Pflichtenheft

Projekt „Roomie Boobie“
Version 1.0
31.05.2019
Trinh, Peters, Vu, Laibold

Inhalt

Zielsetzung

Ziel ist ein 2D-Kinderspiel, bei dem die Spielenden vorgegebene Einrichtungsgegenstände unter bestimmten Regeln in einem Raum platzieren müssen. Zur Auswahl stehen die Spielmodi “Level-Modus” und “Kreativ-Modus”. Beim Level-Modus geht es darum, vorgegebene Räume mit einer bestimmten Anzahl Gegenständen zu füllen. Dabei wird pro Gegenstand eine Sekundenanzahl vorgegeben. Diese muss eingehalten werden, um nicht zu verlieren. Außerdem müssen bei der Platzierung gewisse Regeln befolgt werden. Zum Beispiel darf kein Teppich auf einem Schrank oder kein Bett vor die Tür gestellt werden. Wird eine dieser Regeln gebrochen, geht das Spiel zwar weiter, der Gegenstand muss allerdings neu platziert werden. Sind alle Objekte an einem validen Platz, ist das Level bestanden und das nächste wird freigeschaltet. Pro Level gibt es einen Highscore, in dem der Name des Users und Punkte, die unter Anderem aus der benötigten Zeit berechnet werden, enthalten.
Alternativ kann man im Kreativ-Modus ein eigenes Zimmer mit Wänden, Türen und Fenstern erstellen. Dabei wird auch ausgewählt, welche Gegenstände in den Raum passen müssen. Diese selbst erstellten Level werden persistent gespeichert und können separat von allen Usern gespielt werden. Auch hier gibt es Highscores, außerdem können die Räume auch nachträglich noch bearbeitet werden.

Anwendungsszenarien

Szenario 1: Erstes Level-Spiel

Ein User startet das Spiel zum ersten Mal. Er gibt seinen Namen ein und startet den Level-Modus. Er liest sich kurz die Regeln durch spielt ein kurzes Probe-Level, in dem er ohne Zeitdruck die einzelnen Funktionen testen kann.
Im ersten bewerteten Level hat der User nun 5 Sekunden für die Platzierung jedes Gegenstandes. Dies meistert er ohne Fehler, da er die Gegenstände clever dreht. Nach dem Level erhält der Benutzer seine aktuelle Punktzahl und den Highscore dieses Levels und und kann das nächste Level weiterspielen, um seine Bewertung zu verbessern.
Beim nächsten Level muss der Spielende mehr Objekte in einem komplizierteren Raum platzieren. Während der Spielzeit macht er einen Fehler (Schrank vor die Tür gestellt) und wird mit einer Fehlermeldung darauf hingewiesen. Er schafft es nicht, alle Gegenstände in der vorgegebenen Zeit korrekt zu platzieren und hat damit verloren. Er bekommt seine Punktzahl auf dem Highscore angezeigt und landet wieder im Menü.

Szenario 2: Motivation durch Highscore

Der User möchte sehen, wie sein Geschwisterkind im Spiel abgeschnitten hat. Dazu wählt er im Menü den Punkt “Highscore” aus. Dort sieht er, wer es bis zu welchem Level geschafft hat und wie viele Punkte dabei erreicht wurden. Er sieht auch seinen eigenen Namen mit dem erreichten Level von letzter Woche und möchte in der Rangliste weiter oben stehen. Dazu startet er den Level-Modus und vergleicht danach sein Ergebnis, das nach dem Spiel angezeigt wird.

Szenario 3: Im Kreativ-Modus erstellen und spielen

Der User möchte sein eigenes Level gestalten und startet den Kreativ-Modus. Dieser Modus erlaubt es ihm sein eigenes Level mit allen möglichen Elementen zu erstellen. Dabei muss er zuerst einen Grundriss aus Wänden erstellen. Diese füllt er dann mit einem Fenster und einer Tür, was Mindestvorgabe ist. Danach bestimmt er die dazugehörigen Objekte, die beim Spielen des Levels platziert werden müssen. Zum Schluss speichert der Spielende seine Kreation unter einem ausgewählten Namen ab.
Danach will der User eines seiner eigenen Level ausprobieren. Er wählt es zwischen den erstellten Kreationen von anderen Spielern aus. Daraufhin wird die ausgewählte Spielatmosphäre geladen und der User kann mit dem Spielen beginnen. Nach dem Spielen eines selbst erstellten Level wird ein Highscore für dieses Level angezeigt, bei dem er sehen kann, wer das Level am schnellsten lösen konnte.
Der User möchte dieses Level nun noch einmal bearbeiten und lädt es im Bearbeiten-Modus aus. Dann lässt er seiner Kreativität erneut freien Lauf und kann alles an dem Level verändern, vom Namen bis hin zu den Auswahl der Objekten, dem Grundriss, die Platzierung der Fenster und Türen. Zum Schluss muss dies nur den vorgegebenen Regeln entsprechend gerecht sein und wenn der User fertig mit der Bearbeitung ist, bestätigt er die Veränderung und überschreibt die alte Datei.

Use Cases

Eigenes Level erstellen

Akteure: User, System
Fachlicher Auslöser: User möchte ein eigenes Level erstellen
Vorbedingungen: Anwendung muss gestartet sein und User muss bei der Auswahl auf erstellen gedrückt haben
Standardablauf:

1. User: legt Wände fest
2. User: platziert Fenster und Türen
3. System: überprüft ob der Raum geschlossen ist
4. System: überprüft, ob die Platzierung der Türen und Fenster regelgerecht ist
5. User: wählt die Objekte aus, die das Level bereitstellen soll
6. System: überprüft ob die Auswahl der Objekte regelgerecht ist
7. User: gibt dem erstellten Level einen Namen und drückt auf speichern
8. System: speichert das Level und all dessen Attribute als JSON Datei ab

Alternative Abläufe/ Fehlersituationen/Sonderfälle:

3a. System merkt, dass der Raum nicht geschlossen ist
3aa. User muss nochmal die Wände festlegen - weiter mit 3
4a. System lehnt die Platzierung ab
4aa1. User muss die Platzierung anpassen
4ab1. User bearbeitet noch mal die Größe und Anzahl der Räume (Punkt 4)
6a. System lehnt die Auswahl der Objekte ab
5a1. User muss die Auswahl anpassen (Punkt 5)

Nachbedingung/Ergebnis:

Level ist spielbar

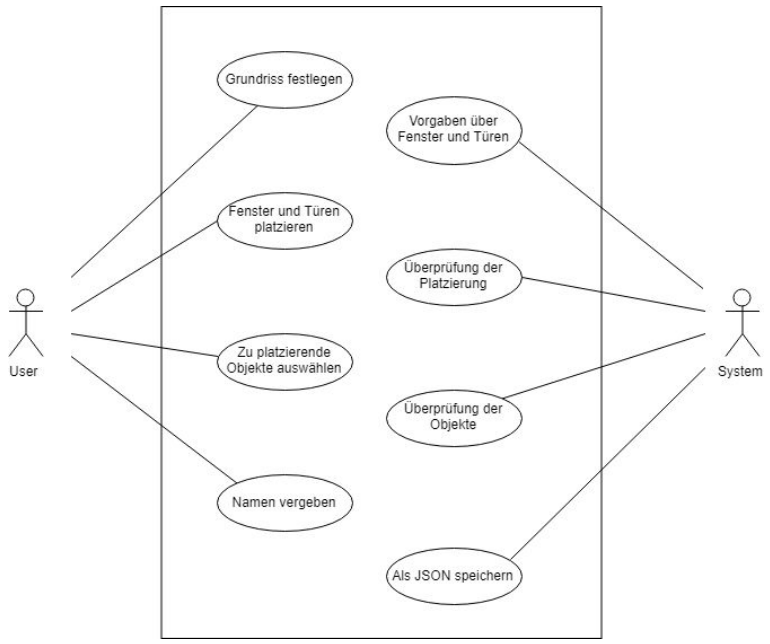
Nicht-funktionale Anforderungen:

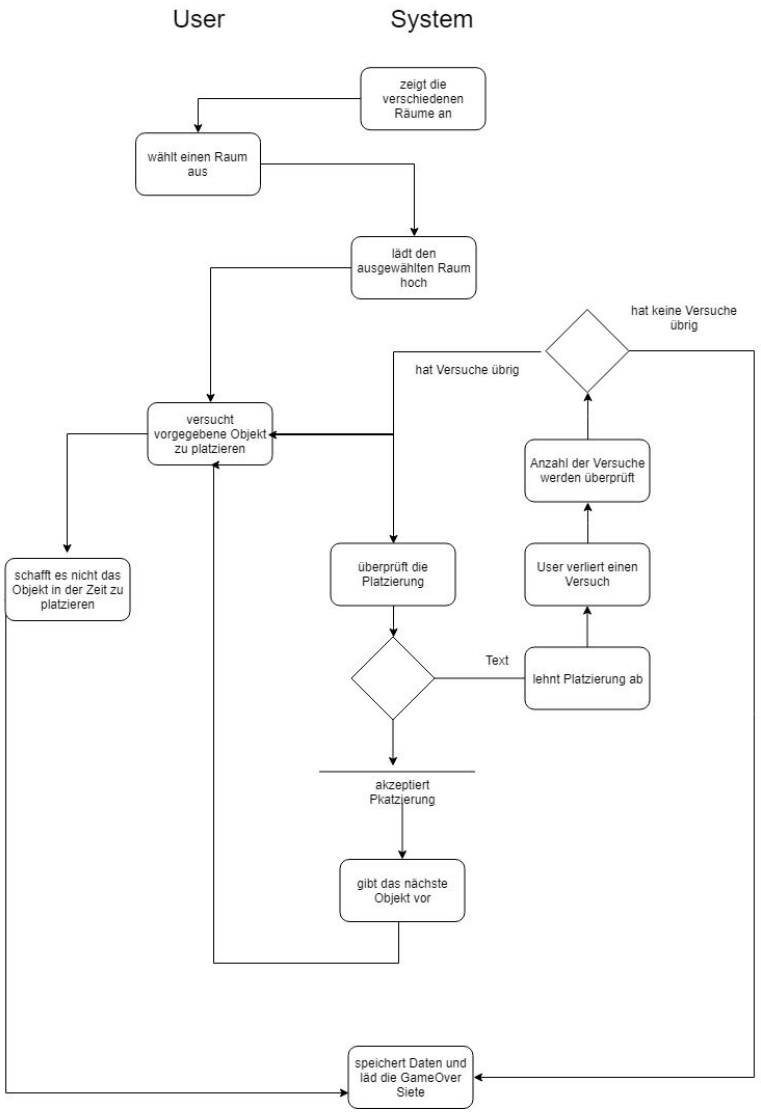
Verständliche Rückmeldung bei Fehlern,
Programm ist tolerant gegenüber falschen Platzierungen
Nur korrekte Level werden gespeichert

Parametrisierbarkeit/Flexibilität:

Wann immer ein User ein eigenes Level erstellen mag

Nutzungshäufigkeit/Mengengerüst: undefiniert


Grundriss festlegen
Vorgaben über Fenster und Türen
Fenster und Türen platzieren
Überprüfung der Platzierung
Zu platzierende Objekte auswählen
Überprüfung der Objekte
Namen vergeben
Als JSON speichern
User
System


User
System
zeigt die verschiedenen Räume an
wählt einen Raum aus
lädt den ausgewählten Raum hoch
hat keine Versuche übrig
hat Versuche übrig
versucht vorgegebene Objekt zu platzieren
Anzahl der Versuche werden überprüft
schafft es nicht das Objekt in der Zeit zu platzieren
überprüft die Platzierung
User verliert einen Versuch
Text
lehnt Platzierung ab
akzeptiert Pkatzierung
gibt das nächste Objekt vor
speichert Daten und läd die GameOver Siete

Eigenes Level bearbeiten

Akteure: User, System

Fachlicher Auslöser: User möchte ein bereits existierendes eigenes Level bearbeiten

Vorbedingungen: Anwendung muss gestartet sein und User muss bei der Auswahl auf bearbeiten gedrückt haben

Standardablauf:

1. User: wählt von der Auswahl ein existierendes Level aus

2. System: lädt das ausgewählte Level hoch und zeigt den Raum im Raum Editor

3aa. User: bearbeitet die Wände

3aa. User: bearbeitet die Auswahl der Objekte

3ab. System: überprüft ob die Auswahl regelgerecht ist

3ba. User: bearbeitet die PLatzierung der Türen und Fenster

3bb. System: überprüft ob die Auswahl der Objekte regelgerecht ist

3c. User: gibt dem erstellten Level einen Namen und drückt auf speichern

4. System: überschreibt das Level mit den neuen Attributen

Alternative Abläufe/ Fehlersituationen/Sonderfälle:

