```
package test;

import java.applet.*;
import java.awt.*;
import java.util.*;
import java.io.*;
import java.net.*;

/**
 * This applet was generated by a SmartGuide.
 *
 */
public class ChatApplet extends Applet implements
java.awt.event.ActionListener {
      ChatListener chatListen = null;
      Socket client = null;
      BufferedReader in = null;
      InputStreamReader inISR = null;
      private Button ivjButton1 = null;
      private Button ivjButton2 = null;
      private Button ivjButton3 = null;
      private Button ivjButton4 = null;
      private TextArea ivjChatRecord = null;
      private Dialog ivjDialog1 = null;
      private Label ivjLabel1 = null;
      private Label ivjLabel2 = null;
      private Label ivjLabel3 = null;
      private Panel ivjPanel1 = null;
      private Panel ivjPanel2 = null;
      private Button ivjSendButton = null;
      private TextField ivjTextField1 = null;
      private TextField ivjToServer = null;
      PrintWriter out = null;
      String host = "flurry";
      int port = 80;

      String version = "1.07";


/**
 * Method to handle events for the ActionListener interface.
 * @param e java.awt.event.ActionEvent
 */
/* WARNING: THIS METHOD WILL BE REGENERATED. */
public void actionPerformed(java.awt.event.ActionEvent e) {
      // user code begin {1}
      // user code end
      if ((e.getSource() == getSendButton()) ) {
            conn0(e);
      }
      if ((e.getSource() == getButton1()) ) {
            conn1(e);
      }
      if ((e.getSource() == getButton3()) ) {
            conn2(e);
      }
      if ((e.getSource() == getButton2()) ) {
            conn3(e);
      }
      if ((e.getSource() == getButton4()) ) {
```

```
                conn4(e);
        }
        // user code begin {2}
        // user code end
}

/**
 * conn0:  (SendButton.action.actionPerformed(java.awt.event.ActionEvent) --
> ChatApplet.sendText())
 * @param arg1 java.awt.event.ActionEvent
 */
/* WARNING: THIS METHOD WILL BE REGENERATED. */
private void conn0(java.awt.event.ActionEvent arg1) {
        try {
                // user code begin {1}
                // user code end
                this.sendText();
                // user code begin {2}
                // user code end
        } catch (java.lang.Throwable ivjExc) {
                // user code begin {3}
                // user code end
                handleException(ivjExc);
        }
}

/**
 * conn1:  (Button1.action.actionPerformed(java.awt.event.ActionEvent) -->
Dialog1.show())
 * @param arg1 java.awt.event.ActionEvent
 */
/* WARNING: THIS METHOD WILL BE REGENERATED. */
private void conn1(java.awt.event.ActionEvent arg1) {
        try {
                // user code begin {1}
                // user code end
                getDialog1().show();
                // user code begin {2}
                // user code end
        } catch (java.lang.Throwable ivjExc) {
                // user code begin {3}
                // user code end
                handleException(ivjExc);
        }
}

/**
 * conn2:  (Button3.action.actionPerformed(java.awt.event.ActionEvent) -->
Dialog1.dispose())
 * @param arg1 java.awt.event.ActionEvent
 */
/* WARNING: THIS METHOD WILL BE REGENERATED. */
private void conn2(java.awt.event.ActionEvent arg1) {
        try {
                // user code begin {1}
                // user code end
                getDialog1().dispose();
                // user code begin {2}
                // user code end
        } catch (java.lang.Throwable ivjExc) {
                // user code begin {3}
                // user code end
```

```
                handleException(ivjExc);
        }
}
/**
 * conn3:  (Button2.action.actionPerformed(java.awt.event.ActionEvent) -->
ChatApplet.connect())
 * @param arg1 java.awt.event.ActionEvent
 */
/* WARNING: THIS METHOD WILL BE REGENERATED. */
private void conn3(java.awt.event.ActionEvent arg1) {
        try {
                // user code begin {1}
                // user code end
                this.connect();
                // user code begin {2}
                // user code end
        } catch (java.lang.Throwable ivjExc) {
                // user code begin {3}
                // user code end
                handleException(ivjExc);
        }
}

/**
 * conn4:  (Button4.action.actionPerformed(java.awt.event.ActionEvent) -->
ChatApplet.disconnect())
 * @param arg1 java.awt.event.ActionEvent
 */
/* WARNING: THIS METHOD WILL BE REGENERATED. */
private void conn4(java.awt.event.ActionEvent arg1) {
        try {
                // user code begin {1}
                // user code end
                this.disconnect();
                // user code begin {2}
                // user code end
        } catch (java.lang.Throwable ivjExc) {
                // user code begin {3}
                // user code end
                handleException(ivjExc);
        }
}

/**
 * This method was created by a SmartGuide.
