



JSP程序设计教程

第7章 JSP实用组件



第7章 JSP实用组件

- 7.1 JSP文件操作 ✓
- 7.2 发送E-mail ✓
- 7.3 JSP动态图表 ✓
- 7.4 JSP报表 ✓



7.1 JSP文件操作

在Web开发中，对文件操作是一项非常实用的功能，例如，文件的上传与下载。在JSP中，常用的文件上传与下载组件是jspSmartUpload，该组件是一个可免费使用的全功能的文件上传下载组件。通过该组件可以很方便地实现文件的上传与下载。



7.1 JSP文件操作

7.1.1 jspSmartUpload组件的安装与配置 ✓

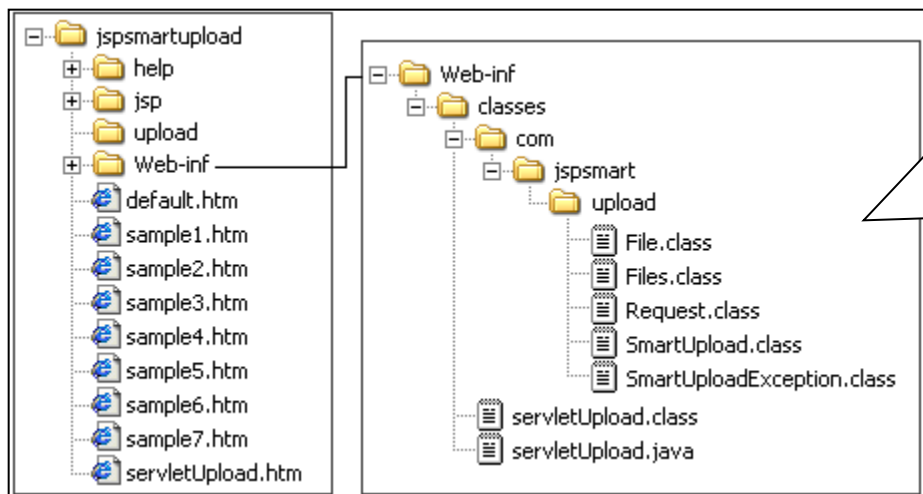
7.1.2 jspSmartUpload组件中的常用类 ✓

7.1.3 采用jspSmartUpload组件进行文件操作 ✓



7.1.1 jspSmartUpload组件的安装与配置

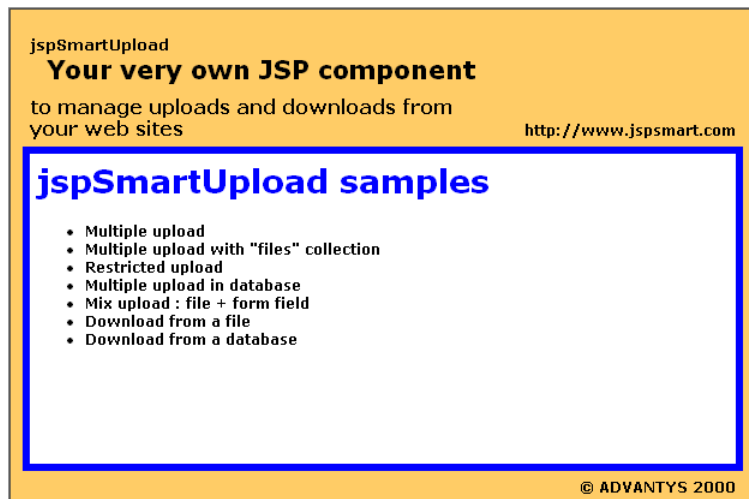
jspSmartUpload组件可以通过网络搜索找到相关网站进行下载。下载的文件名为jspSmartUpload.zip，解压后得到的是一个Web应用程序，其目录结构如下图所示。



default.htm为Web应用的首页面，sample1.htm~sample7.htm文件分别为7个实例中的供用户选择上传文件和下载文件的静态页面，help目录下存放了jspSmartUpload组件的说明文件，jsp目录下存放了与sample1.htm~sample7.htm文件对应的JSP文件，用来实现当前实例中的动态内容，在这些JSP文件中将调用jspSmartUpload组件中的类来实现文件的上传或下载，Web-inf目录下存放的就是jspSmartUpload组件中的类文件。

7.1.1 jspSmartUpload组件的安装与配置

若想运行该Web应用，首先将Web-inf目录名更改为WEB-INF，然后将jspsmartupload整个文件夹拷贝到Tomcat安装目录下的webapps目录下，最后访问地址“<http://localhost:8080/jspsmartupload/default.htm>”即可进入Web应用的首页面，运行结果如下图所示。



7.1.1 jspSmartUpload组件的安装与配置

可以通过如下的方法将Web-inf\classes目录下的文件打包成自己的JAR文件，以便在以后的程序开发时，可直接通过将该文件拷贝到应用的WEB-INF\lib目录下来应用jspSmartUpload组件实现文件的上传与下载。

(1) 若JDK安装在了C:\jdk1.6.0_03目录下，则环境变量的系统变量中应存在如下的配置：

```
JAVA_HOME=C:\jdk1.6.0_03  
PATH=%JAVA_HOME%\bin
```

(2) 打开“命令提示符”窗口，进入到jspSmartUpload.zip文件解压后的目录的classes子目录下，输入以下命令行进行文件打包：

7.1.1 jspSmartUpload组件的安装与配置

可以通过如下的方法将Web-inf\classes目录下的文件打包成自己的JAR文件，以便在以后的程序开发时，可直接通过将该文件拷贝到应用的WEB-INF\lib目录下来应用jspSmartUpload组件实现文件的上传与下载。

(1) 若JDK安装在了C:\jdk1.6.0_03目录下，则环境变量的系统变量中应存在如下的配置：

```
JAVA_HOME=C:\jdk1.6.0_03  
PATH=%JAVA_HOME%\bin
```

