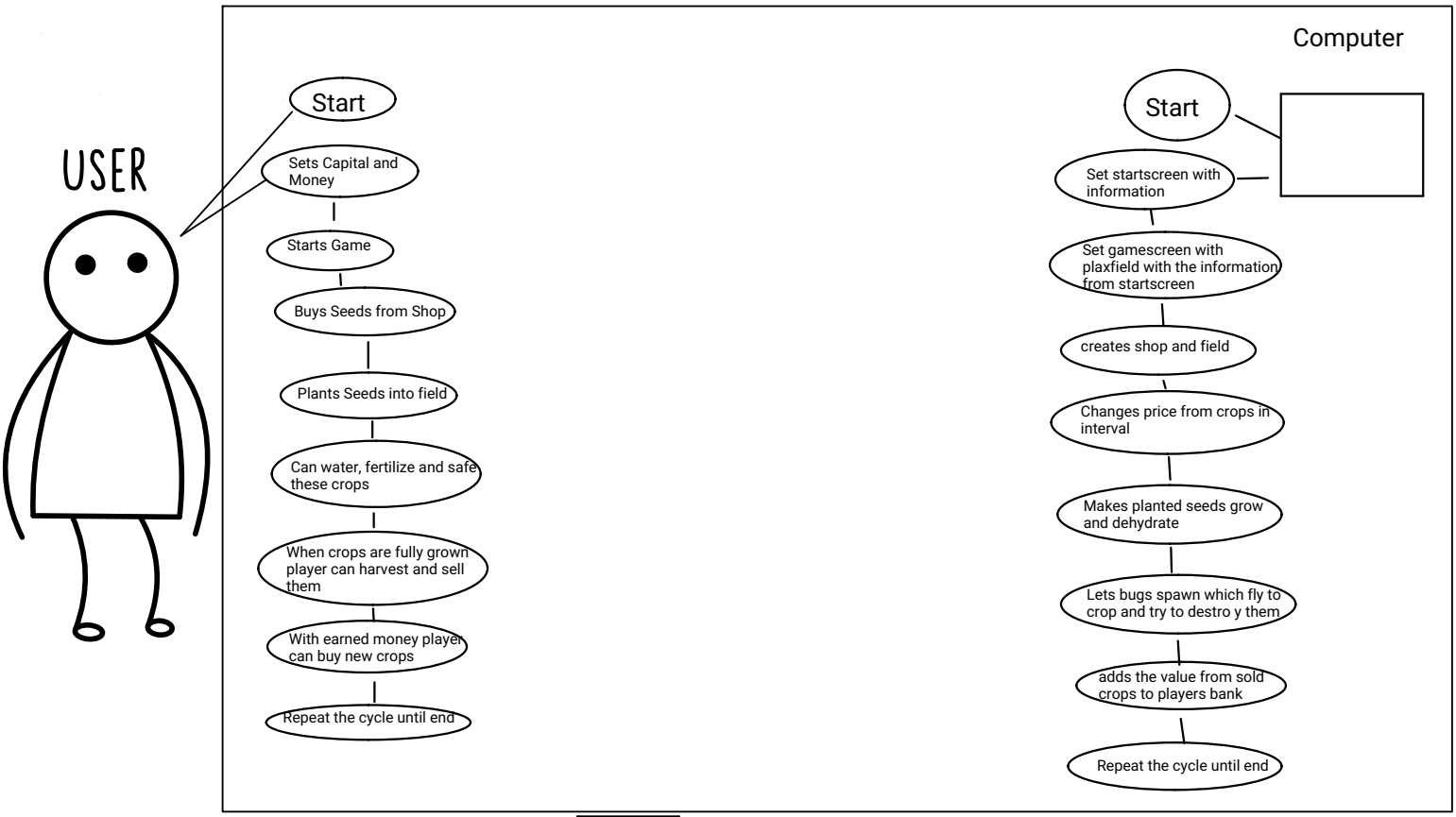
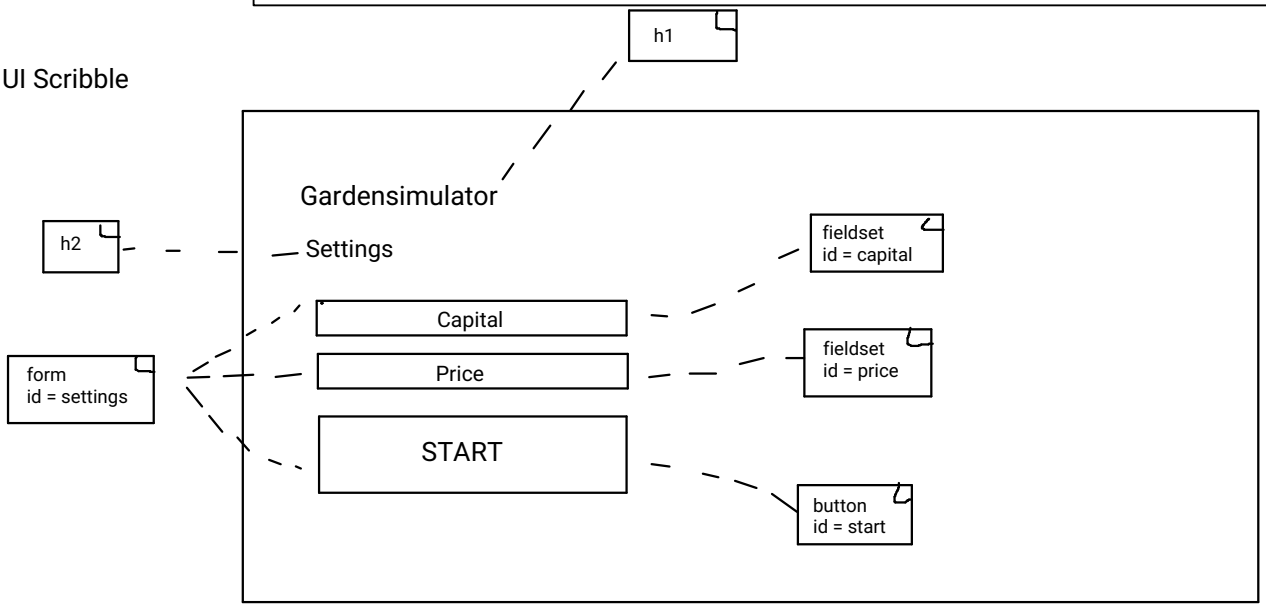
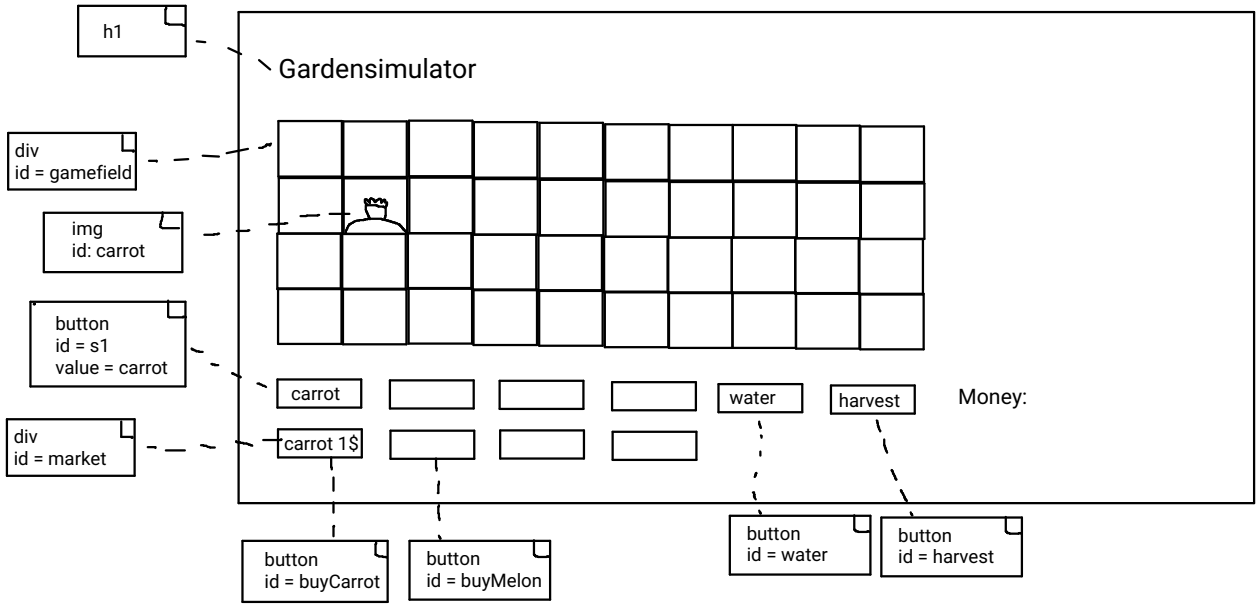

USER
Start
Sets Capital and Money
Starts Game
Buys Seeds from Shop
Plants Seeds into field
Can water, fertilize and safe these crops
When crops are fully grown player can harvest and sell them
With earned money player can buy new crops
Repeat the cycle until end
Computer
Start
Set startscreen with information
Set gamescreen with plaxfield with the information from startscreen
creates shop and field
Changes price from crops in interval
Makes planted seeds grow and dehydrate
Lets bugs spawn which fly to crop and try to destro y them
adds the value from sold crops to players bank
Repeat the cycle until end

UI Scribble


h1
Gardensimulator
h2
Settings
fieldset
id = capital
Capital
form
id = settings
Price
fieldset
id = price
START
button
id = start


h1
Gardensimulator
div
id = gamefield
img
id: carrot
button
id = s1
value = carrot
carrot
water
harvest
Money:
div
id = market
carrot 1$
button
id = buyCarrot
button
id = buyMelon
button
id = water
button
id = harvest

Class Diagram

Load

static ananasS: HTMLImageElement;
static ananasM: HTMLImageElement;
static ananasB: HTMLImageElement;
static blueberryS: HTMLImageElement;
static blueberryM: HTMLImageElement;
static blueberryB: HTMLImageElement;
static melonS: HTMLImageElement;
static melonM: HTMLImageElement;
static melonB: HTMLImageElement;
static raspberryS: HTMLImageElement;
static raspberryM: HTMLImageElement;
static raspberryB: HTMLImageElement;
static strawberryS: HTMLImageElement;
static strawberryM: HTMLImageElement;
static strawberryB: HTMLImageElement;
static dropS: HTMLImageElement;
static dropM: HTMLImageElement;
static dropB: HTMLImageElement;
static fertilizS: HTMLImageElement;
static fertilizM: HTMLImageElement;
static fertilizB: HTMLImageElement;
static bug: HTMLImageElement;
static empty: HTMLImageElement;

static loading( );

