**程序设计说明书**

目录

[一、 设计目的 2](#_Toc73624976)

[二、 设计目标与设计说明 2](#_Toc73624977)

[1. 任务和目标 2](#_Toc73624978)

[2. 运行环境概述 2](#_Toc73624979)

[3. 条件与限制 3](#_Toc73624980)

[三、 具体设计步骤 3](#_Toc73624981)

[四、 总结和心得 4](#_Toc73624982)

# 设计目的

此设计是对S3 Browser进行文件同步、上传、下载、删除等基本操作的综合运用，学习使用JAVA-AWS-SDK实现上述操作。

# 设计目标与设计说明

## 任务和目标

1. 指定本地目录与S3的某个Bucket进行文件同步。
2. 程序启动时把Bucket中的文件同步到本地，需要处理文件冲突。
3. 本地添加、修改了文件，需要上传到S3；本地删除了文件，也要删除S3上对应的文件。
4. 对于超过20MB的文件，需要使用分块上传/下载、传输中断、程序重启可以继续原来的进度。

## 运行环境概述

硬件环境：一台连接网络的电脑

软件环境：

* 操作系统：Windows10
* 数据库系统：无
* 软件：OpenVPN, Eclipse, S3 Browser
* 第三方工具包：JDK1.8, JAVA-AWS-SDK

## 条件与限制

# 具体设计步骤

1. 设计如何连接S3 Browser账户以及桶。
2. 设计基本的功能，如上传、下载、删除、获取桶内文件目录：
3. 上传 **public** **static** **void** uploadFile(String object)：

首先判断本地文件大小是否大于20MB，是则启用分块上传；否则直接进行上传。

1. 下载 **public** **static** **void** downloadFile(String object):

类似于下载，首先判断桶内文件大小是否大于20MB，是则启用分块下载；否则直接进行下载。

1. 删除：

直接使用AmazonS3的API，可直接删除桶内的文件。

1. 获取桶内文件目录 **public** **static** List<String> getFile():

通过创建ListObjectsRequest和ObjectListing对象，可获取到桶内的文件目录，最终返回一个String类型的列表。

1. 设计复杂功能，如同步本地文件至桶，同步桶内文件至本地：
2. 同步本地文件至桶 **public** **static** **void** sync()：
   1. 若本地文件进行了增加或删除，则直接比较本地文件目录和桶内文件目录即可找到本地增加或删除的文件，再将其上传至桶或将其从桶内删除。
   2. 若本地文件进行了修改，则需要比较本地文件和桶内同文件的ETag值：

**public** **static** String calculateETag(String object) **public** **static** **boolean** compare(String object)

找出修改过的文件名，再通过此文件名将文件上传至桶。

1. 同步桶内文件至本地 **public** **static** **void** downloadAllFile()：

直接将桶内所有的文件下载到本地即可。

# 总结和心得

通过设计、开发这个简单的文件同步器，我对通过JAVA-AWS-SDK操作S3 Browser有了更加深入的了解，熟悉基本的上传、下载、删除、同步等基本功能。虽然在此过程中遇到许多困难，从一开始的无从下手，到查阅各类文档资料，最终还是粗略地完成了这个程序。相信在接下来的课程中，我可以面对更难的问题和挑战。