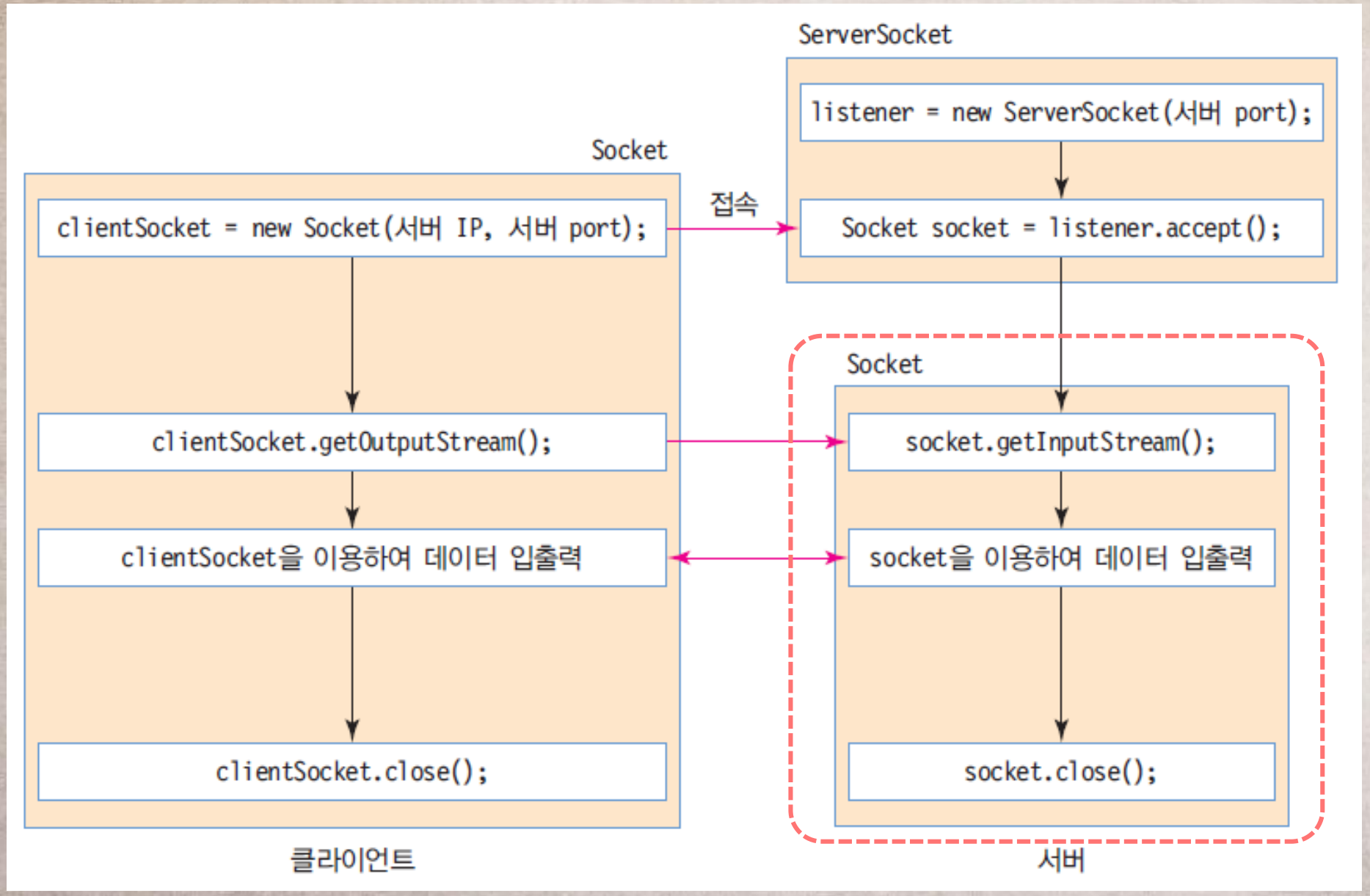
1:N 채팅

comsi.java@gmail.com
