Entwicklung Interaktiver Anwendungen II – Abschlussaufgabe Isabell Müller

Funktionale Analyse

Mit der Anwendung soll der User für einen kurzen Augenblick abgelenkt werden. Dieser soll mit der Anwendung per Mausklick interagieren. Ziel der Anwendung ist es, die Neugier der Benutzer anzuregen. Sie sollen Sammys Riff per Mausklick entdecken.

Sammy ist eine Schildkröte, die durch das Meer schwimmt. Der Nutzer soll sich die Anwendung selbst erklären, damit soll auf einen Überraschungseffekt abgezielt werden.

Die Anwendung hat kein konkretes Ende und ist theoretisch immer erweiterbar. Sobald sich der Nutzer langweilt, hat er die Möglichkeit die Anwendung zu verlassen.

Technische Analyse/ Umsetzung

Im HTML- Dokument habe ich ein unsichtbares Div Element erstellt. In dieses habe ich mein Hintergrundbild eingebunden. Aufgerufen wird dieses in meiner TypeScript Datei über ID und in meinem Canvas Element angezeigt.

Zunächst habe ich mir Gedanken über die Gestaltung meiner Schildkröte Sammy gemacht. Ihn habe ich mit Zeichenbefehlen auf dem Canvas erzeugt. Um ihn zu animieren benötige ich die Funktionen move und draw.

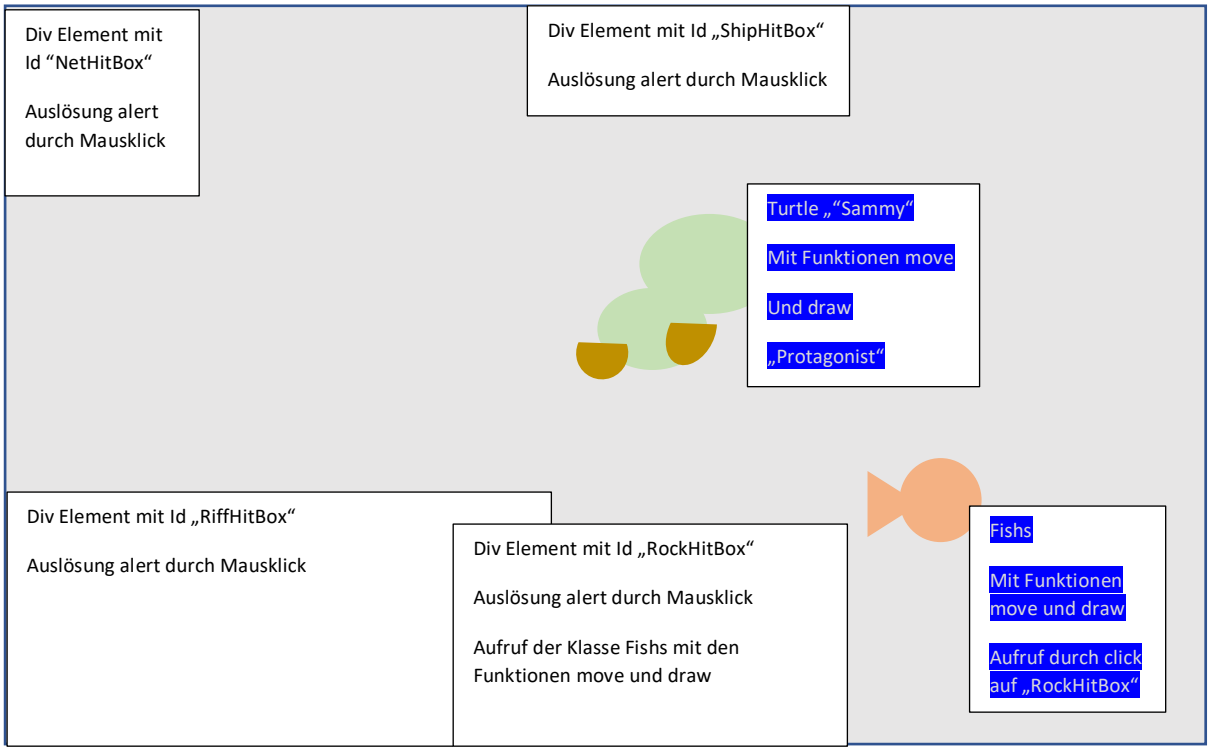
Diese Funktionen/ Daten brauche ich ebenfalls für weitere Zeichenelemente. Deshalb bietet sich die Erstellung von Klassen an. Meine Superklasse mit dem Namen MovingObjects, beinhaltet diese Methoden. Elemente der Subklasse sind Turtle, Ship und Fishs. Danach rufe ich meine animate Funktion auf welche erst später deklariert wird.

