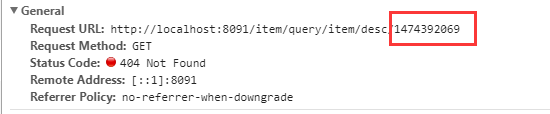
# 富文本编辑器实现

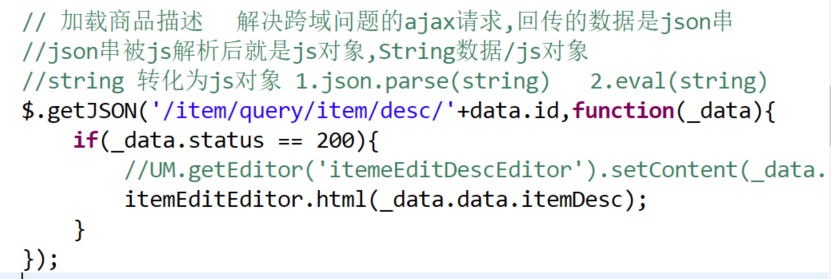
## 商品详情回显

### 页面分析

1. 页面分析



1. 页面JS



### 编辑Controller

//实现商品描述信息的查询

@RequestMapping("/query/item/desc/{itemId}")

@ResponseBody

**public** SysResult findItemDescById(@PathVariable Long itemId){

**try** {

ItemDesc itemDesc = itemService.findItemDescById(itemId);

**return** SysResult.*oK*(itemDesc);

} **catch** (Exception e) {

e.printStackTrace();

}

**return** SysResult.*build*(201, "商品详情查询失败");

}

### 编辑Service

@Override

**public** ItemDesc findItemDescById(Long itemId) {

**return** itemDescMapper.selectByPrimaryKey(itemId);

}

### 页面效果



## 商品修改

### 编辑Controller

//实现商品的更新操作

@RequestMapping("/update")

@ResponseBody

**public** SysResult updateItem(Item item,String desc){

**try** {

itemService.updateItem(item,desc);

//System.out.println("sadfasdfasdfasdafsdfdsa");//执行性能太低

***logger***.info("~~~~~~~~~~~~~~~~~~~商品更新成功}");

***logger***.debug("~~~~~~~~~~~~~~~~~~~商品更新成功}");

**return** SysResult.*oK*();

} **catch** (Exception e) {

e.printStackTrace();

***logger***.error(e.getMessage());

}

**return** SysResult.*build*(201,"修改商品数据失败");

}

### 编辑Service

@Override

**public** **void** updateItem(Item item,String desc) {

item.setUpdated(**new** Date());

//动态更新操作 将对象中不为null的数据充当set条件 主键除外

itemMapper.updateByPrimaryKeySelective(item);

ItemDesc itemDesc = **new** ItemDesc();

itemDesc.setItemId(item.getId());

itemDesc.setItemDesc(desc);

itemDesc.setUpdated(item.getUpdated());

itemDescMapper.updateByPrimaryKeySelective(itemDesc);

}

## 商品描述删除

### 编辑Service

说明:因为商品表与商品描述表是一对一对应.所以删除商品表时,删除商品描述信息

@Override

**public** **void** deleteItems(Long[] ids) {

itemMapper.deleteByIDS(ids);

itemDescMapper.deleteByIDS(ids);

}

# 文件上传

## 文件上传步骤

### 实现步骤

1. 准备文件上传的按钮 type=”file”
2. 一般定义一个form表单进行提交,提交方式POST,一般文本提交不需要添加任何的参数.但是如果提交多媒体信息时需要修改请求类型.
3. 采用输入流进行数据接收InputStream
4. FileAPI实现 定义输出文件夹, 文件格式:D:jt-upload/1.jpg
5. 调用文件的输出流进行输出outputStream.
6. 用完流之后关闭流.

