springboot对接阿里云视频点播

原创

阿演 于 2021-07-31 12:04:50 发布 162 收藏 1
分类专栏： springboot 文章标签： 阿里云 java 视频点播

> 在阿里云的视频点播官方文档中，可以看到是有一个上传SDK和一个服务端SDK的，上传视频可以用**上传SDK里面的服务端SDK里面的java上传SDK**，其他操作查询删除什么的只能用**服务端SDK里面的javaSDK**

点开java上传SDK，可以看到有一个demo可以下载

下载这个demo，这个demo不是一个maven项目

根据文档中写的，引入maven依赖，然后把demo里面的代码复制到我们的项目，发现里面有几个类是找不到的，比如UploadStreamRequest，这些类所在的包不是开源的，在demo中是有这个jar包的，就是下面这个包，自己用命令打到本地仓库或者私库。

```
<dependency>
    <groupId>com.aliyun</groupId>
    <artifactId>aliyun-java-vod-upload</artifactId>
    <version>1.4.14</version>
</dependency>
```

全部依赖：

```
<!-- 阿里云视频点播 -->
        <dependency>
            <groupId>com.aliyun</groupId>
            <artifactId>aliyun-java-sdk-core</artifactId>
            <version>4.5.1</version>
        </dependency>
        <dependency>
            <groupId>com.aliyun</groupId>
            <artifactId>aliyun-java-sdk-vod</artifactId>
            <version>2.15.11</version>
        </dependency>
        <dependency>
            <groupId>com.aliyun</groupId>
            <artifactId>aliyun-java-vod-upload</artifactId>
            <version>1.4.14</version>
        </dependency>
        <dependency>
            <groupId>com.alibaba</groupId>
            <artifactId>fastjson</artifactId>
            <version>1.2.62</version>
        </dependency>
        <dependency>
            <groupId>com.aliyun.oss</groupId>
            <artifactId>aliyun-sdk-oss</artifactId>
            <version>3.1.0</version>
        </dependency>
```

