



# Java程序设计

## 第06-02讲 文件上传和下载

Java课程组



# 本讲教学目标



- 熟悉各种表单域
- 掌握Java中实现文件上传
- 掌握Java中实现文件下载





## 本讲内容

- 表单元素
- 文件上传
- 文件下载



# 常用的表单标签



- 通常使用<input> 标签向服务器端提交数据
- <input> 标签常用类型有9种类型
  - text: 文本域
  - password: 密码文本域
  - radio: 单选按钮
  - checkbox: 多选按钮
  - file: 文件上传
  - hidden: 隐藏域
  - reset: 重置按钮
  - submit: 提交按钮
  - button: 按钮

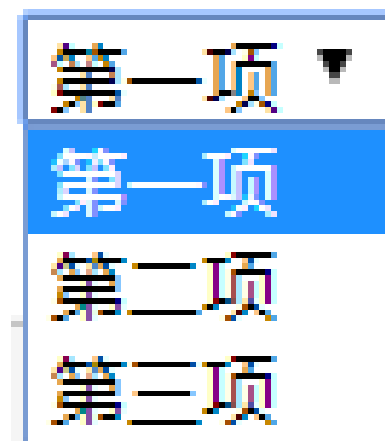
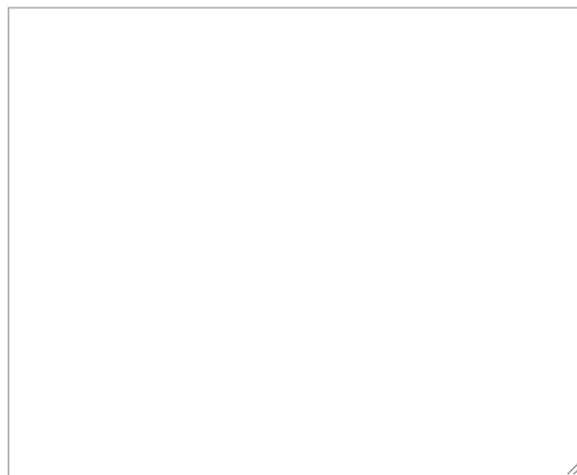


# 常用表单标签(续)



- 其他常用表单标签

- `<textarea name= "" rows= "3" cols= "3" >`多行文本  
`</textarea>`
- `<select name="i_select">`
  - `<option value="1" >`第一项`</option>`
  - `<option value="2" >`第二项`</option>`
  - `<option value="3" >`第三项`</option>``</select>`



# 值的提交与获取



- 观察实验中示例，通常表单域的值都是value值，在服务器端使用`request.getParameter("name")`取得
- 几个较特殊的标签
  - `<textarea>`这里是该标签的值`</textarea>`
  - `<select name="i_select">`
    - `<option value="值">第一项</option>`
    - `</select>`
  - `<input type="checkbox" />`
    - 如果没有value，默认选中为"on"，否则为null
    - 如果有value值，选中的值为其value值
    - 使用`request.getParameterValues()`方法得到值的字符串数组
  - `<input type="radio">`
    - 一组radio的名字相同，value要设置且不同
  - 文件提交`<input type="file" />`





## 本讲内容

- 表单元素
- 文件上传
- 文件下载



- 表单中文件输入类型允许将客户端计算机内的文件上传至服务器端
  - `<form action="..." enctype="multipart/form-data">`
  - `<input type="file" name="upfile" />`
  - `<input type="submit" />`
  - `</form>`
- 文件输入域的属性
  - accept: 设定用户能够上传文件之MIME类型
  - size: 文件输入字段的大小
  - name: 文件输入字段的名称
  - value: 默认的文件名称
  - maxlength: 文件名称最大长度限制





# 文件上传的问题



- 文本类型数据(如: Text、Select等)传送数据到服务器端时, 使用的编码方式为:
  - `enctype="application/x-www-form-urlencoded"`
- 若要传送文件至服务器端, 必须使用
  - `enctype="multipart/form-data"`
- 由于在传送数据时所使用的编码方式不同, 因此在接收客户端传来的文件数据时, 不能直接使用`request.getParameter()`方法



# request.getInputStream



- 使用request.getInputStream方法可以获得客户端浏览器POST方式传来的数据流，但得到的数据流还包括一些其他信息
- `getServletContext().getRealPath()`方法取得服务器端某文件的实际路径



# jspSmartUpload



- 是一套免费的组件，简单好用，但官方网站已经关闭，且没有源代码
- 示例
- 步骤
  - 创建SmartUpload对象
  - 初始化SmartUpload对象
  - 根据Form的内容上传
  - 保存文件到服务器端



# jspSmartUpload示例



```
protected void doPost(HttpServletRequest request,
    HttpServletResponse response) throws ServletException, IOException {
    SmartUpload smartUpload = new SmartUpload();
    // 初始化
    ServletConfig config = this.getServletConfig();
    smartUpload.initialize(config, request, response);
    try {
        // 上传文件
        smartUpload.upload();
        // 得到上传的文件对象
        File smartFile = smartUpload.getFiles().getFile(0);
        // 保存文件
        smartFile.saveAs("C:/\" + smartFile.getFileName(),
            smartUpload.SAVE_PHYSICAL); // 保存文件到C盘根目录
    } catch (SmartUploadException e) {
        e.printStackTrace();
    }
    String msg = "Upload Success!";
    request.setAttribute("msg", msg);
    RequestDispatcher rd = request.getRequestDispatcher("/uploadForm.jsp");
    rd.forward(request, response);
}
```



