**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

学成在线-第14天-讲义-媒资管理



* 视频处理



1.1需求分析



原始视频通常需要经过编码处理，生成m3u8和ts文件方可基于HLS协议播放视频。通常用户上传原始视频，系统自动处理成标准格式，系统对用户上传的视频自动编码、转换，最终生成m3u8文件和ts文件，处理流程如下：

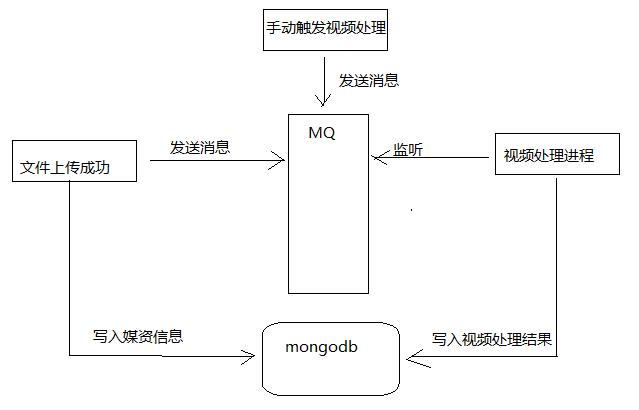
1、用户上传视频成功

2、系统对上传成功的视频自动开始编码处理

3、用户查看视频处理结果，没有处理成功的视频用户可在管理界面再次触发处理

4、视频处理完成将视频地址及处理结果保存到数据库

视频处理流程如下：



视频处理进程的任务是接收视频处理消息进行视频处理，业务流程如下：

1、监听MQ，接收视频处理消息。

2、进行视频处理。

3、向数据库写入视频处理结果。

**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

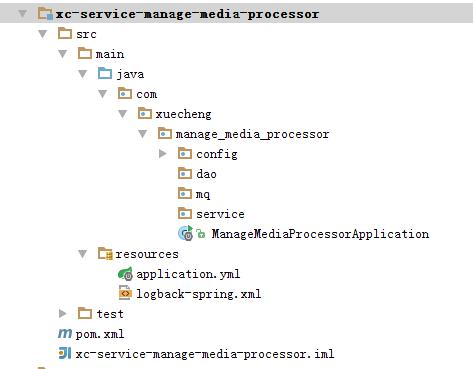
视频处理进程属于媒资管理系统的一部分，考虑提高系统的扩展性，将视频处理单独定义视频处理工程。

1.2 视频处理开发



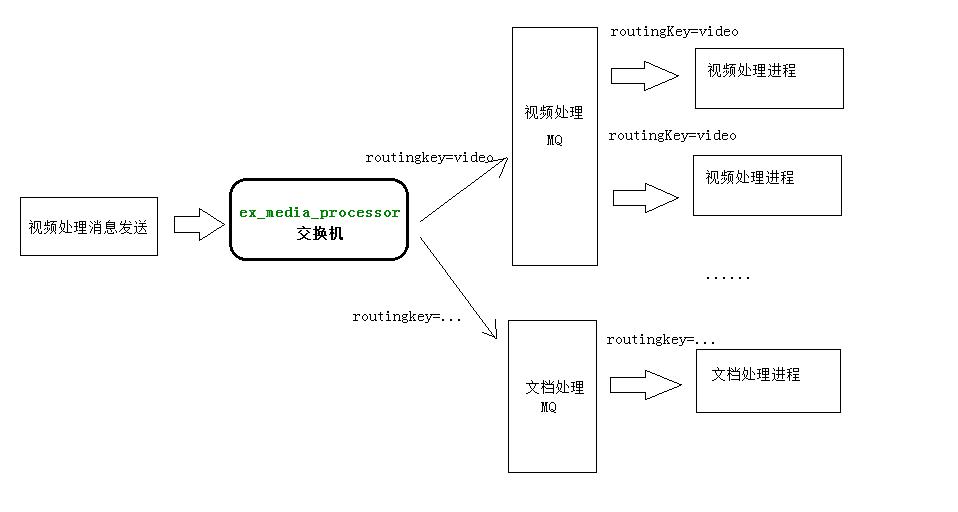
1.2.1 视频处理工程创建

1、导入“资料”下的视频处理工程：xc-service-manage-media-processor



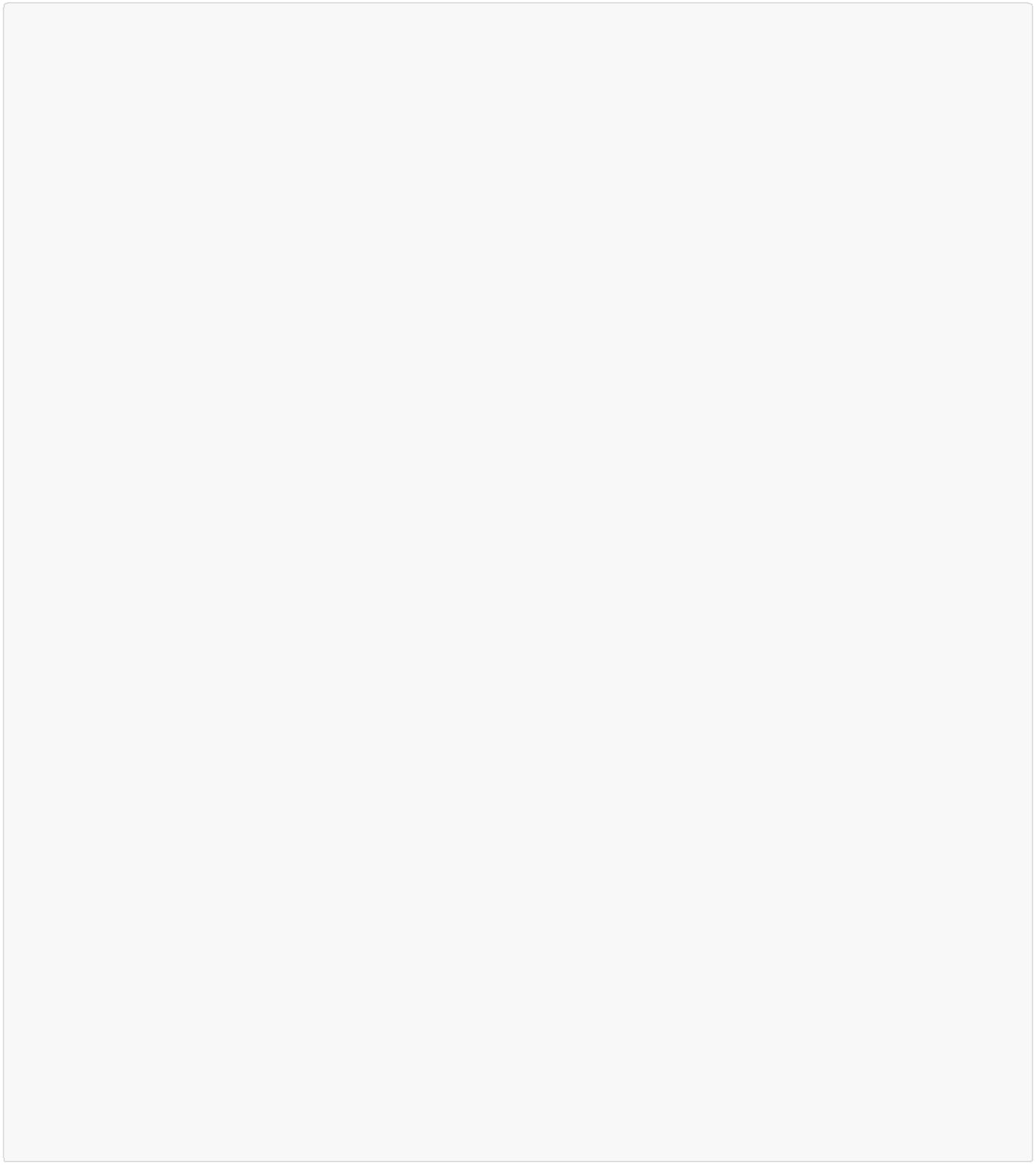
2、RabbitMQ配置

使用rabbitMQ的routing交换机模式，视频处理程序监听视频处理队列，如下图：



**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

RabbitMQ配置如下：



@Configuration

public class RabbitMQConfig {

public static final String EX\_MEDIA\_PROCESSTASK = "ex\_media\_processor";

//视频处理队列

@Value("${xc‐service‐manage‐media.mq.queue‐media‐video‐processor}")

public String queue\_media\_video\_processtask;

//视频处理路由

@Value("${xc‐service‐manage‐media.mq.routingkey‐media‐video}")

public String routingkey\_media\_video;

/\*\*

* 交换机配置
* @return the exchange \*/

@Bean(EX\_MEDIA\_PROCESSTASK)

public Exchange EX\_MEDIA\_VIDEOTASK() {

return ExchangeBuilder.directExchange(EX\_MEDIA\_PROCESSTASK).durable(true).build();

}

//声明队列

@Bean("queue\_media\_video\_processtask")

public Queue QUEUE\_PROCESSTASK() {

Queue queue = new Queue(queue\_media\_video\_processtask,true,false,true); return queue;

}

/\*\*

* 绑定队列到交换机 .

\* @param queue the queue

* @param exchange the exchange
* @return the binding

\*/

@Bean

public Binding binding\_queue\_media\_processtask(@Qualifier("queue\_media\_video\_processtask") Queue queue, @Qualifier(EX\_MEDIA\_PROCESSTASK) Exchange exchange) {

return BindingBuilder.bind(queue).to(exchange).with(routingkey\_media\_video).noargs();

}

}

在application.yml中配置队列名称及routingkey



xc‐service‐manage‐media:

mq:

queue‐media‐video‐processor: queue\_media\_video\_processor

routingkey‐media‐video: routingkey\_media\_video

**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

1.2.2 视频处理技术方案

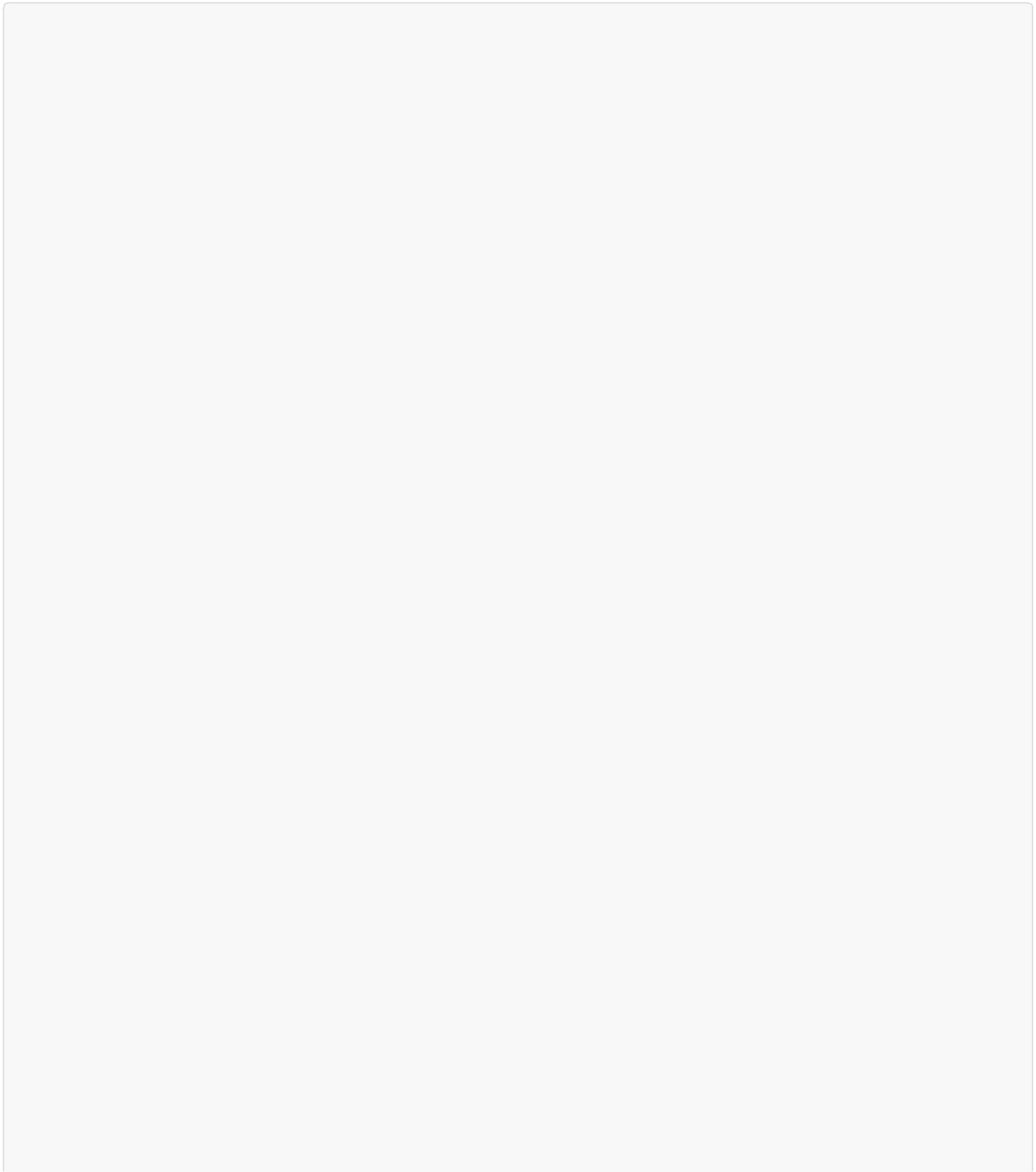
如何通过程序进行视频处理？

ﬀmpeg是一个可行的视频处理程序，可以通过Java调用ﬀmpeg.exe完成视频处理。

在java中可以使用Runtime类和Process Builder类两种方式来执行外部程序，工作中至少掌握一种。

本项目使用Process Builder的方式来调用ﬀmpeg完成视频处理。

关于Process Builder的测试如下：



@Test

public void testProcessBuilder(){

ProcessBuilder processBuilder = new ProcessBuilder();

* processBuilder.command("ping","127.0.0.1");

processBuilder.command("ipconfig");

//将标准输入流和错误输入流合并，通过标准输入流读取信息 processBuilder.redirectErrorStream(true);

try { //启动进程

Process start = processBuilder.start(); //获取输入流

InputStream inputStream = start.getInputStream(); //转成字符输入流

InputStreamReader inputStreamReader = new InputStreamReader(inputStream,"gbk");

int len = ‐1;

char[] c = new char[1024];

StringBuffer outputString = new StringBuffer();

//读取进程输入流中的内容

while ((len= inputStreamReader.read(c))!=‐1) {

String s = new String(c,0,len);

outputString.append(s);

System.out.print(s);

}

inputStream.close();

} catch (IOException e) { e.printStackTrace();

}

}

@Test

public void testFFmpeg(){

ProcessBuilder processBuilder = new ProcessBuilder(); //定义命令内容

List<String> command = new ArrayList<>();

command.add("D:\\Program Files\\ffmpeg‐20180227‐fa0c9d6‐win64‐static\\bin\\ffmpeg.exe");

command.add("‐i");

command.add("E:\\ffmpeg\_test\\1.avi");

command.add("‐y");//覆盖输出文件

command.add("‐c:v");

command.add("libx264");

**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

command.add("‐s");

command.add("1280x720");

command.add("‐pix\_fmt");

command.add("yuv420p");

command.add("‐b:a");

command.add("63k");

command.add("‐b:v");

command.add("753k");

command.add("‐r");

command.add("18");

command.add("E:\\ffmpeg\_test\\1.mp4");

processBuilder.command(command);

//将标准输入流和错误输入流合并，通过标准输入流读取信息

processBuilder.redirectErrorStream(true);

try {

//启动进程

Process start = processBuilder.start();

//获取输入流

InputStream inputStream = start.getInputStream();

//转成字符输入流

InputStreamReader inputStreamReader = new InputStreamReader(inputStream,"gbk"); int len = ‐1;

char[] c = new char[1024];

StringBuffer outputString = new StringBuffer();

//读取进程输入流中的内容

while ((len= inputStreamReader.read(c))!=‐1) {

String s = new String(c,0,len);

outputString.append(s);

System.out.print(s);

}

inputStream.close();

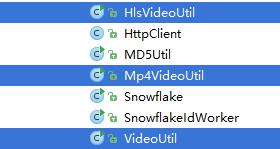
} catch (IOException e) {

e.printStackTrace();

}

}

上边的代码已经封装成工具类，参见：



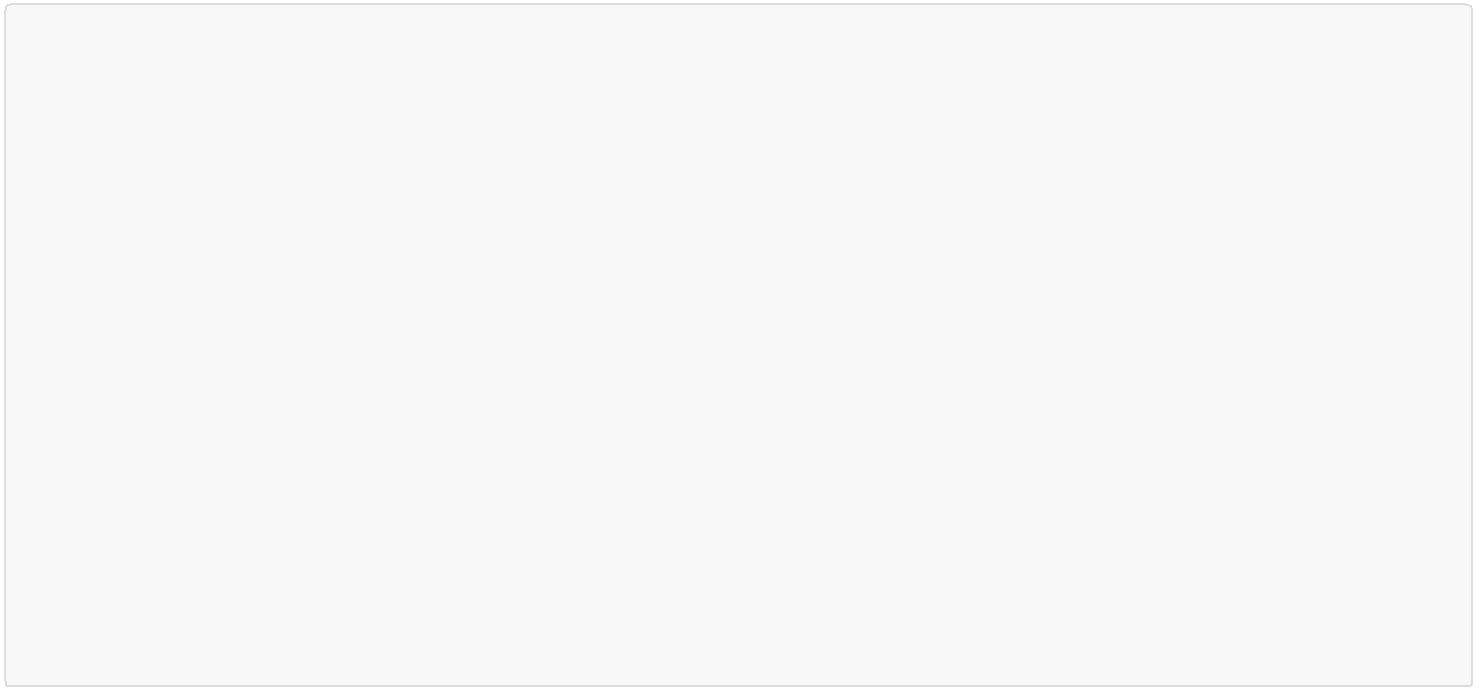
上边的工具类中：

Mp4VideoUtil.java完成avi转mp4

**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

HlsVideoUtil.java完成mp4转hls

分别测试每个工具类的使用方法。



public static void main(String[] args) throws IOException { //ffmpeg的路径

String ffmpeg\_path = "D:\\Program Files\\ffmpeg‐20180227‐fa0c9d6‐win64‐ static\\bin\\ffmpeg.exe";//ffmpeg的安装位置

