
Chapter 8

Events - Lecture Slides

Events

- An *event* is when
 -
 -
 -
 -
 -
 -
- We can choose to respond to events
- Usually events occur from the user (e.g. selected a button, moving the mouse) but other events can occur too (e.g. timer expires, thread finishes)

Events and Listeners

- Components can generate (fire) events based on user interactions
 -
 -
- We designate that our applet wants to *listen* to the events we care about, so we can respond appropriately
 - Button clicks
 - Mouse moves
 - Checkbox selected

4 Things to get Events to Work

1. Import the events package

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

2. Implement the listener(s)

```
public class X extends JApplet
```

3. Add listeners to objects

```
button.addActionListener(this);
```

4. Write the required listener methods

```
public void actionPerformed( ActionEvent ae )
```

Event Listeners

ActionListener	button clicks
ItemListener	
ListSelectionListener	selecting from JList
KeyListener	any key on the keyboard pressed
MouseListener	mouse entered the component, exited the component, mouse button pressed, mouse button released, mouse button clicked
MouseMotionListener	
FocusListener	on JTextField, JTextArea when user's focus leaves the component

Required methods for Listeners

ActionListener	<code>public void actionPerformed((ActionEvent ae)</code>
ItemListener	<code>public void itemStateChanged(ItemEvent ie)</code>
ListSelectionListener	<code>public void valueChanged(ListSelectionEvent e)</code>
KeyListener	<code>public void keyReleased(KeyEvent ke)</code> <code>public void keyTyped(KeyEvent ke)</code> <code>public void keyPressed(KeyEvent ke)</code>
FocusListener	<code>public void</code>
MouseEvent	<code>public void mouseClicked(MouseEvent me)</code> <code>public void mousePressed(MouseEvent me)</code> <code>public void mouseReleased(MouseEvent me)</code> <code>public void mouseEntered(MouseEvent me)</code> <code>public void mouseExited(MouseEvent me)</code>
MouseMotionListener	<code>public void mouseMoved(MouseMovedEvent me)</code> <code>public void mouseDragged(MouseMovedEvent me)</code>

getSource()

- Find out who caused the event – was it the JButton was clicked, JCheckBox selected, etc.
- For any event type (ActionEvent, ItemEvent, etc)

```
JButton goButton, addButton
```

```
public void actionPerformed((ActionEvent event) )  
{  
    Object o = event.getSource( );  
    if ( o == goButton )  
        ...  
    else if ( o == addButton )  
        ...  
}
```

ActionEvents

ActionEvent

- Occurs when
 -
 -
 -
- Methods:
 - `getActionCommand()`
 - if button, get the label
 - if list, get text of selected item
 - if textfield, get the contents of textfield
- Listener method:

JButton Example

- When the user presses a button, the button component creates an `ActionEvent` object and calls the `actionPerformed` method of the listener
- The `actionPerformed` method is where you handle what you want to do when the button is clicked

JButton ActionListener Example

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class IfButton extends JApplet
    implements ActionListener
{
    JButton go, stop;
    JTextArea textarea;
    public void init( )
    {
        setLayout( new FlowLayout( ) );
        setupButtons( );
        textarea = new JTextArea( 3,10);
        add( textarea );
    }
    public void setupButtons( )
    {
        go = new JButton( "Go!" );
        stop = new JButton( "Stop!" );
        //add events
        go.addActionListener( this );
        stop.addActionListener( this );
        add( go ); // add to applet
        add( stop );
    }
}
```



```
public void actionPerformed((ActionEvent ae) )
{
    Object obj = ae.getSource( );
    if( obj == go )
    {
        textarea.setText( "Go GO GOOO!!!!" );
    }
    else if ( obj == stop )
    {
        textarea.setText( "STOP." );
    }
}
```

ActionEvent Example - Clicking a JButton changes the Image



```

import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class ButtonImages extends JApplet
    implements ActionListener
{
    JButton windowButton, statueButton,
    flowerButton;
    Image windowImg, statueImg, flowerImg;
    ImageIcon icon;
    JLabel imagelabel;
    JPanel butpanel;
    public void init( )
    {
        setLayout( new BorderLayout( ) );
        setupButtons( );
        windowImg=getImage(getCodeBase( ),
                           "ChinaWindow.jpg");
        statueImg=getImage(getCodeBase( ),
                           "Statue.jpg");
        flowerImg=getImage(getCodeBase( ),
                           "flowers.jpg");
        icon = new ImageIcon( );
        imagelabel=new JLabel( icon, JLabel.CENTER);
        add( imagelabel, BorderLayout.CENTER );
        setupImage( windowImg );
    }
    public void setupImage( Image img )
    {
        icon.setImage( img );
        imagelabel.setIcon( icon );
        repaint( );
    }
}

```

```

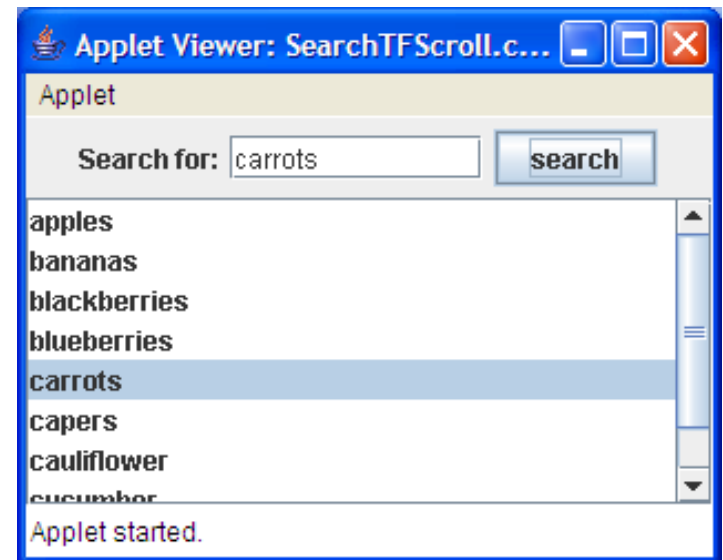
public void setupButtons( )
{
    windowButton = new JButton( "China" );
    statueButton = new JButton( "Germany" );
    flowerButton = new JButton( "Slovakia" );
    windowButton.addActionListener( this );
    statueButton.addActionListener( this );
    flowerButton.addActionListener( this );

