Hochschule für Technik und Wirtschaft Berlin
University of Applied Sciences

Technische Spezifikation

Ein 2D Top-Down Pixel Art Spiel

Autoren des Dokuments : Adib Ghassani Waluya
Minh Hoang Do
Michael Reno
Letzte Änderung : Berlin, 29.07.2020
Dateiname : TechSpec - Dungeorus
Version : 3.0

Inhaltsverzeichnis

Versionshistorie

Version	Datum	Autoren	Bemerkung
0.1	31.05.2020	Michael Reno	Initiale Dokumenterstellung
0.2	05.06.2020	Michael Reno, Adib Ghassani Waluya, Minh Hoang Do	Erweiterung
0.3	08.06.2020	Minh Hoang Do, Michael Reno, Adib Ghassani Waluya	Erweiterung und Erstellung von Diagramme
1.0	10.06.2020	Adib Ghassani Waluya, Minh Hoang Do, Michael Reno	Erweiterung, Erstellung von Diagramme und Fertigung
1.1	06.07.2020	Adib Ghassani Waluya, Michael Reno, Minh Hoang Do	Erweiterung, Erstellung von Diagramme
2.0	08.07.2020	Adib Ghassani Waluya, Michael Reno, Minh Hoang Do	Erweiterung, Erstellung von Diagramme und Fertigung
2.1	24.07.2020	Adib Ghassani Waluya, Michael Reno, Minh Hoang Do	Erweiterung, Erstellung von Diagramme
3.0	28.07.2020	Adib Ghassani Waluya, Michael Reno, Minh Hoang Do	Erweiterung, Erstellung von Diagramme und Fertigung

Vorhandene Dokumente

Dokumente	Autoren	Datum
Lastenheft	Adib Ghassani Waluya, Michael Reno, Minh Hoang Do	29.04.2020
Projektplan	Prof. Dr.-Ing. Mohammad Abuosba, Adib Ghassani Waluya, Minh Hoang Do, Michael Reno	21.05.2020
Pflichtenheft	Adib Ghassani Waluya, Michael Reno, Minh Hoang Do	21.05.2020
Qualitätssicherung	Adib Ghassani Waluya, Michael Reno, Minh Hoang Do	28.07.2020

1 Überblick

Um die Funktionen von Dungeorus zu testen wird ein Android-Smartphone bzw. ein Emulator benötigt, auf dem das Spiel über eine APK-Datei installiert wird.
Nach dem Starten der App öffnet sich das Hauptmenü, in dem der Benutzer wählen kann, ob er ein neues Spiel startet, sich in das “Optionen”-Menü begibt oder sich die Highscores anschaut.
Im “Optionen”-Menü kann die Lautstärke des Spiels eingestellt werden. Wählt der Benutzer ein “Neues Spiel” wird er zur Charakterauswahl weitergeleitet, wo er zwischen mindestens zwei verschiedenen Helden auswählen kann, die verschiedene Fähigkeiten sowie Eigenschaften besitzen.

Daraufhin wird das Level geladen mit den verschiedenen Gegnern. Das Ziel des Benutzers ist es diese zu besiegen, um einen neuen Highscore zu erreichen. Das Spiel endet, wenn der Lebensbalken des Helden auf 0 sinkt.

Während des Spielverlaufs kann der Benutzer Items einsammeln, wie z.B. Masken und Desinfektionsmittel, um sich zu heilen, oder Munition und Waffen, um Fähigkeiten freizuschalten.

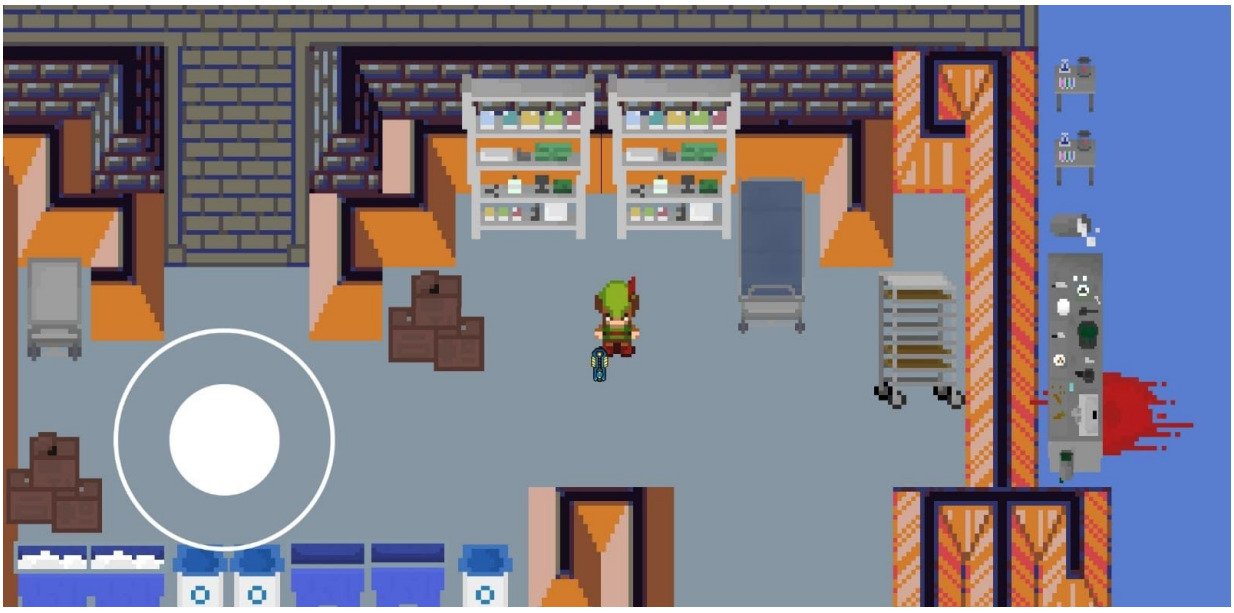

Abbildung 1: Ein Screenshot vom Dungeorus v0.2

2 Prozessüberblick

2.1 Realisierungsprozess

Zur Realisierung dieses Projektes müssen zuerst fertige Dummy-Assets (Platzhalterobjekte) genutzt werden, damit parallel die Skripte geschrieben und die eigenen Assets designed werden können. Es werden unter anderem Skripte für die Bewegung, die Kollision, die Animationen, den Highscore und die Fähigkeiten benötigt.

