**FastDFS文件系统使用说明书**

|  |  |  |
| --- | --- | --- |
| 文件状态:  [ ] 草稿  [ ] 正式发布  [ √] 正在修改 | 文件标识 |  |
| 当前版本 | V1.0 |
| 作 者 | huangjianwei |
| 审 核 |  |
| 完成日期 | 2017/12/11 |

版本历史

|  |  |  |  |
| --- | --- | --- | --- |
| 版本 | 作者 | 起止日期 | 备注 |
| V1.0.0 | huangjianwei | 2017-12-09-2017-12-11 | 基本使用 |
| V2.0.0 | huangjianwei | 2017-12-13-2017-12-13 | FastDFSClient使用 |
| V3.0.0 | huangjianwei | 2017-12-22-2017-12-22 | 加入防盗链功能 |
|  |  |  |  |
|  |  |  |  |

**FastDFS文件系统使用说明书**

1. 项目配置

步骤：1.配置pom.xml文件，加入fastdfs依赖包

<dependency>

<groupId>com.szwcyq.ggw.share</groupId>

<artifactId>ggw-share-fastdfs-common</artifactId>

<version>0.3.0-SNAPSHOT</version>

</dependency>

1. 在application-local.yml加入fastdfs连接属性

fdfs:

#获取文件超时时间

soTimeout: 1501

#连接超时时间

connectTimeout: 601

#缩列图大小定义

thumbImage:

width: 150

height: 150

#前端获取url

webServerUrl: <http://192.168.93.132:8888/>

#防盗链key

secret\_key: FastDFS1234567890

#fastdfs服务器地址，配置的是集群版时，即添加多个地址

trackerList:

- 192.168.93.132:22122

# - 192.168.174.47:22122

# - 192.168.174.48:22122

1. 在config在引入fastdfs配置

@Import(FdfsClientConfig.**class**)

1. 基本配置完成
2. FastFileStorageClient接口使用
3. 方法名：uploadFile

说明：上传一般文件

方法体：StorePath uploadFile(InputStream inputStream, **long** fileSize, String fileExtName, Set<MateData> metaDataSet);

参数说明：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 说明 |
| inputStream | InputStream | 文件流 |
| fileSize | **long** | 文件大小 |
| fileExtName | String | 文件后缀名 |
| metaDataSet | Set<MateData> | 文件元数据 |
|  |  |  |

返回参数说明：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 说明 |
| StorePath | Object | 返回一个对象 |
| storeParh对象参数如下 | | |
| group | String | 组名 |
| path | String | 路径名 |

1. 方法名：uploadImageAndCrtThumbImage

说明：上传图片并且生成缩略图（标准150\*150），支持的图片格式包括"JPG", "JPEG", "PNG", "GIF", "BMP", "WBMP"

方法体：StorePath uploadImageAndCrtThumbImage(InputStream inputStream, **long** fileSize, String fileExtName,Set<MateData> metaDataSet);

参数说明：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 说明 |
| inputStream | InputStream | 文件流 |
| fileSize | **long** | 文件大小 |
| fileExtName | String | 文件后缀名 |
| metaDataSet | Set<MateData> | 文件元数据 |
|  |  |  |

返回参数说明：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 说明 |
| StorePath | Object | 返回一个对象 |
| storeParh对象参数如下 | | |
| group | String | 组名 |
| path | String | 路径名 |

1. 方法名：deleteFile

说明：删除文件

方法体：**void** deleteFile(String filePath);

参数说明：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 说明 |
| filePath | String | 文件路径(groupName/path) |

返回参数说明：

无

4.方法名：downloadFile

说明：下载整个文件

方法体：<T> T downloadFile(String groupName, String path, DownloadCallback<T> callback);

参数说明：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 说明 |
| groupName | String | 组名 |
| path | String | 路径url |
| callback | DownloadCallback<T> | 返回数据定义 |

返回参数说明：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 说明 |
| T | DownloadCallback<T> | 返回数据 |

5.方法名：deleteFile

说明：删除文件

方法体：**void** deleteFile(String groupName, String path);

参数说明：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 说明 |
| groupName | String | 组名 |
| path | String | 路径url |

返回参数说明：

无

1. 使用演示（上传文件）

@Autowired

**protected** FastFileStorageClient storageClient;

@Autowired

**private** ThumbImageConfig thumbImageConfig;

方法体代码如下：

InputStream in = **null**;

File file = **new** File("F:/hjw/gaofutongworkspace/share/projects/web/web-fastdfs-base/src/main/resources/images/gs.jpg");

*log*.debug("--file:{}",file.getName());

String fileExtName = FilenameUtils.*getExtension*(file.getName());

*log*.debug("--:fileExtName:{}",fileExtName);

**long** fileSize = file.length();

*log*.debug("--:fileSize:{}",fileSize);

**try** {

in=**new** FileInputStream(file);

*log*.debug("storageClient:{}",storageClient);

//StorePath uploadImageAndCrtThumbImage = storageClient.uploadImageAndCrtThumbImage(in, fileSize, fileExtName, createMateData());

//StorePath uploadImageAndCrtThumbImage = storageClient.uploadImageAndCrtThumbImage(in, fileSize, fileExtName, null,300,300);

StorePath uploadImageAndCrtThumbImage = storageClient.uploadFile(in, fileSize, fileExtName, createMateData());

*log*.debug("--上传成功");

} **catch** (FileNotFoundException e) {

e.printStackTrace();

