随手记点笔记，不全，自用

本文思维导图(PDF)

本文思维导图(xmind)

上传过程

1.Storage Server定时上传状态信息给tracker

2.Client上传连接请求

3.TrackerServer查询可用的Storage Server

4.返回查询结果(Storage Server的IP和端口号)

5.Client开始上传文件(file content & metadata)

6.Storage Server生成file_id

7.Storage Server将上传内容写入磁盘

8.返回file_id(路径信息和文件名)

- 组名：文件上传后所在的 storage 组名称，在文件上传成功后有storage 服务器返回，需要客户端自行保存。
- 虚拟磁盘路径：storage 配置的虚拟路径，与磁盘选项store_path*对应。如果配置了store_path0 则是 M00，如果配置了 store_path1 则是 M01，以此类推。
- 数据两级目录：storage 服务器在每个虚拟磁盘路径下创建的两级目录，用于存储数据文件。
- 文件名：与文件上传时不同。是由存储服务器根据特定信息生成，文件名包含：源存储服务器 IP 地址、文件创建时间戳、文件大小、随机数和文件拓展名等信息。

9.Client存储文件信息

搭建流程

拉取镜像：

```
docker pull morunchang/fastdfs
```

运行tracker：

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

运行storage：

```
docker run -d --name storage --net=host -e TRACKER_IP=<your tracker
server address>:22122 -e GROUP_NAME=<group name> morunchang/fastdfs sh
storage.sh
```

- 使用的网络模式是–net=host, 替换为你机器的IP即可
- 如果想要增加新的storage服务器，再次运行该命令，注意更换 新组名

进入storage的容器内部，修改nginx.conf：

```
docker exec -it storage /bin/bash
```

vi /data/nginx/conf/nginx.conf 添加以下设置

```
location /group1/M00 {
    proxy_next_upstream http_502 http_504 error timeout invalid_header;
        proxy_cache http-cache;
        proxy_cache_valid 200 304 12h;
        proxy_cache_key $uri$is_args$args;
        proxy_pass http://fdfs_group1;
        expires 30d;
}
```

退出容器：

```
exit
```

重启storage容器：

```
docker restart storage
docker run -d --name storage --net=host -e
TRACKER_IP=<your tracker
server address>:22122 -e GROUP_NAME=<
group name> morunchang/fastdfs sh
storage.sh
```

代码部分

导包

resources下建立配置文件 fdfs_client.conf

```
connect_timeout = 60
network_timeout = 60
charset = UTF-8
http.tracker_http_port = 8080
tracker_server = 192.168.200.128:22122
```

resources下建立配置文件 application.yml

```
spring:
servlet:
    multipart:
        max-file-size: 10MB
        max-request-size: 10MB
server:
port: 9008
eureka:
```

```
client:
    service-url:
        defaultZone: http://127.0.0.1:6868/eureka
    instance:
        prefer-ip-address: true
feign:
hystrix:
    enabled: true
```

创建com.changgou.file包，创建启动类FileApplication

```
@SpringBootApplication
@EnableEurekaClient
public class FileApplication {
    public static void main(String[] args) {
        SpringApplication.run(FileApplication.class);
    }
}
```

