# Fastdfs

## 主要内容

1. fastdfs原理及入门
2. fastdfs在Docker下的安装
3. fastdfs使用springboot完成文件上传下载删除

## 学习目标

|  |  |
| --- | --- |
| 知识点 | 要求 |
| fastdfs原理及入门 | 掌握 |
| fastdfs在Linux下安装 | 自学了解 |
| fastdfs在Docker下的安装 | 掌握 |
| 使用Java代码并测试上传 | 自学了解 |
| fastdfs使用springboot完成文件上传下载删除 | 掌握 |

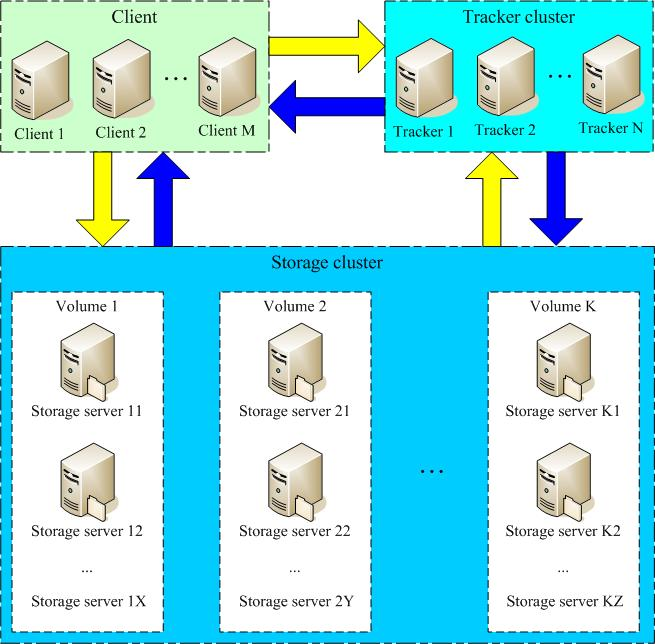
# 【熟悉】FastDFS原理及入门

它就是一个可以存放文件的服务器

千牛云

阿里OSS

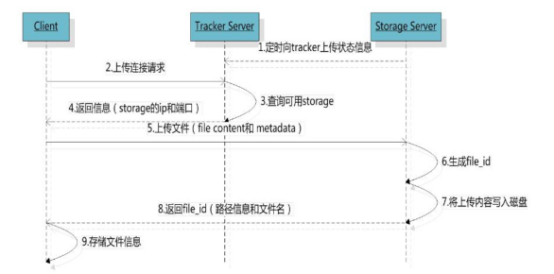
## 系统结构图



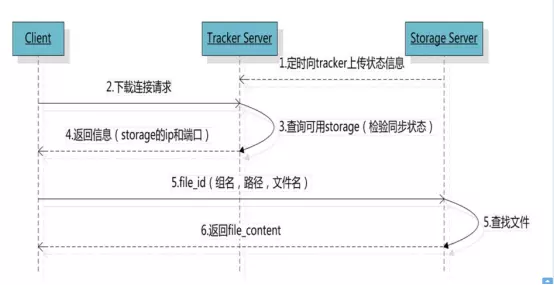
* FastDFS分为Tracker、Storage，其中Storage负责存储文件，Tracker负责存储文件所在地址，主要作用是负载均衡和资源调度。
* Tracker、Storage都可以实现集群部署，Tracker的每个节点地位平等，而Storage可以分为多个组，每个组之间保存的文件是不同的，组内部分为多个成员，每个成员保存的内容是一样，组成员地位一致，没有主从概念。
* 使用FastDFS存储文件优点：可以应对互联网的海量文件存储，一旦文件较多，可以随时横向扩展，且集群的实现也使系统不存在单点故障问题，用户不会因为服务器宕机而无法访问文件资源。

## 工作流程详解

文件上传：Client会先向Tracker询问存储地址，Tracker查询到存储地址后返回给Client，Client拿着地址直接和对应的Storage通讯，将文件上传至改Storage。



文件下载：同样，Client会向Tracker询问地址，并带上要查询的文件名和组名，Tracker查询后会将地址返回给Client，Client拿着地址和指定Storage通讯并下载文件。



# 【自学】Linux下的安装部署fastdfs

## 安装fastDFS需要分别安装如下内容

fastdfs-nginx-module，

fastdfs

nginx

libfastcommon

## 安装gcc(编译时需要)

 yum install -y gcc gcc-c++

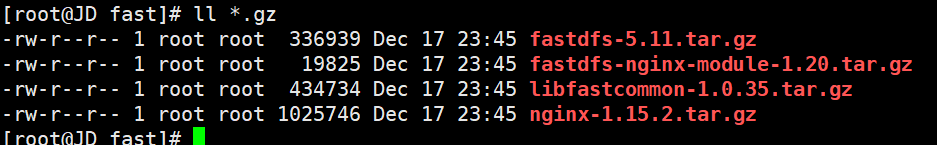
## 安装libevent(运行时需要)

