JAVAMAIL API - SENDING AN HTML EMAIL

http://www.tutorialspoint.com/javamail_api/javamail_api_send_html_in_email.htm

Here is an example to send an HTML email from your machine. Here we have used JangoSMPT server via which emails are sent to our destination email address. The setup is explained in the Environment Setup chapter.

This example is very similar to sending simple email, except that, here we are using setContent method to set content whose second argument is "text/html" to specify that the HTML content is included in the message. Using this example, you can send as big as HTML content you like.

To send a email with HTML content, the steps followed are:

- Get a Session
- Create a default MimeMessage object and set *From, To, Subject* in the message.
- Set the actual message using setContent method as below:

```
message.setContent("<h1>This is actual message embedded in
   HTML tags</h1>", "text/html");
```

- Send the message using the Transport object.

Create Java Class

Create a java class file **SendHTMLEmail**, the contents of which are as follows:

```
package com.tutorialspoint;

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 SendHTMLEmail {
   public static void main(String[] args) {
      // Recipient's email ID needs to be mentioned.
      String to = "destinationemail@gmail.com";

      // Sender's email ID needs to be mentioned
      String from = "fromemail@gmail.com";
      final String username = "manishaspatil";//change accordingly
      final String password = "*******";//change accordingly

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

      Properties props = new Properties();
      props.put("mail.smtp.auth", "true");
      props.put("mail.smtp.starttls.enable", "true");
      props.put("mail.smtp.host", host);
      props.put("mail.smtp.port", "25");

      // 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");

    // Send the actual HTML message, as big as you like
    message.setContent(
               "<h1>This is actual message embedded in HTML tags</h1>",
              "text/html");

    // Send message
    Transport.send(message);

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

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

As we are using the SMTP server provided by the host provider JangoSMTP, we need to authenticate the username and password. The *javax.mail.PasswordAuthentication* class is used to authenticate the password.

Compile and Run

Now that our class is ready, let us compile the above class. I've saved the class SendHTMLEmail.java to directory : **/home/manisha/JavaMailAPIExercise**. We would need the jars *javax.mail.jar* and *activation.jar* in the classpath. Execute the command below to compile the class $boththejarsareplacedin/home/manisha/directory$ from command prompt:

```
javac -cp /home/manisha/activation.jar:/home/manisha/javax.mail.jar: SendHTMLEmail.java
```

Now that the class is compiled, execute the below command to run:

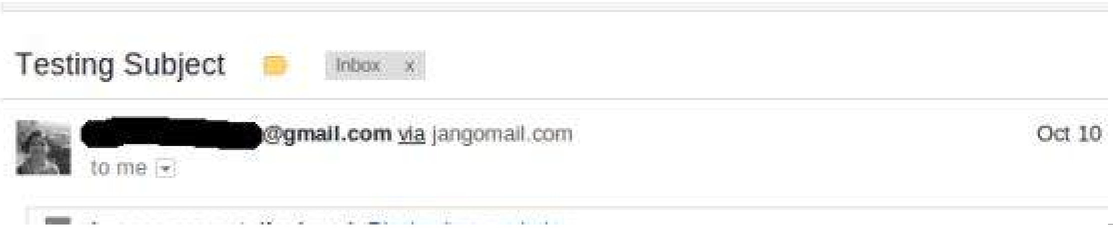
```
java -cp /home/manisha/activation.jar:/home/manisha/javax.mail.jar: SendHTMLEmail
```

Verify Output

You should see the following message on the command console:

```
Sent message successfully....
```

As I'm sending an email to my gmail address through JangoSMTP, the following mail would be received in my gmail account inbox:



Testing Subject

[redacted]@gmail.com via jangomail.com

Oct 10

to me

Images are not displayed. Display images below

This is actual message embedded in HTML tags

Loading [MathJax]/jax/output/HTML-CSS/jax.js