Model/View/Controller

Model

Data values and logic to manipulate them

Observable

View

How the data are displayed

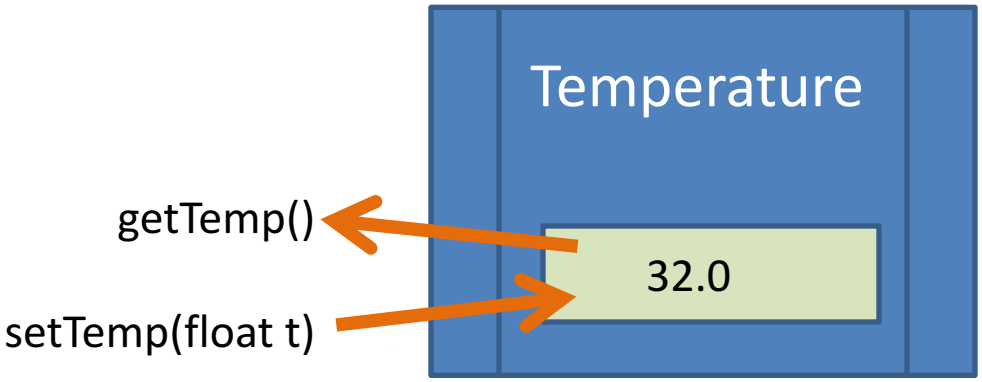
Observer

Controller

Receives data values and updates the Model

Model

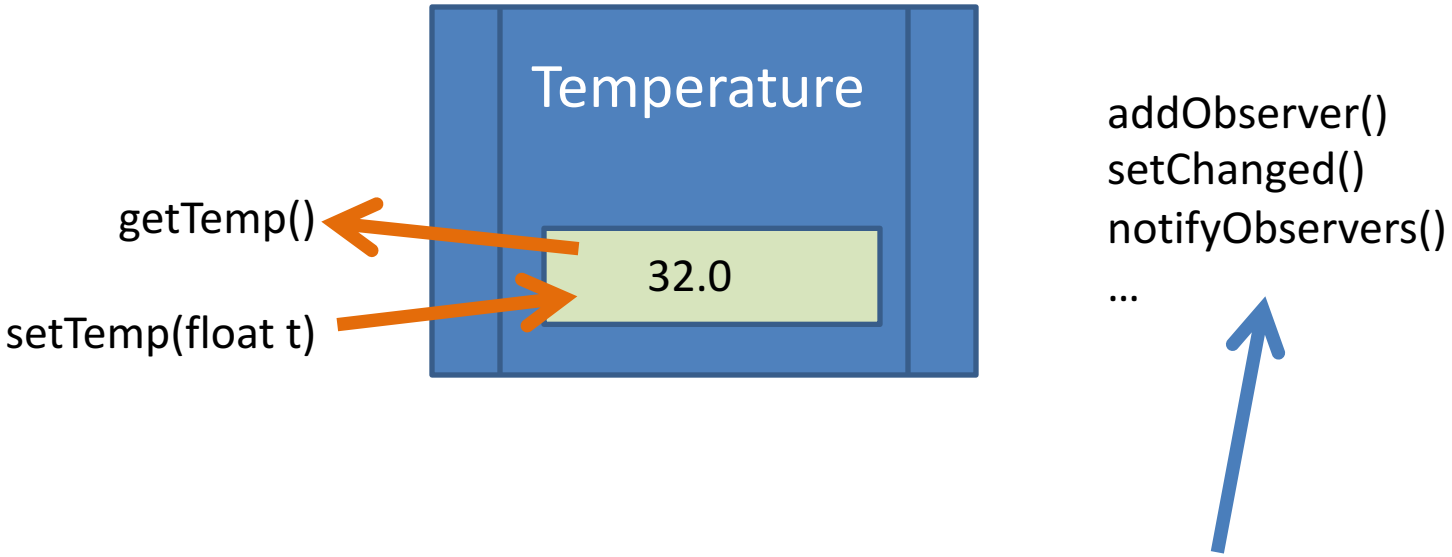
A representation of the data with no concern for how it will appear to the user.



class TemperatureModel

Model

A representation of the data with no concern for how it will appear to the user.

