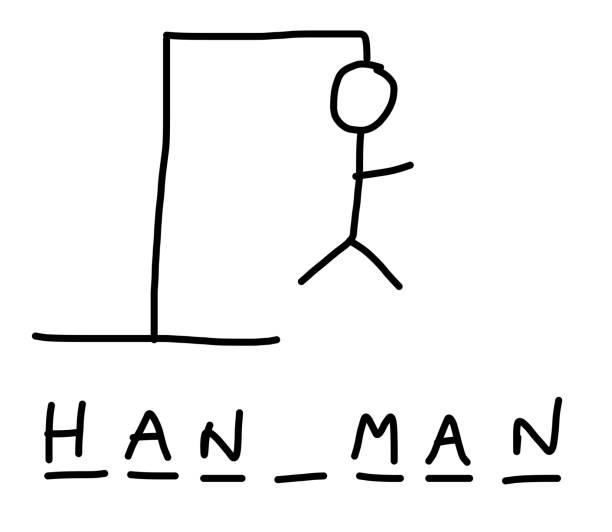
**Dokumentation zum Projekt**

**GALGENMÄNNCHEN**



Projektarbeit von

Celine Gugel Nr. 2707682

Lena Steinbrink Nr. 5513041

Valentin Müller Nr. 5461128

Jannik Oßwald Nr. 1527904

und Thi Tuong Vy Nguyen Nr. 8610085

Inhalt

[1. Ideenfindung 2](#_Toc122348663)

[**2.** **Spielerklärung** 2](#_Toc122348664)

[3. Benötigte Funktionen 2](#_Toc122348665)

[3.1 Auswahl Spielmodus​ 2](#_Toc122348666)

[3.2 Buchstabeneingabe 3](#_Toc122348667)

[3.3 Ausgabe der Konsole 3](#_Toc122348668)

[3.4 Spielablauf 3](#_Toc122348669)

[4. Spielaufbau 4](#_Toc122348670)

[4.1 Draw Hangman 4](#_Toc122348671)

[4.2 Hangman Java 4](#_Toc122348672)

[4.3 Spiel.java 6](#_Toc122348673)

[**5.** **Zusammenarbeit** 7](#_Toc122348674)

# Ideenfindung

Um eine Idee davon zu bekommen, welches Spiel wir umsetzen wollen, haben wir uns zunächst selbst informiert, um uns anschließend für ein Spiel zu entscheiden. Zuerst kam uns hierbei das Spiel Flappy Bird in den Sinn. Da hierzu jedoch eine Grafik benötigt werden würde und wir vor allem das in den Vorlesungen angeeignete Wissen nutzen wollten, haben wir uns schlussendlich für das altbekannte Spiel Galgenmännchen entschieden.

1. **Spielerklärung**

Bei dem Spiel Galgenmännchen muss ein Begriff erraten werden. Es sind jedoch nur Striche in der Anzahl der zu erratenen Buchstaben des Wortes gegeben. Der Spieler muss nun einen Buchstaben eingeben. Ist der Buchstabe im Wort vorhanden, erscheint er an der passenden Stelle. Bei falsch geratenen Buchstaben bildet sich nach und nach eine Abbildung eines Galgens. Wurden zu viele Buchstaben falsch genannt, sodass der Galgen vervollständigt ist, so hat der Spieler das Spiel verloren. Sollte der Spieler jedoch vorher das Wort erraten, so kann er sich als Sieger verkünden.

# Benötigte Funktionen

Damit wir mit der Programmierung beginnen können, muss zunächst festgelegt werden, wie wir das Spiel aufbauen wollen.

3.1 Auswahl Spielmodus​

Wenn der Spieler das Spiel startet, soll er zunächst die Auswahl haben, ob er gegen den Computer spielen möchte oder ein Spiel mit eigener Worteingabe starten möchte. Bei einem Spiel gegen den Computer soll aus einem Array mit vorgegebenen Wörtern ein Wort zufällig ausgegeben werden und das Spiel startet. Wird ein Spiel mit eigener Worteingabe gewählt, so wird über die Scanner-Funktion ein Wort als String abgefragt. Damit bei dem Beginn des Spiels die Spielauswahl nicht mehr in der Konsole zu sehen ist, fügen wir 60 Leerzeichen ein. Es gibt auch noch einen dritten Modus, mit welchem man das Spiel verlassen kann.

