# Springmvc 第二天 课堂笔记

回顾springmvc

* 前端控制器
* 处理器映射器
* 处理器适配器
* 视图解析器(jsp,ftl,pdf,vm)
* 注解开发
* 注解的处理器映射器
* 注解的处理器适配器
* 视图解析器
* RequestMapping:请求映射
* URL模版映射：springmvc实现restfull：requestMapping和PathVariable
* ModelAttribute
* RequestParam
* 参数传递
* 基本类型的参数传递
* Pojo类型的参数传递
* 集合：List map
* 转发与重定向

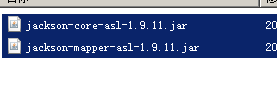
# 今天课程

# 返回json，请求json

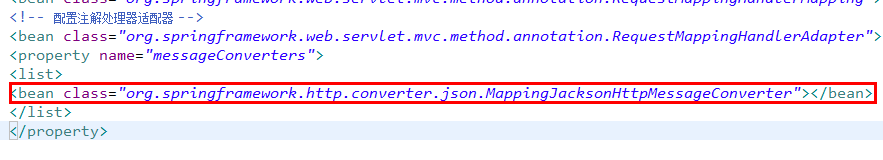
@RequestBody：请求json：把请求的json数据封装成javabean。

@responseBody：响应结果就是json格式的数据

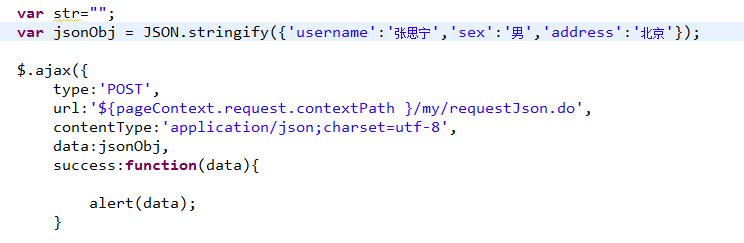
Springmvc整合了Jackson这个jar文件：



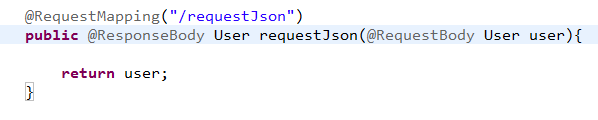
在注解适配器中注入Jackson的bean：



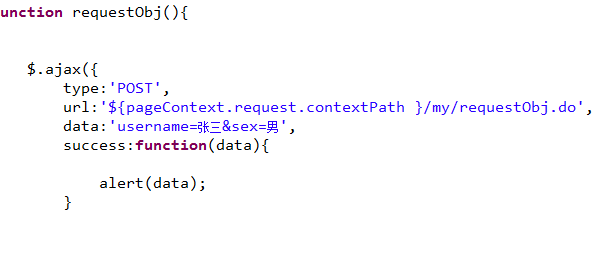
定义ajax请求json响应json页面：



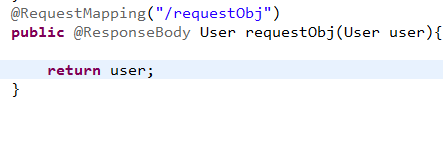
定义响应代码：



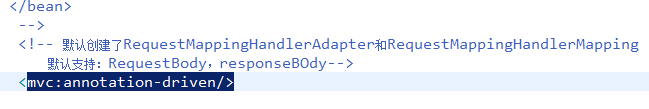
请求普通对象，返回json数据：



代码：

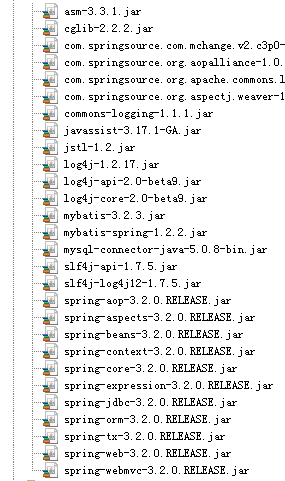


总结：



# 二．Ssm整合

1.创建web工程并且导入jar文件



2.配置beans.xml配置文件

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xmlns:mvc=*"http://www.springframework.org/schema/mvc"*