//源avi视频的路径

String video\_path = "E:\\ffmpeg\_test\\1.avi";

//转换后mp4文件的名称

String mp4\_name = "1.mp4";

//转换后mp4文件的路径

String mp4\_path = "E:\\ffmpeg\_test\\";

//创建工具类对象

Mp4VideoUtil videoUtil = new Mp4VideoUtil(ffmpeg\_path,video\_path,mp4\_name,mp4\_path); //开始视频转换，成功将返回success

String s = videoUtil.generateMp4();

System.out.println(s);

}

1.2.3 视频处理实现

1.2.3.1确定消息格式

MQ消息统一采用json格式，视频处理生产方会向MQ发送如下消息，视频处理消费方接收此消息后进行视频处理：

｛“mediaId”:XXX｝

1.2.3.2处理流程

1）接收视频处理消息

2）判断媒体文件是否需要处理（本视频处理程序目前只接收avi视频的处理）

当前只有avi文件需要处理，其它文件需要更新处理状态为“无需处理”。

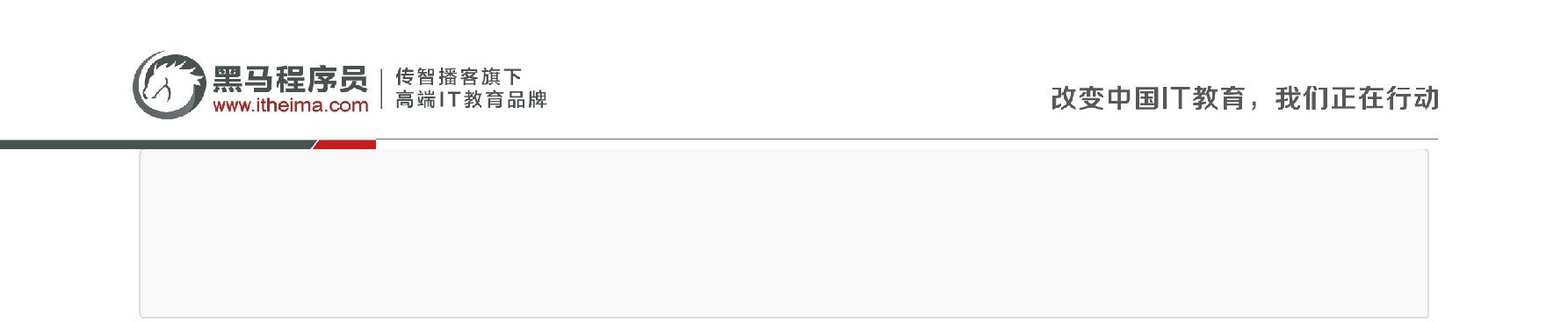
3）处理前初始化处理状态为“未处理”

4）处理失败需要在数据库记录处理日志，及处理状态为“处理失败”

5）处理成功记录处理状态为“处理成功”

1.2.3.3数据模型

在MediaFile类中添加mediaFileProcess\_m3u8属性记录ts文件列表，代码如下：

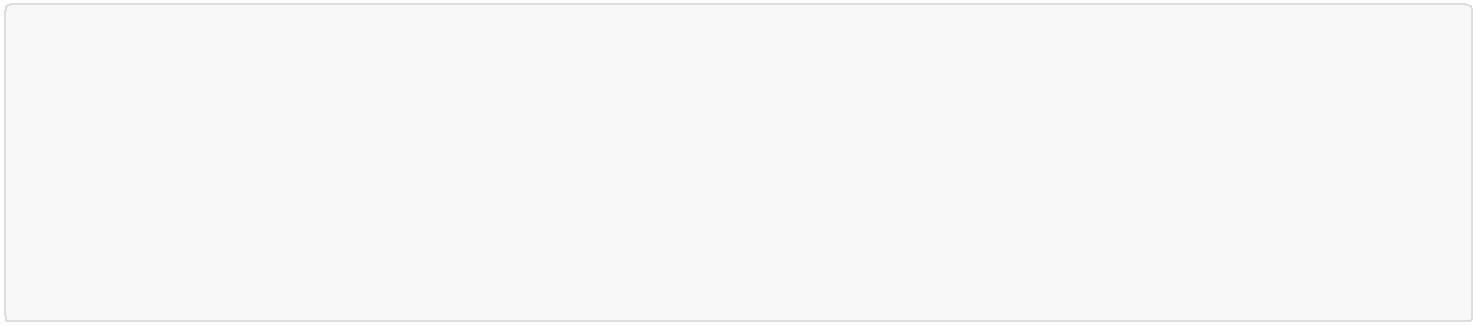
**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

//处理状态

private String processStatus;

//hls处理

private MediaFileProcess\_m3u8 mediaFileProcess\_m3u8;



@Data

@ToString

public class MediaFileProcess\_m3u8 extends MediaFileProcess { //ts列表

private List<String> tslist;

}

1.2.3.4 视频处理生成Mp4

1、创建Dao

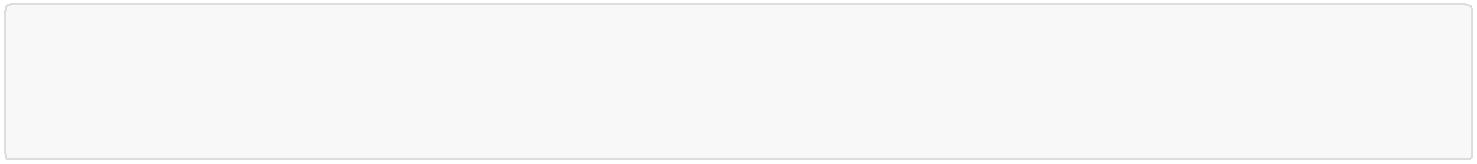
视频处理结果需要保存到媒资数据库，创建dao如下：



public interface MediaFileRepository extends MongoRepository<MediaFile,String> {

}

2、在application.yml中配置ﬀmpeg的位置及视频目录的根目录：



xc‐service‐manage‐media:

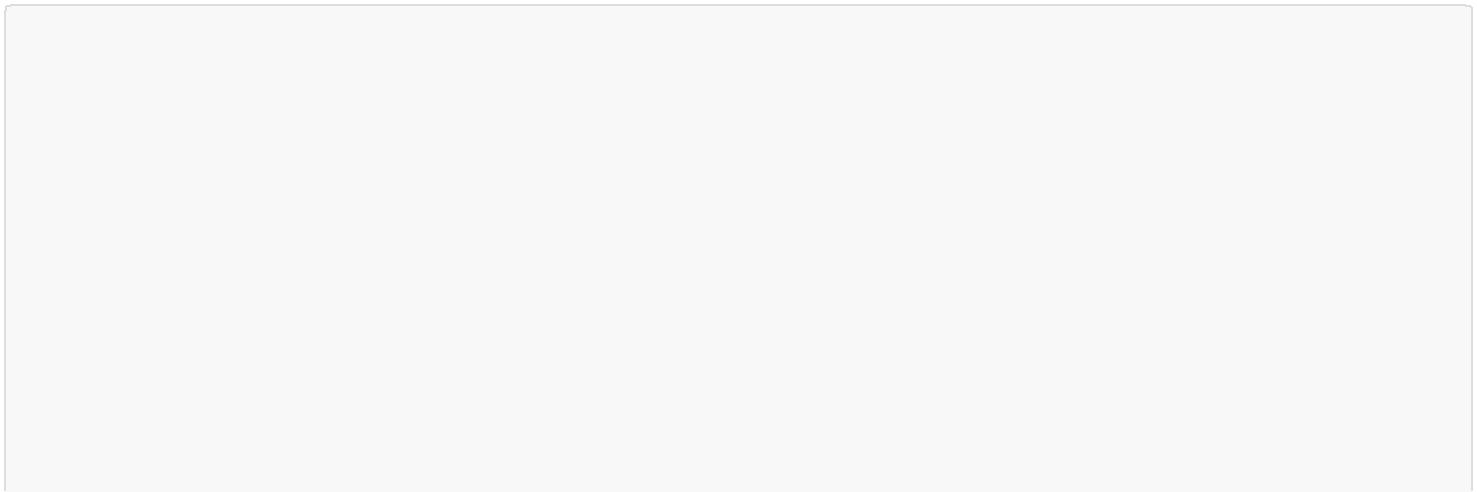
video‐location: F:/develop/video/

ffmpeg‐path: D:/Program Files/ffmpeg‐20180227‐fa0c9d6‐win64‐static/bin/ffmpeg.exe

3、处理任务类

在mq包下创建MediaProcessTask类，此类负责监听视频处理队列，并进行视频处理。

整个视频处理内容较多，这里分两部分实现：生成Mp4和生成m3u8，下边代码实现了生成mp4。



@Component

public class MediaProcessTask {

private static final Logger LOGGER = LoggerFactory.getLogger(MediaProcessTask.class);

//ffmpeg绝对路径

@Value("${xc‐service‐manage‐media.ffmpeg‐path}")

String ffmpeg\_path;

//上传文件根目录

@Value("${xc‐service‐manage‐media.upload‐location}")

String serverPath;

**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

@Autowired

MediaFileRepository mediaFileRepository;

@RabbitListener(queues = "${xc‐service‐manage‐media.mq.queue‐media‐processtask}")

public void receiveMediaProcessTask(String msg) throws IOException {

Map msgMap = JSON.parseObject(msg, Map.class); LOGGER.info("receive media process task msg :{} ",msgMap); //解析消息

//媒资文件id

String mediaId = (String) msgMap.get("mediaId");

//获取媒资文件信息

Optional<MediaFile> optional = mediaFileRepository.findById(fileMd5); if(！optional.isPresent()){

return ;

}

MediaFile mediaFile = optional.get();

//媒资文件类型

String fileType = mediaFile.getFileType();

if(fileType == null || !fileType.equals("avi")){//目前只处理avi文件mediaFile.setProcessStatus("303004");//处理状态为无需处理 mediaFileRepository.save(mediaFile);

return ;

}else{

mediaFile.setProcessStatus("303001");//处理状态为未处理 mediaFileRepository.save(mediaFile);

}

//生成mp4

String video\_path = serverPath + mediaFile.getFilePath()+mediaFile.getFileName(); String mp4\_name = mediaFile.getFileId()+".mp4";

String mp4folder\_path = serverPath + mediaFile.getFilePath(); Mp4VideoUtil videoUtil = new

Mp4VideoUtil(ffmpeg\_path,video\_path,mp4\_name,mp4folder\_path); String result = videoUtil.generateMp4();

if(result == null || !result.equals("success")){

//操作失败写入处理日志

mediaFile.setProcessStatus("303003");//处理状态为处理失败

MediaFileProcess\_m3u8 mediaFileProcess\_m3u8 = new MediaFileProcess\_m3u8();

mediaFileProcess\_m3u8.setErrormsg(result);

mediaFile.setMediaFileProcess\_m3u8(mediaFileProcess\_m3u8);

mediaFileRepository.save(mediaFile);

return ;

}

//生成m3u8...

}

}

说明：

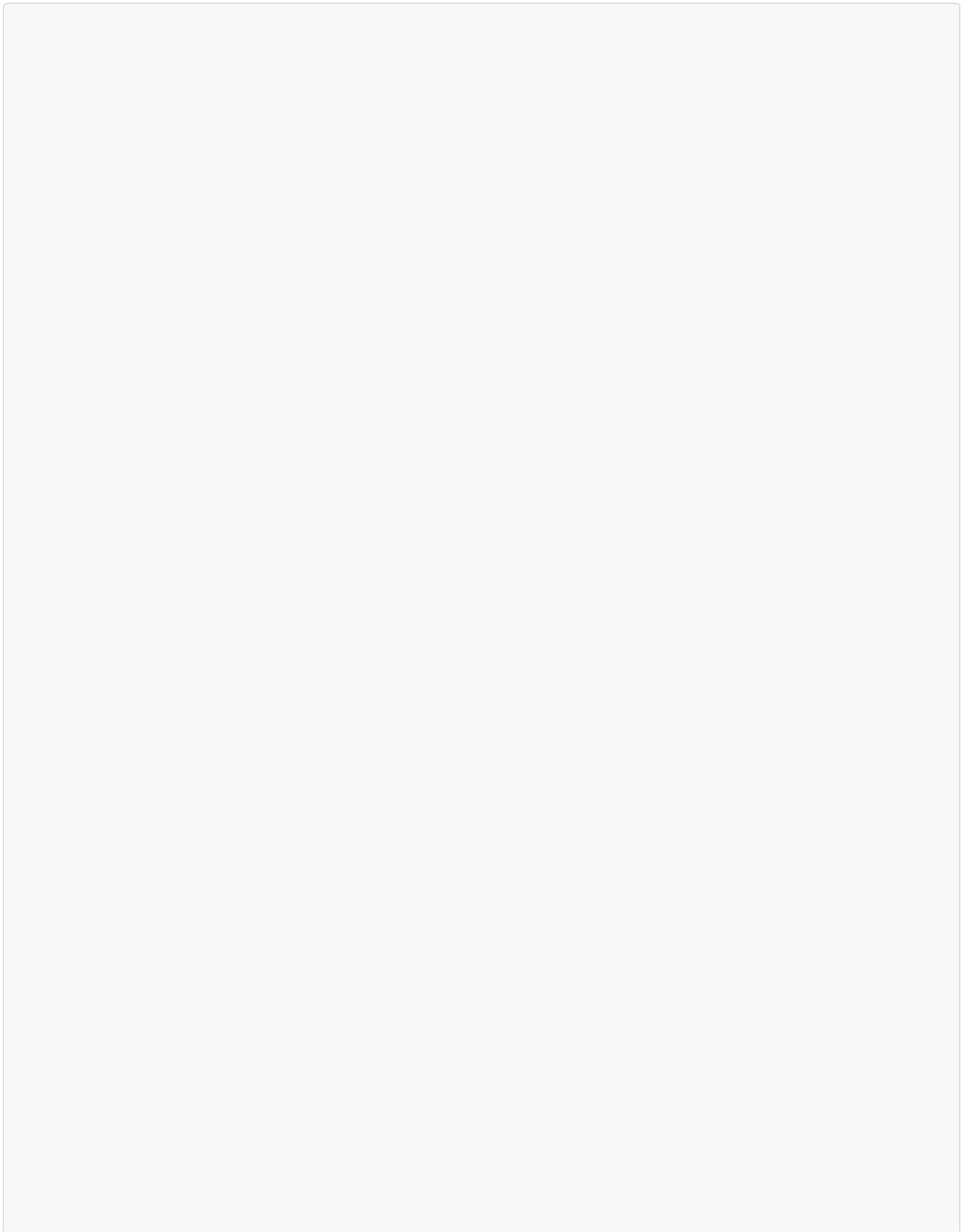
1、原始视频转成mp4如何判断转换成功？

**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

根据视频时长来判断，取原视频和转换成功视频的时长（时分秒），如果相等则相同。

1.2.3.5 视频处理生成m3u8

下边是完整的视频处理任务类代码，包括了生成m3u8及生成mp4的代码。



@Component

public class MediaProcessTask {

private static final Logger LOGGER = LoggerFactory.getLogger(MediaProcessTask.class);

//ffmpeg绝对路径

@Value("${xc‐service‐manage‐media.ffmpeg‐path}")

String ffmpeg\_path;

//上传文件根目录

@Value("${xc‐service‐manage‐media.upload‐location}")

String serverPath;

@Autowired

MediaFileRepository mediaFileRepository;

@RabbitListener(queues = "${xc‐service‐manage‐media.mq.queue‐media‐processtask}")

public void receiveMediaProcessTask(String msg) throws IOException {

Map msgMap = JSON.parseObject(msg, Map.class); LOGGER.info("receive media process task msg :{} ",msgMap); //解析消息

//媒资文件id

String mediaId = (String) msgMap.get("mediaId");

//获取媒资文件信息

Optional<MediaFile> optional = mediaFileRepository.findById(fileMd5); if(！optional.isPresent()){

return ;

}

MediaFile mediaFile = optional.get();

//媒资文件类型

String fileType = mediaFile.getFileType();

if(fileType == null || !fileType.equals("avi")){//目前只处理avi文件mediaFile.setProcessStatus("303004");//处理状态为无需处理 mediaFileRepository.save(mediaFile);

return ;

}else{

mediaFile.setProcessStatus("303001");//处理状态为未处理 mediaFileRepository.save(mediaFile);

}

//生成mp4

String video\_path = serverPath + mediaFile.getFilePath()+mediaFile.getFileName(); String mp4\_name = mediaFile.getFileId()+".mp4";

String mp4folder\_path = serverPath + mediaFile.getFilePath(); Mp4VideoUtil videoUtil = new

Mp4VideoUtil(ffmpeg\_path,video\_path,mp4\_name,mp4folder\_path); String result = videoUtil.generateMp4();

**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

if(result == null || !result.equals("success")){

//操作失败写入处理日志

mediaFile.setProcessStatus("303003");//处理状态为处理失败

MediaFileProcess\_m3u8 mediaFileProcess\_m3u8 = new MediaFileProcess\_m3u8();

mediaFileProcess\_m3u8.setErrormsg(result);

mediaFile.setMediaFileProcess\_m3u8(mediaFileProcess\_m3u8);

mediaFileRepository.save(mediaFile);

return ;

}

//生成m3u8

video\_path = serverPath + mediaFile.getFilePath()+mp4\_name;//此地址为mp4的地址 String m3u8\_name = mediaFile.getFileId()+".m3u8";

String m3u8folder\_path = serverPath + mediaFile.getFilePath()+"hls/"; HlsVideoUtil hlsVideoUtil = new

HlsVideoUtil(ffmpeg\_path,video\_path,m3u8\_name,m3u8folder\_path); result = hlsVideoUtil.generateM3u8();

if(result == null || !result.equals("success")){

//操作失败写入处理日志

mediaFile.setProcessStatus("303003");//处理状态为处理失败

MediaFileProcess\_m3u8 mediaFileProcess\_m3u8 = new MediaFileProcess\_m3u8();

mediaFileProcess\_m3u8.setErrormsg(result);

mediaFile.setMediaFileProcess\_m3u8(mediaFileProcess\_m3u8);

mediaFileRepository.save(mediaFile);

return ;

}

//获取m3u8列表

List<String> ts\_list = hlsVideoUtil.get\_ts\_list();

