1. 文件上传的回顾

1.1 文件上传的必要前提

1. form 表单的 enctype 取值必须是： `multipart/form-data` (默认值是: `application/x-www-form-urlencoded`)； enctype是表单请求正文的类型
2. method 属性取值必须是 `Post`
3. 提供一个文件选择域 选择文件 未选择任何文件

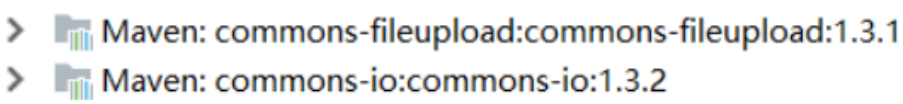
1.2 文件上传的原理分析

当 form 表单的 enctype 取值不是默认值后， `request.getParameter()` 将失效。

enctype="application/x-www-form-urlencoded" 时，form 表单的正文内容是：key=value&key=value&key=value ， 当 form 表单的 enctype 取值为 Mutilpart/form-data 时，请求正文内容就变成： 每一部分都是 MIME 类型描述的正文

1.3 借助第三方组件实现文件上传

使用 Commons-fileupload 组件实现文件上传，需要导入该组件相应的支撑 jar 包： Commons-fileupload 和commons-io。commons-io 不属于文件上传组件的开发 jar 文件，但Commons-fileupload 组件从 1.1 版本开始，它工作时需要 commons-io 包的支持。



2. springmvc 传统方式的文件上传

2.1 说明

传统方式的文件上传，指的是我们上传的文件和访问的应用存在于同一台服务器上。并且上传完成之后，浏览器可能跳转。

2.2 实现步骤

2.2.1 第一步：拷贝文件上传的 jar 包到工程的 lib 目录

```
<dependency>
    <groupId>org.apache.commons</groupId>
    <artifactId>commons-io</artifactId>
    <version>1.3.2</version>
</dependency>

<dependency>
    <groupId>commons-fileupload</groupId>
  <artifactId>commons-fileupload</artifactId>
     <version>1.3.1</version>
</dependency>
```

2.2.2 第二步：编写 jsp 页面

```
<form action="/fileUpload1" method="post"
enctype="multipart/form-data">
    名称：<input type="text" name="picname"/><br/>
    图片：<input type="file" name="uploadFile"/><br/>
    <input type="submit" value="上传"/>
</form>
```

2.2.3 第三步：编写控制器

```
/**
文件上传控制器
*/
@Controller()
public class FileUploadController {
    /**
    文件上传
    */
    @RequestMapping("/fileUpload1")
    public String fileUpload(String picname, MultipartFile
uploadFile, HttpServletRequest request) throws Exception {
        String fileName = "";
        //1.获取原始的文件名字
        String uploadFileName =
uploadFile.getOriginalFilename();
        //2.截取文件扩展名
        String extendName =
uploadFileName.substring(uploadFileName.lastIndexOf(".") +
1, uploadFileName.length());

        //3.把文件加上随机数，防止文件重复
```

```
        String uuid =
UUID.randomUUID().toString().replace("-",
"").toUpperCase();
        //4.判断是否输入了文件名
        if (!StringUtils.isEmpty(picname)) {
            fileName = uuid + "_" + picname + "." +
extendName;
        } else {
            fileName = uuid + "_" + uploadFileName;
        }
        System.out.println("要上传的文件名是：" + fileName);
        //2.获取上传的真实的服  务器路径
        String basePath =
request.getServletContext().getRealPath("/uploads");
        //3.解决同一文件夹中文件过多问题
        String datePath = new SimpleDateFormat("yyyy-MM-
dd").format(new Date());
        //4.判断路径是否存在
        File file = new File(basePath + "/" + datePath);
        if (!file.exists()) {
            file.mkdirs();
        }
        //5.使用 MulitpartFile 接口中方法，把上传的文件写到指定
位置
        File f = new File(file, fileName);
        uploadFile.transferTo(f);
        System.out.println("文件上传成功:"
+f.getAbsolutePath());
        return "success";
    }
}
```

