# FastDFS介绍

FastDFS开源地址：<https://github.com/happyfish100>

客户端代码案例：<https://gitee.com/zx19890628/spring-boot-example/tree/master/lab_059_util_fastDFS>

客户端源码&教程案例: <https://github.com/tobato/FastDFS_Client>

## 简介

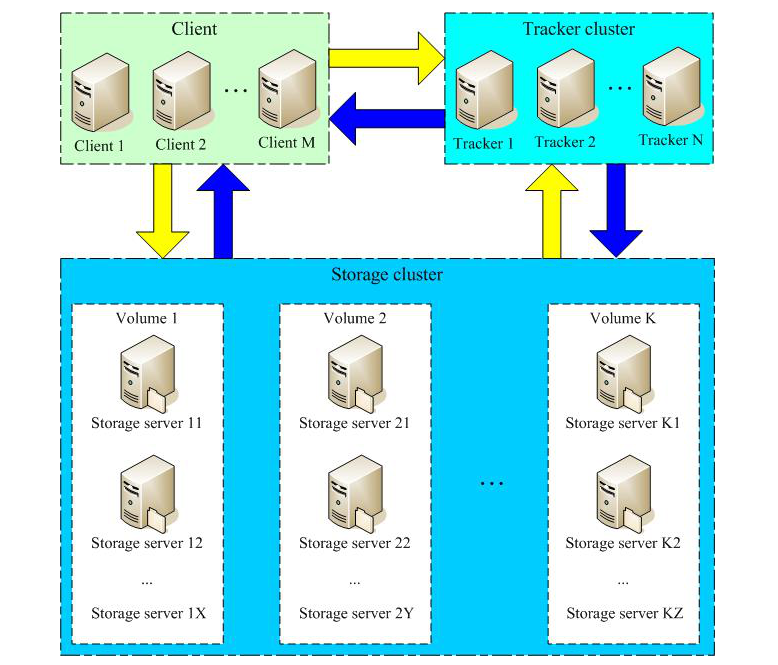
FastDFS 是一个开源的高性能分布式文件系统（DFS）。 它的主要功能包括：文件存储，文件同步和文件访问，以及高容量和负载平衡。主要解决了海量数据存储问题，特别适合以中小文件（建议范围：4KB < file\_size <500MB）为载体的在线服务。

FastDFS 系统有三个角色：跟踪服务器(Tracker Server)、存储服务器(Storage Server)和客户端(Client)。

Tracker Server：跟踪服务器，主要做调度工作，起到均衡的作用；负责管理所有的 storage server和 group，每个 storage 在启动后会连接 Tracker，告知自己所属 group 等信息，并保持周期性心跳。

Storage Server：存储服务器，主要提供容量和备份服务；以 group 为单位，每个 group 内可以有多台 storage server，数据互为备份。

Client：客户端，上传下载数据的服务器，也就是我们自己的项目所部署在的服务器。



## FastDFS的存储策略

为了支持大容量，存储节点（服务器）采用了分卷（或分组）的组织方式。存储系统由一个或多个卷组成，卷与卷之间的文件是相互独立的，所有卷的文件容量累加就是整个存储系统中的文件容量。一个卷可以由一台或多台存储服务器组成，一个卷下的存储服务器中的文件都是相同的，卷中的多台存储服务器起到了冗余备份和负载均衡的作用。

在卷中增加服务器时，同步已有的文件由系统自动完成，同步完成后，系统自动将新增服务器切换到线上提供服务。当存储空间不足或即将耗尽时，可以动态添加卷。只需要增加一台或多台服务器，并将它们配置为一个新的卷，这样就扩大了存储系统的容量。

## FastDFS的上传过程

FastDFS向使用者提供基本文件访问接口，比如upload、download、append、delete等，以客户端库的方式提供给用户使用。

Storage Server会定期的向Tracker Server发送自己的存储信息。当Tracker Server Cluster中的Tracker Server不止一个时，各个Tracker之间的关系是对等的，所以客户端上传时可以选择任意一个Tracker。

