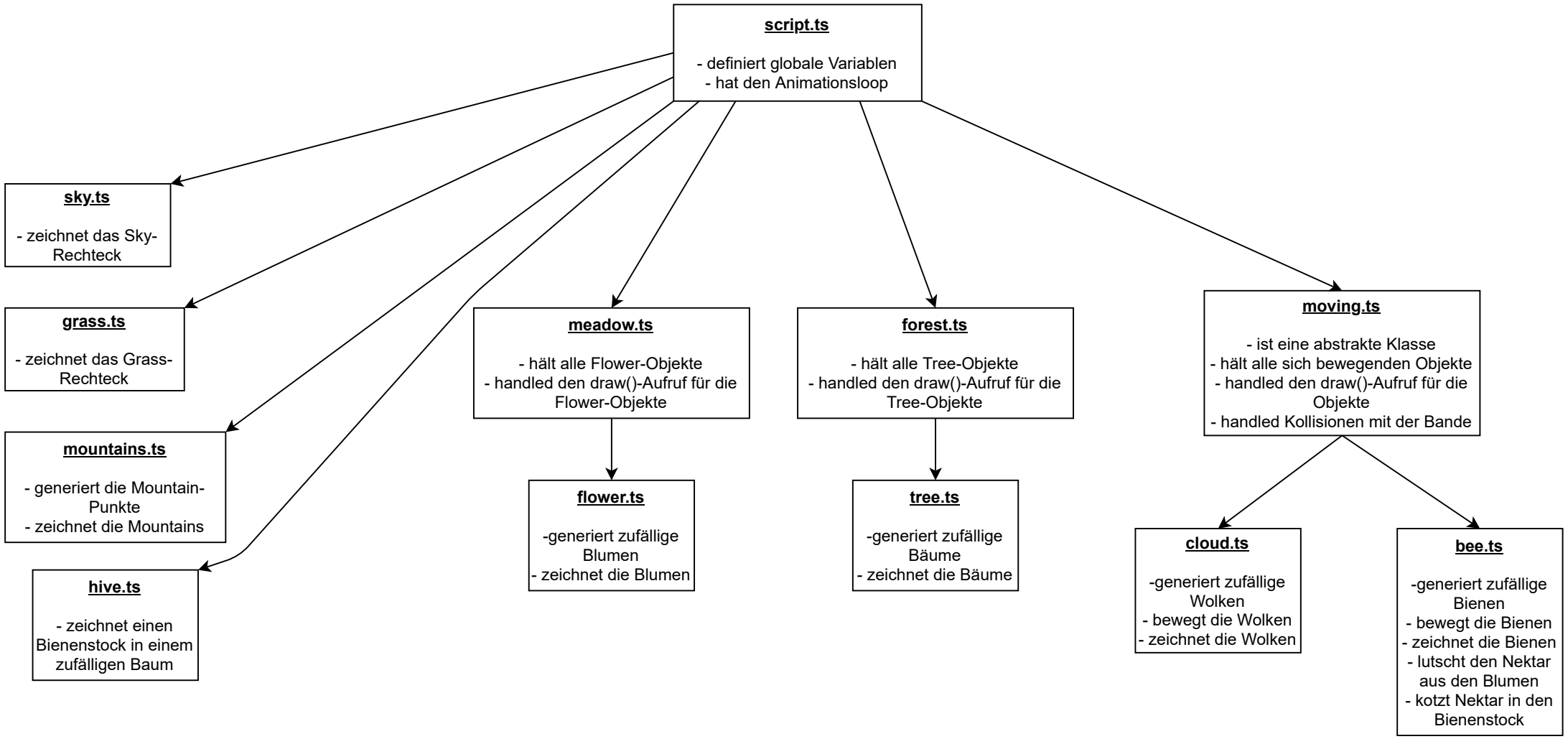
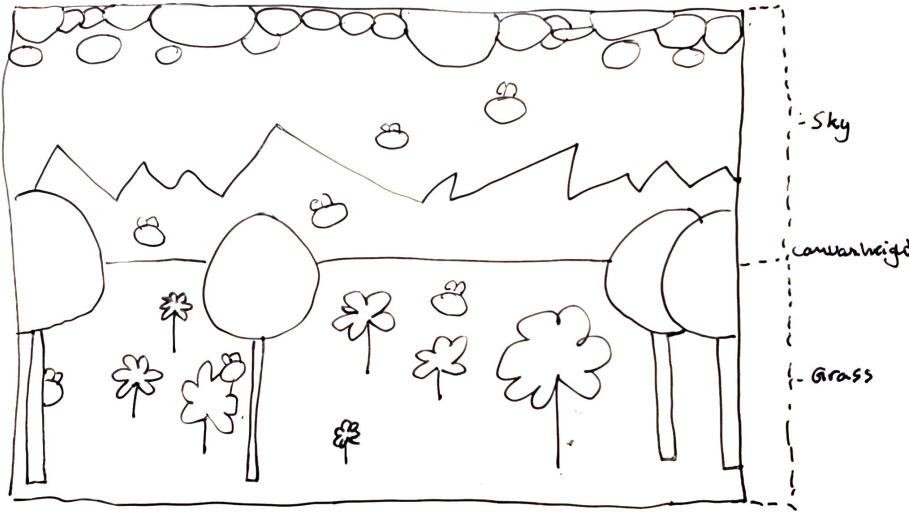
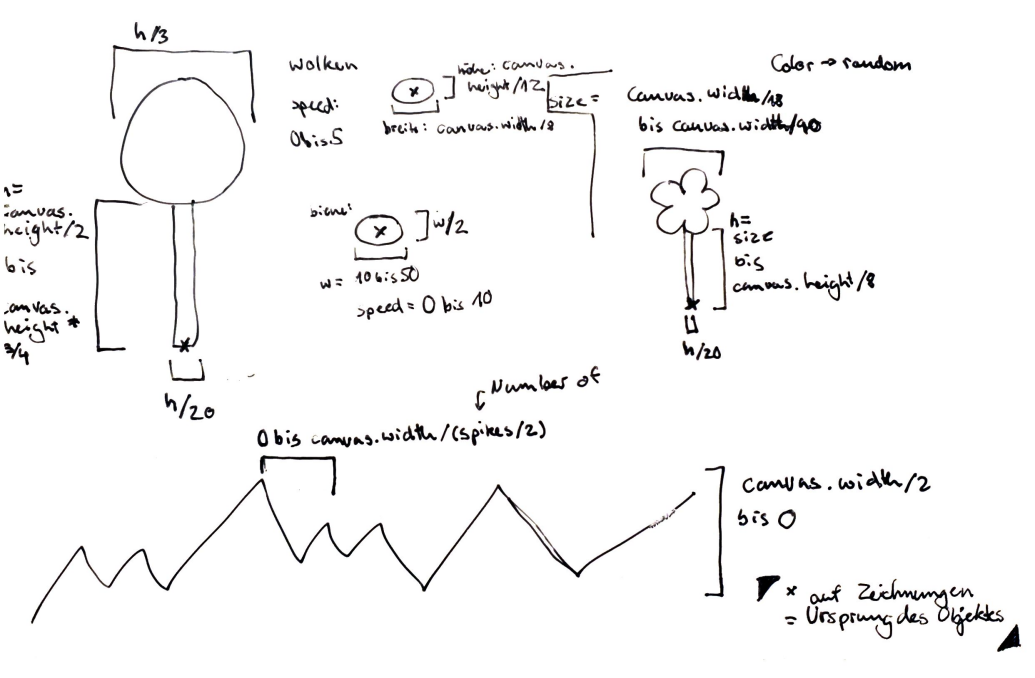
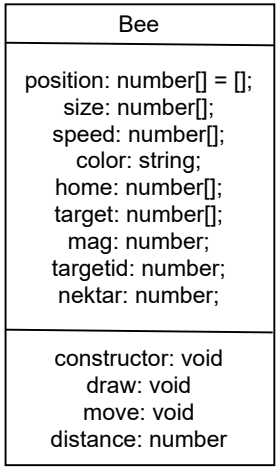
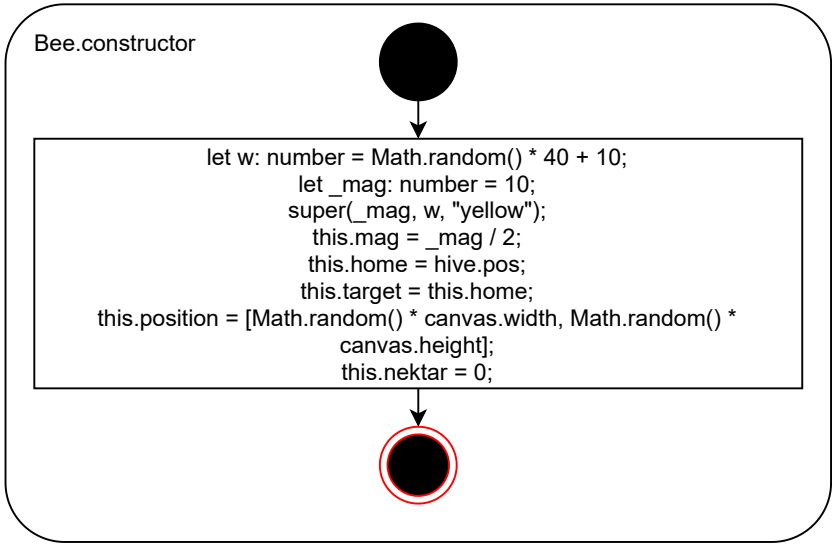