yum -y install libevent

## 安装创建目录上传所有文件

mkdir -p /fileservice/fast

cd /fileservice/fast



## 安装libfastcommon

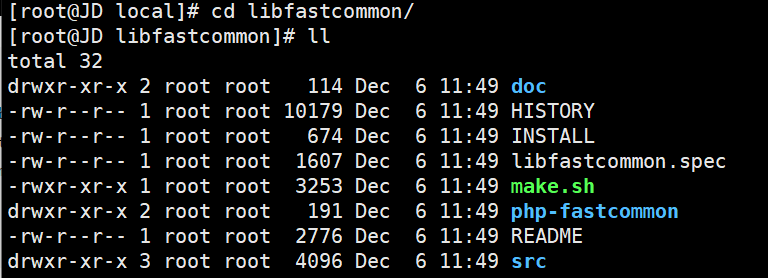
进入fast目录：cd /fileservice/fast

解压文件： tar -zxvf libfastcommon-1.0.35.tar.gz

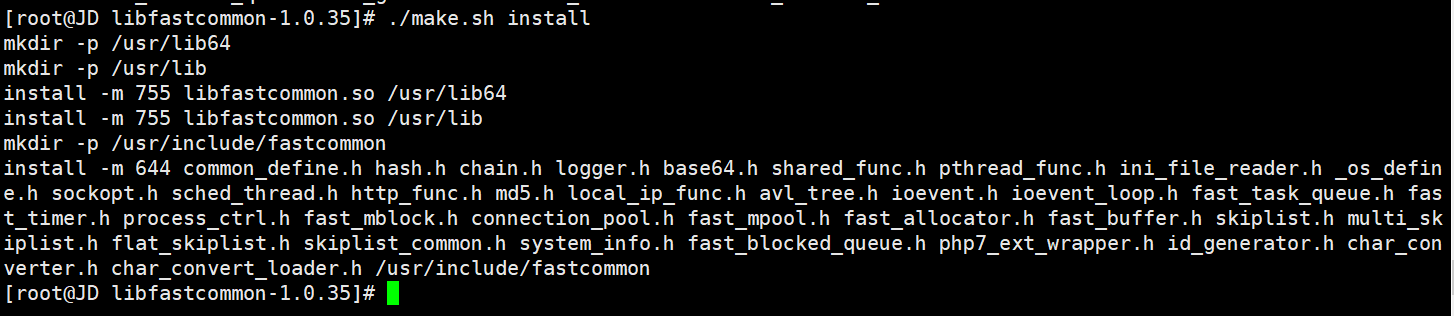
进入libfast文件目录：cd libfastcommon-1.0.35

执行编译：./make.sh

安装：./make.sh install



安装完成之后



## 安装fastdfs

### 下载

<https://sourceforge.net/projects/fastdfs/files/>

网官下载很慢，看我准备的安装文件

### 安装相关依赖库

yum install perl

yum install pcre

yum install pcre-devel

yum install zlib

yum install zlib-devel

yum install openssl

yum install openssl-devel

### 安装fastdfs

进入fast目录：cd /fileservice/fast

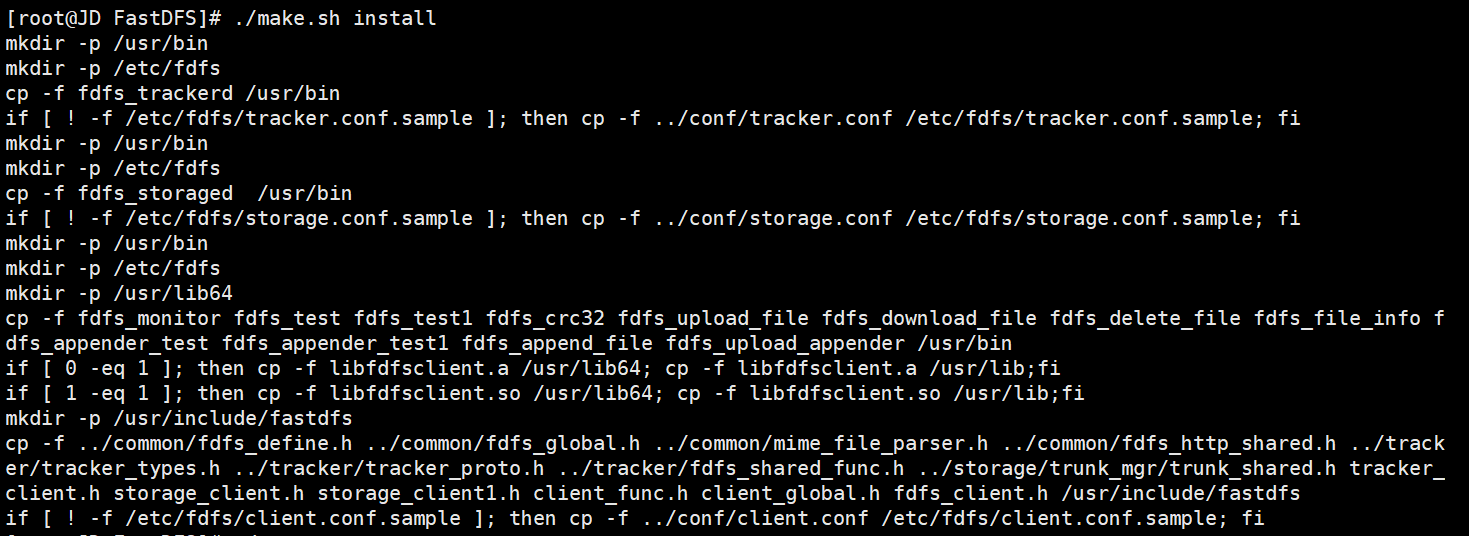
解压文件：tar -zxvf fastdfs-5.11.tar.gz

进入解压后的目录：cd fastdfs-5.11

执行编译：./make.sh

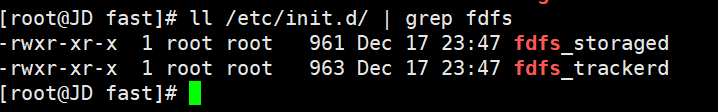
安装：./make.sh install

### 成功之后



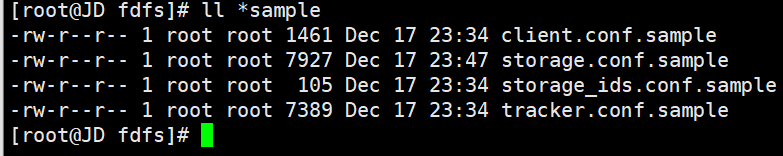
### 查看tracker和storage的可执行脚本(后面有用)

 ll /etc/init.d/ | grep fdfs



### 准备配置文件  默认在/etc/fdfs/下面

cd /etc/fdfs/



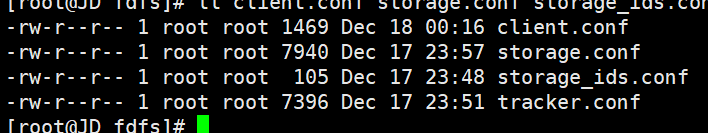
先把配置文件名中的sample去了。[可以复制一份]

cp client.conf.sample client.conf

cp storage.conf.sample storage.conf

cp storage\_ids.conf.sample storage\_ids.conf

cp tracker.conf.sample tracker.conf



然后修改tracker的存放数据和日志的目录。

mkdir -p /home/leige/fastdfs/tracker

## 配置和启动tracker

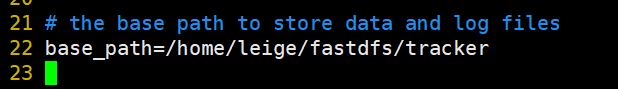
### 切换目录到： /etc/fdfs/ 目录下；

cd /etc/fdfs/

### 修改tracker.conf

vim tracker.conf

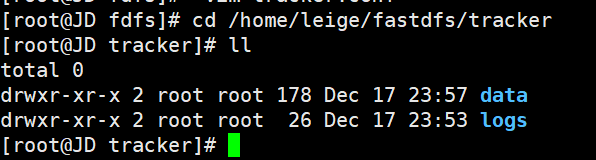
base\_path=/home/yuqing/fastdfs 改为: base\_path=/home/leige/fastdfs/tracker



### 启动tracker，运行如下命令：

service fdfs\_trackerd start

注意：在/home/leige/fastdfs/tracker 目录下生成两个目录， 一个是数据，一个是日志；



## 配置和启动storage

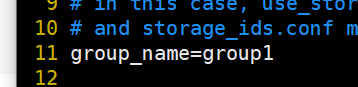
 由于上面已经安装过FastDFS，这里只需要配置storage就好了；

### 切换目录到： /etc/fdfs/ 目录下；

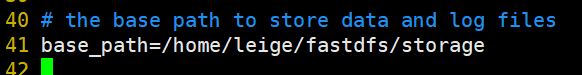
 cd /etc/fdfs/

### 修改storage.conf ; vim storage.conf

group\_name=group1 #配置组名

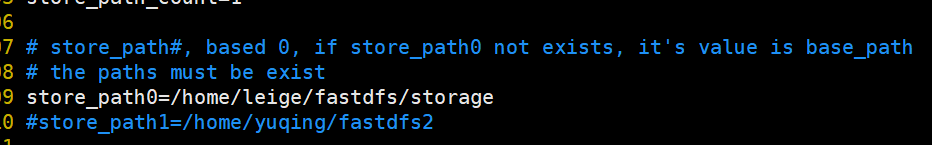


base\_path=/home/yuqing/fastdfs 改为: base\_path=/home/leige/fastdfs/storage



#store存放文件的位置(store\_path)

store\_path0=/home/yuqing/fastdfs 改为： store\_path0=/home/leige/fastdfs/storage



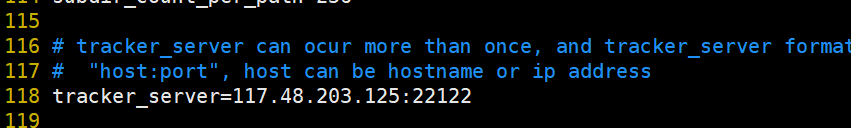
#如果有多个挂载磁盘则定义多个store\_path，如下

#store\_path1=.....

#store\_path2=......

