Slime Slaughter

Prototyping interaktiver Medien-Apps und Games

Sommersemester 2021

Jonathan Rißler 263246

Betreuender Dozent: Prof. Jirka Dell'Oro-Friedl

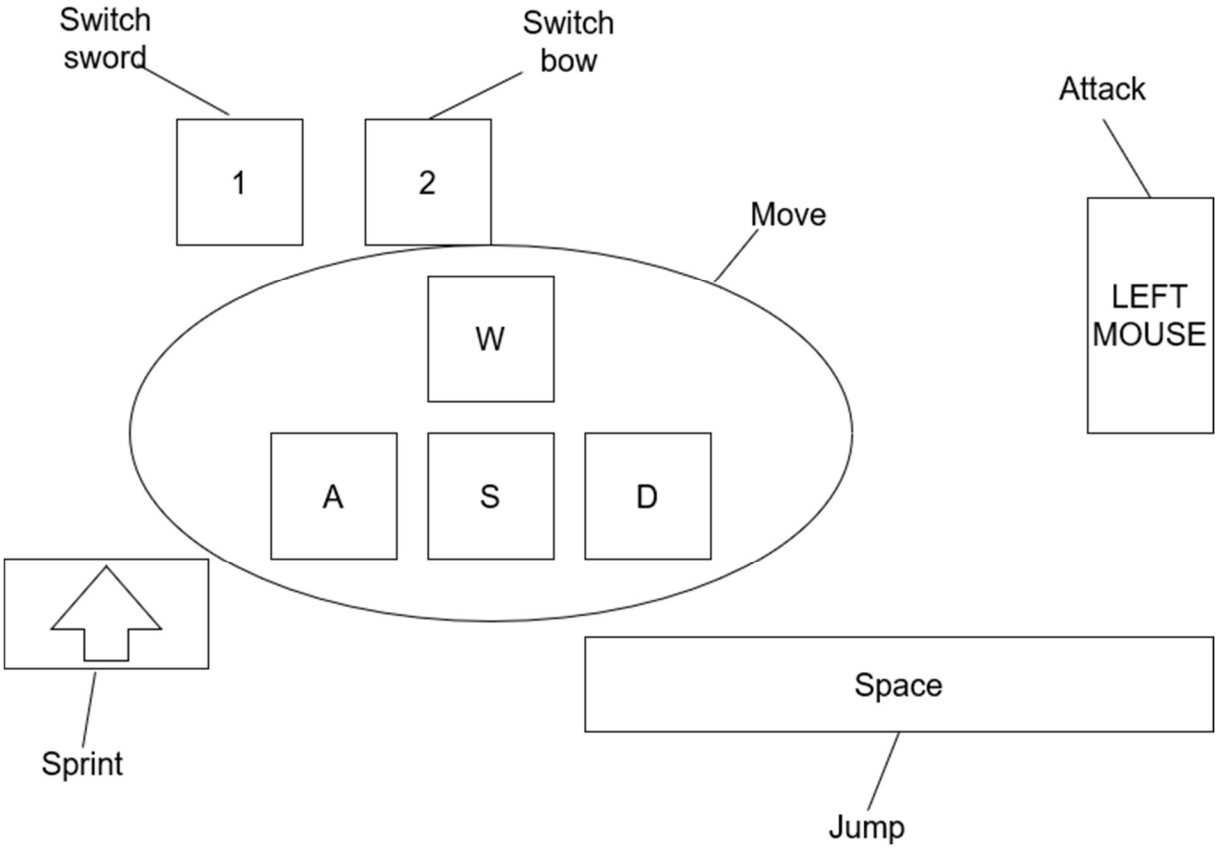
Inhaltsverzeichnis

1. Kurzbeschreibung und Gameplay

Der Avatar kann sich in verschiedenen Levels gegen kleine Gegner sowie einen Boss beweisen. Dabei steigen die Level sowohl in der Schwierigkeit der Gegner als auch im Design des Levels an. Kannst du alle Gegner besiegen, bevor du selbst erledigt wirst?

2. Steuerung und Nutzerinteraktion

Der Avatar kann durch die Benutzung der WASD-Tasten bewegt und gesteuert werden. Zusätzlich hat er die Möglichkeit durch das Halten der linken Shift-Taste sich schneller zu bewegen. Weiterhin kann man zwischen dem Schwert und dem Bogen wechseln, indem man die Taste „1“ für Schwert drückt beziehungsweise die Taste „2“ für den Bogen. Die jeweilige ausgerüstete Waffe kann man dann mit der linken Maustaste verwenden. Zu guter Letzt kann der Avatar mit Hilfe der Leertaste einen Sprung ausführen.