script.ts
- definiert globale Variablen
- hat den Animationsloop
sky.ts
- zeichnet das Sky-Rechteck
grass.ts
- zeichnet das Grass-Rechteck
mountains.ts
- generiert die Mountain-Punkte
- zeichnet die Mountains
hive.ts
- zeichnet einen Bienenstock in einem zufälligen Baum
meadow.ts
- hält alle Flower-Objekte
- handled den draw()-Aufruf für die Flower-Objekte
flower.ts
-generiert zufällige Blumen
- zeichnet die Blumen
forest.ts
- hält alle Tree-Objekte
- handled den draw()-Aufruf für die Tree-Objekte
tree.ts
-generiert zufällige Bäume
- zeichnet die Bäume
moving.ts
- ist eine abstrakte Klasse
- hält alle sich bewegenden Objekte
- handled den draw()-Aufruf für die Objekte
- handled Kollisionen mit der Bande
cloud.ts
-generiert zufällige Wolken
- bewegt die Wolken
- zeichnet die Wolken
bee.ts
-generiert zufällige Bienen
- bewegt die Bienen
- zeichnet die Bienen
- lutscht den Nektar aus den Blumen
- kotzt Nektar in den Bienenstock

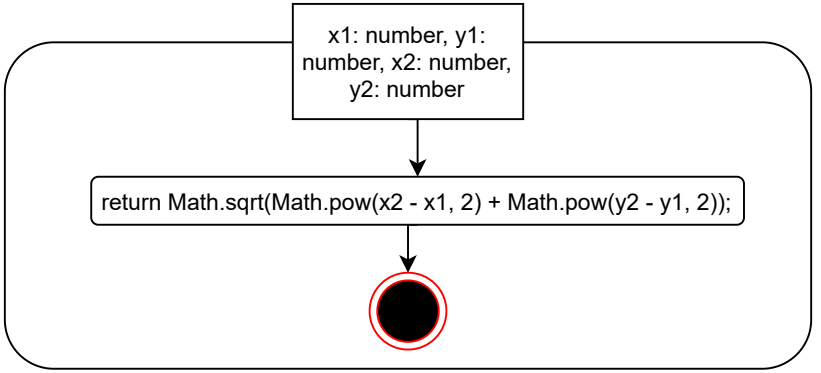
V1- Scribble

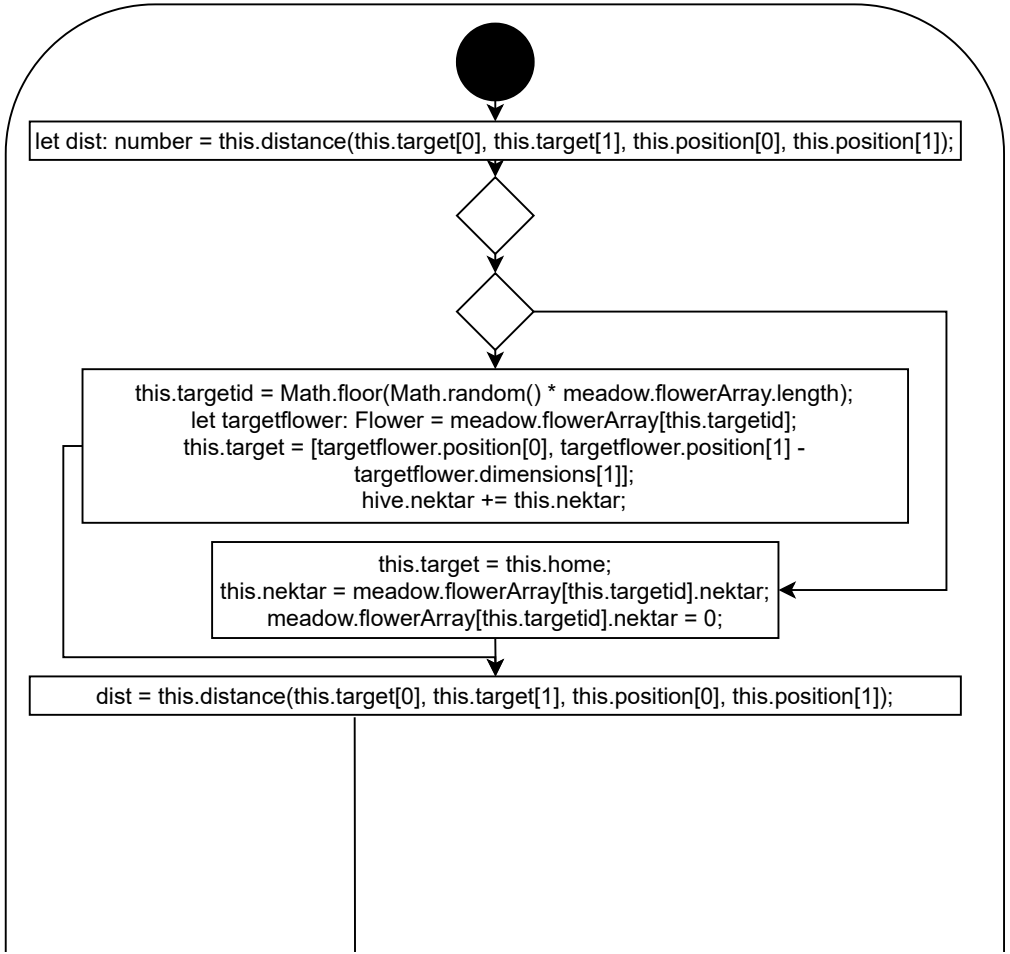


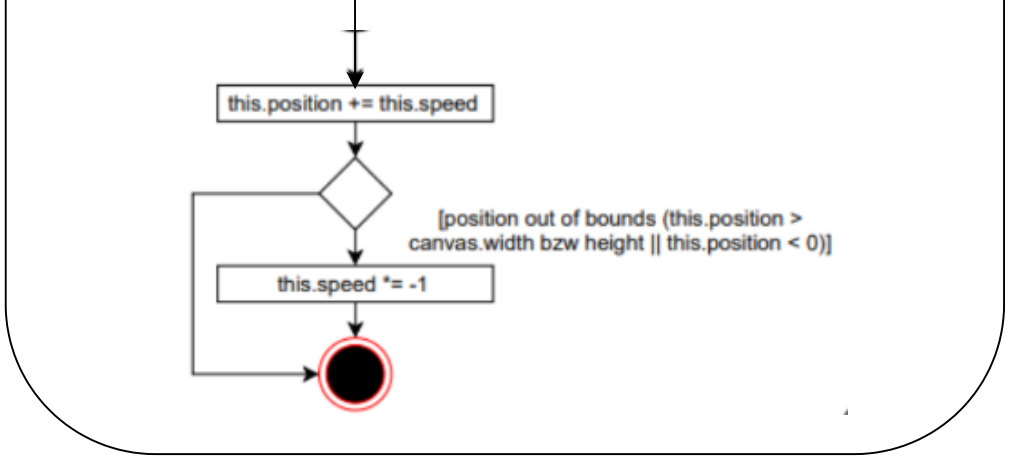

h/3
Wolken
speed:
0 bis 5
höhe: canvas. height/12
breite: canvas.width/8
size =
canvas.width/18
bis canvas.width/90
Color → random
bis
canvas. height/2
bis
canvas. height * 3/4
h/20
size
w/2
w = 10 bis 50
speed = 0 bis 10
h=
size
bis
canvas. height/8
h/20
Number of
0 bis canvas.width/(spikes/2)
canvas. width/2
bis 0
× auf Zeichnungen
= Ursprung des Objektes


Bee
position: number[] = [];
size: number[];
speed: number[];
color: string;
