

1 Teilnehmer/innen des Teams:

Klasse: BI20C	Team: Gabriel Franz, Cornel Forster
------------------	--

2 Anforderungsdefinition (Meilenstein A)

SPACE INVADERS

Auftrag:

(Allgemeine Beschreibung)

Nutzen: Wir wollen unser Wissen vom Breakout-Spiel erneut anwenden. Mit dem Spiel soll die Reaktionsgeschwindigkeit eines Users getestet werden.

Szenario:

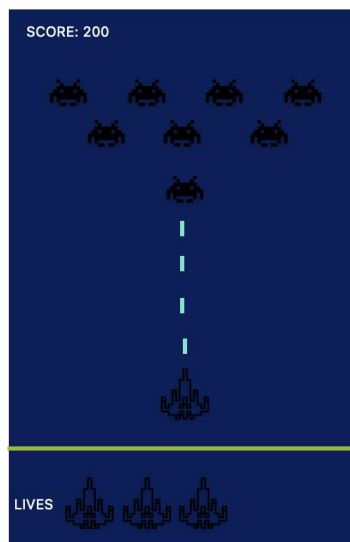
Für das Spiel wird ein eigenes Szenario erstellt.

Details:

- Das Spiel beinhaltet ein schiessendes Raumschiff und verschiedene UFOs, welche abgeschossen werden müssen
- Ausserdem soll das Spiel ein Score beinhalten.
- ...

Machbarkeitsabklärung:

- Folgende Features sind vorab untersucht worden:
Keine, da wir uns sicher sind, dass wir das Game erstellen können mit den nötigen Features.

Mockup:

MUSS**Kriterien:**

(Konkrete Features, die umzusetzen sind)

Folgende Features sollen implementiert werden (Funktionalität):

- Dass Raumschiff lässt sich nach links und rechts bewegen
- Dass Raumschiff kann auf Befehl schießen
- Der User hat zu Beginn des Spiels 3 Leben.
- Der User kann Leben verlieren, wenn er von einem UFO getroffen wird.
- Die UFOs können von Schüssen getroffen und eliminiert werden
- Scoreanzeige

KANN**Kriterien:**

(Konkrete Features, die optional sind)

Folgende Features können zusätzlich implementiert werden: (Kreativität)

- Spezielle UFOs können Geschosse reflektieren
- Durch Perks kann das Geschoss aufgerüstet werden
- Der Alltime Highscore wird während des Spiels angezeigt
- Die Formation der UFOs wird generisch erstellt
- Verschieden Starke und Grösse Gegner

3 Lösungsdesign (Meilenstein B: Aufgabe 1)

Anhand der Analyse wurde folgendes Lösungsdesign entworfen:

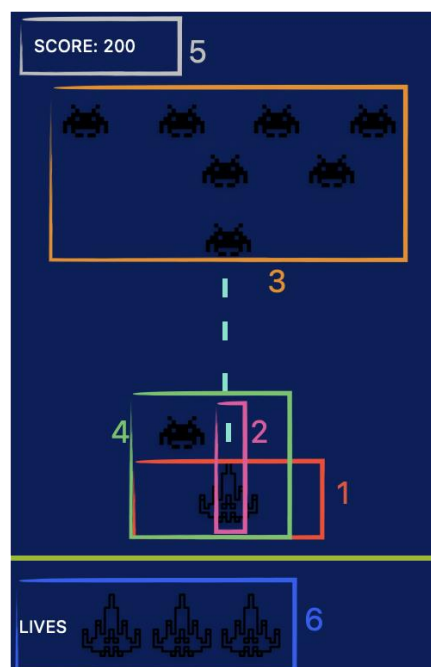
3.1 Funktionsmodell

Im Folgenden sind die erwarteten Eingaben und Ausgaben beschrieben / dargestellt:

Objekte: Raumschiff, Schuss, diverse UFOs

Konzepte: Bewegung von Raumschiff und UFO, Schussauslösung, Kollisionserkennung, Scoreboard, Anzahl Leben Raumschiff

In folgendem wird das Konzept dargestellt:



