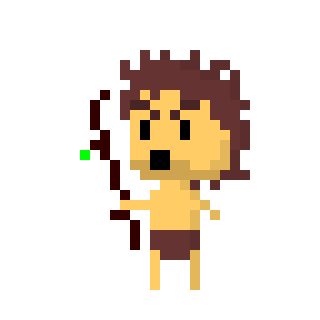
|  |  |
| --- | --- |
|  | **2015** |
|  | CavemenArena  undevined |

|  |
| --- |
| **[Dokumentation]** |
|  |



Inhalt

[Projektplanung 2](#_Toc417985196)

[Alter Projektplanung 2](#_Toc417985197)

[Analyse 3](#_Toc417985198)

[Plattform 3](#_Toc417985199)

[Entwicklungsumgebung 3](#_Toc417985200)

[Eclipse mit Android Development Tools 3](#_Toc417985201)

[Android Studio 4](#_Toc417985202)

[Vergleich 4](#_Toc417985203)

[Realisierung der Gruppenarbeit 5](#_Toc417985204)

[Use-Case-Diagramm 5](#_Toc417985205)

[Hauptmenü 5](#_Toc417985206)

[HowTo öffnen 5](#_Toc417985207)

[Spiel starten 5](#_Toc417985208)

[Schwierigkeitsstufe wählen 5](#_Toc417985209)

[Spiel 5](#_Toc417985210)

[Abwehren 5](#_Toc417985211)

[Design 6](#_Toc417985212)

[Klassendiagram 6](#_Toc417985213)

[Ablaufdiagramm 7](#_Toc417985214)

[Abfragen der Schärfe des Stocks 7](#_Toc417985215)

[Spieler 1: Angriff, Spieler 2: Stock schärfen 8](#_Toc417985216)

[Spieler 1: Angriff, Spieler 2: Angriff 8](#_Toc417985217)

[Implementierung 9](#_Toc417985218)

[SimpleBrain 9](#_Toc417985219)

[Layout 11](#_Toc417985220)

[Tests 12](#_Toc417985221)

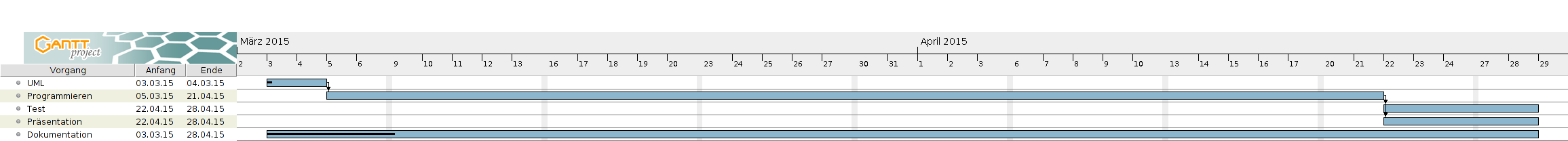
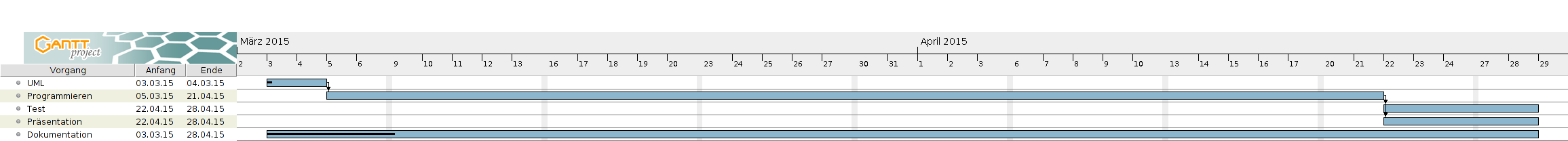
[Black Box 12](#_Toc417985222)

[Stick 12](#_Toc417985223)

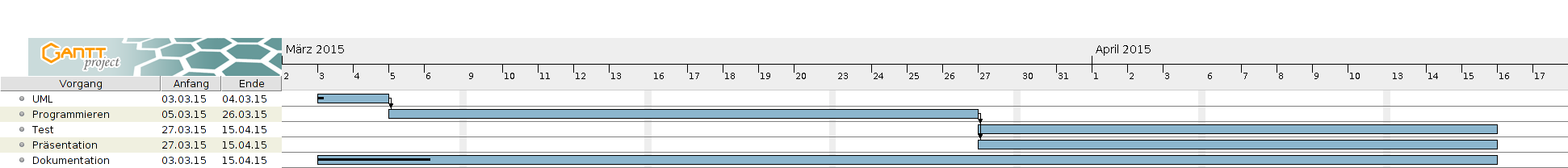
[Actions 12](#_Toc417985224)

[Quellen 13](#_Toc417985225)

# Projektplanung



## Alter Projektplanung



Aufgrund von Problemen mit der Entwicklungsumgebung und fehlenden Resourcen, war die alte Zeitplanung nicht zu halten, daher wurde der Plan korrigiert.

