Logo

Description automatically generated

**Minigolf**

Miigolf

Shape, rectangle

Description automatically generated

**Projektdokumentation**

**Programmieren II - Praktikum**

im Studiengang „Angewandte Informatik und Soziale Medien“

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | vorgelegt von: | | |  |
| **Christoph Gretenkort**  Matrikelnummer: 2190167 | |  | **Evelyn Nolden**  Matrikelnummer: 2210050 | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | *ausgetreten:* |  | |
|  | *Lasse Hauschild*  *Matrikelnummer: 2210209* | | |  |

**Prüfer\*in**: Dr. Benedikt Lindenbeck

1. **Woche (25.03.2022)**

In der ersten Woche wurde sich im Team zusammengefunden und festgelegt, dass das zu programmierende Spiel **Minigolf**werden soll. ***(Lasse H. & Evelyn N.)***

1. A picture containing text, clipart

   Description automatically generated**Woche (01.04.2022)**

In dieser Woche wurde das Git-Repository angelegt

mit einem Textdokument der Teammitglieder.

***(Lasse H. & Evelyn N.)***

Ebenfalls wurden sich Gedanken über das Design des

Spiels gemacht ***(Lasse H. & Evelyn N.)*** sowie eine

dazugehörige erste Skizze angefertigt ***(Evelyn N.)***.

1. **Woche (08.04.2022)**

Shape, rectangle, square

Description automatically generated

Das Design wurde noch einmal überarbeitet,   
da ein neues Teammitglied ***(Christoph G.)*** zur Gruppe

dazu kam. Daher wurde auch eine weitere Skizze

angefertigt ***(Evelyn N.)***. Ebenso wurde über die

Architektur des Spiels gebrainstormt ***(alle Mitglieder)***

und gleichzeitig wurde ein vorläufiges UML-Diagramm

angefertigt ***(Christoph G.)***.

Diagram

Description automatically generated

1. **Woche (15.04.2022)**

Die Aufgaben wurden mithilfe des UML-Diagramms verteilt. ***(alle Mitglieder)*** Folgende erste Aufgabeverteilung ist entstanden:

|  |  |  |
| --- | --- | --- |
|  |  |  |

1. **Woche (22.04.2022)**

Die Logik des Balls wurde entwickelt. Die Idee ist, dass eine Linie gezeichnet werden soll und mithilfe des Start- und Endpunktes soll ein Vektor berechnet werden. Mithilfe des Vektors wird dann die Ballbewegungsrichtung vorgegeben. Die Geschwindigkeit des Balls soll abhängig von der Länge des Vektors sein.

Mit der Implementierung wurde in dieser Woche ebenfalls begonnen. MouseListener sowie MouseMotionListener wurden implementiert, um die Start- und Endposition der Computermaus abzufragen. Die mousePressed()-Methode ist dafür zuständig, die x- und y-Koordinate vom Startpunkt *(der späteren Linie)* an die Variablen startX und startY zuzuweisen. Die mouseDragged()-Methode übernimmt die gleiche Funktion für den Endpunkt und weist die Koordinaten an die Variablen endX sowie endY zu. ***(Evelyn N.)***

1. **Woche (29.04.2022)**

An der Logik des Balls wurde weitergearbeitet. Die Methode mouseReleased() wurde überschrieben und die Methode geschwReduzieren() erstellt.

A picture containing shape

Description automatically generatedmouseReleased() ist dafür zuständig, die Länge zweier Vektoren zu berechnen, um mit

dieser die Bewegungsrichtung sowie die Geschwindigkeit des Balls zu regulieren. Die

Länge der Vektoren ist abhängig von der Länge der gezeichneten Linie, welche in dieser Woche ebenfalls implementiert wurde. Somit wurde auch die Implementierung der grafischen Oberfläche in dieser Woche begonnen. Neben der zeichenbaren Linie wurde noch ein Image für den Ball erstellt und eingefügt, sodass der Ball ein bewegendes Bild ist. ***(Evelyn N.)***

1. **Woche (06.05.2022)**

Shape, square

Description automatically generatedIn dieser Woche wurde das Spielfeld grafisch umgesetzt.

Um dies zu erstellen, wurden die Methoden

rechteckErstellen() sowie rundesRechteckErstellen()

geschrieben. Diese werden dann in der überschriebenen

paint()-Methode aufgerufen, um das Spielfeld zu

zeichnen. Ebenfalls wurde das Bewegen des Balls umgesetzt

mithilfe eines Timers. Um das Herausbewegen des Balls zu

verhindern, wurde ein ActionListener implementiert.

Mithilfe der überschriebenen actionPerformed()-Methode wird abgefragt, ob der Ball den Rand des Fensters erreicht und falls ja, dann wird die Bewegungsrichtung des Balls umgekehrt. ***(Evelyn N.)***

1. **Woche (13.05.2022)**

Wir haben in dieser Woche die Logik der Banden implementiert, sodass der Ball sich nur innerhalb der Banden bewegen kann und außerhalb von diesen abprallt beziehungsweise die Geschwindigkeit invertiert wird. Dafür wurde die Methode bounce() implementiert, um dies sicherzustellen. Diese wurde zunächst wie folgt implementiert: Es wurde bestimmt, in welche Richtung sich der Ball aus dem Spielfeld bewegt und an dieser Achse gespiegelt. Dies war zunächst kein Problem, musste jedoch vor dem Hintergrund der mehrteiligen Bahn 3 neu durchdacht werden. Eine Erklärung dafür folgt in der nächsten Woche. ***(Christoph G.)***

1. **Woche (20.05.2022)**

Die Logik der Banden wurde weiter umgesetzt. Wie in der letzten Woche angedeutet, musste diese weiter angepasst werden. Es ist folglich die Methode getVarianz() dazu gekommen. Diese berechnet aus der aktuellen Ballposition den Abstand zu allen Grenzen der aktuellen Bahn mit Berücksichtigung mehrerer Teilstücke *(das Attribut second in der Banden- Klasse)*. Dabei war es besonders schwer das zweite Bahnstück bespielbar zu machen. Dies funktionierte am Ende dieser Woche jedoch.

