

# PoCs

## Spieler-Character

- Movement
- Sprite
  - Animation?
- Nähe zu Interaktionsobjekte

## Raum

- Gegenstände
- Kollision
- Sprites

## Interaktionsobjekte (Aufgaben/Aktivität)

## Parameter

- Manipulierbar
- Zugriff
- Anzeige
- Interaktion Untereinander

## Zeit

## PoC 1: Player Movement

Es gilt zu beweisen, dass der Spieler Character mittels WASD gesteuert werden kann.

Exit Kriterium: X oder Y Koordinaten der Player-Node werden verändert und der Character bewegt sich auf dem Bild.

Fail Kriterium: Keine Bewegung des Spieler Characters ist erkennbar.

1. Fallback: Der Spieler erhält alternative Steuerungsmöglichkeiten, wie z.B. die Pfeiltasten oder Controller.

2. Fallback: Die Interaktion kann mittels der Mausgetätigt werden.

## PoC 2: Character Sprite laden

Es gilt zu beweisen, dass der korrekte Sprite für den Spieler Charakter verwendet wird.

Exit Kriterium: Der korrekte Sprite ist auf dem Bildschirm sichtbar.

Fail Kriterium: Auf dem Bildschirm ist kein Sprite oder ein nicht vorgesehener Sprite sichtbar.

1. Fallback: Im Ladeprozess auf korrektes Laden der Dateien warten.

2. Fallback: Standard Sprite verwenden.

## PoC 3: Sprite Animation laden

Es gilt zu beweisen, dass für den Spieler Charakter bei Bewegungen die korrekte Sprite Animationen geladen werden.

Exit Kriterium: Die korrekte Sprite Animation ist bei den jeweiligen Bewegungsinteraktionen auf den Bildschirm sichtbar.

Fail Kriterium: Auf dem Bildschirm ist keine Animation oder eine nicht vorhergesehene Animation sichtbar.

1. Fallback: Im Ladeprozess auf korrektes Laden der Dateien warten.

2. Fallback: Es werden keine konkreten Animationen benutzt sondern nur zu passenden Sprites gewechselt.

## PoC 4: Interaktionsdistanz

Es gilt zu beweisen, dass Interaktionsobjekte nur innerhalb der Distanz  $n$  zum Spieler Interaktion erlauben.

Exit Kriterium: Der Spieler kann nur innerhalb der vorgegebenen Distanz mit dem Objekt interagieren.

Fail Kriterium: Der Spieler kann auch außerhalb der vorgegebenen Distanz oder sogar gar nicht mit dem Interaktionsobjekt interagieren.

Fallback 1: Interaktion mit dem Objekt auch mittels der Maus zulassen.

Fallback 2: Interaktion nicht von Distanz, sondern von Kollision oder Position abhängig machen.

## PoC 5: Raum Sprite laden

Es gilt zu beweisen, dass der korrekte Sprite für den Raum verwendet wird.

Exit Kriterium: Der korrekte Sprite ist auf dem Bildschirm sichtbar.

Fail Kriterium: Auf dem Bildschirm ist kein Sprite oder ein nicht vorgesehener Sprite sichtbar.

1. Fallback: Im Ladeprozess auf korrektes Laden der Dateien warten.

2. Fallback: Es wird nur eine Leere Fläche für den Raum verwendet

## PoC 6: Raum Kollision

Es gilt zu beweisen, dass der Spieler Character mit der Wand des Raumes kollidiert und diesen nicht verlassen kann.

Exit Kriterium: Der Spieler Character kann den Raum nicht durch das Durchgehen von Wänden verlassen.

Fail Kriterium: Der Spieler Character kann durch Wände gehen und es findet keine Kollision mit der Wand statt.

1. Fallback: Die X und Y Position des Character kann nur innerhalb der X und Y Werte der Größe des Raumes verwendet werden.

2. Fallback: Spieler Bewegung nur mittels der Maus erlauben und die Raumgröße auf das gesamte Fenster setzen.

## PoC 7: Parameter Manipulation

Es gilt zu beweisen, dass die benutzten Parameter auf intendierte Weise manipulierbar und veränderbar sind.

Exit Kriterium: Die Parameter sind manipulierbar und es ist möglich auf sie zu addieren, subtrahieren oder sie zu überschreiben, wie intendiert.

Fail Kriterium: Die Parameter lassen sich nicht verändern oder verändern sich nicht auf gewünschte Weise.

1. Fallback: Die Interaktionsobjekte verfügen über eigenen Parameter, welche eine Berechnung der Parameter erlauben.

## PoC 8: Parameter Zugriff

Es gilt zu beweisen, dass die Parameter von sich selbst und von anderen Funktionen zugreifbar und veränderbar sind.

Exit Kriterium: Auf die Parameter kann von innerhalb der Funktion und von anderen Funktionen zugegriffen und verändert werden.

Fail Kriterium: Parameter lassen sich nicht verändern und sind nicht oder nur bedingt zugreifbar.

1. Fallback: Standard-Verhalten von Objekten erzeugen.

2. Fallback: Extra-Parameter in Objekten, welche eine Berechnung des Spielerverhaltens erlauben.

## PoC 9: Parameter UI

Es gilt zu beweisen, dass die Darstellung der Parameter mittels des UIs korrekt ist.

Exit Kriterium: Die vom UI dargestellten Parameter stimmen mit den internen Werten überein.

Fail Kriterium: Die vom UI dargestellten Parameter stimmen nicht mit den internen Werten überein.

1. Fallback: Die konkreten Zahlen werden zusätzlich dargestellt, ohne eine grafische Darstellung.

2. Fallback: Zusätzlich an Objekten im Raum das UI darstellen, welche das Lesen der Parameter erlauben.

## PoC 10: UI Lesbarkeit

Es gilt zu beweisen, dass der Spieler das UI lesen und korrekt interpretieren kann.

Exit Kriterium: Der Spieler interpretiert das UI korrekt und kann die dargestellten Werte korrekt ablesen.

Fail Kriterium: Der Spieler kann das UI nicht korrekt interpretieren und die Werte nicht korrekt ablesen.

1. Fallback: Alternative Darstellungsformen der Parameter in den Einstellungen anbieten.
2. Fallback: Abhängig von den Parametern auch den Sprite des Spieler Characters verändern, um den Wert der Parameter über den Spieler Character zu vermitteln.

## PoC 11: Skalierbare Fenstergröße

Es gilt zu beweisen, dass das Fenster auf unterschiedlichen Bildschirmgrößen korrekt skaliert und die Grafiken sich nicht verzerren.

Exit Kriterium: Das Fenster lässt sich beliebig skalieren, ohne dass sich die Grafiken verzerren.

Fail Kriterium: Das Fenster lässt sich nicht skalieren oder die Grafiken werden nicht korrekt angezeigt.

1. Fallback: Es wird nur eine feste Fenstergröße implementiert.
2. Fallback: Der Inhalt des Fensters behält immer dieselbe Größe unabhängig der Fenstergröße.