Chapter 4: Scrolling

Scrolling

Once you get that last program working, see what happens when you click a button more than 10 times. It looks like the program quits working! It doesn't—it keeps adding text to the text area, but there's no more room to display it, so it doesn't show up. We're going to look at how to fix that by adding a *scroll bar* to our window so you can scroll up and down to see all the text.

We only need to use one more class in our program to make it scroll. The JScrollPane object provides scroll bars. It will also manage the mouse clicks in the scroll bar and move the text display accordingly. Let's see how to make it work.

The first thing we need to do is add one more declaration to our list of window components. We can declare a JScrollPane object like this:

```
private JScrollPane scrollArea;
```

To make our text scroll, we'll need to do three things. First, instead of adding the textArea to the window pane, we'll pass it to the scroll pane when we call its constructor. That will tell the scroll pane what component will be scrolled. So let's replace the line that added textArea to contentPane with this line:

```
scrollArea = new JScrollPane(textArea);
```

Second, we need to configure the scrolling area by telling it how we want the scroll bars to appear. There are two scroll bars, a vertical one on the right side that controls up and down motion, and a horizontal one across the bottom that controls left and right scrolling.

We can choose to make each of the bars appear all the time, none of the time (if we know we won't need one of them), or only when they're needed. Since we won't do any right or left scrolling, we'll set the horizontal scroll bar to be invisible. And we'll make the vertical bar invisible until we need it.

To configure the bars, we use two methods from the scroll pane class, *setVerticalScrollBarPolicy()* and *setHorizontalScrollBarPolicy()*. The calls look like this:

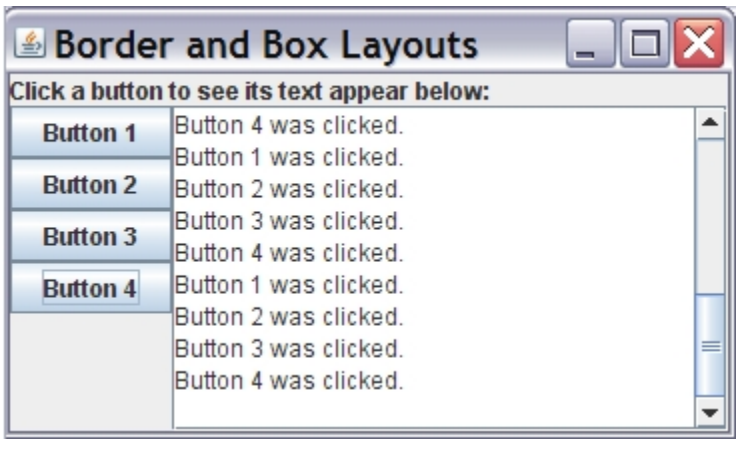
```
scrollArea.setVerticalScrollBarPolicy(ScrollPaneConstants.VERTICAL_SCROLLBAR_AS_NEEDED);

scrollArea.setHorizontalScrollBarPolicy(ScrollPaneConstants.HORIZONTAL_SCROLLBAR_NEVER);
```

The constants that control the scroll bar policies can end with *ALWAYS*, *AS_NEEDED*, or *NEVER*, depending on how we want the scroll bars to behave.

Okay, that's two out of three. The last thing we need to do is add the scroll pane to the window instead of adding the text area. We already know how to use contentPane's add() method to do that:

```
contentPane.add(scrollArea, BorderLayout.CENTER);
```

That's it! We're done! If you run the program now, and click enough buttons, your window should look something like this:



Scrolling text area

Your scroll bar should work exactly like any other, moving the text up and down so you can look at all of it. If you can't get it to work, let me know, or take a look at my finished program here.

Solution: Try not to peek!