3ab. System lehnt die Auswahl der Objekte ab

3aba. User muss die Auswahl anpassen - weiter mit 3ab

3bb. System lehnt die Platzierung ab

3bba. User muss die Platzierung anpassen

3bbb. User bearbeitet nochmal die Größe und Anzahl der Räume
weiter mit 3aba1

Nachbedingung/Ergebnis: -

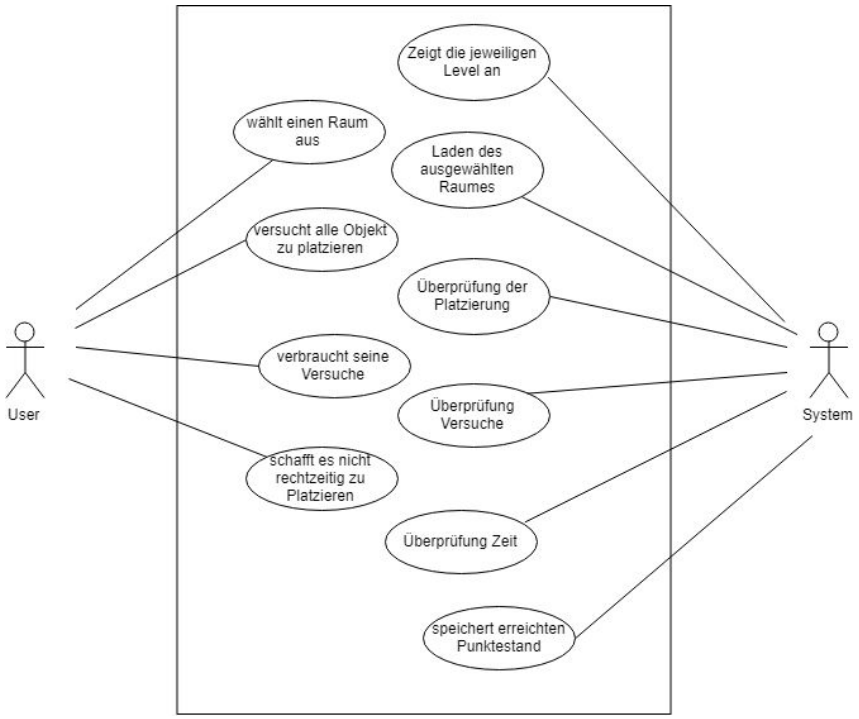
Nicht-funktionale Anforderungen:

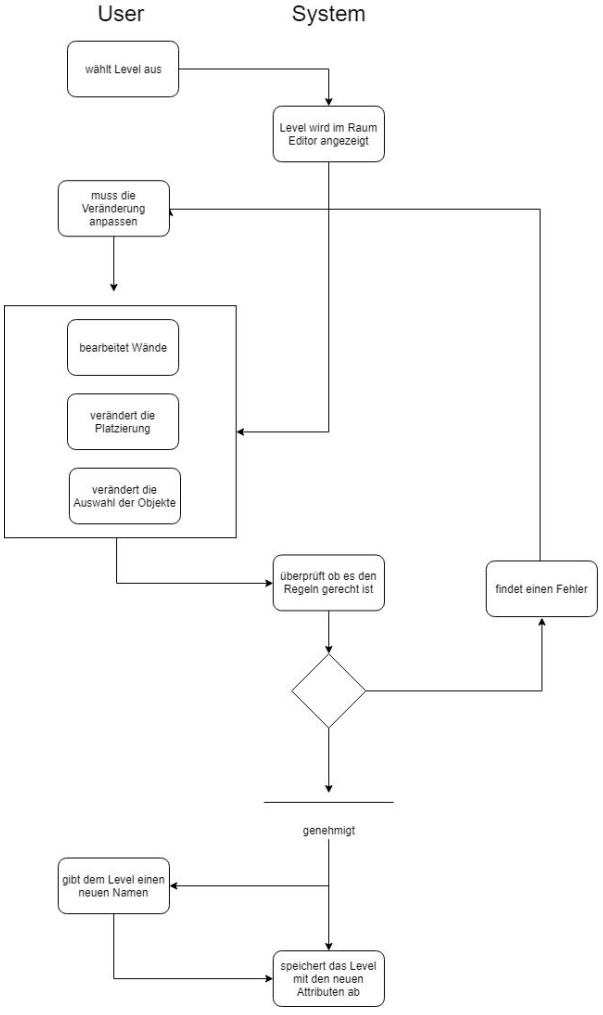
Verständliche Rückmeldung bei Fehlern,
Programm ist tolerant gegenüber falschen Platzierungen
Nur korrekte Level werden gespeichert

Parametrisierbarkeit/Flexibilität:

Wann immer ein User ein eigenes Level bearbeiten mag

Nutzungshäufigkeit/Mengengerüst: undefiniert


Zeigt die jeweiligen Level an
wählt einen Raum aus
Laden des ausgewählten Raumes
versucht alle Objekt zu platzieren
Überprüfung der Platzierung
verbraucht seine Versuche
Überprüfung Versuche
schafft es nicht rechtzeitig zu Platzieren
Überprüfung Zeit
speichert erreichten Punktestand
User
System


User
System
wählt Level aus
Level wird im Raum Editor angezeigt
muss die Veränderung anpassen
bearbeitet Wände
verändert die Platzierung
verändert die Auswahl der Objekte
überprüft ob es den Regeln gerecht ist
findet einen Fehler
genehmigt
gibt dem Level einen neuen Namen
speichert das Level mit den neuen Attributen ab

Level-Modus spielen

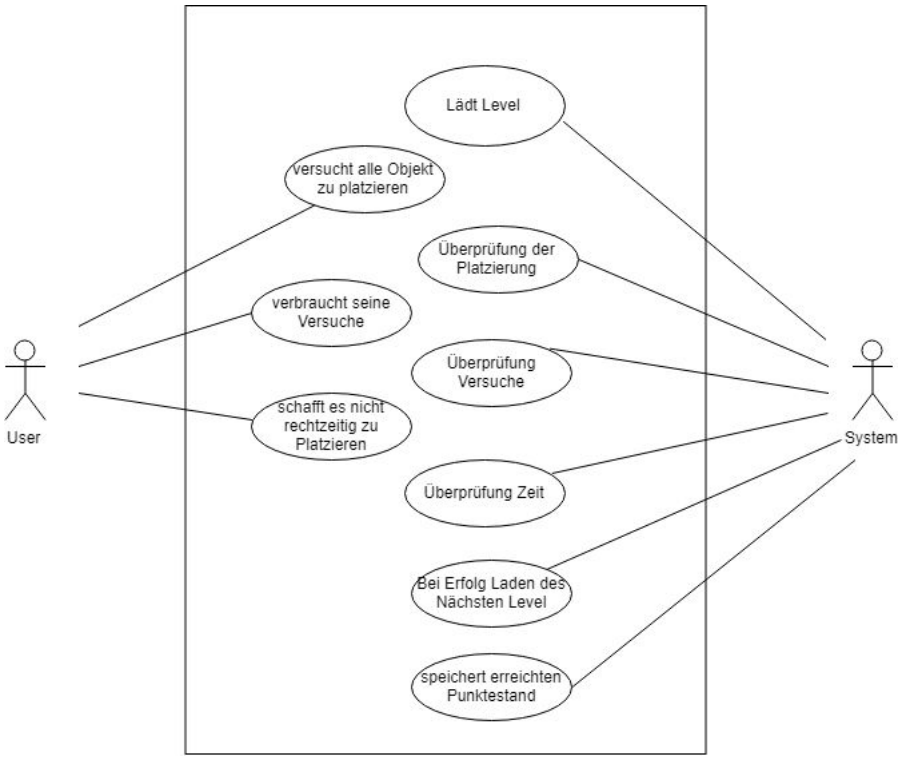
Akteure: User, Systeme
Fachlicher Auslöser: User möchte den Level-Modus spielen
Vorbedingungen: Anwendung muss gestartet sein und User muss bei der Auswahl auf "Level Modus" gedrückt haben, Level muss freigeschaltet sein
Standardablauf:

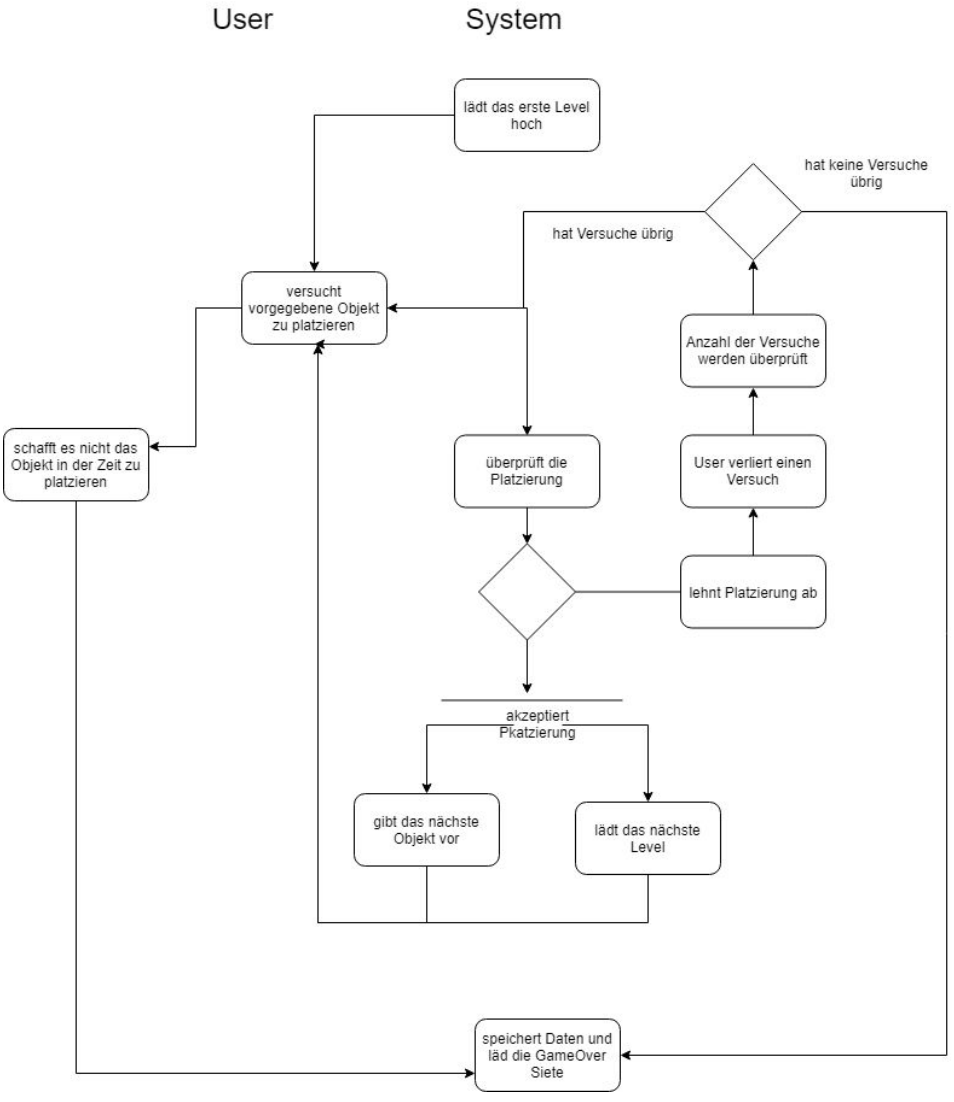
1. System: lädt das erste Level des Level-Modus hoch
2. User: versucht die Objekte in der vorgegebenen Zeit in das Zimmer zu platzieren
3. System: überprüft die Platzierung der Objekte
4. User konnte alle Objekte platzieren
5. System: zeigt die Highscore-Liste des gespielten Levels an
6. System: lädt das nächste Level – weiter mit 2
7a. User: verbraucht seine Versuche
7b. User: schafft es nicht, das Objekt in der Zeit zu platzieren
8. System: speichert die jeweiligen Daten (Name, Punktestand u. erreichtes Level), passt die Anzeige an

Alternative Abläufe/ Fehlersituationen/Sonderfälle:

3a. System findet Fehler bei der Platzierung des Objekts
3aa. System merkt sich den Fehler
3ab. System überprüft, wie viele Fehlversuche der User noch hat
3ac. weiter mit 2 oder weiter mit 7

Nachbedingung/Ergebnis: -
Nicht-funktionale Anforderungen: Verständliche Rückmeldung, Intuitive Bedienung
Parametrisierbarkeit/Flexibilität: Wann immer ein User ein eigenes Level spielen mag
Nutzungshäufigkeit/Mengengerüst: undefiniert


Lädt Level
versucht alle Objekt zu platzieren
Überprüfung der Platzierung
verbraucht seine Versuche
Überprüfung Versuche
schafft es nicht rechtzeitig zu Platzieren
Überprüfung Zeit
Bei Erfolg Laden des Nächsten Level
speichert erreichten Punktestand
User
System


User
System
lädt das erste Level hoch
hat keine Versuche übrig
hat Versuche übrig
versucht vorgegebene Objekt zu platzieren
Anzahl der Versuche werden überprüft
schafft es nicht das Objekt in der Zeit zu platzieren
überprüft die Platzierung
User verliert einen Versuch
lehnt Platzierung ab
akzeptiert Pkatzierung
gibt das nächste Objekt vor
lädt das nächste Level
speichert Daten und läd die GameOver Siete

