Integrating JabberBeans with Java Servlet Technology

Mark T. Yoshikawa
HP Laboratories Palo Alto
HPL-2002-139
May 13th , 2002*

E-mail: marky@hpl.hp.com

Jabber, JabberBeans, Instant Messaging, XML-RPC, Tomcat, servlet

Open Source technologies are interesting in that they allow free experimentation and integration by anyone, anywhere. The Jabber system and the Jakarta Tomcat server are two such examples. In this technical report, the integration of JabberBeans code with servlet technology via the Tomcat servlet engine will be demonstrated. We will quickly examine these technologies and explain why they are interesting. This is followed by a discussion on installing, configuring and coding the software necessary to perform your own demonstration.

Integrating JabberBeans with Java Servlet Technology

Mark T. Yoshikawa
Hewlett Packard Laboratories
1501 Page Mill Road, 2L
Palo Alto, CA 94304-1100, USA
+1-650-857-5831
marky@hpl.hp.com
May 13, 2002

ABSTRACT

Open Source technologies are interesting in that they allow free experimentation and integration by anyone, anywhere. The Jabber system and the Jakarta Tomcat server are two such examples. In this technical report, the integration of JabberBeans code with servlet technology via the Tomcat servlet engine will be demonstrated. We will quickly examine these technologies and explain why they are interesting. This is followed by a discussion on installing, configuring and coding the software necessary to perform your own demonstration.

Keywords

Jabber, JabberBeans, Instant Messaging, XML-RPC, Tomcat, servlet

Introduction

JabberBeans [1], a JavaBean application, enables the integration of Java and various Java-related technologies in Jabber [2], an Open-Source, XML-based, instant messaging client/server application set. This makes Jabber's XML messaging protocol [3] open for development and applicable to other research and organizational needs. One such integration we might consider is Java Servlet technology. Can we integrate Jabber's IM client system with the Web? This report demonstrates one method to do this.

Jabber and JabberBeans

Jabber's protocol is Open Source and is being applied in many ways beyond Instant Messaging. Jabber has the potential to open up various internal and external research opportunities for HP. Here are a number of efforts relating to Jabber.

Jabber Activities at HP

HP Operations (HPO) has jabber.hp.com [4], a Jabber Server for HP's North American Delivery Operations. HPO is also working on licensing for an enterprise version of the server and client from Jabber, Inc [5]. ISE (Infrastructure Strategic Engineering) is investigating Secure real-time XML messaging [6] using Jabber and has demonstrated having an IM buddy which is not a human but a computer which will give information when asked. For more see ISE's Jabber Protocol Awareness report [7]. (See this informative paper for more on activities at HP.) HP Labs is investigating Jabber as well and have contributed to the Open Source Jabber efforts [8]. Internally, HP Labs has contributed to Corporate Source [9] a transport module which extends the Jabber Server called j2j. It connects Jabber's IM system to the Java Agent DEvelopment Framework (JADE) [10].

Jabber's Broad-based Interest

Current efforts regarding Jabber are significant not only within HP but elsewhere. They come in a variety of licensing schemes such as GNU GPL [11], JOSL (Jabber Open Source License) [12], proprietary (available at no cost), and proprietary (with cost). Keep in mind that many of these efforts are in the beginning states of development, yet this highlights the fact that there are many Jabber opportunities.

Here are some examples of these efforts: IBM has a Jabber client written using their Sash development environment that uses HTML, XML, and JavaScript to write programs which look like Windows programs [13]. There is an XML editor called the XML Cooktop [14] that has a built-in Jabber client to do collaboration. Many other Jabber clients are listed at JabberCentral's clients site [15]. The Jabber server itself is Open Source and may be downloaded and used at no monetary cost [16]. Jabber, Inc. sells an enterprise ready Jabber server as well as services to brand their Jabber Instant Messenger (JIM) client to a company's needs. Also the Walt Disney Internet Group, and BellSouth Corporation have contracts with Jabber, Inc. One person began a project to integrate Jabber with Emacs [17]. Muse is another API for various IM platforms including Jabber. The Jabber protocol is being extended to interoperate with XML-RPC (Jabber-RPC) [18]. There are many more applications of the Jabber protocol such as APIs. JabberBeans is the one described in this paper.

Jabber is a technology as well as a protocol

As a technology, there are tangible attributes such as a Jabber Instant Messaging client application, Perl modules, APIs, and so forth. However, these would not be practical without establishing Jabber as a protocol.