Ich möchte Funktionen per Mausklick auf bestimmten Canvas-Zeichnungen auslösen. Dafür habe ich mir die Positionierung und das Flussverhalten von HTML Div-Elementen zur Hilfe genommen und per CSS an die benötigten Stellen geschoben.

Anschließend habe ich die benötigten Klickfunktionen für meine Div-Boxen beziehungsweise HitBoxen initialisiert.

Am Ende befülle ich meine animate Funktion in dem ich meine Objekte der Superklasse MovingObjects erzeuge und der Funktion ein Timeout mitgebe.

Skizze/ Veranschaulichung



Klassendiagramme

MovingObjects
x: number; y:number;

Constructor (_x, _y)
move(): void;
draw(): void;

Turtle (extends MovingObjects)
dx: number; dy: number;

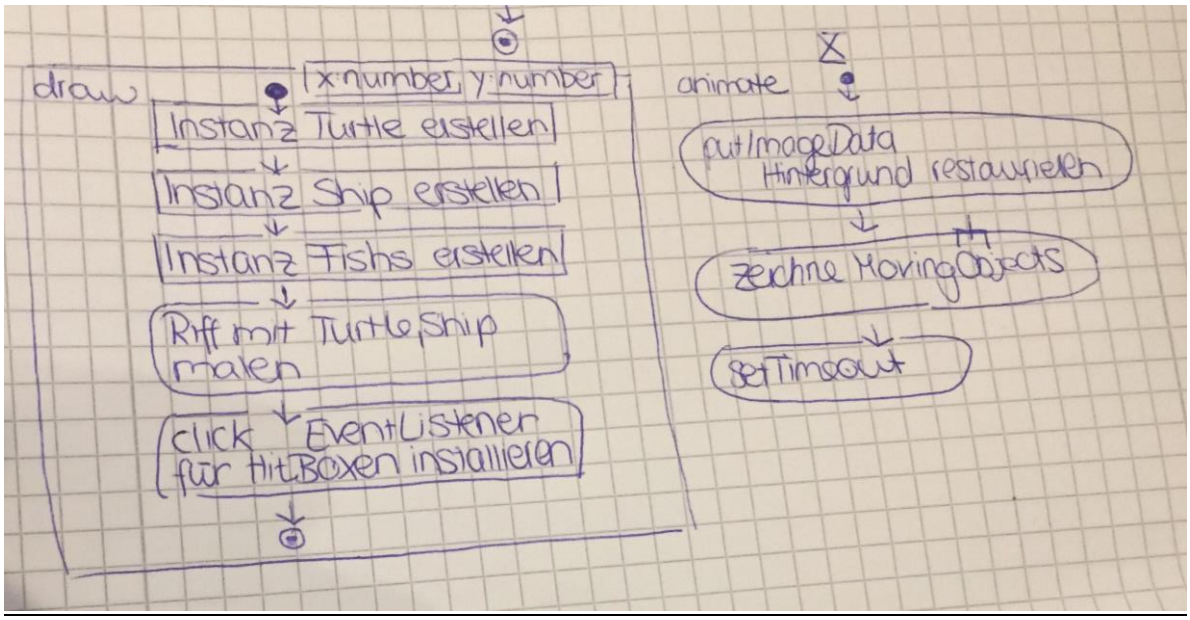
MovingObjects (_x, _y)
move(): void;
draw(): void;