Eigenes Level spielen

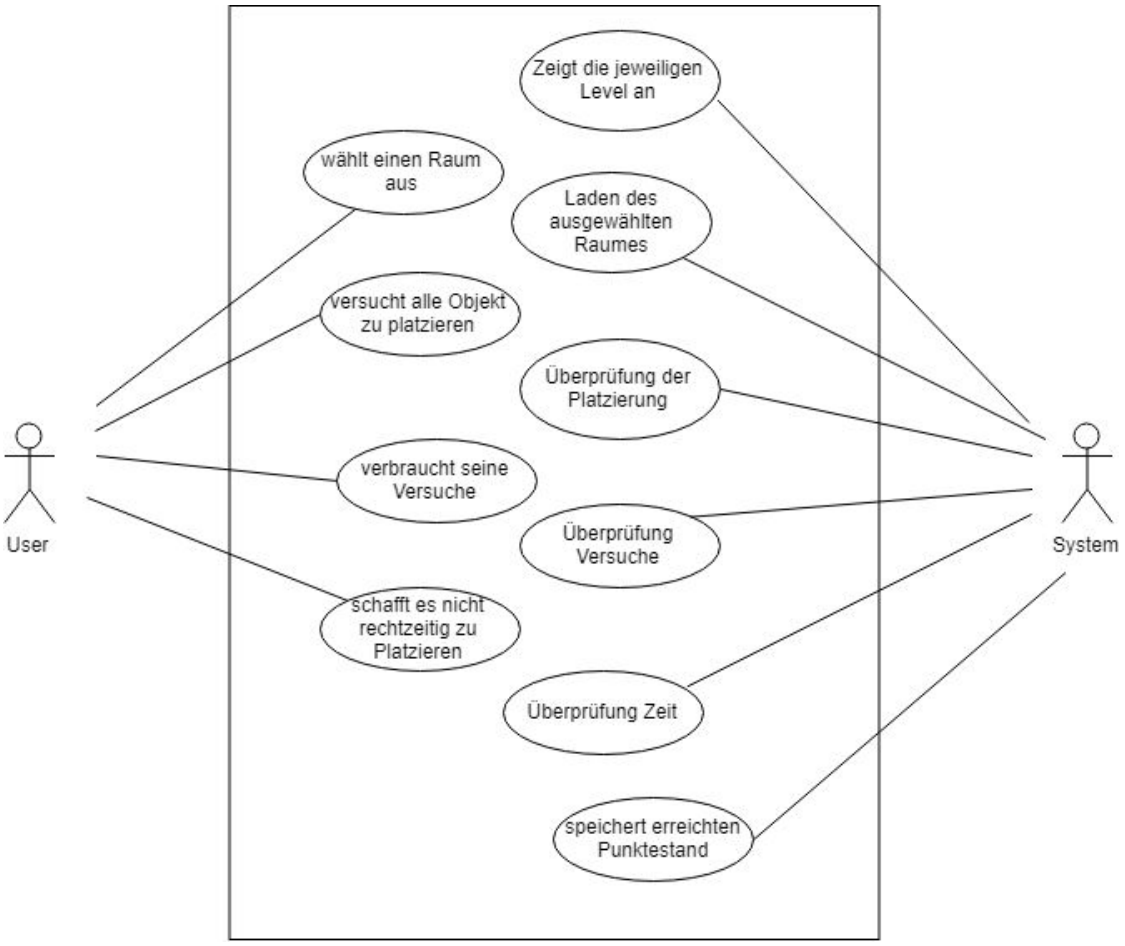
Akteure: User, Systeme
Fachlicher Auslöser: User möchte ein eigenes erstelltes Level spielen
Vorbedingungen: Anwendung muss gestartet sein und User muss bei der Auswahl auf "Kreativ-Modus" gedrückt haben
Standardablauf:

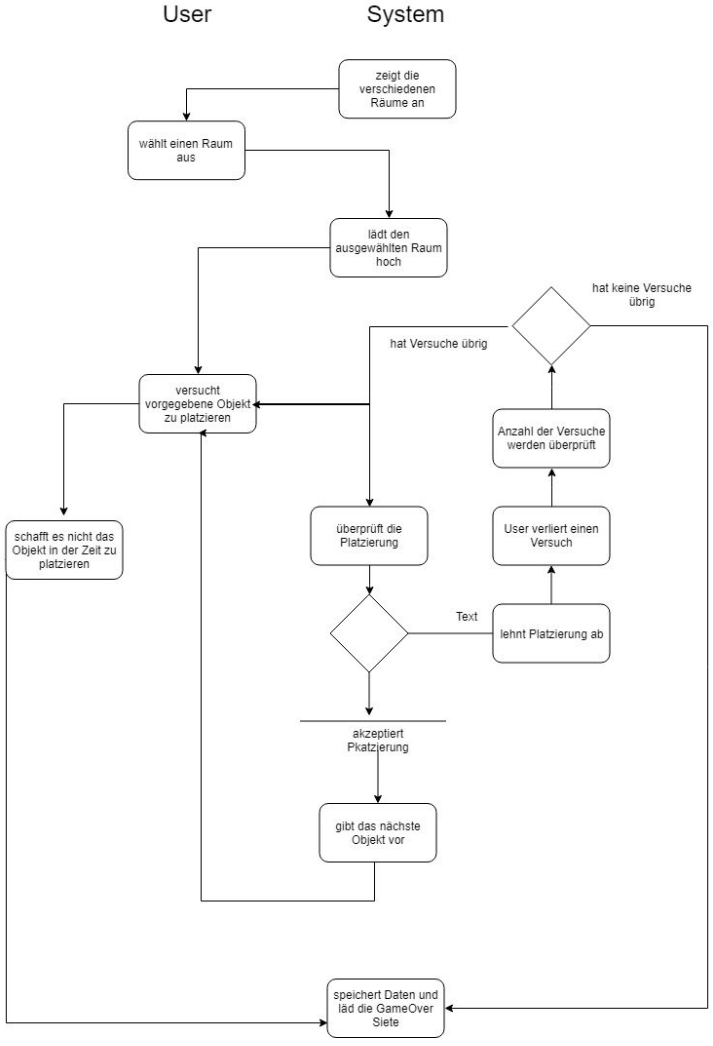
1. System: zeigt die verschiedenen Räume an, die kreiert wurden
2. User: wählt einen Raum aus
3. System: zeigt das ausgewählte Spiel/ den Raum an
4. User: versucht die Objekte in der vorgegebenen Zeit in das Zimmer zu platzieren
5. System: überprüft die Platzierung der Objekte
6. User konnte alle Objekte platzieren
7. System: lädt das nächste Level - weiter mit 2
8a. User: verbraucht seine Versuche
8b. User: schafft es nicht, das Objekt in der Zeit zu platzieren
9. System: speichert die jeweiligen Daten (Name, Punktestand u. erreichtes Level) passt die Anzeige an