Außerdem wurde das BandenCounter-Attribut hinzugefügt. Dies ist nötig, um die aktuelle Bahn zu bestimmen und funktioniert analog zum LochCounter. Diese sind separat voneinander, da das Wasserstück der zweiten Bahn sonst keine Berücksichtigung finden würde. Die Banden und Löcher sind nun in je einer ArrayList zu finden, somit ist eine dynamische Anzahl an Banden ab jetzt möglich. Wir haben uns für eine ArrayList entschieden, da hier ein Zugriff auf die Banden mithilfe des entsprechenden Counters möglich ist aber dennoch eine variable Länge / Anzahl an Bahnen. ***(Christoph G.)***

1. **Woche (27.05.2022)**

Hier wurde die Logik der Banden weitergeführt und funktioniert nun mit seltenen, nicht-deterministischen Ausnahmen, bei welchen wir uns dazu entschieden haben, mit ihnen zu rechnen, da ein Lösen der Bugs uns in der gegebenen Zeit leider nicht möglich ist.

Die Methode nextCourse() wurde implementiert, um einen sicheren Ablauf des Spiels zu gewährleisten. Diese Methode teleportiert den aktuellen Ball an den Startpunkt der nächsten Bahn und zählt die entsprechenden counter hoch. ***(Christoph G.)***

1. **Woche (03.06.2022)**

Diese Woche wurde dazu genutzt, um Teile des Codes aus der MyPanel-Klasse auszulagern in andere. Somit wurden die folgende Klassen für das Spiel neu erstellt: Ball, Game und Player. ***(Evelyn N.)***

1. **Woche (10.06.2022)**

Da die Zweispielerkomponente bisher noch nicht angefangen wurde, wurden die Grundbausteine für den Zweispielermodus begonnen zu implementieren. Ebenso wurde in dieser Woche ein Endscreen sowie ein Scoreboard grafisch umgesetzt. ***(Evelyn N.)***

Für den Zweispielermodus wurden zwei Variablen zum Zählen der Schläge in der Klasse MyPanel erstellt: schlagzähler1 und schlagzähler2. Mithilfe der mouseReleased()-Methode werden die beiden Integer vorübergehend nach jedem Schlag hochgezählt. Mithilfe dessen wurde dann ein Scoreboard gebaut, in welchem sich die Schlagzahlen jeweils aktualisieren. Das Scoreboard besteht aus einer Liste mit zwei Einträgen.

Ebenso entstand in dieser Woche ein Endscreen, welcher

den\*die Gewinner\*in ausgeben soll. Mithilfe der schlagzähler-

A screenshot of a computer

Description automatically generated with medium confidenceVariablen soll am Ende überprüft werden, welcher Spieler

weniger Schläge gebraucht hat und dieser soll dann auf dem

Bildschirm angegeben werden.

1. **Woche (17.06.2022)**

Die 13. Woche wurde dafür genutzt, um das Scoreboard sowie den Endscreen zu überarbeiten. ***(Evelyn N.)*** Ebenso wurde der Zweispielermodus vollendet. ***(Christoph G.)***

Das Scoreboard besteht nun nicht mehr aus einer Liste, sondern ist ein Rechteck, welches gezeichnet wird und welches mittels zwei Strings die jeweils aktuelle Schlagzahl des Spielers anzeigt. Dem Endscreen wurde noch eine Konfetti-Animation hinzugefügt.

Graphical user interface, text, application

Description automatically generatedA picture containing text, device, gauge

Description automatically generated

Der Zweispielermodus wird über die Klasse Game realisiert. Diese verfügt über zwei elementare Attribute hierfür: PlayerPrimary und PlayerSecondary. Der entsprechend aktive Spieler ist nun über die Methode getAktPlayer() erreichbar. Um diese zu unterscheiden, beinhält die Player-Klasse nun ein weiteres Attribut aktiverSpieler, um diesen eindeutig zu bestimmen.

Außerdem hat sich in dieser Woche das Team verkleinert, da sich ***Lasse Hauschild*** von diesem Projekt abgemeldet hat. Die Gruppe besteht ab diesem Zeitpunkt nun offiziell aus den Teammitgliedern ***Christoph Gretenkort*** und ***Evelyn Nolden***.

1. **Woche (24.06.2022)**

Zu Anfang dieser Woche wurde nochmals ausprobiert, eine RMI-Funktion in das Projekt einzubinden. Diese Funktionalität hat sich jedoch durch das Nicht-Funktionieren des rmic-tools als deutlich schwieriger herausgestellt als erwartet. Vor dem Hintergrund der nahen Abgabe sowie der Klausurenphase, wurde entschieden, diese Funktionalität auszulassen und sich auf den bereits funktionierenden Teil des Programms zu konzentrieren, um diesen einheitlicher zu schreiben, alte auskommentierte Code-Snippets zu entfernen und am Feinschliff des Codes zu arbeiten.

Außerdem fangen wir in dieser Woche bereits die Präsentationvorbereitung an, da dafür in der folgenden Woche nicht mehr viel Zeit bleibt. Auch hier sind die parallelen Klausuren Grund dafür. ***(Christoph G. und Evelyn N.)***