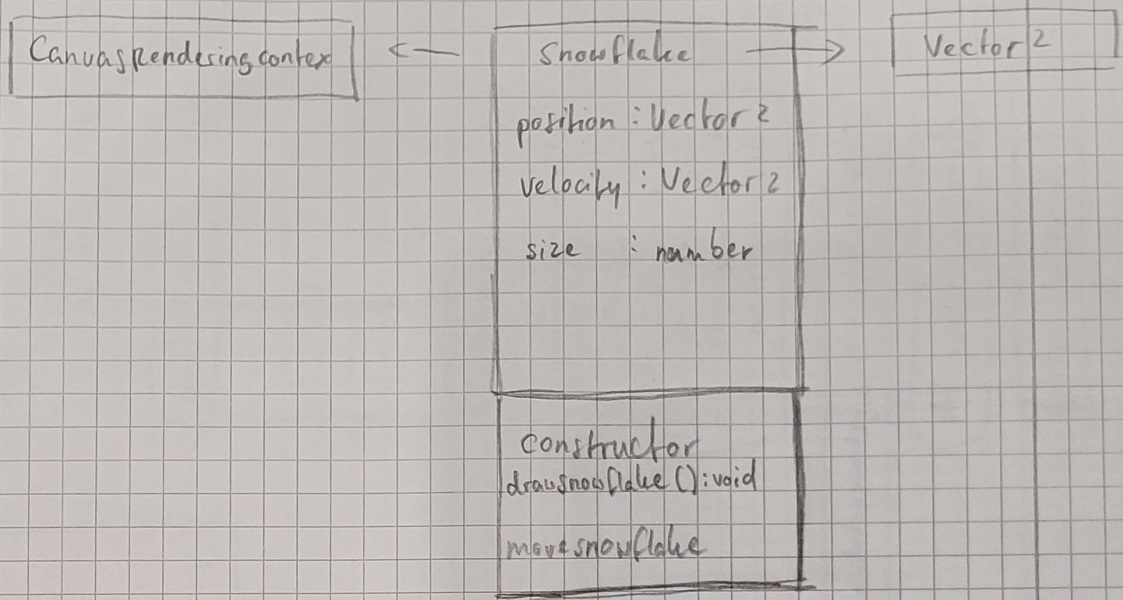
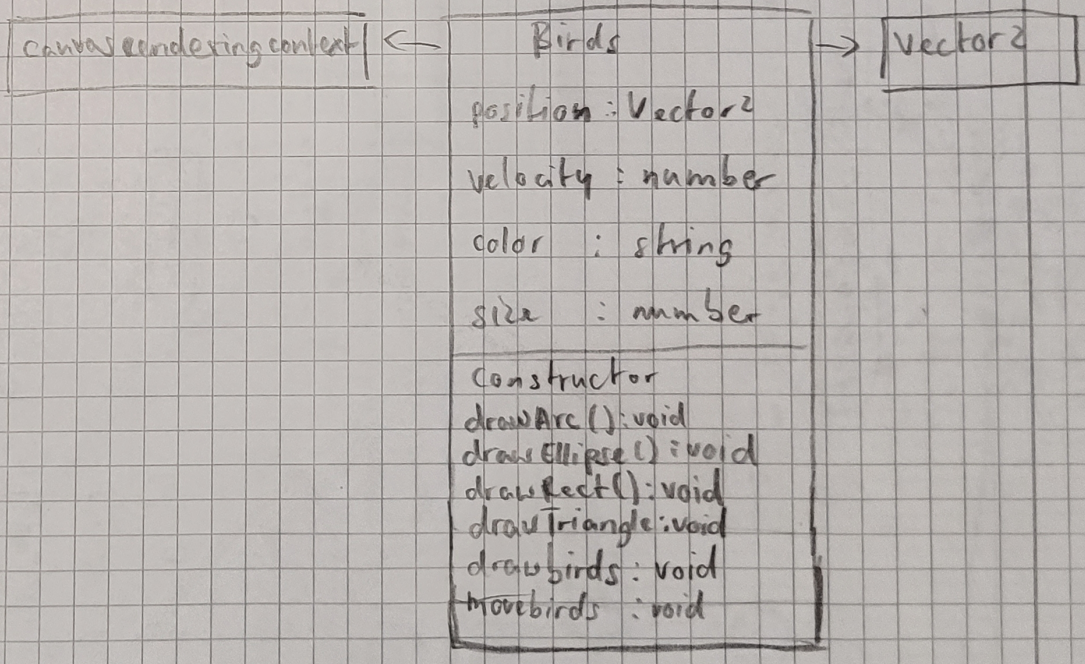
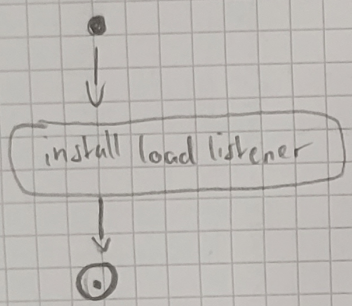
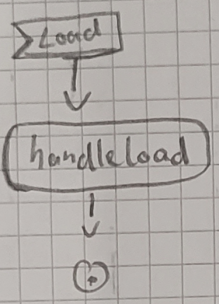
Vogelhaus : ClassDiagramm