3. Objektinteraktion

Alle Objektinteraktionen finden durch die Kollisionen in der Physik statt:

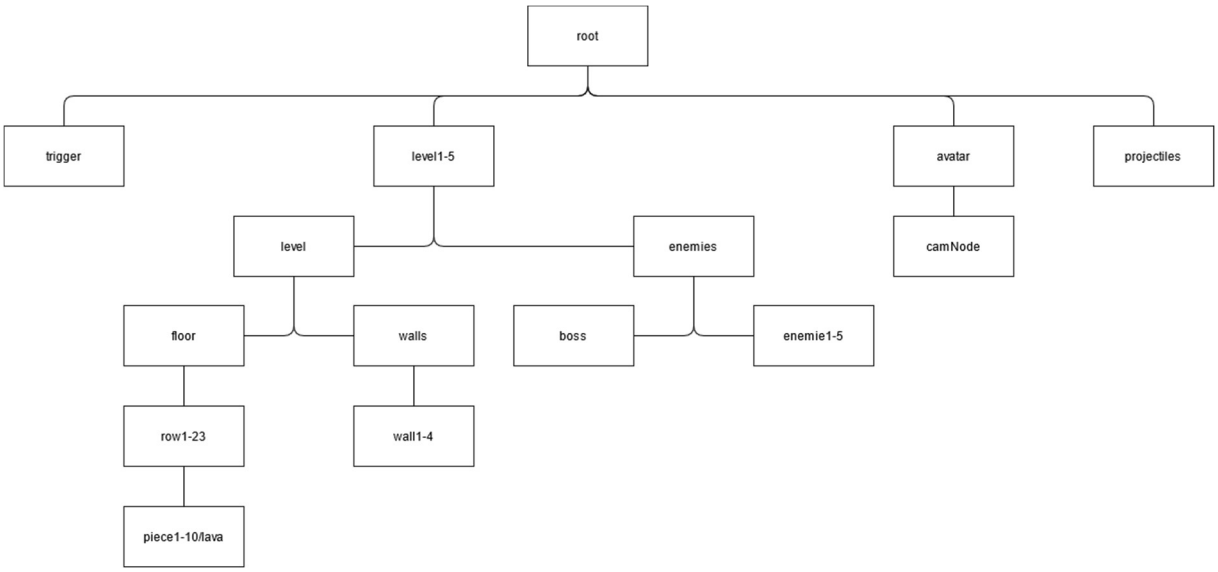
- Alle Projektile werden bei einer Kollision kurzzeitig auf PHYSICS_TYPE.STATIC gesetzt und bleiben an ihrer Stelle stecken beim Auftreffen, eine Sekunde später werden diese mit ihrem RigidBody aus der Szene entfernt

- Treffen die Projektile der Gegner den Avatar oder die Pfeile des Avatars die Gegner beziehungsweise schlägt der Avatar mit seinem Schwert wird eine Schadensberechnung durchgeführt
- Tritt der Avatar auf ein Lava Feld bekommt dieser ebenso Schaden
- Fällt der Avatar vom Level setzt ein Trigger das Spiel auf Verloren
- Beim Schlagen mit dem Schwert sowie beim Prüfen, ob der Avatar wieder springen kann, wird ein Raycast verwendet

4. Objektanzahl variabel

Die Objektanzahl der Projektile ist variabel, da sowohl die Feuerbälle der Gegner als auch die Pfeile des Avatars zur Laufzeit generiert werden.

