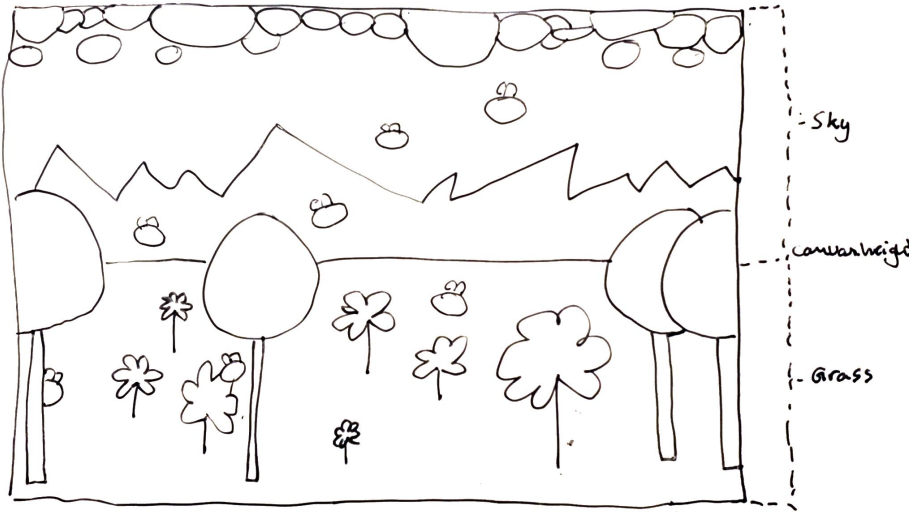
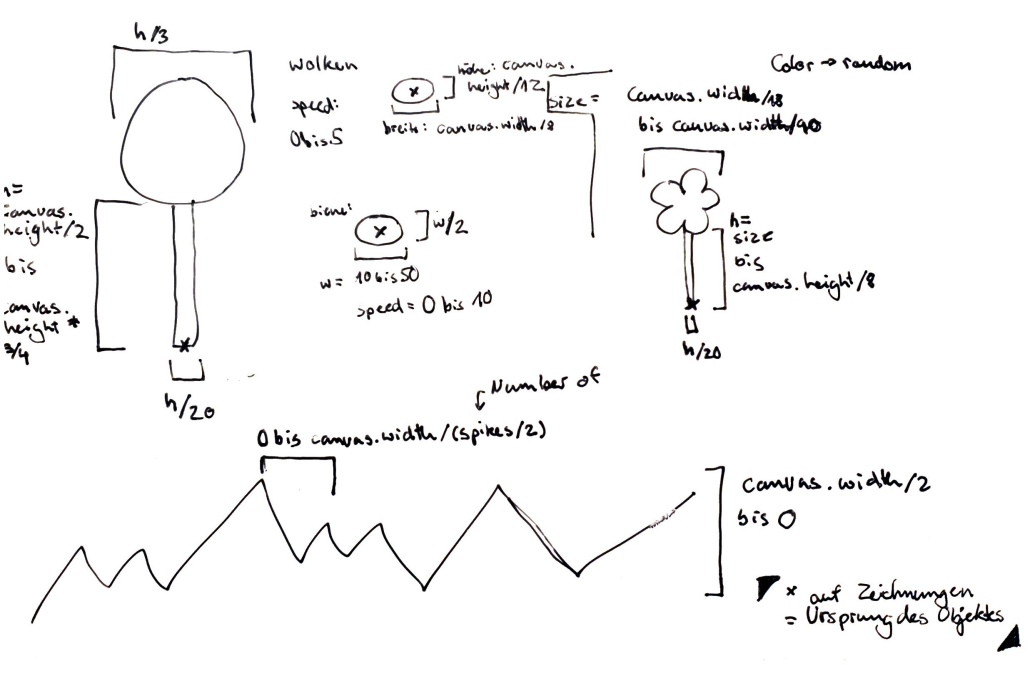
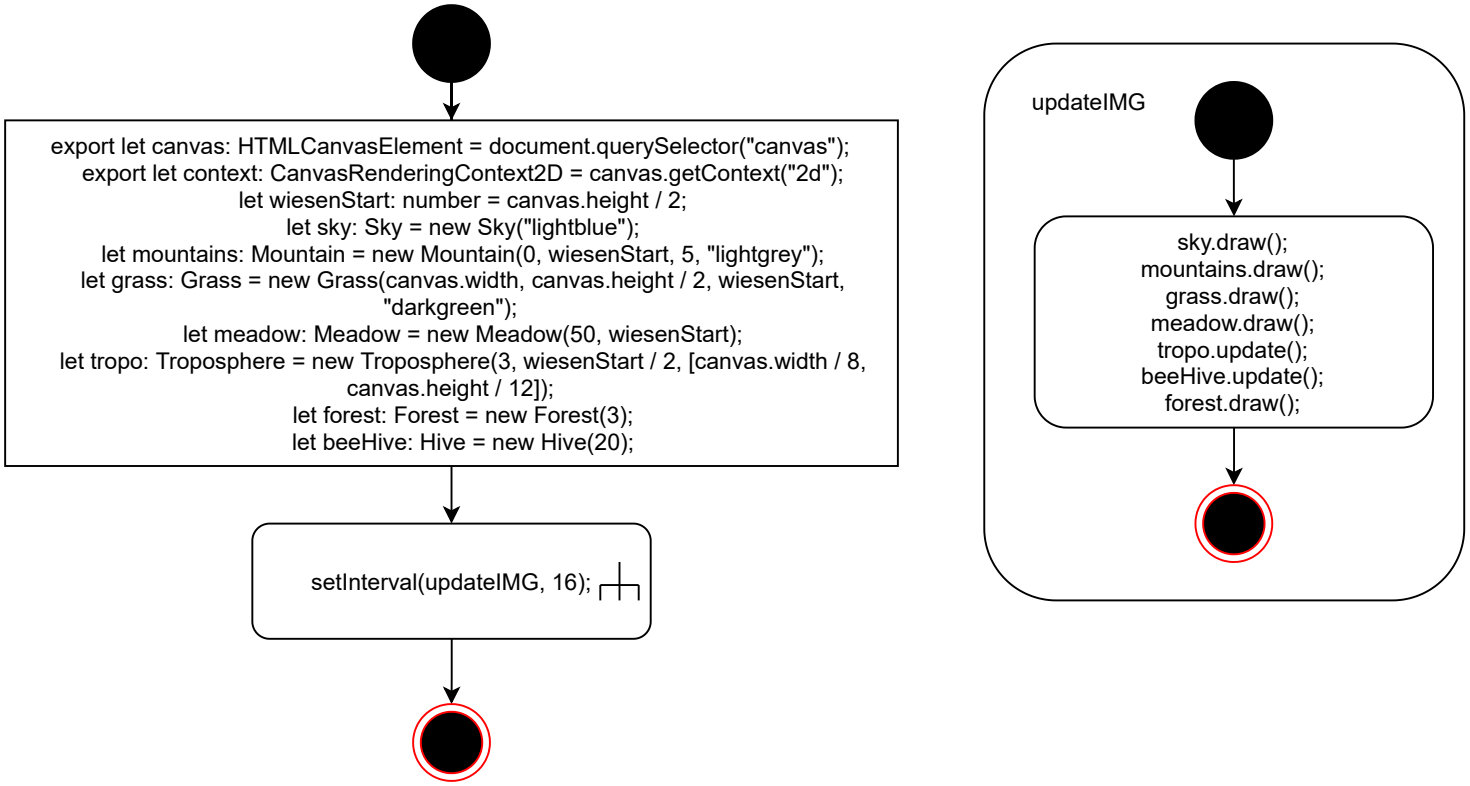
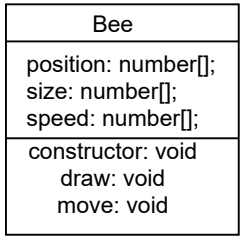
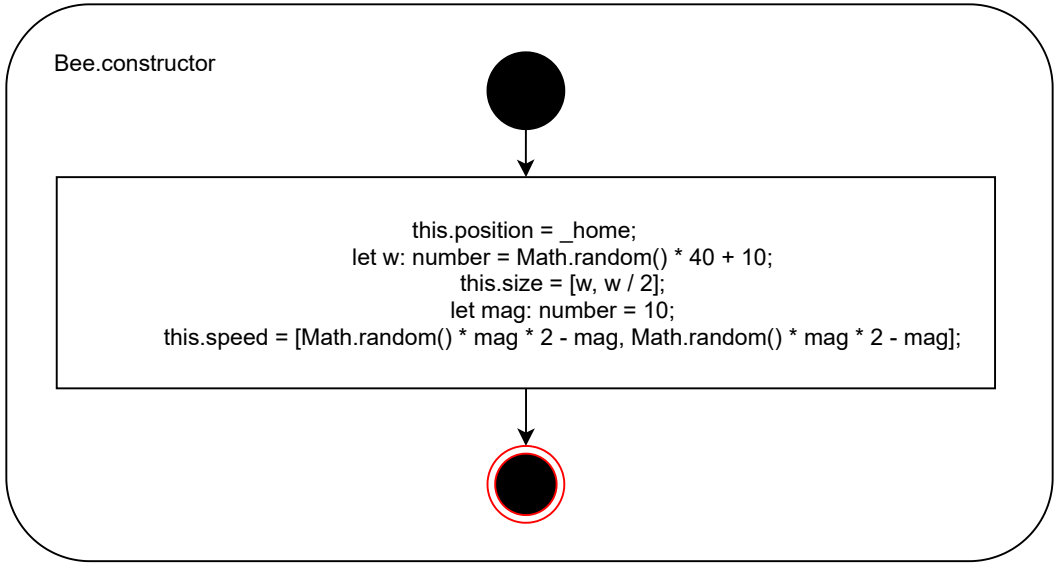
V1- Scribble