 */
public void connect( ) {

        String newPort = null;
        URL chatURL = null;

        // get alias
        String alias = ivjTextField1.getText();
        ivjDialog1.dispose();

        System.out.println("Entering connect()");
        try {
                // point to server
                chatURL = new URL("http", host, port, "/servlet/ChatServlet");
        } catch (MalformedURLException e) {
                System.out.println("URL problem: " + e);
        }
```

```
      System.out.println("Formed URL");
      try {
            // contact server
            InputStream serverIn = (InputStream) chatURL.getContent();
            BufferedReader bSI = new BufferedReader(new
InputStreamReader(serverIn));
            System.out.println("Contacted server");
            newPort = bSI.readLine();
            int thePort = new Integer(newPort).intValue();
            System.out.println("Contacted servlet and got a response of " +
newPort);

            if (thePort == 0) {
                  // can't connect to server
                  ivjChatRecord.append("Sorry, cannot connect to server;
try later.\n");
                  return;
            }

            // attach to new server port
            client = new Socket(host, thePort);

            // get i/o streams
            inISR = new InputStreamReader(client.getInputStream());
            in = new BufferedReader(inISR);
            out = new PrintWriter(client.getOutputStream(),true);

      } catch (IOException e) {
            System.out.println("IOException:  " + e);
      }

      // send the name string
      out.println(alias);

      // establish a listener for the server
      chatListen = new ChatListener(in, ivjChatRecord);
      chatListen.start();

      return;
}
 /**
 * This method was created by a SmartGuide.
 */
public void disconnect( ) {

      // stop the server
      out.println("QUIT");

      // now wait for the listener to quit
      chatListen.stopListening();

      // close all the things
      try {
            in.close();
            out.close();
            client.close();
      } catch (IOException e) {
            System.out.println("IOException:  " + e);
      }

      return;
```

```
}
/**
 * Gets the applet information.
 * @return java.lang.String
 */
public String getAppletInfo() {
      return "test.ChatApplet created using VisualAge for Java.";
}
/**
 * Return the Button1 property value.
 * @return java.awt.Button
 */
/* WARNING: THIS METHOD WILL BE REGENERATED. */
private Button getButton1() {
      if (ivjButton1 == null) {
            try {
                  ivjButton1 = new java.awt.Button();
                  ivjButton1.setName("Button1");
                  ivjButton1.setLabel("Connect");
                  // user code begin {1}
                  // user code end
            } catch (java.lang.Throwable ivjExc) {
                  // user code begin {2}
                  // user code end
                  handleException(ivjExc);
            }
      };
      return ivjButton1;
}

/**
 * Return the Button2 property value.
 * @return java.awt.Button
 */
/* WARNING: THIS METHOD WILL BE REGENERATED. */
private Button getButton2() {
      if (ivjButton2 == null) {
            try {
                  ivjButton2 = new java.awt.Button();
                  ivjButton2.setName("Button2");
                  ivjButton2.setLabel("OK");
                  // user code begin {1}
                  // user code end
            } catch (java.lang.Throwable ivjExc) {
                  // user code begin {2}
                  // user code end
                  handleException(ivjExc);
            }
      };
      return ivjButton2;
}
/**
 * Return the Button3 property value.
 * @return java.awt.Button
 */
/* WARNING: THIS METHOD WILL BE REGENERATED. */
private Button getButton3() {
      if (ivjButton3 == null) {
            try {
                  ivjButton3 = new java.awt.Button();
                  ivjButton3.setName("Button3");
                  ivjButton3.setLabel("Cancel");
```

```
                        // user code begin {1}
                        // user code end
                } catch (java.lang.Throwable ivjExc) {
                        // user code begin {2}
                        // user code end
                        handleException(ivjExc);
                }
        };
        return ivjButton3;
}

/**
 * Return the Button4 property value.
 * @return java.awt.Button
 */
/* WARNING: THIS METHOD WILL BE REGENERATED. */
private Button getButton4() {
        if (ivjButton4 == null) {
                try {
                        ivjButton4 = new java.awt.Button();
                        ivjButton4.setName("Button4");
                        ivjButton4.setLabel("Disconnect");
                        // user code begin {1}
                        // user code end
                } catch (java.lang.Throwable ivjExc) {
