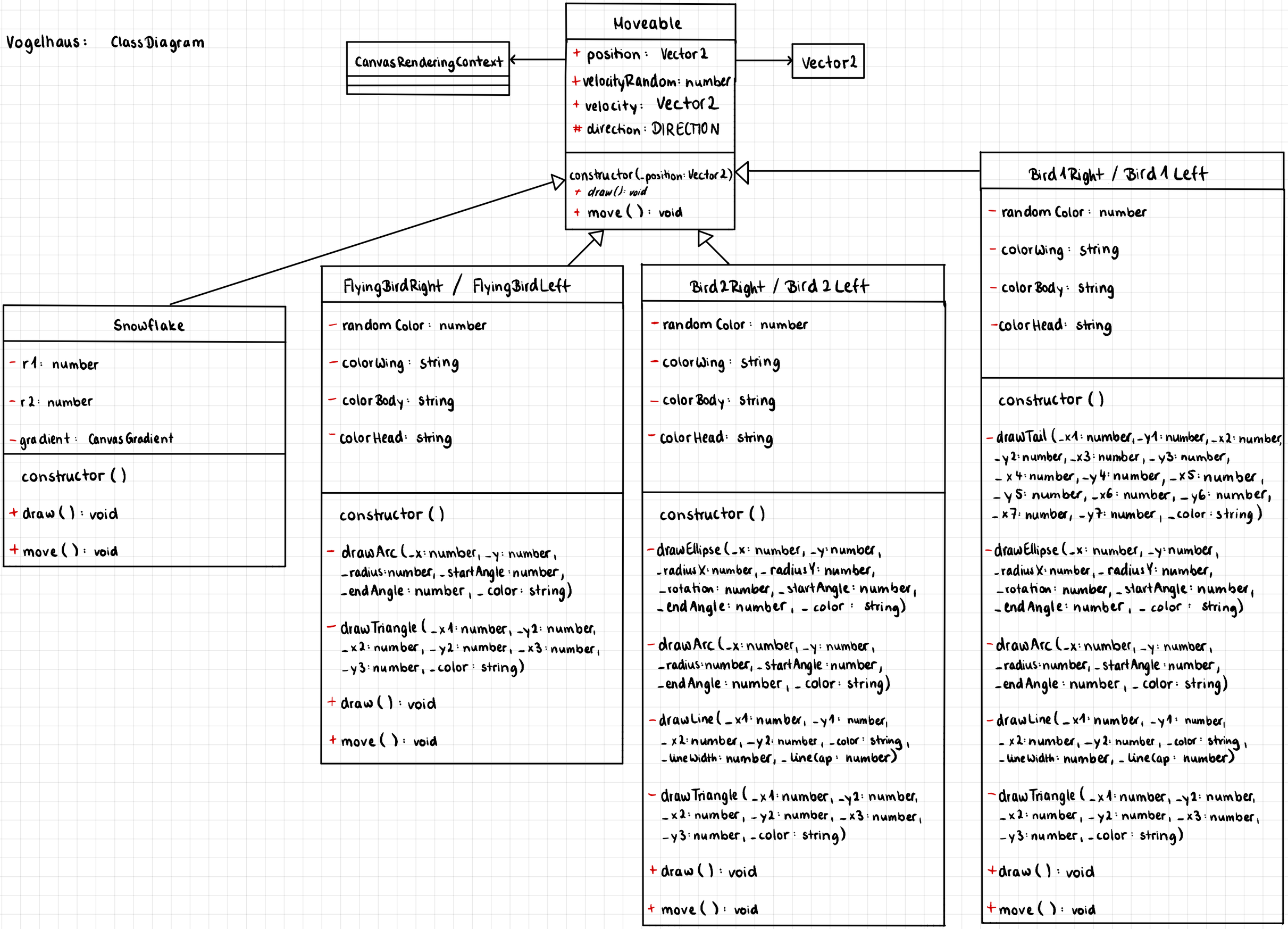
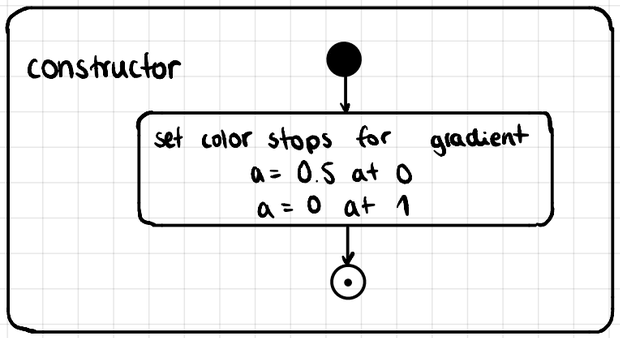
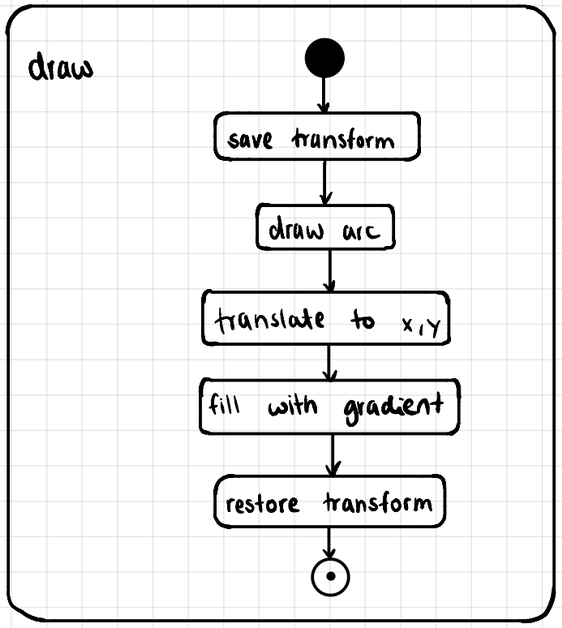
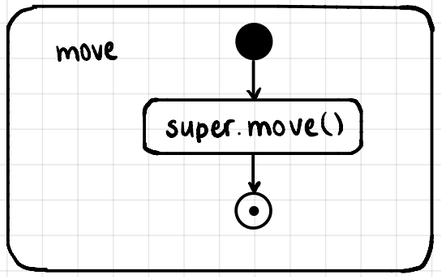
Vogelhaus: Class Diagram



