Melissa Federer

Wiss |

Catch Them All

MODUL 226

Ein Bild, das Text, Elektronik, Screenshot, Anzeige enthält.

Automatisch generierte Beschreibung

Inhalt

[Was ist mein Projekt 2](#_Toc93947230)

[Inspiration 2](#_Toc93947231)

[Spielumfang 2](#_Toc93947232)

[Codebeschreibung 3](#_Toc93947233)

[Grundlegende Funktionen 3](#_Toc93947234)

[Abstrakte Klasse und vererbung 4](#_Toc93947235)

[Collision Detection 4](#_Toc93947236)

[Zustandsautomaten 5](#_Toc93947237)

[Vorgehen 6](#_Toc93947238)

[Mindmap 6](#_Toc93947239)

[Zeitplan 6](#_Toc93947240)

[UML - Klassendiagramm 6](#_Toc93947241)

[Probleme 7](#_Toc93947242)

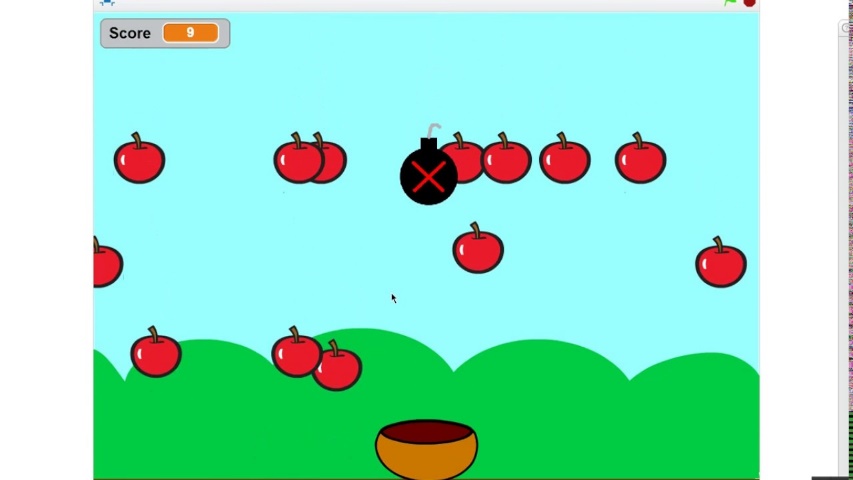
[Fazit 7](#_Toc93947243)

[Quellen 7](#_Toc93947244)

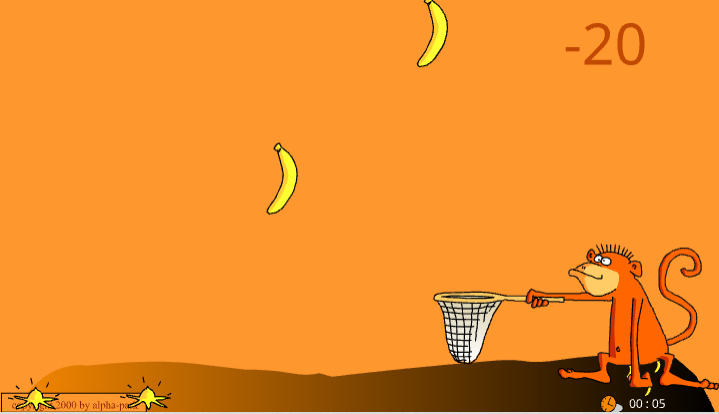
# Was ist mein Projekt

In dem Modul 226B besteht die LB daraus, ein eigenes Spiel zu Programmieren. Das Ziel von meinem kleinen Spiel ist simpel: So viele Früchte wie möglich mit einem Korb einzusammeln.

## Inspiration

Spiele dieser Art gibt es in dutzenden Ausführungen. Das Spielprinzip ist jedoch immer das gleiche.

Beispiel Quelle



Beispiel Quelle

Der Spieler steuert ein Objekt, welches sich nur von rechts nach links steuern lässt. Bei manchen funktioniert diese Steuerung mit der Tastatur, bei anderen mit der Maus.

## Spielumfang

Der Spieler kann das Korb-Objekt mit der Maus steuern. Dieses Korb Objekt hat einen fixen y-Wert, damit dieser immer am gleichen Ort steht. Mit diesem Korb gilt es nun, verschiedene fallende Früchte einzusammeln.

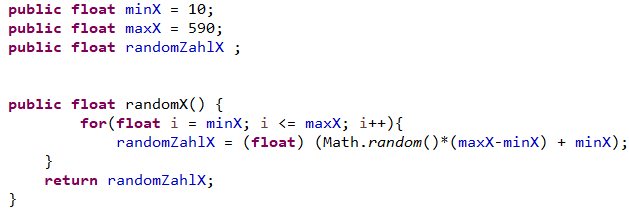
Diesen Früchten wurde ein fixer y-Startwert gegeben. Die x-Position ist jedoch Random, somit kann der Spieler nie vorhersehen, wo die Früchte erscheinen.

Jede einzelne Frucht gibt beim auffangen Punkte.