(2) 打开“命令提示符”窗口，进入到jspSmartUpload.zip文件解压后的目录的classes子目录下，输入以下命令行进行文件打包：

7.1.1 jspSmartUpload组件的安装与配置

```
jar cvf jspSmartUpload.jar com servletUpload.class  
servletUpload.java
```

com为classes目录下的com文件夹，jspSmartUpload.jsp文件即为打包后的文件。



7.1.2 jspSmartUpload组件中的常用类

在jspSmartUpload组件中主要包含了File，Files，Request和SmartUpload核心类，下面对这些核心类分别进行介绍。

1. File类

该类不同于java.io.File类，在编写程序时应注意使用。File类用于保存单个上传文件的相关信息，如上传文件的文件名、文件大小、文件数据等，File类的常用方法如下表所示。

7.1.2 jspSmartUpload组件中的常用类

方 法	说 明
saveAs()	该方法用于保存文件
isMissing()	该方法用于判断用户是否选择了文件，即表单中对应的<input type="file">标记实现的文件选择域中是否有值，该方法返回boolean型值，选择了文件时，返回false，否则返回true
getFieldName()	获取Form表单中当前上传文件所对应的表单项的名称
getFileName()	获取文件的文件名，该文件名不包含目录
getFilePathName()	获取文件的文件全名，获取的值是一个包含目录的完整文件名
getFileExt()	获取文件的扩展名，即后缀名，不包含“.”符号
getContentType()	获取文件MIME类型，如“text/plain”
getContentString()	获取文件的内容，返回值为String型
getSize()	获取文件的大小，单位byte，返回值为int型
getBinaryData(int index)	获取文件数据中参数index指定位置处的一个字节，用于检测文件

7.1.2 jspSmartUpload组件中的常用类

Files类中的**saveAs()**方法用于保存文件，在**File**类中提供了以下两种形式的**saveAs()**方法：

saveAs(String destFilePathName)方法

saveAs(String destFilePathName, int optionSaveAs)方法

这两个方法都没有返回值，第一种形式与**saveAs(destFilePathName, 0)**执行效果相同。
destFilePathName：指定文件保存的路径，包括文件名，其值应以“/”开头。
optionSaveAs：保存目标选项。该选项有3个值，分别是**SAVEAS_AUTO**、**SAVEAS_VIRTUAL**和**SAVEAS_PHYSICAL**。它们是**File**类中的静态字段，分别表示整数0、1和2。

7.1.2 jspSmartUpload组件中的常用类

将optionSaveAs参数设为SAVEAS_VIRTUAL选项值，则通知jspSmartUpload组件以web应用的根目录为文件根目录，然后加上destFilePathName参数指定的路径来保存文件；设为SAVEAS_PHYSICAL值，则一种情况是通知jspSmartUpload组件将以Web服务器的安装路径中的磁盘根目录为文件根目录，然后加上destFilePathName参数指定的路径来保存文件，另一种情况则以destFilePathName参数指定的目录为最终目录来保存文件；设为SAVEAS_AUTO值，则首先以SAVEAS_VIRTUAL方式来保存文件，若Web应用下由destFilePathName参数指定的路径不存在，则以SAVEAS_PHYSICAL方式保存文件。例如，若Web服务器（Tomcat）的安装目录为“C:\Tomcat 6.0”，当前web应用为“FileUpDown”时，下面分别应用这3个选项保存文件。

7.1.2 jspSmartUpload组件中的常用类

(1) 使用SAVEAS_VIRTUAL选项值

```
saveAs("/file/myfile.txt",File.SAVEAS_VIRTUAL)或  
saveAs("/file/myfile.txt",1)
```

若FileUpDown应用下存在“file”子目录，则将上传的文件以“myfile.txt”为文件名进行保存，实际的保存路径如下：

```
C:\Tomcat 6.0\webapps\FileUpDown\file\myfile.txt
```

若不存在“file”子目录，则抛出下面的异常：

```
This path does not exist (1135)
```

(2) 使用SAVEAS_PHYSICAL选项值

```
saveAs("/file/myfile.txt",File.SAVEAS_PHYSICAL)或  
saveAs("/file/myfile.txt",2)
```

7.1.2 jspSmartUpload组件中的常用类

因为Tomcat安装在C盘，因此若E盘根目录下存在“file”子目录，则将上传的文件以“myfile.txt”为文件名进行保存，实际的保存路径如下：

```
C:\file\myfile.txt
```

若C盘根目录下不存在file子目录，而FileUpDown应用的根目录下存在file子目录，则抛出下面的异常：

```
The path is not a physical path
```

否则抛出下面的异常：

```
This path does not exist (1135)
```

7.1.2 jspSmartUpload组件中的常用类

使用SAVEAS_PHYSICAL选项值时，可以将上传的文件保存到由destFilePathName参数指定的一个具体的目录下，如：

```
saveAs("D:/temp/myfile.txt",File.SAVEAS_PHYSICAL)
```

最终文件的实际保存路径如下：

```
D:\temp\myfile.txt
```

（3）使用SAVEAS_AUTO选项值

```
saveAs("/file/myfile.txt",File.SAVEAS_AUTO)或  
saveAs("/file/myfile.txt",0)
```

若FileUpDown应用根目录下存在“file”子目录，则以SAVEAS_VIRTUAL方式保存文件，否则以SAVEAS_PHYSICAL方式保存文件。通常情况下应使用SAVEAS_VIRTUAL方式保存文件，以便程序的移植。

7.1.2 jspSmartUpload组件中的常用类

2. Files类

Files类存储了所有上传的文件，通过类中的方法可获得上传文件的数量和总长度等信息。**Files**类中的常用方法如下表所示。

方 法	说 明
getCount()	获取上传文件的数目，返回值为int型
getSize()	获取上传文件的总长度，单位byte，返回值为long型
getFile(int index)	获取参数index指定位置处的com.jspsmart.upload.File对象
getCollection()	将所有File对象以Collection形式返回
getEnumeration()	将所有File对象以Enumeration形式返回

7.1.2 jspSmartUpload组件中的常用类

Files类中的**getCollection()**方法和**getEnumeration()**方法将所有的**File**对象分别以**Collection**和**Enumeration**形式返回，它们的源代码如下。