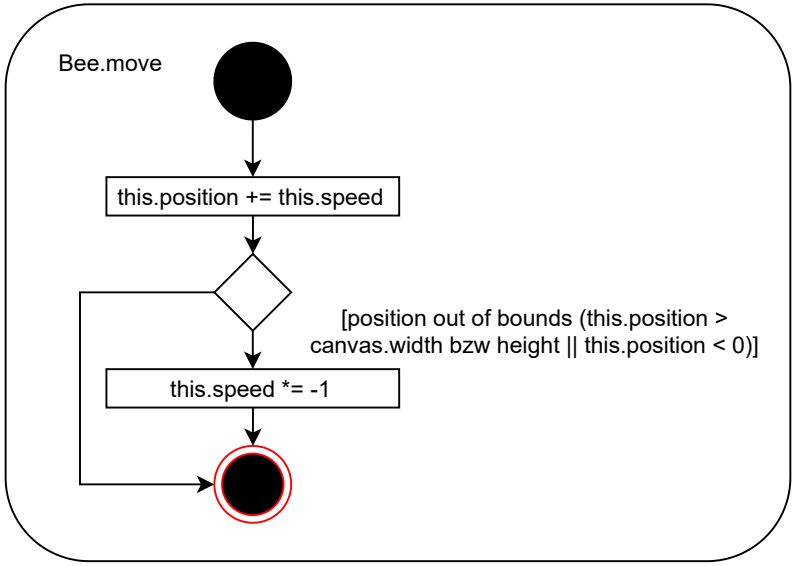


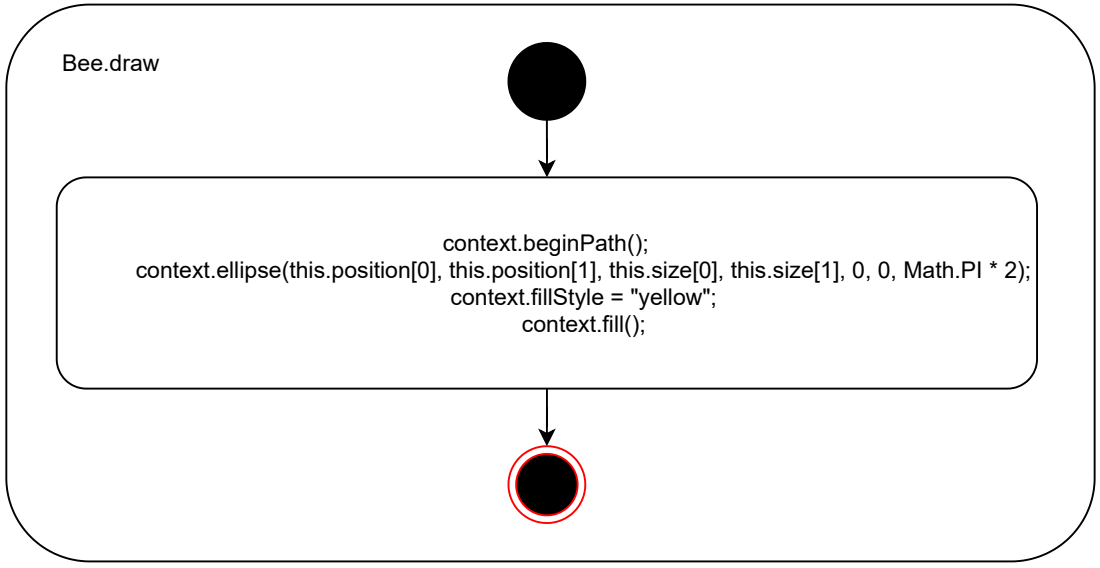

h/3
Wolken
speed:
0 bis 5
höhe: canvas. height/12
breite: canvas.width/8
size =
canvas.width/18
bis canvas.width/90
Color → random
bis
canvas. height/2
bis
canvas. height * 3/4
h/20
size
w/2
w = 10 bis 50
speed = 0 bis 10
h=
size
bis
canvas. height/8
h/20
Number of
0 bis canvas.width/(spikes/2)
canvas.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


export let canvas: HTMLCanvasElement = document.querySelector("canvas");
export let context: CanvasRenderingContext2D = canvas.getContext("2d");
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 tropo: Troposphere = new Troposphere(3, wiesenStart / 2, [canvas.width / 8, canvas.height / 12]);
let forest: Forest = new Forest(3);
let beeHive: Hive = new Hive(20);
setInterval(updateIMG, 16);
updateIMG
sky.draw();
mountains.draw();
grass.draw();
meadow.draw();
tropo.update();
beeHive.update();
forest.draw();