当Tracker收到客户端上传文件的请求时，会为该文件分配一个可以存储文件的group，当选定了group后就要决定给客户端分配group中的哪一个storage server。当分配好storage server后，客户端向storage发送写文件请求，storage将会为文件分配一个数据存储目录。然后为文件分配一个fileid，最后根据以上的信息生成文件名存储文件

## FastDFS的文件同步

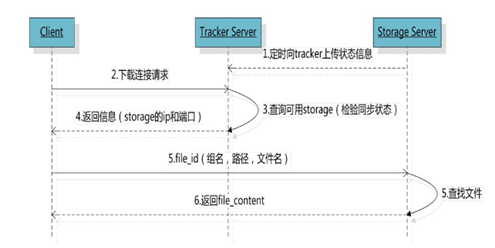
写文件时，客户端将文件写至group内一个storage server即认为写文件成功，storage server写完文件后，会由后台线程将文件同步至同group内其他的storage server。

每个storage写文件后，同时会写一份binlog，binlog里不包含文件数据，只包含文件名等元信息，这份binlog用于后台同步，storage会记录向group内其他storage同步的进度，以便重启后能接上次的进度继续同步；进度以时间戳的方式进行记录，所以最好能保证集群内所有server的时钟保持同步。

storage的同步进度会作为元数据的一部分汇报到tracker上，tracke在选择读storage的时候会以同步进度作为参考。

## FastDFS的文件下载

客户端uploadfile成功后，会拿到一个storage生成的文件名，接下来客户端根据这个文件名即可访问到该文件。



跟upload file一样，在downloadfile时客户端可以选择任意tracker server。tracker发送download请求给某个tracker，必须带上文件名信息，tracke从文件名中解析出文件的group、大小、创建时间等信息，然后为该请求选择一个storage用来服务读请求

# 安装FastDFS环境

操作环境：CentOS7 X64，以下操作都是单机环境。

我把所有的安装包下载到/softpackages/下，解压到当前目录。

先做一件事，修改hosts，将文件服务器的ip与域名映射(单机TrackerServer环境)，因为后面很多配置里面都需要去配置服务器地址，ip变了，就只需要修改hosts即可。

# vim /etc/hosts

增加如下一行，这是我的IP

192.168.51.128 file.ljzsg.com

如果要本机访问虚拟机，在C:\Windows\System32\drivers\etc\hosts中同样增加一行

## 下载安装 libfastcommon

libfastcommon是从 FastDFS 和 FastDHT 中提取出来的公共 C 函数库，基础环境，安装即可 。

① 下载libfastcommon

# wget https://github.com/happyfish100/libfastcommon/archive/V1.0.7.tar.gz

② 解压

# tar -zxvf V1.0.7.tar.gz

# cd libfastcommon-1.0.7

③ 编译、安装

# ./make.sh

# ./make.sh install

④ libfastcommon.so 安装到了/usr/lib64/libfastcommon.so，但是FastDFS主程序设置的lib目录是/usr/local/lib，所以需要创建软链接。

# ln -s /usr/lib64/libfastcommon.so /usr/local/lib/libfastcommon.so

# ln -s /usr/lib64/libfastcommon.so /usr/lib/libfastcommon.so

# ln -s /usr/lib64/libfdfsclient.so /usr/local/lib/libfdfsclient.so

# ln -s /usr/lib64/libfdfsclient.so /usr/lib/libfdfsclient.so

## 下载安装FastDFS

① 下载FastDFS

# wget https://github.com/happyfish100/fastdfs/archive/V5.05.tar.gz

② 解压

# tar -zxvf V5.05.tar.gz

# cd fastdfs-5.05

③ 编译、安装

# ./make.sh

# ./make.sh install

④ 默认安装方式安装后的相应文件与目录

　　A、服务脚本：

/etc/init.d/fdfs\_storaged

/etc/init.d/fdfs\_tracker

B、配置文件（这三个是作者给的样例配置文件） :

