Landesinstitut für Pädagogik und Medien
SAARLAND

Einführung in Godot

Eine Open Source Entwicklungsumgebung für Videospiele

Saarbrücken, 06.03.2023

Ablauf

- Warum Spieleentwicklung?
- Was ist eine Game Engine / Warum Godot?
- Den Editor kennenlernen
- Einen Prototypen entwickeln
- Ende

Warum Spieleentwicklung

- Lebensweltbezug
- Motivation der Schüler:innen
- Bei Projektarbeit fächerübergreifendes Arbeiten möglich, durch die Vielzahl an Gewerken, die bei der Spieleerstellung beteiligt ist (Game Design, Programmierung, Grafikdesign, Animation, Writing, …)
- Modularisierung von Aufgaben
- Adaptive Zielformulierung

Was ist eine Game Engine / Warum Godot?

- Game Engines (z. B. Unreal, Unity, CryEngine, Godot, GameMaker, Construct, Stencyl, …)
- Game Engines sind Toolsammlungen, die von professionellen Spieleentwickler:innen erstellt werden, um Videospiele möglichst effizient zu entwickeln.
  Visuelle Oberfläche, die per Drag&Drop bedient werden kann
  Vorgefertigte Objektauswahl
  Objektinspektoren mit Einstellungsoptionen
  Leveleditoren
  Skripteditoren
  …

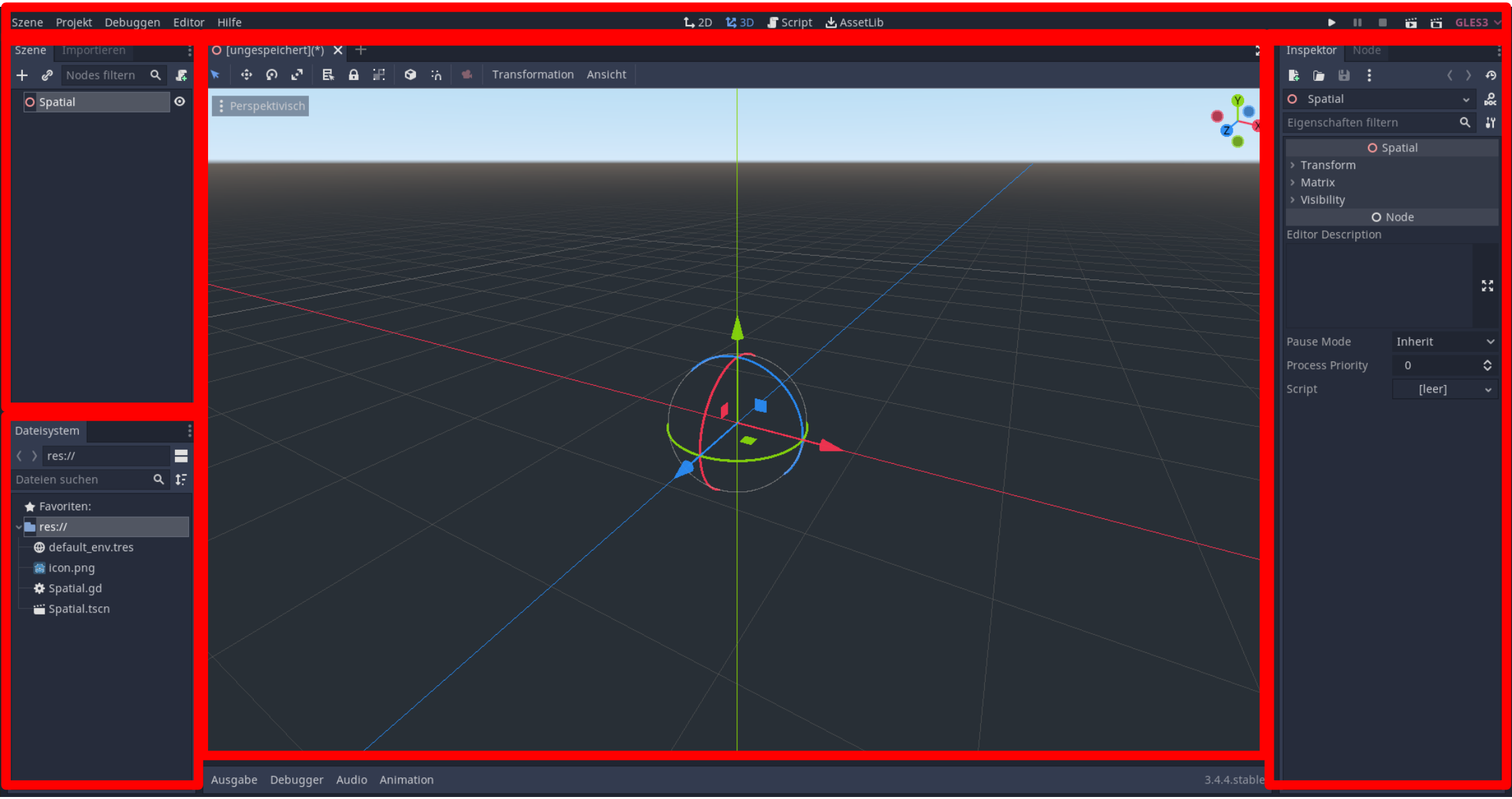
- Warum Godot?
  Open Source
  Echte Trennung von 2D und 3D (Vector2 vs. Vector3)
  Skripteditor in die Engine integriert
  Programmierung mittels GDScript (auf Python basierend)
  „Leichtgewicht“