xmlns:context=*"http://www.springframework.org/schema/context"*

xmlns:aop=*"http://www.springframework.org/schema/aop"* xmlns:tx=*"http://www.springframework.org/schema/tx"*

xsi:schemaLocation=*"http://www.springframework.org/schema/beans*

*http://www.springframework.org/schema/beans/spring-beans-3.1.xsd*

*http://www.springframework.org/schema/mvc*

*http://www.springframework.org/schema/mvc/spring-mvc-3.1.xsd*

*http://www.springframework.org/schema/context*

*http://www.springframework.org/schema/context/spring-context-3.1.xsd*

*http://www.springframework.org/schema/aop*

*http://www.springframework.org/schema/aop/spring-aop-3.1.xsd*

*http://www.springframework.org/schema/tx*

*http://www.springframework.org/schema/tx/spring-tx-3.1.xsd "*>

<context:component-scan base-package=*"cn.itcast"*></context:component-scan>

<!-- 数据源 -->

<context:property-placeholder location=*"classpath:jdbc.properties"*/>

<bean id=*"dataSource"* class=*"com.mchange.v2.c3p0.ComboPooledDataSource"*>

<property name=*"driverClass"* value=*"${jdbc.driver}"*></property>

<property name=*"jdbcUrl"* value=*"${jdbc.url}"*></property>

<property name=*"user"* value=*"${jdbc.username}"*></property>

<property name=*"password"* value=*"${jdbc.password}"*></property>

</bean>

<!-- session工厂：sqlSessionFactory用来生成sqlSesion会话 -->

<bean id=*"sqlSessionFactory"* class=*"org.mybatis.spring.SqlSessionFactoryBean"*>

<property name=*"dataSource"* ref=*"dataSource"*></property>

<property name=*"configLocation"* value=*"classpath:sqlMapConfig.xml"*></property>

</bean>

<bean class=*"org.mybatis.spring.mapper.MapperScannerConfigurer"*>

<property name=*"basePackage"* value=*"cn.itcast.dao"*></property>

<property name=*"sqlSessionFactoryBeanName"* value=*"sqlSessionFactory"*></property>

</bean>

<!-- 事务 -->

<bean id=*"transactionManager"* class=*"org.springframework.jdbc.datasource.DataSourceTransactionManager"*>

<property name=*"dataSource"* ref=*"dataSource"*></property>

</bean>

<tx:advice id=*"advice"* transaction-manager=*"transactionManager"*>

<tx:attributes>

<tx:method name=*"save\*"* propagation=*"REQUIRED"*/>

<tx:method name=*"update\*"* propagation=*"REQUIRED"*/>

<tx:method name=*"select\*"* propagation=*"REQUIRED"*/>

<tx:method name=*"delete\*"* propagation=*"REQUIRED"*/>

</tx:attributes>

</tx:advice>

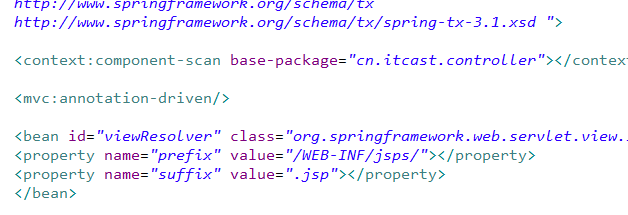
<aop:config>

<aop:advisor advice-ref=*"advice"* pointcut=*"execution(\* cn.itcast.service.\*.\*(..))"*/>

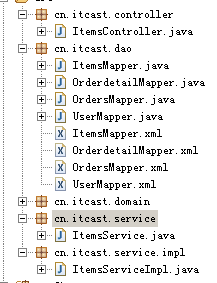
</aop:config>

</beans>

3.配置springmvc.xml配置文件



4.三层结构代码



跨服务器上传图片：



定义图片回显页面ajax：



定义跨服务器上传图片java代码：

@RequestMapping("uploadPic")

**public** **void** uploadPic(HttpServletRequest request,HttpServletResponse response,String fileName,PrintWriter out){

//把Request对象转换成多部件上传类型

MultipartHttpServletRequest mh = (MultipartHttpServletRequest) request;

CommonsMultipartFile cm = (CommonsMultipartFile) mh.getFile(fileName);

