

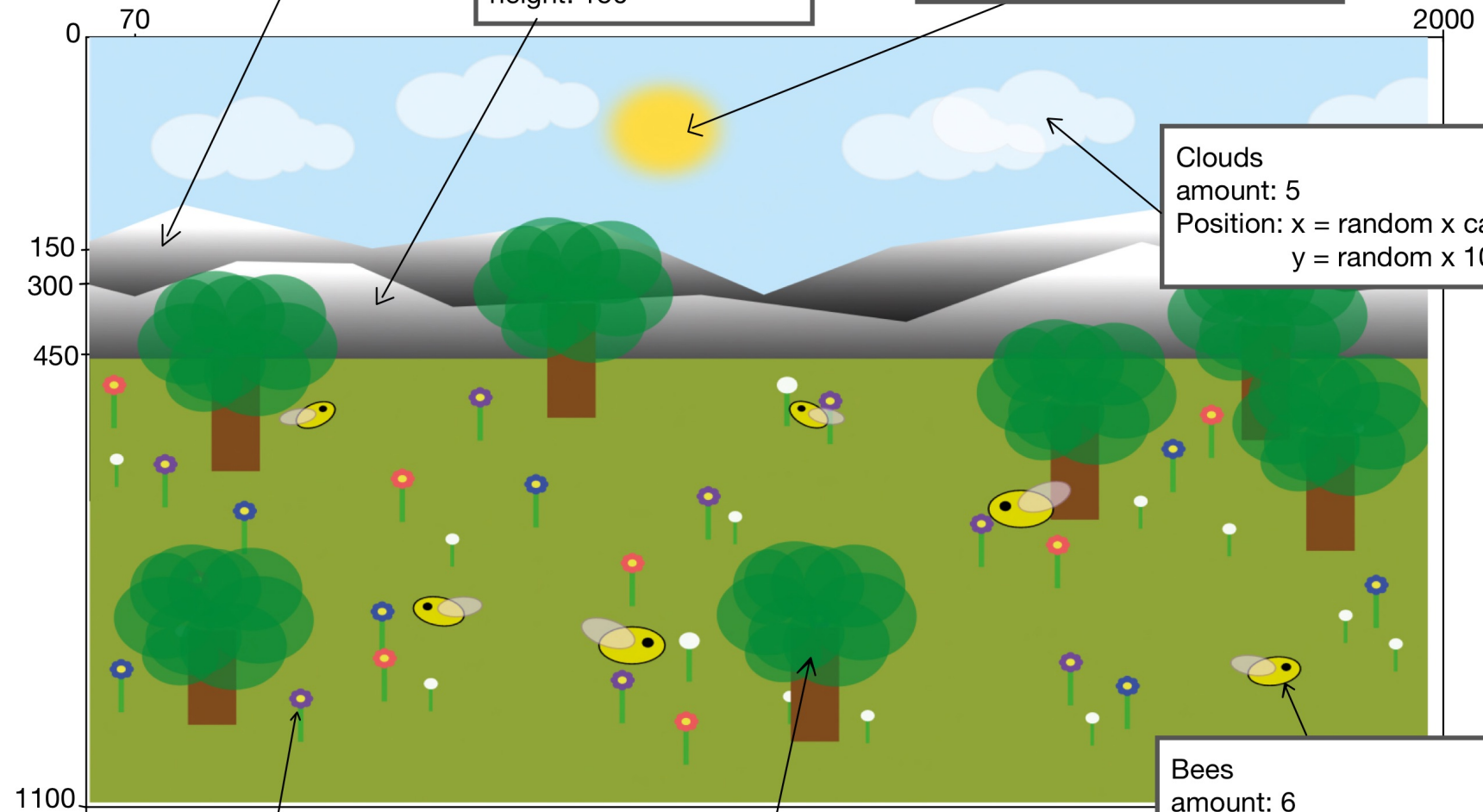
Canvas:
1100 x 2000px

mountain
Position: $x = 0$; $y = 150$
height: 300

mountain2
Position: $x = 0$; $y = 300$
height: 150

DayNightCycle
Position: $x = 70$; $y = \text{Math.PI}$
cycle = timeScale

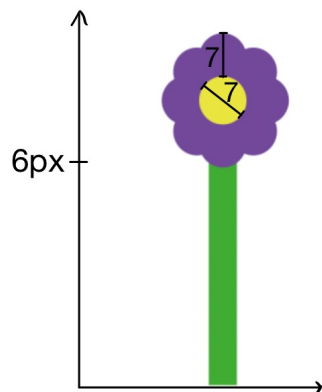
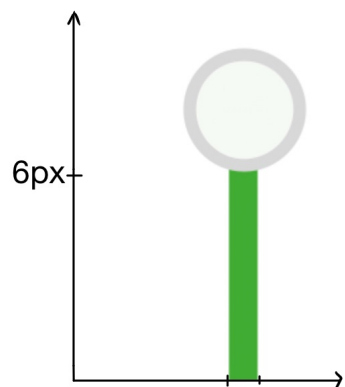
Clouds
amount: 5
Position: $x = \text{random} \times \text{canvas.width}$
 $y = \text{random} \times 100 + 50$



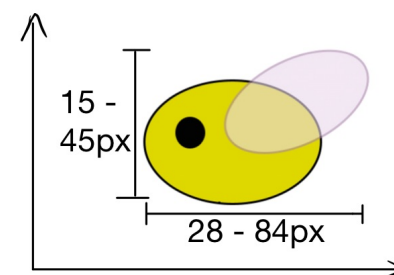
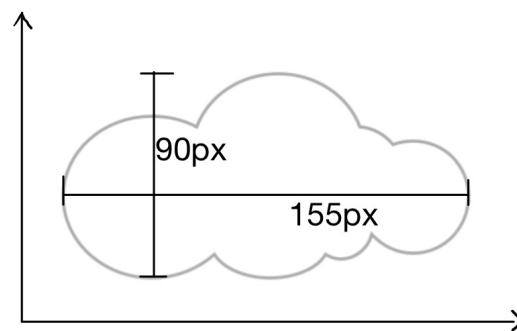
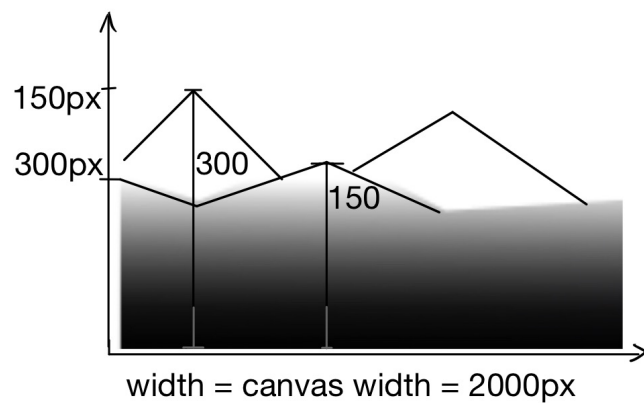
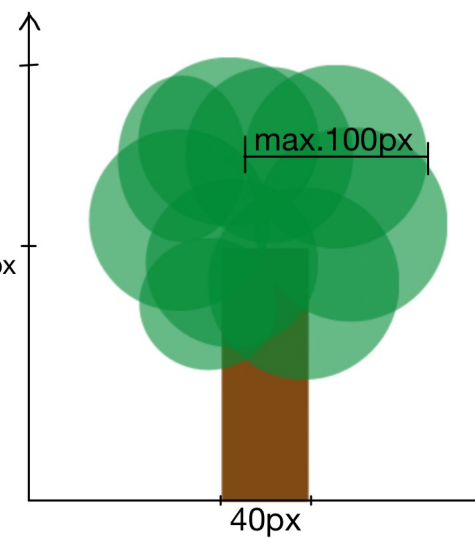
Flowers
amount: 200
Position: $x = \text{random} \times \text{canvas.width}$
 $y = \text{random} \times 1050 + 460$