Ein Satz zur Modularisierung

- Wenn Sie keinen Kunst-/Musikunterricht betreiben und Grafiken + Animationen + Hintergrundmusik + Soundeffekte nicht von den Schüler:innen erstellen lassen wollen, dann finden Sie im Internet eine Reihe von Seiten mit Assets, die unter freier Nutzungslizenz stehen, z. B.:

- Einige Inhalte auf https://itch.io/
- Alle Inhalte auf https://www.kenney.nl/
- Inhalte auf https://opengameart.org/
- Einige Inhalte auf https://craftpix.net/
- (Einige Inhalte auf https://assetstore.unity.com/) - (nur eingeschränkt zu empfehlen)
- Inhalte auf https://freesound.org/
- Etc.

Den Editor kennenlernen



Einen Prototypen entwickeln

Schritt 1 – Das Projekt vorbereiten

- Projekt starten
- Ggf. Editor auf Deutsch einstellen (Editor → Editoreinstellungen → de)
- In der Objekthierarchie „2D-Szene“ anklicken
- Bildausschnitt einrichten (Projekt → Projekteinstellungen → Window → 600 x 1024)
- Grafiken importieren
  Paket herunterladen: https://ansimuz.itch.io/spaceship-shooter-environment
  ZIP entpacken
  In Projektordner verschieben (Rechtsklick auf res:// → „im Dateimanager öffnen“)
- Grafiken verbessern
  Bild anklicken
  Tab „Import“ auswählen
  Haken bei „Filter“ entfernen

Schritt 2 – Hintergrund und Kamera einrichten

- „desert-background.png“ per Drag&Drop aus dem Filesystem in den blauen Rahmen im Hauptfenster ziehen

- Kamera hinzufügen
  Node2D anwählen
  „+“ anklicken
  Camera2D ins Suchfeld eingeben
  Doppelklick auf Camera2D oder anwählen und „Erstellen“ klicken
  Camera2D in der Objekthierarchie anwählen
  Im Objektinspektor „Current“ aktivieren

- Hintergrund ausrichten
  Spriteobjekt „desert-background“ in der Objekthierarchie anwählen
  Im Objektinspektor Transform → Position → x und y = 0