(1) **getCollection()**方法

将所有**File**对象以**Collection**的形式返回，以便其他应用程序引用，该方法的具体代码如下：

```
public Collection getCollection(){  
    return m_files.values();  
}
```

其中**m_files**为**Files**类中的属性，其类型为**Hashtable**，它存储了所有的**File**对象。

7.1.2 jspSmartUpload组件中的常用类

(2) getEnumeration()方法

将所有File对象以Enumeration形式返回，以便其他应用程序引用，该方法的具体代码如下：

```
public Enumeration getEnumeration(){  
    return m_files.elements();  
}
```

m_files为Files类中的属性，其类型为Hashtable，它存储了所有的File对象。

3. Request类

7.1.2 jspSmartUpload组件中的常用类

设置该类的目的，是因为当Form表单用来实现文件上传时，通过JSP的内置对象request的getParameter()方法无法获取其他表单项的值，所以提供了该类来获取，Request类中提供的方法如下表所示。

方 法	说 明
getParameter(String name)	获取Form表单中由参数name指定的表单元素的值，如<input type="text" name="user">，当该表单元素不存在时，返回null
getParameterNames()	获取Form表单中除<input type="file">外的所有表单元素的名称，它返回一个枚举型对象
getParameterValues(String name)	获取Form表单中多个具有相同名称的表单元素的值，该名称由参数name指定，该方法返回一个字符串数组

7.1.2 jspSmartUpload组件中的常用类

4. SmartUpload类

SmartUpload类用于实现文件的上传与下载操作，该类中提供的方法如下。

(1) 文件上传与文件下载必须实现的方法
在使用jspSmartUpload组件实现文件上传与下载时，必须先实现initialize()方法。在SmartUpload类中提供了该方法的3种形式：

```
initialize(ServletConfig config, HttpServletRequest request,  
HttpServletResponse response)  
initialize(ServletContext application, HttpSession session, HttpServletRequest  
request, HttpServletResponse response, JspWriter out)  
initialize(PageContext pageContext)
```

7.1.2 jspSmartUpload组件中的常用类

通常应用第3种形式的方法，该方法中的pageContext参数为JSP的内置对象（页面上下文）。

（2）文件上传使用的方法

实现文件上传，首先应实现initialize()方法，然后实现如下的两个方法即可将文件上传到服务器中。

① upload()方法

实现了initialize()方法后，紧接着就应实现该方法。upload()方法用来完成一些准备操作。首先在该方法中调用JSP的内置对象request的getInputStream()方法获取客户端的输入流，然后通过该输入流的read()方法读取用户上传的所有文件数据到字节数组中，然后在循环语句中从该字节数组中提取每个文件的数据，并将当前提取出的文件的信息封装到File类对象中，最后将该File类对象通过Files类的addFile()方法添加到Files类对象中。

7.1.2 jspSmartUpload组件中的常用类

② save()方法

在实现了initialize()方法和upload()方法后，通过调用该方法就可将全部上传文件保存到指定目录下，并返回保存的文件个数。该方法具有以下两种形式：

```
save(String destPathName)
save(String destPathName, int option)
```

第一个方法等同于save(destPathName,0)或save(destPathName,File. SAVE_AUTO)。

实际上在SmartUpload类的save()方法中最终是调用File类中的saveAs()方法保存文件的，所以save()方法中的参数使用与File类的saveAs()方法中的参数使用是相同的。但在save()方法中option参数指定的保存选项的可选值为SAVE_AUTO，SAVE_VIRTUAL和SAVE_PHYSICAL。它们是SmartUpload类中的静态字段，分别表示整数0、1和2。

7.1.2 jspSmartUpload组件中的常用类

仅仅通过以上的两个方法就实现了文件的上传。下面介绍SmartUpload类中可用来限制上传文件和获取其他信息的主要方法。

① setDeniedFilesList(String deniedFilesList)方法

该方法用于设置禁止上传的文件。其中参数deniedFilesList指定禁止上传文件的扩展名，多个扩展名之间以逗号分隔。若禁止上传没有扩展名的文件，以“,,”表示。例如，setDeniedFilesList("exe,jsp,,bat")表示禁止上传*.exe、*.jsp、*.bat和不带扩展名的文件。

② setAllowedFilesList(String allowedFilesList)方法

该方法用于设置允许上传的文件。其中参数allowedFilesList指定允许上传文件的扩展名，多个扩展名之间以逗号分隔。若允许上传没有扩展名的文件，以“,,”表示。例如，setAllowedFilesList("txt,doc,,")表示只允许上传*.txt、*.doc和不带扩展名的文件。

7.1.2 jspSmartUpload组件中的常用类

③ setMaxFileSize(long maxFileSize)方法

该方法用于设定允许每个文件上传的最大长度，该长度由参数maxFileSize指定。

④ setTotalMaxFileSize(long totalMaxFileSize)方法

该方法用于设置允许上传文件的总长度，该长度由参数totalMaxFileSize指定。

上述的对上传文件进行限制的方法，需在upload()方法之前调用。下面为SmartUpload类中的获取文件信息的方法。

① getSize()方法

该方法用于获取上传文件的总长度，其具体代码如下：

7.1.2 jspSmartUpload组件中的常用类

```
public int getSize(){  
    return m_totalBytes;  
}
```

其中m_totalBytes为SmartUpload类中的属性，表示上传文件的总长度，它是在upload()方法中通过调用JSP内置对象request的getContentLength()方法被赋值的。

② getFiles()方法

获取全部上传文件，以Files对象形式返回。

③ getRequest()方法

获取com.jspsmart.upload.Request对象，然后通过该对象获得上传的表单中其他表单项的值。

（3）文件下载使用的方法

7.1.2 jspSmartUpload组件中的常用类

① setContentDisposition(String contentDisposition)方法

该方法用于将数据追加到MIME文件头的CONTENT-DISPOSITION域。参数contentDisposition为要添加的数据。进行文件下载时，将contentDisposition设为null，则组件将自动添加“attachment”，表示将下载的文件作为附件，IE浏览器会弹出“文件下载”对话框，而不是自动打开这个文件（IE浏览器一般根据下载的文件扩展名决定执行什么操作，扩展名为doc的文件将用Word打开）。

