# 京淘项目实现文件上传

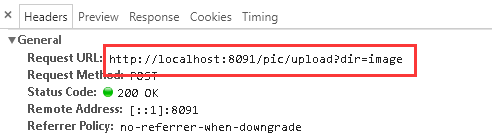
## 富文本文件上传

### 页面分析

1.页面js



2.url



### 编辑Controller

//实现文件上传

@RequestMapping("/pic/upload")

@ResponseBody

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

**return** fileUploadService.uploadPic(uploadFile);

}

### 编辑Service

/\*\*

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

\* error:0 是图片 error:1 表示不是图片

\* url: 真实的物理路径/网站虚拟路径 image.jt.com

\* width":图片的宽度,"height":图片的高度

\* https://img20.360buyimg.com/vc/jfs/t19297/80/1703731107/320815/da350442/5ad44f3dNc1d94408.jpg"

\*

\* 文件上传的思路

\* 1.判断文件的类型是否为图片

\* 2.判断是否为恶意程序( 转化为字节码文件 ImageBuffer)

\* 3.图片如何存储---分文件存储 作用:能够保证图片的量在一个可控范围内

\* 4.应该将文件名称复杂化 UUID + 随机数999

\* 5.实现文件上传

\* 6.准备虚拟路径进行返回

\*/

@Override

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

PicUploadResult result = **new** PicUploadResult();

//1.判断是否为图片 abc.jpg

String fileName = uploadFile.getOriginalFilename();

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

result.setError(1); //表示不是图片

**return** result;

}

**try** {

//2.判断是否为恶意程序

BufferedImage bufferedImage =

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

//3.获取图片的宽度和高度

**int** height = bufferedImage.getHeight();

**int** width = bufferedImage.getWidth();

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

result.setError(1);

**return** result;

}

//4.准备文件存储的路径 通过动态注入方式获取

//5.实现分文件存储 yyyy/MM/dd

String dateDir =

**new** SimpleDateFormat("yyyy/MM/dd")

.format(**new** Date());

//6.形成保存图片的文件夹 E:/jt-upload/yyyy/MM/dd

String picDir = localDir + dateDir;

//7.生成文件夹

File picFile = **new** File(picDir);

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

picFile.mkdirs();

}

//8.1为了让文件不重名 形成UUIDasdfasdfasdfasdfdfasdf

String uuid = UUID.*randomUUID*()

.toString().replace("-", "");

//8.2添加随机数

**int** randomNum = **new** Random().nextInt(999);

//8.3获取文件后缀 abc.jpg

String fileType =

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

//8.4 形成最终文件名称

String realFileName = uuid + randomNum + fileType;

//9.生成文件的本地磁盘路径全名

//E:/jt-upload/yyyy/MM/dd/werqwer100.jpg

String localPath = picFile + "/" + realFileName;

//10.实现文件上传

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

//11.添加文件的宽度和高度

result.setHeight(height+"");

result.setWidth(width+"");

//12.准备文件的虚拟路径 http://image.jt.com/2018/05/07/文件名称

//通过动态取值的方式获取数据

String urlPath = urlPre + dateDir + "/" + realFileName;

result.setUrl(urlPath);

} **catch** (Exception e) {

e.printStackTrace();

result.setError(1);//表示为恶意程序

}

**return** result;

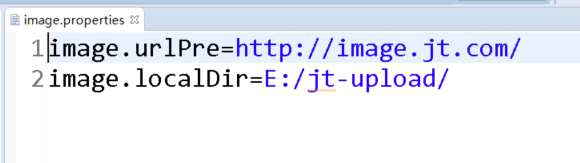
}

### 属性动态取值

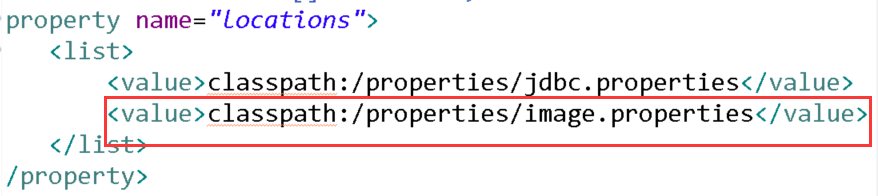
说明:文件存储的地址和虚拟路径的地址应该动态获取,减少程序的耦合性

采用@Value注解实现.

1. spring容器添加image.properties文件



1. 引入文件



1. 动态取值

/\*\*

\* spring容器中如何动态的获取数据???

