**J office 整合外部邮件详细设计**

1. **需求分析**

实现的功能类似Outlook，但不需要这么强大，仅需要能接收外部的邮件，并且能正常显示出来，并且供用户进行全文检索，用来支持外部邮件收发功能。

1. **设计要点**

将节点放在通讯管理下，取名外部邮件，并树的形式加入两叶子结点

外部邮件-

+新建外部邮件

+个人外部邮箱

+外部邮箱设置

当点击“新建外部邮件”时跳转到外部邮件发件页面，实现利用外部邮件服务器进行邮件发送。

当点击“个人外部邮箱”时查询数据库，将也同步过来的邮件显示出来。

显示的数据的列表工具栏上做如下按妞“收取外部邮件”，等以实现外部邮件同步收取与查询。

1. **表设计**

Out\_mail\_user\_seting外部邮件的用户设置表

|  |  |  |
| --- | --- | --- |
| id | 自增ID | pk |
| userId | Oa系统用户ID | Fk ref\_to\_ AppUser |
| userName | Oa系统用户名 |  |
| mailAddress | 外部邮件地址 |  |
| mailPass | 密码 |  |
| smtpHost | Smtp主机 |  |
| smtpPort | Smtp端口 |  |
| smtpAuth | 不详 | true |
| popHost | Pop主机 |  |
| popPort | Pop端口 |  |

Out\_mail 接收外部邮件表

|  |  |  |
| --- | --- | --- |
| mailId | 自增列 | pk |
| userid | Oa系统用ID | Fk\_ref\_to\_Out\_mail\_user\_seting |
| title | 主题 |  |
| content | 内容 | Html形式 |
| senderAddresses | 发件人地址 |  |
| senderName | 发件人名称 |  |
| receiverAddresses | 收件人地址 | 多人以豆号分隔 |
| receiverNames | 收件人名称 | 多人以豆号分隔 |
| cCAddresses | 抄送人地址 | 多人以豆号分隔 |
| cCNames | 抄送人名称 | 多人以豆号分隔 |
| *bCCAddresses* | 暗送人地址 | 多人以豆号分隔 |
| *bCCNames* | 暗送人名称 | 多人以豆号分隔 |
| mailDate | 日期 | 多人以豆号分隔 |
| fileIds | 附件 | 多附件以豆号分隔 |
| fileNames | 附件 |  |
| mailFlag | 收发标志 | S:为发(send),R为收(Receive) |

Out\_mail\_file

|  |  |  |
| --- | --- | --- |
| mailId | 邮件id | Fk\_ref\_to\_Out\_mai |
| fileid | 文件id | Fk\_ref\_to\_file\_attach |

1. **实现要点**
2. 根据表Out\_mail\_user\_seting, Out\_mail利用代码生成器创建DAO对像的接口与实现，并将其放于src\com\htsoft\oa\dao\communicate文件夹下。
3. 根据表Out\_mail\_user\_seting, Out\_mail利用代码生成器创建MODEL对像的接口与实现，并将其放于src\com\htsoft\oa\model\communicate文件夹下。
4. 根据表Out\_mail\_user\_seting, Out\_mail利用代码生成器创建SERVICE对像的接口与实现, 并将其放于src\com\htsoft\oa\service\communicate文件夹下。
5. 根据表Out\_mail\_user\_seting, Out\_mail利用代码生成器创建ACTION对像的接口与实现, 并将其放于src\com\htsoft\oa\action\communicate文件夹下。

修改DAO层代码，向外部邮件服务器收件与发件的方法，并将附件放在jofficev131\web\userfiles\outmailfiles\文件夹下，并为每一用户建一个文件夹，供SERVICE调用,调方法主要运用javamail实现,主要代码如下，

**//发邮件**

**public** **void** send(String user, String pass, String to, String subject,

String html,String affix) **throws** Exception {

Security.*addProvider*(**new** com.sun.net.ssl.internal.ssl.Provider());

