CARREFOUR

Mini-Project

Made by :

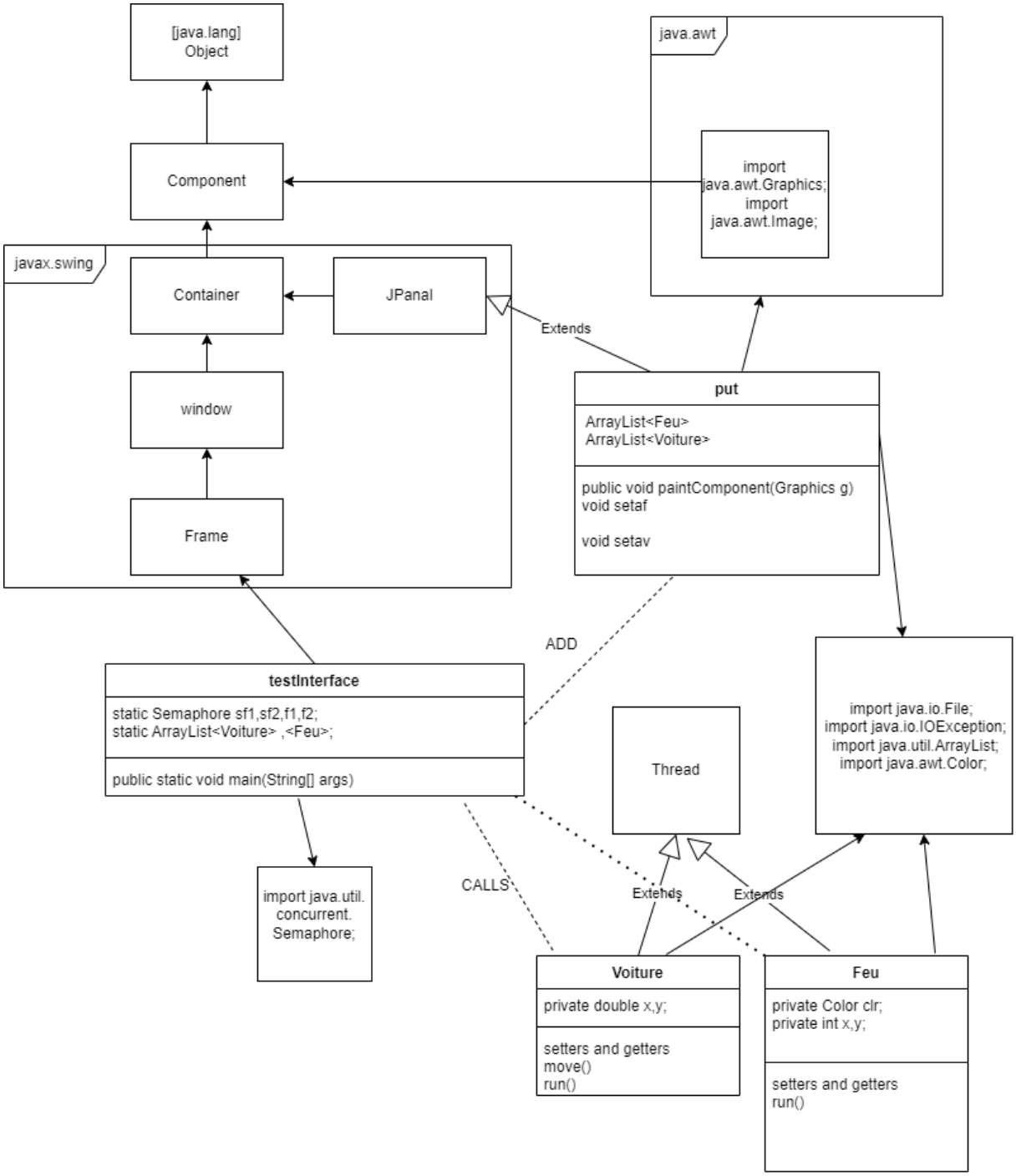
- **RACHA LABIDI**
- **HIRECH DJOUMANA**