/etc/fdfs/client.conf.sample

/etc/fdfs/storage.conf.sample

/etc/fdfs/tracker.conf.sample

C、命令工具在 /usr/bin/ 目录下：

fdfs\_appender\_test

fdfs\_appender\_test1

fdfs\_append\_file

fdfs\_crc32

fdfs\_delete\_file

fdfs\_download\_file

fdfs\_file\_info

fdfs\_monitor

fdfs\_storaged

fdfs\_test

fdfs\_test1

fdfs\_trackerd

fdfs\_upload\_appender

fdfs\_upload\_file

stop.sh

restart.sh

⑤ FastDFS 服务脚本设置的 bin 目录是 /usr/local/bin， 但实际命令安装在 /usr/bin/ 下。

两种方式：

一是修改FastDFS 服务脚本中相应的命令路径，也就是把 /etc/init.d/fdfs\_storaged 和 /etc/init.d/fdfs\_tracker 两个脚本中的 /usr/local/bin 修改成 /usr/bin。

　　　　# vim fdfs\_trackerd

　　　　使用查找替换命令进统一修改:%s+/usr/local/bin+/usr/bin

　　　　# vim fdfs\_storaged

　　　　使用查找替换命令进统一修改:%s+/usr/local/bin+/usr/bin

二是建立 /usr/bin 到 /usr/local/bin 的软链接，我是用这种方式。

# ln -s /usr/bin/fdfs\_trackerd /usr/local/bin

# ln -s /usr/bin/fdfs\_storaged /usr/local/bin

# ln -s /usr/bin/stop.sh /usr/local/bin

# ln -s /usr/bin/restart.sh /usr/local/bin

## 配置FastDFS跟踪器(Tracker)

配置文件详细说明参考：FastDFS 配置文件详解

① 进入 /etc/fdfs，复制 FastDFS 跟踪器样例配置文件 tracker.conf.sample，并重命名为 tracker.conf。

# cd /etc/fdfs

# cp tracker.conf.sample tracker.conf

# vim tracker.conf

② 编辑tracker.conf ，标红的需要修改下，其它的默认即可。

# 配置文件是否不生效，false 为生效

disabled=false

# 提供服务的端口

port=22122

# Tracker 数据和日志目录地址(根目录必须存在,子目录会自动创建)

base\_path=/ljzsg/fastdfs/tracker

# HTTP 服务端口

http.server\_port=80

③ 创建tracker基础数据目录，即base\_path对应的目录

# mkdir -p /ljzsg/fastdfs/tracker

④ 防火墙中打开跟踪端口（默认的22122）

# vim /etc/sysconfig/iptables

添加如下端口行：

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

重启防火墙：

# service iptables restart

⑤ 启动Tracker

初次成功启动，会在 /ljzsg/fdfsdfs/tracker/ (配置的base\_path)下创建 data、logs 两个目录。

可以用这种方式启动

# /etc/init.d/fdfs\_trackerd start

也可以用这种方式启动，前提是上面创建了软链接，后面都用这种方式

