```java
package client;

import ocsf.client.*;
import common.*;
import java.io.*;
import java.util.*;

public class ChatClient implements Observer {

  ChatIF clientUI;

  private ObservableClient client;

  public ChatClient(ObservableClient client, ChatIF clientUI) {
    this.client = client;
    this.clientUI = clientUI;
    client.addObserver(this);

    try {
      // Change in phase 3: Server prompts for login
      client.openConnection();
    } catch(IOException e) {
      handleMessageFromClientUI("#logoff");
      clientUI.display("Cannot open connection.  Awaiting command.");
    }
  }

  public void update(Observable obs, Object msg) {
    if (!(msg instanceof String))
      return;

    String message = (String)msg;

    // The first 3 if statements deal with messages sent by
    // ObservableClient when notable events occur.
    if (message.startsWith(ObservableClient.CONNECTION_CLOSED))
      clientUI.display("Connection to server closed.");
    else if (message.startsWith(ObservableClient.CONNECTION_ESTABLISHED))
      clientUI.display("Connection to server established.");
    else if (message.startsWith(ObservableClient.CONNECTION_EXCEPTION))
      clientUI.display("Connection to server lost.");
    else
      clientUI.display(message);
  }

  public void handleMessageFromClientUI(String message) {
    if (message.startsWith("#login")) {
      try {
        client.openConnection();
      } catch(IOException e) {
        clientUI.display("Cannot establish connection. Awaiting command.");
      }
      return;
    }

    //If the command is #quit.  Added in phase 2
    if (message.startsWith("#quit"))
      quit();

```

```
60      //If the command is #logoff.  Added in phase 2
61      if (message.startsWith("#logoff")) {
62        try {
63          client.closeConnection();
64        } catch(IOException e) {
65          clientUI.display("Cannot logoff normally.  Terminating client.");
66          quit();
67        }
68        return;
69      }
70
71      //If the command is #gethost.  Added in phase 2
72      if (message.startsWith("#gethost")) {
73        clientUI.display("Current host: " + client.getHost());
74        return;
75      }
76
77      //If the command is #getport.  Added in phase 2
78      if (message.startsWith("#getport")) {
79        clientUI.display("Current port: " + client.getPort());
80        return;
81      }
82
83      //If the command is #sethost.  Added in phase 2
84      if (message.startsWith("#sethost")) {
85        if (client.isConnected())
86          clientUI.display("Cannot change host while connected.");
87        else {
88          try {
89            client.setHost(message.substring(9));
90            clientUI.display("Host set to: " + client.getHost());
91          } catch(IndexOutOfBoundsException e) {
92            clientUI.display("Invalid host. Use #sethost <host>.");
93          }
94        }
95        return;
96      }
97
98      if (message.startsWith("#setport")) {
99        if (client.isConnected())
100         clientUI.display("Cannot change port while connected.");
101       else {
102         try {
103           int port = 0;
104           port = Integer.parseInt(message.substring(9));
105
106           //If the port number is invalid
107           if ((port < 1024) || (port > 65535)) {
108             clientUI.display("Invalid port number.  Port unchanged.");
109           } else {
110             client.setPort(port);
111             clientUI.display("Port set to " + port);
112           }
113         } catch(Exception e) {
114           clientUI.display("Invalid port. Use #setport <port>.");
115           clientUI.display("Port unchanged.");
116         }
117       }
118       return;
119     }
```

```
120
121     if (message.startsWith("#help") || message.startsWith("#?")) {
122       clientUI.display("\nClient-side command list:"
123       + "\n#block <loginID> -- Block messages from the specified client."
124       + "\n#channel <channel> -- Connects to the specified channel."
125       + "\n#fwd <loginID> -- Forward all messages to the specified client."
126       + "\n#getchannel -- Gets the channel the client is currently connected to."
127       + "\n#gethost -- Gets the host to which the client will connect/is connected."
128       + "\n#getport -- Gets the port on which the client will connect/is connected."
129       + "\n#help OR #? -- Lists all commands and their use."
130       + "\n#login -- Connects to a server."
131       + "\n#logoff -- Disconnects from a server."
132       + "\n#nochannel -- Returns the client to the main channel."
133       + "\n#private <loginID> <msg> -- Sends a private message to the specified
client."
134       + "\n#pub -- Sends a public message."
135       + "\n#quit -- Terminates the client and disconnects from server."
136       + "\n#sethost <newhost> -- Specify the host to connect to."
137       + "\n#setport <newport> -- Specify the port on which to connect."
138       + "\n#unblock -- Unblock messages from all blocked clients."
139       + "\n#unblock <loginID> -- Unblock messages from a specific client."
140       + "\n#unfwd -- Stop forwarding messages."
141       + "\n#whoblocksme -- List all the users who are blocking messages from you."
142       + "\n#whoiblock -- List all users you are blocking messages from."
143       + "\n#whoison -- Gets a list of all users and the channel they are connected
to.");
144       return;
145     }
146
147     //If not a client-side command or is message to be displayed
148     if ((!(message.startsWith("#")))
149        || message.startsWith("#whoison")        //Added phase 3
150        || message.startsWith("#private")        //Added phase 3
151        || message.startsWith("#channel")        //Added phase 3
152        || message.startsWith("#pub")            //Added phase 3
153        || message.startsWith("#nochannel")      //Added phase 3
154        || message.startsWith("#getchannel")     //Added phase 3
155        || message.startsWith("#fwd")            //Added phase 3
156        || message.startsWith("#unfwd")          //Added phase 3
157        || message.startsWith("#block")          //Added phase 3
158        || message.startsWith("#unblock")        //Added phase 3
159        || message.startsWith("#whoiblock")      //Added phase 3
160        || message.startsWith("#whoblocksme"))   //Added phase 3
161       try {
162         client.sendToServer(message);
163       } catch(IOException e) {
164         clientUI.display("Cannot send the message to the server. Disconnecting.");
165         try {
166           client.closeConnection();
167         } catch(IOException ex) {
168           clientUI.display("Cannot logoff normally.  Terminating client.");
169           quit();
170         }
171       }
172     else
173       clientUI.display("Invalid command.");
174   }
175
176   public void quit() {
177     try {
```

```
178       client.closeConnection();
179     } catch(IOException e) {}
180     System.exit(0);
181   }
182 }
```