

Game Design Dokument



Snack 'n Roll

„Ein physikbasierter Platformer, mit der Extraportion Sammelfieber!“

Entwickelt von: Goran Janosevic

Version: 1.0 – Jänner 2026

Engine: Unreal Engine 5.6.1

Plot

Der Herbst neigt sich dem Ende zu, und ein kalter Hauch kündigt bereits den nahenden Winter an. Für Iggy, den kleinen Igel mit dem großen Appetit, beginnt nun die wichtigste Zeit des Jahres: Bevor er sich voller Vorfreude in seine lange Winterruhe kuscheln kann, bemerkt er ein großes Problem – sein Futtervorrat reicht bei Weitem nicht aus!

Mit neugieriger Nase und wachen Augen macht sich Iggy daher auf den Weg durch raschelndes Laub und feuchte Waldböden. Alles, was er jetzt entdeckt, bringt ihn ein Stück sicherer durch die frostigen Monate.

Charaktere

Iggy

Iggy ist ein kleiner, neugieriger Igel mit einem großen Appetit, doch Iggy ist spät dran: Bevor der Winter endgültig über den Wald hereinbricht, muss er dringend sein Futterreservoir auffüllen.

Mit wachen Augen und flinkem Näschen stapft er durch Laub, Gras und Gestrüpp – immer auf der Suche nach Leckerbissen, die ihm durch die kalte Jahreszeit helfen sollen.

Konzept

Im Zuge seiner Futtersuche muss sich Iggy durch unterschiedliche Landstriche manövrieren, was sich immer wieder als knifflige Herausforderung entpuppt. Manche Passagen gleichen regelrechten Rätseln, die nur mit Geschick, Timing und einem guten Blick für die Umgebung zu meistern sind.

Iggy kann sowohl gemächlich gehen als auch sich zu einer kompakten Kugel zusammenrollen, um schneller voranzukommen – vorausgesetzt, das Gelände lässt es zu. Dabei gilt es, physische Hindernisse wie Baumstämme, Abgründe oder steile Hänge geschickt zu umgehen oder sogar für sich zu nutzen..

Das Spiel lässt sich als Physik-Geschicklichkeit-Platformer bzw. Adventure einordnen, bei dem insbesondere Iggys Roll-Fähigkeit die Möglichkeiten der Unreal-Physikengine voll ausschöpft.

Spielmechaniken

Den Spielenden stehen zwei unterschiedliche Bewegungsformen zur Verfügung, die jeweils eigenes Handling und eigene Vorteile mit sich bringen:

- **Laufmodus**
 - Steuerung in Third-Person-Perspektive
 - Geringere Geschwindigkeit
 - Präzise und kontrollierte Bewegung
 - Animationen:
 - Idle (mit Schnupper-Animation für zusätzliche Immersion)
 - Gehen
 - Hüpfen

- **Rollmodus**
 - Iggy rollt sich zu einer kompakten Kugel zusammen
 - Die Kamera zoomt leicht heraus, um besseren Überblick zu ermöglichen
 - Deutlich höhere Geschwindigkeit von A nach B (sofern physikalisch möglich)
 - Die Steuerung wird anspruchsvoller
 - Das Rollverhalten basiert auf der Physik-Engine
 - Kollisionen können zu einem Crash führen (Geschwindigkeitsverlust)

Entwicklungsumgebung

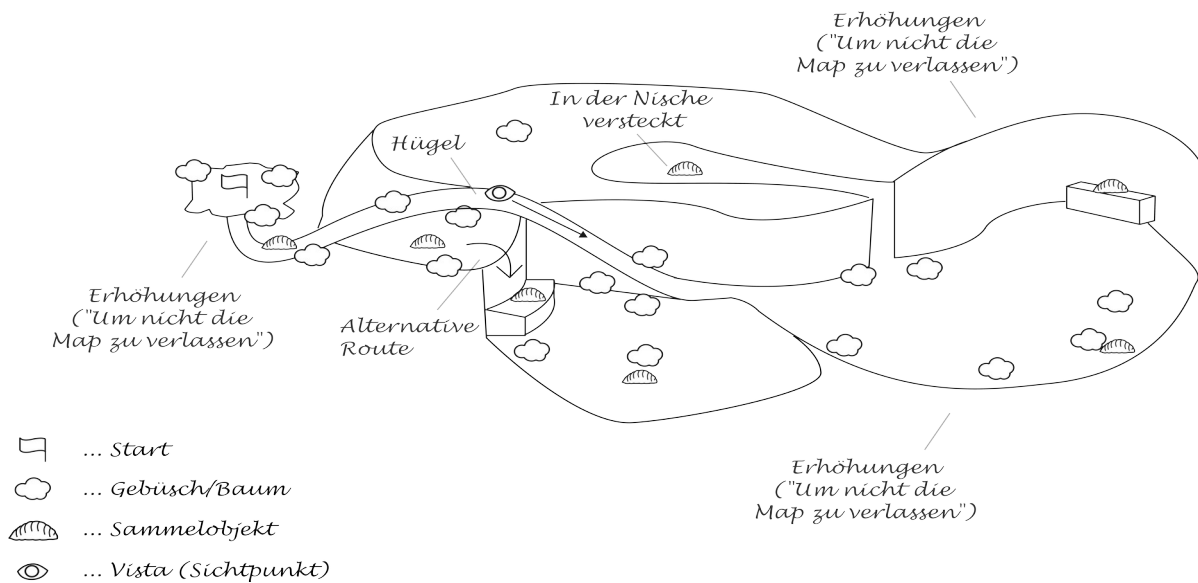
Das Spiel wird mithilfe der Unreal Engine 5.6.1 entwickelt. Für sämtliche Gameplay-Interaktionen innerhalb der Engine kommen primär Blueprints zum Einsatz.

