mail4rpg

Inhaltsverzeichnis

Abbildungsverzeichnis

Zusammenfassung

mail4rpg Dokumentation

1 Was ist mail4rpg?

mail4rpg ist ein einfach verwendbares Tool zum Versenden von E-Mail in RPG-Programmen. Einmal installiert, genügen wenige Zeilen Code zum Erstellen und Verschicken von E-Mails.

```
0001.00 H THREAD(*SERIALIZE)
0002.00 H DFTACTGRP(*NO)
0003.00 /COPY MAIL4RPG/SRC,MAIL4RPG_H
0004.00
0005.00 D m S O CLASS(*JAVA: 'mail4rpg.Mail4RPG')
0006.00 D rc S 10I 0
0007.00
0008.00 /FREE
0009.00
0010.00 m = init('192.168.1.1' : '/home/mail4rpg.log'); // Smtp-Server/Logfile
0011.00 setFrom (m: 'max.muster@invalid.tld');
0012.00 addTo (m: 'moritz.muster@invalid.tld');
0013.00 setSubject (m: 'mail4rpg');
0014.00 addBody (m: 'Heute ist das Wetter schön');
0014.00 addBodyHtml (m: '<h1>Überschrift</h1>');
0015.00 addAttachment (m: '/home/bild.jpg');
0016.00 setUser('username');
0017.00 setPass('passwort');
0018.00 rc = sendMail(m);
0019.00
0020.00 *INLR = *ON;
0021.00
0022.00 /END-FREE
```

Wie das Beispiel zeigt, ist *mail4rpg* einfach und unkompliziert in der Anwendung. Das Beispiel ist nahezu selbsterklärend. In Zeile 5 wird die Objektvariable `m` deklariert, die nach der Anweisung in Zeile 9 ein Objekt der Klasse `Mail4RPG` enthält. Angegeben wird der zu verwendende Smtp-Server und das Logfile, in das *mail4rpg* seine Aktionen protokollieren soll.

Im weiteren Programmverlauf wird die E-Mail durch Aufrufe von Subprocedures Schritt für Schritt aufgebaut und schließlich mit der Subprocedure `sendMail` an den Smtp-Server übergeben, der die E-Mail versendet.

Verlangt der Smtp-Server keine Autorisierung, lässt man die Aufrufe von `setUser` und `setPass` einfach weg.

2 Systemvoraussetzungen

Die Verwendung von *mail4rpg* setzt ein auf der AS/400 (iSeries/IBM i) installiertes IBM Developer Kit für Java (Lizenzprogramm 5761JV1) voraus. *mail4rpg* läuft mit Java-Version 1.5 oder höher ab V5R4M0 oder höher.

3 Lieferumfang

mail4rpg wird als ZIP-File `mail4rpg.zip` ausgeliefert. Das ZIP-Archiv enthält:

- `mail4rpg.jar`
- `mail.jar`
- `activation.jar`
- `mail4rpg.savf` (Save-File der Bibliothek `MAIL4RPG`)
- `mail4rpg.pdf` (Diese Dokumentation)

- `mail4rpg-license_de.pdf` (Lizenzvereinbarung, Deutsch)
- `mail4rpg-license_en.pdf` (Lizenzvereinbarung, Englisch)

4 Installation in drei Schritten

4.1 Schritt 1 - Entpacken von `mail4rpg.zip`

Die Datei `mail4rpg.zip` wird auf dem PC in das Verzeichnis `C:\mail4rpg` entpackt.

4.2 Schritt 2 - Installieren der Jar-Files

Die drei Jar-Files `mail4rpg.jar`, `mail.jar` und `activation.jar` werden auf die AS/400 ins Verzeichnis `/qibm/userdata/java400/ext` kopiert.

Da zum Schreiben in `/qibm/userdata/java400/ext` `QSECOFR`-Rechte erforderlich sind, sollte dieser Schritt nur von erfahrenen Systemadministratoren durchgeführt werden! Höggschde Konzentration und Dischdsiblien!

Zum Kopieren der Jar-Files wird die AS/400-Freigabe `QIBM` als Netzlaufwerk unterdem Benutzer `QSECOFR` verbunden. Der folgende Screenshot zeigt exemplarisch, was zu tun ist. Statt `as400` gibt man den Namen oder die IP-Adresse der eigenen AS/400 an.

Abbildung 1: Kopieren der Jar-Files

4.3 Schritt 3 - Einspielen der Bibliothek `MAIL4RPG`

Das Save-File `mail4rpg.savf` wird auf die AS/400 übertragen. Beispielhaft zeigt das der folgende Screenshot.

