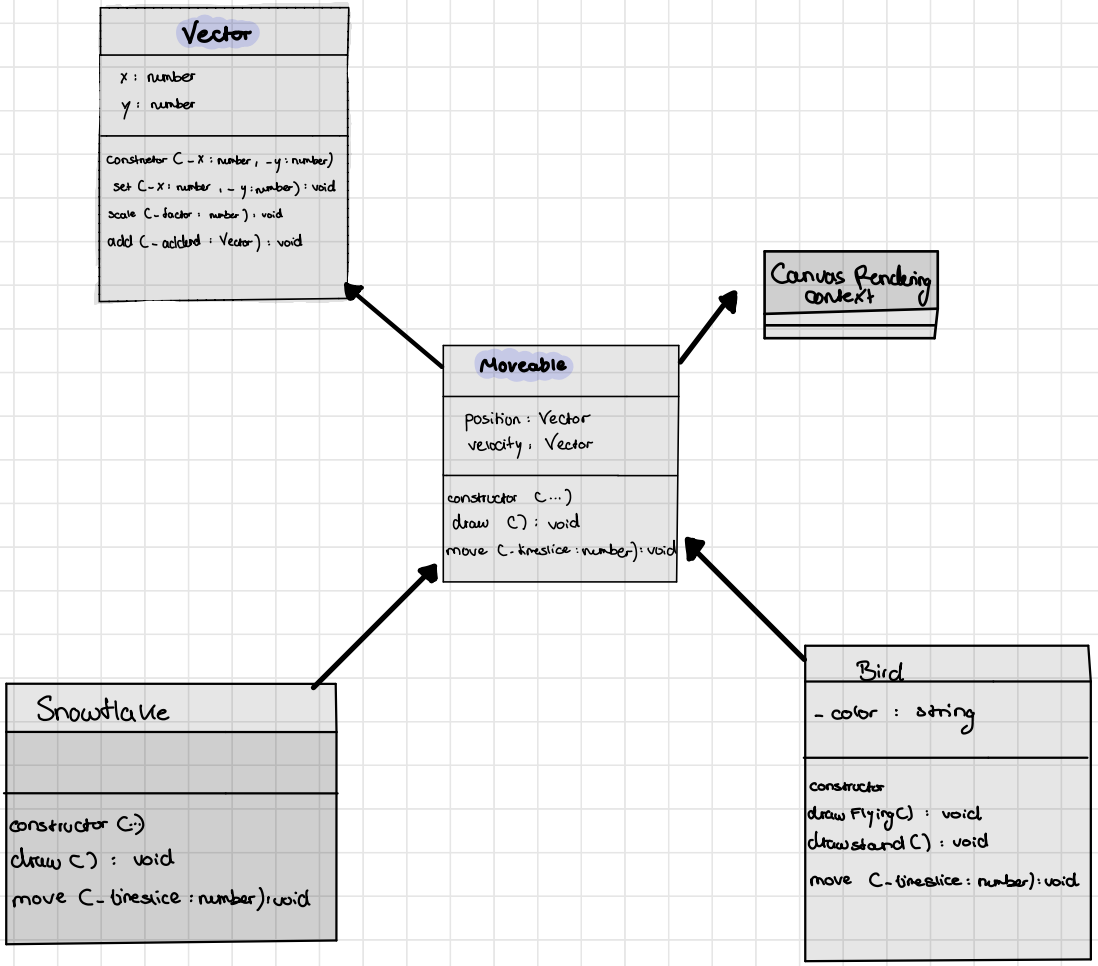
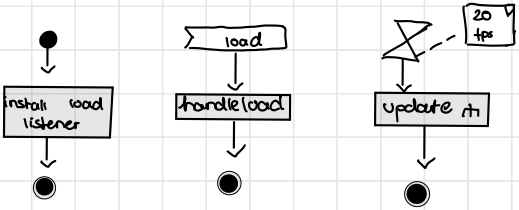
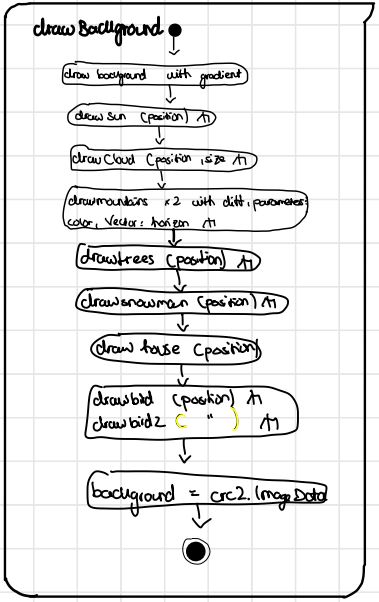
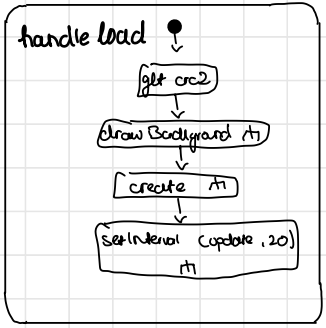