# Analyse

## Plattform

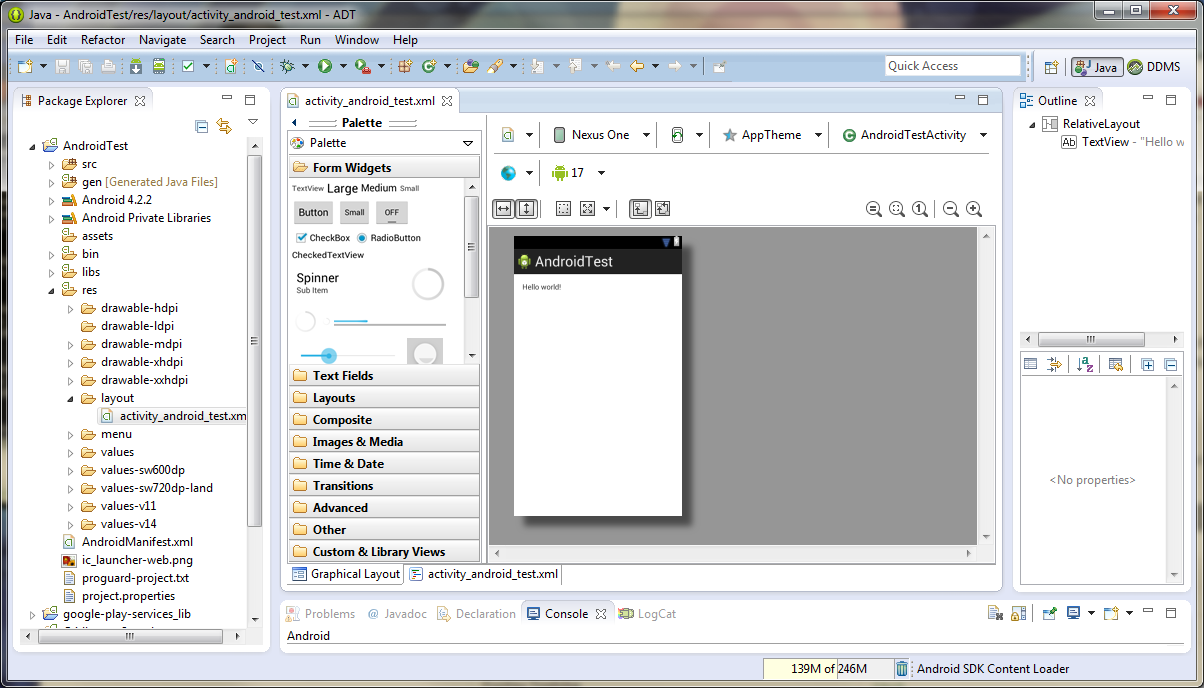
Um möglichst viele Benutzer zu erreichen haben wir die am weitesten verbreitete Plattform gewählt. Aus der Statistik von 2014 ist dies klar mit über 80 % **Android**.

Quelle: Quellverweis 1

## Entwicklungsumgebung

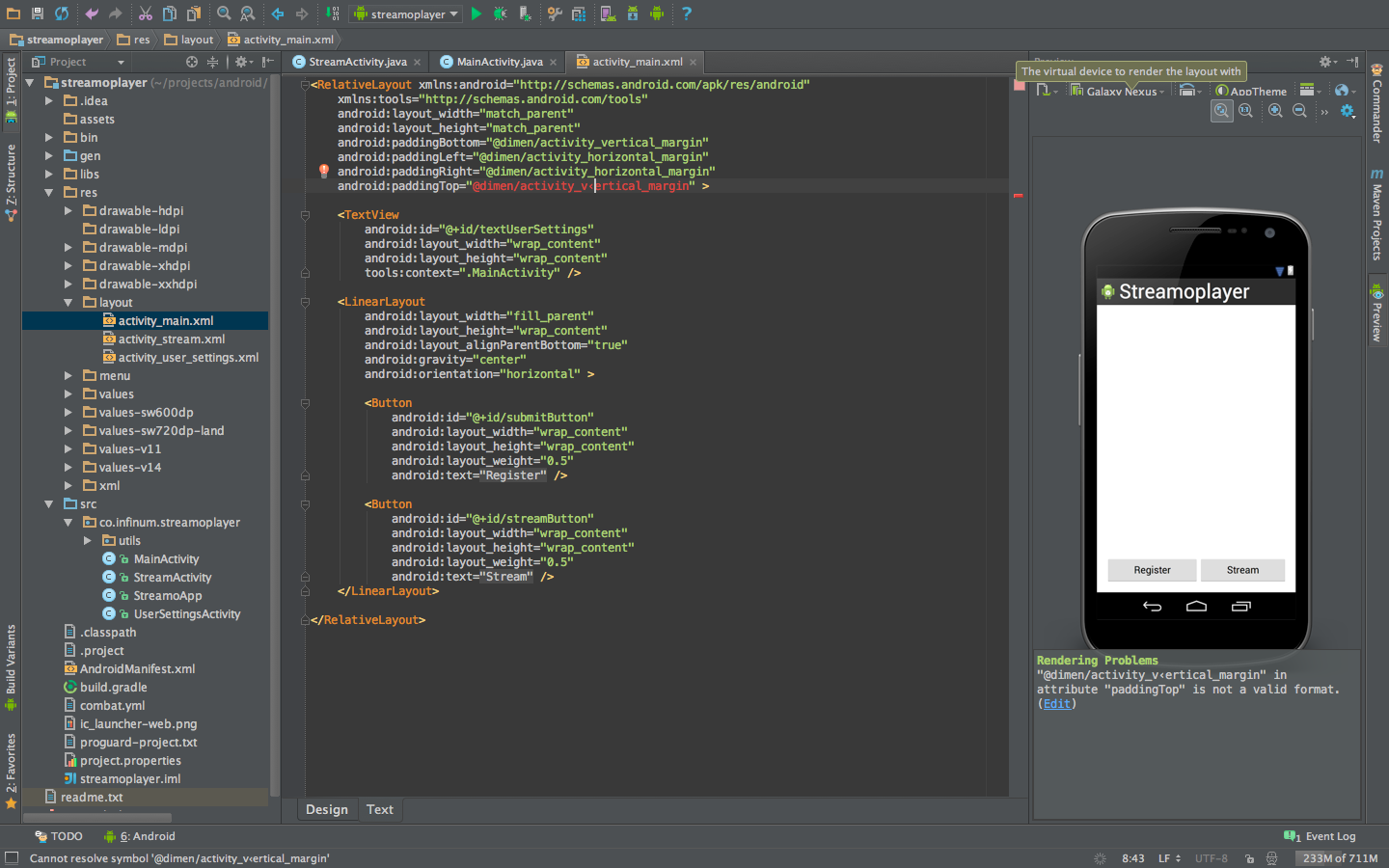
Eine anständige Entwicklungsumgebung ist maßgeblich für den Erfolg einer Anwendung. Im folgenden sind Entwicklungsumgebungen für Android aufgeführt, die wir bewertet haben um die für uns am besten passende wählen zu können.