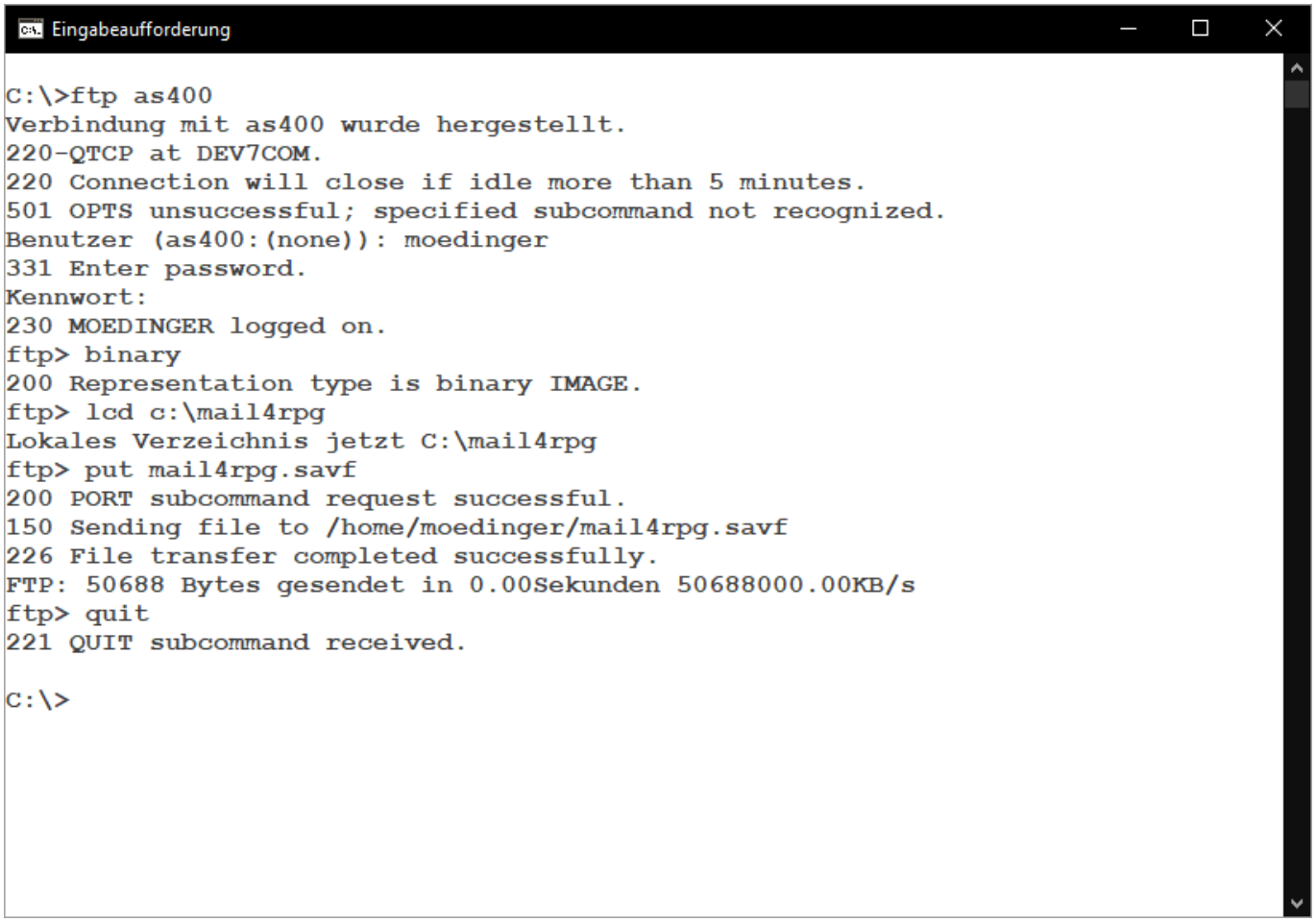

Abbildung 2: Übertragen des Save-Files `mail4rpg.savf`

Nun kann die Bibliothek `MAIL4RPG` restored werden:

```
      RSTLIB SAVLIB(MAIL4RPG) DEV(*SAVF) SAVF(QGPL/MAIL4RPG)
```

5 Verwendung von *mail4rpg*

Die folgenden Unterabschnitte beschreiben alle verfügbaren Subprocedures. Die Syntax ist immer in Form eines Beispiels erklärt. Die literalen Werte in den Beispielen können selbstverständlich durch Variablen ersetzt werden.

In den Beispielen bedeutet `m` immer eine Objektvariable, die in einer RPG D-Bestimmung so deklariert wurde:

```
     D  m               S               O   CLASS(*JAVA: 'mail4rpg.Mail4RPG')
```

5.1 init

Initialisiert *mail4rpg*. Anzugeben ist der zu verwendende SmtpServer, sowie der Filename des Logfiles.

