JavaMail | Add to Wunderlist

From Wikipedia, the free encyclopedia

JavaMail is a Java API used to send and receive email via SMTP, POP3 and IMAP. JavaMail is built into the Java EE platform, but also provides an optional package for use in Java SE.

The current version is 1.5.1, released on November 13, 2013. Another open source JavaMail implementation exists - GNU JavaMail - while supporting only version 1.3 of JavaMail specification, it provides the only free NNTP backend, which makes possible to use this technology to read and send news group articles.

Contents

Licensing

As of April 3, 2013,[1] JavaMail is now hosted as an open source project on Java.net.

Most of the JavaMail source code is licensed under the following licences:[2]

- CDDL-1.1
- GPL-2.0 with Classpath Exception license
- The source code for the demo programs is licensed under the BSD license

Examples

```
import java.util.*;
import javax.mail.*;
import javax.mail.internet.*;
import javax.activation.*;

// Send a simple, single part, text/plain e-mail
public class TestEmail {

    public static void main(String[] args) {

        // SUBSTITUTE YOUR EMAIL ADDRESSES HERE!
        String to = "sendToMailAddress";
        String from = "sendFromMailAddress";
        // SUBSTITUTE YOUR ISP'S MAIL SERVER HERE!
        String host = "smtp.yourisp.invalid";
```

```
        // Create properties, get Session
        Properties props = new Properties();

        // If using static Transport.send(),
        // need to specify which host to send it to
        props.put("mail.smtp.host", host);
        // To see what is going on behind the scene
        props.put("mail.debug", "true");
        Session session = Session.getInstance(props);

        try {
            // Instantiate a message
            Message msg = new MimeMessage(session);

            //Set message attributes
            msg.setFrom(new InternetAddress(from));
            InternetAddress[] address = {new InternetAddress(to)};
            msg.setRecipients(Message.RecipientType.TO, address);
            msg.setSubject("Test E-Mail through Java");
            msg.setSentDate(new Date());

            // Set message content
            msg.setText("This is a test of sending a " +
                        "plain text e-mail through Java.\n" +
                        "Here is line 2.");

            //Send the message
            Transport.send(msg);
        }
        catch (MessagingException mex) {
            // Prints all nested (chained) exceptions as well
            mex.printStackTrace();
        }
    }
}//End of class
```