That protocol is maintained by the Jabber Software Foundation and has been submitted as a draft to the Internet Engineering Task Force (IETF) [19]. Any reference to that work should be viewed as a work in progress.

JabberBeans is an application of JavaBeans capability for the Jabber Instant Messaging platform

JabberBeans is a JavaBeans API to the Jabber protocol, making that protocol available to any application written in Java. JabberBeans was written by David Waite and is Open Source. Note that the current revision at the time of this writing was 0.9.0-pre4. The build of the JabberBean API source and example source code were done with Java SE 1.3.1_01 [20].

Tomcat

Tomcat is an Open Source reference implementation of Java Servlet and JavaServer Pages specifications. Tomcat is part of the Open Source project named Jakarta which is a sub-project of the extremely popular Apache project [21]. The specifications for Java Servlet and JavaServer Pages come from Sun Microsystems. Java servlets are server-executable Java classes that enable dynamic web interactions.

Since both JabberBeans and Tomcat are Open Source efforts which beckon experimentation, the question is whether we can we integrate JabberBeans with servlet technology such as Tomcat? Here we show how.

Integrating JabberBeans with Tomcat 4.0

Downloading Jabber Beans

Go to http://jabberbeans.org/ and follow the "Download" link to the download area for JabberBeans. There you will find *three* different zip files:

JabberBeans-0.9.0-pre4-dist.zip (recommended)
To compile and run your own applications using JabberBean and/or want the JabberBeans source code to examine or build your own *jabberbeans.jar* file, download this file. To build, you will need to edit the build script, for your JAVA_HOME, and define the "cp" environment variable. So to build correctly, without errors you need have the following line (note the addition of jaxp.jar):

```
set
cp=.\lib\ant.jar;.\lib\crimson.jar;.\lib\jaxp.jar;%JAVA_HOME%\lib\tools
.jar
```

Which originally read:

```
set cp=.\lib\ant.jar;.\lib\projectx-tr2.jar;%JAVA_HOME%\lib\tools.jar
```

Note that *projectx-tr2.jar* does not come with the download. Now you should be able to build the *jabberbeans.jar* file.

JabberBeans-0.9.0-pre4-docs.zip (recommended)
If you want the javadoc loaded locally, you can place this in the *docs* folder of the *JabberBeans-0.9.0-pre4* distribution directory. However, there is a link to the same online javadoc at http://jabberbeans.org/.

JabberBeans-0.9.0-pre4-bin.zip
To compile and run your own applications using JabberBeans with the documentation and example code, download this version. After unpacking the download you can use the *jabberbeans.jar* file in your classpath.

Test your JabberBeans installation

Once you have added the jabberbeans.jar file or path to your classpath, you are ready to compile the example programs in the tree *<installation directory>\JabberBeans-0.9.0-pre4\src\jabberbeans-tests\simple-case*. Note that you will be prompted to use the javac option: *–deprecation* when compiling if your Java SDK version is greater than 1.2. Also in some of the examples you may need to *implement* some code. For instance, when compiling *jtest2.java* two errors will appear indicating that "jtest2.JTestPacketListener" and "jtest2.JTestConnectionListener should be declared abstract." This is because you must implement the two missing methods, one in the static class *JTestConnectionListener* and the other in the static class *JTestPacketListener*. Adding the following code as instructed will suffice.

```
// Add to JTestConnectionListener static class in jtest2.java
public void connectionChanged(ConnectionEvent ce)
{
    System.out.println("CLDebug: Connection Changed");
}

// Add to JTestPacketListener static class in jtest2.java
public void sendFailed(PacketEvent pe)
{
    System.out.println("PLDebug: Send Failed : " +
    pe.getPacket().toString());
}
```

Note that the example programs come only with the *JabberBeans-0.9.0-pre4-dist.zip* file, which is why it is recommended.

Download Tomcat and make it JabberBeans aware

1. Follow the directions on the Tomcat 4.0 tutorial, found at: http://javaboutique.internet.com/tutorials/Tomcat/.
2. Add *jabberbeans.jar* to the JDK classpath in addition to servlet.jar in the tutorial. (See Appendix A for setting up the Windows environment batch file.)
3. Copy JabberBeans.jar to *%Catalina_Home%\Common\Lib* directory. (For me this worked w/o additional configuration, although your Tomcat setup may be different.)