home: number[];
target: number[];
mag: number;
targetid: number;
nektar: number;
constructor: void
draw: void
move: void
distance: number

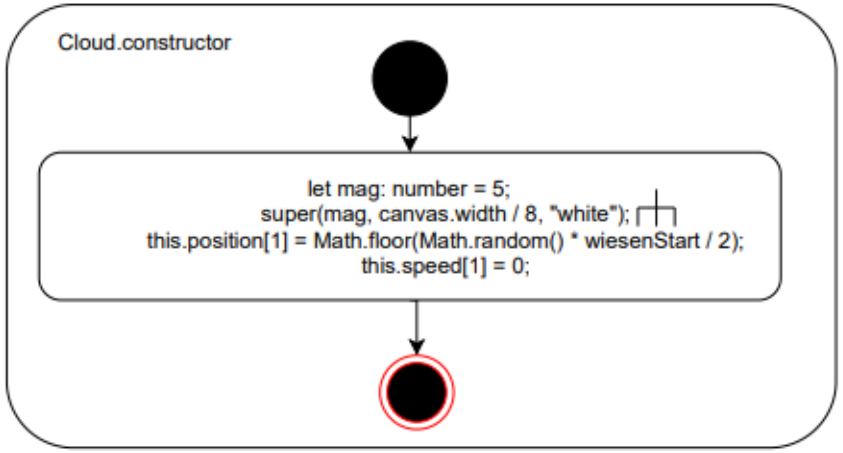

Bee.constructor
let w: number = Math.random() * 40 + 10;
let _mag: number = 10;
super(_mag, w, "yellow");
this.mag = _mag / 2;
this.home = hive.pos;
this.target = this.home;
this.position = [Math.random() * canvas.width, Math.random() *
canvas.height];
this.nektar = 0;


x1: number, y1:
number, x2: number,
y2: number
return Math.sqrt(Math.pow(x2 - x1, 2) + Math.pow(y2 - y1, 2));

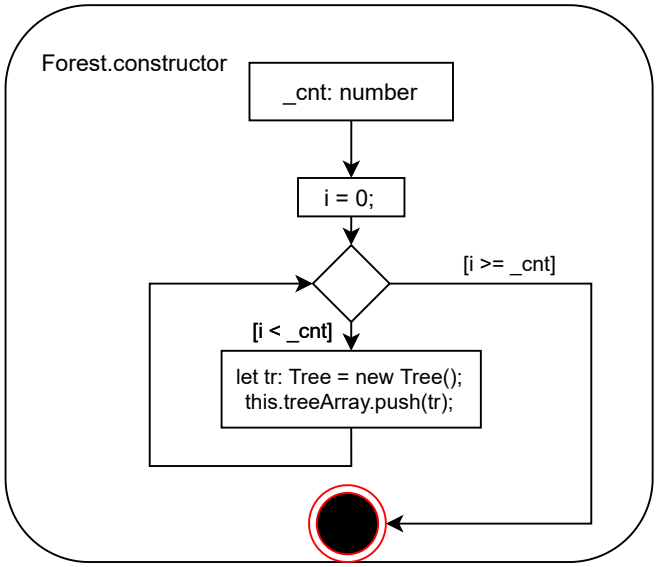

let dist: number = this.distance(this.target[0], this.target[1], this.position[0], this.position[1]);
this.targetid = Math.floor(Math.random() * meadow.flowerArray.length);
let targetflower: Flower = meadow.flowerArray[this.targetid];
this.target = [targetflower.position[0], targetflower.position[1] -
targetflower.dimensions[1]];
hive.nektar += this.nektar;
this.target = this.home;
this.nektar = meadow.flowerArray[this.targetid].nektar;
meadow.flowerArray[this.targetid].nektar = 0;
dist = this.distance(this.target[0], this.target[1], this.position[0], this.position[1]);

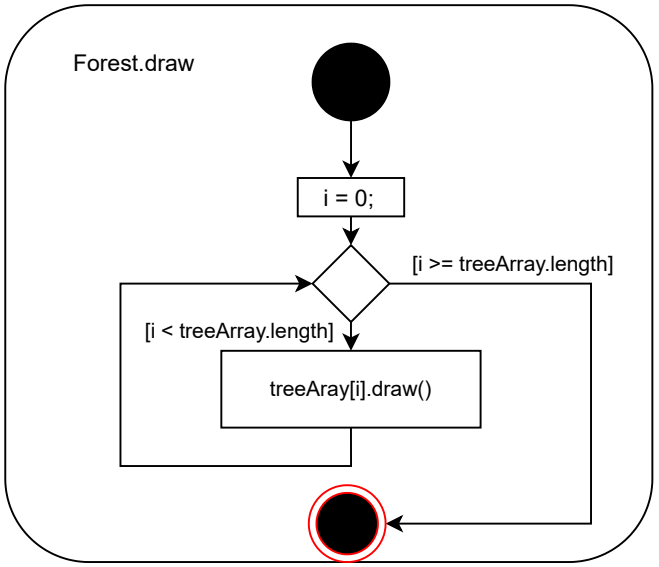


Cloud extends Moving
position: number[] = []; size: number[]; speed: number[]; color: string;
constructor: void draw: void move: void

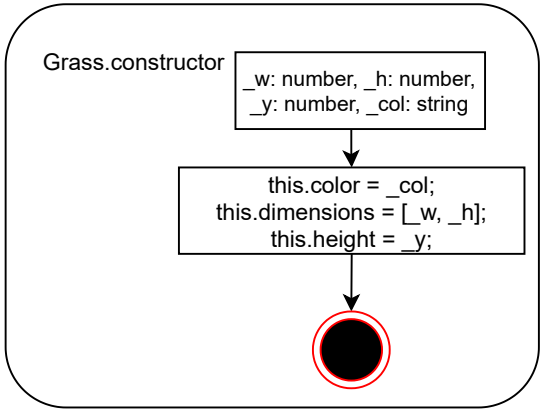


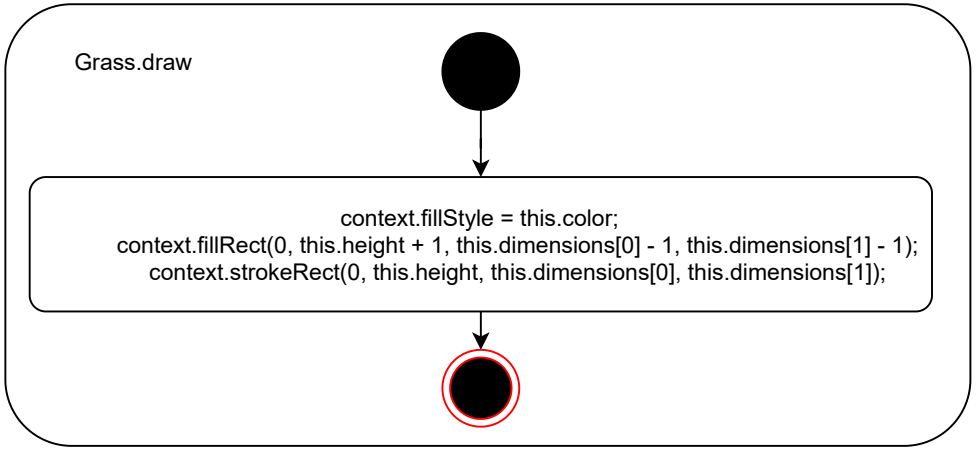
Forest
treeArray: Tree[] = [];
constructor: void draw: void

