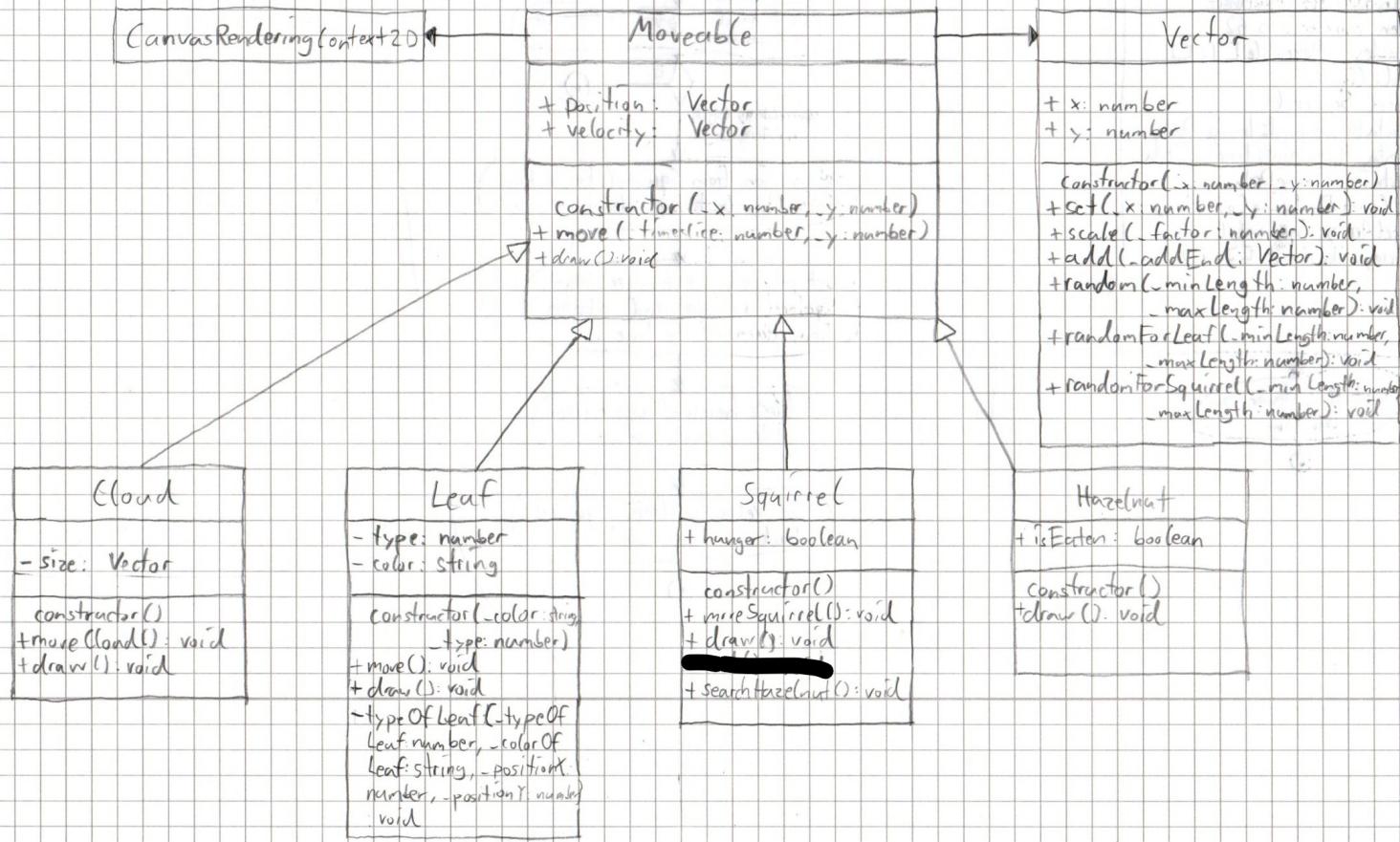
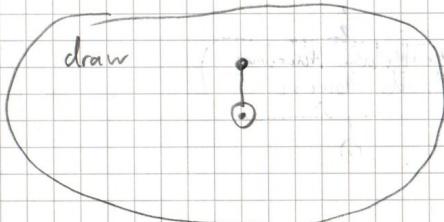
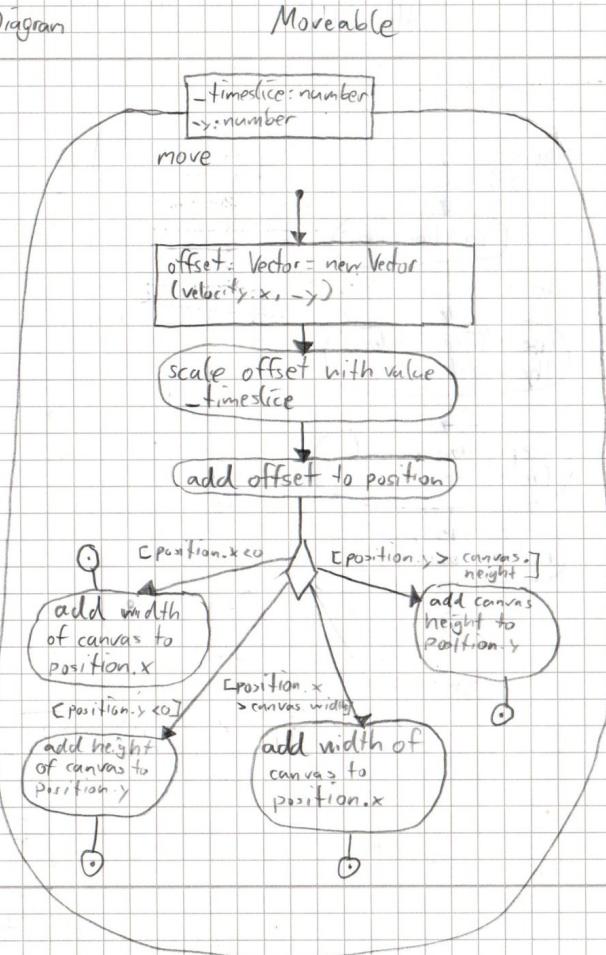
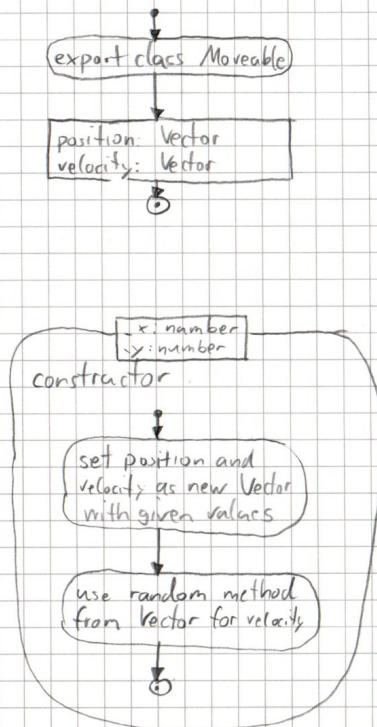