- Szene testen
  Play
  Speichern
  Startszene setzen
  Betrachten
  Schließen

Schritt 3 – Spieler hinzufügen

- Neue Szene erstellen
- In Objekthierarchie KinematicBody2D erstellen
  StaticBody vs. KinematicBody vs. RigidBody
- AnimatedSprite als Child erstellen
  Neues SpriteFrame
  Unten bei Sprite-Einzelbilder ship.png aus Spritesheet einfügen (Waben, 5,2)
  Mittlere Raumschiffe auswählen (oben + unten)
- Szene speichern

- Spieler dem Hintergrund/Spiel hinzufügen (Drag&Drop oder Instanziieren)
  Spieler positionieren und skalieren (über Transform → Position/Scale → X und Y)

- Spieler programmieren
  - Spielerszene auswählen, Skript hinzufügen, Erstellen

```
10 func _ready():
11     pass # Replace with function body.
12 func _physics_process(delta):
13     if (Input.is_action_pressed("ui_left")):
14         if (position.x > -300):
15             move_and_collide(Vector2(-5,0))
16     if (Input.is_action_pressed("ui_right")):
17         if (position.x < 300):
18             move_and_collide(Vector2(5,0))
```

Schritt 4 – Schüsse hinzufügen

- Neue Szene
  KinematicBody2D
  AnimatedSprite → Effekte → laser-bolts.png → Checkbox „Playing“
  CollisionShape2D → Rectangle → Anpassen
  Szene speichern

- Schuss programmieren
  - Skript anhängen/erstellen

  - Player-Skript aktualisieren
    (alternativ auch
    projektil.position = $Position2D.position
    möglich)

```
11 func _physics_process(delta):
12     move_and_collide(Vector2(0, -10))
```

```
29     if Input.is_action_just_pressed("schuss"):
30         var projektil = load("res://Schuss.tscn").instance()
31         get_parent().add_child(projektil)
32         projektil.position = position
33         projektil.position.y -= 50
```

Schritt 5 – Gegner hinzufügen

- Neue Szene
  KinematicBody2D
  AnimatedSprite → Raumschiffe → enemy-big → Checkbox „Playing“
  CollisionShape2D → Rectangle → Anpassen
  Szene speichern

- Timer in Hauptszene einfügen (2sec, One Shot aus, Autostart ein)
  Signal timeout mit Skript der Hauptszene verbinden

```
5   var rng = RandomNumberGenerator.new()
```

```
19  func _on_Timer_timeout():
20      var gegner = load("res://GegnerGerade.tscn").instance()
21      rng.randomize()
22      gegner.position = Vector2(rng.randf_range(-250.0, 250.0), -600)
23      add_child(gegner)
```

Schritt 6 – Schuss mit Gegner interagieren lassen

- Im Skript der Szene Schuss

```
11 func _physics_process(delta):
12     var beruehrt = move_and_collide(Vector2(0, -10))
13     if(beruehrt != null):
14         beruehrt.collider.queue_free()
15         queue_free()
16
```

Weitere Optionen

- Explosion bei Treffer
- Schüsse und Gegner beim Verlassen des Spielfeldes löschen
- Gegner zerstören eigenes Schiff bei Kollision
- Punktezähler
- Ablaufende Spielzeit
- Gegner schießen zurück
- Gegner bewegen sich ebenfalls seitwärts
- Abschlussbildschirm (Gewonnen / Verloren)
- …

Ende

Zeit für Fragen / Anregungen / Kritik

Zeit für Wünsche (z. B. weitere Fortbildungen zu Godot, zu anderen Game Engines, zur Spieleentwicklung allgemein, zu Games im Unterricht allgemein, zu anderen Informatik-Themen, zum Aufbau eines Netzwerks „Spieleentwicklung im Unterricht“, …)

Vielen Dank für Ihre Aufmerksamkeit

Viel Spaß mit Godot