然后我一开始用流方式上传一直提示连接超时，阿里云控制台也是一直显示上传中。用本地文件上传的方式就可以成功上传，不确定什么原因，可能是网络原因。

过了几个小时再去用流方式上传就可以上传成功了（这个问了阿里云客服也没说清楚什么原因）

代码如下

```
import com.aliyun.vod.upload.impl.*;
import com.aliyun.vod.upload.req.*;
import com.aliyun.vod.upload.resp.*;
import org.springframework.beans.factory.annotation.Value;
import org.springframework.stereotype.Component;

import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.InputStream;
import java.net.URL;

/**
 * 阿里云上传音视频
 */
@Component
public class UploadVideoConfiguration {

    //账号AK信息请填写(必选)
    @Value("${aliyun.accessKeyId}")
```

```
    private String accessKeyId;
    //账号AK信息请填写(必选)
    @Value("${aliyun.accessKeySecret}")
    private String accessKeySecret;

    @Value("${aliyun.endPoint}")
    private String endPoint;

    private String callBackUrl;

    /**
     * 流式上传接口
     *
     * @param title
     * @param fileName
     * @param inputStream
     */
    public UploadStreamResponse testUploadStream(String title, String fileName, InputStream inputStream,Lon
                                                 String key) throws IOException {
        UploadStreamRequest request = new UploadStreamRequest(accessKeyId, accessKeySecret, title, fileName
        /* 是否使用默认水印(可选)，指定模板组ID时，根据模板组配置确定是否使用默认水印*/
        //request.setShowWaterMark(true);
        /* 自定义消息回调设置，参数说明参考文档 https://help.aliyun.com/document_detail/86952.html#UserData */
//        request.setUserData("{\"Extend\":{\"test\":\"www\",\"localId\":\"xxxx\"}," +
//                "\"MessageCallback\":{\"CallbackURL\":\"" + callBackUrl + "\"}}");
        /* 视频分类ID(可选) */
        request.setCateId(cateId);
        /* 视频标签,多个用逗号分隔(可选) */
        //request.setTags("标签1,标签2");
        /* 视频描述(可选) */
        //request.setDescription("视频描述");
        /* 封面图片(可选) */
        //request.setCoverURL("http://cover.sample.com/sample.jpg");
        /* 模板组ID(可选) */
        //request.setTemplateGroupId("8c4792cbc8694e7084fd5330e56a33d");
        /* 工作流ID(可选) */
        //request.setWorkflowId("d4430d07361f0*be1339577859b0177b");
        /* 存储区域(可选) */
//        request.setStorageLocation(endPoint);
        /* 开启默认上传进度回调 */
        request.setPrintProgress(true);
        /* 设置自定义上传进度回调 (必须继承 VoDProgressListener) */
//        request.setProgressListener(new PutObjectProgressListener());
        request.setProgressListener(new MyVoDProgressListener(inputStream.available(),key));
        /* 设置应用ID*/
        //request.setAppId("app-1000000");
        /* 点播服务接入点 */
//        request.setApiRegionId("cn-shanghai");
        /* ECS部署区域*/
//        request.setEcsRegionId("cn-shanghai");

        UploadVideoImpl uploader = new UploadVideoImpl();
        UploadStreamResponse response = uploader.uploadStream(request);
        System.out.print("RequestId=" + response.getRequestId() + "\n");  //请求视频点播服务的请求ID
        if (response.isSuccess()) {
            System.out.println("上传成功！！！");
            System.out.print("VideoId=" + response.getVideoId() + "\n");
        } else { //如果设置回调URL无效，不影响视频上传，可以返回VideoId同时会返回错误码。其他情况上传失败时，VideoId为
            System.out.print("VideoId=" + response.getVideoId() + "\n");
            System.out.print("ErrorCode=" + response.getCode() + "\n");
```

```
        System.out.print("ErrorMessage=" + response.getMessage() + "\n");
    }
    return response;

}

/**
 * 本地文件上传接口
 *
 * @param title
 * @param fileName
 */
public UploadVideoResponse testUploadVideo(String title, String fileName) {
    UploadVideoRequest request = new UploadVideoRequest(accessKeyId, accessKeySecret, title, fileName);
    /* 可指定分片上传时每个分片的大小，默认为2M字节 */
    request.setPartSize(2 * 1024 * 1024L);
    /* 可指定分片上传时的并发线程数，默认为1，(注：该配置会占用服务器CPU资源，需根据服务器情况指定）*/
    request.setTaskNum(1);
    /* 是否开启断点续传，默认断点续传功能关闭。当网络不稳定或者程序崩溃时，再次发起相同上传请求，可以继续未完成的上传
    注意：断点续传开启后，会在上传过程中将上传位置写入本地磁盘文件，影响文件上传速度，请您根据实际情况选择是否开启*/
    //request.setEnableCheckpoint(false);
    /* OSS慢请求日志打印超时时间，是指每个分片上传时间超过该阈值时会打印debug日志，如果想屏蔽此日志，请调整该阈值。单
    //request.setSlowRequestsThreshold(300000L);
    /* 可指定每个分片慢请求时打印日志的时间阈值，默认为300s*/
    //request.setSlowReque stsThreshold(300000L);
    /* 是否显示水印(可选)，指定模板组ID时，根据模板组配置确定是否显示水印*/
    //request.setIsShowWaterMark(true);
    /* 自定义消息回调设置(可选)，参数说明参考文档 https://help.aliyun.com/document_detail/86952.html#UserDat
    // request.setUserData("{\"Extend\":{\"test\":\"www\",\"localId\":\"xxxx\"},\"MessageCallback\":{\"
    /* 视频分类ID(可选) */
    request.setCateId(1000325496L);
    /* 视频标签,多个用逗号分隔(可选) */
    //request.setTags("标签1,标签2");
    /* 视频描述(可选) */
    //request.setDescription("视频描述");
    /* 封面图片(可选) */
    //request.setCoverURL("http://cover.sample.com/sample.jpg");
    /* 模板组ID(可选) */
    //request.setTemplateGroupId("8c4792cbc8694e7084fd5330e56a33d");
    /* 工作流ID(可选) */
    //request.setWorkflowId("d4430d07361f0*be1339577859b0177b");
    /* 存储区域(可选) */
    //request.setStorageLocation("in-201703232118266-5sejdln9o.oss-cn-shanghai.aliyuncs.com");
    /* 开启默认上传进度回调 */
    //request.setPrintProgress(false);
    /* 设置自定义上传进度回调 (必须继承 VoDProgressListener) */
    //request.setProgressListener(new PutObjectProgressListener());
    /* 设置您实现的生成STS信息的接口实现类*/
    // request.setVoDRefreshSTSTokenListener(new RefreshSTSTokenImpl());
    /* 设置应用ID*/
    //request.setAppId("app-1000000");
    /* 点播服务接入点 */
    //request.setApiRegionId("cn-shanghai");
    /* ECS部署区域*/
    // request.setEcsRegionId("cn-shanghai");
    UploadVideoImpl uploader = new UploadVideoImpl();
    UploadVideoResponse response = uploader.uploadVideo(request);
    System.out.print("RequestId=" + response.getRequestId() + "\n");  //请求视频点播服务的请求ID
    if (response.isSuccess()) {
```

```
            System.out.print("VideoId=" + response.getVideoId() + "\n");
        } else {
            /* 如果设置回调URL无效，不影响视频上传，可以返回VideoId同时会返回错误码。其他情况上传失败时，VideoId为空，
            System.out.print("VideoId=" + response.getVideoId() + "\n");
            System.out.print("ErrorCode=" + response.getCode() + "\n");
            System.out.print("ErrorMessage=" + response.getMessage() + "\n");
        }
        return response;
    }
}
```

