DFS分布式文件系统

目录

[小应用文件存储方案 3](#_Toc30963)

[工具类 3](#_Toc6952)

[存放在webapp目录下面 5](#_Toc446)

[存放在target目录下面 5](#_Toc30667)

[分布式文件存储系统 6](#_Toc9190)

[环境准备 6](#_Toc4543)

[使用的系统软件 6](#_Toc23274)

[编译环境 6](#_Toc19382)

[磁盘目录 6](#_Toc18203)

[安装libfatscommon 6](#_Toc4992)

[安装FastDFS 7](#_Toc15044)

[安装fastdfs-nginx-module 7](#_Toc1806)

[安装nginx 7](#_Toc26013)

[安装FastDHT 7](#_Toc8859)

[单机部署 11](#_Toc14540)

[tracker配置 11](#_Toc18726)

[storage配置 11](#_Toc12293)

[tracker服务和storage服务启动 11](#_Toc16791)

[client测试 12](#_Toc20610)

[配置nginx访问 13](#_Toc22095)

[分布式部署方案 13](#_Toc29934)

[Nginx的安装 15](#_Toc29169)

[SpringBoot结合分布式文件系统FastDFS实现文件上传功能 18](#_Toc30072)

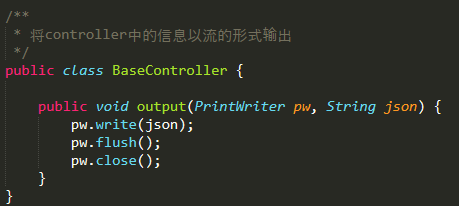
[简介 18](#_Toc11166)

