目录

SDK for PHP 使用指南

版本	时间	更新内容
V1.0	2016/12/14	新建
V1.1	2017/2/7	修改参数名及返回值为大驼峰
V1.2	2017/3/30	新增更新模版接口
V1.3	2017/4/27	新增轮播相关接口

1.概述

此 SDK 适用于PHP 5.5 及以上版本。基于直播转码 API 构建。使用此 SDK 构建您的网络应用程序，能让您以非常便捷地方式调用金山云的直播转码服务。

2.初始化

2.1 ak/sk配置

在金山云控制台获取到ak/sk后创建以下文件：

```
mkdir ~/.ksyun && vi ~/.ksyun/config
```

config文件内容

```
{
    "ak":"********************",
    "sk":"****************************************"
}
```

2.2 composer安装

```
mkdir test && cd test
composer require kscsdk/ksyun_sdk
```

如果需要使用最新版本，安装完之后，修改composer.json为

```
{
    "require": {
        "kscsdk/ksyun_sdk": "dev-master"
    }
}
```

然后再执行更新操作

```
composer update
```

2.3 调试demo

```
cp vendor/kscsdk/ksyun_sdk/examples/demo_Ket.php .
php demo_Ket.php GetPresetList
```

2.4 调用示例

```
<?php
require('vendor/autoload.php');
use Ksyun\Service\Ket;
$response = Ket::getInstance()->request('GetPresetList');
echo $response->getBody();
```

注：接口的传递的参数及返回值请参考《视频云直播转码接入说明》API接口说明

3. 添加模板

调用示例

```
<?php
require('vendor/autoload.php');
use Ksyun\Service\Ket;
$app = 'live'; // 频道名
$uniqname = 'test'; // 用户名
$preset = 'preset_demo'; // 模板名
$preset_data = [
    'Preset' => $preset,
    'App' => $app,
    'Description' => 'desc: preset demo',
    'Output' => [
        [
            'format' => [
                'output_format' => 256,
                'abr' => 60000,
                'vbr' => 800000,
                'fr' => 25,
                'logo_switch' => 1
            ]
        ],
        [
            'format' => [
                'output_format' => 257,
                'abr' => 70000,
                'vbr' => 700000,
                'fr' => 23,
                'logo_switch' => 0
            ]
        ]
    ],
    'Video' => [
        'logo' => [
            [
                'pic' => 'wangshuai9/ksyun.png',
                'short_side' => 480,
                'disable_scale' => 0,
                'offsetX' => 10,
                'offsetY' => -20
            ]
```

```
        ]
    ]
];
$response = Ket::getInstance()->request('Preset', ['query' => ['Uniqname' => $uniqname], 'json' => $preset_data]);
echo $response->getBody();
```

4.修改模板

调用示例

```
<?php
require('vendor/autoload.php');
use Ksyun\Service\Ket;
$app = 'live'; // 频道名
$uniqname = 'test'; // 用户名
$preset = 'preset_demo'; // 模板名
$preset_data = [
    'Preset' => $preset,
    'App' => $app,
    'Description' => 'desc: preset demo',
    'Output' => [
        [
            'format' => [
                'output_format' => 256,
                'abr' => 60000,
                'vbr' => 800000,
                'fr' => 25,
                'logo_switch' => 1
            ]
        ],
        [
            'format' => [
                'output_format' => 257,
                'abr' => 70000,
                'vbr' => 700000,
                'fr' => 23,
                'logo_switch' => 0
            ]
        ]
    ],
    'Video' => [
        'logo' => [
            [
                'pic' => 'wangshuai9/ksyun.png',
```

```
                'short_side' => 480,
                'disable_scale' => 0,
                'offsetX' => 10,
                'offsetY' => -20
            ]
        ]
    ]
];
$response = Ket::getInstance()->request('UpdatePreset', ['query' => ['Uniqname' => $uniqname], 'json' => $preset_data]);
echo $response->getBody();
```

5. 删除模板

调用示例

```
<?php
require('vendor/autoload.php');
use Ksyun\Service\Ket;
$app = 'live'; // 频道名
$uniqname = 'test'; // 用户名
$preset = 'preset_demo'; // 模板名
$response = Ket::getInstance()->request('DelPreset', ['query' => ['Uniqname' => $uniqname, 'App' => $app, 'Preset' => $preset]]);
echo $response->getBody();
```

6. 查询模板列表

调用示例

```
<?php
require('vendor/autoload.php');
use Ksyun\Service\Ket;
$app = 'live'; // 频道名
$uniqname = 'test'; // 用户名
$response = Ket::getInstance()->request('GetPresetList', ['query' => ['Uniqname' => $uniqname, 'App' => $app]]);
echo $response->getBody();
```

7. 查询模板详情

调用示例

```
<?php
require('vendor/autoload.php');
use Ksyun\Service\Ket;
$app = 'live'; // 频道名
$uniqname = 'test'; // 用户名
$preset = 'preset_demo'; // 模板名
$response = Ket::getInstance()->request('GetPresetDetail', ['query' => ['Uniqname' => $uniqname, 'App' => $app, 'Preset' => $prese
t]]);
echo $response->getBody();
```

