Struts 结构的 JAVA Web 开发中实现文件上传与下载

To Upload And Download the Files Under the Struts Frame

汪健峰 （解放军后勤指挥学院 北京 100858）

摘要：Struts 结构的 JAVA Web 开发中，实现文件上传有两种方式，即上传到服务器和上传到数据库。实现文件下载也有两种方式，即从服务器上直按下载和从数据库中下载。

关键词：Struts JAVA Web 上传 下载

1 前言

如今，随着基于 B/S（浏览器/服务器）模式的 Web 系统的广泛使用，文件的上传与下载成为系统开发中经常用到功能。比如，在基于 web 的 email 中，进行收发附件的操作；在管理信息系统中，存储和显示注册人员的照片等信息；在技术支持站点，发送和接收错误日志和缺陷报告文档；在 web 应用中，通过友好的 web 界面在用户间共享文件等，这些不同程序地都要用到文件的上传与下载功能。本文将对如何在 Struts 结构的 JAVA WEB 编程中实现上下载进行讨论，并给出具体解决方法。Struts 是基于 MVC（模型－视图－控制）的 Web 应用框架，通过它可以实现模型、显示和控制层的分离，将程序员从大量繁杂的工作中解放出来，而把精力主要集中在如何解决实际业务问题上。目前 Struts 已经成为众多软件架构师和 Web 程序员的首选。这里我们使用 Struts 提供的 upload 类，其中 org. apache. struts. upload. FormFile 是 Struts 中用来实现文件上传下载的核心。文章中所有实例均已在 JBuiildX + Oracle9i + Tomcat4. 1 环境下调试通过。

2 文件上载的方式

在 TCP/IP 中最早出现的文件上下载机制是 FTP，它是将文件由客户端发送到服务器的标准机制。它很可靠，能够考虑到跨平台的文本和二进制格式文件。但在 JSP 编程中不能使用 FTP 方法来上载文件，这是由 JSP 的运行机制所决定的。我们知道：JSP（JavaServer Pages）是由 Sun MicroSystems 公司倡导、许多公司参与，一起建立的一种动态网页技术标准。在传统的网页 HTML 文件中加入 JAVA 程序片段和 JSP 标记就构成了 JSP 网页。web 服务器在遇到访问 JSP 网页请求时，首先执行其中的程序片段，然后将执行结果以 HTML 形式返回客户。这种运行机制就要求客户与服务器的联系需要使用 HTTP 协议而不能是 FTP 协议。

在 HTTP 协议中上下载文件主要运用 RFC1867 方式。RFC1867 在作为标准发布之前首先被 Netscape 的 Navigator 2.0 中采用。随后被 Microsoft 的 IE3. 0 的附件和 IE3. 03 的一部分使用。它是一种简单实用的方法。只在表单字段中定义一个 file 类型的 input：

```
<input type = " file" >
```

并且在表单本身中加入了不同的编码方案，它不再使用典型的：

```
<form action = " test. jsp" method = " post" >
```

而是使用

```
<form action = " test. jsp" method = " post" enctype = " multipart/form - data" >
```

这种编码方案在传送大量数据时比缺省的" application/x - url - encoded" 表单编码方案要效率高得

多。因为 URL 编码只有很有限的字符集。当使用任何超出字符集的字符时，必须用"%nn"代替（这里的 nn 表示相应的两个十进制数）。这样即使是普通的空格字符也要用"%20"代替。那么，通过 URL 编码方式上载的文件将会是原来的 2－3 倍大。而使用 RFC1867 编码方式则只是在传送数据的周围加上很简单的头部来标识文件内容。

3 文件上载的实现

我们知道，将文件上传到服务器可以有两种实现。第一种实现是将上载的内容以文件的形式存储在到服务器的硬盘。至于文件存放的物理位置，可以通过编程人为设置，也可以将文件存放的有关信息放在数据库中，需要时从数据库中取出；第二种实现是将文件直接上传到数据库服务器的数据库中。以 Oracle 数据库为例，这就要求在数据库中相应地设置一个大字段（Blob 字段），然后通过编写程序实现将客户端的文件存储到服务器端数据库的大字段中。这两种方法在工程上各有应用，应根据不同需求进行选择。以下的实现都是利用了 Struts 的 upload 类，因此在程序中要导入 org. apache. struts. upload. FormFile。