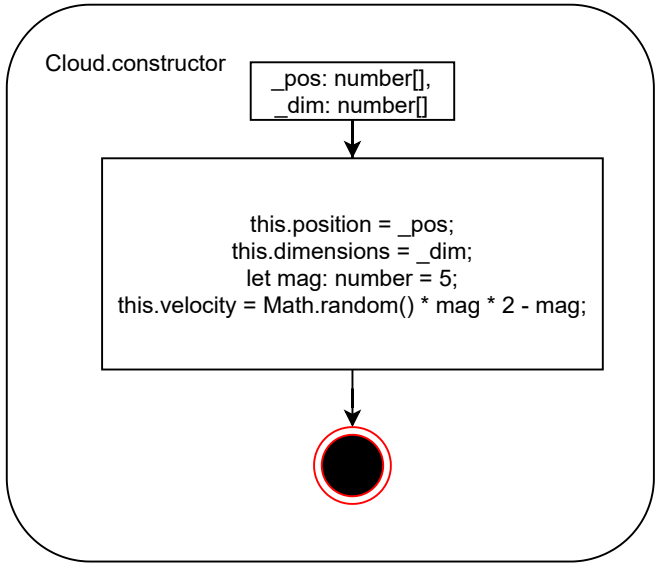

Bee
position: number[];
size: number[];
speed: number[];
constructor: void
draw: void
move: void

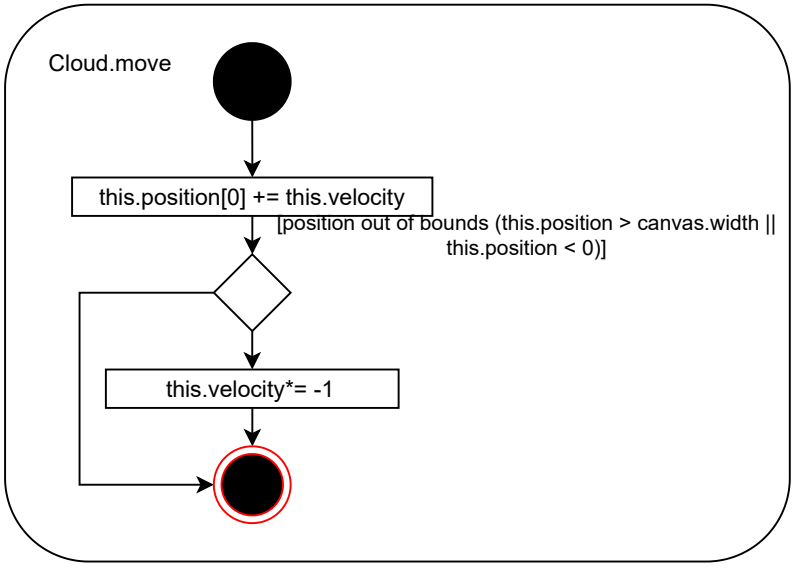

Bee.constructor
this.position = _home;
let w: number = Math.random() * 40 + 10;
this.size = [w, w / 2];
let mag: number = 10;
this.speed = [Math.random() * mag * 2 - mag, Math.random() * mag * 2 - mag];

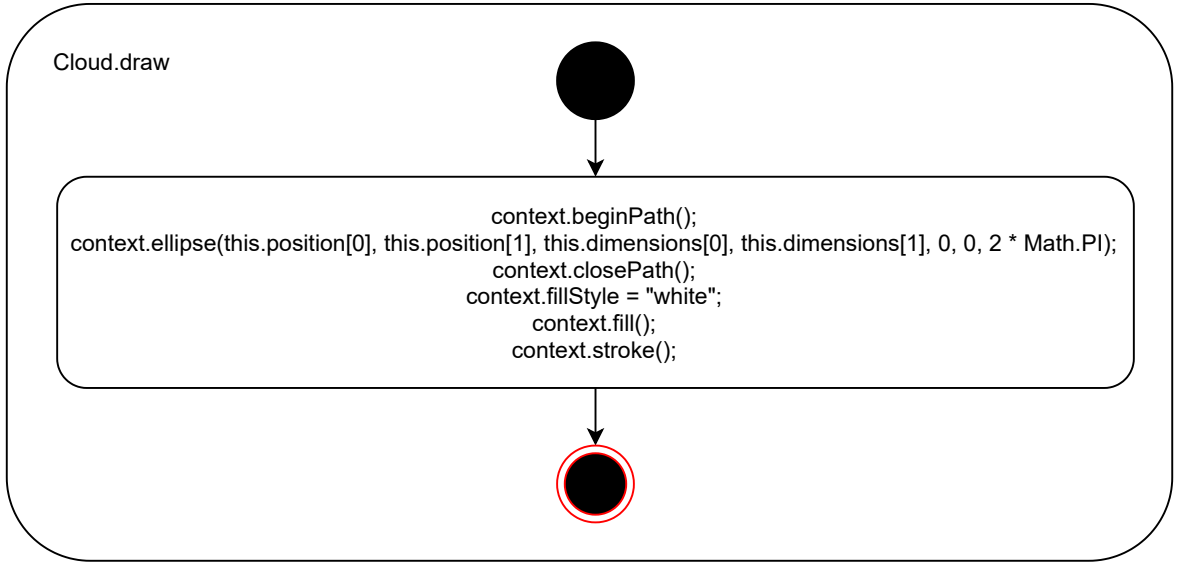

Bee.move
this.position += this.speed
[position out of bounds (this.position >
canvas.width bzw height || this.position < 0)]
this.speed *= -1