Vector
x : number
y : number
constructor (_x : number , _y : number)
set (_x : number , _y : number) : void
scale (_factor : number) : void
add (_addend : Vector) : void
Canvas Rendering Context
Moveable
position : Vector
velocity : Vector
constructor (...)
draw () : void
move (_timeslice : number) : void
Snowflake
constructor (...)
draw () : void
move (_timeslice : number) : void
Bird
- color : string
constructor
drawFlying() : void
drawStand() : void
move (_timeslice : number) : void

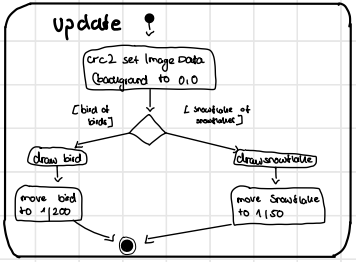
activity diagram - main

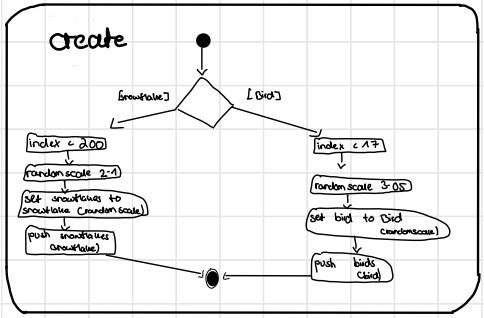


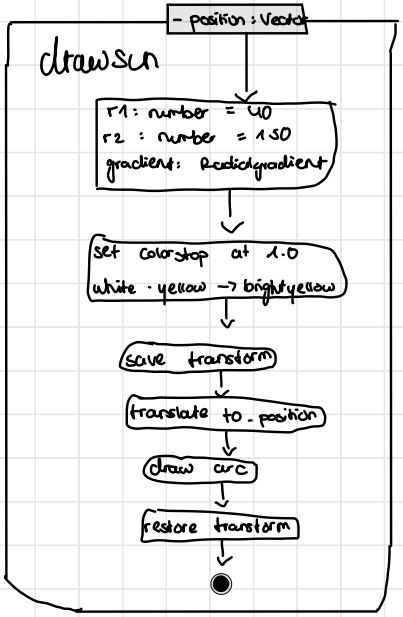
crc2
golden
x : number
y : number
background : ImageData
snowflakes : Snowflake [ ]
moveables : Moveable [ ] • [ ]