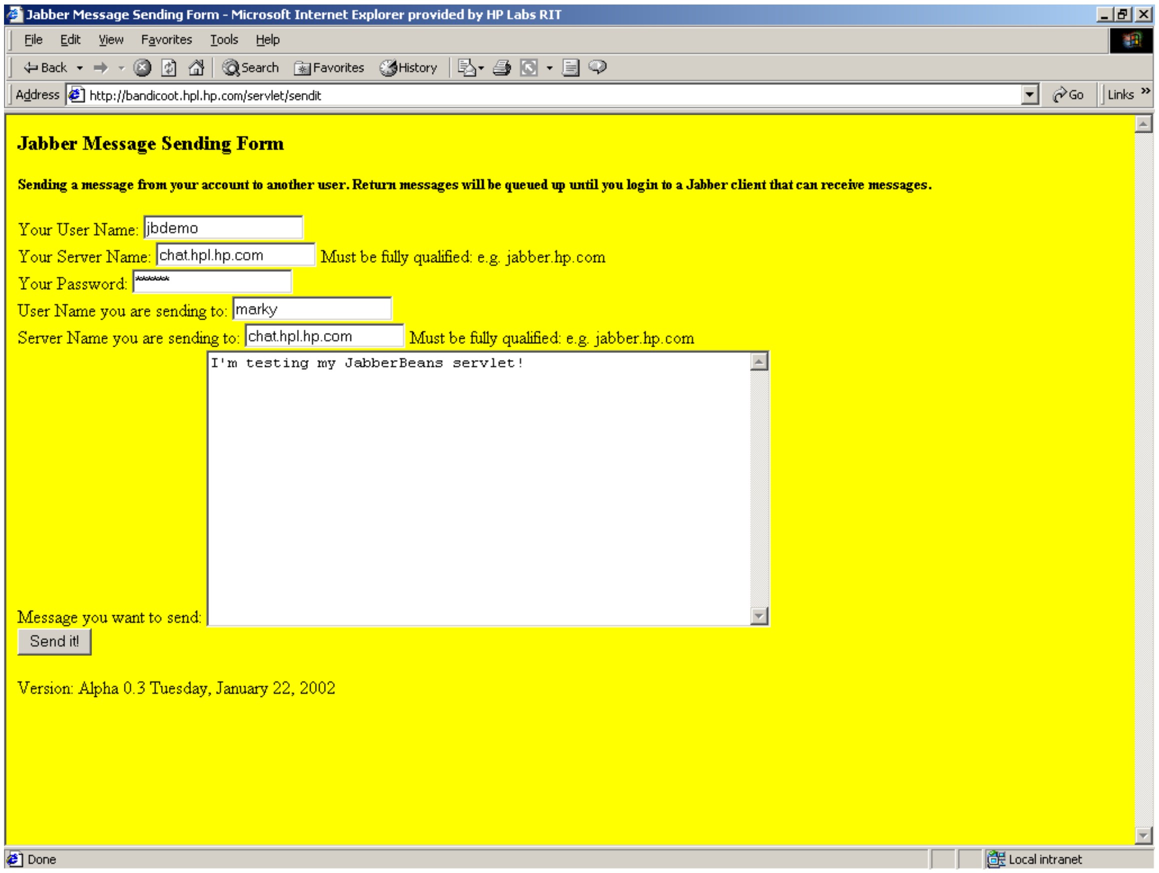
4. Restart Tomcat 4.0.

Write code to test the integration

1. Code the servlet with the JabberBeans API and compile to the .class file.
2. Copy the .class file to publish servlet.

The Result

There was success in integrating an example JabberBeans code with an example servlet code, with some minor edits. This code is named "sendit.java" and a screen capture of it running is shown in Figure 1.


