OBJECT ORIENTED ANALYSIS AND DESIGN

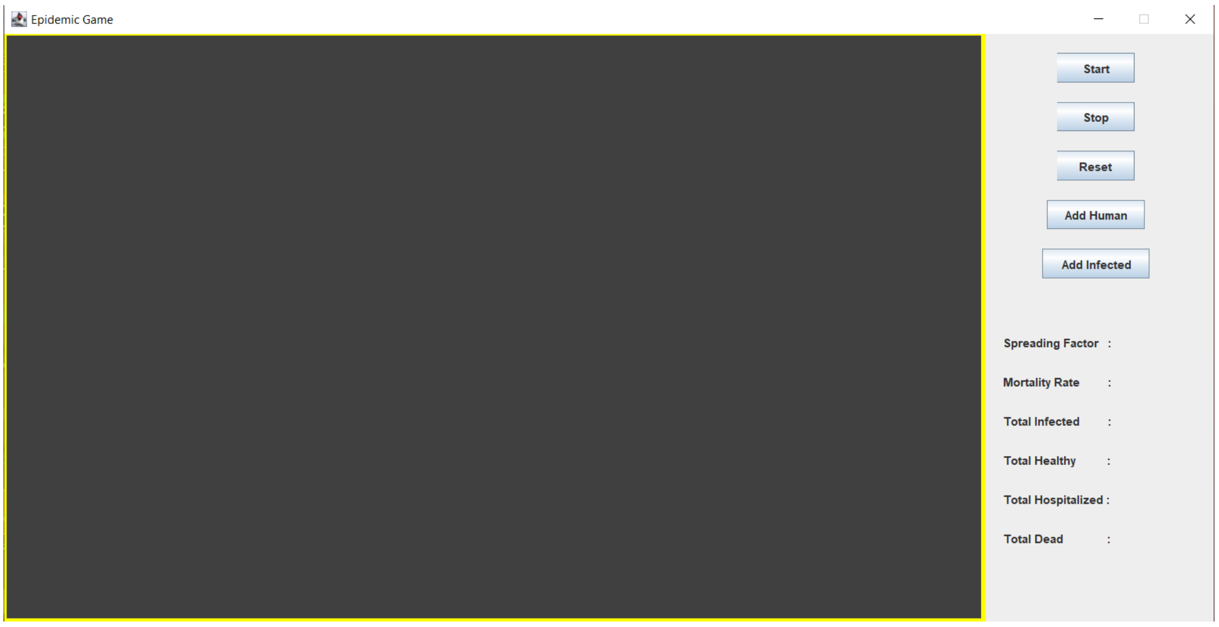
FINAL PROJECT REPORT

OĞUZHAN ŞENTÜRK

Program Explanation

1)The society will be modeled as an empty 2D canvas of size 1000x600 pixels.

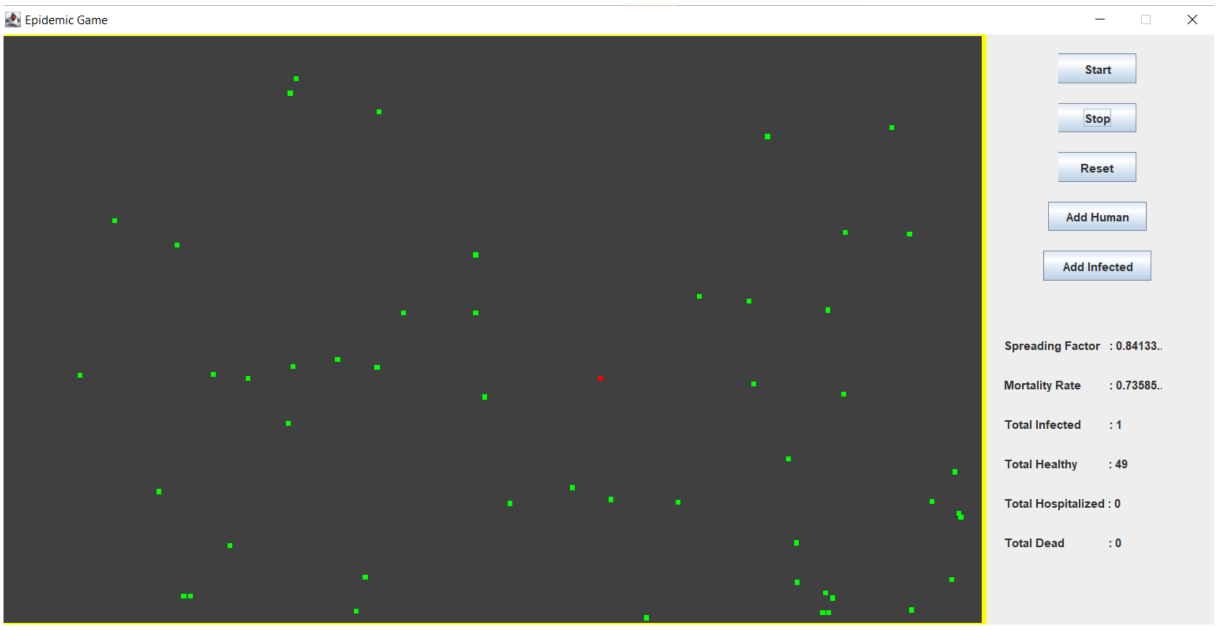
Society is 1000x600 pixels. Frame is 1265x600 pixels.



2)Each individual in this society of population Po will be modeled as a square of size 5x5 pixels on this canvas, positioned randomly.

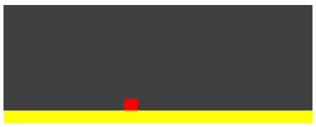
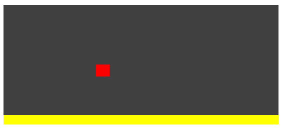
```
size = 5;
```

Model.java 56



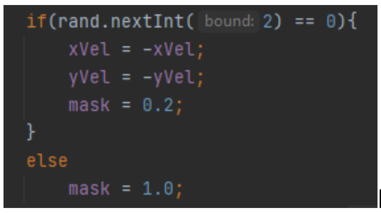
```
Spreading Factor  : 0.84133..
Mortality Rate    : 0.73585..
Total Infected    : 1
Total Healthy     : 49
Total Hospitalized : 0
Total Dead        : 0
```

3)if an individual reaches the edge of the map, her/his movement direction should be once again randomized.