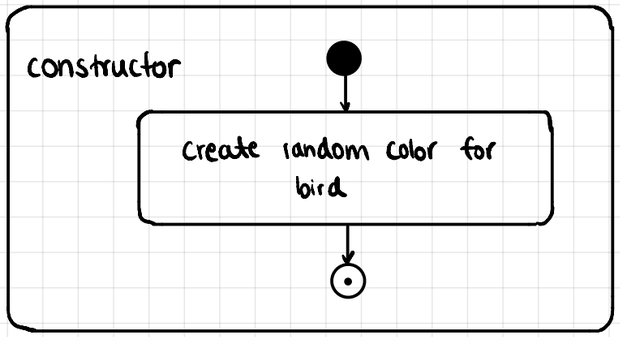
Vogelhaus: Activity Diagram - Snowflake

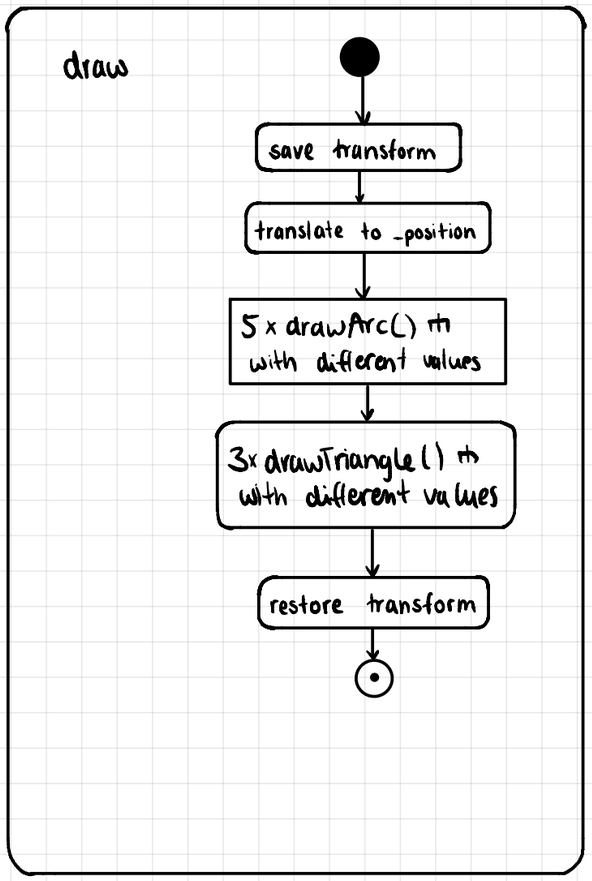


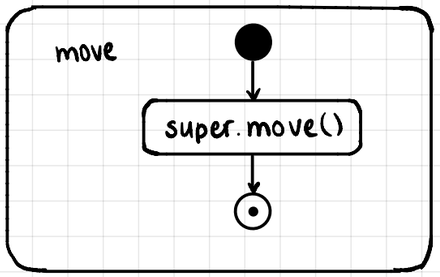




Vogelhaus: Activity Diagram - FlyingBirdRight

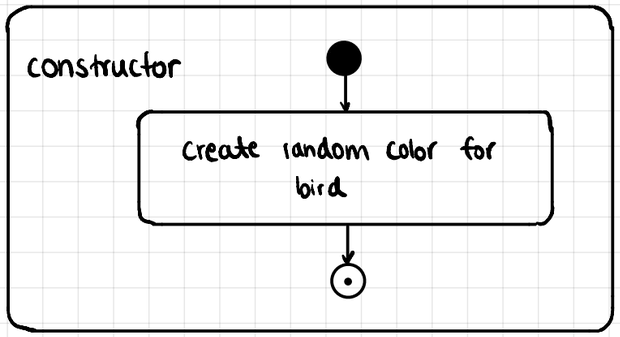


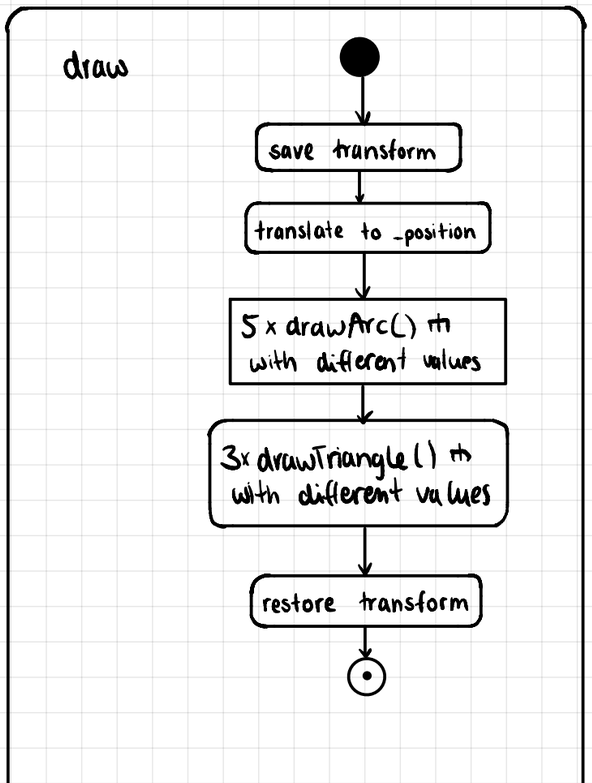


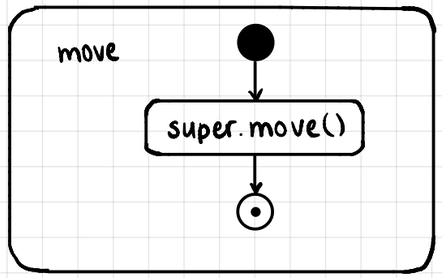


drawArc
drawTriangle
} siehe AD Main

Vogelhaus: Activity Diagram - FlyingBirdLeft

