**Game Design Document**



Projekt Knossos

Prototyp

**Letztes Update:**

02.07.2021

**Autoren:**

Franz Mörike,

René Kraus,

Jan Pagel

**Inhaltverzeichnis**

[Game Analyse & Konzept 3](#_Toc80899842)

[Genre 4](#_Toc80899843)

[Plattformen 4](#_Toc80899844)

[Zielgruppe 4](#_Toc80899845)

[Gameplay 4](#_Toc80899846)

[Gameplay Überblick 4](#_Toc80899847)

[Gameplay Mechaniken 5](#_Toc80899848)

[Items 5](#_Toc80899849)

[Sound/Music 5](#_Toc80899850)

[Performance enhancing 6](#_Toc80899851)

[KI 6](#_Toc80899852)

[Kontrollschema 8](#_Toc80899853)

[Spiel Ästhetik / Design & Benutzeroberfläche 10](#_Toc80899854)

[Skripte 11](#_Toc80899855)

[Arbeitsaufteilung 11](#_Toc80899856)

[Quellen 12](#_Toc80899857)

[TODO Liste 13](#_Toc80899858)

# Game Analyse & Konzept

**Grundidee:**

Der Spieler hat sich in einem unterirdischen Labyrinth verlaufen (Abb. 1). Dort warten gefährliche Untote auf ihn und versuchen ihn zu töten und seine Seele aus dem Leib zu reißen. Der Spieler versucht vor diesen zu fliehen und den Ausgang aus dem verwirrenden Kellergebilde zu finden. Die Flucht gestaltet sich aber nicht ganz so einfach, da das Labyrinth diverse böse Überraschungen für den Spieler parat hat. Das Labyrinth ist an die griechische Mythologie um das Labyrinth von Knossos angelehnt.

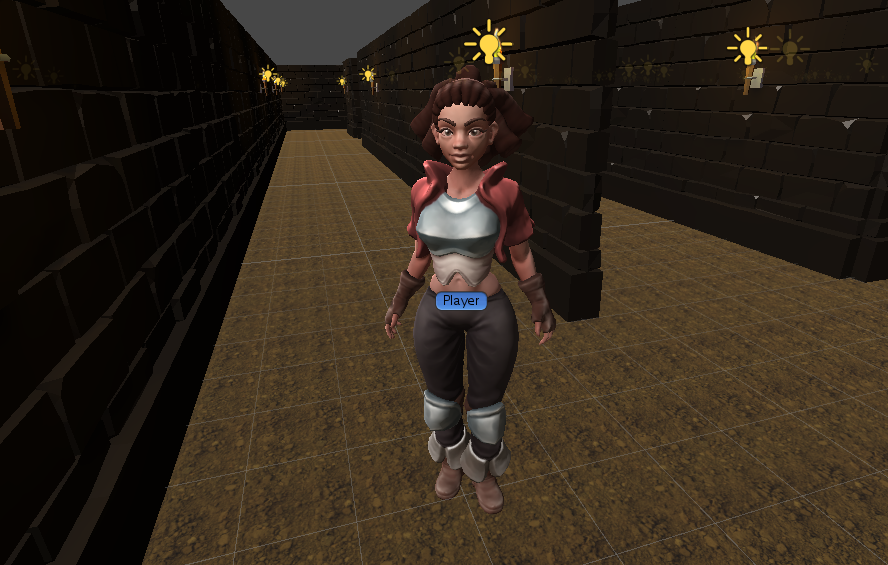


Abb. 1 Spieler inmitten des Labyrinths

**Skriptreferenzen:**

* GameController.cs

# Genre

* Creepy Survival Adventure

# Plattformen

* PC

# Zielgruppe

Da die Steuerung simpel gehalten ist, ist das Spiel auch für unerfahrene Spieler geeignet.

Es soll vor allem Spieler ansprechen, die sich gerne erschrecken und sich für fast ausweglose Labyrinthe begeistern.

# Gameplay

## Gameplay Überblick

Das Gameplay besteht grundsätzlich darin, aus dem Labyrinth zu entkommen und zu überleben, damit die Untoten nicht die Seele des Spielers fressen. Außerdem muss der Spieler taktische Lösungen finden, um sowohl die Untoten zu umgehen als auch die Fallmechaniken auszutricksen.