#配置tracker服务器:IP

tracker\_server=117.48.203.125:22122



#如果有多个则配置多个tracker

#tracker\_server=117.48.203.126:22122

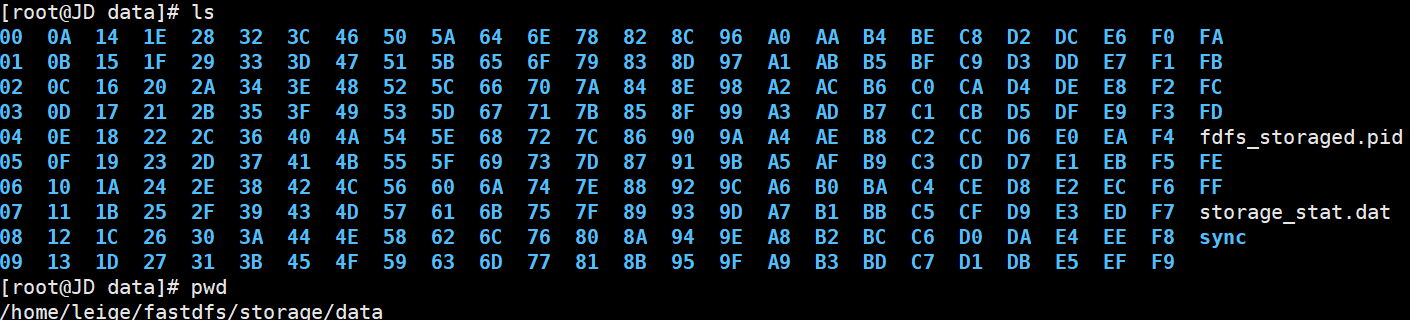
### 创建/home/leige/fastdfs/storage 目录

mkdir -p /home/leige/fastdfs/storage

### 启动storage， 运行命令如下：

service fdfs\_storaged start

启动完成后进入 /home/leige/fastdfs/storage/data 目录下，显示目录如下：



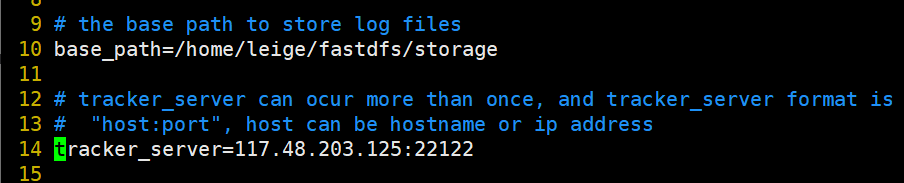
## 使用FastDFS自带工具测试

### 切换目录到 /etc/fdfs/ 目录下；

cd /etc/fdfs/cd

### 修改client.conf ; vim client.conf，

修改基本路径和tracker\_server如下：

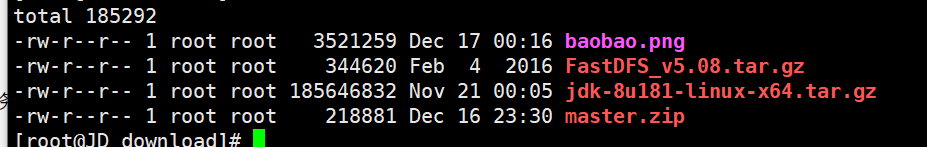


  注意：若tracker有多个，可以配置多个，如下:

　　#tracker\_server=......

　　#tracker\_server=......

### 拷贝一张图片baobao.png到Centos服务器上的 /root/目录下；

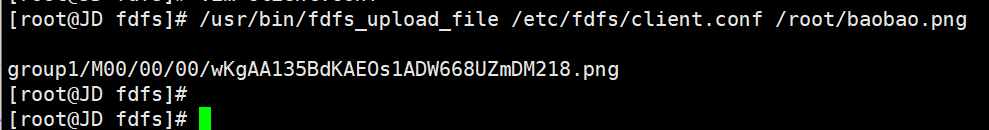


**8.4,进行测试**

运行如下（运行测试程序，读取/etc/fdfs/client.conf 文件，上传/root/目录下的baobao.png文件）

/usr/bin/fdfs\_upload\_file /etc/fdfs/client.conf /root/baobao.png

 结果如下，表示搭建成功；



以上图中的文件地址：http://117.48.203.125/group1/M00/00/00/wKgAA135BdKAEOs1ADW668UZmDM218.png 对应storage服务器上的/home/leige/fastdfs/storage/data/00/00/wKgAA135BdKAEOs1ADW668UZmDM218.png文件；

由于现在还没有和nginx整合无法使用http下载。

## FastDFS 和nginx整合

### 在tracker上安装 nginx

 在每个tracker上安装nginx，的主要目的是做负载均衡及实现高可用。如果只有一台tracker可以不配置nginx。

 一个tracker对应多个storage，通过nginx对storage负载均衡；

### 在storage 上安装nginx

 （1）上传fastdfs-nginx-module-1.20.tar.gz 到Centos服务器上；



 （2）解压fastdfs-nginx-module-1.20.tar.gz 并移动到 /usr/local目录下；

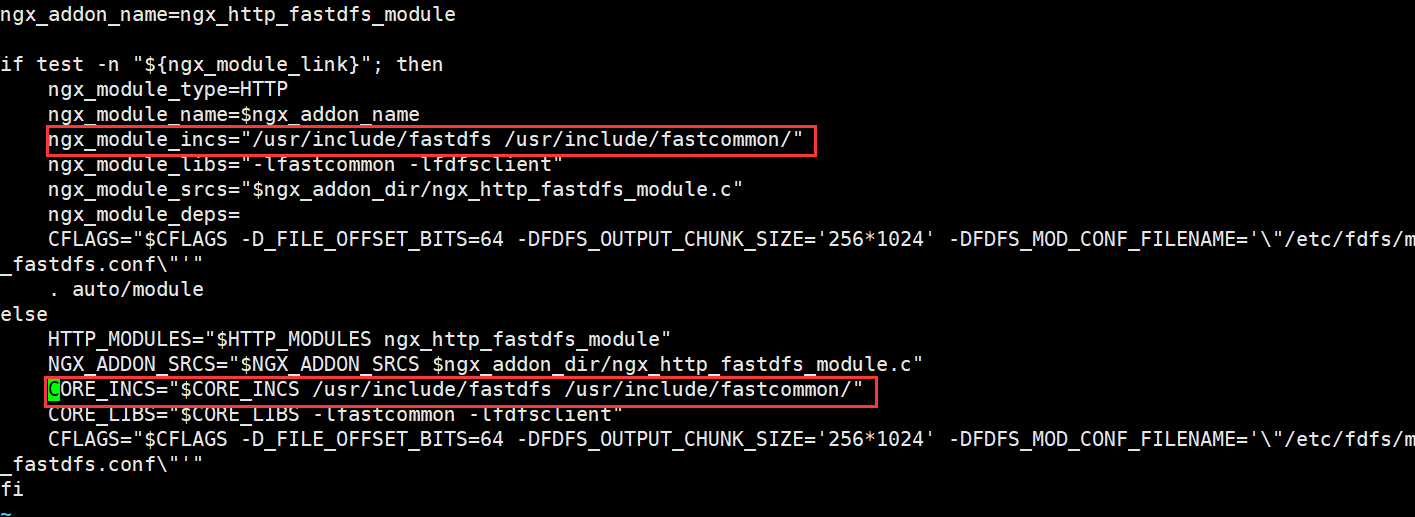
tar -zxvf fastdfs-nginx-module-1.20.tar.gz 解压