박 경 태

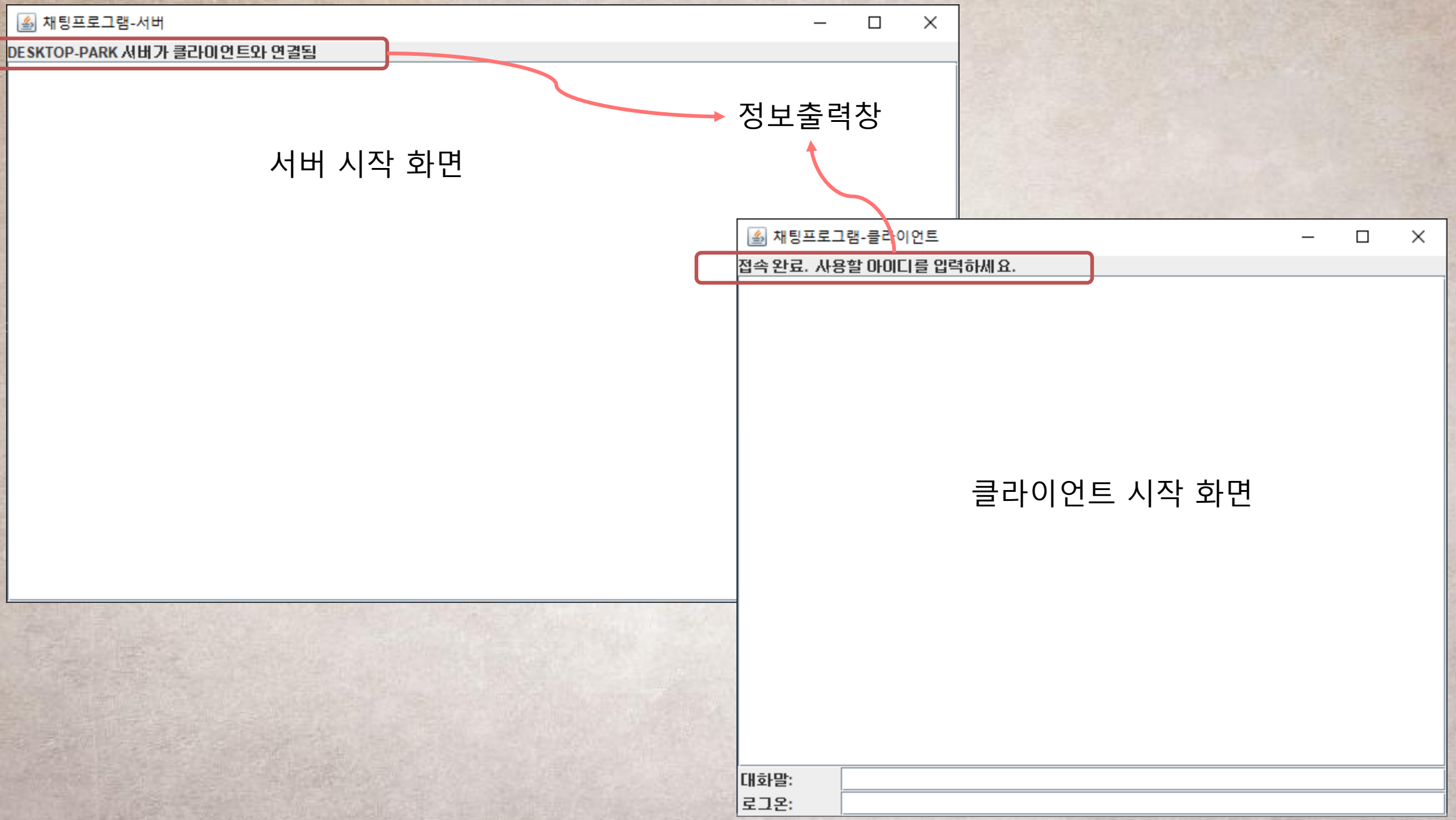
소켓을 이용한 서버 클라이언트 통신 프로그램의 구조

