Announcements

Next Owl assignment due friday, ch 13 up - the end…
Programming assignment 8 up - will be a “work with a friend” assignment. An alternate assignment also possible..

Final exam: 12/19, 8 AM

Today: Graphics and the event model II - the mouse!
import java.awt.*; import javax.swing.*;

public class DisplayWindow extends JFrame{

    private Container c;

    public DisplayWindow(){
        super("Display");
        c = this.getContentPane();
    }

    public void addPanel(JPanel p){
        p.setPreferredSize(new Dimension(500,400));
        c.add(p);
    }

    public void showFrame(){
        this.pack();
        this.setVisible(true);
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}

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

public class TryoutPanel extends JPanel{

private Color myColor;

public TryoutPanel(Color c){
    myColor = c;
}
}
Java’s event model

What modern computing is like..

What Scanner-based interactions are like.. (a script)

Your program needs to know:

1) What to listen for

2) Clarify who’s listening

3) Provide a mechanism for reacting properly to events
import java.awt./*;

public class BabyButtonDriver{

    public static void main(String[] args){
        DisplayWindow d = new DisplayWindow();
        BabyControlPanel p = new BabyControlPanel();
        d.addPanel(p);
        d.showFrame();
    }
}
import java.awt.*; import javax.swing.*;
import java.awt.event.*; // needed for event handling

public class BabyControlPanel extends JPanel implements ActionListener{
    JButton quit = new JButton("Quit"); // make button object

    public BabyControlPanel(){
        setBackground(Color.red);
        this.add(quit); // place button in panel
        quit.addActionListener(this); // panel is listener for button
    }

    public void actionPerformed(ActionEvent e){ // what happens
        if (e.getSource() == quit) // when panel hears button
            System.exit(0);
    }
}

public class BabyControlPanel2 extends JPanel implements ActionListener{

    JButton quit = new JButton("Quit"); // make button object

    JButton blue = new JButton("Blue"); // button to change color

    public BabyControlPanel2(){ // constructor
        setPreferredSize(new Dimension(700,300));
        setBackground(Color.red);
        this.add(quit); // place button in panel
        this.add(blue); // add blue button
        quit.addActionListener(this); // panel is listener for button
        blue.addActionListener(this); // panel listener for blue button
    }
}
public void actionPerformed(ActionEvent e) { // what happens
    if (e.getSource() == quit) // when panel hears quit button
        System.exit(0);
    if (e.getSource() == blue) // when panel hears blue button
        setBackground(Color.blue);
}
Now we’re going to start working directly with the mouse: FUN

Here’s a simple program - as you’ve seen in the book - that writes out locations when you click the mouse
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;

public class LineClicker extends JPanel{
    implements MouseListener{
        int oldX = Integer.MIN_VALUE;
        int oldY = Integer.MIN_VALUE;
        int newX = 0;
        int newY = 0;

        public LineClicker(){
            addMouseListener(this);
        }
    }
}
public void paintComponent(Graphics g){
    g.drawLine(oldX, oldY, newX, newY);
}

public void mouseClicked(MouseEvent e){
    newX = e.getX();
    newY = e.getY();
    repaint();
}

public void mouseEntered(MouseEvent e){}
public void mouseExited(MouseEvent e){}
public void mousePressed(MouseEvent e){}
public void mouseReleased(MouseEvent e){}
The main points:

• **MouseListener** - not **ActionListener**

• The listener registration code is different

• There are five methods in the interface - but here we’re only using 1!

• Another interface: **MouseMotionListener** (deals with mouse motion, not just button pressing..)
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;

public class Mouse2 extends JPanel implements MouseMotionListener{

    int oldX = 0; int oldY = 0;
    int newX = 0; int newY = 0;

    public Mouse2(){
        addMouseMotionMotionListener(this);
    }
}
public void paintComponent(Graphics g) {
    g.drawLine(oldX, oldY, newX, newY);
}

public void mouseDragged(MouseEvent e) {
    oldX = newX; oldY = newY;
    newX = e.getX(); newY = e.getY();
    repaint();
}

public void mouseMoved(MouseEvent e) {}
Layouts

Classes that allow you to organized GUI components physically

FlowLayout

GridLayout

BorderLayout

Another deeper point: Panels can be nested (that is, embedded in other panels)