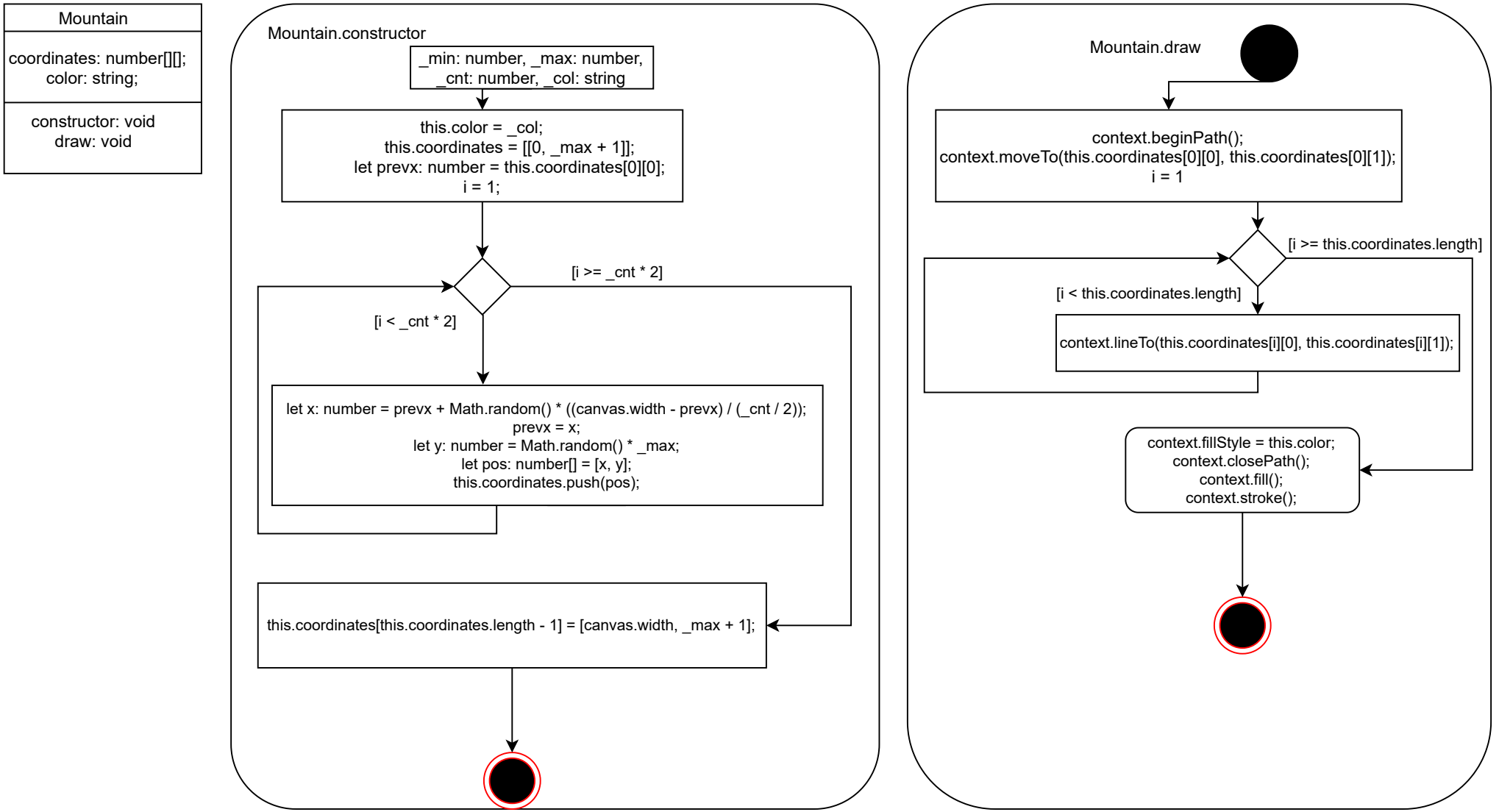


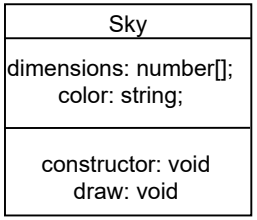


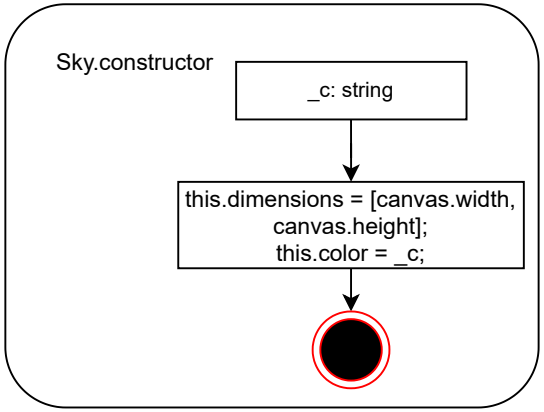
Grass
color: string; dimensions: number[]; height: number;
constructor: void draw: void