    butpanel=new JPanel(new GridLayout(3,1));
    butpanel.add( windowButton );
    butpanel.add( statueButton );
    butpanel.add( flowerButton );
    add( butpanel, BorderLayout.WEST );
}
public void actionPerformed(ActionEvent ae)
{
    Object source = ae.getSource( );
    if( source == windowButton )
        setupImage( windowImg );
    else if( source == statueButton )
        setupImage( statueImg );
    else if( source == flowerButton )
        setupImage( flowerImg );
}

```

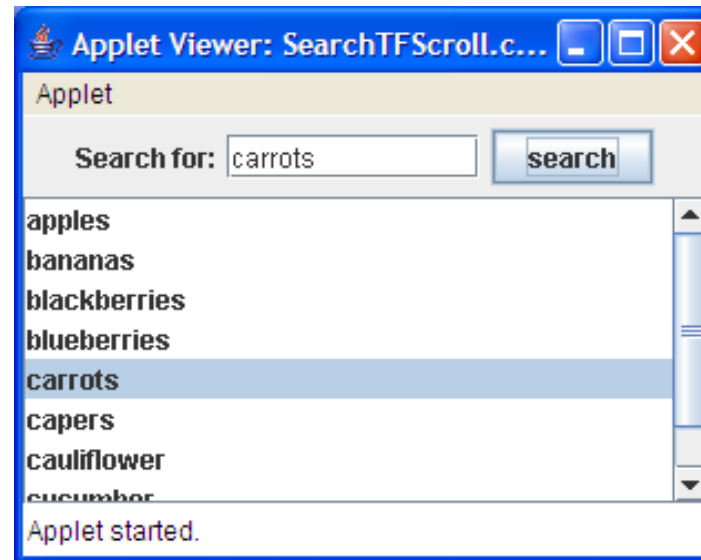
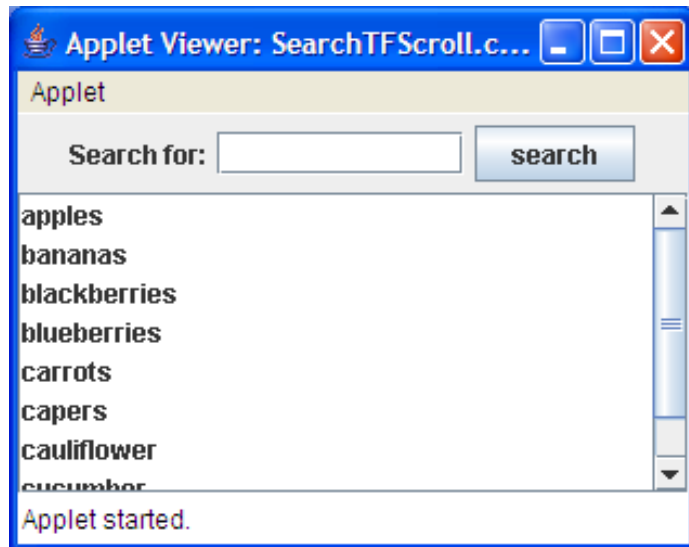


ActionEvent Example - Searching a JList



Searching a JList

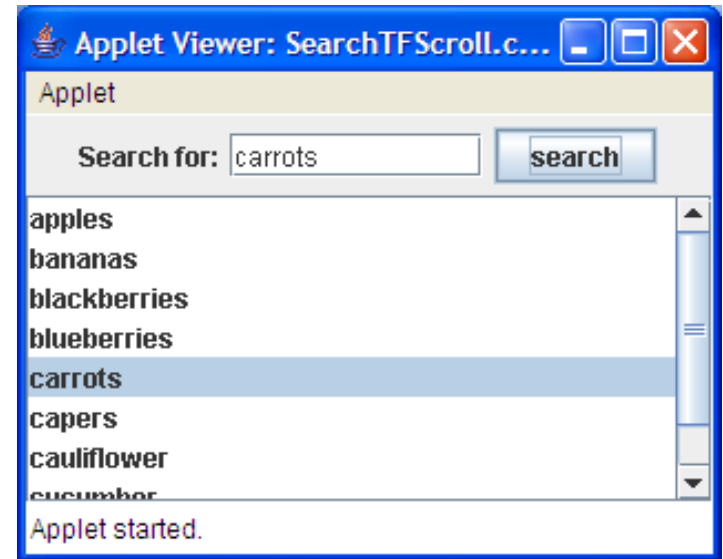
- User enters text to search for in `JTextField`
- When user select 'search' button, our code will go through the items in the list and highlight the match



Searching a JList

```
JTextField searchField;  
JPanel searchPanel;  
JButton searchButton;  
JList list;
```

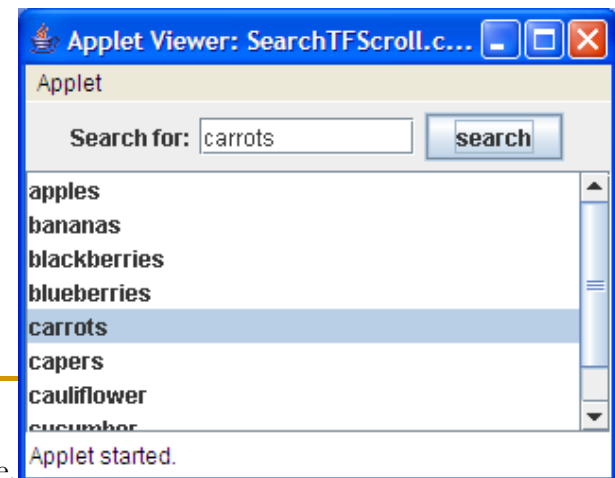
```
public void actionPerformed (ActionEvent e)  
{  
    String field=searchField.getText( );  
    if (field != null)  
        findNode(field);  
}  
public void findNode(String field)  
{  
    list.clearSelection( );  
    list.setSelectedValue(field, true);  
}
```



Example

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
public class SearchTF extends JApplet
    implements ActionListener
{
    JLabel searchLabel;
    JTextField searchField;
    JPanel searchPanel;
    JButton searchButton;
    JList list;
    DefaultListModel model;
    JScrollPane scrollPane;
    public void init( )
    {
        setLayout( new BorderLayout( ) );
        searchLabel = new JLabel("Search for:");
        searchField = new JTextField(10);
        searchPanel = new JPanel();
        searchButton = new JButton("search");
        model = new DefaultListModel( );
        list = new JList( model );
        addListItems( );
        scrollPane = new JScrollPane(list);
        searchButton.addActionListener( this );
        searchPanel.add(searchLabel);
        searchPanel.add(searchField);
        searchPanel.add(searchButton);
        add(scrollPane, BorderLayout.CENTER);
        add(searchPanel, BorderLayout.NORTH);
    }
}
```

```
public void addListItems( )
{
    model.addElement( "apples" );
    model.addElement( "bananas" );
    model.addElement( "blackberries" );
    model.addElement( "blueberries" );
    model.addElement( "carrots" );
    model.addElement( "capers" );
    model.addElement( "cauliflower" );
    model.addElement( "cucumber" );
    model.addElement( "kiwifruit" );
}
public void actionPerformed (ActionEvent e)
{
    String field=searchField.getText( );
    if ( field != null )
        findNode( field );
}
public void findNode( String thefield )
{
    list.clearSelection( );
    list.setSelectedValue( thefield, true );
}
}
```



