Targeting a JTextArea

Having a System.out.println() message displayed when a button is clicked is a fairly simplistic demonstration of GUI components.
Lets modify the program in the previous less so that the results show up in a JLabel components instead.

We will instantiate an object of type JLabel in the class EventDemo2 and connect it to our listener class. In the MyButtonListener2 class we will use a setText() method to change the text within the JLabel.

Oops!!

cannot resolve symbol constructor MyButtonListener (javax.swing.JLabel)

cannot resolve symbol variable l1

EventDemoOops.java

```
1  import java.awt.*;
2  import java.awt.event.*;
3  import javax.swing.*;
4
5  public class EventDemoOops
6  {
7      public static void main(String args[])
8      {
9          JFrame jf = new JFrame("My First Event Application");
10         Container c = jf.getContentPane();
11
12         FlowLayout f = new FlowLayout();
13         c.setLayout(f);
14
15
16         JButton b1 = new JButton("Click");
17         c.add(b1);
18
19         JLabel l1 = new JLabel("I have not been clicked yet?");
20         c.add(l1);
21
22         MyButtonListener mbl = new MyButtonListener(l1);
23         b1.addActionListener(mbl);
24
25         jf.pack();
26         jf.setVisible(true);
27     }
28 }
29
30  class MyButtonListener implements ActionListener
31 {
32
33     public void actionPerformed(ActionEvent e)
34     {
35         l1.setText("I have been clicked!");
36
37     }
38 }
```

The problem centers around the JLabel object l1.
It has been instantiated in the class EventDemoOops.

When the user goes to make reference to the object in the Listener class the compiler complains because the object is not in focus in this class.

Some how we must get a copy of the JLabel object reference from the first class to the event listener class.

How??

Passing a Reference as a Parameter

We have had lots of experience passing primitives and Strings as arguments in a constructor call. Now we will pass an object reference as an argument through a constructors parameter.

Consider the following statements;

If we can do this…..

```
int x = 5;
Cars c = new Cars(x);
```

…then why not this?

```
JLabel  l1 = new Jlabel("I have not been clicked yet")
MyButtonListener mbl = new MyButtonListener(l1);
```

Provide we modify the action event class so that it includes a constructor which takes in a single JLabel object reference, we will be able pass the reference l1 the same way we would any other argument in a constructor call.

Passing l1 as a Constructor Argument

Re-written class to include constructor which takes in a single JLabel object reference.

Private JLabel reference

l1 gets passes as an argument from the constructor call.

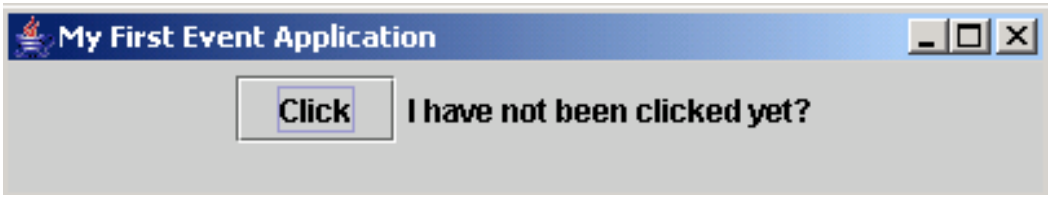
value of parameter is assigned to private reference.

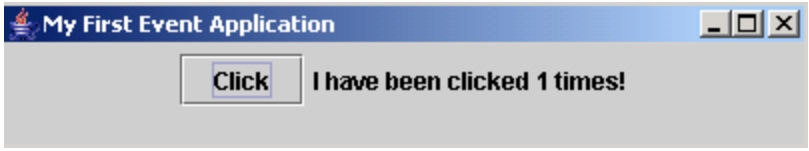
EventDemo2.java

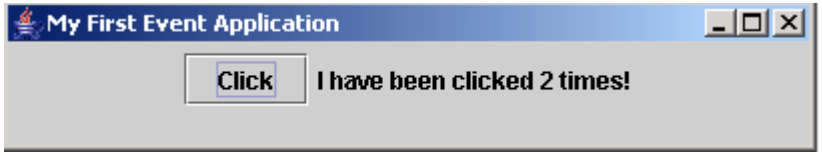
```
29
30  class MyButtonListener implements ActionListener
31  {
32      private JLabel label;
33      int num = 1;
34      public MyButtonListener(JLabel l1)
35      {
36          label = l1;
37      }
38      public void actionPerformed(ActionEvent e)
39      {
40          label.setText("I have been clicked " + num + " times!");
41          num++;
42      }
43  }
```

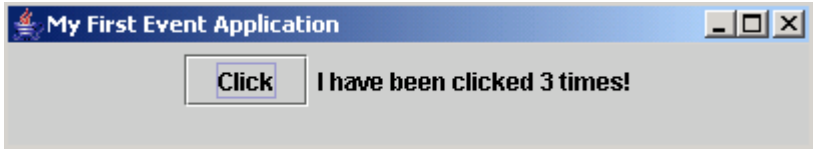
label is a local reference with the same value as the parameter. It can now be used with the class and more specifically within the actionPerformed() method.

The Results


My First Event Application
Click
I have not been clicked yet?


My First Event Application
Click
I have been clicked 1 times!


My First Event Application
Click
I have been clicked 2 times!