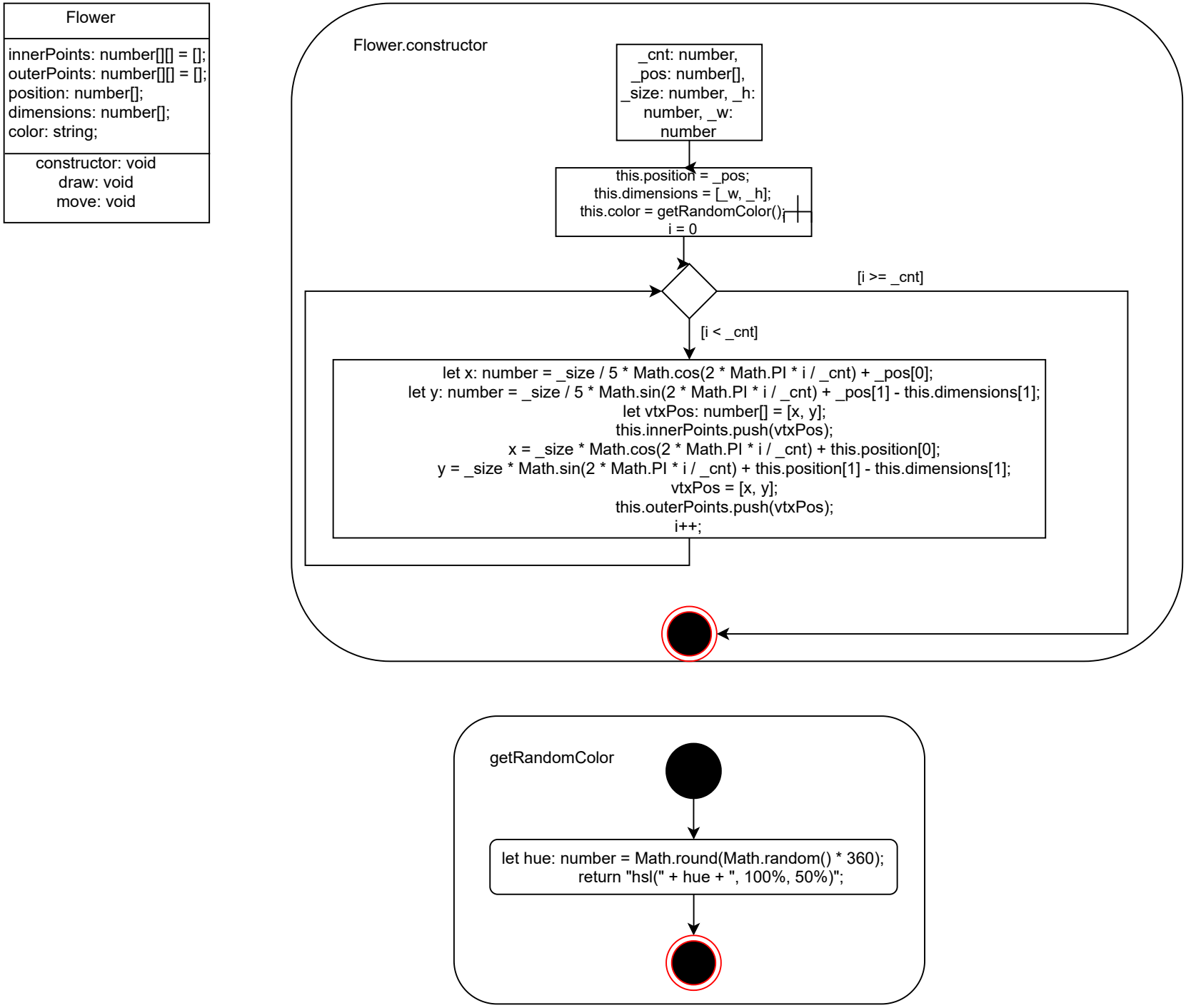

Bee.draw
context.beginPath();
context.ellipse(this.position[0], this.position[1], this.size[0], this.size[1], 0, 0, Math.PI * 2);
context.fillStyle = "yellow";
context.fill();

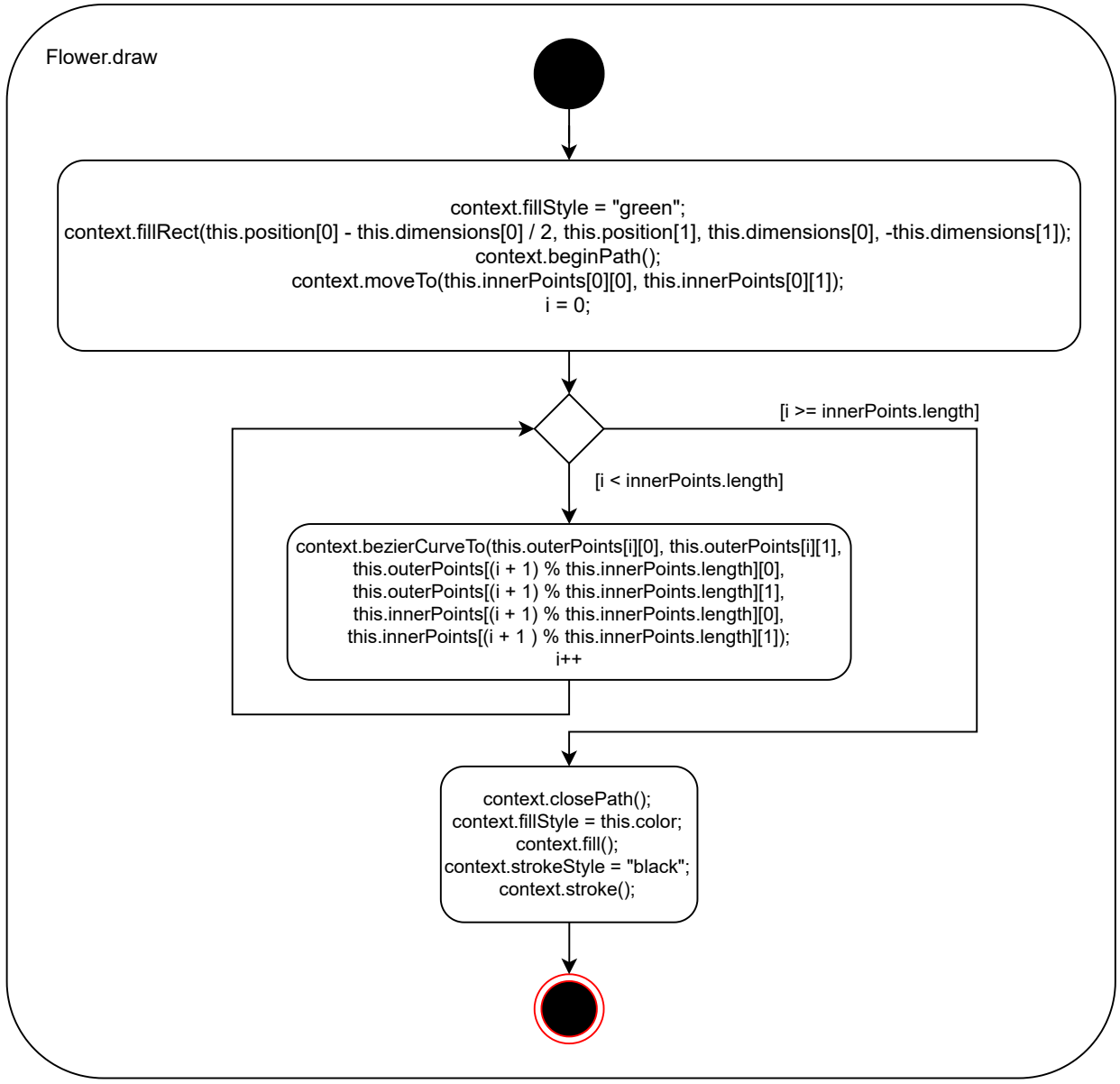
Cloud
position: number[]; dimensions: number[]; velocity: number;
constructor: void draw: void move: void