## 3.2 Buchstabeneingabe

Wir legen eine neue Klasse Spiel. Somit kann man die Funktion zur Buchstabeneingabe programmieren mit den verschiedenen Bedingungen. Die erste Bedingung ist ein Char, damit nur der erste Buchstabe genommen wird. Die zwei anderen Bedingungen sind, dass das Wort zum einen mindestens zwei Zeichen hat und zum anderen, dass es keine Leerzeichen oder Sonderzeichen enthält. Zudem braucht man eine Funktion, die den Buchstaben mit dem gesuchten Wort vergleicht und uns später diesen schon benutzten Buchstaben auch ausgibt. Außerdem benötigen wir ein Array mit den Buchstaben an der entsprechenden Stelle vom Wort für die Ausgabe in der Konsole.

## 3.3 Ausgabe der Konsole

* „Hurra! Deine Eingabe war richtig!”, wenn Eingabe richtig ist​
* „Falsche Eingabe“, wenn Eingabe falsch ist​
* "Das Spiel Hangman wurde gestartet!“ wenn das Spiel gestartet wurde ​
* "Spielmodus 1: Spiele gegen den Computer"​
* "Spielmodus 2: Gib ein eigenes Wort ein (Multiplayer)"​
* „Wenn du das Spiel beenden möchtest, kannst du exit eingeben.“ damit der Spieler weiß, wie er das Spiel beenden kann​
* „Es wurde Spielmodus 1/2 gewählt.“ um dem Spieler mitzuteilen, in welchem Spielmodus er sich befindet​
* „Ein Buchstabe bitte:“ als Aufforderung für den Spieler ein Wort einzugeben​
* „Das Spiel wird beendet“, wenn der Spieler das Spiel beendet

## 3.4 Spielablauf

Eine andere Funktion, die wir benötigen „zeichneSpiel“, um die Zeichnung dann auch ausgeben zu können. Als weitere Funktion haben wir eingebaut, dass die Konsole “Hurra! Deine Eingabe war richtig!” ausgibt. Beispiele für benötigte variablen sind unter anderem *anzahlFalscheBuchstaben (int), spielVerloren (boolean), spielGewonnen (boolean), arrFalscheBuchstaben, arrGegebeneWörter.* Mögliche Zusatzfunktionen sind auch, dass die Konsole eine Nachricht ausgibt bei falscher Eingabe oder wenn man Sonderzeichen/Zahlen eingibt.

# Spielaufbau

## 4.1 Draw Hangman

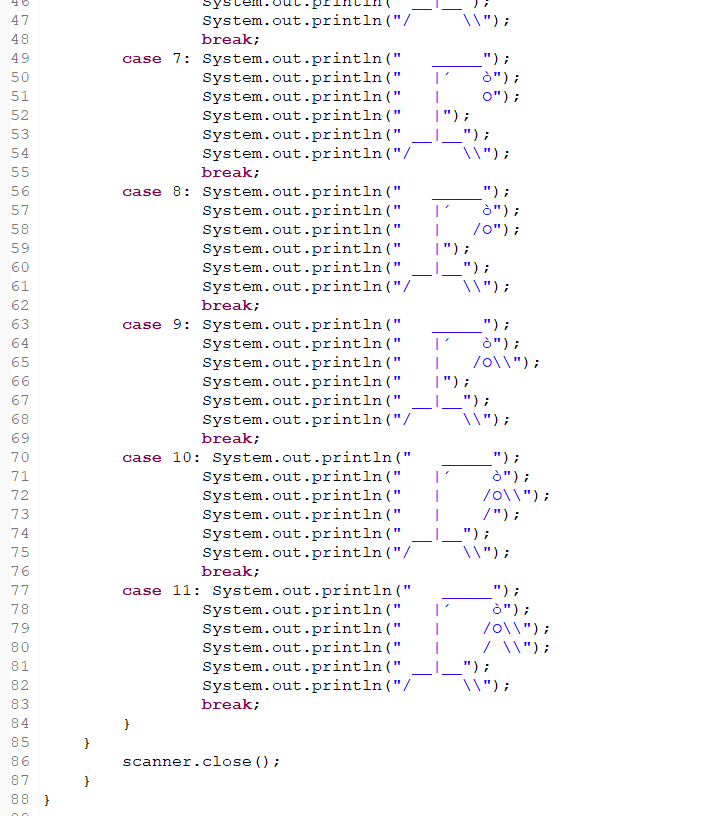
Wir haben uns Gedanken darüber gemacht, wie wir den Galgen und die kleine Person darstellen möchten. Dazu haben wir zunächst einige Optionen ausprobiert und uns am Ende für die Folgende entschieden. Um diese nun darstellen zu können, haben wir zunächst eine Klasse namens DrawHangman erzeugt.

public static void main(String[] args) {

// TODO Auto-generated method stub

Scanner scanner = new Scanner(System.in);

int a=0;

**** while (a<13) {

System.out.println("get new number:");

a = scanner.nextInt();

switch (a) {

Wir haben eine switch-Anweisung angelegt und 11 cases erstellt. Bei jedem case kommt immer ein weiterer Strich für den Galgen und das Strichmännchen hinzu. Durch System.out.println werden die Striche abgebildet. Nach jedem break wird die Schleife komplett verlassen und nach der Schleife fortgesetzt. Ist der letzte case durchlaufen, so wird das Spiel durch scanner.close() beendet.