Following the lead of the text, we’ll do a dummy calculator: +,-,.*,/, and four other components
public FlowDisplay()
{
    setLayout(new FlowLayout());
    add(plus);
    add(minus);
    add(times);
    add(divide);
    add(enterLabel);
    add(enter);
    add(result);
    add(quit);
}

Grid Layout
public GridDisplay()
{
    setLayout(new GridLayout(4,2));
    add(plus);
    add(minus);
    add(times);
    add(divide);
    add(enterLabel);
    add(enter);
    add(result);
    add(quit);
}

<table>
<thead>
<tr>
<th>North</th>
<th>Center</th>
<th>East</th>
</tr>
</thead>
<tbody>
<tr>
<td>West</td>
<td></td>
<td></td>
</tr>
<tr>
<td>South</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Border Layout Zones
Border Layout
import java.awt.*;
import javax.swing.*;

public class BorderDisplay extends JPanel {

    JButton plus = new JButton("+");
    JButton minus = new JButton("-");
    JButton times = new JButton("x");
    JButton divide = new JButton("/");
    JButton quit = new JButton("Quit");
    JLabel enterLabel = new JLabel("enter: ");
    JTextField enter = new JTextField(8);
    JLabel result = new JLabel("0.0");

    JPanel northPanel = new JPanel();
    JPanel centerPanel = new JPanel();
public BorderDisplay()
{
    setLayout(new BorderLayout);
    northPanel.add(plus);
    northPanel.add(minus);
    northPanel.add(times);
    northPanel.add(divide);
    add(northPanel,BorderLayout.NORTH);
    centerPanel.add(enterLabel);
    centerPanel.add(enter);
    centerPanel.add(result);
    add(centerPanel,BorderLayout.CENTER);
    add(quit,BorderLayout.SOUTH);
}
}
MouseStretch extends JPanel implements ActionListener, Item

CenterPanel centerPanel = new CenterPanel();

Private class CenterPanel extends JPanel implements MouseListener

void mousePressed..

void mouseReleased..

void paintComponent..
public class MouseStretch extends JPanel implements ItemListener, ActionListener {

    private ImageIcon mao = new ImageIcon("Mao.png");
    private ImageIcon arches = new ImageIcon("Arches.png");
    private ImageIcon ada = new ImageIcon("Ada.png");
    ImageIcon which = mao;
    int width = 0; int height = 0; JButton quit = new JButton("Quit");

    CheckboxGroup PixGroup = new CheckboxGroup();
    Checkbox MaoBox = new Checkbox("Mao", true, PixGroup);
    Checkbox ArchesBox = new Checkbox("Arches", false, PixGroup);
    Checkbox AdaBox = new Checkbox("Ada", false, PixGroup);

    JPanel northPanel = new JPanel();
public MouseStretch()
{
    setLayout(new BorderLayout());
    add(quit, BorderLayout.SOUTH);
    quit.addActionListener(this);

    CenterPanel centerPanel = new CenterPanel();
    northPanel.add(MaoBox);
    MaoBox.addItemListener(this);
    northPanel.add(ArchesBox);
    ArchesBox.addItemListener(this);
    northPanel.add(AdaBox);
    AdaBox.addItemListener(this);
    add(northPanel, BorderLayout.NORTH);
    add(centerPanel, BorderLayout.CENTER);
}
private class CenterPanel extends JPanel implements MouseListener {

    // (x1, y1), (x2, y2), are corners of rectangle that will hold
    // current image

    int x1 = 0; int y1 = 0; int x2 = 0; int y2 = 0;

    private CenterPanel() {
        addMouseListener(this);
    }

    public void mousePressed(MouseEvent e) {
        x1 = e.getX();
        y1 = e.getY();
    }
}
```java
public void mouseReleased(MouseEvent e){
    x2 = e.getX(); y2 = e.getY();
    repaint();
}

public void paintComponent(Graphics g){
    if(x1 != 0){
        width = x2-x1;
        height = y2-y1;
        g.drawImage(which.getImage(),x1,y1,width, height,this);
    }
}

public void mouseEntered(MouseEvent e){}
public void mouseExited(MouseEvent e){}
public void mouseClicked(MouseEvent e){}
} // end of embedded class
} // end of main class
```