Software Engineering II
Maintenance

Gruppe C

Bauhaus-Universität Weimar

10. Dezember 2023

Maintenance

Zusätzliches Feature:

Implementation eines zusätzlichen Feautures:

Als zusätzliches Feature haben wir uns überlegt, 'Barrieren' einzubauen. Sobald der Spieler auf das Feld einer Barriere kommt, stirbt seine Schlange.
Zu diesem Zweck haben wir eine neue Klasse, Barriere, erstellt.
In der Hauptanwendung, also Slither, wurde ein Feld von Barrieren hinzugefügt. Diese Barrieren werden auf dem Spielfeld angezeigt. Bei jeder Kollisionsüberprüfung, dh. bei jedem Aufruf der collision-Funktion wird jetzt auch das Feld an Barrieren übergeben und es wird zusätzlich auf eine Kollision mit einer Barriere getestet.Dafür wird dann über den Container mit den Barrieren iteriert und geprüft, ob der Schlangenkopf gleich der Koordinaten einer Bariere ist. Ist dies der Fall, stirbt die Schlange.

Verbesserung:

Bereits implementiertes Feauture, das wir erweitern/verbessern:

Wir haben es uns als Ziel gesetzt, einerseits die Oberfläche zu optimieren, und an anderer Stelle haben wir das Ziel die Timeline zu optimieren und darüber hinaus wollten wir den Kopf der Schlange zur besseren Anschaulichkeit markieren.
´

Oberfläche:
Wir haben festgestellt, dass es Probleme mit der Oberfläche gab. So haben sich die Schlangen nicht bündig auf den Kacheln bewegt, sondern, um 2 bis 3 Pixel versetzt. Unser Ziel war es, dies zu verbessern.
Dafür haben wir zuvor implementierte Gridpane gelöscht und stattdessen durch ein Canvas ersetzt, in welches wir nun Rechtecke kachelförmig angeordnet, einzeichnen.
Durch unsere Erweiterung des Spiels mit Barriers haben noch ein zusätzliches Canvas erstellt in welchem die Barriers gezeichnet werden.
Wir haben diese Unterteilung in drei verschiedene Ebenen vorgenommen, da sich die einzelnen Elemente des Spiels unterschiedlich oft verändern und somit unterschiedlich oft neu erstellt werden müssen.So muss das Schlangencanvas nach jedem Zug erneuert werden, das Barriercanvas nach jedem Spiel und der Hintergrund ist unabhängig uns bleibt dauerhaft bestehen. Um zu verhindern, dass Kacheln oder Barrieren unötig bei jeden Frame neu gezeichnet werden, liegen die drei Element auf verschiedenen Flächen.
Das oberste Canvas stellt das Canvas dar, in welchem die Barriers gezeichnet werden, darunter liegt der mit dem Schlangen und zuletzt kommt der Hintergrund. Der Barrier-Canvas liegt über dem Schlangen Canvas, damit eine Schlange, die auf ein Barrier stößt nicht über dem Barrier zu sehen ist.

Timeline:
Wir haben auch festgestellt, dass es einen Fehler beim Ablauf des Spiels gibt.
Die Schalgen sollen während des Spielverlaufs in regelmäßigen Intervallen länger werden. Jedoch werden der Schlange auch schon vor offiziellem Beginn das Spiel, also bevor eine Taste gedrückt wurde, schon neue Schlangenelemente hinzugefügt, die dann bei Beginn des Spiel in sehr kurzen Abständen sichtbar werden.
Um dem Abhilfe zu schaffen haben wird die Struktur des Spielverlaufs geändert. So haben wir jetzt einen Methode new_game. In dieser Methode wird eine neue Timeline mit ihren keyframes erstellt, jedoch wird die Timeline, erst mit dem drücken einer Taste, also mit dem ofiziellen Spielbeginn gestartet, nicht wie zuvor beim Aufrufen des Programms. Damit wird die Schlange jetzt

erst länger, wenn das Spiel gestartet ist.

Kopf markieren:
Uns fiel es auf, dass es der besseren Steuerung dienlich wäre, wenn der Kopf explizit markiert ist. So muss man den Kopf der Schlange nicht immer exakt mit den Augen verfolgen und es ist leichter zu sehen, in welche Richtung sich die gegenerische Schlange bewegt.
Dafür haben wir in der Draw-Methode der Schlange, noch zwei kleine Kreise auf dem Kopfsegment zeichnen lassen, um so die Form von Augen zu erzeugen.