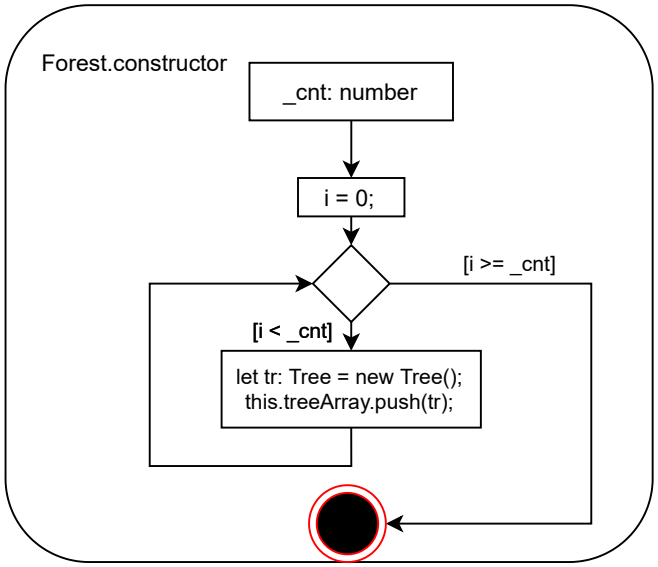


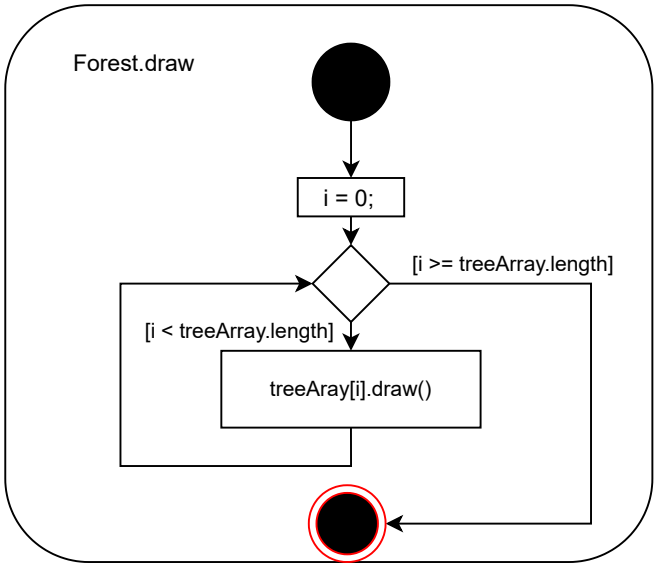



Flower
innerPoints: number[][] = [];
outerPoints: number[][] = [];
position: number[];
dimensions: number[];
color: string;
constructor: void
draw: void
move: void
Flower.constructor
_cnt: number, _pos: number[], _size: number, _h: number, _w: number
this.position = _pos;
this.dimensions = [_w, _h];
this.color = getRandomColor();
i = 0
[i >= _cnt]
[i < _cnt]
let x: number = _size / 5 * Math.cos(2 * Math.PI * i / _cnt) + _pos[0];
let y: number = _size / 5 * Math.sin(2 * Math.PI * i / _cnt) + _pos[1] - this.dimensions[1];
let vtxPos: number[] = [x, y];
this.innerPoints.push(vtxPos);
x = _size * Math.cos(2 * Math.PI * i / _cnt) + this.position[0];
y = _size * Math.sin(2 * Math.PI * i / _cnt) + this.position[1] - this.dimensions[1];
vtxPos = [x, y];
this.outerPoints.push(vtxPos);
i++;
getRandomColor
let hue: number = Math.round(Math.random() * 360);
return "hsl(" + hue + ", 100%, 50%)";

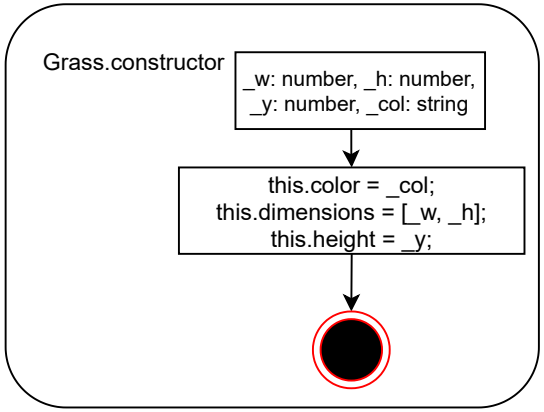

Flower.draw
context.fillStyle = "green";
context.fillRect(this.position[0] - this.dimensions[0] / 2, this.position[1], this.dimensions[0], -this.dimensions[1]);
context.beginPath();
context.moveTo(this.innerPoints[0][0], this.innerPoints[0][1]);
i = 0;
[i >= innerPoints.length]
[i < innerPoints.length]
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();

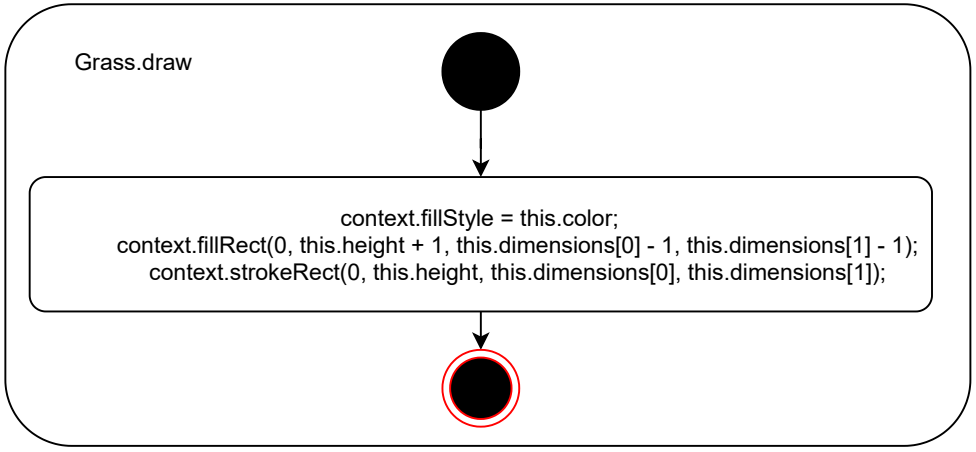
Forest
treeArray: Tree[] = [];
constructor: void draw: void