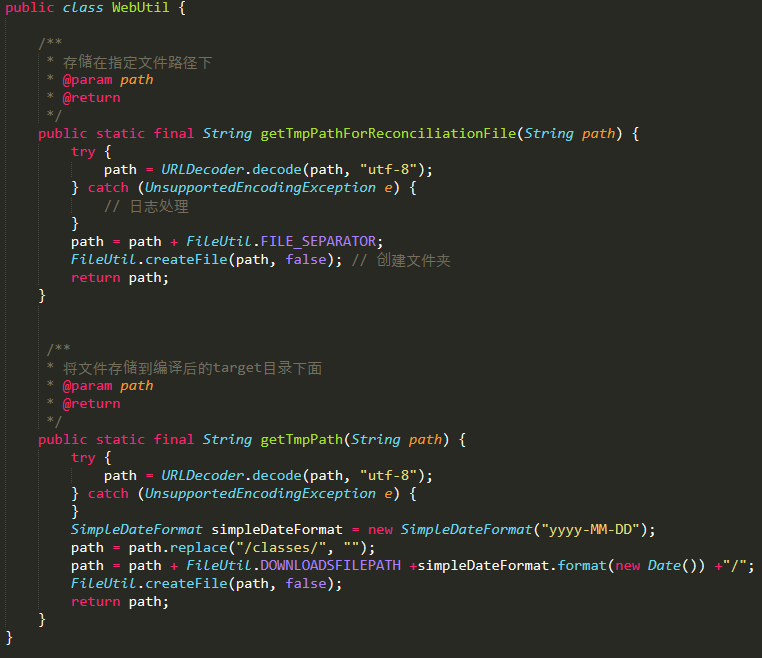
# 小应用文件存储方案

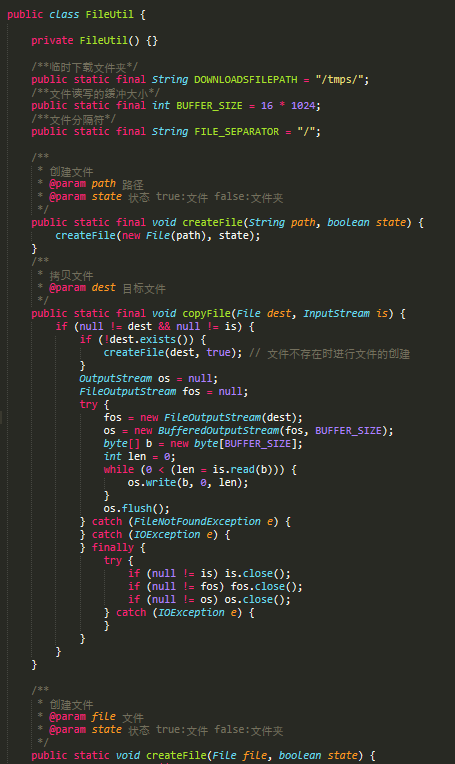
在开发中上传文件存储是个问题，而一般小应用开发会选择服务器本地目录做为存储地址，通过保存文件的相对路径来获取文件，在应用不要求高可用的情况下这是一种普遍的方案。

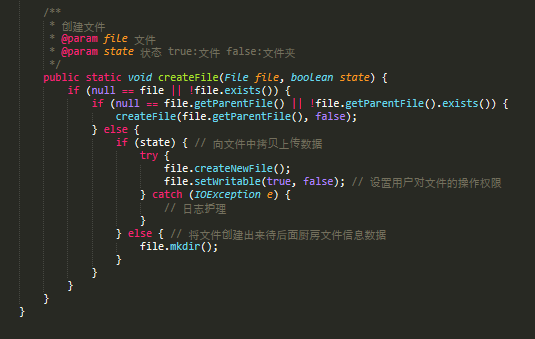
问题: 在每次更新Java版本时，都需要将文件目录做一次备份，不然直接将打包好的war包替换原来的版本时可能会丢失文件（有时可能会直接删除原来的版本目录然后在将新版本放到tomcat的webapps目录）。

