Use Case Diagram (top section)

Actor

- Settings
  - set start budget
  - set fluctuation
- click field
- display with actions pops up
  - field is empty
    - field is ready → plant → reduce owned seeds
    - field is not ready
  - field is not empty
    - plant has vermin → fight vermin → delete all vermin on the plant
    - water the plants → raise field's water
    - fertilize → raise field's fertilizer → reduce owned fertilizer
    - harvest
      - veggie is unripe → Alert "You ain't gonna get nothin kid. Be patient."
      - veggie is ripe → sell veggie
      - veggie is overripe → Alert "Ain't no more than half for this one. Watch out next time kid." → Sell Veggie for half the price
    - view veggie price
    - view veggie health
    - view veggie growth
- click shop
  - go shopping
    - buy seeds → add fertilizer → subtract price from budget
    - buy fertilizer → add seeds → subtract price from budget

GardenSim: ClassDiagram

Class Diagram (bottom section)

<<enumeration>> GROWTH
+ SEEDLING
+ UNRIPE
+ RIPE
+ OVERRIPE
+ DEAD

Account
+ money: number
+ carrot: number
+ cucumber: number
+ potato: number
+ pumpkin: number
+ spinach: number
+ fertilizer: number
---
constructor

Vermin
- position: Vector
- target: Vector
- action: ACTION
- plant: Plant
---
constructor(_plants: Plant[])
+ update(): void
+ die(): void
# draw(): void
# eat(): void
# fly(): void

Field
+ status: STATUS
+ position: Vector
+ water: number
+ fertilizer: number
# plant:Plant[]
- div: HTMLDivElement
- rect: Description
---
constructor(_position: Vector)
+ clicked(_event: MouseEvent)
+ return(): void
+ water(): void
+ fertilize(): void
+ info(): void
+ plant(): Plant

Plant
+ position: Vector
+ health: number
+ vermins: Vermin[]
+ price: number
+ name: string
+ growth: string
+ filed: Field
+ level: number
+ growthRate: number
+ maxHealth: number
+ maxWater: number
+ maxFertilizer: number
---
constructor(_position: Vector)
+ liberate(): void
+ exist(): void
# rot(_speed ? number = 1)
# wither(): void

Carrot
constructor(_position?: Vector, _field?: Field)

Cucumber
constructor(_position?: Vector, _field?: Field)

Potato
constructor(_position?: Vector, _field?: Field)

Pumpkin
constructor(_position?: Vector, _field?: Field)

Spinach
constructor(_position?: Vector, _field?: Field)

Extends (Carrot, Cucumber, Potato, Pumpkin, Spinach → Plant)

GardenSim: ActivityDiagram - Plant

constructor — _position: Vector, _field: Field

- this.position = _position
- this.field = _field
- this.growth = "seedling"
- this.level = 1
- this.health = this.maxHealth = 100
- this.vermins = []
- this.exist()

rot — _speed: number = 1

- this.health -= _speed
- [this.health > 0]
- this.growth = GROWTH.DEAD

die

- this.position.setPosition(-10, -10)
- this.field.reset()
- this.field.fieldPlant.pop()

liberate

- this.vermins = []

constructor — _position?: Vector, _field?: Field

- super(_position, _field)
- this.name = "carrot"
- this.maxWater = 500
- this.maxFertilizer = 275

constructor — _position?: Vector, _field?: Field

- super(_position, _field)
- this.name = "potato"
- this.maxWater = 350
- this.maxFertilizer = 200

constructor — _position?: Vector, _field?: Field

- super(_position, _field)
- this.name = "cucumber"
- this.maxWater = 400
- this.maxFertilizer = 250

constructor — _position?: Vector, _field?: Field

- super(_position, _field)
- this.name = "pumpkin"
- this.maxWater = 800
- this.maxFertilizer = 300

constructor — _position?: Vector, _field?: Field

- super(_position, _field)
- this.name = "spinach"
- this.maxWater = 200
- this.maxFertilizer = 50

constructor — _money: number

- this. money = _money
- give five seeds each
- this.fertilizer = 0

after this.growthRate * 2 seconds

after this.growthRate seconds

exist

- [this.growth == GROTH.UNRIPE] → this.maxHealth = 150
- [this.growth == GROWTH.RIPE] → this.maxHealth = 200
- [this.growth == GROWTH.OVERRIPE] → this.rot(this.level - 100)
- [this.growth == GROTH.DEAD] → this.die