G4

TEACHER: Meriem BELGUIDOUM

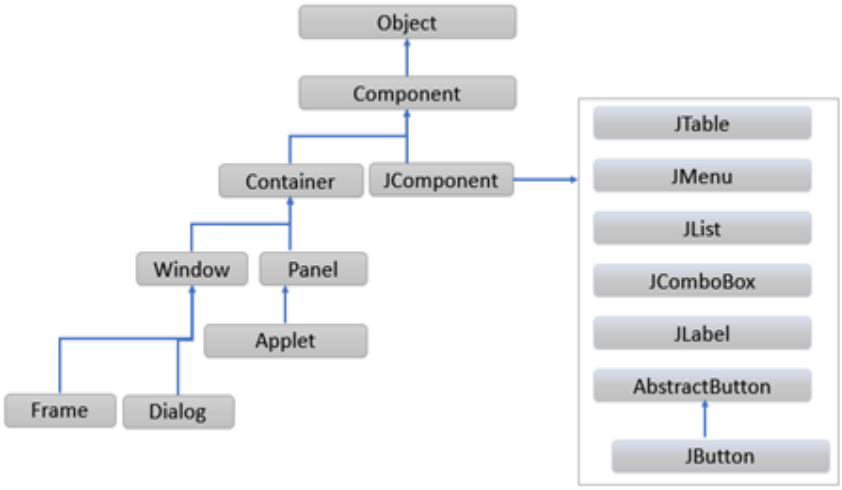
12/02/2022

❖ Application architecture :



❖ SWING:

Swing is a set of program component s for Java programmers that provide the ability to create graphical user interface (GUI) components, such as buttons and scroll bars, that are independent of the windowing system for specific operating system . Swing components are used with the Java Foundation Classes (JFC).

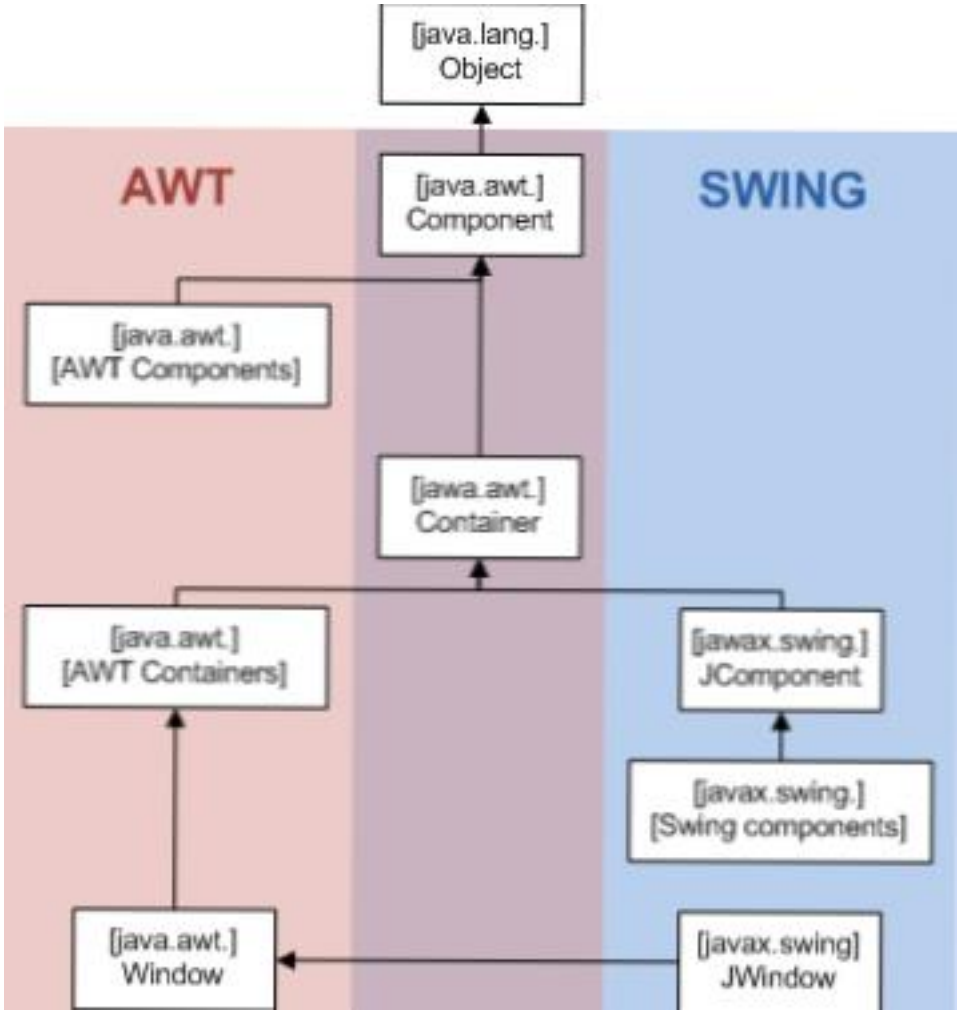


❖ AWT :

AWT stands for Abstract Window Toolkit. It is a platform-dependent API to develop GUI (Graphical User Interface) or window-based applications in Java. It was developed by heavily Sun Microsystems In 1995. It is heavy-weight in use because it is generated by the system's host operating system. It contains a large number of classes and methods, which are used for creating and managing GUI.