### Eclipse mit Android Development Tools



Quelle: Quellverweis 2

### Android Studio



Quelle: Quellverweis 3

### Vergleich

|  |  |  |
| --- | --- | --- |
| Funktion / Kriterium | Android Studio | Eclipse |
| Geschwindigkeit | Gut | Mittel |
| Stabilität | Gut | Schlecht |
| Übersichtlichkeit | Gut | Mittel |

Aus den Tests beider Entwicklungsumgebungen hat sich klar Android Studio als das Mittel der Wahl dargestellt.

Eclipse hat klar durch Probleme geglänzt. Wurde im Designer z.B. ein Button zur Oberfläche hinzugefügt oder das Projekt gespeichert und erneut geöffnet, war es möglich, dass sich die Anwendung nicht mehr erzeugen hat lassen.

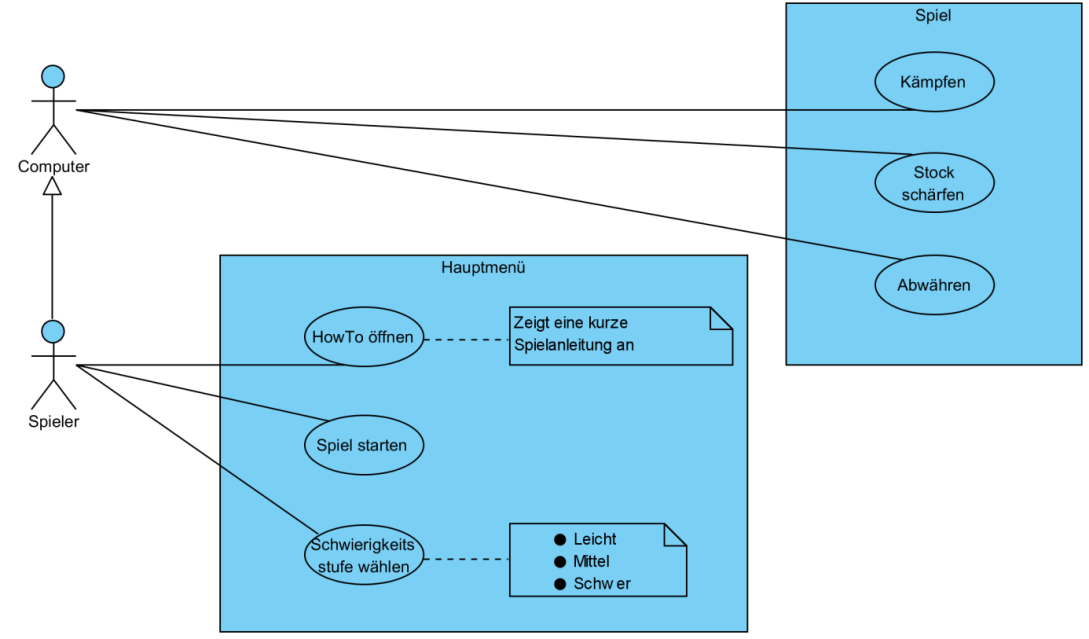
Das für Android angepasste Android Studio hatte klare Vorteile, da es kein Allrounder, sondern ein genau für ein Produkt angepasstes Stück Software ist. Performance und Stabilität waren von Google frei Haus geliefert wurden.

Letztendliche fiel unsere Wahl gegen das Urgestein Eclipse, da wir uns mit der Entwicklung unserer Anwendung und nicht mit den Problemen von Eclipse befassen wollten.

## Realisierung der Gruppenarbeit

Um gemeinsam an einem Projekt arbeiten zu können haben wir GIT eingesetzt, da dort viele Funktionen bereitstehen, wie z.B. eine Versionierung.

## Use-Case-Diagramm



## Hauptmenü

### HowTo öffnen

Es wird eine neue Seite geöffnet, in der mit Text eine einfache Anleitung angezeigt wird.

### Spiel starten

Sobald der Benutzer das spiel startet, wird er nach der Schwierigkeitsstufe gefragt.