② downloadFile()方法

downloadFile()方法实现文件下载，SmartUpload类中提供了以下4种形式的downloadFile()方法：

```
downloadFile(String sourceFilePathName)
```

```
downloadFile(String sourceFilePathName, String contentType)
```

```
downloadFile(String sourceFilePathName, String contentType, String destFileName)
```

```
downloadFile(String sourceFilePathName, String contentType, String destFileName, int blockSize)
```

7.1.2 jspSmartUpload组件中的常用类

sourceFilePathName: 用于指定要下载文件的文件名（可带目录，如/file/myfile.txt或E:/file/ myfile.text），若该文件名存在当前应用下，则
`sourceFilePathName=pageContext.getServletContext().
getRealPath(sourceFilePathName)`。
contentType: 定一个文件内容类型（MIME格式的文件类型信息）。
destFileName: 指定下载的文件另存为的文件名。
blockSize: 指定存储读取的文件数据的字节数组的大小，默认值为65000。

通常使用第一种方法，如果需要更改文件的内容类型，或者更改下载文件另存为的文件名，或者更改用来存储读取的文件数据的字节数组的大小时，可应用后面的三种方法。



7.1.3 采用jspSmartUpload组件进行文件操作

jspSmartUpload组件最常用的功能就是实现文件的上传与下载。本节将通过一个具体的实例介绍应用jspSmartUpload组件实现文件上传与下载的方法。

【例7-1】 采用jspSmartUpload组件实现文件上传及下载





7.2 发送E-mail

7.2.1 Java Mail组件简介 ✓

7.2.2 Java Mail核心类简介 ✓

7.2.3 搭建Java Mail的开发环境 ✓


7.2.4 在JSP中应用Java Mail组件发送E-mail ✓





7.2.1 Java Mail组件简介

Java Mail是Sun公司发布用来处理E-mail的API，是一种可选的、用于读取、编写和发送电子消息的包（标准扩展）。使用Java Mail可以创建MUA（邮件用户代理“Mail User Agent”的简称）类型的程序，它类似于Eudora、Pine及Microsoft Outlook等邮件程序。其主要目的不是像发送邮件或提供MTA（邮件传输代理“Mail Transfer Agent”的简称）类型程序那样用于传输、发送和转发消息，而是可以与MUA类型的程序交互，以阅读和撰写电子邮件。MUA依靠MTA处理实际的发送任务。





7.2.2 Java Mail核心类简介

Java Mail API中提供很多用于处理E-mail的类，其中比较常用的有：**Session**（会话）类、**Message**（消息）类、**Address**（地址）类、**Authenticator**（认证方式）类、**Transport**（传输）类、**Store**（存储）类和**Folder**（文件夹）类等7个类。这7个类都可以在Java Mail API的核心包mail.jar中找到。

1. Session类

Java Mail API中提供了**Session**类，用于定义保存诸如**SMTP**主机和认证的信息的基本邮件会话。通过**Session**会话可以阻止恶意代码窃取其他用户在会话中的信息（包括用户名和密码等认证信息），从而让其他工作顺利执行。



7.2.2 Java Mail核心类简介

每个基于Java Mail的程序都需要创建一个Session或多个Session对象。由于Session对象利用java.util.Properties对象获取诸如邮件服务器、用户名、密码等信息，以及其他可在整个应用程序中共享的信息，所以在创建Session对象前，需要先创建java.util.Properties对象。创建java.util.Properties对象的代码如下：

```
Properties props=new Properties();
```

创建Session对象可以通过以下两种方法，不过，通常情况下会使用第二种方法创建共享会话。

（1）使用静态方法创建Session的语句如下：

```
Session session = Session.getInstance(props, authenticator);
```



7.2.2 Java Mail核心类简介

props为java.util. Properties类的对象，authenticator为Authenticator对象，用于指定认证方式。

(2) 创建默认的共享Session的语句如下：

```
Session defaultSession = Session.getDefaultInstance(props, authenticator);
```

props为java.util. Properties类的对象，authenticator为Authenticator对象，用于指定认证方式。

如果在进行邮件发送时，不需要指定认证方式，可以使用空值（null）作为参数authenticator的值，例如，创建一个不需要指定认证方式的Session对象的代码如下：

```
Session mailSession=Session.getDefaultInstance(props,null);
```



7.2.2 Java Mail核心类简介

2. Message类

Message类是电子邮件系统的核心类，用于存储实际发送的电子邮件信息。Message类是一个抽象类，要使用该抽象类可以使用其子类MimeMessage。该类保存在javax.mail.internet包中，可以存储MIME类型和报头（在不同的RFC文档中均有定义）消息，并且将消息的报头限制成只能使用US-ASCII字符，尽管非ASCII字符可以被编码到某些报头字段中。

如果想对MimeMessage类进行操作，首先要实例化该类的一个对象，在实例化该类的对象时，需要指定一个Session对象，这可以通过将Session对象传递给MimeMessage的构造方法来实现，例如，实例化MimeMessage类的对象message的代码如下：



7.2.2 Java Mail核心类简介

```
MimeMessage msg = new MimeMessage(mailSession);
```

实例化MimeMessage类的对象msg后，就可以通过该类的相关方法设置电子邮件信息的详细信息。

MimeMessage类中常用的方法包括以下几个。

（1）setText()方法

setText()方法用于指定纯文本信息的邮件内容。该方法只有一个参数，用于指定邮件内容。setText()方法的语法格式如下：

```
setText(String content)
```

content: 纯文本的邮件内容。



7.2.2 Java Mail核心类简介

(2) setContent()方法

setContent()方法用于设置电子邮件内容的基本机制，多数应用在发送HTML等纯文本以外的信息。该方法包括两个参数，分别用于指定邮件内容和MIME类型。

setContent()方法的语法格式如下：

```
setContent(Object content, String type)
```

content：用于指定邮件内容。

type：用于指定邮件内容类型。

例如，指定邮件内容为“你现在好吗”，类型为普通的文本，代码如下：

```
message.setContent("你现在好吗", "text/plain");
```