ItemEvents

ItemEvent

- Select/De-select

-
-
-
-

- Methods:

- `getStateChanged()` :: returns value of state
 - `ItemEvent.SELECTED`
 - `ItemEvent.DESELECTED`

- Listener:

```
public void itemStateChanged( ItemEvent ie )
```

ItemEvents

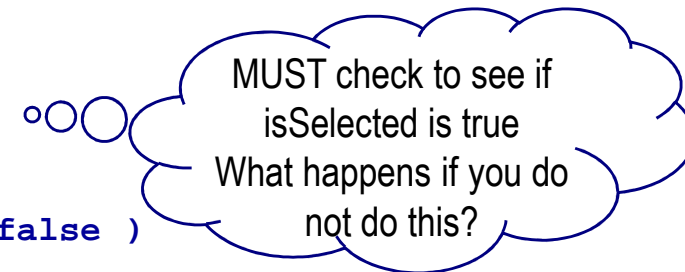


JCheckBox Example



Item Event Example - JCheckBox

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class CheckEvent extends JApplet
    implements ItemListener
{
    JCheckBox isHappy;
    JTextArea textarea;
    public void init( )
    {
        setLayout( new FlowLayout( ) );
        isHappy = new JCheckBox( "Happy?", true );
        textarea = new JTextArea( 3,10);
        // add listener
        isHappy.addItemListener( this );
        add( isHappy );
        add( textarea );
    }
    public void itemStateChanged( ItemEvent ie )
    {
        Object source = ie.getSource( );
        if( source == isHappy && isHappy.isSelected( ) )
            textarea.setText( "You're so happy!" );
        else if (source == isHappy && isHappy.isSelected( ) == false )
            textarea.setText( "Why not happy?" );
    }
}
```



Item Event Example (another way)

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class CheckEvent extends JApplet implements ItemListener
{
    JCheckBox isHappy;
    JTextArea textarea;
    public void init( )
    {
        setLayout( new FlowLayout( ) );
        isHappy = new JCheckBox( "Happy?", true );
        textarea = new JTextArea( 3,10);
        // add listener
        isHappy.addItemListener( this );
        add( isHappy );
        add( textarea );
    }
    public void itemStateChanged( ItemEvent ie )
    {
        Object source = ie.getSource( );
        if ( ie.getStateChanged( ) == ItemEvent.SELECTED )
        {
            if( source == isHappy )
                textarea.setText( "You're so happy!" );
        }
        else // item was deselected
        {
            if (source == isHappy )
                textarea.setText( "Why not happy?" );
        }
    }
}
```



ItemEvents



JCheckBox Complex Example

- 2 checkboxes
- Respond based on recent selection

JCheckBox – More Complex Example

```
public void itemStateChanged( ItemEvent ie )
{
    Object source = ie.getSource( );
    if ( source == isHappy  && isHappy.isSelected( ) )
        textarea.setText( "You're so happy!" );
    else if ( source == isHappy  && ! isHappy.isSelected( ) )
        textarea.setText( "Why not happy?" );

    if ( source == havingFun && havingFun.isSelected( ) )
        textarea.setText( "WOO HOO" );
    else if ( source == havingFun && havingFun.isSelected( ) ==
false )
        textarea.setText("let's go skiing" );
}
```



JCheckBox – More Complex Example

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class CheckEvent2 extends JApplet implements ItemListener
```

```
{
    JCheckBox isHappy, havingFun;
    JTextArea textarea;
    public void init( )
    {
        setLayout( new FlowLayout( ) );
        isHappy = new JCheckBox( "Happy?", true );
        havingFun = new JCheckBox( "Fun?" );
        textarea = new JTextArea( 3,10);
            // add listener
        isHappy.addItemListener( this );
        havingFun.addItemListener(this );
        add( isHappy );
        add( havingFun );
        add( textarea );
    }
}
```

```
public void itemStateChanged( ItemEvent ie )
{
    Object source = ie.getSource( );
    if ( source == isHappy && isHappy.isSelected( ) )
        textarea.setText( "You're so happy!" );
    else if ( source == isHappy && ! isHappy.isSelected( ) )
        textarea.setText( "Why not happy?" );

    if ( source == havingFun && havingFun.isSelected( ) )
        textarea.setText( "WOO HOO" );
    else if ( source == havingFun && havingFun.isSelected( ) == false )
        textarea.setText("let's go skiing" );
}
}
```



ItemEvents

JRadioButton Example



Item Event Example - JRadioButton

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class JRadioEvent extends JApplet
    implements ItemListener
{
    ButtonGroup grp = new ButtonGroup();
    JRadioButton red, fuscia, pink;
    JTextArea ta = new JTextArea( 5, 10 );

    public void init()
    {
        setLayout( new FlowLayout( ) );
        setupButtons();
        addListeners();
        add( red );
        add( fuscia );
        add( pink );
        add( ta );
    }
    public void addListeners()
    {
        red.addItemListener( this );
        fuscia.addItemListener( this );
        pink.addItemListener( this );
    }
}
```

```
public void setupButtons()
{
    red = new JRadioButton( "red" );
    fuscia = new JRadioButton( "fuscia" );
    pink = new JRadioButton( "pink" );
    grp.add( red );
    grp.add( fuscia );
    grp.add( pink );
}
public void itemStateChanged( ItemEvent ie )
{
    Object obj = ie.getSource();
    if ( obj == red && red.isSelected() )
        ta.append( "we like RED \n" );
    else if ( obj == fuscia && fuscia.isSelected() )
        ta.append( "We like fuscia \n" );
    else if ( obj == pink && pink.isSelected() )
        ta.append( "We prefer pink \n" );
}
```