Player

static task: <<TASK>>
static pesticideAmount: number;
static fertilizerAmount: number;
static fertilizerPrice: number;
static pesticidesPrice: number;
static duengerPrice: number;
static pesticidePriceNew: number;

plant (_value: string) : void
harvest(): void
water(): void
fertilize (): void
buy (_value: string): void

<<enum>> TASK

PLANTANANAS,
PLANTBLUEBERRY,
PLANTRASPBERRY,
PLANTMELON,
PLANTSTRAWBERRY,
HARVEST,
WATER,
FERTILIZE

Field

row: number;
collum: number;
color: string
isFieldClear: boolean
plant: Plant;

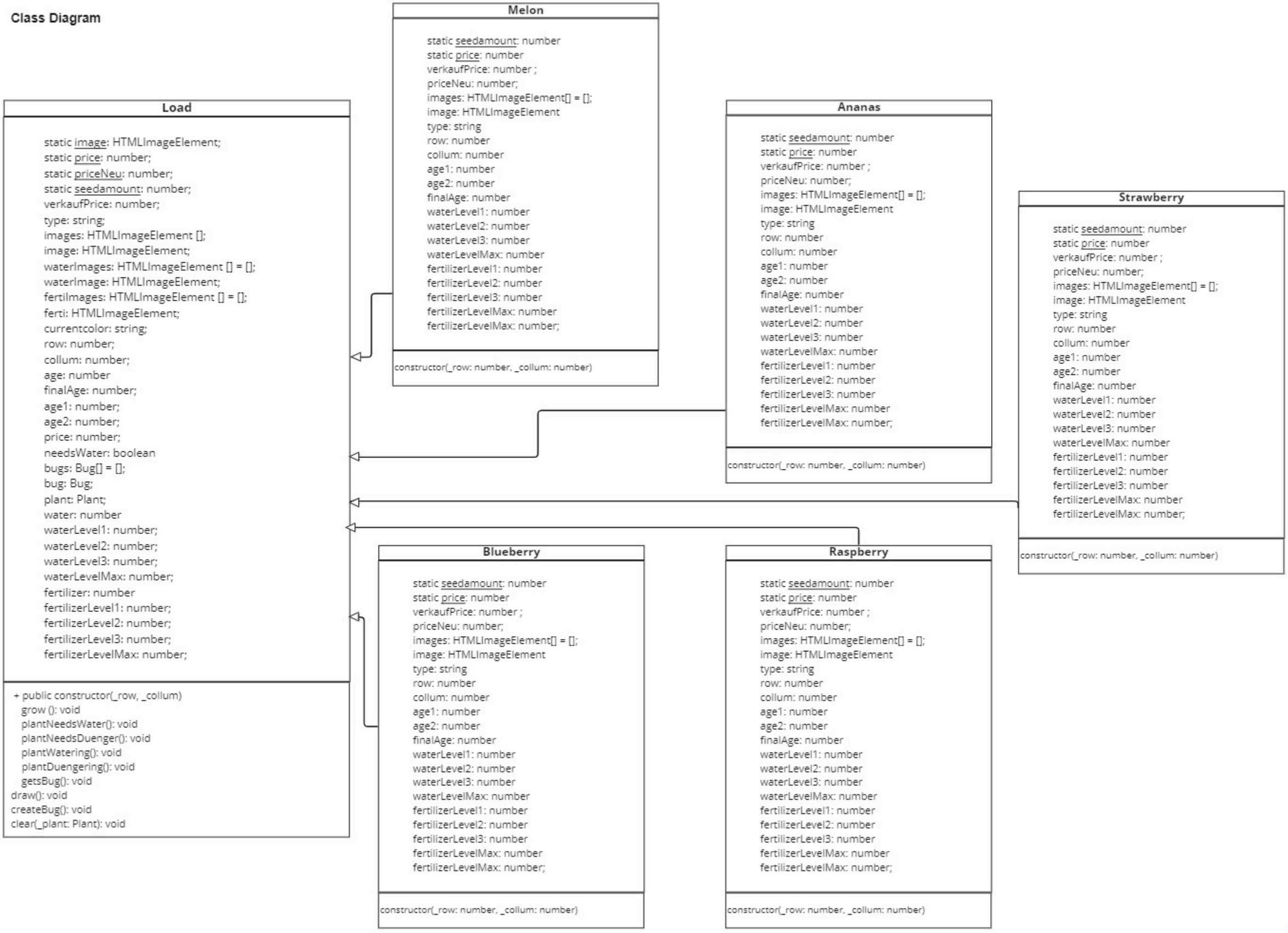
constructor(_row, _collum)
isHit(): void
draw(): void

Bug

+ public position: number
- private row: number;
- private collum: number;

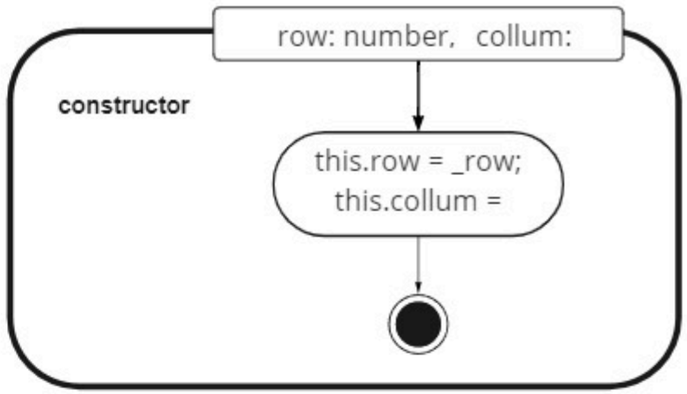
constructor(_row, _collum)
+ public draw (): void
- private move (): void