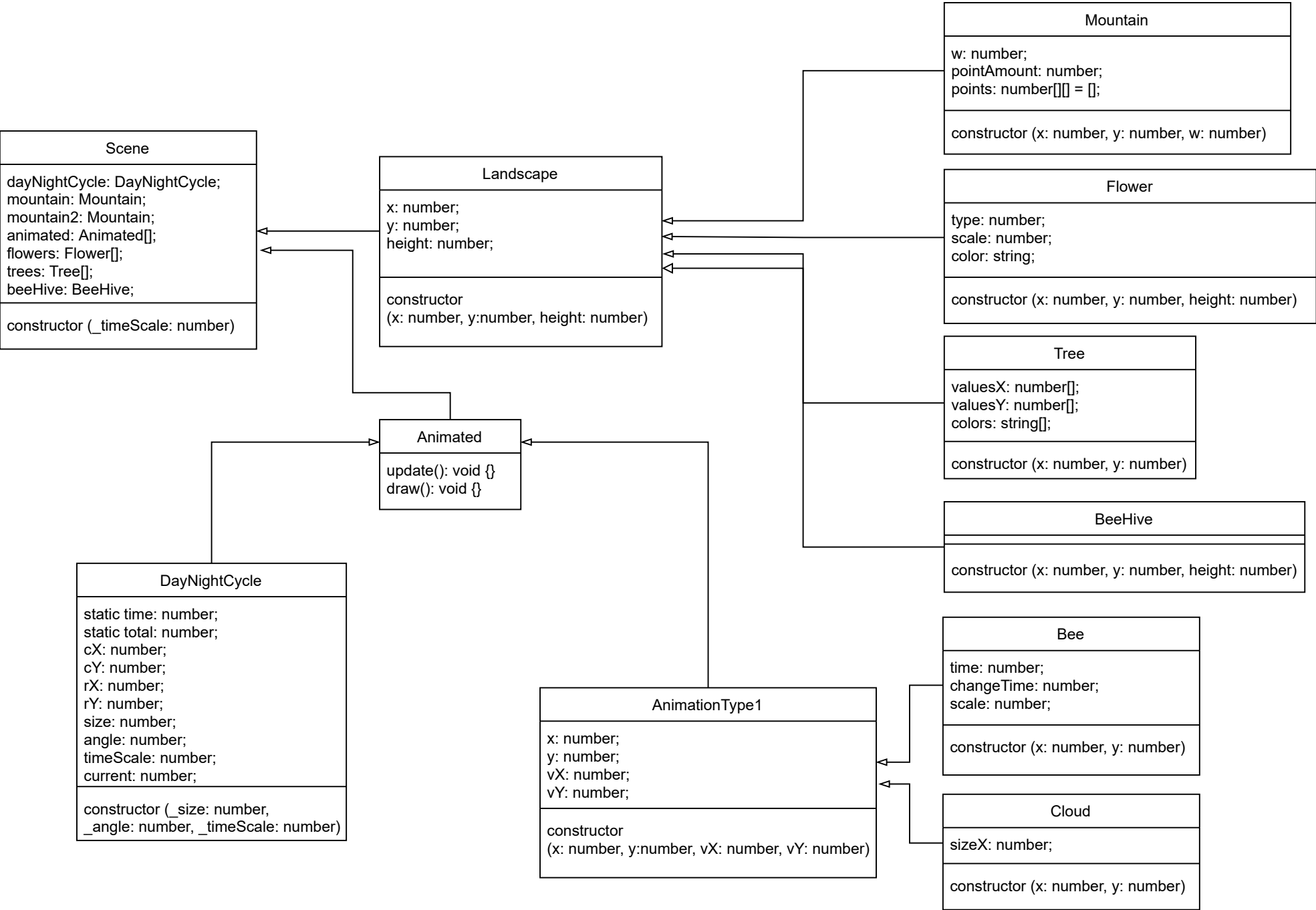
Trees
amount: 12
Position: $x = \text{random} \times \text{canvas.width}$
 $y = \text{random} \times 1050 + 450$

Bees
amount: 6
Position: $x = \text{random} \times \text{canvas.width}$
 $y = \text{random} \times 800 + 300$

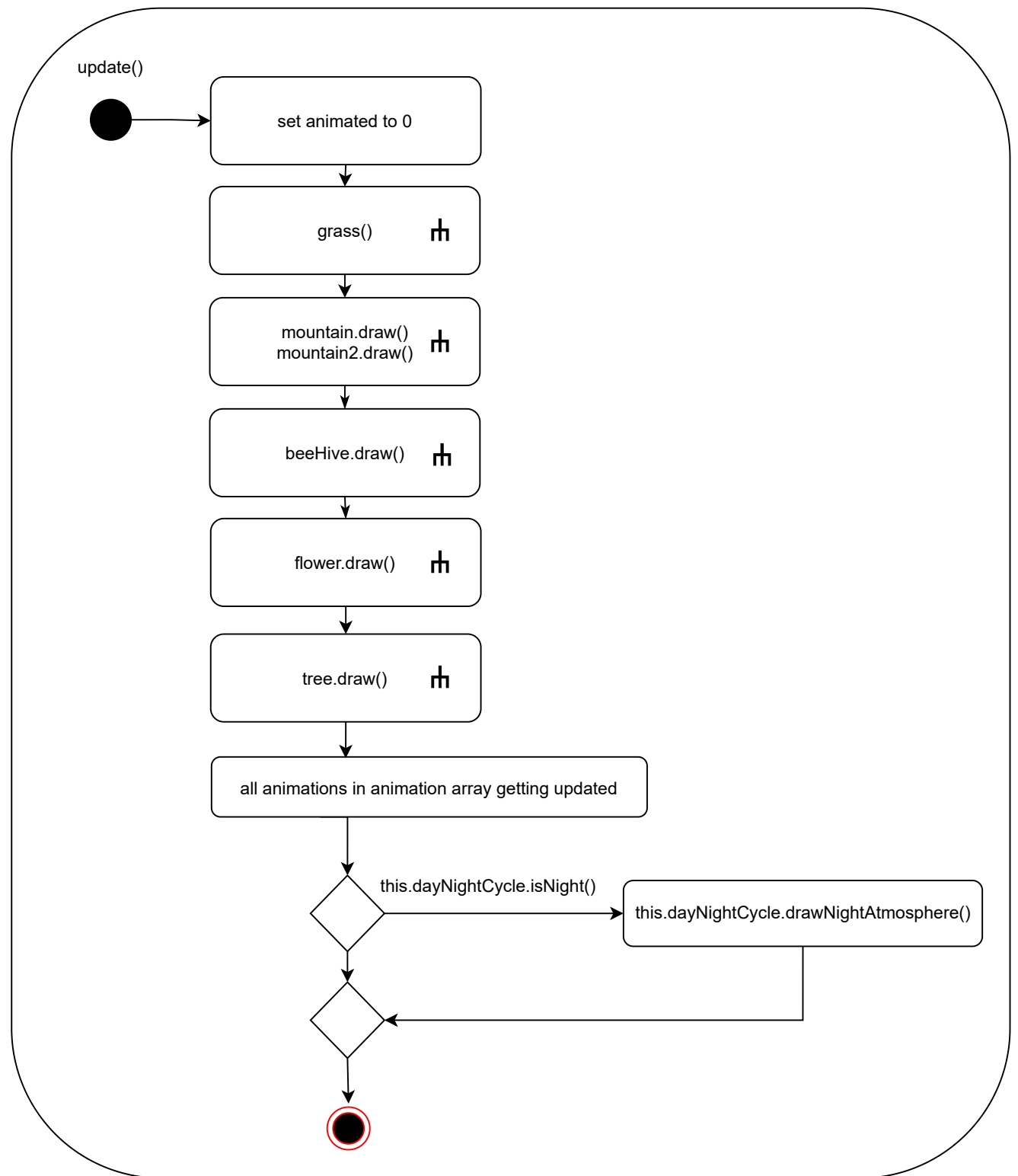
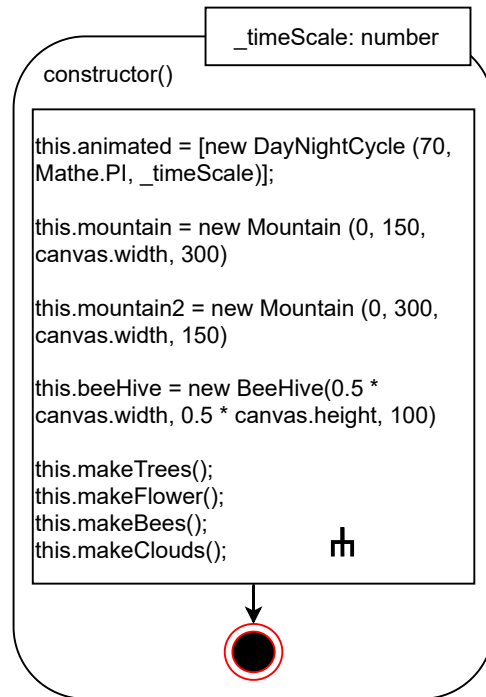