5. Szenenhierarchie

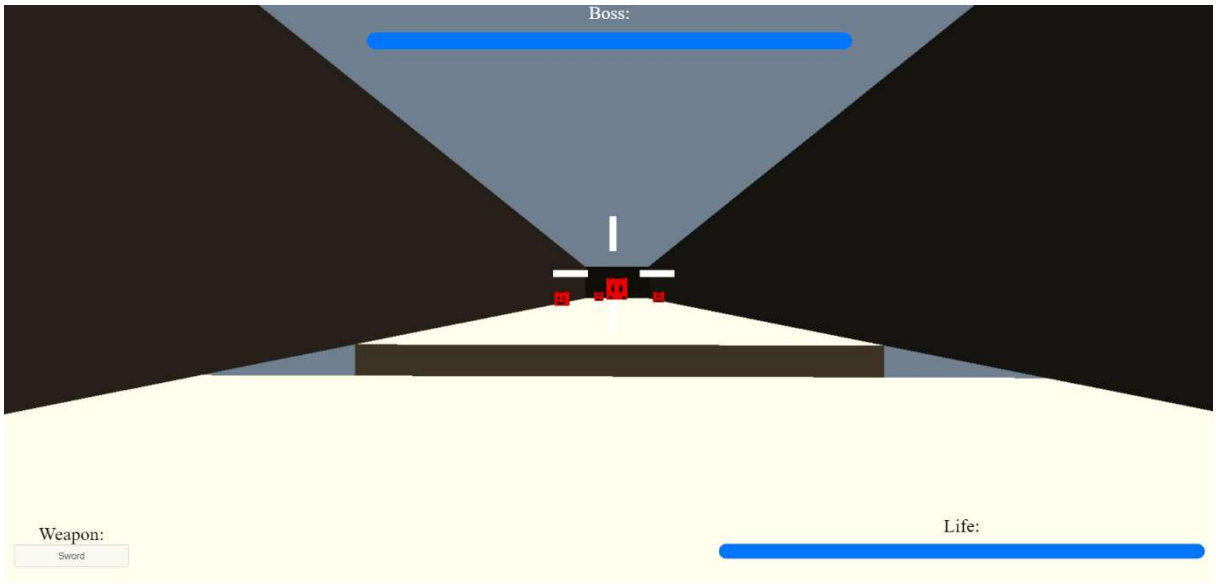


Root ist der Knoten, welcher im Viewport angezeigt wird und ist der oberste Knoten. Dieser wurde genauso wie die fünf Level mit der Belichtung im Fudge-Editor erstellt. In den Level gibt es sowohl einen Knoten, in dem die Gegner sowie der Boss liegen, als auch einen Knoten, in dem der Boden und die Wände liegen. Weiterhin liegt direkt unter dem Root-Knoten ein Trigger, der Avatar, welcher einen Knoten für die Kamera hat. Zudem liegen alle Projectiles, welche während des Spiels erzeugt werden, unter dem Root-Knoten.

6. Sound

Während dem Spiel werden mehrere Sounds verwendet. Außer der Hintergrundmusik wurden diese alle selbst aufgenommen. Während der Dauer eines Levels läuft im Hintergrund für die Atmosphäre eine feine Musik. Weiterhin werden Sounds abgespielt beim Avatar, wenn dieser springt, der Avatar mit dem Bogen schießt oder mit dem Schwert schlägt. Zudem wird jedes Mal ein Sound abgespielt, wenn der Avatar Schaden durch die Gegner oder durch die Lava erhält. Der Boss gibt dauerhaft einen Sound von sich und alle Gegner spielen einen Sound ab, wenn diese Schaden erhalten und wenn diese sterben. Zu guter Letzt wird von der Lava ein leises Blubbern abgespielt.

7. GUI

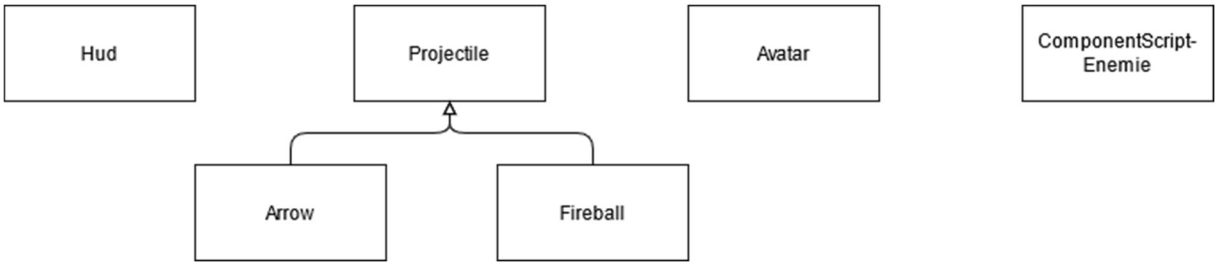


Oben in der Mitte wird die Lebensanzeige des Bosses dargestellt. Weiterhin wird unten rechts die Lebensanzeige des Avatars dargestellt und links unten wird angezeigt, welche Waffe der Avatar momentan ausgerüstet hat. Zu guter Letzt wird in der Mitte des Bildschirms ein Crosshair, welches variiert je nach ausgerüsteter Waffe, angezeigt.

