

Class Diagrams

Clouds

```

Constructor(-position: Vector, -velocity: Vector)
update(): void
draw(): void
    
```

Move

```

posX: number,
posY: number,
velocityX: number,
velocityY: number,
    
```

Bees

```

randomNumber: number = (Math.floor(Math.random() * 2000) + 1000);
randomScale: number;
Counter: number,
    
```

```

Constructor(-position: Vector, -velocity: Vector)
update(): void
draw(): void
    
```

```

Constructor(-position: Vector, -velocity: Vector, -randomScale: number)
draw()
update()
    
```

Flowers

```

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

Vector

```

x: number;
y: number;
    
```

Flower1

```

Constructor(-flowerType: number,
-xPos: number, -yRandomMin:
number, -yRandomMax: number)
draw(): void
    
```

Flower2

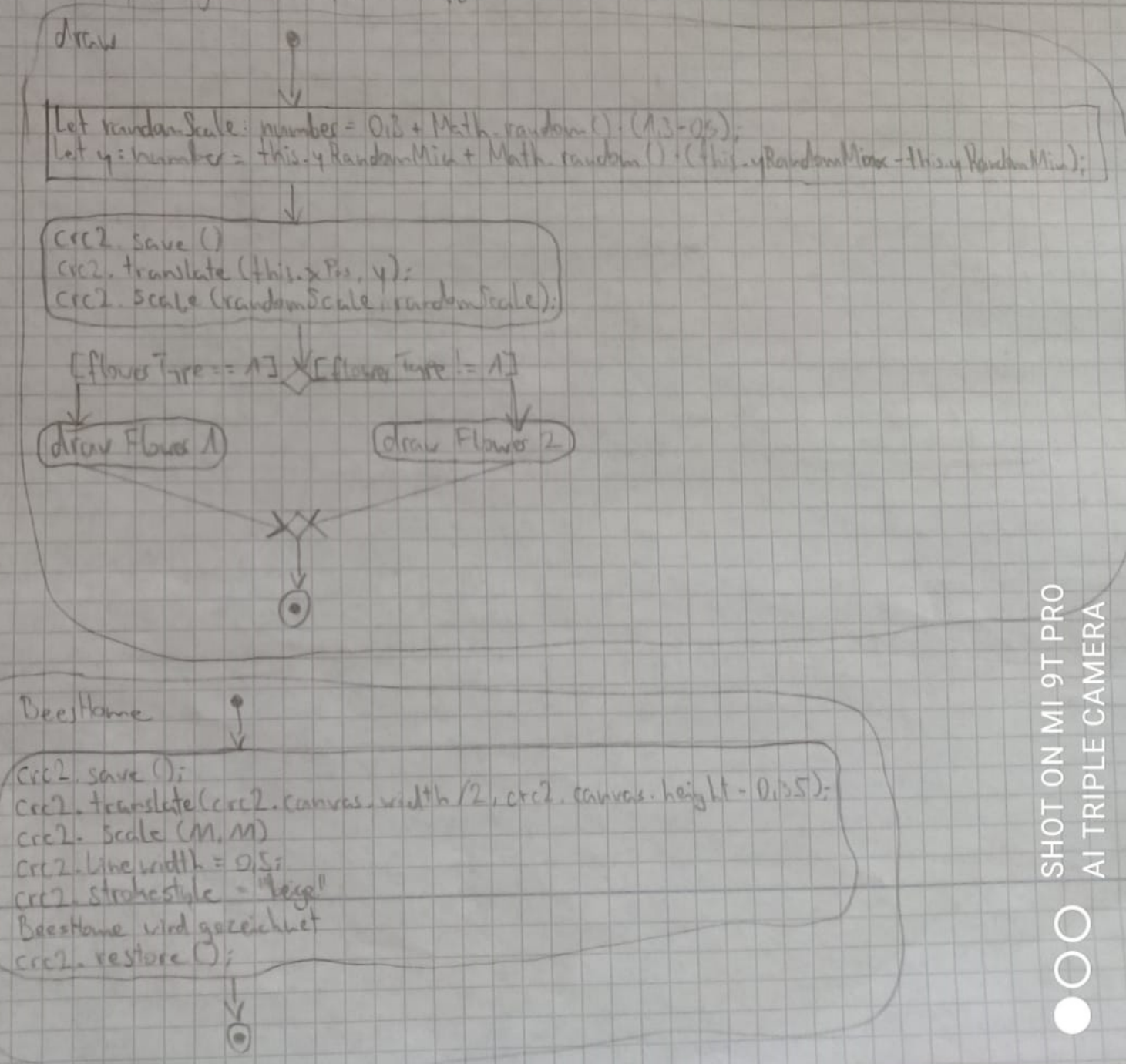
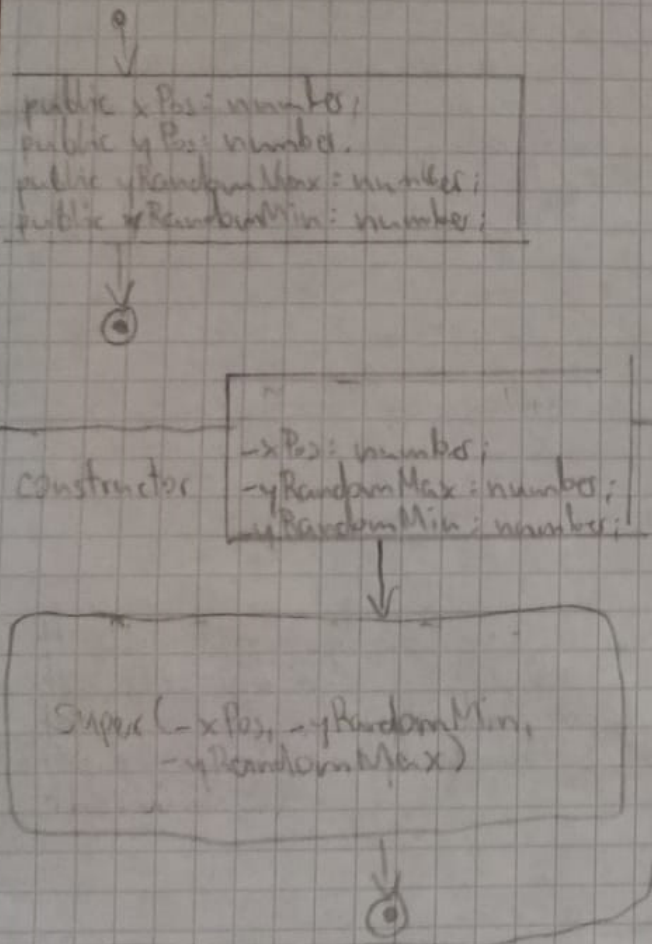
```