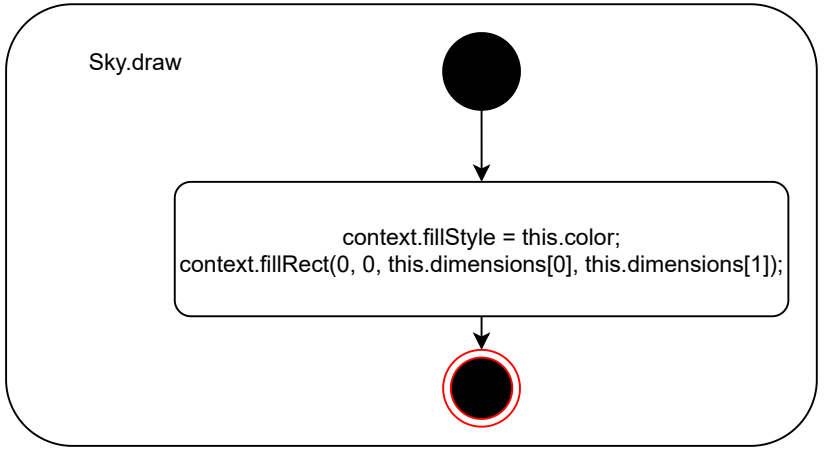


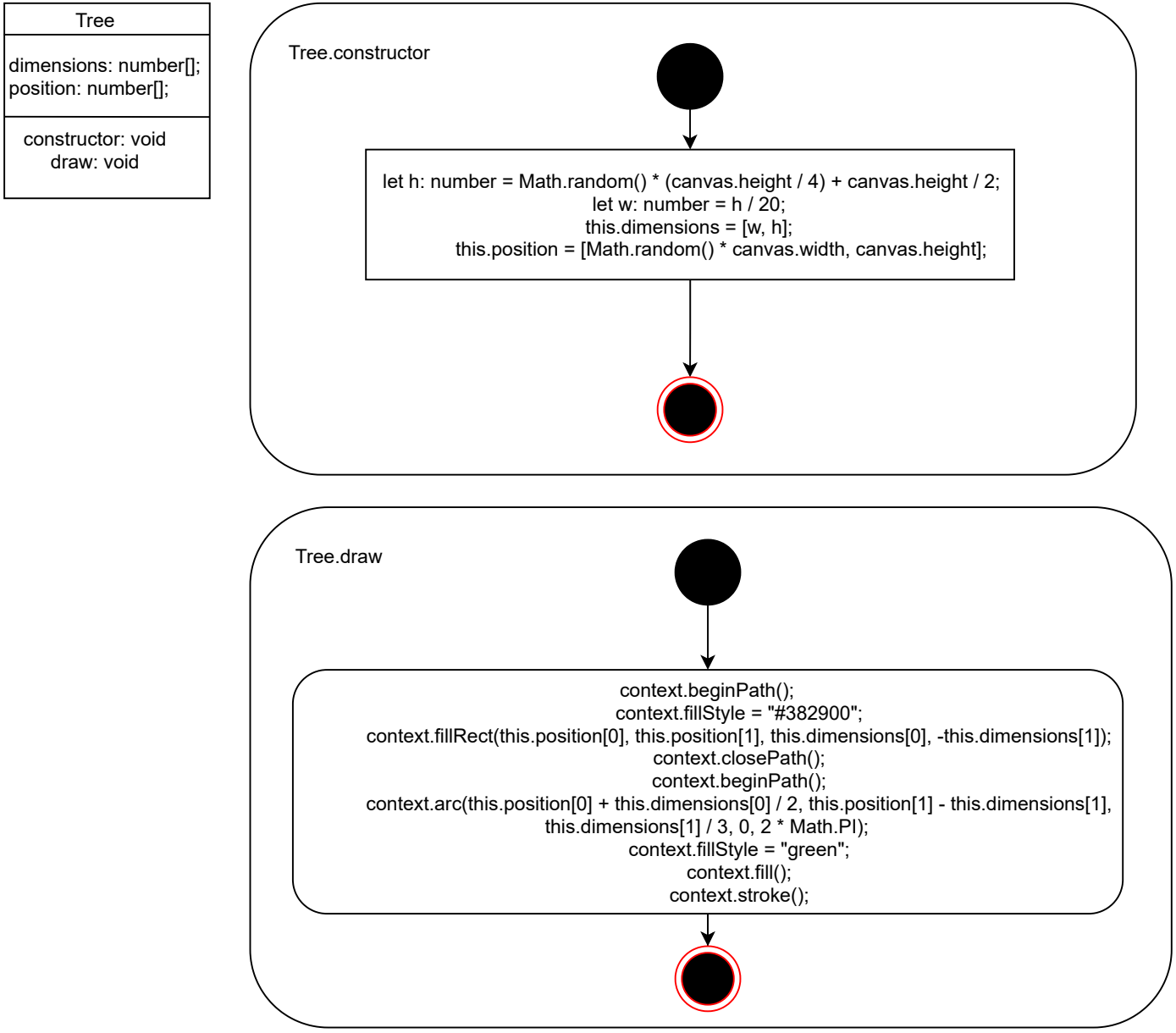


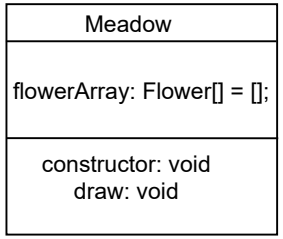

Mountain
coordinates: number[][];
color: string;
constructor: void
draw: void
Mountain.constructor
_min: number, _max: number,
_cnt: number, _col: string
this.color = _col;
this.coordinates = [[0, _max + 1]];
let prevx: number = this.coordinates[0][0];
i = 1;
[i >= _cnt * 2]
[i < _cnt * 2]
let x: number = prevx + Math.random() * ((canvas.width - prevx) / (_cnt / 2));
prevx = x;
let y: number = Math.random() * _max;
let pos: number[] = [x, y];
this.coordinates.push(pos);
this.coordinates[this.coordinates.length - 1] = [canvas.width, _max + 1];
Mountain.draw
context.beginPath();
context.moveTo(this.coordinates[0][0], this.coordinates[0][1]);
i = 1
[i >= this.coordinates.length]
[i < this.coordinates.length]
context.lineTo(this.coordinates[i][0], this.coordinates[i][1]);
context.fillStyle = this.color;
context.closePath();
context.fill();
context.stroke();

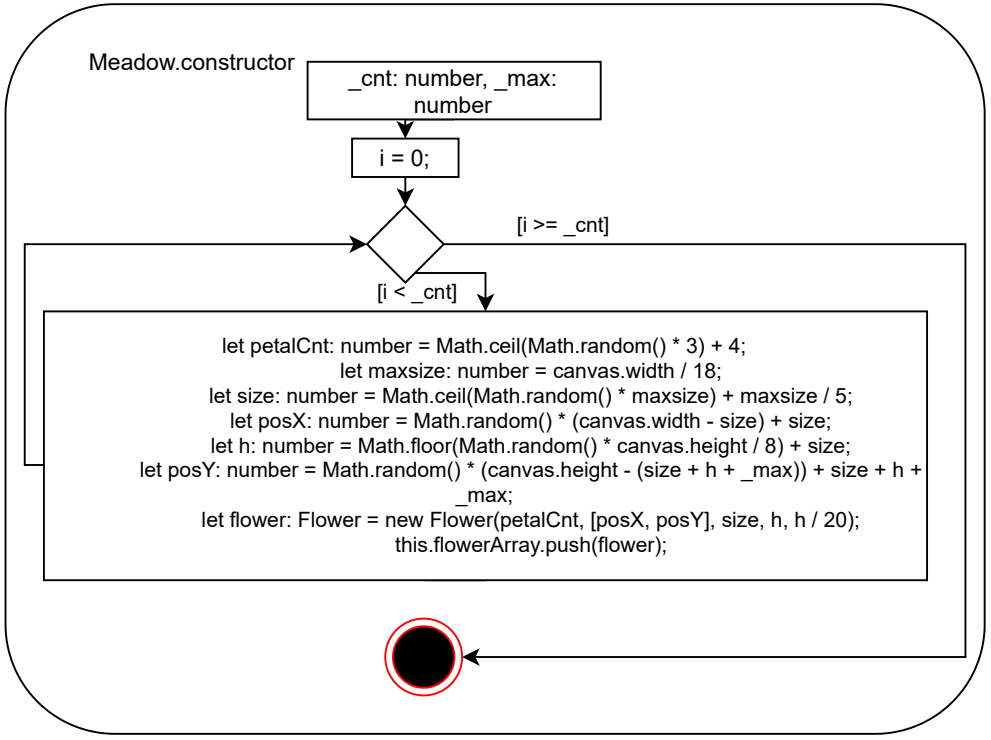

Sky
dimensions: number[];
color: string;
constructor: void
draw: void

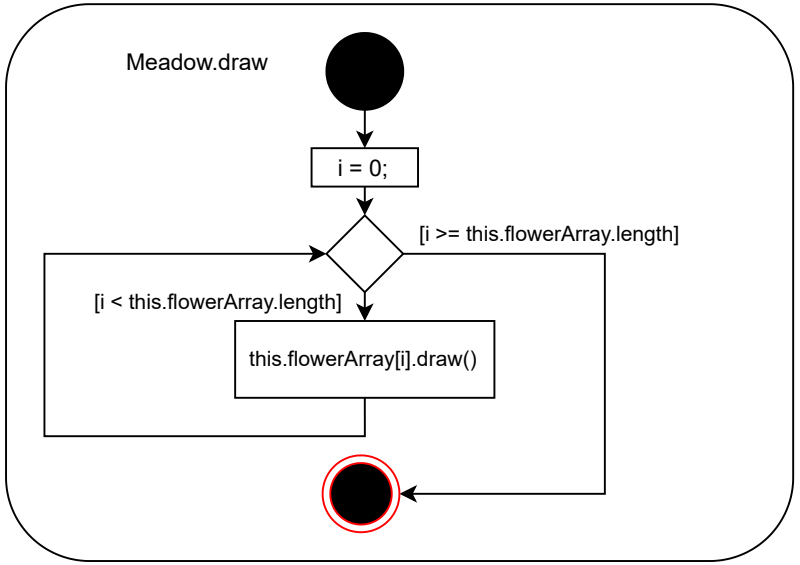

Sky.constructor
_c: string
this.dimensions = [canvas.width,
canvas.height];
this.color = _c;


