# Envoyer des mails en Java

*Michel CASABIANCA*
*casa@sweetohm.net*

Le mail est certainement le service Internet le plus utilisé, il est donc tentant d'en envoyer d'une applet ou d'un programme java. Nous allons voir dans la suite de cet article comment procéder, avec le code Java illustrant cette méthode.

## Créer un socket vers le port 25 du serveur SMTP

On crée un socket vers le serveur SMTP dont l'adresse est hôte et le port est 25 et on intercepte une éventuelle exception avec les clauses try et catch :

```
try {
  socket=new Socket(hote,port);
  sortie=new DataOutputStream(new
    BufferedOutputStream(socket.getOutputStream()));
}
catch() {
  System.out.println("Erreur de connexion au serveur :");
  e.printStackTrace();
}
```

Si cette connexion se fait d'une applet Java, il faut impérativement que le serveur SMTP tourne sur la même machine que le serveur HTTP où se trouve la page. Si ce n'est pas le cas, une SecurityException sera levée et la connexion ne se fera pas. Cette restriction est due au modèle de sécurité des navigateurs Java qui n'autorisent une connexion de l'applet que vers le serveur d'où elle est issue. En pratique, dans le cas d'une applet, l'hôte est déterminé avec les méthode getCodeBase() (qui renvoie l'url de la machine sur laquelle sont stockées les classes java de l'applet) et getHost() qui en extrait le nom de domaine du serveur :

```
hote=getCodeBase().getHost();
```

## Envoyer l'en-tête du message

Il nous faut respecter le protocole SMTP, pour ce faire on commence par annoncer l'expéditeur et le destinataire :

```
sortie.writeBytes("MAIL FROM: "+expediteur+"\n");
sortie.writeBytes("RCPT TO: "+destinataire+"\n");
sortie.writeBytes("DATA\n");
```

On utilise ici la méthode writeBytes(String s) qui envoie les caractères sous forme ASCII. En effet,

les caractères en Java sont codés sur 16 bits d'après le standard UNICODE qui permet de représenter les alphabets et idéogrammes des principales langues contemporaines.

Le "DATA\n" indique au serveur SMTP que le message suit.

## Envoi du message

Il ne reste plus qu'à envoyer le corps du message, que l'on doit terminer par "\n.\n", soit un point seul sur une ligne.

```
      sortie.writeBytes("From: "+expediteur+"\n");
      sortie.writeBytes("To: "+destinataire+"\n");
      sortie.writeBytes("Subject: "+sujet+"\n");
      sortie.writeBytes("Date: "+
        (new Date()).toGMTString()+"\n\n");
      sortie.writeBytes(message);
      sortie.writeBytes("\n.\n");
```

Comme nous avons utilisé un BufferedOutputStream, il faut le purger avant de quitter, c'est que qui est fait avec :

```
      Sortie.flush();
```

Le source complet de cette applet est java-mail.zip. Dans le même fichier ZIP, on trouvera la classe java et un exemple de tag HTML pour insérer cette applet dans une page. On notera que cet applet ne peut fonctionner que si votre fournisseur d'accès fait tourner un serveur SMTP sur le serveur HTTP (ce qui se fait rare).

## Démonstration de l'applet

Je ne peux plus vous montrer l'applet en fonctionnement dans la mesure où mon fournisseur d'accès ne fait plus tourner de serveur SMTP sur le serveur HTTP servant ces pages. De plus, cela m'évitera de recevoir des mails de testeurs fous :o)