- this.field.water > 0
- [this.health == this.maxHealth]
- this.health++
- this.field.water--
- [this.field.fertilizer == 0] → this.level++
- [this.field.fertilizer is between 1 and maxFertilizer] → this.level++ → this.field.fertilizer--
- this.field.fertilizer > this.maxFertilizer → this.rot
- this.growth = GROWTH.UNRIPE
- this.growth = GROWTH.RIPE
- this.growth = GROWTH.OVERRIPE

- this.field.water > this.maxWater → this.rot(0.5)
- this.field.water == 0 → this.rot
- [this.growth == GROWTH.DEAD]
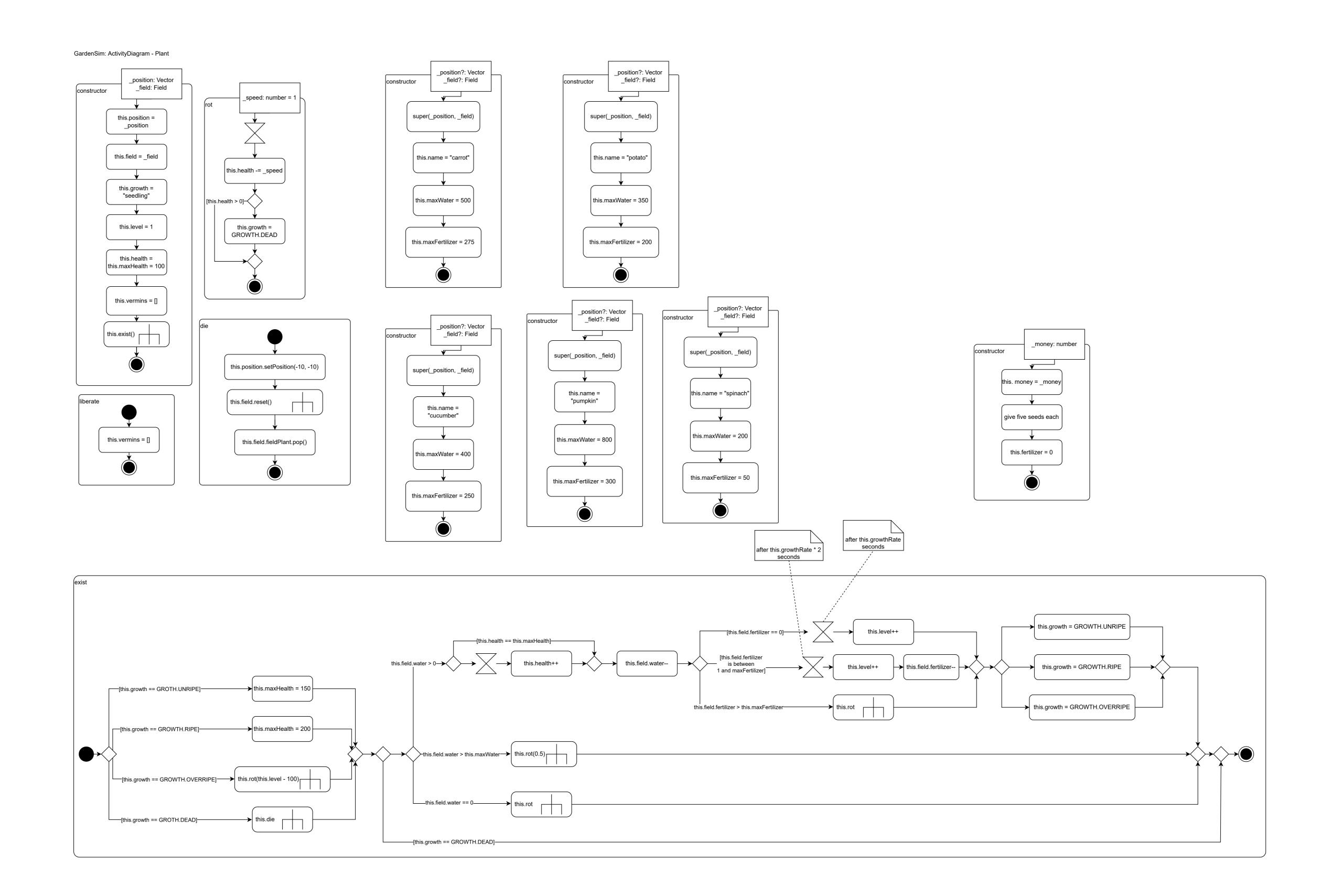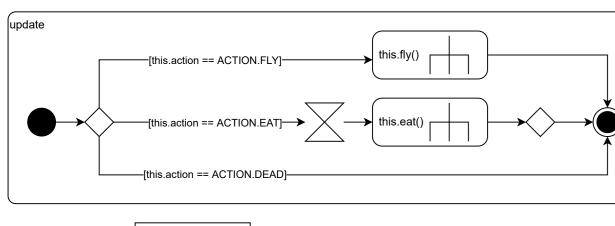
GardenSim Activity Diagram - Vermin