\* **@Value注解** 实现数据的动态的获取

\* 说明: **@Value注解可以直接为String或基本类型赋值.**

\* 同时.可以利用Spring的机制可以为array list Map set properties赋值

\* 有时间再说...

\*

\* 注意事项:

\* 1.在SSM框架中,该注解需要依赖@Service注解(Spring容器内部才能使用)

\* 2.如果使用SpringBoot则任何地方都能用

\*/

@Value("${image.localDir}")

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

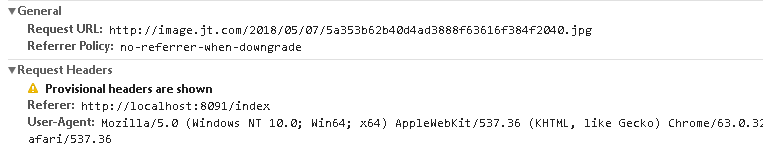
@Value("${image.urlPre}")

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

## 图片的回显

### 图片回显介绍

说明:当图片上传成功后,返回给客户端一个图片的虚拟路径



分析:

1. 图片之所以没有回显的原因是网址不能正常的访问.

url:**http://image.jt.com**/2018/05/07/5a353b62b40d4ad3888f63616f384f2040.jpg

图片的真实的地址

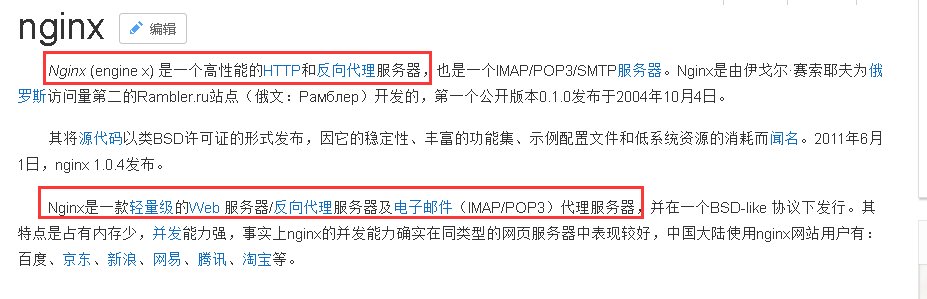
**E:/jt-upload**/2018/05/07/5a353b62b40d4ad3888f63616f384f2040.jpg

**解决方案:**

**说明:采用Nginx反向代理**

## Nginx介绍

### 介绍



特点说明:

1. **实现反向代理**
2. 实现负载均衡
3. Nginx主要实现了请求的**转发**以及响应.Nginx的开发的语言C语言开发.Nginx可以实现秒开秒关

### Nginx的下载

网址: http://nginx.org/en/download.html



### Nginx安装及配置

1. 安装说明

将课前资料中的nginx的安装包导入到本机,其中不要有中文/空格/不要放C盘



1. 启动Nginx

说明:如果是第一次使用Nginx,则以管理员身份启动

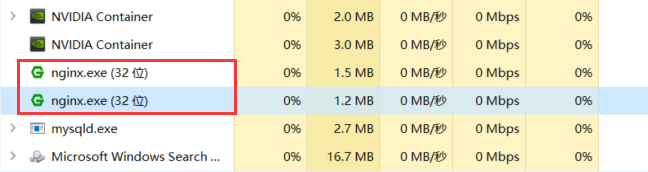


### Nginx关于线程的说明

说明:Nginx的启动会每次启动2个进程.一个是主进程 一个是守护进程.

主进程是提供服务的主要的程序,**守护进程是为了防止主进程意外关闭的**.

主进程占用资源较多,守护相对较少.



### Nginx的命令

**1.启动 start nginx**

2.停止 nginx -s stop

**3.重启 nginx -s reload**

**说明:**如果通过命令启动Nginx则要求必须在nginx的可执行文件夹下运行命令(根目录)

**E:\software\jt\1712\nginx-1.9.0**

### Nginx的配置文件介绍



### 实现图片的回显

说明:

需要将用户发起的请求<http://image.jt.com>/abc.jpg的路径转向到具体的图片文件夹下E:/jt-upload/abc.jpg下

配置文件配置:

#图片服务器

server {

#监听的端口

listen 80;

#服务名称

server\_name image.jt.com;

#请求拦截后做的具体的操作 根据要求转发到具体的文件夹中

location / {

root E:\jt-upload;