Sky.draw
context.fillStyle = this.color;
context.fillRect(0, 0, this.dimensions[0], this.dimensions[1]);

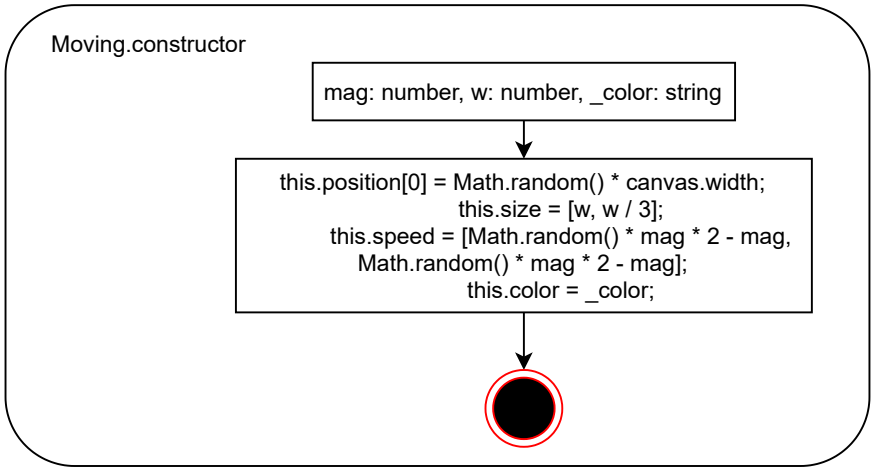

Tree
dimensions: number[];
position: number[];
constructor: void
draw: void
Tree.constructor
let h: number = Math.random() * (canvas.height / 4) + canvas.height / 2;
let w: number = h / 20;
this.dimensions = [w, h];
this.position = [Math.random() * canvas.width, canvas.height];
Tree.draw
context.beginPath();
context.fillStyle = "#382900";
context.fillRect(this.position[0], this.position[1], this.dimensions[0], -this.dimensions[1]);
context.closePath();
context.beginPath();
context.arc(this.position[0] + this.dimensions[0] / 2, this.position[1] - this.dimensions[1],
this.dimensions[1] / 3, 0, 2 * Math.PI);
context.fillStyle = "green";
context.fill();
context.stroke();

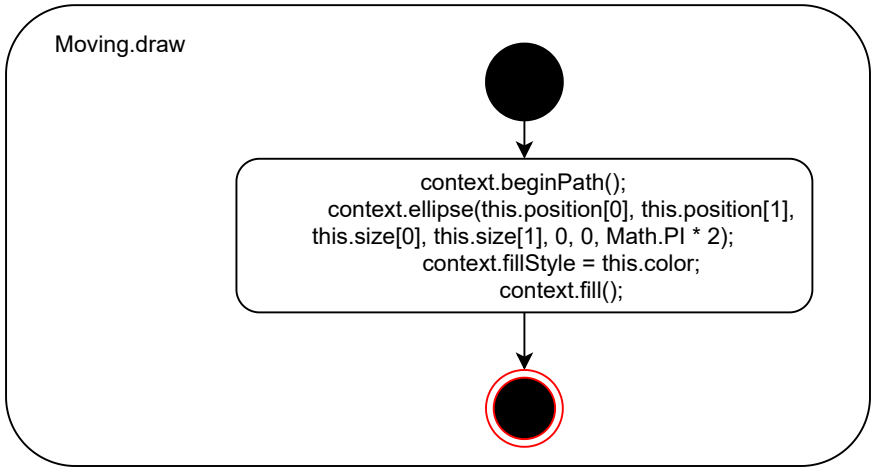

Meadow
flowerArray: Flower[] = [];
constructor: void
draw: void

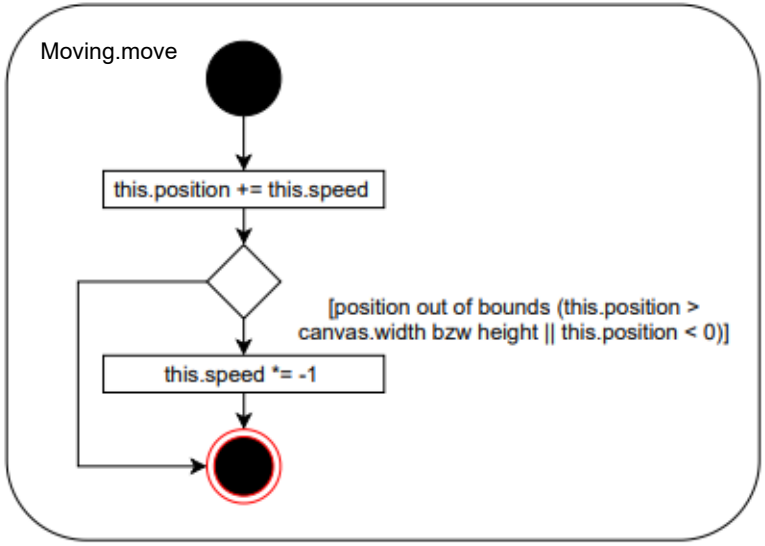

Meadow.constructor
_cnt: number, _max: number
i = 0;
[i >= _cnt]
[i < _cnt]
let petalCnt: number = Math.ceil(Math.random() * 3) + 4;
let maxsize: number = canvas.width / 18;
let size: number = Math.ceil(Math.random() * maxsize) + maxsize / 5;
let posX: number = Math.random() * (canvas.width - size) + size;
let h: number = Math.floor(Math.random() * canvas.height / 8) + size;
let posY: number = Math.random() * (canvas.height - (size + h + _max)) + size + h + _max;
let flower: Flower = new Flower(petalCnt, [posX, posY], size, h, h / 20);
this.flowerArray.push(flower);