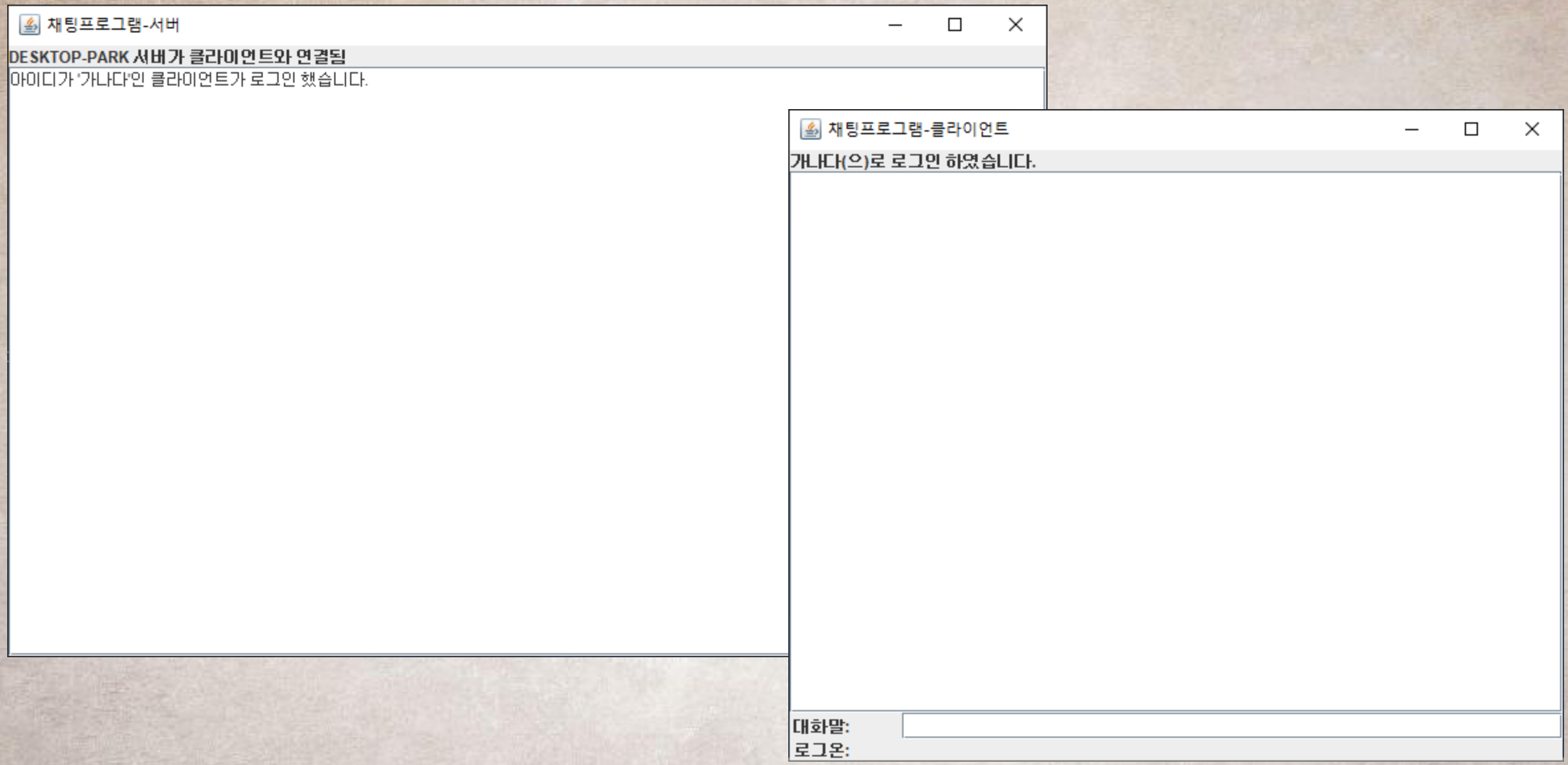