## 工具类



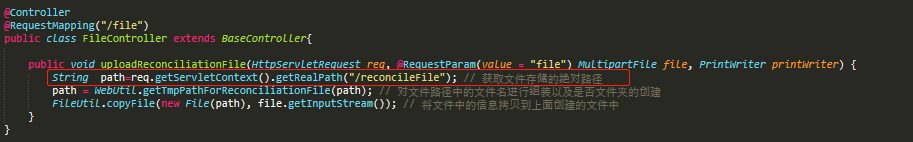




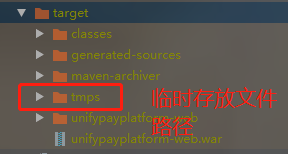


## 存放在webapp目录下面





## **存放在target目录下面**

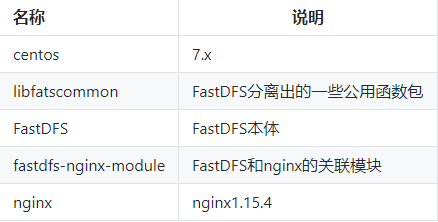




# **分布式文件存储系统**

## **环境准备**

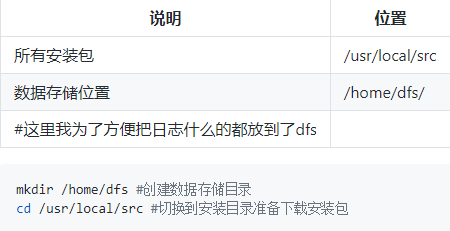
### 使用的系统软件



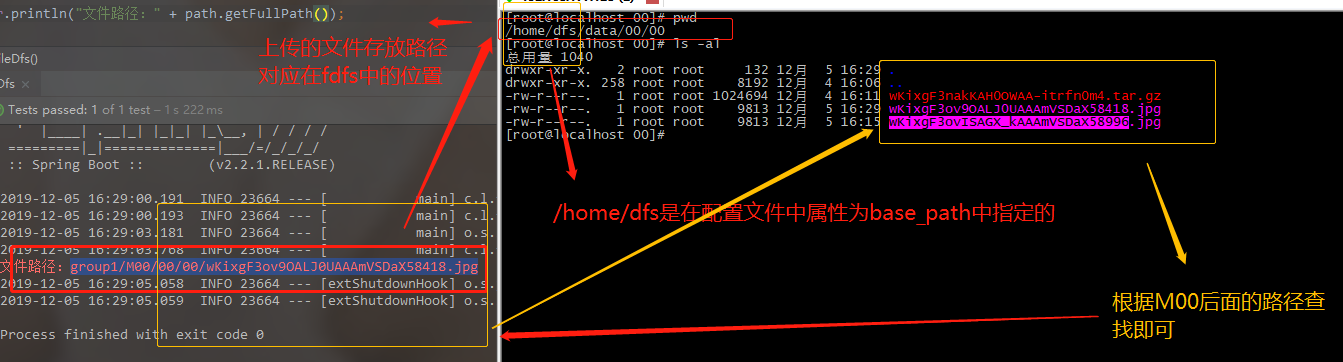
### **编译环境**



### **磁盘目录**



数据上传后文件存放在fdfs中的位置如下图:



### **安装libfatscommon**

git clone https://github.com/happyfish100/libfastcommon.git --depth 1

cd libfastcommon/

./make.sh && ./make.sh install #编译安装

### **安装FastDFS**

cd ../ #返回上一级目录

git clone https://github.com/happyfish100/fastdfs.git --depth 1

cd fastdfs/

./make.sh && ./make.sh install #编译安装#配置文件准备

cp /etc/fdfs/tracker.conf.sample /etc/fdfs/tracker.conf

cp /etc/fdfs/storage.conf.sample /etc/fdfs/storage.conf

cp /etc/fdfs/client.conf.sample /etc/fdfs/client.conf #客户端文件，测试用

cp /usr/local/src/fastdfs/conf/http.conf /etc/fdfs/ #供nginx访问使用

cp /usr/local/src/fastdfs/conf/mime.types /etc/fdfs/ #供nginx访问使用

### **安装fastdfs-nginx-module**

cd ../ #返回上一级目录

git clone https://github.com/happyfish100/fastdfs-nginx-module.git --depth 1

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

### **安装nginx**

wget http://nginx.org/download/nginx-1.15.4.tar.gz #下载nginx压缩包

tar -zxvf nginx-1.15.4.tar.gz #解压

cd nginx-1.15.4/

#添加fastdfs-nginx-module模块

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

make && make install #编译安装

### **安装FastDHT**

#### 简介

由于FastDFS本身不能对重复上传的文件进行去重，而FastDHT可以做到去重。FastDHT是一个高性能的分布式哈希系统，它是基于键值对存储的，而且它需要依赖于Berkeley DB作为数据存储的媒介，同时需要依赖于libfastcommon。

#### 安装Berkley db-6.2.23

cd /usr/local/src

wget http://download.oracle.com/berkeley-db/db-6.2.23.tar.gz

tar -zxvf db-6.2.23.tar.gz

cd db-6.2.23

./dist/configure --prefix=/usr/local/db

Make && make install

#### 安装FastDHT

cd /usr/local/src

git clone https://github.com/happyfish100/fastdht.gitcd fastdht#修改make.sh

vim make.sh

CFLAGS='-Wall -D\_FILE\_OFFSET\_BITS=64 -D\_GNU\_SOURCE'#改为:

CFLAGS='-Wall -D\_FILE\_OFFSET\_BITS=64 -D\_GNU\_SOURCE -I/usr/local/db/include/ -L/usr/local/db/lib/'