Item Event – Selection Alternative Implementation

```
public void itemStateChanged( ItemEvent ie )
{
    Object o = ie.getSource( );
    if ( ie.getStateChange( ) == ItemEvent.DESELECTED )
        return; // ignore if deselecting
    if( o == red )
        ta.append( "we like RED \n" );
    else if( o == fuchsia )
        ta.append( "We like fuchsia \n" );
    else if( o == pink )
        ta.append( "We prefer pink \n" );
}
```

ItemEvents

JComboBox Example



Item Event Example - JRadioButton

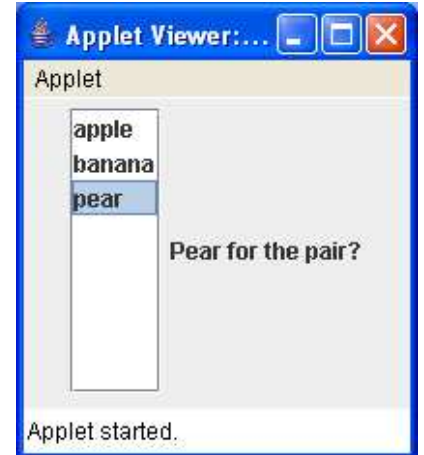
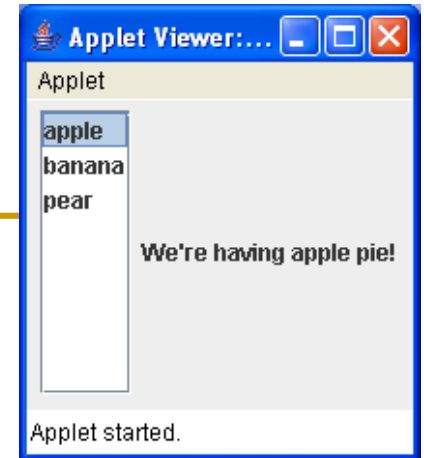
```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class JComboBoxEvent
    extends JApplet
    implements ItemListener
{
    JComboBox list;
    JLabel dessert;
    public void init( )
    {
        list = new JComboBox( );
        list.addItem( "apple" );
        list.addItem( "banana" );
        list.addItem( "pear" );
        list.addItemListener( this );
        dessert = new JLabel( );
        setLayout( new FlowLayout( ) );
        add( list );
        add( dessert );
    }
}
```

```
public void itemStateChanged( ItemEvent ie )
{
    int index = list.getSelectedIndex( );
    if ( index == 0 )
        dessert.setText( "We're having apple pie!" );
    else if ( index == 1 )
        dessert.setText( "Banana split anyone?" );
    else if ( index == 2 )
        dessert.setText( "Pear for the pair?" );
}
```



ListSelectionEvents

JList Example



ListSelectionEvent Example - JList

```
import java.awt.*;
import javax.swing.*;
import javax.swing.event.*;
public class JListEvent extends JApplet
    implements ListSelectionListener
{
    JList list;
    DefaultListModel model;
    JLabel dessert;
    public void init( )
    {
        model = new DefaultListModel( );
        list = new JList( model );
        model.addElement( "apple" );
        model.addElement( "banana" );
        model.addElement( "pear" );
        list.addListSelectionListener( this );
        JScrollPane pane = new JScrollPane( list );
        dessert = new JLabel( );
        setLayout( new FlowLayout( ) );
        add( pane );
        add( dessert );
    }
}
```

```
public void valueChanged( ListSelectionEvent le )
{
    int index = list.getSelectedIndex( );
    if ( index == 0 )
        dessert.setText( "We're having apple pie!" );
    else if ( index == 1 )
        dessert.setText( "Banana split anyone?" );
    else if ( index == 2 )
        dessert.setText( "Pear for the pair?" );
}
```



KeyEvents

KeyEvent

- Occurs when key is typed on keyboard
- Methods:
 - `getKeyCode()` :: returns the identifier for which key was pressed
- Listener
 - `KeyListener`
 - `KeyListenerAdapter`
- Listener Methods:

```
public void keyPressed( KeyEvent ke )  
public void keyReleased( KeyEvent ke )  
public void keyTyped( KeyEvent ke )
```

- Need to set focus to enable listening for key events

KeyEvent

■ Keys

- ❑ Up arrow KeyEvent.VK_UP
- ❑ Down arrow KeyEvent.VK_DOWN
- ❑ Left arrow KeyEvent.VK_LEFT
- ❑ Right arrow KeyEvent.VK_RIGHT
- ❑ 0 KeyEvent.VK_0
- ❑ 1 KeyEvent.VK_1
- ❑ 2 KeyEvent.VK_2
- ❑ A KeyEvent.VK_A
- ❑ B KeyEvent.VK_B
- ❑ C KeyEvent.VK_C
- ❑ <enter> KeyEvent.VK_ENTER
- ❑ <tab> KeyEvent.VK_TAB

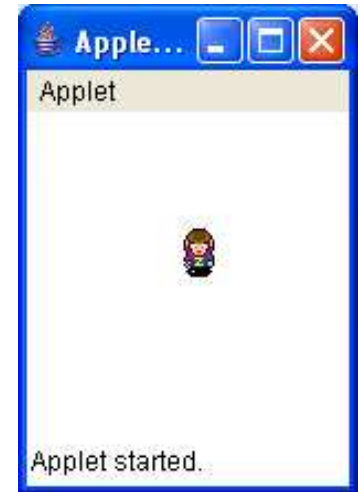
- ❑ (VK stands for Virtual Key)

Key Event Example

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class KeyEventAnimation extends JApplet
    implements KeyListener
{
    Image img;
    JLabel penny;
    // x,y coordinates and # pixels move
    int x=0, y=0, speed=10;
    JPanel pane;
    public void init( )
    {
        pane = new JPanel( );
        pane.setLayout( null );
        pane.setBackground( Color.WHITE );
        img = getImage( getCodeBase( ), "zil.gif" );
        penny = new JLabel( new ImageIcon(img) );
        penny.setSize( img.getWidth(this),
            img.getHeight(this) );

        addKeyListener(this);
        pane.add( penny, 0, 0 );
        add( pane, BorderLayout.CENTER );
        setFocusable(true);
    }
}
```