8. 获取任务列表

调用示例

```
<?php

require('vendor/autoload.php');

use Ksyun\Service\Ket;

$app = 'live'; // 频道名

$uniqname = 'test'; // 用户名

$response = Ket::getInstance()->request('GetStreamTranList', ['query' => ['Uniqname' => $uniqname, 'App' => $app]]);

echo $response->getBody();
```

9. 发起外网拉流

调用示例

```
<?php

require('vendor/autoload.php');

use Ksyun\Service\Ket;

$app = 'live'; // 频道名

$uniqname = 'test'; // 用户名

$streamid = 'stream20170101'; // 流名

$outpull_data = [

    'App' => $app,

    'StreamID' => $streamid,

    'SrcUrl' => 'rtmp://test.rtmplive.ks-cdn.com/live/streamdemo',

    'Params' => ''

];

$response = Ket::getInstance()->request('StartStreamPull', ['query' => ['Uniqname' => $uniqname], 'json' => $outpull_data ]);

echo $response->getBody();
```

10. 停止外网拉流

调用示例

```
<?php

require('vendor/autoload.php');

use Ksyun\Service\Ket;

$app = 'live'; // 频道名

$uniqname = 'test'; // 用户名

$streamid = 'stream20170101'; // 流名

$response = Ket::getInstance()->request('StopStreamPull', ['query' => ['Uniqname' => $uniqname], 'json' => ['App' => $app, 'StreamID' => $streamid]]);

echo $response->getBody();
```

11. 发起轮播

调用示例

```
<?php

require('vendor/autoload.php');
```

```
use Ksyun\Service\Ket;

$app = 'live'; // 频道名

$preset = 'preset_demo'; // 模板名

$uniqname = 'test'; // 用户名

$streamid = 'stream20170101'; // 流名

$loop_data = [
    'App' => $app,
    'Preset' => $preset,
    'StreamID' => $streamid,
    'PubDomain' => 'test.uplive.ksyun.com',
    'DurationHour' => 168,
    'SrcInfo' => array(
        array(
            'Path' => 'http://wangshuai9.ks3-cn-beijing.ksyun.com/ksyun.flv',
            'Index' => 0,
        )
    )
];

$response = Ket::getInstance()->request('StartLoop', ['query' => ['UniqName' => $uniqname], 'json' => $loop_data]);

echo $response->getBody();
```

12.停止轮播

调用示例

```
<?php

require('vendor/autoload.php');

use Ksyun\Service\Ket;

$app = 'live'; // 频道名

$uniqname = 'test'; // 用户名

$streamid = 'stream20170101'; // 流名

$response = Ket::getInstance()->request('StopLoop', ['query' => ['UniqName' => $uniqname], 'json' => ['App' => $app, 'StreamID' =
> $streamid]]);

echo $response->getBody();
```

13.更新轮播时长

调用示例

```
<?php

require('vendor/autoload.php');

use Ksyun\Service\Ket;

$app = 'live'; // 频道名

$uniqname = 'test'; // 用户名

$streamid = 'stream20170101'; // 流名

$response = Ket::getInstance()->request('UpdateLoop', ['query' => ['UniqName' => $uniqname], 'json' => ['App' => $app, 'StreamID'
 => $streamid, 'DurationHour' => 168]]);
```

```
echo $response->getBody();
```

14.获取轮播列表

调用示例

```
<?php
require('vendor/autoload.php');
use Ksyun\Service\Ket;
$app = 'live'; // 频道名
$uniqname = 'test'; // 用户名
$response = Ket::getInstance()->request('GetLoopList', ['query' => ['UniqName' => $uniqname, 'App' => $app]]);
echo $response->getBody();
```

15.获取配额使用数据

调用示例

```
<?php
require('vendor/autoload.php');
use Ksyun\Service\Ket;
$uniqname = 'test'; // 用户名
$response = Ket::getInstance()->request('GetQuotaUsed', ['query' => ['Uniqname' => $uniqname]]);
echo $response->getBody();
```

SDK for Python 使用指南

安装python sdk

1.git 安装

```
git clone https://github.com/KscSDK/ksc-sdk-python.git
cd ksc-sdk-python
python setup.py install
```

2.pip 安装

```
pip install ksc-sdk-python
```

3.通过文件配置及管理密钥

- 所在位置: '/etc/kscore.cfg' 或 './.kscore.cfg'
- 注意: 使用相对路径时，需与运行目录保持一致
- ks_access_key_id和ks_secret_access_key是金山云控制台身份与管理里面生成密钥对

```
[Credentials]
ks_access_key_id='xxxxxxxxxxxxxxxxxxx'
ks_secret_access_key='xxxxxxxxxxxxxxxxxxxxxxx'
```