**byte**[] fbytes = cm.getBytes();

String newFileName="";

SimpleDateFormat sdf = **new** SimpleDateFormat("yyyyMMddHHmmssSSS");

newFileName=sdf.format(**new** Date());

Random randon = **new** Random();

**for**(**int** i=0;i<3;i++){

newFileName = newFileName+randon.nextInt(10);

}

String originalFilename = cm.getOriginalFilename();

String suffix = originalFilename.substring(originalFilename.lastIndexOf("."));

Client client = Client.*create*();

//绑定上传路径到客户端

WebResource resource = client.resource(SSMConstants.*PIC\_HOST*+"/upload/"+newFileName+suffix);

resource.put(String.**class**, fbytes);

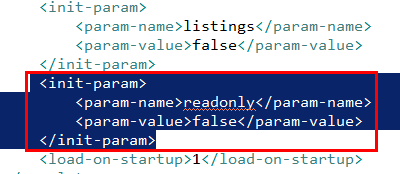
String fullPath=SSMConstants.*PIC\_HOST*+"/upload/"+newFileName+suffix;

String relativePath="/upload/"+newFileName+suffix;

String result="{\"fullPath\":\""+fullPath+"\",\"ralativePath\":\""+relativePath+"\"}";

out.print(result);

}

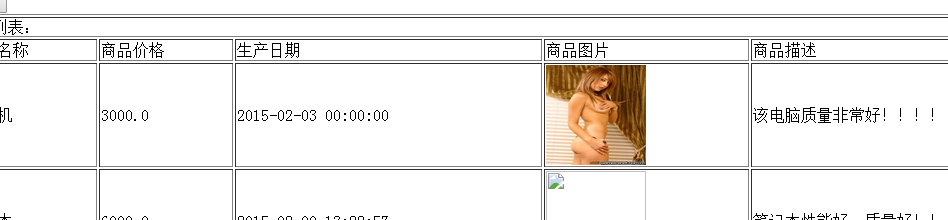


java.lang.ClassCastException: org.apache.catalina.connector.RequestFacade cannot be cast to org.springframework.web.multipart.MultipartHttpServletRequest

错误原因1：没有配置上传图解析器

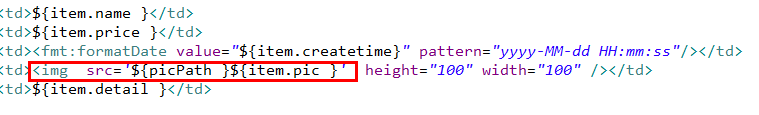
跨服务器上传必须是提交表达，使用ajax提交没有提交表单，这样就会类型转换错误。

列表图片回显：



拼接图片路径进行回显



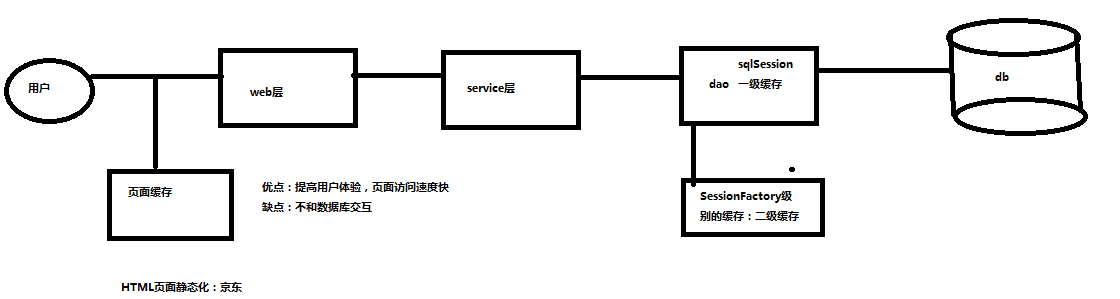
\*

# 有了struts2,为什么还用springmvc？

* 实现机制
* Struts2依赖于过滤器，springmvc依赖Servlet
* 运行效率
* Struts2是多列，springmvc是单例。
* Struts2的参数是基于属性进行封装的，springmvc基于方法进行封装。