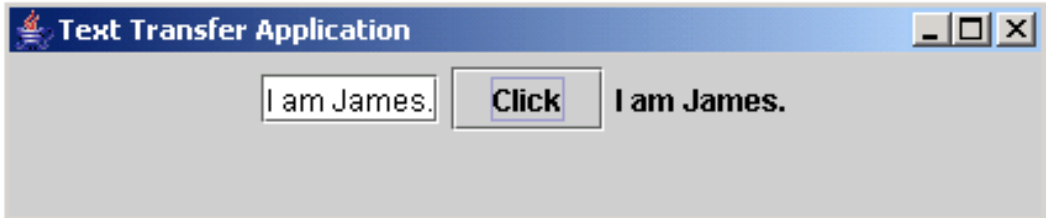

My First Event Application
Click
I have been clicked 3 times!

Your Turn

Write a class that allows the user to type a message in a JTextArea box, click a button and have the text show up in a JLabel box.

Hints:
You will need to pass references for a JTextArea component as well as a JLabel

Use the method getText() to retrieve text from the first component and setText() to display the text in the second component.



The Solution

```
JTextField t1 = new JTextField("Write something here");
c.add(t1);

JButton b1 = new JButton("Click");
c.add(b1);

JLabel l1 = new JLabel("I have not been clicked yet?");
c.add(l1);
```

Private component references

```
MyButtonListener mbl = new MyButtonListener(t1,l1);
b1.addActionListener(mbl);
```

Instantiation and call to constructor passing two component references

EventTextTransfer.java *

```
33
34  class MyButtonListener implements ActionListener
35  {
36      private JLabel label;
37      private JTextField field;;
38      private String s1;
39      public MyButtonListener(JTextField t1, JLabel l1)
40      {
41          label = l1;
42          field = t1;
43      }
44      public void actionPerformed(ActionEvent e)
45      {
46          s1 = field.getText();
47          label.setText(s1);
48
49      }
50  }
```

Constructor with JTextField and JLabel parameters

Assign parameter values to private reference values.

Get and set text using local private object references.

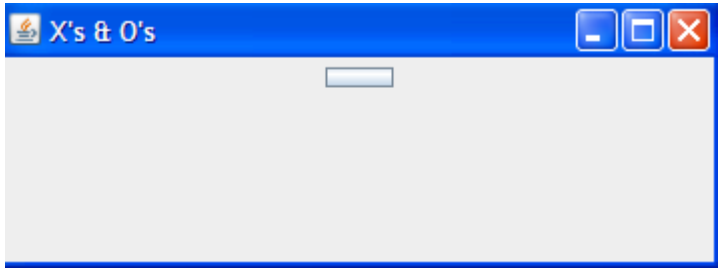
X's and O's

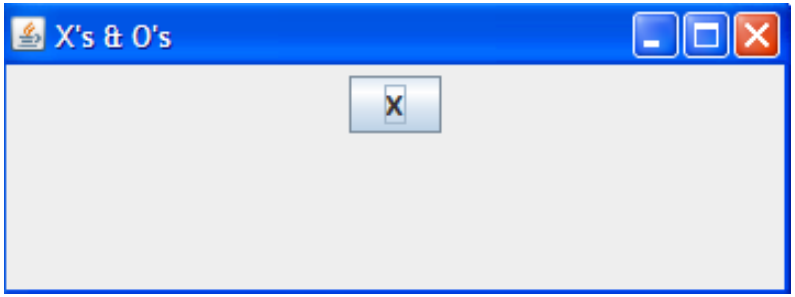
```
import java.awt.event.*;
import java.awt.*;
import javax.swing.*;

public class XOGame
{
    public static void main(String args[])
    {
        JFrame jf = new JFrame("X's & O's");
        jf.setSize(400,400);

        Container c = jf.getContentPane();

        FlowLayout fl = new FlowLayout();
        c.setLayout(fl);

        JButton b1 = new JButton();
        b1.setSize(60,60);
        c.add(b1);

        MyButtonListener mbl = new MyButtonListener(b1);
        b1.addActionListener(mbl);
        jf.setVisible(true);
    }
}
```

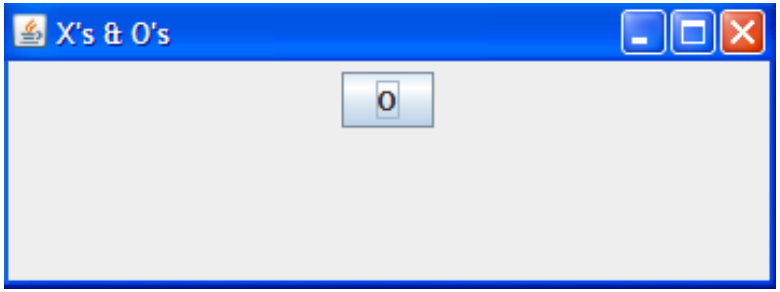
The Listener

```
class MyButtonListener implements ActionListener
{
    JButton b = new JButton();
    int turn = 0;

    public MyButtonListener(JButton b1)
    {
        b = b1;
    }
    public void actionPerformed(ActionEvent e)
    {
        if(turn%2 == 0)
        {
            b.setText("X");
        }
        else
        {
            b.setText("O");
        }
        turn++;

    }
}
```

Running the Program







Finish The Game