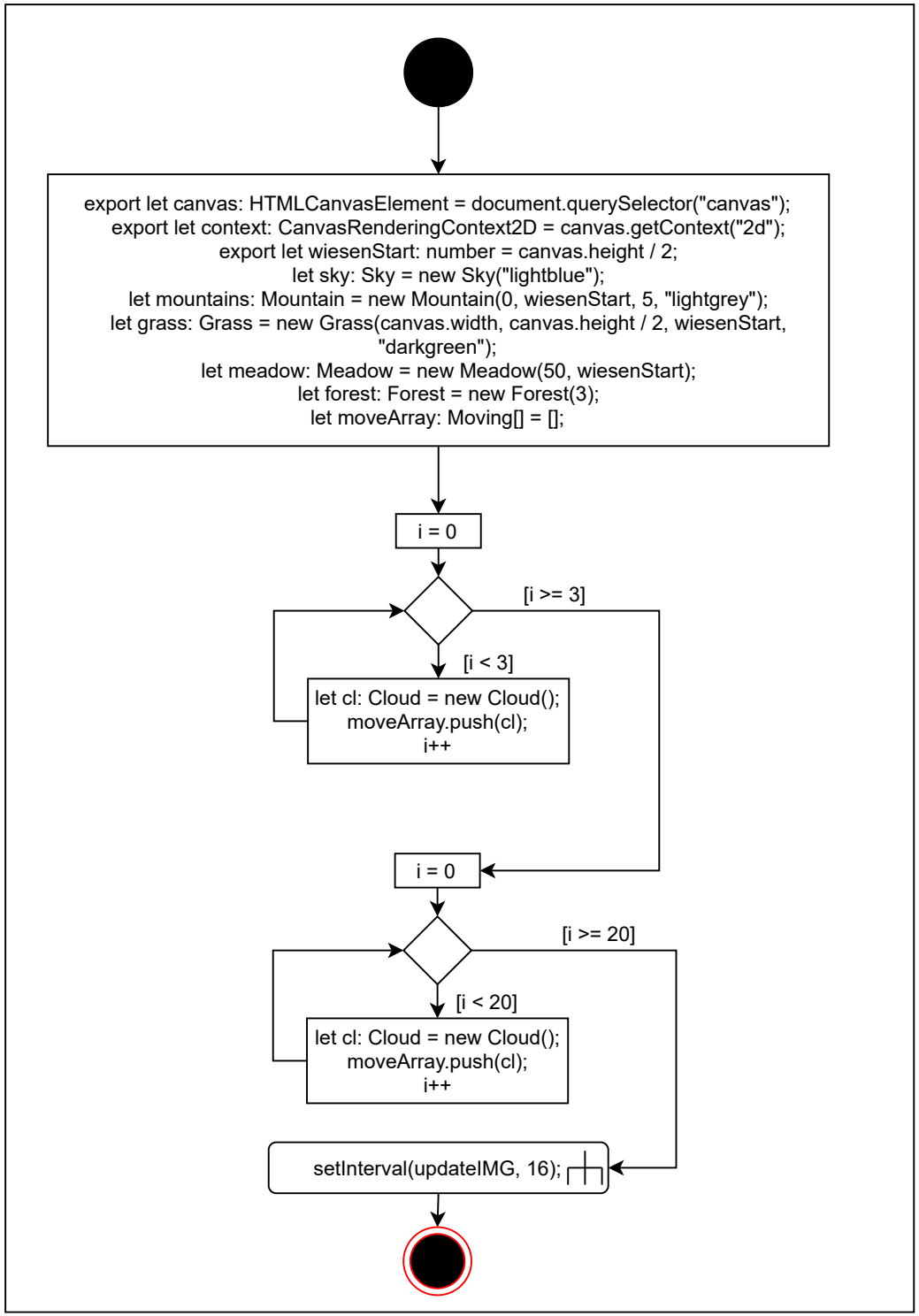

Meadow.draw
i = 0;
[i >= this.flowerArray.length]
[i < this.flowerArray.length]
this.flowerArray[i].draw()

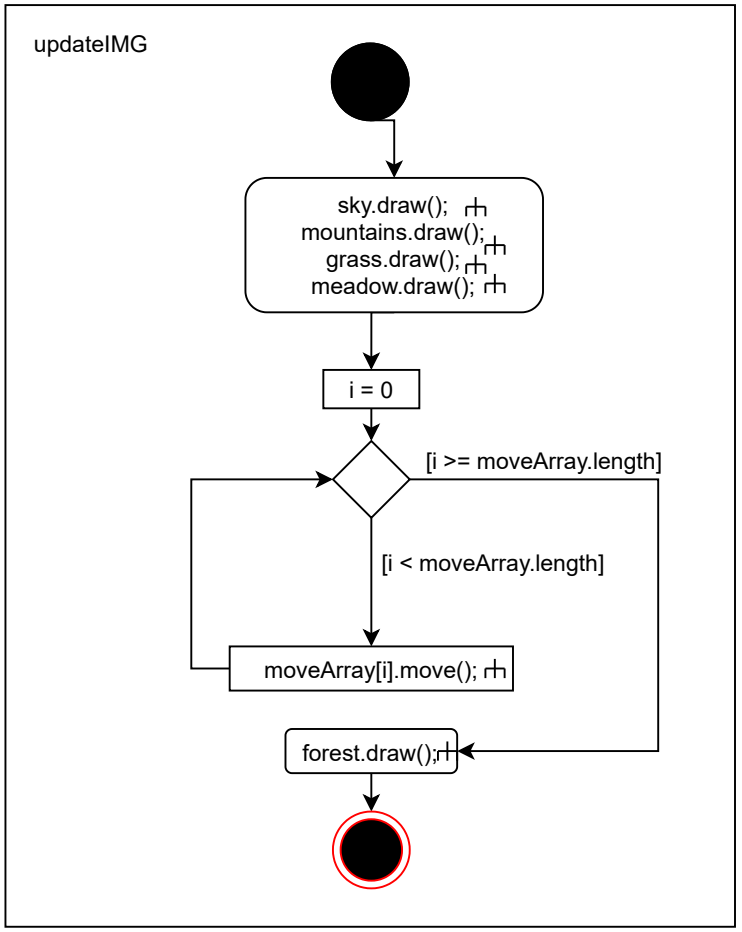
Moving
position: number[] = []; size: number[]; speed: number[]; color: string;
constructor: void draw: void move: void