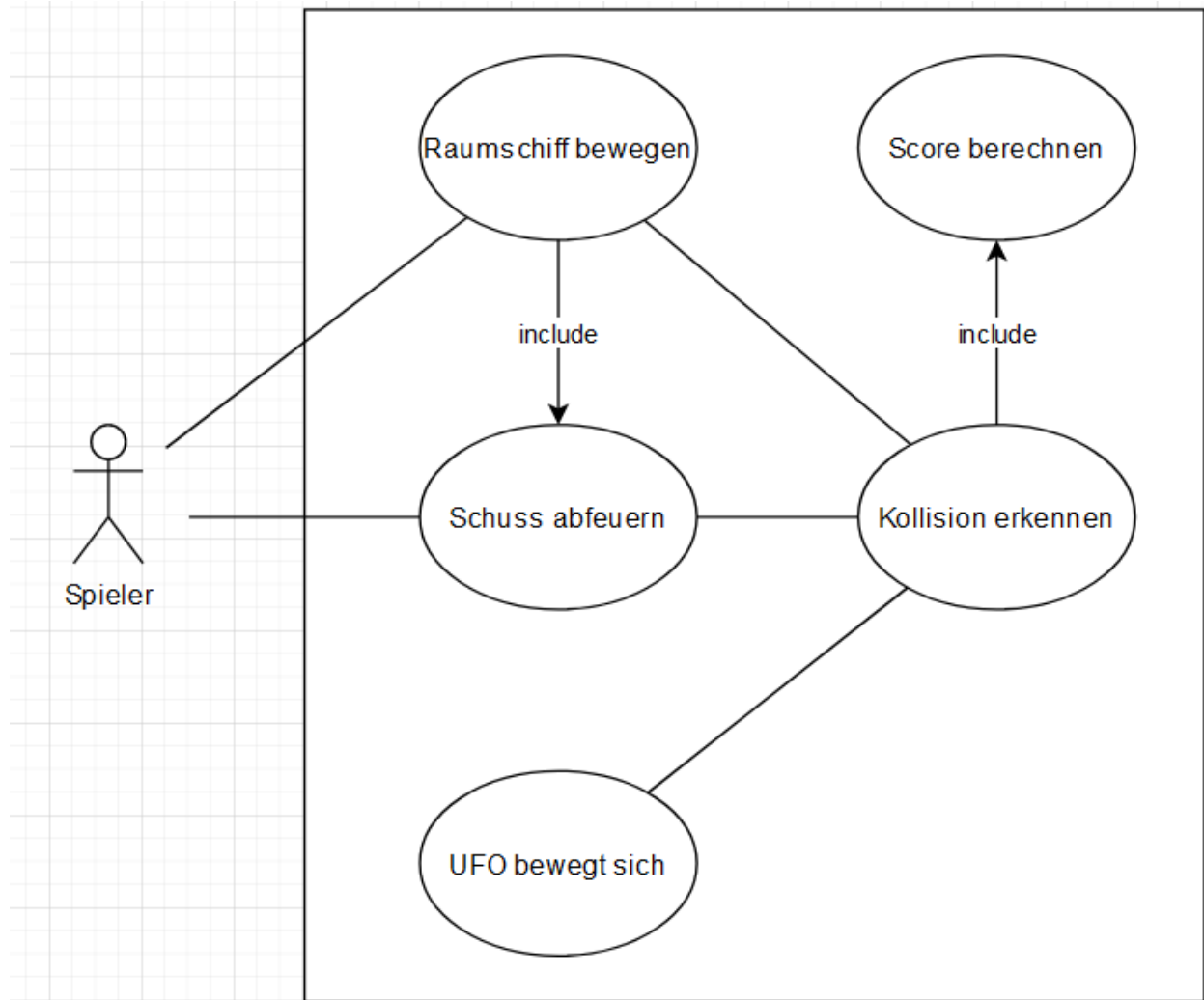
Legende:

Die nachfolgenden Nummern zeigen die Funktionen im Konzept

- Raumschiff bewegt sich vertikal (1)
- Raumschiff schießt (2)
- UFO bewegen sich horizontal nach unten (3)
- UFO kann von Schuss getroffen/ zerstört werden (3)
- UFO kann Raumschiff berühren und Leben abziehen (4,6)
- Anzeige des Scores(5)

3.2 Anwendungsfälle (UseCases)

Folgende Anwendungsfälle sind hier detailliert dokumentiert:

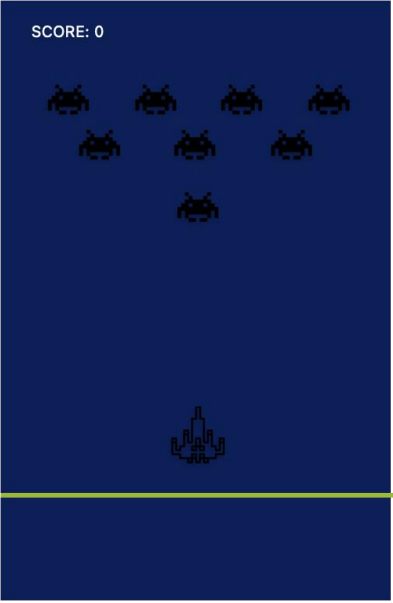
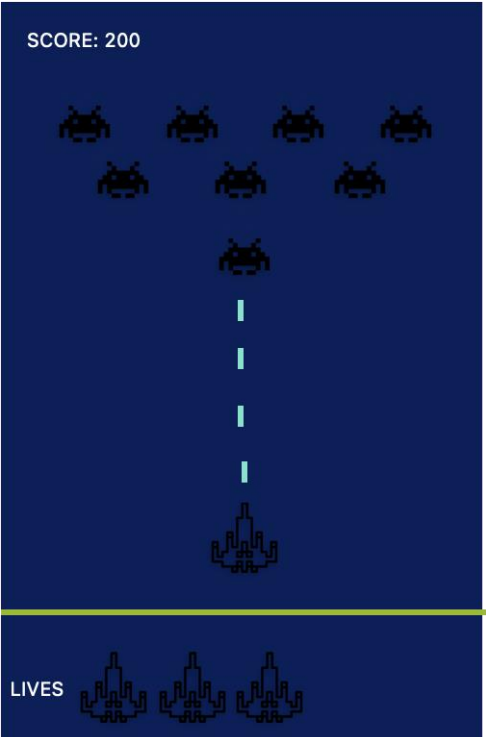


Legende:

- Der User bedient das Spiel, indem er das Raumschiff steuert und Schüss abgibt
- Raumschiff.
- Das Raumschiff wird animiert und visualisiert.
- Die UFOs werden animiert und visualisiert.
- Die Schüsse werden animiert und visualisiert.
- Kollisionen werden erkennt und haben Einfluss auf den Score

3.3 Ablauf

Aus Benutzersicht ist folgender Ablauf des Programms zu erwarten:
(Storyboard)

	<p>Startsituation:</p> <p>Szenario muss gestartet werden (RUN >)</p> <p>1 Raumschiff steht zentriert am unteren Ende</p> <p>Mehrere UFOs sind oberhalb bereit in einer Formation</p>
	<p>Raumschiffaktivität:</p> <p>Mit Pfeilen kann das Raumschiff vertikal bewegt werden</p> <p>Mit der Space-Taste können Schüsse ausgelöst werden</p>

	Kollisionserkennung UFO wird bei Kollision mit Schuss zerstört Raumschiff wird bei Kollision mit UFO zerstört, resp. ein Leben abgezogen
	Anzeige Score Beim Abschuss eines UFOs erhält der User Punkte dazu Wird ein UFO nicht getroffen und befindet sich auf der Höhe des Raumschiffes, werden Punkte abgezogen