4.创建一个session （初始化）

```
from kscore.ket import getKetClient
```

第一种：

```
client = getKetClient("ket", "cn-beijing-6",use_ssl=False)
```

参数说明：

service_name: 服务名，填写"ket"

region_name: 区域名，填写"cn-beijing-6"

use_ssl: 是否https访问，填写False

第二种：

没有配置kscore.cfg调用方式

```
ks_access_key_id='xxxxxxxxxxxxxxxxxxx'
```

```
ks_secret_access_key='xxxxxxxxxxxxxxxxxxxxxxx'
```

参数：服务service_name,大区region_name

```
client = getKetClient("ket", "cn-beijing-6",use_ssl=False,ks_access_key_id=ks_access_key_id,ks_secret_access_key=ks_secret_access_key)
```

5. 运行环境

适用于2.6、2.7、3.3、3.4的Python版本

SDK介绍和使用

接口目录说明

- Preset 创建模板
- DelPreset 删除模板
- GetPresetList 查询模板列表
- GetPresetDetail 查询模板详情
- GetStreamTranList 获取任务列表
- StartStreamPull 发起外网拉流
- StopStreamPull 停止外网拉流
- StartLoop 发起轮播
- StopLoop 停止轮播
- UpdateLoop 更新轮播时长
- GetLoopList 获取轮播列表
- GetQuotaUsed 获取配额使用数据
- CreateDirectorTask 创建选流任务
- UpdateDirectorTask 更新选流任务
- QueryDirectorTask 查询选流任务
- DelDirectorTask 删除选流任务

注：接口的传递的参数及返回值请参考《视频云直播转码接入说明》API接口说明

1. 创建和更新模板

- 接口名: Preset/UpdatePreset
- 调用方式

```
presetParam = {
    "UniqName":"test",
```

```
    "Preset":presetname,
    "App":app,
    "Description":description,
    "Output":[
        {
            "format":{
                "output_format":256,
                "abr": 70000,
                "vbr": 700000,
                "fr": 23
            }
        },
        {
            "format":{
                "output_format":257
            }
        }
    ]
}
```

注：

1.presetParam必须是json格式数据

```
res = client.Preset(presetParam)
```

2.删除模板

- 接口名: DelPreset
- 调用方式

```
res = client.DelPreset(App="live",UniqName="test",Preset=presetname)
```

3.查询模板列表

- 接口名: GetPresetList
- 调用方式

```
res = client.GetPresetList(App="live",UniqName="test")
```

4.查询模板详情

- 接口名: GetPresetDetail
- 调用方式

```
res = client.GetPresetDetail(App="live",UniqName="test",Preset=presetname)
```

5.获取任务列表<

- 接口名: GetStreamTranList
- 调用方式

```
res = client.GetStreamTranList(App="live",UniqName="test",StreamID="test",OutPull=-1)
```

6.发起外网拉流

- 接口名: StartStreamPull
- 调用方式

```
StartStreamPullParam = {
    "UniqName":"test",
    "App":"live",
    "StreamID":"testName",
    "SrcUrl":"test.uplive.ks-cdn.com"
}
```

注:

1.StartStreamPullParam必须是json格式数据

```
res = client.StartStreamPull(StartStreamPullParam)
```

7.停止外网拉流

- 接口名: StopStreamPull
- 调用方式

```
StopStreamPullParam = {
    "UniqName":"test",
    "App":"live",
    "StreamID":"testName"
}
```

注:

1.StopStreamPullParam必须是json格式数据

```
res = client.StopStreamPull(StopStreamPullParam)
```

8.发起轮播

- 接口名: StartLoop
- 调用方式

```
StartLoopParam = {
   "UniqName": "test",
   "App": "live",
   "StreamID": "testName",
   "Preset": "1080",
   "PubDomain": "test.uplive.ksyun.com",
   "DurationHour":168,
   "SrcInfo": [
     {
        "Path": "http://wangshuai9.ks3-cn-beijing.ksyun.com/ksyun.flv",
        "Index": 0
     }
   ]
}
```

注:

1.StartLoopParam必须是json格式数据

```
res = client.StartLoop(StartLoopParam)
```

9.停止轮播

- 接口名: StopLoop
- 调用方式

```
StopLoopParam = {
   "UniqName": "test",
   "App": "live",
   "StreamID": "testName"
}注:
1.StopLoopParam必须是json格式数据
res = client.StopLoop(StopLoopParam)
```

10.更新轮播时长

- 接口名: UpdateLoop
- 调用方式

```
UpdateLoopParam = {
   "UniqName": "test",
   "App": "live",
   "StreamID": "testName",
   "DurationHour":100,
}注:
1.UpdateLoopParam必须是json格式数据
res = client.UpdateLoop(UpdateLoopParam)
```

11.查询轮播列表

- 接口名: GetLoopList
- 调用方式

```
res = client.GetLoopList(App=appname, UniqName=uniqname, StreamID=streamid)
```

