编写FastDFS工具类

FastDFSClient.java

```
package com.xxxx.fastdfsdemo;

import org.csource.common.NameValuePair;
import org.csource.fastdfs.ClientGlobal;
import org.csource.fastdfs.FileInfo;
import org.csource.fastdfs.StorageClient;
import org.csource.fastdfs.TrackerClient;
import org.csource.fastdfs.TrackerServer;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.core.io.ClassPathResource;

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

/**
 * 文件上传工具类
 *
 * @author zhoubin
 * @since 1.0.0
 */
public class FastDFSClient {

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

   //ClientGlobal.init 方法会读取配置文件，并初始化对应的属性。
   static{
      try {
         String filePath = new
ClassPathResource("fdfs_client.conf").getFile().getAbsolutePath();
         ClientGlobal.init(filePath);
      } catch (Exception e) {
         logger.error("FastDFS Client Init Fail!",e);
      }
   }

   /**
    * 上传文件
    * @param file
    * @return
    */
   public static String[] upload(FastDFSFile file) {
      logger.info("File Name: " + file.getName() + "File Length:" +
file.getContent().length);

      //文件属性信息
```

```
        NameValuePair[] meta_list = new NameValuePair[1];
        meta_list[0] = new NameValuePair("author", file.getAuthor());

        long startTime = System.currentTimeMillis();
        String[] uploadResults = null;
        StorageClient storageClient=null;
        try {
            //获取storage客户端
            storageClient = getStorageClient();
            //上传
            uploadResults = storageClient.upload_file(file.getContent(), file.getExt(),
meta_list);
        } catch (IOException e) {
            logger.error("IO Exception when uploadind the file:" + file.getName(), e);
        } catch (Exception e) {
            logger.error("Non IO Exception when uploadind the file:" + file.getName(), e);
        }
        logger.info("upload_file time used:" + (System.currentTimeMillis() - startTime) + "
ms");

        //验证上传结果
        if (uploadResults == null && storageClient!=null) {
            logger.error("upload file fail, error code:" + storageClient.getErrorCode());
        }
        //上传文件成功会返回 groupName。
        logger.info("upload file successfully!!!" + "group_name:" + uploadResults[0] + ",
remoteFileName:" + " " + uploadResults[1]);
        return uploadResults;
    }


    /**
     * 获取文件信息
     * @param groupName
     * @param remoteFileName
     * @return
     */
    public static FileInfo getFile(String groupName, String remoteFileName) {
        try {
            StorageClient storageClient = getStorageClient();
            return storageClient.get_file_info(groupName, remoteFileName);
        } catch (IOException e) {
            logger.error("IO Exception: Get File from Fast DFS failed", e);
        } catch (Exception e) {
            logger.error("Non IO Exception: Get File from Fast DFS failed", e);
        }
        return null;
    }

    /**
     * 下载文件
     * @param groupName
     * @param remoteFileName
```

```
     * @return
     */
    public static InputStream downFile(String groupName, String remoteFileName) {
        try {
            StorageClient storageClient = getStorageClient();
            byte[] fileByte = storageClient.download_file(groupName, remoteFileName);
            InputStream ins = new ByteArrayInputStream(fileByte);
            return ins;
        } catch (IOException e) {
            logger.error("IO Exception: Get File from Fast DFS failed", e);
        } catch (Exception e) {
            logger.error("Non IO Exception: Get File from Fast DFS failed", e);
        }
        return null;
    }

    /**
     * 删除文件
     * @param groupName
     * @param remoteFileName
     * @throws Exception
     */
    public static void deleteFile(String groupName, String remoteFileName)
            throws Exception {
        StorageClient storageClient = getStorageClient();
        int i = storageClient.delete_file(groupName, remoteFileName);
        logger.info("delete file successfully!!!" + i);
    }


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

    /**
     * 生成Tracker服务器端
     * @return
     * @throws IOException
     */
    private static TrackerServer getTrackerServer() throws IOException {
        TrackerClient trackerClient = new TrackerClient();
        TrackerServer trackerServer = trackerClient.getTrackerServer();
        return  trackerServer;
    }

    /**
```

```
     * 获取文件路径
     * @return
     * @throws IOException
     */
    public static String getTrackerUrl() throws Exception {
      TrackerClient trackerClient = new TrackerClient();
      TrackerServer trackerServer = trackerClient.getTrackerServer();
      StorageServer storeStorage = trackerClient.getStoreStorage(trackerServer);
      return "http://"+storeStorage.getInetSocketAddress().getHostString()+":8888/";
    }
}
```

编写Controller

```
package com.xxxx.fastdfsdemo;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.multipart.MultipartFile;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;

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

/**
 * 文件上传Controller
 * @author zhoubin
 */
@Controller
public class UploadController {
    private static Logger logger = LoggerFactory.getLogger(UploadController.class);

    /**
     * 页面跳转
     * @return
     */
    @GetMapping("/")
    public String index() {
        return "upload";
    }

    /**
     * 上传文件
     * @param file
     * @param redirectAttributes
     * @return
     */
    @PostMapping("/upload")
    public String singleFileUpload(@RequestParam("file") MultipartFile file,
```

```
                                    RedirectAttributes redirectAttributes) {
        if (file.isEmpty()) {
            redirectAttributes.addFlashAttribute("message", "Please select a file to
upload");
            return "redirect:uploadStatus";
        }
        try {
            // 上传文件拿到返回的文件路径
            String path=saveFile(file);
            redirectAttributes.addFlashAttribute("message",
                    "You successfully uploaded '" + file.getOriginalFilename() + "'");
            redirectAttributes.addFlashAttribute("path",
                    "file path url '" + path + "'");
        } catch (Exception e) {
            logger.error("upload file failed",e);
        }
        return "redirect:/uploadStatus";
    }

    /**
     * 页面跳转
     * @return
     */
    @GetMapping("/uploadStatus")
    public String uploadStatus() {
        return "uploadStatus";
    }

    /**
     * 上传文件
     * @param multipartFile
     * @return
     * @throws IOException
     */
    public String saveFile(MultipartFile multipartFile) throws Exception {
        String[] fileAbsolutePath={};
        String fileName=multipartFile.getOriginalFilename();
        String ext = fileName.substring(fileName.lastIndexOf(".") + 1);
        byte[] file_buff = null;
        InputStream inputStream=multipartFile.getInputStream();
        if(inputStream!=null){
            int len1 = inputStream.available();
            file_buff = new byte[len1];
            inputStream.read(file_buff);
        }
        inputStream.close();
        FastDFSFile file = new FastDFSFile(fileName, file_buff, ext);
        try {
            //上传文件
            fileAbsolutePath = FastDFSClient.upload(file);
        } catch (Exception e) {
            logger.error("upload file Exception!",e);
        }
```

```
        if (fileAbsolutePath==null) {
            logger.error("upload file failed,please upload again!");
        }
        String path=FastDFSClient.getTrackerUrl()+fileAbsolutePath[0]+
"/"+fileAbsolutePath[1];
        return path;
    }
}
```