```
public void keyReleased( KeyEvent ke ) { }
public void keyTyped( KeyEvent ke ) { }
public void keyPressed( KeyEvent ke )
{
    int code = ke.getKeyCode( );
    if ( code == KeyEvent.VK_UP )
        y -= speed;
    else if ( code == KeyEvent.VK_DOWN )
        y += speed;
    else if ( code == KeyEvent.VK_LEFT )
        x -= speed;
    else if ( code == KeyEvent.VK_RIGHT )
        x += speed;
    penny.setLocation(x,y);
}
```



MouseEvents

MouseEvent

- When the mouse
 - Enters component
 - Exits component
 - Button pressed
 - Button clicked
 - Button released

- Methods:
 - `getX()` :: returns the x coordinate
 - `getY()` :: returns the y coordinate

- Listener:

```
public void mouseClicked( MouseEvent me )
public void mousePressed( MouseEvent me )
public void mouseReleased( MouseEvent me )
public void mouseEntered( MouseEvent me )
public void mouseExited( MouseEvent me )
```

Event methods

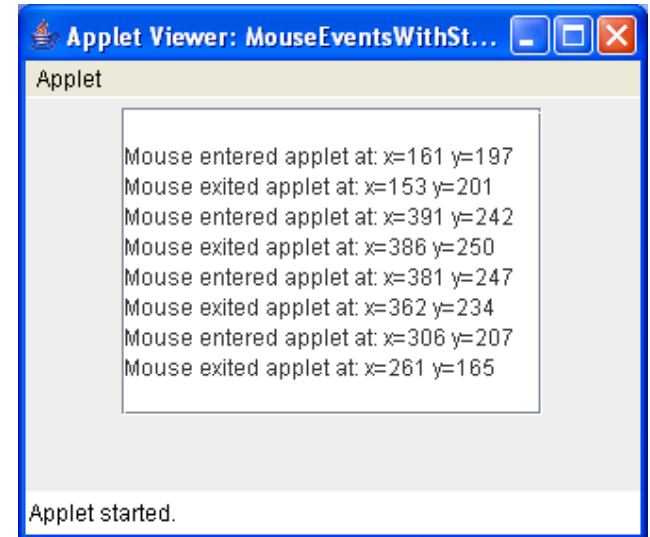
- When implementing the listeners,
- If you don't want to use some of the methods, you still need to code them but you can code them as a `stub`.
- A **stub** is a method with nothing inside.

```
public void mouseExited( MouseEvent me )  
{  
  
  
}
```

Mouse Event example

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class MouseEventsWithStubs extends JApplet implements MouseListener
{
    JTextArea msg;
    JScrollPane spane;
    public void init( )
    {
        addMouseListener( this );
        msg = new JTextArea( 10,20 );
        spane = new JScrollPane(msg);

        setLayout( new FlowLayout( ) );
        add( spane );
    }
    public void mouseEntered( MouseEvent me )
    {
        msg.append( "\nMouse entered applet at: x=" + me.getX( ) + " y=" + me.getY( ) );
    }
    public void mouseExited( MouseEvent me )
    {
        msg.append( "\nMouse exited applet at: x=" + me.getX( ) + " y=" + me.getY( ) );
    }
    public void mousePressed( MouseEvent me )    {    }
    public void mouseReleased( MouseEvent me )   {    }
    public void mouseClicked( MouseEvent me )    {    }
}
```



Mouse Events - Imagemaps

■ Imagemaps

- ❑ Image that the user can click on
- ❑ Depending where the user clicks on the image, a different effect will occur
- ❑ Common for navigation bars, family tree
- ❑ Determine coordinates (x,y) using a paint program



Mouse Event example (con't)

```
public void mouseClicked( MouseEvent me )
{
    int x = me.getX( );
    int y = me.getY( );
    if ( x > 65 && x < 145 && y > 135 && y < 185 )           // Carpool or bike
    {
        details.setText("<HTML><CENTER>Carpooling is an excellent way to save the air."
            + "<BR>Get together with a group of friends and share a ride.<
            + "<BR>Or hop on a bike!<BR>Great exercise and no pollution!");
    }
    else if ( x > 135 && x < 225 && y > 222 && y < 276 )       // Recycle
    {
        details.setText("<HTML><CENTER>RECYCLE!«
            + "<BR>Everybody's doing it.«
            + "<BR>So should YOU!<BR>Check your local garbage collection«
            + " to determine what items can be recycled." );
    }
}
```

Mouse Event example

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class MouseClickImgmap extends JApplet implements MouseListener
{
    Image img;
    ImageIcon icon;
    JLabel map, title;
    JLabel details;
    JPanel center = new JPanel( );
    public void init( )
    {
        setLayout( new BorderLayout( ) );
        setTitle( );
        setupMap( );
        details = new JLabel("<HTML><CENTER>Click on a bubble above<BR>"
            + " to get more info!<BR>", JLabel.CENTER );
        details.setForeground( Color.WHITE );
        details.setBackground( Color.BLACK );
        details.setOpaque( true );
        add( details, BorderLayout.SOUTH );
    }
    public void setTitle( )
    {
        title = new JLabel( "Pollution Solutions", JLabel.CENTER);
        title.setFont( new Font( "Serif", Font.BOLD, 26 ) );
        title.setForeground( Color.WHITE );
        title.setOpaque( true );
        title.setBackground( Color.BLACK );
        add( title, BorderLayout.NORTH );
    }
}
```

Mouse Event example (con't)

```
public void setupMap( )
{
    img = getImage( getCodeBase( ), "PollutionMapSm.jpg" );
    icon = new ImageIcon( img );
    map = new JLabel( icon );
    map.addMouseListener(this);
    center.setBackground( Color.BLACK );
    center.add(map);
    add( center, BorderLayout.CENTER );
}
public void mouseClicked( MouseEvent me )
{
    int x = me.getX( );
    int y = me.getY( );
    if ( x > 65 && x < 145 && y > 135 && y < 185) // Carpool or bike
    {
        details.setText("<HTML><CENTER>Carpooling is an excellent way to save the air and your wallet!"
            + "<BR>Get together with a group of friends and share a ride."
            + "<BR>Or hop on a bike!<BR>Great exercise and no pollution!");
    }
    else if ( x > 135 && x < 225 && y > 222 && y < 276) // Recycle
    {
        details.setText("<HTML><CENTER>RECYCLE!"
            + "<BR>Everybody's doing it."
            + "<BR>So should YOU!<BR>Check your local garbage collection"
            + " to determine what items can be recycled." );
    }
}
}
```