❖ SWING VS AWT :

Java AWT	Java Swing
AWT components are platform-dependent.	Java swing components are platform-independent.
AWT components are heavyweight.	Swing components are lightweight.
AWT doesn't support pluggable look and feel.	Swing supports pluggable look and feel.
AWT provides less components than Swing.	Swing provides more powerful components such as tables, lists, scrollpanes, colorchooser, tabbedpane, etc.
AWT doesn't follows MVC(Model View Controller) where model represents data, view represents presentation and controller acts as an interface between model and view.	Swing follows MVC.



❖ Semaphore:

is simply a variable that is non-negative and shared between threads. A semaphore is a signaling mechanism, and a thread that is waiting on a semaphore can be signaled by another thread. It uses two atomic operations, 1) `acquire()`, and 2) `release()` for the process synchronization.

We used 4 semaphore :

```
static Semaphore sf1=new Semaphore(1);
static Semaphore sf2=new Semaphore(1);
static Semaphore f1=new Semaphore(0);
```

```
static Semaphore f2=new Semaphore(1);
```

sf1: **it's used to stops the cars if there is a car in front of them in the FIRST road .**

sf2: **it's used to stops the cars if there is a car in front of them in the SECOND road .**

f1: **it's used to stops the cars if there is a RED LIGHT in the first road .**

f2: **it's used to stops the cars if there is a RED LIGHT in the SECOND road .**

❖ SOURCE CODE EXPLAINING:

➢ Classe Feu :

This class extends thread , and contains the attributes that are related the the light like the color , x and y The constructor and the getters and the setters Also it overrides the method run () Like any subclass of Thread So , in this method we are going to change the light from green to red in road 1 and the opposite in the second road + waiting some milliseconds between every changing

```
package carrefour;
import java.awt.Color;
public class Feu extends Thread{
	private Color clr;
	private int x,y;
	Feu(Color clr,int x ,int y){
		this.setClr(clr);
		this.setX(x);
		this.setY(y);
	}
	public Color getClr() {
		return clr;
	}
	public void setClr(Color clr) {
		this.clr = clr;
	}
	public int getX() {
		return x;
	}
	public void setX(int x) {
		this.x = x;
	}
	public int getY() {
		return y;
	}
	public void setY(int y) {
		this.y = y;
	}
		public void run() {
		while(true) {
			if(getClr()==Color.green) {
				try {
					testInterface.f1.acquire();
					setClr(Color.red);
					testInterface.f2.release();
					sleep(4000);
				} catch (InterruptedException e) {
					e.printStackTrace();
```

```
                        }
                }else {
                        try {
                                testInterface.f2.acquire();
                                setClr(Color.green);
                                testInterface.f1.release();
                                sleep(4000);
                        } catch (InterruptedException e) {
                                e.printStackTrace();
                        }
                }
        }

}
```

}

Classe Voiture :

This class extends thread , and contains the attributes that are related the cars like the x and y The constructor and the getters and the setters to set or edit this attributes, and a method MOVE() :to change the x and the y of the car and make it moving.

Also it overrides the method run () Like any subclass of Thread So , in this method we are going to stop the cars or let it move :

