对象存储

(Object-Oriented Storage,OOS)

Java SDK 开发者指南

中国电信股份有限公司

云计算分公司

目录

1. 简介

本文档主要介绍 OOS Java SDK 的安装和使用。本文档假设您已经开通了对象存储 OOS 服务，并创建了 AccessKeyId 和 SecretKey。OOS API 服务端地址请参见 OOS 开发者文档。

2. 配置 SDK

导入 SDK 包到项目中

首先将 OOS Java SDK 添加到编译路径中。

下载第三方包，并添加到编译路径中，地址：

https://oos-cn.ctyunapi.cn/sdk/oos/java/third-party.zip

这样即可以在项目中调用 OOS SDK。

使用 SDK 访问服务器

首先，创建 Client 对象，用于向服务器端发送请求，java 代码如下：

```
  AmazonS3 client= new AmazonS3Client(new
PropertiesCredentials(S3Sample.class.getResourceAsStream("AwsCrede
ntials.properties")));
  client.setEndpoint(http://oos.ctyunapi.cn);
```

其中 AwsCredentials.properties 用于存储 accessKey 和 secretKey(此信息可以在控制台—账号管理—API 密钥管理中查看到)，例如：

```
  accessKey =test
  secretKey =test
```

http://oos.ctyunapi.cn 是 OOS 服务器的地址。

然后就可以调用 SDK 中的方法，对 Service,Bucket,Object 进行 PUT,GET,DELETE,HEAD 等操作。

3. SDK FAQ

1. 如何设置请求的重试功能

在创建 AmazonS3Client 时，可以配置客户端发送请求的最大重试次数，当服务端返回 5XX 错误，或连接超时等错误信息时，AmazonS3Client 会自动重发请求。配置方法如下：

```
ClientConfiguration cc = newClientConfiguration();
cc.setMaxErrorRetry(2);//这里设置重试两次，如果不设置的话，默认重试三次
AmazonS3Client oosclient = newAmazonS3Client(new AWSCredentials() {
public String getAWSAccessKeyId() {
return"yourAccessKey";
        }

public String getAWSSecretKey() {
return"yourSecretKey";
        }
   }, cc);
```

2. 如何设置超时时间

在创建 AmazonS3Client 时，可以配置客户端到服务器端的连接超时和 socket 超时时间，代码示例如下：

```
ClientConfiguration cc = newClientConfiguration();
cc.setConnectionTimeout(30*1000);//设置连接的超时时间，单位毫秒
cc.setSocketTimeout(30*1000) ;//设置socket超时时间，单位毫秒
AmazonS3Client oosclient = newAmazonS3Client(new AWSCredentials() {
public String getAWSAccessKeyId() {
return"yourAccessKey";
        }

public String getAWSSecretKey() {
return"yourSecretKey";
        }
   }, cc);
```

3. 如何设置 https 的 endpoint

通过 https 访问 OOS 服务的代码示例如下：

```
  System.setProperty("com.amazonaws.sdk.disableCertChecking",
"true");
  oosclient.setEndpoint("https://oos.ctyunapi.cn");//设置通过 https 方
式访问 oos 服务
```