Alternative Abläufe/ Fehlersituationen/Sonderfälle:

5a. System findet Fehler bei der Platzierung des Objekts
5aa. System merkt sich den Fehler
5ab. System überprüft wie viele Fehlversuche der User noch hat
5ac. weiter mit 2 oder weiter mit 6

Nachbedingung/Ergebnis: -
Nicht-funktionale Anforderungen: Korrekte Zeiterfassung, Verständliche Rückmeldung
Parametrisierbarkeit/Flexibilität: Wann immer ein User ein eigenes Level spielen mag
Nutzungshäufigkeit/Mengengerüst: undefiniert


Zeigt die jeweiligen Level an
wählt einen Raum aus
Laden des ausgewählten Raumes
versucht alle Objekt zu platzieren
Überprüfung der Platzierung
verbraucht seine Versuche
Überprüfung Versuche
schafft es nicht rechtzeitig zu Platzieren
Überprüfung Zeit
speichert erreichten Punktestand
User
System


User
System
zeigt die verschiedenen Räume an
wählt einen Raum aus
lädt den ausgewählten Raum hoch
hat keine Versuche übrig
hat Versuche übrig
versucht vorgegebene Objekt zu platzieren
Anzahl der Versuche werden überprüft
schafft es nicht das Objekt in der Zeit zu platzieren
überprüft die Platzierung
User verliert einen Versuch
Text
lehnt Platzierung ab
akzeptiert Pkatzierung
gibt das nächste Objekt vor
speichert Daten und läd die GameOver Siete

Funktionale Anforderungen:

Reaktion auf Fehlersituationen im Spiel
Wenn der Benutzer sich nicht an die oben genannten Regeln hält wird eine Meldung mit dem Fehler erscheinen.

Berechnung der Punktzahl
Es wird für jedes richtig gesetzte Objekt eine Punktzahl vergeben, die abhängig von der Objektgröße ist. Die erreichten Punkte werden am Ende des Spiels summiert und ausgegeben.

Berechnung der Zeit
Für jeden Gegenstand hat man 5 Sekunden Zeit, um ihn zu platzieren.

Speicherung von Benutzerdaten
Die Namen der einzelnen User werden mit den erzielten Highscores in eine JSON-Datei gespeichert und bei jedem Start geladen.

Speicherung der erstellten Level
Die im „Kreativ-Modus“ erstellten Level werden in einem Textformat gespeichert.

Abspielen von Musik
Es wird eine (Default)Playlist geben mit mp3-Dateien, die der MediaPlayer abspielen wird. Es wird zwischen Hauptmenü, Ingame und Erstellmusik unterschieden.

Funktionen im Spiel
Die Objekte werden mit Drag & Drop reingezogen und platziert. Objekte können weiterhin gedreht werden. Die Länge von Wänden sowie Fenstern ist änderbar.

Spielregeln

Gegenstände
Im Spiel tauchen folgende Einrichtungsgegenstände auf:

Dekoration	Bodenobjekte	Wandobjekte
Dino Einhorn Pflanze Teddybär	Teppich Tisch (Ablage) Fernsehtisch (Ablage) Stuhl Hocker Couch Bett Kleiderschrank (hoch) Hochbett (hoch, unterstellbar) Eckcouch	Wandschrank (hoch) Wandregal (Ablage, unterstellbar) Wanduhr (unterstellbar) Bild (unterstellbar)

Ziel eines Levels
Ein Level ist gewonnen, wenn alle vorgegebenen Gegenstände auf korrekte Art und Weise platziert wurde. Für jeden Gegenstand hat ein Spieler 5 Sekunden Zeit und bei einer Fehlplatzierung (s.u.) bekommt der Spieler zunächst einen neuen Versuch. Das Level geht verloren, wenn eine der folgenden Aktionen eintritt:
- Die Zeit für einen Gegenstand läuft ab
- Es finden insgesamt 3 Fehlplatzierungen statt

Wie darf ein Gegenstand platziert/ nicht platziert werden?
Kein Gegenstand darf vor einer Tür platziert werden.
Bodenobjekte dürfen nur auf dem Boden oder auf einem Teppich platziert werden.
Wandobjekte müssen an einer Wand platziert werden.
Dekorationen dürfen nur auf dem Boden oder auf Ablagen platziert werden.
Es dürfen keine Teppiche aufeinander liegen.
Vor/unter einem unterstellbaren Gegenstand darf kein hoher Gegenstand stehen.
Vor einem Fenster dürfen keine hohen oder unterstellbaren Gegenstände stehen.
Dino und Einhorn brauchen einen Mindestabstand.

Wie setzen sich die Punkte/Highscores zusammen?
Für jeden Gegenstand sind Punkte vorgegeben. Je größer, desto mehr Punkte. Somit bekommt man beim Bestehen eines Levels immer dieselbe Anzahl Punkte. Besteht man ein Level nicht, bekommt man dennoch die Punkte der Gegenstände, die man korrekt platzieren konnte. Haben im Highscore mehrere Spieler dieselbe Punktzahl, ist derjenige höherrangig, der weniger Zeit benötigt hat. Eine Fehlplatzierung stellt zusätzlich eine Zeitstrafe dar.
Bei den Spielen im Kreativ-Modus wird der Rang pro Level anhand der erreichten Punkte und an der benötigten Zeit berechnet.

Nicht-funktionale Anforderungen

Bedienbarkeit und Sprache
Um die Anwendung kindgerecht zu halten, soll sie intuitiv bedienbar sein. Der Großteil der Interaktion soll über die Maus erfolgen. Die Elemente der Benutzungsoberfläche und ihre Interaktion sollen vor der Nutzung erklärt werden. Dabei soll möglichst eine einfache Sprache und kurze Sätze verwendet werden.
Allgemein soll die Oberfläche kindgerecht gehalten werden.

Fehlertoleranz
Die Positionierung eines Gegenstandes soll zunächst im ganzen Raum möglich sein, eine Fehlplatzierung soll nur in die Bewertung des Users einfließen und keinen Einfluss auf die Funktion des Programms haben.