Die Fallen die dem Spieler das Überleben erschweren können:

* Falltüren
* Einstürzende Decken
* Wände, die den Spieler zerquetschen

Um das Gameplay noch interessanter zu gestalten, sind *progressive gameplay mechanics* geplant, die dafür sorgen, dass der Spieler gewisse Schritte erfüllen muss, um im Level weiterzukommen und schlussendlich den Ausgang zu finden. Diese beinhalten Items die im Labyrinth gefunden werden können und dem Spieler beim Überleben helfen. Der Schlüssel wird benötigt, um schlussendlich entkommen zu können.

## Gameplay Mechaniken

|  |  |
| --- | --- |
| **Game Modus** |  |
| Überlebensmodus | Der Spieler versucht aus dem Labyrinth zu entkommen. |

**Skriptreferenzen:**

* HiddenDoor\_Behavior.cs
* Portal\_Behavior.cs
* TrapDoor\_Behavior.cs
* TrapRocks\_Behavior.cs

## Items

Die Items helfen dem Spieler beim Überleben. Sie werden benutzt, indem man die dem Platz im Inventar entsprechende Taste drückt.

|  |  |
| --- | --- |
| **Items** |  |
| Heiltrank | Heilt den Spieler |
| Ausdauertrank | Gibt dem Spieler kurzzeitig mehr Ausdauer |
| Maps | Karten vom Labyrinth |

## Sound/Music

Alle in der GameScene vorhanden AudioSource werden durch einen AudioMixer gesteuert. Durch die Unterteilung in Gruppen lassen sich die einzelnen AudioSources leicht und übersichtlich verwalten. Da nicht alle Audiodateien permanent abgespielt werden, besitzen einige einen Trigger in Form eines leeren GameObject mit einem BoxCollider. Dieser kann einmalig durch den Spieler ausgelöst werden und zerstört sich danach selbst.

Das Master Volumen des AudioMixer wird durch das AudioManager Script gesteuert.

**Skriptreferenzen:**

* AudioManager.cs
* AudioTrigger.cs

# Performance enhancing

Um die Performance des Spiels zu verbessern, werden einige Anpassungen vorgenommen (Abb 2/3).

* Für die Lichtquellen (z. B. Fackeln) an den Wänden werden „lightmaps“ generiert.
* Die Performance ziehenden Mesh Collider werden durch Box Collider ersetzt.
* Zum Rendern des Levels wird Occlusion culling eingesetzt. Dadurch werden nur die Teile des Levels gerendert, die auch vom Spieler eingesehen werden.

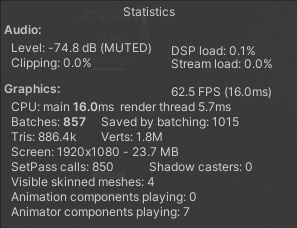


Abb. 2 Werte vor der Optimierung

Abb. 3 Werte vor der Optimierung

# KI

Das Verhalten der Untoten wird mit dem EnemyAI.cs Skript gesteuert. Dieses bestimmt die Bewegungsabläufe und -richtungen anhand von Pathfinding-mechaniken (Pathfinding.cs). Der für die KI begebbare Bereich wird durch die NavMesh bestimmt. (Abb.4/5/6 NavMesh).



