ADAMS

Advanced **D**ata mining **A**nd **M**achine learning **S**ystem

Module: adams-net

Peter Reutemann

December 24, 2014

Contents

List of Figures

Chapter 1

Email

Flows are ideal for being run as background jobs ("-headless" flag). For example, importing or processing data in batches can be done at night time. Of course, you want to be notified if something went wrong or some predictions are off. Adding the *SendEmail* sink to existing flows, allows for automatic sending of emails that were generated by the *CreateEmail* transformer: if everything is OK then send an email to the customer, otherwise send an email to sysadmin. The *CreateEmail* actor adds all incoming file names, e.g., the array output of a *DirectoryLister* source, as attachments before sending the email off to the specified recipients. You can also define a custom subject and body. Variables can be placed in subject and body alike, as they get expanded when creating the email.

In order to be able to send emails, ADAMS needs to know what SMTP server to connect to. The following example configures ADAMS to send emails using a Gmail account [1].

There are two ways of configuring Email:

- *globally* – using the preferences
- *per flow* – using the *SMTPConnection* standalone actor

1.1 Global settings

For configuring email globally, use the dialog available from the main menu (*Program → Preferences → Email*), as depicted in Figure 1.1. The placeholders *USER* and *PASSWORD* have to be replaced with the actual user credentials and *YOUR NAME* with the actual user's name, of course.

Alternatively, you can also simply create a properties file in the `$HOME/.adams` directory, called `Email.props`. The content for the Gmail setup would look like this:

```
Enabled=true
SmtpStartTls=true
SmtpPassword=
SmtpServer=smtp.gmail.com
```

---

[1]For more details, see the following Gmail help page:
`http://support.google.com/mail/bin/answer.py?hl=en&answer=13287`

```
SmtpTimeout=30000
SmtpPort=587
SmtpRequiresAuthentication=true
SmtpUser=USER
DefaultAddressFrom=YOUR NAME <USER@gmail.com>
DefaultSignature=
```
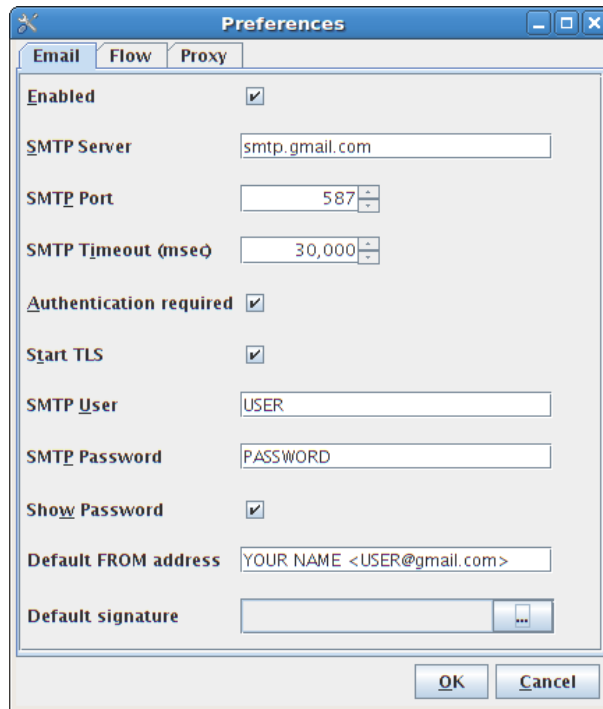

Figure 1.1: Email preferences

1.2   Actors

The following standalone actors are available:

- *SMTPConnection* – for configuring a SMTP server connection.

The following transformers are available:

- *CreateEmail* – creates an Email object. Interprets incoming files as at-
  tachments.
- *EmailFileReader* – reads email files with the specified email reader.

The following sinks are available:

- *EmailFileWriter* – writes Email objects to files using the specified email
  writer.
- *EmailViewer* – displays an Email object; can be used in conjunction with
  the *DisplayPanelManager* as well.

- *SendEmail* – sends Email objects to a SMTP server.

SMTPConnection

If you do not want to store the password with the flow - after all, the password is only obscured with base64 encoding - you can enable the *promptForPassword* option. This will prompt the user with a dialog for entering a password to be used for the connection.

1.3 Addressbook

ADAMS also offers a very simple addressbook for emails. Figure 1.2 shows a screenshot of it.
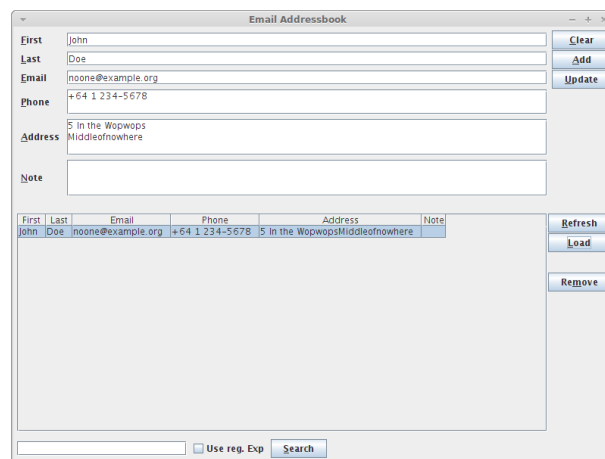