Benutzerinteraktion
Um das Spiel für Kinder verständlich zu machen, sollen Interaktionsschritte deutlich klar gemacht werden. Bei der Platzierung werden die Objekte durch ein Raster geleitet. Bei einer fehlerhaften Platzierung soll eine entsprechende Rückmeldung stattfinden. Die optische Rückmeldung soll durch Geräusche unterstützt werden.

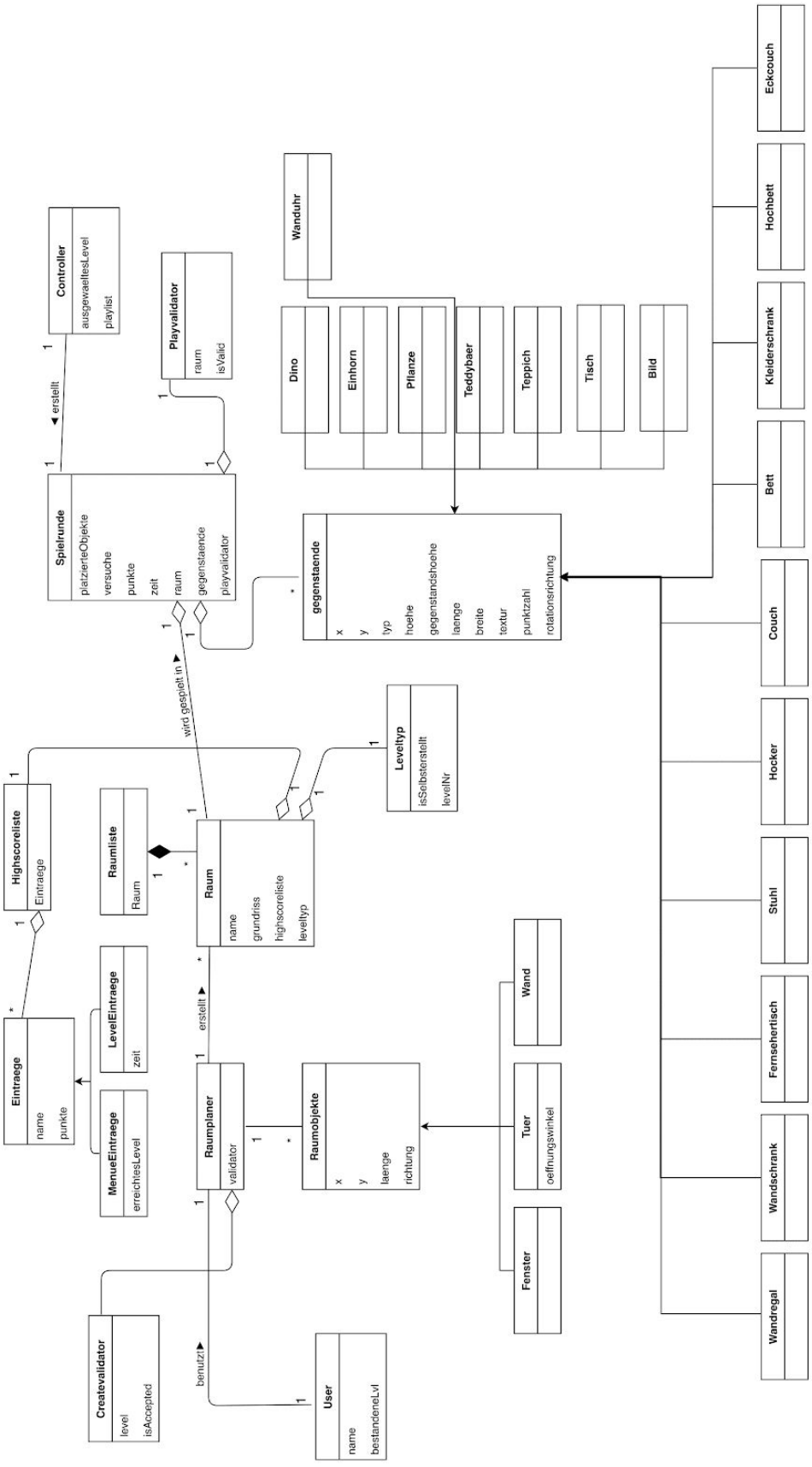
Korrektheit
Bei gleichen Eingaben während der gleichen Zeit in denselben Leveln soll immer dieselbe Punktzahl herauskommen. Es soll den Benutzern immer möglich sein, die Platzierung der Objekte unter den gleichen Performance-Bedingungen durchzuführen.

Austauschbarkeit
Grundrisse (auch selbst erstellte) sollen als Dateien persistent gespeichert werden können und auch zwischen mehreren Systemen austauschbar sein.

Technische Voraussetzungen

Eingabegeräte: Maus, Tastatur
Ausgabegeräte: Bildschirm, Auflösung minimal 800x600, optional Lautsprecher
Software: Java 11

Domänendiagramm



Datentypenverzeichnis

Objekte:

Raum
- grundriss = Bytearray

Gegenstaende
- typ= Enum (Dekoration, Bodenobjekte, Wandobjekte)
- style = Path

Raumliste
- Arraylist aus Räumen

Anwendungsfunktionen

Darstellung der Räume
Ein Raum wird zur Laufzeit in einem 2D-Byte-Array gespeichert. Die Feldwerte geben an, was sich an dieser Position befindet (Wand, Gegenstand, Tür etc.)

Überprüfung beim Erstellen eines Raums
Bei der Erstellung eines Raums mit Fenstern und Türen überprüft ein Validator, ob der Raum geschlossen ist und mindestens ein Fenster oder eine Tür vorhanden ist. Erst, wenn beides zutrifft, kann mit der Auswahl der Objekte fortgefahren werden.

Überprüfung der Platzierung
Um die Platzierung zu vereinfachen, soll die Spielfläche in Raster eingeteilt werden. Das Raster soll dem Array entsprechen und anhand dieser Koordinaten gibt ein Validator zurück, ob eine Platzierung regelgerecht ist.