### Schwierigkeitsstufe wählen

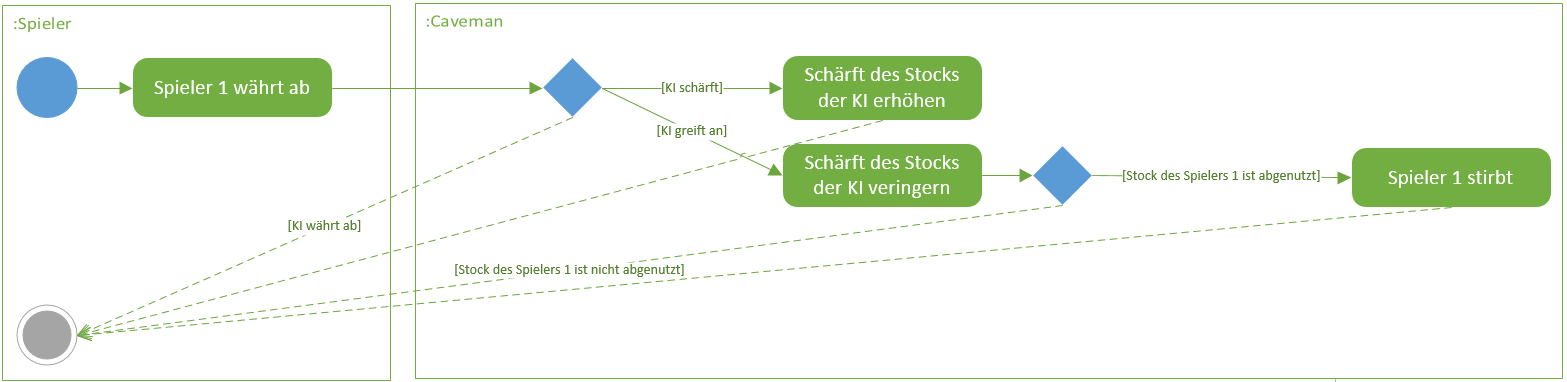
Benutzer kann die Schwierigkeit wählen, in der er das Spiel spielen möchte.