```
public void mouseEntered( MouseEvent me ) { }
public void mouseExited( MouseEvent me ) { }
public void mousePressed( MouseEvent me ) { }
public void mouseReleased( MouseEvent me ) { }
```

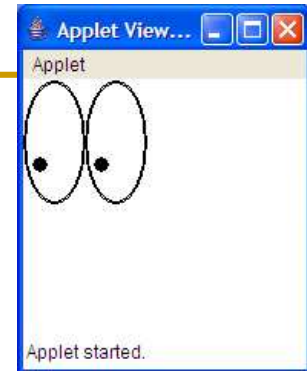
MouseEvent

MouseEvent

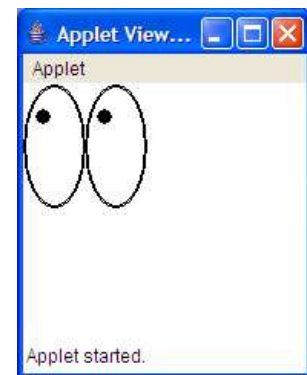
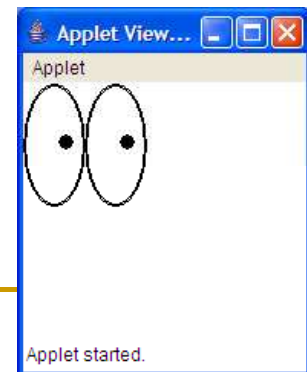
- **Occurs when mouse**
 - Moved
 - Dragged
- **Methods:**
 - `getX()` :: returns the x coordinate
 - `getY()` :: returns the y coordinate
- **Listener**
 - `MouseListener`
- **Listener Methods:**

```
public void mouseMoved( MouseEvent me )  
public void mouseDragged( MouseEvent me )
```

MouseEvent



MouseEyes Example



Mouse Event example

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

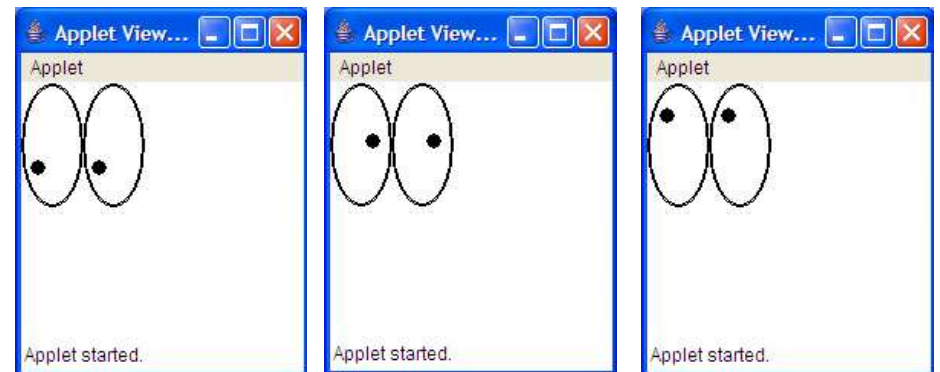
import javax.swing.JApplet;
public class MouseEyes extends JApplet
    implements MouseMotionListener
{
    int mouseX, mouseY;
        // current location of mouse

public void init( )
{
    addMouseMotionListener( this );
}

public void mouseMoved( MouseEvent me )
{
    // track current location of mouse
    mouseX = me.getX();
    mouseY = me.getY();
    repaint( );
}

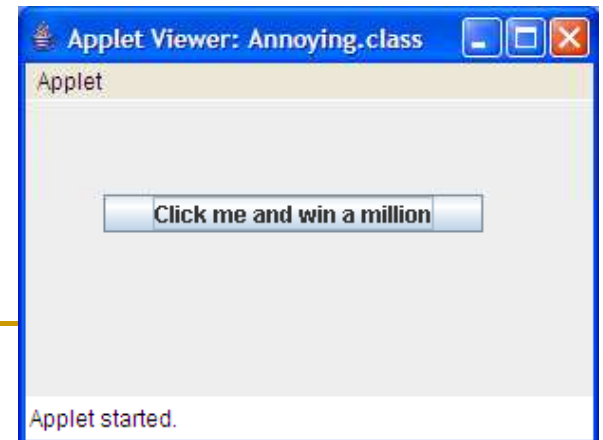
public void mouseDragged( MouseEvent me ) { }
}
```

```
public void paint( Graphics g )
{
    g.clearRect( 0,0,getWidth( ),getHeight( ) );
    int eyeWidth = 40;
    int eyeHeight = 80;
    g.setColor( Color.WHITE ); // color the whites of the eyes
    g.fillOval( 0,0,eyeWidth,eyeHeight );
    g.fillOval( eyeWidth,0, eyeWidth, eyeHeight );
    g.setColor( Color.BLACK ); // draw outline
    g.drawOval( 0,0, eyeWidth, eyeHeight ); // left eye
    g.drawOval( 1,1, eyeWidth-2, eyeHeight-2 );
    g.drawOval( eyeWidth,0, eyeWidth, eyeHeight ); // right eye
    g.drawOval( eyeWidth+1,1, eyeWidth-2, eyeHeight-2 );
        // eyes, x,y as percent of where mouse is
    int eyeX = (mouseX * 100 / getWidth( )) * eyeWidth / 100;
    int eyeY = (mouseY * 100 / getHeight( )) * eyeHeight / 100;
    g.fillOval( eyeX, eyeY, 10,10 );
    g.fillOval( eyeX+eyeWidth, eyeY, 10,10 );
}
}
```



MouseEvent

Annoying Random Example



Random

- Get a pseudo-random number to ‘randomly’ move the button around

```
// need a Random object to randomly select a new x and y coordinates  
// when we move the button to a new location
```

```
Random random = new Random( );  
int newX = random.nextInt(300);  
int newY = random.nextInt( 400 );
```

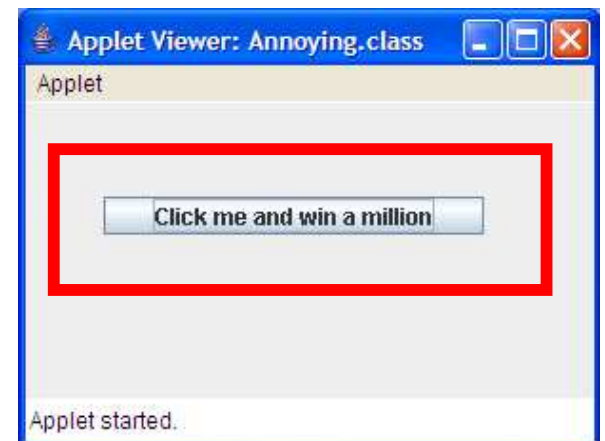
MouseEventExample

- Maintain information about the 'bounding box' surrounding the actual button.
- Whenever the user's mouse gets within the 'bounding box' region, move the button
- 'Bounding box' is invisible to the user
- We'll use a **Rectangle** object to keep track of the (x,y) coordinates and width/height of the 'bounding box' to determine if the mouse coordinates are within the 'bounding box' region.

```
Rectangle rectangle = new Rectangle( );  
rectangle.setBounds( 90,190,220,40);
```

```
if ( rectangle.contains( mouseX, mouseY ) )
```

```
...
```