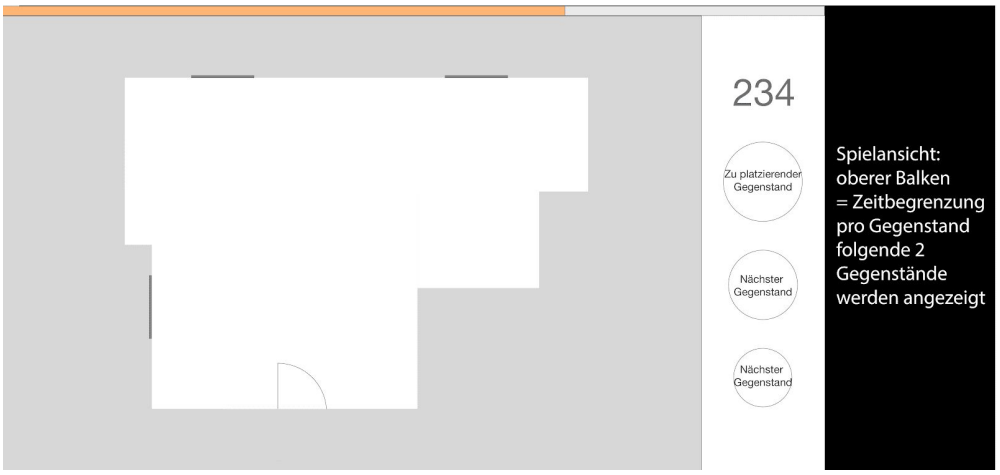
Visualisierung
Die Darstellung eines geladenen Raums oder einem in der Erstellung soll über JavaFX-Paths stattfinden.

Objekt-Erstellung eines erstellten Raums
Beim Klicken und dem damit verbundenen Erstellen von Wänden und Einrichtungsgegenständen sollen die Koordinaten direkt zwischengespeichert und in einem Byte-Array verwaltet werden. Löschen/verschieben oder abbrechen von Aktionen sollen analog dazu umgesetzt werden.

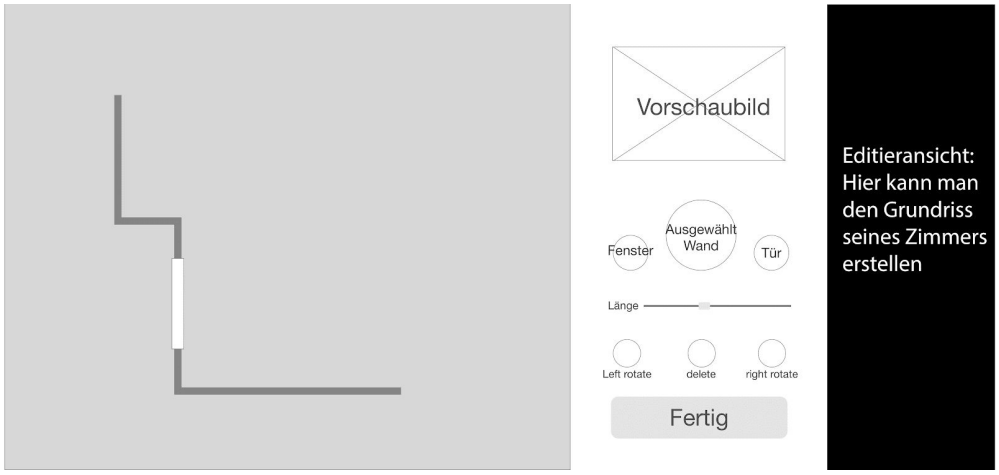
Persistente Speicherung eines Raums
Informationen über einen Raum werden in JSON-Dateien gespeichert.

Wireframes

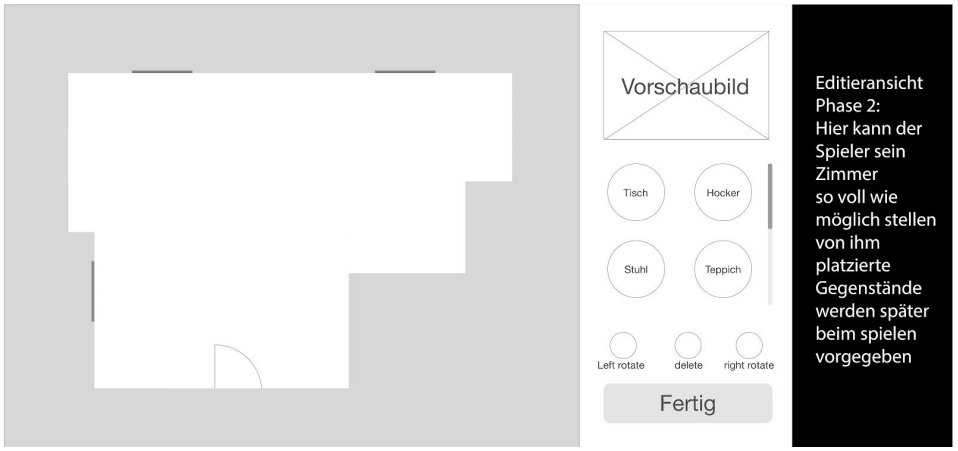
Spielansicht



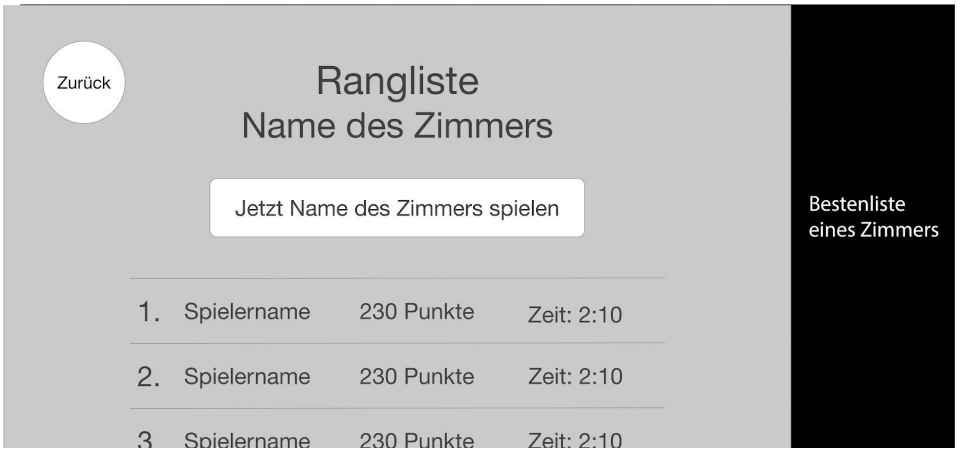
Editieransicht



Editieransicht 2



Bestenliste eines Zimmers



Schnittstellen

JSON
Die Array-Struktur eines Raums wird für die persistente Speicherung in ein JSON-Array übertragen und mit Zusatzinformationen (Name etc.) in einer Datei gespeichert. Außerdem werden Daten wie Name, Punktestand sowie weitere Attribute die für die Highscore-Einträge benötigt werden auch in dem Dateiformat gespeichert und überschrieben.

MediaPlayer
Für das Abspielen der Musik benutzen wir eine vorgefertigte API von JAVA FX.