Introduction to Event-Driven Programs

Section 7.5 Graphical/Internet Java: Event-Driven Programming

/*
GUITemperature.java converts Celsius temperatures to Fahrenheit. It uses a graphical user interface to interact with the user.
Author: L. Nyhoff
Date: Dec. 7, 2002
*/

import ann.gui.*;   // CloseableFrame
import javax.swing.*; // JLabel, JTextField, JPanel
import java.awt.*;  // Color
import java.awt.event.*; // ActionEvent, ...

class GUITemperature extends CloseableFrame implements ActionListener {

//-- GUI Constructor
public GUITemperature() {
    setTitle("Temperature Converter");
    myCelsiusLabel = new JLabel("Celsius: ", SwingConstants.RIGHT);
    myCelsiusField = new JTextField(12);
    myCelsiusField.addActionListener(this);
    myCelsiusLabel = new JLabel("Fahrenheit: ", SwingConstants.RIGHT);
    myFahrenheitField = new JTextField(12);
    myPanel = new JPanel();
    myPanel.setLayout( new GridLayout(2, 2));
    myPanel.add(myCelsiusLabel);
    myPanel.add(myCelsiusField);
    myPanel.add(myFahrenheitLabel);
    myPanel.add(myFahrenheitField);
    setContentPane(myPanel);
}

/**
    ActionEvent handler
    * Receive: an ActionEvent event
    * Precondition: event was generated by an "Enter" in myCelsiusField
    * Postcondition: event has been processed
    */
public void actionPerformed(ActionEvent event) {
    String celsiusString = myCelsiusField.getText();
    double celsius = Double.parseDouble(celsiusString);
    double fahrenheit = ((9.0/5.0)*celsius) + 32;
    myFahrenheitField.setText("" + fahrenheit);
}
public static void main(String [] args) {
    GUITemperature aGUITemp = new GUITemperature();
aGUITemp.setBackground(Color.white);
aGUITemp.pack();
aGUITemp.setVisible(true);
}

private JLabel myCelsiusLabel, myFahrenheitLabel;
private JTextField myCelsiusField, myFahrenheitField;
private JPanel myPanel;

Java's Event Model

It's called the **event delegation model**. It consists of:

**Event sources**: objects that *generate* events (buttons, text fields, etc.). They are said to *fire* events.

**Event listeners**: objects that *respond* to events.

Event Sources

A GUI program must define an event-generating component in the GUI, usually in the constructor.

Example: `myCelsiusField`, a `JTextField` that fires an `ActionEvent` when the user presses the Enter key.

Note that the program *implements* the `ActionListener` interface.

Our simple GUITemperature example has a single event source: `myCelsiusField`

The GUI Temperature example in Section 7.5 has three event sources: a Celsius field, a Fahrenheit field, a Kelvin field. (The program converts a temp. on any of these scales to the corresponding temperature in the others.)
Event Listeners

To have a GUI respond to an event:
- **Create a listener** for that event source
- **Register** the listener with that event source

Usually the listener is the GUI app itself. For example, a `GUITemperature` object is also:
- a `CloseableFrame` object
- an `ActionListener` object

Registering Event Listeners with Event Sources

Action event sources provide an `addActionListener()` method.

In the `GUITemperature` constructor we have:

```java
myCelsiusField.addActionListener(this);
```

- `this` refers to the object being constructed
- the object registers itself as an `ActionListener`

Now the listener has been **bound** to the event source.

The `actionPerformed()` Method

- It is invoked when an `ActionEvent` source fires an `ActionEvent`
  - the GUI class has been specified as the listener
- It must specify what to do when the event occurs. In our example:
  - get a string from `myCelsiusField`
  - convert it to a double
  - compute corresponding Fahrenheit temp.
  - put it (as a `String`) in `myFahrenheitField`

Summary of Common Structure of a GUI Constructor

1. Create components & listeners, register listeners with sources that fire events
2. Create a `JPanel` for components
3. Specify a layout manager for the `JPanel`
4. Mount components on the `JPanel`, usually via the `add()` method
5. Make the `JPanel` the content pane of window frame