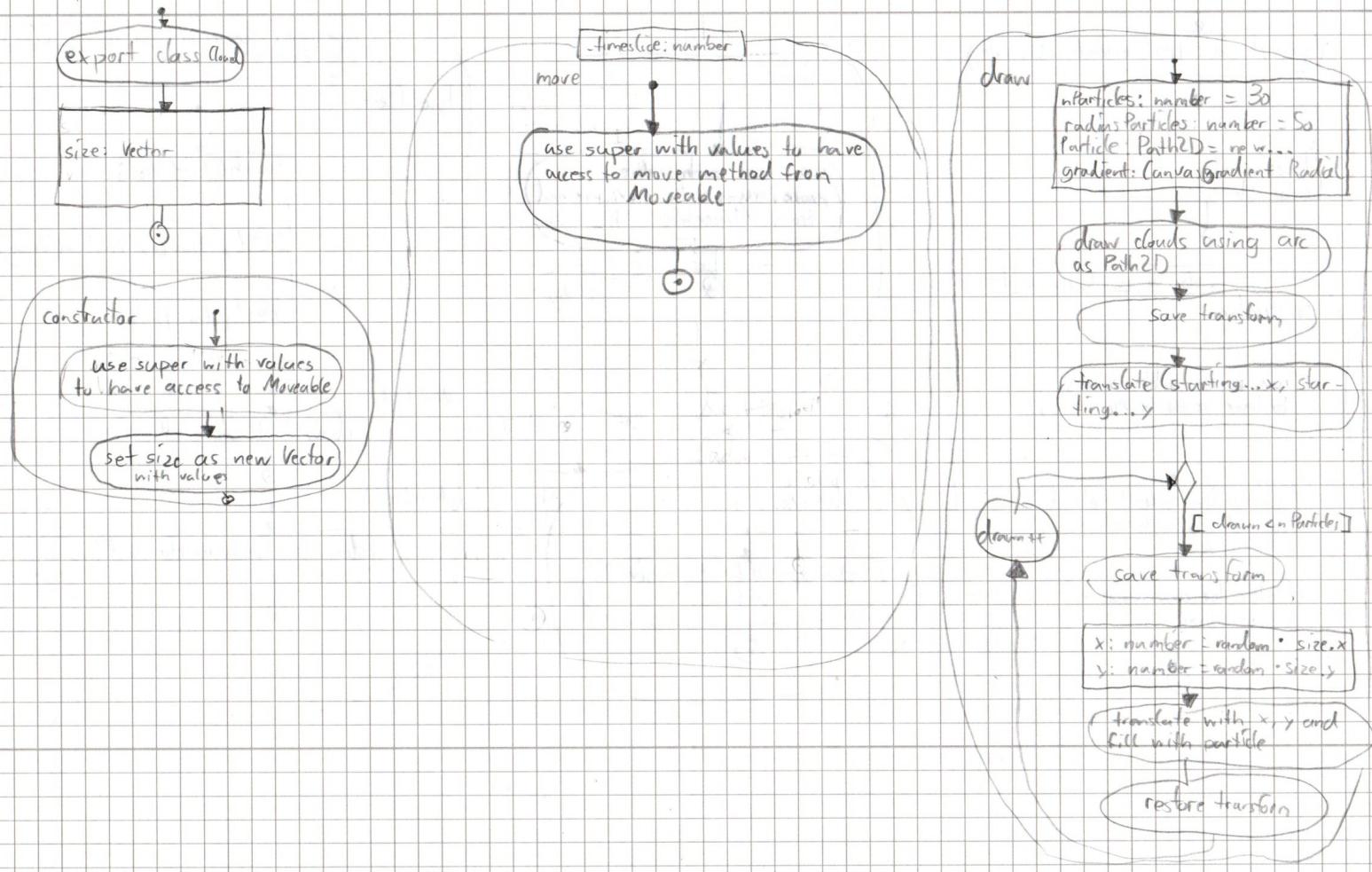
## Goldener Herbst Polymorphie: Classdiagramm



