Elisabeth- von-Thüringen-Gymnasium

Schuljahr 2019/2020

Facharbeit im Fach Informatik

BetreuerIn: Frau Becker

**Konzepte der Informatik in Minecraft erläutern**

vorgelegt von

Felix Mansel

Köln, den 27. Februar 2020

Felix Mansel

Schellenhofweg 11

50858 Köln

Tel: 0221 6110818

E-Mail: [xilefma@web.de](mailto:xilefma@web.de)

Inhaltsverzeichnis

1. Einleitung X
2. Auswahl der IDE X
3. Einrichten des Servers
4. Aufbau eines Spigot Plugins (Spigot API)
5. Aufbau von Listener
6. Aufbau von Command
7. Mein Projekt
8. **Hashmap**
9. **Doppelte While-schleife**
10. **externe Methodenaufrufe (in anderen Konzepten enthalten)**
11. **Vererbung (in anderen Konzepten enthalten)**
12. **For-Each Schleifen**
13. **Switch-Statements**
14. **Rückgabewerte (in anderen Konzepten enthalten)**
15. **Singelton (in anderen Konzepten enthalten)**
16. DOKU
17. DIAGRAMM
18. Zusammenfassung warum Minecraft sich anbietet

# Einleitung

Im Rahmen meiner Facharbeit habe ich mir die Frage gestellt in wie weit gängige Konzepte der Informatik anhand des Computerspiels Minecraft demonstriert werden können.

Minecraft ist ein auf Java-basierendes Open-World-Spiel, in dem die gesamte Welt aus Blöcken besteht, die ein Spieler beliebig abbauen oder platzieren kann. So hat der Spieler die Möglichkeit allein oder im mehrspieler-Modus auf Servern die Welt nach eigenen Vorstellungen zu verändern. Zusätzlich werden dem Spieler verschiedene Modi zur Verfügung bestellt die entweder auf Kreativität oder überleben im Open-World Spiel setzen.

Nachdem ich mich intensiv mit Minecraft und dessen Programmierung auseinander gesetzt habe bin ich zum Schluss gekommen, dass für meine Facharbeit der serverbasierte Mehrspielermodus am interessantesten ist und die meisten Möglichkeiten bietet meine Ideen umzusetzen.

Viele der ebenerwähnten Server basieren auf der SpigotAPI, die neben den normalen Spielfunktionalitäten im Mehrspielermodus auch als ein Plugin-Entwicklungs-Framework dient. Anders als bei vielen anderen Modifikationen liegt durch Plugins nur eine serverseitige Modifikation vor. Es besteht also nicht die Möglichkeit neue Inhalte, wie beispielsweise neue [Texturen](https://minecraft-de.gamepedia.com/Texturen) oder [Sounds](https://minecraft-de.gamepedia.com/Sound) hinzuzufügen. Abgesehen davon hat man aber viele Möglichkeiten das Spielgeschehen neu zu gestalten.

Hier sind einige gängige Erweiterungen durch Plugins aufgelistet:

* Verwalten von Spielerdaten in Datenbanken (Beispielsweise ob er bereits gespielt hat und somit in der Datenbank steht)
* Spielmodi, die standartmäßig nicht zur Verfügung gestellt werden, werden ermöglicht (Zum Beispiel „Capture the Flag“ ein Eroberungsspiel, dass von Plugin Entwicklern durch Schutzzonen realisiert wird.)

Generell sind die Möglichkeiten hier nur durch die Fantasie und Programmierkünste begrenzt, da Spigot für nahezu alle Aktionen und Events im Spiel Modifikationsmöglichkeiten bietet.

# **Auswahl einer IDE (Integrated Development Environment)**

Mit welcher Java-IDE das Plugin entwickelt wird ist für das Resultat nicht wichtig.

Ich habe mich für Eclipse entschieden, weil ich ich mich in der IDE schon auskenne und mich so nicht in eine komplett neue Programmierumgebung einarbeiten muss.

# Aufbau eines Plugins für die SpigotAPI

Grundaufbau

Jedes Plugin, das für die SpigotAPI geschrieben wird, unterliegt einigen Konventionen. Es besteht aus einem Java-Projekt in welches zunächst die SpigotAPI eingebunden werden muss, um zu gewährleisten, dass man Zugriff auf die Bibliothek erhält.

Anschließend muss die Hauptklasse des Projektes modifiziert werden, denn Sie muss der Klasse JavaPlugin der SpigotAPI untergeordnet sein. In meinem Fall sieht das also wie folgt aus:

public class Facharbeit extends JavaPlugin {…}

Später beim Starten des Servers werden einige Methoden ausgeführt, die für die Aktivierung der Funktionen des Plugins notwendig sind.

In diesen Methoden müssen EventListener oder erweiterte Commands aus anderen oder derselben Klasse registriert werden, da jene letztendlich das Spielgeschehen beeinflussen.

public void onEnable() //wird nach dem starten des Servers ausgeführt

public void onDisable()//wird nach dem stoppen ausgeführt

Des Weiteren muss Jedes Projekt eine plugin.yml Datei beinhalten. In dieser Datei sind alle Basis-Informationen wie Pluginname, Autor, Version und eine Referenz auf die Hauptklasse enthalten. Da es sich um eine YAML bzw yml Datei handelt sind Einrückungen und Zeilenabstände sehr wichtig.

In meinem Beispiel sieht diese Datei wie folgt aus:

name: Facharbeit

main: de.felix.facharbeit.main.Facharbeit

version: 1.0

commands:

countdown:

description: Mit diesem Command werden Switch-Statements demonstriert

hashmap:

description: Mit diesem Command werden Hashmaps demonstriert

foreach:

description: Mit diesem Command werden For-Each Schleifen demonstriert

sortieren:

Die Grundstruktur ist gesetzt und das Plugin muss nur noch als JAR-Datei exportiert und in den von Spigot erstellten Ordner Plugins auf dem Server abgelegt werden.

Allerdings hat das Plugin nach diesen Modifizierungen keinerlei Funktion.

Hauptsächlich wird das Spielgeschehen durch Commands oder EventListenern verändert.

Im folgenden möchte ich den Aufbau eines Listeners anhand eines Beispiels erläutern.

Aufbau eines Listeners

Ich möchte ein Plugin programmieren und vorstellen, dass Konzepte der Informatik im Spiel umsetzt und demonstrieren kann.

Hierfür habe ich mir einige Konzepte überleg, die Ich vorstellen möchtet:

1. **Hashmap**
2. **Doppelte While-schleife**
3. **externe Methodenaufrufe (in anderen Konzepten enthalten)**
4. **Vererbung (in anderen Konzepten enthalten)**
5. **For-Each Schleifen**
6. **Switch-Statements**
7. **Rückgabewerte (in anderen Konzepten enthalten)**
8. **Singelton (in anderen Konzepten enthalten)**

**Zunächst muss aber erstmal deutlich werden wie Plu**

Aufbau eines Listeners

Aufbau eines Commands

Die API

Eine vollständige Dokumentation der SpigotAPI ist hier zu finden:

VERSION aber 1.15

MEINE VERSION :

<https://hub.spigotmc.org/nexus/content/groups/public/org/spigotmc/spigot-api/1.8.8-R0.1-SNAPSHOT/>

Die Versionen unterscheiden sich nicht wesentlich.

Da ich aber Minecraft nur in der Version 1.8 gespielt habe keine ich mich in dieser Version besser auf und habe mich deshalb für diese entschieden.