} **finally** {

**if** (**null** != in) {

**try** {

in.close();

} **catch** (IOException e) {

e.printStackTrace();

}

}

}

1. FastDFSClient接口使用（简易版）

1.方法名：uploadFile

说明：上传文件

方法体：String uploadFile(MultipartFile file) **throws** IOException

参数说明：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 说明 |
| file | MultipartFile或者File |  |

返回参数说明：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 说明 |
| fileName | String | 返回访问图片的地址 |

2.方法名：uploadFile

说明：将一段字符串生成一个文件上传

方法体：String uploadFile(String content, String fileExtension)

参数说明：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 说明 |
| content | String | 内容 |
| fileExtension | String | 文件名后缀 |

返回参数说明：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 说明 |
| fileName | String | 返回访问图片的地址 |

3.方法名：deleteFile

说明：删除文件

方法体： **void** deleteFile(String fileUrl)

参数说明：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 说明 |
| fileUrl | String | 文件路径 |

返回参数说明：

无

4.方法名：uploadImageAndCrtThumbImage

说明：上传图片并且生成缩略图（可自定义大小）

方法体： String uploadImageAndCrtThumbImage(File file,**int** width,**int** height) **throws** IOException

参数说明：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 说明 |
| file | File | 必填参数 |
| width | **int** | 宽，默认150，可选参数 |
| height | **int** | 高，默认150，可选参数 |

返回参数说明：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 说明 |
| filePath | String | 访问图片路径 |

补充：获取缩略图路径：FastDFSUtil.*getThumbImagePath*(uploadFile);

5.方法名：downloadFile

说明：下载整个文件

方法体： **byte**[] downloadFile(String fileUrl) **throws** IOException

参数说明：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 说明 |
| fileUrl | String | 图片路径 |

返回参数说明：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 说明 |
| **byte** | **byte**[] |  |

补充：如需更加灵活的处理，可使用FastFileStorageClient接口,如果开启防盗链功能：在类中注入FastDFSUtil,然后调用getUrlPermissions方法，得到权限url，例：fastDFSUtil.getUrlPermissions(path);

简易版接口使用说明

可参考如下代码：

@Autowired

**private** FastDFSClient fastDFSClient;

@Autowired

**private** FastDFSUtil fastDFSUtil;

/\*\* 日志 \*/

**protected** **static** Logger *log* = LoggerFactory.*getLogger*(FastDfsController.**class**);

/\*\*

\* 上传文件

\* **@param** file 文件对象

\* **@return** 文件访问地址

\* **@throws** IOException

\*/

@RequestMapping(value = "/uploadFile")

**public** String uploadFile(){

String uploadFile="";

**try** {

File file=**new** File("F:/hjw/gaofutongworkspace/share/projects/web/web-fastdfs-base/src/main/resources/images/gs.jpg");

uploadFile = fastDFSClient.uploadFile(file);

//获得图片能够访问的url

uploadFile=fastDFSUtil.getUrlPermissions(uploadFile);

*log*.info("--上传图片成功：URL：{}",uploadFile);

} **catch** (Exception e) {

*log*.error("--异常：{}",JacksonUtil.*toJson*(e));

**return** "上传图片出现异常";

}

**return** "上传图片成功：图片URL："+uploadFile;

}

/\*\*

\* 删除文件

\* **@return**

\*/

@RequestMapping(value = "/deleteFile")

**public** String deleteFile(@RequestParam(value="url",defaultValue="") String url){

**try** {

fastDFSClient.deleteFile(url);

*log*.info("--删除图片成功：URL：{}",url);

} **catch** (Exception e) {

*log*.error("--异常：{}",JacksonUtil.*toJson*(e));

**return** "删除图片出现异常";

}

**return** "删除图片成功：图片URL："+url;

}

/\*\*

\* 上传图片并且生成缩略图

\* **@param** file 文件对象

\* **@return** 文件访问地址

\* **@throws** IOException

\*/

@RequestMapping(value = "/uploadImageAndCrtThumbImage")

**public** String uploadImageAndCrtThumbImage(){

String uploadFile="";

**try** {

File file=**new** File("F:/hjw/gaofutongworkspace/share/projects/web/web-fastdfs-base/src/main/resources/images/gs.jpg");

uploadFile = fastDFSClient.uploadImageAndCrtThumbImage(file);

*log*.info("--上传图片成功：URL：{}",uploadFile);

*log*.info("--缩略图：URL：{}",FastDFSUtil.*getThumbImagePath*(uploadFile));

} **catch** (Exception e) {

*log*.error("--异常：{}",JacksonUtil.*toJson*(e));

**return** "上传图片出现异常";

}

**return** "上传图片成功：图片URL："+uploadFile+" :缩略图url："+FastDFSUtil.*getThumbImagePath*(uploadFile);

}

/\*\*

\* 下载

\* **@param** url

\* **@return**

\*/

@RequestMapping(value = "/downloadFile")

**public** String downloadFile(@RequestParam(value="url",defaultValue="") String url){

**byte**[] downloadFile=**null**;

**try** {

downloadFile = fastDFSClient.downloadFile(url);

*log*.info("--下载图片成功：URL：{}",downloadFile);

} **catch** (Exception e) {

*log*.error("--异常：{}",JacksonUtil.*toJson*(e));

**return** "下载图片出现异常";

}

**return** "下载图片成功：内容："+downloadFile;

}