- **If the light is green and the car is the first in the road it can moves else it will wait until it becomes the first (no car in front of it)**
- **If the car is the first and the light is red it waits until the light turns to green**

```
- package carrefour;
-
- import java.awt.Color;
- import java.util.ArrayList;
-
- //import java.awt.Color;
- public class Voiture extends Thread{
-       private double x,y;
-       int m=0,n=0;
-       static ArrayList<Voiture> v = new ArrayList <Voiture>();
-
-       Voiture(double x, double y){
-               this.setX(x);
-               this.setY(y);
-       }
-       public double getY() {
-               return y;
-       }
-       public void setY(double y) {
-               this.y = y;
-       }
-       public double getX() {
-               return x;
-       }
-       public void setX(double x) {
-               this.x = x;
-       }
-       public void move() {
-               if(x==290) {
-                       y=y+0.000001;
-               }else {
-                       if(y==215) {
```

```
-                               x=x+0.000001;
-                         }
-                   }
-             }
-
-             public void run() {
-                   while(true) {
-
-
             if(getY()<100&&testInterface.f.get(1).getClr().equals(Color.red
      )) {
-                               try {
-                                     testInterface.sf1.acquire();
-                                     testInterface.f1.acquire();
-
-                                     move();
-                                     if(getY()<260&&getY()>100) {
-                                           while(getY()<260) {
-                                                 move();
-                                           }
-                                     }
-
-                                     testInterface.f1.release();
-                                     testInterface.sf1.release();
-                               } catch (InterruptedException e) {
-                                     e.printStackTrace();
-                               }
-                         }else
-
            if(getX()<150&&testInterface.f.get(0).getClr().equals(Color.red
      )) {
-                                     try {
-                                           testInterface.sf2.acquire();
-                                           testInterface.f2.acquire();
-
-                                           move();
-                                           if(getX()<350&&getX()>150) {
-                                                 while(getY()<350) {
-                                                       move();
-                                                 }
-                                           }
-
-                                           testInterface.f2.release();
-                                           testInterface.sf2.release();
-                                     } catch (InterruptedException e) {
-                                           e.printStackTrace();
-                                     }
-                               }else {
-                                     move();
-                               }
-
-                   }
-             }
-       }
-
```

Classe testInterface :

This class contains the main method , and the declaration and the initialisation of the semaphores and calling an arraylist of our 2 classes feu and voiture . We puted the lights on their places ,I used matlab to get the coordination of the pic , the first light starts with the color green and it's place is on (200,300),

And the second one is initialised with a red color and it's placed on (400.120) ,

Then by running the method run in the class fue they are going to switch the colors - i choosed to put 4 cars on the first road

..every car has it's own x and y The cars in this road are moving on the y-line which is fixed on y=215

So they gonna use x=x+0.00..1 to move

Par contre , the cars on the second road are moving on the x-line and it's fixed on x=230

And they are moving using y=y+0.00...1 ....

We created an instance of the class put to add the cars and the lights Then we add this instance to our frame

```
-   package carrefour;
-
-   import java.awt.Color;
-   import java.util.ArrayList;
-   import java.util.concurrent.Semaphore;
-
-   import javax.swing.JFrame;
-
-   public class testInterface {
-         static Semaphore sf1=new Semaphore(1);
-         static Semaphore sf2=new Semaphore(1);
-         static Semaphore f1=new Semaphore(0);
-         static Semaphore f2=new Semaphore(1);
-         static ArrayList<Feu> f = new ArrayList <Feu>();
-         static ArrayList<Voiture> v = new ArrayList <Voiture>();
-               public static void main(String[] args) {
-                     Feu f1 = new Feu(Color.green,200,300);
-                     Feu f2 = new Feu(Color.red,400,120);
-                     put j=new put();
-                     f.add(f1);
-                     f.add(f2);
-                     j.setaf(f);
-                     Voiture v1=new Voiture(290,50);
-                     Voiture v2=new Voiture(70,215);
-                     Voiture v3=new Voiture(290,-70);
-                     Voiture v4=new Voiture(-240,215);
-                     Voiture v5=new Voiture(290,-220);
-                     Voiture v6=new Voiture(-350,215);
-                     Voiture v7=new Voiture(290,-340);
-                     Voiture v8=new Voiture(-600,215);
-                     v.add(v1);
-                     v.add(v2);
-                     v.add(v3);
```

