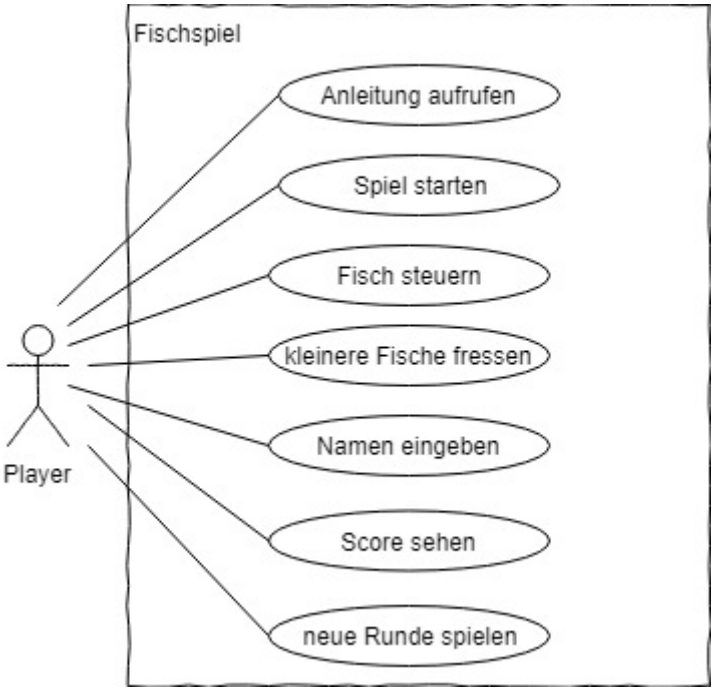
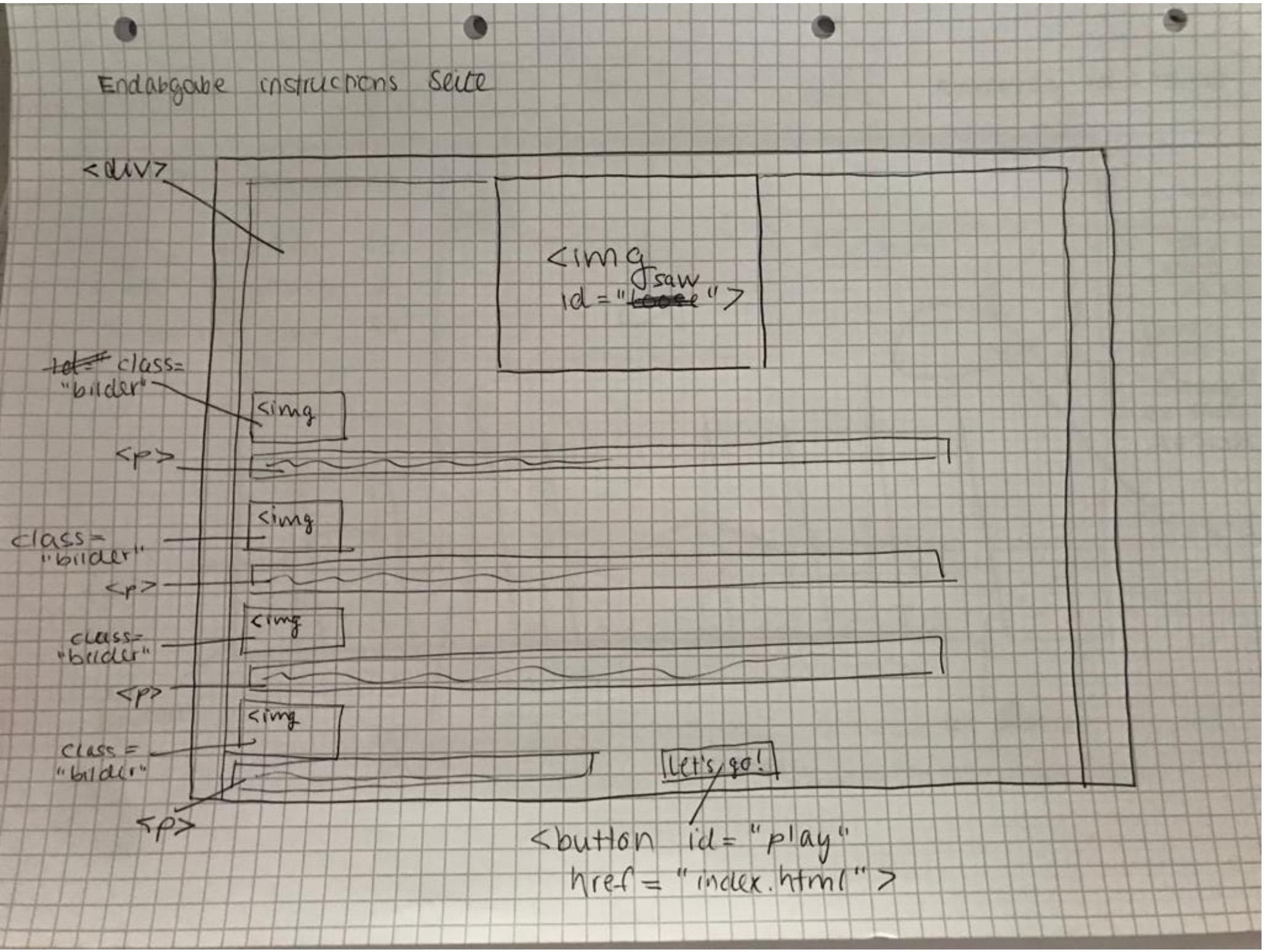
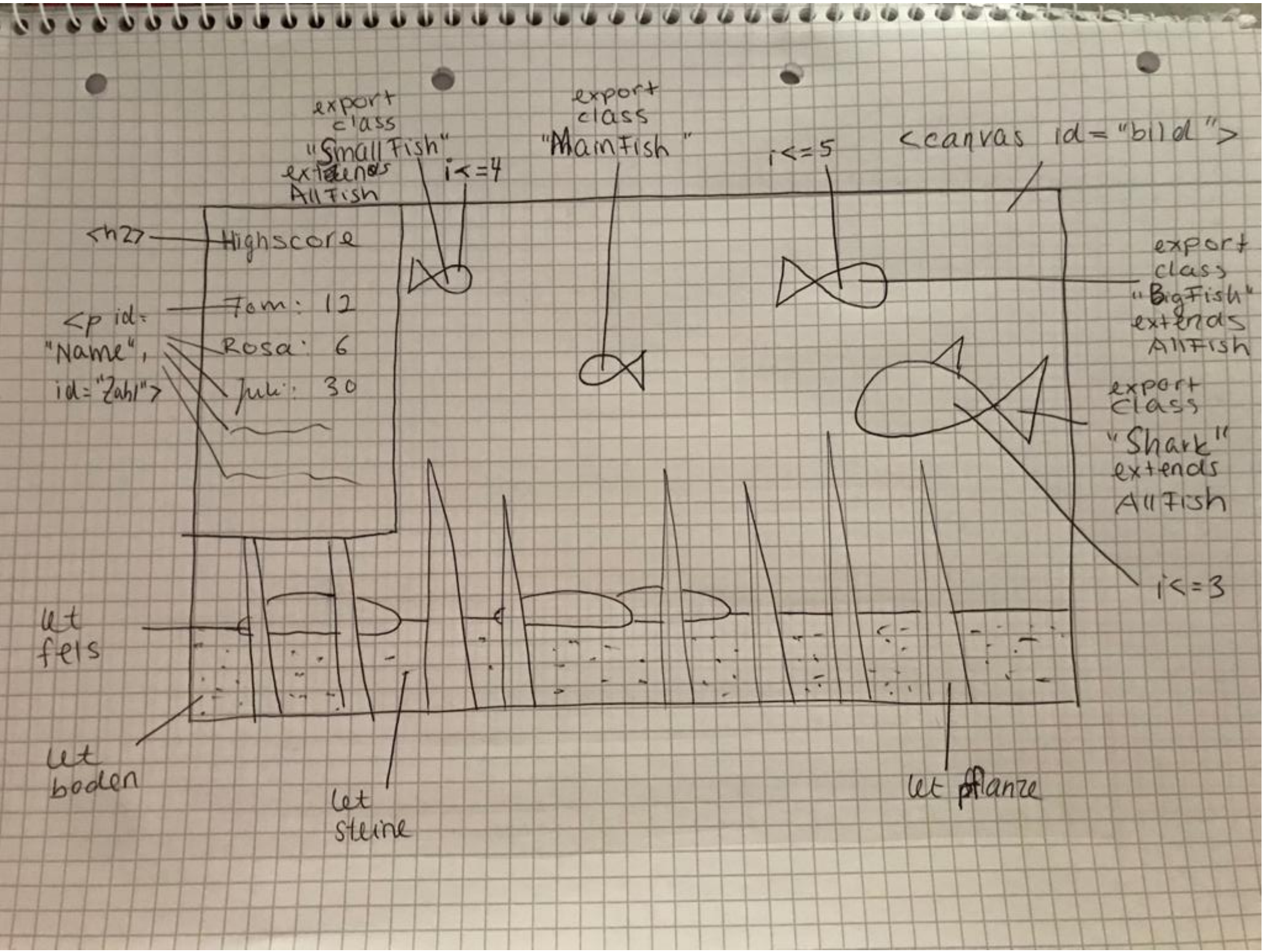
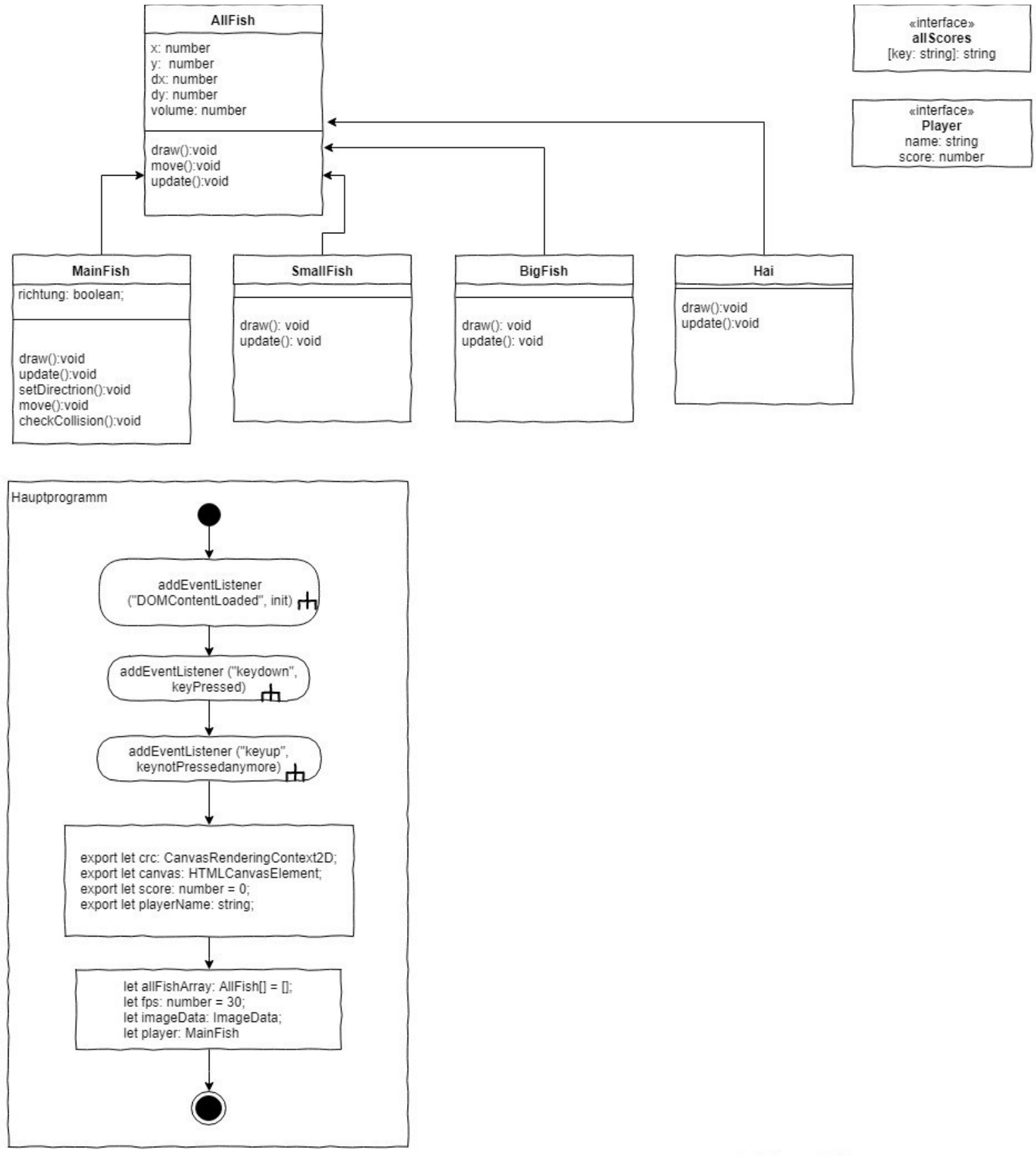
# Technische Analyse