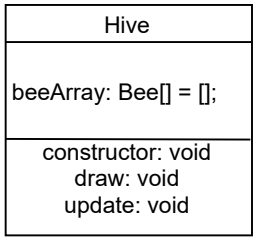


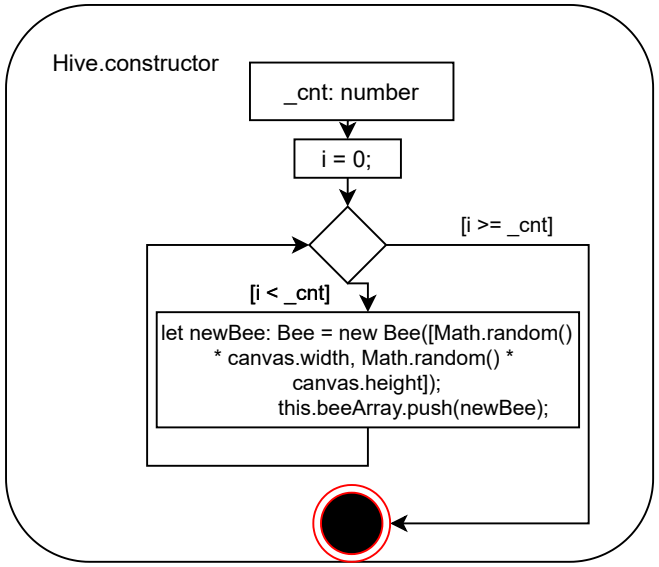


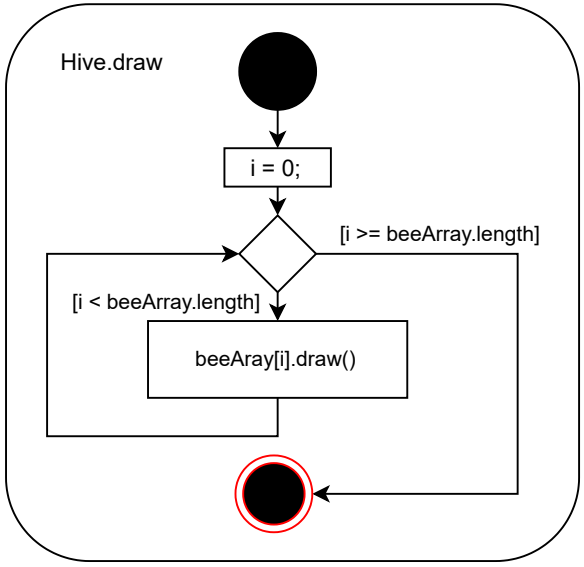
Grass
color: string; dimensions: number[]; height: number;
constructor: void draw: void

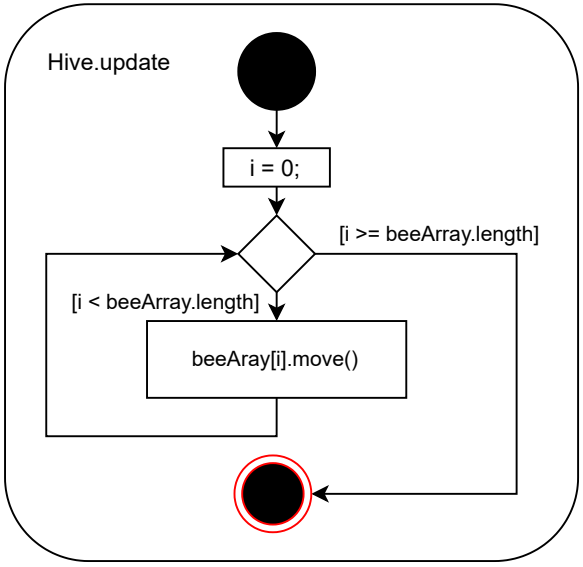


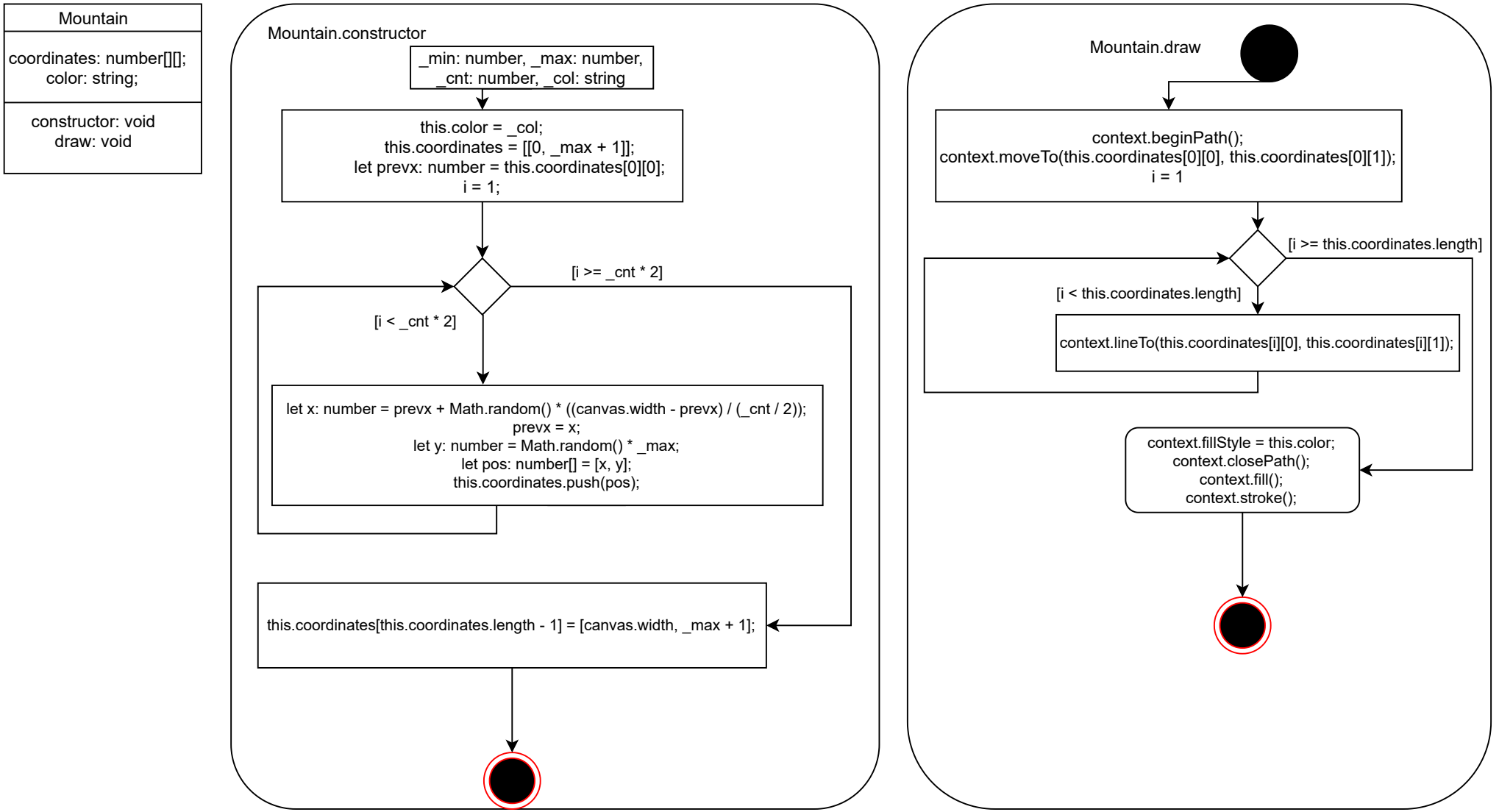


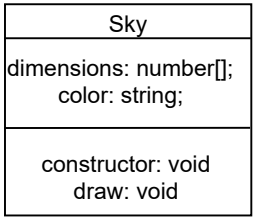

Hive
beeArray: Bee[] = [];
constructor: void
draw: void
update: void

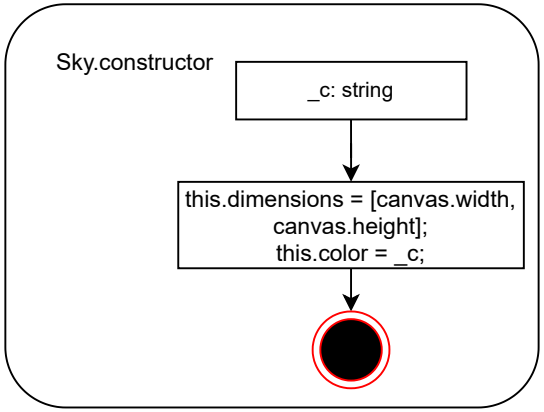

Hive.constructor
_cnt: number
i = 0;
[i >= _cnt]
[i < _cnt]
let newBee: Bee = new Bee([Math.random()
* canvas.width, Math.random() *
canvas.height]);
this.beeArray.push(newBee);