12.获取配额使用数据

- 接口名: GetQuotaUsed
- 调用方式

```
res = client.GetQuotaUsed(UniqName="test")
```

13.创建选流任务

- 接口名：CreateDirectorTask
- 调用方式

```
param4 = {
   "UniqName": uniqname,
   "App": appname,
   "Preset": presetname,
   "SrcInfo": [
     {
```

```
        "Url": "rtmp://host/app/outernetStreamForSwitch",
        "Idx": 0
      },
      {
        "Streamid": "streamForSwitch",
        "Idx": 1
      }
    ],
    "DstInfo": [
      {
        "Streamid":"stream0ForMonitor",
        "Idx": 0
      },
      {
        "Streamid": "stream1ForSwitch",
        "Idx": 1
      },
      {
        "Streamid": "stream2ForSwitch",
        "Idx": 2
      }
    ]
}
res = client.CreateDirectorTask(param4)
print json.dumps(res)
```

14.更新选流任务

- 接口名：UpdateDirectorTask
- 调用方式

```
param4["TaskID"] = taskid
res = client.UpdateDirectorTask(param4)
print json.dumps(res)
```

15.查询选流任务

- 接口名：QueryDirectorTask
- 调用方式

```
res = client.QueryDirectorTask(App=appname, UniqName=uniqname, TaskID=taskid)
print json.dumps(res)
```

16.删除选流任务

- 接口名：DelDirectorTask
- 调用方式

```
res = client.DelDirectorTask(App=appname, UniqName=uniqname, TaskID=taskid)
print json.dumps(res)
```

SDK for Java 使用指南

安装java sdk

1.git 安装

```
git clone https://github.com/KscSDK/ksc-sdk-java.git
mvn clean install
```

2.mvn 中央仓库安装

设置pom文件即可

3.工程使用sdk录

建议使用Maven构建自己的项目，导入ksc-sdk-java,添加ksc-sdk-java-ket的依赖。

新建maven项目

进入到pom文件下，进行如下配置

```
<dependencies>
    <dependency>
        <groupId>com.ksyun</groupId>
        <artifactId>ksc-sdk-java-ket</artifactId>
        <version>0.3.9</version>
    </dependency>
</dependencies>
```

4.通过文件配置及管理密钥

本地文件配置:

```
~/.aws/credentials on Linux, OS X or unix
C:\Users\USERNAME\.aws\credentials on Windows
```

该文件包含下述内容：

```
[default]
aws_access_key_id = your_access_key_id
aws_secret_access_key = your_secret_access_key
```

如不通过本地文件读入ak和sk信息，则需要在执行的类文件中增加以下代码：

```
AWSCredentials credentials = new BasicAWSCredentials(aws_access_key_id, aws_secret_access_key);
```

SDK介绍和使用

在调用接口前，先初始化 KSCOFFJsonClient，如下用例：

```
KSCKETJsonClient ksc = new KSCKETJsonClient();
//可传入AWSCredentials参数 如:KSCKETJsonClient ksc = new KSCKETJsonClient(credentials);
ksc.setEndpoint("http://ket.cn-beijing-6.api.ksyun.com/");
```

字符集：utf-8

SDK介绍和使用

接口目录说明

注：接口的传递的参数及返回值请参考《视频云直播转码接入说明》API接口说明

1.创建和更新模板

- 接口名：Preset和UpdatePreset
- 调用方式

```
PresetRequest presetRequest = new PresetRequest();
String data = PresetSet("preset", 2);
presetRequest.setData(data);
ErrResult presetResult = ksc.Preset(presetRequest);
System.out.println("ErrNum: " + presetResult.getErrNum() + ",ErrMsg: " + presetResult.getErrMsg());
//请求内容生成接口
private static String PresetSet(String preset, int num) throws JSONException {
    int formatSet[] = new int[] { 256, 257, 258, 259 };
    JSONObject data = new JSONObject();
    JSONArray output = new JSONArray();
    JSONArray logo = new JSONArray();
    JSONObject video = new JSONObject();
    data.put("UniqName", "mytest");
    data.put("Preset", preset);
    data.put("Description", "desc:" + preset);
```

```
    data.put("App", "live");
    for (int i = 0; i < num; i++) {
        JSONObject output_tmp = new JSONObject();
        JSONObject format = new JSONObject();
        JSONObject _switch = new JSONObject();
        format.put("output_format", formatSet[i]);
        format.put("vbr", 800000);
        _switch.put("sn", 0);
        _switch.put("sm", 0);
        _switch.put("hv", 0);
        output_tmp.put("format", format);
        output_tmp.put("switch", _switch);
        output.put(output_tmp);
    }
    data.put("Output", output);
    video.put("orientationAdapt", 1);
    for (int i = 0; i < num; i++) {
        JSONObject logo_tmp = new JSONObject();
        logo_tmp.put("pic", "/wangshuai9/1.png");
        logo_tmp.put("short_side", 640);
        logo.put(logo_tmp);
    }
    video.put("logo", logo);
    data.put("Video", video);
    return data.toString();
}
```