Constructor(flowerType: number,
-xPos: number, -yRandomMin:
number, -yRandomMax: number)
draw(): void
    
```

```

Constructor(flowerType: number, xPos: number, yRandomMin: number,
-yRandomMax: number)
draw(): void
    
```

Activity Diagram: flowers



Activity diagram: main10

```

    Let crc2: CanvasRenderingContext2D;
    Let golden: = 0.4;
    Let move: Move[] = [];
    Let flowers: Flower[] = [];

    Let imageData: ImageData;
  
```

Install Load Listener

handleLoad

```

    let canvas: HTMLCanvasElement =
    document.querySelector
  
```

```

    crc2 = canvas.getContext("2D");
    canvas.width = window.innerWidth;
    canvas.height = window.innerHeight;
  
```

createBees(30);

createBackground();

createFlowers();

createCloud();

```

    imageData = crc2.getImageData(0, 0, width, height);
  
```

animation();

Load

handleLoad

createBees

-nbrBeeNumber

Let index: number = 0

[i] > -nbrBee

[i] < -nbrBee

```

    Let randomScale: number = 0.8 + Math.random() * (4.5 - 1.8);
    Let randomVelocityX: number = (Math.random() - 0.8) * 8;
    Let randomVelocityY: number = (Math.random() - 0.8) * 5;
  
```