Abb. 4 NavMesh vom Default Level



Abb. 5 NavMesh vom Default Level

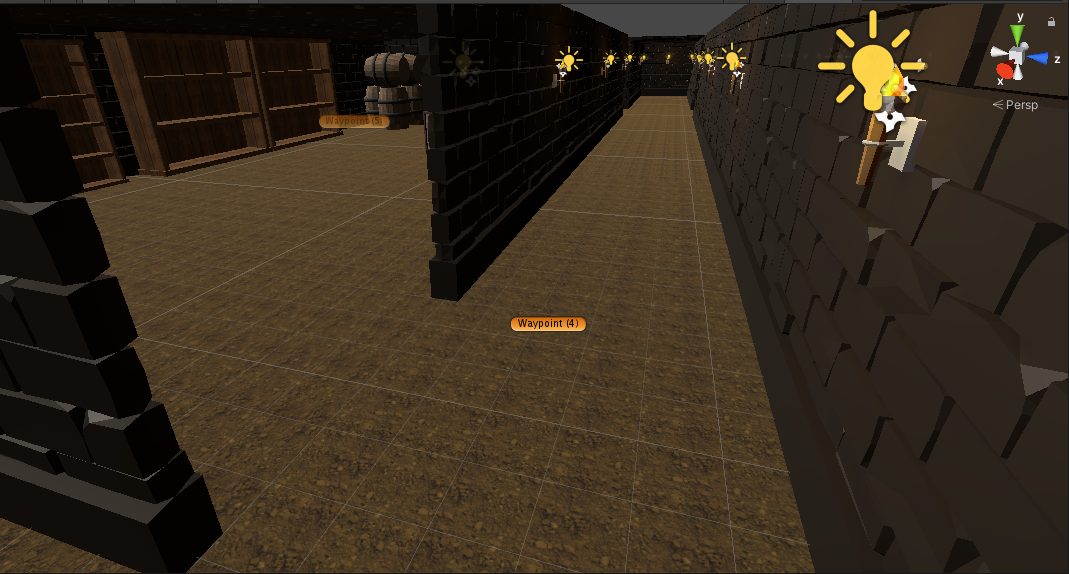


Abb. 7 Wegpunkt

Abb. 6 KI beim Ablaufen der Wegpunkte

Sobald der Spieler sich in der Nähe vom(/n) Gegner(n) befindet, verfolgt die KI den Spieler durch die eingebaute smart-player-detection (SearchPlayer.cs) und greift diesen an, wenn er in Reichweite ist.

Um die Wahrscheinlichkeit zu erhöhen, dass die KI den Spieler findet, wurde das Level in Sektoren unterteilt. Befindet sich der Spieler in einem Sektor, wird dessen Aufenthaltsdauer dort ermittelt und gespeichert. Jeder Sektor besitzt festgelegte Wegpunkte, die von der KI abgelaufen werden, der Sektor mit der höchsten Aufenthaltsdauer des Spielers übergibt diese an die KI.

Folglich ist es das Ziel des Spielers, möglichst den Untoten aus dem Weg zu gehen,

und sollte er entdeckt werden, schnellstmöglich zu entkommen.

**Skriptreferenzen:**

* Pathfinding.cs
* SearchPlayer.cs
* SectorManager.cs
* Sector.cs
* Waypoint.cs

# Kontrollschema

Das Spiel steuert sich mit Maus und Tastatur.

|  |  |
| --- | --- |
| **Button Input** | **Umsetzung im Spiel** |
| W, A, S, D | Spieler bewegt sich in die entsprechende Richtung |
| Maus-Bewegung | Rotiert den Spieler („Umschauen“) |
| L shift – Linke-Umschalt-Taste | Sprinten |
| Linke Maustaste | Item aufsammeln |
| 1,2,3,4,5 | Item an der entsprechenden Stelle im Inventar einsetzen |

**Skriptreferenzen:**

* PlayerController.cs

# Spiel Ästhetik / Design & Benutzeroberfläche

**Ästhetik:**

Das Spieldesign ist an eine düstere Kellerumgebung angelehnt, die spärlich durch Fackeln ausgeleuchtet ist (Abb. 8). Die Fantasy-Atmosphäre entsteht durch das an den Mythos von Knossos erinnernde Labyrinth. Dieses ist sehr verwinkelt und sorgt dadurch dafür, dass der Spieler sich leicht verläuft und in Räumen wiederfindet, die er vorher bereits besucht hatte. Außerdem ist das Labyrinth sowohl von gruseligen, skelettartigen Monstern besetzt, die versuchen den Spieler zu töten, als auch mit verschiedenen Fallen ausgestattet, die den Spieler zusätzlich behindern. Die düstere Atmosphäre unterstreicht animierte Elemente, wie Ratten, tropfende Decken und höhlenartige Geräuschkulissen. Die Assets sind dementsprechend von der Ästhetik her im low-Fantasy-genre, denn es werden keine „unrealistischen“ Assets verwendet abgesehen von dem Untoten. Die verlassenen größeren Kellerräume, in denen der Spieler sich anfangs wiederfindet, lassen Spielraum für Interpretation auf den Zweck, den diese Räume vor langer Zeit erfüllt haben (Abb. 9).



