TP 6 : MIME, URL, Serveur Mail SMTP et HTTP

Documentation :

1. http://www.tutorialspoint.com/javamail_api/
2. http://www.jmdoudoux.fr/java/dej/chap-javamail.htm

1. MIME depuis JAVA

1.1. En lisant le document "Get the Mime Type from a File - Real's Java How-to.html", tester le type MIME d'un fichier de votre choix puis d'un fichier html en ligne.

2. Envoie d'un mail

2.1. Configuration pour l'envoi du mail avec le serveur smtp de gmail

2.1.1. Activation du compte Gmail pour autorisé les envois de mail par les applications (il faut se loger chez gmail avant, vous pouvez utiliser votre mail académique) : https://www.google.com/settings/security/lesssecureapps

2.1.2. Installation de l'API javax.mail.jar

a. Téléchargement https://javaee.github.io/javamail/#Download_JavaMail_Release

b. Sous Windows : inclure le jar dans le projet

c. Sous linux (on va pas le faire): en root

i. [root@iga ext]# update-alternatives --display java
(par exemple /usr/lib/jvm/jre-1.6.0-openjdk/)
Choisir le meilleur endroit (*)

ii. Puis placer le jar dans lib/ext de ce chemin

d. Désactiver l'anti-virus (plutôt le parefeu)

2.1.3. Conditions d'utilisation des serveurs mails de google

e. A lire https://support.google.com/a/answer/176600?hl=fr Et a commenter. Quelle est la solution que nous allons utiliser

2.1.4. Code a adapter avec votre compte

```
import java.util.Properties;
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 mail {
   public static void main(String[] args) {



      // Recipient's email ID needs to be mentioned.
    String to = "XXXXX";
```

```
        // Sender's email ID needs to be mentioned
              String from = "XXXX";
        final String username = "XXXX";//change accordingly
        final String password = "XXXX";//change accordingly

        // Assuming you are sending email through relay.jangosmtp.net
        String host = "XXXXX";




        Properties props = new Properties();
    // Nous supposons ici que le serveur smtp de l'entreprise demain
    //l'authentification et utilise un tunnel SSL (port 465) ou
    //TLS (port 587)
        props.put("mail.smtp.auth", "true");
        props.put("mail.smtp.starttls.enable", "true");
        props.put("mail.smtp.host", host);
        props.put("mail.smtp.port", "587");

        // Get the Session object.
        Session session = Session.getInstance(props,
        new javax.mail.Authenticator() {
           protected PasswordAuthentication getPasswordAuthentication() {
              return new PasswordAuthentication(username, password);
           }
        });

        try {
           // Create a default MimeMessage object.
           Message message = new MimeMessage(session);

           // Set From: header field of the header.
           message.setFrom(new InternetAddress(from));

           // Set To: header field of the header.
           message.setRecipients(Message.RecipientType.TO,
           InternetAddress.parse(to));

           // Set Subject: header field
           message.setSubject("Testing Subject");

           // Now set the actual message
           message.setText("Hello, this is sample for to check send "
              + "email using JavaMailAPI ");

           // Send message
           Transport.send(message);

           System.out.println("Sent message successfully....");

        } catch (MessagingException e) {
              throw new RuntimeException(e);
        }
     }
}
```

Remplir les champs et tester

2.2. Ajouter dans le Body du mail, un contenu Text et un contenu HTML

```
Message message = new MimeMessage(session);

Multipart multiPart = new MimeMultipart("alternative");

MimeBodyPart textPart = new MimeBodyPart();
textPart.setText(text, "utf-8");

MimeBodyPart htmlPart = new MimeBodyPart();
htmlPart.setContent(html, "text/html; charset=utf-8");

multiPart.addBodyPart(textPart); <-- first
multiPart.addBodyPart(htmlPart); <-- second
message.setContent(multiPart);
```