Before :  After : 

4)At the beginning there will be one random infected individual in the population (See option 2)

5)Each individual will possess a numerical value indicating whether they wear a mask (M=0.2) or not (M=1.0)

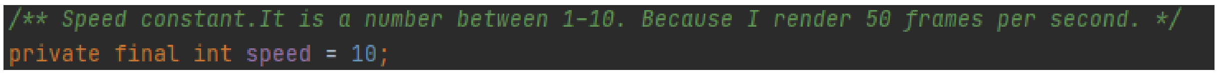


```
if(rand.nextInt( bound: 2) == 0){
    xVel = -xVel;
    yVel = -yVel;
    mask = 0.2;
}
else
    mask = 1.0;
```

Model.java 63

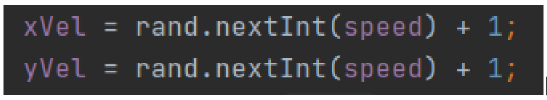
6)Their speed S \in [1,500] of movement in pixels/second,

Since it has to move between 1 and 500 pixels per second and to get more smoother view, I render 50 times in a second and call move functions 50 time. Therefore, I keep the speed between 1 and 10.



```
/** Speed constant.It is a number between 1-10. Because I render 50 frames per second. */
private final int speed = 10;
```

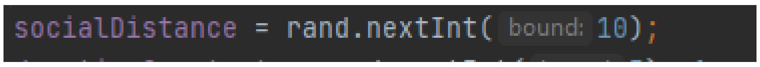
Model.java 49



```
xVel = rand.nextInt(speed) + 1;
yVel = rand.nextInt(speed) + 1;
```

Model.java 61

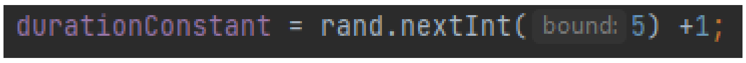
7)The social distance D \in [0,9] (in pixels) that they practice when they collide with other individuals



```
socialDistance = rand.nextInt( bound: 10);
```

Model.java 70

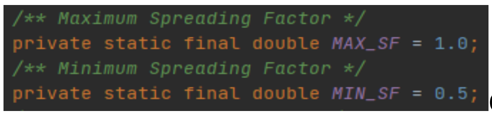
8)How social they are in terms of C seconds \in [1,5] they spend with every individual they collide with.



```
durationConstant = rand.nextInt( bound: 5) +1;
```

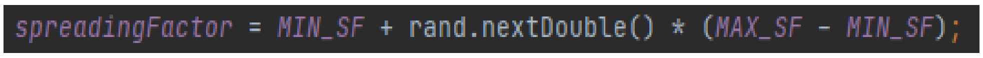
Model.java 71

9)The disease will have a constant spreading factor R \in [0.5,1.0]



```
/** Maximum Spreading Factor */
private static final double MAX_SF = 1.0;
/** Minimum Spreading Factor */
private static final double MIN_SF = 0.5;
```

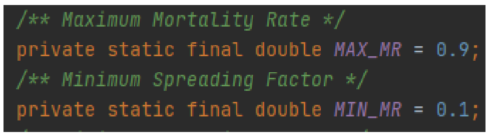
Controller.java 15



```
spreadingFactor = MIN_SF + rand.nextDouble() * (MAX_SF - MIN_SF);
```

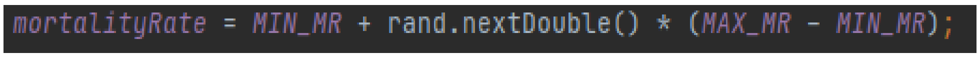
Controller.java 47

10)Constant mortality rate Z \in [0.1, 0.9].



```
/** Maximum Mortality Rate */
private static final double MAX_MR = 0.9;
/** Minimum Spreading Factor */
private static final double MIN_MR = 0.1;
```

Controller.java 19



```
mortalityRate = MIN_MR + rand.nextDouble() * (MAX_MR - MIN_MR);
```

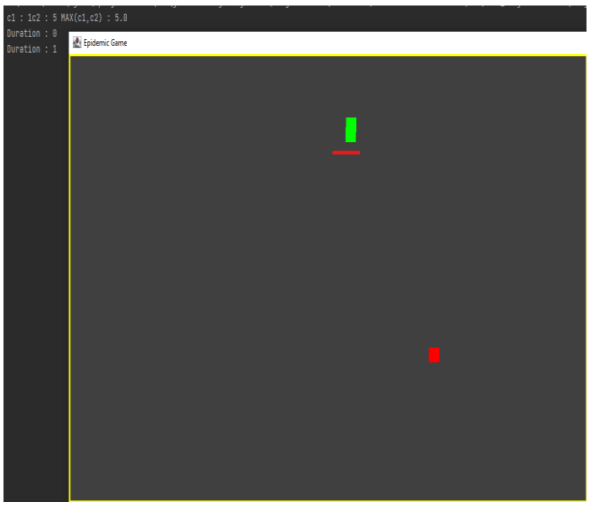
Controller.java 48

11)When two individuals with coefficients C_1 and C_2 collide they stay together (at collision position) for time C=max{C_1,C_2} to simulate interaction and then continue their randomized courses

In order to demonstrate it better, I wrote the print condition to inside mediator class, and after the collision, I wrote the print condition again in the timer class inside the mediator class to show that they stay together dynamically for the minimum time determined by the mediator class.

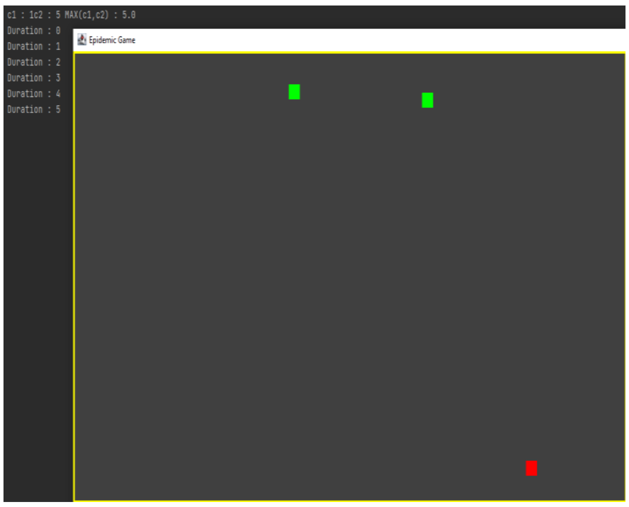
I changed the size of person for your better view.