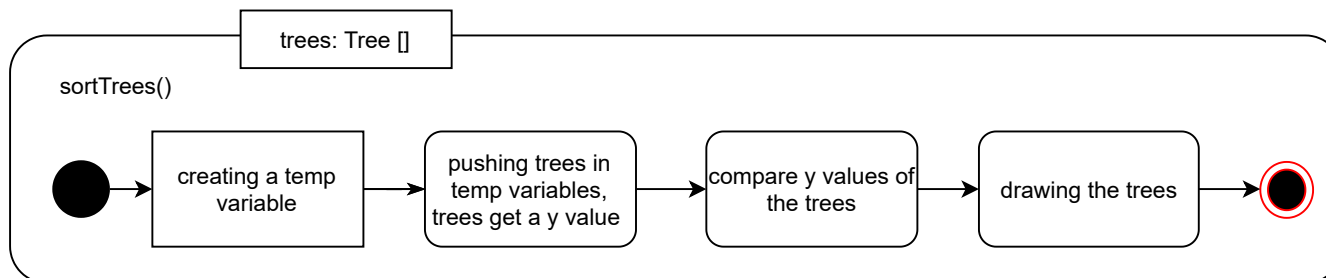
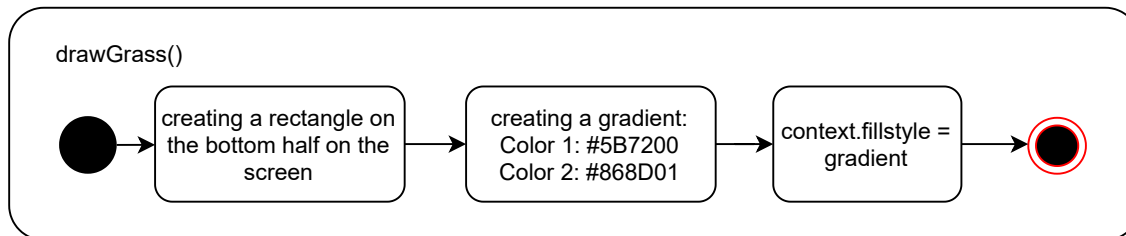
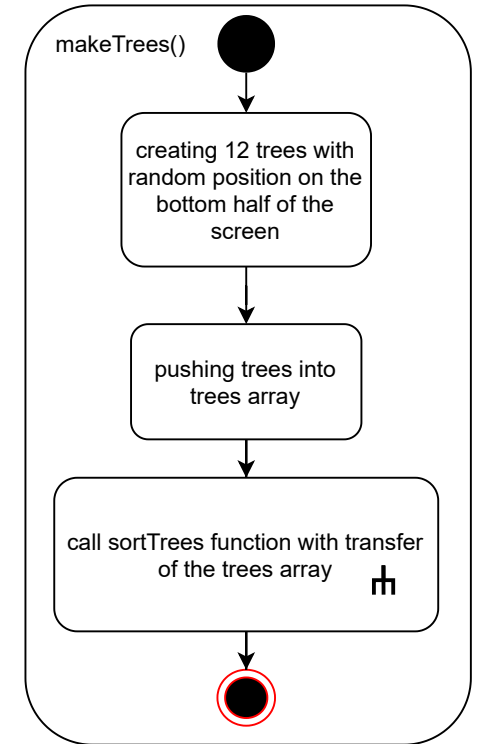
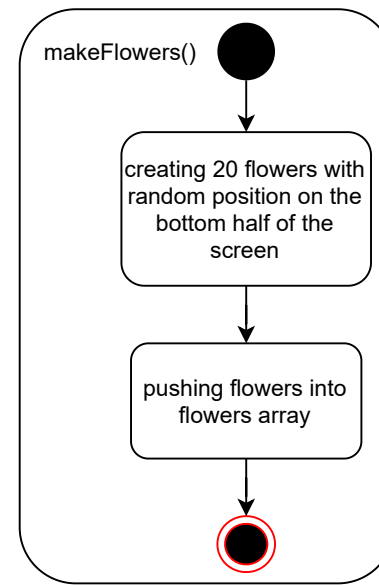
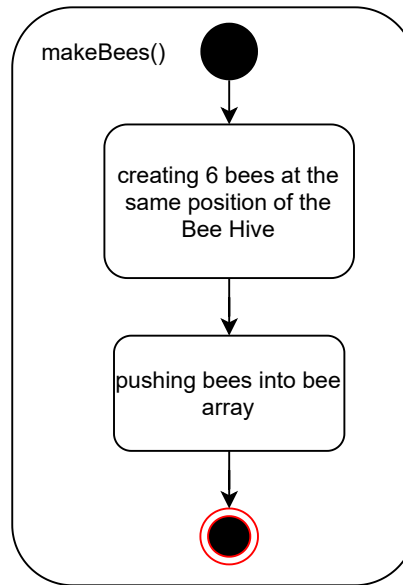
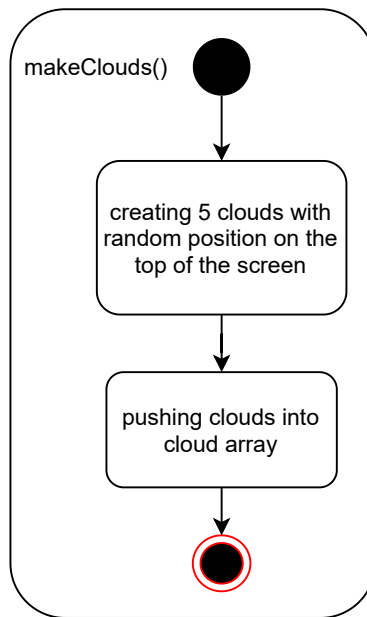
height trunk:
-150px - random x 100px