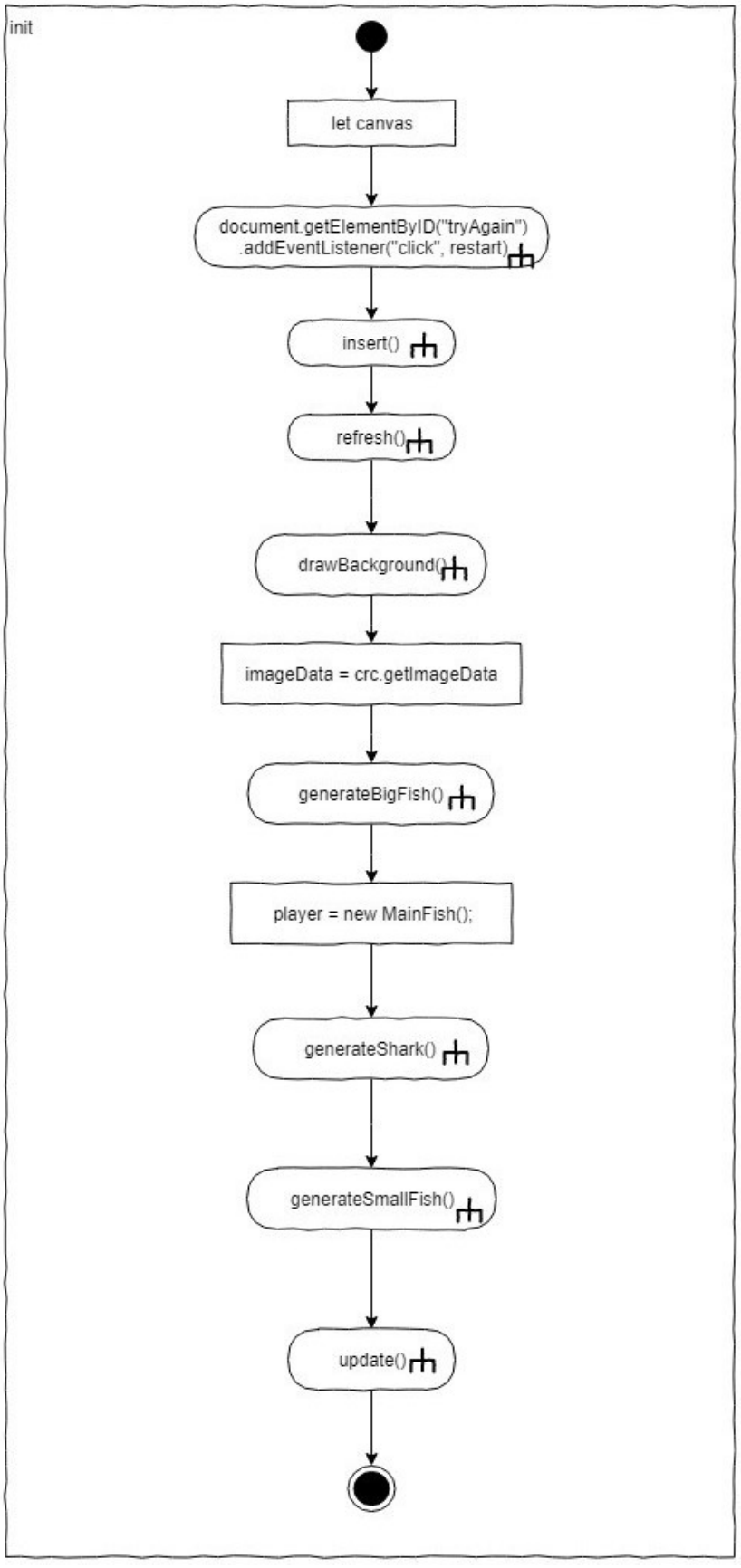


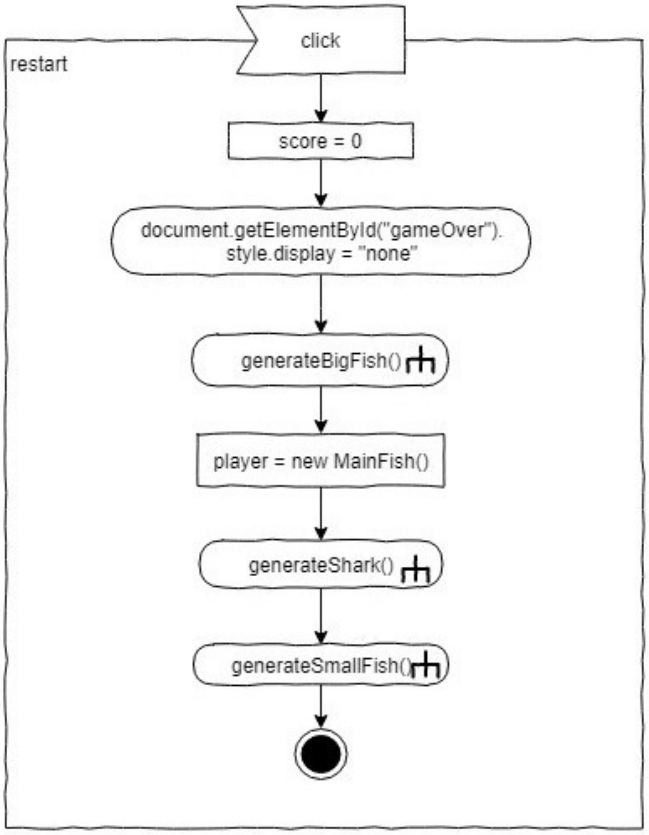
Endabgabe instructions Seite

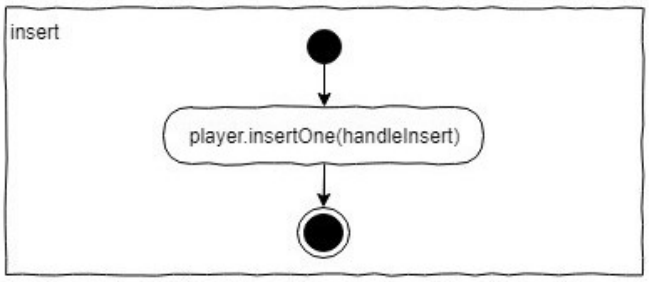


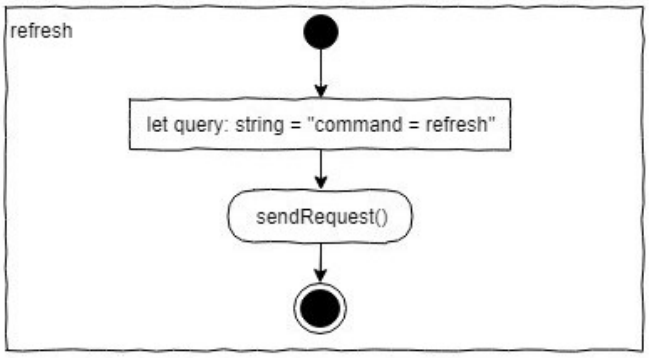

export class "SmallFish" extends AllFish
i<=4
export class "MainFish"
i<=5
<canvas id="bild">
<h2>
Highscore
Tom: 12
Rosa: 6
Juli: 30
<p id="Name", id="Zahl">
export class "BigFish" extends AllFish
export class "Shark" extends AllFish
i<=3
let fels
let boden
let steine
let pflanze