2.删除模板

- 接口名: DelPreset
- 调用方式

```
DelPresetRequest delpresetRequest = new DelPresetRequest();
delpresetRequest.setApp("live");
delpresetRequest.setUniqName("mytest");
delpresetRequest.setPreset("preset");
ErrResult delpresetResult = ksc.DelPreset(delpresetRequest);
System.out.println("ErrNum: " + delpresetResult.getErrNum() + ",ErrMsg: " + delpresetResult.getErrMsg());
```

3.查询模板列表

- 接口名: GetPresetList
- 调用方式

```
GetPresetListRequest getPresetListRequest = new GetPresetListRequest();
getPresetListRequest.setApp("live");
```

```
getPresetListRequest.setUniqName("mytest");
GetPresetListResult getpresetlistResult = ksc.GetPresetList(getPresetListRequest);
```

4. 查询模板详情

- 接口名: GetPresetDetail
- 调用方式

```
GetPresetDetailRequest getPresetDetailRequest = new GetPresetDetailRequest();
getPresetDetailRequest.setApp("live");
getPresetDetailRequest.setUniqName("mytest");
getPresetDetailRequest.setPreset("preset");
GetPresetDetailResult getPresetDetailResult = ksc.GetPresetDetail(getPresetDetailRequest);
```

5. 获取任务列表

- 接口名: GetStreamTranList
- 调用方式

```
GetStreamTranListRequest getStreamTranListRequest = new GetStreamTranListRequest();
getStreamTranListRequest.setApp("live");
getStreamTranListRequest.setUniqName("mytest");
GetStreamTranListResult getStreamTranListResult = ksc.GetStreamTranList(getStreamTranListRequest);
```

6. 发起外网拉流

- 接口名: StartStreamPull
- 调用方式

```
StartStreamPullRequest startStreamPullRequest = new StartStreamPullRequest();
String data1 =  StreamPullSet("test123");
startStreamPullRequest.setData(data1);
ErrResult startStreamPullResult = ksc.StartStreamPull(startStreamPullRequest);
System.out.println("ErrNum: " + startStreamPullResult.getErrNum() + ",ErrMsg: " + startStreamPullResult.getErrMsg());
//请求内容接口
private static String StreamPullSet(String StreamID) {
    JSONObject data = new JSONObject();
    data.put("UniqName", "mytest");
    data.put("StreamID", StreamID);
    if (type == 0) {
        data.put("SrcUrl", "test.uplive.ks-cdn.com");
    }
    data.put("App", "live");
    return data.toString();
}
```

7. 停止外网拉流

- 接口名: StopStreamPull

- 调用方式

```
StopStreamPullRequest stopStreamPullRequest = new StopStreamPullRequest();
String data2 =  StreamPullSet("test123");
stopStreamPullRequest.setData(data2);
ErrResult stopStreamPullResult = ksc.StopStreamPull(stopStreamPullRequest);
System.out.println("ErrNum: " + stopStreamPullResult.getErrNum() + ",ErrMsg: " + stopStreamPullResult.getErrMsg());
//请求内容接口
private static String StreamPullSet(String StreamID) {
    JSONObject data = new JSONObject();
    data.put("UniqName", "mytest");
    data.put("StreamID", StreamID);
    if (type == 0) {
        data.put("SrcUrl", "test.uplive.ks-cdn.com");
    }
    data.put("App", "live");
    return data.toString();
}
```

8. 发起轮播

- 接口名: StartLoop
- 调用方式

```
StartLoopRequest startLoopRequest = new StartLoopRequest();
String startLoopData = StartLoopSet();
startLoopRequest.setData(startLoopData);
StartLoopResult startLoopResult = ksc.StartLoop(startLoopRequest);
System.out.println(startLoopResult.getList().get(0).getTaskID());
//请求内容接口
private static String StartLoopSet() {
    JSONObject data = new JSONObject();
    JSONArray srcInfo = new JSONArray();
    data.put("PubDomain", "videoqa.uplive.ks-cdn.com");
    data.put("UniqName", UniqName);
    data.put("Preset", "looppreset");
    data.put("StreamID", "java_sdk_1234");
    data.put("App", "live");
    data.put("DurationHour", 168);
    JSONObject tmp = new JSONObject();
    tmp.put("Path", "http://ks3-cn-beijing-internal.ksyun.com/qa-screenshot/offline_source_file/offline_mkv.mkv");
    tmp.put("Index", 0);
    srcInfo.put(tmp);
    data.put("SrcInfo", srcInfo);
    return data.toString();
```

```
}
```

9. 停止轮播

- 接口名: StopLoop
- 调用方式

```
StopLoopRequest stopLoopRequest = new StopLoopRequest();
String stopLoopData = StopLoopSet();
stopLoopRequest.setData(stopLoopData);
ErrResult stopLoopResult = ksc.StopLoop(stopLoopRequest);
//请求内容接口
private static String StopLoopSet() {
    JSONObject data = new JSONObject();
    data.put("App", "live");
    data.put("UniqName", UniqName);
    data.put("StreamID", "java_sdk_1234");
    return data.toString();
}
```

