fastdfs 文件上传 java 代码

一.添加依赖

1.下载依赖

下载地址:https://github.com/happyfish100/fastdfs-client-java

2.安装依赖

将项目导入到 IDE 中,或者直接使用 maven 指令将依赖安装到仓库中

3.项目中添加依赖

```
<dependency>
    <groupId>org.csource</groupId>
    <artifactId>fastdfs-client-java</artifactId>
    <version>1.27-SNAPSHOT</version>
</dependency>
```

4.配置文件

4.1 conf 格式

```
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
```

4.2 properties 格式

```
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
```

5.加载配置示例(后面用)

```
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);
```

6.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
}
```

二. 测试方式演示

1.直接测试

```
@Test
    public void test1() throws Exception {
        //加载配置文件,需要告诉它向导服务器在哪
        ClientGlobal.init("/home/jackiechan/文
档/workspace/gp1701/testfastdfs/src/main/resources/server.conf");
        //创建客户端
        TrackerClient trackerClient=new TrackerClient();
        TrackerServer trackerServer = trackerClient.getConnection();
        StorageServer server=null;
        StorageClient storageClient=new StorageClient(trackerServer, server);
        String[] strings = storageClient.upload_file("/home/jackiechan/12333.jpg",
"jpg", null);//参数1 文件路径,参数2 文件的扩展名,参数3文件的元数据
        for (String string : strings) {
            System.err.println(string);//返回的结果就是整个地址,不包括域名 ip
        }
        // group1/M00/00/00/rBFdZ1oU8z2AAKfLABx_3eMqYVk200.jpg
    }
```

2.封装简单工具类

```
/**
    工具类中只封装了部分方法,更多方法请自己封装
*/
public class FastDFSClient {
    private TrackerClient trackerClient;
    private TrackerServer trackerServer;
    private StorageClient1 storageClient;
    private StorageServer storageServer;
```

```
    public FastDFSClient(String conf) throws Exception {
        if (conf.startsWith("classpath")) {//如果配置文件是 classpath 开头的,则代表是
在类路径中,去掉 classpath: 然后拼接类路径
            conf=conf.replace("classpath:",
getClass().getResource("/").getPath());
        }
        ClientGlobal.init(conf);//加载路径
        trackerClient=new TrackerClient();
        trackerServer=trackerClient.getConnection();
        storageClient=new StorageClient1(trackerServer, storageServer);
    }
    /**
        上传文件,参数是文件的路径,后缀名和元数据
    */
    public String  upload_file(String fileName,String ext_name,NameValuePair[]
pairs) throws Exception {
        return storageClient.upload_file1(fileName, ext_name, pairs);
    }

    public String upload_file(String fileName)  throws Exception{
        return upload_file(fileName, null, null);
    }


    public String upload_file(String fileName,String ext_name)  throws Exception{
        return upload_file(fileName, ext_name, null);
    }

    public String upload_file(String fileName,NameValuePair[] pairs)  throws
Exception{
        return upload_file(fileName, null, pairs);
    }
    /**
        上传二进制数据,需要将文件先转换为二进制
    */
    public String upload_file(byte[]source,String ext_name,NameValuePair[] pairs)
throws Exception{
        return storageClient.upload_file1(source, ext_name, pairs);
    }
}
```

3. 测试工具类

```
@Test
    public void test2()  throws Exception{
        FastDFSClient fastDFSClient =new FastDFSClient("/home/jackiechan/文
档/workspace/gp1701/testfastdfs/src/main/resources/server.conf");
        String upload_file =
fastDFSClient.upload_file("/home/jackiechan/12333.jpg");
        System.err.println(upload_file);
    }
```

三. SpringMvc 中使用

> 环境配置和前面四步一致

代码

```
@Controller
public class FileUploadController {
/**
 * MultipartFile的形参的名字和form表单中文件的域name一致
 * @param file
 * @return
 */
    @ResponseBody
    @RequestMapping(value="/upload",produces= {"text/plain;charset=utf-8"})
    public String upLoad(MultipartFile file) {
        String path=null;
        Map<String, Object> map=new HashMap<>();//用于返回 json 数据的map 对象
        try {
            //file是用户上传的文件
            //需要将文件转存到文件服务器
            //file.transferTo(dest); 存储到本地文件
            /*file.getOriginalFilename()
            file.getName()*/
            FastDFSClient fastDFSClient=new
FastDFSClient("classpath:server.conf");//此处应当注入对象过来
            path= fastDFSClient.upload_file(file.getBytes(), "jpg", null);
            map.put("error", 0);//设置返回内容
            map.put("url", "http://jiaoxue.chenjunbo.xin/"+path);
        } catch (Exception e) {
            map.put("error", 1);
            map.put("message", "发斯蒂芬斯蒂芬斯蒂芬");
            e.printStackTrace();
        }
```

```
        return JsonUtils.objectToJson(map);//将数据转成 json 数据返回,此处可以自己写或者直接返回map
    }
}
```