（3）切换目录到： fastdfs-nginx-module-1.20/src 目录下

cd fastdfs-nginx-module-1.20/src

（4）修改config文件，将文件中的所有 /usr/local/ 路径改为 /usr/

修改之后



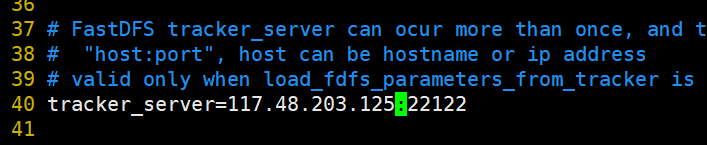
（5）将fastdfs-nginx-module/src下的mod\_fastdfs.conf拷贝至/etc/fdfs/下

cp mod\_fastdfs.conf /etc/fdfs/

（6）并修改 /etc/fdfs/mod\_fastdfs.conf 的内容；

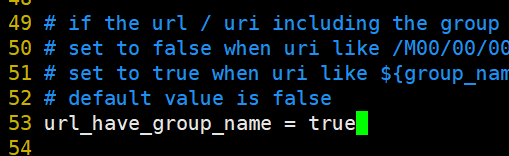
vi /etc/fdfs/mod\_fastdfs.conf

tracker\_server=117.48.203.125:22122

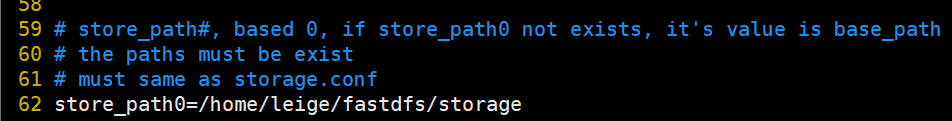


#tracker\_server=192.168.172.20:22122 #(多个tracker配置多行)

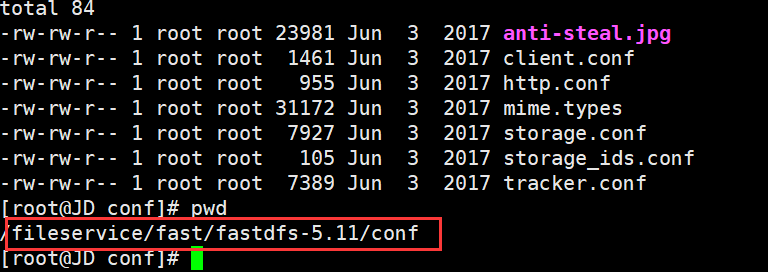
url\_have\_group\_name=true #url中包含group名称



store\_path0=/home/fdfs\_storage #指定文件存储路径（上面配置的store路径）



### 进入之前解压的fastdfs目录下，把http.conf、mime.conf移动至/etc/fdfs



cp http.conf mime.types /etc/fdfs/

## Nginx的安装

### 上传 nginx-1.15.2.tar.gz 到Centos服务器上；



### 解压 nginx-1.15.2.tar.gz

cd /fileservice/fast/

tar -zxvf nginx-1.15.2.tar.gz

### 进入nginx解压的目录下

cd nginx-1.15.2/

### 加入模块命令配置

./configure --prefix=/opt/nginx --sbin-path=/usr/bin/nginx --add-module=/fileservice/fast/fastdfs-nginx-module-1.20/src

### 编译并安装

make && make install

### 修改nginx配置

cd /opt/nginx/conf

vim nginx.conf



### 启动nginx

cd /usr/bin/

./nginx #启动

**11、在浏览器中访问上传到fastDFS的图片**

因为Centos系统有防火墙，需要先关闭掉，才可以在浏览器中访问；

（1）CentOS 7.0默认使用的是firewall作为防火墙；若没有启用iptables 作为防火墙，则使用以下方式关闭防火墙：

systemctl stop firewalld.service #停止firewall

systemctl disable firewalld.service #禁止firewall开机启动

firewall-cmd --state #查看默认防火墙状态（关闭后显示notrunning，开启后显示running）

（2）若已经启用iptables作为防火墙，则使用以下方式关闭：

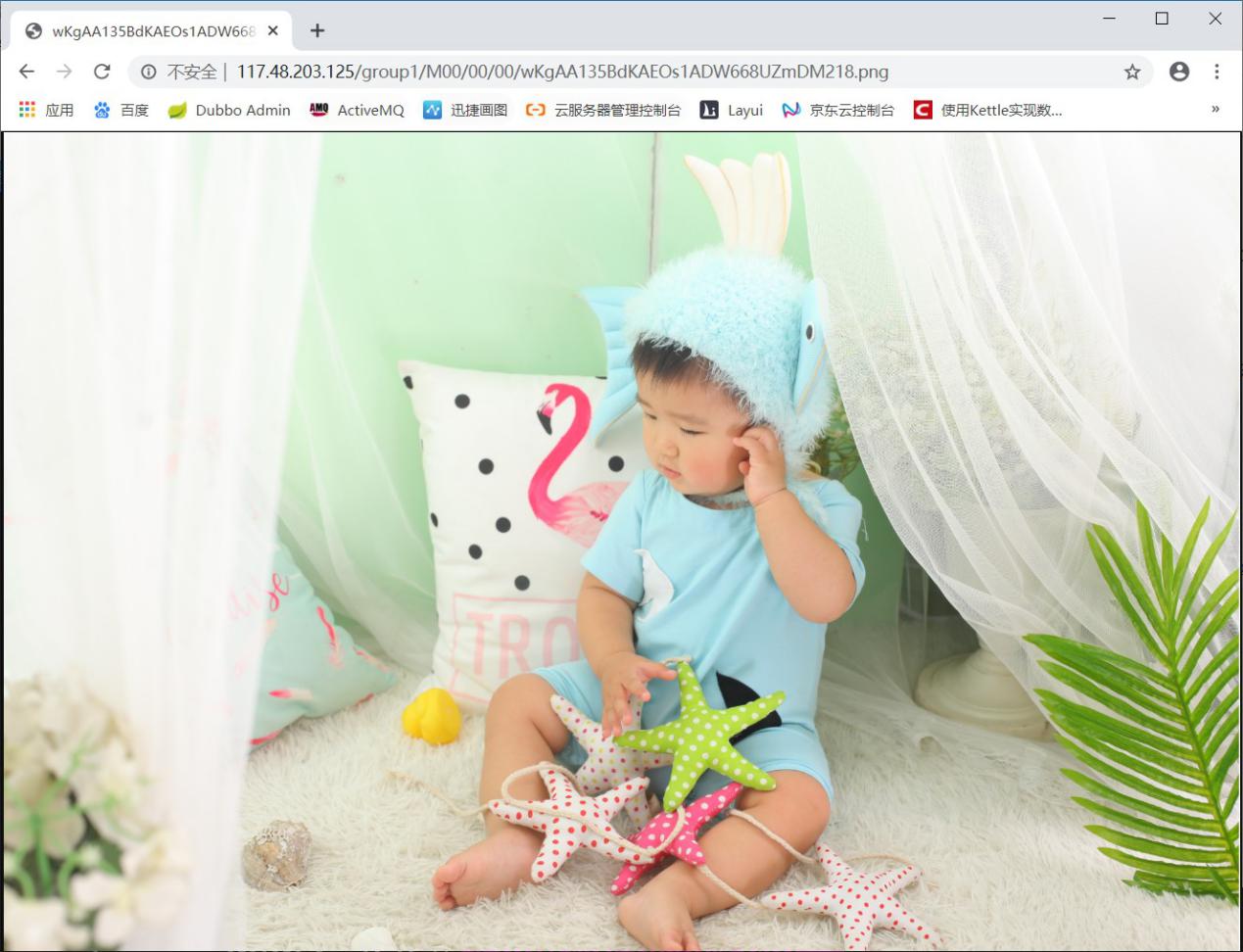
service iptables stop #临时关闭防火墙

chkconfig iptables off #永久关闭防火墙

 （3）在谷歌浏览器中访问刚才上传的图片：

刚才上传的图片地址为：<http://117.48.203.125/group1/M00/00/00/wKgAA135BdKAEOs1ADW668UZmDM218.png>