}

}

### 配置hosts文件

说明:host文件的路径C:\Windows\System32\drivers\etc\hosts文件.添加图片的转发路径

# 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

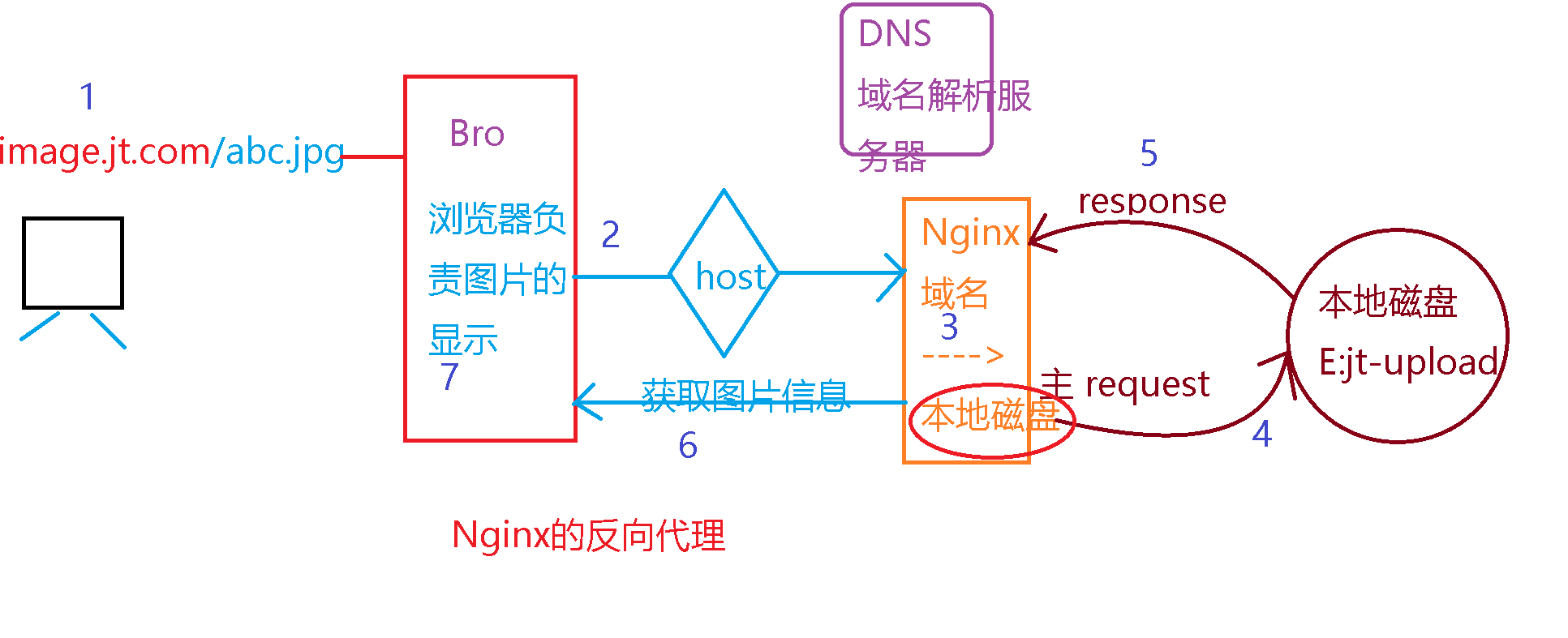
127.0.0.1 order.jt.com

127.0.0.1 solr.jt.com

### 图片回显效果



### Nginx的反向代理



步骤:

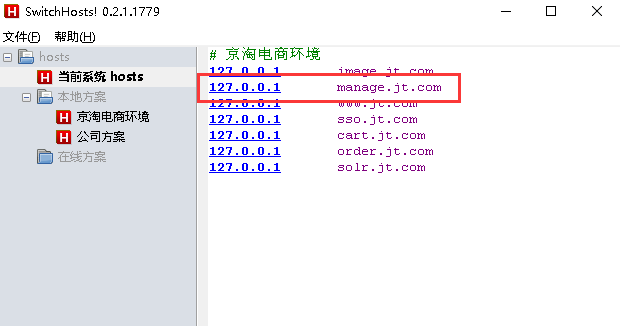
1. 通过通过域名请求获取图片信息 image.jt.com/abc.jpg
2. 通过hosts文件将请求发往本机
3. Nginx通过监听及服务项拦截请求,之后根据配置文件中配置的代理原则.将image.jt.com替换为本地磁盘路径E:jt-upload这样的路径,之后由Nginx实现图片的获取.(反向代理技术)
4. 通过物理的磁盘路径找到请求的图片
5. 将图片的信息进行返回给nginx
6. nginx将图片信息返还给浏览器
7. 浏览器接收到图片信息后进行页面展现.