Nach der Fertigstellung der eigenen Designs werden diese animiert und in das Spiel implementiert. Zum Schluss wird das Spiel als APK-Datei exportiert und kann somit auf Android-Geräten installiert werden.

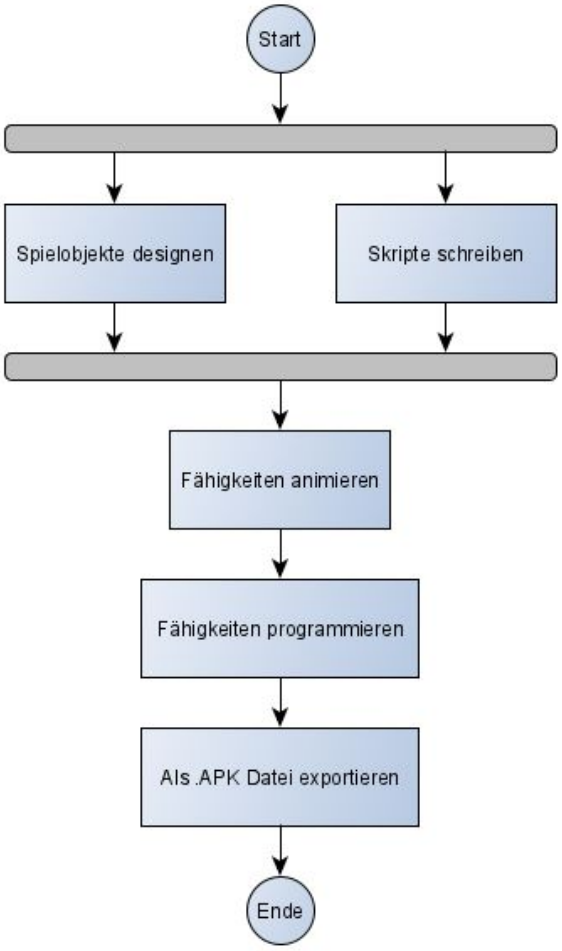

Abbildung 2: Diagramm des Realisierungsprozesses

2.2 Fachlicher Workflow