# service fdfs\_trackerd start

查看 FastDFS Tracker 是否已成功启动 ，22122端口正在被监听，则算是Tracker服务安装成功。

# netstat -unltp|grep fdfs

关闭Tracker命令：

# service fdfs\_trackerd stop

⑥ 设置Tracker开机启动

# chkconfig fdfs\_trackerd on

或者：

# vim /etc/rc.d/rc.local

加入配置：

/etc/init.d/fdfs\_trackerd start

⑦ tracker server 目录及文件结构

Tracker服务启动成功后，会在base\_path下创建data、logs两个目录。目录结构如下：

${base\_path}

|\_\_data

| |\_\_storage\_groups.dat：存储分组信息

| |\_\_storage\_servers.dat：存储服务器列表

|\_\_logs

| |\_\_trackerd.log： tracker server 日志文件

## 配置 FastDFS 存储 (Storage)

① 进入 /etc/fdfs 目录，复制 FastDFS 存储器样例配置文件 storage.conf.sample，并重命名为 storage.conf

# cd /etc/fdfs

# cp storage.conf.sample storage.conf

# vim storage.conf

② 编辑storage.conf

# 配置文件是否不生效，false 为生效

disabled=false

# 指定此 storage server 所在 组(卷)

group\_name=group1

# storage server 服务端口

port=23000

# 心跳间隔时间，单位为秒 (这里是指主动向 tracker server 发送心跳)

heart\_beat\_interval=30

# Storage 数据和日志目录地址(根目录必须存在，子目录会自动生成)

base\_path=/ljzsg/fastdfs/storage

# 存放文件时 storage server 支持多个路径。这里配置存放文件的基路径数目，通常只配一个目录。

store\_path\_count=1

# 逐一配置 store\_path\_count 个路径，索引号基于 0。

# 如果不配置 store\_path0，那它就和 base\_path 对应的路径一样。

store\_path0=/ljzsg/fastdfs/file

# FastDFS 存储文件时，采用了两级目录。这里配置存放文件的目录个数。

# 如果本参数只为 N（如： 256），那么 storage server 在初次运行时，会在 store\_path 下自动创建 N \* N 个存放文件的子目录。

subdir\_count\_per\_path=256

# tracker\_server 的列表 ，会主动连接 tracker\_server

# 有多个 tracker server 时，每个 tracker server 写一行

tracker\_server=file.ljzsg.com:22122

# 允许系统同步的时间段 (默认是全天) 。一般用于避免高峰同步产生一些问题而设定。

sync\_start\_time=00:00

sync\_end\_time=23:59

# 访问端口

http.server\_port=80

③ 创建Storage基础数据目录，对应base\_path目录

# mkdir -p /ljzsg/fastdfs/storage

# 这是配置的store\_path0路径

# mkdir -p /ljzsg/fastdfs/file

④ 防火墙中打开存储器端口（默认的 23000）

