Sendmail Action (2.1 legacy document)

Table of contents

Table of contents

Warning:
This document was copied as is from the Cocoon 2.1 documentation, but has not yet been fully reviewed or moved to its new home.

1. Description

The Sendmail action is an action which makes the capability of sending an email message available to the sitemap. This includes attaching binary files which come either from another Cocoon pipeline or from HTML file uploads. The Sendmail action make the primary methods of the Java Mail API available to the sitemap.

The Sendmail action is an alternative to the Sendmail logicsheet.

2. Usage

Before the Sendmail action can be used, it must be activated - either in Cocoon's main sitemap, or in a subsitemap which is used for your application. Include the following map:action element as child respectively grandchild of the map:components -> map:actions elements.

```
<map:components> <map:actions> <map:action name="sendmail"
logger="sitemap.action.sendmail" src="org.apache.cocoon.acting.Sendmail"/>
</map:actions> </map:components>
```

3. Example Code

This example uses a simple HTML form which provides entry fields for subject, cc, uploaded_file1 and the email body.

```
<form action="/cocoon/mail/send-a-mail" method="POST" enctype="multipart/form-data">
<input type="text" name="subject" size="56" /> <input type="text" name="cc" size="56" />
<input type="file" name="uploaded_file1" size="56" /> <textarea name="body" rows="5"
cols="72"> </textarea> </form>
```

Please keep in mind that it is important to use enctype="multipart/form-data" if you want to upload files which should be attached to an email message.

The posted HTTP request data is processed by this sitemap fragment.

```
<map:match pattern="mail/*"> <map:act type="sendmail"> <!-- To override defaults specified
in cocoon.xconf: <map:parameter name="smtp-host" value="localhost"/> <map:parameter
name="smtp-user" value="john"/> <map:parameter name="smtp-password" value="john"/>
--> <map:parameter name="from" value="cocoon@localhost"/> <map:parameter name="to"
value="mailinglist@somewhere.com"/> <map:parameter name="subject"
value="{request-param:subject}"/> <map:parameter name="body"
value="{request-param:body}"/> <map:parameter name="cc" value="{request-param:cc}"/>
<map:parameter name="bcc" value="censor@somewhere.com"/> <map:parameter
name="attachments" value="uploaded_file1 context://welcome.xml"/> <map:generate
src="mail/{status}.xml"/> <map:serialize type="xml"/> </map:act> </map:match>
```

The input modules are used to supply some of the input parameters for the Sendmail action. In this example, apart from the uploaded_file1 request parameter, a second file is attached to the email message, using the Cocoon protocol notation (the file welcome.xml from the Cocoon context directory). The result of sending the email message is passed back into the sitemap through the status parameter and is used to provide the user with a feedback. (The transformation is left as an exercise to the reader).

Please consider the security implications if you let a user specify an email address in an input form. A malicious user might abuse this to send SPAM emails. So, this example is probably only useful in an intranet application, where users can mostly be considered well behaved.

4. Input/Output Parameter Reference

The following is the list of parameters which can be passed from the sitemap into the Sendmail action.

Name:	Required?	Description:
smtp-host	no	The IP address or the name of the host, which should deliver the email message. Better known as the mail transfer agent or short MTA.
smtp-user	no	User name
smtp-password	no	Password
to	yes	Sets the destination/to address of the email message. This can be a list of comma separated email addresses.
cc	no	Sets the recipients of a carbon copy of this email. This can be a list of comma separated email addresses.
bcc	no	Sets the recipients of a black carbon copy of the email. This can be a list of comma separated email addresses.
from	yes	Sets the from address of the message.
subject	no	Sets the subject line of the message.
body	no	Sets the body text of the message. This parameter is ignored if the src parameter is specified.
src	no	A url specifying the source of the text body of the email. If src is specified, the body parameter is ignored.
srcMimeType	no	The optional mime type of the source of the text body of the email if you specified src.
charset	no	Sets the character set for encoding the message. This tag has only an effect if no attachments are send.
attachment	no	Sets the attachment for this

		email. This is a blank separated list. If an argument contains a ":", it is assumed that a Cocoon internal protocol is specified (ex: context://welcome.xml). This means that the attachment comes from a Cocoon pipeline (internally an org.apache.excalibur.source.Source object). If the argument does not contain a colon, the argument names a request parameter which is a file upload through a HTML form (internally an org.apache.cocoon.components.request.multipart.Fil object).

The following is the list of parameters which are passed from Sendmail action back into the sitemap.

Name:	Description:
status	This parameter can take three values: success, user-error and server-error. success means that the email message has been successfully delivered to the mail transfer host. user-error means that there was a problem with at least one of the specified email addresses (to, cc, bcc or from). server-error means that there was some problem delivering the message to the mail transfer host. In effect, this parameter can be used to inform the user about the outcome of sending the email message.
message	This parameter contain some explanatory text if the message couldn't be delivered. It is unset if the email message had been successfully sent.

5. Additional Hint

Cocoon provides the capability to automatically parse a file upload out of an incoming HTTP request. Depending on the setting of the parameter autosave-uploads (default is false) in Cocoon's web.xml file, the file upload is either stored in memory (false) for further processing or stuffed into Cocoons upload directory (true).

In theory, it should be equal whether the file upload data comes from memory of from file. In my particular setup (Linux, Tomcat 4.0.4, Mozilla 1.3 and JDK 1.4.1_02) I was unable to get the file uploading working with autosave-upload=false. Somehow the attached binary data was distorted (a GIF file does not appear to be a GIF any more).

With autosave-upload=true it worked flawlessly, even attaching multiple files.

1. Comments

add your comments