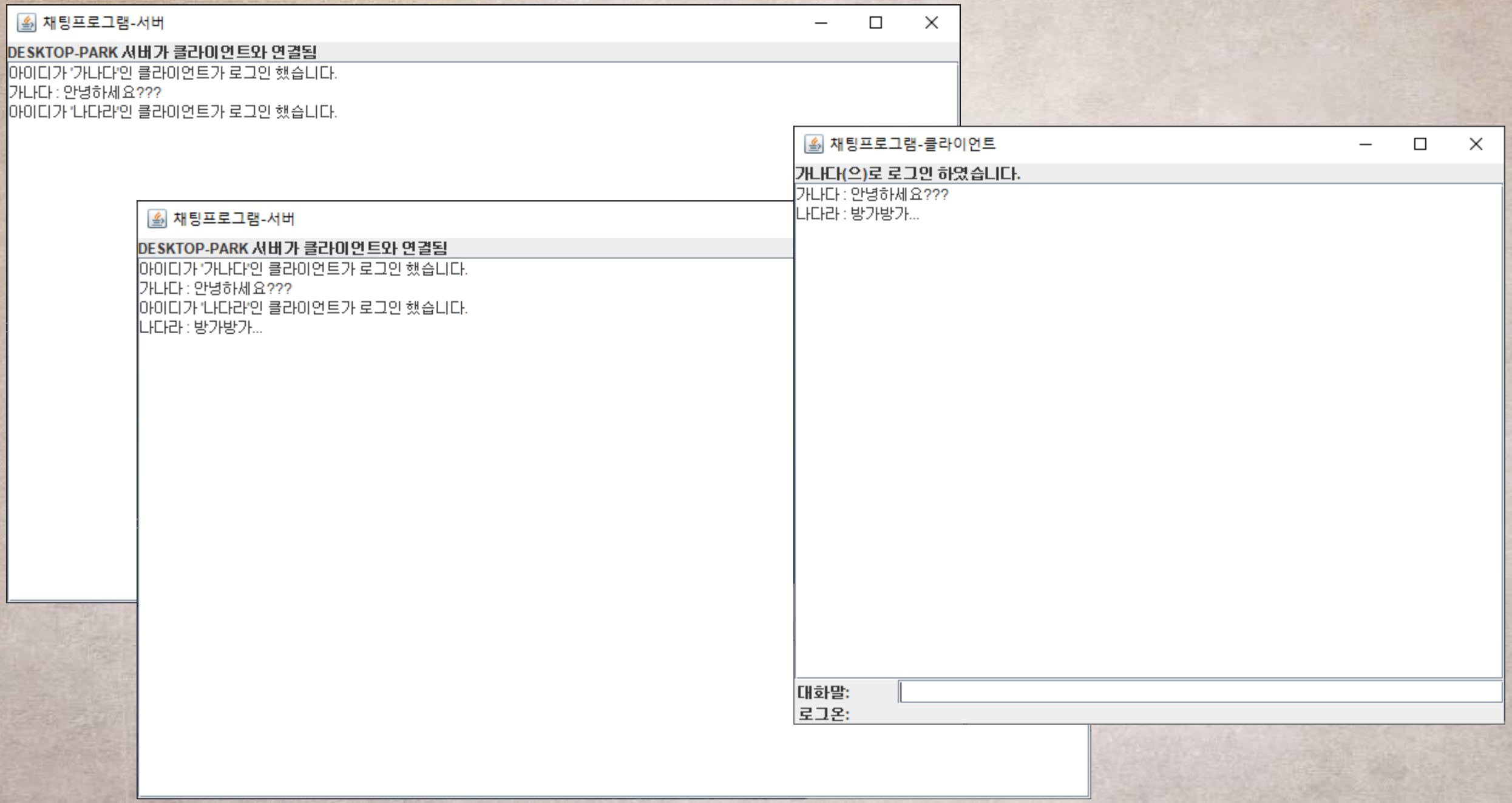
ServerThread(
- 서버클래스
- 소켓
- 대화표시창
- 라벨(로그표시))

