搭建分布式文件管理系统

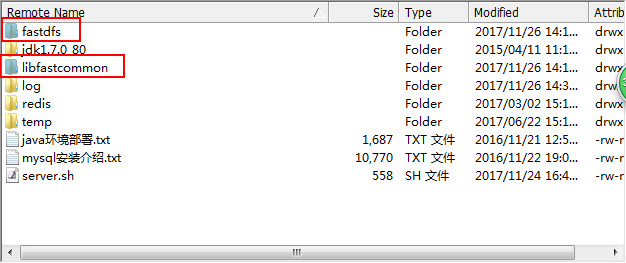
1.下载[fastdfs代码](https://github.com/happyfish100/fastdfs.git" \t "_blank)

* 下载地址：<https://github.com/happyfish100/fastdfs>

2.下载[libfastcommon fastdfs通用函数库](https://github.com/happyfish100/libfastcommon.git" \t "_blank)

* 下载地址：<https://github.com/happyfish100/libfastcommon.git>

3.解压缩拷贝到你准备的文件系统中，我使用的是liunx系统



4.编译安装libfastcommon通用函数库

进入到文件夹下

* cd /root/libfastcommon

进行编译

* ./make.sh

进行安装

* ./make.sh install

5.编译安装FastDFS

进入到文件夹下

* cd /root/fastdfs

进行编译

* ./make.sh

进行安装

* ./make.sh install

6.修改FastDFS配置文件

在修改之前创建一个文件存储目录

* mkdir /root/data

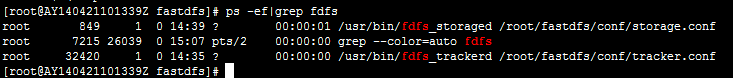
进入 fastdfs/conf/修改配置文件

* tracker.conf 分发 修改 base\_path地址为刚才创建的/root/data文件夹、port:与storage.conf里面的tracker\_server端口一致
* storage.conf 存储 修改 base\_path地址为刚才创建的/root/data文件夹地址 store\_path0为存储地址， tracker\_server为tracker服务器地址，server\_port 为http端口
* client.conf 客户端 修改 base\_path地址为刚才创建的/root/data文件夹地址 tracker\_server为tracker服务器地址

7.启动FastDFS

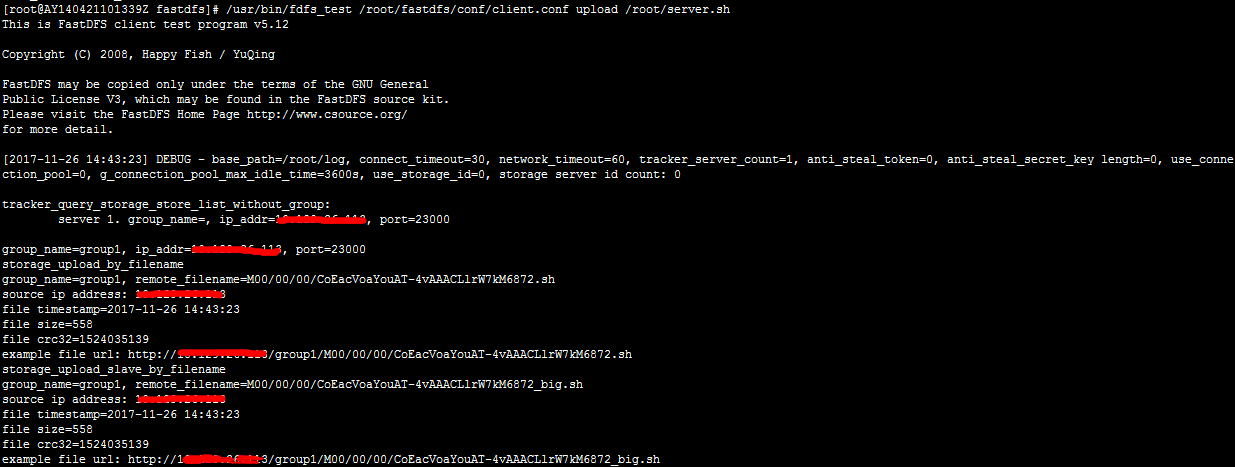
* /usr/bin/fdfs\_trackerd /root/fastdfs/conf/tracker.conf
* /usr/bin/fdfs\_storaged /root/fastdfs/conf/storage.conf
* 查看进程是否启动成功
* 查看服务是否运行