8. Externe Daten

Der Aufbau der verschiedenen Levels ist in einer externen JSON-Datei gespeichert wie der restliche Aufbau der Szene, welche durch den Fudge-Editor kreiert wurde. Zusätzlich lassen sich diverse Parameter für die Gegner und den Boss jedes Levels in einer weiteren JSON-Datei anpassen. Dabei kann eingestellt werden wieviel Leben die normalen Gegner und wieviel Leben der Boss besitzt sowie wieviel Schaden die normalen Gegner und der Boss machen. Alle Werte werden als number gespeichert.

9. Verhaltensklassen und Subklassen



Die Klasse Hud verwaltet die Gamestatevariablen, welche zur Anzeige des GUIs verwendet werden. Die Avatarklasse erstellt den Avatar sowie die Kamera und kümmert sich um sein Movement. Die Projektilklasse ist die Superklasse von der Klasse Arrow und Fireball und initialisiert das Mesh, den Rigidbody, das jeweilige Material sowie setzt die Projektile ein kleines Stück in ihre geschossene Richtung, sodass diese nicht mit dem Objekt, von dem sie losgeschossen werden, kollidieren. In den beiden Subklassen muss also lediglich die Größe der jeweiligen Projektile angegeben werden. Weiterhin wird für die Gegner ein Componentscriptklasse ComponentScriptEnemie erstellt, welche die Gegner als Component

erhalten. In dieser Klasse wird den Gegnern ihre Textur zugewiesen, über einen Timer Feuerbälle abgeschossen sowie das Kollisionshandling der Feuerbälle definiert.