ESC

:wq

./make.sh

./make.sh install

#### 配置文件

mkdir /home/fdht

# 如没有/etc/fdht目录及目录下文件

mkdir /etc/fdht

cp /root/fastdht/conf/fdht\_client.conf /etc/fdht

cp /root/fastdht/conf/fdhtd.conf /etc/fdht

cp /root/fastdht/conf/fdht\_servers.conf /etc/fdht

#### 配置fdht\_client.conf

vim /etc/fdht/fdht\_client.conf

#本选项关联 storaged.conf文件

[keep\_alive=1](#_配置storaged.conf文件)

base\_path=/data/fdht#include /etc/fdht/fdht\_servers.conf

ESC

:wq

#### 配置storaged.conf文件

vim /etc/fdfs/storage.conf

#是否检测上传文件已经存在。如果已经存在，则建立一个索引链接以节省磁盘空间

check\_file\_duplicate=1

#当上个参数设定为1时 ， 在FastDHT中的命名空间

key\_namespace=FastDFS

#长连接配置选项，如果为0则为短连接 1为长连接

keep\_alive=1

#此处特别需要注意配置

#include /etc/fdht/fdht\_servers.conf

#### 配置fdht\_servers.conf（11411）

vim /etc/fdht/fdht\_servers.conf

group\_count=1

group0 = 192.168.177.128:11411

ESC

:wq

#### 配置fdhtd.conf

vim /etc/fdht/fdhtd.conf

bind\_addr=192.168.177.128

port=11411

base\_path=/home/fdht

cache\_size=64MB

#include /etc/fdht/fdht\_servers.conf

ESC

:wq

#### 引入libdb.so

ln -s /usr/local/db/lib/libdb-6.2.so /usr/lib/libdb-6.2.so

ln -s /usr/local/db/lib/libdb-6.2.so /usr/lib64/libdb-6.2.so

#### 开启11411端口

vim /etc/sysconfig/iptables-config

-A INPUT -m state --state NEW -m tcp -p tcp --dport 11411 -j ACCEPT

#### 启动FastDHT

/usr/local/bin/fdhtd /etc/fdht/fdhtd.conf

#### 开机启动

vim /etc/rc.local

/usr/local/bin/fdhtd /etc/fdht/fdhtd.conf

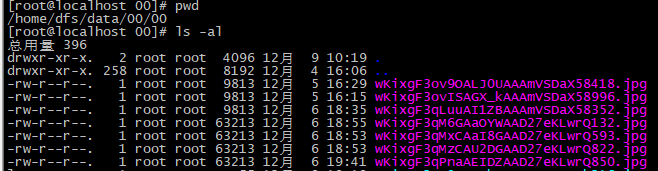
ESC

:wq

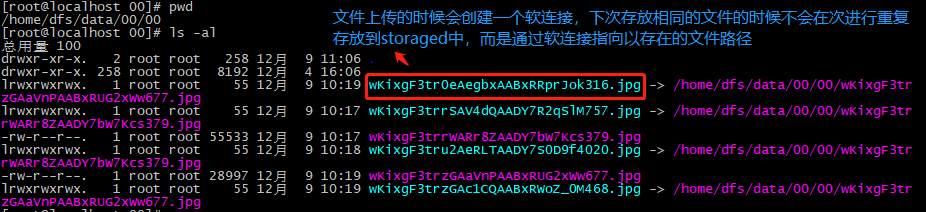
chmod +x /etc/rc.local

#### 验证效果

该效果图是在fdfs安装好的情况下:



下面图示为安装fdfs和fdht的情况下:



[Nginx的安装](#_Nginx的安装)

## **单机部署**

### **tracker配置**

#服务器ip比如为192.168.52.1(虚拟机的IP地址)