//更新处理状态为成功

mediaFile.setProcessStatus("303002");//处理状态为处理成功

MediaFileProcess\_m3u8 mediaFileProcess\_m3u8 = new MediaFileProcess\_m3u8();

mediaFileProcess\_m3u8.setTslist(ts\_list);

mediaFile.setMediaFileProcess\_m3u8(mediaFileProcess\_m3u8);

//m3u8文件url

mediaFile.setFileUrl(mediaFile.getFilePath()+"hls/"+m3u8\_name);

mediaFileRepository.save(mediaFile);

}

}

说明：

mp4转成m3u8如何判断转换成功？

第一、根据视频时长来判断，同mp4转换成功的判断方法。

第二、最后还要判断m3u8文件内容是否完整。

1.3 发送视频处理消息



**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

当视频上传成功后向MQ 发送视频 处理消息。

修改媒资管理服务的文件上传代码，当文件上传成功向MQ发送视频处理消息。

1.3.1 RabbitMQ配置

1、将media-processor工程下的RabbitmqConfig配置类拷贝到media工程下。

2、在media工程下配置mq队列等信息

修改application.yml



xc‐service‐manage‐media:

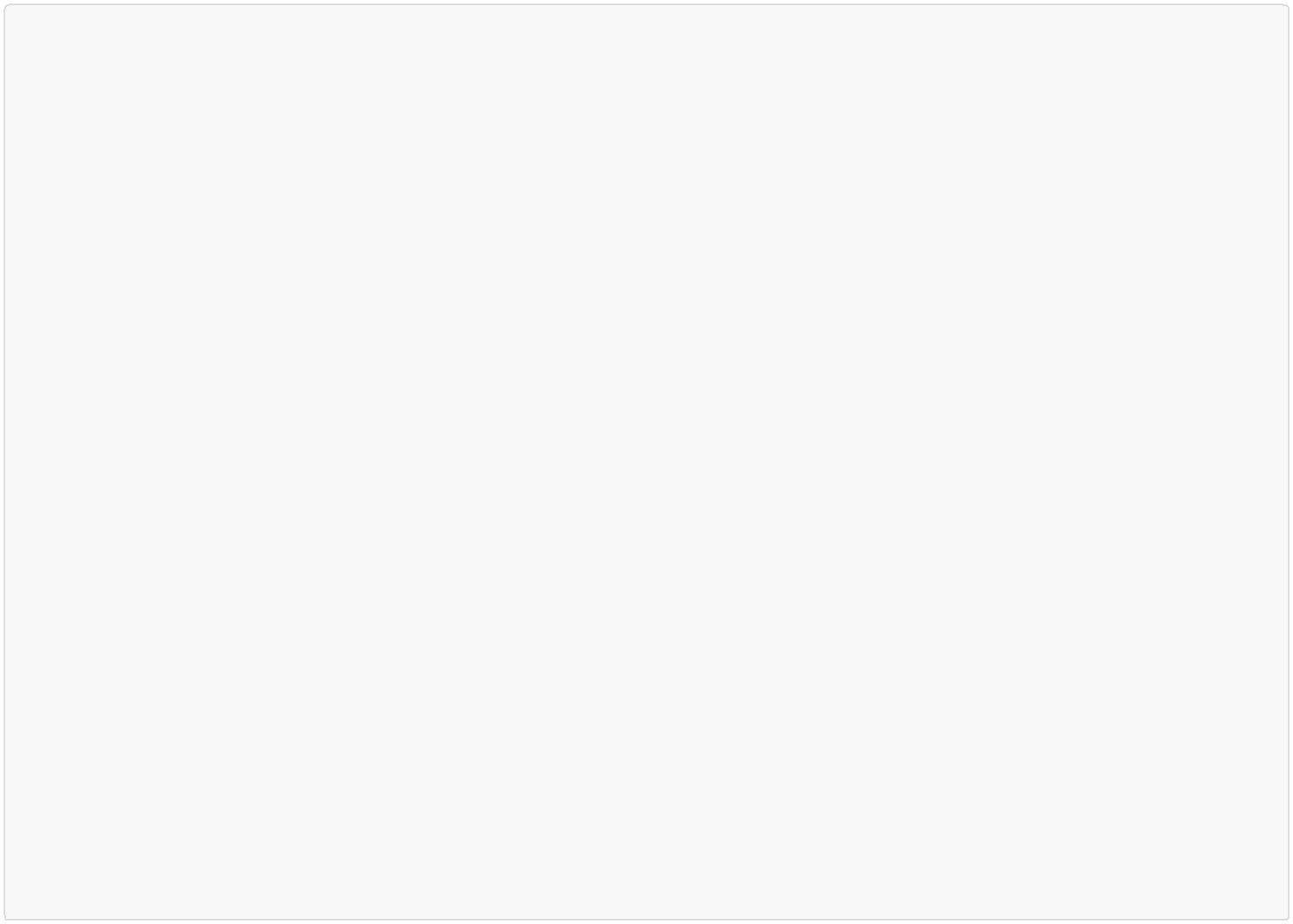
mq:

queue‐media‐video‐processor: queue\_media\_video\_processor

routingkey‐media‐video: routingkey\_media\_video

1.3.2 修改Service

在文件合并方法中添加向mq发送视频处理消息的代码：



//向MQ发送视频处理消息

public ResponseResult sendProcessVideoMsg(String mediaId){ Optional<MediaFile> optional = mediaFileRepository.findById(fileMd5);

if(！optional.isPresent()){

return new ResponseResult(CommonCode.FAIL);

}

MediaFile mediaFile = optional.get();

//发送视频处理消息

Map<String,String> msgMap = new HashMap<>();

msgMap.put("mediaId",mediaId);

//发送的消息

String msg = JSON.toJSONString(msgMap);

try {

this.rabbitTemplate.convertAndSend(RabbitMQConfig.EX\_MEDIA\_PROCESSTASK,routingkey\_media\_video, msg);

LOGGER.info("send media process task msg:{}",msg); }catch (Exception e){

e.printStackTrace();

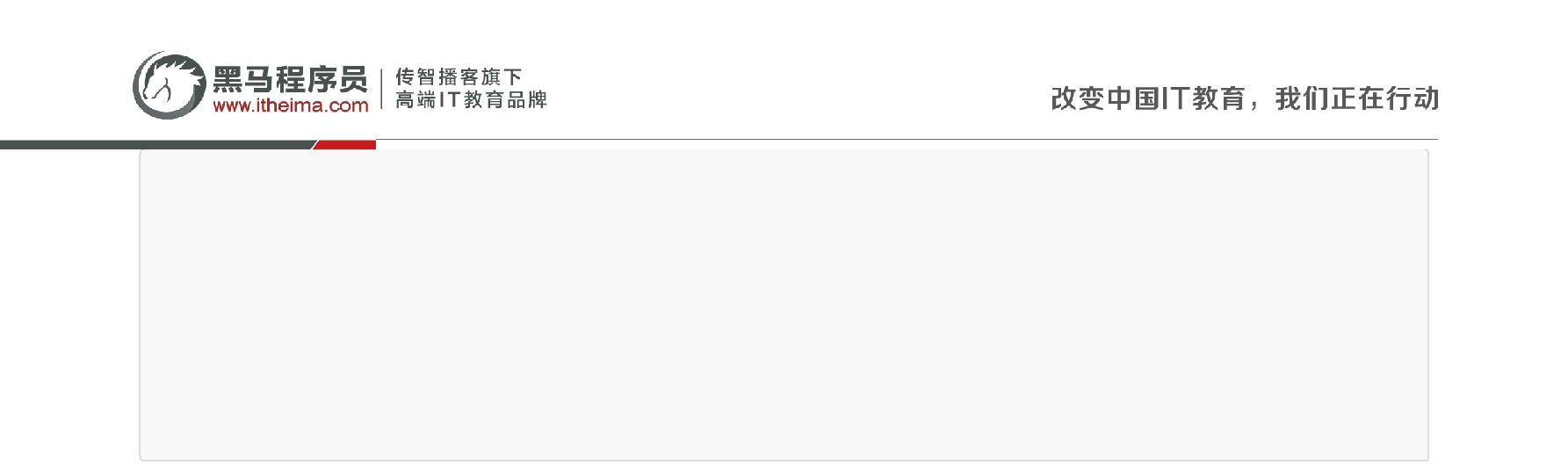
LOGGER.info("send media process task error,msg is:{},error:{}",msg,e.getMessage()); return new ResponseResult(CommonCode.FAIL);

}

return new ResponseResult(CommonCode.SUCCESS);

}

在mergechunks方法最后调用sendProcessVideo方法。

**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

......

//状态为上传成功

mediaFile.setFileStatus("301002");

mediaFileRepository.save(mediaFile);

String mediaId = mediaFile.getFileId();

//向MQ发送视频处理消息

sendProcessVideoMsg(mediaId);

......

1.4 视频处理测试



测试流程：

1、上传avi文件

2、观察日志是否发送消息

3、观察视频处理进程是否接收到消息进行处理

4、观察mp4文件是否生成

5、观察m3u8及 ts文件是否生成

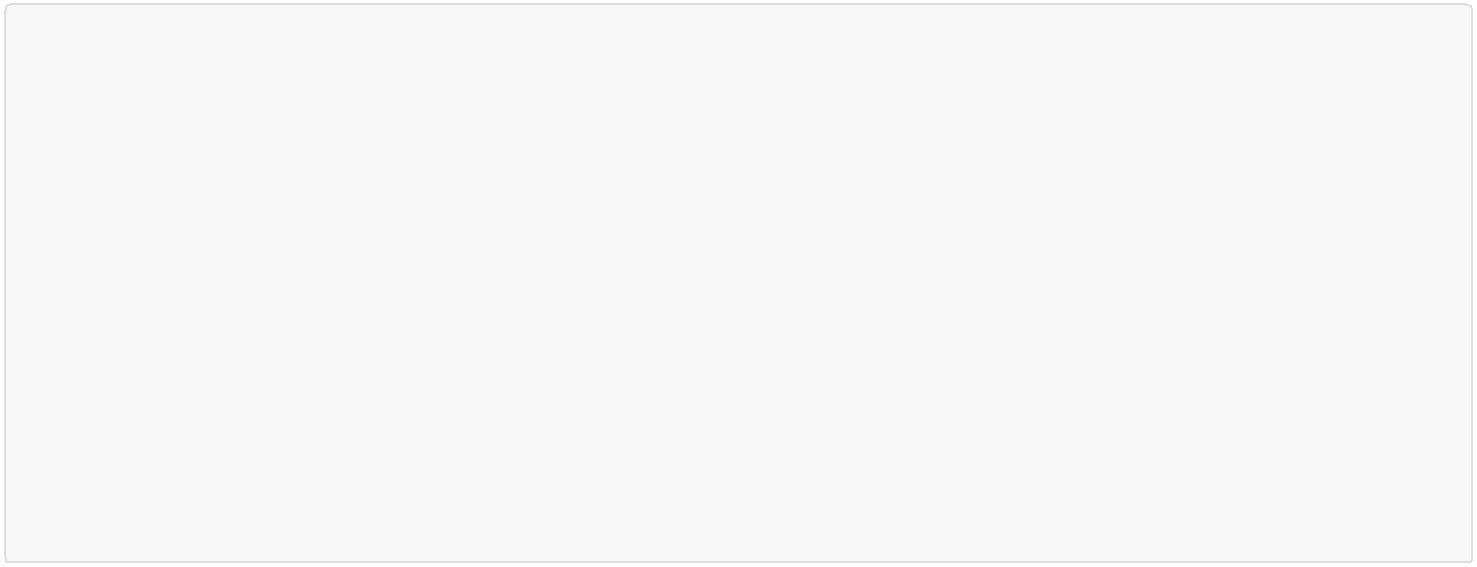
1.5 视频处理并发设置



代码中使用@RabbitListener注解指定消费方法，默认情况是单线程监听队列，可以观察当队列有多个任务时消费端每次只消费一个消息，单线程处理消息容易引起消息处理缓慢，消息堆积，不能最大利用硬件资源。

可以配置mq的容器工厂参数，增加并发处理数量即可实现多线程处理监听队列，实现多线程处理消息。

1、在RabbitmqConfig.java中添加容器工厂配置：



//消费者并发数量

public static final int DEFAULT\_CONCURRENT = 10;

@Bean("customContainerFactory")

public SimpleRabbitListenerContainerFactory

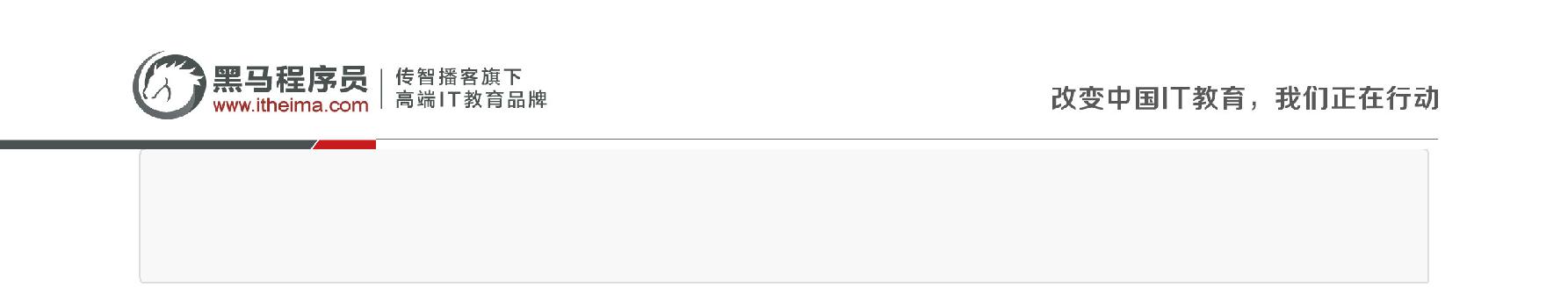
containerFactory(SimpleRabbitListenerContainerFactoryConfigurer configurer, ConnectionFactory

connectionFactory) {

SimpleRabbitListenerContainerFactory factory = new SimpleRabbitListenerContainerFactory(); factory.setConcurrentConsumers(DEFAULT\_CONCURRENT); factory.setMaxConcurrentConsumers(DEFAULT\_CONCURRENT); configurer.configure(factory, connectionFactory); return factory;

}

2、在@RabbitListener注解中指定容器工厂

**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

//视频处理方法

@RabbitListener(queues = {"${xc‐service‐manage‐media.mq.queue‐media‐video‐processor}"}, containerFactory="customContainerFactory")

再次测试当队列有多个任务时消费端的并发处理能力。

* 我的媒资



2.1 需求分析



通过我的媒资可以查询本教育机构拥有的媒资文件，进行文件处理、删除文件、修改文件信息等操作，具体需求如下：

1、分页查询我的媒资文件

2、删除媒资文件

3、处理媒资文件

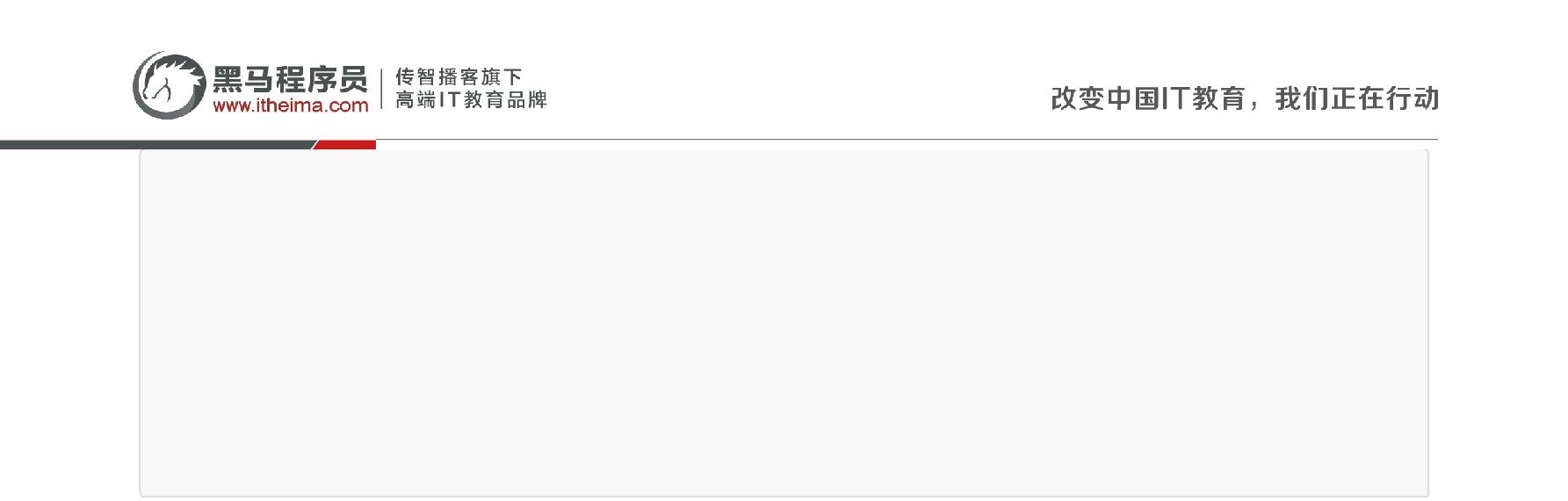
4、修改媒资文件信息



2.2 API



本节讲解我的媒资文件分页查询、处理媒资文件，其它功能请学员自行实现。

**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

@Api(value = "媒体文件管理",description = "媒体文件管理接口",tags = {"媒体文件管理接口"}) public interface MediaFileControllerApi {

@ApiOperation("查询文件列表")

public QueryResponseResult findList(int page, int size, QueryMediaFileRequest queryMediaFileRequest) ;

