Konzeptdokumentation
von Erwin Oudomvylay und Thanh Pham

Einleitung
Wir haben uns überlegt, dass wir eine Experience im Berich „Unterdrückung“ ansprechen wollen. Dabei soll die Message der Mechanic sein, dass man während der Unterdückung beschränkt ist, was die Steuerung angeht. Spricht, man kann nur in bestimmte Richtungen laufen und man darf sich nicht schneller fortbewegen als man eigentlich will.

Ebenfalls soll die Situationen der individuellen Charaktere in Form von Dialogen dargestellt werden, weshalb sehr viel Wert auf die Erstellung vom Dialogsystem gelegt wurde. Auf World Building und die dabei entstehende Erschaffung der Welt war die Priorität sehr hoch.

Wir wollten einen Kontrast am Anfang und am Ende des Spiels haben. Am Anfang hat man die komplette Kontrolle über den Charakter und man kann sich frei rumbewegen.
Im zweiten Level ist man direkt beschränkt.
Am Ende hat man wieder die Kontrolle über den Character.

Somit ist die Idee für ein Top-Down 2D Exploration in den Sinn gekommen. Man hat vier verschiedene Richtungen und man muss nach dem Hauptziel suchen. Wenn man mit NPCs spricht, geben sie einem Tipps, was man ungefähr als nächstes machen soll.

Geschichte
Am Anfang der Zeit war die Welt einst, alle haben in kompletter Balance gelebt. Jedes Dorf war unabhängig von anderen und niemand wollte in die Affäre anderer eingreifen.

Dies hat angehalten, bis die ersten Kristalle auf der Welt entdeckt wurden. Es besteht aus 8 Elementen, die andere Eigenschaften mit sich geben. Diese besaßen Kräfte Jenseits der Welt; Mit dieser Kraft konnte man andere überzwingen und unterdrücken.
Durch die Konkurrenz vermehrte sich die Interesse und der Anreiz mehrerer Menschen. Dies löste negativen Emotionen auf die Welt aus. Dadurch kam ein Überwesen, der dies alles beenden wollte, damit die Balance der Welt erhalten bleibt: Der Dämonenlord.

Niemand war zufrienden mit dem Dämonenlord, da er als böse angesehen wurde und die ganze Kraft für sich haben wolle. Aufgrunddessen haben sich alle zusammengetan und nur die Qualifiziertesten wurden auserwählt, gegen den Dämonenlord zu kämpfen: Die vier Legenden.

Am Ende des alten Zeitalters kämpften sie zu viert gegen ihn. Nur mit der Macht der verschmolzenen Kristalle konnte der Dämonenlord versiegelt werden. Jedoch gab es kein Ende ohne Unfälle; Durch die Verschmelzung der Kristalle kam es zu einem starken Impuls, sodass alle vier gefallen sind.

Die Kristalle wurden in viele einzelne Fragmente aufgeteilt. Nun sind sie überall in der Welt verteilt, jedoch sind die meisten bis heute unentdeckt. Bis heute leben die meisten in Harmonie und Balance, jedoch gibt es noch einige, die sich nach den Kristallen sehnen und diese ergattern wollen.

Steuerung

Die Steuerung wird direkt am Anfang des Title Screens gezeigt, welches man mit der linken Shifttaste ein- oder ausblenden kann. Im Default wird die Steuerung angezeigt für den Fall, dass man das Spiel startet, ohne einen Blick auf die Steuerung gehabt zu haben.

Mit der Leertaste oder der Z-Taste kann man Aktionen vollführen (z.B. bestätigen, untersuchen, sprechen etc.).
Mit WASD oder den Pfeiltasten kann man den Player bewegen.
Mit der linken Shifttaste gehalt gedrückt kann man sprinten.

Die Bewegungssteuerung WASD wurde genommen, da dies die Standardsteuerung in PC Games ist.
Dementsprechend wurde wurde zum Bestätigen für die Leertaste entschieden, da man mit WASD in der Regel die Leertaste als Sprungstaste verwerndet. Da dies ein 2D Top-Down Spiel ist und in dem Genre in den seltensten Fällen keine Sprungmechanik existiert, wurde für die Leertaste entschieden.

Schon mit dem Title Screen hat man einen guten Überblick, was die Steuerung ist. Mit den auf dem Title Screen stehenden Tasten muss man fortfahren, damit man mit dem Spiel anfangen kann.
Die einzige Ausnahme wäre die Z-Taste, die im Folgenden Abschnitt näher erläuter wird, weshalb für diese Taste entschieden wurde.

