Mail Engine

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME
1.0	2013-12		SF

Contents

1 Pengantar

Mail engine adalah program Java yang memungkinkan program RPG di AS/400 untuk mengirimkan email dengan mudah. Program Java di PC akan membaca data queue di AS/400. Jika ada data dalam data queue maka akan dikirimkan email berdasarkan data tersebut. Program RPG di AS/400 hanya perlu menuliskan data ke dalam data queue untuk mengirimkan email.

2 Perangkat Lunak

Dalam menuliskan program digunakan perangkat lunak berikut:

1. Java SDK versi 7
2. Maven versi 3.1.1
3. NetBeans 8

Silahkan melakukan instalasi perangkat lunak diatas sesuai dengan petunjuk dari situs mereka.

3 Mail Engine

Untuk melakukan test program maka penulis membuat library MAIL di AS/400.

```
===>CRTLIB MAIL
```

Sesudah itu dibuat data queue.

```
===>CRTDTAQ DTAQ(MAIL/MAILQ) MAXLEN(1000)
```

Jadikan library MAIL sebagai current library dan buat source file QSRC. Masuk ke dalam source file itu dengan WRKMBRPDM. Buatlah member PGSEND dengan tipe RPGLE.

```
===>CHGCURLIB MAIL
===>CRTSRCPF FILE(QSRC)
===>WRKMBRPDM FILE(MAIL/QSRC)
===>STRSEU SRCFILE(MAIL/QSRC) SRCMBR(PGSEND) TYPE(RPGLE)
```

Lalu masukkan code berikut:

```
     D QNAME           S             10A   INZ(’MAILQ’)
     D LIB             S             10A   INZ(’MAIL’)
     D QLEN            S              5  0 INZ(1000)
      *
     D MAILDS          DS
     D  FROM                        100A
     D  TO                          100A
     D  SUBJECT                     100A
     D  TEXT                        700A
      *
     C                   EVAL       FROM=’sfranklyn@gmail.com’
     C                   EVAL       TO=’sfranklyn@gmail.com’
     C                   EVAL       SUBJECT=’Send mail from RPG’
     C                   EVAL       TEXT=’It Works’
      *
     C                   CALL      ’QSNDDTAQ’
     C                   PARM                    QNAME
     C                   PARM                    LIB
     C                   PARM                    QLEN
     C                   PARM                    MAILDS
      *
     C                   EVAL      *INLR=*ON
     C                   RETURN
```

Refresh tampilan anda dengan F5 lalu compile member PGSEND dengan menggunakan option 14.

Saatnya membuat program java anda dengan menggunakan Maven. Jalankan perintah berikut di command line

```
$ mvn archetype:generate \
   -DarchetypeGroupId=org.apache.maven.archetypes \
   -DarchetypeArtifactId=maven-archetype-quickstart \
   -DarchetypeVersion=1.1
```

Isi nilai groupId dengan com.gamya, nilai artifactId dengan mailengine, nilai version dengan 1.0 dan nilai package dengan com.gamya.mailengine. Maven akan membuat file pom.xml, source file App.java dan AppTest.java. Hapuslah source file App.java dan AppTest.java. Sesuaikan isi file pom.xml anda hingga seperti dibawah ini:

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
                      http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>com.gamya</groupId>
    <artifactId>mailengine</artifactId>
    <version>1.0</version>
    <packaging>jar</packaging>

    <name>mailengine</name>
    <url>http://maven.apache.org</url>

    <properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <maven.compiler.source>1.7</maven.compiler.source>
        <maven.compiler.target>1.7</maven.compiler.target>
    </properties>

    <build>
        <plugins>
            <plugin>
                <groupId>org.codehaus.mojo</groupId>
                <artifactId>exec-maven-plugin</artifactId>
                <version>1.2.1</version>
                <configuration>
                    <mainClass>com.gamya.mailengine.MailEngine</mainClass>
                </configuration>
            </plugin>
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-jar-plugin</artifactId>
                <version>2.4</version>
                <configuration>
                    <archive>
                        <manifest>
                            <addClasspath>true</addClasspath>
                            <mainClass>com.gamya.mailengine.MailEngine</mainClass>
                        </manifest>
                    </archive>
                </configuration>
            </plugin>
        </plugins>
    </build>

    <dependencies>
        <dependency>
            <groupId>net.sf.jt400</groupId>
            <artifactId>jt400-full</artifactId>
            <version>4.7.0</version>
        </dependency>
        <dependency>
            <groupId>javax</groupId>
            <artifactId>javaee-api</artifactId>
            <version>7.0</version>
        </dependency>
    </dependencies>
</project>
```

Buat file MailEngine.java dengan isi seperti ini:

```
/*
 * Copyright 2013 Samuel Franklyn <sfranklyn@gmail.com>.
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *      http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.gamya.mailengine;

import com.ibm.as400.access.AS400;
import com.ibm.as400.access.AS400SecurityException;
import com.ibm.as400.access.AS400Text;
import com.ibm.as400.access.CharacterFieldDescription;
import com.ibm.as400.access.DataQueue;
import com.ibm.as400.access.DataQueueEntry;
import com.ibm.as400.access.ErrorCompletingRequestException;
import com.ibm.as400.access.IllegalObjectTypeException;
import com.ibm.as400.access.ObjectDoesNotExistException;
import com.ibm.as400.access.Record;
import com.ibm.as400.access.RecordFormat;
import java.io.IOException;
import java.io.InputStream;
import java.util.Properties;
import java.util.logging.FileHandler;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.mail.Message;
import javax.mail.MessagingException;
import javax.mail.PasswordAuthentication;
import javax.mail.Session;
import javax.mail.Transport;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;

public class MailEngine {
    private static final Logger logger = Logger.getLogger(MailEngine.class.getName());
    private final Properties configProperties;

    public static void main(String[] args) {
        MailEngine mailEngine = new MailEngine();
        mailEngine.readQueue();
    }

    public MailEngine() {
        configProperties = new Properties();
        InputStream inputStream = this.getClass().getClassLoader().
                getResourceAsStream("config.properties");
        try {
            configProperties.load(inputStream);
            FileHandler fileHandler = new FileHandler("log.xml");
            logger.addHandler(fileHandler);
            logger.setLevel(Level.ALL);
        } catch (IOException ex) {
            logger.log(Level.SEVERE, null, ex);
        }
    }

    public void sendMail(String from, String to, String subject, String text) {
        final String username = configProperties.getProperty("mail.username");
        final String password = configProperties.getProperty("mail.password");

        Session session = Session.getInstance(configProperties,
                new javax.mail.Authenticator() {
                    @Override
                    protected PasswordAuthentication getPasswordAuthentication() {
                        return new PasswordAuthentication(username, password);
                    }
                });

        try {
            Message message = new MimeMessage(session);
            message.setFrom(new InternetAddress(from));
            message.setRecipients(Message.RecipientType.TO,
                    InternetAddress.parse(to));
            message.setSubject(subject);
            message.setText(text);

            Transport.send(message);
        } catch (MessagingException ex) {
            logger.log(Level.SEVERE, null, ex);
        }
    }

    public void readQueue() {
        String systemName = configProperties.getProperty("as400.systemname");
        String userName = configProperties.getProperty("as400.username");
        String password = configProperties.getProperty("as400.password");
        String dataQueue = configProperties.getProperty("as400.dataqueue");

        AS400 as400 = new AS400(systemName, userName, password);
        DataQueue dq = new DataQueue(as400, dataQueue);

        CharacterFieldDescription fromFD
                = new CharacterFieldDescription(new AS400Text(100, as400),
                        "FROM");
        CharacterFieldDescription toFD
                = new CharacterFieldDescription(new AS400Text(100, as400),
                        "TO");
        CharacterFieldDescription subjectFD
                = new CharacterFieldDescription(new AS400Text(100, as400),
                        "SUBJECT");
        CharacterFieldDescription textFD
                = new CharacterFieldDescription(new AS400Text(700, as400),
                        "TEXT");

        RecordFormat recordFormat = new RecordFormat();
        recordFormat.addFieldDescription(fromFD);
        recordFormat.addFieldDescription(toFD);
        recordFormat.addFieldDescription(subjectFD);
        recordFormat.addFieldDescription(textFD);

        String from;
        String to;
        String subject;
        String text;
        Record record;
        try {
            logger.log(Level.INFO, "Read mail data queue");
            while (true) {
                DataQueueEntry DQData = dq.read(-1);
                record = recordFormat.getNewRecord(DQData.getData());
                from = record.getField("FROM").toString().trim();
                logger.log(Level.INFO, "From: {0}", from);
                to = record.getField("TO").toString().trim();
                logger.log(Level.INFO, "To: {0}", to);
                subject = record.getField("SUBJECT").toString().trim();
                logger.log(Level.INFO, "Subject: {0}", subject);
                text = record.getField("TEXT").toString().trim();
                logger.log(Level.INFO, "Text: {0}", text);
                sendMail(from, to, subject, text);
            }
        } catch (AS400SecurityException | ErrorCompletingRequestException |
                IOException | IllegalObjectTypeException |
                InterruptedException | ObjectDoesNotExistException ex) {
            logger.log(Level.SEVERE, null, ex);
        }
    }

}
```

Untuk bisa menjalankan program java ini anda harus menyesuaikan isi dari config.properties di direktori src/main/java/resources. Contoh isi dari config.properties

```
mail.username=user@gmail.com
mail.password=password
mail.smtp.auth=true
mail.smtp.starttls.enable=true
mail.smtp.host=smtp.gmail.com
mail.smtp.port=587
as400.systemname=pub1.de
as400.username=USER
as400.password=PASSWORD
as400.dataqueue=/QSYS.LIB/MAIL.LIB/MAILQ.DTAQ
```

Untuk membuat file jar untuk menjalankan program java gunakan perintah berikut di command line.

```
$ mvn clean package
```

Untuk menjalankan jar gunakan perintah:

```
$ mvn exec:java
```

Lalu jalankan program program PGSEND di AS/400 untuk mengirim surat.

```
===>CALL PGSEND
```