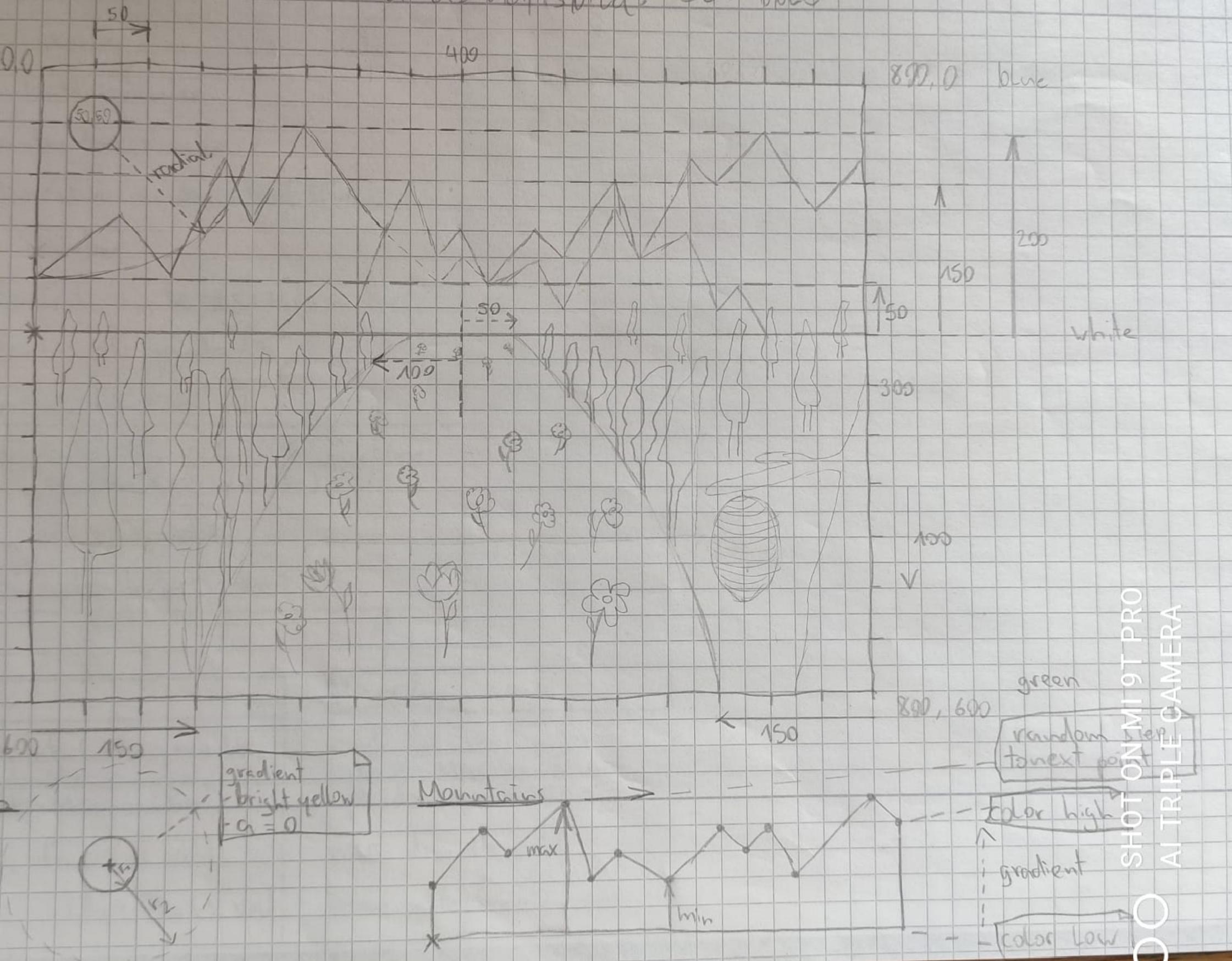


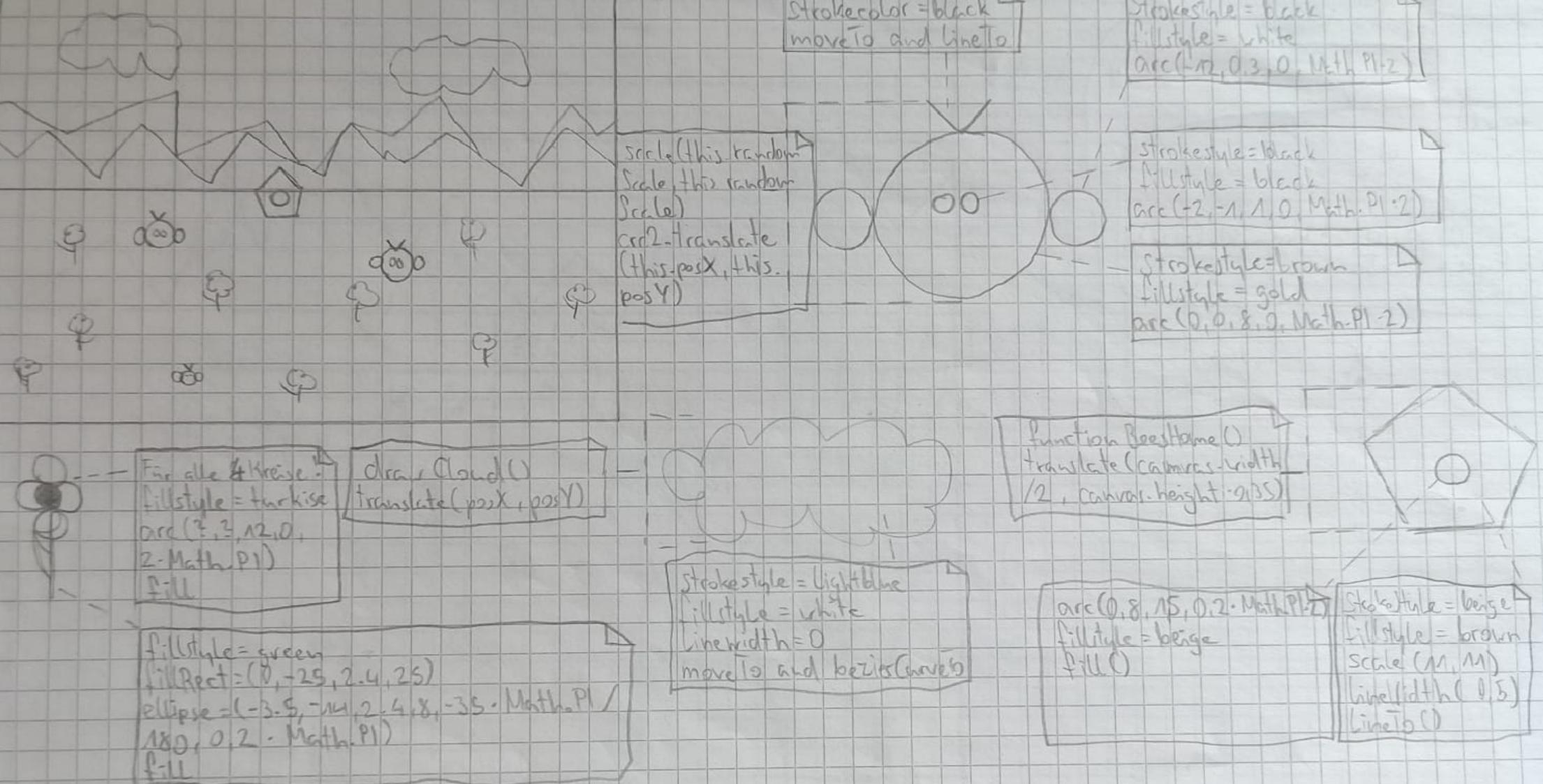
Landschaftsbild: Scribble



SHOT ON MI 9T PRO
AI TRIPLE CAMERA



Scribble



Class Diagrams

FlowerM

-xPos: number

-yRandomMin: number

-yRandomMax: number

nectarValue: number = Math.floor...

-yPos: number

randomScale: number = 0.5 + Math.random...

nectarLength: number = 5

nectarCounter: number = 0

constructor(-xPos: number, -yRandomMin: number, -yRandomMax: number)

public draw(): void

public updateNectar(): void

Flower1

constructor(-xPos: number,
-yRandomMin: number,
-yRandomMax: number)

public draw(): void

public updateNectar(): void

Flower2

constructor(-xPos: number,
-yRandomMin: number,
-yRandomMax: number)