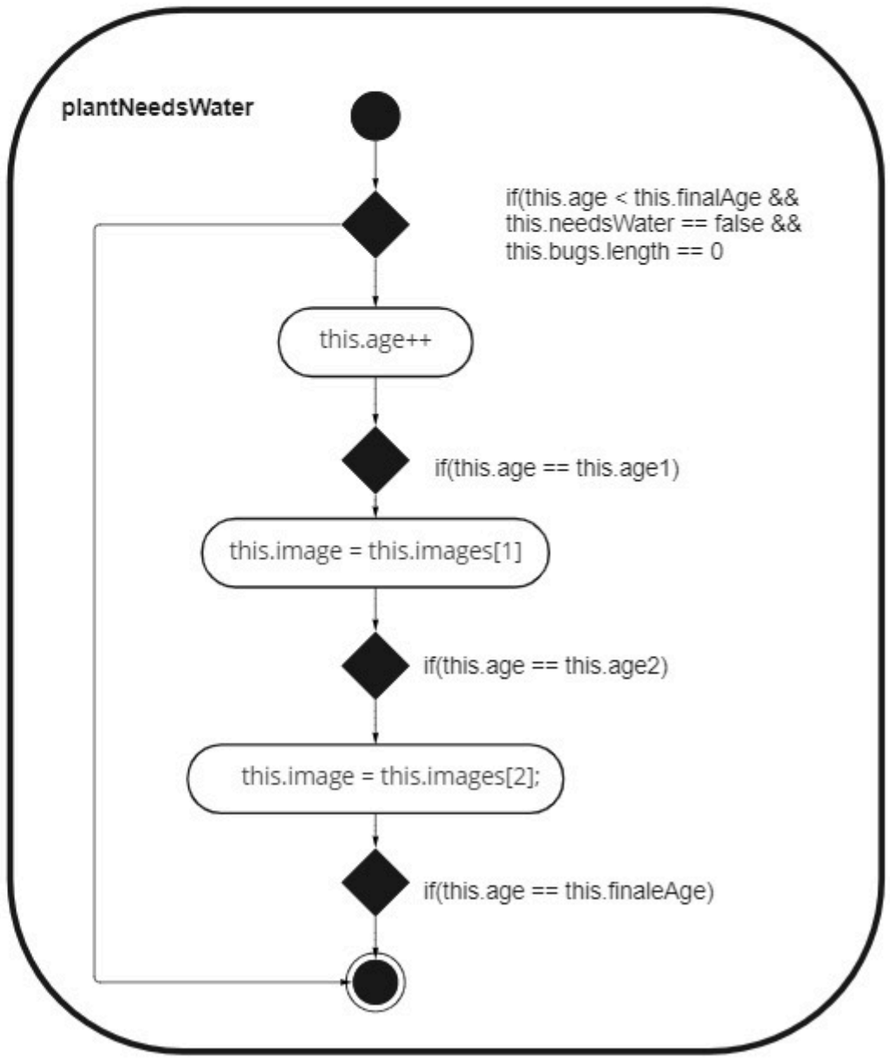
Class Diagram

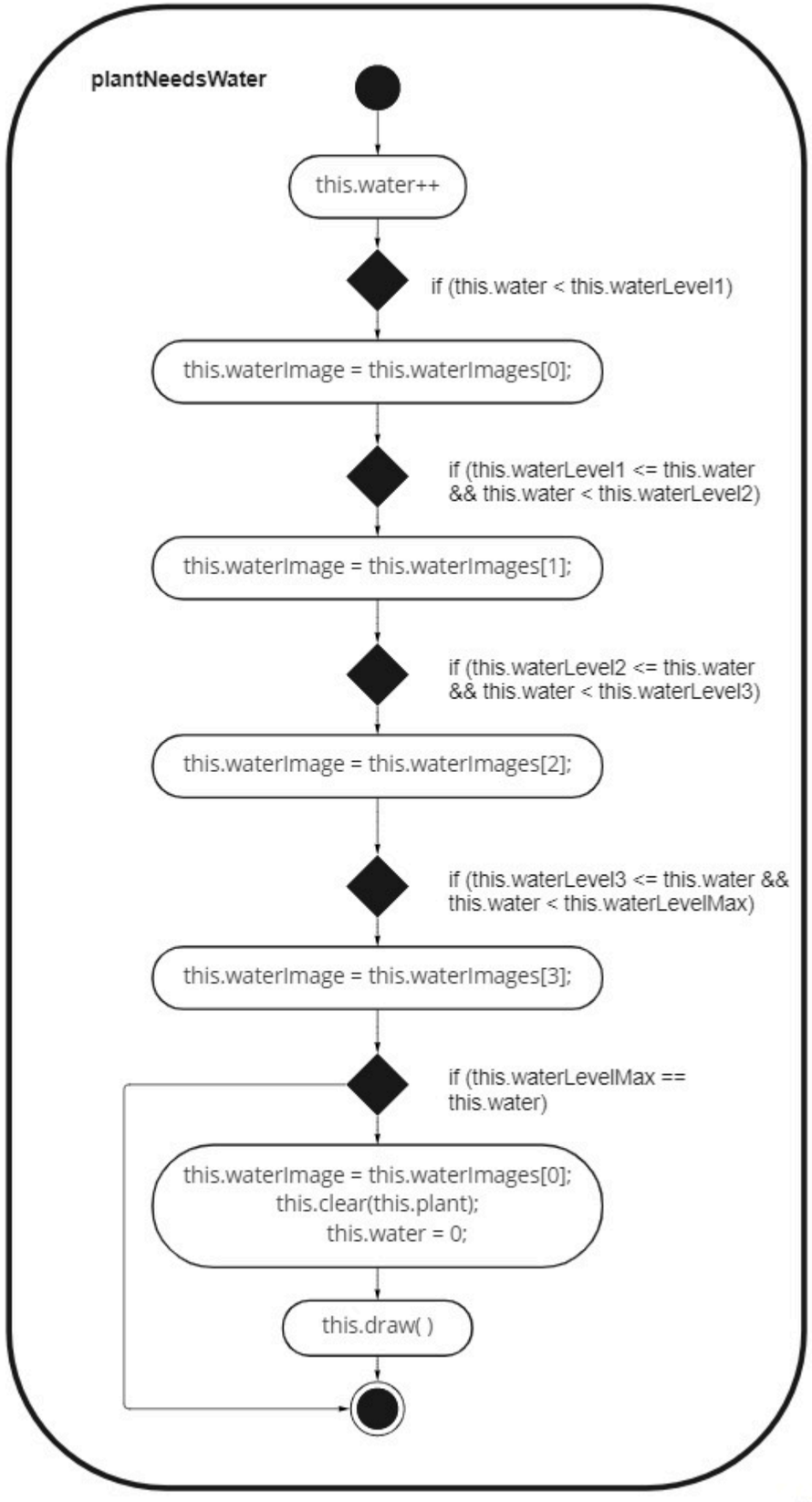


Plant

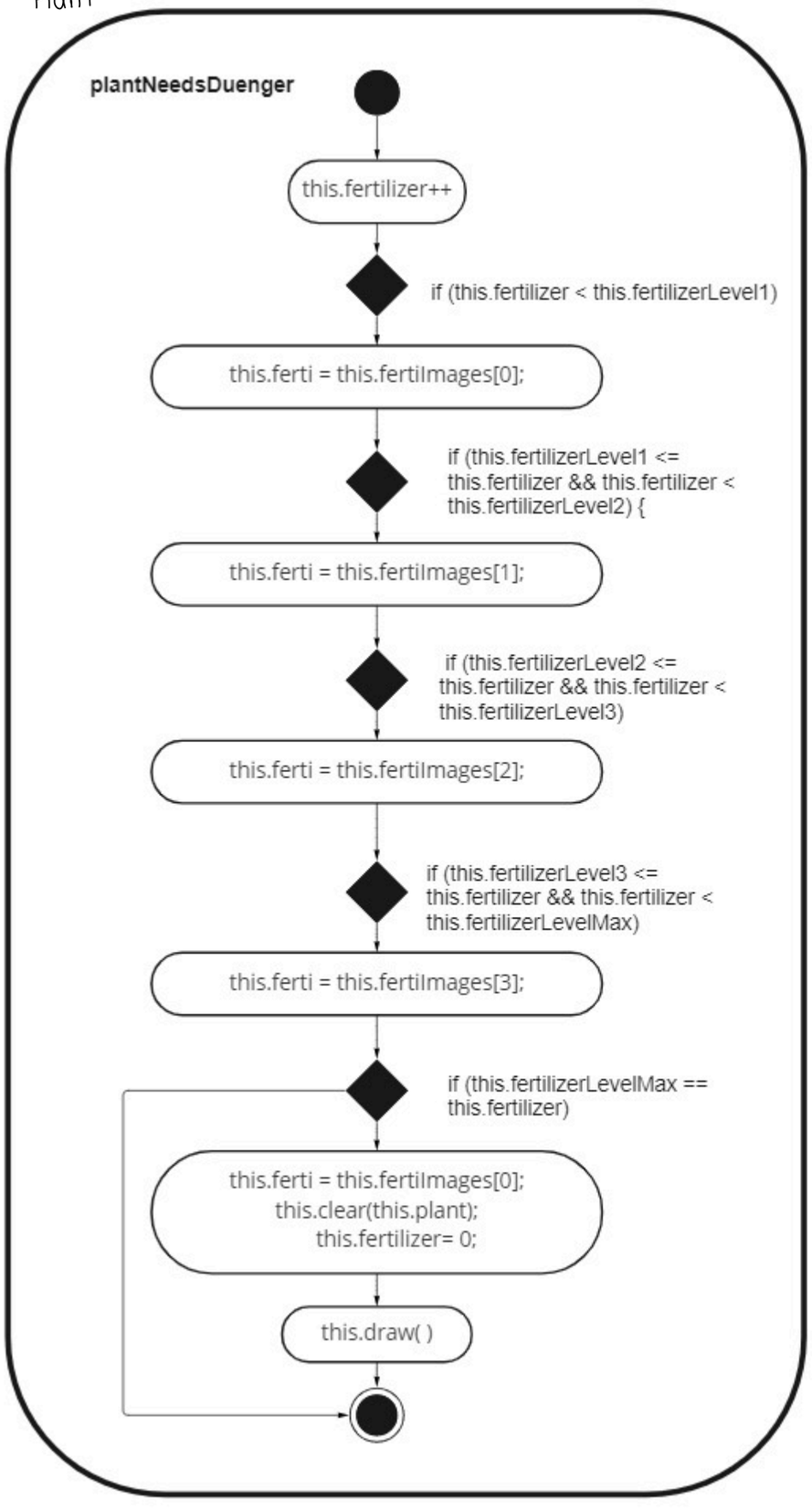
```
static image: HTMLImageElement;
static price: number;
static priceNeu: number;

static seedamount: number;
verkaufPrice: number;
type: string;

images: HTMLImageElement [];
image: HTMLImageElement;
waterImages: HTMLImageElement [] = [Load.empty,
Load.dropS, Load.dropM, Load.dropB];
waterImage: HTMLImageElement;
fertiImages: HTMLImageElement [] = [Load.empty,
Load.fertilizS, Load.fertilizM, Load.fertilizB];
ferti: HTMLImageElement;

currentcolor: string;
row: number;
collum: number;
age: number = 1;
finalAge: number;
age1: number;
age2: number;
price: number;
needsWater: boolean = false;

bugs: Bug[] = [];
bug: Bug;

plant: Plant;

water: number = 0;
waterLevel1: number;
waterLevel2: number;
waterLevel3: number;
waterLevelMax: number;

fertilizer: number = 0;
fertilizerLevel1: number;
fertilizerLevel2: number;
fertilizerLevel3: number;
fertilizerLevelMax: number;
```