**Verfügbare Schwierigkeitsstufen:**

* Leicht
* Mittel
* Schwer

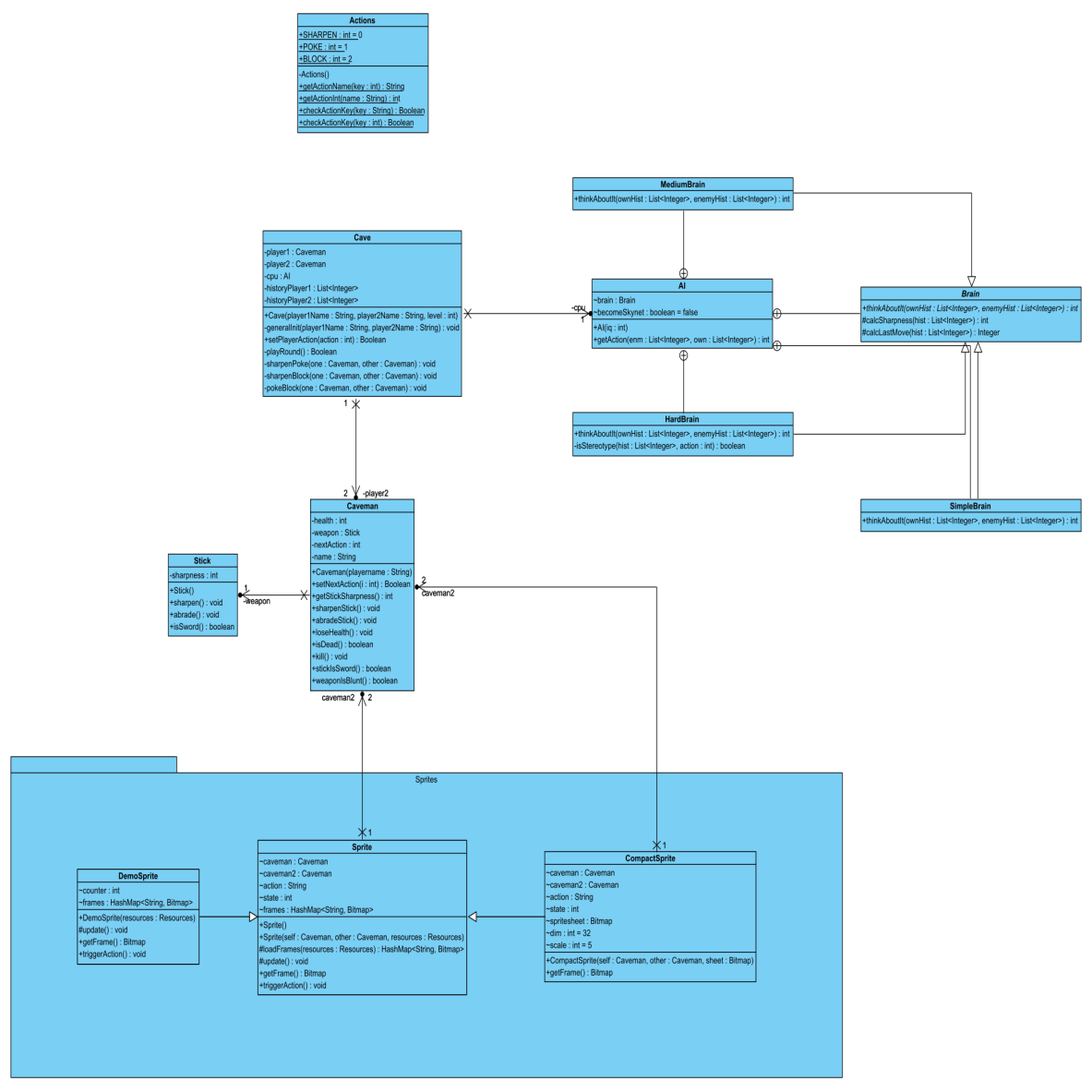
## Spiel

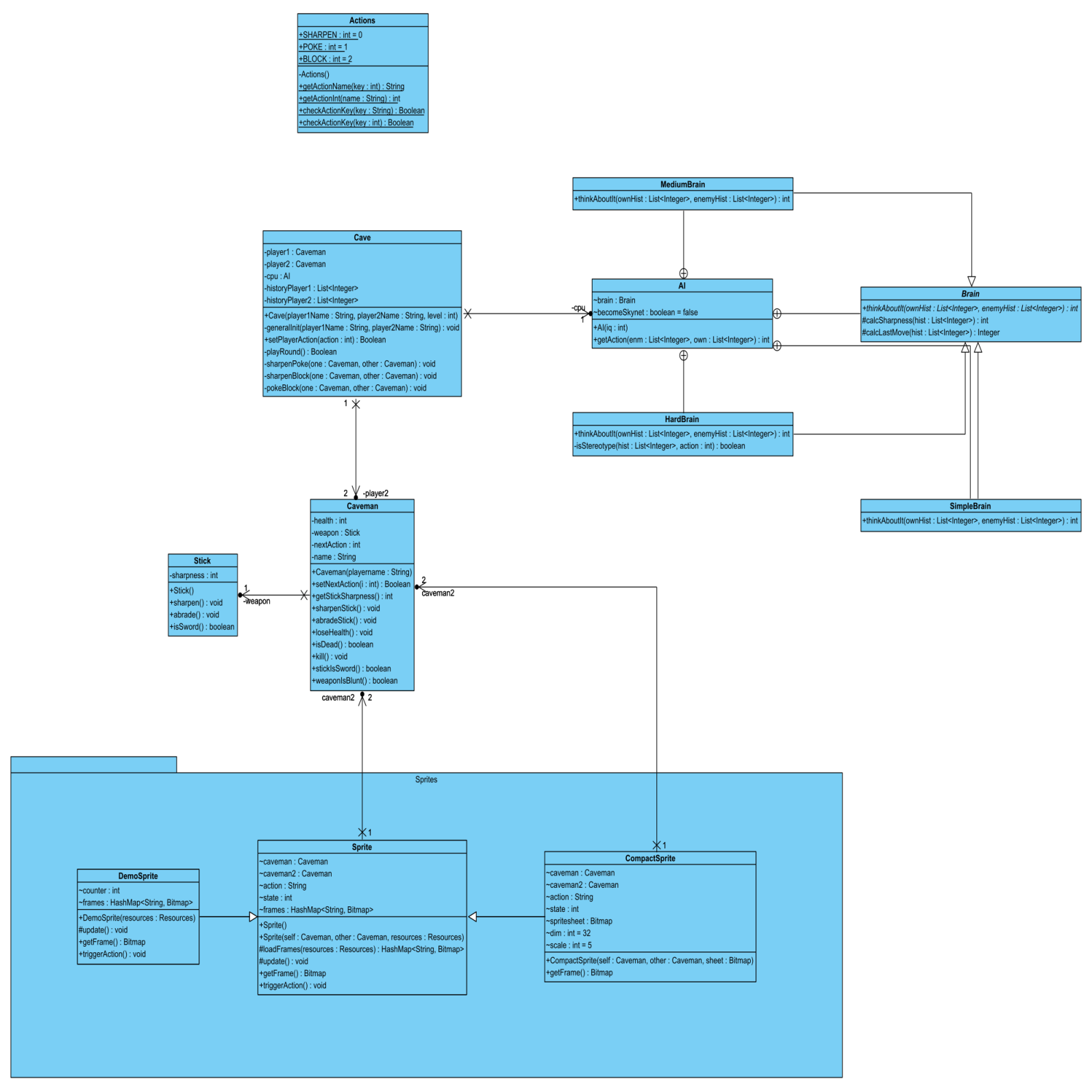
### Abwehren



# Design

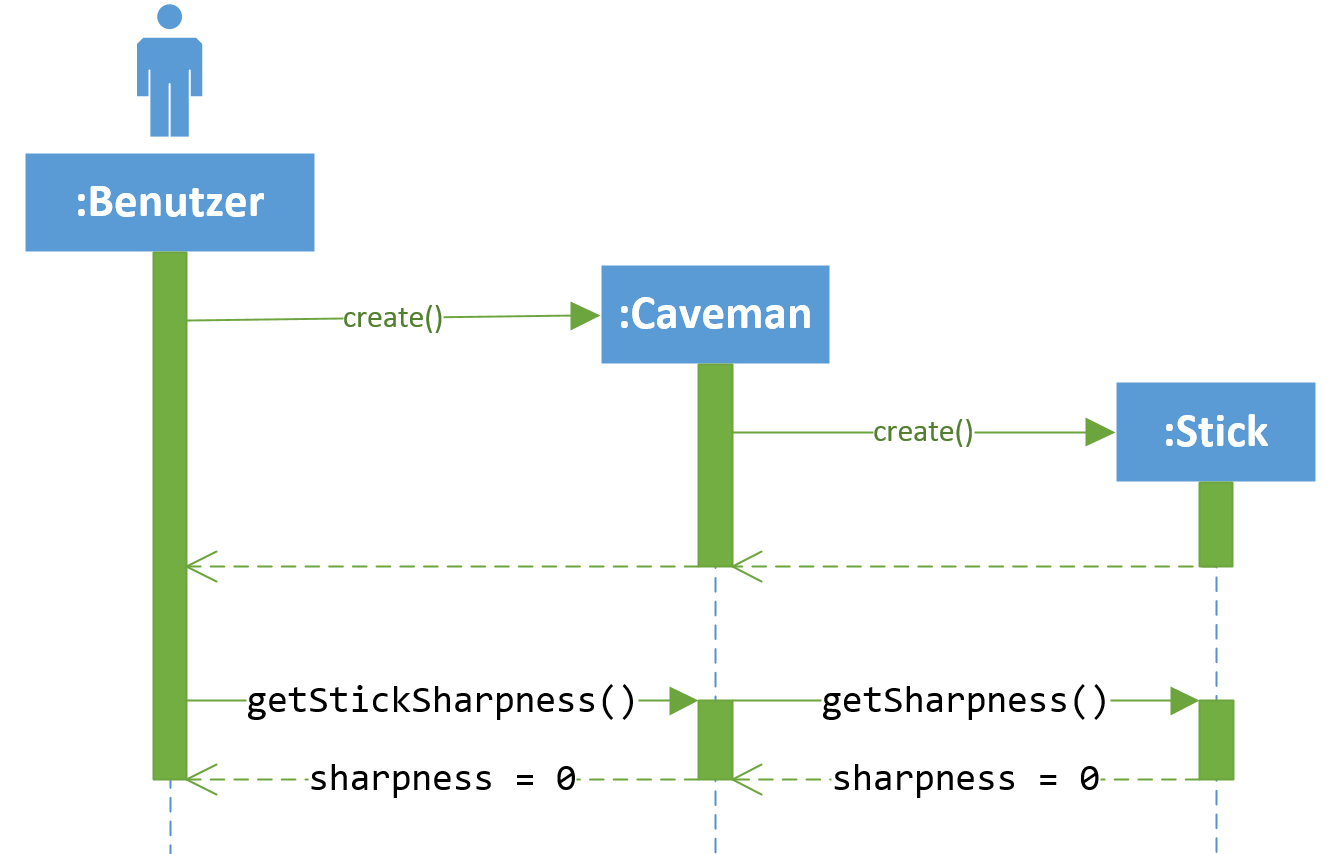
## Klassendiagram





## Ablaufdiagramm

### Abfragen der Schärfe des Stocks



Abfrage der Schärfe