## 入门案例

### 编辑文件上传页面

<%@ page language="java" contentType="text/html; charset=UTF-8"

pageEncoding="UTF-8"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>入门案例</title>

</head>

<body>

<h1>文件上传入门案例</h1>

<!--开启多媒体标签 -->

<form action="http://localhost:8091/file" method="post"

enctype="multipart/form-data">

<input name="image" type="file"/>

<button type="submit"/>提交

</form>

</body>

</html>

### 编辑文件上传解析器

<!--文件上传视图解析器

要求:要求bean的Id必须为multipartResolver

规范参数:

1.定义最大上传数量

2.可能会定义上传的字符集编码

-->

<bean id=*"multipartResolver"*

class=*"org.springframework.web.multipart.commons.CommonsMultipartResolver"*>

<property name=*"maxUploadSize"* value=*"10485760"*/>

<property name=*"defaultEncoding"* value=*"UTF-8"*/>

</bean>

### 实现文件上传

/\*\*

\* 如果参数需要赋值,需要通过springMVC提供的解析器才可以.

\* 注意事项:

\* 参数接收,必须和页面提交的name属性相同.否则参数不能提交.

\* 程序报404报错.

\* **@param** image

\* **@return**

\* **@throws** IllegalStateException

\* **@throws** IOException

\*/

@RequestMapping("/file")

**public** String imageFile(MultipartFile image) **throws** IllegalStateException, IOException{

//1.定义文件上传的目录 E:\jt-upload

File imageFile = **new** File("E:/jt-upload");

//2.判断文件是否存在

**if**(!imageFile.exists()){

//创建文件夹

imageFile.mkdirs();

}

//3.获取文件名称

String fileName = image.getOriginalFilename();

//4.实现文件上传 文件路径/文件名称

image.transferTo(**new** File("E:/jt-upload/"+fileName));

System.***out***.println("文件上传实现成功!!!!!");

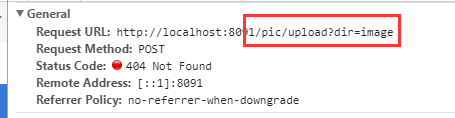
//跳转到index页面

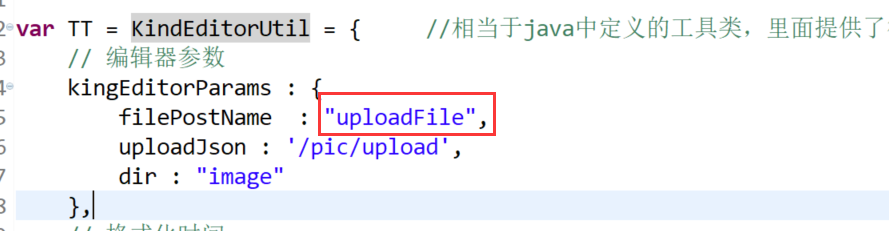
**return** "index";

}

## 富文本编辑器实现文件上传

### 页面js分析





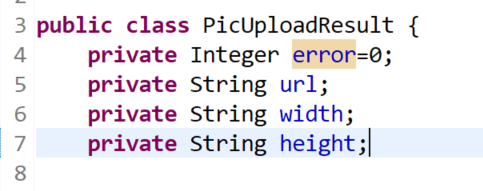
### 编辑VO对象

格式要求:

**{"error":0,"url":"图片的保存路径","width":图片的宽度,"height":图片的高度}**

参数说明： 0代表是一张图片，如果是0，前台才可以解析并显示。1代表不是图片，

不显示如果不设置宽度和高度，则默认用图片原来的大小，所以不用设置



### 编辑Controller

@RequestMapping("/pic/upload")

@ResponseBody

**public** PicUploadResult fileUpload(MultipartFile uploadFile){

**return** fileService.fileUpload(uploadFile);

}

### 编辑Service

@Service

**public** **class** FileServiceImpl **implements** FileService {

**private** String localPath = "E:/jt-upload/";

**private** String urlPath = "http://image.jt.com/";

/\*\*

\* 1.判断图片的类型 jpg|png|git

\* 2.判断是否为恶意程序

\* 3.为了提高检索效率.将文件分文件存储.