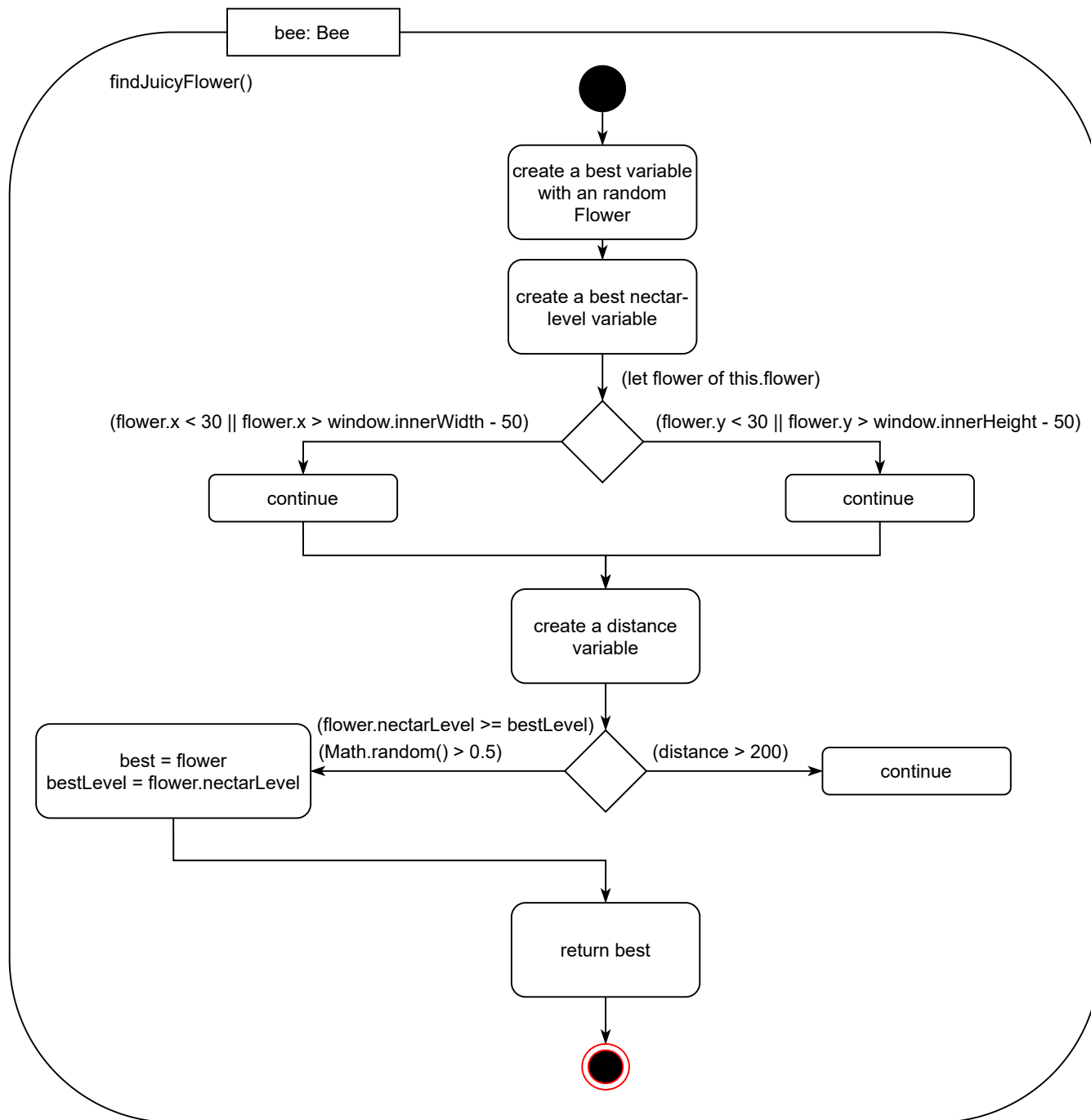




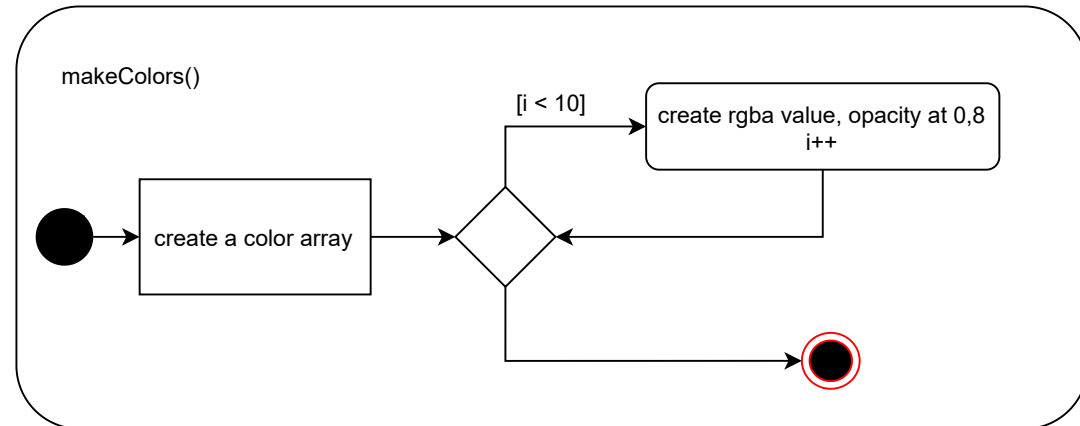
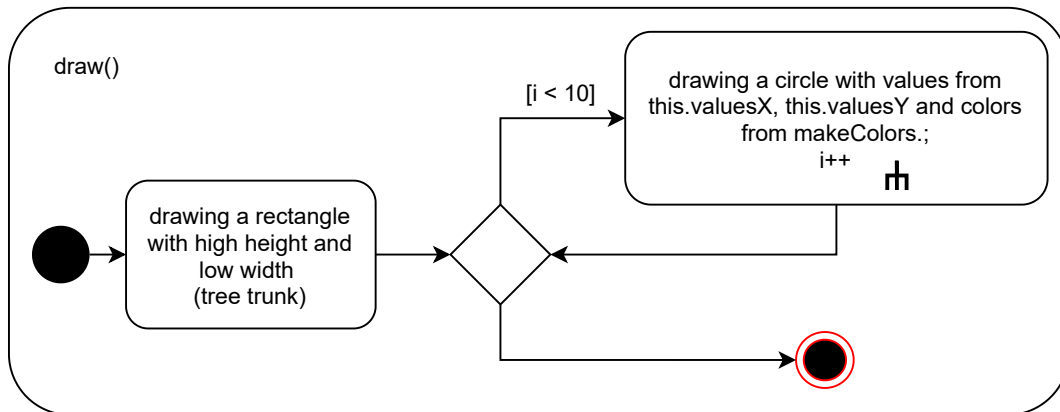
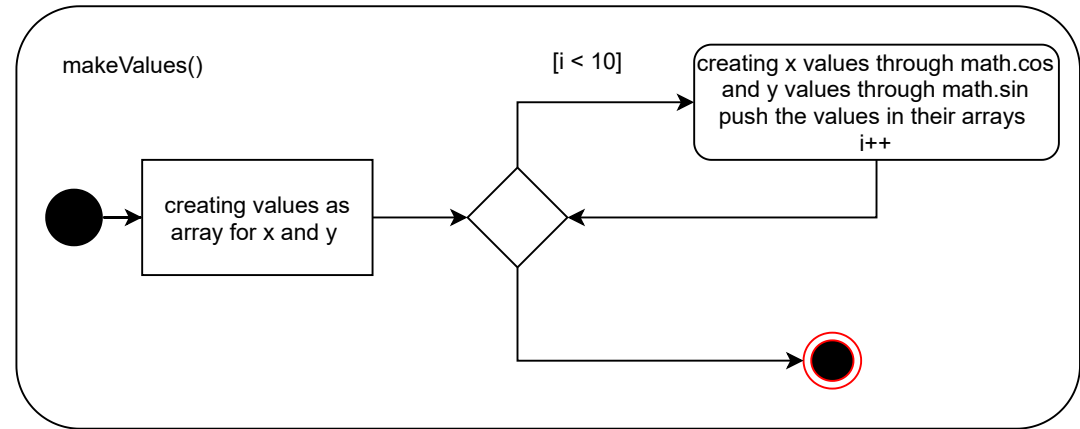
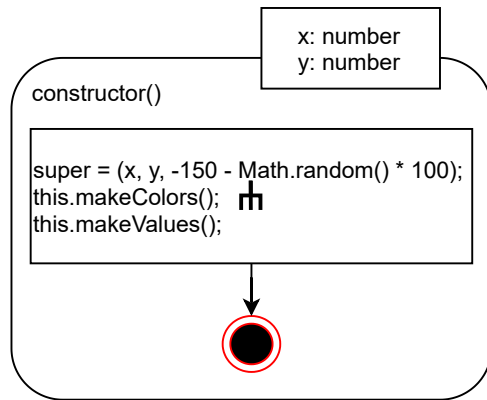
Aktivitätsdiagramm: Scene







Aktivitätsdiagramm: Trees



Aktivitätsdiagramm: DayNightCycle

size: number
angle: number
timeScale: number

constructor()

```
super();  
this.timeScale = _timeScale;  
DayNightCycle.time = 0;  
this.current = 0;  
DayNightCycle.total = 2 * Math.PI  
/this.timeScale;  
this.cX = this.canvas.width / 2;  
this.cY = this.canvas.height / 3;  
this.rX = this.canvas.width * 0,4;  
this.rY = this.canvas.height * 0.28;  
this.size = _size;  
this.angle = _angle,  
this.timeScale = _timeScale;
```



colorFade()

creating a red

creating a green

creating a blue

return rgb

rS: number
gS: number
bS: number
rE: number
gE: number
bE: number
i: number
steps: number



sun()

defining x and y for the starting point,
used later to distinguish the position of
the sun

creating a radialGradient
should be brighter on the outside
to look like a sun

colors of the gradient:
#FC9601
#FFCC33
rgba(255,204,51,0)

filling the gradient so it looks like a
circle,
because of the opacity there
should be no sign of the rectangle



moon()

define x and y + Math.PI.
use Math.PI to have the moon fly
opposite to the sun

create radialGradient thats
white on the in- and outside


colors for the gradient:
rgba (255, 255, 255, 1)
rgba(255, 255, 255, 1)
rgba (255, 255, 255, 0)

filling the radiantGradient so it looks
like a circle,
because of the rgba opacity there
should be no sign of a rectangle



update()

this.angle += this.timeScale

draw() 


increase time by 1


this.current = this.time % (this.total)



draw()

sky()

sun() 

moon() 