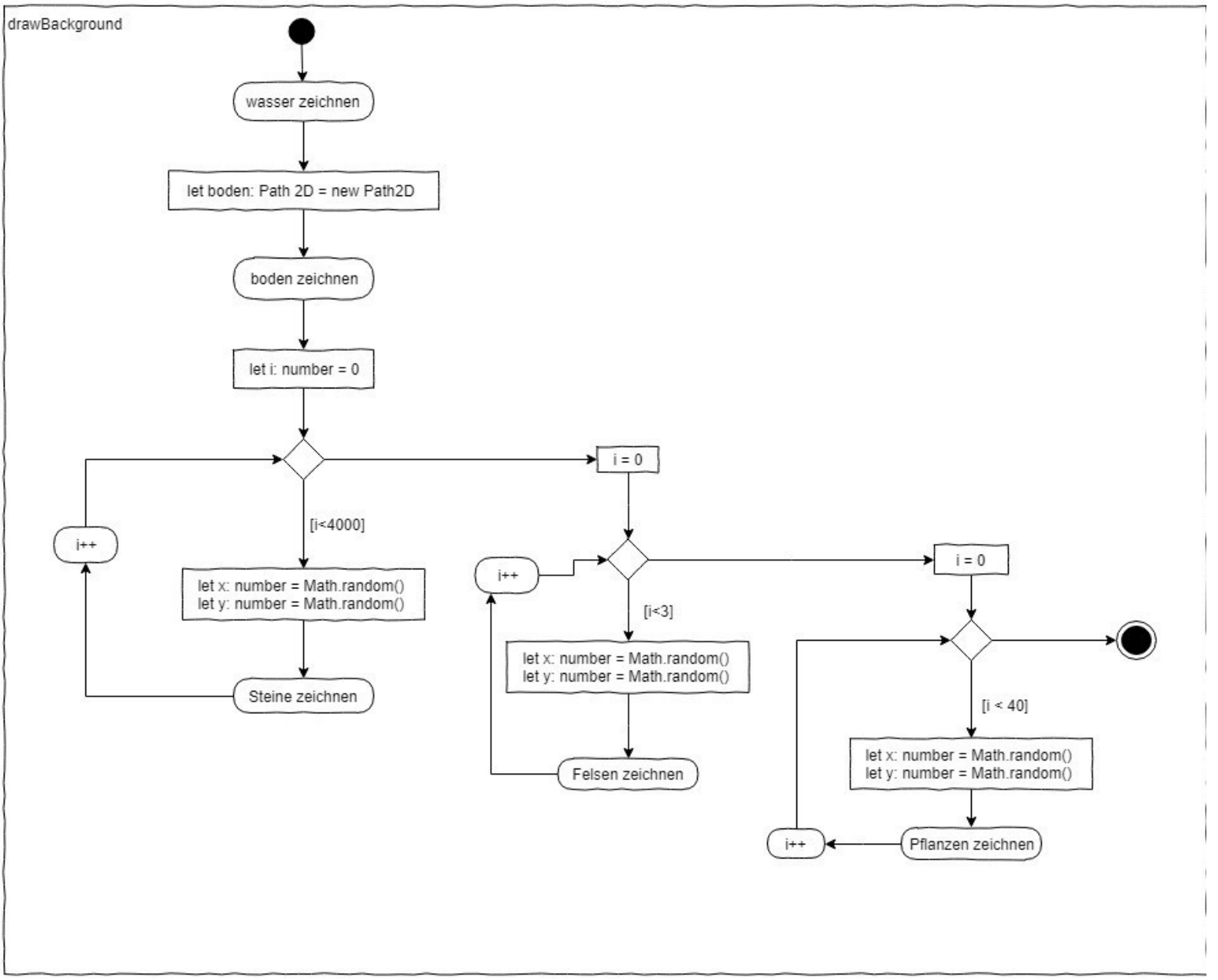

AllFish
x: number
y: number
dx: number
dy: number
volume: number
draw():void
move():void
update():void
«interface»
allScores
[key: string]: string
«interface»
Player
name: string
score: number
MainFish
richtung: boolean;
draw():void
update():void
setDirectrion():void
move():void
checkCollision():void
SmallFish
draw(): void
update(): void
BigFish
draw(): void
update(): void
Hai
draw():void
update():void
Hauptprogramm
addEventListener
("DOMContentLoaded", init)
addEventListener ("keydown",
keyPressed)
addEventListener ("keyup",
keynotPressedanymore)
export let crc: CanvasRenderingContext2D;
export let canvas: HTMLCanvasElement;
export let score: number = 0;
export let playerName: string;
let allFishArray: AllFish[] = [];
let fps: number = 30;
let imageData: ImageData;
let player: MainFish

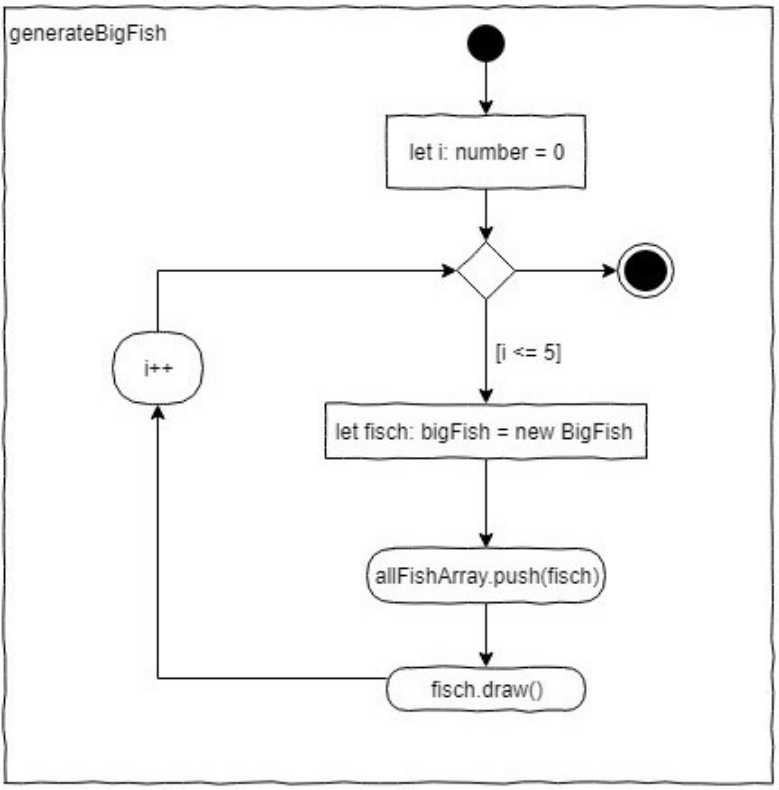

init
let canvas
document.getElementByID("tryAgain")
.addEventListener("click", restart)
insert()
refresh()
drawBackground()
imageData = crc.getImageData
generateBigFish()
player = new MainFish();
generateShark()
generateSmallFish()
update()

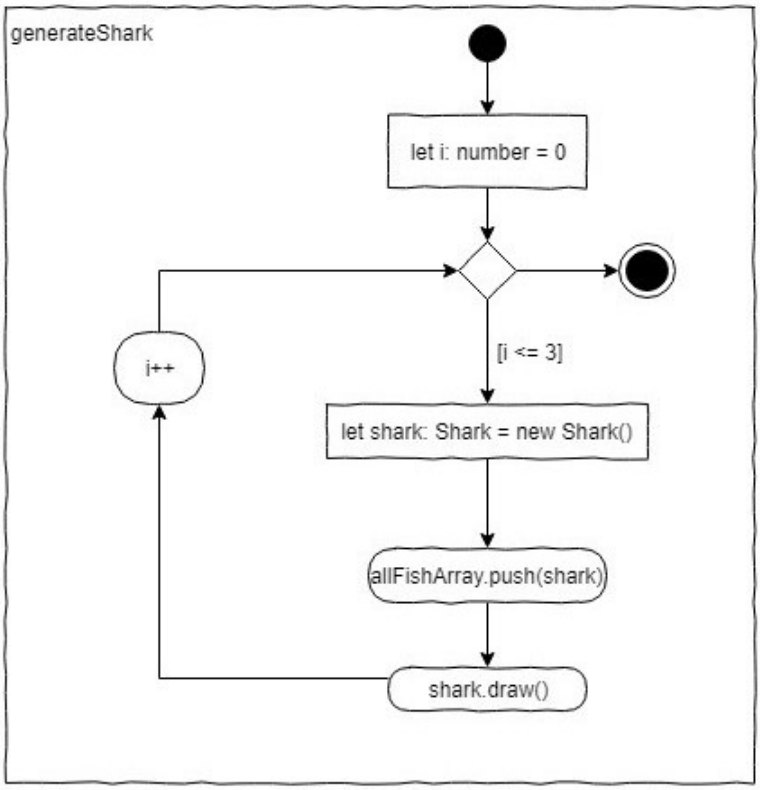

restart
click
score = 0
document.getElementById("gameOver").
style.display = "none"
generateBigFish()
player = new MainFish()
generateShark()
generateSmallFish()

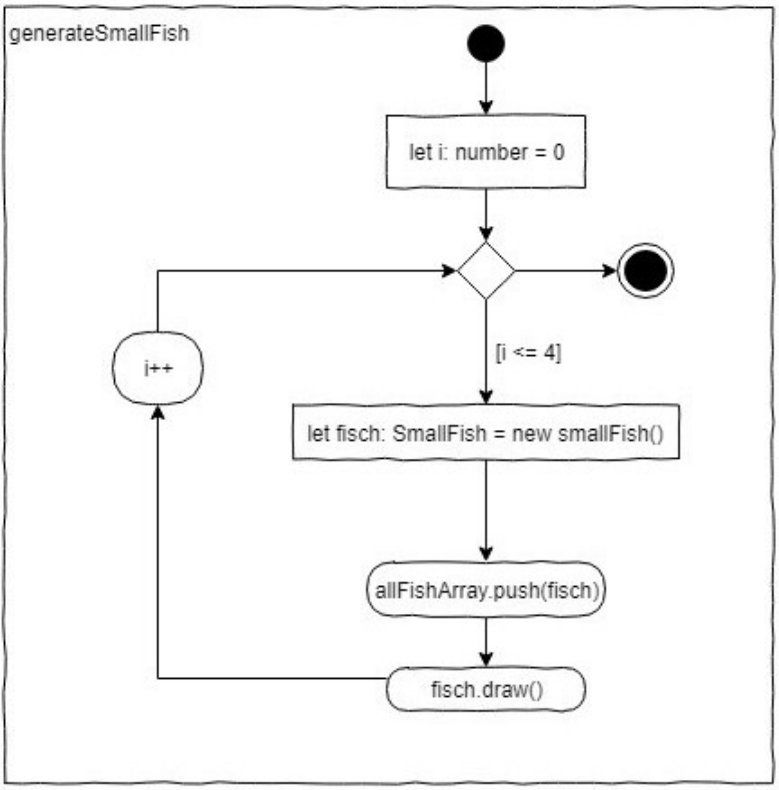

insert
player.insertOne(handleInsert)