7.2.2 Java Mail核心类简介

(3) setSubject ()方法

setSubject()方法用于设置邮件的主题。该方法只有一个参数，用于指定主题内容。setSubject()方法的语法格式如下：

```
setSubject(String subject)
```

subject: 用于指定邮件的主题。

(4) saveChanges()方法

saveChanges()方法能够保证报头域同会话内容保持一致。saveChanges ()方法的使用方法如下：

```
msg.saveChanges();
```

(5) setFrom()方法



7.2.2 Java Mail核心类简介

setFrom()方法用于设置发件人地址。该方法只有一个参数，用于指定发件人地址，该地址为**InternetAddress**类的一个对象。**setFrom()**方法的使用方法如下：

```
msg.setFrom(new InternetAddress(from));
```

（6）setRecipients()方法

setRecipients()方法用于设置收件人地址。该方法有两个参数，分别用于指定收件人类型和收件人地址。

setRecipients()方法的语法格式如下：

```
setRecipients(RecipientType type, InternetAddress addres);
```

type：收件人类型。可以使用以下3个常量来区分收件人的类型。



7.2.2 Java Mail核心类简介

- ① Message.RecipientType.TO //发送
- ② Message.RecipientType.CC //抄送
- ③ Message.RecipientType.BCC //暗送

address: 收件人地址，可以为InternetAddress类的一个对象或多个对象组成的数组。

例如，设置收件人的地址为“wgh8007@163.com”的代码如

```
address=InternetAddress.parse("wgh8007@163.com",false);  
msg.setRecipients(Message.RecipientType.TO, toAddrs);
```

(7) setSentDate()方法

setSentDate()方法用于设置发送邮件的时间。该方法只有一个参数，用于指定发送邮件的时间。**setSentDate ()**方法的语法格式如下：



7.2.2 Java Mail核心类简介

```
setSentDate(Date date);
```

date: 用于指定发送邮件的时间。

(8) **getContent()**方法

getContent()方法用于获取消息内容，该方法无参数。

(9) **writeTo()**方法

writeTo()方法用于获取消息内容（包括报头信息），并将其内容写到一个输出流中。该方法只有一个参数，用于指定输出流。**writeTo()**方法的语法格式如下：

```
writeTo(OutputStream os)
```

os: 用于指定输出流。



7.2.2 Java Mail核心类简介

3. Address类

Address类用于设置电子邮件的响应地址。Address类是一个抽象类，要使用该抽象类可以使用其子类InternetAddress，该类保存在javax.mail.internet包中，可以按照指定的内容设置电子邮件的地址。如果想对InternetAddress类进行操作，首先要实例化该类的一个对象，在实例化该类的对象时，有以下两种方法。

(1) 创建只带有电子邮件地址的地址，可以把电子邮件地址传递给InternetAddress类的构造方法，代码如下：

```
InternetAddress address = new  
InternetAddress("wgh717@sohu.com");
```



7.2.2 Java Mail核心类简介

(2) 创建带有电子邮件地址并显示其他标识信息的地址，可以将电子邮件地址和附加信息同时传递给 `InternetAddress` 类的构造方法，代码如下：

```
InternetAddress address = new  
InternetAddress("wgh717@sohu.com","Wang GuoHui");
```

说明：Java Mail API没有提供检查电子邮件地址有效性的机制。如果需要可以自己编写检查电子邮件地址是否有效的方法。



7.2.2 Java Mail核心类简介

4. Authenticator类

Authenticator类通过用户名和密码来访问受保护的资源。Authenticator类是一个抽象类，要使用该抽象类首先需要创建一个Authenticator的子类，并重载 getPasswordAuthentication()方法，具体代码如下：

```
class WghAuthenticator extends Authenticator {  
    public PasswordAuthentication getPasswordAuthentication() {  
        String username = "wgh"; //邮箱登录账号  
        String pwd = "111";      //登录密码  
        return new PasswordAuthentication(username, pwd);  
    }  
}
```



7.2.2 Java Mail核心类简介

(2) 首先从指定协议的会话中获取一个特定的实例，然后传递用户名和密码，再发送信息，最后关闭连接，代

```
Transport transport =sess.getTransport("smtp");  
transport.connect(servername,from,password);  
transport.sendMessage(message,message.getAllRecipients());  
transport.close();
```

在发送多个消息时，建议采用第二种方法，因为它将保持消息间活动服务器的连接，而使用第一种方法时，系统将为每一个方法的调用建立一条独立的连接。

注意：如果想要查看经过邮件服务器发送邮件的具体命令，可以用`session.setDebug(true)`方法设置调试标志。



7.2.2 Java Mail核心类简介

然后再通过以下代码实例化新创建的Authenticator的子类，并将其与Session对象绑定：

```
Authenticator auth = new WghAuthenticator ();  
Session session = Session.getDefaultInstance(props, auth);
```

4. Authenticator类

Transport类用于使用指定的协议（通常是SMTP）发送电子邮件。Transport类提供了以下两种发送电子邮件的方法。

（1）只调用其静态方法send()，按照默认协议发送电子邮件，代码如下：

```
Transport.send(message);
```



7.2.2 Java Mail核心类简介

6. Store类

Store类定义了用于保存文件夹间层级关系的数据库，以及包含在文件夹之中的信息，该类也可以定义存取协议的类型，以便存取文件夹与信息。

在获取会话后，就可以使用用户名和密码或Authenticator类来连接Store类。与Transport类一样，首先要告诉Store类将使用什么协议：

使用POP3协议连接Store类，代码如下：

```
Store store = session.getStore("pop3");  
store.connect(host, username, password);
```

