# Einleitung

Dies ist die Dokumentation zu unserem Java Schulprojekt. Wir haben das Projekt „Mensch Ärgere Dich nicht“ gewählt, da wir ein einfaches Spiel in Java umsetzen wollten und einige Teammitglieder in unserem Projekt schon ein wenig Erfahrung mit der Programmierung eines Brettspieles hatten.

# Projektantrag

Hier folgt der E-Mail Antrag zu unserem Java Projekt.

Hallo Herr Peschel,

wir (Team: Philip Langer, David John, Chris Deter) würden ganz gerne eine Mensch ärgere dich nicht Adaption in Java programmieren.

Es sollen 4 spielbare Parteien (Rot, Blau, Grün, Gelb) im Spielfeld mit jeweils 4 Figuren zur Verfügung stehen.

Zusätzlich gibt es einen Optionsfenster und einen Würfel.

Nachdem man eine Zahl gewürfelt hat, soll man per Klick auf eine Figur der Farbe, die aktuell am Zug ist einen Zug auslösen.

Die Figur soll entsprechend der gewürfelten Zahl auf dem Spielfeld weiter ziehen.

Dabei soll sie sowohl eigene als auch gegnerische Figuren erkennen und die gegnerischen gegebenenfalls herausschmeißen und so an den Start zurücksetzten.

Je nach Zeit um Umfang des Projektes werden eventuell weitere Funktionen implementiert (weitere Regeln, KI, ...).

lg Chris

# Beschreibung des Projektziels

Das Projektziel ist eine funktionierende graphische Umsetzung von Mensch Ärgere Dich Nicht. Alle Farben sollen auf dem Spielfeld spielbar sein. Dazu sollen die wichtigsten Regeln ordnungsgemäß funktionieren.

Zu dem wichtigsten Regeln gehören:

* Wer eine Sechs würfelt, muss eine eigene Spielfigur aus der Startposition heraus auf sein Startfeld des Spielfeldes stellen
* Anschließend wird erneut gewürfelt und mit der Figur entsprechend viele Felder vorgerückt
* Das Startfeld muss so bald wie möglich freigemacht werden
* Bei einer 6 darf erneut gewürfelt werden
* Gegnerische Spielfiguren können aus dem Spiel geschmissen werden

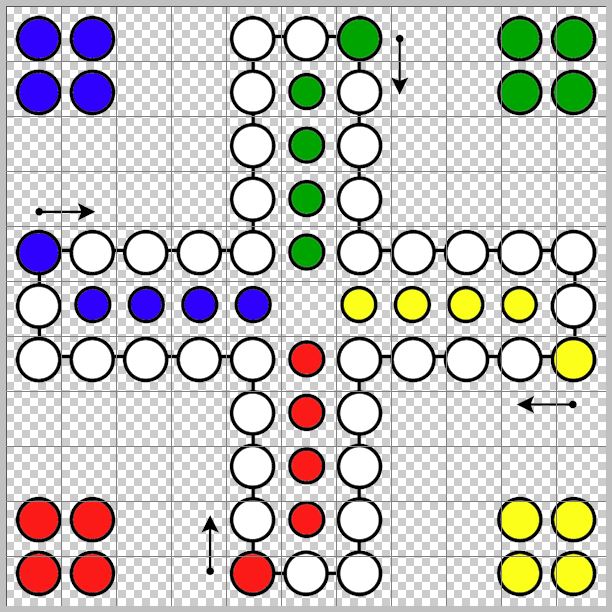
Dazu gibt es einige optionale Regeln, die sich bei der Umsetzung auch als Optionale Ziele gesetzt wurden:

* Wenn sich keine Figur im Spielfeld befindet, darf 3-mal versucht werden, eine 6 zu würfeln.
* „Schlagzwang“ – Wenn die Möglichkeit besteht, muss eine Spielfigur aus dem Spiel geschmissen werden, ansonsten wird die Spielfigur selber zurück auf ihr Startfeld gesetzt
* „Barrieren“ – Wenn zwei Figuren derselben Farbe auf einem Feld stehen, wird eine Barriere errichtet, die von anderen Spielern nicht überschritten werden kann.

Zusätzlich haben wir uns als optionales Ziel die Entwicklung einer KI vorgenommen.