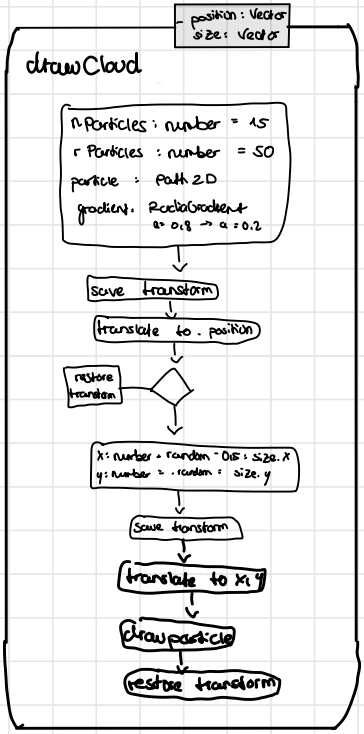


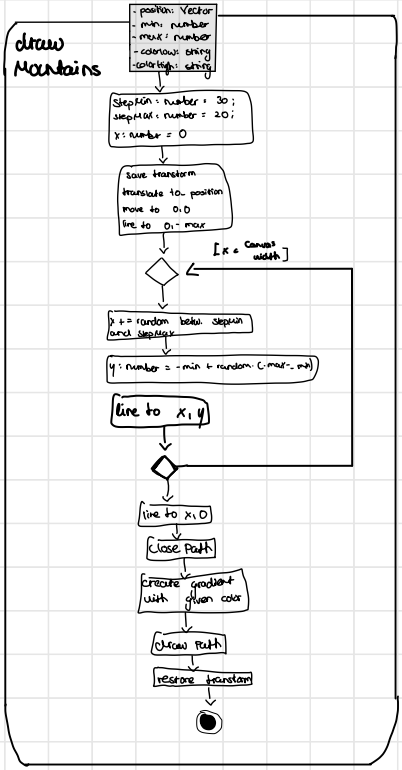


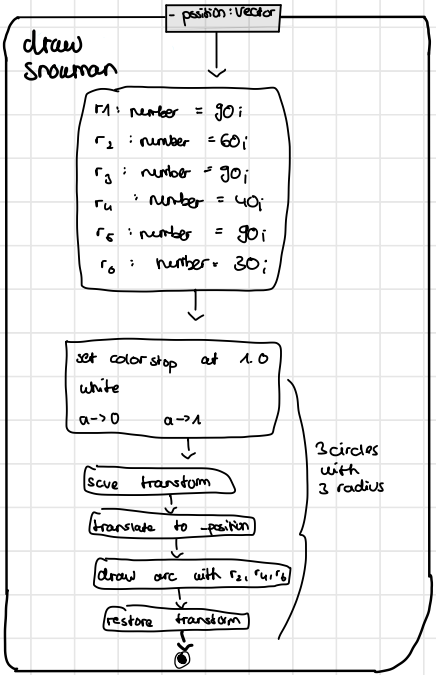


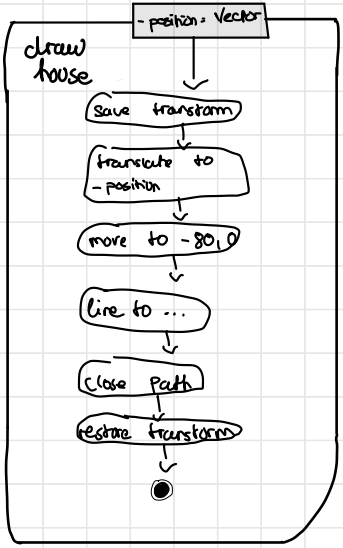


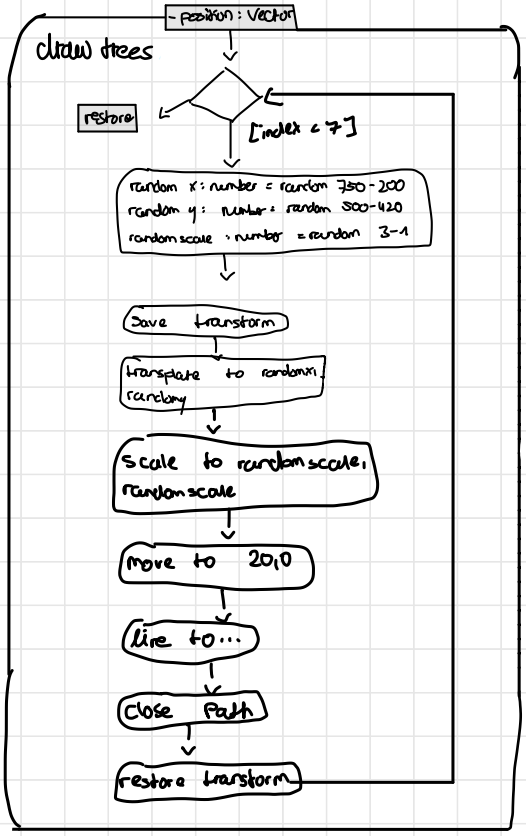


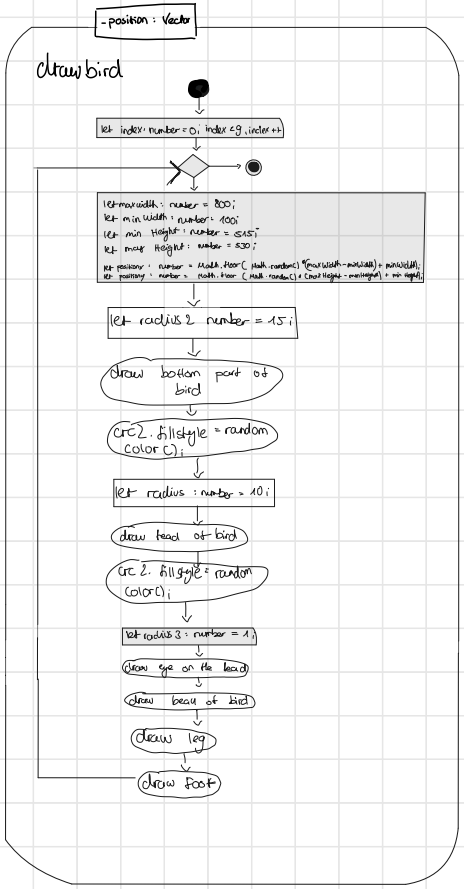

drawCloud
- position : Vector
size : Vector
nParticles : number = 15
rParticles : number = 50
particle : Path2D
gradient : RadialGradient
a= 0.8 → a = 0.2
save transform
translate to -position
restore transform
x: number = random · 0.5 · size.x
y: number = random · size.y
save transform
translate to x, y
draw particle
restore transform

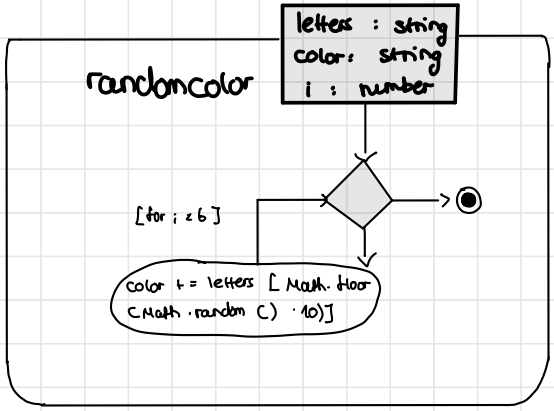

drawMountains
- position: Vector
- min: number
- max: number
- colorLow: string
- colorHigh: string
stepSize : number = 30;
stepHalf : number = 20;
x: number = 0
save transform
translate to position
move to 0,0
line to 0, -max
[x < canvas width]
x += random betw. stepSize and stepSize
y: number = -min + random · (-max - min)
line to x, y
line to x, 0
close Path
create gradient with given color
draw Path
restore transform


drawSnowman
- position : Vector
r1 : number = 90;
r2 : number = 60;
r3 : number = 90;
r4 : number = 40;
r5 : number = 90;
r6 : number = 30;
set colorstop at 1.0
white
a->0 a->1
save transform
translate to -position
draw arc with r2, r4, r6
restore transform
3 circles with 3 radius