MouseEventExample

- Use the **null** layout so we can place the button *exactly* where we want.
 - Need to call **.setBounds(x, y, width, height)** for the button to appear

```
JButton winner = new JButton( "Click me and win a million" );  
setLayout( null );  
winner.setBounds(100,200,200,20);
```

Mouse Event example (con't)

```
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;
public class Annoying extends JApplet implements MouseMotionListener
{
    JButton winner = new JButton( "Click me and win a million" );
    Random random = new Random( );
    Rectangle rectangle = new Rectangle( );
    public void init( )
    {
        setLayout( null );
        winner.setBounds(100,200,200,20);
        rectangle.setBounds( 90,190,220,40);    // "too close" boundary
        addMouseMotionListener(this);
        add(winner);
    }
    public void mouseMoved( MouseEvent me )
    {
        int x = me.getX( );
        int y = me.getY( );
        if( rectangle.contains( x, y ) )
        {
            winner.setLocation( random.nextInt(300), random.nextInt(490) );
            // reset the boundary for the rectangle, such that it's 10 pixels
            // buffer from the new location of the button
            rectangle.setBounds( winner.getX()-10, winner.getY()-10,
                winner.getWidth()+20, winner.getHeight()+20 );
        }
    }
    public void mouseDragged( MouseEvent me ) { }
}
```

Tip: Code the looks and actions separately

- For a complicated GUI, break up the work into two parts
 - Code the appearance:
 -
 -
 - Code the actions
 -

Revisit component methods

- Methods to get information the user entered in our forms
 - **JTextField, JTextArea, JPasswordField**
 - `component.getText()` - returns a
 - **JList**
 - `component.getSelectedIndex()` - returns an
 - `component.getSelectedValue()` - returns an
 - `component.getSelectedValues()` - returns an
 - **JComboBox**
 - `component.getSelectedIndex()` - returns an
 - `component.getSelectedItem()` - returns an
 - **JRadioButton, JCheckBox**
 - `component.isSelected()` - returns true or false (boolean)

Handling Object data type

- Methods that return an object
 - **JList**
 - `component.getSelectedValue()` - returns an *object*
 - **JComboBox**
 - `component.getSelectedItem()` - returns an *object*

- Transforming an object
 - In order to use the Object, we need to *cast* it to a type we're familiar with so we can use the methods for the type.
 - To cast to a data type, we put the type in parenthesis before the Object variable
 - Example
 - `String listItemSelected =`
 - `String cmbboxSelected =`
 - If an image was selected, we cast to the `ImageIcon` object
 -

Handling Object data type

- If an image was selected, we cast to the ImageIcon object

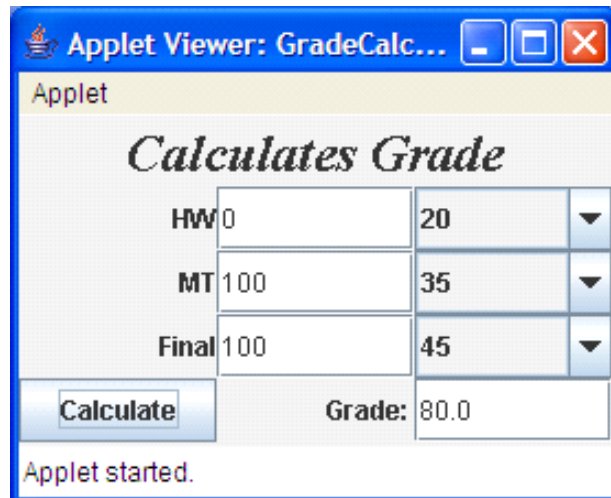
```
□ ImageIcon icon = (ImageIcon) component.getSelectedItemAt( );  
□ if ( icon == myIcon2 )  
    ...  
    else if ( icon == myIcon3 )  
        ...
```

- Full Example

```
Image img2 = getImage( getCodeBase( ), "idea.png" );  
Image img3 = getImage( getCodeBase( ), "house.png" );  
ImageIcon myIcon2 = new ImageIcon( img2 );  
ImageIcon myIcon3 = new ImageIcon( img3 );  
JComboBox cmbx = new JComboBox( );  
cmbx.add( myIcon2 );  
cmbx.add( myIcon3 );  
  
...  
public void itemStateChanged( ItemEvent ie )  
{  
    Object obj = ie.getSource( );  
    if ( obj == cmbx )  
    {  
        JComboBox cb = (JComboBox) obj;  
        ImageIcon ic = (ImageIcon) cb.getSelectedItem( );  
        if ( ic == myIcon2 )  
            ...  
        else if ( ic == myIcon3 )  
            ...  
    }  
}
```

Events - example

■ GradeCalculator



Applet Viewer: GradeCalc... [min] [max] [close]

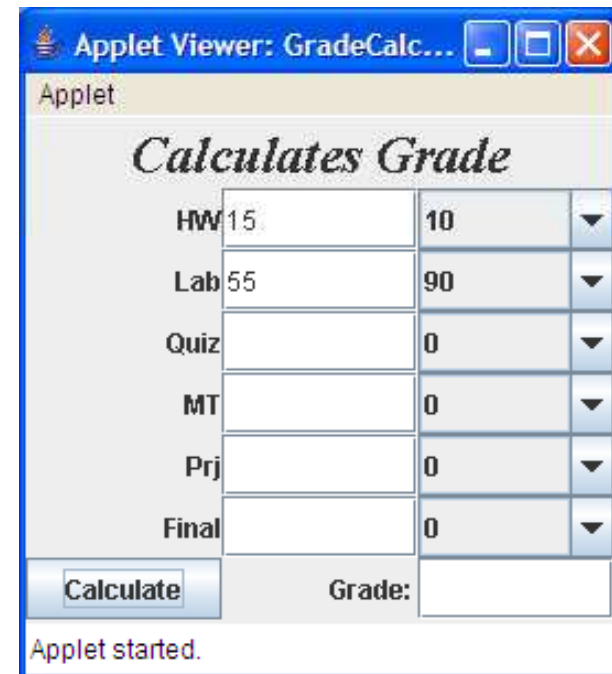
Applet

Calculates Grade

HW	<input type="text" value="0"/>	<input type="text" value="20"/>	▼
MT	<input type="text" value="100"/>	<input type="text" value="35"/>	▼
Final	<input type="text" value="100"/>	<input type="text" value="45"/>	▼

Grade:

Applet started.