Before:



```
c1 : 1c2 : 5 MAX(c1,c2) : 5.0
Duration : 0
Duration : 1
```

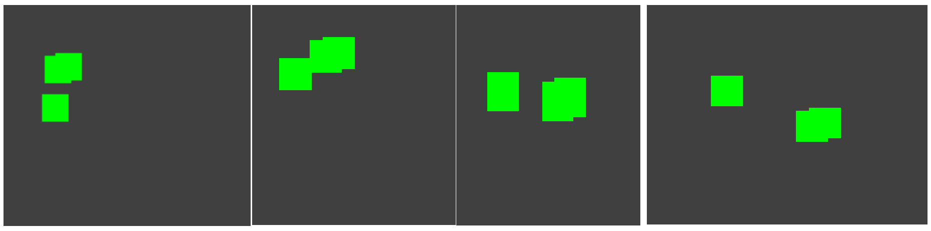
After:



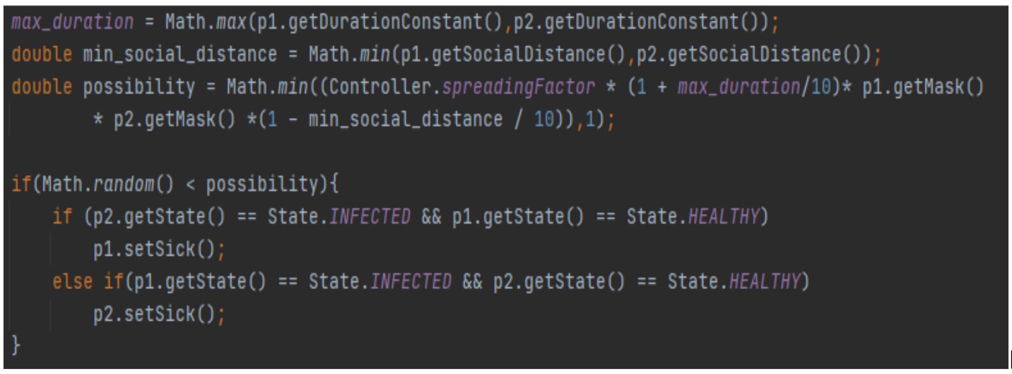
```
c1 : 1c2 : 5 MAX(c1,c2) : 5.0
Duration : 0
Duration : 1
Duration : 2
Duration : 3
Duration : 4
Duration : 5
```

12)If another individual is in collision course with either of them in the meantime, s/he cannot interact with them and ignores them as if they weren't there.

I put these screenshots in order, although it passes from a distance that should normally be collision, if it collision with someone next to it, it does not enter into collision



13)The probability of I_1 infecting I_2 is P = min(R * (1+C/10) * M_1 * M_2 * (1-D/10),1)

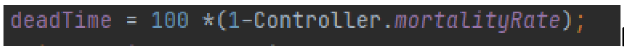


```java
max_duration = Math.max(p1.getDurationConstant(),p2.getDurationConstant());
double min_social_distance = Math.min(p1.getSocialDistance(),p2.getSocialDistance());
double possibility = Math.min((Controller.spreadingFactor * (1 + max_duration/10)* p1.getMask()
        * p2.getMask() *(1 - min_social_distance / 10)),1);

if(Math.random() < possibility){
    if (p2.getState() == State.INFECTED && p1.getState() == State.HEALTHY)
        p1.setSick();
    else if(p1.getState() == State.INFECTED && p2.getState() == State.HEALTHY)
        p2.setSick();
}
```

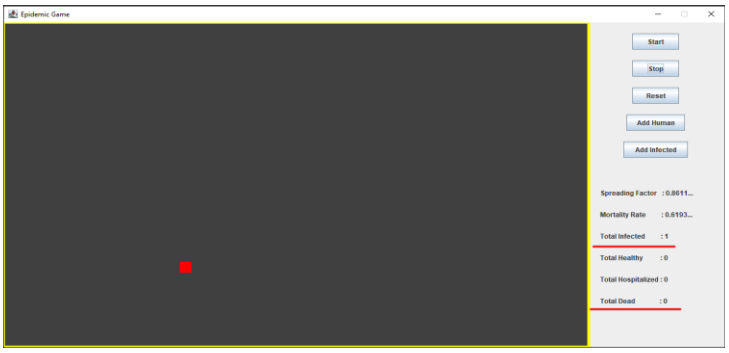
Mediator.java 37

14)An infected individual will die after 100 * (1-Z) seconds and disappear from the canvas.

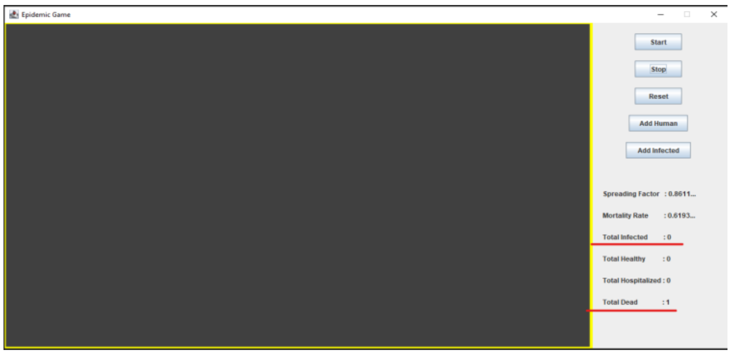


```java
deadTime = 100 *(1-Controller.mortalityRate);
```

Model.java 76



```
Spreading Factor   : 0.8611...
Mortality Rate     : 0.6193...
Total Infected     : 1
Total Healthy      : 0
Total Hospitalized : 0
Total Dead         : 0
```



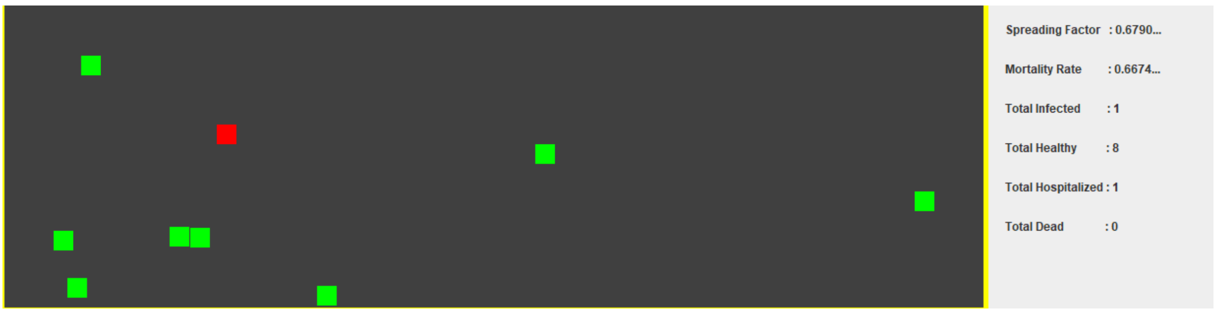
```
Spreading Factor   : 0.8611...
Mortality Rate     : 0.6193...
Total Infected     : 0
Total Healthy      : 0
Total Hospitalized : 0
Total Dead         : 1
```

15)Update the canvas every second or less; provide a timer,

I updated canvas 50 times per second.