10. 更新轮播时长

- 接口名: UpdateLoop
- 调用方式

```
UpdateLoopRequest updateLoopRequest = new UpdateLoopRequest();
String updateLoopData = UpdateLoopSet();
updateLoopRequest.setData(updateLoopData);
ErrResult updateLoop = ksc.UpdateLoop(updateLoopRequest);
System.out.println(updateLoop.getErrMsg());
//请求内容接口
private static String UpdateLoopSet() {
    JSONObject data = new JSONObject();
    data.put("App", "live");
    data.put("UniqName", UniqName);
    data.put("StreamID", "java_sdk_1234");
    data.put("DurationHour", 10);
    return data.toString();
}
```

11. 获取轮播列表

- 接口名: GetLoopList
- 调用方式

```
GetLoopListRequest getLoopListRequest = new GetLoopListRequest();
getLoopListRequest.setApp("live");
getLoopListRequest.setUniqName(UniqName);
```

```
GetLoopListResult getLoopListResult = ksc.GetLoopList(getLoopListRequest);
System.out.println(getLoopListResult.getList().get(0).getTaskID());
```

12. 获取配额使用数据

- 接口名: GetQuotaUsed
- 调用方式

```
GetQuotaUsedRequest getQuotaUsedRequest = new GetQuotaUsedRequest();
getQuotaUsedRequest.setUniqName(UniqName);
GetQuotaUsedResult getQuotaUsedResult = ksc.GetQuotaUsed(getQuotaUsedRequest);
System.out.println("ErrNum: " + getQuotaUsedResult.getErrNum() + ",ErrMsg: " + getQuotaUsedResult.getErrMsg());
```

13. 创建选流任务

- 接口名：CreateDirectorTask
- 调用方式

```
CreateDirectorTaskRequest createDirectorTaskRequest = new CreateDirectorTaskRequest();
createDirectorTaskRequest.setData(DirectorTaskSet());
ErrResult createDirectorTaskResult = ksc.CreateDirectorTask(createDirectorTaskRequest);
System.out.println(createDirectorTaskResult.getTaskID());
private static String DirectorTaskSet(){
    JSONObject data = new JSONObject();
    data.put("App", "live");
    data.put("UniqName", UniqName);
    JSONArray SrcInfo = new JSONArray();
    for (int i = 0; i < 2; i++) {
        JSONObject info = new JSONObject();
        info.put("SrcUrl","rtmp://host/app/outernetStreamForSwitch");
        info.put("Index",i);
        SrcInfo.put(info);
    }
    data.put("SrcInfo", SrcInfo);
    return data.toString();
}
```

14. 更新选流任务

- 接口名：UpdateDirectorTask
- 调用方式

```
UpdateDirectorTaskRequest updateDirectorTaskRequest = new UpdateDirectorTaskRequest();
updateDirectorTaskRequest.setData(DirectorTaskSet());
ErrResult updateDirectorTaskResult = ksc.UpdateDirectorTask(updateDirectorTaskRequest);
System.out.println(updateDirectorTaskResult.getErrNum());
private static String DirectorTaskSet(){
    JSONObject data = new JSONObject();
```

```
    data.put("App", "live");
    data.put("UniqName", UniqName);
    JSONArray SrcInfo = new JSONArray();
    for (int i = 0; i < 2; i++) {
        JSONObject info = new JSONObject();
        info.put("SrcUrl","rtmp://host/app/outernetStreamForSwitch");
        info.put("Index",i);
        SrcInfo.put(info);
    }
    data.put("SrcInfo", SrcInfo);
    return data.toString();
}
```

15. 查询选流任务

- 接口名：QueryDirectorTask
- 调用方式

```
QueryDirectorTaskRequest queryDirectorTaskRequest = new QueryDirectorTaskRequest();
queryDirectorTaskRequest.setApp("live");
queryDirectorTaskRequest.setUniqName(UniqName);
QueryDirectorTaskResult queryDirectorTaskResult = ksc.QueryDirectorTask(queryDirectorTaskRequest);
System.out.println(queryDirectorTaskResult.getErrNum());
```

16. 删除选流任务

- 接口名：DelDirectorTask
- 调用方式

```
DelDirectorTaskRequest delDirectorTaskRequest = new DelDirectorTaskRequest();
delDirectorTaskRequest.setApp("live");
delDirectorTaskRequest.setUniqName(UniqName);
ErrResult delDirectorTaskResult = ksc.DelDirectorTask(delDirectorTaskRequest);
System.out.println(delDirectorTaskResult.getErrNum());
```

SDK for PHP 使用指南

概述

此 SDK 适用于PHP 5.5 及以上版本。基于离线转码 API 构建。使用此 SDK 构建您的网络应用程序，能让您以非常便捷地方式调用金山云的离线转码服务。

初始化

ak/sk配置

在金山云控制台获取到ak/sk后创建以下文件：

```
mkdir ~/.ksyun && vi ~/.ksyun/config
```

config文件内容

```
{
    "ak":"********************",
    "sk":"****************************************"
}
```

composer安装

```
mkdir test && cd test
composer require kscsdk/ksyun_sdk
```

如果需要使用最新版本，安装完之后，修改composer.json为

```
{
    "require": {
        "kscsdk/ksyun_sdk": "dev-master"
    }
}
```