                        // user code begin {2}
                        // user code end
                        handleException(ivjExc);
                }
        };
        return ivjButton4;
}
/**
 * Return the ChatRecord property value.
 * @return java.awt.TextArea
 */
/* WARNING: THIS METHOD WILL BE REGENERATED. */
private TextArea getChatRecord() {
        if (ivjChatRecord == null) {
                try {
                        ivjChatRecord = new java.awt.TextArea();
                        ivjChatRecord.setName("ChatRecord");
                        // user code begin {1}
                        // user code end
                } catch (java.lang.Throwable ivjExc) {
                        // user code begin {2}
                        // user code end
                        handleException(ivjExc);
                }
        };
        return ivjChatRecord;
}

/**
 * Return the Dialog1 property value.
 * @return java.awt.Dialog
 */
/* WARNING: THIS METHOD WILL BE REGENERATED. */
private Dialog getDialog1() {
        java.awt.GridBagConstraints constraintsLabel3 = new
java.awt.GridBagConstraints();
        java.awt.GridBagConstraints constraintsTextField1 = new
```

```
java.awt.GridBagConstraints();
      java.awt.GridBagConstraints constraintsButton2 = new java.awt.GridBagConstraints();
      java.awt.GridBagConstraints constraintsButton3 = new java.awt.GridBagConstraints();
      if (ivjDialog1 == null) {
            try {
                  ivjDialog1 = new java.awt.Dialog(new java.awt.Frame());
                  ivjDialog1.setName("Dialog1");
                  ivjDialog1.setLayout(new java.awt.GridBagLayout());
                  ivjDialog1.setBounds(23, 284, 405, 99);
                  ivjDialog1.setTitle("Chat Connection");

                  constraintsLabel3.gridx = 0; constraintsLabel3.gridy = 0;
                  constraintsLabel3.gridwidth = 1; constraintsLabel3.gridheight = 1;
                  constraintsLabel3.anchor = java.awt.GridBagConstraints.WEST;
                  constraintsLabel3.weightx = 0.0;
                  constraintsLabel3.weighty = 0.0;
                  constraintsLabel3.insets = new java.awt.Insets(10, 2, 0, 0);
                  ((java.awt.GridBagLayout)
getDialog1().getLayout()).setConstraints(getLabel3(), constraintsLabel3);
                  getDialog1().add(getLabel3());

                  constraintsTextField1.gridx = 1;constraintsTextField1.gridy = 0;
                  constraintsTextField1.gridwidth = 3;constraintsTextField1.gridheight = 1;
                  constraintsTextField1.fill = java.awt.GridBagConstraints.HORIZONTAL;
                  constraintsTextField1.anchor = java.awt.GridBagConstraints.EAST;
                  constraintsTextField1.weightx = 1.0;
                  constraintsTextField1.weighty = 0.0;
                  constraintsTextField1.insets=new java.awt.Insets(10, 0, 0, 2);
                  ((java.awt.GridBagLayout)
getDialog1().getLayout()).setConstraints(getTextField1(), constraintsTextField1);
                  getDialog1().add(getTextField1());

                  constraintsButton2.gridx = 0; constraintsButton2.gridy = 1;
                  constraintsButton2.gridwidth = 1; constraintsButton2.gridheight = 1;
                  constraintsButton2.fill = java.awt.GridBagConstraints.HORIZONTAL;
                  constraintsButton2.anchor = java.awt.GridBagConstraints.WEST;
                  constraintsButton2.weightx = 0.0;
                  constraintsButton2.weighty = 1.0;
                  constraintsButton2.insets = new java.awt.Insets(0, 2, 0, 10);
                  ((java.awt.GridBagLayout)
getDialog1().getLayout()).setConstraints(getButton2(), constraintsButton2);
                  getDialog1().add(getButton2());

                  constraintsButton3.gridx = 1; constraintsButton3.gridy = 1;
                  constraintsButton3.gridwidth = 3; constraintsButton3.gridheight = 1;
                  constraintsButton3.fill = java.awt.GridBagConstraints.HORIZONTAL;
                  constraintsButton3.anchor = java.awt.GridBagConstraints.CENTER;
                  constraintsButton3.weightx = 1.0;
                  constraintsButton3.weighty = 1.0;
                  constraintsButton3.insets = new java.awt.Insets(0, 10, 0, 0);
                  ((java.awt.GridBagLayout)
getDialog1().getLayout()).setConstraints(getButton3(), constraintsButton3);
                  getDialog1().add(getButton3());
                  // user code begin {1}
                  // user code end
            } catch (java.lang.Throwable ivjExc) {
                  // user code begin {2}
                  // user code end
                  handleException(ivjExc);
            }
      };
      return ivjDialog1;
}
```

```
/**
 * Return the Label1 property value.
 * @return java.awt.Label
 */
/* WARNING: THIS METHOD WILL BE REGENERATED. */
private Label getLabel1() {
