5.4 Effekte und Übergänge

Bla

### 3.5.5 Der Feinschliff

# 4.Schluss

## 4.1 Beantwortung der Fragestellung

### 4.1.1 Wie schreibt man eine Interaktive Geschichte?

Eine Interaktive Geschichte unterscheidet sich von einer Linearen Geschichte nur in wenigen Elementen. Im Grunde sind Interaktive Geschichten lediglich eine Sammlung normaler Linearer Geschichten, welche individuell kombiniert und dann nacheinander gelesen werden. Um eine Interaktive Geschichte zu schreiben, muss man deshalb zu Beginn sehr ähnlich vorgehen wie bei einer Normalen Geschichte. Man plant eine zentrale Handlung und baut danach auf diese die verschiedenen Pfade auf. Die simpelste variante eine solche Geschichte zu schreiben, beginnt mit einem der Pfade bis hin zum ersten Ende. Von diesem spaltet man dann die weiteren verzweigenden Handlungen und Enden ab. Falls wie bei dieser Geschichte eines der Enden bevorzugt werden soll, eignet es sich von diesem Pfad auszugehen. Um für alle Pfade eine sinnreiche und Spannende Geschichte zu schreiben, erfordern Interaktive Geschichten natürlich etwas mehr Planung als Herkömmliche, doch schlussendlich beruhen beide auf denselben literarischen Grundlagen und nutzen die selben Rhetorischen und Sprachlichen Mittel.

### 4.1.2 Wie setzt man eine Geschichte in eine Spielbare form um/Welche Methoden gibt es?

Um eine Geschichte in eine spielbare Form umzusetzen, gibt es viele Methoden.

### 4.1.3 Was braucht es, um einen Visual Novell zu kreieren?

Um einen simplen Visual Novel zu erstellen, braucht es grundsätzlich nur ein Skript und simple Grafiken. Es gibt viele Formen, in welchen dieses Genre auftreten kann und von Spiel zu Spiel werden die

### 4.1.4 Wie verstärken visuelle Elemente den Eindruck einer Geschichte?

Die Grafiken und Hintergründe

### 4.1.5 Wie kann ich meinen Code strukturiert notieren (Clean Code)?

Bla.

# 5. Reflexion

## 5.1 Inhalt

Bla.

## 5.2 Methoden

Bla.

# 6. Quellenverzeichnis

**Internetquellen:**

* 1000 gefahren bücher: <https://www.ravensburger.de/de-DE/suche?query=1000+gefahren&productCategories=Ravensburger&productCategories=BRIO&productCategories=Thinkfun>
* Bandersnatch <https://www.ign.com/wikis/black-mirror/Bandersnatch_Endings>
* Ren’ Py <https://www.renpy.org>
* Ren’ Py documentation <https://www.renpy.org/doc/html/>
* <https://videlais.com/2018/07/02/working-with-renpy-part-5-variables-and-conditional-statements/>
* Flowchart maker : <https://app.diagrams.net>
* Github: <https://github.com>
* Visual novel <https://en.wikipedia.org/wiki/Visual_novel>
* Visual novel <https://emmarie.sites.gettysburg.edu/visual-novels/is-a-visual-novel/#:~:text=origins,towards%20one%20of%20multiple%20endings>.
* Persona <https://persona.atlus.com/series/>
* Ddlc <https://ddlc.moe>
* <https://teamsalvato.com>

**Bildquellen:**

* Titelbild: eigenes Bild
* Office hallway: <https://www.flickr.com/photos/markjsebastian/1264424156/>
* Back ally <https://maartenhof.artstation.com/projects/dOGYNK>

# 7. Anhang

Bla