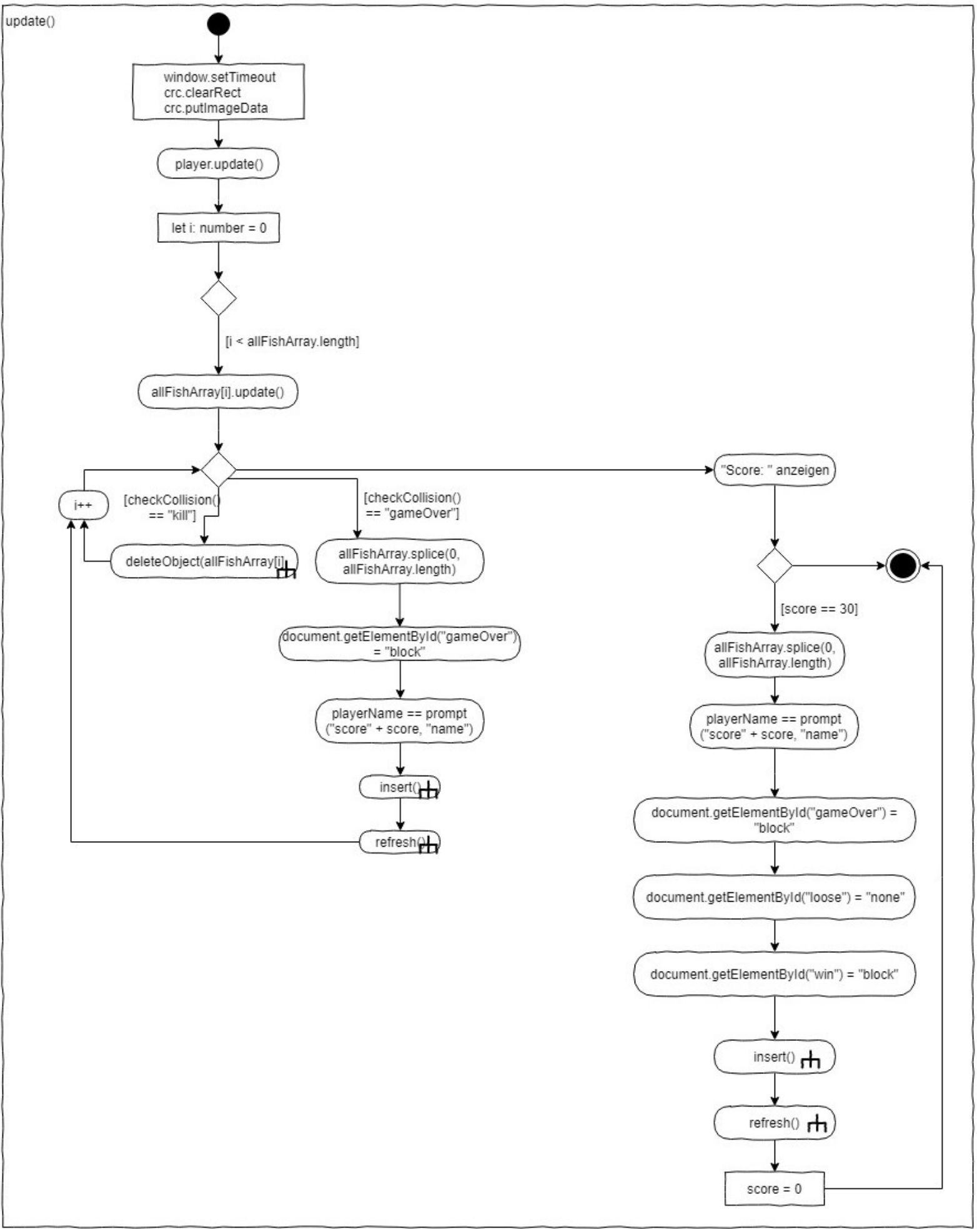

refresh
let query: string = "command = refresh"
sendRequest()

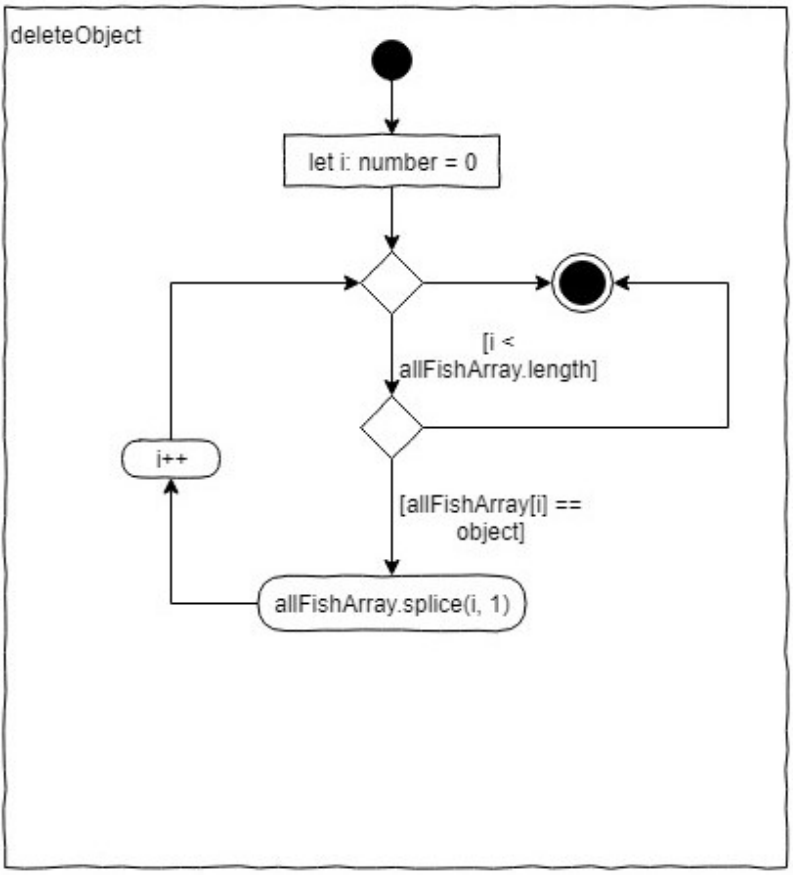

drawBackground
wasser zeichnen
let boden: Path 2D = new Path2D
boden zeichnen
let i: number = 0
i = 0
[i<4000]
i++
let x: number = Math.random()
let y: number = Math.random()
Steine zeichnen
i++
i = 0
[i<3]
let x: number = Math.random()
let y: number = Math.random()
Felsen zeichnen
i = 0
[i < 40]
let x: number = Math.random()
let y: number = Math.random()
Pflanzen zeichnen
i++

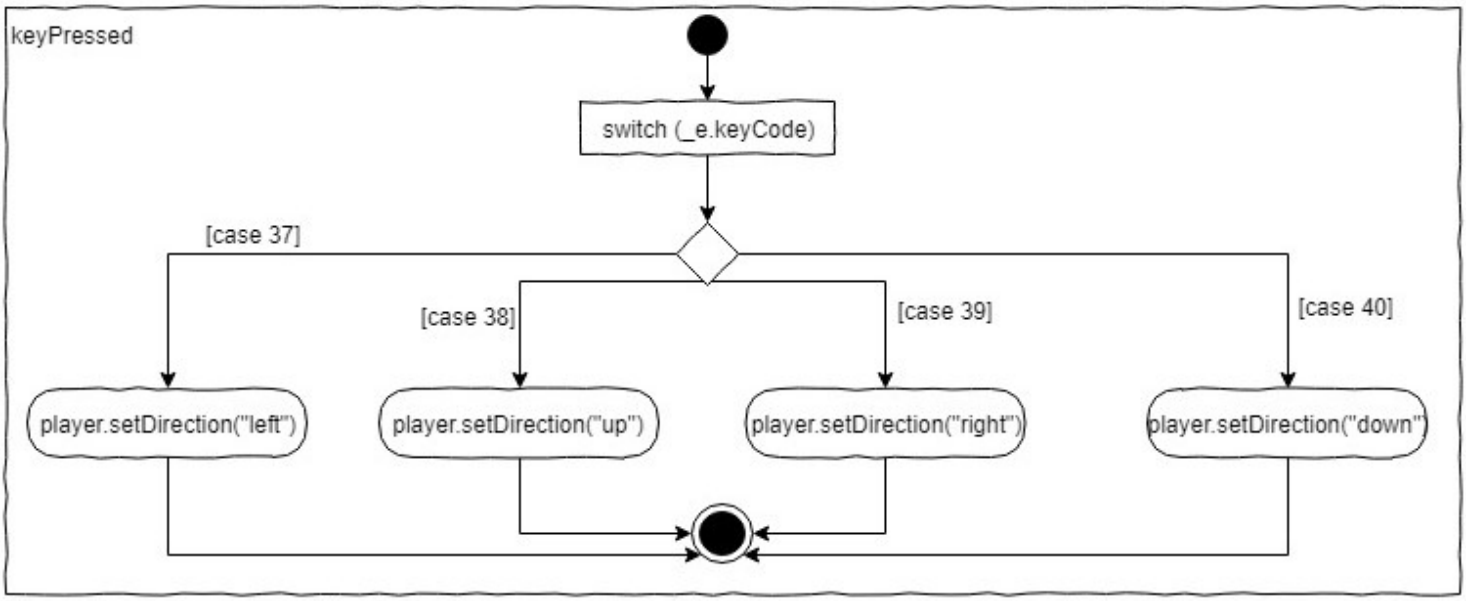

generateBigFish
let i: number = 0
[i <= 5]
i++
let fisch: bigFish = new BigFish
allFishArray.push(fisch)
fisch.draw()