Abb. 8 Fackeln

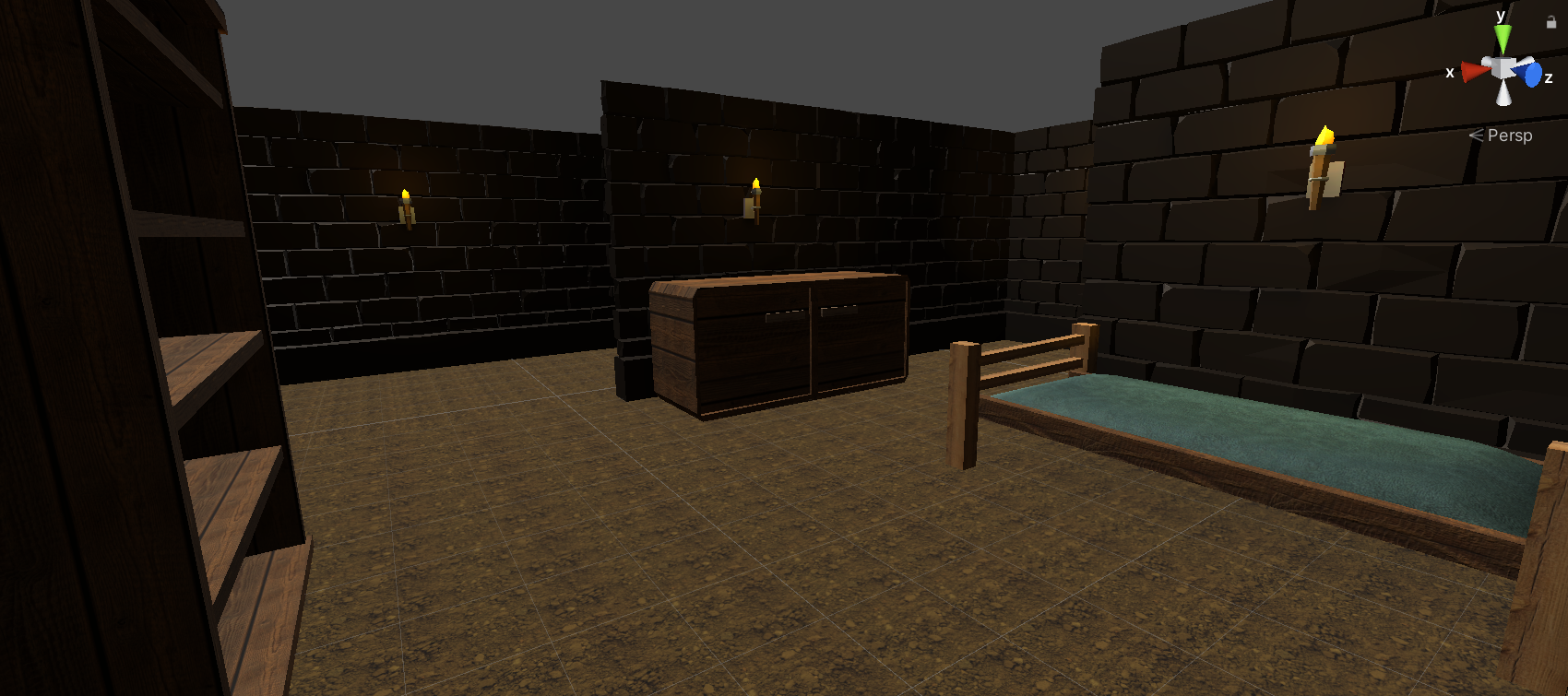


Abb. 9 Einrichtungsgegenstände der Kellerräume

**GUI:**

Die Nutzeroberfläche beschränkt sich größtenteils auf simple Knöpfe. Diese starten per Knopfdruck die gewünschte Aktion. Die graphische Nutzeroberfläche zeigt dem Nutzer die Lebensanzeige, die sich verringert, wenn Schaden genommen wird. Außerdem verdeutlicht ein Herz anhand der Farbe den Gesundheitszustand des Spielers. (Updated) Die Lebensanzeige wird nun im PlayerController gehandhabt.

**Skriptreferenzen:**

* GameController.cs
* GUIDeath.cs
* GUIMainMenu.cs
* GUIOptionMenu.cs
* PlayerController.cs

# Skripte

Die Autoren der Skripte *René Kraus*, *Jan Pagel* und *Franz Mörike* sind in jedem Script festgehalten und der Programmierfortschritt mit Timestamps geloggt. Es werden bislang keine externen Skripte verwendet.

# Arbeitsaufteilung

Auch wenn sich die Arbeitsbereiche teilweise überschneiden, wollen wir Übersicht über unsere Fortschritte behalten.