Activity Diagram - Field

update

[this.action == ACTION.FLY] → this.fly()

[this.action == ACTION.EAT] → this.eat()

[this.action == ACTION.DEAD]

constructor

_plants: Plant[]

let x be a random number between 0 and 39

this.plant = _plants[x]

this.target = this.plant.position

this.position = new Vector offscreen

this.action = ACTION.FLY

this.draw()

die

this.action = ACTION.DEAD

this.position.setPosition(-10, -10)

constructor

_wrapper: HTMLDivElement

this.div = document.createElement("div")

this.rect= div.getBoundingClientRect()

this.position = new Vector(this.rect.left.toFixed() + 50, this.rect.top.toFixed() + 50)

this.div.className = "ready"

_wrapper.appendChild(this.div)

this.plant = []

this.water = 100

this.fertilizer = 0

this.status = STATUS.READY

add click listener with this.clicked to this.div

info

_event: MouseEvent

let button: HTMLButtonElement = new Button

delete existing form element

display the plant's and field's stats in the info div

install listener on button with this.clicked

append info div by the button

water

this.water += 50

reset

this.status = STATUS.READY

fertilize

_account: Account

this.fertilizer += 25

_account.fertilizer--

harvest

_account: Account

[this.plant[0].growth == GROWTH.UNRIPE] → send alert "You ain't gonna get nothing kid. Be patient"

[this.plant[0].growth == GROWTH.RIPE] → _account.money += this.plant[0].price → this.plant.pop() → [this.div.hasChildNodes] this.div.removeChild(this.div.firstChild) → this.div.className = ready → this.status = STATUS.READY

[this.plant[0].growth == GROWTH.OVERRIPE] → _account.money += this.plant[0].price / 2 → Alert "Ain't no more than half for this one. Watch out next time kid." → [this.div.hasChildNodes] this.div.removeChild(this.div.firstChild) → this.div.className = ready → this.status = STATUS.READY

Activity Diagram - Main

carrot — _event: MouseEvent

- (start)
- install load-listener
- (end)

carrot — _event: MouseEvent
- let carrot: Plant = new Carrot(clickedField[0].position, clickedField[0]);
- clickedField[0].pictures = ["sprossling.png", "", "", ""];
- clickedField[0].fieldPlant.push(carrot);
- clickedField[0].occupied = true;
- account.carrots--;
- document.getElementById("carrot").innerHTML = `Carrot: ${account.carrots}`;
- (end)

cucumber — _event: MouseEvent
- let cucumber: Plant = new Cucumber(clickedField[0].position, clickedField[0]);
- clickedField[0].pictures = ["sprossling.png", "", "", ""];
- clickedField[0].fieldPlant.push(cucumber);
- clickedField[0].occupied = true;
- account.cucumber--;
- document.getElementById("cucumberLabel").innerHTML = `Cucumber: ${account.cucmber}`;
- (end)

buy
- (start)
- let formData: FormData = FormData(document.forms.shop)
- account.carrots += parseInt(`${formData.get("carrotSeeds")}`)
- account.cucumber += parseInt(`${formData.get("cucumberSeeds")}`)
- account.potatos += parseInt(`${formData.get("potatoSeeds")}`)
- account.pumpkin += parseInt(`${formData.get("pumpkinSeeds")}`)
- account.spinach += parseInt(`${formData.get("spinachSeeds")}`)
- account.fertilizer += parseInt(`${formData.get("fertilizer")}`)
- account.pesticide += parseInt(`${formData.get("pesticide")}`)
- subtract total price from account.money
- Reset shop
- (end)

_event: MouseEvent (createGarden)
- let wrapper: HTMLDivElement = <HTMLDivElement>document.getElementById("wrapper"); let field: Field; let waterButton: HTMLButtonElement = <HTMLButtonElement>document.getElementById("water"); let fertilizeButton: HTMLButtonElement = <HTMLButtonElement>document.getElementById("fertilize"); let formData: FormData = new FormData(document.forms[1]);
- set the vegetables' min- and maxPrice to your formData values
- turn the divs "#info" and "#shop" visible
- document.getElementById("wallet").innerHTML = `Wallet: ${account.money} TSh.`;
- wrapper.innerHTML = "";
- (decision)
- let i: number = 0
- (decision) — stonkSwing / (end)
- let j: number = 0
- (decision)
- let x: number = 125 * j; let y: number = 150 * i;
- field = new Field(wrapper, x, y, account);
- field.div.style.left = `${x}px`;
- field.div.style.top = `${y}px`;
- garden.push(field);
- waterButton.addEventListener("click", field.water);
- fertilizeButton.addEventListener("click", field.fertilize);
- (decision)