Das Ziel des Spiels ist es so viele Punkte wie möglich zu ergattern, ohne die gefährlichen Bomben zu erwischen.

# Codebeschreibung

## Grundlegende Funktionen



1 So wurde die randomX Funktion für die Früchte umgesetzt

Dem Spieler werden aber auch Hindernisse gestellt. Um das Spiel noch ereignisvoller zu gestalten, fällt ab und zu eine Bombe da hinunter, wo eigentlich eine Frucht fallen sollte. Wird diese gefangen, ist das Spiel sofort vorbei.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

2 Sobald der Korb die Bombe berührt, kommt man in den End Screen

Um das Spiel noch etwas interessanter zu gestalten, wurde dem Spiel noch ein Schwierigkeitsgrad eingebaut. Dieser wurde mit einer Modulo Funktion programmiert. Sobald der Spieler 10 Punkte erreicht. Wird der Speed der fallenden Objekte um Zwei erhöht.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

3 Modulo Funktion für die Schwierigkeit

Dies funktioniert dann auch mit 20 Punkten, 30 Punkten und so weiter

## Abstrakte Klasse und vererbung

Damit alle Frucht-Klassen nicht unzählige Code Zeilen beanspruchen, habe ich mir überlegt, eine Abstrakte Klasse «Frucht» zu erstellen.

Somit ist hier als Beispiel die Apfel Klasse merklich kleiner:

Ein Bild, das Text enthält.

Automatisch generierte BeschreibungEin Bild, das Text enthält.

Automatisch generierte Beschreibung

4 In den einzelnen Früchte Klassen ist nur noch die Draw Methode individuell

1 In der Frucht Klasse findet man nun alle Getter und Setter

## Collision Detection

Die Vererbung hat ebenfalls die Collision Detection simpler gemacht

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

5 Gesamte Collision-Detection im Game Controller

In der Collision Class sieht das dann wie folgt aus

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

6 Collision der Melone, Dazu kommt noch der Score

Nur für die Bombe musste man eine kleine Änderung machen



7 Sobald die Bombe getroffen wurde, gehe direkt in den End Screen

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

8 Da es für die Bombe einen True oder False Wert geben muss, musste man void zu einem Boolean ändern

## Zustandsautomaten

Ein Bild, das Text enthält.

Automatisch generierte BeschreibungEin Zustandsautomat ist ein Steuerungskonzept, welches eine Reihe von Zuständen besitzt, durch die sich ihre Funktion definiert.

2 verschiedene Zustände wurden mit einem ENUM erstellt

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Mit Switch-Case wird je nach Zustand ein anderer Code ausgeführt.

3 Switch-Case

Ein Bild, das Text enthält.

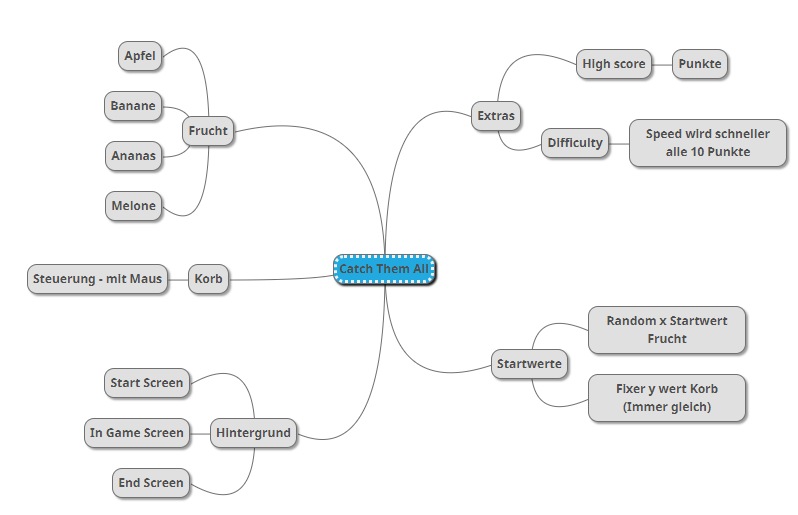
Automatisch generierte Beschreibung

Mithilfe eines Inputs des Users kann der Spielzustand geändert werden:

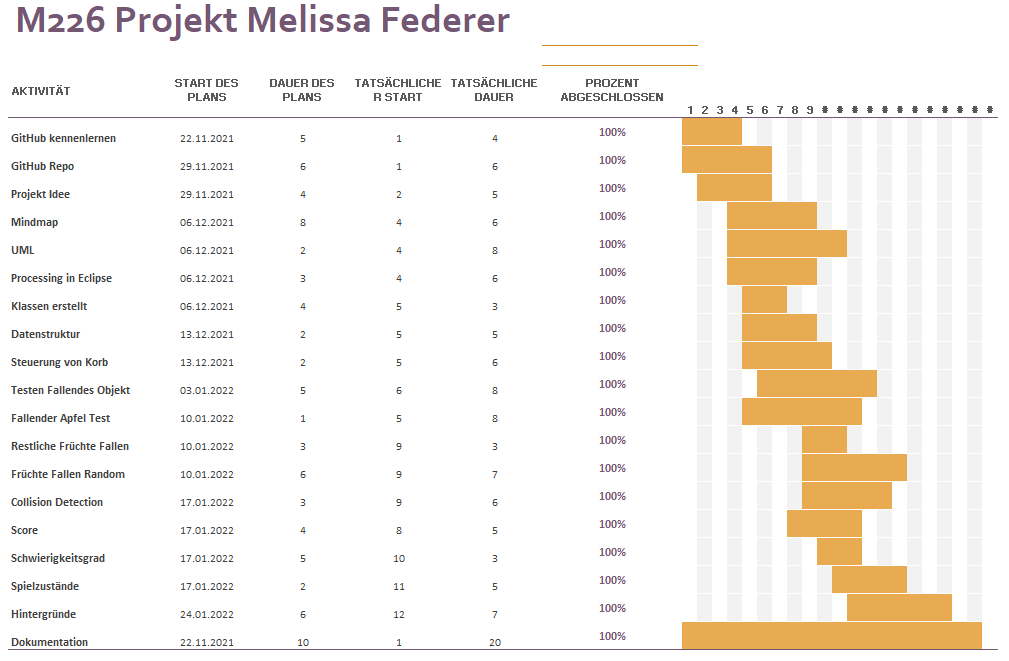
4 drückt der Spieler "s" geht das Spiel los

# Hilfestellungen

## Mindmap

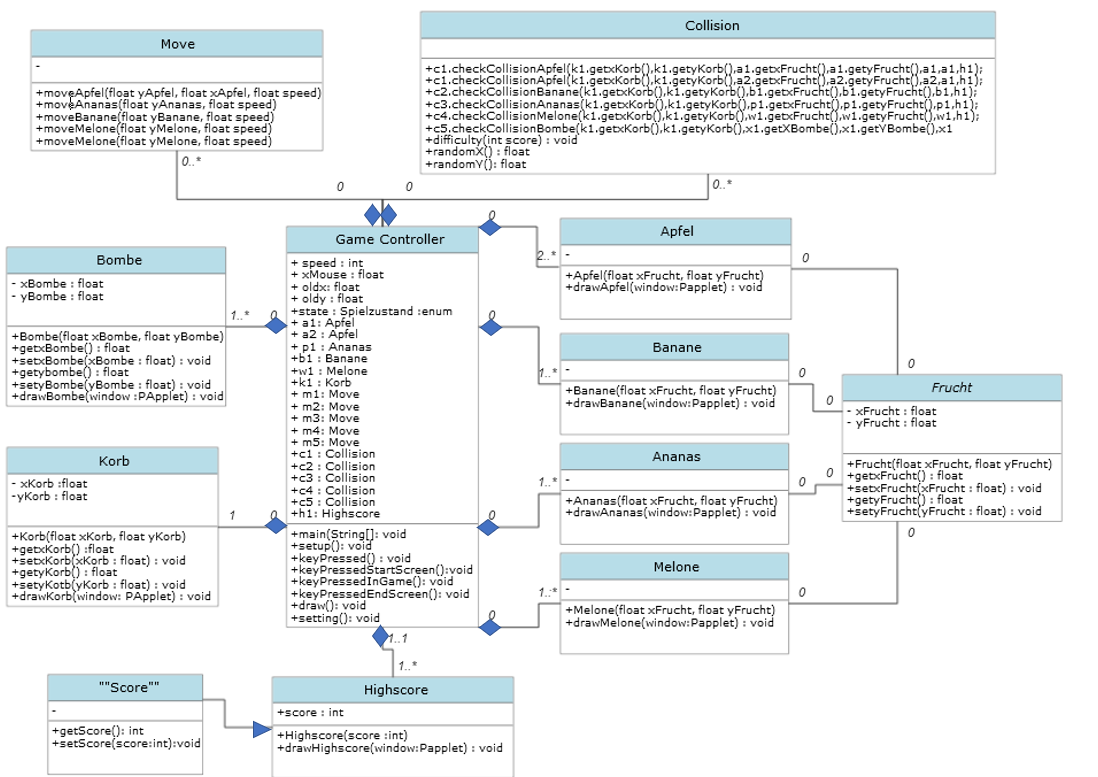


## Zeitplan



## UML – Klassendiagramm

## UML



# Testen

Mit den JUnit Tests habe ich zum Beispiel getestet, ob die Move Funktion korrekt ist.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

9 Hier wurde getestet, ob sich das Melonen Objekt bewegt

# Das konnte ich lernen

Ich konnte verstehen, wie man mit abstrakten Klassen arbeitet und diese umsetzt. Ich konnte allgemein viel lernen und mehr verstehen, wie der Prozess zur Erstellung eines grösseren Codes aussieht.

Mit der Zeitplanung habe ich mich ein Wenig überschätzt. Manche Funktionen nehmen dann doch mehr Zeit in Anspruch als geplant. Doch auch daraus konnte ich einiges lernen.

# Quellen

Quelle 1

https://www.youtube.com/watch?v=3QKiK4rJIB0

Quelle 2

https://langeneggers.ch/spiele/games/72/Bono.html