16)Show the total count of infected, healthy, hospitalized and dead.

In order to demonstrate it better, I changed the size of people, added only 10 people and changed the hospital capacity (PO/10) this means 1. You see 1 infected people,8 healthy people and one person in the hospital.



```
Spreading Factor  : 0.6790...
Mortality Rate    : 0.6674...
Total Infected    : 1
Total Healthy     : 8
Total Hospitalized : 1
Total Dead        : 0
```

17)Every infected individual, 25 seconds after her/his initial infection will be assumed to be at the hospital and will be removed temporarily from the canvas.

```
recoveryTime = 25;
```
Model.java 74(See option 16)

18)The hospital however is assumed to have only B=Po/100 ventilators.

```
/** Minimum Spreading Factor */
private static final int PO_CONSTANT = 100;
```
Controller.java 23

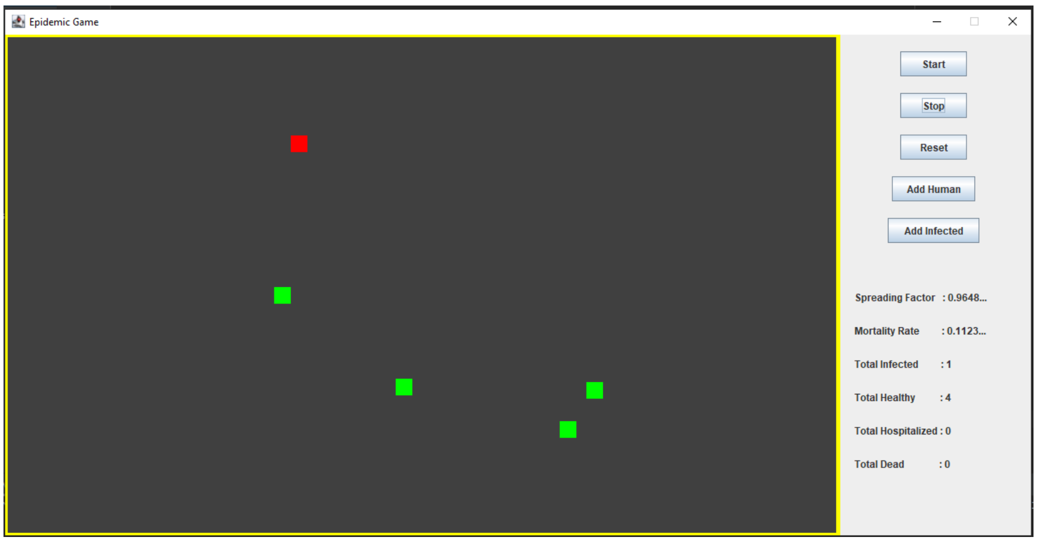
The hospital capacity is changed dynamically as people are added to the game.

```
hospital.setCapacity(modelList.size()/PO_CONSTANT);
```
Contoller.java 94

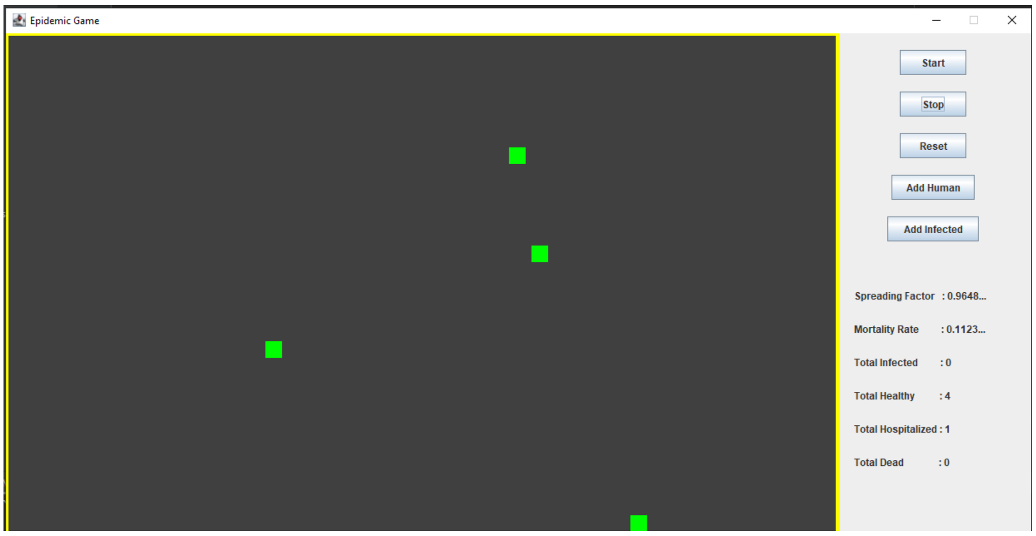
19)After staying at the hospital for 10 seconds s/ he will return to the society at a random position as healthy.

```
dischargeTime = 10;
```
Model.java 75

I changed the hospital capacity (PO/5) this means 1.

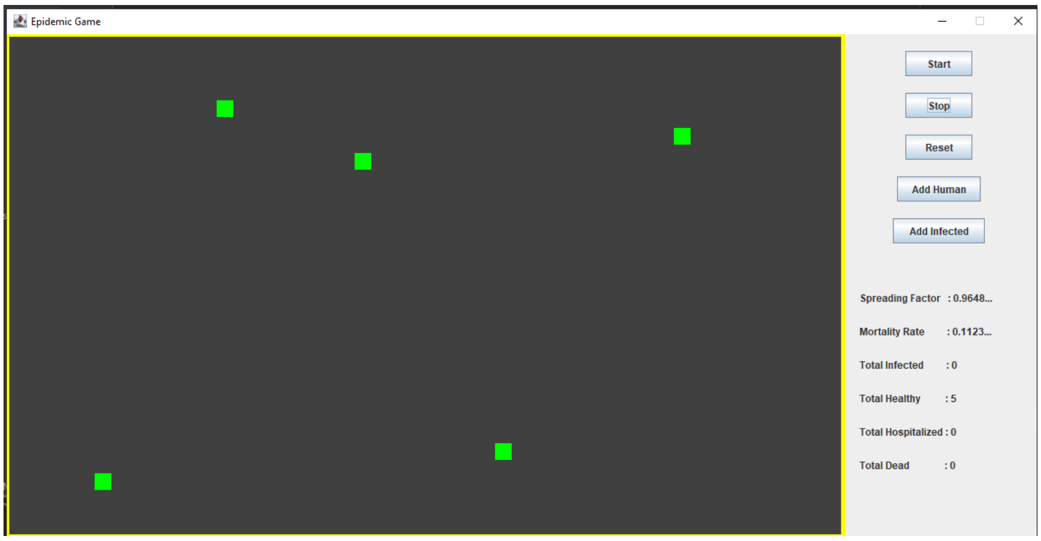


```
Spreading Factor  : 0.9648...
Mortality Rate    : 0.1123...
Total Infected    : 1
Total Healthy     : 4
Total Hospitalized : 0
Total Dead        : 0
```