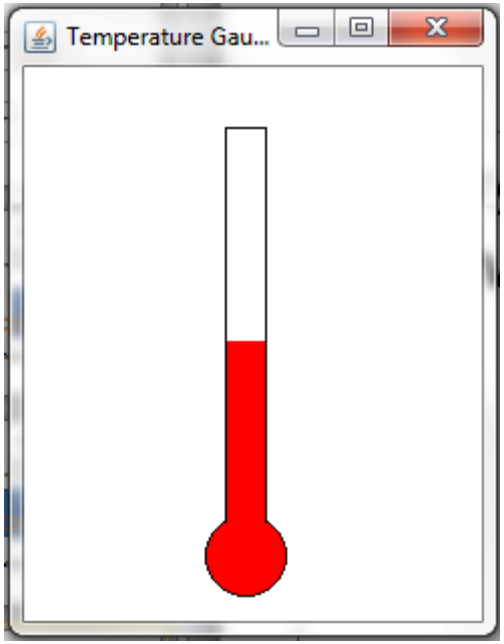


class TemperatureModel extends Observable {

View

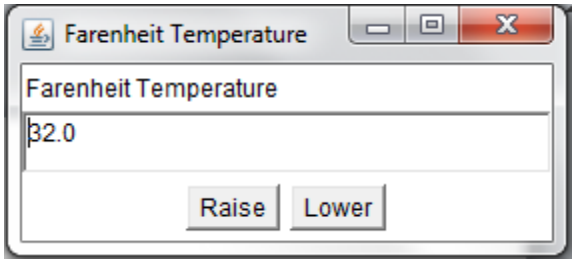
Display of the data using GUI components by **observing** the Model.

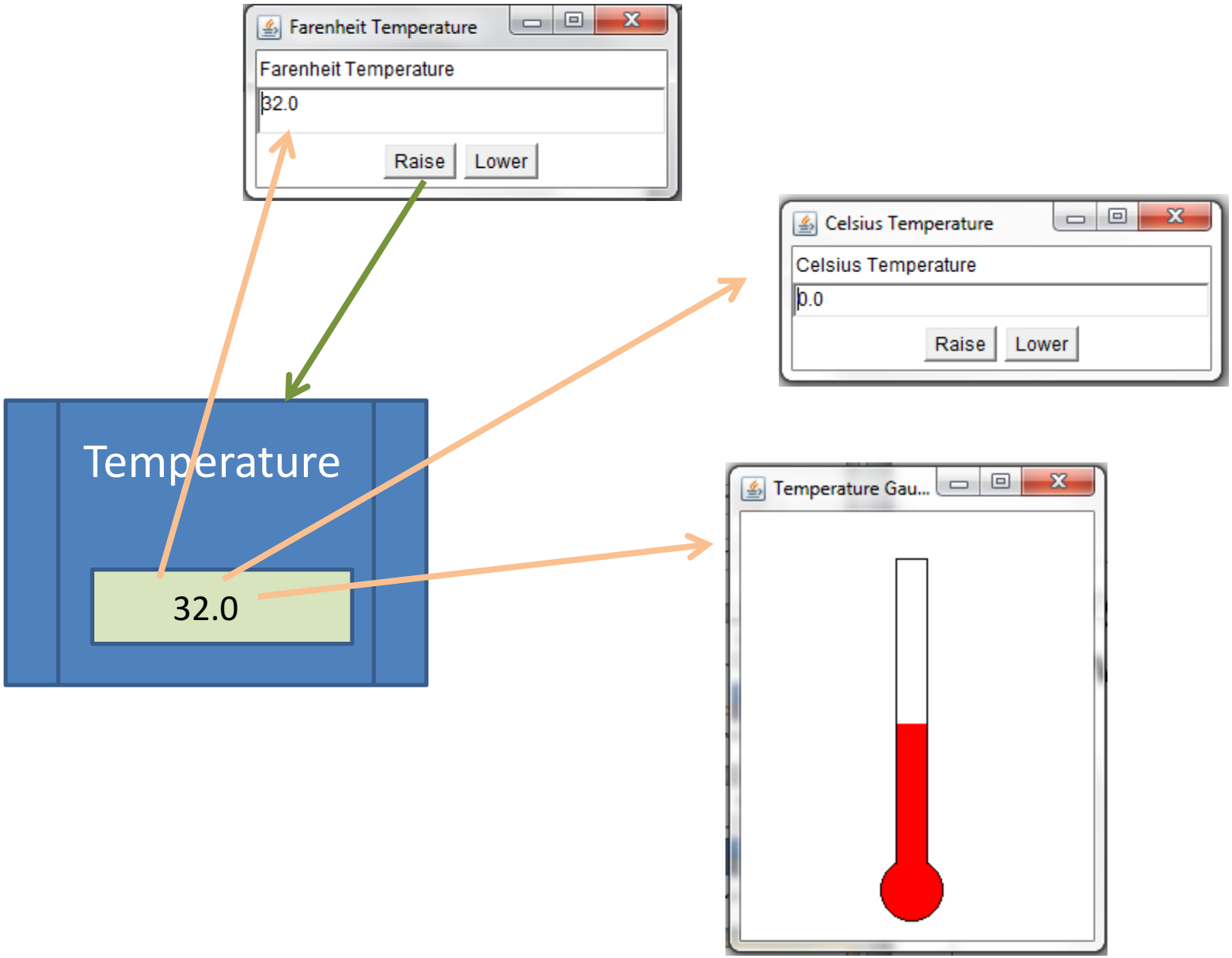
```
class Thermometer implements Observer {
```



Controller

- A *Listener* that responds to events and updates the *Model*.
- For example, push a button to raise the temperature one degree.