}

2.3 服务端开发



2.3.1 Dao



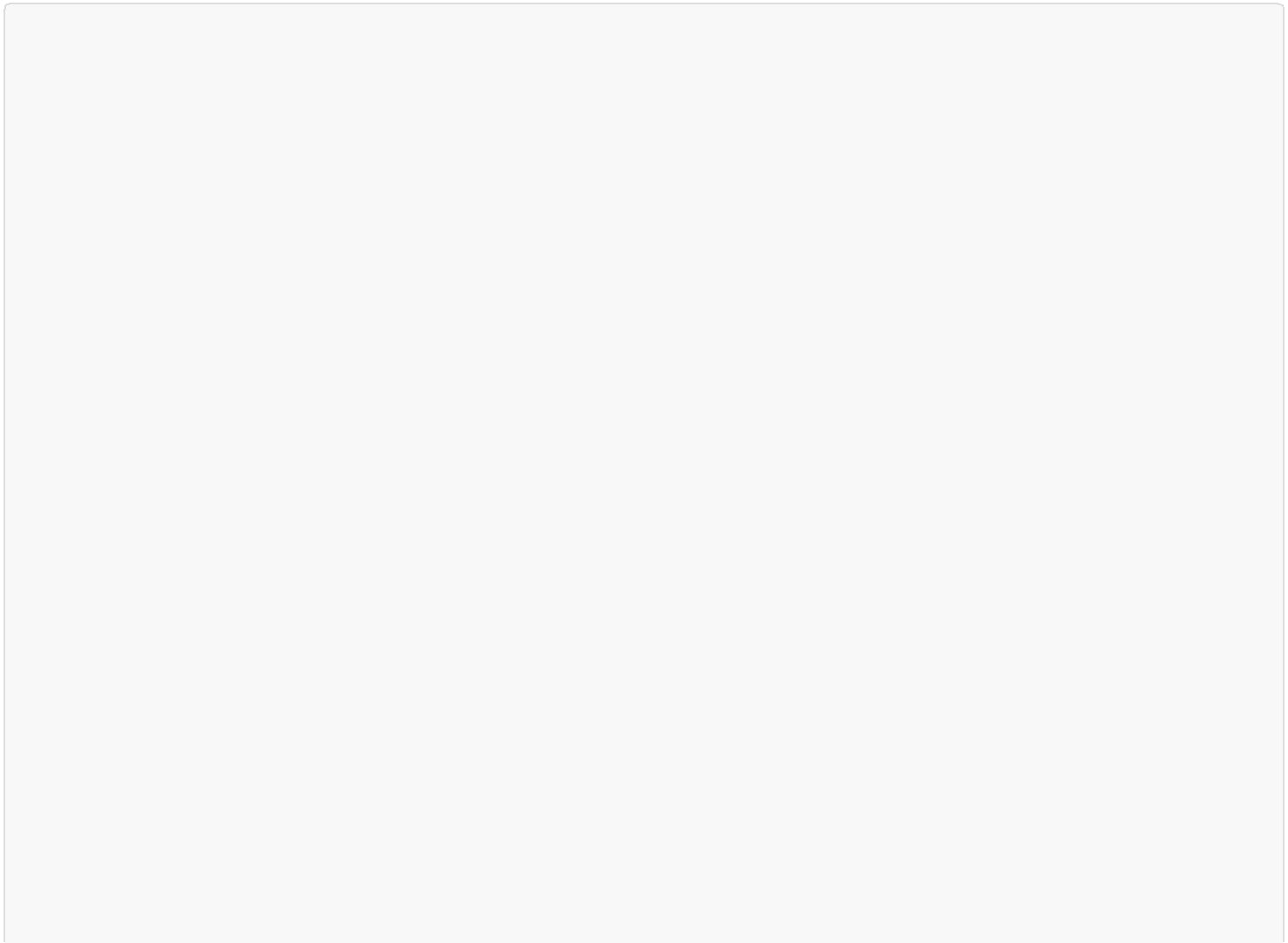
@Repository

public interface MediaFileDao extends MongoRepository<MediaFile,String> {

}

2.3.2 Service

定义findList方法实现媒资文件查询列表。



@Service

public class MediaFileService {

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

@Autowired

MediaFileRepository mediaFileRepository;

//文件列表分页查询

public QueryResponseResult findList(int page,int size,QueryMediaFileRequest queryMediaFileRequest){

//查询条件

MediaFile mediaFile = new MediaFile();

if(queryMediaFileRequest == null){

queryMediaFileRequest = new QueryMediaFileRequest();

}

//查询条件匹配器

ExampleMatcher matcher = ExampleMatcher.matching()

.withMatcher("tag", ExampleMatcher.GenericPropertyMatchers.contains())//tag字段

模糊匹配

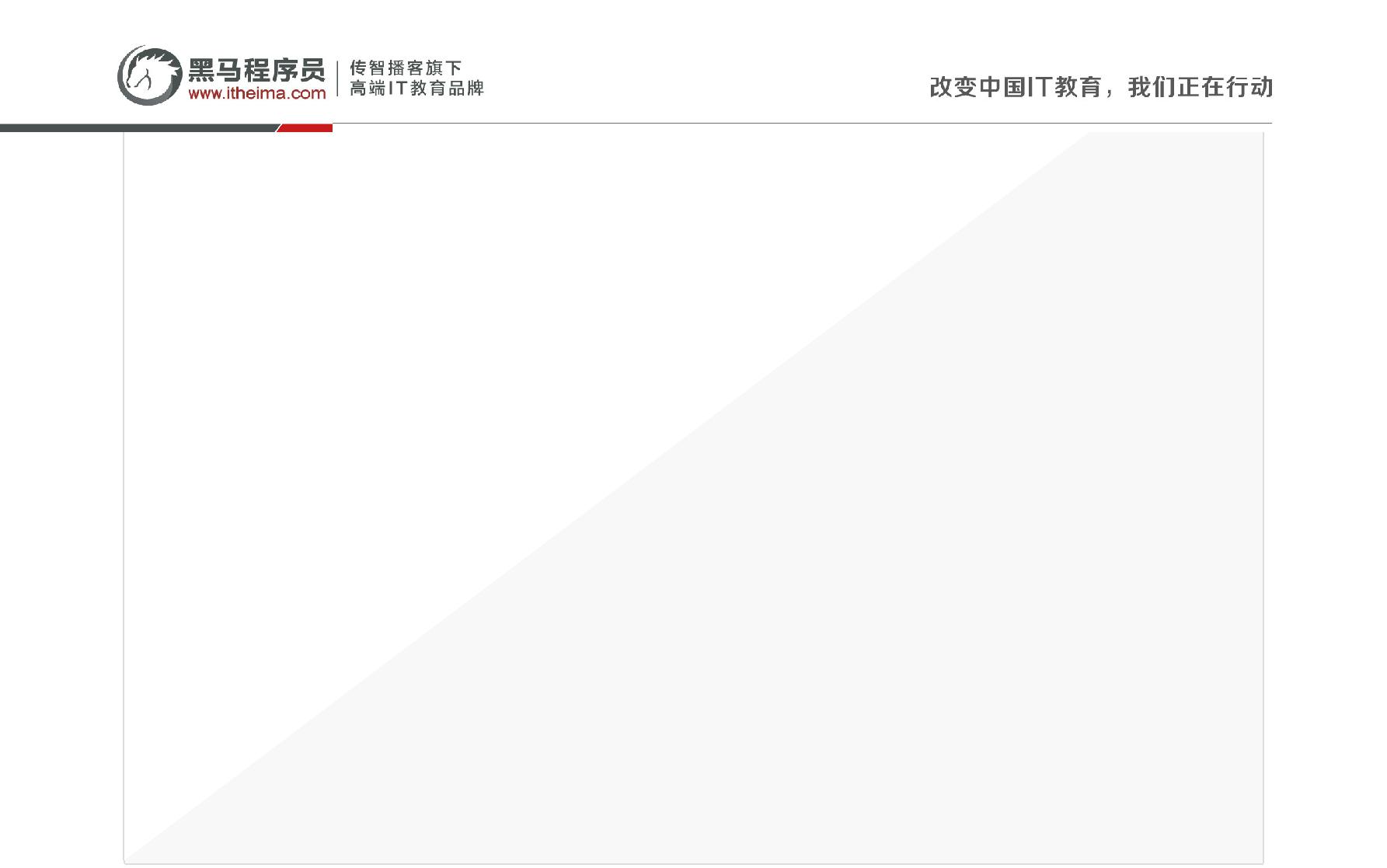
.withMatcher("fileOriginalName",

ExampleMatcher.GenericPropertyMatchers.contains())//文件原始名称模糊匹配

.withMatcher("processStatus", ExampleMatcher.GenericPropertyMatchers.exact());// 处理状态精确匹配（默认）

//查询条件对象

if(StringUtils.isNotEmpty(queryMediaFileRequest.getTag())){

**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

mediaFile.setTag(queryMediaFileRequest.getTag());

}

if(StringUtils.isNotEmpty(queryMediaFileRequest.getFileOriginalName())){

mediaFile.setFileOriginalName(queryMediaFileRequest.getFileOriginalName());

}

if(StringUtils.isNotEmpty(queryMediaFileRequest.getProcessStatus())){

mediaFile.setProcessStatus(queryMediaFileRequest.getProcessStatus());

}

//定义example实例

Example<MediaFile> ex = Example.of(mediaFile, matcher);

page = page‐1;

//分页参数

Pageable pageable = new PageRequest(page, size);

//分页查询

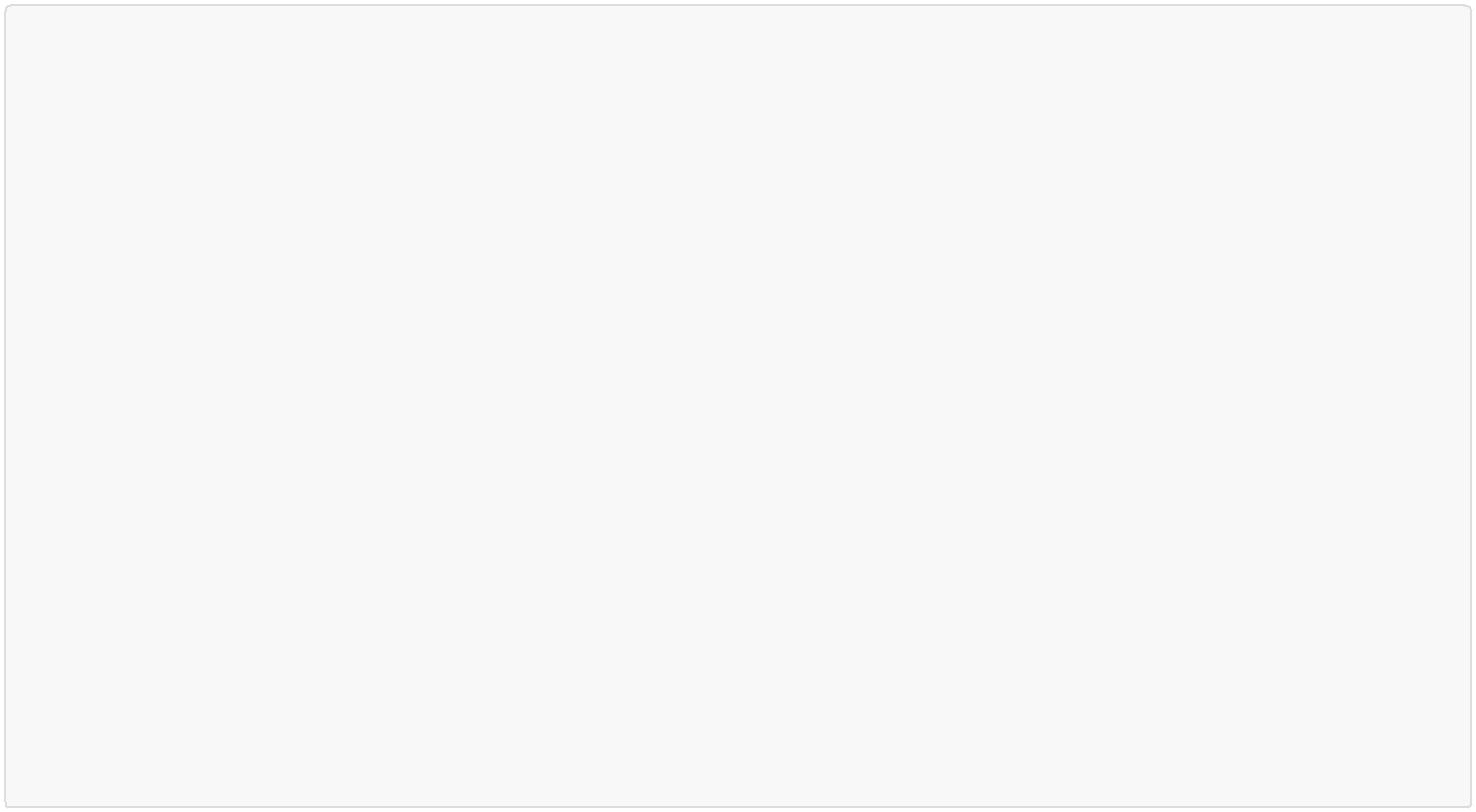
Page<MediaFile> all = mediaFileRepository.findAll(ex,pageable); QueryResult<MediaFile> mediaFileQueryResult = new QueryResult<MediaFile>(); mediaFileQueryResult.setList(all.getContent()); mediaFileQueryResult.setTotal(all.getTotalElements());

return new QueryResponseResult(CommonCode.SUCCESS,mediaFileQueryResult);

}

}

2.3.3 Controller



@RestController

@RequestMapping("/media/file")

public class MediaFileController implements MediaFileControllerApi {

@Autowired

MediaFileService mediaFileService;

@Autowired

MediaUploadService mediaUploadService;

@Override

@GetMapping("/list/{page}/{size}")

public QueryResponseResult findList(@PathVariable("page") int page, @PathVariable("size") int size, QueryMediaFileRequest queryMediaFileRequest) {

//媒资文件查询

return mediaFileService.findList(page,size,queryMediaFileRequest);

}

}

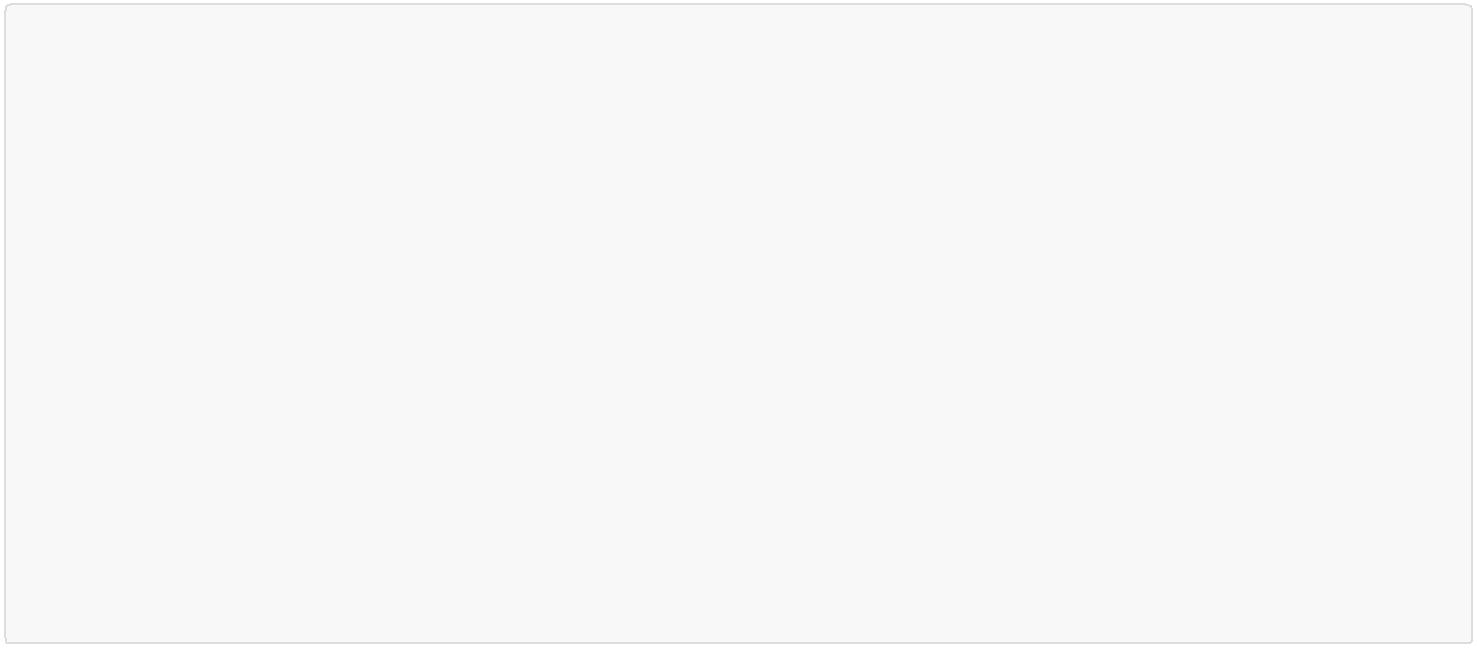
2.4 前端开发



**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

2.4.1 API方法

在media模块定义api方法如下：



import http from './../../../base/api/public'

import querystring from 'querystring'

let sysConfig = require('@/../config/sysConfig')

let apiUrl = sysConfig.xcApiUrlPre;

/\*页面列表\*/

export const media\_list = (page,size,params) => {

//params为json格式

//使用querystring将json对象转成key/value串

let querys = querystring.stringify(params)

return http.requestQuickGet(apiUrl+'/media/file/list/'+page+'/'+size+'/?'+querys)

}

/\*发送处理消息\*/

export const media\_process = (id) => {

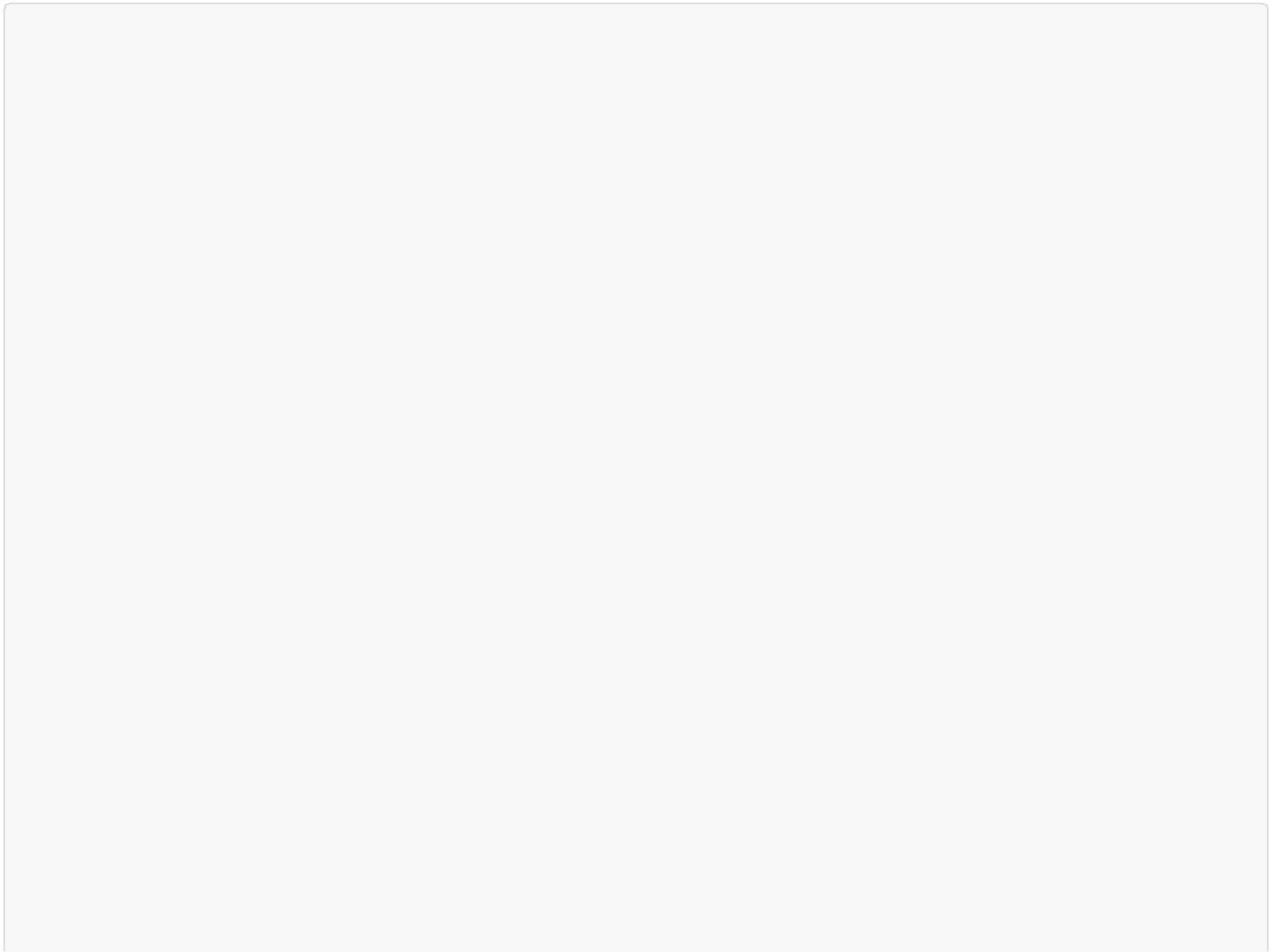
return http.requestPost(apiUrl+'/media/file/process/'+id)

