Certamen 1: Pregunta 2 de Desarrollo

ELO329, 1º Sem. 2022

En la primera etapa de la tarea 1, usted tuvo que crear las clases para controlar una lámpara. En http://profesores.elo.utfsm.cl/~agv/elo329/1s22/C1/

usted encontrará un código **similar** a la primera etapa de la tarea 1, donde algunas funcionalidades son probadas desde la clase Main dada. La salida de este programa es:

Time	State	L0R	L0G	L0B
0.0	OFF	255	255	255
2.0	ON	255	255	255
5.0	OFF	255	255	255

a) A partir del código anterior, se pide crear las clases AirConditioner y AC_Control y otros cambios necesarios para que la clase MainAC pueda ser compilada y ejecutada sin errores.

La clase MainAC está en el código ya dado.

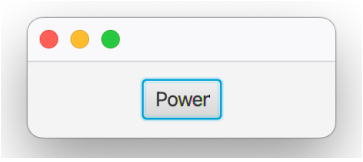
Usando una instancia de AC_Control es posible definir la temperatura solicitada al aire acondicionado como también su encendido y apagado.

La salida esperada para MainAC es:

Time	State	L0R	L0G	L0B	State	AC0T
0.0	OFF	255	255	255	OFF	20
2.0	ON	255	255	255	OFF	20
5.0	OFF	255	255	255	OFF	20
6.0	OFF	255	255	255	OFF	20
6.0	OFF	255	255	255	ON	20
6.0	OFF	255	255	255	ON	21

Notar que mientras el aire acondicionado está apagado, éste no modifica su temperatura.

b) A partir del código dado, cree la clase MainFX como alternativa a Main. En esta clase se pide crear una interfaz JavaFX con un botón como el mostrado. Cada vez que el usuario presiona el botón, una lámpara (instancia de Lamp) cambia su estado y se imprime por consola el estado de la lámpara. Ej. Luego de presionar tres veces, se debe tener:



ON	255	255	255
OFF	255	255	255
ON	255	255	255

Para a) suba a AULA su respuesta como a.zip, a.rar o a.tar. Para b) suba el archivo MainFX.java.

a)

```
public class AirConditioner extends DomoticDevice {    // 12 pts.
   public AirConditioner (int channel){
      super(nextId++, channel);
      temp = 20;
      state = PowerState.OFF;
   }
   public void changePowerState(){
         state = state==PowerState.ON ? PowerState.OFF : PowerState.ON;
   }
   public void setTemp(int t){
      if (state==PowerState.ON)
         temp = t;
   }
   public String getHeader(){
      String s = "State\t" + "AC" + getId() + "T";
      return s;
   }
   public String toString(){
      return state.toString() + "\t" + temp;
   }
   private int temp;
   private PowerState state;
   private static int nextId=0;
}

public class AC_Control extends DomoticDeviceControl{.  // 8 pts.
   public AC_Control(int channel, Cloud c){
      super(channel,c);
   }
   public void setT(int temp){
      cloud.setACtemp(channel, temp);
   }
   public void pressPower(){
      cloud.changeACpowerState(channel);
   }
}
```

Adicionalmente, es necesario adaptar la clase Cloud ya provista.

En Cloud los métodos y atributos que se deben agregar son:

```
public class Cloud {      // 10 pts.
   public Cloud() {
      lamps = new ArrayList<DomoticDevice>();
      airConds = new ArrayList<DomoticDevice>();  // <--- New
   }
   public void addLamp(Lamp l){
      lamps.add(l);
   }
```

```
    public void addAirConditioner(AirConditioner ac) {   // <--- New
        airConds.add(ac);
    }
    private DomoticDevice getDomoticDeviceAtChannel( ArrayList<DomoticDevice> devices, int channel){
        for (DomoticDevice device: devices)
            if (device.getChannel() ==channel)
                return device;
        return null;
    }
    public void changeLampPowerState(int channel) {
        Lamp l=(Lamp) getDomoticDeviceAtChannel(lamps, channel);
        if (l != null) l.changePowerState();
    }
    public void changeACpowerState(int channel) {  // <--- New
        AirConditioner ac = (AirConditioner) getDomoticDeviceAtChannel(airConds, channel);
        if (ac != null) ac.changePowerState();
    }
    public void setACtemp(int channel, int temp) {  // <--- New
        AirConditioner ac = (AirConditioner) getDomoticDeviceAtChannel(airConds, channel);
        if (ac != null) ac.setTemp(temp);
    }
    public String getHeaders() {
        String header = "";
        for (DomoticDevice l: lamps)
            header += l.getHeader()+"\t";
        for (DomoticDevice  rs: airConds)   //  <--- New
            header += rs.getHeader()+"\t";
        return header;
    }
    public String getState() {
        String state="";
        for (DomoticDevice l: lamps) {
            state += l+ "\t";
        }
        for (DomoticDevice rs: airConds)  // <--- New
            state += rs+"\t";
        return state;
    }
    private ArrayList<DomoticDevice> airConds;   // <--- New
    private ArrayList<DomoticDevice> lamps;
}
```

a) 20 pts.

```
public class MainFX extends Application {
    public void start(Stage primaryStage) {
        Cloud cloud = new Cloud();
        Lamp lamp = new Lamp(1);
        cloud.addLamp(lamp);
        LampControl lampControl = new LampControl(1, cloud);
        StackPane pane = new StackPane();
        Button lampB = new Button("Power");
```

```
        pane.getChildren().add(lampB);
        lampB.setOnAction(e->{lampControl.pressPower();System.out.println(cloud.getState());});
        Scene scene = new Scene(pane, 200, 50);
        primaryStage.setScene(scene);
        primaryStage.show();
    }
}
```