\* UUID:hash随机算法(当前毫秒数+算法+hash)=32位hash值 2^32=21亿

\* 3.1 aaaaaaaa-bbbbbbbb-cccccccc-dddddddd/1.jpg

\* 优点:几乎永远不需要修改代码

\* 3.2 yyyy/MM/dd

\*

\* 4.如何杜绝文件重名现象

\* uuid+随机数(100).jpg

\* 5.实现文件上传

\*/

@Override

**public** PicUploadResult fileUpload(MultipartFile uploadFile) {

PicUploadResult uploadResult = **new** PicUploadResult();

//1.判断是否为图片类型

String fileName = uploadFile.getOriginalFilename();//abc.jpg

fileName = fileName.toLowerCase(); //将字符全部转化为小写

**if**(!fileName.matches("^.\*\\.(jpg|png|gif)$")){

uploadResult.setError(1);

**return** uploadResult; //文件不是图片

}

//2.判断是否为恶意程序

**try** {

BufferedImage bufferedImage =

ImageIO.*read*(uploadFile.getInputStream());

**int** height = bufferedImage.getHeight(); //获取图片的高度

**int** width = bufferedImage.getWidth(); //获取图片的宽度

**if**(height == 0 || width == 0){

//表示不是图片

uploadResult.setError(1);

**return** uploadResult;

}

//程序执行到这里 证明是一张图片

//为文件进行分文件存储 yyyy/MM/dd

String datePath =

**new** SimpleDateFormat("yyyy/MM/dd").format(**new** Date());

//定义文件保存的路径 E:/jt-upload/2018/12/12

String dirPath = localPath + datePath;

//判断文件夹是否存在

File dirFile = **new** File(dirPath);

**if**(!dirFile.exists()){

//文件不存在时,应该创建文件夹

dirFile.mkdirs();

}

//4.动态生成文件名称 UUID+三位随机数 ss-aa-cc ssaacc

String uuid = UUID.*randomUUID*().toString().replace("-", "");

**int** randomNum = **new** Random().nextInt(1000);

String imageFileType =

fileName.substring(fileName.lastIndexOf(".")); //.jpg

String imageFileName = uuid + randomNum + imageFileType;

//5.实现文件上传 E:/jt-upload/2018/12/12/ssdfsd.jpg

String imageLocalPath = dirPath +"/" + imageFileName;

uploadFile.transferTo(**new** File(imageLocalPath));

//6.封装参数

uploadResult.setHeight(height+"");

uploadResult.setWidth(width+"");

//7.准备虚拟路径 "http://image.jt.com/yyyy/MM/dd/abc.jpg";

String imageUrlPath = urlPath + datePath + "/" + imageFileName;

uploadResult.setUrl(imageUrlPath);//添加虚拟路径

} **catch** (Exception e) {

e.printStackTrace();

uploadResult.setError(1);

**return** uploadResult;

}

**return** uploadResult;

}

}

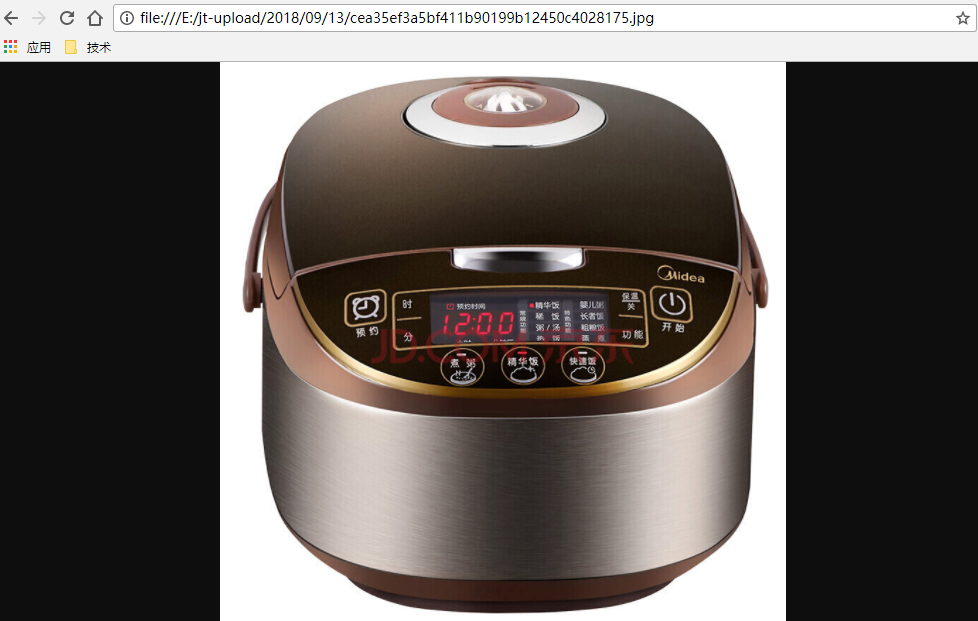
### 文件上传测试

当文件上传时,获取虚拟路径

<http://image.jt.com/2018/09/13/cea35ef3a5bf411b90199b12450c4028175.jpg>

将虚拟路径改为绝对路径.检测是否正确.

