Events, Listeners and Actions

2 – 4 pm Tuesday 8/14/2008 @JD2211

1

Agenda

- Review
  - GUI Components, Layout and Appearance
- Events
- Listeners
- Actions

2

GUI

- In the 1980s, the transition from terminal-based interface to GUI revolutionized and simplified the use of computers
- A GUI program is event-driven and is constructed quite differently from a program that communicates through a terminal

3

GUI Program

- GUI programs are complex
- To construct a GUI program within a reasonable time, we have to utilize libraries with ready-made functions
  - Xlib, Motif in UNIX
  - Win API (Application Programming Interface) in Windows

4

Program-Driven

- In a traditional program, the reading of input data is program-driven

5

Event-Driven

- Two phases of execution of an event-driven program
  - Initialization phase
  - Waiting phase

6

Event Classes

- Each event is described by event classes
  - P. 412
- When an event occurs, an object is created of the class that describes the event
- After this, a method is called in the listener that has been defined to listen to this type of the event

7

Event Classes (cont'd)

- All event classes has a common superclass called `EventObject`
  - Classes that describe event related to Java AWT are founded in `java.awt.event`
- Classes particular to Swing include `ChangeEvent`
  - in `javax.swing.event`.

8

Event Classes (cont'd)

- For each event, there is a graphic component in which the event occurred
- One way to find this component is to call the method `getSource()`, which is defined in class `EventObject`

9

Listener Classes

- To trap an event, you should create a listener
  - A listener is an object of a listener class
- There must be an interface for each and every event class which the listener class must implement
- This listener interface will have the same name as the corresponding event class, but with `Listener` as a suffix instead of `Event`

10

Listener Classes (cont'd)

- The methods the listener class has are defined in the interface
- As parameter, a listener method will always get an object of the current event class

11

Listener Classes (cont'd)

- A listener must be registered
- We do this by calling the method `add?Listener()` for the component the listener is to be connected to
  - The character ? Is then replaced by the name of the event

12

Listener Classes (cont'd)

```
class C extends C0 implements ActionListener{
        C(){
                …
                b.addActionListener(this); // register the listener
                …
        }

        public void actionPerformed(ActionEvent e){ // listener method
                …
        }
}
```

13

Listener Classes (cont'd)

```
class C extends C0 {
        C(){
                …
                MyListener listener = new MyListener();
                b.addActionListener(listener); // register the listener
                …
        }

        // listener class
        private MyListener implements ActionListener{
                public void actionPerformed (ActionEvent e) {
                …
        }
}
```

14

Example 1

15

```
import java.awt.*;
import java.awt.event.*; // contains listener classes
import javax.swing.*;

public class JButtonDemo extends JFrame implements ActionListener {
  private JLabel l1 = new JLabel("Welcome!", JLabel.CENTER);
  private JButton exit = new JButton("Exit");
  private JButton eng = new JButton("English");
  private JButton dutch = new JButton("Dutch");

  public JButtonDemo(){
    super("JBUTTON DEMO");
    Container c = getContentPane();
    c.setLayout(new GridLayout(2,2));

    c.add(l1);  c.add(exit);   c.add(eng); c.add(dutch);
    c.setBackground(Color.white);

    l1.setFont(new Font("SansSerif", Font.ITALIC, 20));
    exit.setFont(new Font("SansSerif", Font.BOLD, 14));
```

16

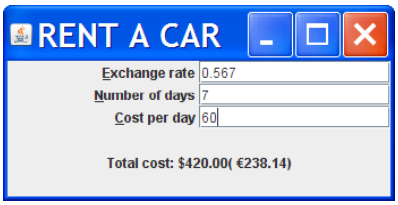
```
      // register listners
          exit.addActionListener(this);
          eng.addActionListener(this);
          dutch.addActionListener(this);

          eng.setEnabled(false); // English is shown

          setSize(350, 100);
          setVisible(true);
          setDefaultCloseOperation(EXIT_ON_CLOSE);
        }
```

17

```
      public void actionPerformed(ActionEvent e){ // listener method
        if (e.getSource() == exit)
          System.exit(0);
        else if (e.getSource() == eng){
          l1.setText("Welcome!");
          exit.setText("Exit");
          eng.setEnabled(false);
          dutch.setEnabled(true);
        }
        else if (e.getSource() == dutch){
          l1.setText("Welkom!");
          exit.setText("Uitgang");
          eng.setEnabled(true);
          dutch.setEnabled(false);
        }
      }
      public static void main (String[] args){
        JButtonDemo j = new JButtonDemo();
      }
    }
```

18

Example 2



19

```
// This program calculates the cost of renting a car.
// We know the number of days for which we wish to rent the car, together
//  with the rent charges per day.
// The program prints out the total cost in both euros and US dollars.
// To this end, we will give the current exchange rate as input data.

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.text.*;

public class RentCar extends JFrame implements ActionListener{
  JLabel l1 = new JLabel("Exchange rate ", JLabel.RIGHT);
  JLabel l2 = new JLabel("Number of days ", JLabel.RIGHT);
  JLabel l3 = new JLabel("Cost per day ", JLabel.RIGHT);

  JTextField t1 = new JTextField(15);   JTextField t2 = new JTextField(15);
  JTextField t3 = new JTextField(15);

  JPanel p = new JPanel();
  JLabel result = new JLabel();
  result.setHorizontalAlignment(JLabel.CENTER);
```

20

```
public RentCar(){
    super("RENT A CAR");
    Container c = getContentPane();
    c.setLayout(new GridLayout(2,1));
    c.add(p);      c.add(result);

    p.setLayout(new GridLayout(3,2));
    p.add(l1); p.add(t1);      p.add(l2); p.add(t2);      p.add(l3); p.add(t3);

    //connect the forms and assign keyboard shortcuts
    l1.setLabelFor(t1); l1.setDisplayedMnemonic('E');
    l2.setLabelFor(t2); l2.setDisplayedMnemonic('N');
    l3.setLabelFor(t3); l3.setDisplayedMnemonic('C');

    //connect the listener
    t1.addActionListener(this);   t2.addActionListener(this);   t3.addActionListener(this);

    pack();
    setVisible(true);
    setDefaultCloseOperation(EXIT_ON_CLOSE);
}
```

21

```
  public void actionPerformed(ActionEvent e){ // Listener method
    String input = t1.getText();   // read exchange rate
    double exchange = Double.parseDouble(input);

    input = t2.getText();  // read number of days
    int noOfDays = Integer.parseInt(input);

     input = t3.getText();  // read cost per day
    double costPerDay = Double.parseDouble(input);
    double totalCost = noOfDays * costPerDay;

    NumberFormat nf = NumberFormat.getInstance();
    nf.setMinimumFractionDigits(2); // 2 decimals

    String output = "Total cost: $" + nf.format(totalCost) +
                        "( \u20AC" + nf.format(totalCost*exchange) + ")";
    result.setText(output);
  }
  public static void main (String[] args){
    RentCar rc = new RentCar();
  }
}
```

22

Lab Assignment

- http://www.csun.edu/~twang/Java/Lab/8-14-Lab.pdf

23