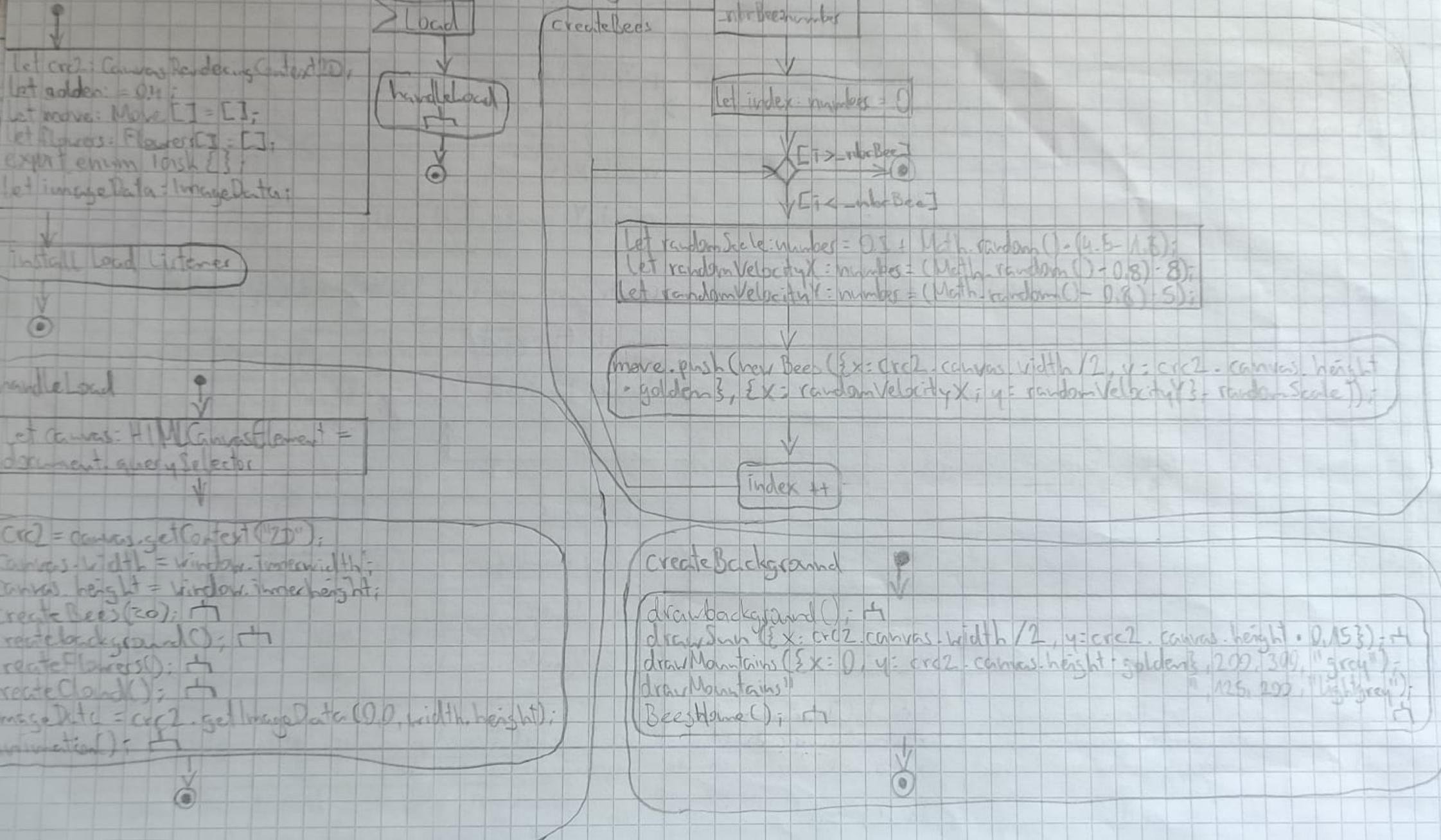
public draw(): void

public updateNectar(): void



SHOT ON MI 9T PRO
AI TRIPLE CAMERA

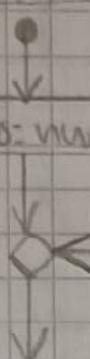
Activity diagram: MainAA



Activitydiagramm: Main/M

Create Flowers

Let xPos: number = 0



Let flowerType: number = Math.floor(...)