vim /etc/fdfs/tracker.conf

#需要修改的内容如下

port=22122 # tracker服务器端口（默认22122,一般不修改）

base\_path=/home/dfs # 存储日志和数据的根目录

### **storage配置**

vim /etc/fdfs/storage.conf

#需要修改的内容如下

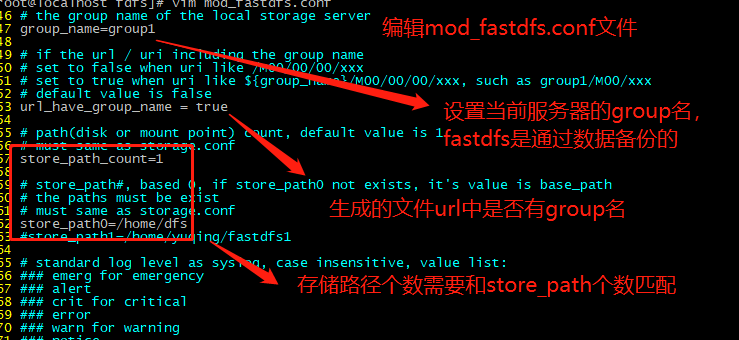
port=23000 # storage服务端口（默认23000,一般不修改）

base\_path=/home/dfs # 数据和日志文件存储根目录

store\_path0=/home/dfs # 第一个存储目录

tracker\_server=192.168.52.1:22122 # tracker服务器IP和端口（tracker追踪服务可以通过该设置找到该storage存储服务）

http.server\_port=8888 # http访问文件的端口(默认8888,看情况修改,和nginx中保持一致)



### **tracker服务和storage服务启动**

具体可以参考网站:

<https://blog.csdn.net/yelllowcong/article/details/78300037>

#### 前提开发tracker/storage服务器端口

vim /etc/sysconfig/iptables-config

#添加规则链路fastdfs tracker

-A INPUT -p tcp -m tcp --dport 22122 -m state --state NEW,ESTABLISHED -j ACCEPT

#fastdfs storage

-A INPUT -p tcp -m tcp --dport 23000 -m state --state NEW,ESTABLISHED -j ACCEPT

#fastdfs web

-A INPUT -p tcp -m tcp --dport 8888 -m state --state NEW,ESTABLISHED -j ACCEPT

#重启防火墙

service iptables restart （CentOS 6.x使用）

systemctl signal firewalld（CenOS 7使用）

#### **启动tracker**

#需要使用绝对路径

/etc/init.d/fdfs\_trackerd start

#### **storage服务启动**

#启动服务

/etc/init.d/fdfs\_storaged start

### **client测试**

vim /etc/fdfs/client.conf#需要修改的内容如下

base\_path=/home/dfs

tracker\_server=192.168.52.1:22122

#tracker服务器IP和端口#保存后测试,返回ID表示成功 如：group1/M00/00/00/xx.tar.gz

/usr/bin/fdfs\_upload\_file /etc/fdfs/client.conf /usr/local/src/nginx-1.15.4.tar.gz

Ps:正确退出Fast DFS:



### **配置nginx访问**

vim /etc/fdfs/mod\_fastdfs.conf

#需要修改的内容如下

tracker\_server=192.168.52.1:22122 #tracker服务器IP和端口

url\_have\_group\_name=true

store\_path0=/home/dfs

#配置nginx.config

vim /usr/local/nginx/conf/nginx.conf

#添加如下配置

server {

listen 8888; ## 该端口为storage.conf中的http.server\_port相同

server\_name 192.168.52.1;

location ~/group[0-9]/ {

ngx\_fastdfs\_module;

}

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

location = /50x.html {

root html;

}

}

#测试下载，用外部浏览器访问刚才已传过的nginx安装包,引用返回的ID

http://192.168.52.1:8888/group1/M00/00/00/wKgAQ1pysxmAaqhAAA76tz-dVgg.tar.gz

#弹出下载单机部署全部跑通

## **分布式部署方案**

主要就是通过在每个服务器上安装的storage的配置文件中添加所有tracker\_server地址

详细搭建可以见：

[https://github.com/happyfish100/fastdfs/wiki#%E5%88%86%E5%B8%83%E5%BC%8F%E9%83%A8%E7%BD%B2](https://github.com/happyfish100/fastdfs/wiki" \l "%E5%88%86%E5%B8%83%E5%BC%8F%E9%83%A8%E7%BD%B2)