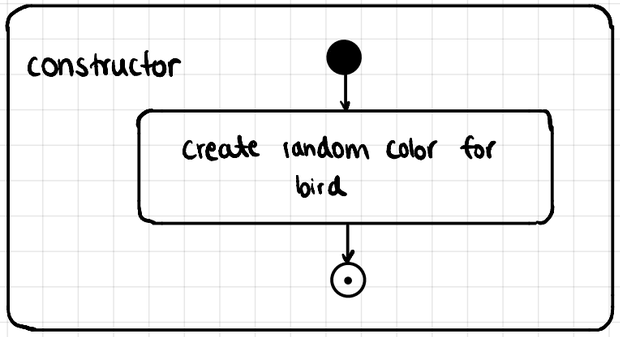


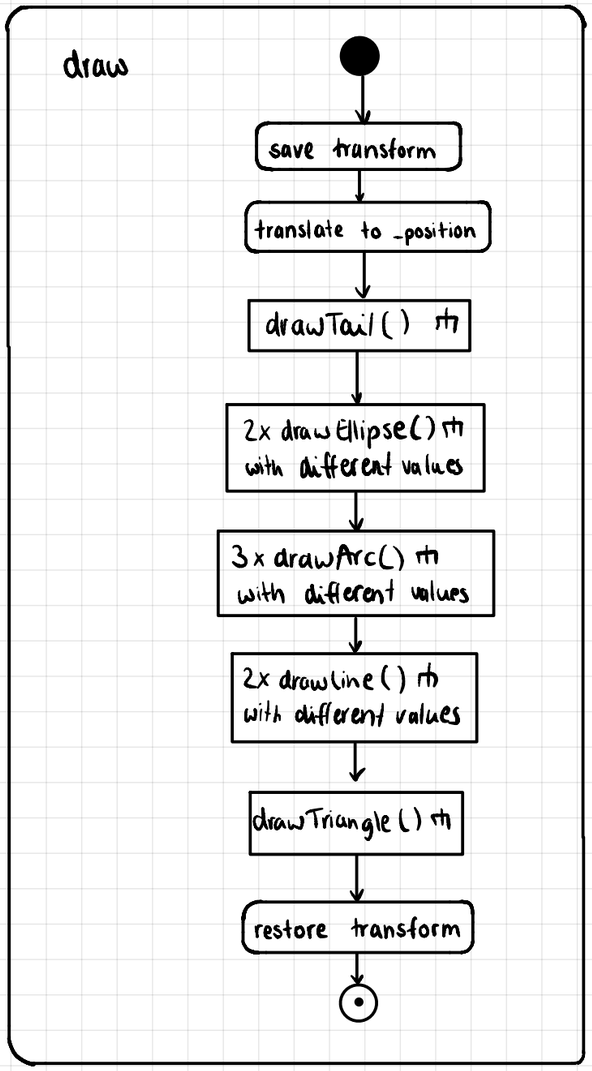


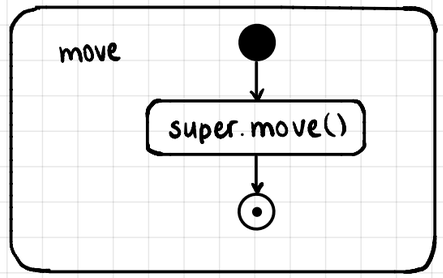


drawArc
drawTriangle
} siehe AD Main

Vogelhaus: Activity Diagram - Bird 1 Right



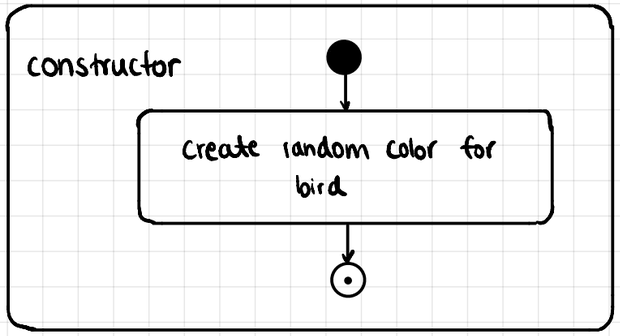


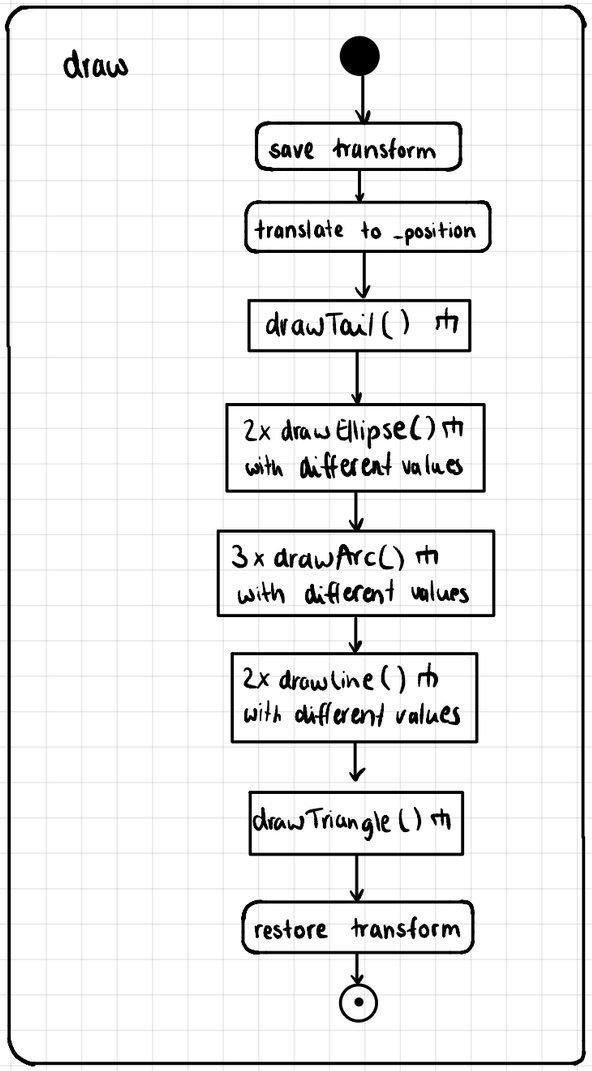


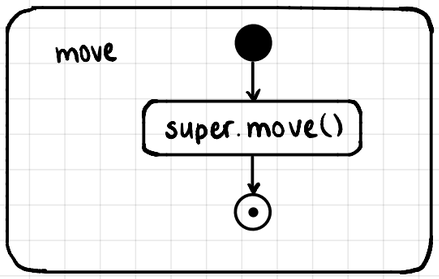
drawTail
drawEllipse
drawArc
drawLine
drawTriangle