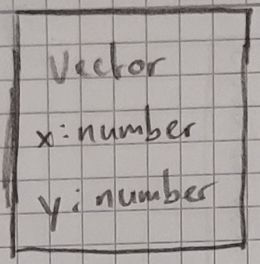


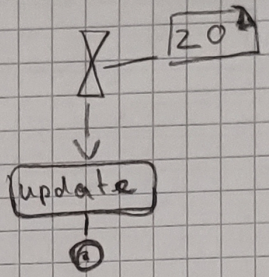



Vogelhaus : AD - scripts



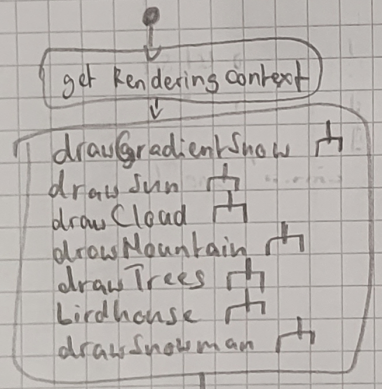




```
let getimageData : ImageData;
let snowflakesArray : Snowflakes[] = []
let gbirds : Birds[] = []
```

handleload

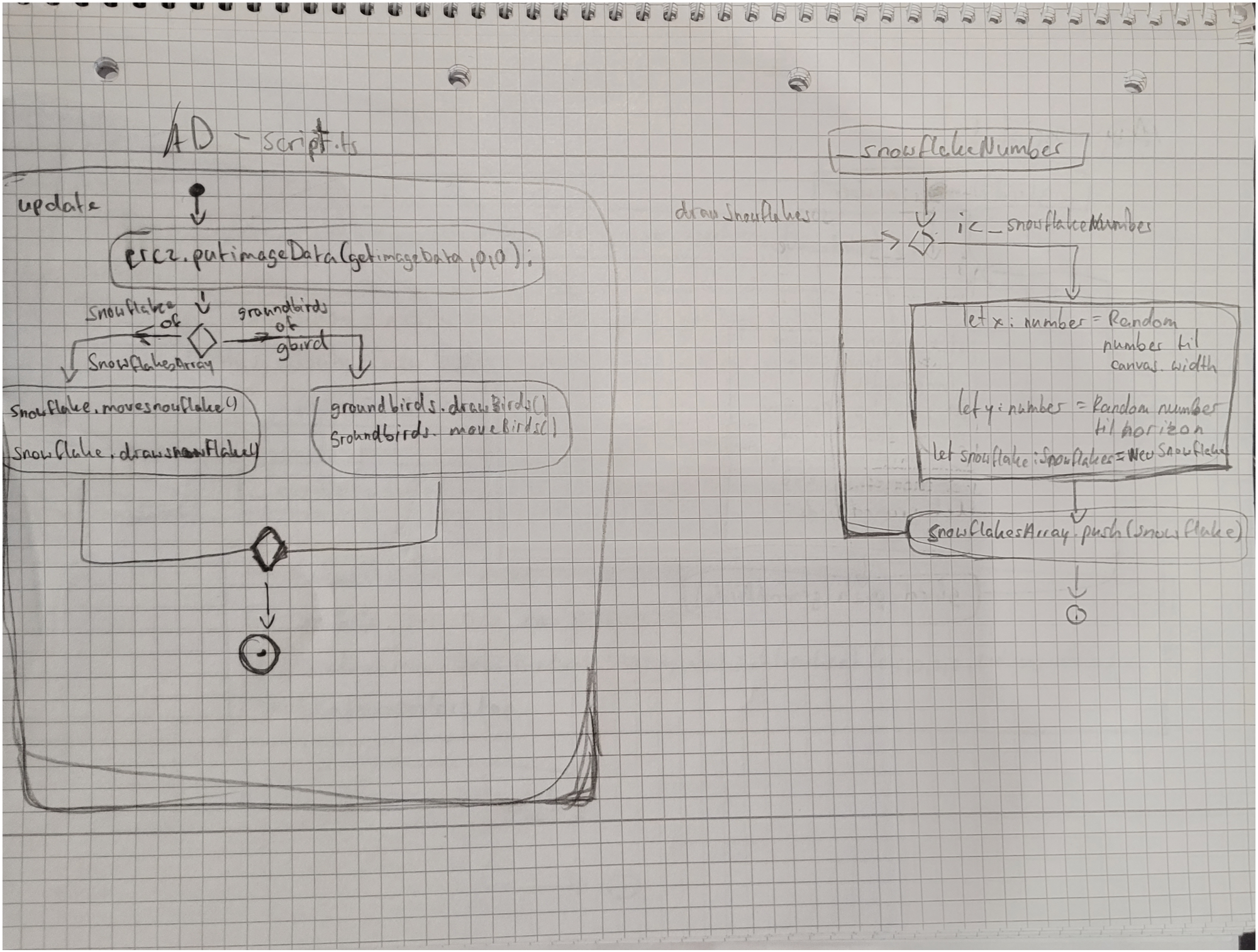



getimageData = crc2.getimageData(0,0, canvaswidth, canvas.height)

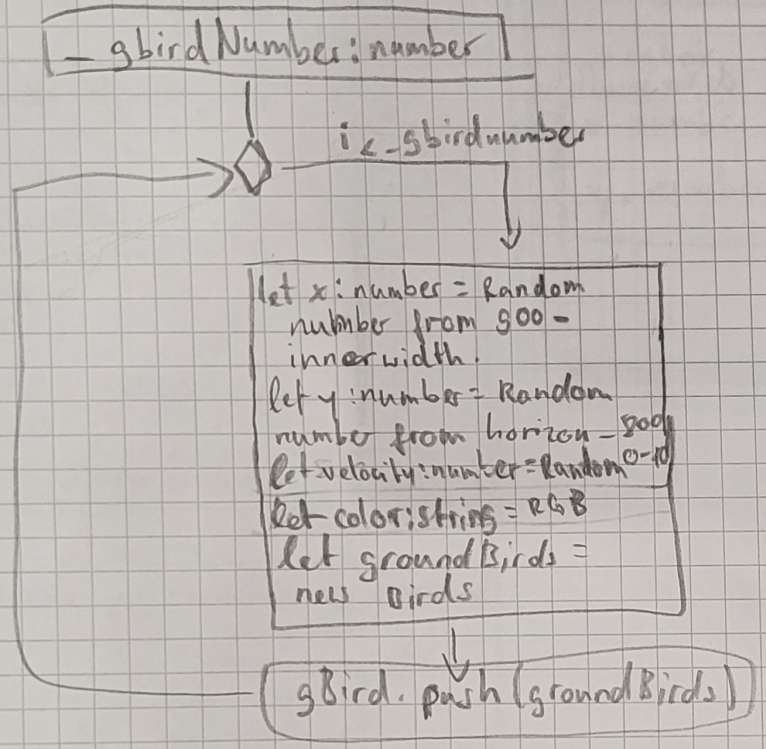
drawSnowflakes

drawbird

window.set.intervall(update, 20); → ⊙


AD - script.ts
update
ctx.putImageData(getImageData,0,0);
snowflakes of SnowflakesArray
groundbirds of gbird
snowflake.movesnowflake()
snowflake.drawsnowflake()
groundbirds.drawBirds()
groundbirds.moveBirds()
drawSnowflakes
snowflakeNumbers
ic_snowflakeNumbers
let x: number = Random number til canvas.width
let y: number = Random number til horizon
let snowflake: Snowflakes = New Snowflake
snowflakesArray.push(snowflake)
i

AD - script.ts


_gbirdNumber: number
i < _gbirdnumber
let x: number = Random number from 900 - innerwidth.
let y: number = Random numbe from horizon - 500
let velocity: number = Random 0-10
let color: string = RGB
let groundBirds = new Birds
gBird.push(groundBirds)

# AD – Snowflake

Constructor

```
_position: Vector, _velocity: Vector, _size: number,
```

```
this.position = new Vector2(_position.x, _position.y);
this.velocity = new Vector2(_velocity.x, _velocity.y);
this.size = _size
```

drawSnowflakes

```
_numberparticles: number,
```

```
let radiusParticle: number = this.size;
let particle: Path2D = new Path2D();
let gradientsnow: CanvasGradient = crc2.createRadialGradient(0,0,0,0,0,radiusparticle);
gradiensnow.addcolorstop
```

draw particle.arc

restore → ⊙

AD - SnowFlake