Bei der Z-Taste muss man anmerken, dass es sich um das QWERTY Keyboard-Layout handelt. Jedoch kann man mit jedem Layout spielen, es ist nur die Position der Z-Taste gemeint, was links vom X ist. Beim QWERTZ Layout (deutschsprachige Tastatur) ist somit die Y-Taste gemeint, was beim QWERTY Layout der Z-Taste entspricht.

Der Grund für die alternative Wahl der Z-Taste, ist, dass die meisten 2D RPG Games die Pfeiltaste als Steuerung nutzen und die Z-Taste als Bestätigung verwendet wird. Für diejenigen, die das Muscle Memory als RPG Player haben, ist diese Steuerung viel angenehmer als die WASD Steuerung.

Wenn man RPG Player ist, muss man nicht direkt Umdenken, was die Z-Taste ist, denn es ist immer die Position links von der X-Taste gemeint. Wenn man als Non-RPG Player bezüglich der Z-Taste verwirrt ist, kann man immer auf WASD + Leerstaste zurückkehren.

Die linke Shifttaste ist die Default Sprinttaste. Ob man Leertaste oder Z-Taste als Aktion verwendet, der linke Kleinfinger liegt immer natürlich auf der linken Schifttaste.

Alle Steuerungstasten sind während des Spiels verfügbar. Man muss nicht auswählen, für welche Steuerung man sich entschieden muss.

Playtesting
Am 09.01.2024 wurde an der Hochschule Ansbach das Spiel getestet. Die ungewohnte Steuerung aus Pfeiltaste und der Z-Taste war für viele schwer, sich dran zu gewöhnen, weshalb die alternativen Steuerungen WASD + Leertaste eingeführt wurde.

Zugleich war der Stein, der hinter der Pflanze versteckt war, zu schwer aufzufinden, weshalb der Fels direkt rechts beim Spawnen angezeigt wird.
Ebenfalls sagt nun Bob, kein optionaler NPC mehr, was die linke Shifttaste aussagt.

Mit einem Icon über dem Character weiß man nun als Player, dass man interagieren kann.
Mehrere Tipps wurden dem Player per Dialogen implementiert, was man wo machen kann.

Wenn Sie den Prototyp selber playtesten wollen, müssen Sie die Szenenvariable „sceneCount“ auf 3 setzen, damit Bob den Player am Anfang von Level 03 folgt. Ansonsten kann der Wert immer 3 bleiben, da keine andere Startbedingungen abhängig von der Variable sind, bevor Unity die Variablen abliest und dann in Laufzeit bringt und der Script „PartyMember“ diese Variable zu spät mitnimmt.

Animation Controller

Charaktere
Es wurde eine Spritesheet Animation für Charaktere genommen. Es gibt vier Richtungen zwischen den Player Sprite und dem Soldier Sprite. Beide haben eine Idle und eine Walk Animation, jedoch hat der Soldier zusätzlich eine Run Animation. Je nach Situation wird die passende Animation abgespielt.

Beide Animation Controller verwenden den Blend Tree. Die Variable „animationValue“ hat drei Bedingungen:
animationValue = 0: Idle animation
animationValue = 1: Walk animation
animationValue = 2: Run animation
Die horizontal und vertical Variablen verändert die Blickrichtung je nach Input des Players mit WASD / Pfeiltaste.

UI
Die UI hat primär die Funktion, den Canvas mit flüssige Übergängen auszustatten. Das Spiel soll damit ein natürliches Aussehen hervorheben.

Grafik
Wir haben uns für Pixelgrafik, bestehend aus einer Dimension von 32x32, entschieden, da es einen simplen Look haben soll. Weil der Fokus auf die Geschichte, die Charaktere und die Welt liegt, wie in viele RPGs mit Pixelgrafik, wurde dafür entschieden.
Es sollte zwischen Menschen und Tieren unterschieden werden, damit man als Player hineininterpretieren kann.

Assets
Alle Assets wurden von itch.io heruntergeladen und besitzen eine Lizenz von CC0.
Unter Assets/GameAssets sind alle Assets vorhanden und geordnet.

Hierarchy Structure in Unity

Mit leeren GameObject wurde die Hierchy ordentlich sortiert, um den Überblick beizubehalten.

