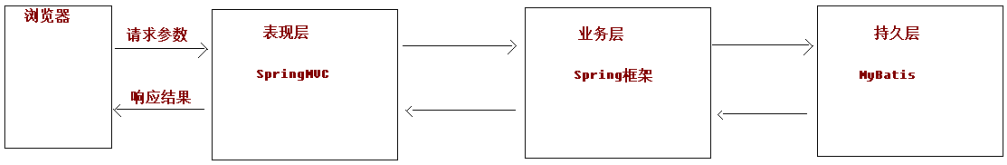
服务器端分成三层框架



MVC设计模型

M　　model模型 JavaBean
V　　View视图 JSP
C　　Controller控制器 Servlet

restful编程风格

原来的方式：模拟增删改查的请求地址

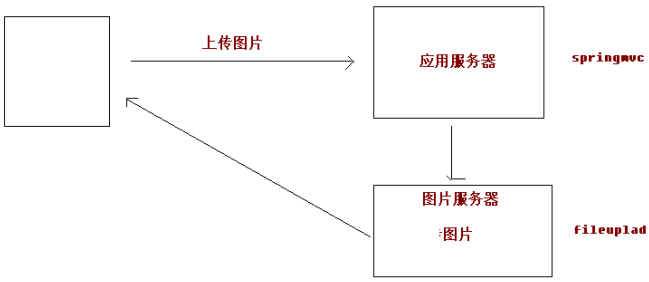
springmvc/saveUser　　新增
springmvc/updteUser　　修改
springmvc/findAll　　查询所有

restful风格：请求路径以后只有一个

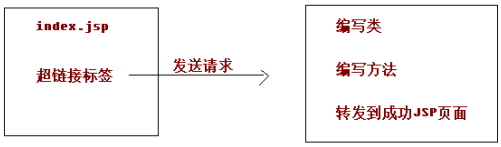
springmvc/save　新增（POST）
springmvc/user　修改（PUT）
springmvc/user　查询所有（GET）
springmvc/user/{id}　　通过ID查询（GET）
https://item.jd.com/113.html

<a href="springmvc/user/3">超链接<a>　　根据不同的请求方式让不同方法去执行，发送get请求

跨服务器的文件上传



入门程序的需求：

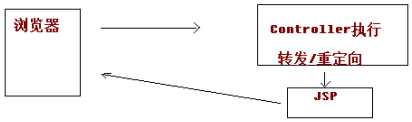


1. 需要搭建开发的环境

2. 编写入门的程序

响应的方式

同步

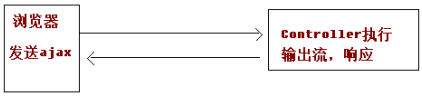


转发重定向的区别

转发是1次请求，同一个request对象　不用写项目名

重定向是2次请求 编写项目名

异步

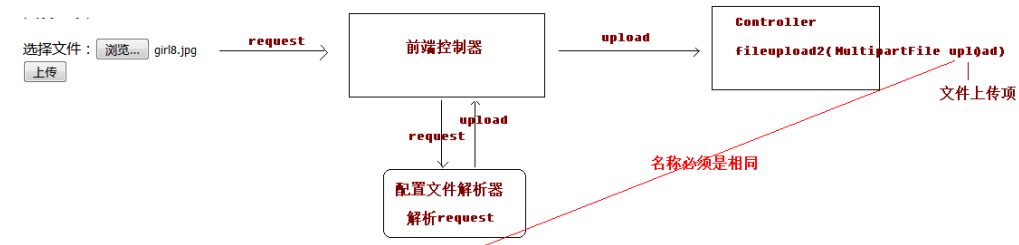


1. 启动服务器，加载一些配置文件
    * DispatcherServlet对象创建
    * springmvc.xml被加载了
    * HelloController创建成对象