}

2.4.2 页面

在media模块创建media\_list.vue，可参考cms系统的page\_list.vue来编写此页面。

1、视图



<template>

<div>

<!‐‐查询表单‐‐>

<el‐form :model="params">

标签：

<el‐input v‐model="params.tag" style="width:160px"></el‐input> 原始名称：

<el‐input v‐model="params.fileOriginalName" style="width:160px"></el‐input> 处理状态：

<el‐select v‐model="params.processStatus" placeholder="请选择处理状态"> <el‐option

v‐for="item in processStatusList"

:key="item.id"

:label="item.name"

:value="item.id">

</el‐option>

</el‐select>

<br/>

<el‐button type="primary" v‐on:click="query" size="small">查询</el‐button> <router‐link class="mui‐tab‐item" :to="{path:'/media/upload'}">

<el‐button type="primary" size="small" >上传文件</el‐button> </router‐link>

</el‐form>

<!‐‐列表‐‐>

<el‐table :data="list" highlight‐current‐row v‐loading="listLoading" style="width: 100%;">

<el‐table‐column type="index" width="30">

**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

</el‐table‐column>

<el‐table‐column prop="fileOriginalName" label="原始文件名称" width="220"> </el‐table‐column>

<el‐table‐column prop="fileName" label="文件名称" width="220"> </el‐table‐column>

<el‐table‐column prop="fileUrl" label="访问url" width="260"> </el‐table‐column>

<el‐table‐column prop="tag" label="标签" width="100"> </el‐table‐column>

<el‐table‐column prop="fileSize" label="文件大小" width="120"> </el‐table‐column>

<el‐table‐column prop="processStatus" label="处理状态" width="100"

:formatter="formatProcessStatus">

</el‐table‐column>

<el‐table‐column prop="uploadTime" label="创建时间" width="110"

:formatter="formatCreatetime">

</el‐table‐column>

<el‐table‐column label="开始处理" width="100" >

<template slot‐scope="scope">

<el‐button

size="small" type="primary" plain @click="process(scope.row.fileId)">开始处理

</el‐button>

</template>

</el‐table‐column>

</el‐table>

<!‐‐分页‐‐>

<el‐col :span="24" class="toolbar">

<el‐pagination background layout="prev, pager, next" @current‐change="changePage" :page‐ size="this.params.size"

:total="total" :current‐page="this.params.page"

style="float:right;">

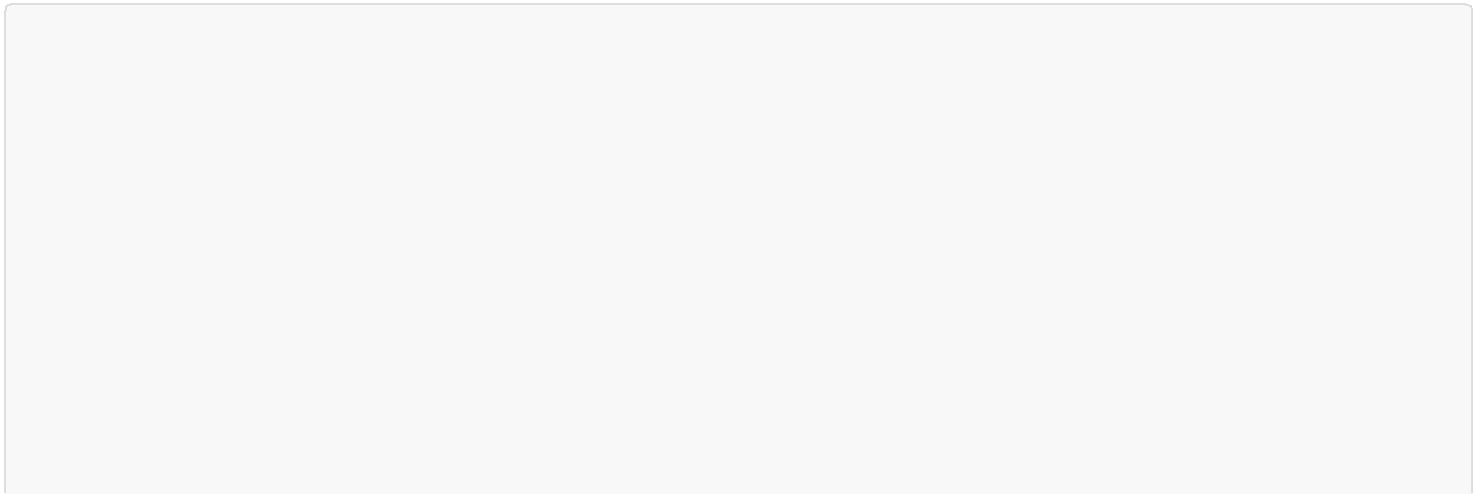
</el‐pagination>

</el‐col>

</div>

</template>

2、数据对象



import \* as mediaApi from '../api/media'

import utilApi from '@/common/utils';

export default{

data(){

return {

params:{

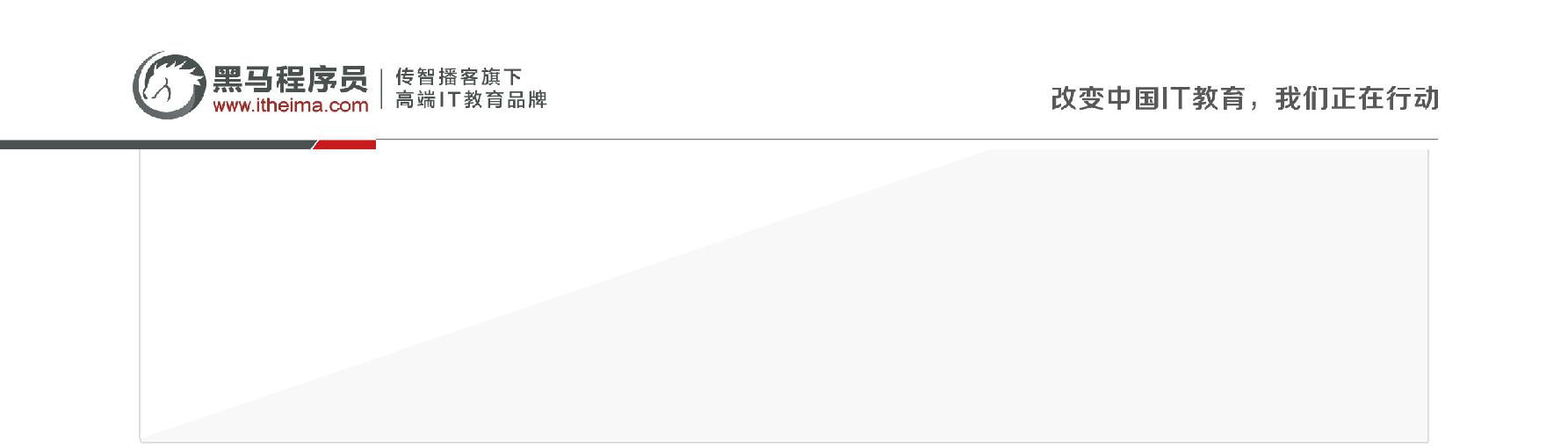
page:1,//页码

size:2,//每页显示个数

tag:'',//标签

fileName:'',//文件名称

processStatus:''//处理状态

**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

},

listLoading:false,

list:[],

total:0,

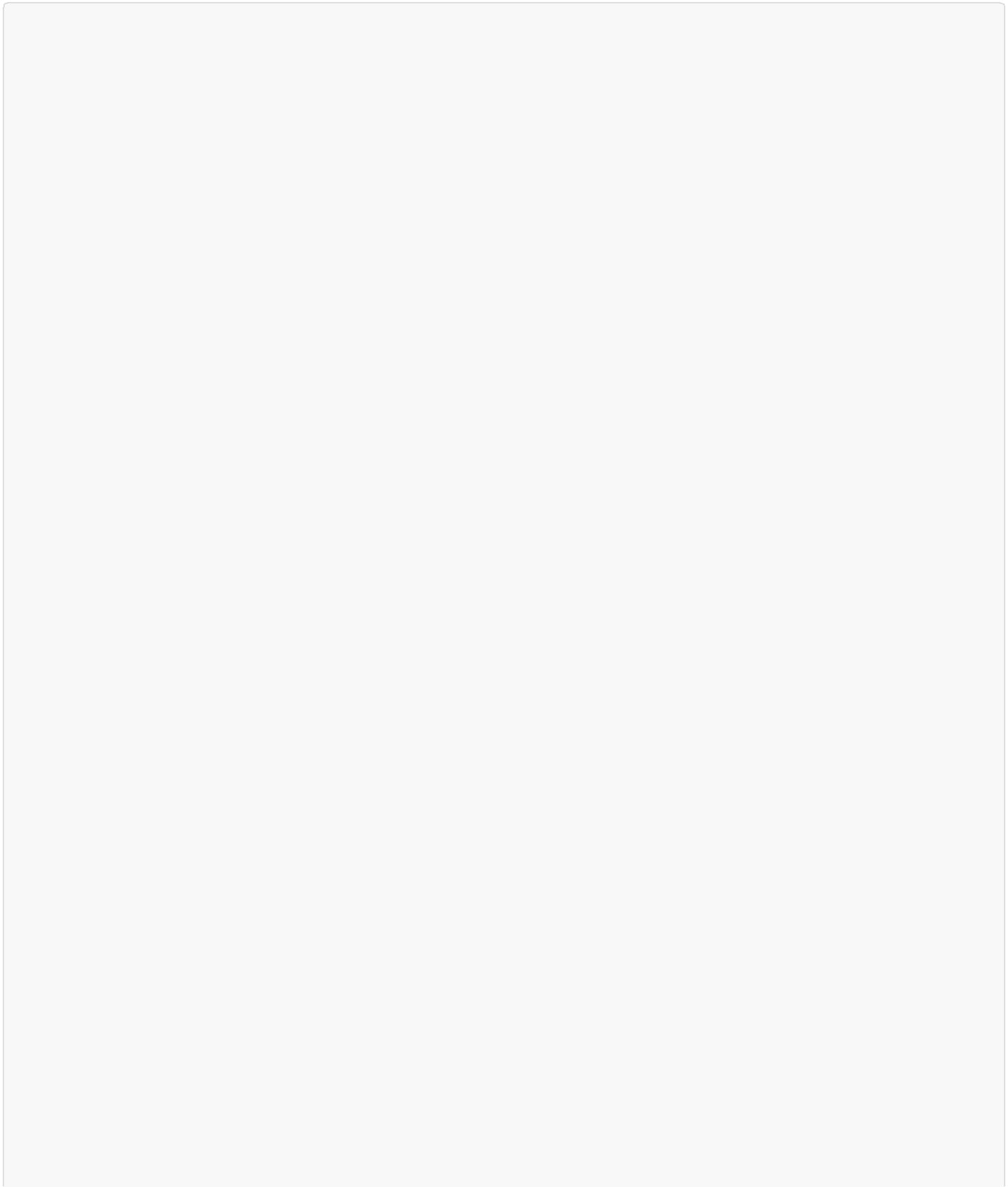
processStatusList:[]

}

}

。。。

3、方法



methods:{

formatCreatetime(row, column){

var createTime = new Date(row.uploadTime);

if (createTime) {

return utilApi.formatDate(createTime, 'yyyy‐MM‐dd hh:mm:ss');

}

},

formatProcessStatus(row,column){

var processStatus = row.processStatus;

if (processStatus) {

if(processStatus == '303001'){

return "处理中";

}else if(processStatus == '303002'){

return "处理成功";

}else if(processStatus == '303003'){

return "处理失败";

}else if(processStatus == '303004'){

return "无需处理";

}

}

},

changePage(page){

this.params.page = page;

this.query()

},

process (id) {

* console.log(id)

mediaApi.media\_process(id).then((res)=>{

console.log(res)

if(res.success){

this.$message.success('开始处理，请稍后查看处理结果'); }else{

this.$message.error('操作失败，请刷新页面重试');

}

})

},

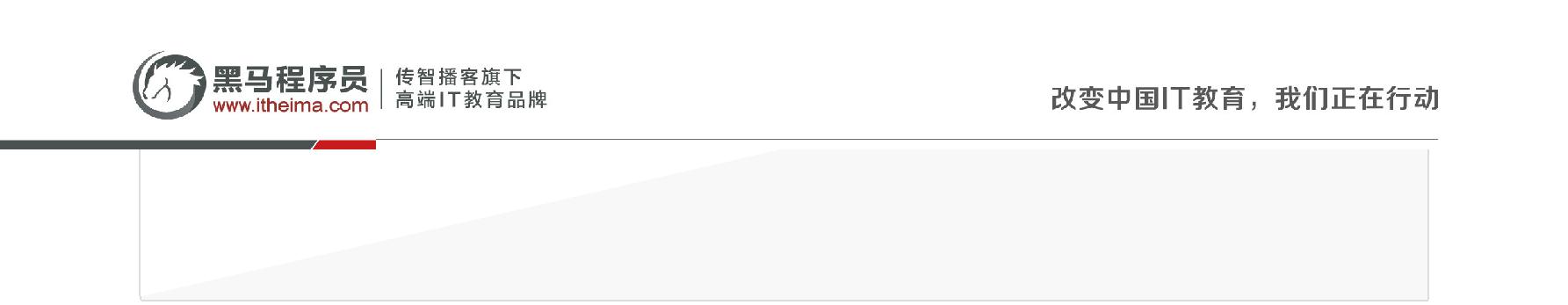
query(){

mediaApi.media\_list(this.params.page,this.params.size,this.params).then((res)=>{

console.log(res)

this.total = res.queryResult.total

this.list = res.queryResult.list

**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

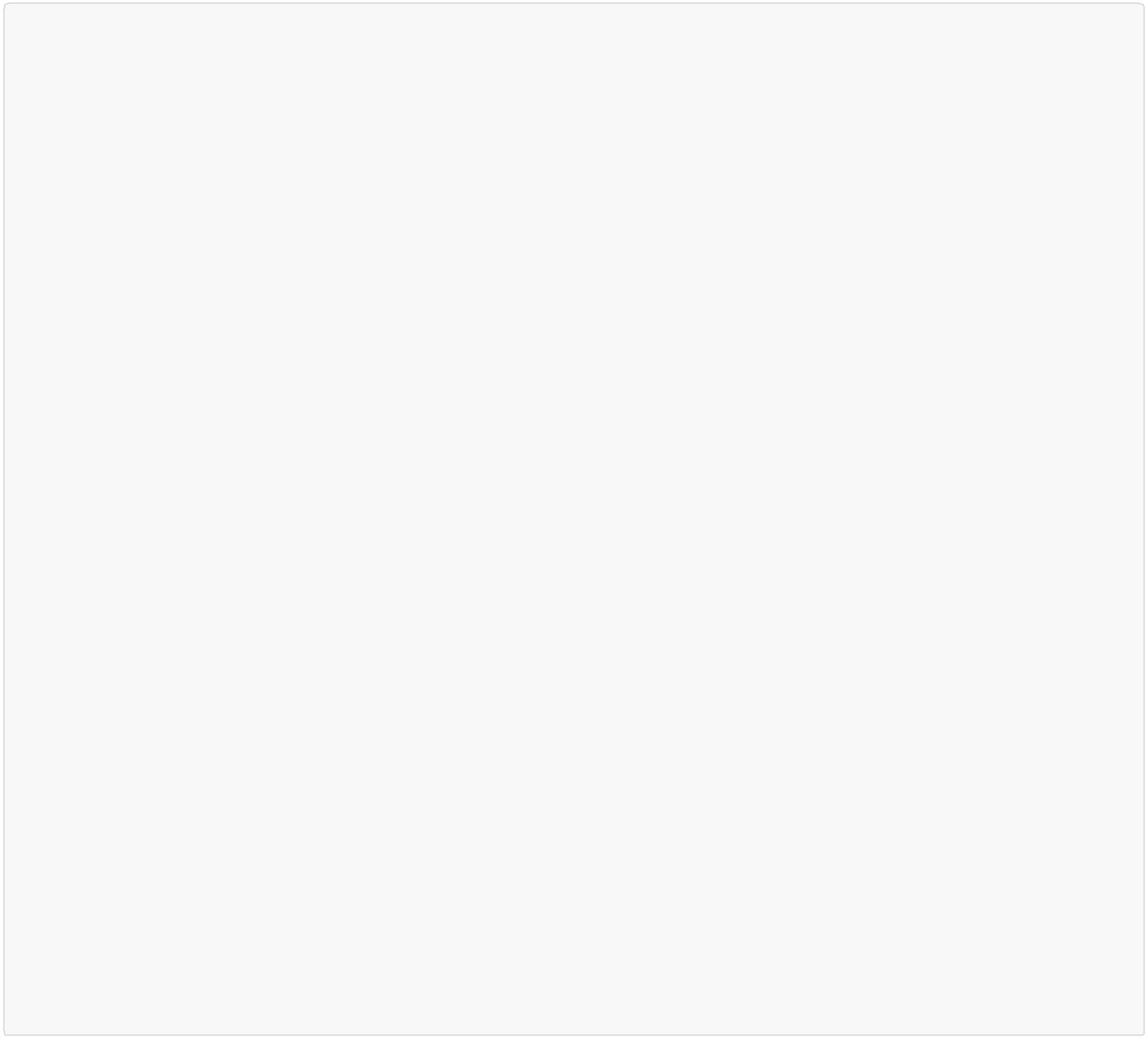
})

}

}

...