      if (ivjLabel1 == null) {
            try {
                  ivjLabel1 = new java.awt.Label();
                  ivjLabel1.setName("Label1");
                  ivjLabel1.setText("Send This:");
                  // user code begin {1}
                  // user code end
            } catch (java.lang.Throwable ivjExc) {
                  // user code begin {2}
                  // user code end
                  handleException(ivjExc);
            }
      };
      return ivjLabel1;
}

/**
 * Return the Label2 property value.
 * @return java.awt.Label
 */
/* WARNING: THIS METHOD WILL BE REGENERATED. */
private Label getLabel2() {
      if (ivjLabel2 == null) {
            try {
                  ivjLabel2 = new java.awt.Label();
                  ivjLabel2.setName("Label2");
                  ivjLabel2.setText("Message Area:");
                  // user code begin {1}
                  // user code end
            } catch (java.lang.Throwable ivjExc) {
                  // user code begin {2}
                  // user code end
                  handleException(ivjExc);
            }
      };
      return ivjLabel2;
}

/**
 * Return the Label3 property value.
 * @return java.awt.Label
 */
/* WARNING: THIS METHOD WILL BE REGENERATED. */
private Label getLabel3() {
      if (ivjLabel3 == null) {
            try {
                  ivjLabel3 = new java.awt.Label();
                  ivjLabel3.setName("Label3");
                  ivjLabel3.setText("Enter your Chat Alias:");
                  // user code begin {1}
                  // user code end
            } catch (java.lang.Throwable ivjExc) {
                  // user code begin {2}
                  // user code end
                  handleException(ivjExc);
```

```
            }
      };
      return ivjLabel3;
}

/**
 * Return the Panel1 property value.
 * @return java.awt.Panel
 */
/* WARNING: THIS METHOD WILL BE REGENERATED. */
private Panel getPanel1() {
      java.awt.GridBagConstraints constraintsLabel1 = new java.awt.GridBagConstraints();
      java.awt.GridBagConstraints constraintsToServer = new
java.awt.GridBagConstraints();
      java.awt.GridBagConstraints constraintsSendButton = new
java.awt.GridBagConstraints();
      java.awt.GridBagConstraints constraintsButton1 = new
java.awt.GridBagConstraints();
      java.awt.GridBagConstraints constraintsButton4 = new
java.awt.GridBagConstraints();
      if (ivjPanel1 == null) {
            try {
                  ivjPanel1 = new java.awt.Panel();
                  ivjPanel1.setName("Panel1");
                  ivjPanel1.setLayout(new java.awt.GridBagLayout());

                  constraintsLabel1.gridx = 0; constraintsLabel1.gridy = 0;
                  constraintsLabel1.gridwidth = 1; constraintsLabel1.gridheight = 1;
                  constraintsLabel1.anchor = java.awt.GridBagConstraints.WEST;
                  constraintsLabel1.weightx = 0.0;
                  constraintsLabel1.weighty = 1.0;
                  constraintsLabel1.insets = new java.awt.Insets(0, 2, 0, 0);
                  ((java.awt.GridBagLayout)
getPanel1().getLayout()).setConstraints(getLabel1(), constraintsLabel1);
                  getPanel1().add(getLabel1());

                  constraintsToServer.gridx = 1; constraintsToServer.gridy = 0;
                  constraintsToServer.gridwidth = 7; constraintsToServer.gridheight = 1;
                  constraintsToServer.fill = java.awt.GridBagConstraints.HORIZONTAL;
                  constraintsToServer.anchor = java.awt.GridBagConstraints.CENTER;
                  constraintsToServer.weightx = 1.0;
                  constraintsToServer.weighty = 1.0;
                  constraintsToServer.insets = new java.awt.Insets(0, 0, 0, 2);
                  ((java.awt.GridBagLayout)
getPanel1().getLayout()).setConstraints(getToServer(), constraintsToServer);
                  getPanel1().add(getToServer());

                  constraintsSendButton.gridx = 7; constraintsSendButton.gridy = 1;
                  constraintsSendButton.gridwidth = 1;
constraintsSendButton.gridheight = 1;
                  constraintsSendButton.anchor = java.awt.GridBagConstraints.EAST;
                  constraintsSendButton.weightx = 0.0;
                  constraintsSendButton.weighty = 1.0;
                  constraintsSendButton.insets = new java.awt.Insets(0, 0, 0, 2);
                  ((java.awt.GridBagLayout)
getPanel1().getLayout()).setConstraints(getSendButton(), constraintsSendButton);
                  getPanel1().add(getSendButton());

                  constraintsButton1.gridx = 0; constraintsButton1.gridy = 1;
                  constraintsButton1.gridwidth = 1; constraintsButton1.gridheight = 1;
                  constraintsButton1.fill = java.awt.GridBagConstraints.HORIZONTAL;
                  constraintsButton1.anchor = java.awt.GridBagConstraints.WEST;
```

```
                    constraintsButton1.weightx = 0.0;