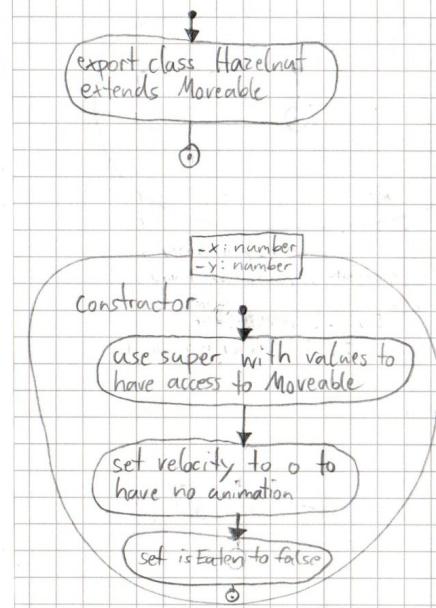
## Goldener Herbst Polymorphie: Activity - Diagram



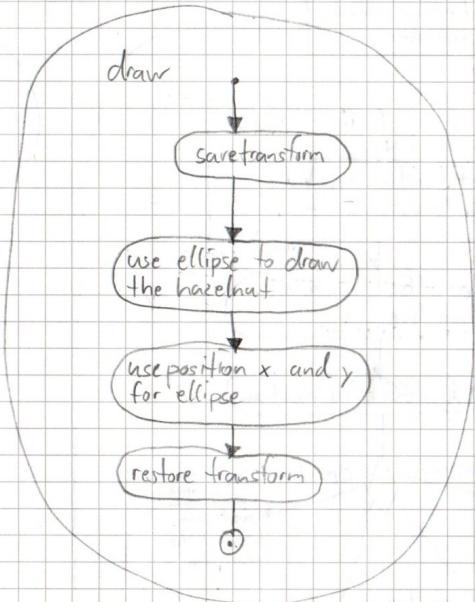
## Goldenherbs/Polyomniptre: Activity diagram