这边这个进度监听器是自己继承它指定的那个监听器类实现的，为了获取上传进度给前端用

```
/**
 * 阿里云上传视频获取进度监听器
 */
public class MyVoDProgressListener implements VoDProgressListener {

    private long bytesWritten = 0L;
    private long totalBytes = -1L;
    private boolean succeed = false;
    private String videoId;
    private String imageId;

    private String key;

    private static String redisHost = "127.0.0.1";
    private static int redisPort = 6379;
    private static String redisPass = "";

    public MyVoDProgressListener(long totalBytes,String key) {
        this.key = key;
        this.totalBytes = totalBytes;
    }

    public void progressChanged(ProgressEvent progressEvent) {
        long bytes = progressEvent.getBytes();
        ProgressEventType eventType = progressEvent.getEventType();
        switch(eventType) {
            case TRANSFER_STARTED_EVENT:
                if (this.videoId != null) {
                    System.out.println("开始上传ID为" + this.videoId + "的视频");
                }

                if (this.imageId != null) {
                    System.out.println("Start to upload imageId " + this.imageId + "......");
                }
                break;
            case REQUEST_CONTENT_LENGTH_EVENT:
                this.totalBytes = bytes;
                System.out.println(this.totalBytes + " bytes in total will be uploaded to OSS.");
                break;
            case REQUEST_BYTE_TRANSFER_EVENT:
                this.bytesWritten += bytes;
                if (this.totalBytes != -1L) {
                    int percent = (int)((double)this.bytesWritten * 100.0D / (double)this.totalBytes);
                    //上传百分比存到redis，接口取出来返给前端
                    Jedis jedis = JedisUtil.getInstance();
```

```
                    jedis.set(this.key, String.valueOf(percent));
                    if(percent == 100){
                        jedis.close();
                    }
                    System.out.println(bytes + " bytes have been written at this time, upload progress: " +
                } else {
                    System.out.println(bytes + " bytes have been written at this time, upload sub total : (
                }
                break;
            case TRANSFER_COMPLETED_EVENT:
                this.succeed = true;
                if (this.videoId != null) {
                    System.out.println("视频ID为 " + this.videoId + " , 总写入字节" + this.bytesWritten + "上
                }

                if (this.imageId != null) {
                    System.out.println("Succeed to upload imageId " + this.imageId + " , " + this.bytesWrit
                }
                break;
            case TRANSFER_FAILED_EVENT:
                if (this.videoId != null) {
                    System.out.println("视频ID为 " + this.videoId + " , 总写入字节" + this.bytesWritten + "上
                }

                if (this.imageId != null) {
                    System.out.println("Failed to upload imageId " + this.imageId + " , " + this.bytesWritt
                }
        }

    }

    public boolean isSucceed() {
        return this.succeed;
    }

    public void onVidReady(String videoId) {
        this.setVideoId(videoId);
    }

    public void onImageIdReady(String imageId) {
        this.setImageId(imageId);
    }

    public String getVideoId() {
        return this.videoId;
    }

    public void setVideoId(String videoId) {
        this.videoId = videoId;
    }

    public String getImageId() {
        return this.imageId;
    }

    public void setImageId(String imageId) {
        this.imageId = imageId;
    }
}
```

到这里上传视频可以了，查询视频信息和删除视频的功能只能用文档里面那个服务端SDK

代码如下：

```
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.vod.model.v20170321.*;
import org.springframework.beans.factory.annotation.Value;
import org.springframework.stereotype.Component;

@Component
public class VideoUtils {

    private DefaultAcsClient client;

    public VideoUtils(@Value("${aliyun.accessKeyId}") String accessKeyId,
                      @Value("${aliyun.accessKeySecret}") String accessKeySecret) {
        String regionId = "cn-shanghai";  // 点播服务接入区域
        DefaultProfile profile = DefaultProfile.getProfile(regionId, accessKeyId, accessKeySecret);
        client = new DefaultAcsClient(profile);
    }

    /**
     * 获取视频信息 带播放地址
     *
     * @param videoId 视频ID
     * @return GetVideoInfoResponse 获取视频信息响应数据
     * @throws Exception
     */
    public GetMezzanineInfoResponse getVideoInfo(String videoId) throws Exception {
        GetMezzanineInfoRequest request = new GetMezzanineInfoRequest();
        request.setVideoId(videoId);
        return client.getAcsResponse(request);
    }

    /**
     * 根据ID删除
     *
     * @param videoIds ID
     * @return 是否成功
     */
    public boolean delVideoByVideoId(String videoIds) throws ClientException {
        DeleteVideoRequest request = new DeleteVideoRequest();
        request.setVideoIds(videoIds);

        client.getAcsResponse(request);
        return true;
    }
}
```