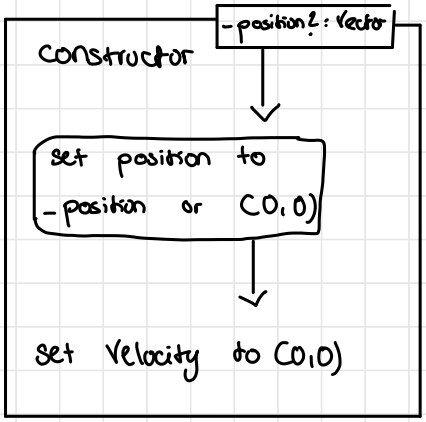

drawHouse
- position = Vector
save transform
translate to -position
move to -80, 0
line to ...
close Path
restore transform

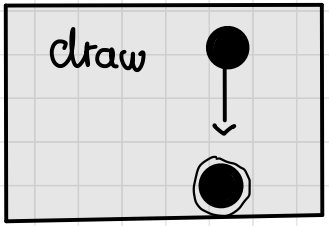

- position : Vector
drawTrees
restore
[index < 7]
random x: number = random 750 - 200
random y: number = random 500 - 420
random scale : number = random 3 - 1
save transform
translate to random x, random y
scale to random scale, random scale
move to 20,0
line to...
close Path
restore transform

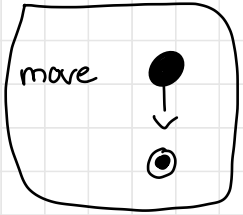

- position : Vector
drawBird
let index: number = 0; index < 9 ; index ++
let maxWidth : number = 800;
let minWidth : number = 100;
let minHeight : number = 345;
let maxHeight : number = 530;
let positionX : number = Math.floor(Math.random() *(maxWidth - minWidth) + minWidth);
let positionY : number = Math.floor (Math.random() * (maxHeight - minHeight) + minHeight);
let radius2 number = 15;
draw bottom part of bird
crc2.fillstyle = random color();
let radius : number = 10;
draw head of bird
crc2.fillstyle = random color();
let radius 3 : number = 1;
draw eye on the head
draw beak of bird
draw leg
draw foot

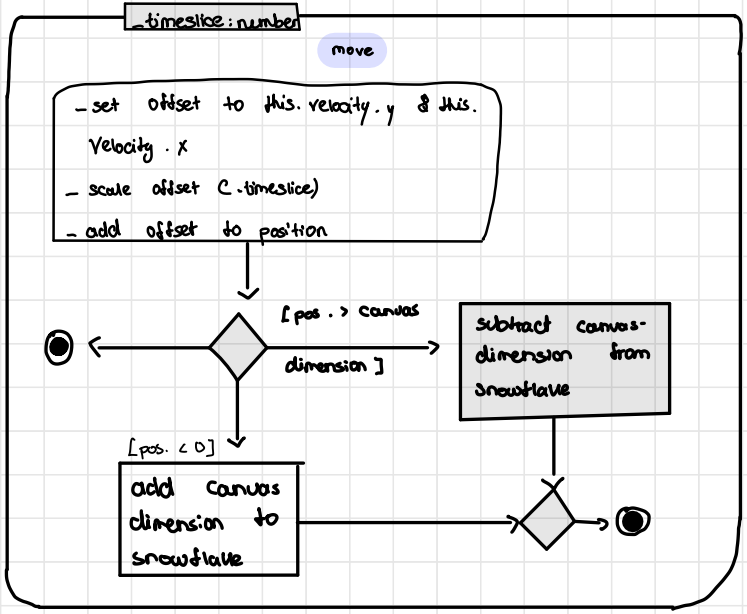

randomcolor
letters : string
color: string
i : number
[for i < 6]
color += letters [Math.floor (Math.random() * 10)]