# vim /etc/sysconfig/iptables

添加如下端口行：

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

重启防火墙：

# service iptables restart

⑤ 启动 Storage

启动Storage前确保Tracker是启动的。初次启动成功，会在 /ljzsg/fastdfs/storage 目录下创建 data、 logs 两个目录。

可以用这种方式启动

# /etc/init.d/fdfs\_storaged start

也可以用这种方式，后面都用这种

# service fdfs\_storaged start

查看 Storage 是否成功启动，23000 端口正在被监听，就算 Storage 启动成功。

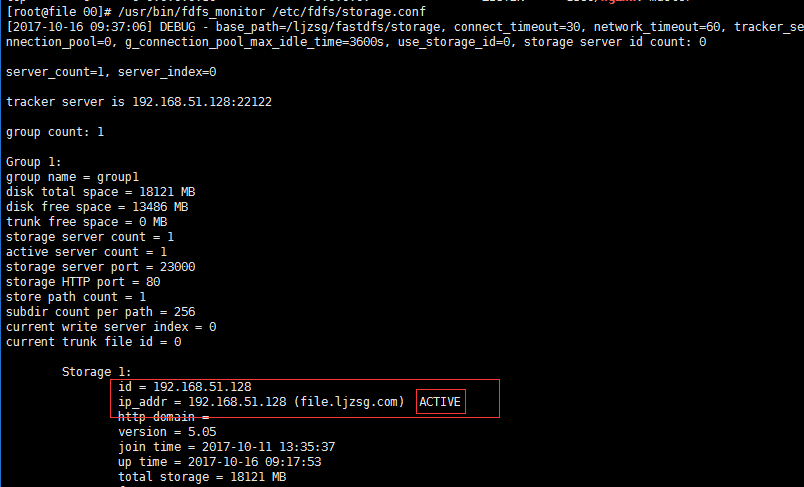
# netstat -unltp|grep fdfs

关闭Storage命令：

# service fdfs\_storaged stop

查看Storage和Tracker是否在通信：

/usr/bin/fdfs\_monitor /etc/fdfs/storage.conf



⑥ 设置 Storage 开机启动

# chkconfig fdfs\_storaged on

或者：

# vim /etc/rc.d/rc.local

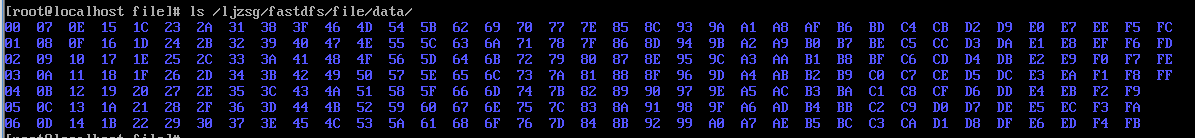
加入配置：

/etc/init.d/fdfs\_storaged start

⑦ Storage 目录

同 Tracker，Storage 启动成功后，在base\_path 下创建了data、logs目录，记录着 Storage Server 的信息。

在 store\_path0 目录下，创建了N\*N个子目录：



## 文件上传测试

① 修改 Tracker 服务器中的客户端配置文件

# cd /etc/fdfs

# cp client.conf.sample client.conf

# vim client.conf

修改如下配置即可，其它默认。

# Client 的数据和日志目录

base\_path=/ljzsg/fastdfs/client

# Tracker端口

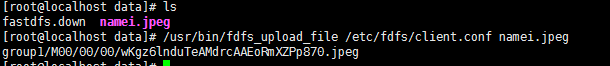
tracker\_server=file.ljzsg.com:22122

② 上传测试

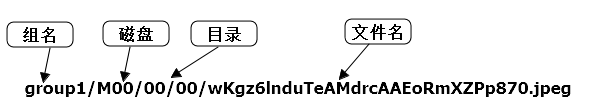
在linux内部执行如下命令上传 namei.jpeg 图片

# /usr/bin/fdfs\_upload\_file /etc/fdfs/client.conf namei.jpeg

上传成功后返回文件ID号：group1/M00/00/00/wKgz6lnduTeAMdrcAAEoRmXZPp870.jpeg



返回的文件ID由group、存储目录、两级子目录、fileid、文件后缀名（由客户端指定，主要用于区分文件类型）拼接而成。



# FastDFS 配置 Nginx 模块

## 安装配置Nginx模块

① fastdfs-nginx-module 模块说明

FastDFS 通过 Tracker 服务器，将文件放在 Storage 服务器存储， 但是同组存储服务器之间需要进行文件复制， 有同步延迟的问题。

假设 Tracker 服务器将文件上传到了 192.168.51.128，上传成功后文件 ID已经返回给客户端。

此时 FastDFS 存储集群机制会将这个文件同步到同组存储 192.168.51.129，在文件还没有复制完成的情况下，客户端如果用这个文件 ID 在 192.168.51.129 上取文件,就会出现文件无法访问的错误。