Jabber Message Sending Form - Microsoft Internet Explorer provided by HP Labs RIT
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://bandicoot.hpl.hp.com/servlet/sendit
Go Links
Done
Local intranet

Jabber Message Sending Form

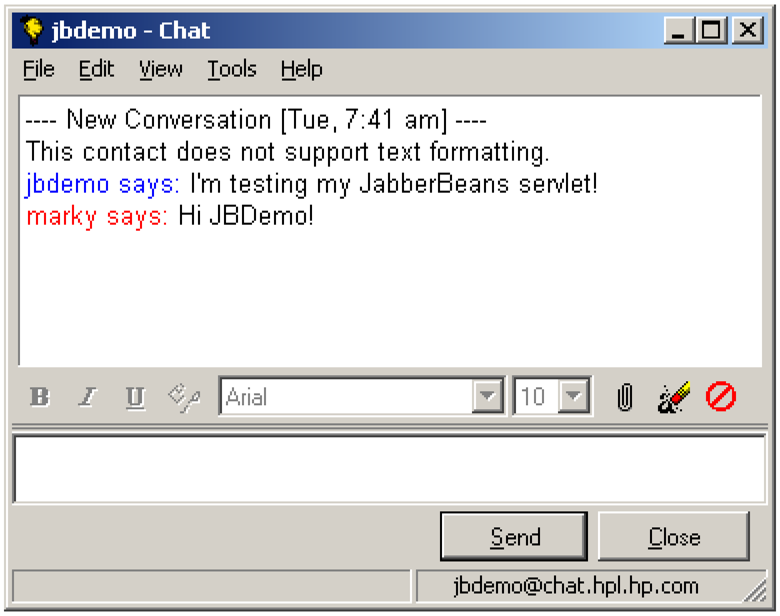
Sending a message from your account to another user. Return messages will be queued up until you login to a Jabber client that can receive messages.

Your User Name: jbdemo
Your Server Name: chat.hpl.hp.com Must be fully qualified: e.g. jabber.hp.com
Your Password: ******
User Name you are sending to: marky
Server Name you are sending to: chat.hpl.hp.com Must be fully qualified: e.g. jabber.hp.com
Message you want to send: I'm testing my JabberBeans servlet!
Send it!

Version: Alpha 0.3 Tuesday, January 22, 2002

Figure 1
Result of running the sendit code

Note that I'm sending a message from the account called *jbdemo* on the server called *chat.hpl.hp.com* to the account *marky* on the server called *chat.hpl.hp.com*. One could send to an account on a different server as long as it is accessible, i.e. both servers behind the firewall, etc. Hitting the "Send it!" button will send the message from the *jbdemo* user to the *marky* user.



---- New Conversation [Tue, 7:41 am] ----
This contact does not support text formatting.
jbdemo says: I'm testing my JabberBeans servlet!
marky says: Hi JBDemo!

jbdemo@chat.hpl.hp.com

Figure 2
Result of hitting the Send it! button in the sendit servlet window

Figure 2 shows marky@chat.hpl.hp.com receives a message from jbdemo@chat.hpl.hp.com. Note that the window in Figure 2 shows a response. That was just for illustration since this code used to test JabberBeans only sends messages but does not receive them. However that response will be received upon logging into a client capable of receiving messages since the Jabber system queues messages until that is true.

Key features of the *sendit.java* code

To understand Jabber programming better it is best to go to the Jabber Developer's Cheat Sheet [22] and to read the book by DJ Adams called Programming Jabber [23]. Here we will only highlight code fragments and make a reference to the complete *sendit.java* code. See Appendix B for the complete source code listing of *sendit.java*.

In *sendit.java* we separated the imports so that we can clearly see which are for servlet technology and which are for JabberBeans:

```
// Servlet imports
import java.io.*;
import java.text.*;
import java.util.*;
import javax.servlet.*;
import javax.servlet.http.*;

// JabberBeans imports
import org.jabber.jabberbeans.*;
import org.jabber.jabberbeans.util.JID;
import org.jabber.jabberbeans.Extension.*;
import java.net.InetAddress;
```

Also, there is a need to convert the command line parameters in the JabberBeans example code to be captured by a web form that the *getParameter* method does for us:

```
            // Parameters we will send to the sendJabber() method.
            String accountName = request.getParameter("accountname");
            String serverName = request.getParameter("servername");
            String passWord = request.getParameter("password");
            String toAccount = request.getParameter("toaccount");
            String toServer = request.getParameter("toserver");
            String messageBody = request.getParameter("messagebody");
```

Here is the beginning code for the form including the first parameter *accountname*:

```
            out.print("<form action=\"");
            out.print("sendit\" ");
            out.println("method=POST>");
            out.println("Your User Name:");
            out.println("<input type=text size=20 name=accountname>");
            out.println("<br>");
```

The parameters are then passed to the JabberBeans method:

```
            sendJabber(accountName, serverName, passWord, toAccount,
      toServer, messageBody);
```