                    constraintsButton1.weighty = 1.0;
                    constraintsButton1.insets = new java.awt.Insets(0, 2, 0, 0);
                    ((java.awt.GridBagLayout)
getPanel1().getLayout()).setConstraints(getButton1(), constraintsButton1);
                    getPanel1().add(getButton1());

                    constraintsButton4.gridx = 2; constraintsButton4.gridy = 1;
                    constraintsButton4.gridwidth = 1; constraintsButton4.gridheight = 1;
                    constraintsButton4.anchor = java.awt.GridBagConstraints.CENTER;
                    constraintsButton4.weightx = 1.0;
                    constraintsButton4.weighty = 0.0;
                    constraintsButton4.insets = new java.awt.Insets(0, 2, 0, 2);
                    ((java.awt.GridBagLayout)
getPanel1().getLayout()).setConstraints(getButton4(), constraintsButton4);
                    getPanel1().add(getButton4());
                    // user code begin {1}
                    // user code end
             } catch (java.lang.Throwable ivjExc) {
                    // user code begin {2}
                    // user code end
                    handleException(ivjExc);
             }
      };
      return ivjPanel1;
}

/**
 * Return the Panel2 property value.
 * @return java.awt.Panel
 */
/* WARNING: THIS METHOD WILL BE REGENERATED. */
private Panel getPanel2() {
      java.awt.GridBagConstraints constraintsLabel2 = new java.awt.GridBagConstraints();
      java.awt.GridBagConstraints constraintsChatRecord = new
java.awt.GridBagConstraints();
      if (ivjPanel2 == null) {
             try {
                    ivjPanel2 = new java.awt.Panel();
                    ivjPanel2.setName("Panel2");
                    ivjPanel2.setLayout(new java.awt.GridBagLayout());

                    constraintsLabel2.gridx = 0; constraintsLabel2.gridy = 0;
                    constraintsLabel2.gridwidth = 1; constraintsLabel2.gridheight = 1;
                    constraintsLabel2.fill = java.awt.GridBagConstraints.HORIZONTAL;
                    constraintsLabel2.anchor = java.awt.GridBagConstraints.WEST;
                    constraintsLabel2.weightx = 1.0;
                    constraintsLabel2.weighty = 0.0;
                    ((java.awt.GridBagLayout)
getPanel2().getLayout()).setConstraints(getLabel2(), constraintsLabel2);
                    getPanel2().add(getLabel2());

                    constraintsChatRecord.gridx = 0; constraintsChatRecord.gridy = 1;
                    constraintsChatRecord.gridwidth = 4;
constraintsChatRecord.gridheight = 1;
                    constraintsChatRecord.fill = java.awt.GridBagConstraints.BOTH;
                    constraintsChatRecord.anchor = java.awt.GridBagConstraints.CENTER;
                    constraintsChatRecord.weightx = 1.0;
                    constraintsChatRecord.weighty = 1.0;
                    ((java.awt.GridBagLayout)
getPanel2().getLayout()).setConstraints(getChatRecord(), constraintsChatRecord);
                    getPanel2().add(getChatRecord());
```

```
                        // user code begin {1}
                        // user code end
                } catch (java.lang.Throwable ivjExc) {
                        // user code begin {2}
                        // user code end
                        handleException(ivjExc);
                }
        };
        return ivjPanel2;
}

/**
 * Return the SendButton property value.
 * @return java.awt.Button
 */
/* WARNING: THIS METHOD WILL BE REGENERATED. */
private Button getSendButton() {
        if (ivjSendButton == null) {
                try {
                        ivjSendButton = new java.awt.Button();
                        ivjSendButton.setName("SendButton");
                        ivjSendButton.setLabel("Send");
                        // user code begin {1}
                        // user code end
                } catch (java.lang.Throwable ivjExc) {
                        // user code begin {2}
                        // user code end
                        handleException(ivjExc);
                }
        };
        return ivjSendButton;
}

/**
 * Return the TextField1 property value.
 * @return java.awt.TextField
 */
/* WARNING: THIS METHOD WILL BE REGENERATED. */
private TextField getTextField1() {
        if (ivjTextField1 == null) {
                try {
                        ivjTextField1 = new java.awt.TextField();
                        ivjTextField1.setName("TextField1");
                        // user code begin {1}
                        // user code end
                } catch (java.lang.Throwable ivjExc) {
                        // user code begin {2}
                        // user code end
                        handleException(ivjExc);
                }
        };
        return ivjTextField1;
}

/**
 * Return the ToServer property value.
 * @return java.awt.TextField
 */
/* WARNING: THIS METHOD WILL BE REGENERATED. */