Party
Hier sind der Player und die Party Member drin.
Alle GameObjects besitzen entweder das Script „Player“ oder „PartyMember“.
Ausnahme: Bob hat am Anfang von Level 01 den NPC Script und nachdem man mit ihn geredet, hat wechselt der Graph zu PartyMember.

Interactable
Hier kommen alle interagierbaren Objekte rein. Unterteilt wird zwischen Props, NPCs und Idle.
Soldaten, die unter Interactable sind, sind nicht feindlich und geben keine Game Over Bedingung.
Alle GameObjects besitzen entweder das „NPC“ oder „Prop“ Script.

Enemies
Hier kommt der Enemy Sprite rein mit der Logik für die Enemy AI.
Alle GameObjects besitzen die State Machine „Enemy“.

Manager
Hier sind alle Manager drinnen mit deren Graph Skript passend zum Namen. Dies beeinflusst die ganze Logik des Spiels.

Hud
Die UI für den Canvas kommt hier rein mit dem Script „HUD“.

Camera
Bei der Camera wurde die 2D Cinemachine Camera verwendet.

Location
Hier werden die Positionen der Orte gespeichert, wo der Player nach einer Room Transition teleportiert wird.

Grid
Hier sind die ganzen Tilemaps drin.

Static
Alle Objekte, die sich in der Szene nicht bewegen.

Lighting
Hier wurde das Light 2D verwendet.

End
Das „End“ ist ein Trigger für den Escape in Level 03.

Szenen
Der Prototyp beinhaltet ein Title Screen und 3 Level: Level 01 – 03. Hier werden die einzelnen Details der Level beschrieben.

Level 00
Der Title Screen führt der Player ein mit einer Steuerungserklärung. Nach dem Screen erhält man eine simple Story über den Main Character und sein Ziel.

Level 01
Gameplay Mechanic
Man kann mit WASD / Pfeiltasten laufen und mit Shift sprinten.
Man kann mit NPCs und Objekten interagieren. Wenn der Player Sprite Richtung Interactable blickt, dann erscheint ein Icon. Mit Leertaste / Z-Taste kann man die Interaktion ausführen.

Walkthrough
Finde Bob und rede mit ihn. Er lebt südöstlich vom Dorf. Man muss die Brücke in der Mitte überqueren. Danach kann man den großen Fels links von Tanukis Haus zerstören. Außerhalb der Mauer kommt man zwei Weg entlang. Wenn man Richtung Süden läuft, sieht man einen blauen Kristall. Beim Interagieren läuft eine Cutscene und es lädt das zweite Level.

Extras
Alle NPCs besitzen zwei Dialoge: Bevor und nachdem man mit Bob geredet hat. Die Dialoge passen sich dementsprechend an.
Shin und Pete haben eine Wette.
Ken liest ein Buch über die Legende.
Das Crypt gibt einen Tipp, wie man weiterkommt.
Die Statuen südlich vom Dorf erzählen von dem Kampf mit dem Dämonenkönig.
Tarti hat ein startText, obwohl man nie die Möglichkeit hat, mit ihm reden zu können, bevor man mit Bob geredet hat. Falls man es als Player irgendwie schafft, außerhalb des Dorfes zu gelangen ohne Bob in der Party zu haben und man mit Tarti redet, wurde ein passender startText geschrieben, sodass man kein Softlock bekommt.

Level 02
Gameplay Mechanic
Man kann während des Level nicht sprinten.
Auf dem Weg zur Mine hin und zurück ist man auf eine Bewegungsrichtung beschränkt.
In der Mine muss man mehrmals auf ein Objekt drücken, damit etwas passiert.

Walkthrough
Rede mit dem Soldat Gretel im Camp und laufe zur Mine. Sammle insgesamt 8 Erze, wobei in der Mitte 5 Erze und oben rechts 3 Erze sind. Jedes Erz muss 3x angeklickt werden, damit es erfolgreich aufgesammelt wird. Rede mit Hansel vor dem Mineneingang und gib die Erze ab. Nun kann man die Mine verlassen und zurück zum Camp laufen. Rede mit Bob im Camp und es lädt das nächste Level.

Extras
Alle NPCs besitzen zwei Dialoge: Bevor und nachdem man 8 Erze gesammelt hat.
Der zweite Dialogtext von einigen NPCs weist draufhin, dass man alle 8 Erze hat, falls man den letzten Text beim Erze sammeln überlesen hat.