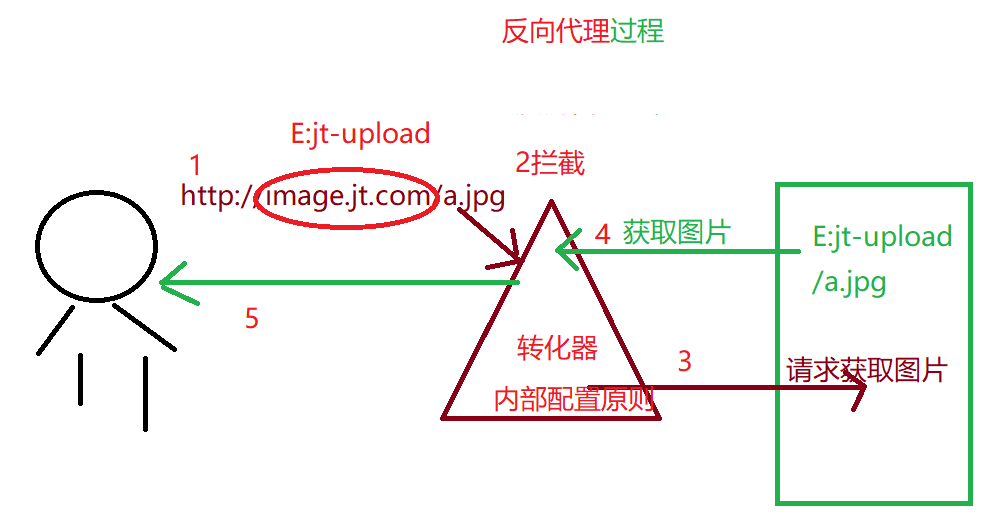
<file:///E:/jt-upload/2018/09/13/cea35ef3a5bf411b90199b12450c4028175.jpg>



## 反向代理

### 反向代理说明

需求:如果用户发起请求<http://image.jt.com/a.jpg>,需要将请求路径自动转化为e:jt-upload/a.jpg,并且获取图片信息.之后将数据返回给用户.



代理步骤:

1. 用户发起请求.
2. 被转化器所拦截,根据内部配置原则.将拦截的内容进行修改.修改为正确的本地磁盘路径进行数据的请求.
3. 根据本地磁盘路径请求图片
4. 将获取到的图片返回给转化器
5. 转化器将获取的图片信息交给用户.用户页面正常显示数据.

## Nginx代理服务器

### Nginx介绍

*Nginx* (engine x) 是一个高性能的[HTTP](https://baike.baidu.com/item/HTTP" \t "_blank)和[反向代理](https://baike.baidu.com/item/%E5%8F%8D%E5%90%91%E4%BB%A3%E7%90%86/7793488" \t "_blank)服务，也是一个IMAP/POP3/SMTP服务。Nginx是由伊戈尔·赛索耶夫为[俄罗斯](https://baike.baidu.com/item/%E4%BF%84%E7%BD%97%E6%96%AF/125568" \t "_blank)访问量第二的Rambler.ru站点（俄文：Рамблер）开发的，第一个公开版本0.1.0发布于2004年10月4日。

其将[源代码](https://baike.baidu.com/item/%E6%BA%90%E4%BB%A3%E7%A0%81" \t "_blank)以类BSD许可证的形式发布，因它的稳定性、丰富的功能集、示例配置文件和低系统资源的消耗而[闻名](https://baike.baidu.com/item/%E9%97%BB%E5%90%8D/2303308" \t "_blank)。2011年6月1日，nginx 1.0.4发布。