private TextField getToServer() {
        if (ivjToServer == null) {
                try {
```

```
                    ivjToServer = new java.awt.TextField();
                    ivjToServer.setName("ToServer");
                    // user code begin {1}
                    // user code end
              } catch (java.lang.Throwable ivjExc) {
                    // user code begin {2}
                    // user code end
                    handleException(ivjExc);
              }
      };
      return ivjToServer;
}

/**
 * Called whenever the part throws an exception.
 * @param exception java.lang.Throwable
 */
private void handleException(Throwable exception) {

      /* Uncomment the following lines to print uncaught exceptions to stdout
*/
      System.out.println("--------- UNCAUGHT EXCEPTION ---------" +
exception);
      // exception.printStackTrace(System.out);
}

/**
 * Handle the Applet init method.
 */
/* WARNING: THIS METHOD WILL BE REGENERATED. */
public void init() {
      super.init();
      try {
              System.out.println("Hey, starting the ChatApplet version
"+version);
              java.awt.GridBagConstraints constraintsPanel1 = new
java.awt.GridBagConstraints();
              java.awt.GridBagConstraints constraintsPanel2 = new
java.awt.GridBagConstraints();
              setName("ChatApplet");
              setLayout(new java.awt.GridBagLayout());
              setSize(426, 240);

              constraintsPanel1.gridx = 0; constraintsPanel1.gridy = 0;
              constraintsPanel1.gridwidth = 1; constraintsPanel1.gridheight = 1;
              constraintsPanel1.fill = java.awt.GridBagConstraints.BOTH;
              constraintsPanel1.anchor = java.awt.GridBagConstraints.CENTER;
              constraintsPanel1.weightx = 1.0;
              constraintsPanel1.weighty = 0.0;
              ((java.awt.GridBagLayout)
this.getLayout()).setConstraints(getPanel1(), constraintsPanel1);
              this.add(getPanel1());

              constraintsPanel2.gridx = 0; constraintsPanel2.gridy = 1;
              constraintsPanel2.gridwidth = 1; constraintsPanel2.gridheight = 4;
              constraintsPanel2.fill = java.awt.GridBagConstraints.BOTH;
              constraintsPanel2.anchor = java.awt.GridBagConstraints.CENTER;
              constraintsPanel2.weightx = 1.0;
              constraintsPanel2.weighty = 1.0;
              constraintsPanel2.insets = new java.awt.Insets(10, 2, 2, 2);
              ((java.awt.GridBagLayout)
this.getLayout()).setConstraints(getPanel2(), constraintsPanel2);
```

```
                this.add(getPanel2());
                initConnections();
                // user code begin {1}
                System.out.println("ChatApplet version "+version+" started");
                String hostname = getParameter("hostname");
                String hostport = getParameter("hostport");
                if (hostname != null) { host = hostname; }
                if (hostport != null) { port = new Integer(hostport).intValue();
}
                // user code end
        } catch (java.lang.Throwable ivjExc) {
                // user code begin {2}
                // user code end
                handleException(ivjExc);
        }
}

/**
 * Initializes connections
 */
/* WARNING: THIS METHOD WILL BE REGENERATED. */
private void initConnections() {
        // user code begin {1}
        // user code end
        getSendButton().addActionListener(this);
        getButton1().addActionListener(this);
        getButton3().addActionListener(this);
        getButton2().addActionListener(this);
        getButton4().addActionListener(this);
}

/**
 * main entrypoint - starts the part when it is run as an application
 * @param args java.lang.String[]
 */
public static void main(java.lang.String[] args) {
        try {
                java.awt.Frame frame;
                try {
                        Class aFrameClass =
Class.forName("uvm.abt.edit.TestFrame");
                        frame = (java.awt.Frame)aFrameClass.newInstance();
                } catch (java.lang.Throwable ivjExc) {
                        frame = new java.awt.Frame();
                }
                test.ChatApplet aChatApplet = new test.ChatApplet();
                frame.add("Center", aChatApplet);
                frame.setSize(aChatApplet.getSize());
                frame.setSize(400,240);
                aChatApplet.init();
                aChatApplet.start();
                frame.setVisible(true);

                aChatApplet.destroy();
        } catch (Throwable exception) {
                System.err.println("Exception occurred in main() of
java.applet.Applet");
        }
}

/**
 * This method was created by a SmartGuide.
```

```
 */
public void sendText( ) {
      //System.out.println("would send now - client");
      String request = ivjToServer.getText();
      out.println(request);
      return;
}
}
```