4. 代码示例

OOSCredentials.properties

用于存储用户名和密码，例如：

```
accessKey =yourAccessKey
secretKey =yourSecretKey
```

OOSSample.java 文件

```
package com.amazonaws.services.s3;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.io.Writer;

import com.amazonaws.AmazonClientException;
import com.amazonaws.AmazonServiceException;
import com.amazonaws.auth.PropertiesCredentials;
import com.amazonaws.services.s3.model.Bucket;
import com.amazonaws.services.s3.model.GetObjectRequest;
import com.amazonaws.services.s3.model.ListObjectsRequest;
import com.amazonaws.services.s3.model.ObjectListing;
import com.amazonaws.services.s3.model.PutObjectRequest;
```

```
import com.amazonaws.services.s3.model.S3Object;
import com.amazonaws.services.s3.model.S3ObjectSummary;

public class OOSSample {
public static void main(String[] args) throws IOException {
  /* 创建 client,其中 OOSCredentials.properties 中存放着用户名和密码*/
  AmazonS3 oos = new AmazonS3Client(new PropertiesCredentials(
             OOSSample.class
                    .getResourceAsStream("OOSCredentials.properties")))
;
  // 设置 endpoint 到 OOS
  oos.setEndpoint("http://oos.ctyunapi.cn");
  String bucketName = "my-first-oos-bucket";
  String key = "MyObjectKey";
  System.out.println("===========================================");
    System.out.println("Getting Started with OOS");
  System.out.println("===========================================");
  try {
      /* 创建 bucket */
      System.out.println("Creating bucket " + bucketName + "\n");
      oos.createBucket(bucketName);

      /*列出账户内的所有 buckets */
      System.out.println("Listing buckets");
      for (Bucket bucket : oos.listBuckets()) {
       System.out.println(" - " + bucket.getName());
                }
       System.out.println();

       /* 上传一个 object 到 bucket 中 */
       System.out.println("Uploading a new object to OOS from a file\n");
       oos.putObject(new PutObjectRequest(bucketName, key,
       createSampleFile()));

       /* 下载 object */
     System.out.println("Downloading an object");
     /* 当使用getObject()方法时，需要非常小心。因为返回的S3Object对象包括一个从
  HTTP连接获得的数据流。底层的HTTP连接不会被关闭，直到用户读完了数据，并关闭了流。因此
  从S3Object中读取inputStream数据后，需要立刻关闭流。否则会导致客户端连接池用满 */
       S3Object object = oos.getObject(new GetObjectRequest(bucketName,
key));
```

```
        System.out.println("Content-Type: "
        + object.getObjectMetadata().getContentType());
        System.out.println("Content:");
        displayTextInputStream(object.getObjectContent());

        /* 拷贝 object */
          String destinationBucketName = "my-copy-oos-bucket";
          String destinationKey = "MyCopyKey";
        System.out.println("Copying an object ,from " + bucketName + "/"
                    + key + " to " + destinationBucketName + "/"
                    + destinationKey);
        oos.createBucket(destinationBucketName);
        oos.copyObject(bucketName, key, destinationBucketName,
        destinationKey);

        /* 下载拷贝的 object */
        System.out.println("Downloading the " + destinationKey + " object");
        object = oos.getObject(new GetObjectRequest(destinationBucketName,
destinationKey));
        System.out.println("Content-Type: "
                    + object.getObjectMetadata().getContentType());
        System.out.println("Content:");
        displayTextInputStream(object.getObjectContent());

        /* 列出 bucket 中的 object，支持 prefix,delimiter,marker,max-keys 等选项 */
        System.out.println("Listing objects");
          ObjectListing objectListing = oos
                    .listObjects(new ListObjectsRequest().withBucketName(
bucketName).withPrefix("My"));
        for (S3ObjectSummary objectSummary : objectListing
                    .getObjectSummaries()) {
           System.out.println(" - " + objectSummary.getKey() + "  "
                                + "(size = " + objectSummary.getSize() + ")");
           }
           System.out.println();

           /* 删除 object */
           System.out.println("Deleting objects\n");
        oos.deleteObject(bucketName, key);
        oos.deleteObject(destinationBucketName, destinationKey);
```

```
            /* 删除 bucket */
        System.out.println("Deleting bucket " + bucketName + "\n");
        oos.deleteBucket(bucketName);
        System.out.println("Deleting bucket " + destinationBucketName
                    + "\n");
        oos.deleteBucket(destinationBucketName);
        } catch (AmazonServiceException ase) {
        System.out.println("Caught an AmazonServiceException, which means your
  request made it "
         + "to OOS, but was rejected with an error response for some reason.");
        System.out.println("Error Message:    " + ase.getMessage());
        System.out.println("HTTP Status Code: " + ase.getStatusCode());
        System.out.println("OOS Error Code:   " + ase.getErrorCode());
        System.out.println("Request ID:       " + ase.getRequestId());
         } catch (AmazonClientException ace) {
          System.out.println("Caught an AmazonClientException, which means the
        client encountered "
          + "a serious internal problem while trying to communicate with OOS, "
        + "such as not being able to access the network.");
        System.out.println("Error Message: " + ace.getMessage());
         }
    }

Private static File createSampleFile() throws IOException {
         File file = File.createTempFile("oos-java-sdk-", ".txt");
file.deleteOnExit();
         Writer writer = new OutputStreamWriter(new FileOutputStream(file));
        writer.write("abcdefghijklmnopqrstuvwxyz\n");
        writer.write("01234567890112345678901234\n");
        writer.write("!@#$%^&*()-=[]{};':',.<>/?\n");
        writer.write("01234567890112345678901234\n");
        writer.write("abcdefghijklmnopqrstuvwxyz\n");
        writer.close();
        return file;
    }

Private static void displayTextInputStream(InputStream input)
throws IOException {
  BufferedReader reader = newBufferedReader(new InputStreamReader(input));
  while (true) {
          String line = reader.readLine();
```

```
    if (line == null)
    break;
    System.out.println("    " + line);
        }
         /* 需要在这里关闭InputStream，原因参见getObject处 */
       input.close();
    System.out.println();
    }
}
```

5. SDK 版本说明

目前 OOS 提供以下几个版本的 SDK：

1. oos-sdk-2.0.0.jar

   此版本的 SDK 兼容亚马逊 SDK 的 1.3.15 版本。

2. oos-sdk-2.1.0.jar

   此版本的 SDK 兼容亚马逊 SDK 的 1.4.7 版本。

   增加了 AccessKey 和 SecretKey 的相关接口，这些方法以 ctyun 为开头命名。

3. oos-sdk-5.0.0.jar

   此版本的 SDK 兼容亚马逊 SDK 的 1.4.7 版本。

   增加了异地互备相关的接口说明，可以实现不同资源池之间的数据互备，参见 PUT/GET/DELETE Bucket Trigger 等接口。同时用户可以记录异地互备相关的日志，参见 PUT/GET Bucket Trigger 接口。

   增加了断点续传接口。

4. oos-sdk-6.0.0.jar

增加 bucket 索引位置、数据位置的相关接口，参见 PUT Bucket 接口。