JAVA NOTES

GRAPHICAL USER INTERFACES

Terry Marris 24 June 2001

5 TEXT AREAS

5.1 LEARNING OUTCOMES

By the end of this lesson the student should be able to

- understand how to get multi-line input from the user
- use text area for input and output

5.2 INTRODUCTION

In the last chapter we saw how to obtain input from the user a line at a time.

Sometimes we need to collect user input that is more than one line long. To do this we use the *JTextArea* component.

With a text area, a user can enter any number of lines by using the enter key to separate them. Each line is terminated with '\n'.

5.3 CAR SERVICE GUI

Figure 5.1 shows what we are aiming for.

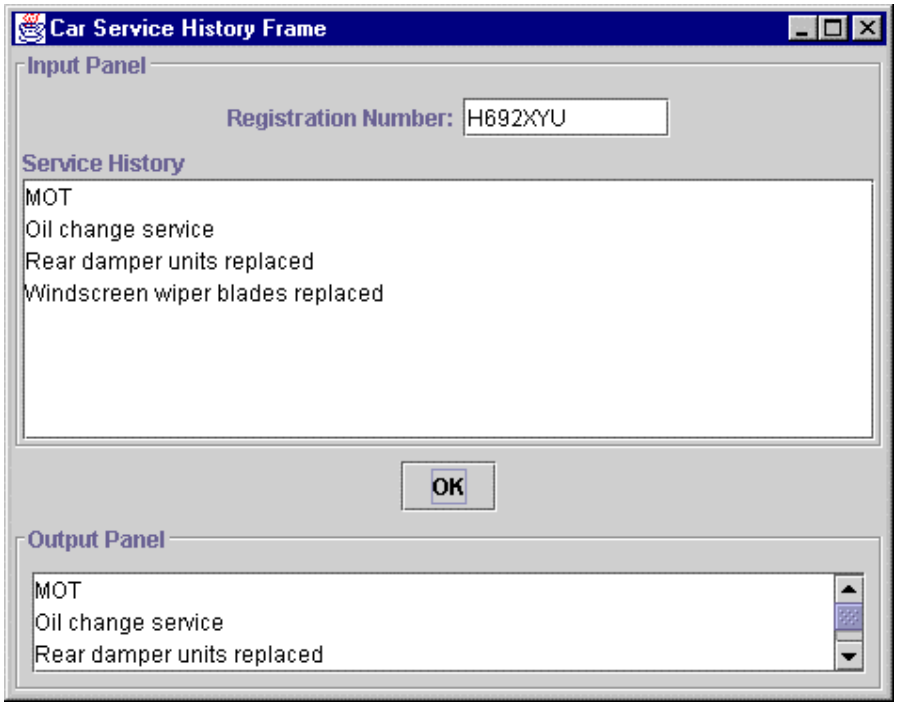

Figure 5.1 *Text Area Input Output*

The user enters a car's registration number and makes some entries in the Service History area of the input panel.

When the ok button is clicked, the input is echoed to the text area of the output panel. But because there are more lines of text than can fit in the allocated space, a scroll bar is generated. A scroll bar allows the user to scroll through text in a text area.

5.4 JTEXT AREAS

We specify the number of rows and columns in the *JTextArea* constructor.

```
serviceHistoryTextArea = new JTextArea(8, 40);
```

This creates a text area with eight lines of forty columns each.

We place the text area in a scroll pane. A scroll pane handles scrolling and scroll bars.

```
JScrollPane scrollPane = new
            JScrollPane(serviceHistoryTextArea);
```

Then we add the scroll pane to a container e.g. a panel.

```
add(scrollPane, "South");
```

If a line of text is too long to fit into its allocated width, we can specify line breaks to occur at word boundaries.

```
outputTextArea.setLineWrap(true);
outputTextArea.setWrapStyleWord(true);
```

This does not insert hard returns or newline characters '\n'; it just provides a view of the text.

