

HOW TO WRITE A *KeyListener* USING THE *KeyEvent* CLASS


A keyboard event is generated whenever the user presses a key on the keyboard. By using keyboard events, you can get the values of the keys that the user pressed to perform an action.

In Java, keyboard event objects belong to a class called **KeyEvent**. Any objects that need to listen for key events must implement the **KeyListener** interface. The object must be registered with a component by calling the component's **addKeyListener()** method. The registration is done with the following command:

```
component.addKeyListener(listener);
```

The **listener** is the object that listens for key events, and **component** is the object that will generate the key events (when it has the input focus).

The **KeyListener** interface defines the following methods to respond to events arising when a key on the keyboard is pressed or released:



METHOD	DESCRIPTION
keyTyped(KeyEvent e)	Called when a key on the keyboard is pressed and then released.
keyPressed(KeyEvent e)	Called when a key on the keyboard is pressed.
keyReleased(KeyEvent e)	Called when a key on the keyboard is released.

Every key event has an associated key character or key code that is returned by the **getKeyChar()** or **getKeyCode()** method in the **KeyEvent** class. The key codes are constants, some of which are outlined in the following table:

CONSTANT	DESCRIPTION
VK_HOME	The HOME key
VK_END	The END key
VK_PGUP	The PAGE UP key
VK_PGDN	The PAGE DOWN key
VK_UP	The UP arrow key
VK_DOWN	The DOWN arrow key
VK_LEFT	The LEFT arrow key
VK_RIGHT	The RIGHT arrow key
VK_ESCAPE	The ESCAPE key
VK_TAB	The TAB key
VK_CONTROL	The CTRL key

CONSTANT	DESCRIPTION
VK_SHIFT	The SHIFT key
VK_BACK_SPACE	The BACKSPACE key
VK_CAPS_LOCK	The CAPS LOCK key
VK_NUM_LOCK	The NUM LOCK key
VK_ENTER	The ENTER key
VK_UNDEFINED	The keyCode unknown
VK_F1 to VK_F12	The FUNCTION keys
VK_0 to VK_9	The number keys
VK_A to VK_Z	The letter keys

The following program demonstrates how the **KeyEvent** class works:

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

public class KeyEventDemo extends JPanel implements KeyListener {

    // Declare class variables
    private int x = 120, y = 100;
    private ImageIcon imgRobot;
    private final int WIDTH = 300, HEIGHT = 300;

    public static void main(String[] args) {

        new KeyEventDemo();
    }

    public KeyEventDemo() {

        // Initialize ImageIcon object
        imgRobot = new ImageIcon("iRobot.png");

        // Set the properties of the JPanel
        setLayout(null);
        addKeyListener(this);
        setFocusable(true);
    }
}
```

```

    // Declare and initialize JFrame
    JFrame frame = new JFrame();
    frame.setContentPane(this);
    frame.setTitle("KeyEvent Demo");
    frame.setSize(WIDTH, HEIGHT);
    frame.setLocationRelativeTo(null);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setVisible(true);
}

public void keyPressed(KeyEvent e) {

    // User clicks the left arrow key
    if (e.getKeyCode() == KeyEvent.VK_LEFT)
    {
        x -= 10;
    }
    // User clicks the right arrow key
    else if (e.getKeyCode() == KeyEvent.VK_RIGHT)
    {
        x += 10;
    }
    // User clicks the up arrow key
    else if (e.getKeyCode() == KeyEvent.VK_UP)
    {
        y -= 10;
    }
    // User clicks the down arrow key
    else if (e.getKeyCode() == KeyEvent.VK_DOWN)
    {
        y += 10;
    }
    repaint();
}

public void keyReleased(KeyEvent e) {}
public void keyTyped(KeyEvent e) {}

public void paintComponent(Graphics g) {

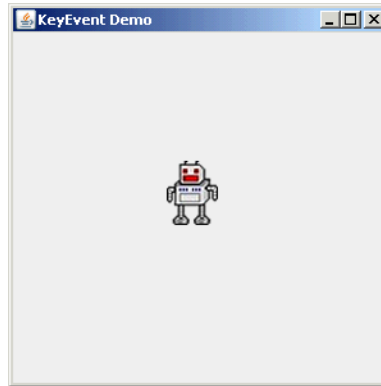
    // Clear the drawing canvas
    super.paintComponent(g);

    // Declare and initialize a Graphics2D object
    Graphics2D g2 = (Graphics2D) g;

    // Draw the image on the panel
    g2.drawImage(imgRobot.getImage(), x, y, this);
}
}

```

When you run the above program, you will see an image of a robot in the centre of the screen. When you click the up arrow key the robot will move up, when you click the down arrow key the robot will move down, when you click the left arrow key the robot will move left, and when you click the right arrow key the robot will move right.



Of course, the robot will go off the screen since we did not account for its position. Create four methods – **moveLeft()**, **moveRight()**, **moveUp()** and **moveDown()** that includes an if statement that ensures that the robot does not go off the screen.

*HINT: You will need to use the **getIconWidth()** and **getIconHeight()** methods in order to get the dimensions of the robot.*

EXTENDING THE *KeyAdapter* CLASS

You'll notice that even though I did not use the **keyReleased()** and **keyTyped()** methods, I still need to declare them in my program. To avoid having to do this extra step, you can create an inner class that extends the **KeyAdapter** class.

The first thing we would have to change from our original version is the class declaration. In the class declaration we would need to remove **implements KeyListener**.

```
public class KeyEventDemo extends JPanel implements KeyListener
```

The next step would be removing the reference to the JPanel when adding the key listener and instead have it point to a new instance of our inner class, which we'll call **RobotListener**:

```
addKeyListener(new RobotListener());
```

The final step requires us to declare our inner class and include the instructions for the **keyPressed()** method:

```
class RobotListener extends KeyAdapter {  
    public void keyPressed(KeyEvent e) {  
        ...  
    }  
}
```