

Mouse & Keyboard Processing

2 ways to add a handler

1: Create an inner class(like eventClass). Create an object of that class (handler). Attach to control.

```
public class mainClass {  
    public mainClass()  
    {  
        JButton helloButton = new JButton( "Hello");  
        eventClass handler = new eventClass();  
        helloButton.addActionListener( handler );  
    }  
  
    private class eventClass implements ActionListener //private inner class  
    {  
        public void actionPerformed ( ActionEvent e )  
        {  
            System.out.println( e.getActionCommand() );  
        }  
    }  
}  
// end mainClass
```

2. Make your mainClass the handler. Attach using "this".

```
public class mainClass extends JFrame implements  
    MouseListener, MouseMotionListener  
{  
    public mainClass ()  
    {  
        // contentPane, container, FlowLayout, etc.  
        addMouseListener( this );  
        addMouseMotionListener( this );  
    }  
    // now add methods required by MouseListener and  
    // MouseMotionListener  
}
```

Mouse Interface Obligations

- mouseClicked (MouseEvent m)
- mousePressed (MouseEvent m)
- mouseReleased (MouseEvent m)
- mouseEntered(MouseEvent m)
- mouseExited(MouseEvent m)
- mouseDragged (MouseEvent m)
- mouseMoved(MouseEvent m)

MouseEvent object methods

mouseClicked(MouseEvent m)

m.getX()
m.getY()

KeyListener

```
public yourClass extends JFrame implements  
KeyListener, MouseListener  
{  
    public void keyPressed( KeyEvent event )  
    {  
        System.out.println(event.getKeyCode( ));  
    }  
}
```

Key Listener obligations

- `keyPressed(KeyEvent event)`
as key is pressed
- `keyTyped(KeyEvent event)`
repeats as key is held
- `keyReleased(KeyEvent event)`
as key is released

KeyEvent object methods

`keyPressed(KeyEvent event)`

`event.getKeyChar()` – actual letter typed

`event.getKeyCode()` - # on keyboard

`event.getKeyText(event.getKeyCode())` -
description

`event.isActionKey()`

JFrame method addKeyListener

- `addKeyListener(handler);`
if handler is an object of private class
- `addKeyListener(this);`
if class itself implement keyListener

What's an adapter class?

- An attempt to thwart an interface.
 - Provides a blank method for each one dictated by the interface, none final.
 - Lets you override only the ones that you want.
- e.g. `new MouseMotionAdapter()` lets you write a class that extends `MouseMotionAdapter` with only `mouseDragged()` overridden.

JFrame method addMouseMotionListener

```
addMouseMotionListener ( adapter class  
here );
```

An anonymous, adapter class

```
addMouseMotionListener( // must specify a class that implements  
                        // mouseMotionListener  
// anonymous inner adapter class  
new MouseMotionAdapter() {  
  
    // can re-write any mouse method here  
    public void mouseDragged( MouseEvent event )  
    {  
        xValue = event.getX();  
        yValue = event.getY();  
        repaint();  
    }  
} // end anonymous inner class  
); // end call to addMouseMotionListener
```

- no object, just adds a method to `MouseMotionListener`