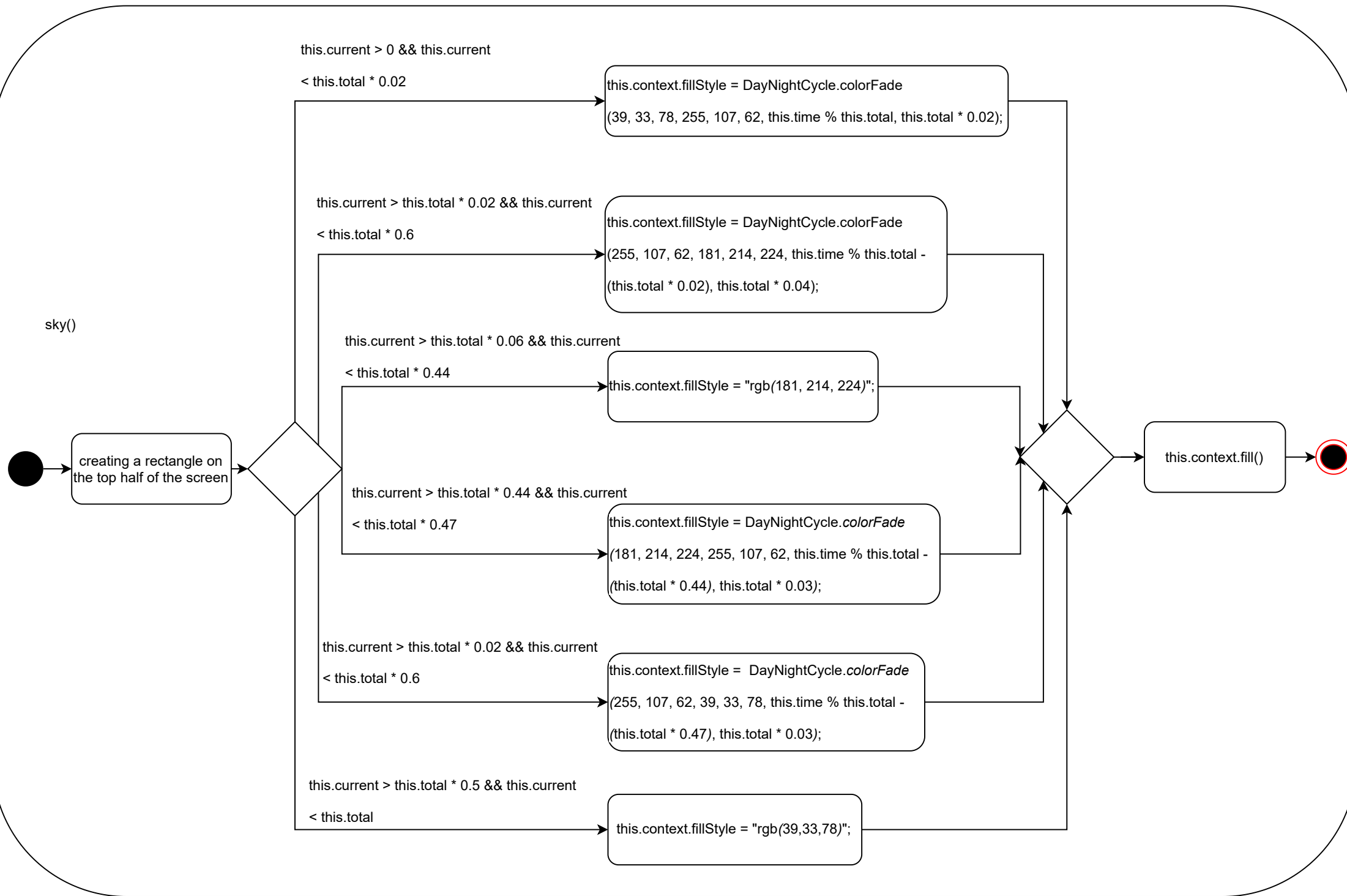
isNight()

[this.time % (this.total)
> (this.total) / 2.1]

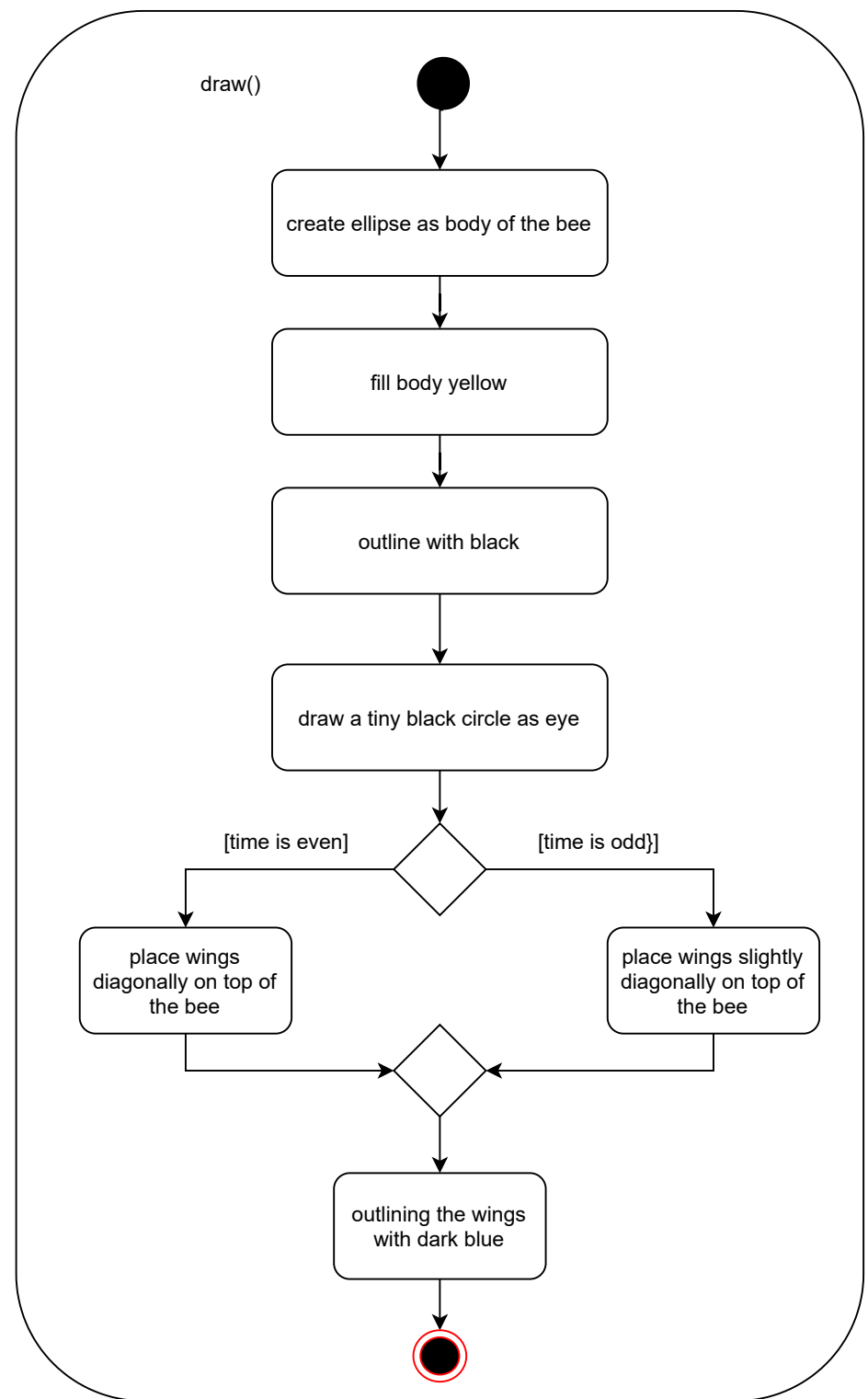
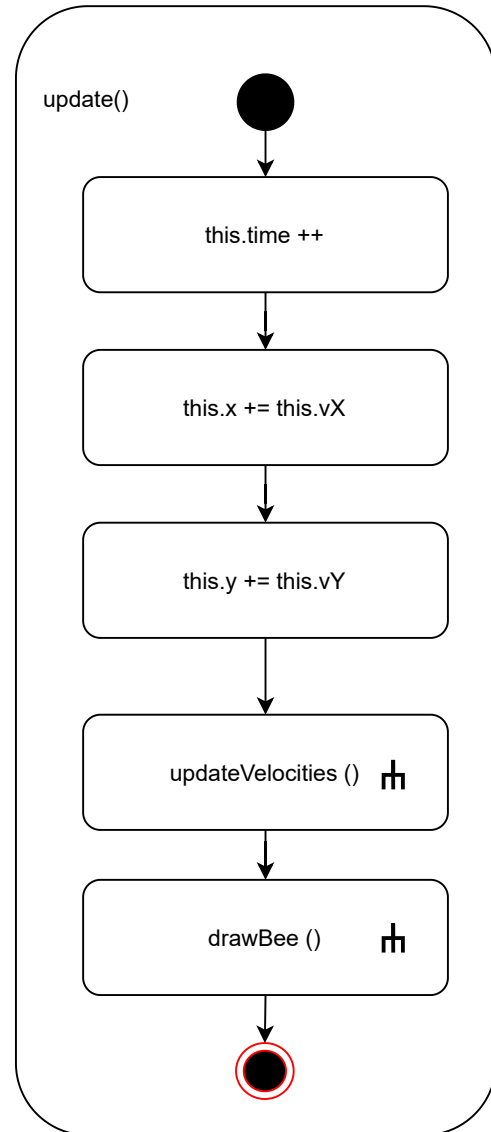
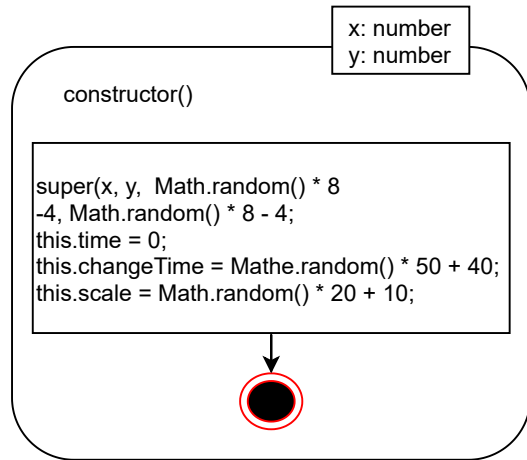
true

