Labs

Lab 4.1: Web Visit Count

What is the purpose?

In this lab, you will study the example “Applet Clients” on pages 988-989 of the textbook. The code in the example creates an applet indicating the number of visits made to a Web page. The count is stored in a file on the server side. Each time the page is visited or reloaded, the applet sends a request to the server, which increases the count and sends it to the applet. The count is stored using a random access file. When the applet is loaded, the server reads the count from the file, increases the count, and saves the count back to the file.

Rewrite the program to improve its performance. The program must read the count from the file when the server starts and save the count to the file when the server is stopped by using the Stop button. Use a variable to store the count when the server is active. Reuse the client program “CountServer.java” in Listing 30.5. Rewrite the server as a GUI application that uses a Stop button to exit the server.

Here is a sample of the output:

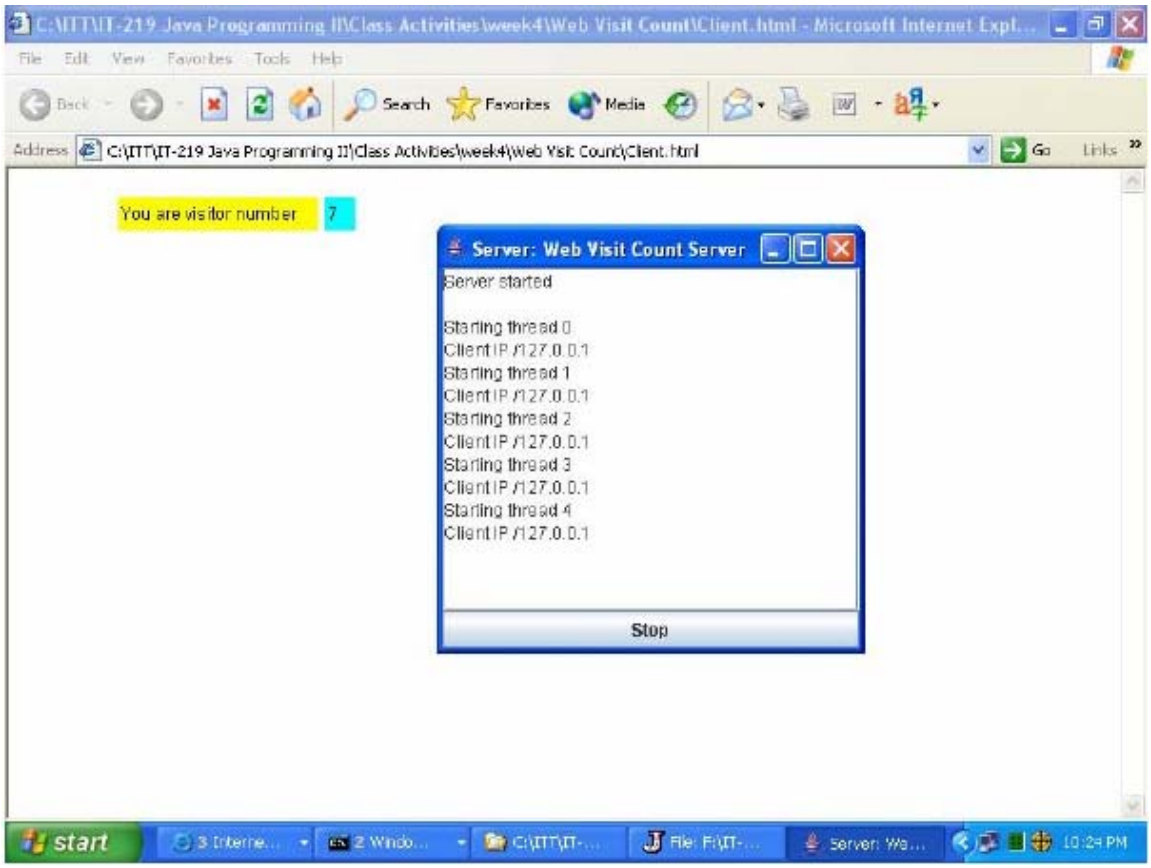

Figure 4-1-1

What are the steps?

- **Task 1**

Procedure:

1. Create a java file named Server.java that extends the JFrame class and implements the ActionListener class. Reference the CountServer.java on pages 987-988.
2. Establish a socket connection with the server using the following code:

```
Socket connectToServer = new Socket("localhost", 8000);
```

Figure 4-1-2

3. Create a java file named Server.java that extends the JFrame class and implements the ActionListener class.
4. Create a scroll pane to contain the text area.
5. Create a container to add a panel and a scroll panel to the frame.
6. Create a JButton for "Stop" and register it to ActionListener.
7. Create a stream socket to get the InetAddress from the Client class using the following code segments:

```
serverSocket = new ServerSocket(8000);
Socket connectToClient = serverSocket.accept();
connectToClient.getInetAddress();
WebVisitHandler thread = new WebVisitHandler(connectToClient);
thread.start();
```

Figure 4-1-3

8. Create an HTML file named client.html to embed the Client class:

```
<APPLET code = "Client.class" width = 300 height = 200>
<P>
You must have a Java-enabled browser to view the applet
</APPLET>
```

Figure 4-1-4

9. Open the client.html file in a Web browser; then run the Server class to see all the captured IP addresses with threads.
10. Submit your Java code and sample output to your instructor.