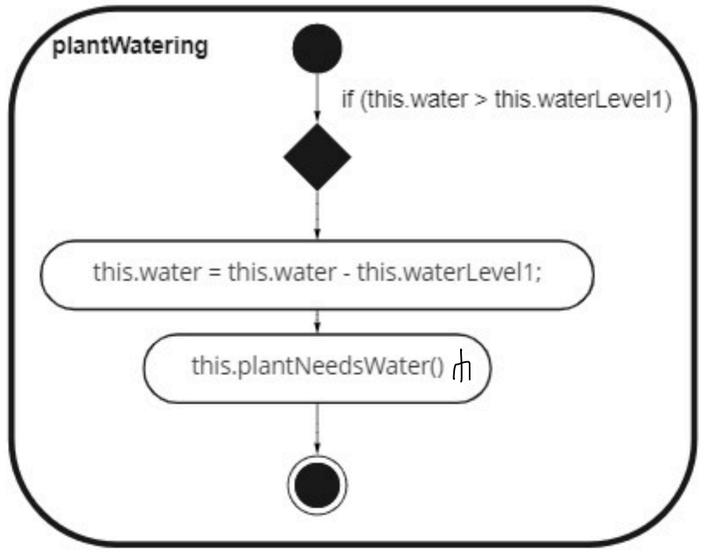


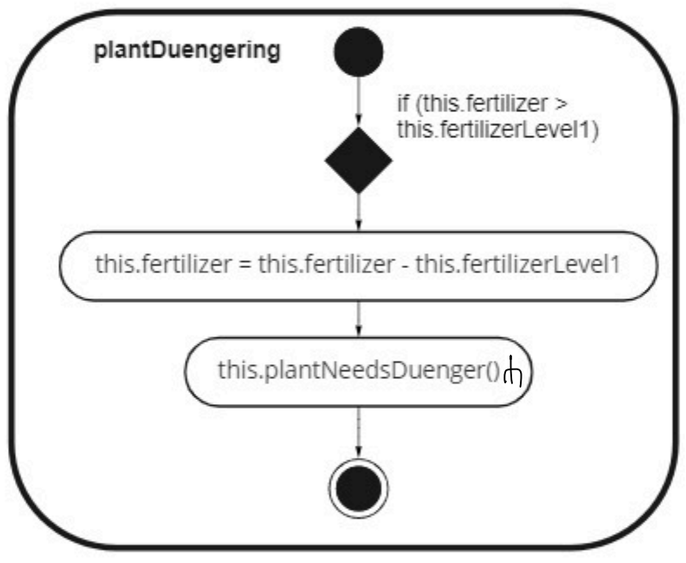


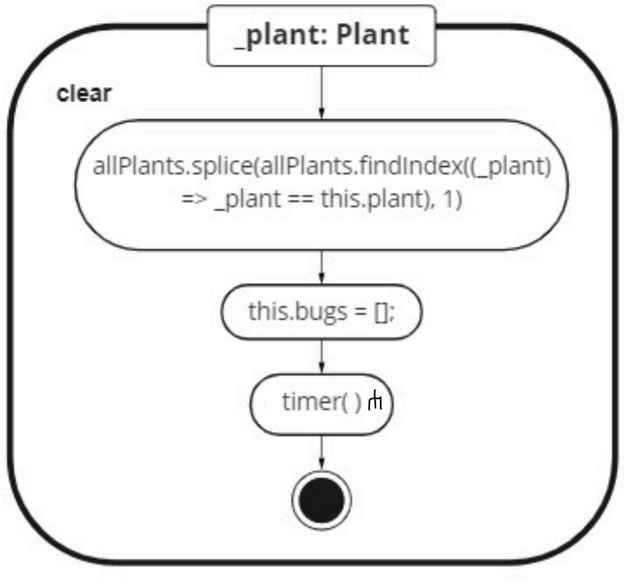


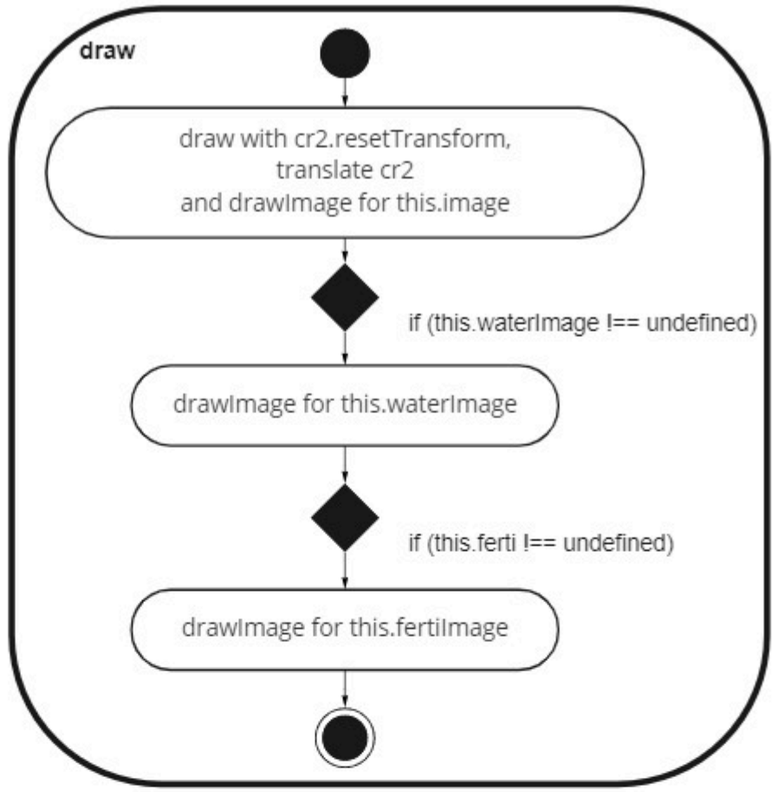
Plant

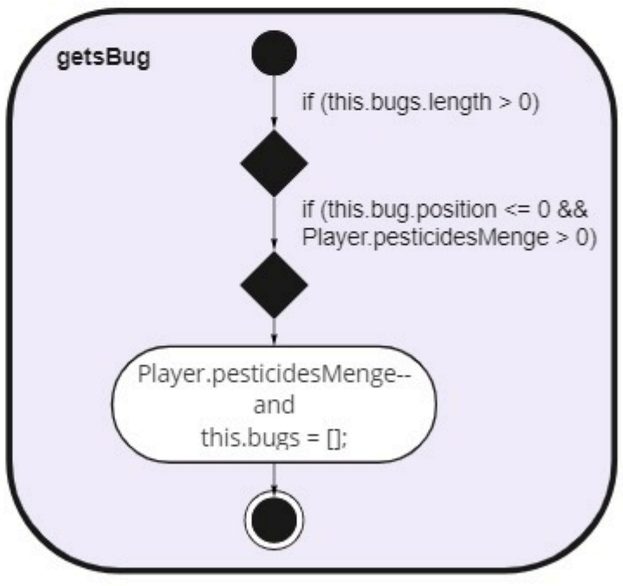

plantNeedsDuenger
this.fertilizer++
if (this.fertilizer < this.fertilizerLevel1)
this.ferti = this.fertiImages[0];
if (this.fertilizerLevel1 <= this.fertilizer && this.fertilizer < this.fertilizerLevel2) {
this.ferti = this.fertiImages[1];
if (this.fertilizerLevel2 <= this.fertilizer && this.fertilizer < this.fertilizerLevel3)
this.ferti = this.fertiImages[2];
if (this.fertilizerLevel3 <= this.fertilizer && this.fertilizer < this.fertilizerLevelMax)
this.ferti = this.fertiImages[3];
if (this.fertilizerLevelMax == this.fertilizer)
this.ferti = this.fertiImages[0];
this.clear(this.plant);
this.fertilizer= 0;
this.draw( )

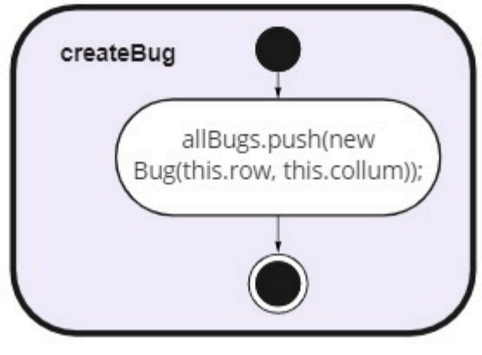

plantWatering
if (this.water > this.waterLevel1)
this.water = this.water - this.waterLevel1;
this.plantNeedsWater()


plantDuengering
if (this.fertilizer > this.fertilizerLevel1)
this.fertilizer = this.fertilizer - this.fertilizerLevel1
this.plantNeedsDuenger()


_plant: Plant
clear
allPlants.splice(allPlants.findIndex((_plant) => _plant == this.plant), 1)
this.bugs = [];
timer( )


draw
draw with cr2.resetTransform, translate cr2 and drawImage for this.image
if (this.waterImage !== undefined)
drawImage for this.waterImage
if (this.ferti !== undefined)
drawImage for this.fertiImage


getsBug
if (this.bugs.length > 0)
if (this.bug.position <= 0 && Player.pesticidesMenge > 0)
Player.pesticidesMenge-- and this.bugs = [];


createBug
allBugs.push(new Bug(this.row, this.collum));

Player

static ananasS: HTMLImageElement;
static ananasM: HTMLImageElement;
static ananasB: HTMLImageElement;

static blueberryS: HTMLImageElement;
static blueberryM: HTMLImageElement;
static blueberryB: HTMLImageElement;

static melonS: HTMLImageElement;
static melonM: HTMLImageElement;
static melonB: HTMLImageElement;

static raspberryS: HTMLImageElement;
static raspberryM: HTMLImageElement;
static raspberryB: HTMLImageElement;

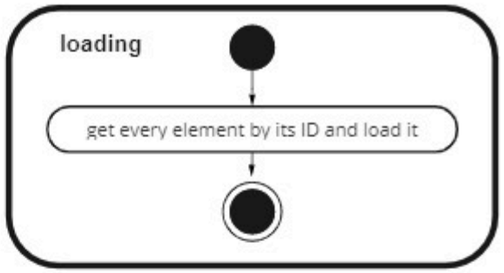
static strawberryS: HTMLImageElement;
static strawberryM: HTMLImageElement;
static strawberryB: HTMLImageElement;

static dropS: HTMLImageElement;
static dropM: HTMLImageElement;
static dropB: HTMLImageElement;

static fertilizS: HTMLImageElement;
static fertilizM: HTMLImageElement;
static fertilizB: HTMLImageElement;

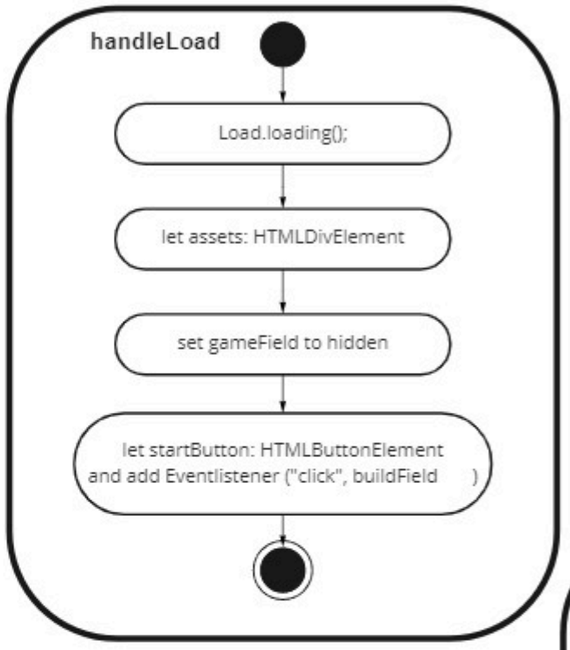
static bug: HTMLImageElement;

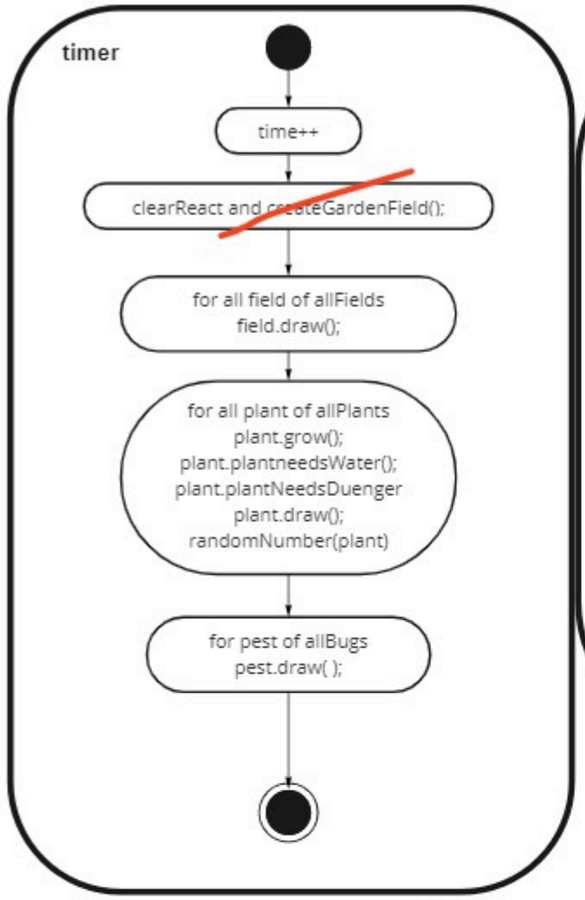
static empty: HTMLImageElement;