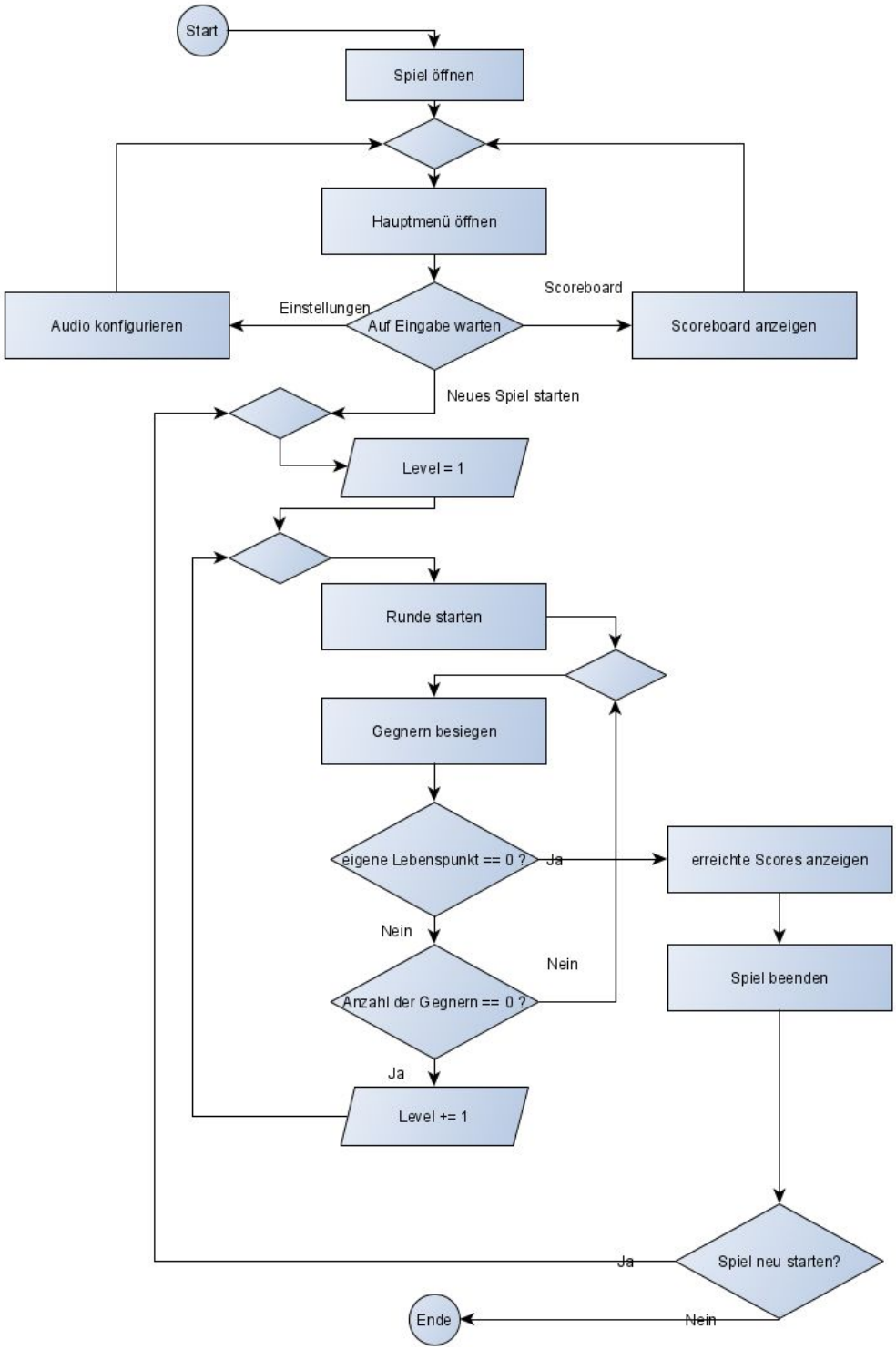

Abbildung 3: Fachlicher Workflow

2.3 Technischer Workflow

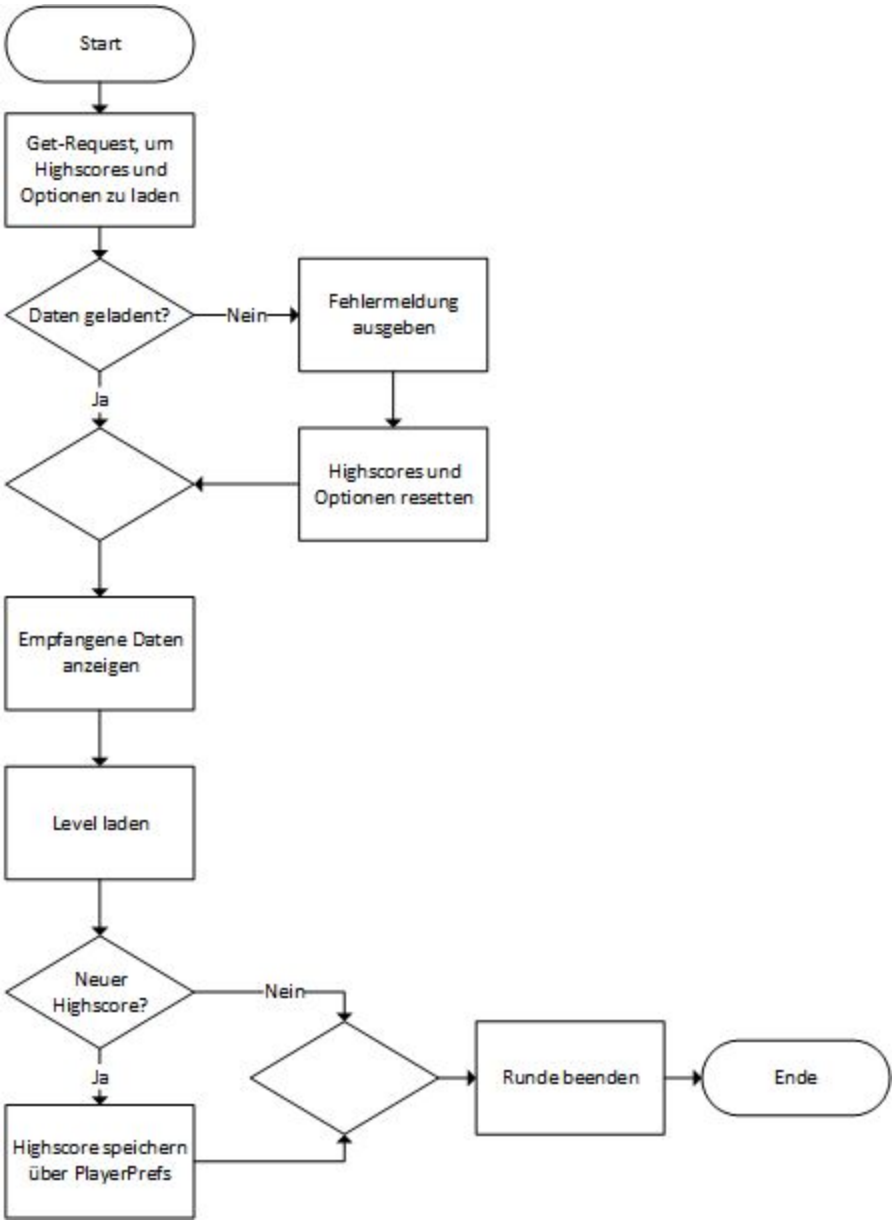

Abbildung 4: Technischer Workflow

3 Systemarchitektur und Infrastruktur

3.1 Systemarchitektur

Die folgende Abbildung zeigt es, wie das System mit dem Benutzer verbindet.