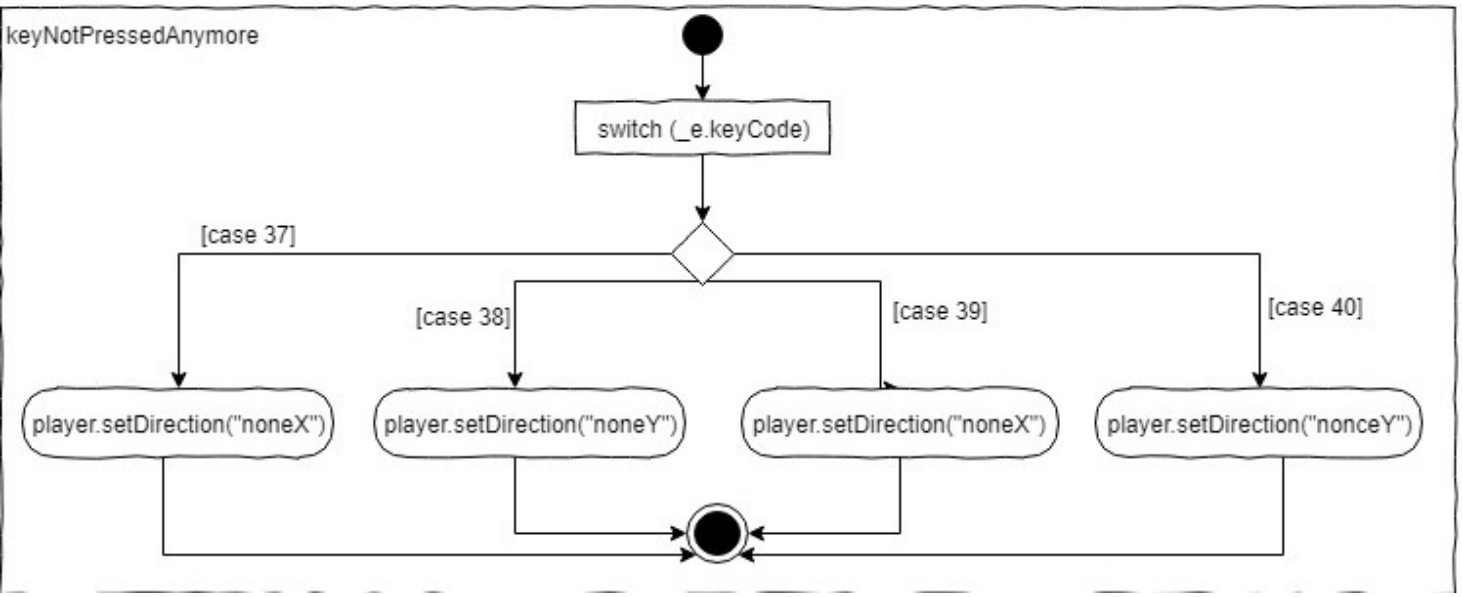

generateShark
let i: number = 0
[i <= 3]
i++
let shark: Shark = new Shark()
allFishArray.push(shark)
shark.draw()


generateSmallFish
let i: number = 0
[i <= 4]
i++
let fisch: SmallFish = new smallFish()
allFishArray.push(fisch)
fisch.draw()

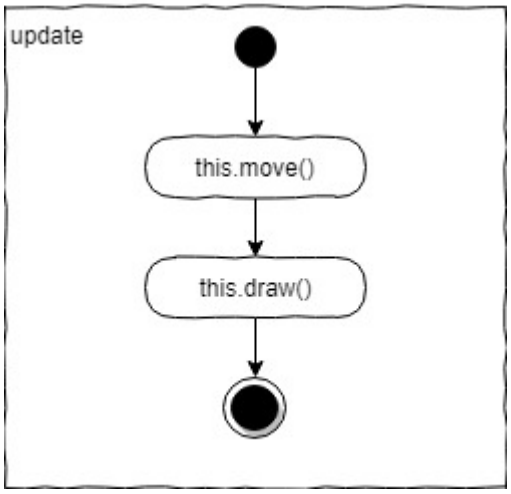

update()
window.setTimeout
crc.clearRect
crc.putImageData
player.update()
let i: number = 0
[i < allFishArray.length]
allFishArray[i].update()
i++
[checkCollision()
== "kill"]
deleteObject(allFishArray[i])
[checkCollision()
== "gameOver"]
allFishArray.splice(0,
allFishArray.length)
document.getElementById("gameOver")
= "block"
playerName == prompt
("score" + score, "name")
insert()
refresh()
"Score: " anzeigen
[score == 30]
allFishArray.splice(0,
allFishArray.length)
playerName == prompt
("score" + score, "name")
document.getElementById("gameOver") =
"block"
document.getElementById("loose") = "none"
document.getElementById("win") = "block"
insert()
refresh()
score = 0

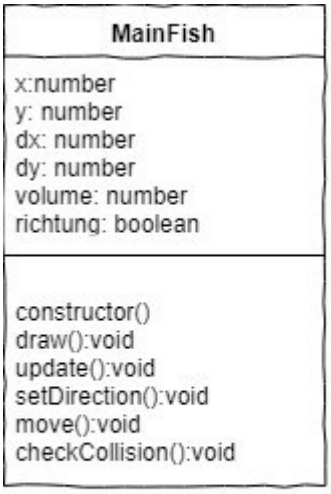

deleteObject
let i: number = 0
[i <
allFishArray.length]
i++
[allFishArray[i] ==
object]
allFishArray.splice(i, 1)

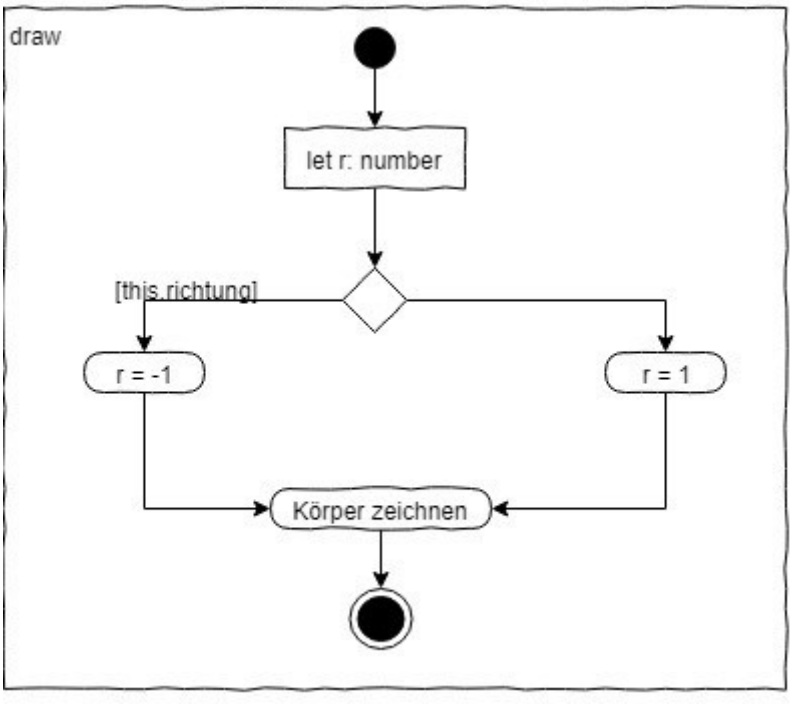

keyPressed
switch (_e.keyCode)
[case 37]
[case 38]
[case 39]
[case 40]
player.setDirection("left")
player.setDirection("up")
player.setDirection("right")
player.setDirection("down")

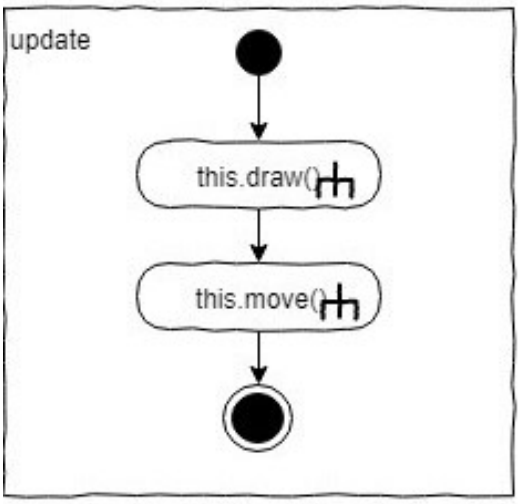

keyNotPressedAnymore
switch (_e.keyCode)
[case 37]
[case 38]
[case 39]
[case 40]
player.setDirection("noneX")
player.setDirection("noneY")
player.setDirection("noneX")
player.setDirection("nonceY")

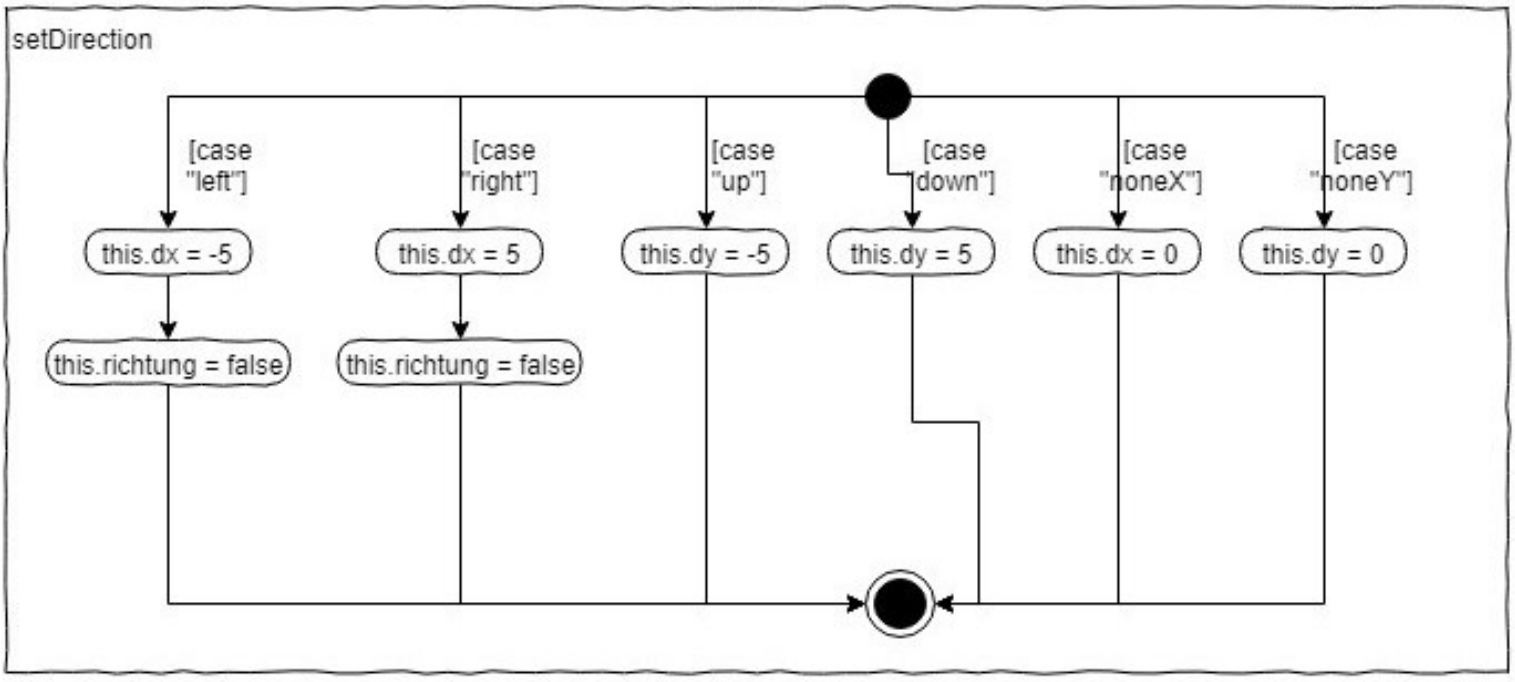

AllFish
volume: number
x: number
y: number
dx: number
dy: number
draw():void
update():void
move():void