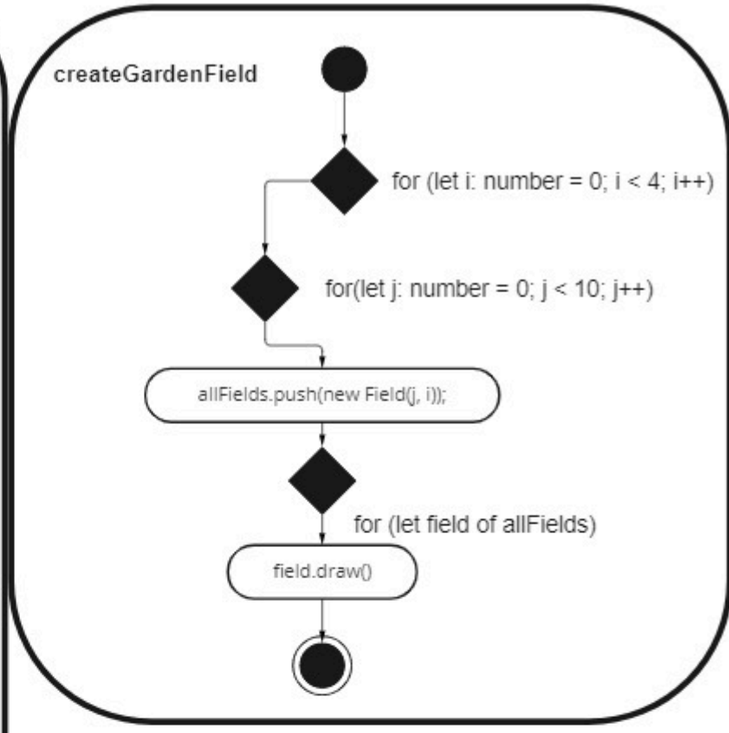


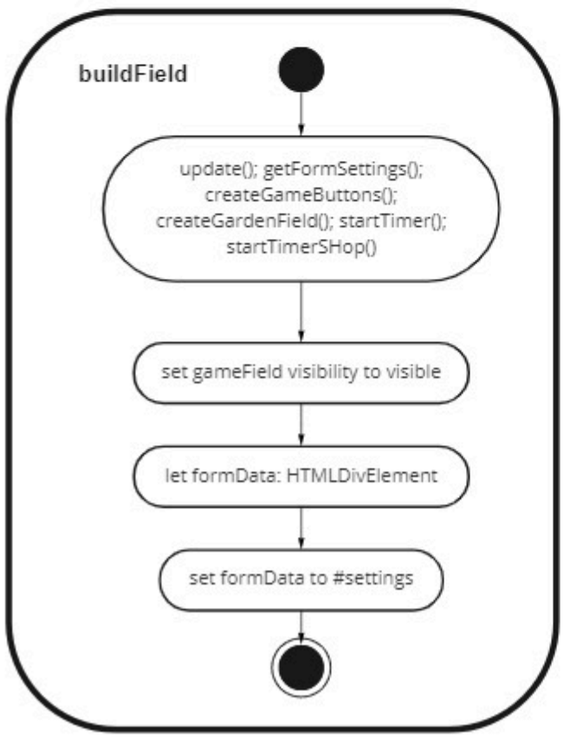
Main

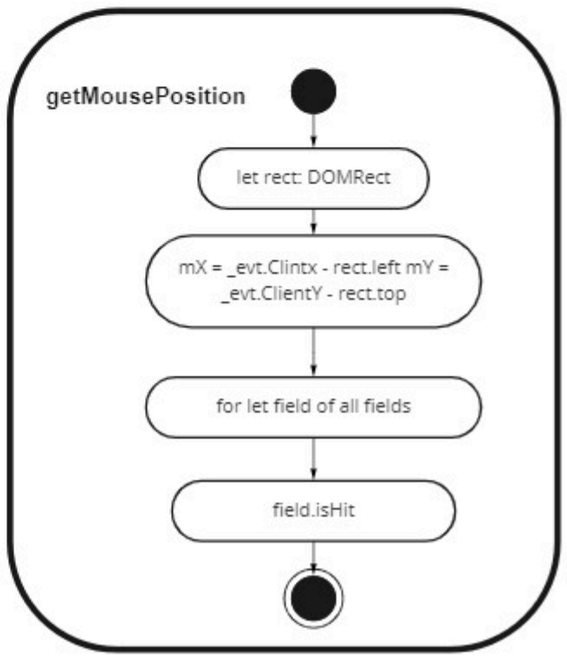
```
export let canvas: HTMLCanvasElement;
export let cr2: CanvasRenderingContext2D;
export let allFields: Field[] = [];
export let allPlants: Plant[] = [];
export let player: Player = new Player();
export let mX: number;
export let mY: number;
export let time: number = 0;
export let animationTime: number = 0;
export let allBugs: Bug[] = [];
let gameField: HTMLDivElement;

let formValues: FormData;
export let money: number;
let moneyChange: number;

export let fertilizerAmount: number = 0;
let fertilizerPrice: number = 10;
let fertilizerPriceNew: number;

export let pesticideAmount: number = 0;
let pesticidePriceNew: number;
let pestizidePrice: number = 5;
```