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.net.*;

public
class ChatServlet extends HttpServlet {

  int basePort = 0;
  int chatClient = 0;

  PrintWriter clientOut[] = null;
  int clientPort[] = null;

  String version = "1.03";

  public void init(ServletConfig config)
       throws ServletException
  {

    super.init(config);

    // here perhaps would set up the maximum number of clients
    int maxLength = 4;
    // initialize the servlet state
    clientOut = new PrintWriter[maxLength];
    clientPort = new int[maxLength];
    for (int i=0; i<clientOut.length; i++) {
      clientOut[i] = null;
      clientPort[i] = 0;
    }
    // set up the base port number for communications
    basePort = 9000;
  }

  public void destroy() {
    // close well known port
    log("about to destroy");
  }
  public void doGet(HttpServletRequest req, HttpServletResponse res)
       throws ServletException, IOException
  {

    int me;

    // set up for returning information
    log("entering doGet");
    ServletOutputStream bout = res.getOutputStream();
    res.setContentType("text/plain");
    // get client number and port (find unused output slot)
    synchronized (clientPort) {
      for (me=0; me<clientPort.length; me++) {
     if (clientPort[me] == 0) {break;}
      }
      // if found an open slot
      if (me < clientOut.length) {
     // reserve the slot
     clientPort[me] = basePort + me;
      }
    }
```

```
    // if did not find an open slot
    if (me == clientOut.length) {
      // indicate can't support another client
      log("doGet, overbooked");
      bout.println(""+0);
    } else {
      // send port back to applet
      log("doGet, accepting another client");
      bout.println(""+clientPort[me]);
      // start a new server thread to listen to client
      ServerListener chatter = new ServerListener(me, clientPort, clientOut,
this);
      log("doGet, about to start chatter");
      chatter.start();
    }
    // finish off
    bout.flush();
    return;
  }
}


class ServerListener extends Thread {

  int me;
  int myPort;
  PrintWriter clientOut[] = null;
  int clientPort[] = null;
  HttpServlet servlet = null;

  ServerSocket ssClient = null;
  Socket socket = null;
  InputStreamReader inISR = null;
  BufferedReader in = null;

  String version = "1.04";

  public ServerListener (int index, int[] port, PrintWriter[] out,
HttpServlet it) {
    super();

    // set up information
    it.log("ServerListener version "+version+": port="+port[index]);
    me = index;
    clientPort = port;
    myPort = clientPort[me];
    clientOut = out;
    servlet = it;
  }

  public void run()
  {

    boolean keepon = true;

    ServerSocket ssClient = null;
    Socket socket = null;
    InputStreamReader inISR = null;
    BufferedReader in = null;

    try {
      // wait for a client to connect
```

```
      servlet.log("ServerListener "+version+ " waiting for client connection
to: "+myPort);
      // get server socket
      ssClient = new ServerSocket(myPort);
      servlet.log("ServerListener: got ServerSocket("+myPort+")");