## 4.2 Hangman Java

Wir haben eine Klasse Namens Hangman erstellt und einen Scanner angelegt. Zuerst soll es eine Abfrage des Spielmodi geben. Durch System.out.println angezeigt, dass das Spiel gestartet wurde. In der nächsten Zeile wird der „Spielmodus1: Gegen den Computer“ und eine Zeile darunter „Spielmodus 2: Gib ein eigenes Wort (Multiplayer)“ ausgegeben.

Damit es keine Fehlermeldung gibt, wird \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Um eine Auswahl zwischen Spielmodus 1 und 2 zu gewährleisten, haben wir eine switch-Anweisung, also eine Mehr-Wege-Verzweigung angelegt. Der Ausdruck „spielmodus“ lässt sich somit in case 1 und 2 untergliedern. Case 1 steht für den Spielmodus 1. Es wird angezeigt „Es wurde Spielmodus 1 gewählt“. Wir haben eine Auswahl an Wörtern hinterlegt und eine zufällige Zahl für die Wortauswahl. Wichtig hierbei ist es, die Wörterlänge als Maximum anzugeben, da \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Damit der Computer nun ein zufälliges Wort von den vorher angelegten Wörtern aussucht, haben wir eine random Methode angelegt.

Durch System.out.println(woerter[randomNumber] wird nun eines der hinterlegten Wörter angezeigt. Die Variable input haben wir auf das zufällig wewählte Wort gesetzt, weil \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

In Case zwei ist der Spieler selbst für die Erstellung eines zu ratenden Wortes verantwortlich. Wieder wird ausgegeben, in welchem Spielmodus man sich befindet, jedoch wird der Spieler aufgefordert ein Wort einzugeben. Die Variable wird auf das eingegebene Wort gesetzt.

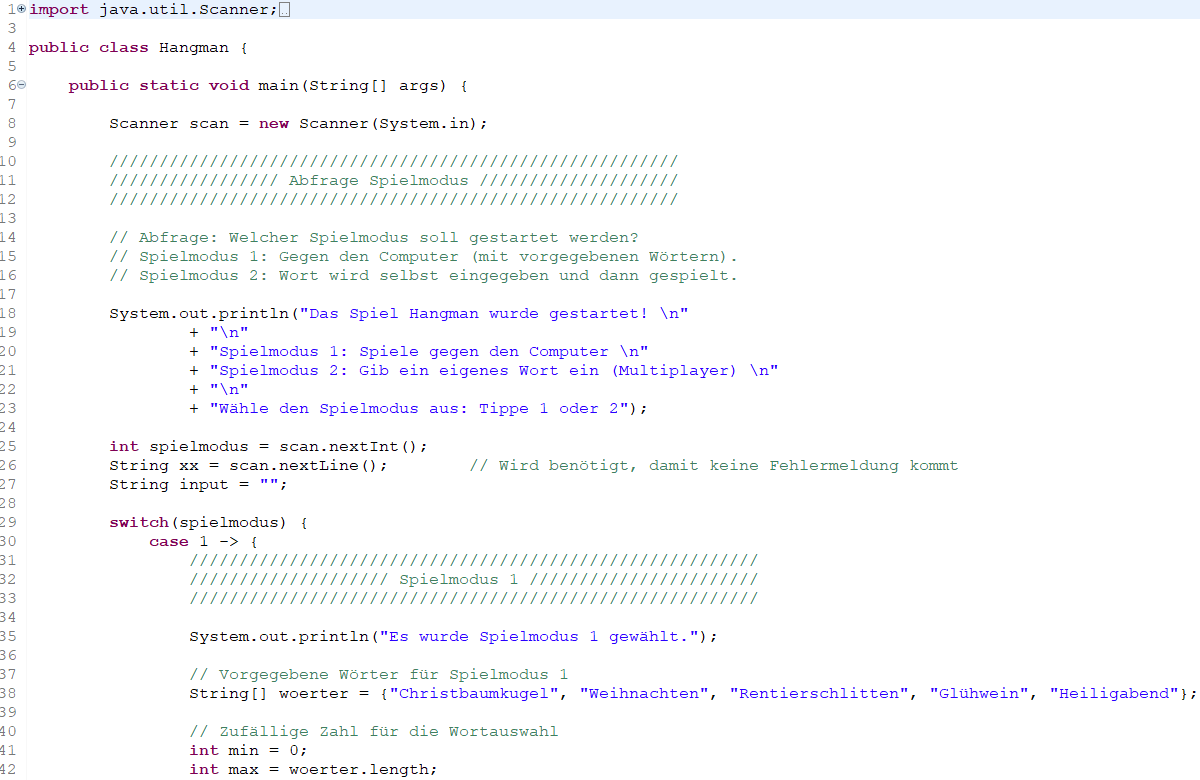
default -> System.out.println("Falsche Eingabe!")