false

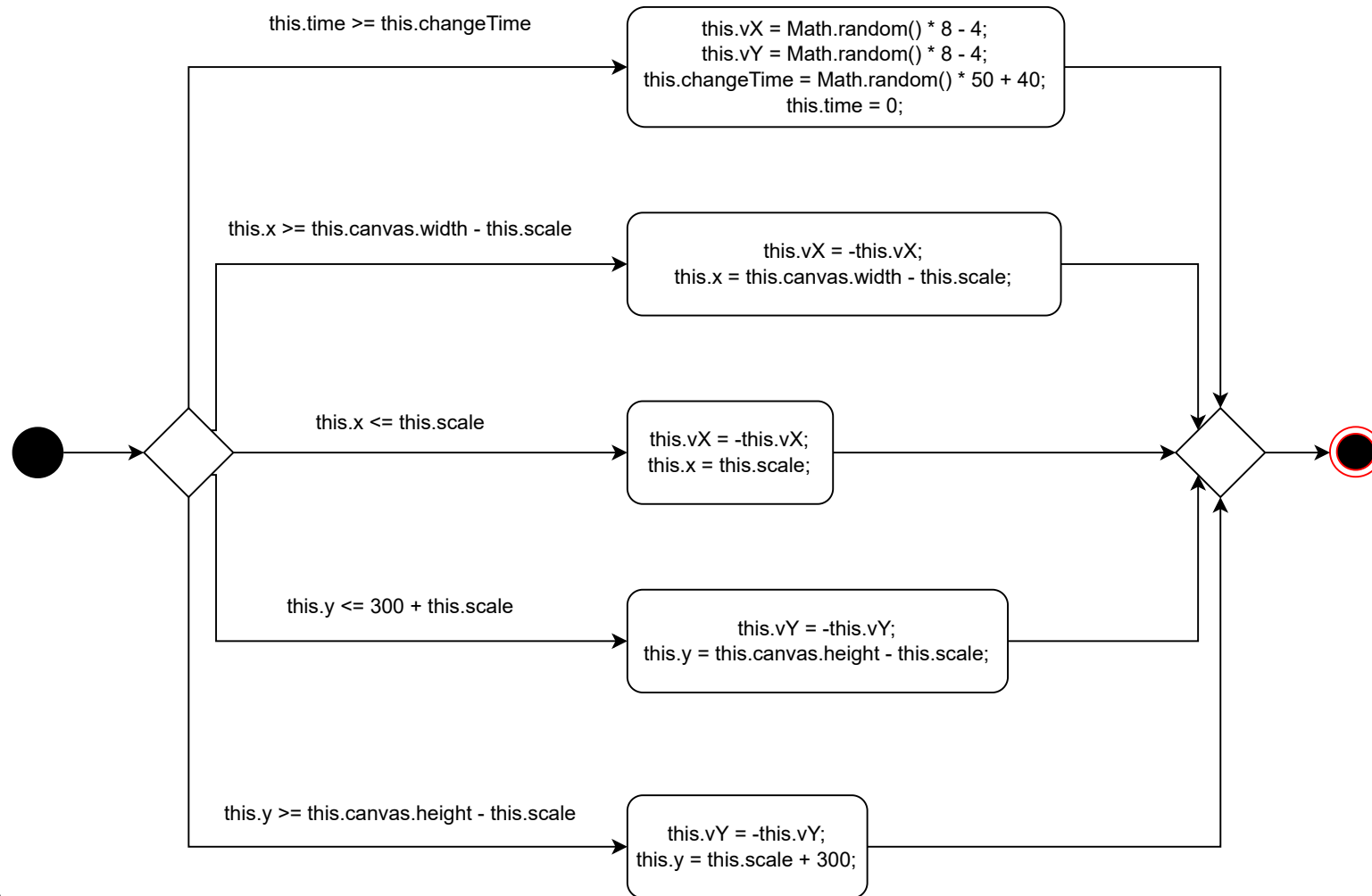




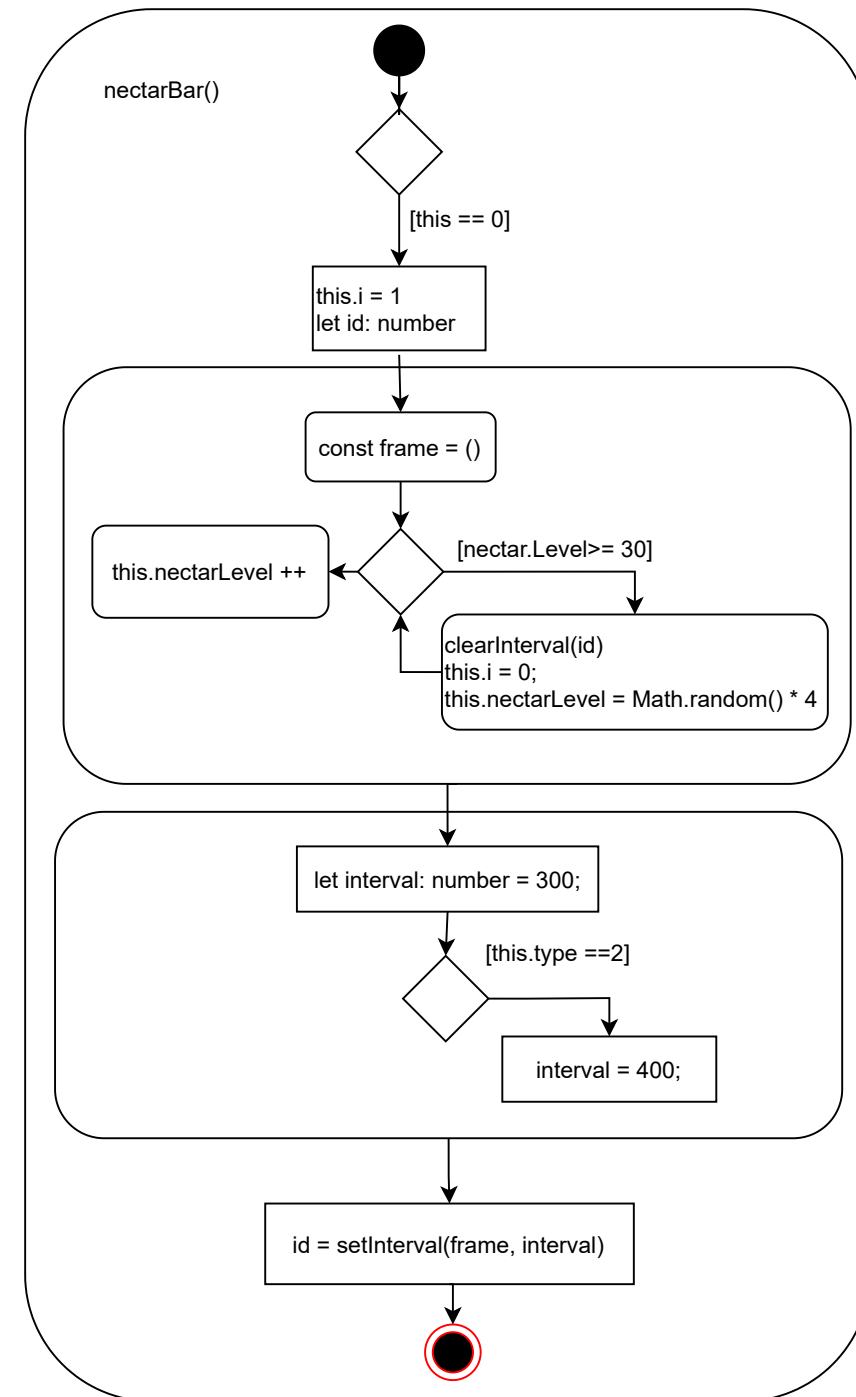
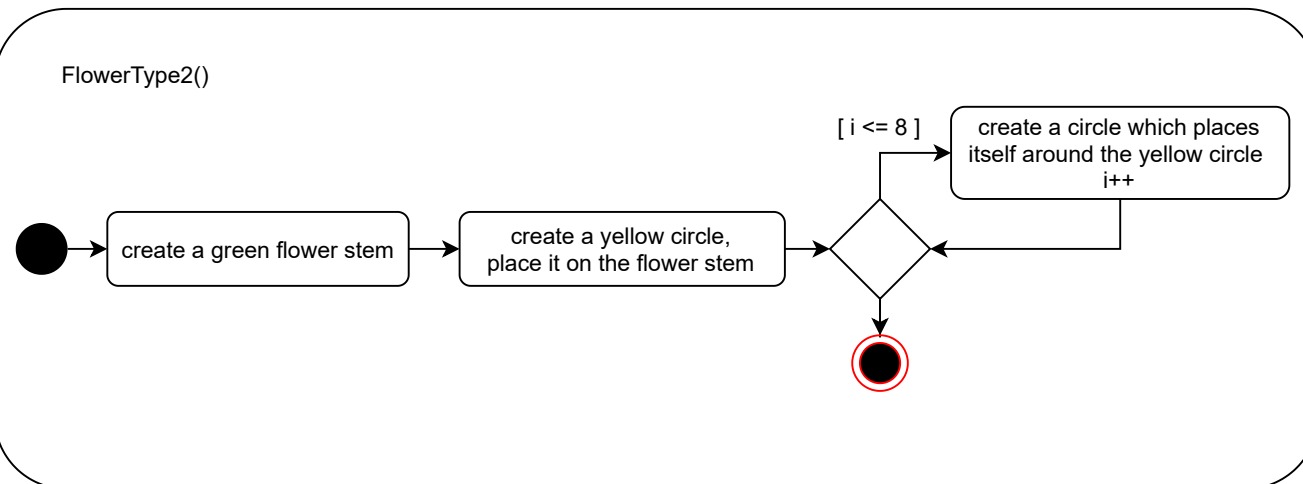
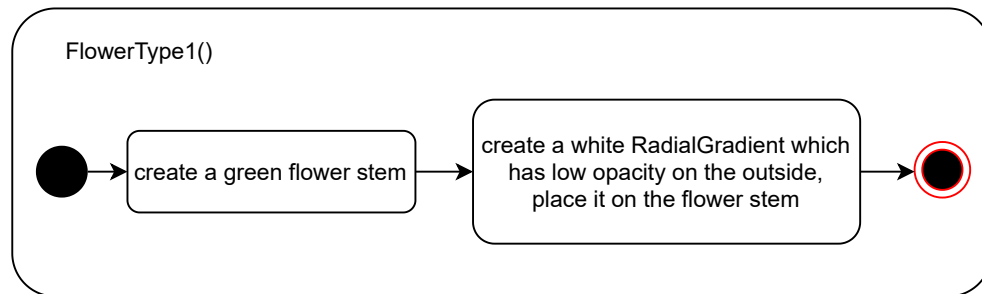