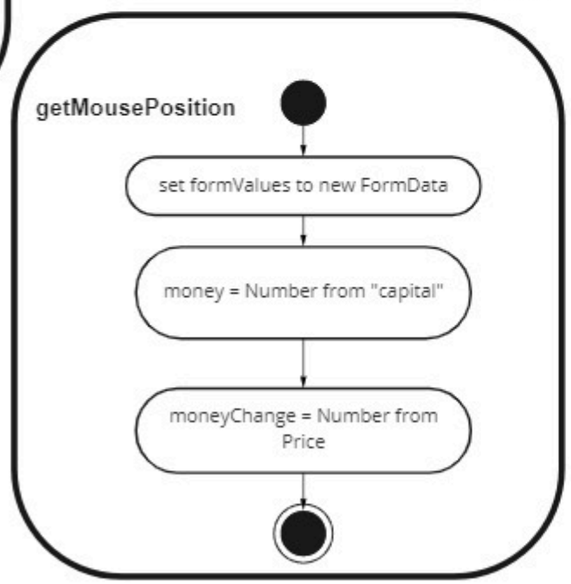


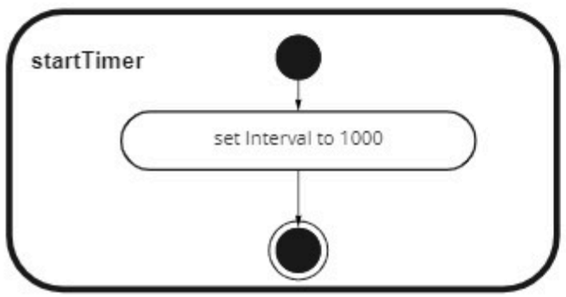


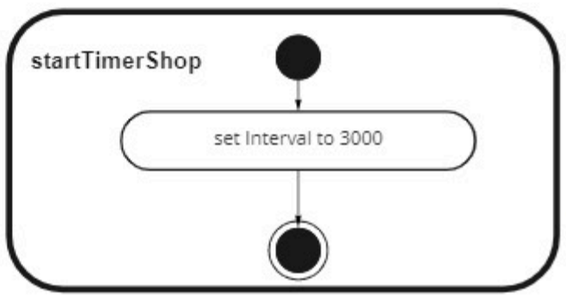




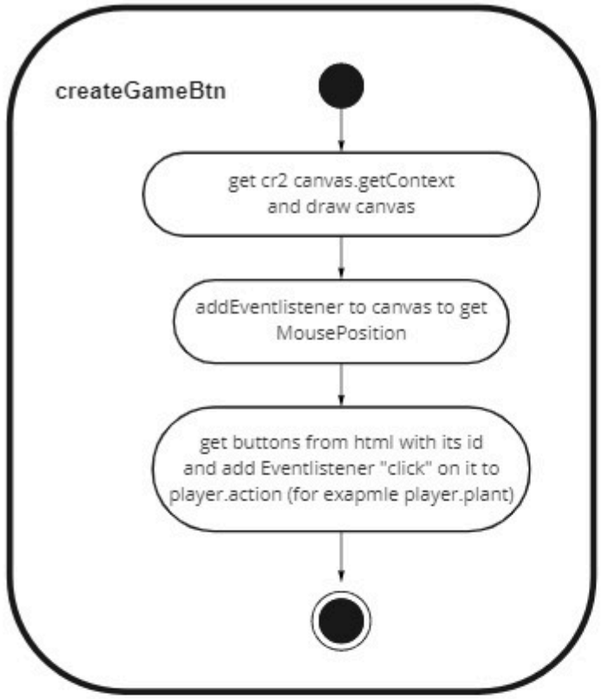


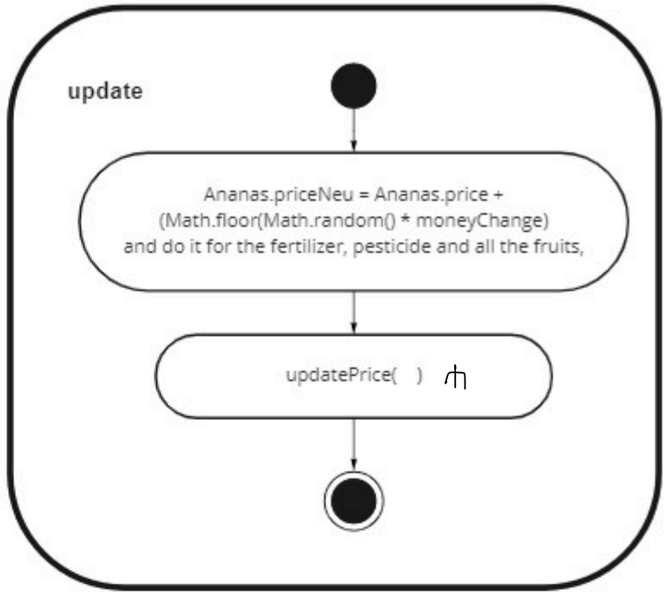


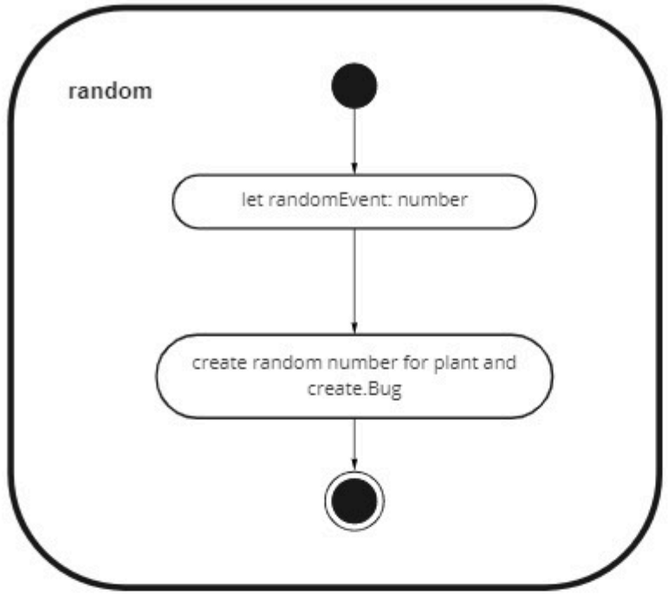


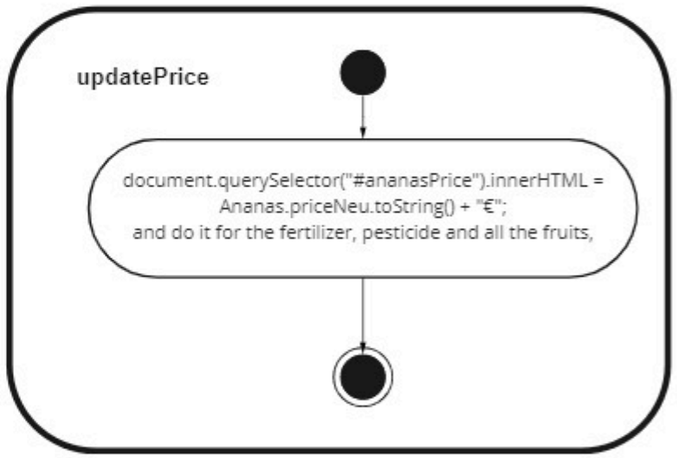


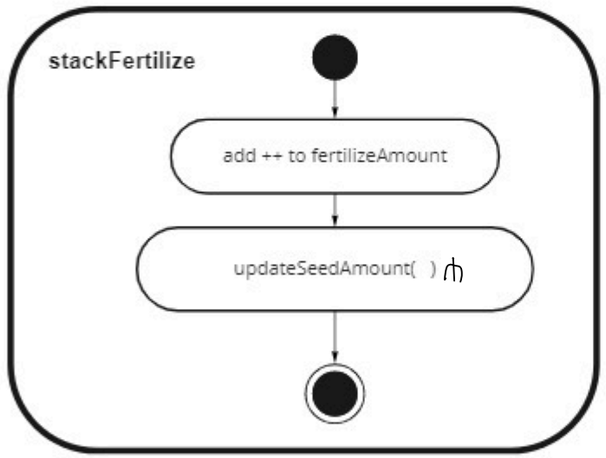
Main

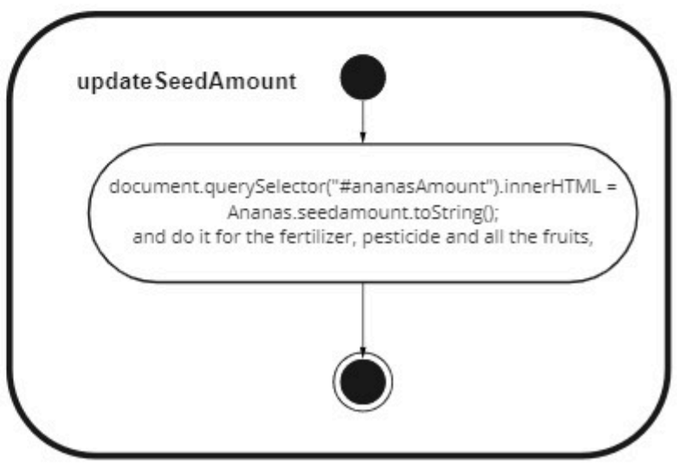

createGameBtn
get cr2 canvas.getContext
and draw canvas
addEventlistener to canvas to get
MousePosition
get buttons from html with its id
and add Eventlistener "click" on it to
player.action (for exapmle player.plant)

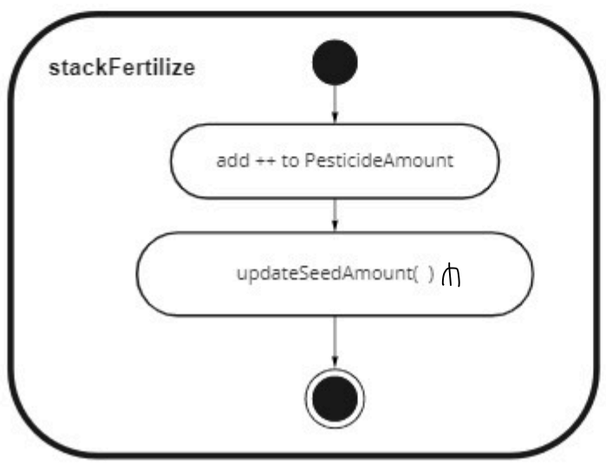

update
Ananas.priceNeu = Ananas.price +
(Math.floor(Math.random() * moneyChange)
and do it for the fertilizer, pesticide and all the fruits,
updatePrice( )


random
let randomEvent: number
create random number for plant and
create.Bug


updatePrice
document.querySelector("#ananasPrice").innerHTML =
Ananas.priceNeu.toString() + "€";
and do it for the fertilizer, pesticide and all the fruits,

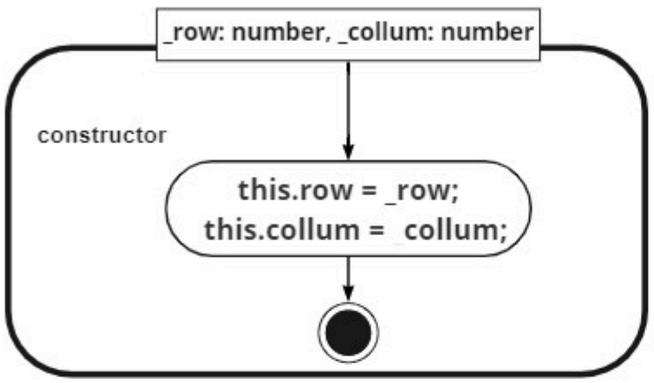

stackFertilize
add ++ to fertilizeAmount
updateSeedAmount( )

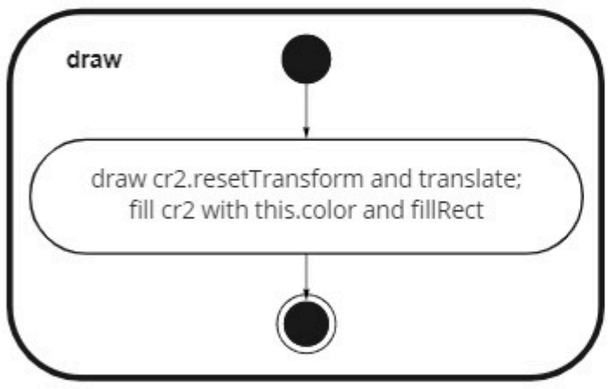

updateSeedAmount
document.querySelector("#ananasAmount").innerHTML =
Ananas.seedamount.toString();
and do it for the fertilizer, pesticide and all the fruits,