      // wait for a client to connect
      servlet.log("ServerListener: waiting for client connection client
"+me);
      socket = ssClient.accept();
      servlet.log("ServerListener: got client connection client");

      // get input and output streams (output with autoflush)
      inISR = new InputStreamReader(socket.getInputStream());
      in = new BufferedReader(inISR);
      synchronized (clientPort) {
     clientOut[me] = new PrintWriter(socket.getOutputStream(),true);
      }

      // get alias
      String alias = in.readLine();

      // send welcome message
      clientOut[me].println("Welcome to Chat 1.0, " + alias);

      // service messages
      while (keepon) {

     // get input from client
     servlet.log("ServerListener: waiting on client "+me);
     String request = in.readLine();

     // if quitting
     if (request.startsWith("QUIT")) {
       // finish up
       servlet.log("ServerListener: about to send to client");
       clientOut[me].println("Goodbye, " + alias);
       servlet.log("ServerListener: sent QUIT to "+me);
       keepon = false;
     } else {
       // send input to all clients
       synchronized (clientPort) {
         for (int i=0; i<clientOut.length; i++) {
           // if there is a client in this slot
           if (clientOut[i] != null) {
            // send to client
            servlet.log("ServerListener: about to send to client "+i);
            clientOut[i].println(alias + ": " + request);
            servlet.log("ServerListener: sent to client");
           }
         }
       }
     }
      }

      // close everything
      servlet.log("ServerListener: about to quit "+me);
      synchronized (clientOut) {
     clientOut[me].close();
     clientOut[me] = null;
     clientPort[me] = 0;
      }
```

```
        in.close();
        inISR.close();
        socket.close();
        ssClient.close();

      } catch (IOException e) {
        servlet.log("ServerListener IOException: " +  e);
      }
    }

  }
```

The Chat Listener

```
package test;

import java.io.*;
import java.net.*;
import java.awt.*;
/**
 * This class was generated by a SmartGuide.
 *
 */
public class ChatListener extends Thread {
      BufferedReader inStream = null;
      TextArea outTarget = null;

      boolean done = false;


/**
 * ChatListener constructor comment.
 */
public ChatListener() {
      super();
}
/**
 * This method was created by a SmartGuide.
 * @param in java.io.BufferedReader
 * @param in2 java.awt.TextArea
 */
public ChatListener ( BufferedReader in, TextArea ta) {
      inStream = in;
      outTarget = ta;
}
/**
 * ChatListener constructor comment.
 * @param target java.lang.Runnable
 */
public ChatListener(Runnable target) {
      super(target);
}
/**
 * ChatListener constructor comment.
 * @param target java.lang.Runnable
 * @param name java.lang.String
 */
public ChatListener(Runnable target, String name) {
      super(target, name);
}
/**
 * ChatListener constructor comment.
 * @param name java.lang.String
 */
public ChatListener(String name) {
      super(name);
}
/**
 * ChatListener constructor comment.
 * @param group java.lang.ThreadGroup
 * @param target java.lang.Runnable
 */
public ChatListener(ThreadGroup group, Runnable target) {
      super(group, target);
```

```
}
/**
 * ChatListener constructor comment.
 * @param group java.lang.ThreadGroup
 * @param target java.lang.Runnable
 * @param name java.lang.String
 */
public ChatListener(ThreadGroup group, Runnable target, String name) {
      super(group, target, name);
}
/**
 * ChatListener constructor comment.
 * @param group java.lang.ThreadGroup
 * @param name java.lang.String
 */
public ChatListener(ThreadGroup group, String name) {
      super(group, name);
}
/**
 * This method was created by a SmartGuide.
 */
public void run( ) {

      // set up to run as long as user wants
      done = false;

      // if sufficient information to proceed
      if ((inStream != null) && (outTarget != null)) {
            // run forever
            while (!done) {
                  // wait for input
                  try {
                        String message = inStream.readLine();
                        // write it out
                        outTarget.append(message + "\n");
                        // check for user finished
                        if (message.startsWith("Goodbye")) {
                              // indicate we are finished
                              done = true;
                        }
                  } catch (IOException e) {
                        System.out.println("Server socket problem in
ChatListener: " +  e);
                  }
            }
      }

      // indicate finished
      done = true;
      return;
}
/**
 * This method was created by a SmartGuide.
 */
public void stopListening() {

      // wait for the run method to finish
      while(!done) {}

      return;
}
}
```