Farenheit Temperature
Farenheit Temperature
32.0
Raise
Lower
Celsius Temperature
Celsius Temperature
0.0
Raise
Lower
Temperature
32.0
Temperature Gau...

Typical Temperature Class

```
public class TemperatureModel  {
    private double temperatureF = 32.0;
    public double getF() {
      return temperatureF;
  }
  public double getC(){
      return (temperatureF - 32.0) * 5.0 / 9.0;
  }
  public void setF(double tempF)
  {
      temperatureF = tempF;
  }
  public void setC(double tempC)
  {  temperatureF = tempC*9.0/5.0 + 32.0;
  }
}
```

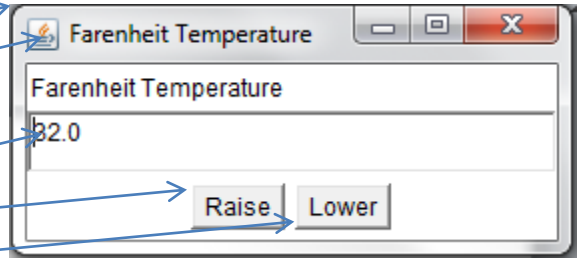
TemperatureModel

```
import java.util.Observable;
public class TemperatureModel extends Observable {
    private double temperatureF = 32.0;
    public double getF() {
      return temperatureF;
   }
  public double getC(){
      return (temperatureF - 32.0) * 5.0 / 9.0;
   }
  public void setF(double tempF)
   {
      temperatureF = tempF;
      setChanged();
      notifyObservers();
   }
  public void setC(double tempC)
   {  temperatureF = tempC*9.0/5.0 + 32.0;
      setChanged();
      notifyObservers();
   }
}
```

Controller (Listener)

```
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

  class UpListener implements ActionListener  {
     TemperatureModel model;

      public UpListener(TemperatureModel m) {
      model = m;
   }

      public void actionPerformed(ActionEvent e) {
         model.setF(model.getF() + 1.0);
      }
   }
```

```
import java.awt.*;
import java.awt.event.*;
abstract class TemperatureGUI implements java.util.Observer {
   private String label;
   private TemperatureModel model;
   private Frame temperatureFrame;
   private TextField display = new TextField();
   private Button upButton = new Button("Raise");
   private Button downButton = new Button("Lower");

   TemperatureGUI(String theLabel, TemperatureModel tModel, int h, int v) {
      label = theLabel;
      model = tModel;
      Frame temperatureFrame;
      temperatureFrame = new Frame(label);
      temperatureFrame.add("North", new Label(label));
      temperatureFrame.add("Center", display);
      Panel buttons = new Panel();
      buttons.add(upButton);
      buttons.add(downButton);
      temperatureFrame.add("South", buttons);
      temperatureFrame.addWindowListener(new CloseListener());
      model.addObserver(this); // Connect the View to the Model
      temperatureFrame.setSize(200,100);
      temperatureFrame.setLocation(h, v);
      temperatureFrame.setVisible(true);

   public void setDisplay(String s){
      display.setText(s);}
   public double getDisplay()  {
      return Double.valueOf(display.getText()).doubleValue();
   }
      continued…
```



```
public void addDisplayListener(ActionListener a){
   display.addActionListener(a);}

public void addUpListener(ActionListener a){
   upButton.addActionListener(a);}

public void addDownListener(ActionListener a){
   downButton.addActionListener(a);}
```

actionPerformed from the DisplayListener class:

```
public void actionPerformed(ActionEvent e) {
   double value = fg.getDisplay();
   model.setF(value);
}
```

```
import java.awt.*;
import java.awt.event.*;
import java.util.Observable;

public class FarenheitGUI extends TemperatureGUI {

   public FarenheitGUI(TemperatureModel model, int h, int v)
{
      super("Farenheit Temperature", model, h, v);
      setDisplay(""+model.getF());
      addUpListener(new UpListener(model));
      addDownListener(new DownListener(model));
      addDisplayListener(new DisplayListener(model,this));
   }

   public void update(Observable t, Object o)
       // automatically called when the model is changed
   {
      setDisplay("" + model().getF());
   }

}
```