moveable

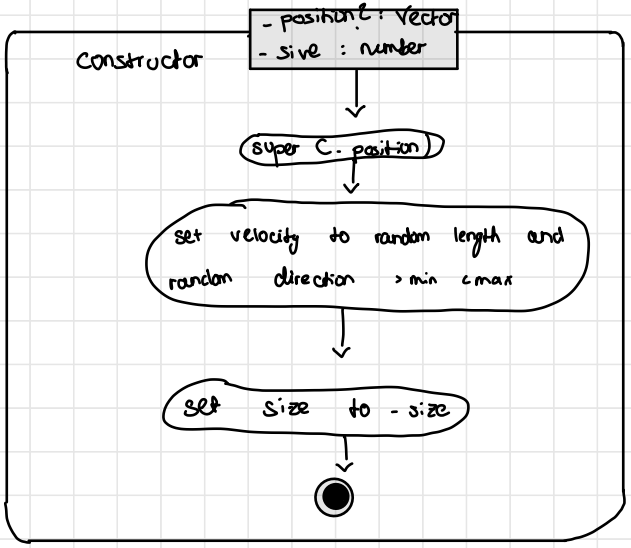

_position? : Vector
constructor
set position to _position or (0,0)
set velocity to (0,0)

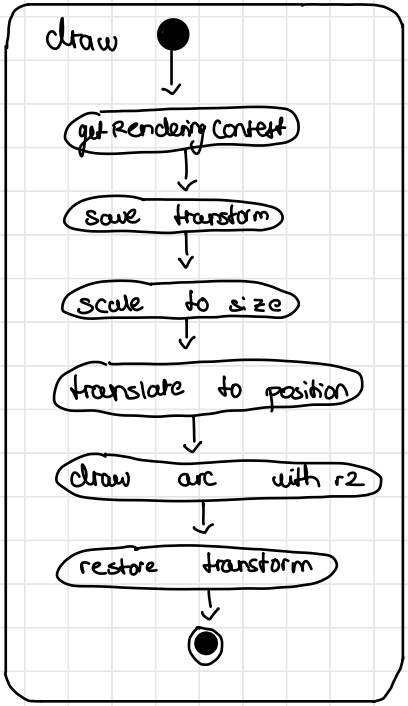

draw

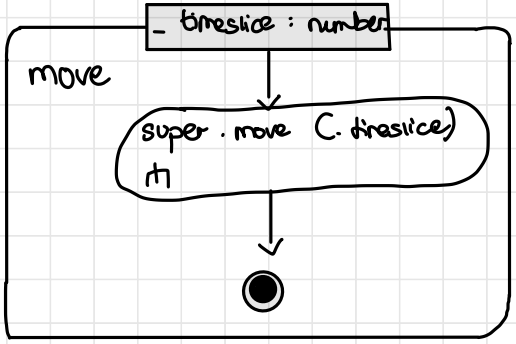

move


_timeslice : number
move
_ set offset to this.velocity.y & this.velocity.x
_ scale offset (_timeslice)
_ add offset to position
[pos. > canvas dimension]
subtract canvas-dimension from snowflake
[pos. < 0]
add canvas dimension to snowflake

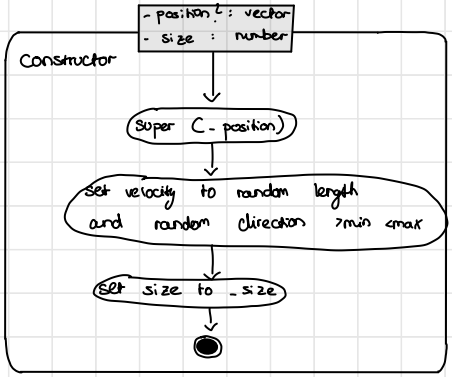
snowflake

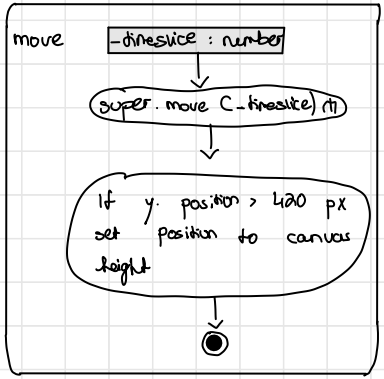

constructor
- position : Vector
- size : number
super C. position)
set velocity to random length and random direction > min < max
set size to - size