The JabberBeans method *sendJabber* is pretty much like the example code found in the JabberBeans download called *jtest.java*. The first thing we want to do is to establish a Jabber session with the server and that is done with a JabberBeans *ConnectionBean* object:

```
            cb.connect(addr=InetAddress.getByName(serverName));
```

Next, we construct the XML packet to send the authentication data, using a JabberBeans *InfoQueryBuilder* object.

```
            // Construct a InfoQuery packet for the auth data
            InfoQueryBuilder iqb=new InfoQueryBuilder();
            InfoQuery iq;

            // and the auth data builder
            IQAuthExtensionBuilder iqAuthb=new
IQAuthExtensionBuilder();

            // Configure to set.
            iqb.setType("set");

            // Set data for the login to a Jabber server.
            iqAuthb.setUsername(accountName);
            iqAuthb.setPassword(passWord);
            iqAuthb.setResource("sendit");
            …
            iqb.addExtension(iqAuthb.build());
```

Next, we send the XML packet that authenticates the sending user, using the JabberBeans *ConnectionBean* send method.

```
            cb.send(iq);
```

Next, we build the message body using a JabberBeans *MessageBuilder* object.

```
            MessageBuilder mb=new MessageBuilder();
            Message msg;

            // Who and where to send the message to.
            mb.setToAddress(new JID(toAccount, toServer, null));

            mb.setBody(messageBody);
            try
            {
                  msg=(Message)mb.build();
            }
```

Finally, we send the message using the JabberBeans *ConnectionBean send* method.

```
            cb.send(msg);
```

Note in all this we did not write a single line of XML code! This is made possible by the JabberBeans API.

<u>Where to get *sendit.java*?</u>

Corporate Source, an open source (internal to HP) service of the HP Lab's Global Library, can be accessed at <u>http://source.hpl.hp.com</u>. Once you login there you can then click the "Browse" link and look for the software named "SENDIT."

Conclusion

The approach mentioned above demonstrates how to integrate JabberBeans with servlet technology. This of course was not a comprehensive coverage of all that can be done, but in general we can say that Jabber IM client features can be integrated into a servlet, in particular a Tomcat 4.0 implementation of servlet technology. The *sendit* servlet application is not a production piece of work, but a springboard to other efforts and hopefully other ideas. Note also that JabberBeans, and thus Java is not the only language in which to develop Jabber client applications. There are libraries in Perl, Python, C, Visual Basic and possibly more. Jabber is interesting now only from the standpoint of human-to-human but between human-to-machine as well as machine-to-machine XML messaging.

Acknowledgements

I would like to thank Jamie Dinkelacker for the guidance he gave to help make this paper possible! Thanks to Kevin Smathers for his pointers regarding Jabber. Final thanks to all those at HP's ISE (Infrastructure Strategic Engineering) and especially Ragavan Srinivasan whose expertise and enthusiasm with Jabber is contagious!

Appendix A: Setting up the Windows environment in a batch file

Setting the *classpath* on the Windows platform has caused trouble for some. Here are my *classpath* and Tomcat 4.0 settings in a batch file. Run this in a command prompt window when programming JabberBeans/Tomcat applications.

```
   set CATALINA_HOME=C:\jakarta-tomcat-4.0.1
   set
   CLASSPATH=.;C:\data\java\JabberServletDevel;%CATALINA_HOME%\common\l
   ib\servlet.jar;C:\JabberBeans-0.9.0-pre4\jabberbeans.jar
```