5.5 CAR SERVICE HISTORY FRAME

The code that produced Figure 4.1 is shown below.

```
/* CarServiceHistoryFrame.java
   Terry Marris  24 June 2001
*/

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


class RegistrationNumberPanel extends JPanel {
  private JTextField regNumField;

  public RegistrationNumberPanel()
  {
    JLabel regNumLabel = new JLabel("Registration Number");
    add(regNumLabel);

    regNumField = new JTextField(10);
    add(regNumField);
  }
}
```

```
class ServiceHistoryPanel extends JPanel {
  private JTextArea serviceHistoryTextArea;

  public ServiceHistoryPanel()
  {
    setLayout(new BorderLayout());

    JLabel serviceHistoryLabel = new
                          JLabel("Service History");
    add(serviceHistoryLabel, "North");

    serviceHistoryTextArea = new JTextArea(8, 40);
    JScrollPane scrollPane = new
                JScrollPane(serviceHistoryTextArea);
    add(scrollPane, "South");
  }


  public String serviceHistory()
  {
    return serviceHistoryTextArea.getText();
  }
}
```

```
class InputPanel extends JPanel {
  private RegistrationNumberPanel regNumPanel;
  private ServiceHistoryPanel serviceHistoryPanel;

  public InputPanel()
  {
    Border etched = BorderFactory.createEtchedBorder();
    Border titled = BorderFactory.createTitledBorder(
                                  etched, "Input Panel");
    setBorder(titled);

    setLayout(new BorderLayout());

    regNumPanel = new RegistrationNumberPanel();
    add(regNumPanel, "North");

    serviceHistoryPanel = new ServiceHistoryPanel();
    add(serviceHistoryPanel, "South");
  }


  public String serviceHistory()
  {
    return serviceHistoryPanel.serviceHistory();
  }
}
```

```
class OutputPanel extends JPanel {
  private JTextArea outputTextArea;

  public OutputPanel()
  {
    Border etched = BorderFactory.createEtchedBorder();
    Border titled = BorderFactory.createTitledBorder(
                                  etched, "Output Panel");
    setBorder(titled);

    outputTextArea = new JTextArea(3, 40);
    outputTextArea.setEditable(false);
    outputTextArea.setBackground(Color.white);
    outputTextArea.setLineWrap(true);
    outputTextArea.setWrapStyleWord(true);
    JScrollPane scrollPane = new JScrollPane(outputTextArea);
    add(scrollPane);
  }


  public void setOutputTextArea(String text)
  {
    outputTextArea.setText(text);
  }
}
```

```
class ButtonPanel extends JPanel {
  private JButton okButton;

  public ButtonPanel()
  {
    okButton = new JButton("OK");
    add(okButton);
  }


  public JButton getOkButton()
  {
    return okButton;
  }
}
```

```
class CarServiceHistoryPanel extends JPanel
                             implements ActionListener {
  private InputPanel inputPanel;
  private OutputPanel  outputPanel;
  private ButtonPanel buttonPanel;

  public CarServiceHistoryPanel()
  {
    setLayout(new BorderLayout());

    inputPanel = new InputPanel();
    add(inputPanel, "North");

    buttonPanel = new ButtonPanel();
    add(buttonPanel, "Center");
    buttonPanel.getOkButton().addActionListener(this);

    outputPanel = new OutputPanel();
    add(outputPanel, "South");
  }


  public void actionPerformed(ActionEvent e)
  {
    Object source = e.getSource();
    if (source == buttonPanel.getOkButton()) {
      String text = inputPanel.serviceHistory();
      outputPanel.setOutputTextArea(text);
    }
  }
}
```

```
public class CarServiceHistoryFrame extends JFrame {

  public CarServiceHistoryFrame()
  {
    setTitle("Car Service History Frame");

    Toolkit tk = Toolkit.getDefaultToolkit();
    Dimension screen = tk.getScreenSize();
    setLocation(0, 0);

    addWindowListener(new WindowAdapter()
      {
        public void windowClosing(WindowEvent e)
        {
          System.exit(0);
        }
      });


    Container contentPane = getContentPane();
    contentPane.add(new CarServiceHistoryPanel());
    pack();
  }


  public static void main(String[] s)
  {
    JFrame aFrame = new CarServiceHistoryFrame();
    aFrame.show();
  }
}
```

5.6 DOCUMENTATION

JTextArea has methods that are the same as *JTextField* because they both have the same inheritance hierarchy.

javax.swing.JTextArea

Constructors	
JTextArea(int rows, int columns)	initialises a new empty text area with the given number of rows and columns.
JTextArea(String text, int rows, int columns)	initialises a new text area with the given text, rows and columns.

Methods		
void	append(String str)	adds the given string to the end of the document.
int	getLineCount()	returns the number of lines contained in this area.
void	insert(String str, int position)	inserts the given string at the given position in this text area.
void	replaceRange(String str, int start, int end)	replaces the text from the start to the end position with the given text.
void	setLineWrap(boolean on)	sets the line-wrapping policy for this area. If true the lines will be wrapped if they are too long to fit within the allocated width.
void	setWrapStyleWord(boolean on)	sets the style of wrapping used if this text area is wrapping lines. If true the lines will be wrapped at word boundaries.

5.7 FURTHER READING

HORSTMANN & CORNELL *Core Java 2 Volume 1* pp 400
www.java.sun.com/docs/books/tutorial

In the next chapter we look at radio buttons.

5.8 REVIEW

5.9 EXERCISES

1 Draw a class diagram that reflects the structure of program *CarServiceHistoryFrame.java* shown in §5.5 above.

2 Explain each line of the OutputPanel class.

3 Explain each line of the CarServiceHistoryPanel class. By diagrams, or otherwise, explain the sequence of messages, how the CarServiceHistoryPanel object obtains a copy of the service history text.