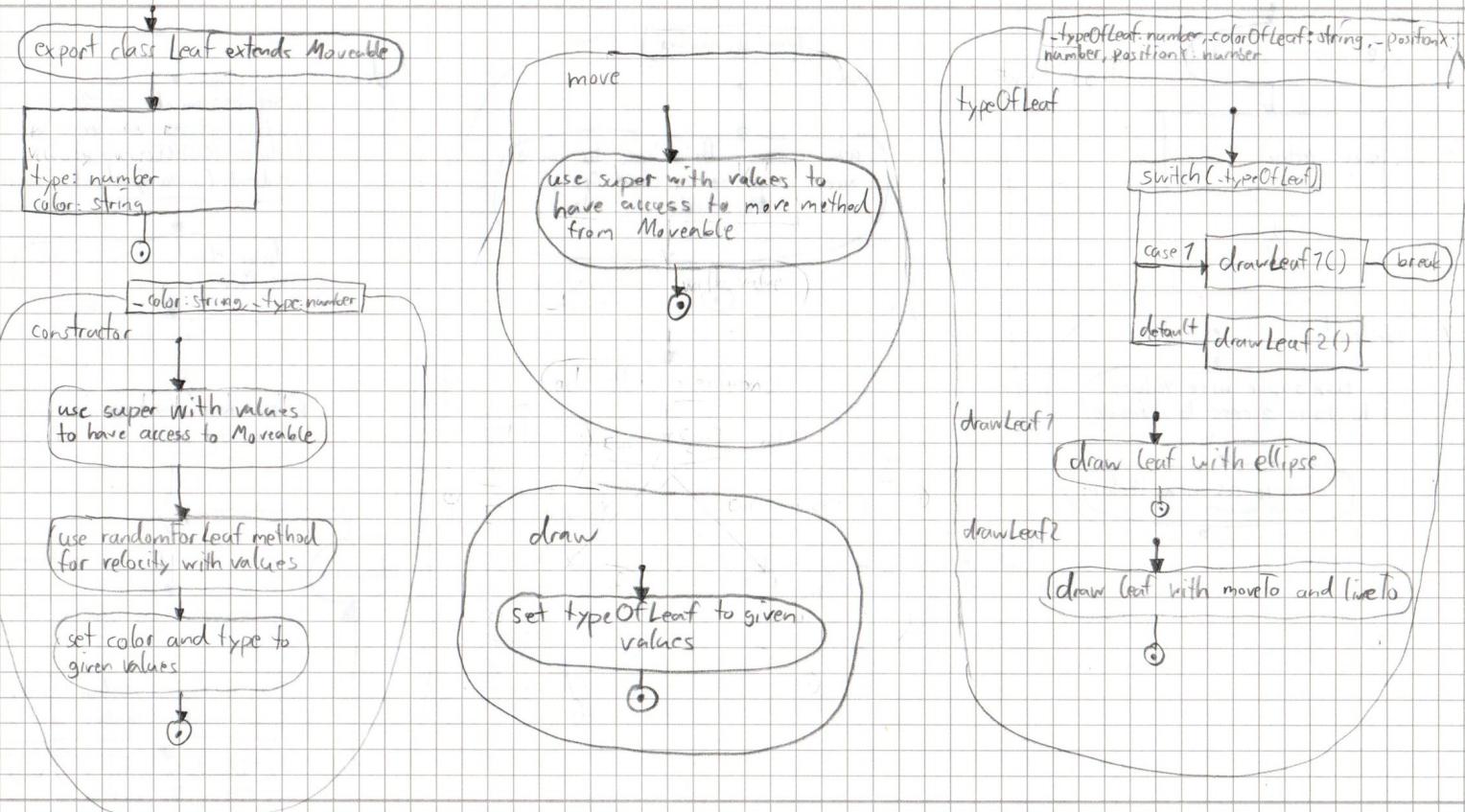
## Goldener Herbst



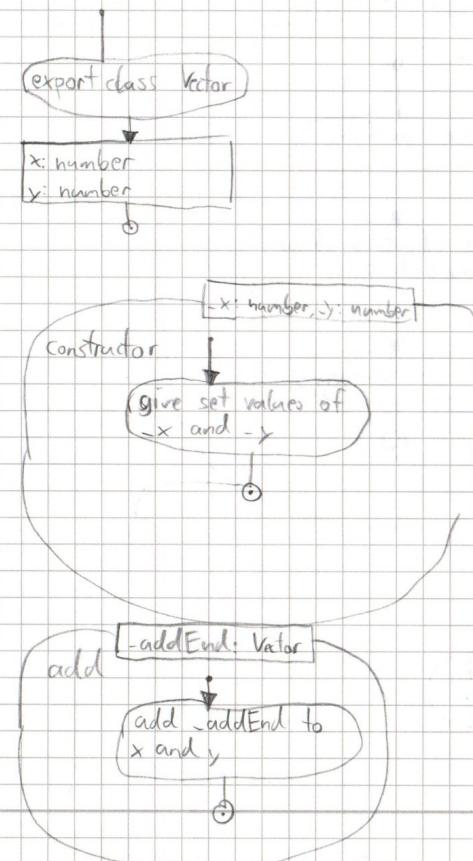
## Hazelnut



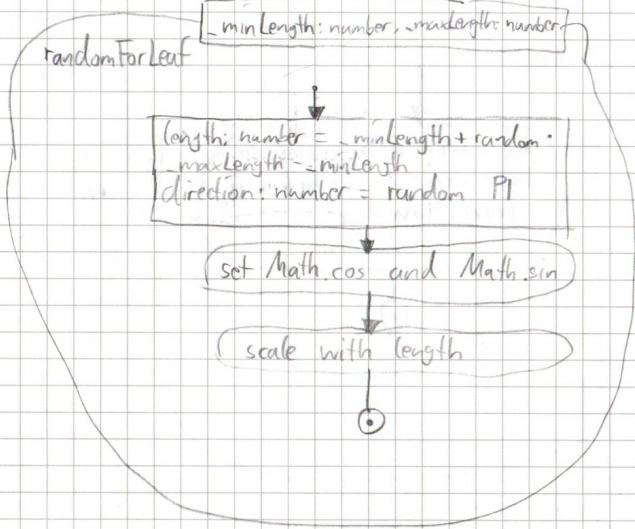