4、钩子方法



created(){

//默认第一页

this.params.page = Number.parseInt(this.$route.query.page||1);

},

mounted() {

//默认查询页面

this.query()

//初始化处理状态

this.processStatusList = [

{

id:'',

name:'全部'

},

{

id:'303001',

name:'处理中'

},

{

id:'303002',

name:'处理成功'

},

{

id:'303003',

name:'处理失败'

},

{

id:'303004',

name:'无需处理'

}

]

}

}

* 媒资与课程计划关联



3.1需求分析



到目前为止，媒资管理已完成文件上传、视频处理、我的媒资功能等基本功能。其它模块已可以使用媒资管理功能，本节要讲解课程计划在编辑时如何选择媒资文件。

**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

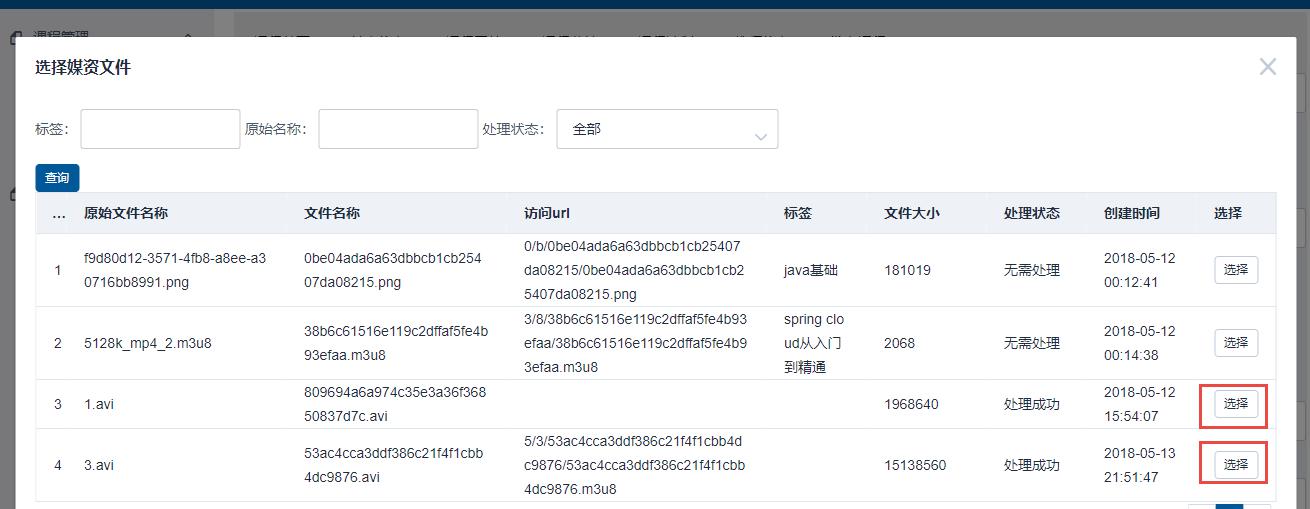
操作的业务流程如下：

1、进入课程计划修改页面

2、选择视频

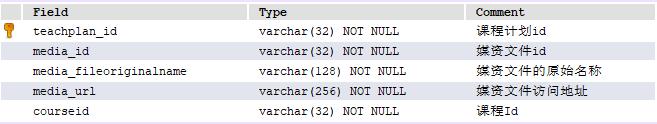
打开媒资文件查询窗口，找到该课程章节的视频，选择此视频。

点击“选择媒资文件”打开媒资文件列表



3、 选择成功后，将在课程管理数据库保存课程计划对应在的课程视频地址。

在课程管理数据库创建表 teachplan\_media 存储课程计划与媒资关联信息，如下：



3.2 选择视频



3.2.1 Vue父子组件通信

上一章已实现了我的媒资页面，所以媒资查询窗口页面不需要再开发，将“我的媒资页面”作为一个组件在修改课程计划页面中引用，如下图：

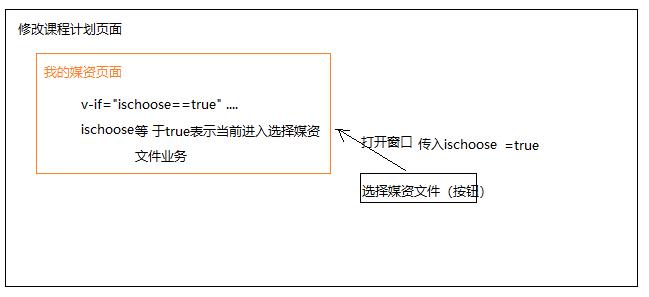
**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

修改课程计划页面为父组件，我的媒资查询页面为子组件。

问题1：

我的媒资页面在选择媒资文件时不允许显示，比如“视频处理”按钮，该如何控制？

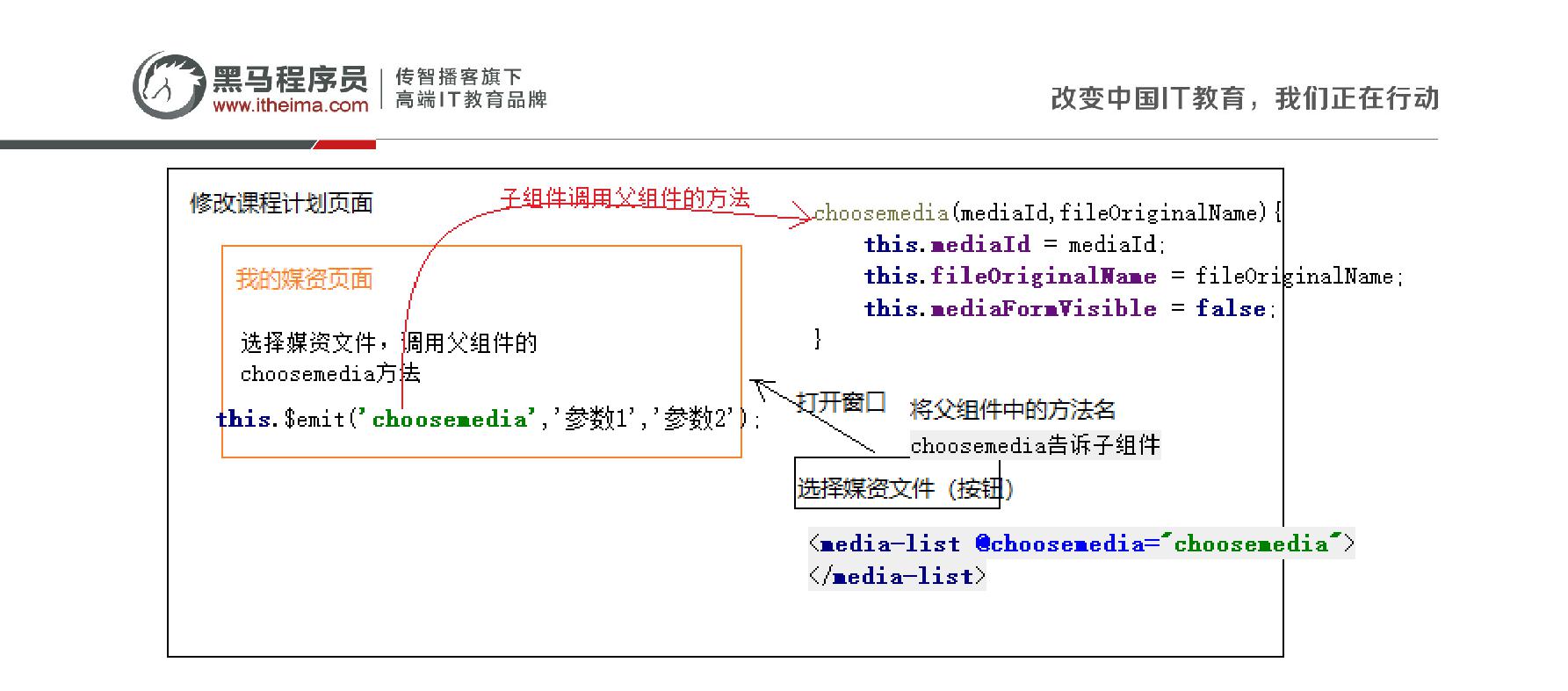
这时就需要父组件（修改课程计划页面）向子组件（我的媒资页面）传入一个变量，使用此变量来控制当前是否进入选择媒资文件业务，从而控制哪些元素不显示，如下图：



问题2：

在我的媒资页面选择了媒资文件，如何将选择的媒资文件信息传到父组件？

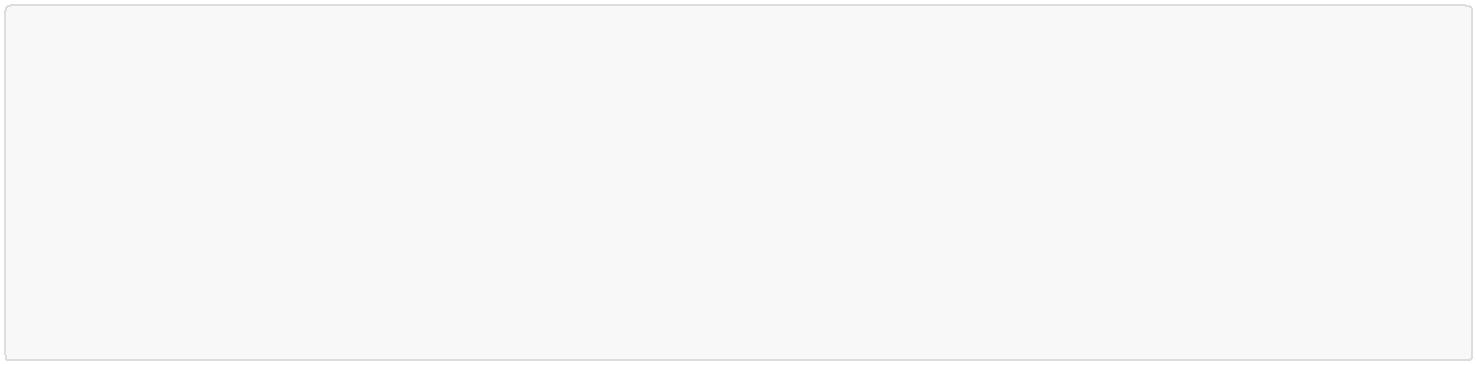
这时就需要子组件调用父组件的方法来解决此问题，如下图：

**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

3.2.2 父组件（修改课程计划）

本节实现功能：在课程计划页面打开我的媒资页面。

1、引入子组件



import mediaList from '@/module/media/page/media\_list.vue';

export default {

components:{

mediaList

},

data() {

....

2、使用子组件

在父组件的视图中使用子组件，同时传入变量ischoose，并指定父组件的方法名为choosemedia

这里使用el-dialog 实现弹出窗口。



<el‐dialog title="选择媒资文件" :visible.sync="mediaFormVisible">

<media‐list v‐bind:ischoose="true" @choosemedia="choosemedia"></media‐list>

</el‐dialog>

3、choosemedia方法

在父组件中定义choosemedia方法，接收子组件调用，参数包括：媒资文件id、媒资文件的原始名称、媒资文件 url



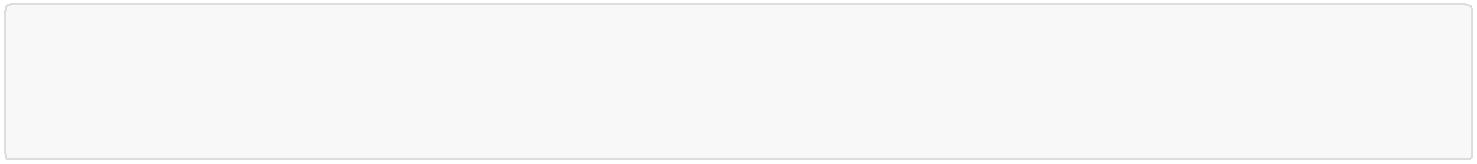
choosemedia(mediaId,fileOriginalName,mediaUrl){

}

**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

4、打开子组件窗口

1）打开子组件窗口按钮定义



添加“选择视频”按钮：

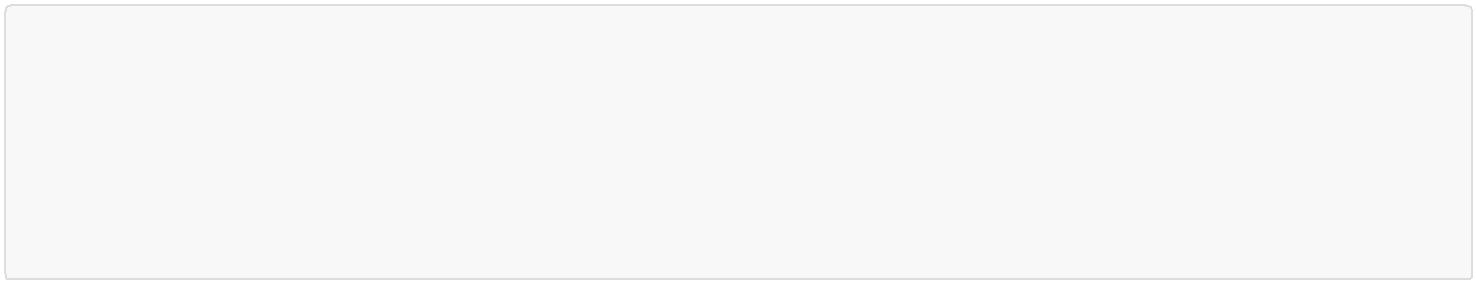
<el‐button style="font‐size: 12px;" type="text" on‐click={ () => this.querymedia(data.id) }>选择视频</el‐button>

效果如下：



2、打开子组件窗口方法

定义querymedia方法：



methods: {

//打开查询媒资文件窗口，传入课程计划id

querymedia(teachplanId){

this.activeTeachplanId = teachplanId;

this.mediaFormVisible = true;

},

3.2.3 子组件（我的媒资查询）

1、定义ischoose变量，接收父组件传入的ischoose



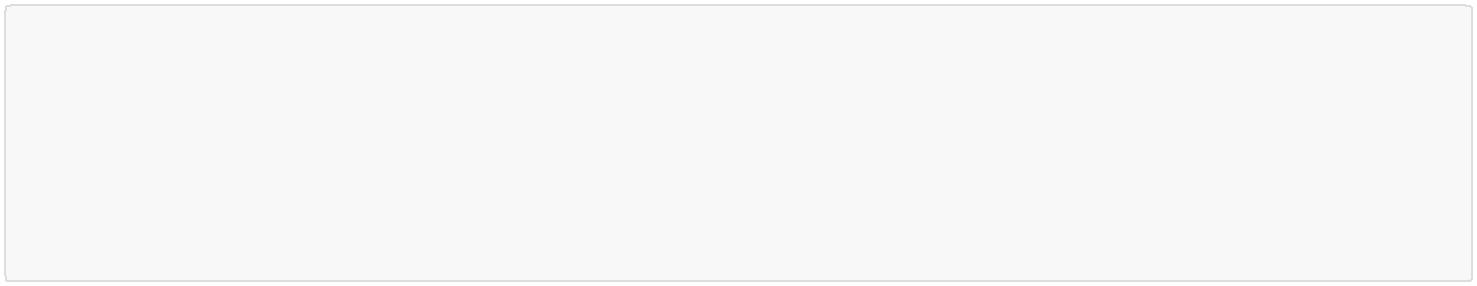
export default{

props: ['ischoose'],

data(){

2、父组件传的ischoose变量为 true时表示当前是选择媒资文件业务，需要控制页面元素是否显示

1）ischoose=true，选择按钮显示



<el‐table‐column label="选择" width="80" v‐if="ischoose == true"> <template slot‐scope="scope">

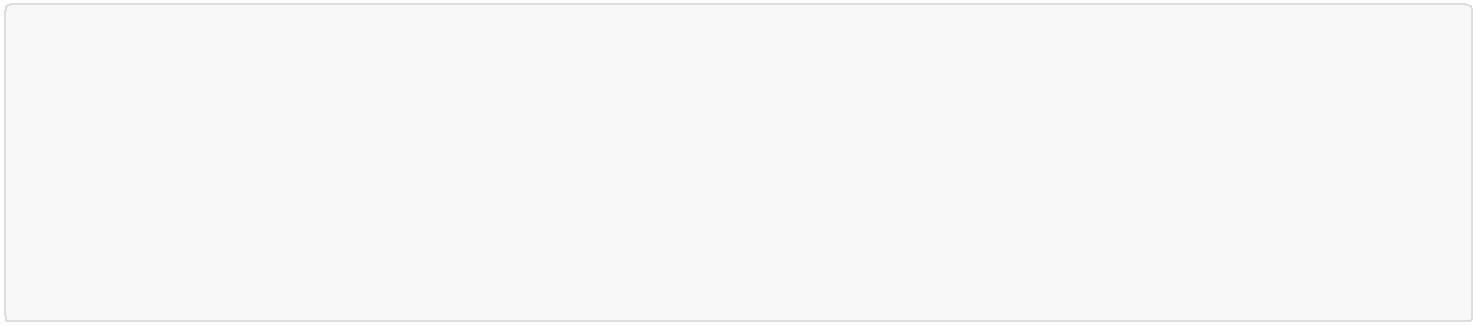
<el‐button

size="small" type="primary" plain @click="choose(scope.row)">选择</el‐button> </template>

</el‐table‐column>

**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

2）ischoose=false，视频处理按钮显示



<el‐table‐column label="开始处理" width="100" v‐if="ischoose != true"> <template slot‐scope="scope">

<el‐button

size="small" type="primary" plain @click="process(scope.row.fileId)">开始处理

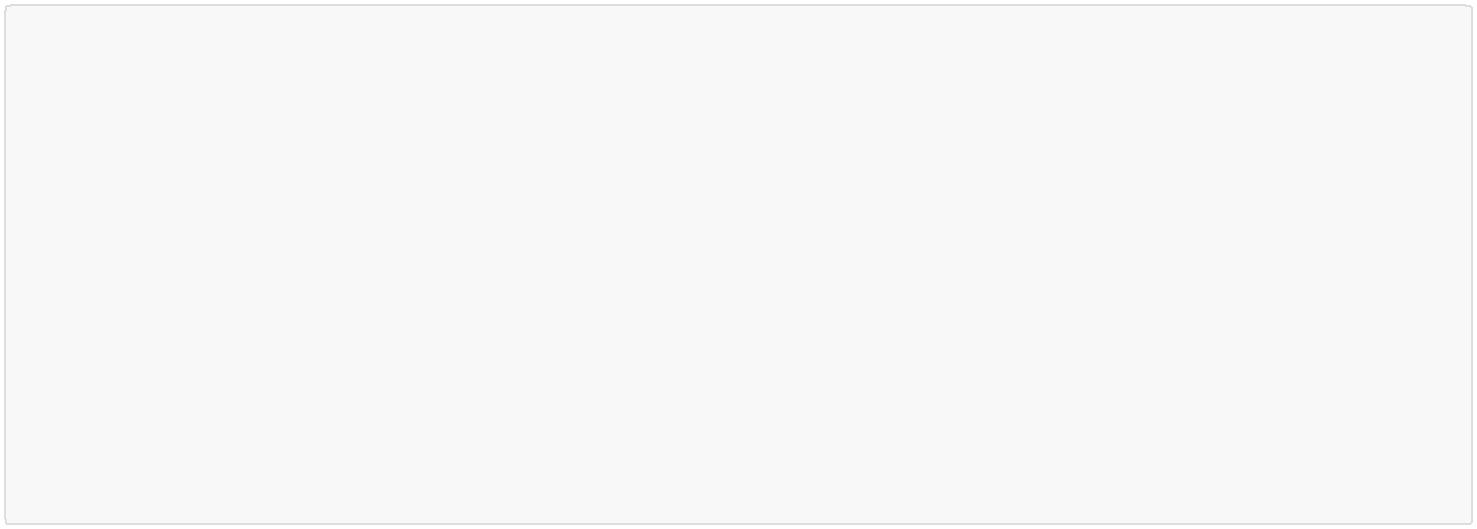
</el‐button>

</template>

</el‐table‐column>

3、选择媒资文件方法

用户点击“选择”按钮将向父组件传递媒资文件信息



choose(mediaFile){

if(mediaFile.processStatus !='303002' && mediaFile.processStatus !='303004'){ this.$message.error('该文件未处理，不允许选择');

return ;