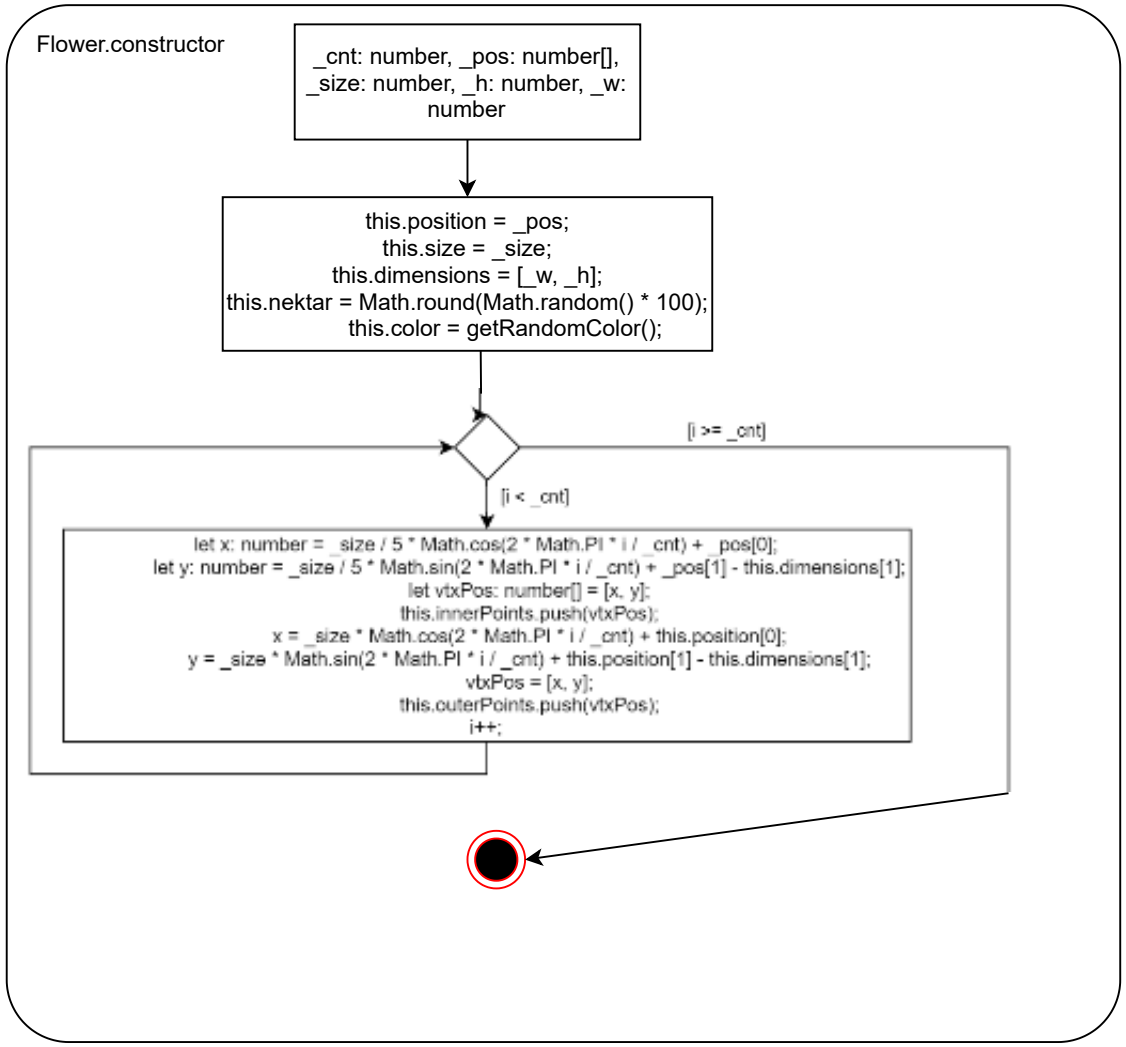

export let canvas: HTMLCanvasElement = document.querySelector("canvas");
export let context: CanvasRenderingContext2D = canvas.getContext("2d");
export let wiesenStart: number = canvas.height / 2;
let sky: Sky = new Sky("lightblue");
let mountains: Mountain = new Mountain(0, wiesenStart, 5, "lightgrey");
let grass: Grass = new Grass(canvas.width, canvas.height / 2, wiesenStart, "darkgreen");
let meadow: Meadow = new Meadow(50, wiesenStart);
let forest: Forest = new Forest(3);
let moveArray: Moving[] = [];
i = 0
[i >= 3]
[i < 3]
let cl: Cloud = new Cloud();
moveArray.push(cl);
i++
i = 0
[i >= 20]
[i < 20]
let cl: Cloud = new Cloud();
moveArray.push(cl);
i++
setInterval(updateIMG, 16);

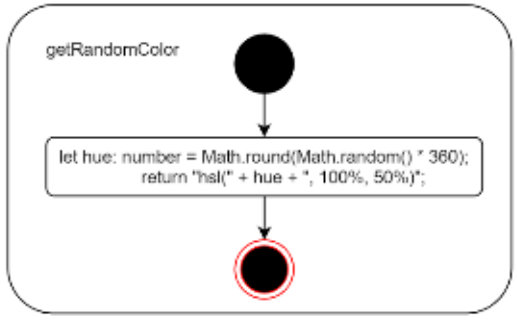

updateIMG
sky.draw();
mountains.draw();
grass.draw();
meadow.draw();
i = 0
[i >= moveArray.length]
[i < moveArray.length]
moveArray[i].move();
forest.draw();

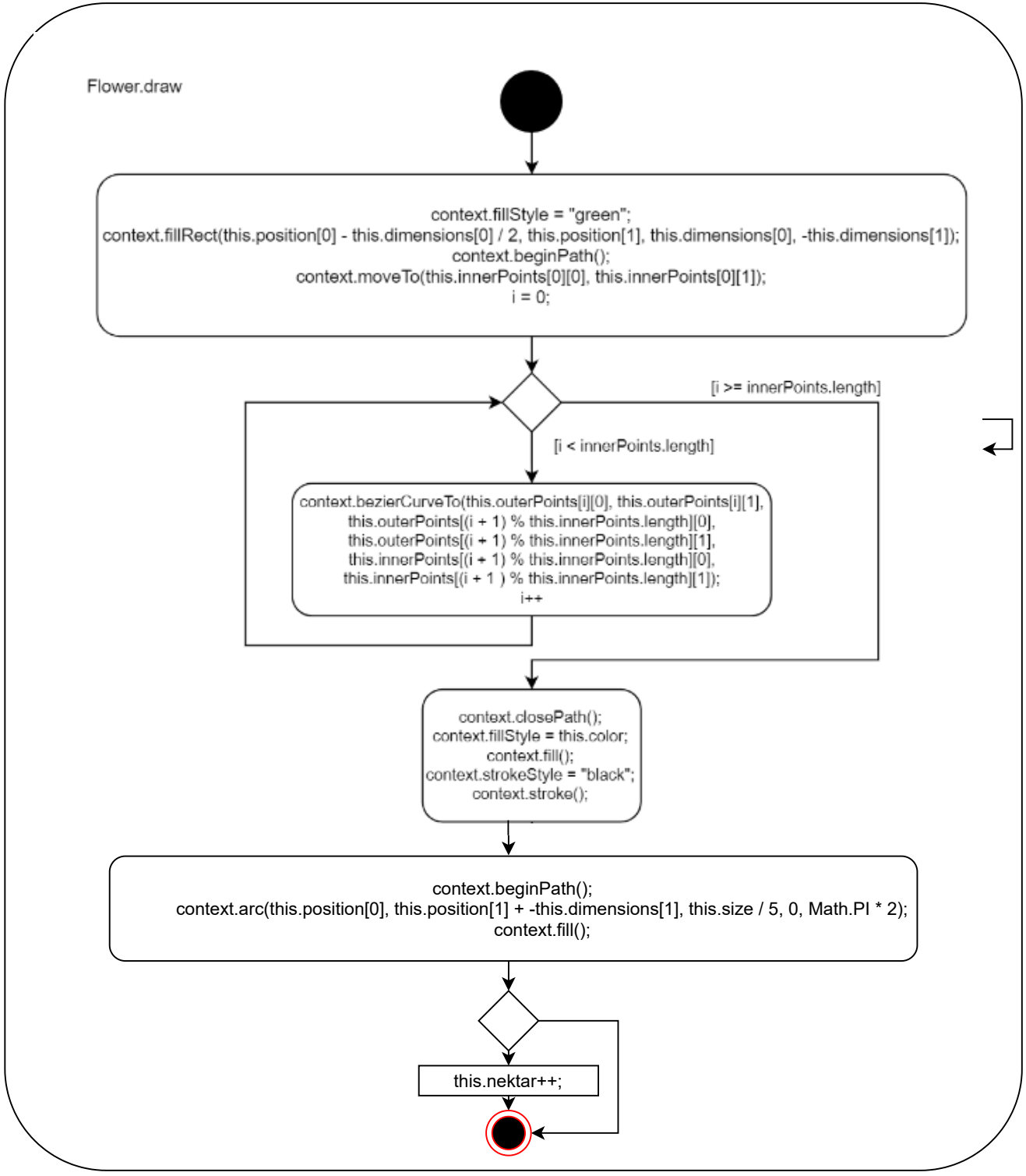
Flower

innerPoints: number[][] = [];
outerPoints: number[][] = [];
position: number[];
dimensions: number[];
color: string;
nektar: number;
size: number;

constructor: void draw: void getRandomColor: string






Flower.draw
context.fillStyle = "green";
context.fillRect(this.position[0] - this.dimensions[0] / 2, this.position[1], this.dimensions[0], -this.dimensions[1]);
context.beginPath();
context.moveTo(this.innerPoints[0][0], this.innerPoints[0][1]);
i = 0;
[i >= innerPoints.length]
[i < innerPoints.length]
context.bezierCurveTo(this.outerPoints[i][0], this.outerPoints[i][1],
this.outerPoints[(i + 1) % this.innerPoints.length][0],
this.outerPoints[(i + 1) % this.innerPoints.length][1],
this.innerPoints[(i + 1) % this.innerPoints.length][0],
this.innerPoints[(i + 1) % this.innerPoints.length][1]);
i++
context.closePath();
context.fillStyle = this.color;
context.fill();
context.strokeStyle = "black";
context.stroke();
context.beginPath();
context.arc(this.position[0], this.position[1] + -this.dimensions[1], this.size / 5, 0, Math.PI * 2);
context.fill();
this.nektar++;