Epidemic Game

Start
Stop
Reset
Add Human
Add Infected

Spreading Factor : 0.9648...
Mortality Rate : 0.1123...
Total Infected : 0
Total Healthy : 4
Total Hospitalized : 1
Total Dead : 0

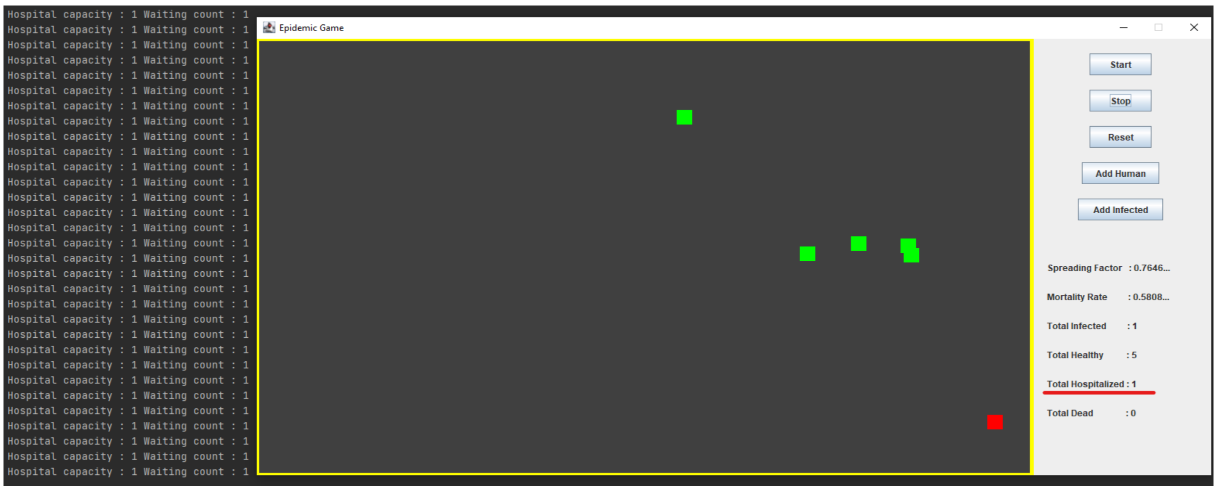

Epidemic Game

Start
Stop
Reset
Add Human
Add Infected

Spreading Factor : 0.9648...
Mortality Rate : 0.1123...
Total Infected : 0
Total Healthy : 5
Total Hospitalized : 0
Total Dead : 0

20)If the hospital ventilators are all full the individual will remain and continue moving/infecting in the society, until a ventilator becomes available or...s/he dies.

In order to demonstrate it better, I wrote the print condition to inside producer run method, you can see hospital capacity and number of waiting in queue from terminal. For now, 5 healthy people, one infected people and one hospitalized people. The infected person was put in queue. Because her/his 25-second time was over and as the hospital capacity is full.

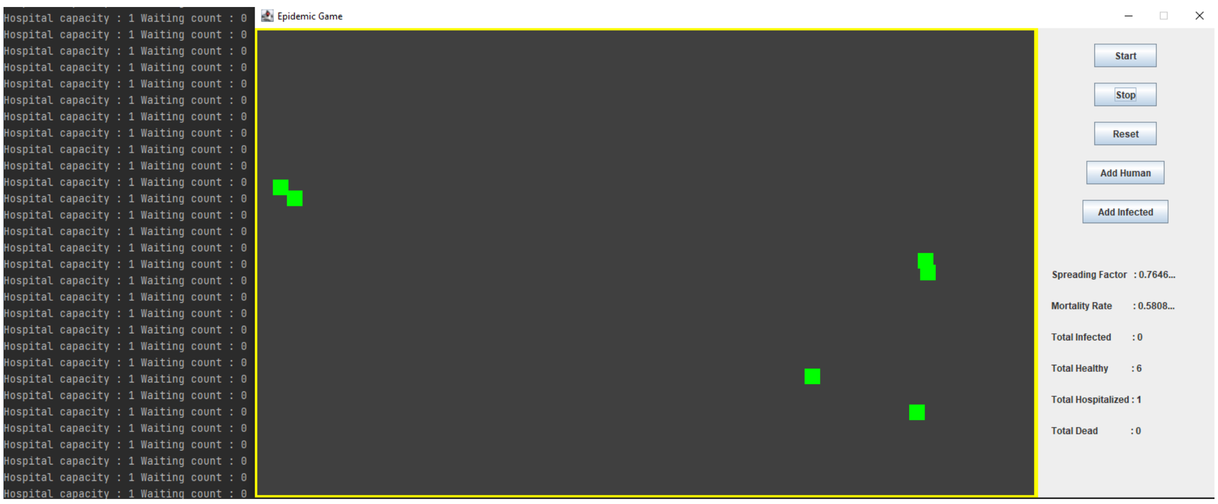

```
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
Hospital capacity : 1 Waiting count : 1
```

Epidemic Game

Start
Stop
Reset
Add Human
Add Infected

Spreading Factor : 0.7646...
Mortality Rate : 0.5808...
Total Infected : 1
Total Healthy : 5
Total Hospitalized : 1
Total Dead : 0