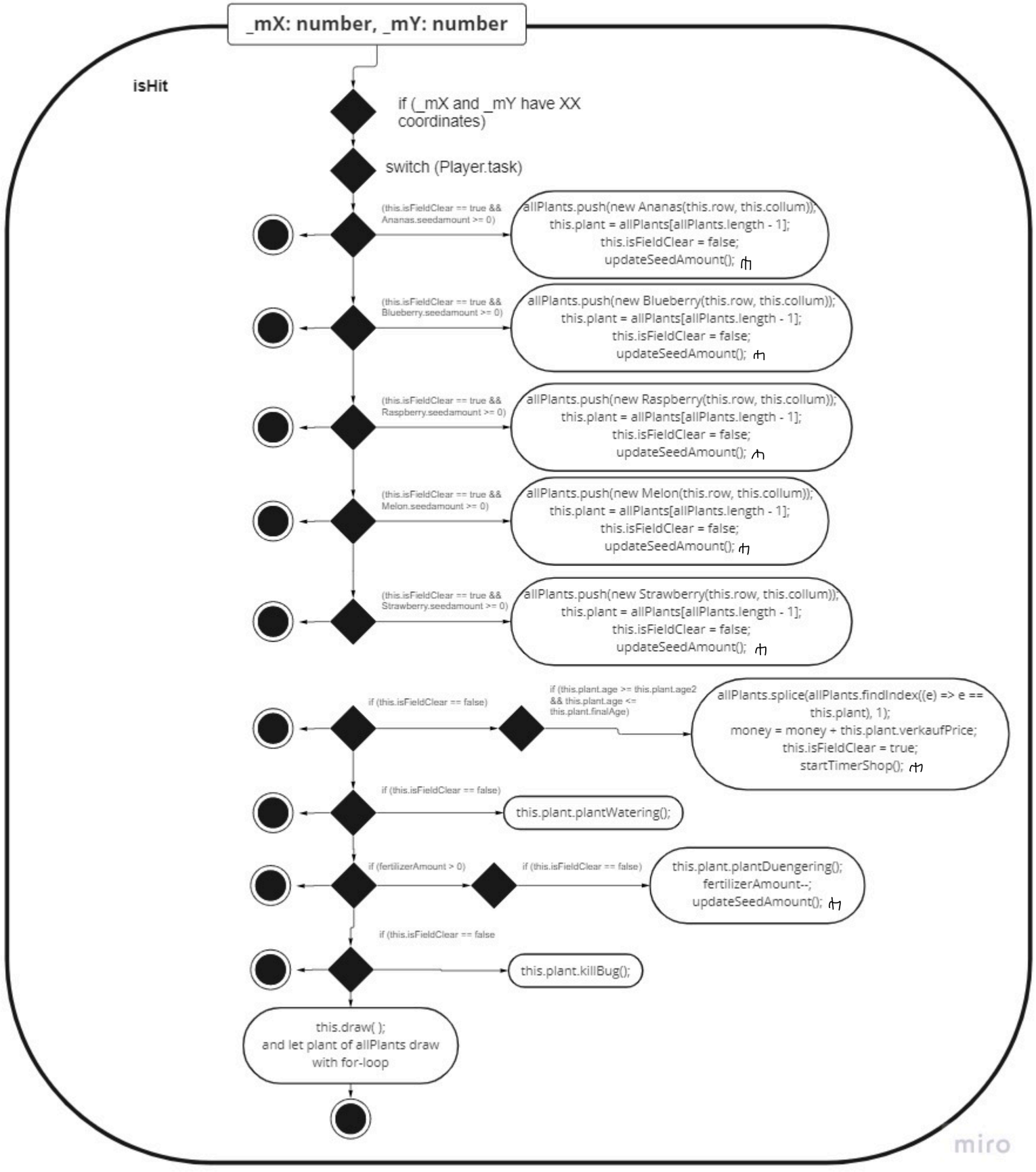

stackFertilize
add ++ to PesticideAmount
updateSeedAmount( )

Field

row: number;
collum: number;
color: string = "tan";
isFieldClear: boolean = true;
plant: Plant;

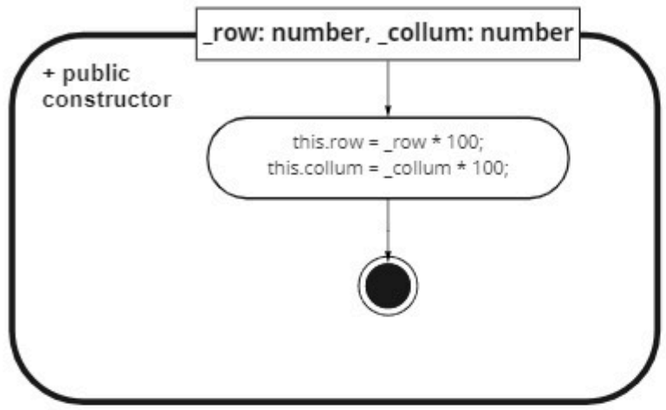


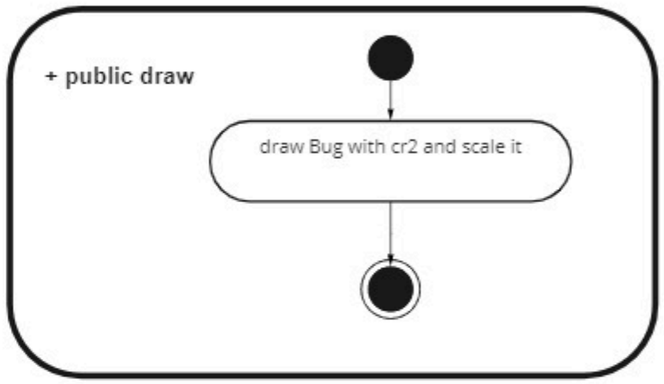


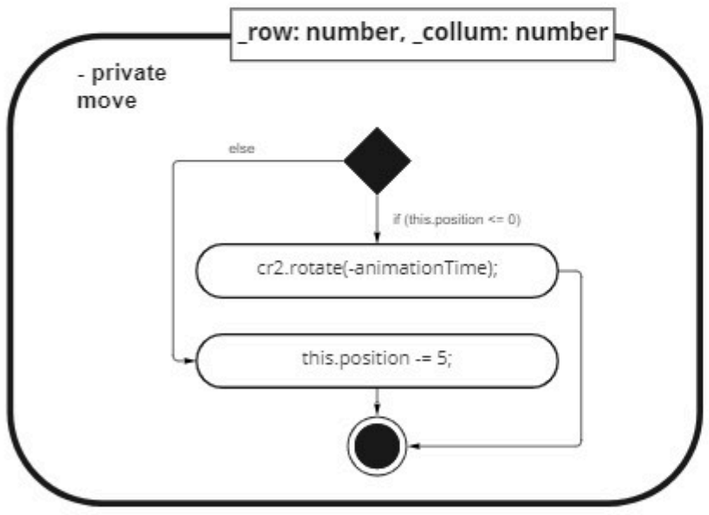


Bug

```
public position: number = 400;
private row: number;
private collum: number;
```