使用IMAP协议连接Store类，代码如下：



7.2.2 Java Mail核心类简介

```
Store store = session.getStore("imap");  
store.connect(host, username, password);
```

说明：如果使用**POP3**协议，只可以使用**INBOX**文件夹，但是使用**IMAP**协议，则可以使用其他的文件夹。在使用**Store**类读取完邮件信息后，需要及时关闭连接。关闭**Store**类的连接可以使用以下代码：

```
store.close();
```

7. Folder类

Folder类定义了获取（**fetch**）、备份（**copy**）、附加（**append**）及以删除（**delete**）信息等的方法。



7.2.2 Java Mail核心类简介

在连接**Store**类后，就可以打开并获取**Folder**类中的消息。打开并获取**Folder**类中的信息的代码如下：

```
Folder folder = store.getFolder("INBOX");  
folder.open(Folder.READ_ONLY);  
Message message[] = folder.getMessages();
```

在使用**Folder**类读取完邮件信息后，需要及时关闭对文件夹存储的连接。关闭**Folder**类的连接的语法格式如下：

```
folder.close(Boolean boolean);
```

boolean：用于指定是否通过清除已删除的消息来更新文件夹。



7.2.3 搭建Java Mail的开发环境

由于目前Java Mail还没有被加在标准的Java开发工具中，所以在使用前必须另外下载Java Mail API，以及Sun公司的JAF（JavaBeans Activation Framework），Java Mail的运行必须依赖于JAF的支持。

1. 下载并构建Java Mail API

Java Mail API是发送E-mail的核心API，它可以到网址“<http://java.sun.com/products/javamail/downloads/index.html>”中下载，目前最新版本的文件名为javamail-1_4.zip。下载后解压缩到硬盘上，并在系统的环境变量CLASSPATH中指定activation.jar文件的放置路径，例如，将mail.jar文件复制到“C:\JavaMail”文件夹中，可以在环境变量CLASSPATH中添加以下代码：

7.2.3 搭建Java Mail的开发环境

C:\JavaMail\mail.jar;

如果不想更改环境变量，也可以把mail.jar放到实例程序的WEB-INF/lib目录下。

2. 下载并构建JAF

目前Java Mail API的所有版本都需要JAF的支持。JAF为输入的任意数据块提供了支持，并能相应地对其进行处理。JAF可以到网址“<http://java.sun.com/products/javabeans/jaf/downloads/index.html>”中下载，当前最新版本的JAF文件名为jaf-1_1-fr.zip，下载后解压缩到硬盘上，并在系统的环境变量CLASSPATH中指定activation.jar文件的放置路径，例如，将activation.jar文件复制到“C:\JavaMail”文件夹中，可以在环境变量CLASSPATH中添加以下代码：

7.2.3 搭建Java Mail的开发环境

C:\JavaMail\activation.jar;

如果不想更改环境变量，也可以把activation.jar放到实例程序的WEB-INF/lib目录下。

2. 下载并构建JAF

目前Java Mail API的所有版本都需要JAF的支持。JAF为输入的任何数据块提供了支持，并能相应地对其进行处理。JAF可以到网址“<http://java.sun.com/products/javabeans/jaf/downloads/index.html>”中下载，当前最新版本的JAF文件名为jaf-1_1-fr.zip，下载后解压缩到硬盘上，并在系统的环境变量CLASSPATH中指定activation.jar文件的放置路径，例如，将activation.jar文件复制到“C:\JavaMail”文件夹中，可以在环境变量CLASSPATH中添加以下代码：



7.2.4 在JSP中应用Java Mail 组件发送E-mail

jspSmartUpload组件最常用的功能就是实现发送E-mail。本节将通过一个具体的实例介绍应用jspSmartUpload组件发送E-mail的方法。

【例7-2】 发送普通文本格式的E-mail



课件制作人：王国辉



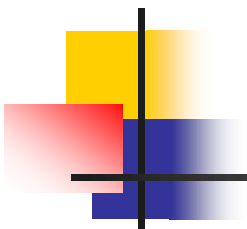
7.3 JSP动态图表

JFreeChart是一个Java开源项目，是一款优秀的Java图表生成插件，它提供了在Java Application、Servlet和JSP下生成各种图片格式的图表，包括柱形图、饼形图、线图、区域图、时序图和多轴图等。

- 7.3.1 JFreeChart的下载与使用 ✓
- 7.3.2 JFreeChart的核心类 ✓
- 7.3.3 利用JFreeChart生成动态图表 ✓



7.3.1 JFreeChart的下载与使用



在JFreeChart的官方网站（<http://www.jfree.org/jfreechart/index.html>）上可以下载到该插件，该插件有两个版本：

- （1）jfreechart-1.0.5.zip，该版本适用于Windows系统；
 - （2）jfreechart-1.0.5.tar.gz，该版本适用于UNIX/Linux系统。
- 下面以Windows系统为例，介绍JFreeChart组件的使用。解压缩jfreechart-1.0.9.zip后将得到一个名为jfreechart-1.0.9的文件夹，只需将lib子文件夹内的jfreechart-1.0.9.jar和jcommon-1.0.12.jar两个文件拷贝到Web应用程序的WEB-INF下的lib文件夹内，并且在该Web应用程序的web.xml文件中，</web-app>前面添加如下代码：

7.3.1 JFreeChart的下载与使用

```
<servlet>
    <servlet-name>DisplayChart</servlet-name>
    <servlet-class>org.jfree.chart.servlet.DisplayChart</servlet-
class>
</servlet>
<servlet-mapping>
    <servlet-name>DisplayChart</servlet-name>
    <url-pattern>/servlet/DisplayChart</url-pattern>
</servlet-mapping>
```

这样，就可以利用JFreeChart组件生成动态统计图表了。利用JFreeChart组件生成动态统计图表的基本步骤如下：

7.3.1 JFreeChart的下载与使用

- (1) 创建绘图数据集合;
- (2) 创建JFreeChart实例;
- (3) 自定义图表绘制属性, 该步可选;
- (4) 生成指定格式的图片, 并返回图片名称;
- (5) 组织图片浏览路径;
- (6) 通过HTML中的标记显示图片。