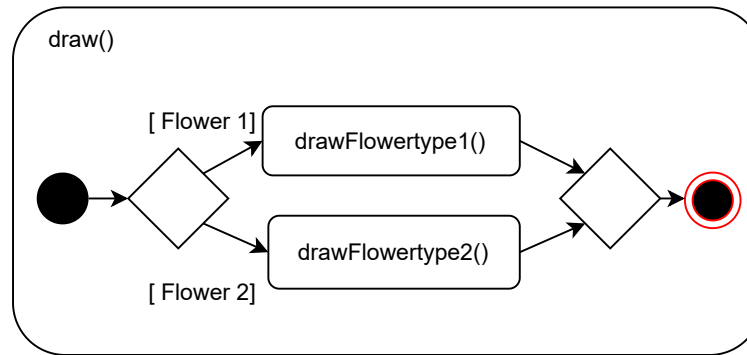
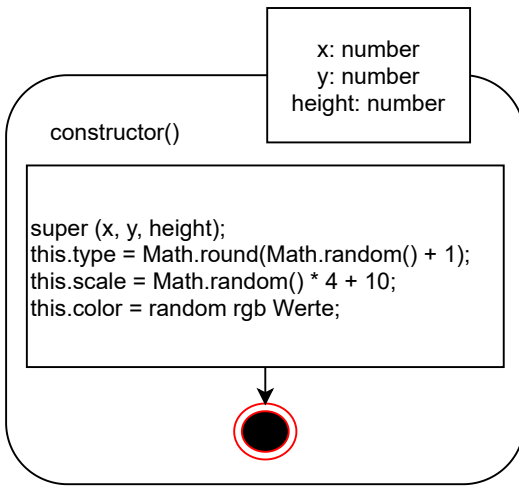
Aktivitätsdiagramm: Bee



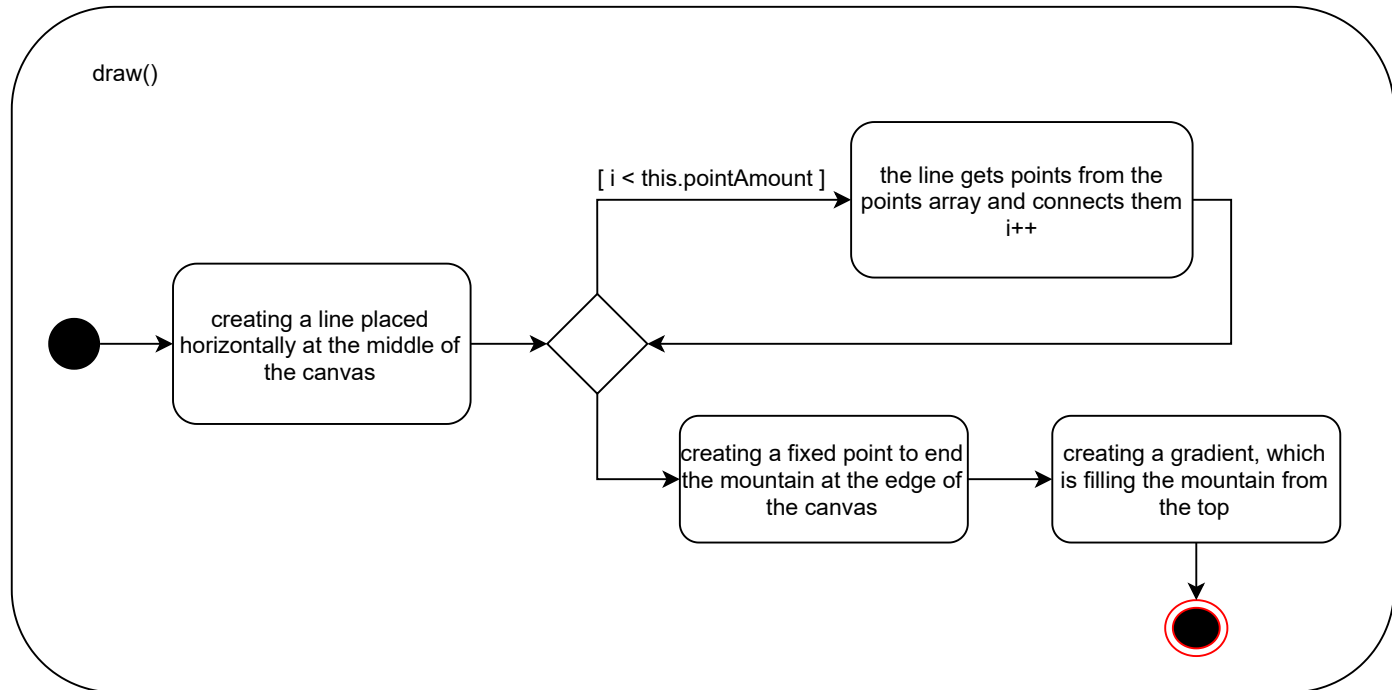
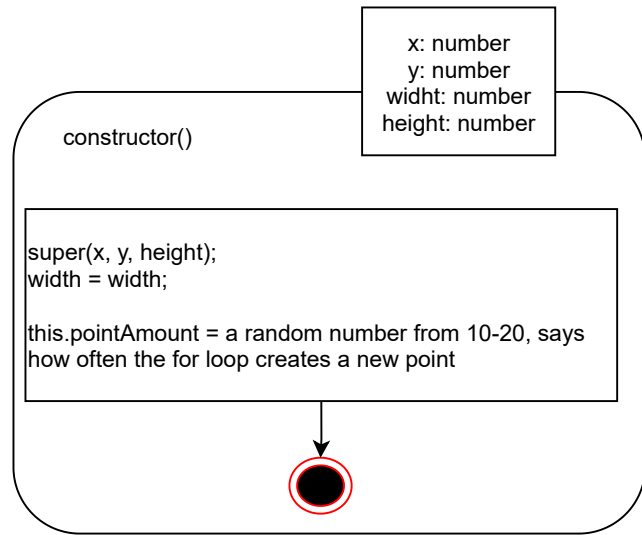
updateVelocities()