ps -ef | grep fdfs

* 
* 成功

8.测试上传文件，后面是自己要上传的文件

* /usr/bin/fdfs\_test /root/fastdfs/conf/client.conf upload /root/server.sh



全部成功

下面配置nginx服务，可以浏览器查看上传内容

1.下载nginx-1.11.9.tar.gz

解压缩

tar zxf nginx-1.11.9.tar.gz

2.下载**安装fastdfs-nginx-module**

**解压缩**

**tar zxf fastdfs-nginx-module\_v1.16.tar.gz**

3.进入nginx目录

cd nginx-1.11.9

执行

./configure --add-module=/usr/local/src/fastdfs-nginx-module/src

make && make install

4.复制 fastdfs-nginx-module 源码中的配置文件到/etc/fdfs 目录,并拷贝文件到相应目录，进行修改

cp /fastdfs-nginx-module/src/mod\_fastdfs.conf /etc/fdfs/

vi /etc/fdfs/mod\_fastdfs.conf

相应目录改成自己的文件存储目录

connect\_timeout=10

     base\_path=/tmp

     tracker\_server=ip01:22122

     storage\_server\_port=23000

     group\_name=group1

     url\_have\_group\_name = true    //此处一定改为true否则浏览器无法访问到

     store\_path0=/fastdfs/storage

5.复制 FastDFS 的部分配置文件到/etc/fdfs 目录

cd /opt/fastdfs/conf

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

6.在/fastdfs/storage 文件存储目录下创建软连接,将其链接到实际存放数据的目录

ln -s /fastdfs/storage/data/ /fastdfs/storage/data/M00

7.配置nginx文件

http {

    include mime.types;

    default\_type application/octet-stream;

    sendfile on;

    keepalive\_timeout 65;

    server {

        listen 8888;

        server\_name localhost;

        location ~/group([0-9])/M00 {

            ngx\_fastdfs\_module;

        }

        error\_page 500 502 503 504 /50x.html;

        location = /50x.html {

            root html;

        }

    }

}

 A、8888 端口值是要与/etc/fdfs/storage.conf 中的 http.server\_port=8888 相对应, 因为 http.server\_port 默认为 8888,如果想改成 80,则要对应修改过来。  
 B、Storage 对应有多个 group 的情况下,访问路径带 group 名,如/group1/M00/00/00/xxx, 对应的 Nginx 配置为:

location ~/group([0-9])/M00 { ngx\_fastdfs\_module; }

8.启动nginx

/usr/local/nginx/sbin/nginx

# FastDfs使用Docker部署（推荐）

在2017/12/04 20:13上被[亮 刘](http://er.msqltec.com:8080/xwiki/bin/view/XWiki/liuliang)修改

上面讲了安装过程。现在使用我已经做好的docker镜像来安装

1.下载镜像

docker pull hub.c.163.com/liu2160619/fastdfs:latest

2.启动tracker镜像

 docker run -d --name tracker --net=host liu2160619/fastdfs sh tracker.sh

3.启动storage镜像

docker run -d --name storage --net=host -e TRACKER\_IP=<your tracker server address>:22122 -e GROUP\_NAME=<group name> liu2160619/fastdfs sh storage.sh

* 因为使用的网络模式是–net=host，所以此处的<your tracker server address>是你的宿主机的IP
* <group name>是组名，表示Storage的组
* 如果你想再增加一个Storage服务器，就再运行一下命令，换个<group name>