7.3.2 JFreeChart的核心类

在使用JFreeChart组件之前，首先应该了解该组件的核心类及其功能。JFreeChart核心类如下表所示。

方 法	说 明
JFreeChart	图表对象，生成任何类型的图表都要通过该对象，JFreeChart插件提供了一个工厂类ChartFactory，用来创建各种类型的图表对象
XXXDataset	数据集对象，用来保存绘制图表的数据，不同类型的图表对应着不同类型的数据集对象
XXXPlot	绘图区对象，如果需要自行定义绘图区的相关绘制属性，需要通过该对象进行设置
XXXAxis	坐标轴对象，用来定义坐标轴的绘制属性
XXXRenderer	图片渲染对象，用于渲染和显示图表
XXXURLGenerator	链接对象，用于生成Web图表中项目的鼠标单击链接
XXXToolTipGenerator	图表提示对象，用于生成图表提示信息，不同类型的图表对应着不同类型的图表提示对象



7.3.2 JFreeChart的核心类

上表中给出的各对象的关系如下：

JFreeChart中的图表对象用JFreeChart对象表示，图表对象由Title（标题或子标题）、Plot（图表的绘制结构）、BackGround（图表背景）、tooltip（图表提示条）等几个主要的对象组成。其中Plot对象又包括了Render（图表的绘制单元——绘图域）、Dataset（图表数据源）、domainAxis（x轴）、rangeAxis（y轴）等一系列对象组成，而Axis（轴）是由更细小的刻度、标签、间距、刻度单位等一系列对象组成。



7.3.3 利用JFreeChart生成动态图表

利用JFreeChart可以很方便的生成柱形图表，下面通过一个具体实例进行介绍。

【例7-3】 利用JFreeChart生成论坛版块人气指数排行的柱形图





7.4 JSP报表

在企业的信息系统中，报表一直占据比较重要的作用。在JSP中可以通过iText组件生成报表。下面将介绍如何使用iText组件生成PDF报表。

7.4.1 iText组件简介 ✓

7.4.2 iText组件的下载与配置 ✓

7.4.3 应用iText组件生成JSP报表 ✓





7.4.1 iText组件简介

iText是一个能够快速产生PDF文件的Java类库，是著名的开放源码站点sourceforge的一个项目。通过iText提供的Java类不仅可以生成包含文本、表格、图形等内容的只读文档，而且可以将XML、HTML文件转化为PDF文件。它的类库尤其与java Servlet有很好的结合。使用iText与PDF能够使用户正确地控制Servlet的输出。



7.4.2 iText组件的下载与配置

iText组件可以到<http://www.lowagie.com/iText/download.html>网站下载。在IE地址栏中输入上面的URL地址后，将进入到如下图所示的下载界面。

在该图中单击iText-2.0.7.jar下载最新版本的iText组件，其中，iText-2.0.7.jar适用Windows操作系统，而iText-2.0.7.tar.gz适用于Linux操作系统。

Address (Q) <http://www.lowagie.com/iText/download.html>

Current release
The release on this site is 2.0.7.

Download the library:

Source code	iText-src-2.0.7.tar.gz or iText-src-2.0.7.zip
Compiled code	iText-2.0.7.jar
Documentation	iText-docs-2.0.7.tar.gz

Please consult the [history overview](#) for more info.
Another option to get the source code is to download it from the [SVN repository](#). There, you'll always find the most recent iText version, and even some branches to build your own jar for the older JDK 1.3 or newer JDKs such as JDK 1.5.

Getting older versions of iText

Java PDF Library
Fully featured
100% Java API.
Sign, Encrypt,
Render. Free Trial!
big.faceless.org

Dynamic PDF Form-Filling
Append, stamp,
split, secure,
merge form-fill PDF

完毕 本anet



7.4.2 iText组件的下载与配置

下载iText-2.0.7.jar文件后，需要把itext-2.0.7.jar包放入项目目录下的WEB-INF/lib路径中，这样在程序中就可以使用iText类库了。如果生成的PDF文件中需要出现中文、日文、韩文字符，则需要访问<http://itext.sourceforge.net/downloads/iTextAsian.jar>下载iTextAsian.jar包。当然，如果想真正了解iText组件，阅读iText文档显得非常重要，读者在下载类库的同时，也可以下载类库文档。



课件制作人：王国辉

7.4.3 应用iText组件生成JSP 报表

1. 建立com.lowagie.text.Document对象的实例

建立com.lowagie.text.Document对象的实例时，可以通过以下3个构造方法实现：

```
public Document();  
public Document(Rectangle pageSize); //定义页面的大小  
public Document(Rectangle pageSize,int marginLeft,int  
marginRight,int marginTop,int marginBottom); /*定义页面的大小，  
参数marginLeft、marginRight、marginTop、marginBottom分别为左、  
右、上、下的页边距*/
```

其中，通过Rectangle类对象的参数可以设定页面大小、面背景色，以及页面横向/纵向等属性。

7.4.3 应用iText组件生成JSP报表

iText组件定义了A0-A10、AL、LETTER、HALFLETTER、_11x17、LEDGER、NOTE、B0-B5、ARCH_A-ARCH_E、FLSA和FLSE等纸张类型，也可以制定纸张大小来自定义，程序代码如下：

```
Rectangle pageSize = new Rectangle(144,720);
```

在iText组件中，可以通过下面的代码实现将PDF文档设定成A4页面大小，当然，也通过Rectangle类中的rotate()方法可以将页面设置成横向。程序代码如下：

```
Rectangle rectPageSize = new Rectangle(PageSize.A4); //定义A4页面大小  
rectPageSize = rectPageSize.rotate(); //加上这句可以实现A4页面的横置  
Document doc = new Document(rectPageSize,50,50,50,50); //其余4个参数  
设置了页面的4个边距
```

