

1 Zork 2 – Game Projekt

Thema: Mit Klassen, Eigenschaften, Objekten, Listen, JavaDoc und JUnit arbeiten.

Partnerarbeit: maximal 2 Lernende

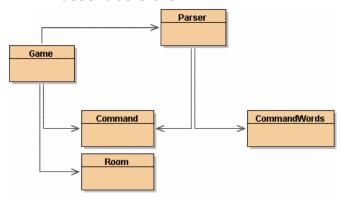
Zeit: ca. 12 Lektionen

Präsentation: Das eigene Spiel wird der Klasse in ca. 5 Minuten Vorträgen präsentiert.

1.1 Einleitung

Das Zork 2 ist eine einfache Basis für ein Abenteuer Spiel. Es implementiert Räume, welche man betreten und wieder verlassen kann.

1.2 Klassenübersicht

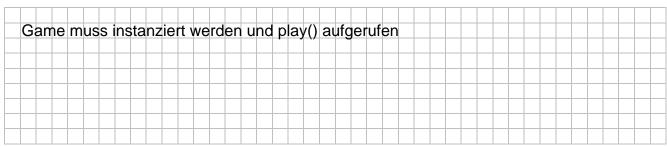


1.3 Hauptklasse für Zork

Eine Hauptklasse ist die Klasse, welche den Entry Point des Programmes bildet. Hier fängt also alles an. Sie besitzt eine statische Methode *main*. Dank dieser Methode weiss die intellij Umgebung wo das Programm mit der Ausführung beginnen soll.

Erstellen Sie die Klasse Zork2, welche uns als Hauptklasse dienen soll. Bei der Erstellung dieser Klasse kann die Erstellung der Methode *main* in eclipse direkt angegeben werden.

Welche Klasse muss in der main Methode instanziert werden, und welche Methode dieser Klasse muss aufgerufen werden, damit das Programm startet?





1.4 Erzeugung der Räume (aus dem Beispiel)

```
Room outside, lab, tavern, gblock, office;
...

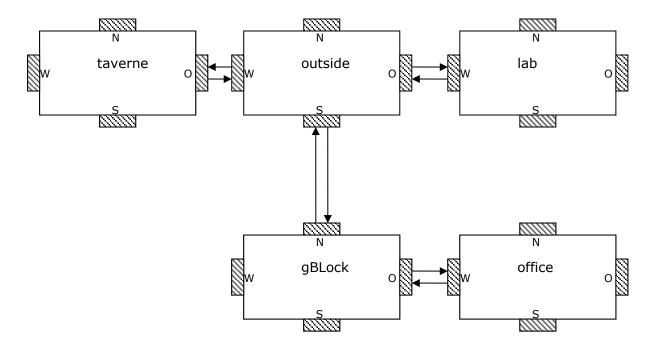
// create the rooms
outside = new Room("outside G block on Peninsula campus");
lab = new Room("a lecture theatre in A block");
tavern = new Room("the Seahorse Tavern (the campus pub)");
gblock = new Room("the G building");
office = new Room("the computing admin office");

// initialise room exits
outside.setExits(null, lab, gblock, tavern);
lab.setExits(null, null, null, outside);
tavern.setExits(null, outside, null, null);
gblock.setExits(outside, office, null, null);
office.setExits(null, null, null, gblock);

currentRoom = outside; // start game outside
```

1.5 Raumsituation

Dies ist eine Skizze der Räume aus dem vorgegebenen Beispiel.

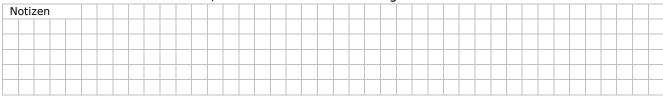




2 Aufgabe 1 – Spielidee - Dokumentation:

2.1 Eigene Räumlichkeiten definieren.

Definieren Sie eingene Räumlichkeiten eines Gebäudes oder einer Landschaft, ganz nach Ihrer Phantasie. Übertreiben Sie nicht, wählen Sie für den Anfang maximal 10 Räume.



a) Erstellen Sie eine Skizze und Beschreibung in ihrer Dokumentation und zeigen Sie diese der Lehrperson.