} siehe AD Main

Vogelhaus: Activity Diagram - Bird 1 Left

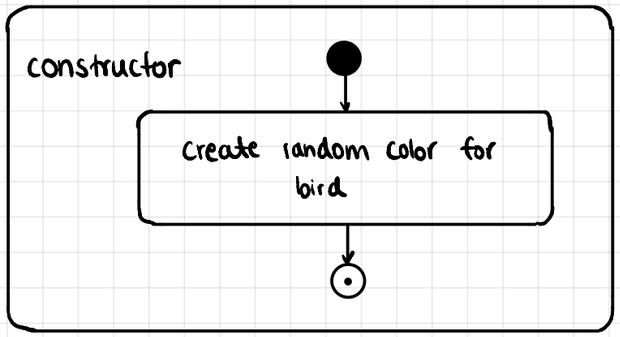


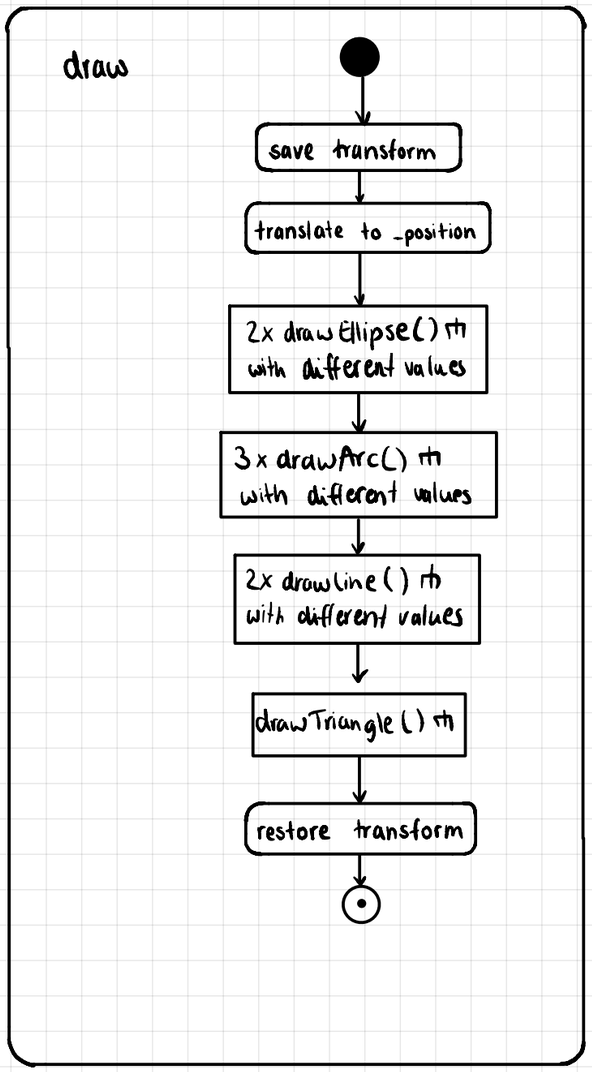


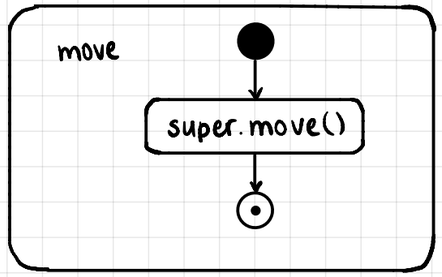


drawTail
drawEllipse
drawArc
drawLine
drawTriangle
} siehe AD Main

Vogelhaus: Activity Diagram - Bird2Right



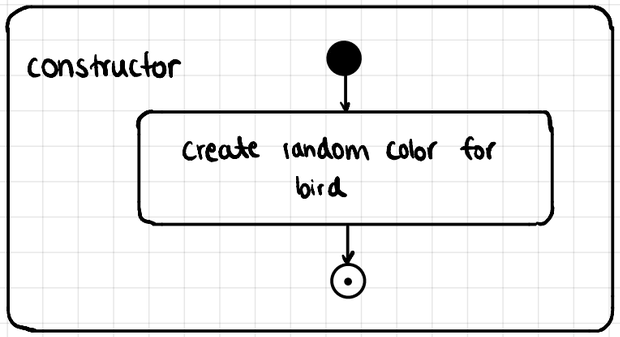


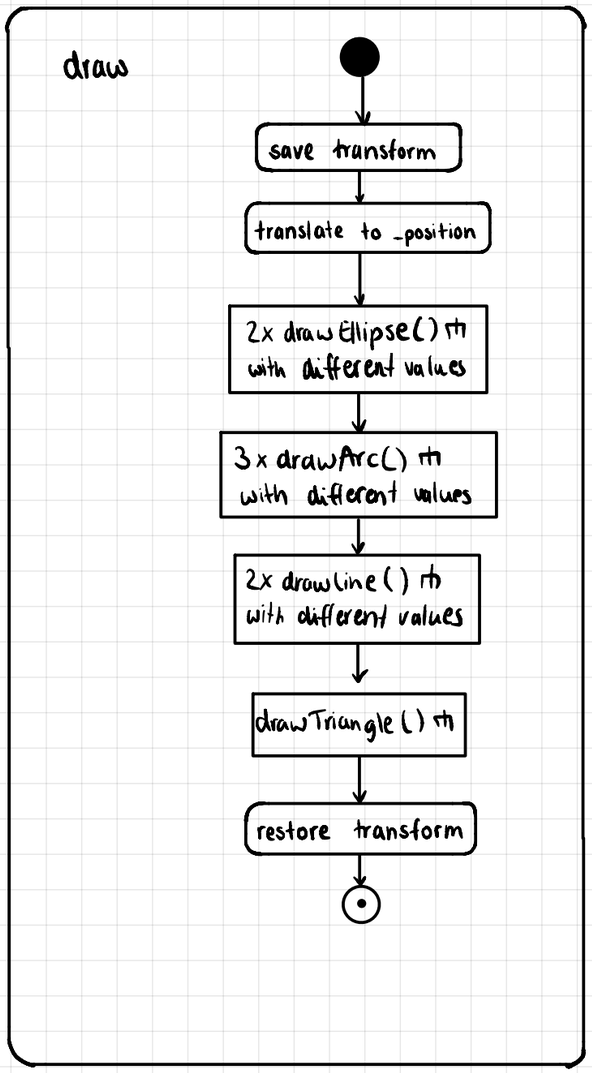


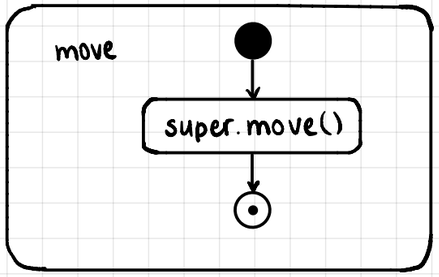
drawEllipse
drawArc
drawLine
drawTriangle