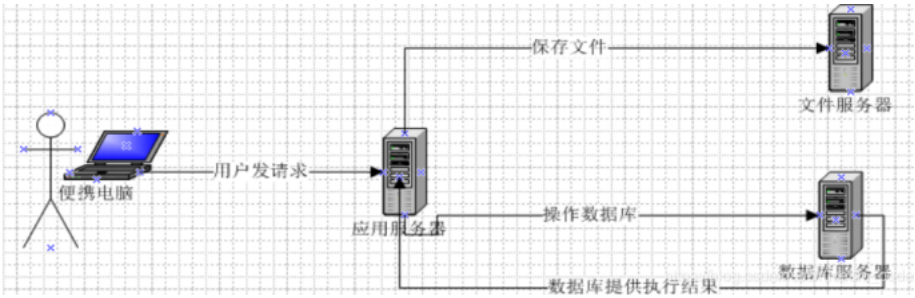
2.2.4 第四步：配置文件解析器

```
<!-- 配置文件上传解析器 -->
<!-- id 的值是固定的-->
<bean id="multipartResolver"
class="org.springframework.web.multipart.commons.CommonsMu
ltipartResolver">
    <!-- 设置上传文件的最大尺寸为 5MB -->
    <property name="maxUploadSize">
        <value>5242880</value>
    </property>
</bean>
```

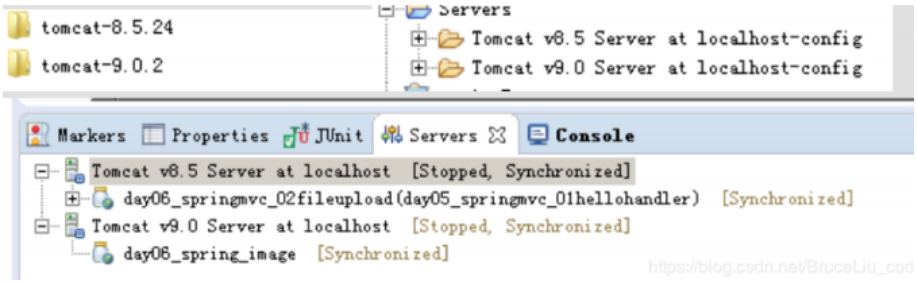
3. springmvc 跨服务器方式的文件上传

3.1 分服务器的目的

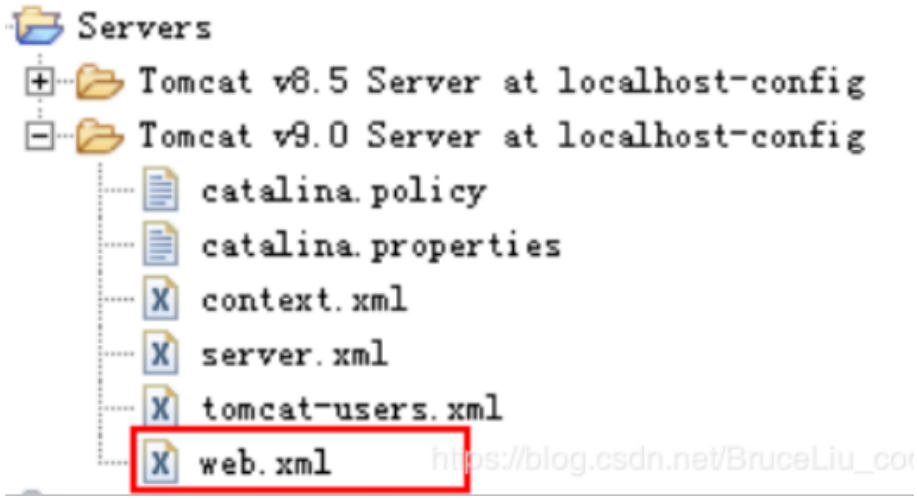
在实际开发中，我们会有很多处理不同功能的服务器。例如： 应用服务器：负责部署我们的应用 数据库服务器：运行我们的数据库 缓存和消息服务器：负责处理大并发访问的缓存和消息 文件服务器：负责存储用户上传文件的服务器。(注意：此处说的不是服务器集群） 分服务器处理的目的是让服务器各司其职，从而提高我们项目的运行效率。