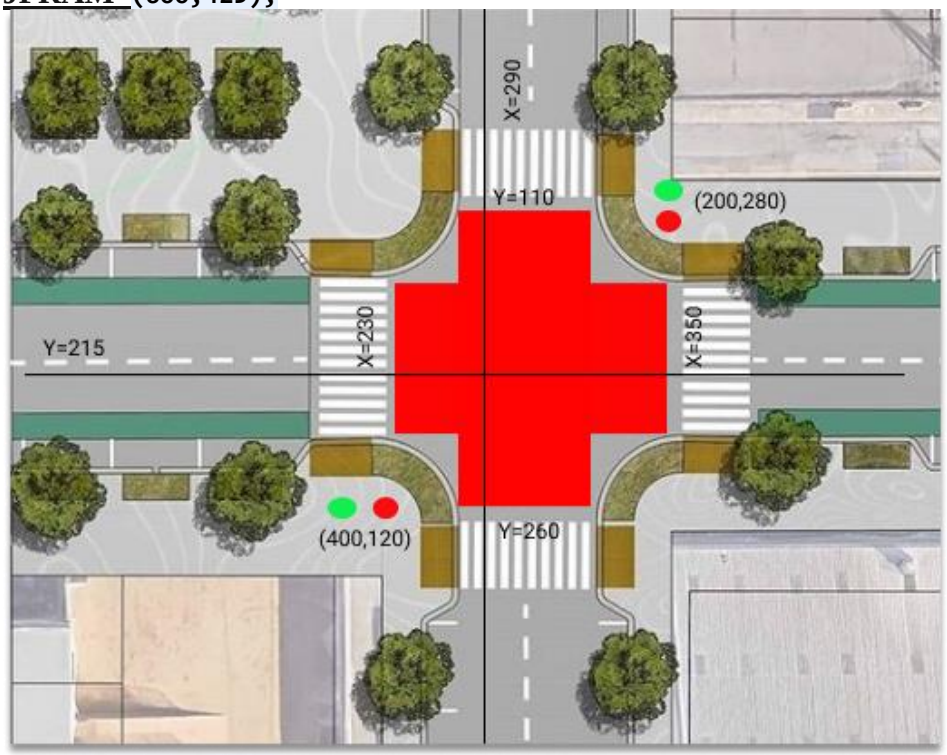
```
-                       v.add(v4);
-                       v.add(v5);
-                       v.add(v6);
-                       v.add(v7);
-                       v.add(v8);
-                       j.setav(v);
-                       j.repaint();
-                       JFrame Fenetre =new JFrame();
-                       Fenetre.setSize(600,429);
-                       Fenetre.setResizable(false);
-                       Fenetre.add(j);
-                       Fenetre.setVisible(true);
-                       for(int i=0;i<f.size();i++)
-                       {
-                             f.get(i).start();
-                       }
-                       for(int i=0;i<v.size();i++)
-                       {
-                             v.get(i).start();
-                       }
-                       while(true) {
-                       j.repaint();
-                       }
-                 }
-          }
-
```

Classe put :

The class put is the class that contains the vue of our application I preferred to use pics so the CARREFOUR looks real That's why i use a pic of an intersection

THE INTERSECTION PIC :

THE PICTURE SIZE USED ON THE PRORAM IS THE SAME AS THE JFRAM `(600,429);`



And we used pics of traffic lights , 2 copies for each

1. **Light**

2. **RED light**

3. **GREEN light**

And some cars and buses :

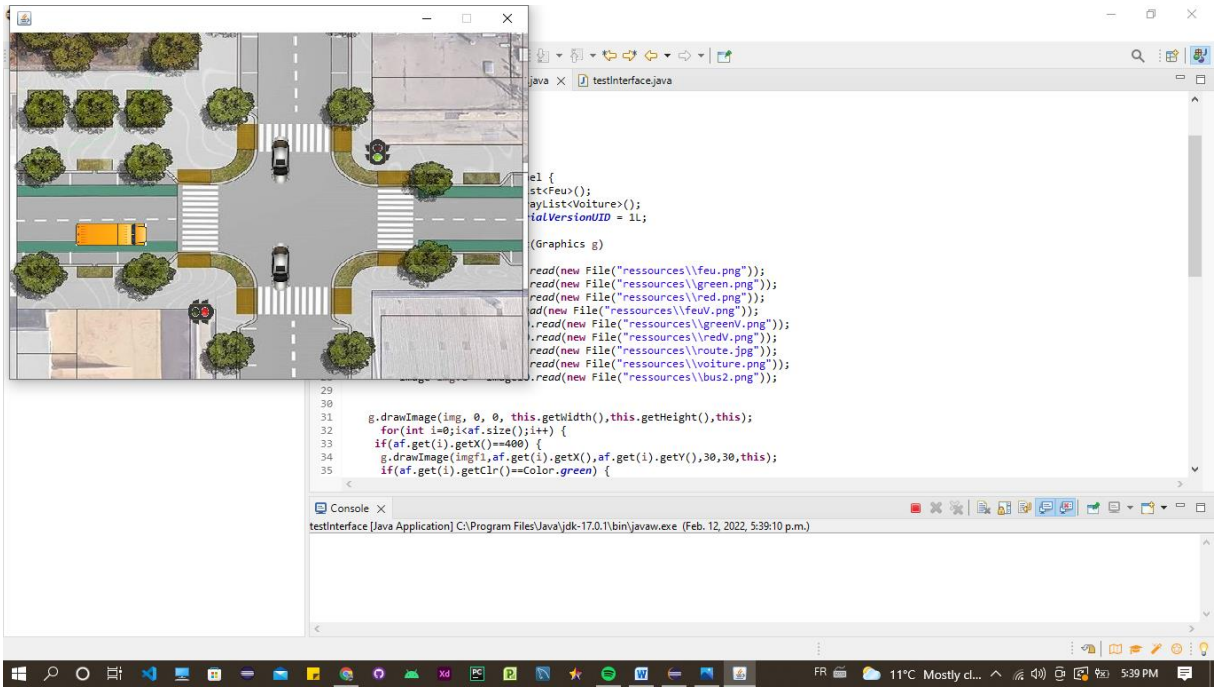
It contains 2 methods to set the cars and the lights

```
package carrefour;


 import java.awt.Color;
 import java.awt.Graphics;
```