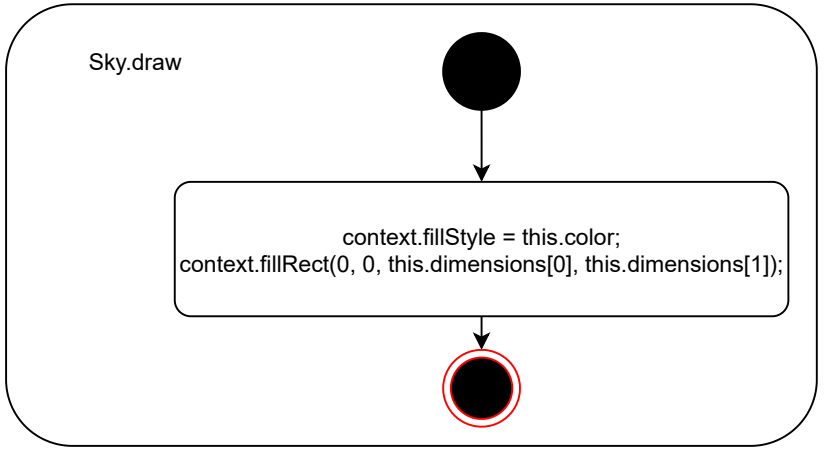

Hive.draw
i = 0;
[i >= beeArray.length]
[i < beeArray.length]
beeAray[i].draw()

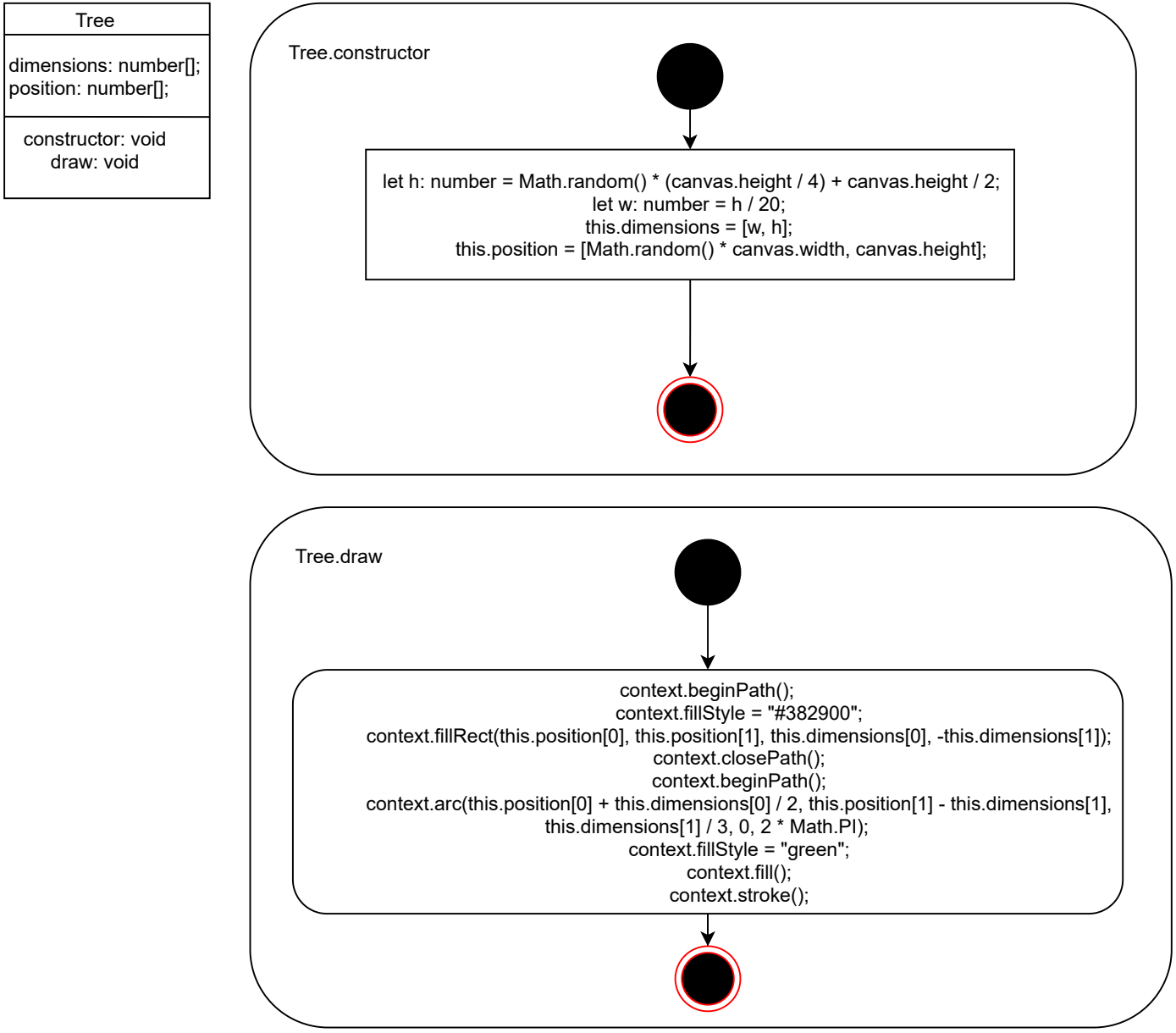

Hive.update
i = 0;
[i >= beeArray.length]
[i < beeArray.length]
beeAray[i].move()


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();