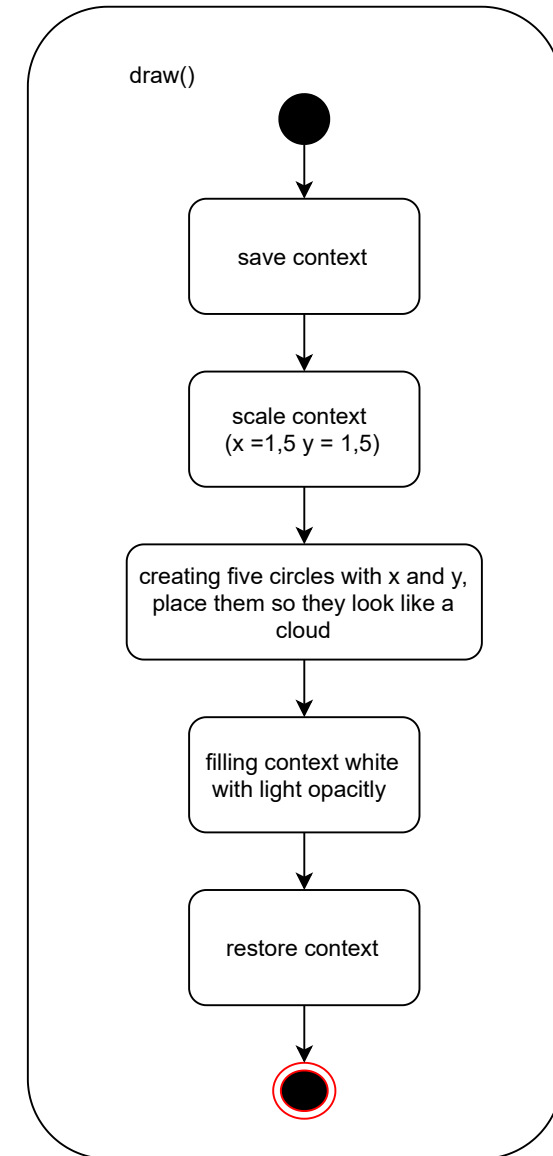
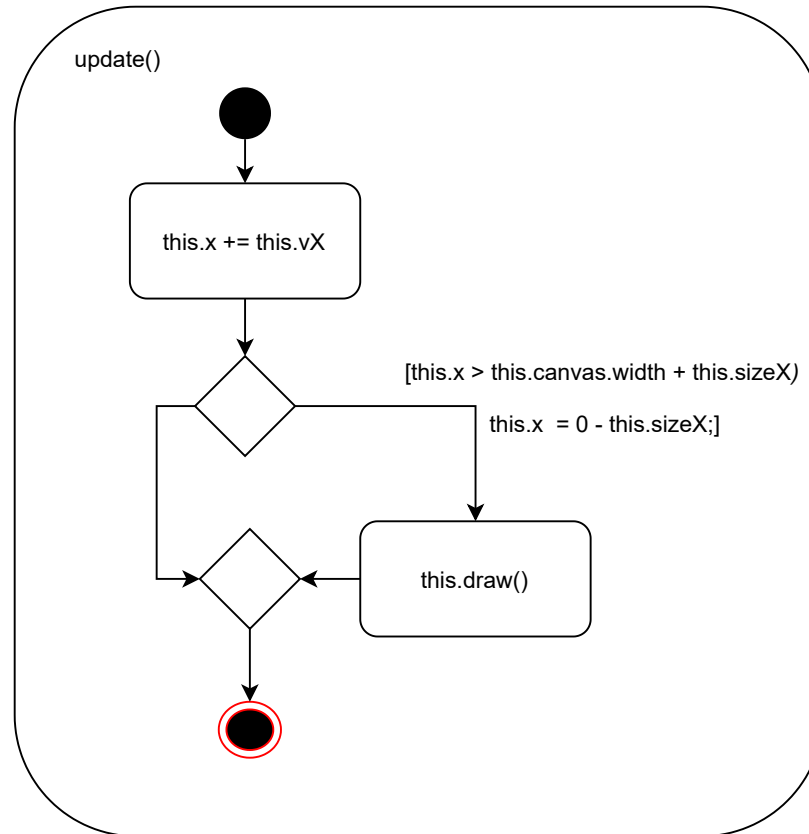
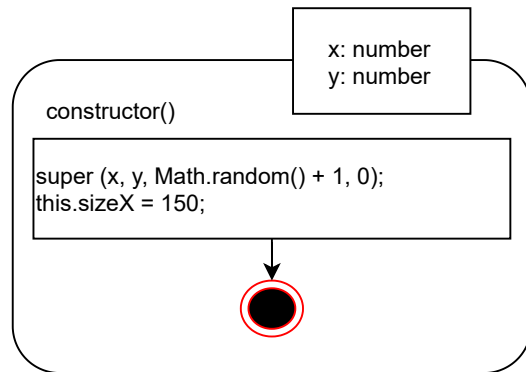
Aktivitätsdiagramm: Flower



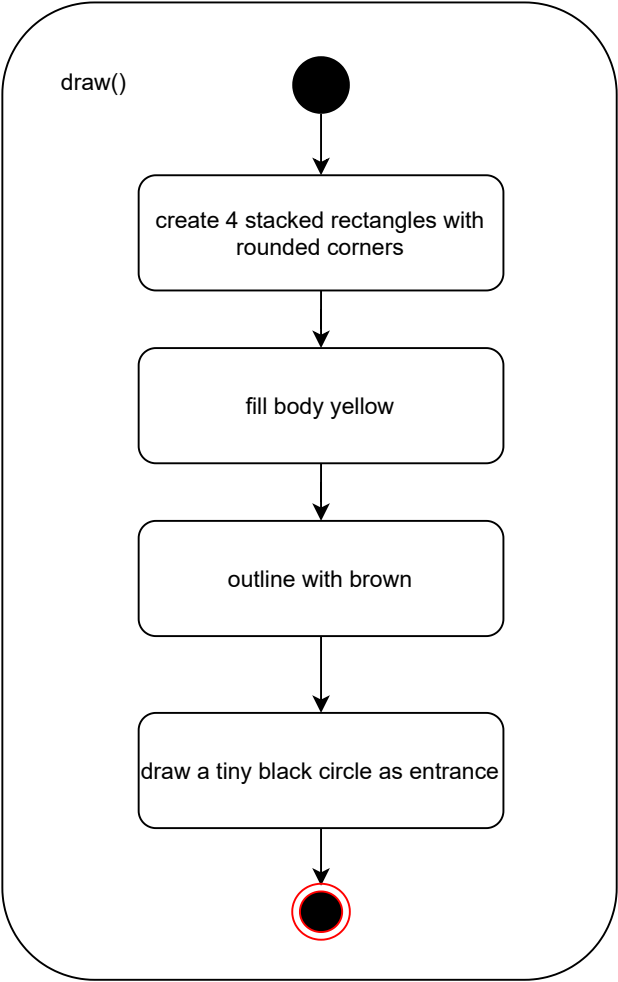
Aktivitätsdiagramm: Mountain



Aktivitätsdiagramm: Cloud



Aktivitätsdiagramm: BeeHive



Aktivitätsdiagramm: Script

```
timeScale: number = 0.005;  
(kann angepasst werden, beeinflusst die Geschwindigkeit,  
in der der Tag zur Nacht wird und umgekehrt)  
scene: Scene = new Scene(timeScale);  
  
setInterval (updateAll, 30)  ⚡
```

