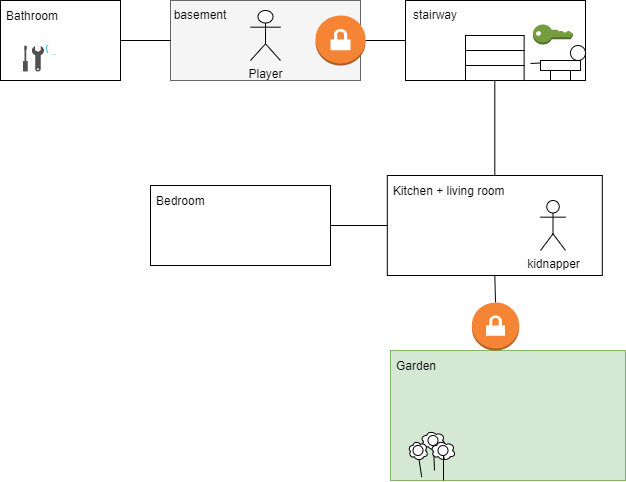
# Spielidee

## 2.1 Eigene Räumlichkeiten

Wir haben uns für die folgenden sechs Räumlichkeiten entschieden:

|  |  |  |
| --- | --- | --- |
| Ort | Items | Ausgänge |
| Outside | - | North: kitchen |
| Bathroom | Hammer, Wardrobe | East: basement |
| Basement | - | East: stairway, West: bathroom |
| Stairway | Key, Plant | South: kitchen, West: basement |
| Kitchen | Winebottle | Norht: stairway, South: outside |
| Bedroom | Wardrobe, Bed | East: kitchen, West: bedroom |

Um dies besser zu verdeutlichen ist hier eine Skizze:



## 2.2 Gewonnen

Wir haben uns für di simple Umgebung von einer Wohnung entschieden, weil diese am besten zum Szenario einer Entführung passt. Unsere eigentliche Idee stammt von einem Film, die Geschichte, die in unserem Spiel erzählt wird, ist jedoch nicht ganz dieselbe. Es geht darum, dass der Spieler entführt wurde. Mit Hilfe von Items, welche man in der Umgebung finden und aufheben kann, kann man seinem Entführer entkommen, was uns auch gleich zum Ziel des Spieles bringt, und zwar nach draussen zukommen.

## 2.3 Befehl «back»

Um den back Command realisieren zu können, haben wir die letzte Richtung, in welche der Spieler gegangen ist im String «currentDirection» gespeichert. Diesen String haben wir dann der «lastRoom» Funktion gegeben, welche anhand dieses Property die entgegen gesetzter Richtung zurückgibt. Also wird aus der Richtung «north» gleich «south».

Methode, welche durch den vom User eingegebenen Command aufgerufen wird:

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Methode, welche die letzte Richtung in die entgegengesetzte Richtung umwandelt und als String returnt.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

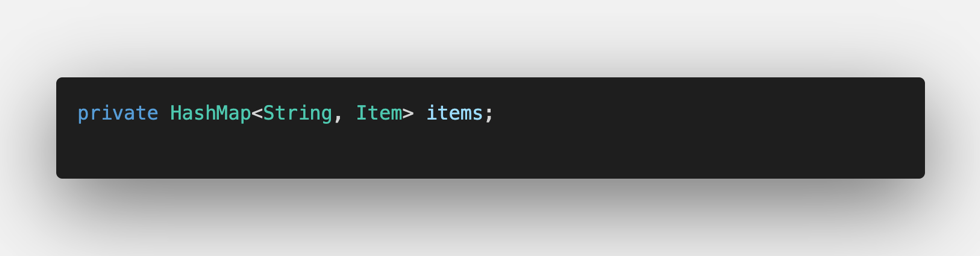
## 2.4 Gegenstand im Raum / 2.5 Mehrere Gegenstände im Raum / 2.6 Gewicht

Als erstes haben wir die Item Klasse erstellt, diese hat 2 Attribute, den Namen des Gegenstandes und dessen Gewicht. Für jedes Attribut wurde ausserdem ein «getter» und ein «setter» generiert.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Als nächstes haben wir in der Room klasse die HashMap items erstellt, welche als key den Namen vom item und das Items selbst als value speichert.