在linux中，可以通过管道指令查询比如/usr/bin目录下面的指令



# **Nginx的安装**

下载：wget http://nginx.org/download/nginx-1.15.4.tar.gz

各种后缀的压缩包的解压方式：https://wenku.baidu.com/view/5a71149416fc700abb68fce7.html

解压：tar -xzvf 压缩包[nginx] -C /usr/local/[指定解压到位置]

// \*\*\*\*\*使nginx配置文件高亮显示：cp -r contrib/vim/\* /usr/share/vim/vim74

安装：[进入到解压后的目录下面 eg: cd usr/local/nginx-1.15.4 后执行下面命令]

1：指定nginx的安装目录 ./configure --prefix=/usr/local/nginx

2：编译命令 make

3：安装 make install

\*\*\*\*\*执行完以上步骤后相关的nginx中的文件会在【1中prefix目录下面建立相关名称的文件】

\*\*\*\*\* nginx中常用的指令如下： 可参考网站：https://blog.csdn.net/qq\_36090419/article/details/80595153 或者使用下面的方式

检查配置文件是否出错： ./nginx -t -c /etc/nginx/nginx.conf 【通过-t进行检查 -c指定的配置文件是否出错】

在nginx启动 cd usr/local/nginx/sbin后執行 ./nginx

启动后有以下几个命令：./nginx -s signal 发送信号也可以用linux中常用的 kill 命令

signal[信号]有下面几种：

1：stop 【停止-快速关机】

2：quit 【退出-优雅退出】

3：reload 【重新加载配置文件当配置文件修改后】

4：reopen 【重新打开日志文件】

nginx通过定时[crontab linux中定时机制]进行nginx logs中的日志管理:大致操作如下

在书写脚本的时候有关于注意点:

1：操作命令用 `` 符号引用起来如下面date和cat操作命令

2：字符串用''引用起来如LOG\_PATH和LOG\_NAME等

//-----------------------脚本---------------------------------

#desc:把当前日志按日期备份，重新生成日志文件[每分钟]

#date:2016-09-18

DATE=`date +%Y%m%d%H%M`

NGINX\_PID=`cat /usr/local/nginx/logs/nginx.pid`

#如果当前Nginx没有运行就退出

if [ "$?" != 0 ]

then

exit 1;

fi

#nginx 日志所在的路径

LOG\_PATH='/usr/local/nginx/logs'

LOG\_NAME='access.log'

RECORD\_LOG\_PATH='/usr/local/nginx/logs/history'

mv ${LOG\_PATH}/${LOG\_NAME} ${RECORD\_LOG\_PATH}/access\_$DATE

#删除7天前旧的备份文件

function deloldbak()

{

olddate=`date +"%Y%m%d" -d "-$1 day"`

if [ -e "${RECORD\_LOG\_PATH}/"access\_"$olddate" ]

then

rm -f ${RECORD\_LOG\_PATH}/access\_$olddate

echo "${RECORD\_LOG\_PATH}/access\_$olddate del OK"

fi

}

#重载nginx配置，重新生成nginx日志文件

kill -USR1 $NGINX\_PID

if [ "$?" == 0 ]

then

deloldbak 7

exit 0;

fi

//-------------------------脚本-------------------------------

\*\*\*\*\*在脚本编写好之后需要对相应得的脚本文件进行权限的赋值操作：如 chmod -x [脚本文件名]，只有赋予权限后该脚本方可起效

在脚本写好后通过crontab命令进行启用定时机制

crontab -e命令后出现的页面中可以进行定时配置

\*/1 \* \* \* \* [加上脚本的路径] 该方式表示每分钟执行脚本操作

crontab -l 命令是查看有哪些定时任务

备注：定时任务的操作见网页：https://blog.csdn.net/a6864657/article/details/82086049

# SpringBoot结合分布式文件系统FastDFS实现文件上传功能

主要详情见如下网址：

<https://www.imooc.com/article/270430>

<https://go.ctolib.com/tobato-FastDFS_Client.html>

<http://bbs.chinaunix.net/forum-240-1.html>

### **简介**