서버와 클라이언트의 메시지 전송

- 사용자가 입력한 대화말 전송
  - A의 대화말이 다른 클라이언트에 전송될 때 전송자의 ID구분자가 필요
  - 로그인 과정 필요하고 대화말과 클라언트의 ID도 같이 전송
- 메시지 전송 형식
  - 메시지 코드 | 아이디 or 메시지 코드 | 아이디 | 대화말
  - 메시지 코드: 1001 – 로그인, 1021 – 대화말
- 클라이언트는 반드시 로그인하고 대화말을 전송해야 함


채팅프로그램-서버
DESKTOP-PARK 서버가 클라이언트와 연결됨
정보출력창
서버 시작 화면
채팅프로그램-클라이언트
접속 완료. 사용할 아이디를 입력하세요.
클라이언트 시작 화면
대화말:
로그온:


채팅프로그램-서버
DESKTOP-PARK 서버가 클라이언트와 연결됨
아이디가 '가나다'인 클라이언트가 로그인 했습니다.
채팅프로그램-클라이언트
가나다(으)로 로그인 하였습니다.
대화말:
로그온:


채팅프로그램-서버
DESKTOP-PARK 서버가 클라이언트와 연결됨
아이디가 '가나다'인 클라이언트가 로그인 했습니다.
가나다 : 안녕하세요???
아이디가 '나다라'인 클라이언트가 로그인 했습니다.
채팅프로그램-서버
DESKTOP-PARK 서버가 클라이언트와 연결됨
아이디가 '가나다'인 클라이언트가 로그인 했습니다.
가나다 : 안녕하세요???
아이디가 '나다라'인 클라이언트가 로그인 했습니다.
나다라 : 방가방가...
채팅프로그램-클라이언트
가나다(으)로 로그인 하였습니다.
가나다 : 안녕하세요???
나다라 : 방가방가...
대화말:
로그온:

ChatMessageS.java

OneToOneS.java | OneToOneC.java | ChatMessageS.java | ChatMessageC.java

```
1  import java.awt.BorderLayout;
2  import java.io.BufferedReader;
3  import java.io.BufferedWriter;
4  import java.io.IOException;
5  import java.io.InputStreamReader;
6  import java.io.OutputStreamWriter;
7  import java.net.ServerSocket;
8  import java.net.Socket;
9  import java.util.ArrayList;
10 import java.util.List;
11 import java.util.StringTokenizer;
12
13 import javax.swing.JFrame;
14 import javax.swing.JLabel;
15 import javax.swing.JScrollPane;
16 import javax.swing.JTextArea;
17 import javax.swing.JTextField;
18
19 public class ChatMessageS extends JFrame {
20
21     private static final long serialVersionUID = 1L;
22
23     JTextArea display;
24     JLabel info;
25     List<ServerThread> listThread;
26
```

```
27      public ChatMessageS() {
28          // creating the frame
29          super("채팅프로그램-서버");
30          setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
31          setSize(800, 500);
32          // info
33          info = new JLabel();
34          // TextArea
35          display = new JTextArea();
36          display.setEditable(false);
37          JScrollPane scrollPane = new JScrollPane(display);
38
39          getContentPane().add(BorderLayout.NORTH, info);
40          getContentPane().add(BorderLayout.CENTER, scrollPane);
41          setVisible(true);
42      }
43
44
45      public static void main(String[] args) {
46          // TODO Auto-generated method stub
47          ChatMessageS s = new ChatMessageS();
48          s.runServer();
49      }
50
```

```
51      public void runServer() {
52          ServerSocket server;
53          Socket sock;
54          ServerThread sThread;
55          try {
56              listThread = new ArrayList<ServerThread>();
57              server = new ServerSocket(5000, 100);
58              try {
59                  while(true) {
60                      sock = server.accept();
61                      sThread = new ServerThread(this, sock, display, info);
