MIME Is a Terrible Thing to Waste: A Programmer's Guide

Erik Werfel

Senior Software Engineer
GalaxE Solutions
http://www.galaxesolutions.com/

TS-4990

Reaching Out Via Email

Integrate Email with Java™ Technology-Based Web Applications

Agenda

Why Use Email

Using the JavaMail™ API

Sending Multipart/Alternative Messages

Including File Attachments

Demo: Generating an Email Newsletter

User Metrics

Demo: User Metrics

Q&A

Agenda

Why Use Email

Using the JavaMail™ API

Sending Multipart/Alternative Messages

Including File Attachments

Demo: Generating an Email Newsletter

User Metrics

Demo: User Metrics

Q&A

Why Use Email

- Essential for workflow
  - Asynchronous
  - Notify a user when a process is ready for the user’s involvement
  - Send documents as attachments for review
  - Notify the user when a review process is complete
- Simple notifications
  - Not really a security feature, but some information may be more appropriately sent via email
    - Your password has been reset
- Newsletters
  - Keep a group up-to-date with new content or information in your application

Agenda

Why Use Email

Using the JavaMail™ API

Sending Multipart/Alternative Messages

Including File Attachments

Demo: Generating an Email Newsletter

User Metrics

Demo: User Metrics

Q&A

Using the JavaMail API

- Configure an email Java Naming and Directory Interface™ (JNDI) name
  - How you do this will depend upon your application server
- Add a resource to your application
  - Do this in web.xml
- Create a JavaMail API session
- Send a message
  - Populate the content
  - Populate recipients
  - Send when ready

Mail Resource

```
<resource-ref>
    <res-ref-name>mail/MyAppMailSession</res-ref-name>
    <res-type>javax.mail.Session</res-type>
    <res-auth>CONTAINER</res-auth>
</resource-ref>
```

Creating the JavaMail API Session

```
javax.naming.InitialContext context = new
javax.naming.InitialContext();
@Resource (mappedName=
"java:comp/env/mail/MyAppMailSession")
javax.mail.Session session;
```

You can override the session properties set at the server level if necessary by creating a new session:

```
Properties props = new Properties();
props.put("mail.transport.protocol", "smtp");
props.put("mail.smtp.host", "mailhost");
props.put("mail.from", “someAddress”);
Session session2 = session.getInstance(props);
```

Send a Simple Message

```
MimeMessage message = new MimeMessage(session);
message.setFrom(new InternetAddress(from));
message.addRecipient(Message.RecipientType.TO, new
InternetAddress(to)); message.setSubject(“Subject
here");
message.setText(“Text here");
Transport.send(message); // Send message
```

Agenda

Why Use Email

Using the JavaMail™ API

Sending Multipart/Alternative Messages

Including File Attachments

Demo: Generating an Email Newsletter

User Metrics

Demo: User Metrics

Q&A

Sending Multipart/ Alternative Messages

- Most mail clients can render HTML messages as rich text
- Some mail clients cannot render HTML, or some users may disable this capability
- SMTP allows a message to be sent with an HTML part along with a plain text alternative part
  - Users (generally) only see one part: the mail client determines which part to render
  - Both parts should therefore have all the information intended to be conveyed

Sending Alternative Messages

```
MimeBodyPart textPart = new MimeBodyPart();
textPart.setText( "Body of message here." ) ;
MimeBodyPart htmlPart = new MimeBodyPart();
htmlPart.setDataHandler(new DataHandler
( "<html><body>Body of message here</body></html>",
"text/html"));
MimeMultipart mp = new MimeMultipart( "alternative" );
mp.addBodyPart(htmlPart);
mp.addBodyPart(textPart);
MimeMessage message = new MimeMessage( session );
message.setContent(mp);
Transport.send( message ) ;
```

Agenda

Why Use Email

Using the JavaMail™ API

Sending Multipart/Alternative Messages

Including File Attachments

Demo: Generating an Email Newsletter

User Metrics

Demo: User Metrics

Q&A

Sending Attachments

- Messages are built as parts
- Include one or more file parts for attachments
- Plain and html alternative parts may be included as well; The mail client determines which parts to display