# jspSmartUpload主要功能



- 使用setMaxFileSize()限制上传文件大小
- getFiles()得到多个上传文件
- setAllowedFilesList来限制上传文件类型
- 还可以进行文件下载的操作



# O'reilly cos组件



- O'reilly公司提供的cos组件，可免费下载且有源代码，会定期更新功能
  - 下载地址：  
<http://www.servlets.com/cos/index.html>
- 示例
  - 创建MultipartRequest对象
    - 在构造方法中传入request、文件存放路径、文件名的编码方式
  - 通过MultipartRequest.getFileNames()得到上传文件的反馈信息



# Apache commons-fileupload



- commons-fileupload是目前JavaWeb应用最广泛的上传组件
- commons-fileupload组件需要有commons-io组件的支持
- 示例
- 步骤
  - 创建DiskFileItemFactory
  - 使用factory创建一个ServletFileUpload
  - 使用parseRequest方法解析request得到FileItem列表
  - 用FileItem的write方法写到一个目标文件中(自己起文件名)



# fileupload组件的更多用法



- 查看commons-fileupload-1.3.3-bin.zip中的帮助文档

OVERVIEW PACKAGE CLASS USE TREE DEPRECATED INDEX HELP

PREV NEXT FRAMES NO FRAMES ALL CLASSES

A B C D F G H I L M N O P R S T U W

**A**

**addHeader(String, String)** - Method in class org.apache.commons.fileupload.util.FileItemHeadersImpl  
Method to add header values to this instance.

**arrayEquals(byte[], byte[], int)** - Static method in class org.apache.commons.fileupload.MultipartStream  
Compares count first bytes in the arrays a and b.

**asString(InputStream)** - Static method in class org.apache.commons.fileupload.util.Streams  
This convenience method allows to read a FileItemStream's content into a string.

**asString(InputStream, String)** - Static method in class org.apache.commons.fileupload.util.Streams  
This convenience method allows to read a FileItemStream's content into a string, using the given character encoding.

**ATTACHMENT** - Static variable in class org.apache.commons.fileupload.FileUploadBase  
Content-disposition value for file attachment.

**available()** - Method in class org.apache.commons.fileupload.MultipartStream.ItemInputStream  
Returns the number of bytes, which are currently available, without blocking.

**B**

**BOUNDARY\_PREFIX** - Static variable in class org.apache.commons.fileupload.MultipartStream  
A byte sequence that precedes a boundary (CRLF—).

**C**

**checkFileName(String)** - Static method in class org.apache.commons.fileupload.util.Streams  
Checks, whether the given file name is valid in the sense, that it doesn't contain any NUL characters.

**close()** - Method in class org.apache.commons.fileupload.MultipartStream.ItemInputStream  
Closes the input stream.

**close(boolean)** - Method in class org.apache.commons.fileupload.MultipartStream.ItemInputStream  
Closes the input stream.

**close()** - Method in interface org.apache.commons.fileupload.util.Closeable





# 文件上传思考



- 多文件上传
- 服务器端的路径及名称要考虑
- 限制上传文件大小
- 文件名中文问题
- 表单里同时有普通域也有文件域
- 上传进度条





## 本讲内容

- 表单提交
- 文件上传
- 文件下载





- `HttpServletResponse`对象实现文件下载
  - 确定下载文件的地址（路径）
  - `response`设置响应的内容类型
  - `response`设置响应头信息
  - 读取文件
  - 通过`response`以文件流的形式响应给客户端



# 文件下载示例



```
//获得请求文件名
```

```
String filename = request.getParameter("filename");
```

```
System.out.println(filename);
```

```
//设置文件MIME类型
```

```
response.setContentType(getServletContext().getMimeType(filename));
```

```
//设置Content-Disposition
```

```
response.setHeader("Content-Disposition", "attachment;filename="+filename);
```

```
//读取目标文件，通过response将目标文件写到客户端
```

```
//获取目标文件的绝对路径
```

```
String fullFileName = getServletContext().getRealPath("/download/" + filename);
```

```
//System.out.println(fullFileName);
```

```
//读取文件
```

```
InputStream in = new FileInputStream(fullFileName);
```

```
OutputStream out = response.getOutputStream();
```

```
//写文件
```

```
int b;
```

```
while((b=in.read())!= -1)
```

```
{
```

```
    out.write(b);
```

```
}
```

```
in.close();
```

```
out.close();
```

```
}
```



# 小结



- 复习常用的表单域
- 文件上传
  - O' reilly cos插件
  - Apache common-fileupload 插件
- 文件下载
  - Response以流的方式将文件响应给客户端





# 实验



- 实验
  - 文件上传
    - 头像上传
    - 多文件上传
    - 文本域、文件域混合提交
  - 文件下载
    - 页面提交文件名下载指定文件





# 本讲结束

- 谢谢大家