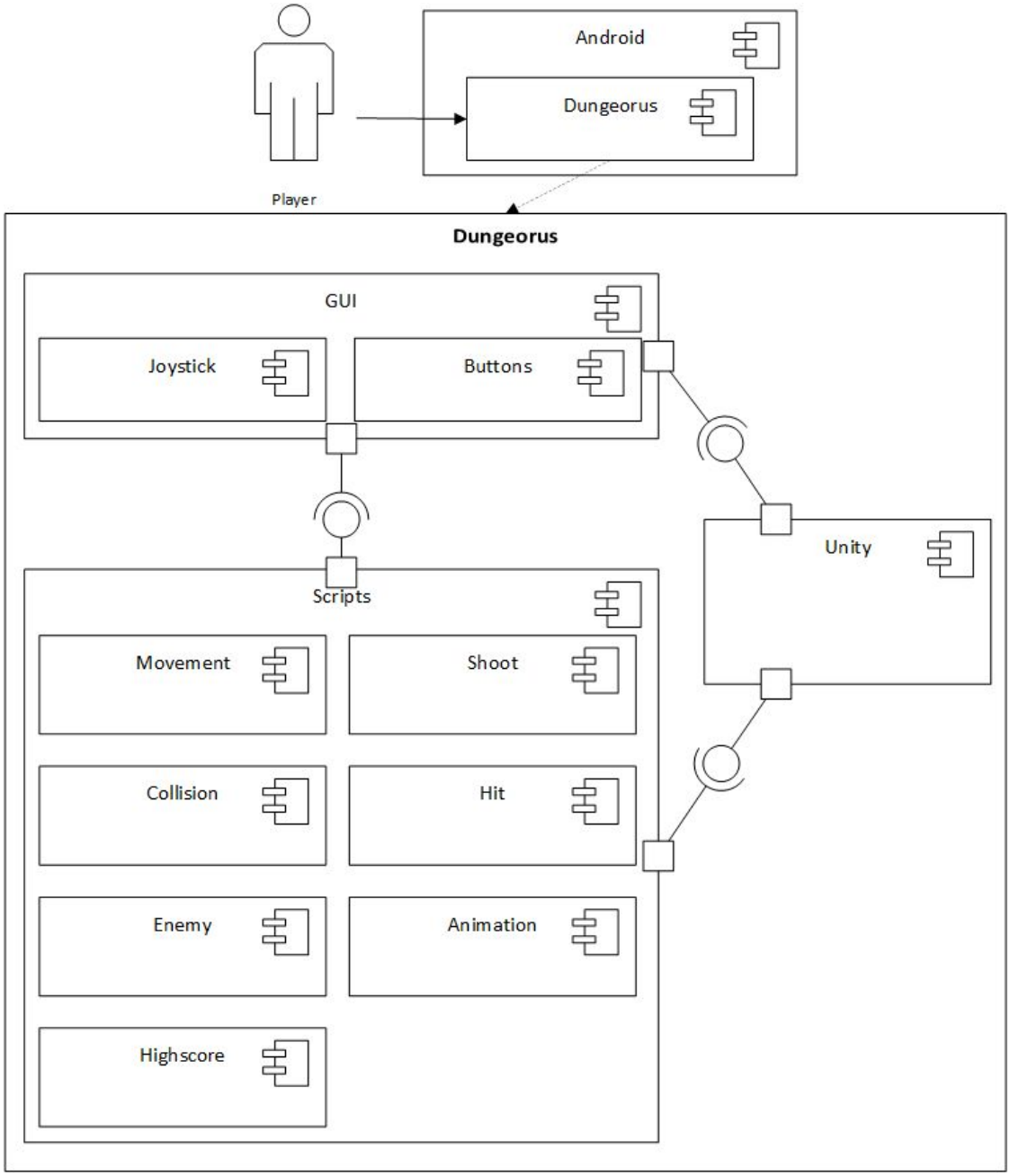

Abbildung 5: Systemarchitektur Diagramm

3.2 System Infrastruktur

Das Spiel läuft lokal auf dem Gerät ohne Multiplayer-Feature, daher wird kein Backend-Service benötigt. Zum Speichern des Highscores, wird auch keine externe oder lokale Datenbank genutzt, da Unity eine Funktion dafür schon integriert hat. Diese nennt sich “PlayerPref” und stellt Funktionen für das Speichern von Daten bereit.

4 Spezifikation Software

4.1 Überblick Komponente

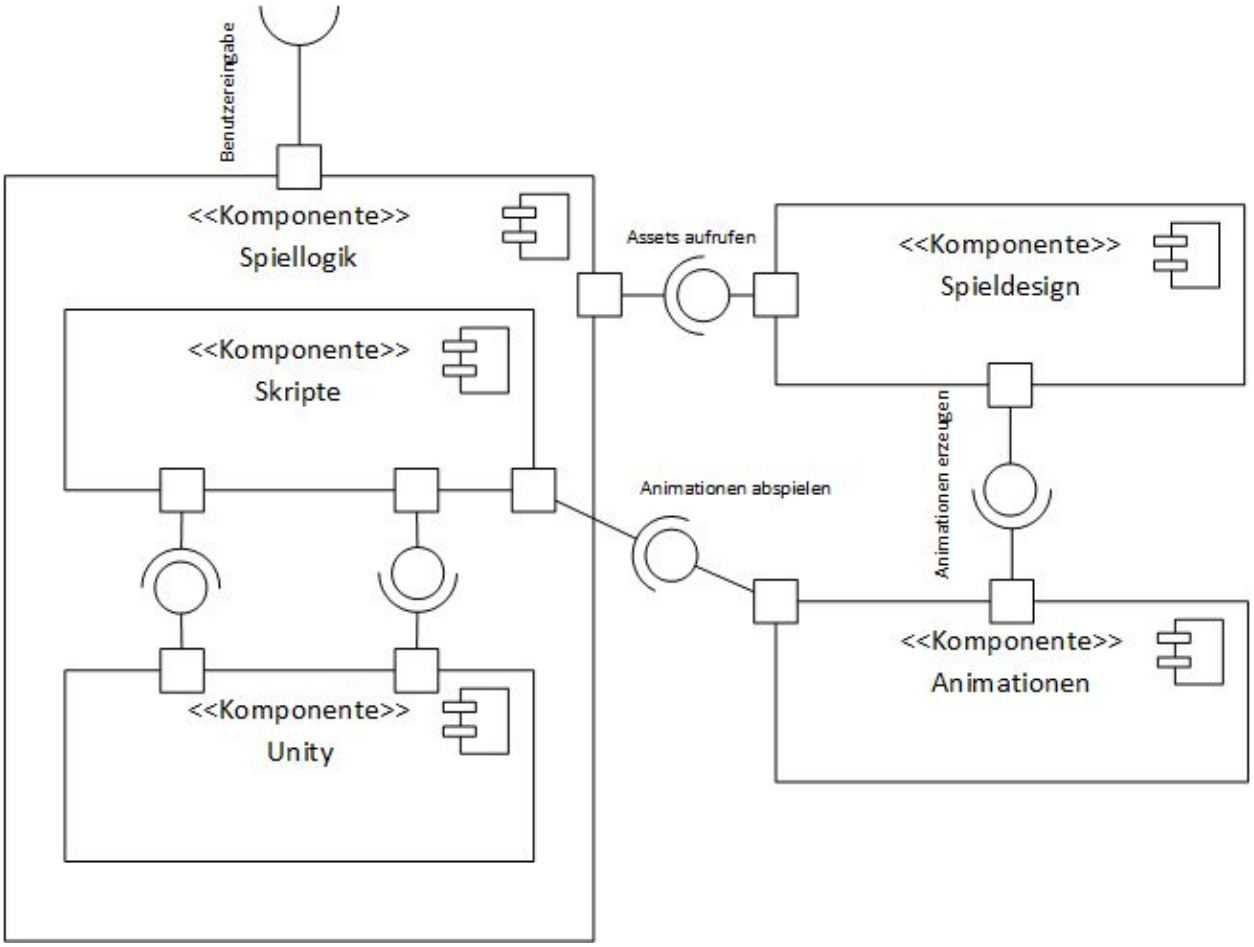

Abbildung 6: Komponentendiagramm

SW Komponente	Funktionen	Sprache / Typ
T1 - Spiellogik	F1 - Neues Spiel starten	C# Skripte, .cs
	F2 - Pause-Menü aufrufen	
	F3 - Charakter auswählen	
	F4 - Charakter bewegen	
	F5 - Fähigkeiten einsetzen	
	F6 - Schlagen	
	F7 - Schießen	
	F10 - Item aufsammeln	
	F11 - Charakter heilen	
	F12 - Gegner erzeugen	
	F13 - Highscore berechnen	
T2 - Spieldesign	F8/9 - Charaktere / Items / Gegner / Levels designen/anlegen	PNG Datei, .png Unity Szene-Datei, .unity
T3 - Animationen	F8/9 - Charaktere / Items / Gegner / Levels designen/anlegen	anim-Datei

Tabelle 1: Beschreibung der Softwarekomponenten und deren Funktionen

4.2 Beschreibung der Implementierung

4.2.1 F1 - Neues Spiel starten

Vom Hauptmenü kann der Spieler mit einem Klick auf den “Neues Spiel starten” Button ein neues Spiel beginnen. Daraufhin wird der Spieler zur Charakterauswahl (F3) geleitet, wo er einen von mehreren Helden wählen kann.
Danach wird das Level geladen und der Spieler kann seinen Helden steuern.