2.2 Gewonnen

Diesem Spiel fehlt eine massgebende Eigenschaft, welche wir jetzt zusätzlich implementieren wollen. Wir wollen implementieren, dass wir auch "gewinnen" können.

a) Beschreiben Sie in ihrer Dokumentation wann Sie gewonnen haben!

Implementieren Sie diese Funktion so, dass man beim Erreichen eines bestimmten Raumes gewinnt. Erweitern Sie die Klasse Raum mit einer Eigenschaft, über welche herausgefunden werden kann ob es sich bei einem bestimmten Raum um den "Gewinner Raum" handelt. Definieren Sie einen solchen Raum (Objekt), bei welchem man gewinnt, sobald man den Raum betritt.



Tip:

Untersuchen Sie wie die Ausgabe der Raumbezeichnung und die möglichen Ausgänge des aktuellen Raumes, programmiert ist. Konkret die Methode *longDescription()* der Klasse Room. Versuchen Sie es zu verstehen und analog zu programmieren.

2.3 Befehl "back"

Implementieren Sie den Befehl *back*. Dieser Befehl bringt uns zu diesem Raum, von dem wir gekommen sind.

a) Beschreiben Sie in ihrer Dokumentation wie Sie es realisieren möchten!

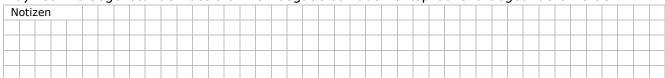
2.4 Gegenstand im Raum

Im Raum kann ein Gegenstand sein. Erzeugen Sie eine Klasse *Item* mit den Eigenschaften und den dazugehörenden Methoden. Erweitern Sie die Klasse *Room* so, dass man dem Objekt einen Gegenstand hinzufügen kann.

- a) Beschreiben Sie in ihrer Dokumenation wie Sie vorgehen!
- b) Wir müssen auch nach dem Gegenstand im Raum "fragen" können. Dokumentieren Sie wie es realisieren.

2.5 Mehrere Gegenstände

Im Raum können mehrere Gegenstände sein. Erweitern Sie die Klasse *Room* so, dass es eine *ArrayList* mit Gegenständen besitzt. Die Ausgabe soll auch entsprechend abgeändert werden.





2.6 Gewicht

Ein Gegenstand hat ein Gewicht. Erweitern Sie die Klasse Item mit einem Attribut weight.

a) Erweitern sie die Klasse und beschreiben Sie wie sie vorgehen!

2.7 Befehl "map"

Ergänzen Sie die Klasse *Game* mit einer Methode, welche auf Befehl alle Räume mit allen Gegenständen ausgibt. Erweitern Sie dazu die Klasse *Game* mit einer *Arraylist*, welche die Räume speichert.

a) Aus der Ausgabe soll auch ersichtlich sein, in welchem Raum man sich gerade befindet.

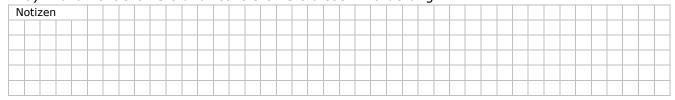
Notizen	



3 Schwierigere Aufgaben

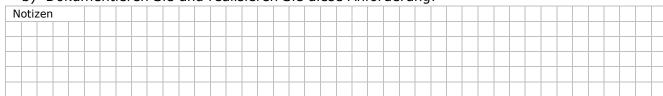
3.1 Der Spieler kann ein Gegenstand bei sich tragen

a) Dokumentieren Sie und realisieren Sie diese Anforderung!



3.2 Der Spieler kann mehrere Gegenstände tragen (mit einem Maximalgewicht)

b) Dokumentieren Sie und realisieren Sie diese Anforderung!



3.3 Der Spieler kann Gegenstände auch wieder ablegen

c) Dokumentieren Sie und realisieren Sie diese Anforderung!