} siehe AD Main

Vogelhaus: Activity Diagram - Bird 2 Left



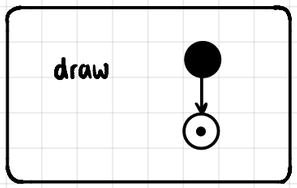


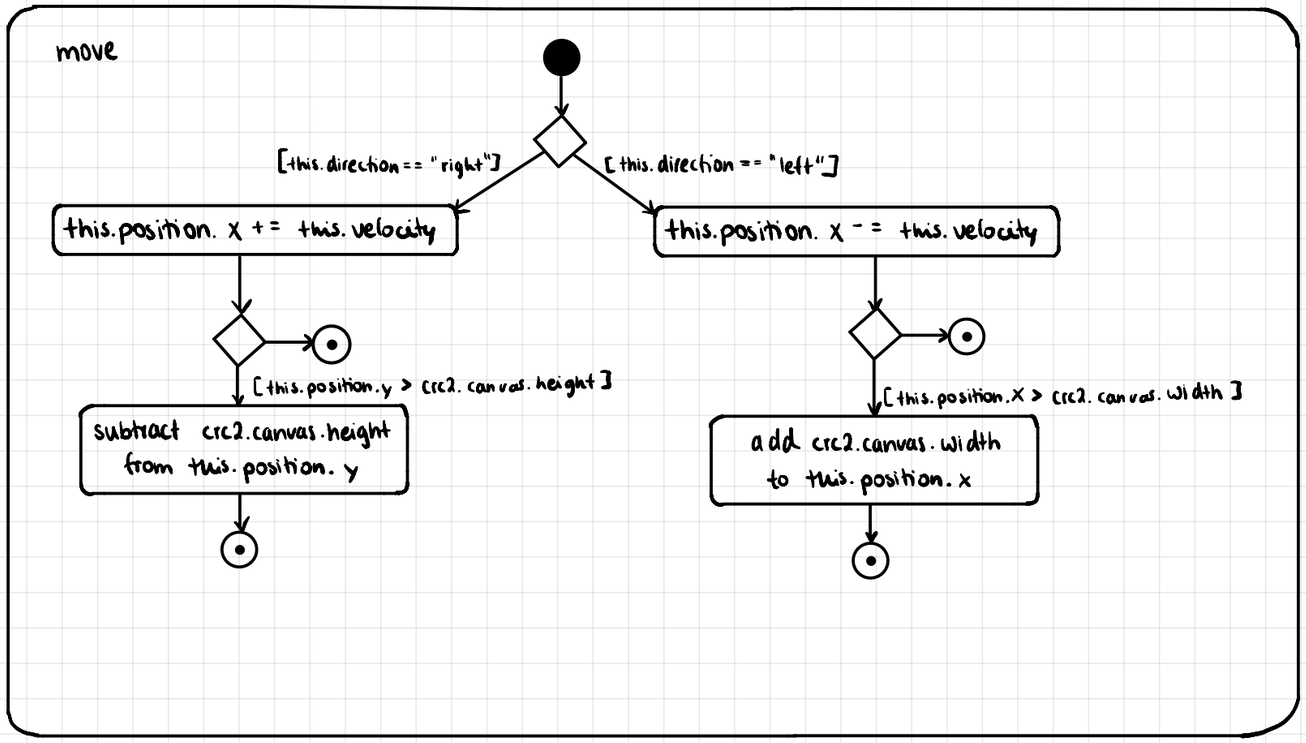


drawEllipse
drawArc
drawLine
drawTriangle

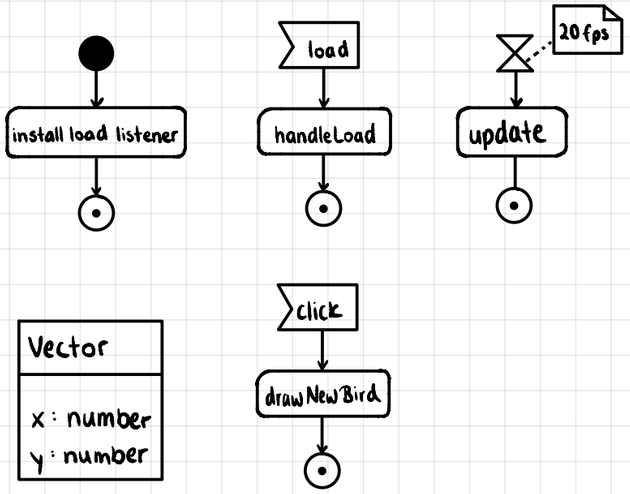
} siehe AD Main

Vogelhaus: Activity Diagram - Moveable



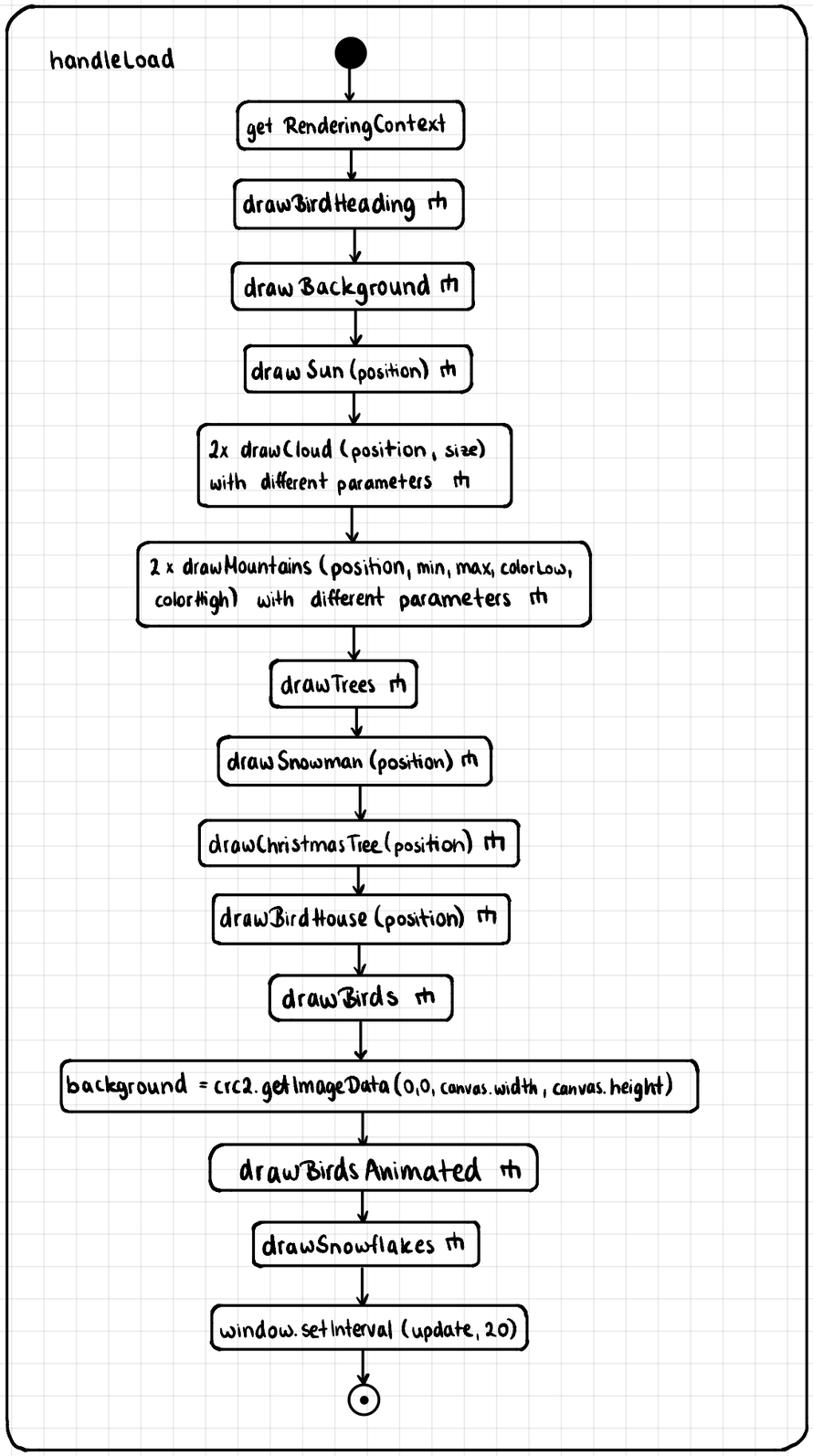


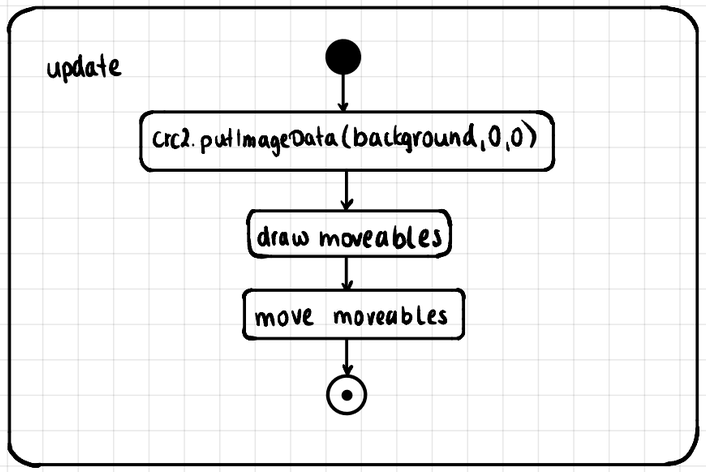
Vogelhaus: Activity Diagram - Main

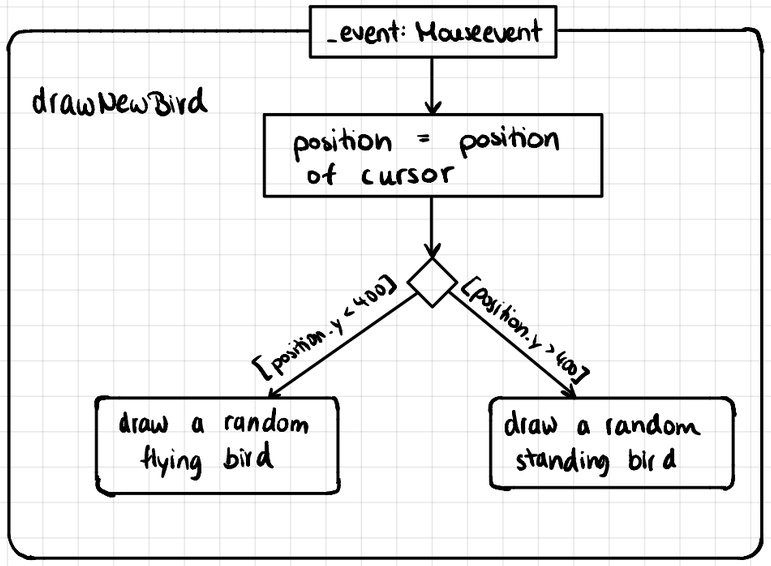


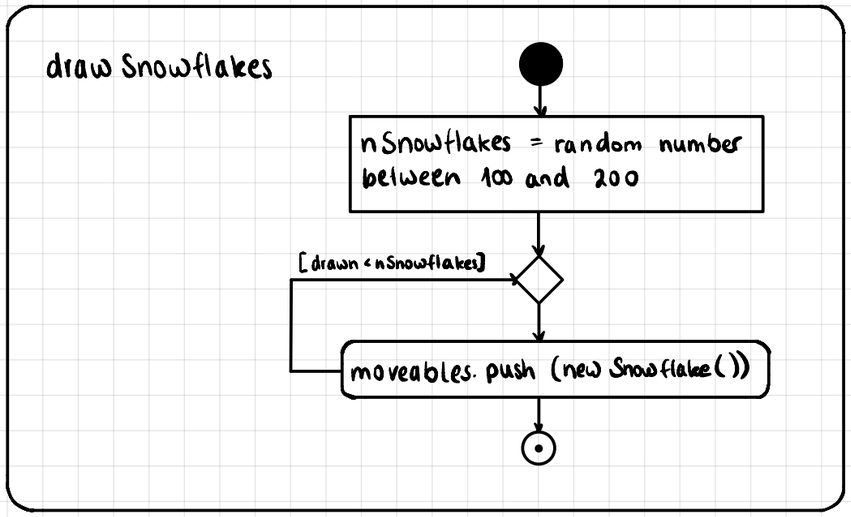
let background: ImageData;

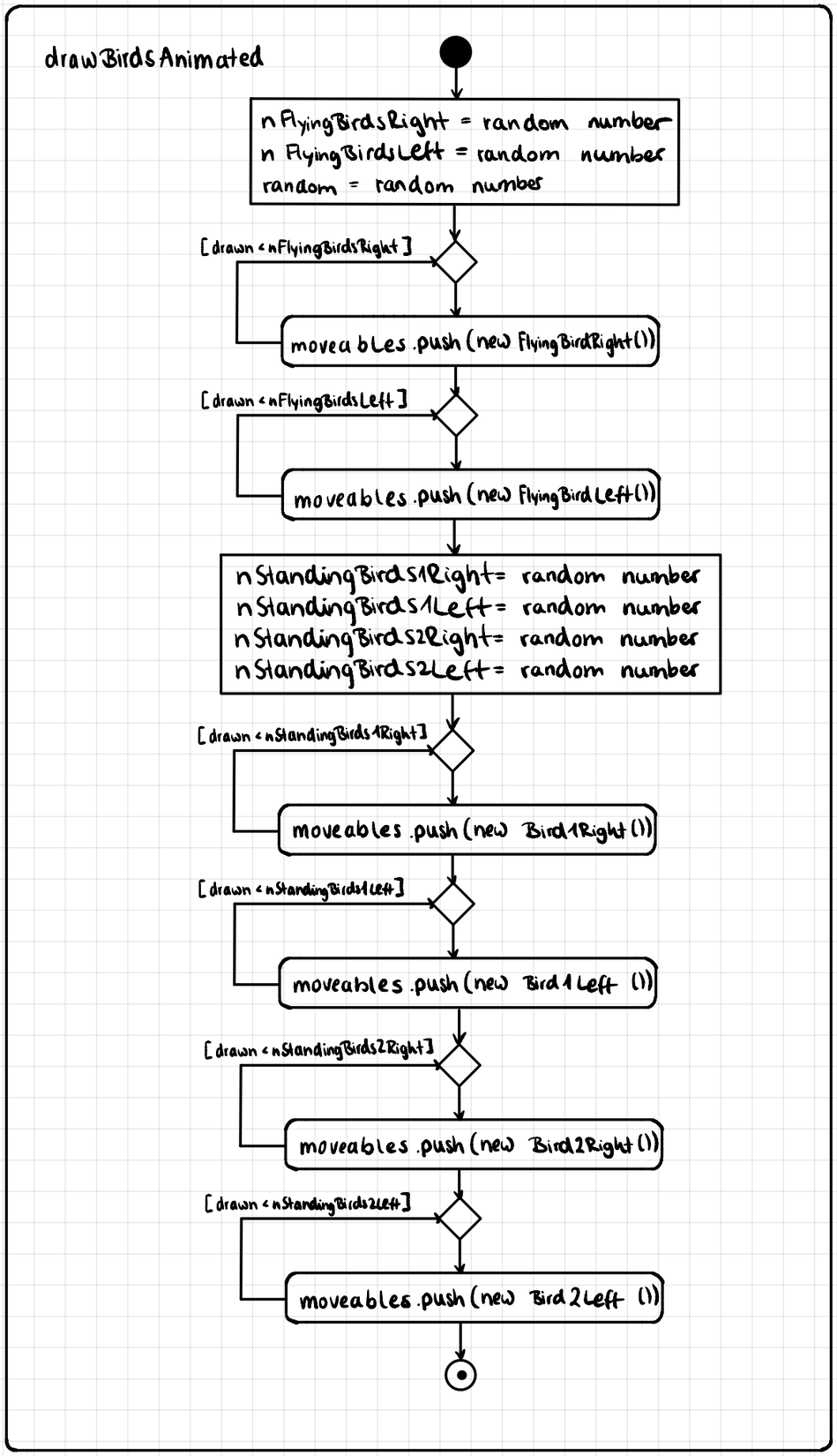
let moveables: Moveable[] = []








drawSnowflakes
nSnowflakes = random number between 100 and 200
[drawn < nSnowflakes]
moveables.push(new Snowflake())


drawBirdsAnimated
nFlyingBirdsRight = random number
nFlyingBirdsLeft = random number
random = random number
[drawn < nFlyingBirdsRight]
moveables.push(new FlyingBirdRight())
[drawn < nFlyingBirdsLeft]
moveables.push(new FlyingBirdLeft())
nStandingBirds1Right= random number
nStandingBirds1Left= random number
nStandingBirds2Right= random number
nStandingBirds2Left= random number
[drawn < nStandingBirds1Right]
moveables.push(new Bird1Right())
[drawn < nStandingBirds1Left]
moveables.push(new Bird1Left())
[drawn < nStandingBirds2Right]
moveables.push(new Bird2Right())
[drawn < nStandingBirds2Left]
moveables.push(new Bird2Left())