#	Komponente	Erforderliche Arbeiten
T1	Spiellogik	Das Level wird immer beim Starten eines neuen Spiels auf eins gesetzt.
T2	Spieldesign	Die Map und alle Spielobjekte werden abhängig vom Level geladen.

Tabelle 2: Komponenten F1

Abbildung 7: aktuelles Design vom Hauptmenü

4.2.2 F2 - Pause-Menü aufrufen

Innerhalb des Levels kann der Spieler über einen Button das Pause-Menü aufrufen. Mit dem Aufruf des Pause-Menüs wird das Spiel automatisch pausiert. Darin kann der Spieler entscheiden, ob er das Spiel fortsetzt, beendet oder ein neues startet. Wenn er das Spiel fortsetzen möchte, wird das Pause-Menü geschlossen und das Spiel fortgesetzt. Wenn das Spiel beendet wird, kommt der Spieler zurück zum Hauptmenü. Wenn der Spieler sich entscheidet, das Spiel neu zu starten, wird er automatisch zum Charakterauswahl weitergeleitet.

#	Komponente	Erforderliche Arbeiten
T1	Spiellogik	Beim Aufrufen des Pause-Menüs wird das Spiel pausiert und der Score für die Fortsetzung des Spiels gespeichert.
T2	Spieldesign	Die Szene vom Pause-Menü wird aufgerufen und geladen.

Tabelle 3: Komponenten F2

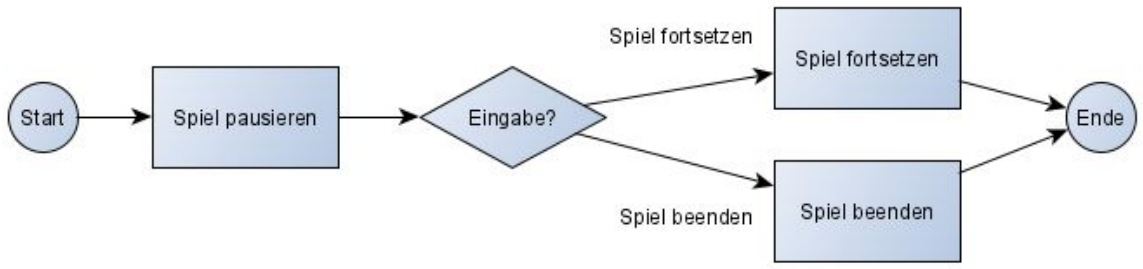

Abbildung 8: Flowchart F2 - Pause-Menü aufrufen

Abbildung 9: Design von In-game Pause Menü

4.2.3 F3 - Charakter auswählen

Der Spieler kann sich in diesem Menü einen von mehreren Charakteren auswählen. Diese unterscheiden sich sowohl im Aussehen als auch von den Fähigkeiten und Eigenschaften wie Lebenspunkte und Bewegungsgeschwindigkeit. Der ausgewählte Charakter kann während des Spiels vom Spieler gesteuert werden.

#	Komponente	Erforderliche Arbeiten
T1	Spiellogik	Der ausgewählte Charakter wird als Parameter für die Fähigkeiten eingesetzt.
T2	Spieldesign	Der ausgewählte Charakter soll als ein Spielobjekt geladen und im Spiel angelegt werden.

Tabelle 4: Komponenten F3

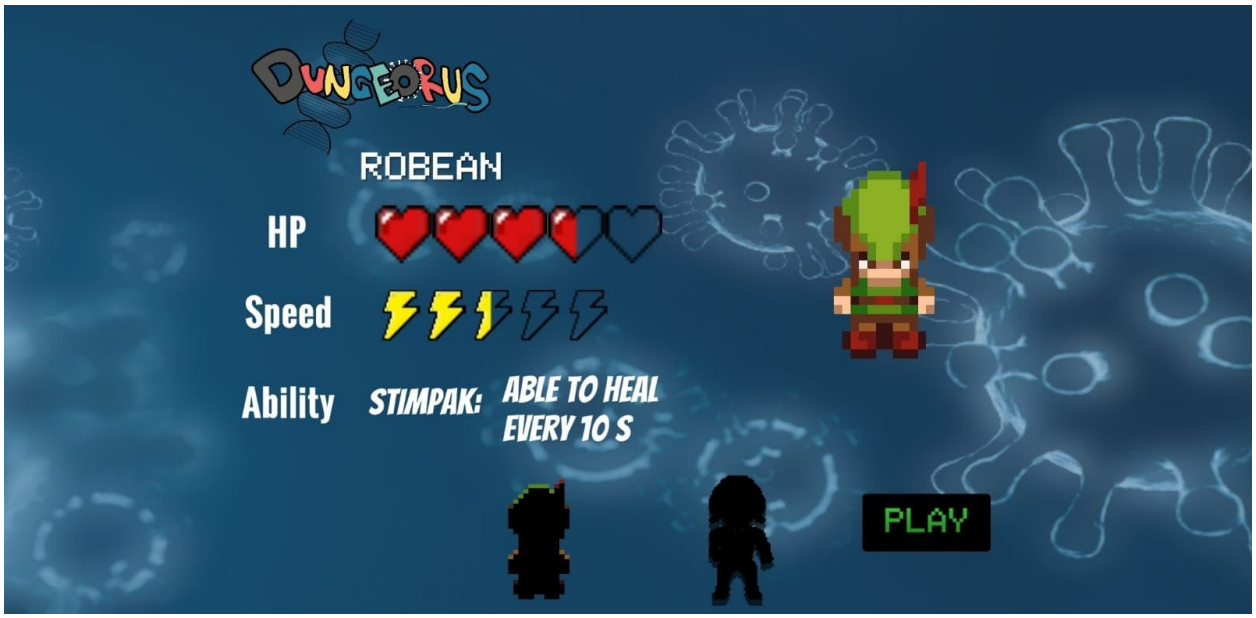

Abbildung 10: Design bei Auswahl von Charakter

4.2.4 F4 - Charakter bewegen

Zur Steuerung des Charakters wird ein digitaler Joystick verwendet. Die Bewegung des Joysticks in x- und y-Richtung wird als Eingabe für die Bewegung eingesetzt.