3.1 上载的内容以文件的形式存储在到服务器的硬盘

下面是实例：

3.1.1 上传的 JSP 界面

如前面所述，在 JSP 中文件上载使用的是 RFC1867 标准。在其中也规定了上载文件所使用的 HTML 表单的格式。此表单的格式被 Netscape 3 和 Microsoft IE3.02 以上的浏览器所支持。上载文件的表单发布具备如下的格式：

（1）单所使用的方法必须是 post；

（2）在 action 属性中的执行程序必须能处理由客户端返回的数据；

（3）表单必须使用 MIME 类型：multipart/form－data；

（4）表单必须至少包含一个 <input type="file" >元素。与 RFC1867 兼容的浏览器将会在客户端界面上显示一个文本框和一个"浏览"按钮，以方便客户端选择上载的文件。

下面是使用此类型表单的 JSP 页面的部分代码：

```
<html:form action="UploadAction.do" enctype
="multipart/form－data" >
    <html:file property="theFile" />
    <html:submit/>
    </html:form>
```

3.1.2 Form Bean

与 JSP 页面对应，FormBean 中定义了一个属性：theFile，同时定义了它的 set 与 get 方法。

3.1.3 Action

主要实现业务逻辑。首先通过 FormFile 类取得上传的文件，然后将该文件读入到服务器的指定路径下。部分的代码如下：

```
FormFile file = theForm.getTheFile();//取得上传的文件
String contentType = file.getContentType();
String size = (file.getFileSize() + " bytes");//文件大小
String fileName = file.getFileName();//文件名
try {
    InputStream stream = file.getInputStream();//把文件读入
    String filePath = request.getRealPath("/");//取当前系统路径
    ByteArrayOutputStream baos = new ByteArrayOutputStream();
    OutputStream bos = new FileOutputStream(filePath +"/upload" + "/" +
        file.getFileName());
//建立一个上传文件的输出流，将上传文件存入 web 应用的 upload 目录下。upload 目录是提前建好的。
int bytesRead = 0; byte[]buffer = new byte[8192]; while ((bytesRead = stream.read(buffer, 0, 8192)) != -1) { bos.write(buffer, 0, bytesRead); }//将文件写入服务器 接下来，配置好 Struts 的配置文件（struts－congfig.xml），并按照 JAVA Web 应用的结构布署好各个文件（此应用中 jsp 页面应在 web 应用的根目录下），设定好应用服务器（如 Tomcat、Weblogic 等）的配置文件（如 Tomcat 的 sever.xml 文件），进行上传。最后，就可以去服务器硬盘（此处是 web 应用下的 upload 文件夹）查看上传的文件
```

了。

3.2 将文件存储在服务器的数据库中

这个例子中，用户先选一个图片，然后按 submit 就可以将图片存入 Oracle 数据库中。如前所述，要将文件存入数据库中，就要在数据库中设立相应的大字段（Oracle 中是 Blob 字段）。首先要在 Oracle 的 ORADB 库中建立一个 test 表，这个表有两个字段：一个是 varchar2 型的大小为 20 的"name"字段，一个是 Blob 型的"pic"字段。此外，要特别注意的是，必须要将 Oracle 的驱动程序包 classes12. jar 拷贝到. . \WEB - INF\lib 目录下，这样才能实现将文件数据存入数据库的 Blob 字段中。下面是实例：

3.2.1 JSP 页面

提供一个文本框和一个"浏览"按钮，以方便客户选择上载的文件，同时还提供一个文本框用来提交文件名。部分代码如下：

```
<html:form action="UploadAction.do" enctype="multipart/form-data">
文件名：<html:text property="name" /><br>
文件：<html:file property="file" />
<html:submit/>
</html:form>
```

3.2.2 相对应的 FormBean

与 JSP 页面对应，FormBean 中定义了两个属性：FormFile 类型的 theFile 和 String 类型的 name，同时定义了它们的 set 与 get 方法。