Paul und Klein sind Brüder.
Kaneki gibt dem Player Tipps, was im nächsten Level zu machen ist, nachdem man 8 Erze gesammelt hat
Jack listet die 8 Attribute aller einzelnen Kristalle auf.
In der Lagerhalle sind Statuen von der Legende und man kann damit interagieren.
Nicht alle Soldaten behandeln die Raccoons wie das Letzte.

Level 03
Gameplay Mechanic
Man kann wieder in jede Richtung laufen und sprinten.
Man kann Tore und Schatztruhen öffnen.
Soldaten bewachen die ganze Map und man muss sich durchschleichen.

Waklthrough
Der Weg links, oben und rechts wird abgedeckt von Soldaten. Man muss unten rum und dann im Gegenuhrzeigersinn laufen. Südwestlich ist die Schatztruhe, was ein Schlüssel besitzt. Danach kann man das Tor im Osten öffnen und entkommen.

Extras
Das Lighting ist dunkler, da es die Nachtvariante ist.
Der Eingang zur Mine wurde abgesperrt.
Wenn man ganz südlich beim gelben und blauen Haus ist und nach links läuft, kann man die Schatztruhe auf der anderen Seite des Flusses sehen.
Man kann nochmal mit der Schatztruhe interagieren, nachdem es schon geöffnet wurde.

Visual Script

Im Folgenden werden alle Visual Scripts ins Detail beschrieben.
Es wird unterteilt zwischen Graph Scripts unter Assets/Graphs und Embed Scripts in Level 00.

CutsceneManager
Der CutsceneManager ist nur in Level 01 betroffen.
Hier wird die Bedingung nur abgespielt, wenn der Player mit dem GameObject „Crystal“ interagiert. Zugleich kann man sich danach nicht mehr bewegen und man schaut eine Zwischensequenz an.

DialogueManager
Hier werden alle Dialoge behandelt. Wenn man mit Sachen interagiert, wird dieses Skript aufgerufen. Alle Buchstaben werden einzeln abgebildet. Man kann nur auf das nächste Textdialog schalten, wenn der ganze Text angezeigt wird. Wenn man die Leertaste / Z-Taste drückt während durch den Buchstaben iteriert wird, wird der ganze Text direkt angezeigt, falls man ausversehen nochmal interagiert hat oder schneller den Text durchlesen will. Wenn die Aot Liste vom getroffenen Interactable den Endindex erreicht hat, wird der Dialog beendet und ein Sound abgespielt.

DialogueText
Dies ist der Subgraph für beide Skripte NPC und Prop. Alle GameObjects besitzen eine Aot List Variable, damit alle einzelnen Items in der Liste gespeichert sind und der passende Dialog zum Interactable angezeigt wird. Zugleich weiß der DialogueManager dann auch bescheid, wie viele Indexes in der Liste vorhanden sind, damit es den Dialog erfolgreich beendet.

Enemy
Die State Machine hat drei Zustände: Idle, Patrol und Catch. Hier wird die Enemy AI behandelt.
Wenn man den Tag „EnemyIdle“ hat, wird die Blickrichtung vom Enemy gar nicht beeinflusst. Das sind für stationäre Soldaten, die bestimmte Weg blocken damit man gezwungen ist, einen anderen Weg zu gehen.
Der Tag „EnemyIdleSwitch“ hat den einzigen Unterschied, dass die Blickrichtung des Soldaten sich in die andere Richtung verändert.
Der Tag „EnemyPatrol“ wechselt den Zustand zwischen Idle und Patrol.
Wenn der Raycast auf die Layer Player oder PartyMember trifft, wechselt es zum Catch und es kommt zum Game Over.