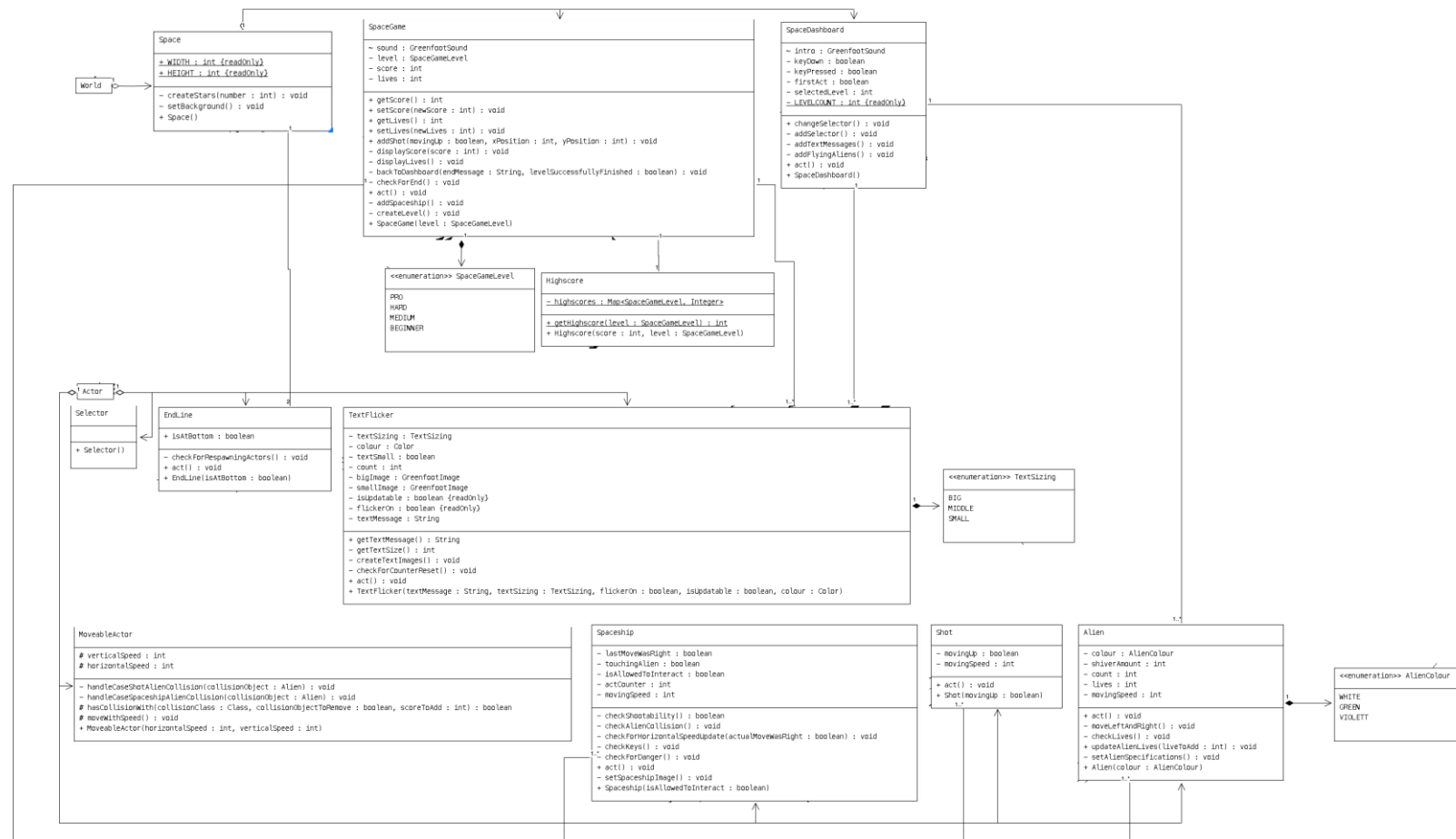
4 Systemdokumentation (**Meilenstein C: Aufgabe 3**)

Das erstellte Java-Projekt (Greenfoot-Szenario) ist hier detailliert abgelegt:

[**M226B_Space_Invaders_Franz_Forster.zip**](#)

4.1 Statisches Design: Klassendiagramm

Folgend die statische Struktur des Szenarios:



4.2 Umfang / Abgrenzung / Änderungen gegenüber Design

Aufgrund unten beschriebener Umstände sind Anpassungen des ursprünglichen Lösungsdesigns gemacht worden:

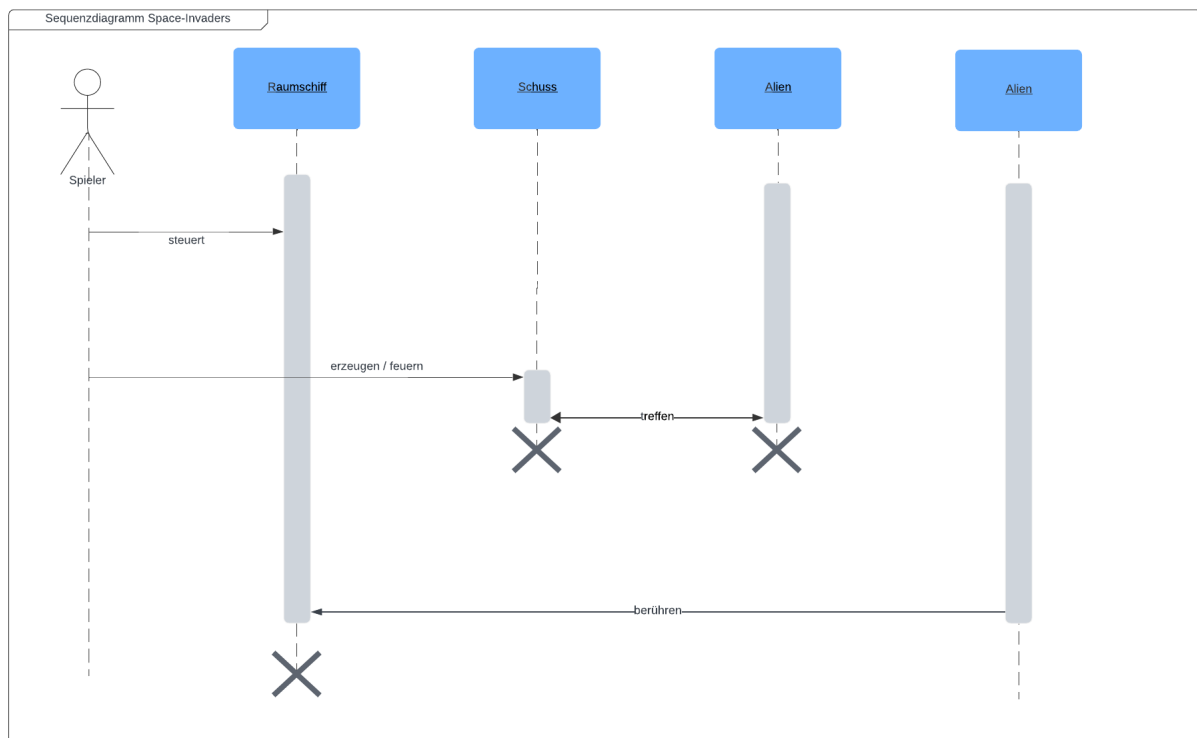
Score: Score wird abgezogen wenn der Schuss das Ende der Welt erreicht.

4.3 Funktionalität der Implementation.

Die einzelnen Funktionalitäten sind im Code ausführlich mit JavaDoc dokumentiert. Aus diesem Grund werden in diesem Abschnitt keine Funktionen detailliert beschrieben.

4.4 Dynamische Struktur: Sequenzdiagramm

Ein zentraler Ablauf eines UseCases ist im Folgenden dargestellt:



5 Bedienungsanleitung (Meilenstein C: Aufgabe 3)

1. Öffne das Szenario in Greenfoot
2. Klicke auf den grünen «Run» Knopf am unteren Ende des Greenfoot Programms
3. Wähle mit den Pfeiltasten (unten und oben) das gewünschte Level (Je höher das Level, desto schwieriger sind die Gegner)
4. Klicke «Enter», das Spiel startet sofort
5. Bewege das Raumschiff am unteren Ende mit den Pfeiltasten (links und rechts) nach links und rechts, um den Aliens auszuweichen
6. Drücke die Leertaste (Space) um einen Schuss abzufeuern und Aliens zu eliminieren.

Ziel

Dein Ziel ist es, mit möglichst vielen Leben und Punkten alle Gegner eliminiert zu haben.

In der linken, oberen Ecke ist deine Lebensanzeige. Sind keine Raumschiffe in dieser Anzeige vorhanden ist das Spiel zu Ende und du verlierst.

Sind alle Gegner eliminiert hast du das Spiel gewonnen und dein Score wird angezeigt.

Der Score erhöht sich mit jeder Elimination eines Aliens.

Der Score verringert sich, wenn:

- Ein Alien dein Raumschiff berührt und ein Leben abgezogen wird.
- Dein Schuss den Weltrand erreicht.

6 Testprotokoll (LB2 Meilenstein C2: Aufgabe 4)

Ausgefülltes Testprotokoll siehe Dokument

[**M226B_LB2_Testvorschrift_MS-C2_Franz_Forster.docx**](#)