然后再执行更新操作

```
composer update
```

调试demo

```
cp vendor/kscsdk/ksyun_sdk/examples/demo_Kvs.php .
php demo_Kvs.php GetPresetList
```

调用示例

```
<?php
require('vendor/autoload.php');
use Ksyun\Service\Kvs;
$response = Kvs::getInstance()->request('GetPresetList');
echo $response->getBody();
```

创建模板

- 接口名

Preset

- 调用方式

```
<?php
require('vendor/autoload.php');
use Ksyun\Service\Kvs;

// 拼凑模板数组
$preset_data = array(
        'Preset' => 'xxx',
        'Description' => 'xxx',
        'PresetType' => 'xxx',
```

```
        'Param' => array(
                'f' => 'xxx',
                'VIDEO' => array(
                        'vr' => '13',
                        'vb' => '780000',
                        'vcodec' => 'h264',
                        'width' => 500,
                        'height' => 600
                ),
                'AUDIO' => array(
                        'ar' => '44100',
                        'ab' => '64k',
                        'acodec' => 'aac',
                        'an' => 0
                )
        )
);

$response = Kvs::getInstance()->request('Preset', ['json' => $preset_data]);

echo $response->getBody();
```

更新模板

- 接口名

UpdatePreset

- 调用方式

```
<?php
require('vendor/autoload.php');
use Ksyun\Service\Kvs;

// 拼凑模板数组
$preset_data = array(
        'Preset' => 'xxx',
        'Description' => 'xxx',
        'PresetType' => 'xxx',
        'Param' => array(
                'f' => 'xxx',
                'VIDEO' => array(
                        'vr' => '13',
                        'vb' => '780000',
                        'vcodec' => 'h264',
                        'width' => 500,
                        'height' => 600
                ),
                'AUDIO' => array(
                        'ar' => '44100',
                        'ab' => '64k',
                        'acodec' => 'aac',
                        'an' => 0
                )
        )
);

$response = Kvs::getInstance()->request('UpdatePreset', ['json' => $preset_data]);

echo $response->getBody();
```

删除模板

- 接口名

DelPreset

- 调用方式

```
<?php
require('vendor/autoload.php');
use Ksyun\Service\Kvs;

$response = Kvs::getInstance()->request('DelPreset', ['query' => ['Preset' => 'xxx']]);

echo $response->getBody();
```

查询模板列表

- 接口名

GetPresetList

- 调用方式

```
<?php
require('vendor/autoload.php');
use Ksyun\Service\Kvs;

$response = Kvs::getInstance()->request('GetPresetList', ['query' => ['PresetType' => 'xxx', 'Presets' => 'xxx']]);

echo $response->getBody();
```

查询模板详情

- 接口名

GetPresetDetail

- 调用方式

```
<?php
require('vendor/autoload.php');
use Ksyun\Service\Kvs;

$response = Kvs::getInstance()->request('GetPresetDetail', ['query' => ['Preset' => 'xxx']]);

echo $response->getBody();
```

创建任务

- 接口名

CreateTask

- 调用方式

```
<?php
require('vendor/autoload.php');
use Ksyun\Service\Kvs;

// 拼凑参数数组
$task_data = array(
    'Preset' => 'xxx',
    'srcInfo' => array(
        array(
            'path' => 'xxx',
            'index' => 0,
            'type' => 'video',
        )
    ),
    'dstBucket' => 'xxx',
    'dstDir' => '',
    'dstObjectKey' => 'xxx',
    'dstAcl' => 'public-read',
    'isTop' => 0,
    'cbUrl' => '',
    'cbMethod' => '',
    'extParam' => ''
);

$response = Kvs::getInstance()->request('CreateTask', ['json' => $task_data]);

echo $response->getBody();
```

置顶任务

- 接口名

TopTaskByTaskID

- 调用方式

```
<?php
require('vendor/autoload.php');
```

```
use Ksyun\Service\Kvs;

$response = Kvs::getInstance()->request('TopTaskByTaskID', ['query' => ['TaskID' => 'xxx']]);

echo $response->getBody();
```

删除任务

- 接口名

DelTaskByTaskID

- 调用方式

```
<?php
require('vendor/autoload.php');
use Ksyun\Service\Kvs;

$response = Kvs::getInstance()->request('DelTaskByTaskID', ['query' => ['TaskID' => 'xxx']]);

echo $response->getBody();
```

查询任务列表

- 接口名

GetTaskList

- 调用方式

```
<?php
require('vendor/autoload.php');
use Ksyun\Service\Kvs;

$response = $response = Kvs::getInstance()->request('GetTaskList', ['query' => ['StartDate' => '20160815', 'EndDate' => '20160816'
, 'Marker' => 0, 'Limit' => 2]]);

echo $response->getBody();
```