des Stock, die direkt nach dem erzeugen der Caveman-Instanz geliefert wird.

public class Caveman  
{  
 private Stick weapon;

/\*\*

\* Ermittelt die Schärfe der Waffe

\*

\* @return int Schärfe der Waffe

\*/

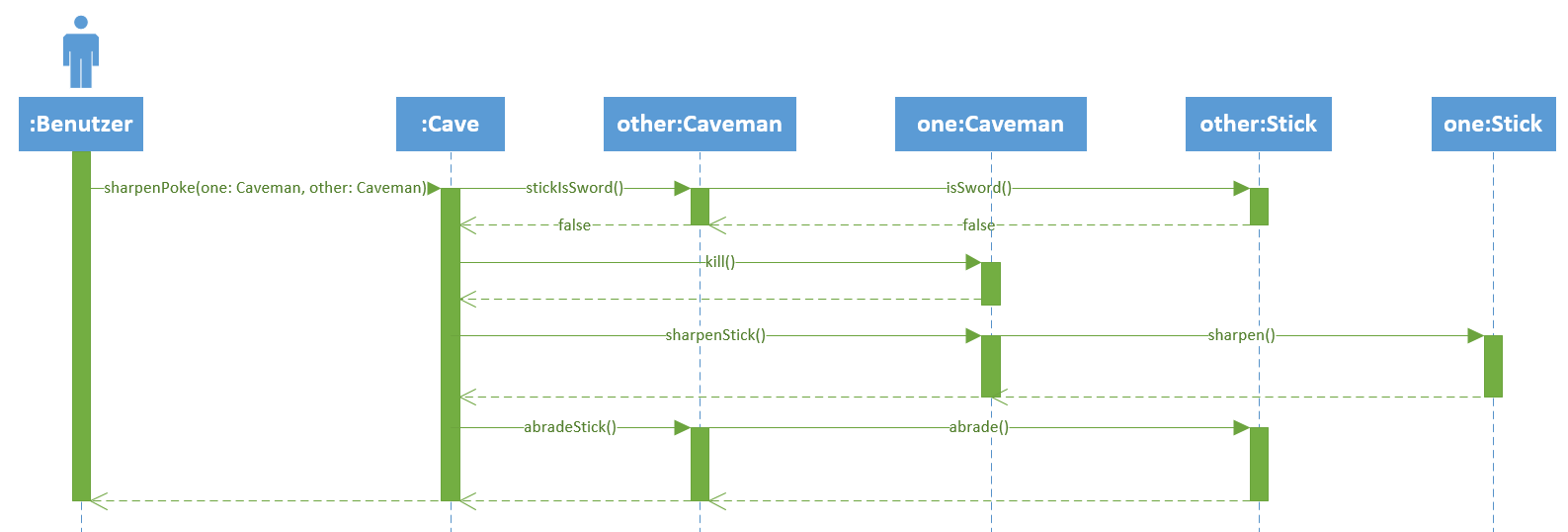
public int getStickSharpness() {

return this.weapon.getSharpness();