Appendix B: Sendit.java source

```
/**
 * sendit.java - A JabberBeans servlet to send a one way Jabber message
to another user.
 *
 */

// Servlet imports
import java.io.*;
import java.text.*;
import java.util.*;
import javax.servlet.*;
import javax.servlet.http.*;

// JabberBeans imports
import org.jabber.jabberbeans.*;
import org.jabber.jabberbeans.util.JID;
import org.jabber.jabberbeans.Extension.*;
import java.net.InetAddress;

/**
 * Sendit.java is a JabberBeans servlet to send a one way Jabber
message to another user.<p>
 * Note: <ol><li>This code is just a demonstration of the JabberBeans
API and is not production work.</li>
 * <li>It was tested only on Tomcat 4.0.</li>
 * <li>It is a derivative work from the example code
RequestParamExample.java written by
 * <a href="mailto:duncan@eng.sun.com">James Duncan Davidson</a> and
the example code jtest.java from the JabberBeans
 * download.<a href="mailto:dwaite@jabber.com">David Waite</a> is the
author of the JabberBeans API.</li></ol><p>
 * <u>Helpful links include:</u><br>
 * <a href="http://javaboutique.internet.com/tutorials/Tomcat/">Setting
up a Tomcat 4.0 server.</a><br>
 * <a href="http://www.jabber.org/">Learn about Jabber.</a><br>
 * <a href="http://jabberbeans.org/">Learn about the JabberBeans
API.</a><br>
 * <a
href="http://homepage.mac.com/jens/Jabber/JabberClientCheatSheet.html">
Jabber Client Developer's cheat sheet</a><br>
 * @author <a href="mailto:marky@hpl.hp.com">Mark Yoshikawa</a>
 * Copyright &copy;2002 <a href="http://www.hp.com">Hewlett Packard
Company</a><br>
 * @version alpha 0.3 for submission to Corporate Source
 */

public class sendit extends HttpServlet {
      public void doGet(HttpServletRequest request,
                        HttpServletResponse response)
            throws IOException, ServletException{

            response.setContentType("text/html");

            PrintWriter out = response.getWriter();
            out.println("<html>");
            out.println("<body>");
            out.println("<head>");

            // Setup the page
            String title = "Jabber Message Sending Form";
            out.println("<title>" + title + "</title>");
            out.println("</head>");
            out.println("<body bgcolor=\"yellow\">");

            out.println("<h3>" + title + "</h3>");
            out.println("<h5>Sending a message from your account to
another user. Return messages will be queued up until you login to a
Jabber client that can receive messages.</h5>");

            // Parameters we will send to the sendJabber() method.
            String accountName = request.getParameter("accountname");
            String serverName = request.getParameter("servername");
            String passWord = request.getParameter("password");
            String toAccount = request.getParameter("toaccount");
            String toServer = request.getParameter("toserver");
            String messageBody = request.getParameter("messagebody");

            // Making the assumption that a null accountName indicates
we are painting the page or data entry is in error. Otherwise call the
sendJabber method.
            if (accountName != null) {
                  sendJabber(accountName, serverName, passWord,
toAccount, toServer, messageBody);
            }

            out.println("<P>");
            out.print("<form action=\"");
            out.print("sendit\" ");
            out.println("method=POST>");
            out.println("Your User Name:");
            out.println("<input type=text size=20 name=accountname>");
            out.println("<br>");
            out.println("Your Server Name:");
            out.println("<input type=text size=20 name=servername>");
            out.println("Must be fully qualified: e.g. 
chat.hpl.hp.com");
            out.println("<br>");
            out.println("Your Password:");
            out.println("<input type=password size=20 name=password>");
            out.println("<br>");
            out.println("User Name you are sending to:");
            out.println("<input type=text size=20 name=toaccount>");
            out.println("<br>");
            out.println("Server Name you are sending to:");
            out.println("<input type=text size=20 name=toserver>");
            out.println("Must be fully qualified: e.g. 
chat.hpl.hp.com");
            out.println("<br>");
            out.println("Message you want to send:");
            out.println("<textarea name=messagebody rows=15 cols=60
></textarea>");
            out.println("<br>");
            out.println("<input type=submit value=\"Send it!\">");
            out.println("</form>");

            // Print version info on page.
            out.println("<p><p><p><p><p>");
            out.println("Version: Alpha 0.3 Tuesday, January 22, 
2002");
            out.println("</body>");
            out.println("</html>");
      }

      public void doPost(HttpServletRequest request,
                         HttpServletResponse response)
            throws IOException, ServletException
      {
            doGet(request, response);
      }

      public void sendJabber(String accountName, String serverName, 
String passWord, String toAccount, String toServer, String 
messageBody){
            // Setup a connection bean. (This is basically the 
JabberBean example with some edits!)
            ConnectionBean cb=new ConnectionBean();
            InetAddress addr;

            //Try to start a Jabber session with the server.
            try
            {
                  // Try the connection now
                  cb.connect(addr=InetAddress.getByName(serverName));
            }

            catch (java.net.UnknownHostException e)
            {
                  // If the attempt to contact via the getByName method
fails.
                  java.lang.System.out.println("DNS error finding your
server:");
                  java.lang.System.out.println(e.toString());
                  return ;
            }
            catch (java.io.IOException e)
            {
                  // if the JabberBeans method, connect, fails
                  java.lang.System.out.println("IO error while
attempting to connect to server:");
                  java.lang.System.out.println(e.toString());
                  return ;
            }

            // Construct a InfoQuery packet for the auth data
            InfoQueryBuilder iqb=new InfoQueryBuilder();
            InfoQuery iq;

            // and the auth data builder
            IQAuthExtensionBuilder iqAuthb=new
IQAuthExtensionBuilder();

            // Configure to set.
            iqb.setType("set");

            // Set data for the login to a Jabber server.
            iqAuthb.setUsername(accountName);
            iqAuthb.setPassword(passWord);
            iqAuthb.setResource("sendit");

            // Build the Auth data.
            try
            {
                  iqb.addExtension(iqAuthb.build());
            }
            catch (InstantiationException e)
            {
                  // If the build failed...
                  java.lang.System.out.println("Fatal Error on Auth
object build:");
                  java.lang.System.out.println(e.toString());
                  return;
            }

            // Build the iq packet.
            try
            {
                  // Build the full InfoQuery packet
                  iq=(InfoQuery)iqb.build();
            }
            catch (InstantiationException e)
            {
                  // If the build failed...
                  java.lang.System.out.println("Fatal Error on IQ
object build:");
                  java.lang.System.out.println(e.toString());
                  return ;
            }

            // Send the InfoQuery packet to the server so we can login.
            cb.send(iq);

            MessageBuilder mb=new MessageBuilder();
            Message msg;

            // Who and where to send the message to.
            mb.setToAddress(new JID(toAccount, toServer, null));

            mb.setBody(messageBody);
            try
            {
                  msg=(Message)mb.build();
            }
            catch (InstantiationException e)
            {
                  // If the build failed...
                  System.out.println("error creating message packet:");
                  System.out.println(e.toString());
                  return;
            }

            //Send the message!
            cb.send(msg);
      }
}
```