2. 发送请求，后台处理请求

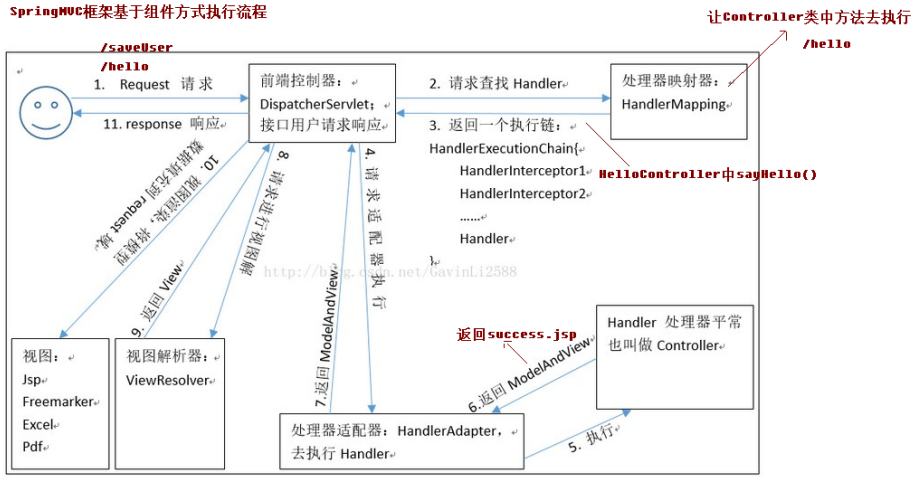
SpringMVC框架文件上传的原理分析

```
<!-- 配置文件解析器对象，要求id名称必须是multipartResolver -->
    <bean id="multipartResolver"
class="org.springframework.web.multipart.commons.CommonsMultipartResolver">
        <property name="maxUploadSize" value="10485760"/>
    </bean>
```

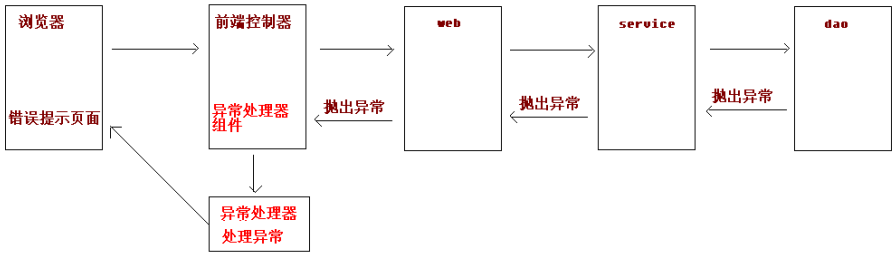



```
<form action="/user/fileupload1" method="post" enctype="multipart/form-data">
    选择文件：<input type="file" name="upload" /><br/>
    <input type="submit" value="上传" />
</form>
```

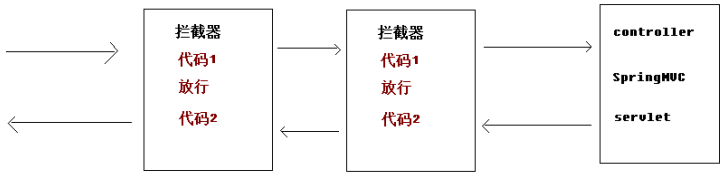
SpringMVC框架基于组件方式执行流程



SpringMVC异常处理

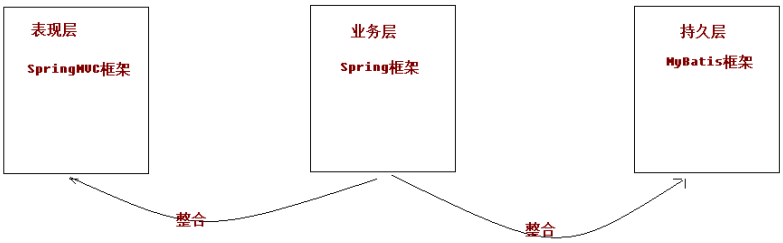


1. 编写自定义异常类（做提示信息的）
2. 编写异常处理器
3. 配置异常处理器（跳转到提示页面）



1. 编写拦截器类，实现HandlerInterceptor接口
2. 配置拦截器

服务器开发，分成3层



选项配置文件 + 注解的方式

Spring整合SpringMVC：启动Tomcat服务器的时候，需要加载spring的配置文件