[flowerType=1] [flowerType=-1]

flowers.push(new Flower1(
xPos, 5 + (crc2.canvas.height * golden), crc1.
canvas.height * 0.9))

flowers.push(new Flower2(
xPos, 5 + (crc2.canvas.height * golden), crc2.
canvas.height * 0.9))

xPos += 10 + Math.random() * (50 - 10)

[xPos >= canvas.width]
[xPos < canvas.width]

Create Cloud

move.push(new Cloud({x: crc2.canvas.width - 10, y: crc2.canvas.height - 10}, {x: 0.5, y: 0.15}));

animation

requestAnimationFrame(animation),
crc2.clearRect(0, 0, crc2.canvas.width,
crc2.canvas.height); crc2.putImageData(
imageData, 0, 0);

Let index: number = 0

index++

move[index].update();
move[index].draw()

index++ < flowers.length

flowers[index].updateNectar

Activity diagram: background/1

interface Vector

drawMountains
- position: Vector;
- min: number
- max: number
- color: string

Let StepMin: number = 50;
let StepMax: number = 150;
let x: number = 0;

crc2.save();
crc2.translate(-position.x, -position.y);
crc2.beginPath();
crc2.moveTo();
crc2.lineTo();

crc2.fillStyle = gradient
crc2.fillRect()

x += StepMin + Math.random() * (StepMax);
let y: number = -height - Math.random() * (-width - height);

line to (x y)

[x < canvas.width]

lineTo(x, 0);
closePath();
crc2.fillStyle = color;
crc2.fill();
crc2.restore();



SHOT ON MI 9T PRO
AI TRIPLE CAMERA

Activity diagram: background10

-position: Vector

```
crc2.save();
crc2.moveTo(20,20);
crc2.translate(20,20);
crc2.beginPath();
crc2.arc(0,0,100,0,2*Math.PI, false);
crc2.fillStyle = colorSun;
crc2.fill();
```



Activity Diagram: MoveM / UpdateNectar

```

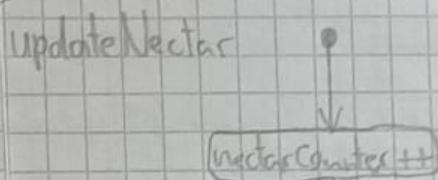
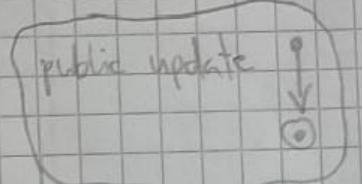
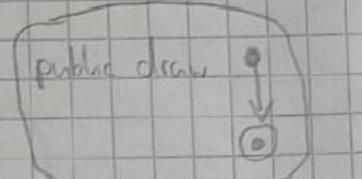
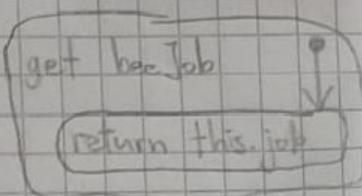
post abstract class Animation
    protected job: boolean = false;
    "Task": Task = Task.FlyAround,
    flowerIndex: number;
    posx: number;
    posy: number;
    flowerTargetX: number;
    flowerTargetY: number;
    velocityX: number;
    velocityY: number;
  
```

```

constructor
  -position: Vector
  -velocity: Vector
  this.posX = position.x
  this.posY = position.y
  this.velocityX = velocity.x
  this.velocityY = velocity.y
  
```

```

setTask
  -task: Task
  this.task = task
  this.job = false
  this.job = true
  
```



```

  if (nectar.length < 15) {
    nectarValue = nectars[counter]
    vectorLength += 2
    nectarValue = Math.random()
  }
  
```

draw NectarCircleBar for
Flower1 & Flower2

setFlowerIndex [index: number]

```

  this.flowerIndex = index
  this.flowerTargetX = flowers[this.flowerIndex].flowerPos.x
  this.flowerTargetY = flowers[this.flowerIndex].flowerPos.y
  