## Golden Tickets + Polymorphism Activity - Diagram



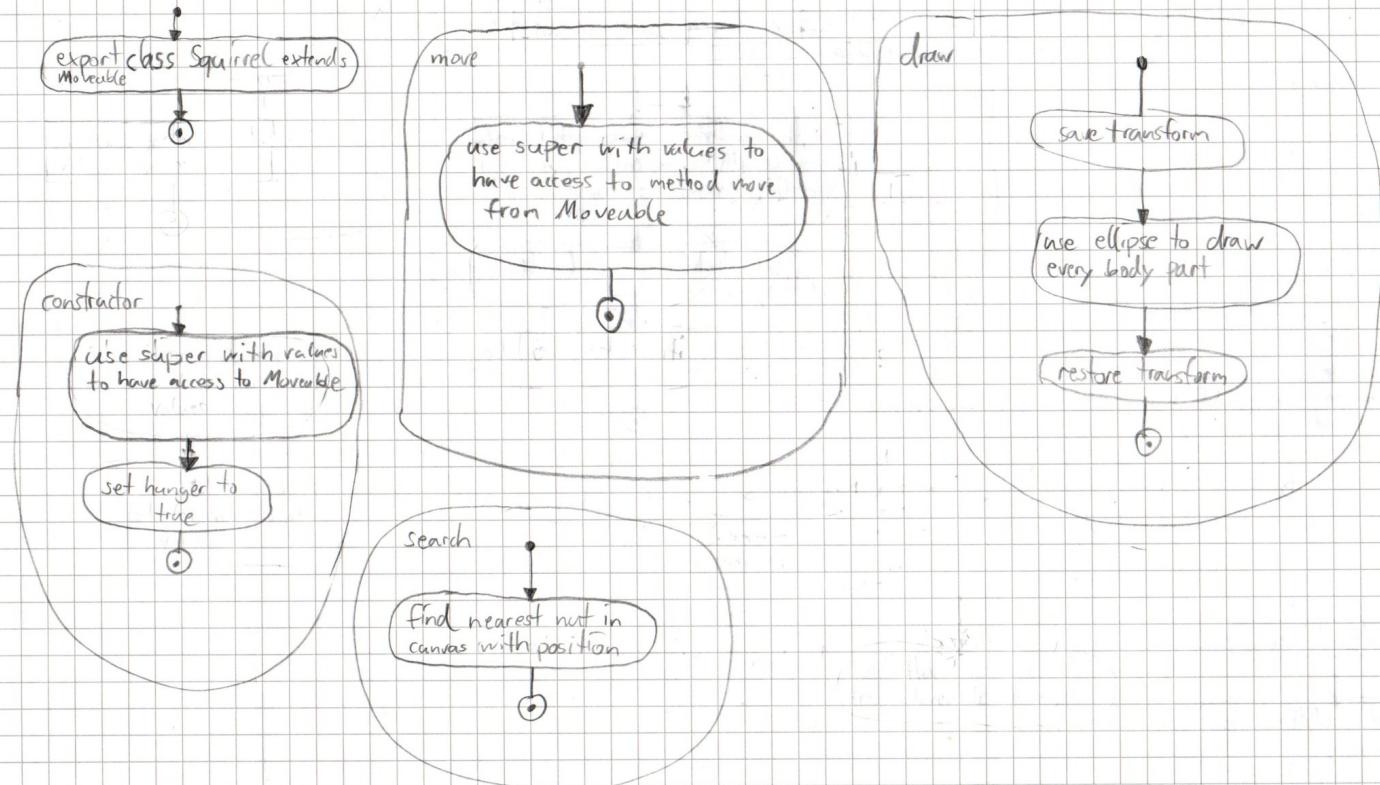
## Goldener Herbst Polymorphie: Activity-Diagram