Did it work?

Were you able to:

- Create an applet that can send a request to the server and record the visit on a file?
- Create a Server class that can read the number of client visits that were captured in a file?
- Add a Stop button on the server GUI that can exit the server?

Lab 4.2: Web Browser

What is the purpose?

In this lab, you will study Listing 30.11 on pages 998-999 of the textbook. Then modify WebBrowser.java so that it will:

- Accept an HTML file from a localhost. Assume that a local HTML file name does not begin with either http:// or www.
- Accept a remote HTML file. A remote HTML file name begins with either http:// or www.

Here is a sample of the output:

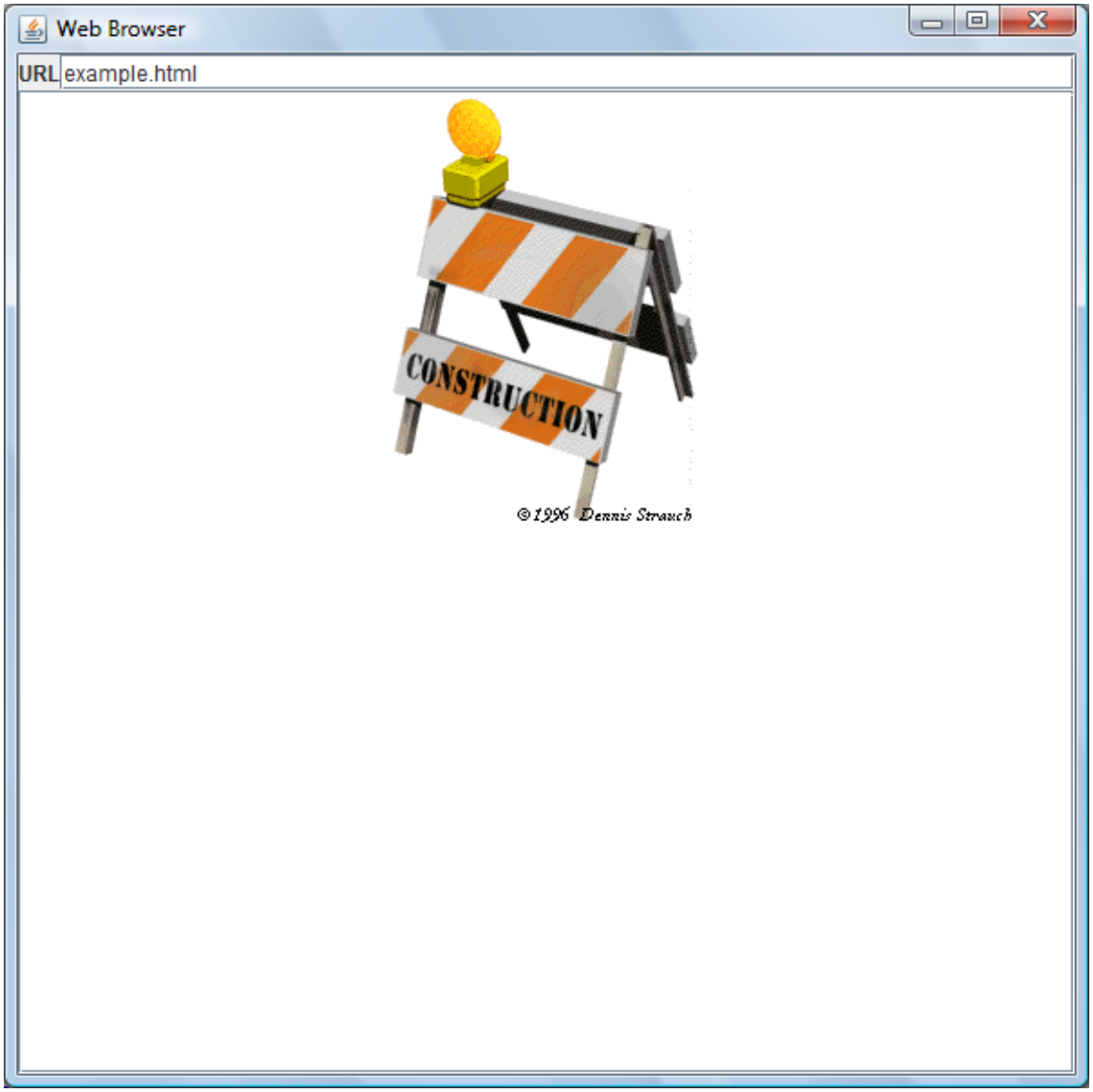

Figure 4-2-1: Sample Output 1

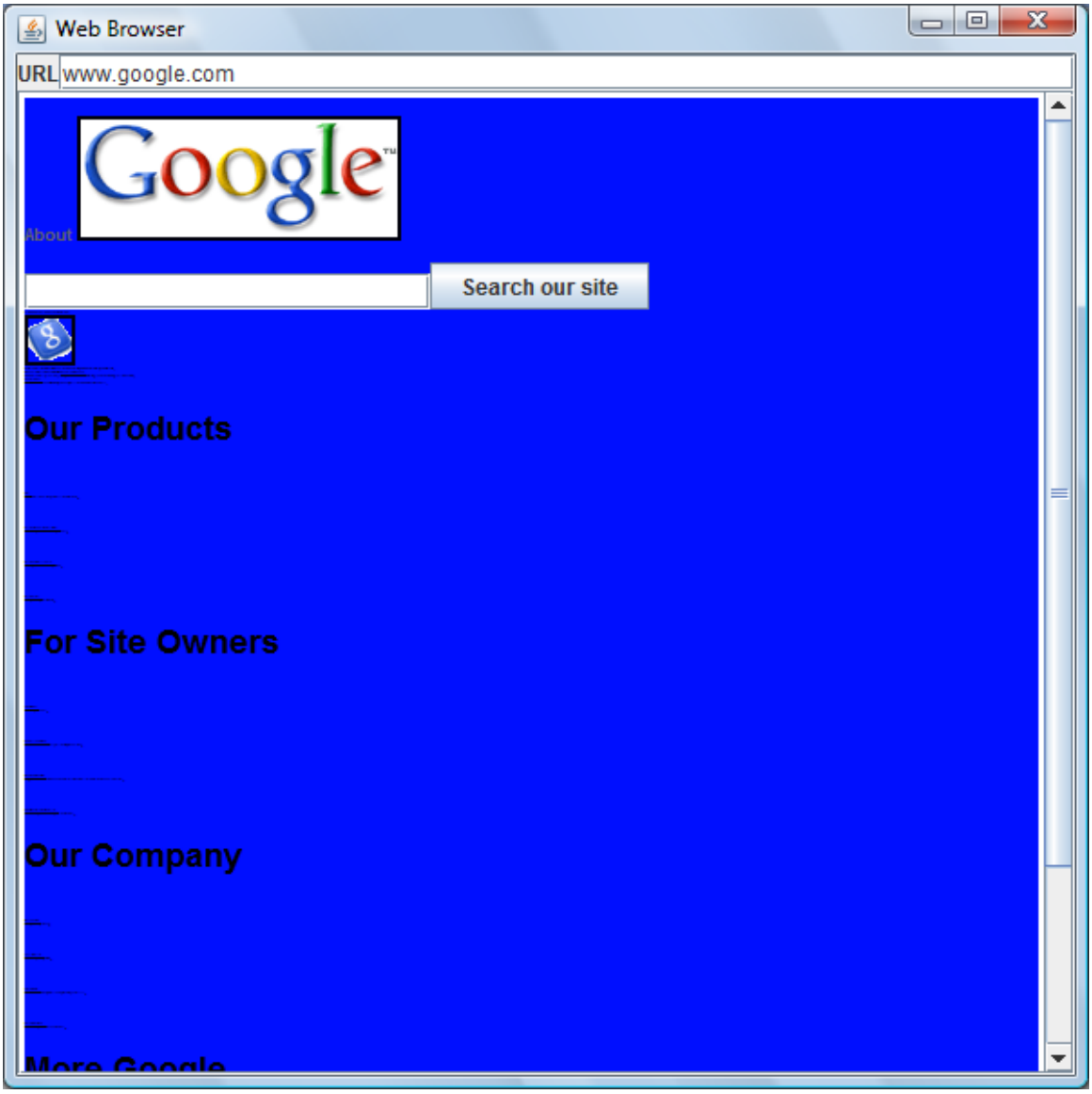

Figure 4-2-2: Sample Output 2

What are the steps?

- **Task 1**

  Procedure:

  1. Create a java file named Browser.java to display the HTML file in JEditorPane.
  2. Extend the class with JApplet and implement with ActionListener, HyperlinkListener.
  3. Add a JEditor pane to view HTML files.
  4. Initialize the applet by creating a panel to hold the label and the text field.

5. Create a scroll pane to hold the JEditorPane.
6. Place the panel and the scroll pane in the applet.
7. Set JEditor to be noneditable.
8. Register with addHyperlinkListener and addActionListener.
9. Submit your Java code and sample output to your instructor.

Did it work?

Were you able to:

- Create a simple web browser by using the JEditorPane?
- Accept a local HTML file?
- Accept a remote HTML file?