GameManager
Hier wird die ganze Logik aus alle Szenen, abgesehen von Level 00, beeinflusst.
Bei einem Switch on Integer mit dem „sceneCount“ als Variable weiß der GameManager immer, in welcher Szene welches Event ausgeführt werden soll.
Der Custom Event „raycastCheck“ überprüft nach dem Layer vom getroffenen GameObject und führt dann andere Events aus.
Nachdem der Dialog fertig ist, wird „afterDialogue“ aufgerunfen. Je nach treffender Bedingung wird ein bestimmtes Event ausgeführt. Wenn das GameObject den Tag „Destroy“ besitzt, wird das Event „destroyObject“ ausgeführt und diese werden dann zerstört.
Die lilane Box in der Mitte des Graphs bezieht sich auf die Szene, wenn man von der Mine zum Camp zurückläuft. Wenn man vor dem Camp steht, wechselt die Richtungsbeschränkung von links nach oben.
Der „gameOver“ Event wird nur ausgeführt, wenn man in Level 03 vom Soldat entdeckt wird und der Game Over Screen wird angezeigt. Nach paar Sekunden kann man das Level neustartenn wenn man die Leertaste / Z-Taste drückt.
„levelDone“ lädt das nächste Level.

Ganz unten in der schwarzen Box ist die Logik für das Ende in Level 03, nachdem man das Tor überquert hat.

HUD
Beim einmaligen Aufrufen von „enableHud“ werden die Textpanels und der Dialog angezeigt. Es displayed nur den Namenpanel an, wenn das getroffene GameObject vom Player den Tag „NPC“, „Idle“ oder „PartyMember“ besitzt und holt den Text. Es schaltet die HUD aus, wenn der Index das letzte Item die Aot Liste ist.

Icon
Wenn der Player auf ein Layer mit dem Raycast trifft, wird ein Interaktionsicon über dem Character angezeigt und soll drauf hinweisen, dass man mit dem Object interagieren kann.

MineOre
Alle Erze besitzen 3 HP. Mit jedem Klick sinkt der Wert um 1 und spielt ein Sound ab. Nachdem man 3x geklickt hat, spielt der Sound für das Zerstören ab. Ein Dialog wird angezeigt mit dem Fortschiritt, wie viel Erze man schon gesammelt hat. Am Ende wird destroyObject aufgerufen und der destroyCounter vom GameManager sinkt um 1. Wenn es 0 erreicht hat, hat man die Mining Quest abgeschlossen und man kann zurück zum Camp laufen, nachdem man mit Gretel gesprochen hat.

NPC
Dieser Graph hat die Logik aller NPCs. Der „idleNPC“ Event, wenn man mit dem NPCs spricht, wird deren Spriterichtung entgegengesetzt der Richtung des Players gesetzt, damit es so aussieht, als würden die beiden sprechen. Nachdem der Dialog fertig ist, wird mit „resetIdle“ die ursprüngliche Richtung des NPCs zurückgesetzt.
Der „getText“ Event wählt zwischen verschiedenen Dialogen aus, welche angezeigt wird je nach GameState in der Szene.
Wenn man ein „Idle“ Tag besitzt, bleibt die Blickrichtung der NPCs behalten. Dies hat die Auswirkung, dass zwei verschiedene NPCs eine Konversation haben oder ein NPC selbstgespräche führt.

PartyMember
Der Party Member folgt dem Player, solange man nicht in Reichweite des Players ist. Das ist der Grund, weshalb der Player zwei Colliders hat: Einmal der normale Collider, der etwas kleiner ist und ein Trigger um den Playerumfang. Man folgt dem Player so lange, bis man den Trigger erreicht und dann stehenbleibt, sodass die Distanz zwischen beiden relativ konstant bleibt.
Man kann durch den PartyMember durchlaufen, falls entweder Player oder PartyMember in einer Ecke feststecken und das Game softlocked.

Player
Hier kann man den Player steuern. Mit dem State Graph „RaycastDirection“ wird die Raycast Richtung des Players nach Bewegungseingabe ausgerichtet. Alle Interaktionen sind vom Raycast des Players abhängig. Anfangs wurde ein Trigger um alle Interactables rum angelegt. Wenn man in Range ist, kann man interagieren. Jedoch hieß es auch, dass man mit den NPCs reden konnte, ohne in deren Richtung zu schauen. Zugleich kam es zu Errors, da der GameManager nicht wusste, was es machen soll, wenn der Player in Range von mehreren Interactables ist.
„RestrictDirection“ ist ein weiterer State Graph im Script, was verantwortlich für die Logik der Bewegungsbeschränkung in Level 02 ist.
Man kann sich nicht bewegen, wenn man in einem Dialog ist oder jegliche Form von Cutscene abgespielt wird.

Die „animationValue“ ist abhängig vom Input der Steuerungen mittels GetAxisRaw. Es wurde für GetAxisRaw entschieden, da der Player eine flüssige Kontrolle ohne auswirkende Physik haben soll. Ebenfalls sind Physikbewegungen vom Player in 2D Top-Down Games selten vertreten.
Mit jeder Bewegung spielt es Sounds für Fußsschritte ab.