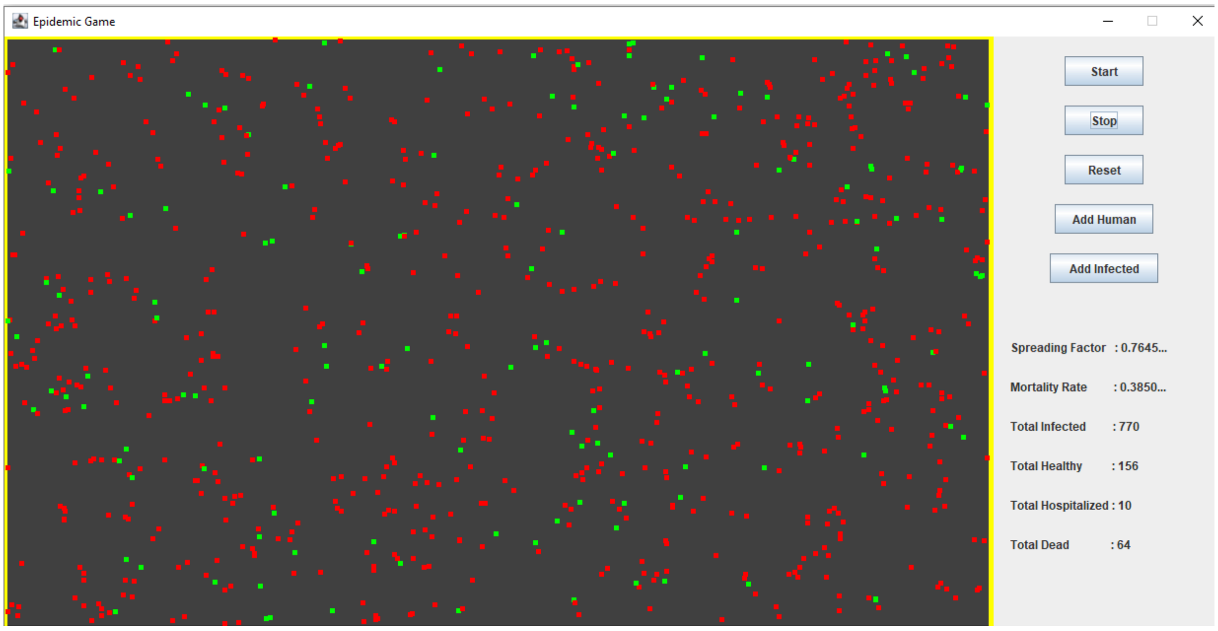
The infected people in the hospital was discharged and the waiting infected was taken to hospital. Because she/he did not die yet.



```
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
Hospital capacity : 1 Waiting count : 0
```

```
Spreading Factor  : 0.7646...
Mortality Rate    : 0.5808...
Total Infected    : 0
Total Healthy     : 6
Total Hospitalized : 1
Total Dead        : 0
```

Reel simulation (1000 person)



```
Spreading Factor  : 0.7645...
Mortality Rate    : 0.3850...
Total Infected    : 770
Total Healthy     : 156
Total Hospitalized : 10
Total Dead        : 64
```

Start button starts the game or resume the game.

Simulation starts after the human is added to the game.

Stop button stops the game.

Reset button starts new game.

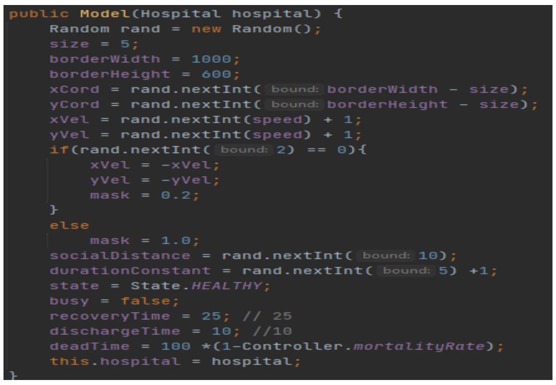
Add human button adds people to the game as many as entered.

Add infected button adds people to the game as many as entered.

You cannot add people while the game is stopped, or the game has not started yet.

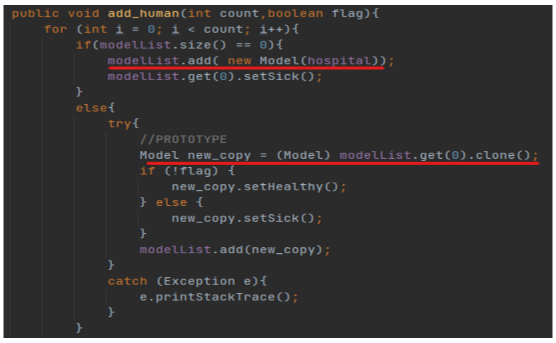
Design Decision

I used Prototype design pattern for creating individuals. User can add people and the people class are instantiated at runtime. Prototype design pattern reduces the need of sub-classing. Cost of creating People object is expensive and little bit complicated as you can see.



```java
public Model(Hospital hospital) {
    Random rand = new Random();
    size = 5;
    borderWidth = 1000;
    borderHeight = 600;
    xCord = rand.nextInt( bound: borderWidth - size);
    yCord = rand.nextInt( bound: borderHeight - size);
    xVel = rand.nextInt(speed) + 1;
    yVel = rand.nextInt(speed) + 1;
    if(rand.nextInt( bound: 2) == 0){
        xVel = -xVel;
        yVel = -yVel;
        mask = 0.2;
    }
    else
        mask = 1.0;
    socialDistance = rand.nextInt( bound: 10);
    durationConstant = rand.nextInt( bound: 5) +1;
    state = State.HEALTHY;
    busy = false;
    recoveryTime = 25; // 25
    dischargeTime = 10; //10
    deadTime = 100 *(1-Controller.mortalityRate);
    this.hospital = hospital;
}
```

Prototype design pattern hides complexities of creating objects, Class that creating object creates objects without know these complexities while creating object. This allows loose coupling. Prototype keep the number of classes in an application minimum. In my system, only one people class is created. All others are cloned.



```java
public void add_human(int count,boolean flag){
    for (int i = 0; i < count; i++){
        if(modelList.size() == 0){
            modelList.add( new Model(hospital));
            modelList.get(0).setSick();
        }
        else{
            try{
                //PROTOTYPE
                Model new_copy = (Model) modelList.get(0).clone();
                if (!flag) {
                    new_copy.setHealthy();
                } else {
                    new_copy.setSick();
                }
                modelList.add(new_copy);
            }
            catch (Exception e){
                e.printStackTrace();
            }
        }
```

I used MVC design architecture in my application.

Controller class controls the data flow into model and updates view whenever data changes. Controller exists between the view and model. Listens to events triggered by the view and decides what to do with that action.