## Vector



# Goldener Herbst Polymorphie: Activity-Diagram



## Goldener Hirsch (B) - Activity-Diagram

```

hazelNotObject: HazelNot
golden: number = 0.67
moveables: Moveable[] = []
cloud: Cloud
Squirrel: Squirrel
imgDataBackground: ImageData
imgDataMountains: ImageData
imgDataTree: ImageData
c2: CanvasRenderingContext2D
    
```

Install listener on window

event: Event

handleLoad

```

canvas: HTMLCanvasElement
c2: CanvasRenderingContext2D
horizon: number = canvas.height
golden
startPointOfFirstMountainX: number
= Math.random()
    
```

drawBackground()  
createCloud()  
drawMountains(startPoint..., horizon)  
drawTree(horizon + 50)  
createSquirrel()  
createLeaf

20x  
per sec.

Install mouseevent on canvas  
do time controlled activity

Σ load  
handleLoad

Σ mousedown  
createHazelNot

Main

drawTree

declare numerous variables to draw tree

save transform

restore  
save tree with get image data

i++

draw tree with moveTo and  
lineTo

drawMountains

startPoint..., horizon: number

```

gradient: CanvasGradient (linear)
x: number = 0
drawLineRandomX: number = 0
    
```

save transform

restore

[i > 0]

drawLineRandomX = random  
drawLineRandomY: number = random

add drawLine Random X to x

use moveTo and lineTo to  
draw mountains

drawBackground

gradient: CanvasGradient (linear)

use FillRect for background

save background with  
get image data

## Goldener Herbst Polymorphie: Activity-Diagram

Main

createLeaf

```
[C < 20]
colors: string [] = [C]
randomNumberForColor: number = random
randomLeafType: number = random
leaf: Leaf = new Leaf(colors[randomNumberForColor], randomLeafType)
```

push the leaf Objects to  
moveables array

use draw method to  
draw the leaf

create榛子

```
hazelnut: ice = new (-event)
(ClientX, -event ClientY)
```

push hazelnutObject to move  
able Array

createCloud

```
Cloud = new Cloud
```

Push cloud Object to  
moveables array

createSquirrel

```
Squirrel = new Squirrel
```

Push squirrel object to  
moveables array and  
squirrelArray

update

use pullImageData to draw the  
images

we draw and move method  
for cloud and squirrel

use for of loop to draw  
and move the leafs

event: Mouse

~~create榛子~~

## Goldener Herbst

```
hazelnutObject: Hazelnut  
golden: number = 0.62  
moveables: Moveable[] = []  
cloud: Cloud  
squirrel: Squirrel  
imgDataBackground: ImageData  
imgDataMountains: ImageData  
imgDataTree: ImageData  
crc2: CanvasRenderingContext2D  
squirrelArray: Squirrel[] = []  
nutArray: Hazelnut[] = []
```

install Listener On Window

## Main (2)

-event.MouseEvent

Create Hazelnut

hazelnutObject = new C.event.ClientX,  
event.ClientY

push hazelnutObject to moveables  
and nut Array

squirrelArray.length  
= nutArray.length

set position from Squirrel to partition  
from Hazelnut and velocity to 0

set property of hanger from  
Squirrel to false and isEaten  
from Hazelnut to true

delete hazelnutObject from  
nutArray and canvas

set velocity from Squirrel  
back to 100, 200