宝宝镇楼，可爱不



# 【掌握】使用Docker搭建

## 拉取镜像并启动

yum -y install docker

systemctl start docker

systemctl enable docker

docker run -d --restart=always --privileged=true --net=host --name=fastdfs -e IP=47.105.128.151 -e WEB\_PORT=8888 -v ${HOME}/fastdfs:/var/local/fdfs registry.cn-beijing.aliyuncs.com/tianzuo/fastdfs

​

其中-v ${HOME}/fastdfs:/var/local/fdfs是指：将${HOME}/fastdfs这个目录挂载到容器里的/var/local/fdfs这个目录里。所以上传的文件将被持久化到${HOME}/fastdfs/storage/data里，IP 后面是自己的服务器公网ip或者虚拟机ip，-e WEB\_PORT=80 指定nginx端口

## 开放端口



## 测试上传

 //进入容器

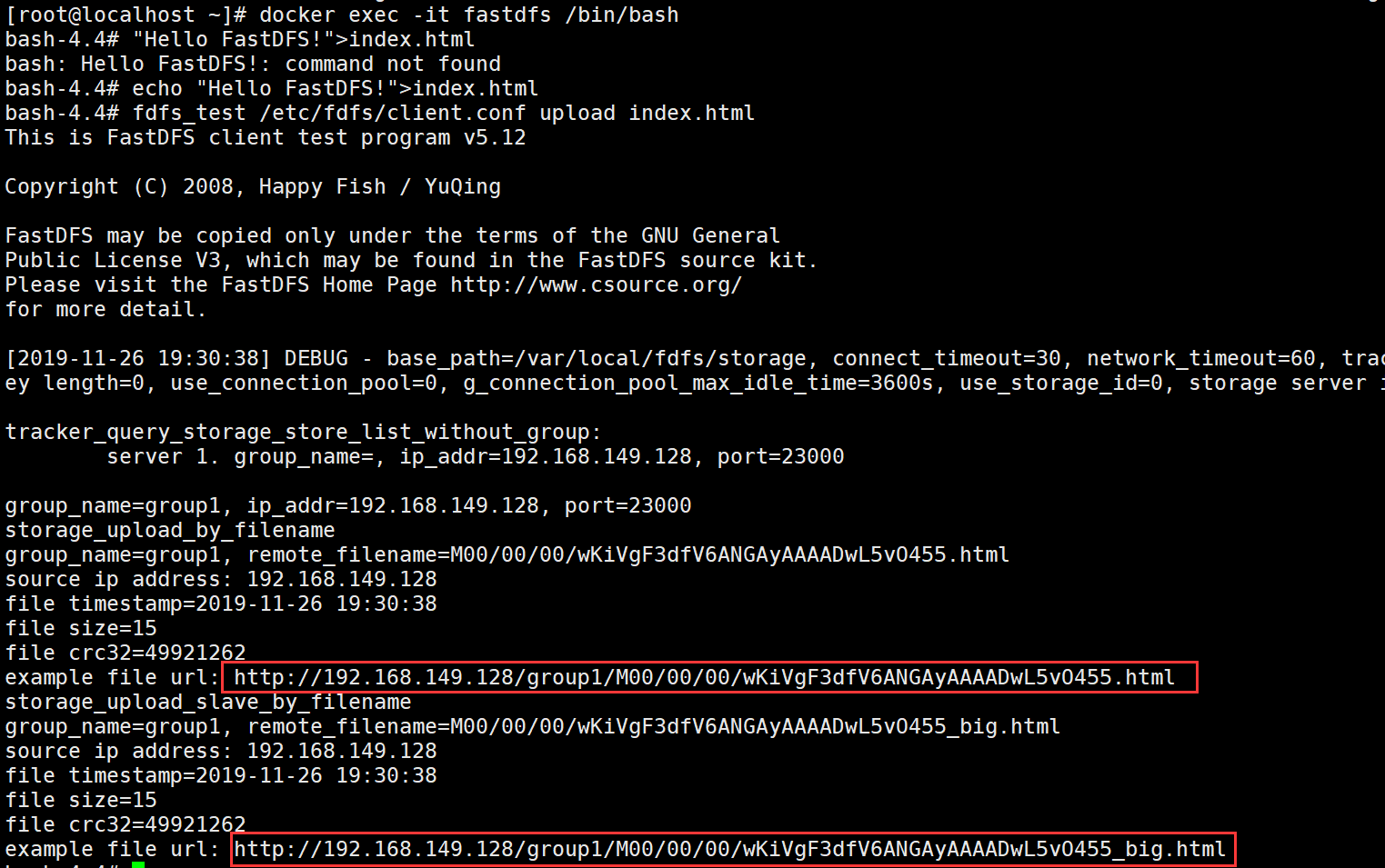
docker exec -it fastdfs /bin/bash

//创建文件

echo "Hello FastDFS!">index.html

//测试文件上传

fdfs\_test /etc/fdfs/client.conf upload index.html



## 配置端口

开放22122 如果这个不开放，远程上传会出现问题

开放8888 nginx对外暴露的端口



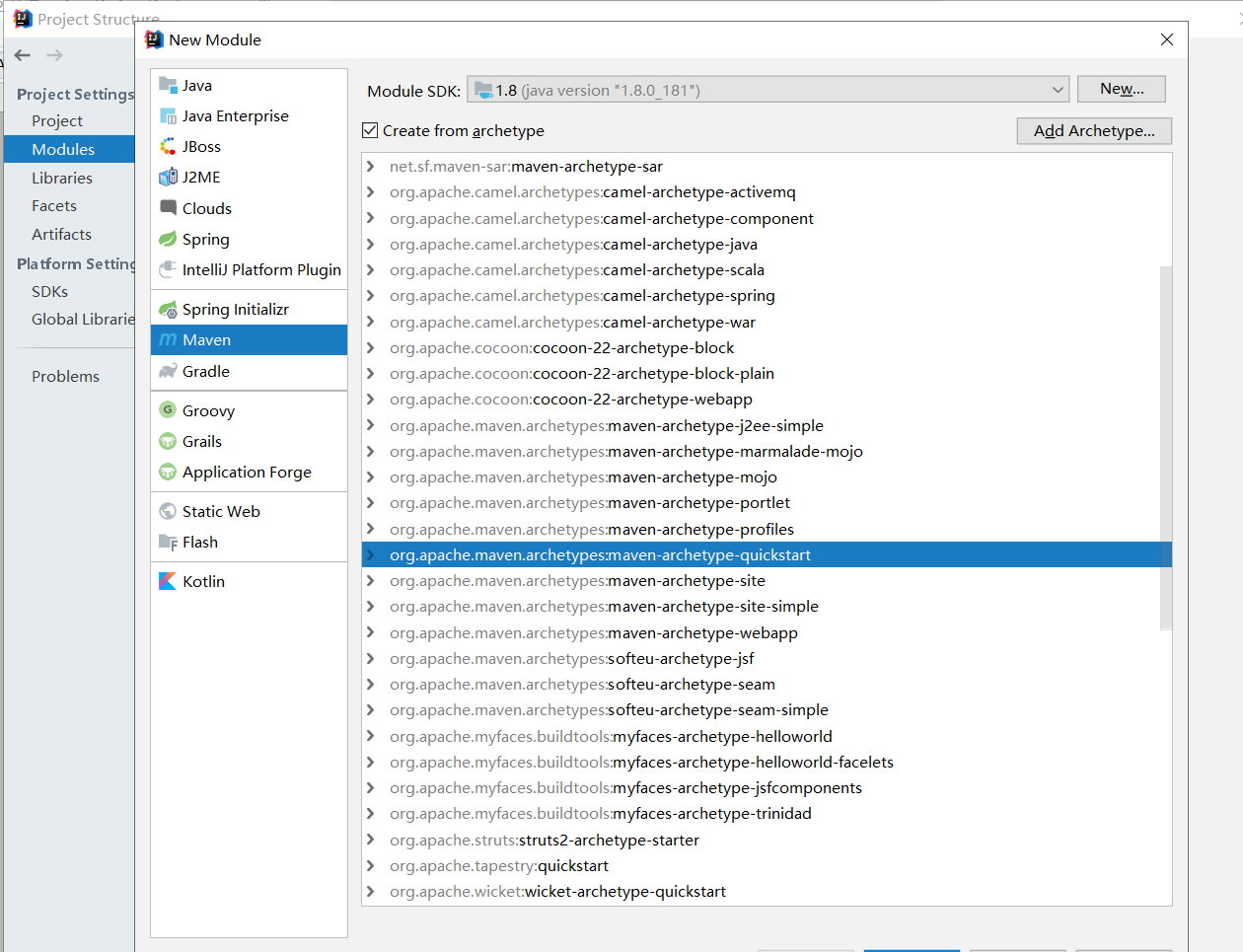
## 测试访问

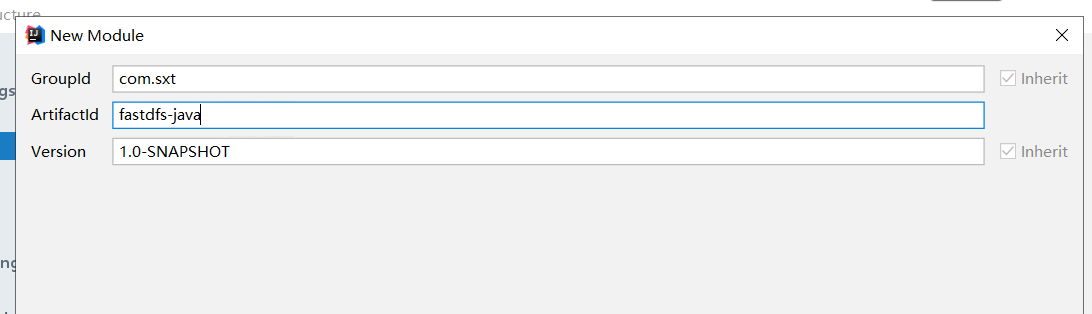
[http://192.168.149.128:8888/group1/M00/00/00/wKiVgF3dfV6ANGAyAAAADwL5vO455\_big.html](http://192.168.149.128/group1/M00/00/00/wKiVgF3dfV6ANGAyAAAADwL5vO455_big.html)



# 【自学】使用Java代码并测试上传

## 创建项目





## 修改pom.xml

|  |
| --- |
| <dependency>  <groupId>net.oschina.zcx7878</groupId>  <artifactId>fastdfs-client-java</artifactId>  <version>1.27.0.0</version>  </dependency>  <!-- spring-core -->  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-core</artifactId>  <version>4.3.25.RELEASE</version>  </dependency> |

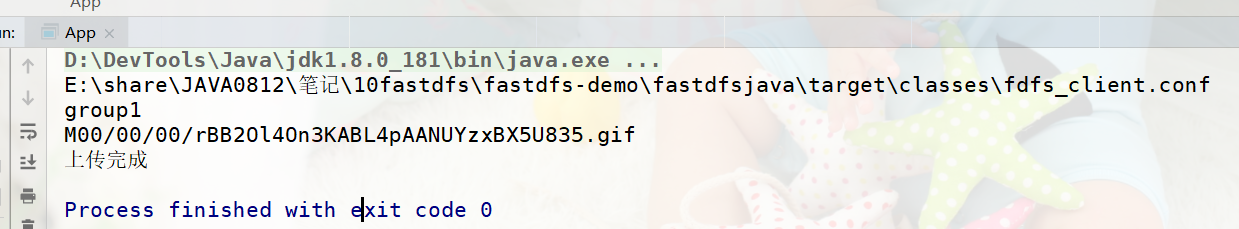
## 创建fdfs\_client.conf

|  |
| --- |