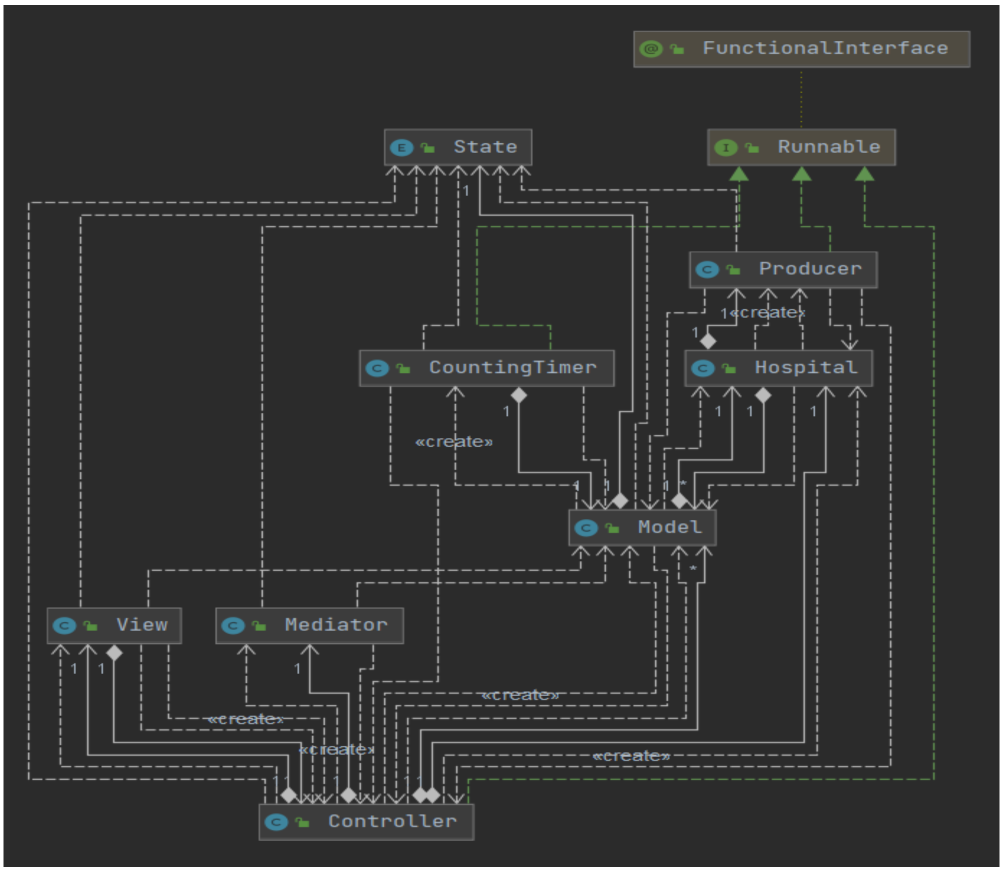
View class represent the visualization of the game and data that model contains

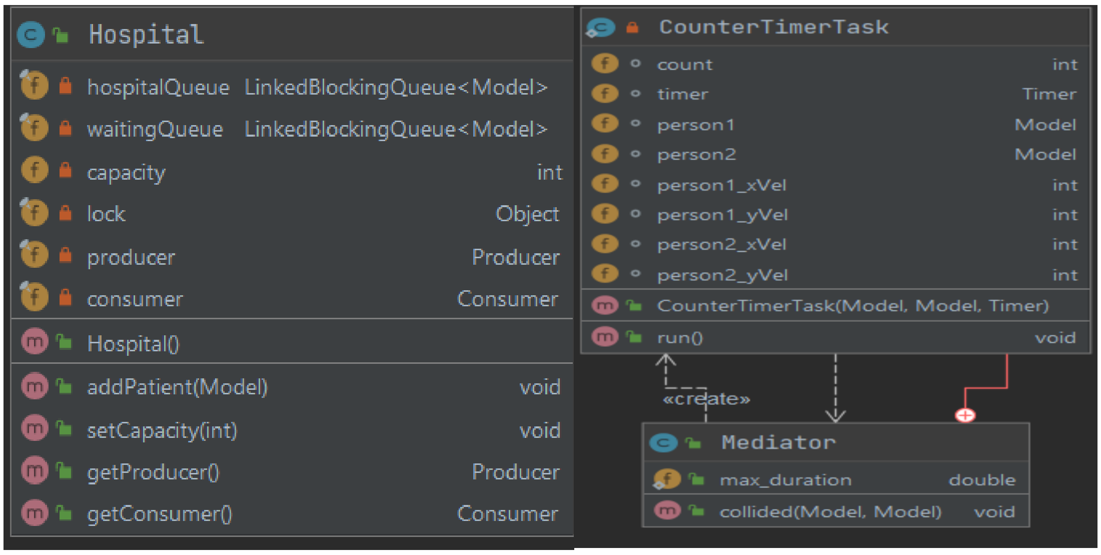
Model class holds data and represent data layer. Model receives user input from the controller.

I used mediator class for model and implement the interaction between individuals. Mediator class gave us a big loose coupling. You only just call black box, you have no information, is there any collision or not, if collision occurred what to do. The mediator does all the work for us you don't know anything. Mediator stops of two users if two users collided, control their time together, checks disease spreading and takes necessary actions etc.

I used producer/consumer paradigm to implement the hospital functionality. I keep two Blocking Queue for this. If infected individual 25 seconds time expires, she/he is immediately taken to the fist Queue. Producer check hospital's capacity is not full, and the consumer is not working and if there are people waiting in line it takes them. Consumer checks if there are people in second Queue and producer is not working, consumes Queue.

Class Diagram





Hospital

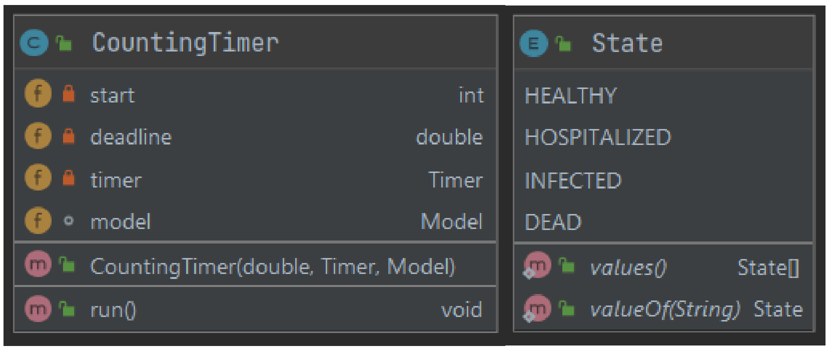
hospitalQueue	LinkedBlockingQueue<Model>
waitingQueue	LinkedBlockingQueue<Model>
capacity	int
lock	Object
producer	Producer
consumer	Consumer
Hospital()	
addPatient(Model)	void
setCapacity(int)	void
getProducer()	Producer
getConsumer()	Consumer

CounterTimerTask

count	int
timer	Timer
person1	Model
person2	Model
person1_xVel	int
person1_yVel	int
person2_xVel	int
person2_yVel	int
CounterTimerTask(Model, Model, Timer)	
run()	void

