

转 JavaMail 发送邮件的实例

2013年10月12日 13:17:45 阅读数：2087

JavaMail是提供给开发者处理电子邮件相关的编程接口。它是Sun发布的用来处理email的API。今天学习了一下JavaMail，javamail发送邮件确实是一个比较麻烦的问题。为了以后使用方便，自己写了段代码，打成jar包，以方便以后使用。呵呵

以下三段代码是我的全部代码，朋友们如果想用，直接复制即可。

第一个类：MailSenderInfo.java

```
[java]    
1.  /**  
2.  * 发送邮件需要使用的基本信息  
3.  */  
4.  import java.util.Properties;  
5.  public class MailSenderInfo {  
6.      // 发送邮件的服务器的IP和端口  
7.      private String mailServerHost;  
8.      private String mailServerPort = "25";  
9.      // 邮件发送者的地址  
10.     private String fromAddress;  
11.     // 邮件接收者的地址  
12.     private String toAddress;  
13.     // 登陆邮件发送服务器的用户名和密码  
14.     private String userName;  
15.     private String password;  
16.     // 是否需要身份验证  
17.     private boolean validate = false;  
18.     // 邮件主题  
19.     private String subject;  
20.     // 邮件的文本内容  
21.     private String content;  
22.     // 邮件附件的文件名  
23.     private String[] attachFileNames;  
24.     /**  
25.     * 获得邮件会话属性  
26.     */  
27.     public Properties getProperties(){  
28.         Properties p = new Properties();  
29.         p.put("mail.smtp.host", this.mailServerHost);  
30.         p.put("mail.smtp.port", this.mailServerPort);  
31.         p.put("mail.smtp.auth", validate ? "true" : "false");  
32.         return p;  
33.     }  
34.     public String getMailServerHost() {  
35.         return mailServerHost;  
36.     }  
37.     public void setMailServerHost(String mailServerHost) {  
38.         this.mailServerHost = mailServerHost;  
39.     }  
40.     public String getMailServerPort() {  
41.         return mailServerPort;  
42.     }  
43.     public void setMailServerPort(String mailServerPort) {  
44.         this.mailServerPort = mailServerPort;  
45.     }  
46.     public boolean isValid() {  
47.         return validate;  
48.     }  
49.     public void setValidate(boolean validate) {  
50.         this.validate = validate;  
51.     }  
52.     public String[] getAttachFileNames() {  
53.         return attachFileNames;  
54.     }  
55.     public void setAttachFileNames(String[] fileNames) {  
56.         this.attachFileNames = fileNames;  
57.     }  
58.     public String getFromAddress() {  
59.         return fromAddress;  
60.     }  
61.     public void setFromAddress(String fromAddress) {  
62.         this.fromAddress = fromAddress;  
63.     }  
64.     public String getPassword() {  
65.         return password;  
66.     }  
67.     public void setPassword(String password) {  
68.         this.password = password;  
69.     }  
70.     public String getToAddress() {  
71.         return toAddress;  
72.     }  
73.     public void setToAddress(String toAddress) {  
74.         this.toAddress = toAddress;
```

```

75.     }
76.     public String getUserName() {
77.         return userName;
78.     }
79.     public void setUserName(String userName) {
80.         this.userName = userName;
81.     }
82.     public String getSubject() {
83.         return subject;
84.     }
85.     public void setSubject(String subject) {
86.         this.subject = subject;
87.     }
88.     public String getContent() {
89.         return content;
90.     }
91.     public void setContent(String textContent) {
92.         this.content = textContent;
93.     }
94. }

```

第二个类：SimpleMailSender.java

```

[java]
1.  import java.util.Date;
2.  import java.util.Properties;
3.  import javax.mail.Address;
4.  import javax.mail.BodyPart;
5.  import javax.mail.Message;
6.  import javax.mail.MessagingException;
7.  import javax.mail.Multipart;
8.  import javax.mail.Session;
9.  import javax.mail.Transport;
10. import javax.mail.internet.InternetAddress;
11. import javax.mail.internet.MimeBodyPart;
12. import javax.mail.internet.MimeMessage;
13. import javax.mail.internet.MimeMultipart;
14.
15. /**
16.  * 简单邮件（不带附件的邮件）发送器
17.  */
18. public class SimpleMailSender {
19. /**
20.  * 以文本格式发送邮件
21.  * @param mailInfo 待发送的邮件的信息
22.  */
23.     public boolean sendTextMail(MailSenderInfo mailInfo) {
24.         // 判断是否需要身份认证
25.         MyAuthenticator authenticator = null;
26.         Properties pro = mailInfo.getProperties();
27.         if (mailInfo.isValidate()) {
28.             // 如果需要身份认证，则创建一个密码验证器
29.             authenticator = new MyAuthenticator(mailInfo.getUserName(), mailInfo.getPassword());
30.         }
31.         // 根据邮件会话属性和密码验证器构造一个发送邮件的session
32.         Session sendMailSession = Session.getDefaultInstance(pro, authenticator);
33.         try {
34.             // 根据session创建一个邮件消息
35.             Message mailMessage = new MimeMessage(sendMailSession);
36.             // 创建邮件发送者地址
37.             Address from = new InternetAddress(mailInfo.getFromAddress());
38.             // 设置邮件消息的发送者
39.             mailMessage.setFrom(from);
40.             // 创建邮件的接收者地址，并设置到邮件消息中
41.             Address to = new InternetAddress(mailInfo.getToAddress());
42.             mailMessage.setRecipient(Message.RecipientType.TO, to);
43.             // 设置邮件消息的主题
44.             mailMessage.setSubject(mailInfo.getSubject());
45.             // 设置邮件消息发送的时间
46.             mailMessage.setSentDate(new Date());
47.             // 设置邮件消息的主要内容
48.             String mailContent = mailInfo.getContent();
49.             mailMessage.setText(mailContent);
50.             // 发送邮件
51.             Transport.send(mailMessage);
52.             return true;
53.         } catch (MessagingException ex) {
54.             ex.printStackTrace();
55.         }
56.         return false;
57.     }
58.
59. /**
60.  * 以HTML格式发送邮件
61.  * @param mailInfo 待发送的邮件信息
62.  */
63.     public static boolean sendHtmlMail(MailSenderInfo mailInfo){
64.         // 判断是否需要身份认证
65.         MyAuthenticator authenticator = null;

