Anforderungsdokument – Crazy Machines (erweiterte Version)

1 Zielsetzung

Ziel dieses Projekts ist die Entwicklung eines minimalistischen Physikpuzzlespiels, inspiriert von *Crazy Machines*. Das Spiel soll eine erweiterbare Architektur bieten, um langfristig KI-gesteuerte Interaktionen zu ermöglichen. Ein besonderer Fokus liegt auf der physikalisch korrekten Simulation, einfacher Bedienbarkeit und der Möglichkeit zur Erstellung und Bearbeitung eigener Level.

2 Funktionsumfang

- **Startscreen mit Navigationsmöglichkeiten:**
  - Spiel starten (Levelauswahl)
  - Settings
  - Levelbuilder (eigene Level erstellen)
- **Levelauswahl:**
  - Drei vorkonfigurierte Level: leicht, mittel, schwer
  - Laden eines benutzerdefinierten Levels (JSON-Datei)
- **Editor-/Spielansicht:**
  - 2D-Ansicht des Spielfelds
  - Drag-and-Drop-Platzierung von Objekten
  - Buttons: Start, Pause, Reset Simulation, Reset platzierte Objekte

3 Settings

- Velocity Iterations
- Position Iterations
- Weitere interne Simulationsparameter (z. B. FPS, Dämpfung – optional)

4 Spielprinzip

Der Spieler platziert Objekte im Editor so, dass durch Start der Simulation ein Ball in die Endzone gelangt. Die Platzierung erfolgt vor Start der Simulation. Bei Erfolg gilt das Level als gelöst.

5 Levelstruktur

- Format: Human-readable JSON-Dateien
- Speicherung von Objektpositionen, Eigenschaften, Startpunkt, Zielbereich, Trigger etc.
- Levelarten:
  - Vorkonfigurierte Levels
  - Eigene Levels (via Levelbuilder)
  - Extern ladbare JSON-Levels

6 Levelbuilder

Der Levelbuilder ermöglicht es Benutzer:innen, vollständig eigene Level zu entwerfen. Dabei steht ein leeres Spielfeld zur Verfügung, auf dem Objekte beliebig platziert werden können. Der Benutzer kann folgende Elemente definieren:

- Platzierung von Objekten aus dem Bauteile-Katalog (Ball, Box, Ballon etc.)
- Definition von Startpunkt des Balls
- Festlegung der Endzone (Ziel)
- Name des Levels
- Die Gravitation des Levels
- Optional; Zeitbegrenzung in der das Level abgeschlossen werden muss
- Optional: Platzierung von Triggern, Ketten und anderen interaktiven Elementen

Die erstellten Level können als JSON-Dateien gespeichert werden. Diese sind im gleichen Format wie die importierbaren Level und somit auch später wieder über die Levelauswahl spielbar. Ziel ist es, sowohl nutzerdefinierte als auch KI-generierte Level flexibel zu unterstützen.

7 Einschränkungen

- Keine oder nur optionale Soundeffekte
- Keine Benutzerkonten oder Online-Features
- Keine mobile Unterstützung
- Fokus auf Funktionalität und korrekte Physik, nicht auf Optik

8 Zielgruppe

- Entwickler:innen, KI-Forscher:innen und Designer:innen
- Proof-of-Concept zur Simulation und KI-Anbindung

9 Erfolgskriterien

- Der Ball erreicht die Endzone durch korrekt platzierte Objekte
- Physiksimulation funktioniert stabil, realistisch und nachvollziehbar
- Benutzeroberfläche ist intuitiv und effizient bedienbar

10 Bauteile (aktuell geplant)

Objekt	**Eigenschaften**
Ball	Dynamisch, reagiert auf Gravitation
Box	Statisch oder dynamisch platzierbar
Ballon	Optional aufsteigend, dynamisch
Domino	Aneinanderreihung von Boxen, dynamisch
Kette (optional)	Verbindet Objekte, interaktive Bewegung möglich
Trigger-Elemente (Optional)	z. B. Schalter, starten Aktionen bei Berührung
Endzone	Markiert den Abschluss eines Levels

11 Technologien

Bereich	**Anforderung**
Programmiersprache	Java 11
GUI-Framework	JavaFX
Physik-Engine	JBox2D
Build-System	Maven