```

    move.push(new Bees({x: crc2.canvas.width/2, y: crc2.canvas.height *
    golden}, {x: randomVelocityX, y: randomVelocityY, randomScale}));
  
```

index ++

createBackground

drawBackground();

drawSun({x: crc2.canvas.width/2, y: crc2.canvas.height * 0.1});

drawMountains({x: 0, y: crc2.canvas.height * golden, 200, 300, "mountain1.png"});

drawMountains({x: 125, 200, "mountain2.png"});

BeesHive();

Activity diagram: main10

createFlowers

let xPos: number = 0;

Flowers.push(new Flowers(Math.floor(Math.random() * 2) + 1, xPos, 5 + (crc2.canvas.height * golden), crc2.canvas.height * 0.8));
xPos += 10 + Math.random() * (50 - 10);

[xPos < canvas.width]

[xPos >= canvas.width]

CreateCloud

move.push(new Cloud({x: crc2.canvas.width * 5, y: crc2.canvas.height * 0.15}));
move.push(new Cloud({x: crc2.canvas.width * 1, y: crc2.canvas.height * 0.85}));

animation

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

let index: number = 0;

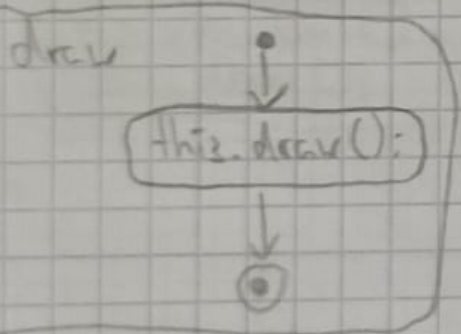
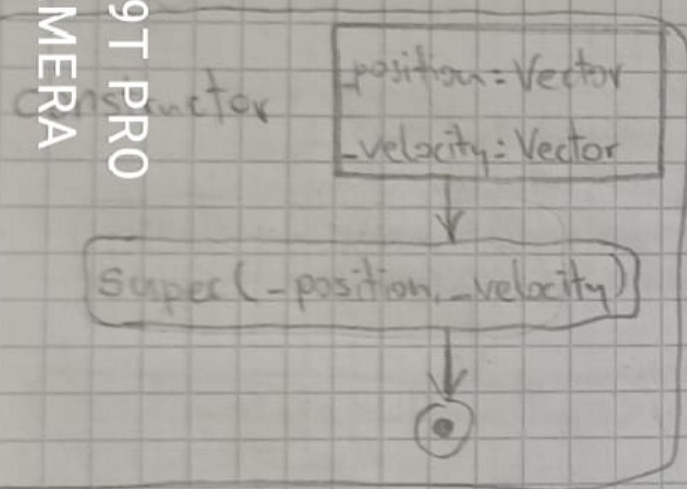
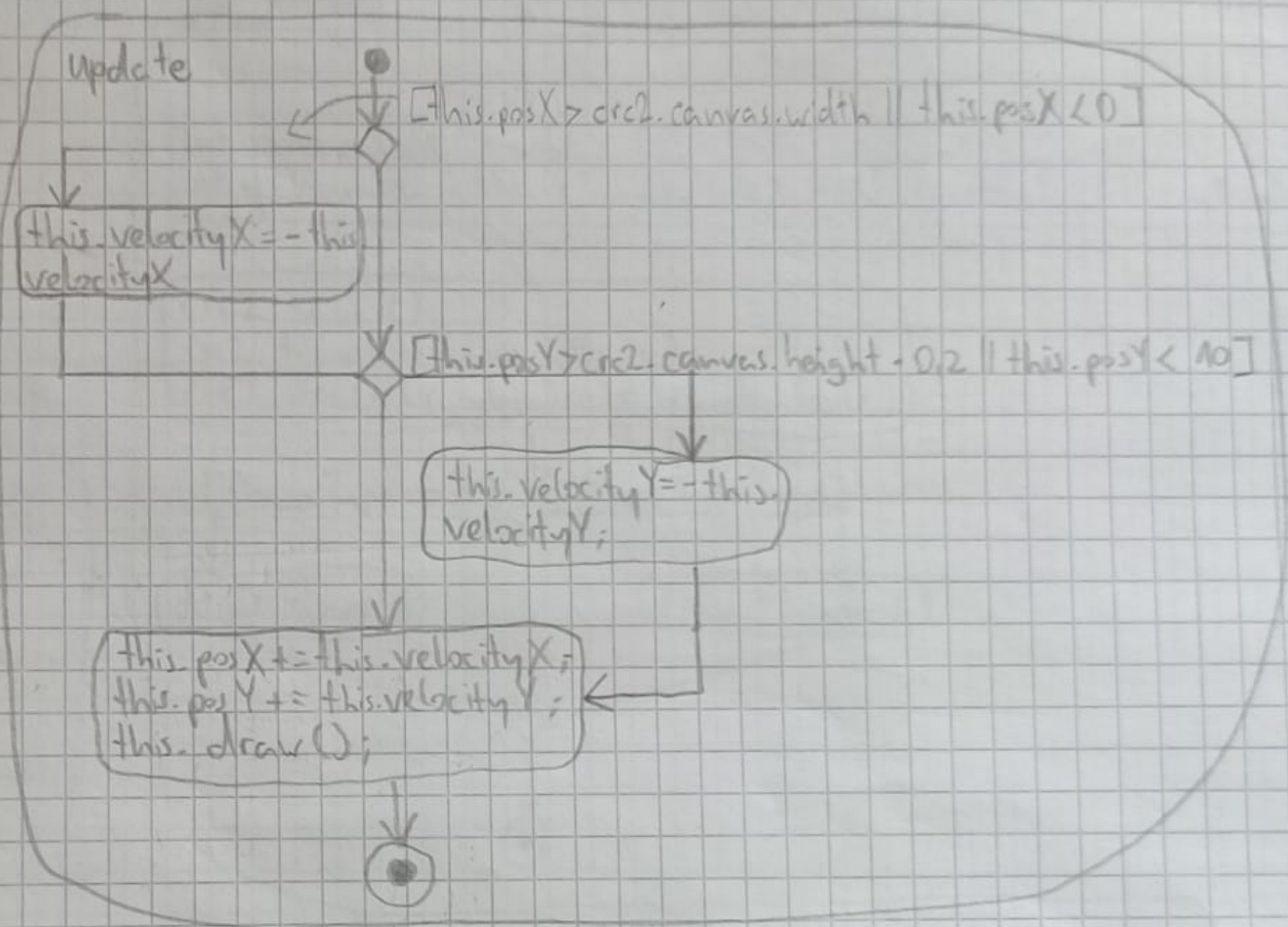
[index > move.length]

[index < move.length]

index++

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

Activity diagram: cloud 10



Activity diagram: background10

draw Mountains

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

let stepMin: number = 50;
let stepMax: number = 100;
let x: number = 0;