Applet Viewer: GradeCalc... [min] [max] [close]

Applet

Calculates Grade

HW	<input type="text" value="15"/>	<input type="text" value="10"/>	▼
Lab	<input type="text" value="55"/>	<input type="text" value="90"/>	▼
Quiz	<input type="text"/>	<input type="text" value="0"/>	▼
MT	<input type="text"/>	<input type="text" value="0"/>	▼
Prj	<input type="text"/>	<input type="text" value="0"/>	▼
Final	<input type="text"/>	<input type="text" value="0"/>	▼

Grade:

Applet started.

Dynamically Changing Components on a JPanel

■ Example:



Changing Components on a JPanel

- Based on an event, you can clear out a panel and put new components in it.
- However, there's a few quirks about how to make this work.
- First,
 - Then in the event method, call the method `removeAll()` on your `JPanel` variable.
 - Then you can add new components to the `JPanel`.
 - Once you're done adding components, call the method `repaint()` and `validate()` on your `JPanel`.

Changing Components on a JPanel (1/3)

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class ClearPanel extends JApplet implements ActionListener
{
    JLabel title;
    JButton productInfo, aboutUs;
    JPanel mainPanel;           // use this JPanel for everything in BorderLayout.CENTER
    JPanel productPanel, aboutUsPanel;
    public void init( )
    {
        setLayout( new BorderLayout( ) );
        title = new JLabel( "<HTML><B><FONT COLOR=RED>Ski Wax For You!", JLabel.CENTER );
        add( title, BorderLayout.NORTH );
        setupButtons();
        setupProductPanel( );
        setupAboutUsPanel( );
        mainPanel = new JPanel( new BorderLayout( ) );
        JLabel tmpMsg = new JLabel( "<HTML><CENTER>Welcome<BR> <BR> to our <BR> <BR> Home!",
            JLabel.CENTER);
        tmpMsg.setFont( new Font( "Serif", Font.BOLD, 18 ) );
        mainPanel.add(tmpMsg, BorderLayout.CENTER);
        add( mainPanel, BorderLayout.CENTER );
    }
}
```

Changing Components on a JPanel (2/3)

```
public void setupAboutUsPanel()  
{  
    aboutUsPanel = new JPanel( new GridLayout( 2, 1 ) );  
    Image img = getImage( getCodeBase(), "pennyWalk.gif" );  
    ImageIcon icon = new ImageIcon( img );  
    aboutUsPanel.add( new JLabel( icon ) );  
    JLabel text = new JLabel( "<HTML><CENTER><B>About Us</B> <BR> <BR>"  
        + "We are a family run business <BR>that has been making wax since 1997."  
        + "<BR>We take pride in our work <BR>and guarantee you will love our wax.", JLabel.CENTER );  
    aboutUsPanel.add( text );  
}  
public void setupProductPanel( )  
{  
    productPanel = new JPanel( new BorderLayout() );  
    JLabel prodinfo = new JLabel( "<HTML><CENTER><B>Product Info</B><BR> <BR>"  
        + "Our ski wax is the best wax you'll ever find.<BR> Comes in two styles:<BR>"  
        + " red: warmer temperatures<BR>blue: colder temperatures", JLabel.CENTER );  
    productPanel.add( prodinfo, BorderLayout.CENTER );  
}  
public void setupButtons()  
{  
    productInfo = new JButton( "Info" );  
    aboutUs = new JButton( "About Us" );  
    productInfo.addActionListener( this );  
    aboutUs.addActionListener( this );  
    JPanel west = new JPanel( new GridLayout( 2,1 ) );  
    west.add( productInfo );  
    west.add( aboutUs );  
    add( west, BorderLayout.WEST );  
}
```

Changing Components on a JPanel (3/3)

```
public void actionPerformed( ActionEvent ae )
{
    Object src = ae.getSource( );
    if ( src == productInfo )
    {
        mainPanel.removeAll( );
        mainPanel.add( productPanel, BorderLayout.CENTER );
    }
    else if ( src == aboutUs )
    {
        mainPanel.removeAll( );
        mainPanel.add( aboutUsPanel, BorderLayout.CENTER );
    }
    mainPanel.validate( );
    mainPanel.repaint( );
}
}
```

Case Study

Applet Viewer: CoffeeClubEvents.class

Applet

Home

Welcome to our Coffee of the Month Club!

*Our coffee club is not just gourmet coffee -
it's an immersion of experience*

Join the Club

We personally select gourmet coffees from around the world. Each month a freshly roasted coffee is sent to your door along with info page about the coffee and other goodies.

FAQ

Each month we send you

- 12 oz. bag of fresh coffee beans
- Information sheets about the coffee and region
- Regional Gift each month (e.g. spices, nuts, chocolates, ...)

Contact Us

FREE SHIPPING!

Buy NOW and receive
a FREE
leather binder
for collecting the fact sheets!

(c) 2008 by CoffeeClubOfTheWorld.com. All Rights Reserved.

Applet started.

Applet Viewer: CoffeeClubEvents.class

Applet

Home

FAQ

Join the Club

Can I get decaffeinated coffee?

No. Our selections are all caffeinated.

Can I get ground coffee instead of the beans?

No. Coffee will stay fresher as whole beans than if pre-ground. This gives you the best flavors to experience.

FAQ

Can I choose which coffees I get?

No. Each month we send the same coffee package to all members.

Contact Us

How do I store the coffee?

Coffee should be kept in a cool place in an air-tight container. Do not freeze your coffee!

(c) 2008 by CoffeeClubOfTheWorld.com. All Rights Reserved.

Applet started.

Summary

■ Events

- ❑ ActionEvents with ActionListener
- ❑ ItemEvents with ItemListener
- ❑ ListSelectionEvents with ListSelectionListener
- ❑ MouseEvents with MouseListener
- ❑ MouseMotionEvents with MouseMotionListener
- ❑ KeyEvents with KeyListener
- ❑ Dynamically Changing Components on a JPanel