drawBackground

gradient = LinearGradient

set color stop for gradient
lightblue at 0
white at golden
HSL(196,40%,91%) at 1

drawSun

_position : Vector

r_1 : number = 30
r_2 : number = 100
gradient = RadialGradient

set color stops for gradient
HSLA(60, 100%, 90%, 1) at 0
HSLA(60, 100%, 50%, 0) at 1

save transform

translate to _position

draw full circle with r_2

restore transform

drawCloud

_position : Vector
_size : Vector

nParticles : number = 60
radiusParticle : number = 25
particle = path with full circle
with radiusParticle
gradient = RadialGradient

set color stops for gradient
(color = white)
a = 0.5 at 0
a = 0 at 1

save transform

translate to _position

restore transform

save transform

[drawn < nParticles]

x : number = (Math.random() - 0.5) * _size.x
y : number = -(Math.random() * _size.y

translate to x, y

draw particle

restore transform

drawMountains

_position: Vector
_min: number
_max: number
_colorLow: string
_colorHigh: string

stepMin: number = 10
stepMax: number = 50
x: number = 0

save transform

translate to _position

move to 0,0

line to 0,-max

[x < canvas.width]

x += random between stepMin and stepMax

y: number = -_min - random * (_max - _min)

line to x,y

line to x,0

closePath

create gradient with given color

drawPath

restore transform

drawTrees

_position: Vector

nTrees = random number between 5 and 15
let horizon = crc2.canvas.height * golden

[drawn <= nTrees]

save transform

x = Math.random * innerWidth
y = horizon

drawTreeSmall(position) rh

translate to position

restore transform

[drawn <= nTrees]

save transform

x = Math.random * innerWidth
y = horizon

drawTreeBig(position) rh

translate to position

restore transform

drawTreeSmall

_position: vector

save transform

drawRect() rh

3x drawTriangle() rh with different values

translate to position

restore transform

drawTreeBig

_position: vector

save transform

drawRect() rh

3x drawTriangle() rh with different values

translate to position

restore transform

drawSnowman

r: number[] = [50,35,20]

save transform

translate to _position

draw body:
3x Arc() rh
with different values for radius

draw hat:
drawRect() rh
2x drawLine() rh
with different values

draw eyes:
2x draw Arcs() rh
with different values for x

draw nose: draw rh Triangle()