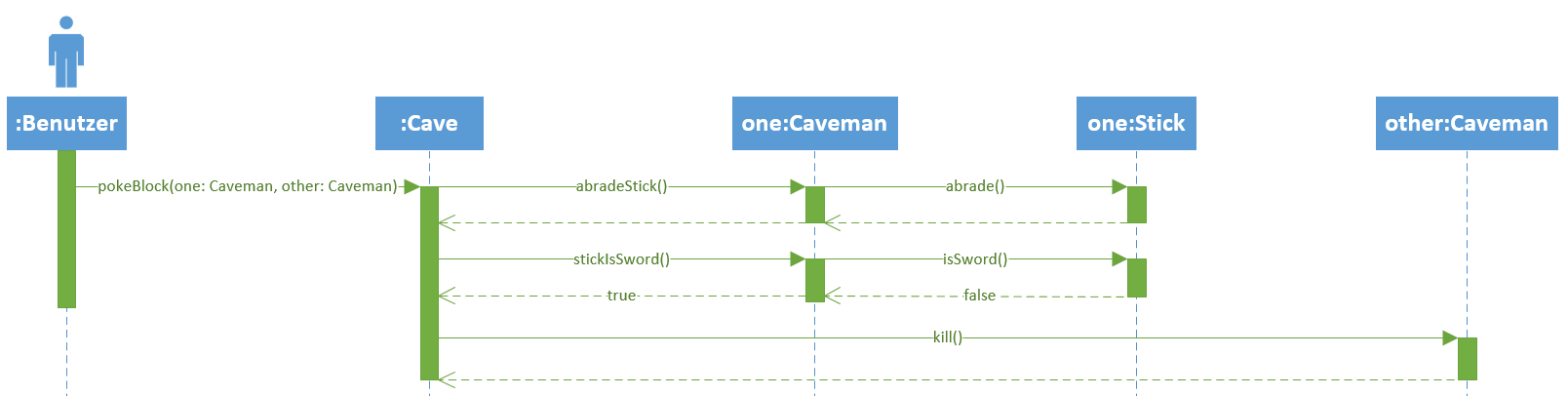
}

}

### Spieler 1: Angriff, Spieler 2: Stock schärfen

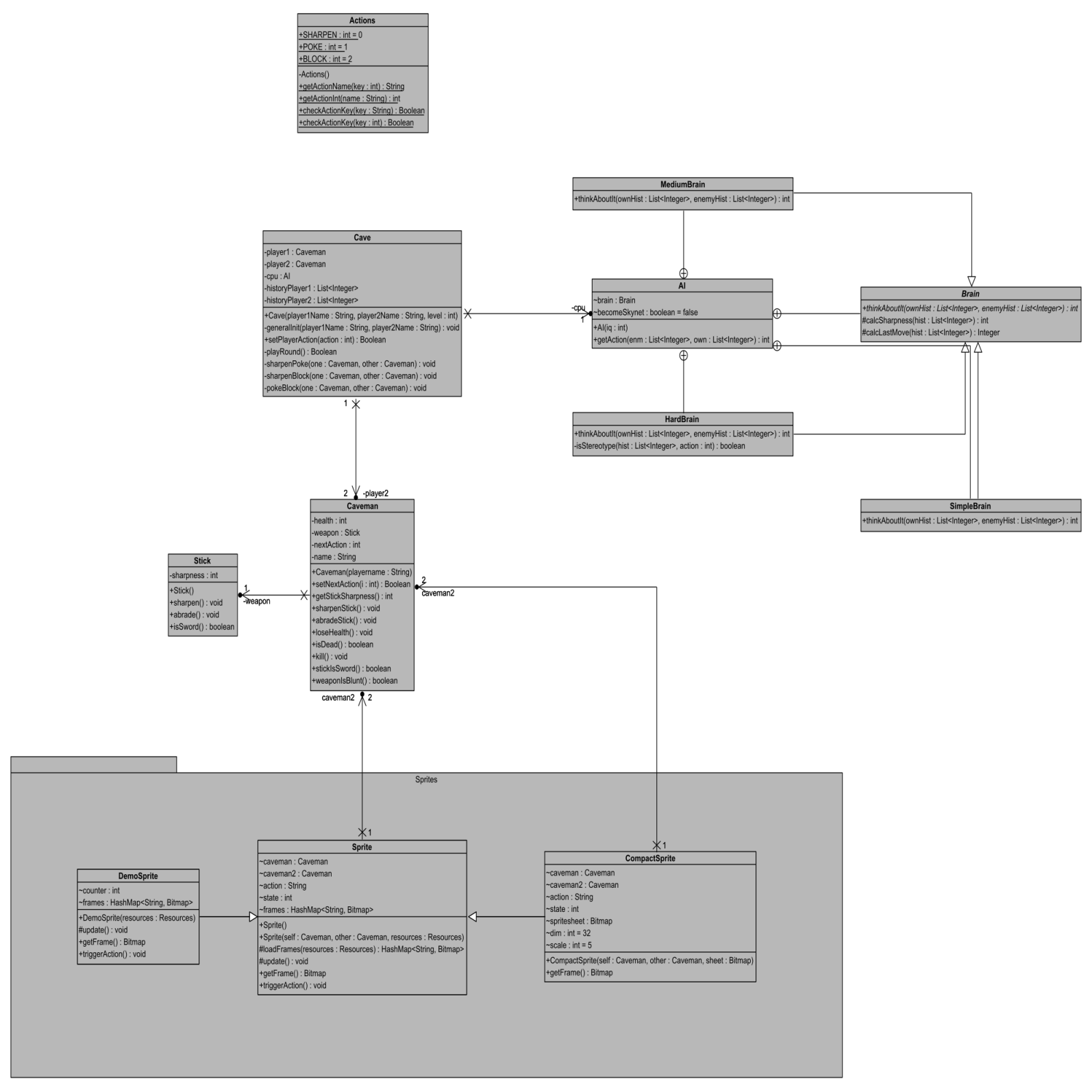


### Spieler 1: Angriff, Spieler 2: Angriff



# Implementierung

## SimpleBrain



/\*\*

\* Simple AI, lowest difficulty

\*/

public class SimpleBrain extends Brain {

/\*\*

\* Wählt die nächste Aktion aus

\*

\* @param ownHist Eigene Vergangenheit der Aktionen

\* @param enemyHist Vergangenheit der Aktionen des Gegners

\*

\* @return Auszuführende Aktion

\*/

@Override

public int thinkAboutIt(List<Integer> ownHist, List<Integer> enemyHist) {

// Ermittelt aus der Vergangenheit die aktuelle Schärfe des eigenen Stocks (Schärfen - Angriffe)

int ownSharpness = calcSharpness(ownHist);