#	Komponente	Erforderliche Arbeiten
T1	Spiellogik	Die Bewegung des Joysticks wird als die Eingabe für die Bewegung des Charakters eingenommen.
T2	Spieldesign	Die entsprechende Animationen für den Charakter werden mit der Bewegung angepasst. Gleichzeitig soll es in der Szene geprüft werden, ob der Charakter durch ein Hindernis geht oder eine freie Strecke.

Tabelle 5: Komponenten F4

Abbildung 11: Eine Bewegungsanimation, die aus mehrere .png Dateien besteht

4.2.5 F5 - Fähigkeiten einsetzen

Jeder Charakter hat eigene Fähigkeiten, die der Spieler während des Spiels einsetzen kann. Das Einsetzen wird durch das Klicken auf den Fähigkeiten Button gestartet. Verschiedene Fähigkeiten haben verschiedene Effekte im Spiel. Man kann zum Beispiel selbst heilen, oder schneller bewegen, und so weiter.

#	Komponente	Erforderliche Arbeiten
T1	Spiellogik	Die Fähigkeiten, die zur Verfügung gestellt werden, sind von dem ausgewählten Charakter (F3) abhängig.
T2	Spieldesign	Die richtige Animationen für die Fähigkeiten wird aufgerufen und auf die Szene geladen. Außerdem stehen auch die Fähigkeiten im Charakterauswahlmenü.

Tabelle 6: Komponenten F5

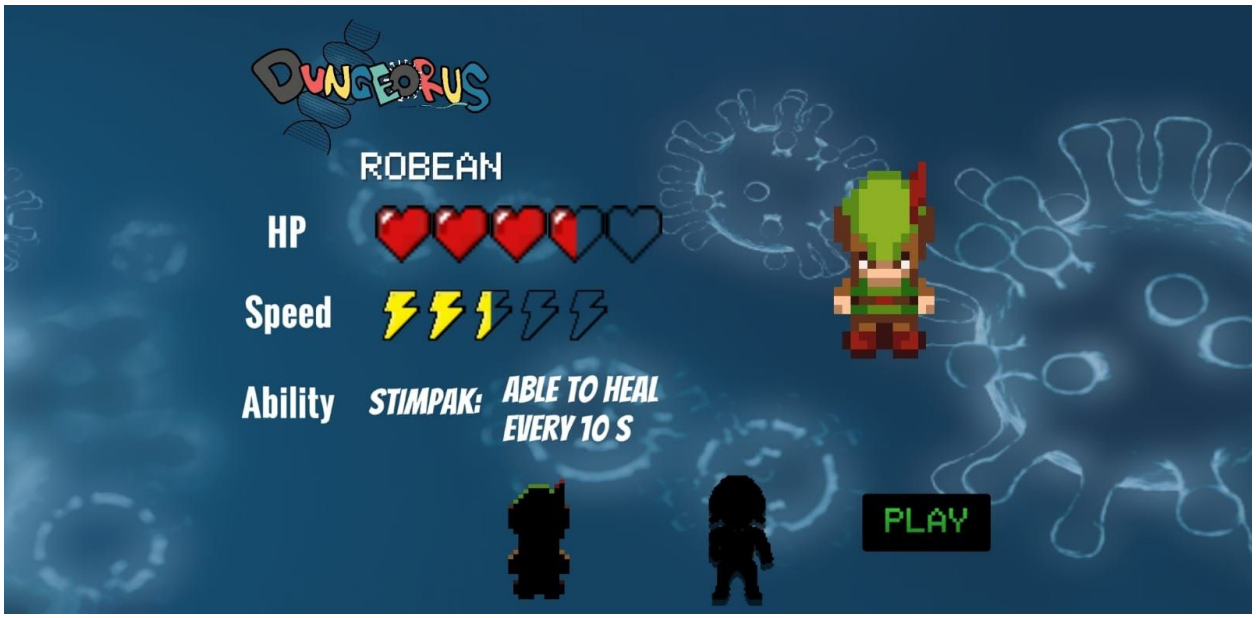

Abbildung 12: Die Fähigkeiten vom Charakter, die im Charakterauswahlmenü angezeigt werden

4.2.6 F6 - Schlagen

Der Charakter kann mit dem Button “Schlagen” Gegner besiegen. Jedes mal wenn die Funktion aufgerufen wird, wird die Animation zum Schlagen geladen. Der Spieler kriegt für jede gelandete Attacke Punkte.

#	Komponente	Erforderliche Arbeiten
T1	Spiellogik	Die Funktion wird nur nach der Eingabe vom Spieler aufgerufen. Die Richtung der Attacke ist von der Eingabe des Spielers abhängig.
T2	Spieldesign	Die richtige Animationen für das Schlagen wird aufgerufen und auf die Szene geladen.

Tabelle 7: Komponenten F6

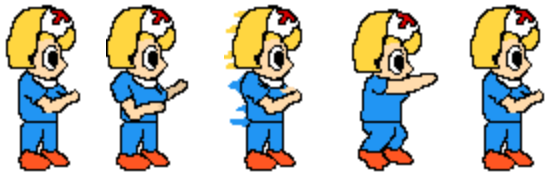

Abbildung 13: Eine Schlagen Animation, die aus mehrere .png Dateien besteht

4.2.7 F7 - Schießen

Der Charakter kann mit dem Button “Schießen” Gegner besiegen. Jedes mal wenn die Funktion aufgerufen wird, wird die Animation zum Schießen geladen. Der Spieler erhält für jede gelandete Attacke Punkte.

#	Komponente	Erforderliche Arbeiten
T1	Spiellogik	Die Funktion wird nur nach der Eingabe vom Spieler aufgerufen. Die Richtung der Attacke ist von der Ausrichtung des Charakters abhängig.
T2	Spieldesign	Die richtige Animationen für das Schießen wird aufgerufen und in die Szene geladen.

Tabelle 8: Komponenten F7

Abbildung 14: Schussanimation von einem Charakter

4.2.8 F8 - Charaktere / Items / Gegner / Levels designen

Charakter-, Item- sowie Gegner-Assets müssen gezeichnet und animiert werden. Diese Assets werden als .png Datei gespeichert. Die Levels werden aus vorhandenen Assets erstellt.