62                      sThread.start();
63                      info.setText(sock.getInetAddress().getHostName() + " 서버가 클라이언트와 연결됨");
64                  }
65
66              }catch(IOException ioe) {
67                  server.close();
68                  ioe.printStackTrace();
69              }
70          }catch(IOException e) {
71
72          }
73      }
74
75  }
76
```

```
77
78  class ServerThread extends Thread{
79      Socket sock;
80      BufferedWriter output;
81      BufferedReader input;
82      JTextArea display;
83      JLabel info;
84      JTextField text;
85      String clientdata;
86      String serverdata="";
87      ChatMessageS cs;
88
89      private static final String SEPERATOR = "¦";
90      private static final int REQ_LOGIN = 1001;
91      private static final int REQ_SENDWORDS = 1021;
92
93
94      public ServerThread(ChatMessageS c, Socket s, JTextArea ta, JLabel l) {
95          this.cs = c;
96          this.sock = s;
97          this.display = ta;
98          this.info = l;
99
100         try {
101             input = new BufferedReader(new InputStreamReader(sock.getInputStream()));
102             output = new BufferedWriter(new OutputStreamWriter(sock.getOutputStream()));
103         }catch(IOException ioe) {
104             ioe.printStackTrace();
105         }
106     }
107
```

```java
    @Override
    public void run() {
        cs.listThread.add(this);

        try {
            while((clientdata = input.readLine()) != null ) {
                StringTokenizer st = new StringTokenizer(clientdata, SEPERATOR);
                int command = Integer.parseInt(st.nextToken());
                int cnt = cs.listThread.size();
                switch(command) {
                    case REQ_LOGIN : { // "1001¦아이디" 인 경우
                        String ID = st.nextToken();
                        display.append("아이디가 '"+ ID +"'인 클라이언트가 로그인 했습니다.\r\n");
                        display.setCaretPosition(display.getDocument().getLength());
                    }
                    break;
                    case REQ_SENDWORDS :{
                        String ID = st.nextToken();
                        String message = st.nextToken();
                        display.append(ID + " : " + message + "\r\n");
                        display.setCaretPosition(display.getDocument().getLength());

                        for(int i=0 ; i<cnt ; i++) {
                            ServerThread sThread = (ServerThread) cs.listThread.get(i);
                            sThread.output.write(ID + " : " + message + "\r\n");
                            sThread.output.flush();
                        }
                    }
                    break;
                } // switch
            }
        }catch(IOException ioe) {
            ioe.printStackTrace();
        }