创建一个实体类，内有文件名、内容、扩展名、MD5值、作者等信息

创建工具类FastDFSClient.java

```
package com.changgou.file.util;

import org.csource.common.NameValuePair;
import org.csource.fastdfs.*;
import org.slf4j.LoggerFactory;
import org.springframework.core.io.ClassPathResource;

import java.io.ByteArrayInputStream;
import java.io.IOException;
import java.io.InputStream;

public class FastDFSClient {

    private static org.slf4j.Logger logger =
LoggerFactory.getLogger(FastDFSClient.class);

    /***
     * 初始化加载FastDFS的TrackerServer配置
     */
    static {
        try {
            String filePath = new
ClassPathResource("fdfs_client.conf").getFile().getAbsolutePath();
            ClientGlobal.init(filePath);
        } catch (Exception e) {
            logger.error("FastDFS Client Init Fail!",e);
        }
    }

    /***
     * 文件上传
     * @param file
```

```
     * @return 1.文件的组名   2.文件的路径信息
     */
    public static String[] upload(File file) {
        //获取文件的作者
        NameValuePair[] meta_list = new NameValuePair[1];
        meta_list[0] = new NameValuePair("author", file.getAuthor());

        //接收返回数据
        String[] uploadResults = null;
        StorageClient storageClient=null;
        try {
            //创建StorageClient客户端对象
            storageClient = getTrackerClient();

            /***
             * 文件上传
             * 1)文件字节数组
             * 2)文件扩展名
             * 3)文件作者
             */
            uploadResults = storageClient.upload_file(file.getContent(),
file.getExt(), meta_list);
        } catch (Exception e) {
            logger.error("Exception when uploadind the file:" +
file.getName(), e);
        }

        if (uploadResults == null && storageClient!=null) {
            logger.error("upload file fail, error code:" +
storageClient.getErrorCode());
        }
        //获取组名
        String groupName = uploadResults[0];
        //获取文件存储路径
        String remoteFileName = uploadResults[1];
        return uploadResults;
    }

    /***
     * 获取文件信息
     * @param groupName:组名
     * @param remoteFileName: 文件存储完整名
     * @return
     */
    public static FileInfo getFile(String groupName, String remoteFileName)
{
        try {
            StorageClient storageClient = getTrackerClient();
            return storageClient.get_file_info(groupName, remoteFileName);
        } catch (Exception e) {
            logger.error("Exception: Get File from Fast DFS failed", e);
        }
        return null;
    }

    /***
     * 文件下载
     * @param groupName
```

```
     * @param remoteFileName
     * @return
     */
    public static InputStream downFile(String groupName, String
remoteFileName) {
        try {
            //创建StorageClient
            StorageClient storageClient = getTrackerClient();

            //下载文件
            byte[] fileByte = storageClient.download_file(groupName,
remoteFileName);
            InputStream ins = new ByteArrayInputStream(fileByte);
            return ins;
        } catch (Exception e) {
            logger.error("Exception: Get File from Fast DFS failed", e);
        }
        return null;
    }

    /***
     * 文件删除
     * @param groupName
     * @param remoteFileName
     * @throws Exception
     */
    public static void deleteFile(String groupName, String remoteFileName)
            throws Exception {
        //创建StorageClient
        StorageClient storageClient = getTrackerClient();

        //删除文件
        int i = storageClient.delete_file(groupName, remoteFileName);
    }

    /***
     * 获取Storage组
     * @param groupName
     * @return
     * @throws IOException
     */
    public static StorageServer[] getStoreStorages(String groupName)
            throws IOException {
        //创建TrackerClient
        TrackerClient trackerClient = new TrackerClient();
        //获取TrackerServer
        TrackerServer trackerServer = trackerClient.getConnection();
        //获取Storage组
        return trackerClient.getStoreStorages(trackerServer, groupName);
    }

    /***
     * 获取Storage信息,IP和端口
     * @param groupName
     * @param remoteFileName
     * @return
     * @throws IOException
     */
```

```
    public static ServerInfo[] getFetchStorages(String groupName,
                                                String remoteFileName)
throws IOException {
        TrackerClient trackerClient = new TrackerClient();
        TrackerServer trackerServer = trackerClient.getConnection();
        return trackerClient.getFetchStorages(trackerServer, groupName,
remoteFileName);
    }

    /***
     * 获取Tracker服务地址
     * @return
     * @throws IOException
     */
    public static String getTrackerUrl() throws IOException {
        return
"http://"+getTrackerServer().getInetSocketAddress().getHostString()+":"+Cli
entGlobal.getG_tracker_http_port()+"/";
    }

    /***
     * 获取Storage客户端
     * @return
     * @throws IOException
     */
    private static StorageClient getTrackerClient() throws IOException {
        TrackerServer trackerServer = getTrackerServer();
        StorageClient storageClient = new StorageClient(trackerServer,
null);
        return  storageClient;
    }

    /***
     * 获取Tracker
     * @return
     * @throws IOException
     */
    private static TrackerServer getTrackerServer() throws IOException {
        TrackerClient trackerClient = new TrackerClient();
        TrackerServer trackerServer = trackerClient.getConnection();
        return  trackerServer;
    }
}

```

创建FileController.java来操作文件上传

```
package com.changgou.file.controller;

import com.changgou.entity.Result;
import com.changgou.entity.StatusCode;
import com.changgou.file.util.FastDFSClient;
import com.changgou.file.util.FastDFSFile;
import org.apache.commons.lang.StringUtils;
import org.springframework.web.bind.annotation.PostMapping;
```

```java
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;
import org.springframework.web.multipart.MultipartFile;

@RestController
@RequestMapping("/file")
public class FileController {

    @PostMapping("/upload")
    public Result uploadFile(MultipartFile file){
        try{
            //判断文件是否存在
            if (file == null){
                throw new RuntimeException("文件不存在");
            }
            //获取文件的完整名称
            String originalFilename = file.getOriginalFilename();
            if (StringUtils.isEmpty(originalFilename)){
                throw new RuntimeException("文件不存在");
            }

            //获取文件的扩展名称  abc.jpg   jpg
            String extName =
originalFilename.substring(originalFilename.lastIndexOf(".") + 1);

            //获取文件内容
            byte[] content = file.getBytes();

            //创建文件上传的封装实体类
            FastDFSFile fastdfsFile = new
FastDFSFile(originalFilename,content,extName);

            //基于工具类进行文件上传,并接受返回参数  String[]
            String[] uploadResult = FastDFSClient.upload(fastdfsFile);

            //封装返回结果
            String url =
FastDFSClient.getTrackerUrl()+uploadResult[0]+"/"+uploadResult[1];
            return new Result(true,StatusCode.OK,"文件上传成功",url);
        }catch (Exception e){
            e.printStackTrace();
        }
        return new Result(false, StatusCode.ERROR,"文件上传失败");
    }
}

```

Postman测试

```
http://localhost:9008/upload
```

```
Key: Content-Type

Value: multipart/form-data
```

选择form-data 然后选择文件file 点击添加文件，最后发送即可