Nginx是一款[轻量级](https://baike.baidu.com/item/%E8%BD%BB%E9%87%8F%E7%BA%A7/10002835" \t "_blank)的[Web](https://baike.baidu.com/item/Web/150564" \t "_blank) 服务器/[反向代理](https://baike.baidu.com/item/%E5%8F%8D%E5%90%91%E4%BB%A3%E7%90%86/7793488" \t "_blank)服务器及[电子邮件](https://baike.baidu.com/item/%E7%94%B5%E5%AD%90%E9%82%AE%E4%BB%B6/111106" \t "_blank)（IMAP/POP3）代理服务器，并在一个BSD-like 协议下发行。其特点是占有内存少，[并发](https://baike.baidu.com/item/%E5%B9%B6%E5%8F%91/11024806" \t "_blank)能力强，事实上nginx的并发能力确实在同类型的网页服务器中表现较好，中国大陆使用nginx网站用户有：百度、[京东](https://baike.baidu.com/item/%E4%BA%AC%E4%B8%9C/210931" \t "_blank)、[新浪](https://baike.baidu.com/item/%E6%96%B0%E6%B5%AA/125692" \t "_blank)、[网易](https://baike.baidu.com/item/%E7%BD%91%E6%98%93/185754" \t "_blank)、[腾讯](https://baike.baidu.com/item/%E8%85%BE%E8%AE%AF/112204" \t "_blank)、[淘宝](https://baike.baidu.com/item/%E6%B7%98%E5%AE%9D/145661" \t "_blank)等。

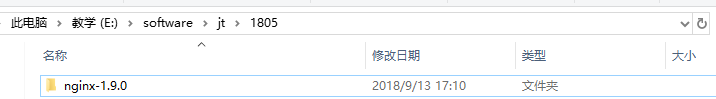
速度:高并发条件下Nginx 5万/秒 实际速度3万/s

### Nginx下载



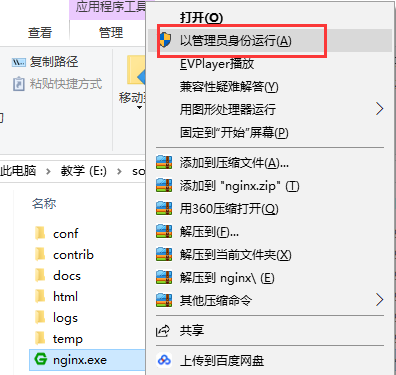
### Nginx安装步骤

1. 将文件解压到本地,不要放C盘不要有中文路径

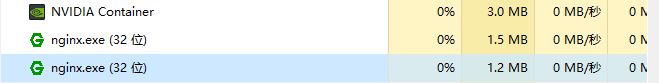


1. 启动Nginx

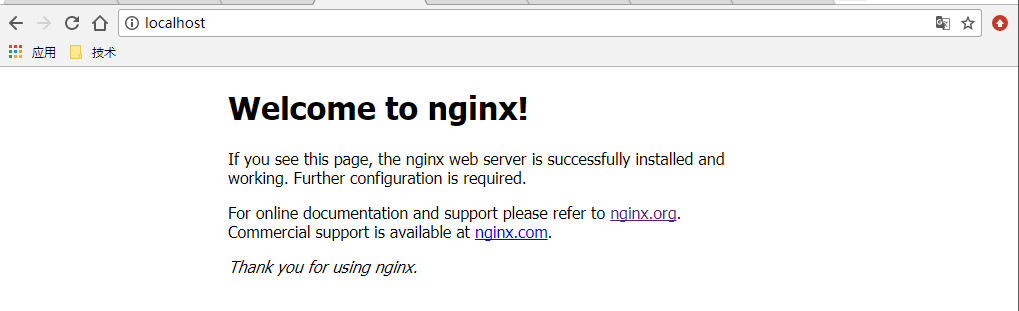
说明:Nginx第一次使用时需要以超级管理员的身份运行.



1. 检查启动进程



1. 访问网址



### Nginx中进程介绍

Nginx中启动后会有2个进程.一个是主进程.另外一个是守护进程.

主进程:提供反向代理等服务. 内存占用较大

守护进程:防止主进程意外关闭.内存占用较小

### Nginx启动命令

如果执行Nginx命令,则windows系统中必须在nginx.exe所在的目录下执行.

