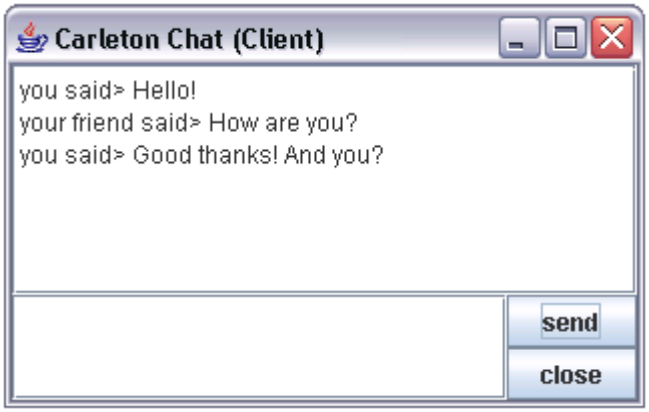
Consider a chat application using the GUI shown here. Part of the code is given below. The chat application has four GUI components:

- A JTextPane, used to display the conversation.
- A JTextField, used to type the text to send.
- Two JButtons, one to send the text typed in the JTextField, the other one is to close the connexion.



In addition to the four GUI components, a chat application has four other instance variables:

- A Socket, to maintain the connexion.
- An BufferedReader, to receive messages.
- An BufferedWriter, to send messages.
- A boolean indicating whether the connexion thread is still active.

The chat application uses the TCP port 1000. Your tasks are the following:

- Program a makeGUI() method which layouts the components as shown above.
- Complete the code for the server and the client side. You have to
  - establish the connexion;
  - initialize the BufferedReader/Writer with the Input/OutputStream of the connexion;
  - start a Thread which Runnable object will be the chat application.
- Complete the appendText() method. That method appends the String given as argument to the JTextPane which holds the conversation.
- Complete the actionPerformed() method.
  - If the event source is the send button, then:
    - send the text which is in the JTextPane to your pal using the BufferedWriter;
    - append that text to the conversation using the prompt “you said”;
    - clear the content of the JTextPane.
  - If the event source is the close button, then:
    - set the **running** variable to **false**;
    - close the connexion.
- Complete the run() method.
  - That method first initializes the **running** variable to **true**.
  - Then, as long as the running variable is true it
    - reads a line from the BufferedReader;
    - if that line is null, then the running variable is set to false; this will happen if the connexion is closed by the remote host;
    - otherwise, the text that is read is appended to the conversation with the prompt “your friend said”.
  - When the **running** variable is **false**, then the connexion is closed.
  - Do not forget that the readLine() method on BufferedReader may throw an IOException if the connexion has been closed. This will happen if the connexion is closed locally.

```
public class ChatApplication extends JFrame implements Runnable, ActionListener{

	public static final int PORT = 1000;

	private JTextPane conversation;
	private JTextField textToSend;
	private JButton send;
	private JButton close;

	private Socket socket;

	private BufferedReader input;
	private BufferedWriter output;

	private boolean running;

	public ChatApplication() throws IOException{
		super("Carleton Chat (Server)");
		makeGUI();
		/* add your code here */

	}

	public ChatApplication(String ip) throws IOException{
		super("Carleton Chat (Client)");
		makeGUI();
		/* add your code here */

	}
	private void appendText(String text){
		/* add your code here */

	}

	public void actionPerformed(ActionEvent e){
		/* add your code here */


	}

	public void run(){
		/* add your code here */


	}
}
```