#	Komponente	Erforderliche Arbeiten
T2	Spieldesign	Jeder Charakter und Gegner soll einfach erkennbar und unterscheidbar sein. Die Fähigkeiten des Charakters sollen möglichst in Verbindung mit dem Charakter (wie Sportler kann sich schneller bewegen). Für jeden Charakter sollen Bewegungs-, Attacke- und Fähigkeiten-Animationen gezeichnet werden. Für jedes Level werden die vorhandenen Assets und die Designs von Charaktere, Items sowie Gegner in verschiedenen Sortiergruppen (Sorting Group) geordnet.
T3	Animationen	Die Animation soll mit dem Charakter und der Eingabe des Spielers angepasst werden.

Tabelle 9: Komponenten F8

Abbildung 15: Design von einem Charakter

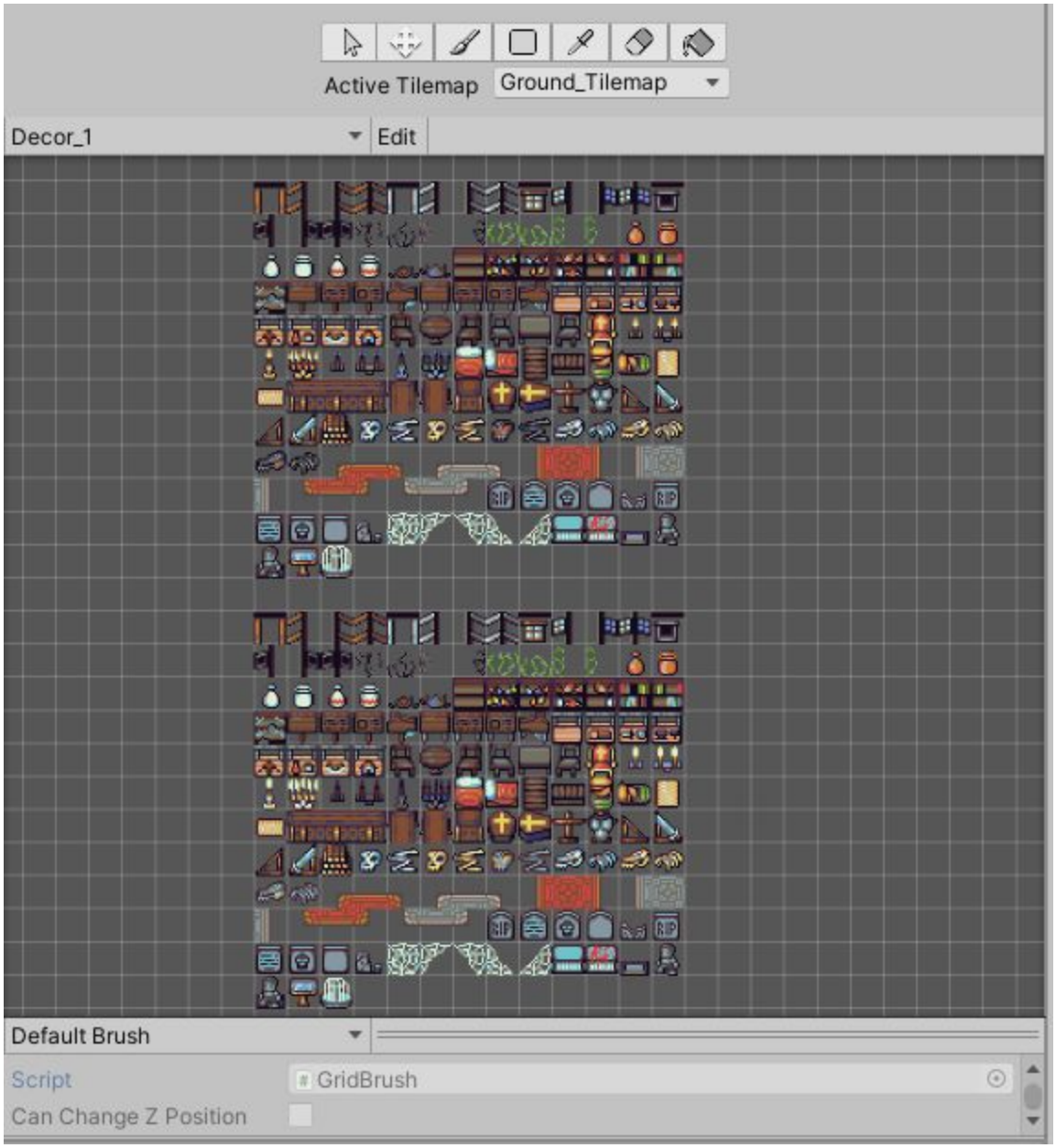

Abbildung 16: die vorhandenen Assets für Dungeorus Spielumgebung

4.2.9 F9 - Charaktere / Items / Gegner / Levels anlegen

Die gezeichnete Charakter-, Item- sowie Gegner-Assets von F8 werden in Unity im entsprechenden Assets Ordnern angelegt. Diese Assets kann man während der Spielentwicklung benutzen und in die Mappe anlegen. Dazu kann man zu jeden Assets ein spezielles Eigenschaft einstellen (wie Kollisionslogik, Animationen usw.).

#	Komponente	Erforderliche Arbeiten
T2	Spieldesign	Die unterschiedlichen Kollisionen, die abhängig von der Sortiergruppe jedes Assets und Designs sind, werden für jedes komplett gezeichnetes Level umgesetzt.
T3	Animationen	Jedes Sprite von jedem Design und Asset wird anhand seiner Sortiergruppe richtig gerendert.

Tabelle 10: Komponenten F9

Abbildung 17: Spielumgebung von Level 1

Abbildung 18: Implementierung von Kollisionen

Abbildung 19: Implentierung von Sprite Renderer (Sortiergruppe vom Tisch > Sortiergruppe von Charakter)

4.2.10 F10 - Item aufsammeln

Ab und zu sollen Items im Spiel zur Verfügung gestellt. Solche Items können aus der Mappe kommen (wie in einem Box) oder von den besiegten Gegnern genommen werden. Die Items sind Hygieneartikeln, die der Spieler aufsammeln und benutzen kann, um die Fähigkeiten seines Charakters zu erhöhen, oder die Gegnern zu besiegen.

#	Komponente	Erforderliche Arbeiten
T1	Spiellogik	Items können aufgesammelt werden, nur wenn es direkt neben dem Charakter ist.
T2	Spieldesign	Jedes Item hat ein erkennbares Design, das auf Hygieneartikeln basiert.