```

```

66. Properties pro = mailInfo.getProperties();
67. //如果需要身份认证,则创建一个密码验证器
68. if (mailInfo.isValidate()) {
69.     authenticator = new MyAuthenticator(mailInfo.getUserName(), mailInfo.getPassword());
70. }
71. // 根据邮件会话属性和密码验证器构造一个发送邮件的session
72. Session sendMailSession = Session.getDefaultInstance(pro,authenticator);
73. try {
74.     // 根据session创建一个邮件消息
75.     Message mailMessage = new MimeMessage(sendMailSession);
76.     // 创建邮件发送者地址
77.     Address from = new InternetAddress(mailInfo.getFromAddress());
78.     // 设置邮件消息的发送者
79.     mailMessage.setFrom(from);
80.     // 创建邮件的接收者地址,并设置到邮件消息中
81.     Address to = new InternetAddress(mailInfo.getToAddress());
82.     // Message.RecipientType.TO属性表示接收者的类型为TO
83.     mailMessage.setRecipient(Message.RecipientType.TO,to);
84.     // 设置邮件消息的主题
85.     mailMessage.setSubject(mailInfo.getSubject());
86.     // 设置邮件消息发送的时间
87.     mailMessage.setSentDate(new Date());
88.     // MiniMultipart类是一个容器类,包含MimeBodyPart类型的对象
89.     Multipart mainPart = new MimeMultipart();
90.     // 创建一个包含HTML内容的MimeBodyPart
91.     BodyPart html = new MimeBodyPart();
92.     // 设置HTML内容
93.     html.setContent(mailInfo.getContent(), "text/html; charset=utf-8");
94.     mainPart.addBodyPart(html);
95.     // 将MiniMultipart对象设置为邮件内容
96.     mailMessage.setContent(mainPart);
97.     // 发送邮件
98.     Transport.send(mailMessage);
99.     return true;
100. } catch (MessagingException ex) {
101.     ex.printStackTrace();
102. }
103. return false;
104. }
105. }

```

第三个类：MyAuthenticator.java

```

[java]
1. import javax.mail.*;
2.
3. public class MyAuthenticator extends Authenticator{
4.     String userName=null;
5.     String password=null;
6.
7.     public MyAuthenticator(){
8.     }
9.     public MyAuthenticator(String username, String password) {
10.         this.userName = username;
11.         this.password = password;
12.     }
13.     protected PasswordAuthentication getPasswordAuthentication(){
14.         return new PasswordAuthentication(userName, password);
15.     }
16. }
17.

```

下面给出使用上面三个类的代码：

注：这段不是转载的，自己试过确实好用~~

[java]  

```
1. public class test {
2.     public static void main(String[] args){
3.         //这个类主要是设置邮件
4.         MailSenderInfo mailInfo = new MailSenderInfo();
5.         mailInfo.setMailServerHost("smtp.163.com");
6.         mailInfo.setMailServerPort("25");
7.         mailInfo.setValidate(true);
8.         mailInfo.setUserName("07gdg");
9.         mailInfo.setPassword("xxx");//邮箱密码，此处省略
10.        mailInfo.setFromAddress("07gdg@163.com");//发送邮件，拿我们系的公共邮箱试一试~
11.        mailInfo.setToAddress("leixiaohua1020@126.com");//接收邮箱
12.        mailInfo.setSubject("2013年存储展望:大数据、云依然是主旋律");
13.        mailInfo.setContent("又到了年底，对于存储领域来说，这一年发生了太多的事，大数据迅速崛起成为IT热词，而与大数据相关的大数据衍生行业也得到了繁荣发展。
2012年大数据从天而降，迅速占领科技报端，混合云存储初现端倪，NAS存储重现胜景，闪存技术和融合式基础架构也跻身主流，可以说2012年存储市场繁荣异常。");
14.        //这个类主要来发送邮件
15.        SimpleMailSender sms = new SimpleMailSender();
16.        sms.setTextMail(mailInfo);//发送文本格式
17.        sms.sendHtmlMail(mailInfo);//发送html格式
18.    }
19.
20. }
```

文章标签：[JavaMail](#) [发送邮件](#) [实例](#)

个人分类：[J2EE](#)

此PDF由spygg生成,请尊重原作者版权!!!

我的邮箱:liushidc@163.com