7.4.3 应用iText组件生成JSP 报表

2. 设定文档属性

在文档打开之前，可以设定文档的标题、主题、作者、关键字、装订方式、创建者、生产者、创建日期等属性，调用的方法分别是：

```
public boolean addTitle(String title)
public boolean addSubject(String subject)
public boolean addKeywords(String keywords)
public boolean addAuthor(String author)
public boolean addCreator(String creator)
public boolean addProducer()
public boolean addCreationDate()
public boolean addHeader(String name, String content)
```

7.4.3 应用iText组件生成JSP 报表

其中方法addHeader()对于PDF文档无效，addHeader()方法仅对HTML文档有效，用于添加文档的头信息。

3. 创建书写器（Writer）对象

文档（document）对象建立好之后，还需要建立一个或多个书写器与对象相关联，通过书写器可以将具体的文档存盘成需要的格式，例如，
om.lowagie.text.PDF.PDFWriter可以将文档存成PDF格式，
而com.lowagie.text.html.HTMLWriter可以将文档存成HTML格式。

【例7-5】 书写器对象示例

7.4.3 应用iText组件生成JSP报表

4. 进行中文处理

iText组件本身不支持中文，为了解决中文输出的问题，需要多下载一个iTextAsian.jar组件。下载后放入项目目录下的WEB-INF/lib路径中。使用这个中文包无非是实例化一个字体类，把字体类应用到相应的文字中，从而可以正常显示中文。可以通过以下代码解决中文输出问题：

```
BaseFont bfChinese = BaseFont.createFont("STSong-Light", "UniGB-UCS2-H", BaseFont.NOT_EMBEDDED);  
//用中文的基础字体实例化了一个字体类  
Font FontChinese = new Font(bfChinese, 12, Font.NORMAL);  
Paragraph par = new Paragraph("简单快乐",FontChinese);//将字体类用到了一个段落中  
document.add(par);                                //将段落添加到了文档中
```

7.4.3 应用iText组件生成JSP 报表

在上面的代码中，STSong-Light定义了使用的中文字体，UniGB-UCS2-H定义文字的编码标准和样式，GB代表编码方式为gb2312，H是代表横排字，V代表竖排字。

【例7-6】 中文处理示例

在JSP页面编写代码实现输出PDF的文档，文档内容为“宝剑峰从磨砺出 梅花香自苦寒来”。

5. 创建表格

iText组件中创建表格的类包括com.lowagie.text.Table和com.lowagie.text.PDF.PDFPTable两种。对于比较简单的表格可以使用com.lowagie.text.Table类创建，但是如果创建复杂的表格，就需要用到com.lowagie.text.PDF.PDFPTable类。

7.4.3 应用iText组件生成JSP 报表

(1) com.lowagie.text.Table类

com.lowagie.text.Table类的构造函数有3个:

Table(int columns)

Table(int columns, int rows)

Table(Properties attributes)

参数columns, rows, attributes分别为表格的列数、行数、表格属性。创建表格时必须指定表格的列数,而对于行数可以不用指定。

创建表格之后,还可以设定表格的属性,如边框宽度、边框颜色、间距(padding space 即单元格之间的间距)大小等属性。

7.4.3 应用iText组件生成JSP 报表

【例7-7】 表格示例1

使用com.lowagie.text.Table类来生成2行3列的带表头的表格。

(2) com.lowagie.text.PdfPTable类

iText组件中一个文档（Document）中可以有很多个表格，一个表格可以有很多个单元格，一个单元格里面可以放很多个段落，一个段落里面可以放一些文字。但是，读者必须明确以下两点内容：

- ① 在iText组件中没有行的概念，一个表格里面直接放单元格，如果一个5列的表格中放进10个单元格，那么就是两行的表格；
- ② 如果一个5列的表格放入4个最基本的没有任何跨列设置的单元格，那么这个表格根本添加不到文档中，而且不会有任何提示。

7.4.3 应用iText组件生成JSP 报表

【例7-8】 表格示例2

使用com.lowagie.text.PdfPTable类来生成2行5列的表格。

6. 图像处理

iText组件中处理图像的类为com.lowagie.text.Image，目前iText组件支持的图像格式有GIF，Jpeg，PNG，wmf等格式，对于不同的图像格式，iText组件用同样的构造函数自动识别图像格式，通过下面的代码可以分别获得gif，jpg，png图像的实例：

```
Image gif = Image.getInstance("face1.gif");  
Image jpeg = Image.getInstance("bookCover.jpg");  
Image png = Image.getInstance("ico01.png");
```

7.4.3 应用iText组件生成JSP 报表

图像的位置主要是指图像在文档中的对齐方式、图像和文本的位置关系。iText组件中通过方法setAlignment(int alignment)设置图像的位置，当参数alignment为Image.RIGHT、Image.MIDDLE、Image.LEFT分别指右对齐、居中、左对齐；当参数alignment为Image.TEXTWRAP、Image.UNDERLYING分别指文字绕图形显示、图形作为文字的背景显示，也可以使这两种参数结合使用以达到预期的效果，如setAlignment(Image.RIGHT|Image.TEXTWRAP)显示的效果为图像右对齐，文字围绕图像显示。如果图像在文档中不按原尺寸显示，可以通过下面的方法进行设定：

7.4.3 应用iText组件生成JSP 报表

```
public void scaleAbsolute(int newWidth, int newHeight)
public void scalePercent(int percent)
public void scalePercent(int percentX, int percentY)
```

方法scaleAbsolute(int newWidth, int newHeight)直接设定显示尺寸；方法scalePercent(int percent)设定显示比例，如scalePercent(50)表示显示的大小为原尺寸的50%；而方法scalePercent(int percentX, int percentY)则表示图像高宽的显示比例。

如果图像需要旋转一定角度之后在文档中显示，可以通过方法setRotation(double r)设定，参数r为弧度，如果旋转角度为 30° ，则参数 $r = \text{Math.PI} / 6$ 。

7.4.3 应用iText组件生成JSP 报表

【例7-9】 图像处理示例

在JSP页面编写代码实现输出PDF的文档，文档内容为一个两行一列的表格，其中第一行的内容为图片，第二行的内容为居中显示的文字harvest。



课件制作人：王国辉