2.3. En cherchant sur google, ajouter une pièce jointe au mail.

3. URL en java

3.1. Création d'une URL

3.1.1. Ici nous étudions le concept de URL et nous développons un simple programme (application JAVA) permettant d'ouvrir un UR avec détection d'erreur (e.g http://www.polytech2go.fr/index.htm)

Commenter et Tester le code suivant :

```
try {
     URL monUrl = new URL("http://www.polytech2go.fr/index.htm");

} catch (MalformedURLException e) {
     // exception handler code here
}

//analyser un URL

System.out.println("protocol = " + aURL.getProtocol());
System.out.println("host = " + aURL.getHost());
System.out.println("filename = " + aURL.getFile());
System.out.println("port = " + aURL.getPort());
System.out.println("ref = " + aURL.getRef());

//lire directement le contenu d'un URL ouvert dans l'application
//(code HTML)

BufferedReader in = new BufferedReader(
                    new InputStreamReader(monUrl.openStream()));

String inputLine;

while ((inputLine = in.readLine()) != null)
     System.out.println(inputLine); // sortie standard

in.close();
```

3.2. Connexion avec un serveur PHP

Dans le cas d'exécution d'un programme script PHP il est important de pouvoir lui fournir les arguments et/ou les données et de lire la réponse généré par le script.

Tester et commenter le code suivant :

```
import java.net.*;
import java.util.*;
import java.io.*;

public class URLConnectionEx {

 public static void main (String args[]) throws IOException {
        if (args.length != 1)
           throw new RuntimeException ("Syntax: java URLConnectionEg
           <url>");

 String stringtosend = URLEncoder.encode(args[0]);

 URL url = new URL("http://www.polytech2go.fr/cgi-bin/montest.php");
 URLConnection c = url.openConnection();

 c.setDoOutput (true);

 PrintWriter out = new PrintWriter(c.getOutputStream());
 out.println("monstring=" + stringtosend); // envoi
 out.close();

 }
}
```

4. http

4.1. Que fait le code suivant

```
import java.net.*;
import java.io.*;

public class jhttp
{
  public static void main ( String[] args ) throws IOException
  {
    try
    {
      URL url = new URL( args[0] );

      URLConnection c = url.openConnection();

      for (int i=0; ; i++)
      {
            String name = c.getHeaderFieldKey(i);
            String value = c.getHeaderField(i);

            if (name == null && value == null)     // end of headers
            {
              break;
            }

            if (name == null)     // first line of headers
            {
              System.out.println("Server HTTP version, Response
code:");
```

```
                    System.out.println(value);
                    System.out.print("\n");
                  }
                  else
                  {
                    System.out.println(name + "=" + value);
                  }
              }
            }
            catch (Exception e) {}
          }
        }
```

4.2. Code de réponse : classe HttpURLConnection

Tester et commenter le code suivant :

```
import java.net.*;
import java.io.*;

public class hrc
{
  public static void main ( String[] args ) throws Exception
  {
    try
    {
      URL url = new URL( args[0] );

      HttpURLConnection  c = (HttpURLConnection) url.openConnection();

      System.out.println( c.getResponseCode() );
    }
    catch (Exception e) {}
  }
}
```

5. Wget en Java : aspirer un site

I. Réalisation d'un programme wget en Java
   a. Le but de cette exercice est de réaliser un programme sémilaire a wget de linux (voir la documentation sur internet)
   b. Exemple de code
      i. https://docs.oracle.com/javase/tutorial/networking/urls/readingURL.html
      ii. http://www.java2s.com/Tutorial/Java/0320__Network/ReaddatafromaURL.htm
   c. Chercher des liens dans le fichier téléchargé
      i. http://stackoverflow.com/questions/5120171/extract-links-from-a-web-page
      ii. http://stackoverflow.com/questions/285619/how-to-detect-the-presence-of-url-in-a-string
      iii. http://www.rgagnon.com/javadetails/java-0639.html
      iv. http://blog.codehangover.com/read-html-with-java-then-7-fun-things-to-do-to-it/ (très bien il y a tout)