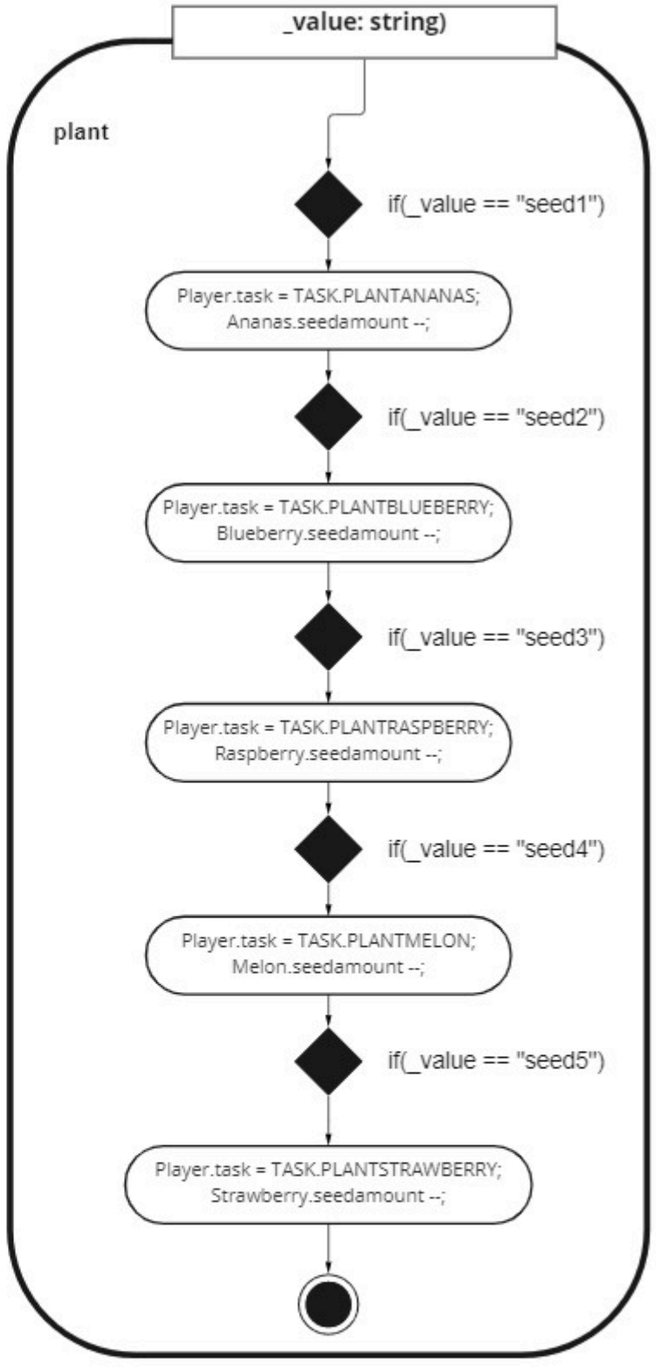


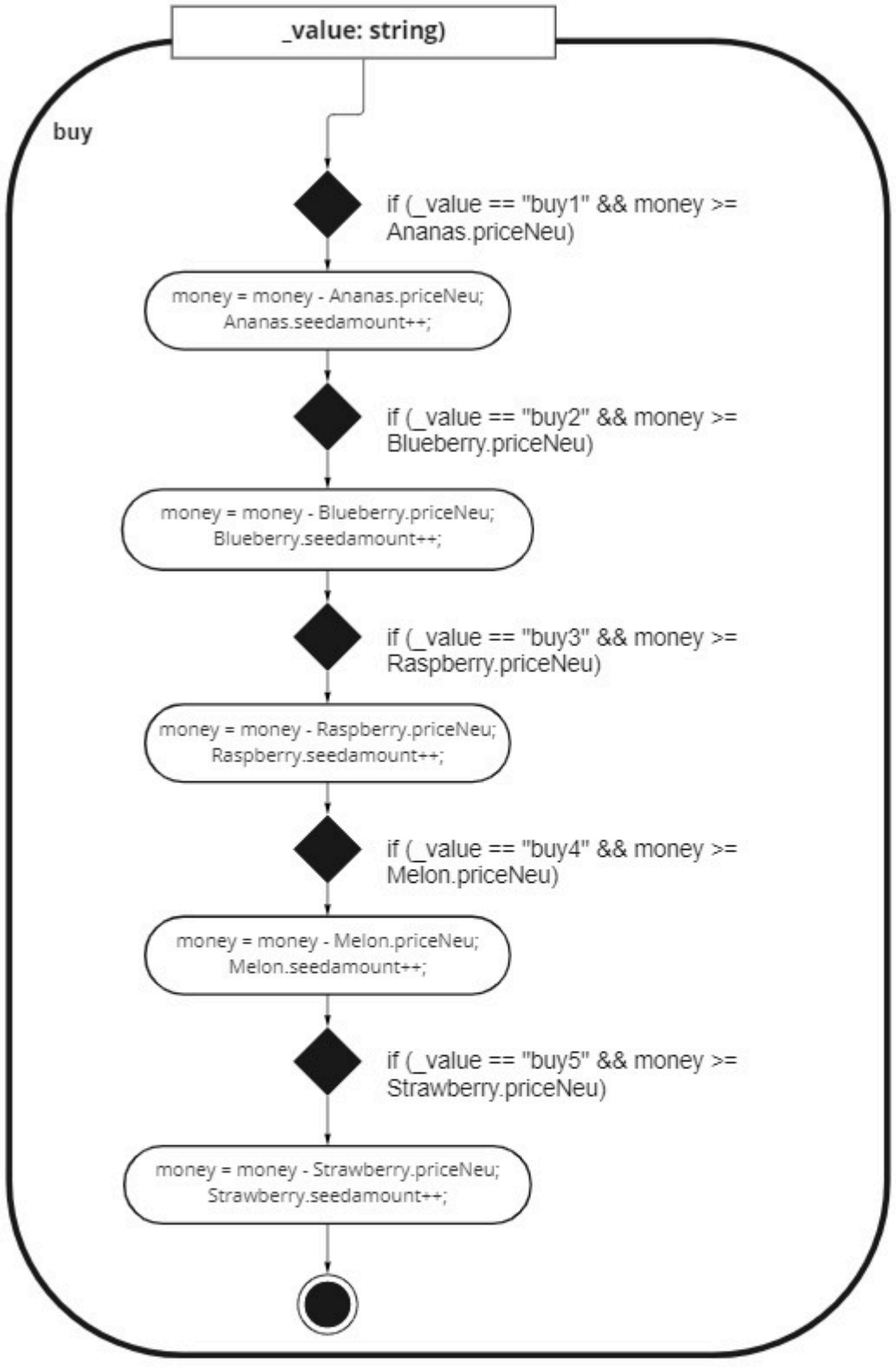
Player

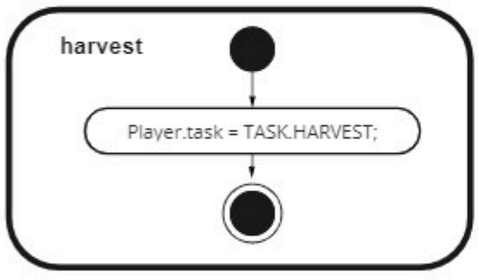
```
static task: TASK;
static pesticideAmount: number = 20;
static fertilizerAmount: number = 20;
static fertilizerPrice: number = 5;
static pesticidesPrice: number = 5;
static fertilizerPriceNew: number;
static pesticidePriceNew: number;
```

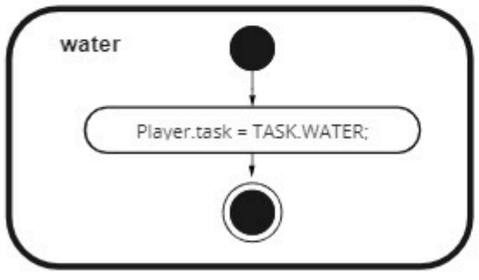
<<enum>>
TASK

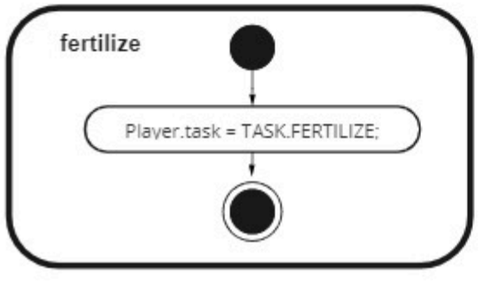
PLANTANANAS,
PLANTBLUEBERRY,
PLANTRASPBERRY,
PLANTMELON,
PLANTSTRAWBERRY,
HARVEST,
WATER,
FERTILIZE,
PESTICIDE

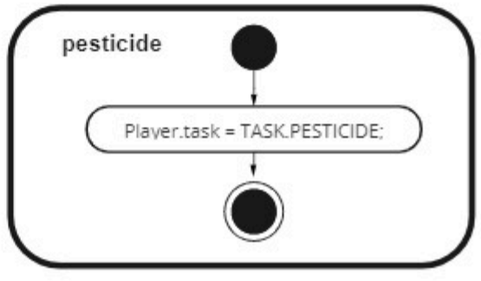












Ananas

extends Plant

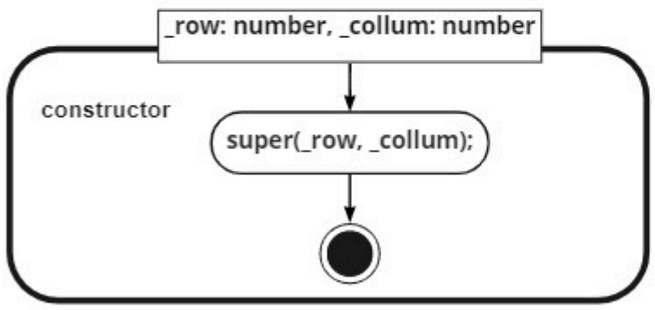
static seedamount: number = 0;
static price: number = 7;
verkaufPrice: number = 15;
priceNeu: number;

images: HTMLImageElement[] = [Load.ananasS, Load.ananasM, Load.ananasB];
image: HTMLImageElement = this.images[0];
type: string = "seed1";
row: number = this.row;
collum: number = this.collum;

age1: number = 10;
age2: number = 20;
finalAge: number = 30;

waterLevel1: number = 5;
waterLevel2: number = 10;
waterLevel3: number = 15;
waterLevelMax: number = 20;

fertilizerLevel1: number = 5;
fertilizerLevel2: number = 10;
fertilizerLevel3: number = 15;
fertilizerLevelMax: number = 20;



Blueberry

extends Plant

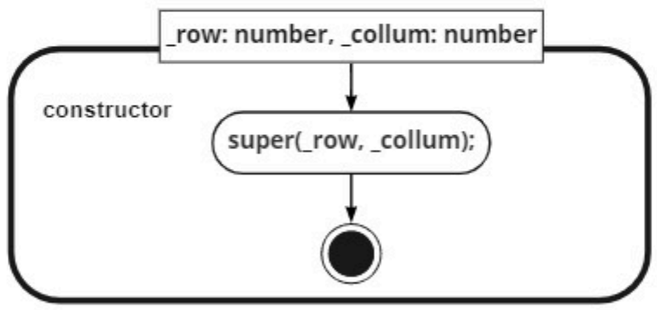
static seedamount: number = 0;
static price: number = 3;
verkaufPrice: number = 7;
priceNeu: number;

images: HTMLImageElement[] = [Load.blueberryS, Load.blueberryM, Load.blueberryB];
image: HTMLImageElement = this.images[0];
type: string = "seed2";
row: number = this.row;
collum: number = this.collum;

finalAge: number = 30;
age1: number = 10;
age2: number = 22;

waterLevel1: number = 5;
waterLevel2: number = 10;
waterLevel3: number = 15;
waterLevelMax: number = 20;

fertilizerLevel1: number = 9;
fertilizerLevel2: number = 16;
fertilizerLevel3: number = 23;
fertilizerLevelMax: number = 30;



Melon

extends Plant

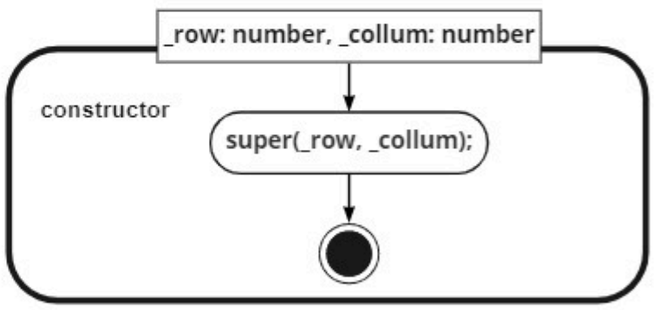
static seedamount: number = 0;
static price: number = 10;
verkaufPrice: number = 18;
priceNeu: number;

images: HTMLImageElement[] = [Load.melonS, Load.melonM, Load.melonB];
image: HTMLImageElement = this.images[0];
type: string = "seed4";
row: number = this.row;
collum: number = this.collum;

age1: number = 18;
age2: number = 30;
finalAge: number = 40;

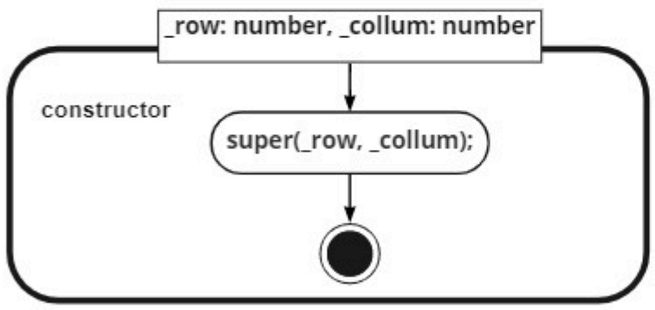
waterLevel1: number = 14;
waterLevel2: number = 20;
waterLevel3: number = 26;
waterLevelMax: number = 32;

fertilizerLevel1: number = 8;
fertilizerLevel2: number = 12;
fertilizerLevel3: number = 16;
fertilizerLevelMax: number = 20;



Raspberry extends Plant

```
static seedamount: number = 0;
static price: number = 2;
verkaufPrice: number = 5;
priceNeu: number;

images: HTMLImageElement[] = [Load.raspberryS
Load.raspberryM, Load.raspberryB];
image: HTMLImageElement = this.images[0];
type: string = "seed3";
row: number = this.row;
collum: number = this.collum;

age1: number = 15;
age2: number = 29;
finalAge: number = 35;

waterLevel1: number = 5;
waterLevel2: number = 10;
waterLevel3: number = 15;
waterLevelMax: number = 20;

fertilizerLevel1: number = 3;
fertilizerLevel2: number = 8;
fertilizerLevel3: number = 13;
fertilizerLevelMax: number = 18;
```



Strawberry extends Plant

```
static seedamount: number = 0;
static price: number = 5;
verkaufPrice: number = 9;
priceNeu: number;

images: HTMLImageElement[] = [Load.strawberryS,
Load.strawberryM, Load.strawberryB];
image: HTMLImageElement = this.images[0];
type: string = "seed5";
row: number = this.row;
collum: number = this.collum;

age1: number = 10;
age2: number = 20;
finalAge: number = 30;

waterLevel1: number = 10;
waterLevel2: number = 14;
waterLevel3: number = 18;
waterLevelMax: number = 22;

fertilizerLevel1: number = 5;
fertilizerLevel2: number = 12;
fertilizerLevel3: number = 19;
fertilizerLevelMax: number = 26;
```