容器中固定端口23000、8080

nginx是8080端口

# 系统默认启动nginx，上传文件成功后直接http://ip:8080/xxxxxxxxx.jpg就可以访问了  文件上传功能开发

在2017/12/05 11:30上被[亮 刘](http://er.msqltec.com:8080/xwiki/bin/view/XWiki/liuliang)修改

使用bootstrap-fileinput/fileinput.js配合FastDfsClient进行文件上传

页面引入

<script src="/resources/sanhu-admin/plugins/bootstrap-fileinput/fileinput.js"/>

<script src="/resources/sanhu-admin/plugins/bootstrap-fileinput/fileinput\_locale\_zh.js"/>

创建一个

<input id="fileupload" type="file" name="files" multiple>

js初始化

//初始化上传控件

$("#fileupload").fileinput({

showPreview: false,

uploadUrl:'/manage/information/informationList/create',

uploadAsync: true, //默认异步上传

showUpload:false, //是否显示上传按钮

showRemove :false, //显示移除按钮

showPreview :true, //是否显示预览

showCaption:false,//是否显示标题

dropZoneEnabled:false,//是否显示拖拽区域

maxFileCount:1,//表示允许同时上传的最大文件个数

overwriteInitial:false,//覆盖初始

initialPreviewShowDelete:false,//初始预览显示删除

elErrorContainer: '#kartik-file-errors',

allowedFileExtensions: ["jpg", "png", "gif"],

enctype:'multipart/form-data',

layoutTemplates:{actionUpload:''},//去除缩略图上面的上传按钮

uploadExtraData: function(previewId, index) { //额外参数的关键点

var params = $("#createForm").serializeArray();

var values = {};

for (var item in params) {

values[params[item].name] = params[item].value;

}

values['informationContent']=UE.getEditor('informationContent').getContent();

console.log(values);

return values;

}

}).on("filebatchselected", function(event, files) {

}).on("fileuploaded", function (event, data, previewId, index) {

//异步上传成功回调

//后台返回数据

var response=data.response;

var code = response.code;

var message = response.message;

var responseData = response.data;

if(code==10001){

createDialog.close();

$('#tableinfo').bootstrapTable('refresh');

}else{

$.confirm({

theme: 'dark',

animation: 'rotateX',

closeAnimation: 'rotateX',

title: false,

content: message,

buttons: {

confirm: {

text: '确认',

btnClass: 'waves-effect waves-button waves-light'

}

}

});

}

}).on('fileerror', function(event, data, msg) {

//异步上传失败回调

}).on("filebatchselected", function(event, files) {

//选择文件后处理事件

}).on("filesuccessremove", function (event, data, previewId, index) {

//删除 文件回调

});

java后台编写

pom.xml引入

<dependency>

<groupId>com.sanhu</groupId>

<artifactId>sanhu-oss-sdk</artifactId>

<version>1.0.0</version>

</dependency>

此类是我公司自己封装的类，在 fastdfsClient基础上进行的封装

创建配置文件fastdfsClient.properties，增加内容

##########链接地址###############

tracker\_server=${tracker\_server}

########### 缩略图配置 #################

#图片宽度

width=350

#图片高度

height=350

#前缀名

cachedPrefixName=\_40

########### 表示文件Web服务器对象,由Nginx服务器承担此角色，通常配置以后就不会再改变 #################

webServerUrl=${webServerUrl}

########### 表示文件Web服务器对象,由Nginx服务器承担此角色，通常配置以后就不会再改变 #################

### 读取时间 ##

soTimeout=600000

### 连接超时时间 ##

connectTimeout=600000

### 字符集 ##

charset=UTF-8

### 设置默认字符集 ##

charsetName=UTF-8

进行spring注入，创建文件applicationContext-fastdfs.xml

写入配置

<bean id="trackerConnectionManager" class="com.sanhu.fastdfs.conn.TrackerConnectionManager">

<property name="trackerList">

<list>

<value>${tracker\_server}</value>

</list>

</property>

</bean>

直接使用

FastDfsUtils.java工具可以实现简单的图片和文件上传

附件提供controller和两个工具类