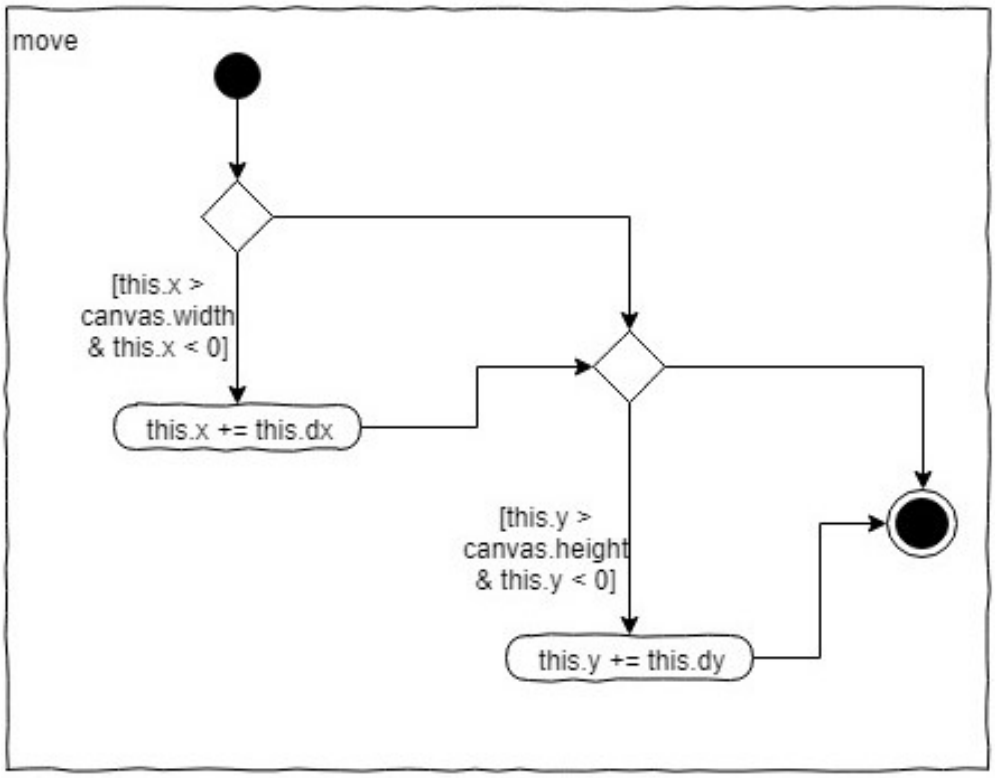

update
this.move()
this.draw()

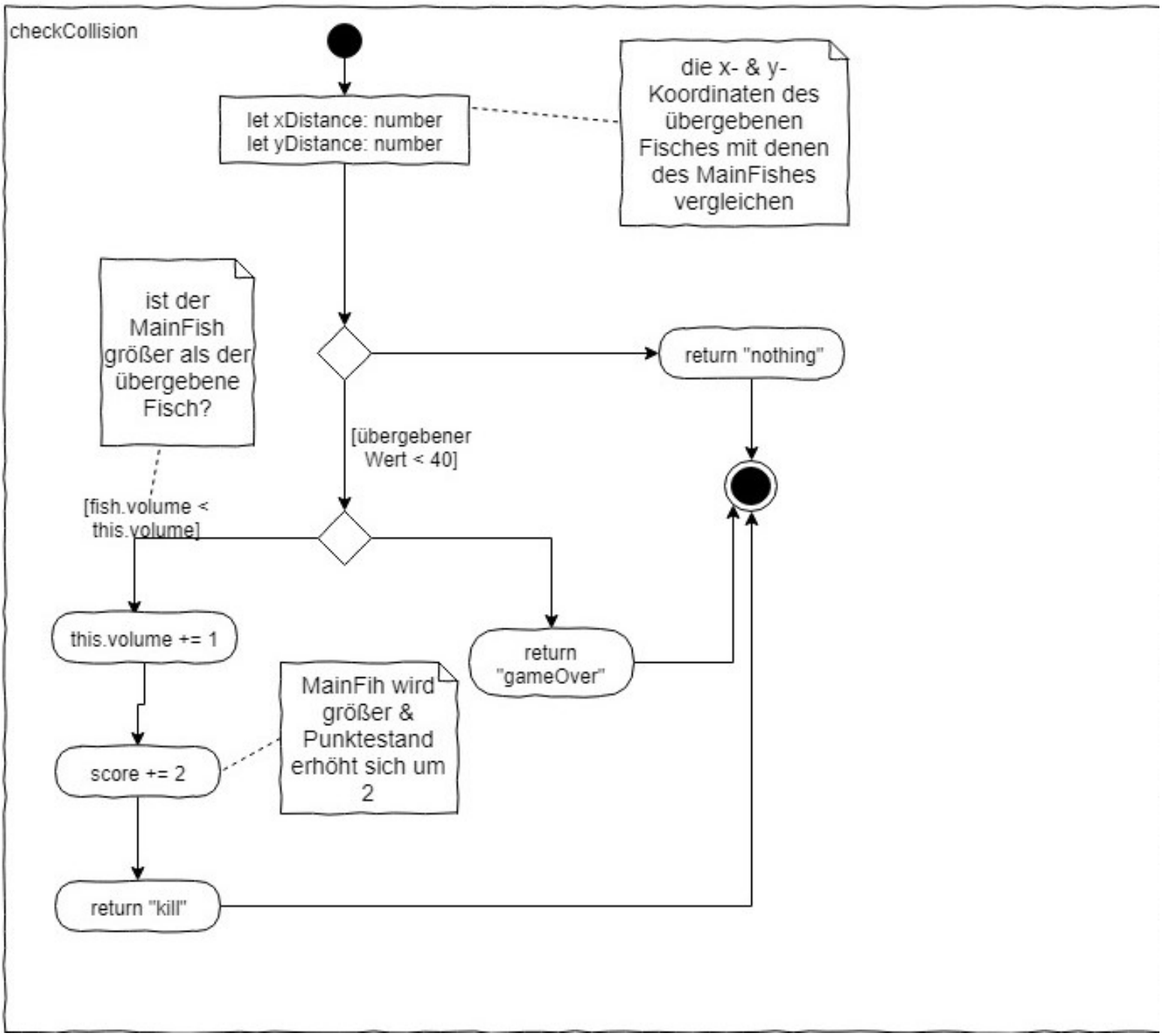

MainFish
x:number
y: number
dx: number
dy: number
volume: number
richtung: boolean
constructor()
draw():void
update():void
setDirection():void
move():void
checkCollision():void

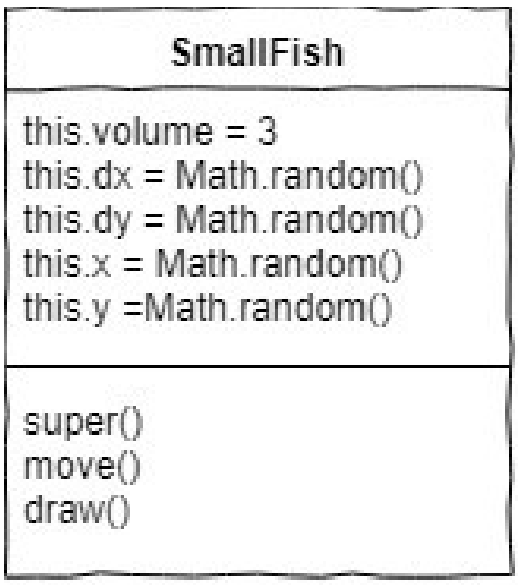

draw
let r: number
[this.richtung]
r = -1
r = 1
Körper zeichnen

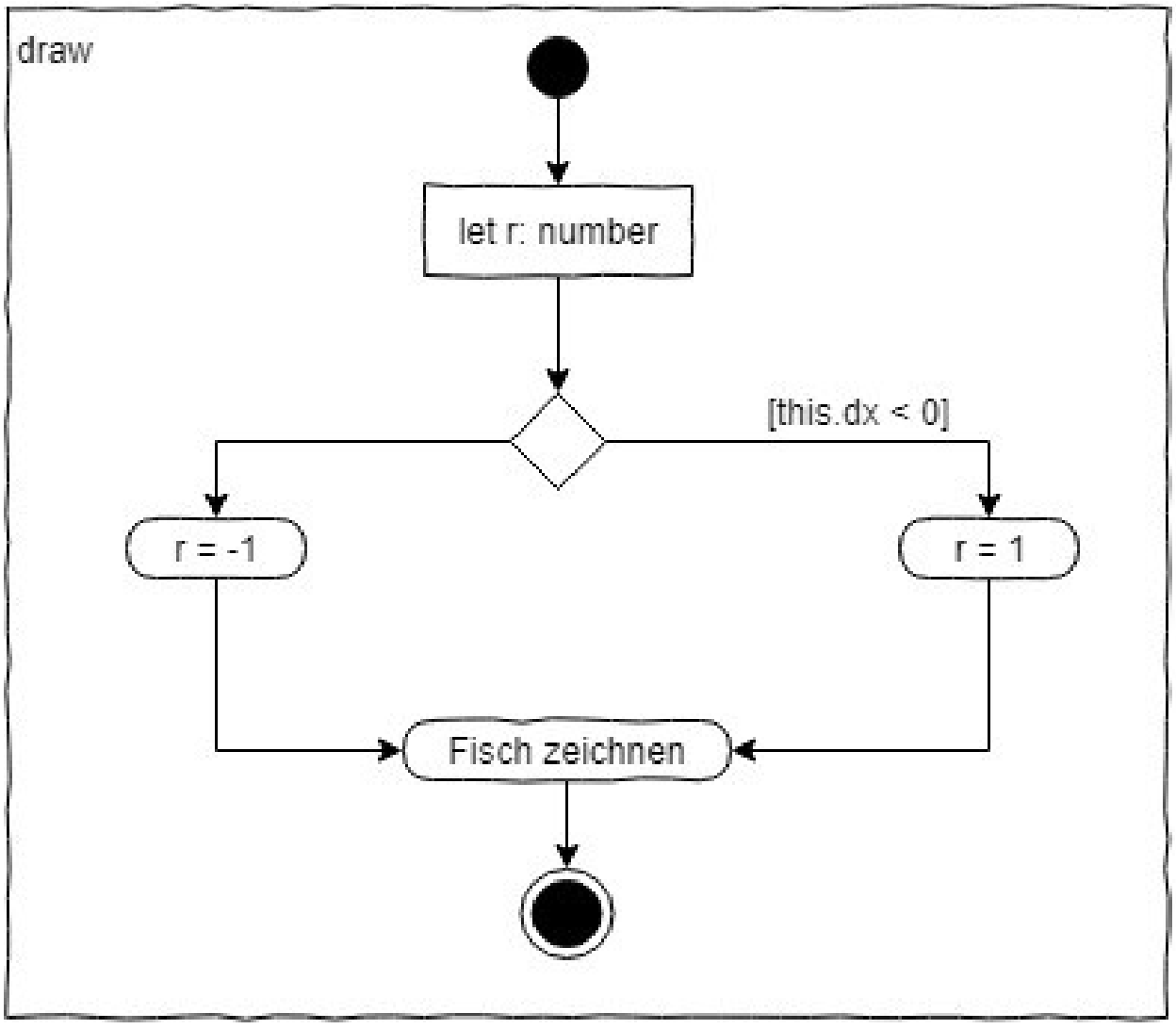

update
this.draw()
this.move()