# Projektanalyse

Das Spielfeld von „Mensch Ärgere Dich Nicht“ enthält 40 Felder auf dem sich die Spielfiguren gemeinsam bewegen können und noch mal 8 Felder die für jede Farbe individuell sind.



Man kann das Spielfeld im Wesentlichen in einem 11 x 11 Blöcke großes Spielfeld zerlegen.  
Wenn man immer von dem Mittelpunkt des Spieles ausgeht und die Gesamtgröße des Spielfeldes durch 11 teilt, bekommt man die Größe der einzelnen Spielquadrate.  
So ist es relativ einfach die einzelnen Punkte auf denen die Spielfiguren sich bewegen können zu bestimmen.

Jede Spielfigur hat ihren individuellen Startpunkt. Zu diesem Startpunkt kehrt sie nach einem Rauswurf immer wieder zurück.  
Danach kann sie die 40 Schritte zum Ziel laufen. Im Ziel kann sie weitere 4 Schritte machen, diese Schritte sind aber auch für jede Farbe individuell. Deswegen war es am naheliegensten jeder Figur einen Wert zuzuweisen. Sie beginnt auf dem Feld 0 und bewegt sich dann von Feld1 bis Feld 44.

Jede Farbe hat dabei einen um etwa 10 Schritte versetzten Startpunkt, auf dem sie das Spielfeld betritt. Nachdem der Würfel geworfen wurde und die Spielfigur geklickt wird müssen diverse Regeln abgeprüft werden.

Zu dem wichtigsten Regeln gehören:

* Wer eine Sechs würfelt, muss eine eigene Spielfigur aus der Startposition heraus auf sein Startfeld des Spielfeldes stellen
* Anschließend wird erneut gewürfelt und mit der Figur entsprechend viele Felder vorgerückt
* Das Startfeld muss so bald wie möglich freigemacht werden
* Bei einer 6 darf erneut gewürfelt werden
* Gegnerische Spielfiguren können aus dem Spiel geschmissen werden

Die optionalen Regeln:

* Wenn sich keine Figur im Spielfeld befindet, darf 3-mal versucht werden, eine 6 zu würfeln.
* „Schlagzwang“ – Wenn die Möglichkeit besteht, muss eine Spielfigur aus dem Spiel geschmissen werden, ansonsten wird die Spielfigur selber zurück auf ihr Startfeld gesetzt
* „Barrieren“ – Wenn zwei Figuren derselben Farbe auf einem Feld stehen, wird eine Barriere errichtet, die von anderen Spielern nicht überschritten werden kann.

Nachdem die Regeln abgefragt wurden, müssen die Spielfiguren auf dem Spielfeld entsprechend der gewürfelten Zahl gezogen werden bis alle Spielfiguren das Ziel erreicht haben. Im Ziel ist es zudem nicht erlaubt, dass die Felder doppelt belegt sind.

Das Spiel wird gewonnen wenn eine Farbe alle ihre Figuren im Ziel hat.

Der Würfel ist eine Zufallszahl von 1 – 6.

# Projektplanung

## Zeitraum

Das Projekt wurde vom 01.02.2013 bis zum 12.04.2013 entwickelt.

## Dokumentation der Aufgabenverteilung innerhalb der Gruppe

|  |  |
| --- | --- |
| Phillip Langer | Terminmanagement, Tests, Quellcodeerstellung, Graphische Gestaltung |
| Chris Deter | Konzeption, Aufgabenmanagement, Quellcodedokumentation, Quellcodeerstellung, Quellcodemanagement, Dokumentation |
| David John | Idee, Graphische Dokumentation, Quellcodeerstellung, Tests |

Terminmanagement: Managt die Treffen und Überprüft den Fortschritt des Projektes  
Tests: Ist zuständig für die ausführlichen Test der Software  
Quellcodeerstellung: Ist beteiligt bei der Entwicklung einzelner Quellcodeabschnitte  
Graphische Gestaltung: Erstellung der verwendeten

(iwann mal fertig schreiben)

## Konzept

Das Projekt soll objektorientiert Umgesetzt werden. Dafür wurde das Spiel in seine Objekte zerlegt.