Prop
Die Logik für diesen Skript ist fast identisch mit dem NPC Skript, weil beide ein „getText“ Event besitzen. Da ein Prop im Default kein Dialogwechsel hat, bleibt der Text konstant. Die einzigen Ausnahmen sind bestimmte GameObjects mit bestimmten Namen in bestimmten Szene.

RaycastPlayer & RaycastEnemy
Die Logik für den Raycast wurden von der Vorlesung GameDesign von Prototyp 03 entnommen. Die beien Skripte unterscheiden sich nur darin, dass Player und Enemy andere Layer als RaycastHit annimmt.

RaycastLayer
Das ist ein Subgraph für den Player und PartyMember. Hier wird die Renderreihenfolge mit einer Raycastlogik angewendet.

SoundManager
Hier wird die ganze Logik für alle Sound abgespielt.

StopRedirection
Dies hat die Logik, dass man die Bewegung und den Raycast vom Player während bestimmten Bedingungen nicht neuausrichten kann.

TransitionManager
Dieser Graph ist nur in Level 02 aktiv. Wenn man in bestimmte Trigger eintritt, wird man während eines Fadescreens in den nächsten Raum transportiert. Hier werden ebenfalls die Steuerungseinschränkungen für verschiedene Räume definiert.

GameManager (Embed Script Level 00)
Dieser Script umfasst die Logik für die Szene in Level 00. Größtenteils wird die UI animiert und entsprechende Logik angewendet, wenn man die Leertaste / Z-Taste drückt.

IntroText (Embed Script Level 00)
Diese Graphs ist eine Vereinfachung vom Script Graphen „DialogueManager“ passend zur Szene. Diesmal ist es nicht abhängig vom Player Raycast sondern es bleibt ein konstanter Text.

Vorhandene Bugs

Party Member Bewegung
Bob hat eine bibbernde Bewegung, was seine Laufanimation komisch darstellen lässt.

Layer Rendering
Dies ist bezogen auf das Script RaycastLayer. Player und Party Member besitzen jeweils dieses Script als Subgraph. Wenn man mit einem folgenden Party Member unter einen GameObject mit dem Tag „House“ steht, wird die falsche Reihenfolge gerendert und es hat den anschein, dass er Character zwischen dem Haus und dem anderen Character im falschen Layer steht.

Single Raycast
Der Player und die Enemies sind abhägnig vom einem single Raycast. In Level 03 kann man theoretisch durch ein Enemy durchlaufen, da der Raycast sehr dünn ist. Ein Fix wäre, dass man ein Cone Raycast oder mehrere Layer Raycast anstelle von einem Einzelnen nutzen würde. Dies hat auch Folgen, dass wenn der Player an der unteren linken oder echten Ecke vom Haus steht, falsch gerendert wird.

Einzelne Leistungsnachweise von Erwin Oudomvylay (Code)
Um das Coding hat sich Erwin gekümmert. Alle Codes wurden mittels Visual Scripting erstellt. Alle Tutorials wurden als Referenz genommen und dann in eigener Bolt Sprache verformt. Alle C# Referenzen wurden danach ins Visual Scripting übersetzt.

Einzelne Leistungsnachweise von Thanh Pham (Art)
Die Grafikauswahl aller Assets, die Layouts der Maps, die Tilemaps und die dazugehörigen Collision wurden von Thanh gemacht. Da es ein 2D Spiel ist, mussten viele Layer für einzelne Tilemaps kreiert werden, damit diese dementsprechend passend im Spiel aussehen.

Fazit
Leider wurden kaum Game Over Bedingungen implementiert, was eigentlich die Message der Mechanic aussagen soll. Ein Feedback, dass, wenn man zu oft mit Soldaten redet, man bestraft wird. Oder wenn man sich weigert, zur Mine zu gehen und vom Weg abweicht, dass man eine Verwarnung bekommt und bei kontinulierlicher Flucht man einen Game Over bekommt.
Viel Bug fixing wurde betrieben wenn eine neue Mechanik implementiert wurde und nahm sehr viel Zeit weg.
Dennoch hoffen wir, dass Ihr durch unser Prototyp eine Experience mitnehmen könnt und natürlich Spaß habt.