Figure 1.2: Email address book

1.4 Command-line

Using the *adams.core.net.SimpleMailer* command-line tool, you can read and send previously saved emails. The following command-line loads/sends all files in the directory */some/where* that end with *.props* using the *PropertiesEmail-FileReader* reader:

```
java -cp lib/* adams.core.net.SimpleMailer
  -env adams.env.Environment
  -reader adams.data.io.input.PropertiesEmailFileReader
  -watch-dir /some/where
  -regexp ".*.props"
  -D 1
```

After a file got sucessfully loaded and sent, it gets (by default) the *.sent* extension appended. If the process failed, the *.failed* extension is used (by default). Use the *-h* or *-help* option on the command-line to list all the available options.

Chapter 2

FTP

In addition to Email, ADAMS also supports FTP (file transfer protocol). Since FTP does not encrypt the commands or transferred data, it is recommended to not use it outside a company's network or for non-anonymous access (e.g., download from a public FTP server).

The following actors are available:

- *FTPConnection* – standalone actor that defines the server connection.
- *FTPDisconnect* – transformer that closes an existing connection, to avoid too many open connections (simply passes through any token it receives).
- *FTPLister* – lists the files and directories on a remote server.
- *FTPGet* – for obtaining a remote file via FTP.
- *FTPNoOp* – sends a NOOP command to the server; useful when trying to keep the connection alive.
- *FTPSend* – for sending a file via FTP to a remote host.

The *FTPConnection* standalone allows you to enter the password at runtime, if you do not want to store the password with the flow setup. At execution time, the user gets prompted with a dialog to enter a password to be used for the connection.

Chapter 3

SSH

ADAMS also comes with SSH support, thanks to the JSch library [1]. This allows for secure remote access. At the time of writing, RSA [2] and DSA [3] are supported for public key encryption schemes.

The following actors are available:

- *SSHConnection* – standalone actor that defines the server connection.
- *SSHExec* – for executing remote commands.
- *ScpFrom* – for copying a file *from* a remote host using secure copy.
- *ScpTo* – for copying a file *to* a remote host using secure copy.
- *SFTPGet* – for obtaining a remote file via secure FTP.
- *SFTPSend* – for sending a file via secure FTP to a remote host.

The *SSHConnection* standalone allows you to prompt the user as runtime to enter a password, in case you do not want to store the connection's password with the flow setup. The user has to enter the password through a dialog that pops up when the flow gets executed.

---

[1] http://www.jcraft.com/jsch/
[2] http://en.wikipedia.org/wiki/RSA_%28algorithm%29
[3] http://en.wikipedia.org/wiki/Digital_Signature_Algorithm

Chapter 4

Miscellaneous

Some other basic, but useful actors are the following:

- *Browser* – opens the system's default browser with the specified URL.
- *DownloadFile* – downloads a file via HTTP and saves it to disk.
- *DownloadContent* – downloads (textual) content via HTTP and forwards it.[1]
- *HTMLFileReader* – reads an HTML file, generating a DOM object.
- *HttpPost* – allows to send string content via HTTP POST.
- *MimeType* – determines the mime-type of files.[2]
- *URLSupplier* – outputs one or more URLs.
- *WebServer* – simple web server for serving static files.

The following conversions are available:

- *HTMLToDOM* – generates a DOM document from an HTML string.
- *HTMLToText* – extracts the text from an HTML string.
- *RenderHTMLAsText* – renders an HTML string as text.
- *SpreadSheetToHTML* – converts a spreadsheet into an HTML page (or just a table fragment).
- *StringToURL* – converts a string into a URL object.
- *URLToString* – converts a URL object into a string.

In case you need to debug a connection, `telnet` is always a useful tool. ADAMS comes with a very simple graphical version, depicted in Figure 4.1.

---

[1] adams-net-download_content.flow
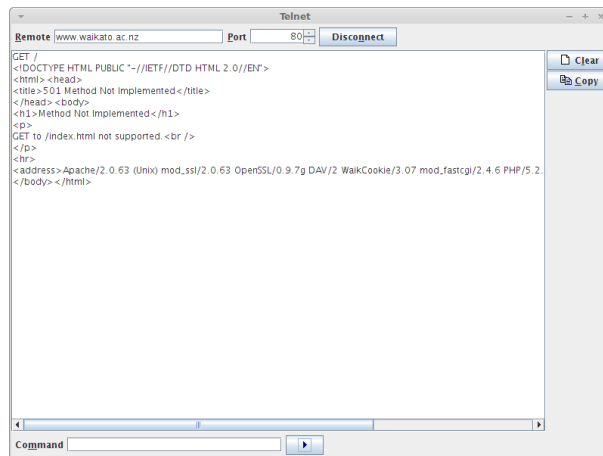[2] adams-net-mimetype.flow

Figure 4.1: Telnet dialog

Bibliography

[1] *ADAMS* – Advanced Data mining and Machine learning System
`https://adams.cms.waikato.ac.nz/`

[2] *Gmail* – Configuring other mail clients
`http://support.google.com/mail/bin/answer.py?hl=en&answer=`
`13287`