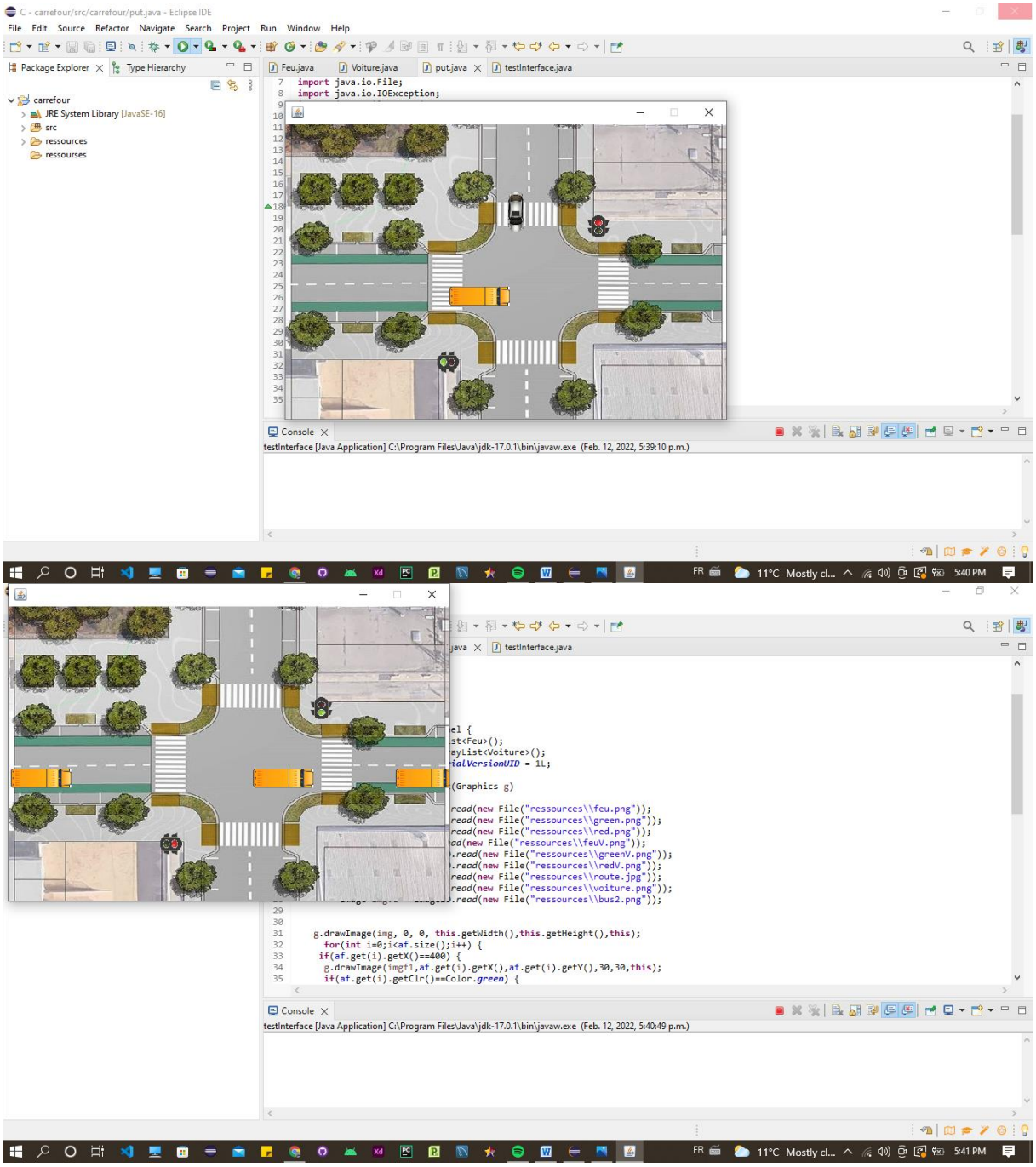
```
import java.awt.Image;
import java.io.File;
import java.io.IOException;
import java.util.ArrayList;

import javax.imageio.ImageIO;
import javax.swing.JPanel;
public class put extends JPanel {
ArrayList<Feu> af=new ArrayList<Feu>();
ArrayList<Voiture> av=new ArrayList<Voiture>();
private static final long serialVersionUID = 1L;

      public void paintComponent(Graphics g)
        {    try {
               Image imgf1=ImageIO.read(new File("ressources\\feu.png"));
               Image imgf2=ImageIO.read(new File("ressources\\green.png"));
               Image imgf3=ImageIO.read(new File("ressources\\red.png"));
              Image imgf=ImageIO.read(new File("ressources\\feuV.png"));
               Image imgf2V=ImageIO.read(new File("ressources\\greenV.png"));
               Image imgf3V=ImageIO.read(new File("ressources\\redV.png"));
               Image img = ImageIO.read(new File("ressources\\route.jpg"));
              Image imgV = ImageIO.read(new File("ressources\\voiture.png"));
              Image imgVG = ImageIO.read(new File("ressources\\bus2.png"));


      g.drawImage(img, 0, 0, this.getWidth(),this.getHeight(),this);
        for(int i=0;i<af.size();i++) {
       if(af.get(i).getX()==400) {
        g.drawImage(imgf1,af.get(i).getX(),af.get(i).getY(),30,30,this);
        if(af.get(i).getClr()==Color.green) {
               g.drawImage(imgf2,af.get(i).getX(),af.get(i).getY(),30,30,this);
        }else  if(af.get(i).getClr()==Color.red) {
               g.drawImage(imgf3,af.get(i).getX(),af.get(i).getY(),30,30,this);
        }
        }else {


               g.drawImage(imgf,af.get(i).getX(),af.get(i).getY(),30,30,this);
               if(af.get(i).getClr()==Color.green) {

g.drawImage(imgf2V,af.get(i).getX(),af.get(i).getY(),30,30,this);
               }else  if(af.get(i).getClr()==Color.red) {

g.drawImage(imgf3V,af.get(i).getX(),af.get(i).getY(),30,30,this);
               }
               }


        }
        for(int i=0;i<av.size();i++) {
            // g.setColor(Color.black);

            // g.fillRect((int)av.get(i).getX(),(int) av.get(i).getY(), 25, 25);
             if(av.get(i).getY()==215) { //140

g.drawImage(imgVG,(int)av.get(i).getX(),(int)av.get(i).getY(),80,25,this);
              }if(av.get(i).getX()== 290) { //330  110

g.drawImage(imgV,(int)av.get(i).getX(),(int)av.get(i).getY(),30,60,this);
               }
```