Sample Code to Send Multipart E-Mail, HTML E-Mail and File Attachments

```
import java.util.*;
import java.io.*;
import javax.mail.*;
import javax.mail.internet.*;
import javax.activation.*;

public class SendMailUsage {

    public static void main(String[] args) {

        // SUBSTITUTE YOUR EMAIL ADDRESSES HERE!!!
        String to = "sendToMailAddress";
        String from = "sendFromMailAddress";
        // SUBSTITUTE YOUR ISP'S MAIL SERVER HERE!!!
        String host = "smtpserver.yourisp.invalid";

        // Create properties for the Session
        Properties props = new Properties();

        // If using static Transport.send(),
        // need to specify the mail server here
        props.put("mail.smtp.host", host);
        // To see what is going on behind the scene
        props.put("mail.debug", "true");

        // Get a session
        Session session = Session.getInstance(props);

        try {
```

```
            // Get a Transport object to send e-mail
            Transport bus = session.getTransport("smtp");

            // Connect only once here
            // Transport.send() disconnects after each send
            // Usually, no username and password is required for SMTP
            bus.connect();
            //bus.connect("smtpserver.yourisp.net", "username", "password");

            // Instantiate a message
            Message msg = new MimeMessage(session);

            // Set message attributes
            msg.setFrom(new InternetAddress(from));
            InternetAddress[] address = {new InternetAddress(to)};
            msg.setRecipients(Message.RecipientType.TO, address);
            // Parse a comma-separated list of email addresses. Be strict.
            msg.setRecipients(Message.RecipientType.CC,
                              InternetAddress.parse(to, true));
            // Parse comma/space-separated list. Cut some slack.
            msg.setRecipients(Message.RecipientType.BCC,
                              InternetAddress.parse(to, false));

            msg.setSubject("Test E-Mail through Java");
            msg.setSentDate(new Date());

            // Set message content and send
            setTextContent(msg);
            msg.saveChanges();
            bus.sendMessage(msg, address);

            setMultipartContent(msg);
            msg.saveChanges();
            bus.sendMessage(msg, address);

            setFileAsAttachment(msg, "C:/WINDOWS/CLOUD.GIF");
            msg.saveChanges();
            bus.sendMessage(msg, address);

            setHTMLContent(msg);
            msg.saveChanges();
            bus.sendMessage(msg, address);

            bus.close();

        }
        catch (MessagingException mex) {
            // Prints all nested (chained) exceptions as well
            mex.printStackTrace();
            // How to access nested exceptions
            while (mex.getNextException() != null) {
                // Get next exception in chain
                Exception ex = mex.getNextException();
                ex.printStackTrace();
                if (!(ex instanceof MessagingException)) break;
                else mex = (MessagingException)ex;
            }
        }
    }

    // A simple, single-part text/plain e-mail.
    public static void setTextContent(Message msg) throws MessagingException {
            // Set message content
            String mytxt = "This is a test of sending a " +
                           "plain text e-mail through Java.\n" +
                           "Here is line 2.";
            msg.setText(mytxt);

            // Alternate form
            msg.setContent(mytxt, "text/plain");

    }

    // A simple multipart/mixed e-mail. Both body parts are text/plain.
```

```
    public static void setMultipartContent(Message msg) throws MessagingException {
        // Create and fill first part
        MimeBodyPart p1 = new MimeBodyPart();
        p1.setText("This is part one of a test multipart e-mail.");

        // Create and fill second part
        MimeBodyPart p2 = new MimeBodyPart();
        // Here is how to set a charset on textual content
        p2.setText("This is the second part", "us-ascii");

        // Create the Multipart.  Add BodyParts to it.
        Multipart mp = new MimeMultipart();
        mp.addBodyPart(p1);
        mp.addBodyPart(p2);

        // Set Multipart as the message's content
        msg.setContent(mp);
    }

    // Set a file as an attachment.  Uses JAF FileDataSource.
    public static void setFileAsAttachment(Message msg, String filename)
                throws MessagingException {

        // Create and fill first part
        MimeBodyPart p1 = new MimeBodyPart();
        p1.setText("This is part one of a test multipart e-mail." +
                   "The second part is file as an attachment");

        // Create second part
        MimeBodyPart p2 = new MimeBodyPart();

        // Put a file in the second part
        FileDataSource fds = new FileDataSource(filename);
        p2.setDataHandler(new DataHandler(fds));
        p2.setFileName(fds.getName());

        // Create the Multipart.  Add BodyParts to it.
        Multipart mp = new MimeMultipart();
        mp.addBodyPart(p1);
        mp.addBodyPart(p2);

        // Set Multipart as the message's content
        msg.setContent(mp);
    }

    // Set a single part html content.
    // Sending data of any type is similar.
    public static void setHTMLContent(Message msg) throws MessagingException {

        String html = "<html><head><title>" +
                      msg.getSubject() +
                      "</title></head><body><h1>" +
                      msg.getSubject() +
                      "</h1><p>This is a test of sending an HTML e-mail" +
                      " through Java.</body></html>";

        // HTMLDataSource is a static nested class
        msg.setDataHandler(new DataHandler(new HTMLDataSource(html)));
    }

    /*
     * Static nested class to act as a JAF datasource to send HTML e-mail content
     */
    static class HTMLDataSource implements DataSource {
        private String html;

        public HTMLDataSource(String htmlString) {
            html = htmlString;
        }

        // Return html string in an InputStream.
        // A new stream must be returned each time.
        public InputStream getInputStream() throws IOException {
            if (html == null) throw new IOException("Null HTML");
```

```
            return new ByteArrayInputStream(html.getBytes());
        }

        public OutputStream getOutputStream() throws IOException {
            throw new IOException("This DataHandler cannot write HTML");
        }

        public String getContentType() {
            return "text/html";
        }

        public String getName() {
            return "JAF text/html dataSource to send e-mail only";
        }
    }

} //End of class
```

References

1. ^ "JavaMail" (http://www.oracle.com/technetwork/java/javamail/index.html). Retrieved 9 Nov 2013.
2. ^ "JavaMail License" (https://java.net/projects/javamail/pages/License). Retrieved 22 Jan 2014.

External links

- JavaMail homepage at Oracle (http://www.oracle.com/technetwork/java/javamail/index.html)
- JavaMail project site at Java.net (https://java.net/projects/javamail/pages/Home)
- GNU JavaMail (http://www.gnu.org/software/classpathx/javamail/javamail.html)