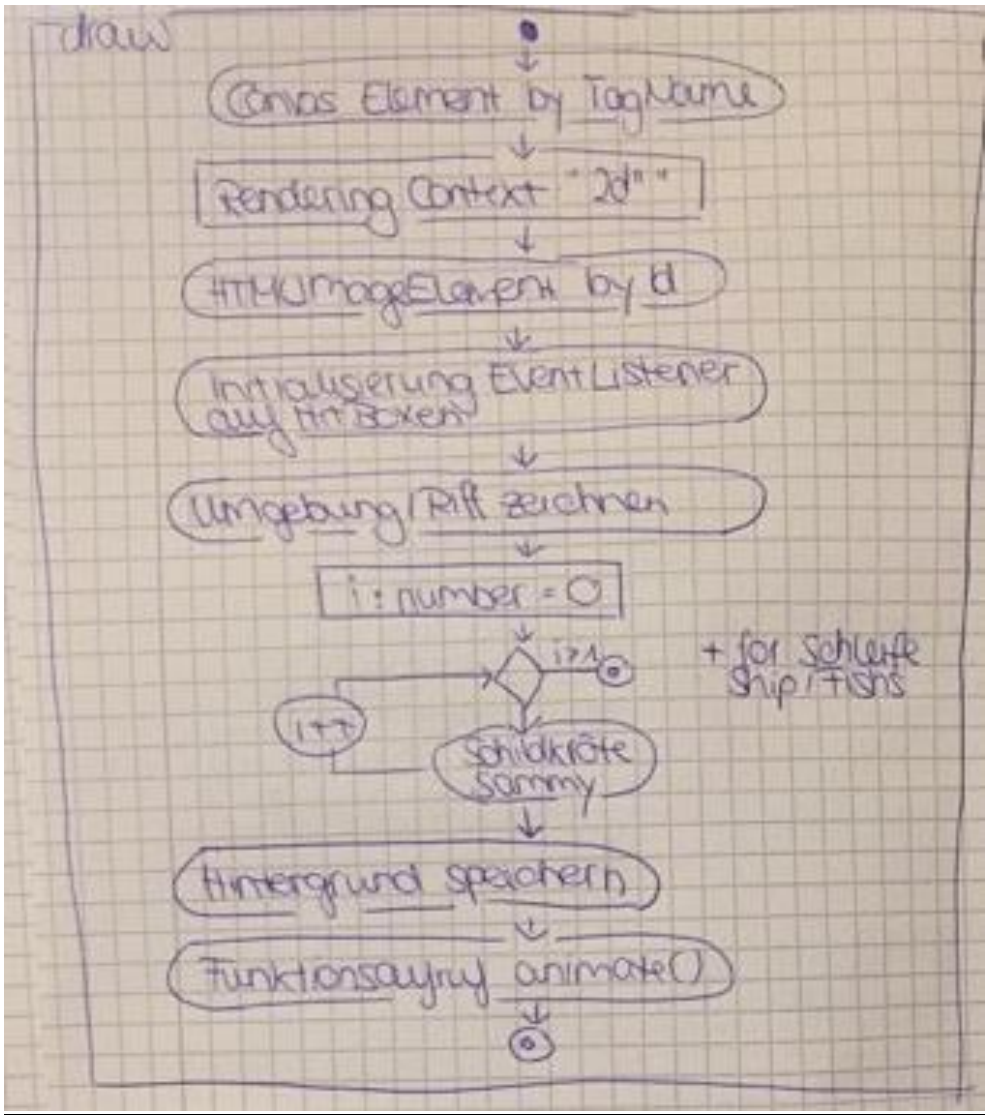
Fishs (extends MovingObjects)
dx: number; dy: number; color: string;

MovingObjects (_x, _y)
move(): void;
draw(): void;

Ship (extends MovingObjects)
MovingObjects (_x, _y) draw(): void;

Aktivitätsdiagramme





Ereignisdiagramme