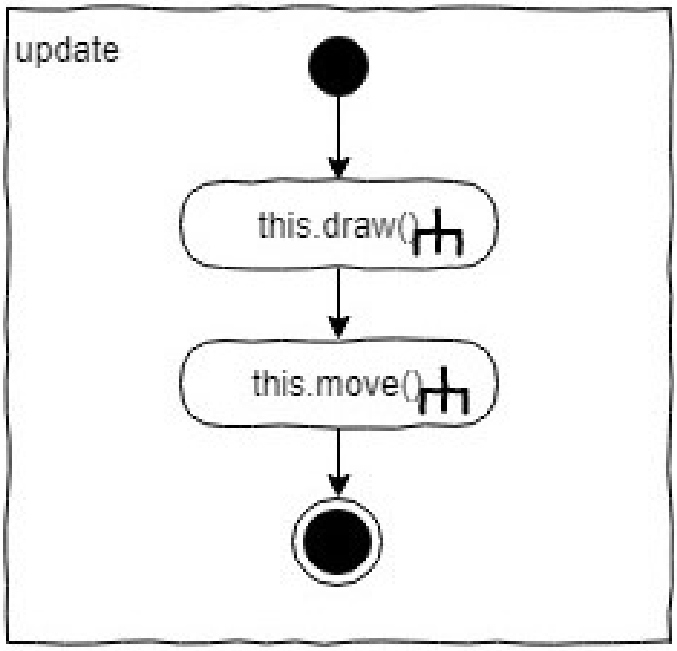

setDirection
[case "left"]
[case "right"]
[case "up"]
[case down"]
[case "noneX"]
[case "noneY"]
this.dx = -5
this.dx = 5
this.dy = -5
this.dy = 5
this.dx = 0
this.dy = 0
this.richtung = false
this.richtung = false

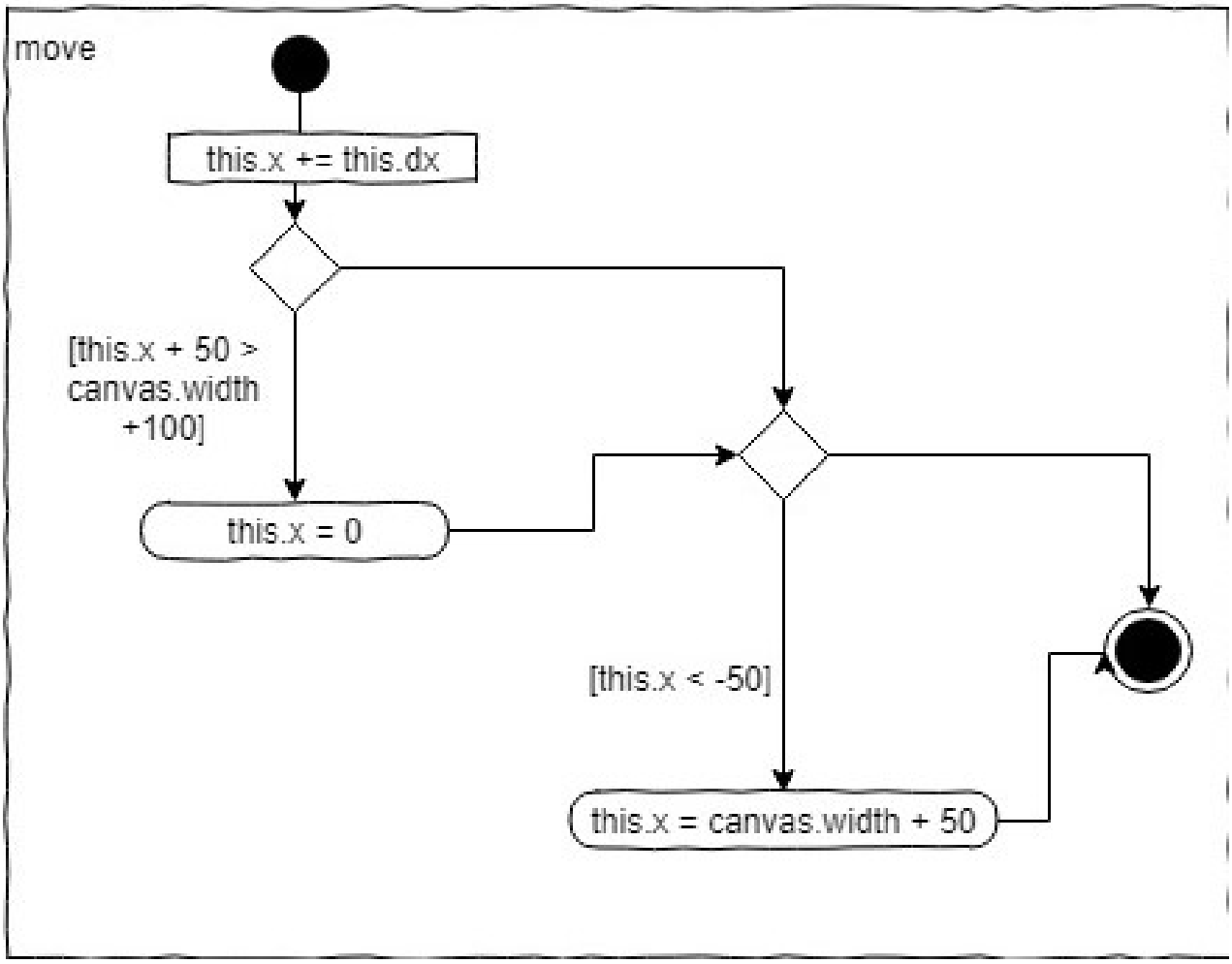

move
[this.x > canvas.width & this.x < 0]
this.x += this.dx
[this.y > canvas.height & this.y < 0]
this.y += this.dy

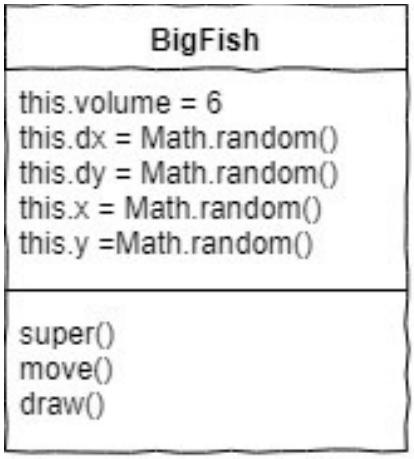

checkCollision
let xDistance: number
let yDistance: number
die x- & y-Koordinaten des übergebenen Fisches mit denen des MainFishes vergleichen
ist der MainFish größer als der übergebene Fisch?
return "nothing"
[übergebener Wert < 40]
[fish.volume < this.volume]
this.volume += 1
return "gameOver"
MainFih wird größer & Punktestand erhöht sich um 2
score += 2
return "kill"

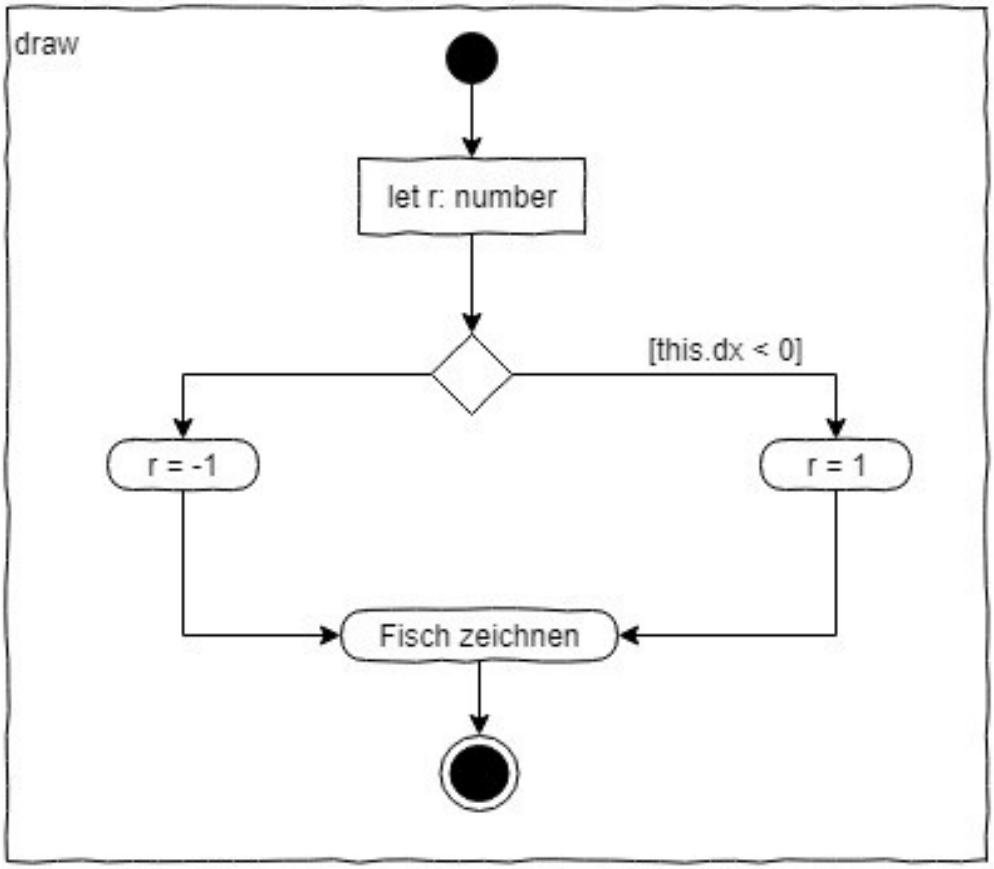

SmallFish
this.volume = 3
this.dx = Math.random()
this.dy = Math.random()
this.x = Math.random()
this.y =Math.random()
super()
move()
draw()

