|  |
| --- |
| dorf10.ch |
| Snake – 71m024 |
| Systemdokumentation |
|  |
| **Timo Bühlmann** |
|  |

1. Einleitung
   1. Zweck des Dokuments

Diese Dokumentation erhält wichtige Informationen zum Snake von Timo Bühlmann. Diese Informationen sind für die Wartung oder Weiterentwicklung dieses Systems notwendig.

* 1. Aufgabenstellung Snake

Das Spiel muss verschiedene Spiel-Elemente beinhalten, welche von einer einheitlichen Klasse erben. Dazu gehören: Schlange(n), Diamanten und die Spielgrenze.

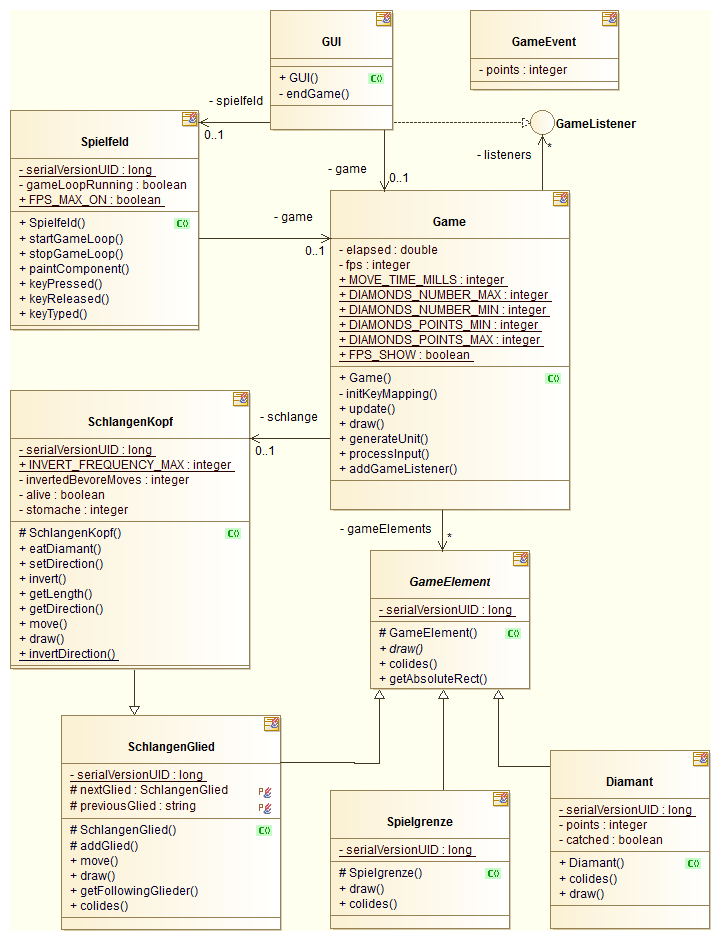
Die Schlange bewegt sich in regelmässigen Abständen in die zuletzt vom Spieler eingegebene Richtung.  
Erkennt das Spiel, wenn die Schlange mit einem anderen Spiel-Element kollidiert. Ist das andere Element eine Schlange oder die Spielgrenze stirbt die Schlange. Kolidiert sie mit einem Diamant, verschwindet dieser vom Spielfeld und die Schlange wächst um die Punktzahl des Diamanten.

1. Resultate aus der Analyse
   * 1. Iteration 1

Ein **Game** hat **Spielelemente**.  
**Spielgrenze** erbt von **Spielelement**.  
**Diamant** erbt von **Spielelement**.  
**Schlange** erbt von **Spielelement**.  
Eine **Schlange** hat einen **Kopf**.  
**Kopf** erbt von **Glied**.

* + 1. Iteration 2

Das **Schlangenglied** hat eine Loop-Methode, welche die **Schlange** in die gesetzte Richtung gewegt.  
Das **Schlangenglied** muss eine **Richtung** in Form eines **Enums** haben.  
Das **Schlangenglied** implementier den **KeyListener** und wird beim **GUI** als **KeyListener** registriert.

1. Architektur
   1. Statische Struktur
2. Das „Game-Loop-Patern“

Dieses Patern befindet sich in der Spielfeld-Klasse und sorgt dafür, dass bei jedem Durchlaufen seines „Loops“ folgende Methoden auf dem Game aufgeruffen werden:

„**Update**“, „**Draw**“ und „**ProcessInput**“

Dabei spielt es keine rolle, in welchen zeitlichen Abständen der „Game-Loop“ durchlaufen wird, weil die Update-Methode sich jeweils dem übergebenen Zeit-Delta anpasst.

1. **Logische Darstellung der Schlange**

Die Schlange besteht nur aus „SchlangenGliedern“ und einem Kopf, welche auch von „SchlangenGlied“ erbt.

Der Kopf zeichnet sich dadurch aus, dass er noch einige „nicht-Glied spezifische“ Informationen über die Schlange weiss, wie z.B. ihre Richtung.  
Die Richtung wird als „Point“ dargestellt, weil dieser X- / Y-Komponenten beinhaltet, welche für den Bewegungs-Vektor von Nöten sind.

Jedes Glied weiss, ob und welches Glied vor beziehungsweise hinter ihm ist. Die meisten Operationen, welche die ganze Schlange betreffen werden also rekursiv über die Glieder ausgeführt.

1. Positionierung mittels eines Rechtecks (Rectangle)

Da das Rechteck X / Y-Koordinaten besitzt sowie eine Höhe und eine Breite eignet sich dieses ideal um um die Spielelemente („GameElements“) zu positionieren / skallieren.

1. Modellierung des Schlangenkörpers

Die Schlange wird in jedem Raster-Feld, welches sie überdeckt mit einem ausfüllenden Kreis dargestellt, welcher eine teils zufällige Farbe besitzt. Auf den Kopf werden zusätzlich noch zwei Augen gezeichnet.

1. Wachstum / Fortbewegung der Schlange

Die Schlange wird unabhängig von den FPS in regelmässigen Abständen in ihre Festgelegte Richtung bewegt und zwar immer um eine Einheit („Unit“).

Wächst die Schlange, wird ans hinterste Glied eines angehängt (als „nextGlied“ hinzugefügt).  
Das letzte Glied wird innerhalb „addGlied“-Methode mittels eines rekursiven Algorithmus ermittelt.

1. Auslösung des Win-Events

Auf der Diamant-Klasse befindet sich ein Klassen-variable (Integer), welche mit jedem erstellen eines Diamanten um 1 erhöht wird. Wird ein Diamant ge-“catched“, wird die Variable wieder um 1 verringert.

In der Update-Methode der „Game“-Klasse wird überprüft, ob die Variable „== 0“ ist. Trifft dies zu, ist das Spiel gewonnen.

* 1. Dynamik

In folgendem Sequenz-Diagramm wird die Schlangenbewegung um eine Einheit stark vereinfacht dargestellt. Dabei wird davon ausgegangen, dass die Schlange bereits 3 Glieder besitzt und einen Punkt im Magen hat.

Es wird also zuerst ein Glied hinzugefügt und anschliessend Glied für Glied bewegt.