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

$x += \text{stepMin} + \text{Math.random()} * (\text{stepMax} - \text{stepMin});$
 $\text{let } y: \text{number} = -\text{height} - \text{Math.random()} * (-\text{width} - \text{height});$

Line to (x, y)

[x < canvas.width]

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

Activity diagram: background 10

-position: Vector



```
cr2.save();  
cr2.moveTo(20, 20);  
cr2.translate(20, 20);  
cr2.beginPath();  
cr2.arc(0, 0, 100, 0, 2 * Math.PI, false);  
cr2.fillStyle = colorSun;  
cr2.fill();
```



Activity diagram: bees10

```

public randomScale: number;
public randomNumber: number;
= (Math.floor(Math.random() * 2000)
+ 1000);
counter = 0;
    
```

Constructor

```

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

Super(-position, -velocity)

draw

```

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

Update

[this.posX > crc2.canvas.width || this.posX < 0]

```

this.velocityX =
-this.velocityX
    
```

[this.posY > crc2.canvas.height || this.posY < crc2.canvas.height - 0.4]

```

this.velocityY = -this.velocityY
    
```

```

this.velocityX = this.
velocityX;
this.velocityY = this.
velocityY;
this.counter = 0;
public randomNumber
= number = (Math.
floor(Math.random
() * 2000) + 1000);
    
```

```

this.posX += this.velocityX;
this.posY += this.velocityY;
this.draw();
    
```