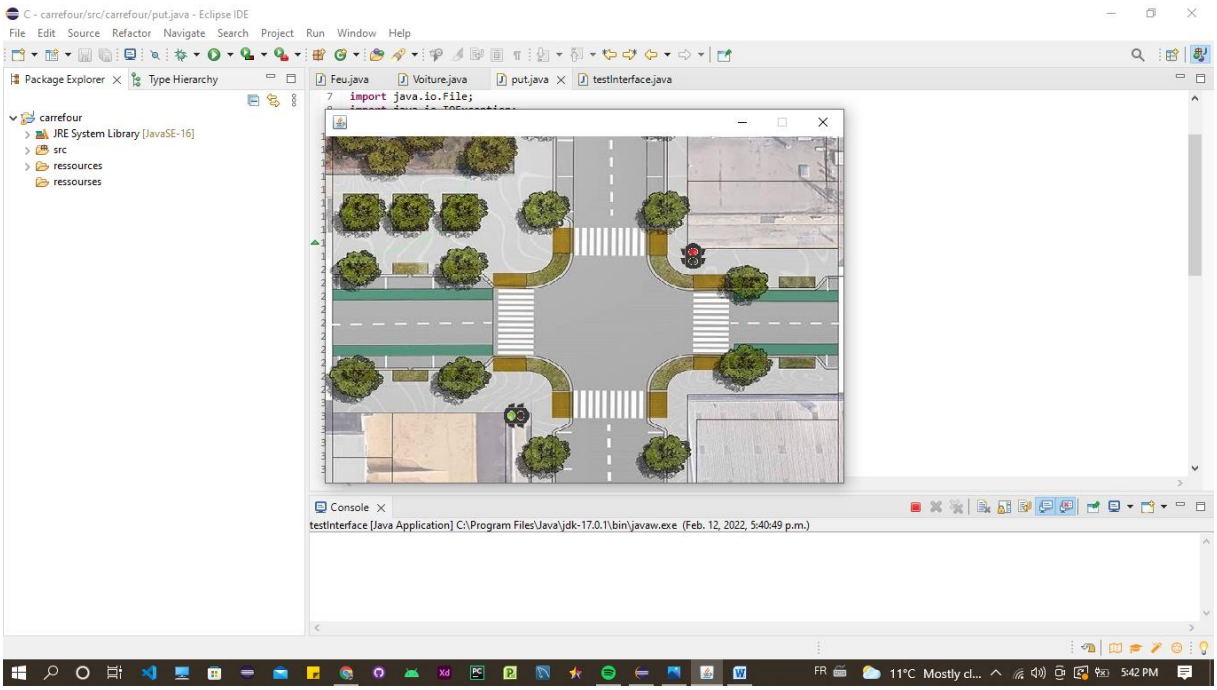
```
            }
        }

        catch (IOException e) {       e.printStackTrace();
        }}
    void setaf(ArrayList<Feu> f) {
            af=f;
    }
    void setav(ArrayList<Voiture> v) {
            av=v;
    }
}
```