Syntax:

```
init (m: '192.168.1.1' : '/home/mail4rpg.log');
```

Das Logfile wird im Unix-Format (mit LF als Zeilentrenner) geschrieben.

5.2 setSubject

Setzt den `Subject:`-Header der E-Mail.

Syntax:

```
setSubject (m : 'Betreff der E-Mail');
```

5.3 setFrom

Setzt den `From:`-Header der E-Mail.

Syntax:

```
setFrom (m : 'user@invalid.tld');
```

5.4 addTo

Setzt einen `To:`-Header in der E-Mail. Ess kann eine einzelne E-Mailadresse oder eine kommagetrennte Liste von E-Mailadressen angegeben werden. Diese Subprocedure kann mehrmals aufgerufen werden.

Syntax:

```
addTo (m : 'user@invalid.tld');
addTo (m : 'user1@invalid.tld, user2@invalid.tld');
```

5.5 addCc

Setzt einen `CC:`-Header in der E-Mail. Es kann eine einzelne E-Mailadresse oder eine kommagetrennte Liste von E-Mailadressen angegeben werden. Diese Subprocedure kann mehrmals aufgerufen werden.

Syntax:

```
addCc (m : 'user@invalid.tld');
addCc (m : 'user1@invalid.tld, user2@invalid.tld');
```

5.6 addBcc

Setzt einen `BCC:`-Header in der E-Mail. Es kann eine einzelne E-Mailadresse oder eine kommagetrennte Liste von E-Mailadressen angegeben werden. Diese Subprocedure kann mehrmals aufgerufen werden.

Syntax:

```
addBcc (m : 'user@invalid.tld');
addBcc (m : 'user1@invalid.tld, user2@invalid.tld');
```

5.7 setReplyTo

Setzt einen `Reply-To:`-Header in der E-Mail. Diese Subprocedure sollte nur einmal mit einer einzelnen E-Mailadresse aufgerufen werden.

Syntax:

```
setReplyTo (m : 'usertoreplyto@invalid.tld');
```

5.8 addBody

Fügt der E-Mail einen Bodypart hinzu. Diese Subprocedure kann mehrmals aufgerufen werden.

Syntax:

```
addBody (m : 'bodytext');
```

5.9 addBodyHtml

Fügt der E-Mail einen HTML-Bodypart hinzu. Diese Subprocedure kann mehrmals aufgerufen werden.

Syntax:

```
addBodyHtml (m : '<h1>Titel</h1>');
```

5.10 addBodyF

Fügt der E-Mail einen Bodypart hinzu. Der Inhalt wird aus einem File gelesen. Diese Subprocedure kann mehrmals aufgerufen werden.

Syntax:

```
addBodyF (m : '/home/bodytext.txt');
```

5.11 addBodyFHtml

Fügt der E-Mail einen Html-Bodypart hinzu. Der Inhalt wird aus einem File gelesen. Diese Subprocedure kann mehrmals aufgerufen werden.

Syntax:

```
addBodyFHtml (m : '/home/bodytext.html');
```

5.12 addAttachment

Fügt der E-Mail einen Anhang hinzu. Anzugeben ist der Filename des Anhangs. Diese Subprocedure kann mehrmals aufgerufen werden.

Syntax:

```
addBodyFHtml (m : '/home/bodytext.html');
```

5.13 sendMail

Versendet die E-Mail an den Smtp-Server, der in der init-Procedure angegeben wurde.

Syntax:

```
rc = sendMail (m);
```

`rc` ist eine Integer-Variable. `sendMail` liefert bei erfolgreicher Ausführung `0` zurück, im Fehlerfall `-1`. Im Fehlerfall enthält das Logfile (anzugeben in der init-Procedure) weitere Informationen über die Fehlerursache.

5.14 setUser

Verlangt der Smtp-Server eine Authentifizierung, dann setzt man mit dieser Subprocedure den Smpt-Benutzernamen.

Syntax:

```
setUser('john_doe');
```

5.15 setPass

Verlangt der Smtp-Server eine Authentifizierung, dann setzt man mit dieser Subprocedure das Smpt-Passwort.

Syntax:

```
setPass('my_password');
```

6 Fragen und Antworten

6.1 Ist *mail4rpg* Open Source?

Nein. *mail4rpg* wird Ihnen von der comSID GmbH & Co. KG *kostenlos* zur Verfügung gestellt, es ist aber nicht Open Source.

6.2 Mir fehlt ein wichtiges Feature! Was kann ich tun?

Kontaktieren Sie die comSID GmbH & Co. KG. Möglicherweise ist das Feature bereits in Entwicklung oder kann in einer späteren Version implementiert werden. Kontaktmöglichkeiten finden Sie auf https://www.comsid.de/kontakt. Oder mailen Sie an mail4rpg@comsid.de

A Anhang