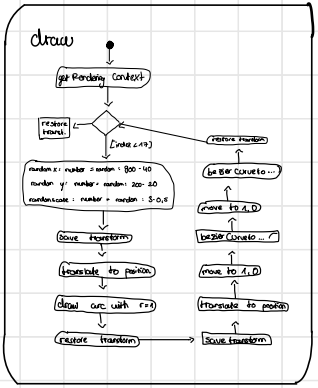

draw
get Rendering Context
save transform
scale to size
translate to position
draw arc with r2
restore transform

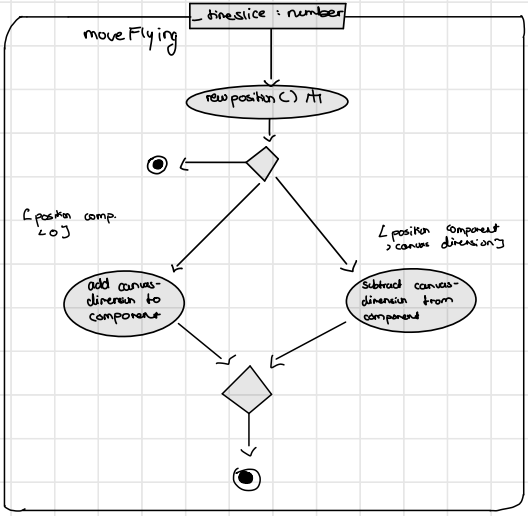

_ timeslice : number
move
super . move C. timeslice)
m

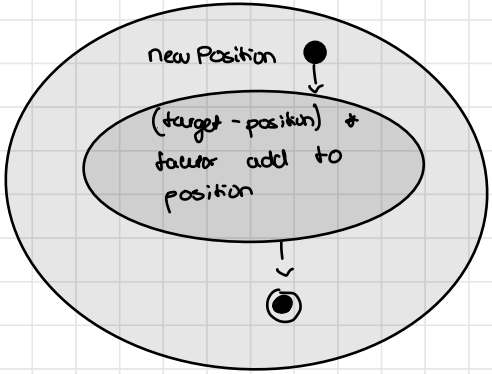
bird

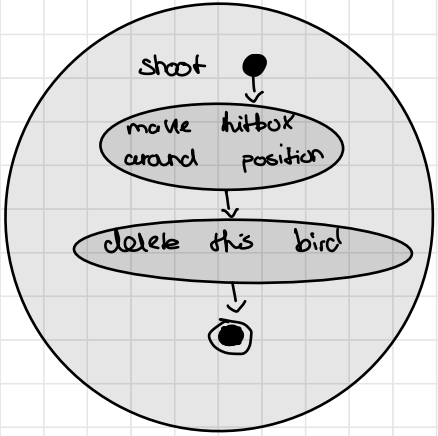

- position : vector
- size : number
constructor
super (_position)
set velocity to random length and random direction >min <max
set size to _size


move
_timeslice : number
super.move (_timeslice) //
if y position > 420 px set position to canvas height


draw
get Rendering context
restore transform
[index < 19]
random x: number = random : 800 - 140
random y: number = random: 200- 20
random scale : number = random : 3-0,8
save transform
translate to position
draw arc with r=4
restore transform
restore transform
bezier curve to ...
move to 1,0
bezier curve to ...
move to 1,0
translate to position
save transform


moveFlying
_timeslice : number
new position () //
[position comp. < 0]
[position component > canvas dimension]
add canvas-dimension to component
subtract canvas-dimension from component


new Position
(target - position) * factor add to position


shoot
move hitbox curound position
delete this bird