3.2.3 相对应的 Action

首先通过 FormFile 类取得上传的文件，然后将该文件读入到 IP 地址是 21. 156. 162. 8 的 Oracle 服务器的 ORADB 数据库中，具体的是方案名为"TESTOA"的"test"表中，这个方案的密码为"test"。部分的代码如下：

```
FormFile file = multiForm.getFile();
String name = multiForm.getName();
try {
Class.forName("oracle.jdbc.driver.OracleDriver");
String url = "jdbc:oracle:thin:@21.156.162.8:1521:ORADB";
Connection con = DriverManager.getConnection(url,"TESTOA","TEST");
String sql = "insert into test (name,pic) values (?,?)";
PreparedStatement ps = con.prepareStatement(sql);
ps.setString(1, name);
//加入图片到数据库
ps.setBinaryStream(2, file.getInputStream(),file.getFileSize());
ps.executeUpdate();
ps.close();
con.close();
}catch (Exception ex) {
System.out.println("++++++++++遇到异常！++++++++++++++++++++");}
```

按下来就是配置好 Struts 和 Tomcat 的配置文件，进行上传测试。需要补充说明的是，上述的实例是应用 Action 中的代码直接通过 JDBC 和 ORACLE 驱动程序实现了与数据库的相连，这样的做法会给数据升迁带来困难。试想如果我们要对数据库重新部署，或者更换数据库的类型，那就要重写所有的 Action 中的代码，在一个大型系统中会有成百上千个部分需要重写，这个工作量是巨大的。这时可以利用 Hibernate 这样一个开源的技术，将数据库映射为类，只在 Tomcat 中配置好有关数据库的连接。这样，我们就可以专心关注于业务逻辑的实现，而不考虑数据库升迁所带来的种种问题。关于这方面，可以参阅 Hibernate 的相关内容。

4 文件下载的实现

文件下载也有两种方式。仍然要用到 FormFile 类，在程序中要导入 org. apache. struts. upload. FormFile。同时可以重用上传的 FormBean。

4.1 直接从服务器下载文件

这种方式是在服务器端将文件读到流中，然后循环取出流中的数据。实现下载的 Action 中部分代码为：

```
String fileName = "karl.doc".toString();
//将文件读到流中
InputStream inStream = new FileInputStream("d:/
```

```
wjf. doc" );
    //设置输出的格式
    response. reset( );
    response. setContentType( " bin" );
    response. addHeader ( " Content - Disposition" ," attachment; filename = "" + fileName + """ );
    //循环取出流中的数据
    byte[ ] b = new byte[100];
    int len;
    while( ( len = inStream. read( b) ) >0)
    response. getOutputStream( ). write( b,0,len) ;
    inStream. close( ) ;
```

4.2 从服务器的数据库中下载文件

这种方式就是把输入流从数据库里读出来，然后转存为文件。Action 中部分代码为：

```
    int bytesum =0;
    int byteread =0;
    try{
      Class. forName( " oracle. jdbc. driver. OracleDriver" ) ;
      Connection conn = DriverManager. getConnection ( " jdbc: oracle: thin: @ 21. 156. 162. 8: 1521: ORADB" ," TESTOA" ," TEST" ) ;
      Statement stmt = conn. createStatement( ) ;
      String sql = " SELECT name, pic FROM test where name =" + name + "" ;
    ResultSet rs = stmt. executeQuery( sql) ;
    if( rs. next( ) ) {
      //读到流中
      InputStream inStream = rs. getBinaryStream ( " pic" ) ;
      String fileName = " wjf. doc" . toString( ) ;
      //设置输出的格式
      response. reset( ) ;
      response. setContentType( " bin" ) ;
      response. addHeader ( " Content - Disposition" ," attachment; filename = "" + fileName + """ ) ;
      //循环取出流中的数据
    byte[ ] b = new byte[100];
    int len;
    while( ( len = inStream. read( b) ) >0)
    response. getOutputStream( ). write( b,0,len) ;
    inStream. close( ) ;
    }
```

至此，就可以完成下载。如果我们配置好各种配置文件，进行测试，就可以在相应的位置查看下载的文件了。

5 总结

在 Struts 结构下的 JAVA WEB 开发中实现文件上传与下载功能，主要使用 Struts 提供的 upload 类，其中 FormFile 是核心。文件可以上载到服务器的硬盘中，也可以上传到数据库中，相应地，下载也有两种实现方式，应根据具体需求和环境进行选择。

参考文献

1 飞思科技产品研发中心，JSP 应用开发详解，电子工业出版社，2004.7。
2 孙卫琴，精通 Struts：基于 MVC 的 JAVA Web 设计与开发，电子工业出版社，2004.11。