3.2 准备两个 tomcat 服务器，并创建一个用于存放图片的 web 工程



在文件服务器的tomcat（9.0） 配置中加入，允许读写操作。文件位置：



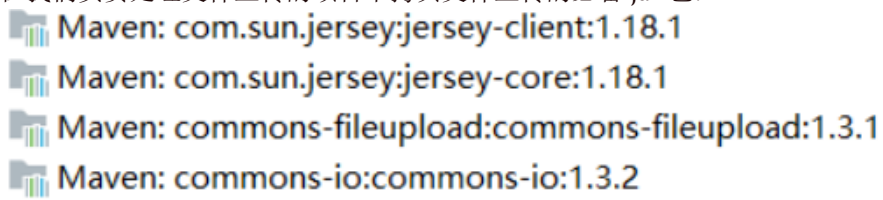
加入内容：

```
<servlet>
    <servlet-name>default</servlet-name>
    <servlet-class>org.apache.catalina.servlets.DefaultServlet</servlet-class>
    <init-param>
        <param-name>debug</param-name>
        <param-value>0</param-value>
    </init-param>
    <init-param>
        <param-name>readonly</param-name>
        <param-value>false</param-value>
    </init-param>
    <init-param>
        <param-name>listings</param-name>
        <param-value>false</param-value>
    </init-param>
    <load-on-startup>1</load-on-startup>
</servlet>
```

加入此行的含义是：接收文件的目标服务器可以支持写入操作。

3.3 拷贝依赖

在我们负责处理文件上传的项目中拷贝文件上传的必备 jar 包：




```
<dependency>
    <groupId>org.apache.commons</groupId>
    <artifactId>commons-io</artifactId>
    <version>1.3.2</version>
</dependency>

<dependency>
    <groupId>commons-fileupload</groupId>
  <artifactId>commons-fileupload</artifactId>
     <version>1.3.1</version>
 </dependency>

```

```
<dependency>
    <groupId>com.sun.jersey</groupId>
    <artifactId>jersey-client</artifactId>
    <version>1.18.1</version>
</dependency>
```

3.4 编写控制器实现上传图片

```
/**
 * @author bruceliu
 * @create 2019-07-20 22:49
 * @description
 */
@Controller("fileUploadController2")
public class FileUploadController2 {


    public static final String FILESERVERURL =

"http://localhost:9090/day06_spring_image/uploads/";

    /**
     * 文件上传，保存文件到不同服务器
     */
    @RequestMapping("/fileUpload2")
    public String testResponseJson(String picname,
MultipartFile uploadFile) throws
            Exception{
        //定义文件名
        String fileName = "";
        //1.获取原始文件名
       String uploadFileName =
uploadFile.getOriginalFilename();
        //2.截取文件扩展名
        String extendName
=uploadFileName.substring(uploadFileName.lastIndexOf(".")
+ 1,
                uploadFileName.length());

        //3.把文件加上随机数，防止文件重复
        String uuid =
UUID.randomUUID().toString().replace("-
","").toUpperCase();
        //4.判断是否输入了文件名
        if (!StringUtils.isEmpty(picname)) {
          fileName = uuid + "_" + picname + "." +
extendName;
        } else {
            fileName = uuid + "_" + uploadFileName;
```

```
31          }
32          System.out.println(fileName);
33          //5.创建sun 公司提供的jersey 包中的Client 对象
34          Client client = Client.create();
35          //6.指定上传文件的地址，该地址是web 路径
36          WebResource resource =
client.resource(FILESERVERURL + fileName);
37          //7.实现上传
38          String result =
resource.put(String.class,uploadFile.getBytes());
39          System.out.println(result);
40          return "success";
41      }
42  }
```

3.5 编写JSP

```
1  <form action="fileUpload2" method="post"
enctype="multipart/form-data">
2       名称：<input type="text" name="picname"/><br/>
3       图片：<input type="file" name="uploadFile"/><br/>
4       <input type="submit" value="上传"/>
5  </form>
```

3.6 配置解析器

```
<!-- 配置文件上传解析器 -->
<!-- id 的值是固定的-->
<bean id="multipartResolver"
class="org.springframework.web.multipart.commons.CommonsMu
ltipartResolver">
    <!-- 设置上传文件的最大尺寸为 5MB -->
    <property name="maxUploadSize">
        <value>5242880</value>
    </property>
</bean>
```