## Nginx实现请求的代理

### 实现后台项目的代理

说明:实现manage.jt.com的访问跳转到后台服务器中

1. 修改hosts文件



1. 修改nginx文件

#京淘后台管理系统

server {

listen 80;

server\_name manage.jt.com;

#编辑拦截后的任务 转发请求 http://localhost:8091

location / {

#转发请求 协议://ip地址:端口号

proxy\_pass http://localhost:8091;

}

}

**修改完成nginx之后,重启nginx,让配置文件生效.之后测试访问**

### 效果测试



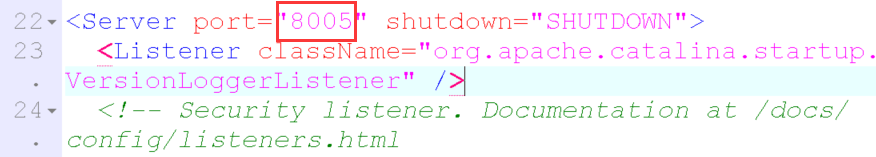
# 搭建tomcat集群

## Tomcat准备

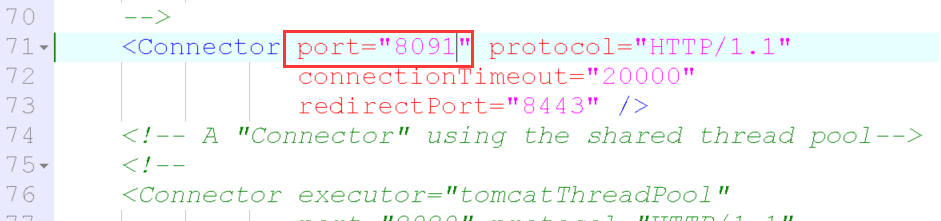
### 复制多台tomcat

说明:根据课前资料中的文件,拷贝多台tomcat,之后修改配置文件

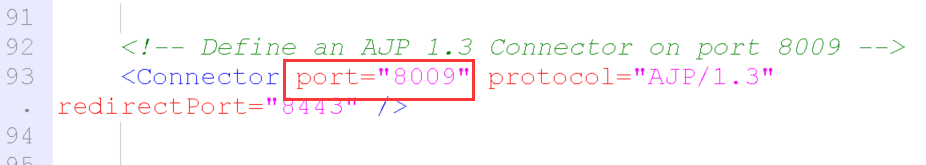
1. 修改tomcat连接apache端口



1. 修改服务端口



1. 修改AJP端口

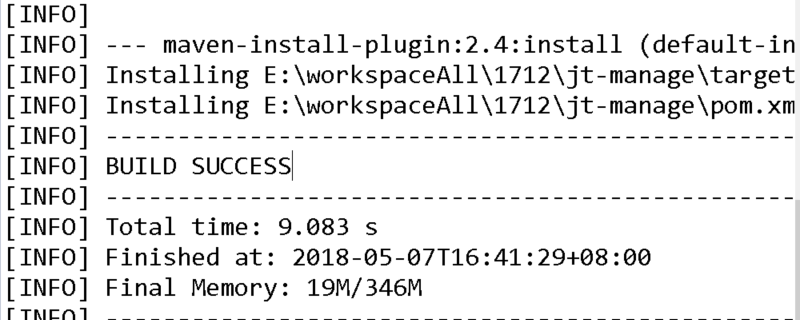


说明:复制3台tomcat.分别修改上述端口 依次加1

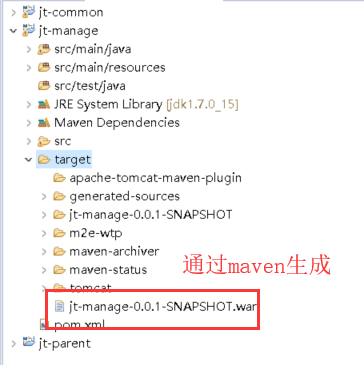
## 项目发布

### 项目打包

说明:通过Maven实现项目的打包



2.生成war包



1. 修改war包的名称

改为ROOT.war,

1. 部署项目

将war包部署到tomcat中webapps下,并且将原来的ROOT文件夹删除，在windos中 暂时不删。

## Nginx实现负载均衡

### 轮询机制

说明:轮询机制根据配置文件的顺序,依次调用tomcat服务

Nginx的配置文件

1.配置轮询



1. 实现配置