**final** String SSL\_FACTORY = "javax.net.ssl.SSLSocketFactory";

// Get a Properties object

Properties props = System.*getProperties*();

props.setProperty("mail.smtp.host", smtpHost);

props.setProperty("mail.smtp.socketFactory.class", SSL\_FACTORY);

props.setProperty("mail.smtp.socketFactory.fallback", "false");

props.setProperty("mail.smtp.port", smtpPort);

props.setProperty("mail.smtp.socketFactory.port", smtpPort);

props.put("mail.smtp.auth", smtpAuth);

**final** String username = user;

**final** String password = pass;

Session session = Session.*getDefaultInstance*(props,

**new** Authenticator() {

**protected** PasswordAuthentication getPasswordAuthentication() {

**return** **new** PasswordAuthentication(username, password);

}

});

// -- Create a new message --

Message msg = **new** MimeMessage(session);

// -- Set the FROM and TO fields --

msg.setFrom(**new** InternetAddress(username));

msg.setRecipients(Message.RecipientType.*TO*, InternetAddress.*parse*(to,

**false**));

msg.setSubject(subject);

mp=**new** MimeMultipart();

BodyPart bp = **new** MimeBodyPart();

bp.setContent("<meta http-equiv=Content-Type content=text/html; charset=gb2312>" + html, "text/html;charset=UTF-8");

mp.addBodyPart(bp);

BodyPart bp1 = **new** MimeBodyPart();

//附件

FileDataSource fileds = **new** FileDataSource(affix);

bp1.setDataHandler(**new** DataHandler(fileds));

bp1.setFileName(fileds.getName());

mp.addBodyPart(bp1);

msg.setSentDate(**new** Date());

msg.setContent(mp);

Transport.*send*(msg);

System.*out*.println("Message sent.");

}

//收邮件

**public** **void** fetch(String username, String password) **throws** Exception {

Security.*addProvider*(**new** com.sun.net.ssl.internal.ssl.Provider());

**final** String SSL\_FACTORY = "javax.net.ssl.SSLSocketFactory";

// Get a Properties object

Properties props = System.*getProperties*();

props.setProperty("mail.pop3.socketFactory.class", SSL\_FACTORY);

props.setProperty("mail.pop3.socketFactory.fallback", "false");

props.setProperty("mail.pop3.port", popPort);

props.setProperty("mail.pop3.socketFactory.port", popPort);

// 以下步骤跟一般的JavaMail操作相同

Session session = Session.*getDefaultInstance*(props, **null**);

// 请将红色部分对应替换成你的邮箱帐号和密码

URLName urln = **new** URLName("pop3", popHost, Integer.*parseInt*(popPort),

**null**, username, password);

Store store = session.getStore(urln);

Folder inbox = **null**;

**try** {

store.connect();

inbox = store.getFolder("INBOX");

inbox.open(Folder.*READ\_ONLY*);

FetchProfile profile = **new** FetchProfile();

profile.add(FetchProfile.Item.*ENVELOPE*);

Message[] messages = inbox.getMessages();

inbox.fetch(messages, profile);

System.*out*.println("收件箱的邮件数：" + messages.length);

**for** (**int** i = 0; i < messages.length; i++) {

// 邮件发送者

String from = *decodeText*(messages[i].getFrom()[0].toString());

InternetAddress ia = **new** InternetAddress(from);

System.*out*.println("FROM:" + ia.getPersonal() + '('

+ ia.getAddress() + ')');

// 邮件标题

System.*out*.println("TITLE:" + messages[i].getSubject());

// 邮件大小

System.*out*.println("SIZE:" + messages[i].getSize());

// 邮件发送时间

System.*out*.println("DATE:" + messages[i].getSentDate());

/\*\*\*

在此处将邮件内容存入数据库

Out\_mial.settitle(messages[i].gettitle());

Out\_mailservice.save(Out\_mial);

\*\*\*/

}

} **finally** {

**try** {

inbox.close(**false**);

} **catch** (Exception e) {

}

**try** {

store.close();

} **catch** (Exception e) {

}

}

}