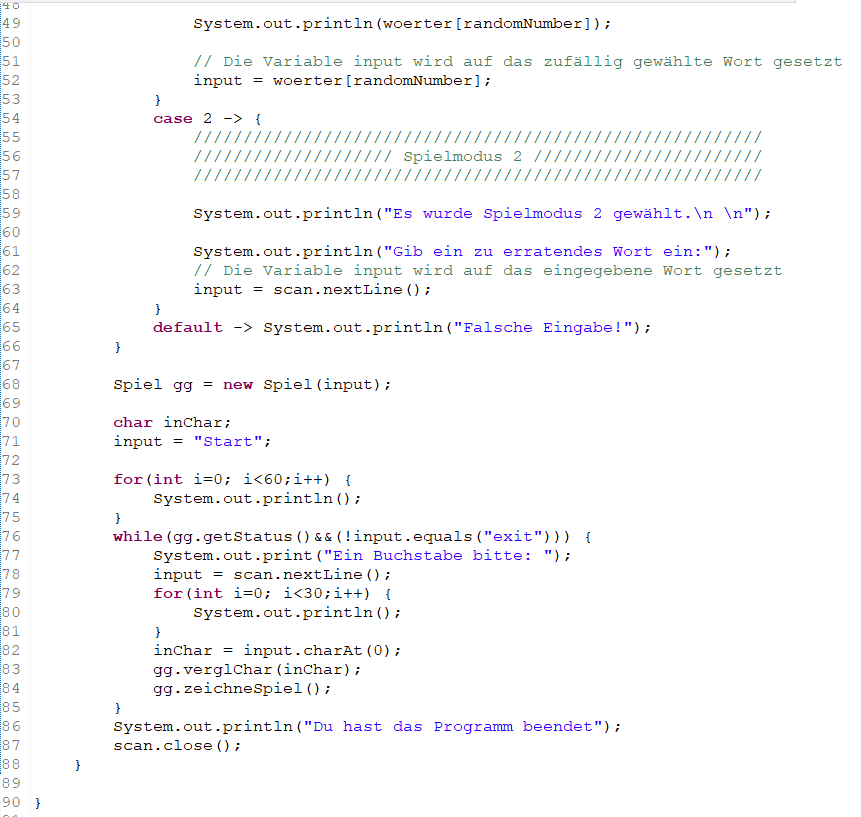
Spiel gg = new Spiel(input);

char inChar;

input = "Start";

Damit der Spieler immer weiter raten kann, haben wir eine for-Schleife angelegt.

****Zudem haben wir die abweisende while-Schleife angelegt, welche, so lange der Spieler nicht „exit“ schreibt, in Kraft tritt. Während der Spieler sich in dieser Schleife befindet, wird „Ein Buchstabe bitte“ beim Spieler angezeigt. Sollte der Spieler „exit schreiben, so wird „Du hast das Spiel beendet“ angezeigt und das Spiel wird durch „scan.close();“ geschlossen.



## 4.3 Spiel.java

1. **Zusammenarbeit**

Für den Start unseres Projektes haben wir uns zusammen Online getroffen. Hier haben wir Java/Eclipse mit GitHub verknüpft und jeder von uns hat einen eigenen Branch angelegt. Des Weiteren hat jeder erstmal über das Wochenende für sich Ideen gesammelt. Unser nächster Gruppentermin hat in Präsenz stattgefunden und wir haben uns alle zusammen und gemeinsam ausgetauscht. Hier haben wir auch festgelegt, wer an welcher Gruppenaufgabe mehr beteiligt sein möchte und wie unser weiteres Vorgehen erfolgt. Insgesamt hat unsere Gruppenarbeit gut funktioniert und wir kamen gut zurecht.