| connect\_timeout=30 network\_timeout=60 base\_path=/home/fastdfs #改为自己服务器的ip tracker\_server=116.62.44.5:22122 log\_level=info use\_connection\_pool = false connection\_pool\_max\_idle\_time = 3600 load\_fdfs\_parameters\_from\_tracker=false use\_storage\_id = false storage\_ids\_filename = storage\_ids.conf http.tracker\_server\_port=80 |

## 创建测试类进行文件上传

|  |
| --- |
| **package** com.sxt; **import** org.csource.common.MyException; **import** org.csource.fastdfs.\*; **import** org.springframework.core.io.ClassPathResource; **import** java.io.IOException; */\*\*  \* Hello world!  \*  \*/* **public class** App  {  **public static void** main( String[] args ) **throws** IOException, MyException {  String uploadFilePath=**"C:/Users/LJH/Pictures/Camera Roll/timg.gif"**;   String filePath = **new** ClassPathResource(**"fdfs\_client.conf"**).getFile().getAbsolutePath();  System.***out***.println(filePath);  *// 1、加载配置文件，配置文件中的内容就是 tracker 服务的地址。* ClientGlobal.*init*(filePath);  *// 2、创建一个 TrackerClient 对象。直接 new 一个。* TrackerClient trackerClient = **new** TrackerClient();  *// 3、使用 TrackerClient 对象创建连接，获得一个 TrackerServer 对象。* TrackerServer trackerServer = trackerClient.getConnection();  *// 4、创建一个 StorageServer 的引用，值为 null* StorageServer storageServer = **null**;  *// 5、创建一个 StorageClient 对象，需要两个参数 TrackerServer 对象、StorageServer 的引用* StorageClient storageClient = **new** StorageClient(trackerServer, storageServer);  *// 6、使用 StorageClient 对象上传图片。  //扩展名不带“.”* String[] strings = storageClient.upload\_file(uploadFilePath, **"gif"**,  **null**);  *// 7、返回数组。包含组名和图片的路径。* **for** (String string : strings) {  System.***out***.println(string);  }  System.***out***.println(**"上传完成"**);  } } |