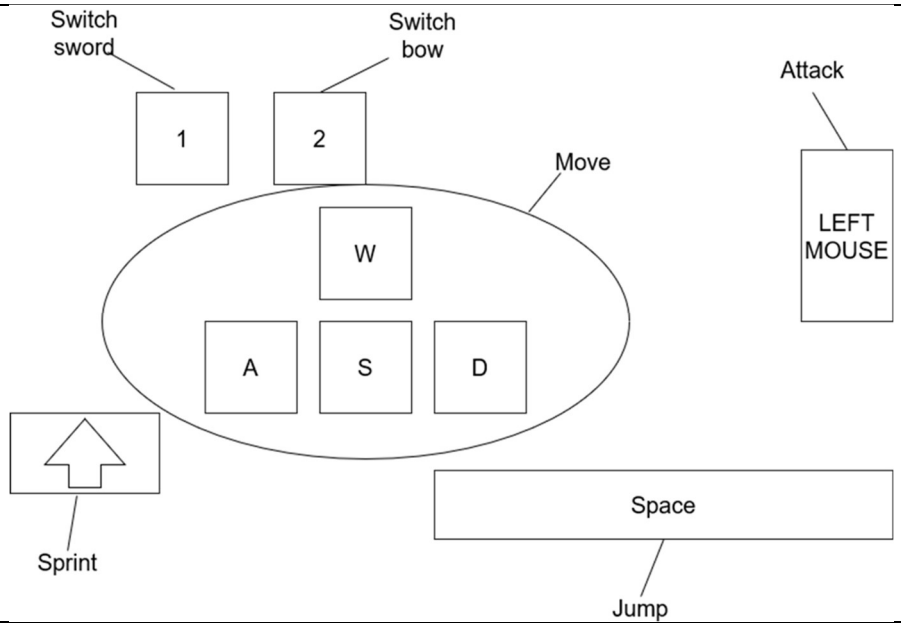
10. Maße und Positionen

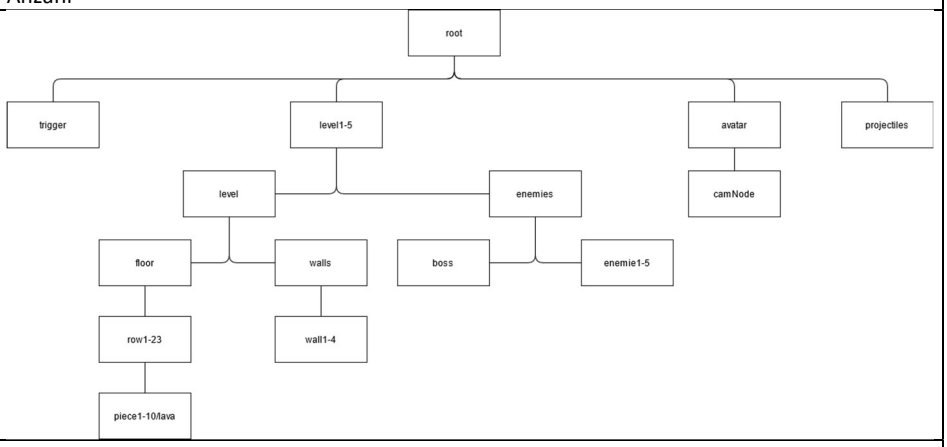
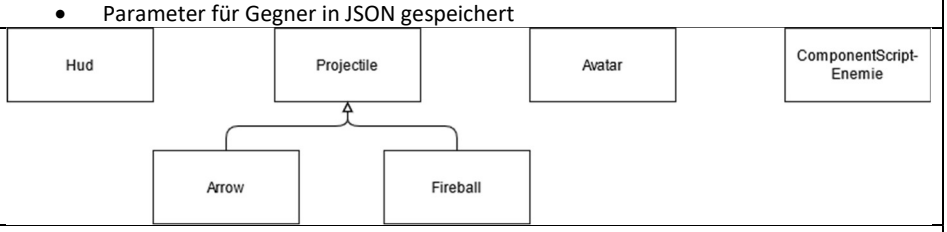
Die Level sind jeweils 66 Einheiten lang und 9 Einheiten breit. Der Avatar sowie die normalen Gegner besitzen die Größe eins in allen Dimensionen. Der Boss ist 3 Einheiten lang, 3 Einheiten breit und hat die Höhe von 4 Einheiten, um bedrohlicher zu wirken. Der Avatar startet beim Starten des Levels im Ursprung. Der ganze Boden ist deshalb um eine halbe Einheit nach unten verschoben. Die Wände sind jeweils 5 Einheiten hoch und umschließen den gesamten Boden. Der Boden besteht aus 23 Reihen, in welcher jeweils 10 Bodenstücke mit der Tiefe 3 Einheiten und den anderen Dimensionen eine Einheit. Dadurch kann beim Erstellen der Levels Löcher in den Boden gesetzt werden.

11. Event-System

Das Eventsystem wird hauptsächlich zur Überprüfung der Kollisionen der Physik verwendet, um die Schadensberechnung des Avatars sowie der Gegner durchzuführen. Weiterhin wird ein Event verwendet, um den Trigger auszulösen, wenn der Avatar von der Plattform fällt. Zudem werden Eventlistener beim Starten des Spiels sowie für das Anklicken der Divs angelegt. Zusätzlich wird die Steuerung des Avatars über Keyboardevents gesteuert.

12. Checkliste

<table>
<tr><th>Nr</th><th>Bezeichnung</th><th>Inhalt</th></tr>
<tr><td></td><td>Titel</td><td></td></tr>
<tr><td></td><td>Name</td><td>Jonathan Rißler</td></tr>
<tr><td></td><td>Matrikelnummer</td><td>263246</td></tr>
<tr><td>1</td><td>Nutzerinteraktion</td><td> 
</td></tr>
<tr><td>2</td><td>Objektinteraktion</td><td>• Kollision Projektile mit Level
• Kollision Projektile mit Gegner/Avatar
• Kollision Avatar Lava Feld
• Trigger beim Herunterfallen vom Level
• Raycast Sprung Avatar sowie Schlag Schwert</td></tr>
</table>

3	Objektanzahl variabel	Feuerbälle sowie Pfeile werden zur Laufzeit erstellt und variieren dadurch ständig in der Anzahl
4	Szenenhierarchie	
5	Sound	• Avatar bekommt Schaden • Hintergrundmusik • Bossgeräusch • Bogenschuss • Gegner bekommt Schaden • Gegner stirbt • Lavablubbern • Schwertgeräusch • Springen
6	GUI	• Boss Lebensanzeige • Avatar Lebensanzeige • Ausgerüstete Waffe • Crosshair
7	Externe Daten	• Root wird aus JSON geladen • Parameter für Gegner in JSON gespeichert
8	Verhaltensklassen	
9	Subklassen	Projectile: • Arrow • Fireball
10	Maße und Positionen	• Avatar & Gegner (1,1,1) • Boss (3,4,3) • Level (66,0.5,9) • Bodenstück (3,1,1) Positionen: • Ursprung: Startpunkt Avatar • Gegnerposition variiert je nach Level
11	Event-System	Physik: • COLLISION_ENTER • TRIGGER_ENTER Weitere: • Load • Click • Mousemove • Pointerlockchange • Keydown • keyup