Mediator

max_duration	double
collided(Model, Model)	void

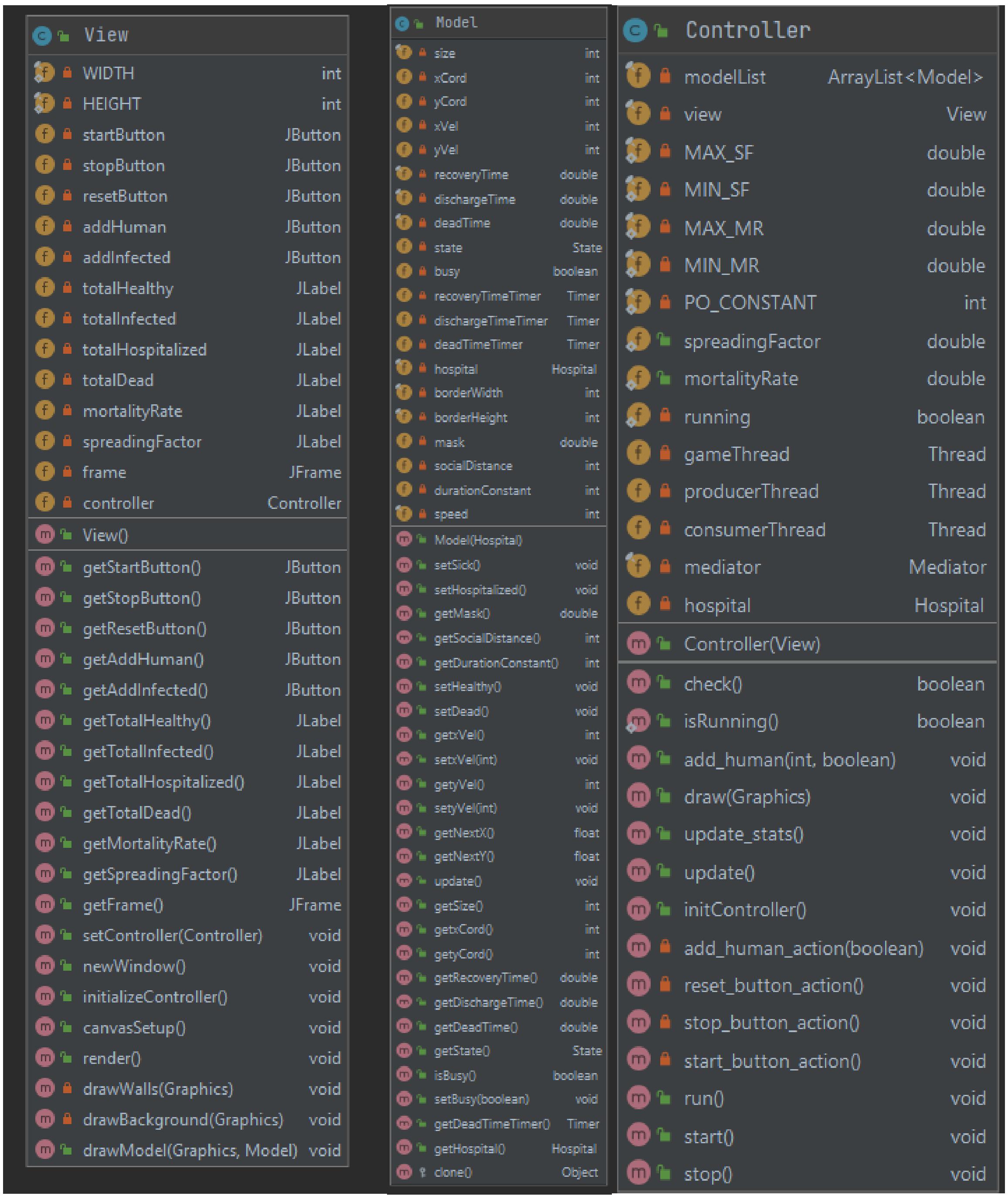


CountingTimer

start	int
deadline	double
timer	Timer
model	Model
CountingTimer(double, Timer, Model)	
run()	void

State

HEALTHY	
HOSPITALIZED	
INFECTED	
DEAD	
values()	State[]
valueOf(String)	State



View	
WIDTH	int
HEIGHT	int
startButton	JButton
stopButton	JButton
resetButton	JButton
addHuman	JButton
addInfected	JButton
totalHealthy	JLabel
totalInfected	JLabel
totalHospitalized	JLabel
totalDead	JLabel
mortalityRate	JLabel
spreadingFactor	JLabel
frame	JFrame
controller	Controller
View()	
getStartButton()	JButton
getStopButton()	JButton
getResetButton()	JButton
getAddHuman()	JButton
getAddInfected()	JButton
getTotalHealthy()	JLabel
getTotalInfected()	JLabel
getTotalHospitalized()	JLabel
getTotalDead()	JLabel
getMortalityRate()	JLabel
getSpreadingFactor()	JLabel
getFrame()	JFrame
setController(Controller)	void
newWindow()	void
initializeController()	void
canvasSetup()	void
render()	void
drawWalls(Graphics)	void
drawBackground(Graphics)	void
drawModel(Graphics, Model)	void

Model	
size	int
xCord	int
yCord	int
xVel	int
yVel	int
recoveryTime	double
dischargeTime	double
deadTime	double
state	State
busy	boolean
recoveryTimeTimer	Timer
dischargeTimeTimer	Timer
deadTimeTimer	Timer
hospital	Hospital
borderWidth	int
borderHeight	int
mask	double
socialDistance	int
durationConstant	int
speed	int
Model(Hospital)	
setSick()	void
setHospitalized()	void
getMask()	double
getSocialDistance()	int
getDurationConstant()	int
setHealthy()	void
setDead()	void
getxVel()	int
setxVel(int)	void
getyVel()	int
setyVel(int)	void
getNextX()	float
getNextY()	float
update()	void
getSize()	int
getxCord()	int
getyCord()	int
getRecoveryTime()	double
getDischargeTime()	double
getDeadTime()	double
getState()	State
isBusy()	boolean
setBusy(boolean)	void
getDeadTimeTimer()	Timer
getHospital()	Hospital
clone()	Object

Controller	
modelList	ArrayList<Model>
view	View
MAX_SF	double
MIN_SF	double
MAX_MR	double
MIN_MR	double
PO_CONSTANT	int
spreadingFactor	double
mortalityRate	double
running	boolean
gameThread	Thread
producerThread	Thread
consumerThread	Thread
mediator	Mediator
hospital	Hospital
Controller(View)	
check()	boolean
isRunning()	boolean
add_human(int, boolean)	void
draw(Graphics)	void
update_stats()	void
update()	void
initController()	void
add_human_action(boolean)	void
reset_button_action()	void
stop_button_action()	void
start_button_action()	void
run()	void
start()	void
stop()	void