Damit man ein Item dann auch in dieser HashMap speichern kann, brauchte es noch einen «setter». Jetzt kann man simpel Items einem Raum geben.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Damit man auch sieht welche Items sich gerade in einem Raum befinden, haben wir die Methode «listItems» erstellt, welche einfach alle Items welche sich im Raum befinden in die Konsole schreibt.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Diese Methode wird dann aufgerufen, wenn der User den Command «show items» eingibt. Die Methode, welche schlussendlich «listItems» aufruft, wird auch noch dafür verwendet die Items im Spieler Inventar anzuzeigen, mehr dazu später.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

## 2.7 Befehl «map»

Wie im Auftrag beschrieben haben wir eine ArrayList erstellt welche alle Räume speichert.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Danach haben wir die Methode «printMap» erstellt, welche die Richtung, aus welcher man kam, den Raum in welchem man sich gerade befindet und alle anderen Räume mit ihren Items in die Konsole printet.

Sobald der User den Command «map» eingibt wird «printMap» aufgerufen.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Gekürzter Output:

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

# 3.Spieler Inventar

Als erstes haben wir die Klasse Inventory erstellt, welche die HashMap «itemList» und den Integer «totaltemWeight» enthält. Für beide sind jeweils getter und setter Methoden erstellt worden.

Mit der Methode setItem, setzten wir ein item in die HashMap und das Gewicht des Items geben wir dem setter «setTotalWeight» mit.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Im «setter» von «totalWeight», setzten wir den jetzigen Wert von «totalWeight» plus den Input, damit wir das totale Gewicht des Spielers berechnen können. Diese Methode wird jedes Mal aufgerufen, wenn der Spieler mit dem Command «take» ein Item aufhebt.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Mit der Methode «showItems», printen wir alle Items und das Gewicht, welches der Spieler gerade hat, in die Konsole, dies wird jedes Mal aufgerufen, wenn der Spieler ein Item aufhebt oder weglegt.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Output Beispiel:

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Mit dieser Methode wird ein Item aus der «itemList» anhand des keys, in dem dem Fall der Name des Items, ein Item entfernt, bzw. es wird vom Spieler Inventar entfernt und dem Raum, in welchem sich der Spieler gerade befindet, übergeben. Ausserdem wird hier «totalItemWeight» auf den richtigen wert gesetzt.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Hier wieder in der Game Klasse, haben wir die Methode «takeItem» implementiert, welche jedes Mal aufgerufen wird, wenn der User den Command «take» eingibt. In der HashMap «currentItems» werden, anhand der Methode «getItems» die Items welche sich in dem Raum, in welchem man sich gerade befindet, gespeichert. Dann wird über diese Items gemappt. Und falls das jetzige Gewicht des Spielers nicht über 100 ist und das 2 Word im User Command gleich ist, wie der Name eines Items, welches sich gerade in diesem Raum befindet, wird dieses Item aus der Raum «itemList» HashMap entfernt und der Inventory «itemList» hinzugefügt.

Damit man auch sieht, was sich gerade im Spieler-Inventar befindet, wird die Methode «showItems» von vorhin aufgerufen.

Ein Bild, das Text, Screenshot, Monitor enthält.

Automatisch generierte Beschreibung

Die Methode «dropItem» ist dafür zuständig, dass Items aus dem Inventar entfernt und einem Raum hinzugefügt werden kann. Im Prinzip wird genau dasselbe wie in der «takeItem» Methode gemacht nur umgekehrt. Der Ablauf ist dabei derselbe. Aus der Inventar HashMap wird das Item entfernt und der Raum HashMap wird das Item hinzugefügt.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Info: Mancher Code könnte sich in der Zwischenzeit verändert haben, denn die Doku wurde während der Entwicklung der Methoden erstellt. Es werden wahrscheinlich noch einige Verbesserungen am Code stattfinden.