```

```
143             cs.listThread.remove(this);
144             try {
145                 sock.close();
146             }catch(IOException e) {
147                 e.printStackTrace();
148             }
149         } // run
150 
151 }
```

ChatMessageC.java

OneToOneS.java | OneToOneC.java | ChatMessageS.java | ChatMessageC.java

```
1  import java.awt.BorderLayout;
2  import java.awt.event.ActionEvent;
3  import java.awt.event.ActionListener;
4  import java.awt.event.KeyEvent;
5  import java.awt.event.KeyListener;
6  import java.io.BufferedReader;
7  import java.io.BufferedWriter;
8  import java.io.IOException;
9  import java.io.InputStreamReader;
10 import java.io.OutputStreamWriter;
11 import java.net.InetAddress;
12 import java.net.Socket;
13
14 import javax.swing.JFrame;
15 import javax.swing.JLabel;
16 import javax.swing.JPanel;
17 import javax.swing.JScrollPane;
18 import javax.swing.JTextArea;
19 import javax.swing.JTextField;
20
21 public class ChatMessageC extends JFrame{
22
23     private static final long serialVersionUID = 1L;
24
25     JTextArea display;
26     JTextField wtext, ltext;
27     JLabel mlbl, wlbl, loglbl;
28
```

```
29      BufferedWriter output;
30      BufferedReader input;
31      Socket client;
32      StringBuffer clientdata;
33      String serverdata;
34      String ID;
35
36      private static final String SEPERATOR = "¦";
37      private static final int REQ_LOGIN = 1001;
38      private static final int REQ_SENDWORDS = 1021;
39
40      public ChatMessageC() {
41          // creating the frame
42          super("채팅프로그램-클라이언트");
43          setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
44          setSize(600, 500);
45          // info
46          mlbl = new JLabel("채팅 상태를 보여줍니다.");
47          getContentPane().add(BorderLayout.NORTH, mlbl);
48
49          // TextArea
50          display = new JTextArea();
51          display.setEditable(false);
52          JScrollPane scrollPane = new JScrollPane(display);
53
54          getContentPane().add(BorderLayout.CENTER, scrollPane);
55
56          // 대화말과 로그인
57          JPanel plabel = new JPanel(new BorderLayout());
```

```
58          // 대화말
59          wlbl = new JLabel(" 대화말:  ");
60          wtext = new JTextField(45); // 대화 입력 필드
61          JPanel wpanel = new JPanel(new BorderLayout()); //
62          wpanel.add(wlbl, BorderLayout.WEST);
63          wpanel.add(wtext, BorderLayout.EAST);
64          //getContentPane().add(BorderLayout.CENTER, wpanel);
65          plabel.add(wpanel, BorderLayout.CENTER);
66
67
68          // 로그인 라벨과 입력
69          JPanel ppanel = new JPanel(new BorderLayout());
70          loglbl = new JLabel(" 로그온:  ");
71          ltext = new JTextField(45); // 로그인 아이디 입력
72          ltext.addActionListener(new ActionListener() {
73
74              @Override
75              public void actionPerformed(ActionEvent arg0) {
76                  if (ID == null) {
77                      ID = ltext.getText();
78                      mlbl.setText( ID + "(으)로 로그인 하였습니다.");
79                      try {
80                          clientdata.setLength(0);
81                          clientdata.append(REQ_LOGIN).append(SEPERATOR).append(ID);
82                          output.write(clientdata.toString()+"\r\n");
83                          output.flush();
84                          ltext.setVisible(false);
85                      }catch(Exception e) {
86                          e.printStackTrace();
87                      }
88                  }
89              }
90          });
91
```

```
92          ppanel.add(loglbl, BorderLayout.WEST);
93          ppanel.add(ltext, BorderLayout.EAST);
94
95          plabel.add(ppanel,BorderLayout.SOUTH);
96          //
97          getContentPane().add(BorderLayout.SOUTH, plabel);
98
99          // 대화 입력필드에서의 키보드 처리
100         wtext.addKeyListener(new KeyListener() {
101
102             @Override
103             public void keyPressed(KeyEvent e) {
104                 if(e.getKeyChar() == KeyEvent.VK_ENTER) {
105                     String message = new String();
106                     message = wtext.getText();
107
108                     if(ID == null) {
109                         mlbl.setText("다시 로그인 하세요!!!");
110                         wtext.setText("");
111                     }else {
112                         try {
113                             clientdata.setLength(0);
114                             clientdata.append(REQ_SENDWORDS).append(SEPERATOR).append(ID).append(SEPERATOR).append(message);
115                             output.write(clientdata.toString()+"\r\n");
116                             output.flush();
117                             wtext.setText("");
118                         }catch(IOException ex) {
119                             ex.printStackTrace();
120                         }
121                     }
122                 }
123             }
124
```

```
125             @Override
126             public void keyReleased(KeyEvent e) {}
127 
128             @Override
129             public void keyTyped(KeyEvent e) {}
130 
131         });
132 
133         setVisible(true);
134     }
135 
136     public void runClient() {
137         try {
138             client = new Socket(InetAddress.getLocalHost(), 5000);
139             mlbl.setText("연결된 서버 이름 : " + client.getInetAddress().getHostName());
140             input = new BufferedReader(new InputStreamReader(client.getInputStream()));
141             output = new BufferedWriter(new OutputStreamWriter(client.getOutputStream()));
142             clientdata = new StringBuffer(2048);
143             mlbl.setText("접속 완료.  사용할 아이디를 입력하세요.");
144             while(true) {
145                 serverdata = input.readLine();
146                 display.append(serverdata + "\r\n");
147                 display.setCaretPosition(display.getDocument().getLength());
148             }
149         }catch(IOException e) {
150             e.printStackTrace();
151         }
152     }
153 
154     public static void main(String[] args) {
155         // TODO Auto-generated method stub
156         ChatMessageC c = new ChatMessageC();
157         c.runClient();
158     }
159 
160 }
```