```

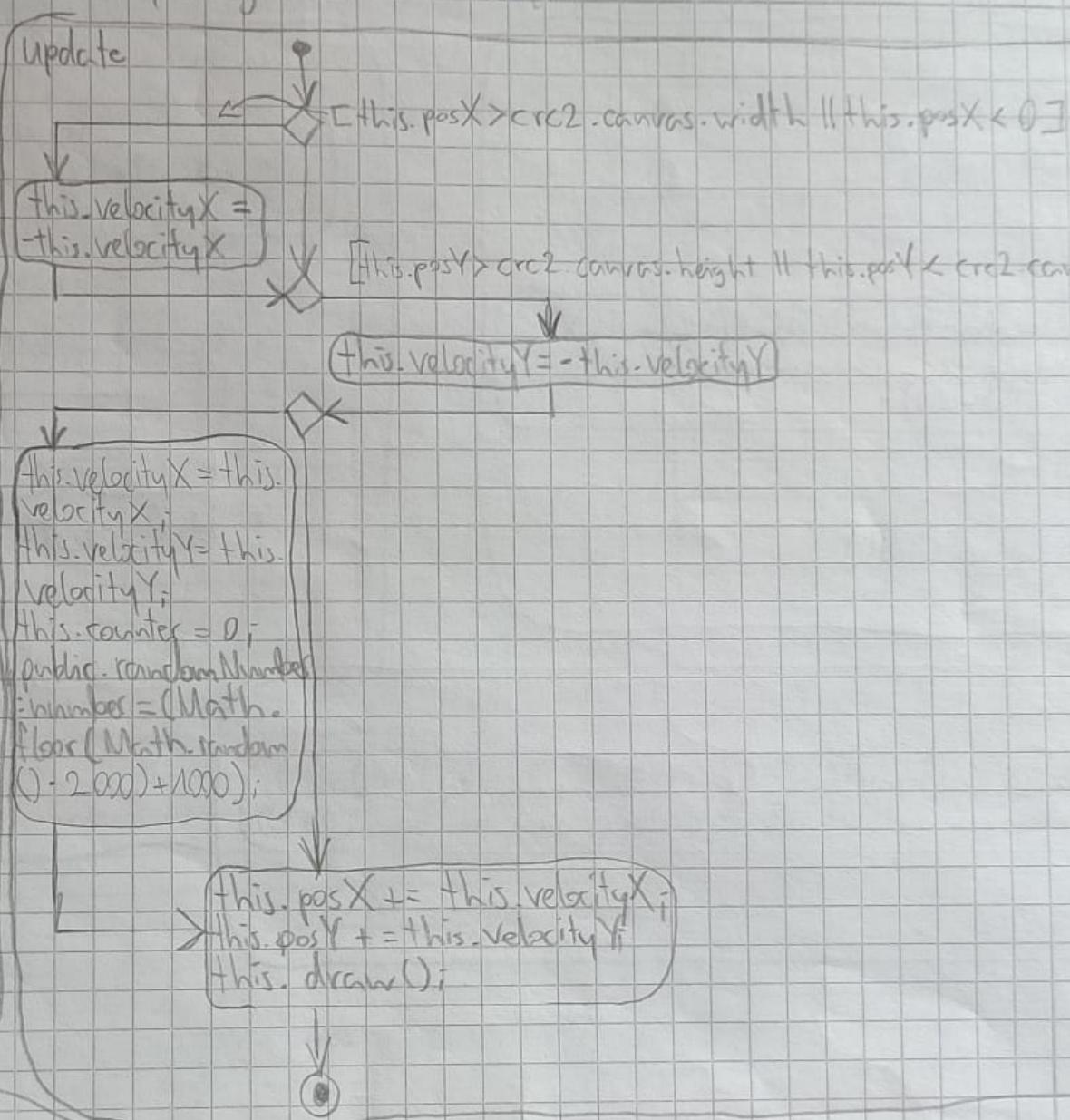
Activity diagram: beeAI

```
private randomScale: number;  
private randomNumber: number;  
= (Math.floor(Math.random() * 2000)  
+ 100);  
counter = 0;
```

Constructor

```
-position: Vector;  
velocity: Vector;  
randomScale: number
```

```
Super(-position, -velocity);  
this.randomScale = randomScale
```



draw

```
crc2.save();  
crc2.translate(this.posX, this.posY);  
crc2.scale(+this.randomScale, this.randomScale);  
crc2.restore();
```



SHOT ON MI 9T PRO
AI TRIPLE CAMERA



ActivityDiagram: flowers()

```
public xPos: number;
public yPos: number;
public yRandomMax: number;
public yRandomMin: number;
```

constructor

```
-xPos: number;
-yRandomMax: number;
-yRandomMin: number;
```

```
Super(-xPos, -yRandomMin,
      -yRandomMax)
```

```
Let randomScale: number = 0.13 + Math.random() * (1.3 - 0.13);
let y: number = this.yRandomMin + Math.random() * (this.yRandomMax - this.yRandomMin);
```

```
crc2.save();
crc2.translate(this.xPos, y);
crc2.scale(randomScale, randomScale);
```

[flowerType == 1] X [flowerType != 1]

draw Flower 1

draw Flower 2

BeestHome

```
crc2.save();
crc2.translate(crc2.canvas.width / 2, crc2.canvas.height - 0.35);
crc2.scale(M, M)
crc2.lineWidth = 0.5;
crc2.strokeStyle = "beige";
BeestHome wird gezeichnet
crc2.restore();
```