## 测试



## 封装UploadService

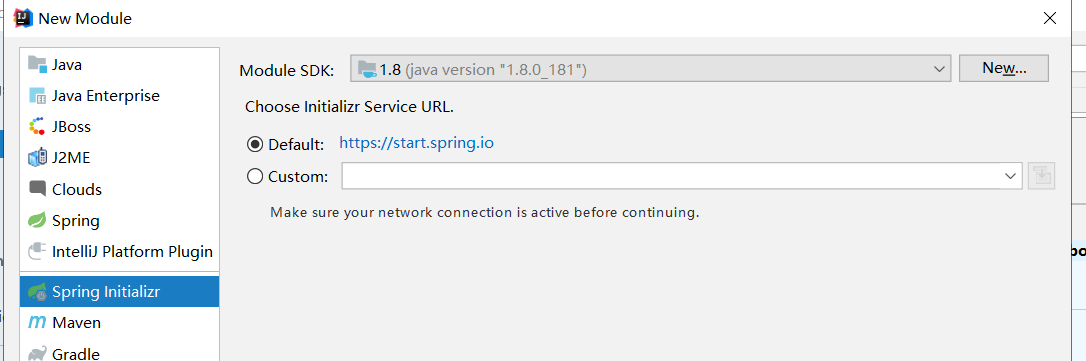
|  |
| --- |
| **package** com.sxt; **import** org.csource.common.MyException; **import** org.csource.fastdfs.\*; **import** org.springframework.core.io.ClassPathResource;  **import** java.io.IOException;  */\*\*  \** ***@Author:*** *尚学堂 雷哥  \*/* **public class** UploadService {   **static** String *configFilePath* =**null**;*//配置文件地址* **static** TrackerClient *trackerClient*=**null**;  **static** TrackerServer *trackerServer*=**null**;  **static** StorageClient *storageClient*=**null**;  **static** {  **try** {  *configFilePath*=**new** ClassPathResource(**"fdfs\_client.conf"**).getFile().getAbsolutePath();  *// 1、加载配置文件，配置文件中的内容就是 tracker 服务的地址。* ClientGlobal.*init*(*configFilePath*);  *// 2、创建一个 TrackerClient 对象。直接 new 一个。  trackerClient* = **new** TrackerClient();  *// 3、使用 TrackerClient 对象创建连接，获得一个 TrackerServer 对象。  trackerServer* = *trackerClient*.getConnection();  *// 4、创建一个 StorageServer 的引用，值为 null* StorageServer storageServer = **null**;  *// 5、创建一个 StorageClient 对象，需要两个参数 TrackerServer 对象、StorageServer 的引用  storageClient* = **new** StorageClient(*trackerServer*, storageServer);  } **catch** (Exception e) {  e.printStackTrace();  }  }   */\*\*  \* 文件上传的方法  \** ***@param filePath*** *文件地址  \** ***@param suffix*** *后缀  \*/* **public static** String upload(String filePath,String suffix){  **try** {  String[] strings = *storageClient*.upload\_file(filePath, suffix,  **null**);  *// 7、返回数组。包含组名和图片的路径。* **for** (String string : strings) {  System.***out***.println(string);  }  System.***out***.println(**"上传完成"**);  **return** strings[0]+**"/"**+strings[1];  } **catch** (Exception e) {  e.printStackTrace();  }  **return null**;  }   */\*\*  \* 文件上传的方法  \** ***@param bytes*** *文件字节数组  \** ***@param suffix*** *后缀  \*/* **public static** String upload(**byte**[] bytes,String suffix){  **try** {  String[] strings = *storageClient*.upload\_file(bytes, suffix,  **null**);  *// 7、返回数组。包含组名和图片的路径。* **for** (String string : strings) {  System.***out***.println(string);  }  System.***out***.println(**"上传完成"**);  **return** strings[0]+**"/"**+strings[1];  } **catch** (Exception e) {  e.printStackTrace();  }  **return null**;  } } |

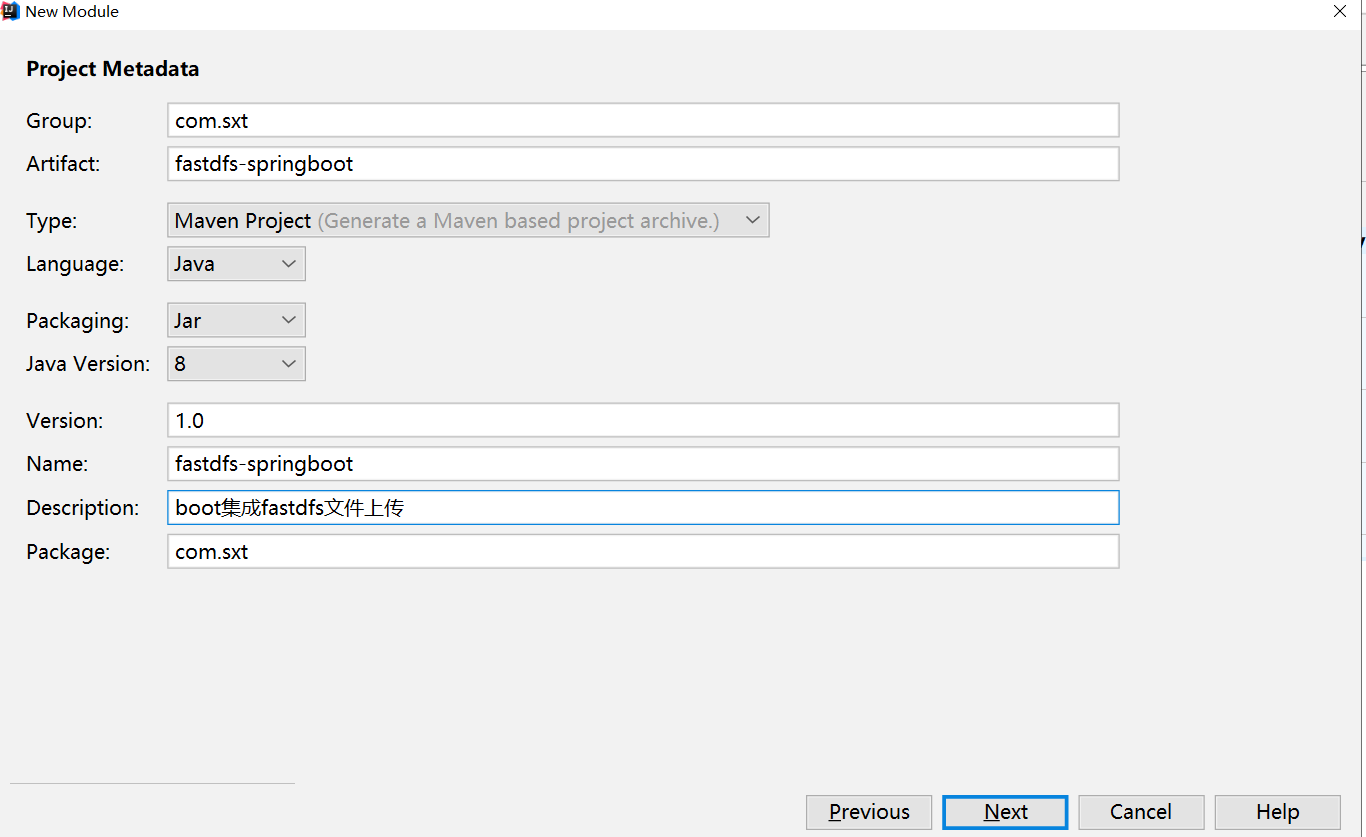
## 测试

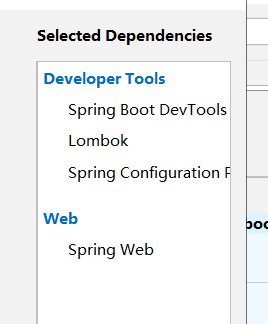
|  |
| --- |
| **package** com.sxt;  **import** org.apache.commons.io.FileUtils; **import** org.csource.fastdfs.\*; **import** org.springframework.core.io.ClassPathResource;  **import** java.io.BufferedInputStream; **import** java.io.ByteArrayInputStream; **import** java.io.File; **import** java.io.FileInputStream;  */\*\*  \** ***@Author:*** *尚学堂 雷哥  \*/* **public class** AppTest {  **public static void** main(String[] args) **throws** Exception {  *//要上传的文件* String uploadFilePath=**"C:/Users/LJH/Pictures/Camera Roll/BG.jpg"**; *// String upload = UploadService.upload(uploadFilePath, "jpg"); // System.out.println(upload);* File file=**new** File(uploadFilePath);  **byte**[] bytes = FileUtils.*readFileToByteArray*(file);  String upload1 = UploadService.*upload*(bytes, **"jpg"**);  String upload2 = UploadService.*upload*(bytes, **"jpg"**);  String upload3 = UploadService.*upload*(bytes, **"jpg"**);  String upload4 = UploadService.*upload*(bytes, **"jpg"**);  System.***out***.println(upload1);  System.***out***.println(upload2);  System.***out***.println(upload3);  System.***out***.println(upload4);  } } |