}

if(!mediaFile.fileUrl){

this.$message.error('该文件的访问url为空，不允许选择');

return ;

}

//调用父组件的choosemedia方法

this.$emit('choosemedia',mediaFile.fileId,mediaFile.fileOriginalName);

}

3.3 保存视频信息



3.3.1 需求分析

用户进入课程计划页面，选择视频，将课程计划与视频信息保存在课程管理数据库中。

用户操作流程：

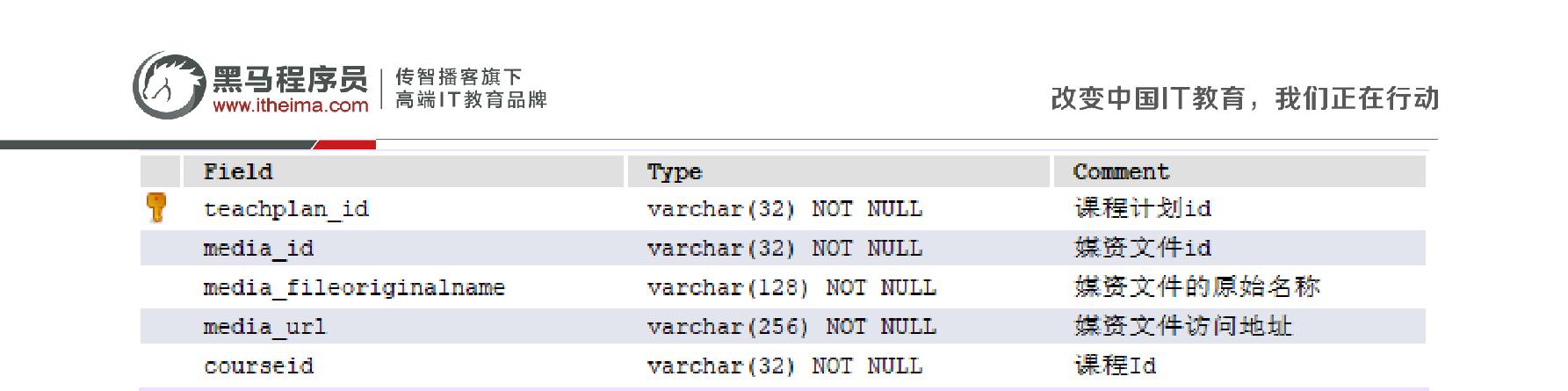
1、进入课程计划，点击”选择视频“，打开我的媒资查询页面

2、为课程计划选择对应的视频，选择“选择”

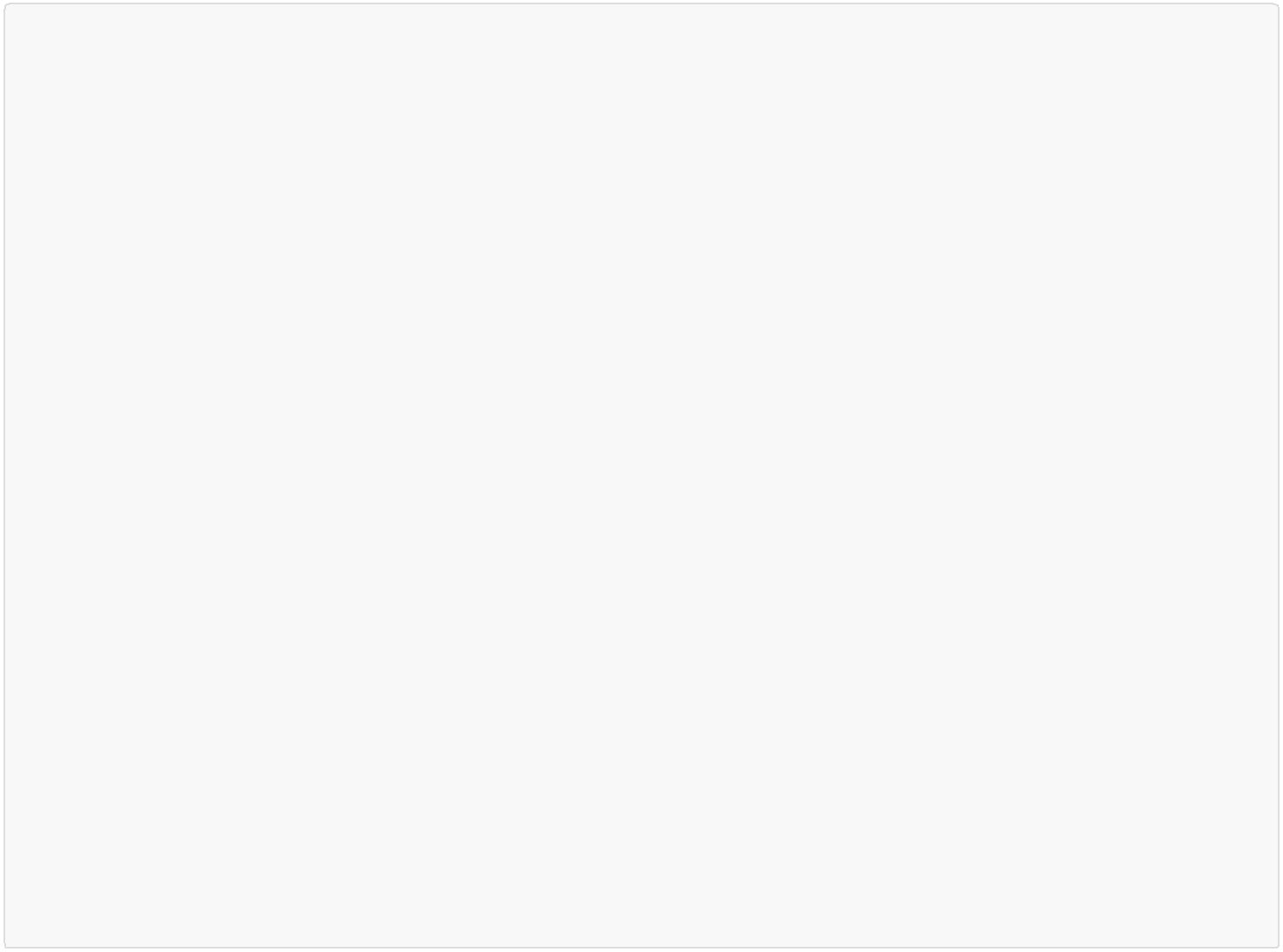
3、前端请求课程管理服务保存课程计划与视频信息。

3.3.2 数据模型

在课程管理数据库创建表 teachplan\_media 存储课程计划与媒资关联信息，如下：

**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

创建teachplanMedia 模型类：



@Data

@ToString

@Entity

@Table(name="teachplan\_media")

@GenericGenerator(name = "jpa‐assigned", strategy = "assigned")

public class TeachplanMedia implements Serializable {

private static final long serialVersionUID = ‐916357110051689485L; @Id

@GeneratedValue(generator = "jpa‐assigned")

@Column(name="teachplan\_id")

private String teachplanId;

@Column(name="media\_id")

private String mediaId;

@Column(name="media\_fileoriginalname")

private String mediaFileOriginalName;

@Column(name="media\_url")

private String mediaUrl;

@Column(name="courseid")

private String courseId;

}

3.3.3 API接口

此接口作为前端请求课程管理服务保存课程计划与视频信息的接口：

在课程管理服务增加接口：



@ApiOperation("保存媒资信息")

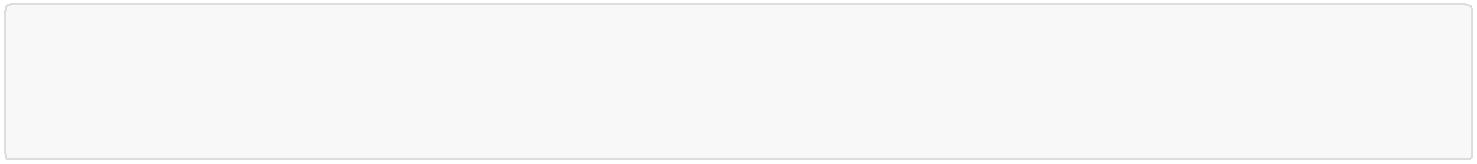
public ResponseResult savemedia(TeachplanMedia teachplanMedia);

3.3.4 服务端开发

3.3.3.1 DAO

**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

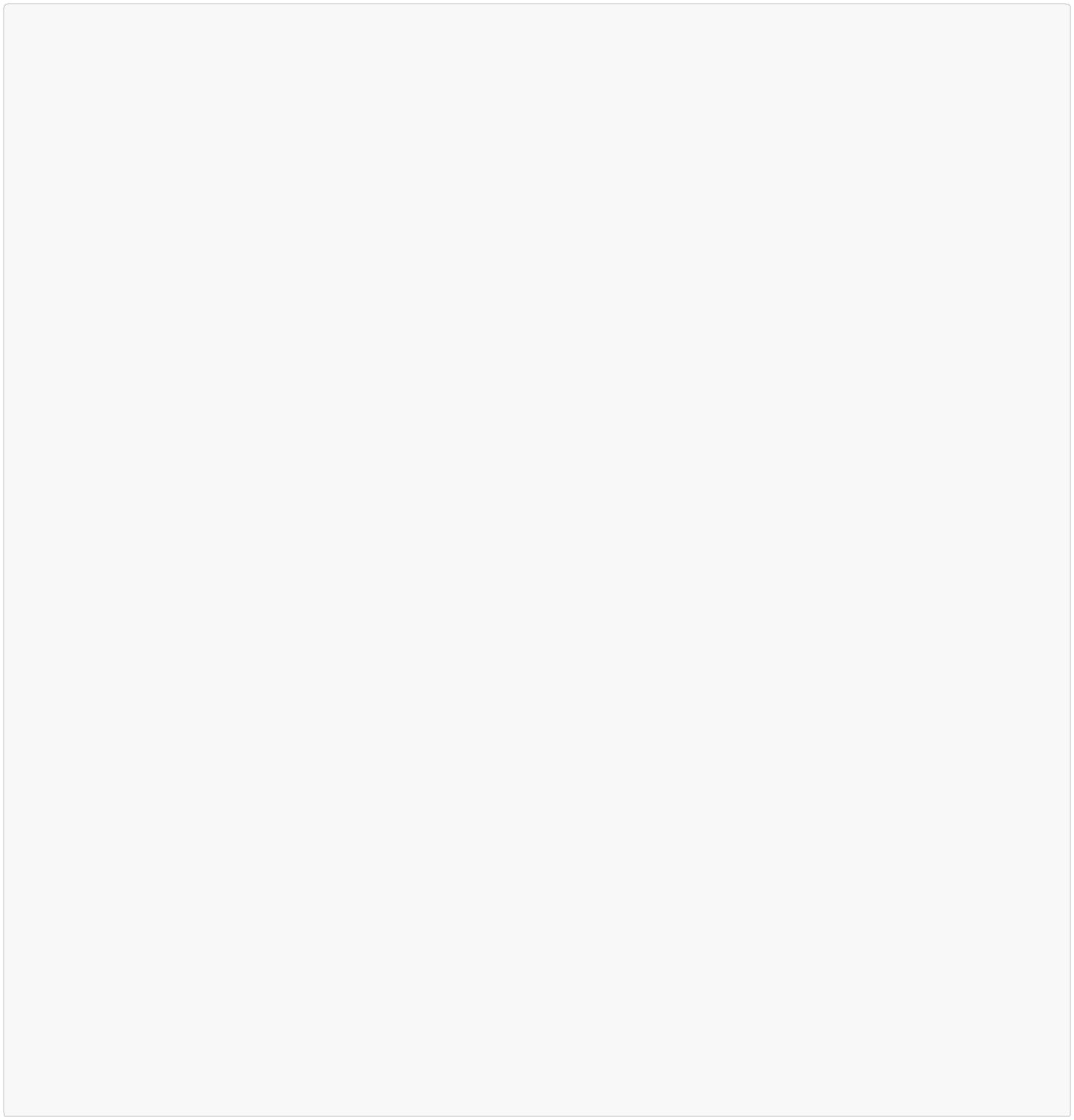
创建TeachplanMediaRepository用于对TeachplanMedia的操作。



public interface TeachplanMediaRepository extends JpaRepository<TeachplanMedia, String> {

}

3.3.3.2 Service



//保存媒资信息

public ResponseResult savemedia(TeachplanMedia teachplanMedia) { if(teachplanMedia == null){

ExceptionCast.cast(CommonCode.INVALIDPARAM);

}

//课程计划

String teachplanId = teachplanMedia.getTeachplanId();

//查询课程计划

Optional<Teachplan> optional = teachplanRepository.findById(teachplanId); if(!optional.isPresent()){

ExceptionCast.cast(CourseCode.COURSE\_MEDIA\_TEACHPLAN\_ISNULL);

}

Teachplan teachplan = optional.get(); //只允许为叶子结点课程计划选择视频 String grade = teachplan.getGrade(); if(StringUtils.isEmpty(grade) || !grade.equals("3")){

ExceptionCast.cast(CourseCode.COURSE\_MEDIA\_TEACHPLAN\_GRADEERROR);

}

TeachplanMedia one = null;

Optional<TeachplanMedia> teachplanMediaOptional =

teachplanMediaRepository.findById(teachplanId);

if(!teachplanMediaOptional.isPresent()){

one = new TeachplanMedia();

}else{

one = teachplanMediaOptional.get();

}

//保存媒资信息与课程计划信息

one.setTeachplanId(teachplanId);

one.setCourseId(teachplanMedia.getCourseId());

one.setMediaFileOriginalName(teachplanMedia.getMediaFileOriginalName());

one.setMediaId(teachplanMedia.getMediaId());

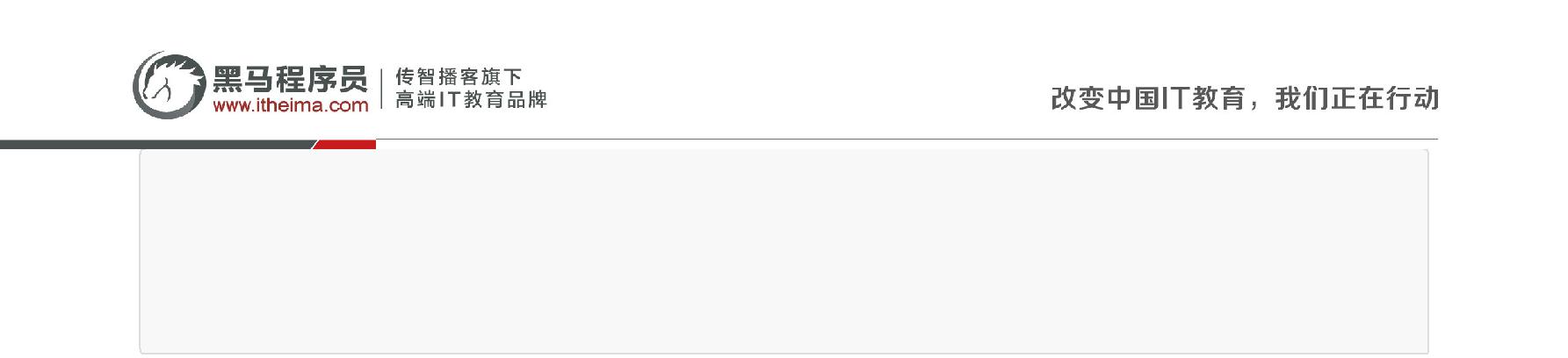
one.setMediaUrl(teachplanMedia.getMediaUrl());

teachplanMediaRepository.save(one);

return new ResponseResult(CommonCode.SUCCESS);

}

3.3.3.3 Controller

**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

@Override

@PostMapping("/savemedia")

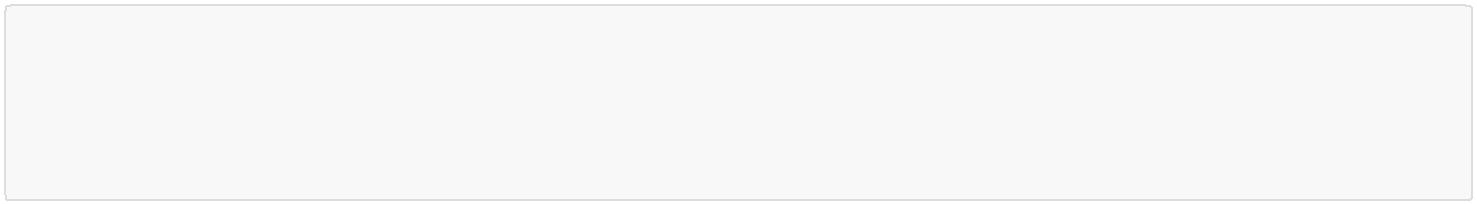
public ResponseResult savemedia(@RequestBody TeachplanMedia teachplanMedia) { return courseService.savemedia(teachplanMedia);