References

1. See http://www.jabberbeans.org - JabberBeans by David Waite.
2. See http://www.jabber.org - Jabber Software Foundation, main Jabber project page.
3. http://www.jabber.org/protocol/ - Standard and draft protocols with links to the draft submitted to IETF.
4. See http://jabber.hp.com/ - For use by North American Operations Delivery staff.
5. See http://www.jabber.com/ - Jabber, Inc.
6. See http://strategy.corp.hp.com/research/jabber.ppt - Secure real-time XML messaging using Jabber
7. See http://strategy.corp.hp.com/research/secure_xml.doc - Jabber Protocol Awareness report from HP-IT's Infrastructure Strategic Engineering, by Jim Harritt and Ragavan Srinivasan.
8. http://mailman.jabber.org/pipermail/jdev/2001-November/009192.html - Kevin Smathers contributes to the Jabber smtp gateway.
9. See http://source.hpl.hp.com - HP's Corporate Source web site.
10. See http://sharon.cselt.it/projects/jade/ - Java Agent DEvelopment Framework.
11. http://www.gnu.org/copyleft/gpl.html - GNU General Public License web page.
12. http://www.jabber.org/josl.html - Jabber Open Source License (JOSL)

13. http://sash.alphaworks.ibm.com/download/sashjab/ - IBM's Jabber client written in Sash.
14. http://www.xmlcooktop.com/ - XML Cooktop - free development environment for DTD, XML, and XSLT documents.
15. See http://www.jabbercentral.org/clients/ - List of Jabber clients.
16. http://jabberd.jabberstudio.org/ - Open Source Jabber server site.
17. See http://intelectronica.net/emacs-jabber/ - Emacs Jabber client integration.
18. See http://www.pipetree.com/jabber/jrpc.html - Formalization of Jabber-RPC in attempt to coordinate many efforts.
19. http://hades.jabber.org/ietf/draft-miller-jabber-00.html - Submission to Internet Engineering Task Force (IETF).
20. See http://java.sun.com/j2se/1.3/ - Here is the latest 1.3.1 version here. Sun has released the 1.4 at the time of this writing.
21. See http://www.apache.org/ - Open Source software projects.
22. See http://homepage.mac.com/jens/Jabber/JabberClientCheatSheet.html - Jabber Client Developer's Cheet Sheet.
23. DJ Adams, Programming Jabber, O'Reilly and Associates, 2002. ISBN: 0-596-00202-5.

Rev: 5/13/2002 10:53 AM