# SendingEmail

This is a quick guide to configuring Cocoon so that you can send email from within an XSP. It draws on the content found at Cocoon Center, updating it for Cocoon 2.0.4.

(The [CocoonCenter] link is broken, as is the whole www.cocooncenter.de site – a new link is needed)[pf]

You need a copy of JavaMail (`mail.jar`) and Java Activation Framework (`activation.jar`). Place them in the `WEB-INF/lib/` directory of your Cocoon installation (or into lib/local/ of the source distribution).

Although the sendmail logicsheet now ships with Cocoon, it is not available by default. To solve this, you need to edit your `cocoon.xconf`. Find the section for `<target-language name="java">`, and within it, add:

```
<builtin-logicsheet>
 <parameter name="prefix" value="sendmail"/>
 <parameter name="uri" value="http://apache.org/cocoon/sendmail/1.0"/>
 <parameter name="href"
value="resource://org/apache/cocoon/components/language/markup/xsp/java/sendmail.xsl"/>
</builtin-logicsheet>
```

Then, create an XSP file similar to the one in the Cocoon Centre article. Email should now work.

See also Cocoon documentation on Sendmail Action and Logicsheet. The cocoon sample block Mail doesn't work in 2.1.2 with Tomcat 4.1.27 on my installation, but is easy to fix using the documentation samples. --Mfonsen

## Encoding mail --Mfonsen

Encoding your email may be tricky, here's a quick-and-dirty solution I've used with Tomcat 4.1.27 and Cocoon 2.1.2. Replace ISO-8859-1 with your favourite encoding.

If you are using Sendmail action, your sitemap could look something like the example below. This just sends email, doesn't show anything and uses input modules to get request parameters. Set-encoding action is used to specify encoding of the request parameters, that are in this case in UTF-8.

```
<map:act type="set-encoding">
        <map:parameter name="form-encoding" value="UTF-8"/>
</map:act>
<map:act type="sendmail">
        <map:parameter name="smtphost" value="smtp.yourisp.com"/>
        <map:parameter name="from" value="{request-param:from}"/>
        <map:parameter name="to" value="{request-param:to}"/>
        <map:parameter name="subject" value="{request-param:subject}"/>
        <map:parameter name="body" value="{request-param:body}"/>
        <map:parameter name="charset" value="iso-8859-1"/>
</map:act>
```

After this you need to add -Dmail.mime.charset=ISO-8859-1 to your JVM options. This is needed because the charset parameter seems to be used only when encoding the body of the message. This property is specified on the JavaMail API Frontpage . You can for example use environment variable JAVA_OPTS that Tomcat Startup uses to execute JVM.
`set JAVA_OPTS=-Dmail.mime.charset=ISO-8859-1`

Everything else should now work fine, except that the subject is encoded in Cp1252 instead of specifed encoding. This is caused by the JavaMail library version packaged to Tomcat 4.1.27. In version 1.3.1 of the JavaMail, this bug (bugparade 4780255) has been fixed. The only solution I've found to replace this old implementation is to replace the actual JAR itself in common/lib directory with the new one.

Now encoding should work fine.