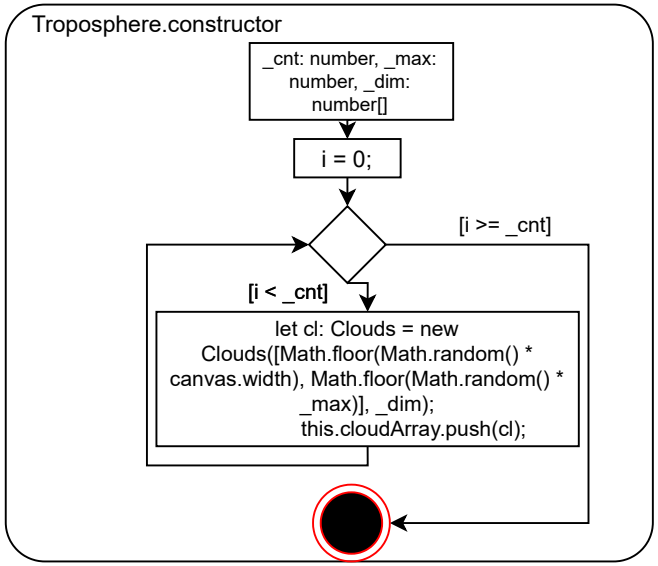

Sky
dimensions: number[];
color: string;
constructor: void
draw: void

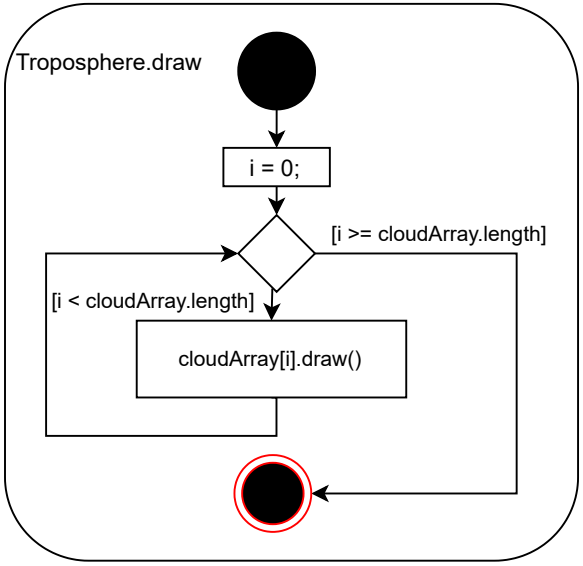

Sky.constructor
_c: string
this.dimensions = [canvas.width,
canvas.height];
this.color = _c;

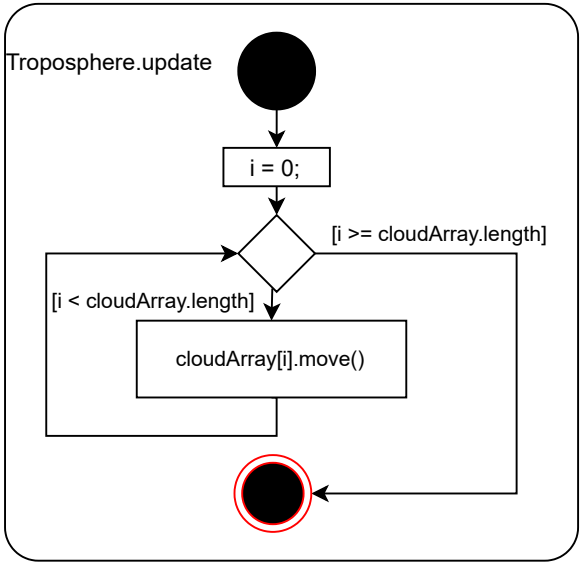

Sky.draw
context.fillStyle = this.color;
context.fillRect(0, 0, this.dimensions[0], this.dimensions[1]);

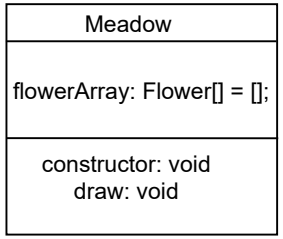

Tree
dimensions: number[];
position: number[];
constructor: void
draw: void
Tree.constructor
let h: number = Math.random() * (canvas.height / 4) + canvas.height / 2;
let w: number = h / 20;
this.dimensions = [w, h];
this.position = [Math.random() * canvas.width, canvas.height];
Tree.draw
context.beginPath();
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();

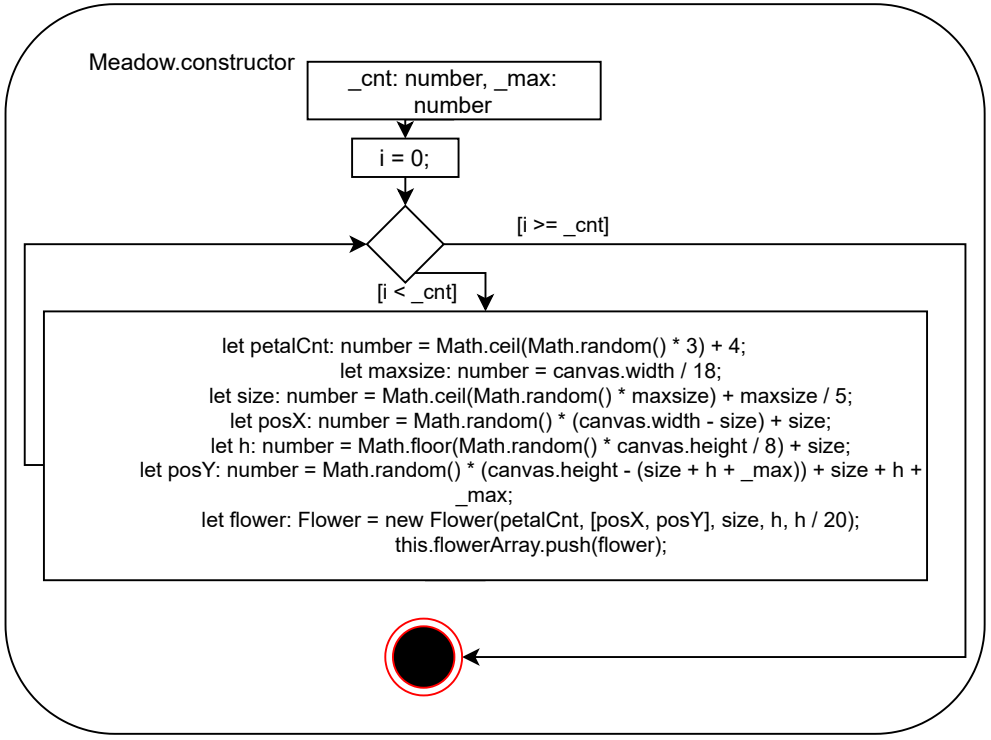
Troposphere
cloudArray: Clouds[] = [];
constructor: void draw: void update: void

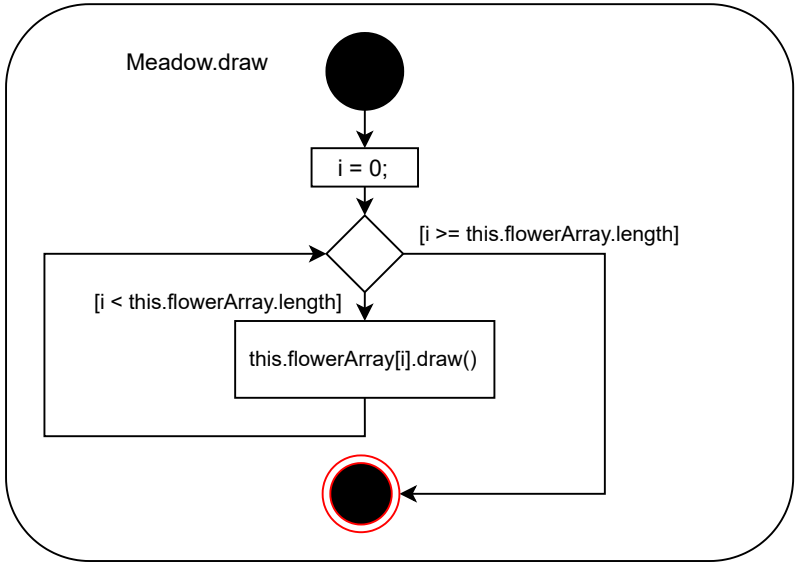







Meadow
flowerArray: Flower[] = [];
constructor: void
draw: void


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()