Level Design

Das Leveldesign orientiert sich an einer herbstlichen Waldlandschaft.

Sketches

Erste Konzeptzeichnungen (z. B. für das 1. Level) dienen als Grundlage für Layout, Plattformpositionen, Hindernisse und Sammelobjekte.



Color-Scheme

Das Farbschema setzt auf warme herbstliche Braun- und Gelb- und Rottöne, ergänzt durch dezente grüne Akzente. Diese spiegeln die Stimmung eines spät-herbstlichen Waldes wider, in dem nur noch wenige Pflanzen ihr Grün behalten.

Sound-Design

Die Klangkulisse soll eine natürliche, ruhige Herbststimmung erzeugen:

- Ruhige Hintergrundmusik
- Blätterrascheln beim Laufen durch Laub
- Rollgeräusch bei aktivem Kugelmodus
- Variierende Soundausgabe beim Hüpfen sowie beim Wechsel in den Kugelmodus und zurück
- Sammelgeräusch beim Aufheben animierter Würmer
- Schnuppergeräusch im Idle-Modus zur Steigerung der Immersion

Mood board

Das Mood Board umfasst:

- Warme, herbstliche Farbtöne, überwiegend Braun-, Gelb- und Rottöne
- Dezent grüne Akzente, passend zum späten Herbst
- Inspirationen zu Waldtexturen, Lichtstimmungen, Laubteppichen sowie gegebenenfalls Baumstümpfen, um einen realistischen Waldeindruck zu vermitteln



Look & Feel

Der Stil des Spiels basiert auf einer abstrahierten Low-Poly-Ästhetik:

- Realitätsnahe Assets wie Bäume, Steine und Pilze, reduziert auf eine stilisierte Detailtiefe
- Leicht stilisierte Naturumgebung mit klaren Formen
- Fokus auf eine stimmungsvolle Atmosphäre und sanfte Farbpaletten

Dieser Ansatz unterstützt eine ruhige und atmosphärische Spielerfahrung.

Technische Spezifikationen

Neben der Unreal Engine 5.6.1 und der Nutzung von Blueprints werden folgende Komponenten für die Umsetzung des Spiels als besonders sinnvoll erachtet:

- **CharacterMovementComponent**
 - Zuständig für den Laufmodus
 - Sorgt für präzise und kontrollierte Bewegungen in der Third-Person-Perspektive
- **Chaos Physics**
 - Zuständig für den Rollmodus
 - Verantwortlich für physikbasierte Beschleunigung, Kollisionen und das generelle Rollverhalten
- **UMG Widgets**

Einsatz für UI-Elemente, wie Menüs oder Hinweise

3D-Modellierung mit Blender

- Erstellung eines Low-Poly-Igels im stehenden Modus, inklusive vollständig geriggtem Skelett zur Animation (Idle, Gehen, Hüpfen, Landen)
- Erstellung einer eingerollten Igel-Kugel als separates Modell bzw. Mesh für den Rollmodus
- Manuell erstellte Assets für das User Interface sowie gegebenenfalls für die Soundausgabe

Zeitplanung

Umsetzung des Projekts im Wochen-Rhythmus

- Woche 1 (16.12.2025 – 23.12.2025)
 - Kleinere Recherchetätigkeiten zur technischen und gestalterischen Umsetzung
- Woche 2 (23.12.2025 – 29.12.2025)
 - Modellierung des Igels
 - Stehende Form und Kugelform
 - Animationen (Idle, Gehen, Springen, etc.)
 - Erste Animationstests
 - Wechsel zwischen Lauf- und Rollmodus
 - Umsetzung der Interaktionen mithilfe von Blueprints
 - Level-Aufbereitung
 - Entscheidung über die Nutzung vorhandener Assets versus Eigenproduktion
 - Recherche und Implementierung von Sounds
 - Einbindung und Abspielen von Sounddateien
 - Auswahl passender Sounds für Bewegungen und Sammelobjekte
 - Finalisierung des Levels
 - Hinzufügen von Details (Assets, Farbgestaltung, User Interface, ...)
- Woche 3 (30.12.2025 – 10.01.2026)
 - Polishing und Bugfixing