# ajax支持补充

关于参数的传递以及接受

1.可以使用普通方式

2.使用json方式

普通方式传递

key=value 后台接受参数的时候就按照=去截取获取

如果说。文本的格式是json  
{key:value}

那么后台就不能按照=去截取获取，应该按照json格式去接受

加一个注解 @requestbody

@responsebody 响应的时候按照json格式去输出

@requestbody 接受参数的时候，按照json格式去接受

他们都是json格式

# 文件上传

前端部分

post enctype=muitlpart/form-data

type=file

后端部分

MultipartResolver 去解析request对象，进一步包装，把request中的参数以及文件封装到MultipartResolver中去

类似于ajax application/json

后台接受的时候也会需要发生变化才可以接受数据的

默认方式都是

application/x-www-form-urlencoded

文件流方式传递

muitlpart/form-data

如果说按照json格式

application/json

在后端

mysql 是数据库

数据库管理系统

还是要严谨

# 拦截器

和以前的过滤器作用是一样的，都是在调用具体的请求之前，先过滤，拦截

过滤器是基于函数的一个回调实现的，在调用这个函数之前先调用目标，目标结束之后，回来再调用过滤器。

拦截器是基于代理类的实现。在调用具体目标对象方法之前，先拦截，然后之后执行目标方法，再调用拦截，之前之后拦截。

springmvc中，可以使用代理类实现过滤，叫拦截器。

拦截器其实也是AOP ，在请求的过程中使用的拦截，通过配置文件实现。

过滤器中，通过filter 配置过滤器对象实例化

配置拦截器有两种方式

全局的，所有的请求都走的拦截器

局部的，针对某个请的拦截

之前做什么

之后做什么

拦截视图中需要渲染的数据

ajax扩展 json格式的支持

文件上传 type post 后台接受 保存到硬盘中 路径 绝对路径 通过配置文件 ，bean 通过id 统一的。

拦截器 和过滤器 代理类 全局 局部等

# thymeleaf

可以理解为框架，也可以理解为页面模板 和jsp是一样的。，都是动态页面的模板

早在以前，所谓的前端

主要是html+css+js 偏向 html+css部分

现在所谓的前端编程

主要是javascript+vue框架

偏向于编程前端

偏向于美化的前端

编写thymeleaf代码，必须要符合规范，可以修改，不需要html5模式。最好还是按照规范。

前后端能分离开发

vue+后端 前后端分离式开发。

还有一种，。返回的是页面，这个页面模板可以独立让前端设计，后端最好不要有影响。