```
package gameutil;

import java.util.*;
import java.awt.*;
import java.awt.image.*;
import javax.swing.*;
import java.awt.event.*;

public abstract class ListeningGameComponent extends GameComponent implements
MouseListener, MouseMotionListener, KeyListener
{
    public boolean mousePressed1 = false, mousePressed2 = false, mousePressed3
    = false;
    public ArrayList<String> keysPressed = new ArrayList();
    public boolean debug = false;

    /**
     * The x location of the mouse.
     */
    public int mouseX = 0;
    /**
     * The y location of the mouse.
     */
    public int mouseY = 0;

    /**
     * Constructs a ListeningGameComponent with a width of w, and a height of
     h.
     *
     * @param  w  width
     * @param  h  height
     * @see         JPanel
     */
    public ListeningGameComponent(int w, int h)
    {
        super(w,h);
        addMouseListener(this);
        addMouseMotionListener(this);
        addKeyListener(this);
    }

    /**
     * The method that draws the component.
     *
     * @param   g   the {@link Graphics} on which the component will be drawn
     */
    public abstract void draw(Graphics g);


    /**
     * The method that updates the component.
     */
    public abstract void update();

    /**
     * Does nothing. Activated when the mouse is pressed and released.
     * @param   e   a mouse event
     */
    public void mouseClicked(MouseEvent e){}

    /**
     * Does nothing. Activated when the mouse enters the component.
     * @param   e   a mouse event
     */
    public void mouseEntered(MouseEvent e){}

    /**
```

```
     * Does nothing. Activated when the mouse exits the component.
     * @param   e   a mouse event
     */
    public void mouseExited(MouseEvent e){}

    /**
     * Updates the mouse variables.
     * @param   e   a mouse event
     */
    public void mousePressed(MouseEvent e)
    {
        if(e.getButton() == e.BUTTON1)
            mousePressed1 = true;
        if(e.getButton() == e.BUTTON2)
            mousePressed2 = true;
        if(e.getButton() == e.BUTTON3)
            mousePressed3 = true;
    }


    /**
     * Updates the mouse variables.
     * @param   e   a mouse event
     */
    public void mouseReleased(MouseEvent e)
    {
        if(e.getButton() == e.BUTTON1)
            mousePressed1 = false;
        if(e.getButton() == e.BUTTON2)
            mousePressed2 = false;
        if(e.getButton() == e.BUTTON3)
            mousePressed3 = false;
    }


    /**
     * Updates the mouse variables.
     * @param   e   a mouse event
     */
    public void mouseDragged(MouseEvent e)
    {
        if(e.getButton() == e.BUTTON1)
            mousePressed1 = !mousePressed1;
        if(e.getButton() == e.BUTTON2)
            mousePressed2 = !mousePressed2;
        if(e.getButton() == e.BUTTON3)
            mousePressed3 = !mousePressed3;
        mouseX = e.getX();
        mouseY = e.getY();
    }


    /**
     * Updates the mouse variables.
     * @param   e   a mouse event
     */
    public void mouseMoved(MouseEvent e)
    {
        mousePressed1 = false;
        mousePressed2 = false;
        mousePressed3 = false;

        mouseX = e.getX();
        mouseY = e.getY();
    }
```

```
    /**
     * Updates the keyboard variables.
     * @param   e   a key event
     */
    public void keyPressed(KeyEvent e)
    {
        if(debug)
            System.out.println(e.getKeyText(e.getKeyCode()));
        keysPressed.add(e.getKeyText(e.getKeyCode()));
    }

    /**
     * Updates the keyboard variables.
     * @param   e   a key event
     */
    public void keyReleased(KeyEvent e)
    {
        for(int i = 0; i < keysPressed.size(); i++)
        {
            if(keysPressed.get(i).equals(e.getKeyText(e.getKeyCode())))
            {
                keysPressed.remove(i);
                i--;
            }
        }
    }

    /**
     * Updates the keyboard variables.
     * @param   e   a key event
     */
    public void keyTyped(KeyEvent e){}

    /**
     * Returns weather a mouse button is pressed.
     * @param   b   button number
     * @return  true if the button is pressed
     */
    public boolean isMousePressed(int b)
    {
        if(b == 1)
            return mousePressed1;
        else if(b == 2)
            return mousePressed2;
        else if(b == 3)
            return mousePressed3;

        return false;
    }

    /**
     * Returns weather any mouse button is pressed.
     * @return  true if the button is pressed
     */
    public boolean isMousePressed()
    {
        if(mousePressed1)
            return mousePressed1;
        else if(mousePressed2)
            return mousePressed2;
        else if(mousePressed3)
            return mousePressed3;

        return false;
    }

    /**
```

```
     * Returns weather a mouse button is pressed.
     * @param   k   the key pressed (Ex. "A", "B", "C"...")
     * @return  true if the key is pressed
     */
    public boolean isKeyPressed(String k)
    {
        for(int i = 0; i < keysPressed.size(); i++)
        {
            if(keysPressed.get(i).equalsIgnoreCase(k))
            {
                return true;
            }
        }
        return false;
    }

    //returns the number of keys pressed.
    public int getKeysPressed()
    {
        return keysPressed.size();
    }

    public void resetKeys()
    {
        keysPressed = new ArrayList();
    }
}
```