❖ Graphic interface and running the code :



CARREFOUR


C - carrefour/src/carrefour/put.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
Type Hierarchy
carrefour
JRE System Library [JavaSE-16]
src
ressources
ressourses
Feu.java
Voiture.java
put.java
testInterface.java
import java.io.File;
import java.io.IOException;
Console
testInterface [Java Application] C:\Program Files\Java\jdk-17.0.1\bin\javaw.exe (Feb. 12, 2022, 5:39:10 p.m.)
11°C Mostly cl...
5:40 PM
testInterface.java
el {
st<Feu>();
ayList<Voiture>();
ialVersionUID = 1L;
(Graphics g)
read(new File("ressources\\feu.png"));
read(new File("ressources\\green.png"));
read(new File("ressources\\red.png"));
ad(new File("ressources\\feuV.png"));
.read(new File("ressources\\greenV.png"));
.read(new File("ressources\\redV.png"));
read(new File("ressources\\route.jpg"));
read(new File("ressources\\voiture.png"));
.read(new File("ressources\\bus2.png"));
g.drawImage(img, 0, 0, this.getWidth(),this.getHeight(),this);
for(int i=0;i<af.size();i++) {
if(af.get(i).getX()==400) {
g.drawImage(imgf1,af.get(i).getX(),af.get(i).getY(),30,30,this);
if(af.get(i).getClr()==Color.green) {
Console
testInterface [Java Application] C:\Program Files\Java\jdk-17.0.1\bin\javaw.exe (Feb. 12, 2022, 5:40:49 p.m.)
11°C Mostly cl...
5:41 PM



❖ Some methods explanation:

Fenetre.setSize(600,429); //setting the size of our window

Fenetre.setResizable(**false**); // we put it false so we can't make the window bigger cz we can't resize the road pic

Fenetre.add(j);// add a component
// here we are adding "j" whith is an object of the class put where we put all of our pics and make their sizes and their X and Y .

Fenetre.setVisible**(**true**);**// Show or hide a component. It is false by default
// here TO MAKE THE WINDOW VISIBLE
v.get(i).start();
f.get(i).start();// TO START THE THREADS

Causes this thread to begin execution; the Java Virtual Machine calls the run method of this thread.

The result is that two threads are running concurrently: the current thread (which returns from the call to the start method) and the other thread (which executes its run method).

It is never legal to start a thread more than once. In particular, a thread may not be restarted once it has completed execution.

j.repaint(); //Repaints this component.

If this component is a lightweight component, this method causes a call to this component's `paint` method as soon as possible. Otherwise, this method causes a call to this component's `update` method as soon as possible