handleLoad — _event: Event
- document.getElementById("start").addEventListener("click", createGarden); document.getElementById("buy").addEventListener("click", buy); document.getElementById("shop").addEventListener("input", shoppingCart); document.getElementById("carrot").addEventListener("click", carrot); document.getElementById("cucumber").addEventListener("click", cucumber); document.getElementById("potato").addEventListener("click", potato); document.getElementById("pumpkin").addEventListener("click", pumpkin); document.getElementById("spinach").addEventListener("click", spinach);
- window.setInterval(stonkSwing, 20000); window.setInterval(update);
- (end)

potato — _event: MouseEvent
- let potato: Plant = new Potato(clickedField[0].position, clickedField[0]);
- clickedField[0].pictures = ["sprossling.png", "", "", ""];
- clickedField[0].fieldPlant.push(potato);
- clickedField[0].occupied = true;
- account.potatoes--;
- document.getElementById("potato").innerHTML = `Potato: ${account.potatoes}`;
- (end)

pumpkin — _event: MouseEvent
- let pumpkin: Plant = new Pumpkin(clickedField[0].position, clickedField[0]);
- clickedField[0].pictures = ["sprossling.png", "Kürbis-unreif.png", "Kürbis-reif.png", "Kürbis-überreif.png"];
- clickedField[0].fieldPlant.push(pumpkin);
- clickedField[0].occupied = true;
- account.pumpkins--;
- document.getElementById("pumpkin").innerHTML = `Pumpkin: ${account.pumpkins}`;
- (end)

buy — _event: MouseEvent
- let formData: FormData = new FormData(document.forms[1]); let carrots: number = parseInt(`${formData.get("carrotSeeds")}`); let cucmbers: number = parseInt(`${formData.get("cucumberSeeds")}`); let potatos: number = parseInt(`${formData.get("potatoSeeds")}`); let pumpkins: number = parseInt(`${formData.get("pumpkinSeeds")}`); let spinaches: number = parseInt(`${formData.get("spinachSeeds")}`); let fertilizer: number = parseInt(`${formData.get("fertilizer")}`); let pesticide: number = parseInt(`${formData.get("pesticide")}`);
- update every value of account
- subtract the total shopping sum from account.money
- Update the wallet, the vegetable buttons and the fertilizer and pesticide button
- (end)

spinach — _event: MouseEvent
- let spinach: Plant = new Spinach(clickedField[0].position, clickedField[0]);
- clickedField[0].pictures = ["sprossling.png", "spinat-unreif.png", "spinat-reif.png", "spinat-überreif.png"];
- clickedField[0].fieldPlant.push(spinach);
- clickedField[0].occupied = true;
- account.spinach--;
- document.getElementById("pumpkin").innerHTML = `Pumpkin: ${account.pumpkins}`;
- (end)

shoppingCart — _event: Event
- let formData: FormData = new FormData(document.forms[1]); let carrots: number = parseInt(`${formData.get("carrotSeeds")}`) * Carrot.price; let cucumbers: number = parseInt(`${formData.get("cucumberSeeds")}`) * Cucumber.price; let potatos: number = parseInt(`${formData.get("potatoSeeds")}`) * Potato.price; let pumpkins: number = parseInt(`${formData.get("pumpkinSeeds")}`) * Pumpkin.price; let spinaches: number = parseInt(`${formData.get("spinachSeeds")}`) * Spinach.price; let fertilizer: number = parseInt(`${formData.get("fertilizer")}`) * 5; let pesticide: number = parseInt(`${formData.get("pesticide")}`) * 10;
- document.getElementById("shoppingCart").innerHTML = `Total: ${(carrots + cucumbers + potatos + pumpkins + spinaches) / 2 + fertilizer + pesticide} TSh.`
- (end)

stonkSwing
- (start)
- Carrot.price = Carrot.setFluctuation(carrotPriceMin, carrotPriceMax);
- Cucumber.price = Cucumber.setFluctuation(cucPriceMin, cucPriceMax);
- Potato.price = Potato.setFluctuation(potPriceMin, potPriceMax);
- Pumpkin.price = Pumpkin.setFluctuation(pumpPriceMin, pumpPriceMax);
- Spinach.price = Spinach.setFluctuation(spinPriceMin, spinPriceMax);
- Update the shop's seed prices to half the vegetable's price
- (end)