draw mouth:
6x draw Arcs() rh
with different values

draw arcs:
3x draw Arcs() rh
with different values

draw arms:
6x drawLine() rh
with different values

restore transform

drawChristmasTree

_position: Vector

save transform

translate to _position

drawRect() rh

3x drawTriangle() rh
with different values

drawChristmasTreeBall() rh

random color for each ball

drawStar:
5x drawTriangle() rh
with different values

restore transform

drawBirdHouse

_position: Vector

save transform

translate to _position

4x drawLine() rh
with different values

drawRect() rh

3x drawPlane() rh
with different values

drawTriangle() rh

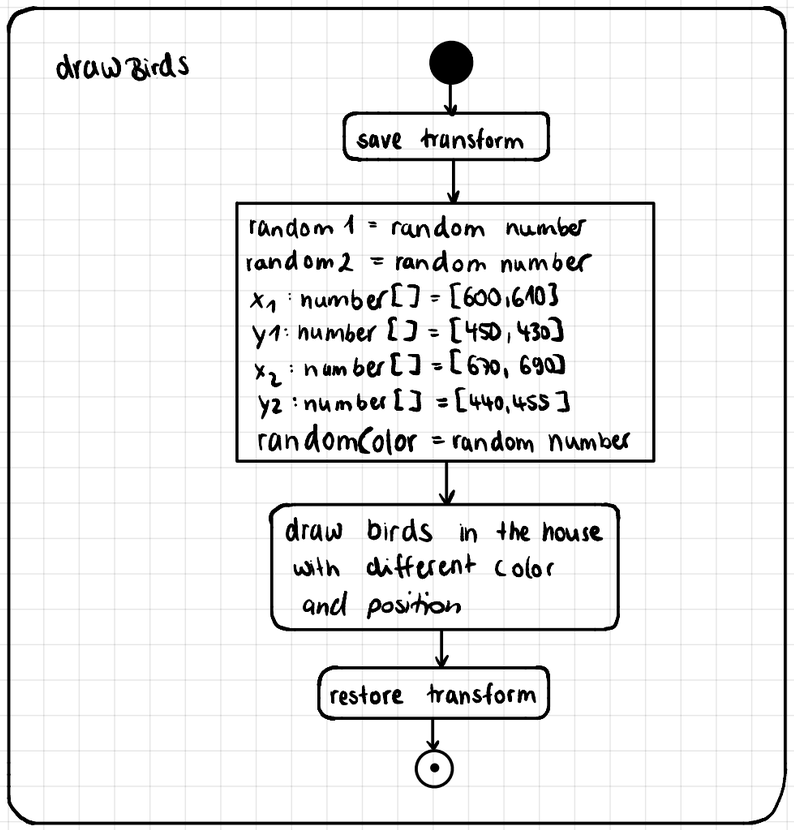
2x drawLine() rh
with different values

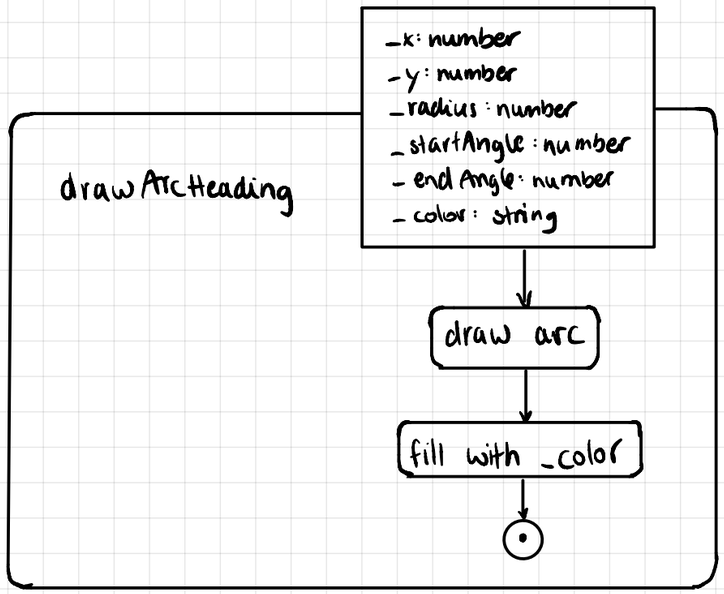
restore transform

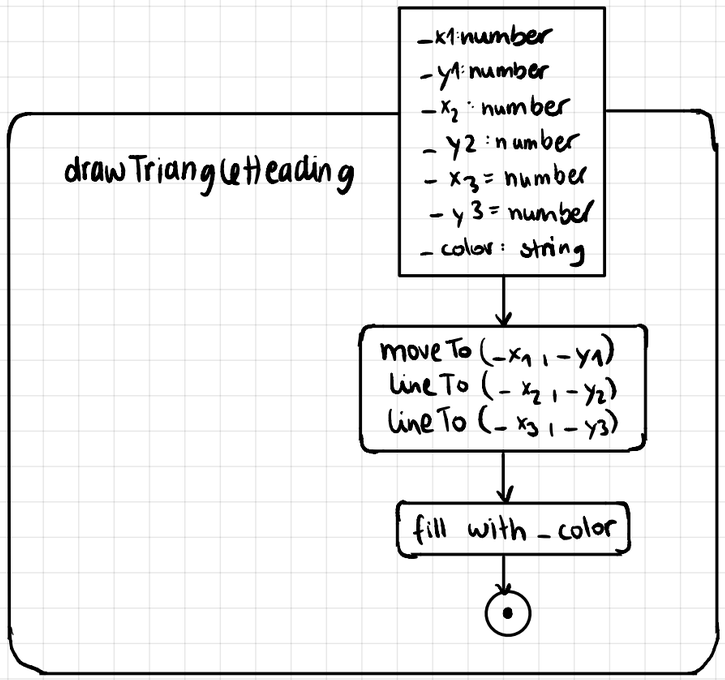
christmalTreeBall

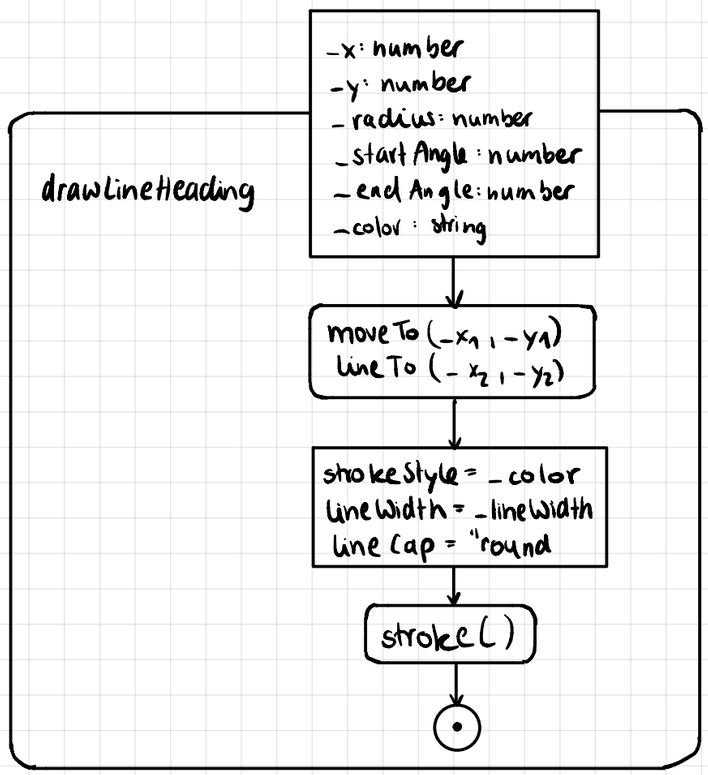
_x: number
_y: number
_radius: number
_startAngle: number
_endAngle: number

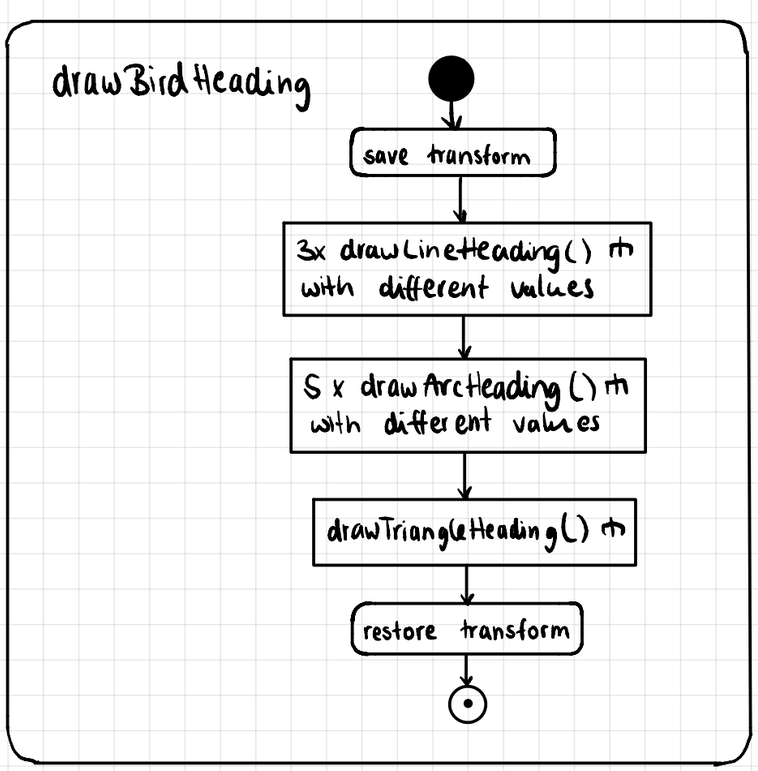
draw arc


drawBirds
save transform
random 1 = random number
random 2 = random number
x1 : number[] = [600,610]
y1: number [] = [450, 430]
x2 : number[] = [670, 690]
y2 : number[] = [440,455]
randomColor = random number
draw birds in the house with different color and position
restore transform


drawArcHeading
_x: number
_y: number
_radius : number
_startAngle : number
_endAngle: number
_color : string
draw arc
fill with _color


drawTriangleHeading
_x1:number
_y1:number
_x2 : number
_y2 : number
_x3 = number
_y3 = number
_color : string
moveTo (_x1 , _y1)
lineTo (_x2 , _y2)
lineTo (_x3 , _y3)
fill with _color


drawLineHeading
_x: number
_y: number
_radius: number
_startAngle : number
_endAngle: number
_color : string
moveTo (_x1 , _y1)
lineTo (_x2 , _y2)
strokeStyle = _color
lineWidth = _lineWidth
lineCap = "round
stroke()


drawBirdHeading
save transform
3x drawLineHeading() with different values
5 x drawArcHeading() with different values
drawTriangleHeading()
restore transform

drawArc

_x: number
_y : number
_radius: numbes
_startAngle : numbes
_color: number

draw arc

fill with _color

drawRect

_x: number
_y : number
_radius: numbes
_startAngle : numbes
_color: number

draw rect

fill with _color

drawEllipse

_x: number
_y : number
_radiusX: number
_radiusY: numbes
_startAngle : numbes
_endAngle : numbes
_color: number

draw ellipse

fill with _color

drawPlane

_x1: numbes
_y1: number
_x2 : number
_y2: number
_color: string
_lineWidth: number
_lineCap: number

moveTo(_x1, _y1)
LineTo(_x2, _y2)

StrokeStyle = _color
Line Width = _lineWidth
LineCap = "butt" or "round

stroke( )

drawPlane

_x1: number
_y2: number
_x2 : number
_y2 : number
_x3: number
_y3: number
_x4: number
_y4: number
_color: string

moveTo(_x1, _y1)
LineTo (_x2, _y2)
LineTo (_x3, _y3)
LineTo (_x4, _y4)

fill with _color

drawTriangle

_x1: number
_y1: number
_x_2 : number
_y2 : number
_x_3 = number
_y3 = number
_color: string

moveTo(_x_1, _y_1)
LineTo (_x_2, _y_2)
LineTo (_x_3, _y_3)

fill with _color