Tabelle 11: Komponenten F10

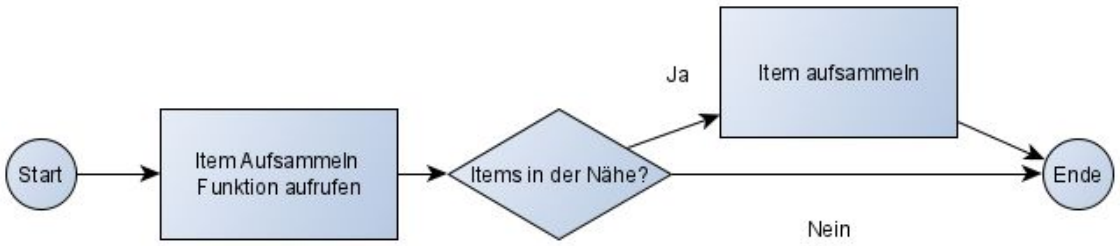

Abbildung 20: Flowchart F10 - Item aufsammeln

4.2.11 F11 - Charakter heilen

Der Charakter kann mit dem “Heilen”-Button seinen Lebensbalken erhöhen. Jedes mal die Funktion aufgerufen wird, wird die Animation zum Heilen geladen.

#	Komponente	Erforderliche Arbeiten
T1	Spiellogik	Die Funktion wird nur nach der Eingabe vom Spieler aufgerufen. Die Funktion erhöht das Lebensbalken des Charakters.
T2	Spieldesign	Die richtige Animationen für das Heilen wird aufgerufen und auf die Szene geladen.

Tabelle 12: Komponenten F11

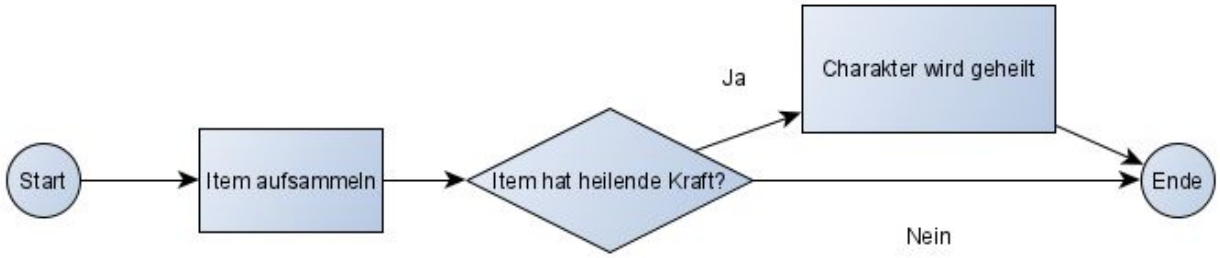

Abbildung 21: Flowchart F11 - Charakter heilen

4.2.12 F12 - Gegner erzeugen

Innerhalb des Levels sollen verschiedene Gegner generiert werden, die der Benutzer zu besiegen hat. Anhand dessen soll der Highscore berechnet werden (F13). Die Gegnern sollen automatisch erzeugt werden. Der Anzahl der zu erzeugenden Gegnern sind vom Level abhängig.

#	Komponente	Erforderliche Arbeiten
T1	Spiellogik	Die Gegnern sollen nach dem Start des Spiels rhythmisch erzeugt werden. Die sollen nicht auf einmal erzeugt werden und der Anzahl hängt vom Level ab.
T2	Spieldesign	Es gibt nur ein paar Eingänge, wodurch die Gegnern erzeugt werden können. Die Gegnern sollen auch animiert werden, wie die Charaktere.

Tabelle 13: Komponenten F12

Abbildung 22: Bewegungsanimation von einem Gegnertyp (CoVID-19) aus mehrere .png Dateien

4.2.13 F13 - Highscore berechnen

Sowohl für jeden besiegten Gegner als auch für jede zehn überlebende Sekunde kriegt der Spieler 10 Punkte. Die Punkte wird in der Skripte geschrieben und wird automatisch berechnet. Als das Spiel endet, wird die erreichte Punkte angezeigt, und falls es besser als die vorherige erreichte Punkte, wird diese Punkte im Scoreboard angezeigt und gespeichert.

#	Komponente	Erforderliche Arbeiten
T1	Spiellogik	Die Berechnung der Punkte wird beim Aufruf des Pausemenü automatisch pausiert. Für jede 10 Sekunde, die der Spieler überlebt, kriegt der Spieler 10 Punkte. Falls das Spiel innerhalb der 10 Sekunde beendet wird, wird keine Punkte vergeben.
T2	Spieldesign	Der Spieler kann die erreichte Punkte auf dem Bildschirm sehen. Die angezeigte Punkte soll mit der Punkte im System gleich sein.

Tabelle 14: Komponenten F13

Gegner besiegt (in n)	**Score (in integer)**
1	10
2	20
n	10 * n
Zeit (in s)	-
10	10
15	10
20	20
n	n (wird aber nur jede 10 Sekunde addiert)

Tabelle 15: “Der Spickzettel” für die Berechnung von Scores

5 Backlog (aktualisiert)

In Sprint 2 haben wir ein paar Änderung zum Backlog gemacht. Geplant war die Schlagen Funktion (F6) in Sprint 2 zu machen, aber wir haben uns beschlossen, statt dieser Funktion die Pause-Menu Aufruf Funktion (F2) zu erschaffen. Der Grund dafür war es, um alle gebrauchte Szene im Spiel so früh wie möglich zu fertigen. Die Änderungen zum Backlog ist in diesem aktualisierten Backlog mit rotem Kreuz gekennzeichnet.

Funktionalitäten	Sprint 1	Sprint 2	Sprint 3
F1 - Neues Spiel starten	x		
F2 - Pause-Menü aufrufen		x	
F3 - Charakter auswählen		x	
F4 - Charakter bewegen	x		
F5 - Fähigkeit einsetzen		x	
F6 - Schlagen			x
F7 - Schießen		x	
F8 - Charaktere / Items / Gegner / Levels designen	x	x	x
F9 - Charaktere / Items / Gegner / Levels anlegen		x	
F10 - Item aufsammeln		x	
F11 - Charakter heilen			x
F12 - Gegner generieren			x
F13 - Highscore berechnen			x

Tabelle 16: Backlog