Belegarbeiten

Fachbereich 1: Computer Engineering
Softwaretechnik - Sommersemester 2016
Luong Tien Dat s0546266@htw-berlin.de
Ardilla Latifasari s0547045@htw-berlin.de
Dozent Herr Prof. Dr. Thomas Baar

1. Absract

Heutige Software besteht in der Regel aus einer Reihe von Komponenten, die miteinander interagieren, um die für die Implementierung der Anwendung nötigen Aufgabe auszuführen. Die Entwicklung umfasst das Erstellen solcher Komponenten, indem Quellcode in einer der zahlreichen verfügbaren Sprachen geschrieben wird. Mithilfe von verschiedener Modelle und die Änderungen des Computerspielklassiker Snake. Die Codestruktur wurde objektorientierter gestaltet und ein neues Spiel implementiert.

2. Die Sofware

Wir haben uns für unsere Projekt das Spiel Snake entschieden. Snake ist ein Computerspielklassiker, bei dem eine sich gerade oder rechtwinklig bewegende Schlange durch ein Spielfeld gesteuert wird. Ziel des Spiel ist, die als Futter angebotenen zufällig erscheinenden “Happen” aufzunehmen und Hindernissen, einschließlich des eigenen Schlangenkörpers auszuweichen. Während die Schlange mit jedem Happen wächst, wird das Manövrieren bei zunehmend vollerem Spielfeld – und bei eventuelle schnellerem Grundtakt – immer schwieriger.

Der Entscheidungspunkt des Spiel Snake liegt daran, dass es nicht nur unsere Kindheit verbunden hat sondern auch die Anforderung ihre Quellcode in Sprache c++ übertragt wird.

3. Die Aufgabenstellung

Als Teil des Projekt mussten wir bestimmte Änderungen im Programm vornehmen. Der Quellcode ist hauptsächlich in den Dateien snake.cpp implementiert und im Dev C++ 5.11 kompiliert worden. Als die erste Aufgabe müssen wir die Code objektorientierter umwandeln. Die zweite Aufgabe sollen wir 2 Schlangen erstellen, die sich Futter aufzunehmen

4. Modelierung

Um die Spiellogik und Funktionalität sowie die Vorgenommen Änderungen besser zu erklären, haben wir Arten von Diagrammen dargestellt:
In den nachfolgenden Kapitel wird jeweils Diagramm besser gezeigt, wie sich unsere Änderungen des Programms auf das Gesamtbild auswirken.

4.1 Use Case

Am ersten Diagramm wird das Use-Case des Spiel Original-Snake dargestellt.

Spielbeschreibung: Der Schlange wurde vom Spieler gesteuert, um die maximale Food aufzunehmen.

Primärer Aktor: Spieler

Vorbedingung: Das Programm wurde gestartet, der Schlage wurde durch die Tastatur: nach rechts (Taste D), nach links (Taste A), nach oben (Taste W), nach unten(Taste S)

Nachbedingung: Der Schlage hat am Anfang mit 3-Leben gestertet, nach 20 Mal die aufgenommene Food ist das Leben um eins gestiegen. Wenn der Schlage sich selbst beißt, ist ihre Leben um eins gesunken.

Prozess:

1. Der Spieler startet das Spiel
2. Das neue Fester wurde mit dem Logo “Snake 2” geöffnet.
3. Das System warten darauf, dass die Spieler das Leer-Taste drückt.
4. Der Schlange wird von dem Spieler durch die gedruckte Taste (ASDW) gesteuert.
5. Das System zeigt die Score der Schlangein im linken Fester und die High Score im rechten Fester. Die aktuelle Food und der Kopf des Schlagen wird im rechten Fester dargestellt.
6. Bis der Schlagen hat kein Leben, wird das Spiel beendet.
7. Falls man eine neue High Score geschafft hat, wird die neue High Score gespeichert und das Spiel beendet.
8. Use Case endet erfolgreich.

4.2 Das Klassendiagramm

Die folgende Klassendiagramm stellt die Klassebeziehungen des originale Spiel (Diagramm1) und die neue objektorientierter Code(Diagramm2) dar. Der Originalcode wurde viel die globalen Variablen benutzt und die ganze Funktionen in File “Snake.cpp” implementiert. Im Gegendsatz wurde die neue Code in zwei neue Klasse: “Until”,“GUI” geteilt. Wegen des Spiel zwei Schlangen benötigt sind, wurde die Funktionen in des Klasse “Snake” neu codierert.