}

3.3.4 前端开发

3.3.4.1 API方法

定义api方法，调用课程管理服务保存媒资信息接口



/\*保存媒资信息\*/

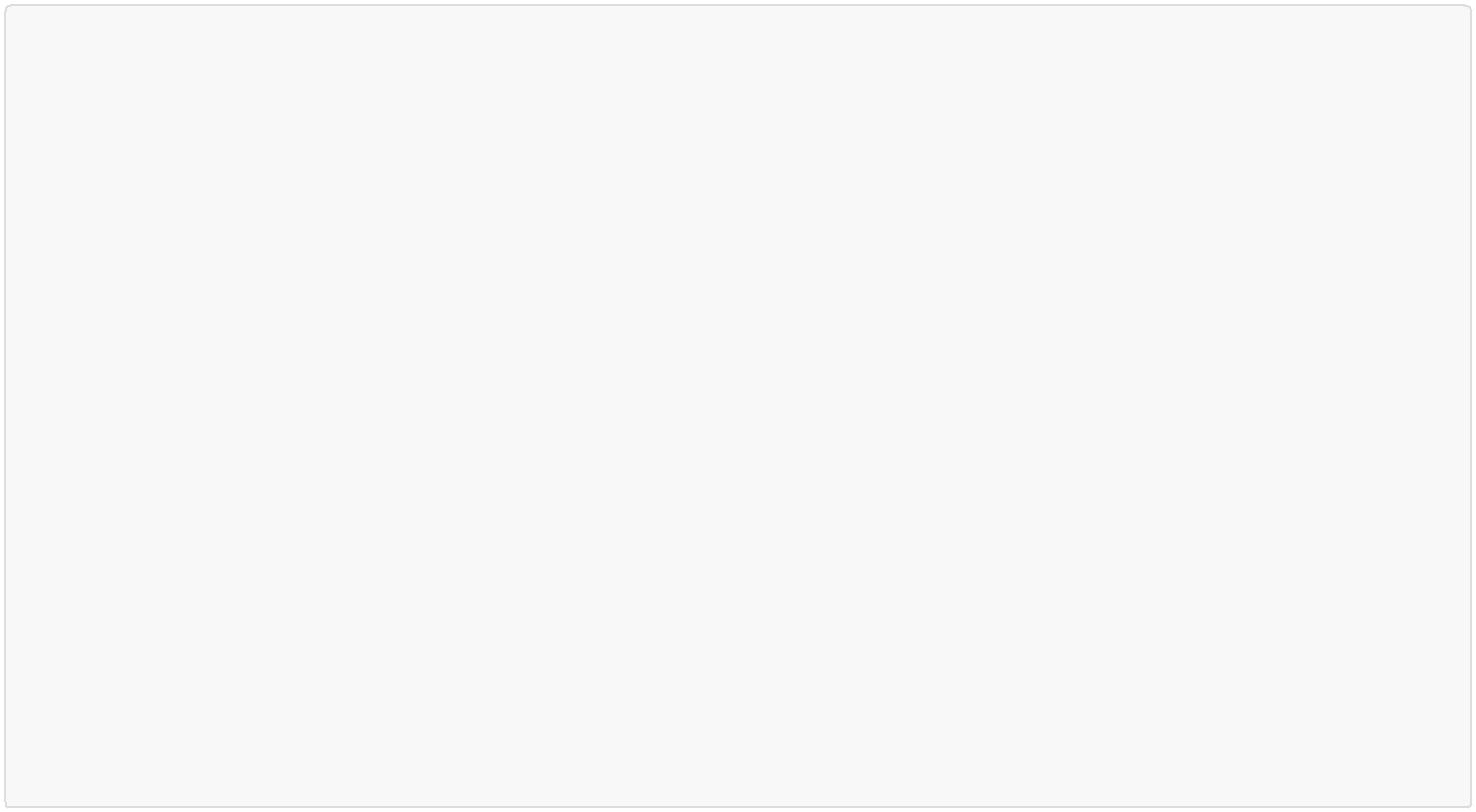
export const savemedia = teachplanMedia => {

return http.requestPost(apiUrl+'/course/savemedia',teachplanMedia);

}

3.3.4.2 API调用

在课程视频方法中调用api：



choosemedia(mediaId,fileOriginalName,mediaUrl){

this.mediaFormVisible = false;

//保存课程计划与视频对应关系

let teachplanMedia = {};

teachplanMedia.teachplanId = this.activeTeachplanId;

teachplanMedia.mediaId = mediaId;

teachplanMedia.mediaFileOriginalName = fileOriginalName;

teachplanMedia.mediaUrl = mediaUrl;

teachplanMedia.courseId = this.courseid;

//保存媒资信息到课程数据库

courseApi.savemedia(teachplanMedia).then(res=>{

if(res.success){

this.$message.success("选择视频成功")

}else{

this.$message.error(res.message)

}

})

},

3.3.4测试

1、向叶子结点课程计划保存媒资信息

操作结果：保存成功

2、向非叶子结点课程计划保存媒资信息

操作结果：保存失败

**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

3.4 查询视频信息



3.4.1 需求分析

课程计划的视频信息保存后在页面无法查看，本节解决课程计划页面显示相关联的媒资信息。

解决方案：

在获取课程计划树结点信息时将关联的媒资信息一并查询，并在前端显示，下图说明了课程计划显示的区域。

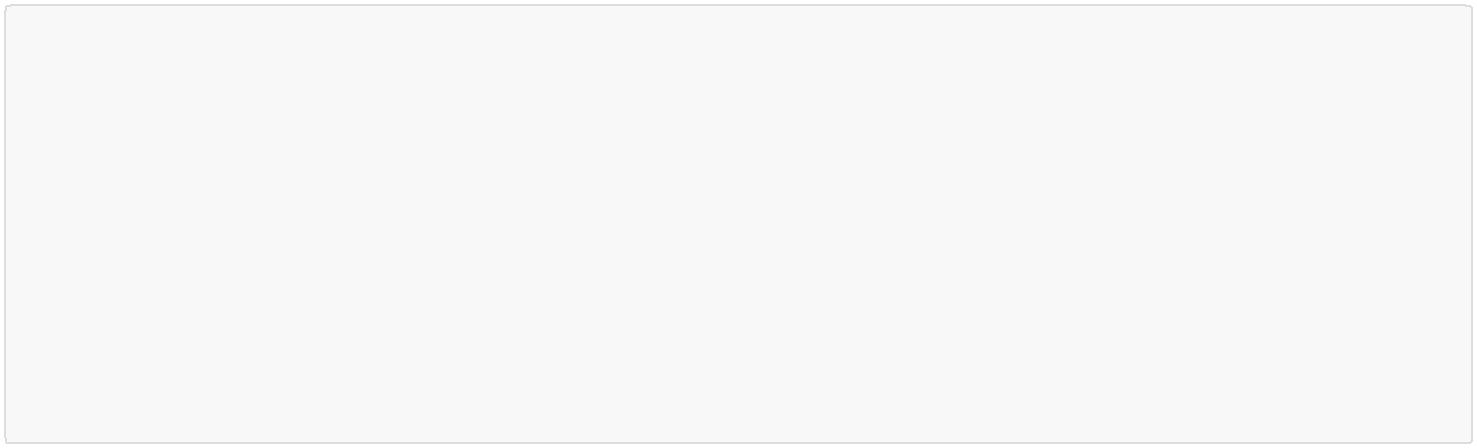


3.4.2 Dao

修改课程计划查询的Dao:

1、修改模型

在课程计划结果信息中添加媒资信息



@Data

@ToString

public class TeachplanNode extends Teachplan {

List<TeachplanNode> children;

//媒资信息

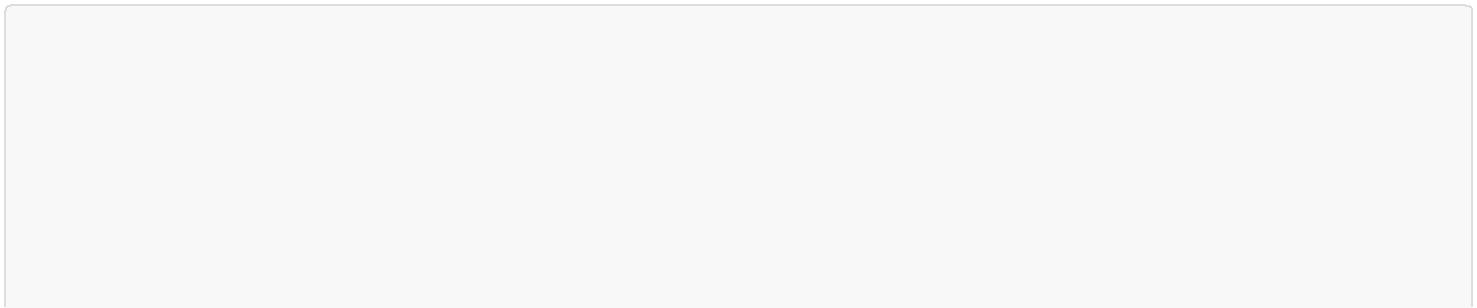
private String mediaId;

private String mediaFileOriginalName;

}

2、修改sql语句，添加关联查询媒资信息

添加mediaId、mediaFileOriginalName



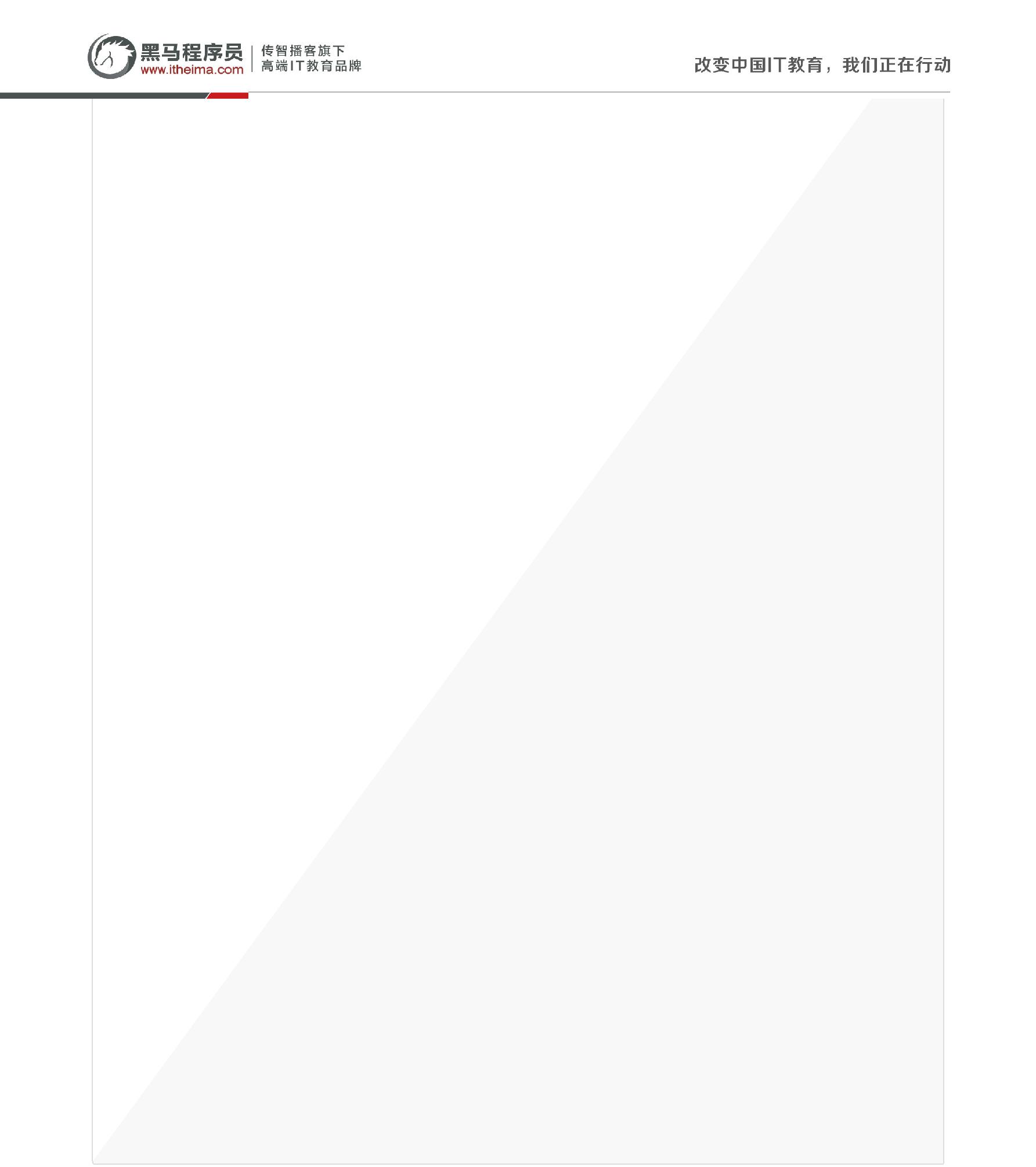
<resultMap type="com.xuecheng.framework.domain.course.ext.TeachplanNode" id="teachplanMap" > <id property="id" column="one\_id"/>

<result property="pname" column="one\_name"/>

<result property="grade" column="one\_grade"/>

<collection property="children"

ofType="com.xuecheng.framework.domain.course.ext.TeachplanNode">

**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

<id property="id" column="two\_id"/>

<result property="pname" column="two\_name"/>

<result property="grade" column="two\_grade"/>

<collection property="children"

ofType="com.xuecheng.framework.domain.course.ext.TeachplanNode"> <id property="id" column="three\_id"/>

<result property="pname" column="three\_name"/>

<result property="grade" column="three\_grade"/>

<result property="mediaId" column="mediaId"/>

<result property="mediaFileOriginalName" column="mediaFileOriginalName"/> </collection>

</collection>

</resultMap>

<select id="selectList" resultMap="teachplanMap" parameterType="java.lang.String" >

SELECT

a.id one\_id,

a.pname one\_name,

a.grade one\_grade,

a.orderby one\_orderby,

b.id two\_id,

b.pname two\_name,

b.grade two\_grade,

b.orderby two\_orderby,

c.id three\_id,

c.pname three\_name,

c.grade three\_grade,

c.orderby three\_orderby,

media.media\_id mediaId,

media.media\_fileoriginalname mediaFileOriginalName

FROM

teachplan a LEFT JOIN teachplan b

ON a.id = b.parentid

LEFT JOIN teachplan c

ON b.id = c.parentid

LEFT JOIN teachplan\_media media

ON c.id = media.teachplan\_id

WHERE a.parentid = '0'

<if test="\_parameter!=null and \_parameter!=''">

and a.courseid=#{courseId}

</if>

ORDER BY a.orderby,

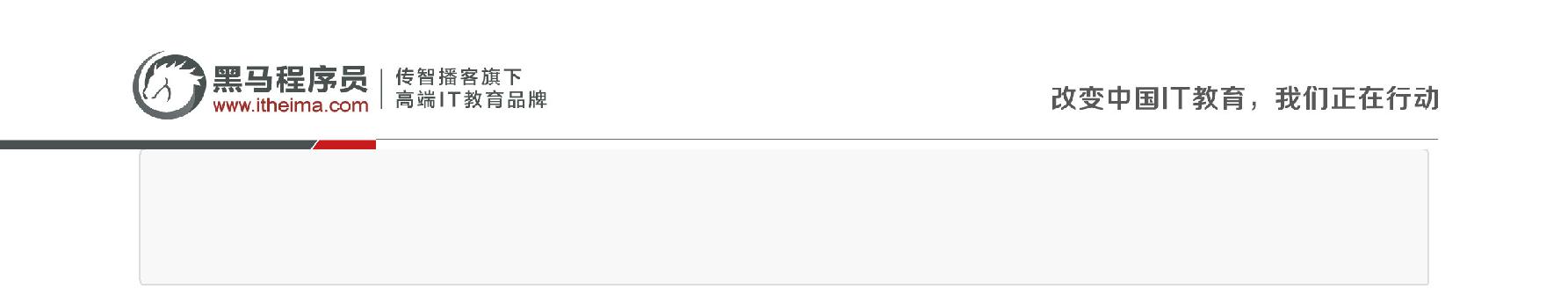
b.orderby,

c.orderby

</select>

3.4.3 页面查询视频

课程计划结点信息已包括媒资信息，可在页面获取信息后显示：

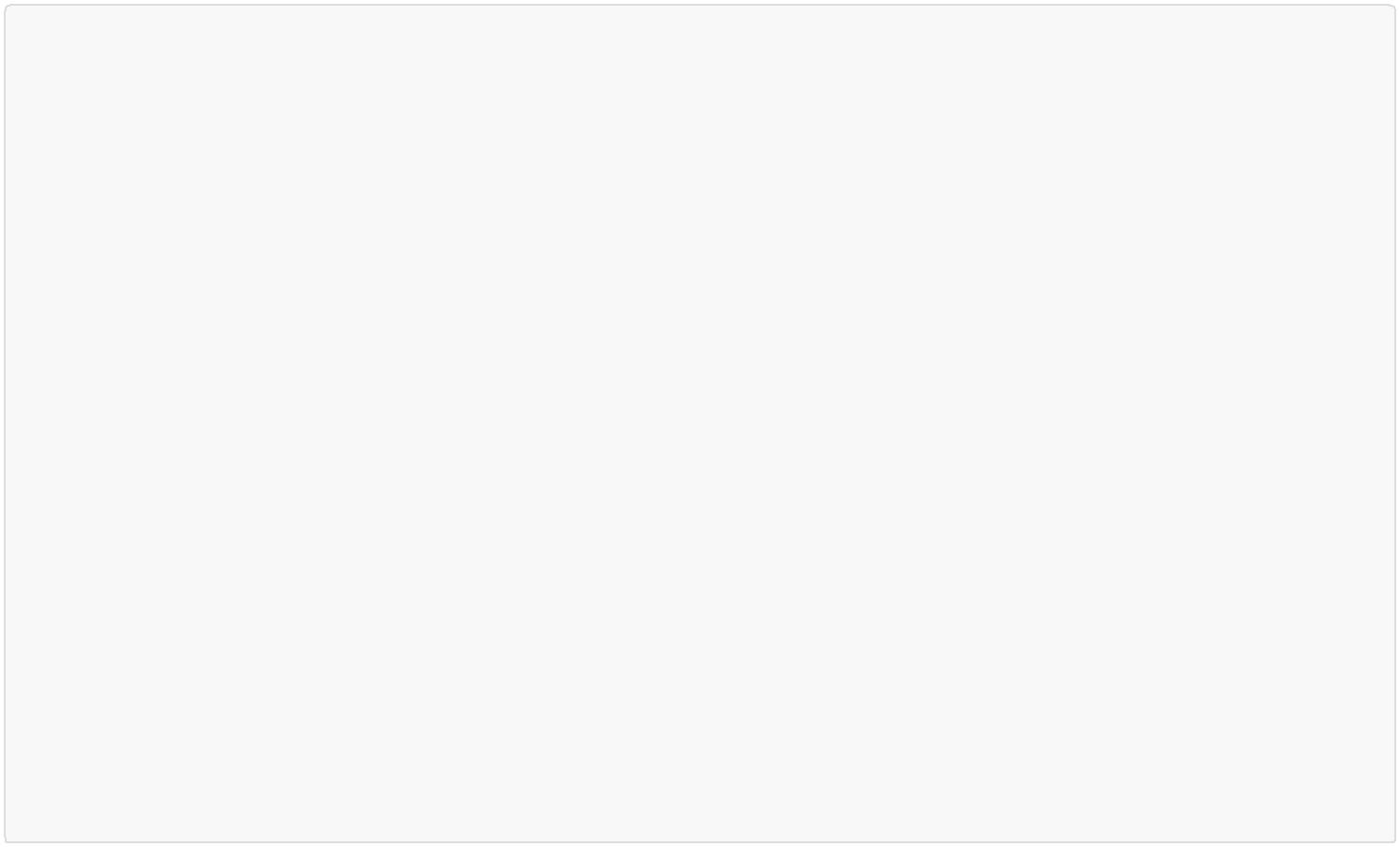
**北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090**

<el‐button style="font‐size: 12px;" type="text" on‐click={ () => this.querymedia(data.id) }> {data.mediaFileOriginalName}&nbsp;&nbsp;&nbsp;&nbsp;选择视频</el‐button>

效果如下：



选择视频后立即刷新课程计划树，在提交成功后，添加查询课程计划代码：this.findTeachplan()，完整代码如下：



choosemedia(mediaId,fileOriginalName,mediaUrl){

this.mediaFormVisible = false;

//保存课程计划与视频对应关系

let teachplanMedia = {};

teachplanMedia.teachplanId = this.activeTeachplanId;

teachplanMedia.mediaId = mediaId;

teachplanMedia.mediaFileOriginalName = fileOriginalName;

teachplanMedia.mediaUrl = mediaUrl;

teachplanMedia.courseId = this.courseid;

//保存媒资信息到课程数据库

courseApi.savemedia(teachplanMedia).then(res=>{

if(res.success){

this.$message.success("选择视频成功")

//查询课程计划

this.findTeachplan()

}else{

this.$message.error(res.message)

}

})

},