命令说明:

1.启动nginx start nginx

2.停止nginx nginx -s stop

3.重启nginx nginx -s reload 先启动在重启

### Nginx调用过程

server {

listen 80;

server\_name localhost;

#charset koi8-r;

#access\_log logs/host.access.log main;

location / {

#root 代表文件夹 index 默认访问

root html;

index index.html index.html;

}

}

## 实现图片回显

### 配置Nginx代理服务

#配置图片服务器

server {

listen 80;

server\_name image.jt.com;

location / {

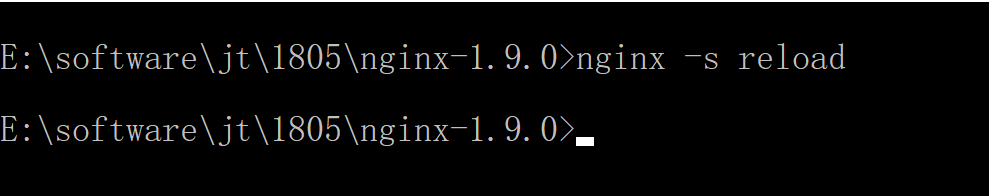
#个别电脑需要编辑 /

root E:\jt-upload;

}

}

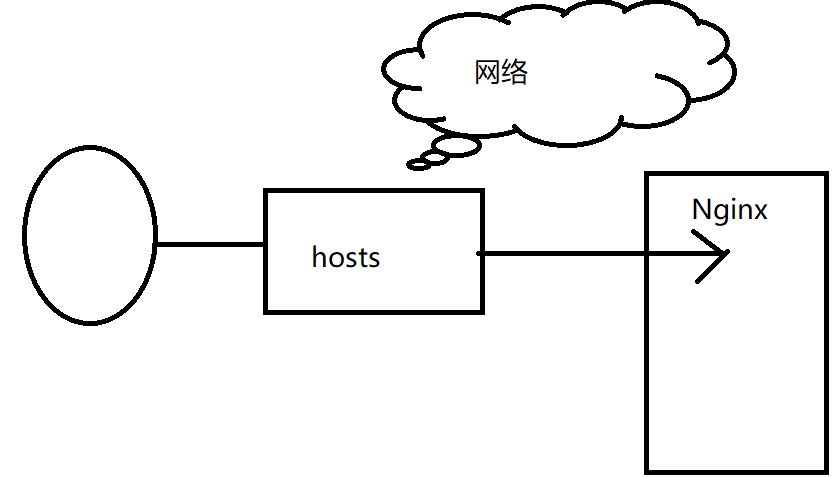
修改完成后重启Nginx

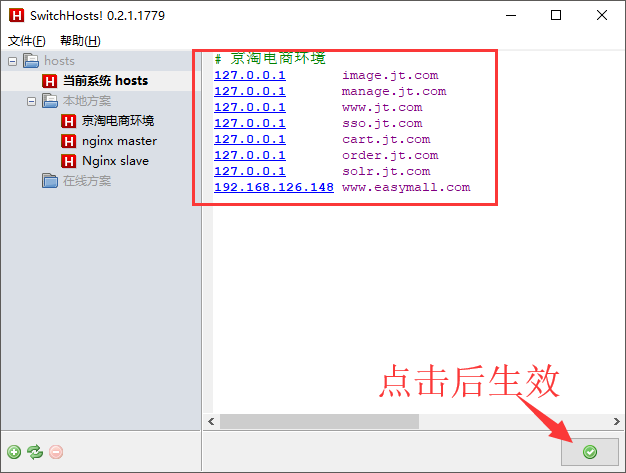


### 修改Hosts文件

说明:因为没有购买<http://image.jt.com的域名.所以暂时修改hosts文件让请求访问本地.之后才能被nginx>的监听所拦截.

路径: C:\Windows\System32\drivers\etc\hosts





# 京淘电商环境

127.0.0.1 image.jt.com

127.0.0.1 manage.jt.com

127.0.0.1 www.jt.com

127.0.0.1 sso.jt.com

127.0.0.1 cart.jt.com

127.0.0.1 order.jt.com

127.0.0.1 solr.jt.com

192.168.126.148 [www.easymall.com](http://www.easymall.com)

### 效果测试

反向代理生效.