// Ermittelt aus der Vergangenheit die aktuelle Schärfe des gegerischen Stocks (Schärfen - Angriffe)

int enemySharpness = calcSharpness(enemyHist);

// Ermittelt die zuletzt ausgeführe Aktion aus der Liste der eigenen Aktionen

int ownLastMove = calcLastMove(ownHist);

int action = Actions.SHARPEN;

// Falls die aktuelle Schärfe des Stocks gleich 0 ist -> schärfen dringend notwendig

if(ownSharpness == 0)

return action;

// Falls der Gegner keinen geschärften Stock besitzt, kann er getötet werden

if (enemySharpness == 0)

action = Actions.POKE;

// Falls der Gegner einen geschärften Stock besitzt, dann soll geblock werden, um weitere Schäden zu verhindern.

else if (enemySharpness > 0 && ownLastMove != 2)

action = Actions.BLOCK;

return action;

}

}

Hier ist das SimpeBrain aufgeführt, dass durch eine einfach Logik versucht den Gegner zu besiegen.

Falls der eigene Stock nicht mehr scharf ist, dann kann nicht angegriffen werden, damit ist das schärfen dringend notwendig.

if(ownSharpness == 0)

return action;

Falls z.B. der Gegner keinen geschärften Stock, dann wird er wenn möglich angegriffen, um einen Sieg zu erzielen.

if (enemySharpness == 0)

action = Actions.POKE;

Falls der Gegner einen geschärften Stock besitzen, dann muss dringend abgewehrt werden.

if (enemySharpness > 0 && ownLastMove != 2)

action = Actions.BLOCK;

## Layout

Im folgendem ist ein Teil von Layout der Anleitung zu sehen.  
Dort befindet sich ein ScrollLayout, damit die über den Bildschirmrand hinausragenden Elemente durch scrollen in den Sichtbereich geholt werden können.

<ScrollView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/scrollView"

android:layout\_gravity="center\_horizontal|top" >

<LinearLayout

android:orientation="vertical"

android:layout\_width="fill\_parent"

android:layout\_height="fill\_parent"

android:id="@+id/fullscreen\_content\_controls">

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:textAppearance="?android:attr/textAppearanceLarge"

android:text="HowTo"

android:id="@+id/textView"

android:textSize="50dp"

android:textColor="#634141" />

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:textAppearance="?android:attr/textAppearanceMedium"

android:text="Ausgangssituation"

android:id="@+id/textView2"

android:layout\_marginTop="20px"

android:textSize="20dp"

android:textColor="#634141" />

Die einzelnen Texte werden in den TextViews bereitgestellt. Damit die XML-Datei nicht unübersichtlich wird, werden die Texte ausgelagert und über eine ID referenziert.

<TextView

android:layout\_marginTop="8px"

…

android:text="@string/inhaltSpielEnde"

android:id="@+id/textView6"

android:textColor="#634141" />

# Tests

## Black Box

### Stick

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Testfall | Funktion | Erwartete Rückgabe | Rückgabe | Gültig |
| 1 | getSharpness | 0 | 0 | Ja |
| 2 | sharpen |  |  | Ja |
| 3 | getSharpness | 1 | 1 | Ja |
| 4 | isSowrd | False | False | Ja |
| 5 | sharpen |  |  | Ja |
| 6 | sharpen |  |  | Ja |
| 7 | sharpen |  |  | Ja |
| 8 | getSharpness | 4 | 4 | Ja |
| 9 | isSowrd | False | False | Ja |
| 10 | sharpen |  |  | Ja |
| 11 | isSowrd | True | True | Ja |
| 12 | sharpen |  |  | Ja |
| 13 | isSowrd |  |  | Ja |
| 14 | abrade |  |  | Ja |
| 15 | getSharpness | 5 | 5 | Ja |
| 16 | abrade |  |  | Ja |
| 17 | isSowrd | False | False | Ja |

### Actions

#### checkActionKey

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Testfall | Parameter | Erwartete Rückgabe | Rückgabe | Gültig |
| 1 | "test" | False | False | Ja |
| 2 | "" | False | False | Ja |
| 3 | "sharpen" | True | True | Ja |
| 4 | "poke" | True | True | Ja |
| 5 | "block" | True | True | Ja |
| 6 | *NULL* | False | Exception | Nein |

# Quellen

1. Prognose zu den Marktanteilen der Betriebssysteme am Absatz vom Smartphones weltweit in den Jahren 2014 und 2018 (http://de.statista.com/statistik/daten/studie/182363/umfrage/prognostizierte-marktanteile-bei-smartphone-betriebssystemen/)
2. Eclipse Screenshot (http://www.techotopia.com/images/0/05/Eclipse\_android\_4.2\_main\_screen.png)
3. Android Studio Screenshot (https://s3.amazonaws.com/infinum.web.production/repository\_items/files/000/000/168/original/android-studio-3.png?1393599622)