而 fastdfs-nginx-module 可以重定向文件链接到源服务器取文件，避免客户端由于复制延迟导致的文件无法访问错误。

② 下载 fastdfs-nginx-module、解压

# 这里为啥这么长一串呢，因为最新版的master与当前nginx有些版本问题。

# wget https://github.com/happyfish100/fastdfs-nginx-module/archive/5e5f3566bbfa57418b5506aaefbe107a42c9fcb1.zip

# 解压

# unzip 5e5f3566bbfa57418b5506aaefbe107a42c9fcb1.zip

# 重命名

# mv fastdfs-nginx-module-5e5f3566bbfa57418b5506aaefbe107a42c9fcb1 fastdfs-nginx-module-master

③ 配置Nginx

在nginx中添加模块

# 先停掉nginx服务

# /usr/local/nginx/sbin/ngix -s stop

进入解压包目录

# cd /softpackages/nginx-1.12.1/

# 添加模块

# ./configure --add-module=../fastdfs-nginx-module-master/src

重新编译、安装

# make && make install

④ 查看Nginx的模块

# /usr/local/nginx/sbin/nginx -V

有下面这个就说明添加模块成功

⑤ 复制 fastdfs-nginx-module 源码中的配置文件到/etc/fdfs 目录， 并修改

# cd /softpackages/fastdfs-nginx-module-master/src

# cp mod\_fastdfs.conf /etc/fdfs/

修改如下配置，其它默认

# 连接超时时间

connect\_timeout=10

# Tracker Server

tracker\_server=file.ljzsg.com:22122

# StorageServer 默认端口

storage\_server\_port=23000

# 如果文件ID的uri中包含/group\*\*，则要设置为true

url\_have\_group\_name = true

# Storage 配置的store\_path0路径，必须和storage.conf中的一致

store\_path0=/ljzsg/fastdfs/file

⑥ 复制 FastDFS 的部分配置文件到/etc/fdfs 目录

# cd /softpackages/fastdfs-5.05/conf/

# cp anti-steal.jpg http.conf mime.types /etc/fdfs/

⑦ 配置nginx，修改nginx.conf

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

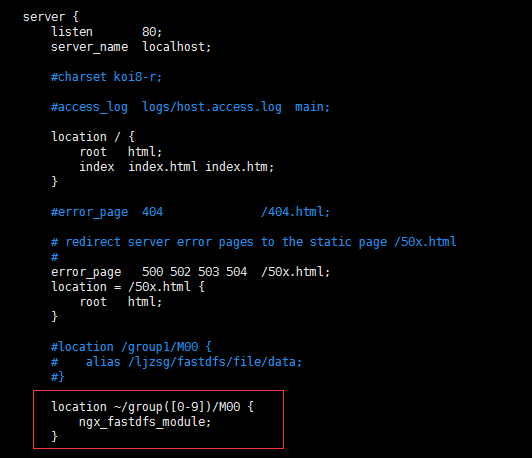
修改配置，其它的默认

在80端口下添加fastdfs-nginx模块

location ~/group([0-9])/M00 {

ngx\_fastdfs\_module;

}



注意：

　　listen 80 端口值是要与 /etc/fdfs/storage.conf 中的 http.server\_port=80 (前面改成80了)相对应。如果改成其它端口，则需要统一，同时在防火墙中打开该端口。

　　location 的配置，如果有多个group则配置location ~/group([0-9])/M00 ，没有则不用配group。

⑧ 在/ljzsg/fastdfs/file 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，这一步可以省略。

# ln -s /ljzsg/fastdfs/file/data/ /ljzsg/fastdfs/file/data/M00

⑨ 启动nginx

# /usr/local/nginx/sbin/nginx

打印处如下就算配置成功

IMG_256

⑩ 在地址栏访问。

能下载文件就算安装成功。注意和第三点中直接使用nginx路由访问不同的是，这里配置 fastdfs-nginx-module 模块，可以重定向文件链接到源服务器取文件。

http://file.ljzsg.com/group1/M00/00/00/wKgz6lnduTeAMdrcAAEoRmXZPp870.jpeg

# FastDFS java client SDK

## 使用ant从源码构建

ant clean package

## 使用maven从源码安装

mvn clean install

## 使用maven从jar文件安装

mvn install:install-file -DgroupId=org.csource -DartifactId=fastdfs-client-java -Dversion=${version} -Dpackaging=jar -Dfile=fastdfs-client-java-${version}.jar