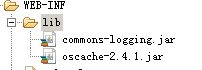
# 三．页面缓存

页面缓存：

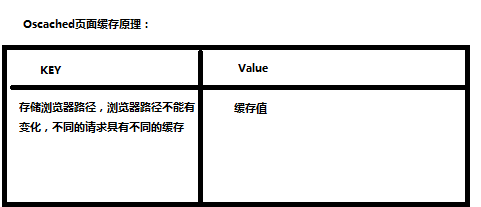


页面缓存需要导入jar文件：

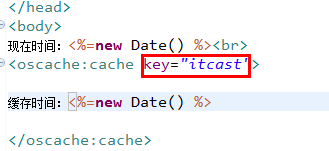
Oscache：



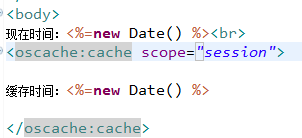
Oscache缓存原理：



无论浏览器地址如何变化,缓存都保持一致：只需要保证key一致：



存储到session中：



如何把Oscache整合到ssm\_day02\_01中：

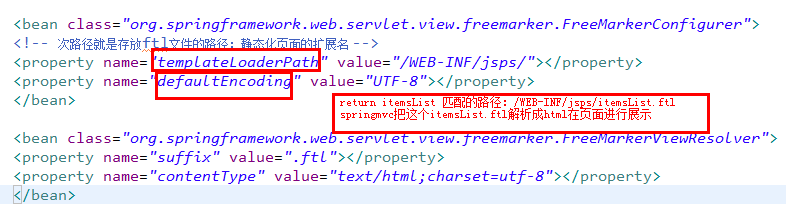
# 四.springmvc整合freemarker

导入jar文件：





配置文件修改springmvc.xml:

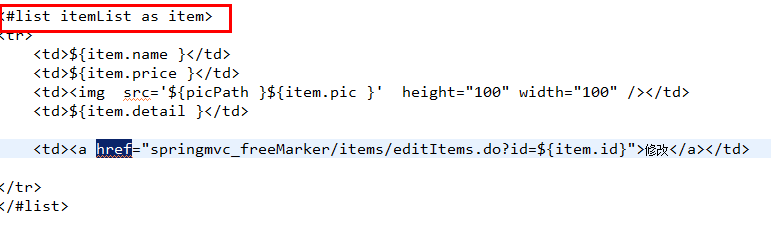


改写模版文件：

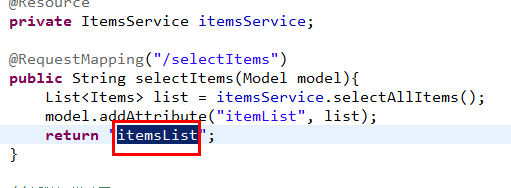
定义一个变量：picPath，这个变量使用来回显图片，picPath和查询数据库的相对路径进行组合，组合成一个绝对路径，图片就显示。



改造回显：

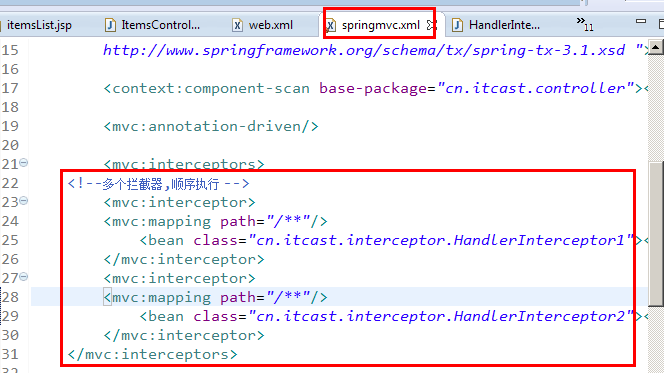


代码显示：



# 五．Springmvc拦截器

定义全局拦截器：



Prehanle里全部放行：

执行效果：

interceptor1................preHandle

interceptor2................preHandle

interceptor2................postHandle

interceptor1................postHandle

interceptor2................afterCompletion

interceptor1................afterCompletion

拦截器1不放行：

interceptor1................preHandle

拦截器2不放行：

interceptor1................preHandle

interceptor2................preHandle

interceptor1................afterCompletion

# 六.springmvc整合hibernate