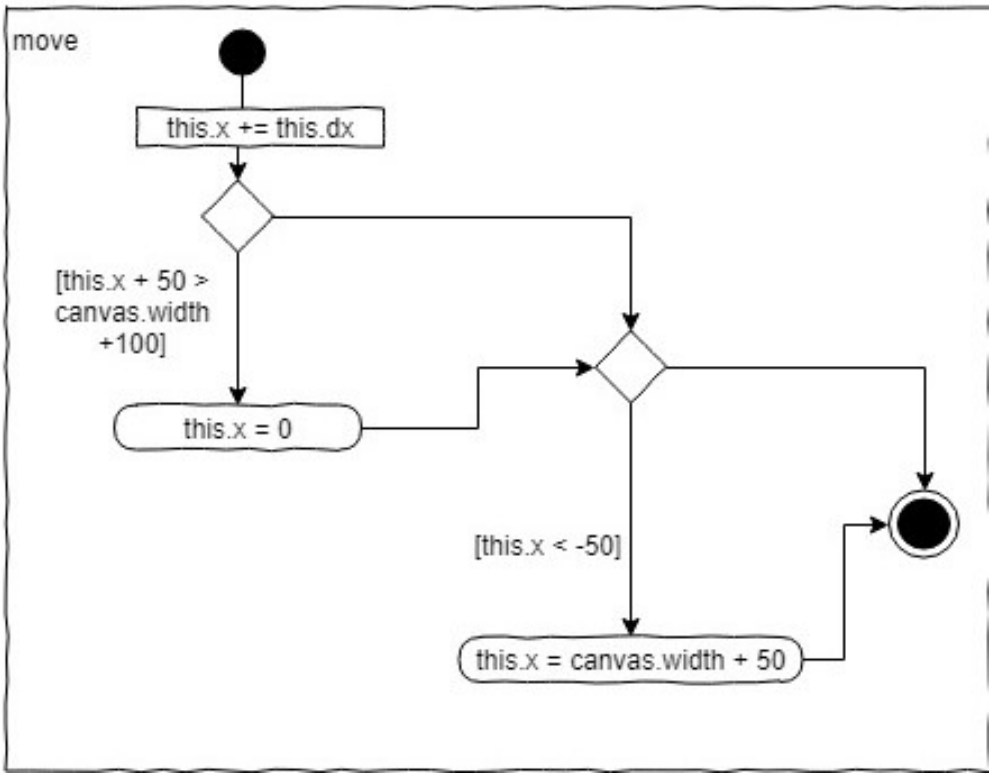

draw
let r: number
[this.dx < 0]
r = -1
r = 1
Fisch zeichnen

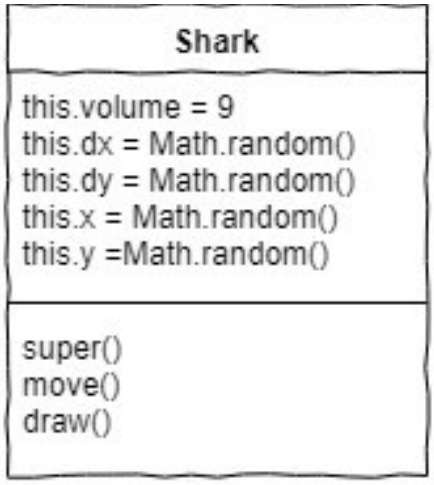

update
this.draw()
this.move()

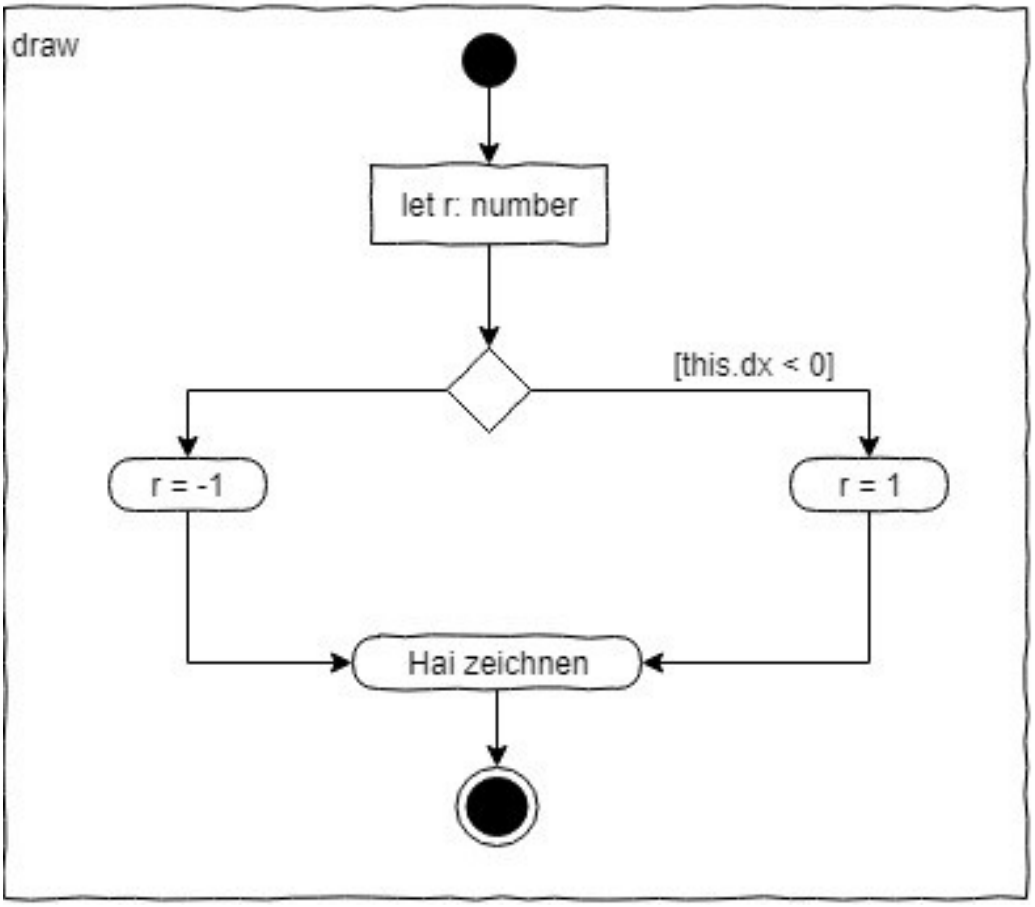

move
this.x += this.dx
[this.x + 50 > canvas.width +100]
this.x = 0
[this.x < -50]
this.x = canvas.width + 50

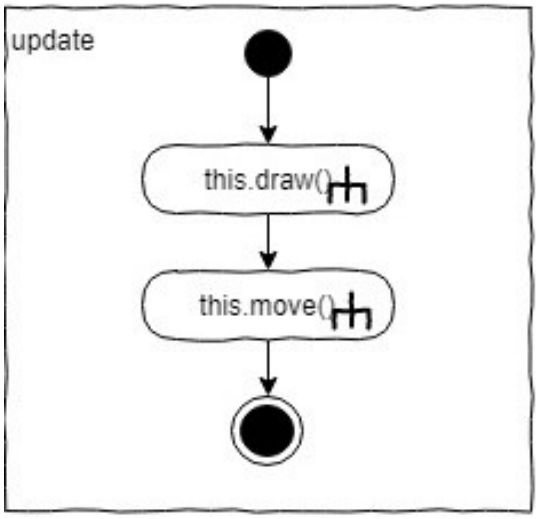

BigFish
this.volume = 6
this.dx = Math.random()
this.dy = Math.random()
this.x = Math.random()
this.y =Math.random()
super()
move()
draw()

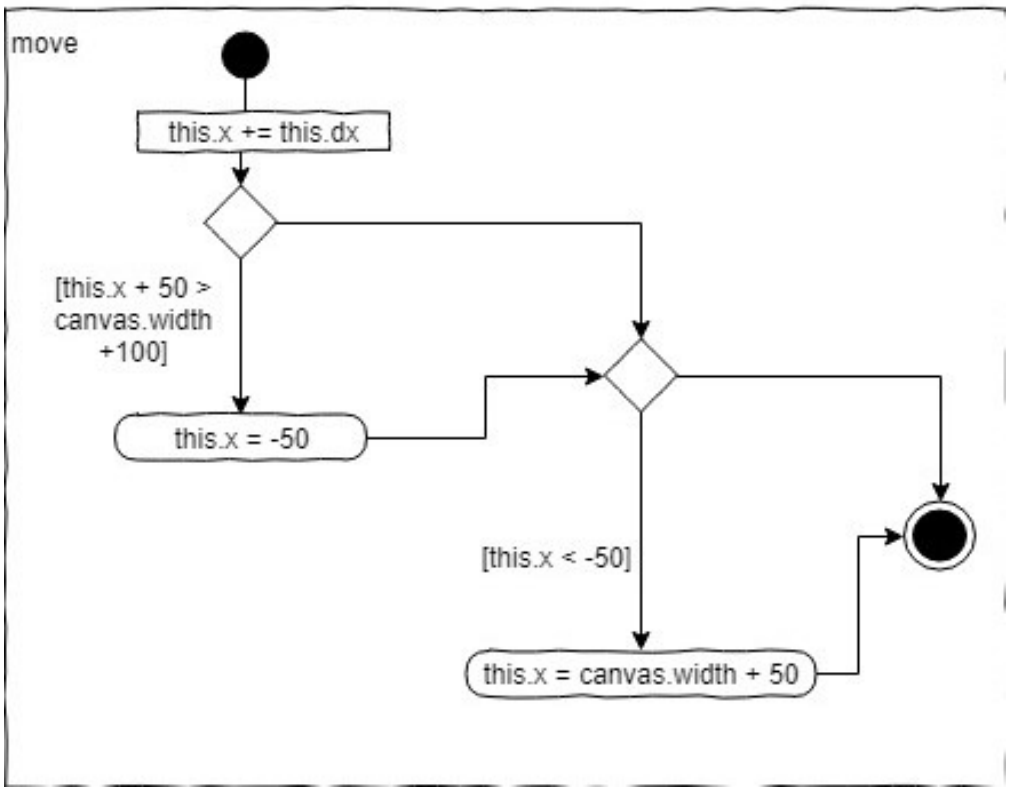

draw
let r: number
[this.dx < 0]
r = -1
r = 1
Fisch zeichnen


move
this.x += this.dx
[this.x + 50 > canvas.width +100]
this.x = 0
[this.x < -50]
this.x = canvas.width + 50


Shark
this.volume = 9
this.dx = Math.random()
this.dy = Math.random()
this.x = Math.random()
this.y =Math.random()
super()
move()
draw()


draw
let r: number
[this.dx < 0]
r = -1
r = 1
Hai zeichnen


update
this.draw()
this.move()


move
this.x += this.dx
[this.x + 50 > canvas.width +100]
this.x = -50
[this.x < -50]
this.x = canvas.width + 50