## 在您的maven项目pom.xml中添加依赖

<dependency>

<groupId>org.csource</groupId>

<artifactId>fastdfs-client-java</artifactId>

<version>1.27-SNAPSHOT</version>

</dependency>

## .conf 配置文件、所在目录、加载优先顺序

配置文件名fdfs\_client.conf(或使用其它文件名xxx\_yyy.conf)

文件所在位置可以是项目classpath(或OS文件系统目录比如/opt/):

/opt/fdfs\_client.conf

C:\Users\James\config\fdfs\_client.conf

优先按OS文件系统路径读取，没有找到才查找项目classpath，尤其针对linux环境下的相对路径比如：

fdfs\_client.conf

config/fdfs\_client.conf

connect\_timeout = 2

network\_timeout = 30

charset = UTF-8

http.tracker\_http\_port = 80

http.anti\_steal\_token = no

http.secret\_key = FastDFS1234567890

tracker\_server = 10.0.11.247:22122

tracker\_server = 10.0.11.248:22122

tracker\_server = 10.0.11.249:22122

注1：tracker\_server指向您自己IP地址和端口，1-n个

注2：除了tracker\_server，其它配置项都是可选的

## .properties 配置文件、所在目录、加载优先顺序

配置文件名 fastdfs-client.properties(或使用其它文件名 xxx-yyy.properties)

文件所在位置可以是项目classpath(或OS文件系统目录比如/opt/):

/opt/fastdfs-client.properties

C:\Users\James\config\fastdfs-client.properties

优先按OS文件系统路径读取，没有找到才查找项目classpath，尤其针对linux环境下的相对路径比如：

fastdfs-client.properties

config/fastdfs-client.properties

fastdfs.connect\_timeout\_in\_seconds = 5

fastdfs.network\_timeout\_in\_seconds = 30

fastdfs.charset = UTF-8

fastdfs.http\_anti\_steal\_token = false

fastdfs.http\_secret\_key = FastDFS1234567890

fastdfs.http\_tracker\_http\_port = 80

fastdfs.tracker\_servers = 10.0.11.201:22122,10.0.11.202:22122,10.0.11.203:22122

注1：properties 配置文件中属性名跟 conf 配置文件不尽相同，并且统一加前缀"fastdfs."，便于整合到用户项目配置文件

注2：fastdfs.tracker\_servers 配置项不能重复属性名，多个 tracker\_server 用逗号","隔开

注3：除了fastdfs.tracker\_servers，其它配置项都是可选的

## 加载配置示例

加载原 conf 格式文件配置：

ClientGlobal.init("fdfs\_client.conf");

ClientGlobal.init("config/fdfs\_client.conf");

ClientGlobal.init("/opt/fdfs\_client.conf");

ClientGlobal.init("C:\\Users\\James\\config\\fdfs\_client.conf");

加载 properties 格式文件配置：

ClientGlobal.initByProperties("fastdfs-client.properties");

ClientGlobal.initByProperties("config/fastdfs-client.properties");

ClientGlobal.initByProperties("/opt/fastdfs-client.properties");

ClientGlobal.initByProperties("C:\\Users\\James\\config\\fastdfs-client.properties");

加载 Properties 对象配置：

Properties props = new Properties();

props.put(ClientGlobal.PROP\_KEY\_TRACKER\_SERVERS, "10.0.11.101:22122,10.0.11.102:22122");

ClientGlobal.initByProperties(props);

加载 trackerServers 字符串配置：

String trackerServers = "10.0.11.101:22122,10.0.11.102:22122";

ClientGlobal.initByTrackers(trackerServers);

## 检查加载配置结果：

System.out.println("ClientGlobal.configInfo(): " + ClientGlobal.configInfo());

ClientGlobal.configInfo(): {

g\_connect\_timeout(ms) = 5000

g\_network\_timeout(ms) = 30000

g\_charset = UTF-8

g\_anti\_steal\_token = false

g\_secret\_key = FastDFS1234567890

g\_tracker\_http\_port = 80

trackerServers = 10.0.11.101:22122,10.0.11.102:22122

}