|  |  |
| --- | --- |
| Figur | Diese Klasse bildet die 16 Spielfiguren auf dem Spielfeld ab. |
| Spieler | Diese Klasse bildet die einzelnen Farben der Spieler ab. |
| Würfel | Diese Klasse bildet den Würfel ab. |
| Window | Diese Klasse bildet das Spielfeld mit den Spielfiguren und den Würfel ab und nimmt die Usereingaben entgegen. |
| Positions | Diese Klasse ist ein Bindeglied zwischen der Game Klasse und der Window Klasse und managt die Bewegungen der Figuren. |
| Game | Diese Klasse dient zur Organisation des Spiels, hier werden die Usereingaben verarbeitet und zurück an das Spielfenster gegeben. |

Wie in der Analyse beschrieben, hat jede Spielfigur einen Wert von 0 bis 44. Die Werte von 1 – 40 werden allerdings für das Spielfeld je nach Farbe um 10 Felder versetzt.  
So steht die Rote Spielfigur mit dem Wert 1 auf dem Feld 1 und die Blaue Spielfigur mit dem Wert 1 auf dem Spielfeld 11.

# Projektumsetzung

## UML-Klassendiagramme

## Entwicklung des Programmes

Das Programm wurde in Eclipse geschrieben. Das Projekt wurde über Google Drive mit allen Projektmitgliedern synchronisiert, sodass jeder die aktuellste Version des Projektes zur Verfügung hatte.

Jedes Teammitglied hat einzelne Funktionen des Quellcodes als Aufgabe bekommen und diese programmiert.

## Darstellung des Ablaufs einer Methode, die Konstrollstrukturen enthält, als Struktogramm

## Erklärung auftretender Probleme und Erläuterung eventueller Lösungswege

Als erstes trat das Problem auf, dass der Zugriff zwischen einzelnen Objekten nur in eine Richtung funktionierte und in der Form auf der die einzelnen Bildboxen mit den Figuren bewegt wurden, konnte nur auf die angeklickte Figur zugegriffen werden. So wurde eine aus dem spiel geschmissene Figur nicht bewegt, obwohl sie vom Spielfeld auf das Startfeld gesetzt hätte werden müssen. Dies wurde durch ein zweites Objekt gelöst, in dem jedes Objekt welches in der Form bewegt werden sollte referenziert wurde. Nachdem dem Verwaltungsobjekt also alle Objekte der Form bekannt waren, konnten diese einzeln angesprochen und bewegt werden. Dadurch wurde es möglich Figuren zu bewegen, die nicht angeklickt wurden.

Beim Erstellen des Regelmechanismus trat häufig ein Verhalten beim Zug einer Figur auf, dass durch die Komplexität und die Anzahl der Abfragen nicht mehr nachvollzogen werden konnte.  
Wir lösten das Problem, in dem wir die einzelnen Quellcodeausschnitte mit einer Konsolenausgabe versahen, die genau den durchgeführten Schritt erklärte. So konnten wir sehen, welche Regeln in der Regelabfrage griffen und so die fehlerhafte Regel identifizieren, die das seltsame Verhalten der Spielfigur verursachte.

JLabel vs Zeilenumbrüche

## Programmablauf (Quellcode inklusive Erläuterung)

# Soll-Ist-Vergleich

Das wesentliche Projektziel wurde erreicht, auch eins der optionalen Ziele konnte umgesetzt werden.

Bei den anderen beiden Regeln reichte die Zeit nicht für eine funktionierende Umsetzung, da diese sich als deutlich komplexer als geplant herausstellten. Zudem wurde die Umsetzung einer KI leider zeitlich nicht geschafft, obwohl ein Konzept für die Erstellung einer KI vorhanden war und einige Funktionen auf die Wiederverwendung in der KI Funktion vorbereitet wurden.

# Fazit

Das Projektziel wurde wie geplant erreicht. Allerdings wurden einige optionale Ziele nicht umgesetzt, da sich die Entwicklung des Projektes deutlich umfangreicher als zuerst vermutet herausstellte und deshalb Zeit eingespart werden musste.

# Quellenangaben

[http://de.wikipedia.org/wiki/Mensch\_ärgere\_Dich\_nicht](http://de.wikipedia.org/wiki/Mensch_%C3%A4rgere_Dich_nicht)

<http://spielfibel.de/pics/pachisi-4er-blanco.jpg>

<http://www.iconarchive.com/show/fast-icon-users-icons-by-fasticon/user-icon.html>