[LEGENDE]

Fertig

In Arbeit

|  |  |  |
| --- | --- | --- |
| **René** | Jan | Franz |
| Basic movement | Sprites | GDD management |
| Mechanics (z.B. path finding) | Textures | GitHub project management |
| Enemy KI (player proximity detection, detection mechanic) | Level Design (Fallen, exit points) | Health mechanic (heart animation, damage / health system) |
| Enemy model | Musik/Sound | Option menu |
| Animationen | Animationen | Death scene |
| Main menu / GUIs | Level-Gameplay | Button behavior |
| Musik/Sound/Soundmanagement | Level-Finetuning | GUI’s |
| Lightmapping |  | JumpscareSound |
|  |  | Inventory / Items mechanics |

# Quellen

Unity Asset Store:

* Alexander Kotov. Potions, Coin And Box of Pandora Pack <https://assetstore.unity.com/packages/3d/props/potions-coin-and-box-of-pandora-pack-71778>
* REXARD. RPG inventory icons

<https://assetstore.unity.com/packages/2d/gui/icons/rpg-inventory-icons-56687>

* Aleksn09. Rust Key

<https://assetstore.unity.com/packages/3d/props/rust-key-167590>

* Nox\_Sound. Footsteps – Essentials
* <https://assetstore.unity.com/packages/audio/sound-fx/foley/footsteps-essentials-189879>
* The Octoverse. Essential Horror Music Collection (v1) [FREE]

<https://assetstore.unity.com/packages/audio/ambient/fantasy/essential-horror-music-collection-v1-free-187099>

* Jose (Dogzerx) Díaz. Medieval Cartoon Warriors <https://assetstore.unity.com/packages/3d/characters/medieval-cartoon-warriors-90079>
* TS WORK. Fantasy Monster (Wizard) DEMO <https://assetstore.unity.com/packages/3d/characters/creatures/fantasy-monster-wizard-demo-103037>

Audio:

* https://audiojungle.net/category/sound/game-sounds?page=2&term=jumpscare#content (Jump Scare sound)
* ID 175850162 © Fxprosound | Dreamstime.com Cartoon female pain shout

<https://www.dreamstime.com/cartoon-female-pain-shout-sound-usable-game-distance-mid-close-acoustic-processed-audio175850162>

* ID 111117454 © SnowMusicStudio | Dreamstime.com Female Ah

<https://www.dreamstime.com/stock-sound-fx-female-ahh-sound-good-your-project-audio111117454>

* ID 175951763 © Fxprosound | Dreamstime.com Female astonished inhales

<https://www.dreamstime.com/sound-voice-female-astonished-inhales-usable-media-foley-distance-mid-close-acoustic-dry-female-astonished-inhales-audio175951763>

* Mike Koenig
  + Breathing <https://soundbible.com/777-Breathing.html>
  + Demon Zombie Ambiance <https://soundbible.com/1025-Demon-Zombie-Ambiance.html>
  + Perfect Thunder Storm <https://soundbible.com/916-Perfect-Thunder-Storm.html>
  + Suicide Bomber 2 (Rolling Stones) <https://soundbible.com/1914-Suicide-Bomber-2.html>
  + Slow Breathing <https://soundbible.com/766-Slow-Breathing.html>
  + Strange Days <https://soundbible.com/1741-Strange-Days.html>
  + Wind Blowing <https://soundbible.com/634-Wind-Blowing.html>
* HopeinAwe. Dark Laugh

<https://soundbible.com/2001-Dark-Laugh.html>

* Unbekannt. Evil Yelling

<https://soundbible.com/98-Evil-Yelling.html>

* PD Ezwa. Female Sigh

<https://soundbible.com/132-Female-Sigh.html>

* NeoPhyTe. Female Scream Horror

<https://soundbible.com/1627-Female-Scream-Horror.html>

* Hipis. No Mercy

<https://soundbible.com/2019-No-Mercy.html>

* Daniel Simon. Sad Woman Crying

<https://soundbible.com/2133-Sad-Woman-Crying.html>

* Seifert. Gate Model & Chains SFX. Hanser Spiele entwickeln mit Unity 5

# TODO Liste

**Must-have:**

* Story ausarbeiten
* Musik / Hintergrundgeräusche
* Inventarsystem
* Progressive gameplay steps

- Schlüssel + Türen die damit geöffnet werden können

- Karte / Teil einer Karte

**Nice-to-have:**

* ~~Bullets / Schießen / stunning the enemy?~~
* ~~Kriech- / Schleichmechanik~~
* ~~Verschiedene Gegnerarten~~
* ~~Springen~~
* Verstecken hinter Objekten

**Probleme:**

* ~~KI bleibt manchmal einfach stehen~~
* ~~Player kann noch durch Wände glitchen (evtl. Auslöser: Sprinten)~~