Sending Attachments

```
BodyPart messageBodyPart = new MimeBodyPart();
messageBodyPart.setText("Pardon Ideas");
Multipart multipart = new MimeMultipart();
multipart.addBodyPart(messageBodyPart);
messageBodyPart = new MimeBodyPart();
DataSource source = new FileDataSource(filename);
messageBodyPart.setDataHandler(new DataHandler(source));
messageBodyPart.setFileName(filename);
multipart.addBodyPart(messageBodyPart);
message.setContent(multipart);
Transport.send(message);
```

Embedding Images

- Same as sending an attachment, only
  - Indicate the part is related in constructor for MimeMultiPart; and
  - Set the content ID for attachment to a unique string that will be used as the source for the img tag
  - The same technique can be used for JavaScript™ technology files, CSS files

Embedding Images

```
BodyPart backgroundImageBodyPart = new MimeBodyPart();
DataSource backgroundUds =
new URLDataSource(
new URL("http://127.0.0.1" + url2ThisDb +
"/tocmenubg.jpg"));
backgroundImageBodyPart.setDataHandler(new
DataHandler(backgroundUds));
backgroundImageBodyPart.setHeader("Content-ID",
"<tocmenubg.jpg>");
multipart.addBodyPart(backgroundImageBodyPart );
messageObject.setContent(multipart);
Transport.send(messageObject);
```

Using Embedded Images

```
<!--Use a CID URL, which is a reference to the content ID
assigned to the image-->

<body class="newsletterbody"
background="cid:tocmenubg.jpg">
<div class="banner"><img src="cid:header.jpg"></div>
```

Agenda

Why Use Email
Using the JavaMail™ API
Sending Multipart/Alternative Messages
Including File Attachments
Demo: Generating an Email Newsletter
User Metrics
Demo: User Metrics
Q&A

DEMO

Generating an Email Newsletter

Agenda

Why Use Email

Using the JavaMail™ API

Sending Multipart/Alternative Messages

Including File Attachments

Demo: Generating an Email Newsletter

User Metrics

Demo: User Metrics

Q&A

Tracking Using Images

- Include an inline image in your email that your application builds
- Include a query string parameter that identifies the recipient
    - <imgsrc=http://myserver.company.com/app/servlet?id=123>

Tracking Using Images

- Create a servlet that processes the ID, then redirects to the transparent gif
- Not all mail systems will show images by default; Tracking using an image that is an important part of the content may increase the likelihood that users will download the image

Tracking Using Images

```
String id = request.getParameter("id");
// processUser is a method that updates the database
appropriately
processUser(id);
String location = "http://home.site.com/portal-
static/home/images/masthead.gif";
response.sendRedirect(location);
```

Include a Form in the Email

- Include an HTML form in the body of the message that posts back to your server
- This technique relies on the user to provide feedback, but is very convenient for the recipient
- This will not work for users who see the plain text email

Include a Form in the Email

```
<form action=http://hostname/App/TrackingServlet
method=post>
   <input type=”hidden” name=”id” value=”123”>
  Did this email meet your needs?:
<input type="radio">Yes
<input type="radio">No
<input type="submit" value="Submit">
</form>
```

Process Incoming Messages with James

- Users can reply to emails using an address that points to a James server
- Apache Enterprise Mail Server for the Java platform (James) is an SMPT server and email application engine
- Emails are processed using the mailet API
- The mailet can process the transaction on the basis of the recipient address or the body of the email

Process Incoming Messages with James

```
Collection recipients = mail.getRecipients();
MailAddress address = getFirstAddress(recipients);
String token = "mailresponseid";
if ( address.toLowerCase().startsWith(token)){
   String idPart = address.substring(token.length());
   StringTokenizer st = new StringTokenizer(idPart, "@");

     processId(st.nextToken());
}
```

DEMO

User Metrics

Summary

- Use JavaMail API to communicate with application users asynchronously
- JavaMail API can send MIME-encoded rich-text HTML messages
- HTML message can include embedded images, CSS files and JavaScript technology files
- The source URL for images that are not embedded can be used to track receipt
- Users can also reply to emails or use in-message forms for tracking

For More Information

- Fundamentals of the JavaMail API: http://java.sun.com/developer/onlineTraining/JavaMail/contents.html
- Apache James: http://james.apache.org/

Q&A

MIME Is a Terrible Thing to Waste: A Programmer's Guide

Erik Werfel

Senior Software Engineer
GalaxE Solutions
ewerfel@galaxesolutions.com
http://www.galaxesolutions.com/

TS-4990