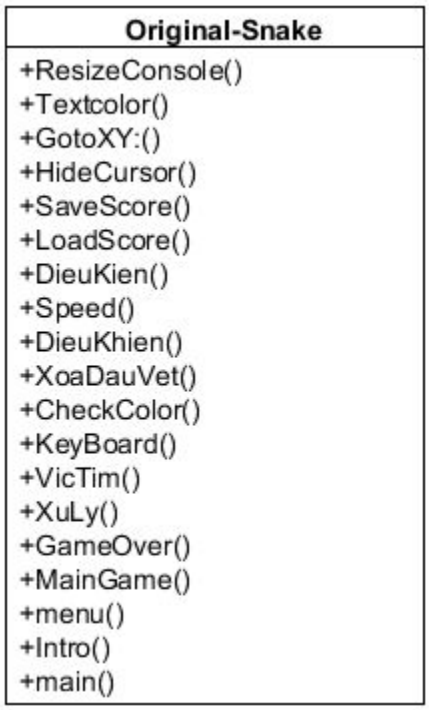

Diagramm 1. Original Snake

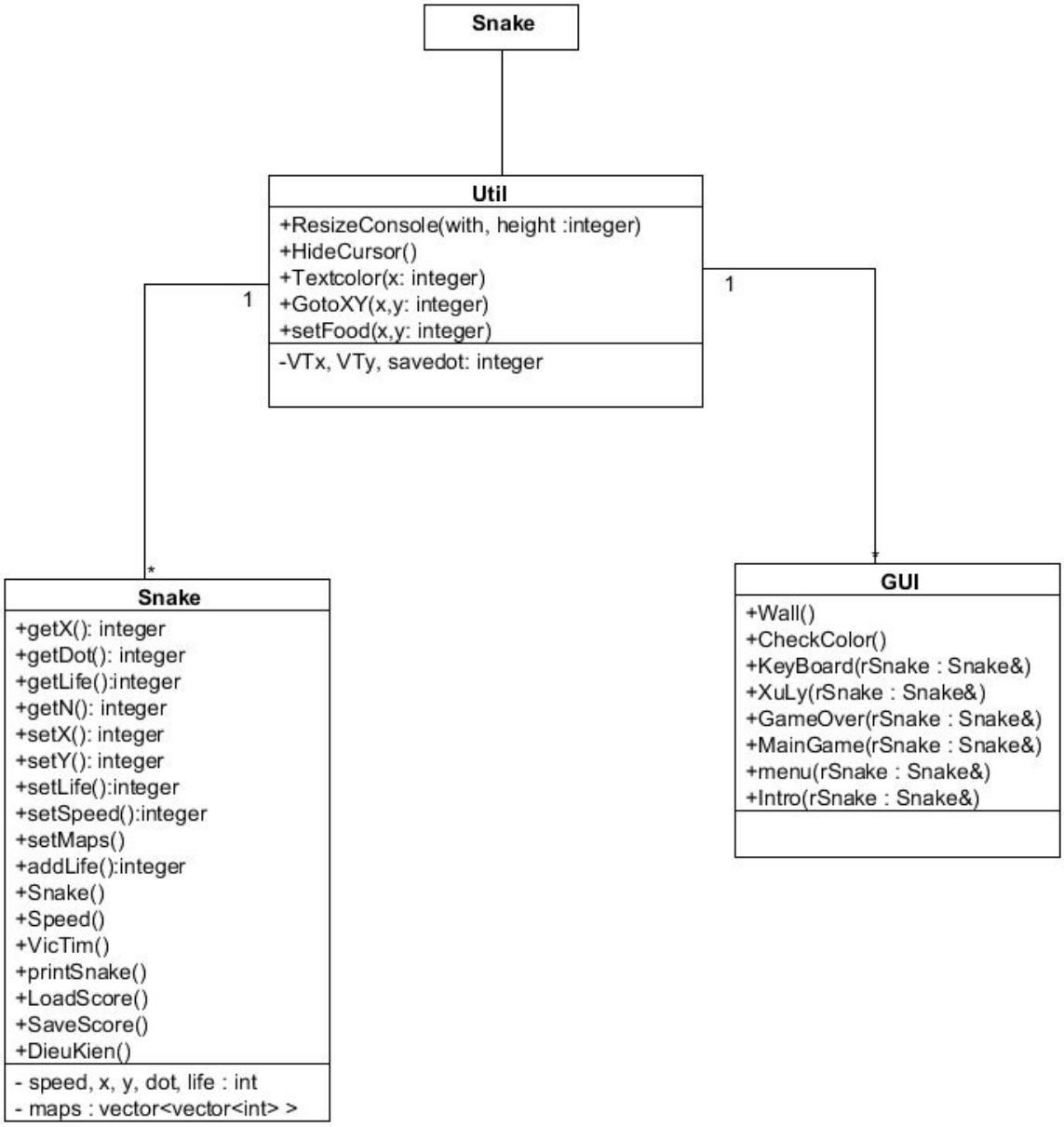

Diagramm2. Die neue Code

5. Fazit

Abschließend ist festzustellen, dass wir von unserem Belegarbeit viel gelernt haben. In Sprache c++ haben wir noch mehr über das Erstellen die Objekorientierung von einem Programm gemacht und die kompliziertere Anforderungen erfüllt. Damit können wir einen Überblick über die Aufgabe von dem Entwickler, wir man ein Software weiter verbessern und codieren kann.
Es war sehr hilfreich mit Git-Repository zu arbeiten, die Code wurde immer aktuellen und sicher speichern. Wir können schneller die Aufgabe teilen, dadurch wird die Lösung umgehende zu finden und die Kommunizieren verbessern.