# 【掌握】使用springboot完成文件上传

## 创建项目







## 修改pom.xml

|  |
| --- |
| *<!-- https://mvnrepository.com/artifact/com.github.tobato/fastdfs-client -->* <**dependency**>  <**groupId**>com.github.tobato</**groupId**>  <**artifactId**>fastdfs-client</**artifactId**>  <**version**>1.27.2</**version**> </**dependency**> |

## 创建配置类UploadProperties

|  |
| --- |
| **package** com.sxt.config; **import** lombok.AllArgsConstructor; **import** lombok.Data; **import** lombok.NoArgsConstructor; **import** org.springframework.boot.context.properties.ConfigurationProperties;  **import** java.util.List;  */\*\*  \** ***@Author:*** *尚学堂 雷哥  \*/* @ConfigurationProperties(prefix = **"upload"**) @Data @AllArgsConstructor @NoArgsConstructor **public class** UploadProperties {  *//访问文件的地址* **private** String **baseUrl**;  *//允许文件上传的类型* **private** List<String> **allowTypes**; } |

## 修改yml文件

|  |
| --- |
| **fdfs**:  **so-timeout**: 2500 *# 读取时间* **connect-timeout**: 600 *# 连接超时时间* **thumb-image**: *# 缩略图* **width**: 100  **height**: 100  **tracker-list**: *# tracker服务配置地址列表* - 47.105.128.151:22122 **upload**:  **base-url**: http://47.105.128.151:8888/  **allow-types**:  - image/jpeg  - image/png  - image/bmp  - image/gif *#文件上传的配置* **spring**:  **servlet**:  **multipart**:  **max-file-size**: 2MB |

## 创建UploadService

|  |
| --- |
| */\*\*  \* Copyright (C),武汉尚学堂 2020-10-31  \* FileName: UploadService  \* Author: 武汉尚学堂【雷哥】  \* Date: 2020/10/31 16:04  \*/* **package** com.bjsxt.config;  **import** com.github.tobato.fastdfs.domain.fdfs.StorePath; **import** com.github.tobato.fastdfs.service.FastFileStorageClient; **import** lombok.AllArgsConstructor; **import** lombok.extern.log4j.Log4j2; **import** org.apache.commons.lang3.StringUtils; **import** org.springframework.beans.factory.annotation.Autowired; **import** org.springframework.boot.context.properties.EnableConfigurationProperties; **import** org.springframework.stereotype.Component; **import** org.springframework.web.multipart.MultipartFile;  **import** javax.imageio.ImageIO; **import** java.awt.image.BufferedImage; **import** java.io.IOException; **import** java.util.HashMap; **import** java.util.Map;  */\*\*  \* <功能简要> <br>  \* <>  \*  \** ***@Author*** *武汉尚学堂【雷哥】   \** ***@createTime*** *2020/10/31 16:04  \** ***@since*** *1.0.0  \*/* @Component @Log4j2 @EnableConfigurationProperties(value = {UploadProperties.**class**}) **public class** UploadService {   @Autowired  **private** UploadProperties **properties**;   @Autowired  **private** FastFileStorageClient **storageClient**;    */\*\*  \* 上传图片  \** ***@param file*** *\** ***@return*** *\*/* **public** Map<String,Object> uploadImage(MultipartFile file) {  *// 1、校验文件类型* String contentType = file.getContentType();  **if** (!**properties**.getAllowTypes().contains(contentType)) {  **throw new** RuntimeException(**"文件类型不支持"**);  }  *// 2、校验文件内容* **try** {  BufferedImage image = ImageIO.*read*(file.getInputStream());  **if** (image == **null** || image.getWidth() == 0 || image.getHeight() == 0) {  **throw new** RuntimeException(**"上传文件有问题"**);  }  } **catch** (IOException e) {  ***log***.error(**"校验文件内容失败....{}"**, e);  **throw new** RuntimeException(**"校验文件内容失败"**+e.getMessage());  }  **return this**.uploadFile(file);   }  */\*\*  \* 上传文件  \** ***@param file*** *\** ***@return*** *\*/* **public** Map<String,Object> uploadFile(MultipartFile file) {  Map<String,Object> map=**new** HashMap<>();  **try** {  *// 3、上传到FastDFS  // 3.1、获取扩展名* String extension = StringUtils.*substringAfterLast*(file.getOriginalFilename(), **"."**);  *// 3.2、上传* StorePath storePath = **storageClient**.uploadFile(file.getInputStream(), file.getSize(), extension, **null**);  *// 返回路径* map.put(**"fileName"**,file.getOriginalFilename()); *//文件名* map.put(**"fullPath"**,**properties**.getBaseUrl() + storePath.getFullPath()); *//全路径* map.put(**"relativePath"**,storePath.getFullPath());*//相对路径* **return** map;  } **catch** (IOException e) {  ***log***.error(**"【文件上传】上传文件失败！....{}"**, e);  **throw new** RuntimeException(**"【文件上传】上传文件失败！"**+e.getMessage());  }  }   */\*\*  \* 从fastdfs里面删除文件  \** ***@param path*** *group1/M00/00/00/rBNPwF9PZDqAaDeDAAF6kn4mN1Y027.jpg  \** ***@return*** *\*/* **public** Map<String,Object> deleteFile(String path) {  Map<String,Object> res=**new** HashMap<>();  **try** {  *// 删除* **storageClient**.deleteFile(path);  res.put(**"code"**,200);  res.put(**"msg"**,**"删除成功"**);  **return** res;  } **catch** (Exception e) {  ***log***.error(**"【文件删除】删除文件失败！....{}"**, e);  res.put(**"code"**,500);  res.put(**"msg"**,**"删除文件失败"**);  **return** res;  }  }      } |

## 创建UploadController

|  |
| --- |
| */\*\*  \* Copyright (C),武汉尚学堂 2020-10-31  \* FileName: UploadController  \* Author: 武汉尚学堂【雷哥】  \* Date: 2020/10/31 16:06  \*/* **package** com.bjsxt.controller;  **import** com.bjsxt.config.UploadService; **import** org.springframework.beans.factory.annotation.Autowired; **import** org.springframework.web.bind.annotation.PostMapping; **import** org.springframework.web.bind.annotation.RequestMapping; **import** org.springframework.web.bind.annotation.RestController; **import** org.springframework.web.multipart.MultipartFile;  **import** java.util.Map;  */\*\*  \* <功能简要> <br>  \* <>  \*  \** ***@Author*** *武汉尚学堂【雷哥】   \** ***@createTime*** *2020/10/31 16:06  \** ***@since*** *1.0.0  \*/* @RestController @RequestMapping(**"upload"**) **public class** UploadController {   @Autowired  **private** UploadService **uploadService**;    */\*\*  \* 上传  \*/* @PostMapping(**"uploadFile"**)  **public** Object uploadFile(MultipartFile mf){  **return uploadService**.uploadImage(mf);  }  */\*\*  \* 删除  \*/* @PostMapping(**"deleteFile"**)  **public** Map<String, Object> deleteFile(String path){  Map<String, Object> map = **uploadService**.deleteFile(path);  **return** map;  }  } |

## 创建index.html测试

|  |
| --- |
| <!DOCTYPE **html**> <**html lang="en"**> <**head**>  <**meta charset="UTF-8"**>  <**title**>文件上传</**title**> </**head**> <**body**> <**h1**>文件上传</**h1**> <**form action="/upload/uploadFile" method="post" enctype="multipart/form-data"**>  <**input type="file" name="mf"**>  <**input type="submit" value="上传"**> </**form**> <**h1**>文件删除</**h1**> <**form action="/upload/deleteFile" method="post"**>  <**input type="text" name="path"**>  <**input type="submit" value="删除"**> </**form**> </**body**> </**html**> |