查询任务详情

- 接口名

GetTaskByTaskID

- 调用方式

```
<?php
require('vendor/autoload.php');
use Ksyun\Service\Kvs;

$response = Kvs::getInstance()->request('GetTaskByTaskID', ['query' => ['TaskID' => 'xxx']]);

echo $response->getBody();
```

查询任务META列表

- 接口名

GetTaskMetaInfo

- 调用方式

```
<?php
require('vendor/autoload.php');
use Ksyun\Service\Kvs;

// 通过任务ID查询
$response = Kvs::getInstance()->request('GetTaskMetaInfo', ['query' => ['TaskID' => 'xxx']]);

// 查询列表
$response = Kvs::getInstance()->request('GetTaskMetaInfo', ['query' => ['StartDate' => '20160929', 'EndDate' => '20160930', 'Mark
er' => 0, 'Limit' => 1]]);

echo $response->getBody();
```

同步获取META信息接口

- 接口名

FetchMetaInfo

- 调用方式

```
<?php
require('vendor/autoload.php');
use Ksyun\Service\Kvs;

$response = Kvs::getInstance()->request('FetchMetaInfo', ['json' => ['SrcPath' => '/{bucket}/xxx.flv']]);

echo $response->getBody();
```

SDK for Python 使用指南

安装Python SDK

1.git 安装

```
git clone https://github.com/KscSDK/ksc-sdk-python.git
cd ksc-sdk-python
python setup.py install
```

2.pip 安装

```
pip install ksc-sdk-python
```

3.通过文件配置及管理密钥

- 所在位置: '/etc/kscore.cfg' 或 './.kscore.cfg'
- 注意: 使用相对路径时，需与运行目录保持一致
- ks_access_key_id和ks_secret_access_key是金山云控制台身份与管理里面生成密钥对

```
[Credentials]
ks_access_key_id=xxxxxxxxxxxxxxxxxxx
ks_secret_access_key=xxxxxxxxxxxxxxxxxxxx
```

4.创建一个session （初始化）

```
from kscore.kvs import getKvsClient

第一种:
client = getKvsClient("kvs", "cn-beijing-6",use_ssl=False)
参数说明:
```

```
    service_name:    服务名，填写"kvs"
    region_name:    区域名，填写"cn-beijing-6"
    use_ssl:        是否https访问，填写False
```

第二种：
没有配置kscore.cfg调用方式
ks_access_key_id='xxxxxxxxxxxxxxxxxxx'
ks_secret_access_key='xxxxxxxxxxxxxxxxxxx'
参数：服务service_name,大区region_name
client = getKvsClient("kvs", "cn-beijing-6",use_ssl=False,ks_access_key_id=ks_access_key_id,ks_secret_access_key=ks_secret_access_key)

5.运行环境

适用于2.6、2.7、3.3、3.4的Python版本

注： 接口的传递的参数及返回值请参考《视频云离线转码接入文档》API接口说明

创建模板

- 接口名

Preset

- 调用方式

```
param = {
    "Preset": "xxxx",
    "Description": "xxxx",
    "PresetType": "xxxx",
    "Param":{
        "f": "xxx",
        "AUDIO": {
            "ab": xxx,
            "ar": xxx,
            "acodec": "xxxx",
            "an": xxx
        },
        "VIDEO": {
            "vr": xxx,
            "vb": "xxxx",
            "vcodec": "xxxx",
            "width": xxx,
            "height": xxx,
            "as": xxx,
            "rotate": "xxxx",
            "vn": xxx
        }
    }
}
注：
1.param必须是json格式数据
res = client.Preset(param)
```

更新模板

- 接口名

UpdatePreset

- 调用方式

```
param = {
    "Preset": "xxxx",
    "Description": "xxxx",
    "PresetType": "xxxx",
    "Param":{
        "f": "xxx",
        "AUDIO": {
            "ab": xxx,
            "ar": xxx,
            "acodec": "xxxx",
            "an": xxx
        },
        "VIDEO": {
            "vr": xxx,
            "vb": "xxxx",
            "vcodec": "xxxx",
```

```
            "width": xxx,
            "height": xxx,
            "as": xxx,
            "rotate": "xxxx",
            "vn": xxx
        }
    }
}
```

注：
1.param必须是json格式数据
res = client.UpdatePreset(param)

查询模板列表

- 接口名

GetPresetList

- 调用方式

```
res = client.GetPresetList(WithDetail=0,PresetType="xxx",Presets="xxx")
```

查询模板详情

- 接口名

GetPresetDetail

- 调用方式

```
res = client.GetPresetDetail(Preset="xxx")
```

删除模板

- 接口名

DelPreset

- 调用方式

```
res = client.DelPreset(Preset="xxx")
```

创建任务

- 接口名

CreateTask

- 调用方式

```
task = {
    "DstDir": "",
    "DstObjectKey": "xxx",
    "DstBucket": "xxx",
    "DstAcl": "public-read",
    "Preset": "xxx",
    "SrcInfo": [
        {
            "path": "xxx",
            "type": "video",
            "index": 0
        }
    ],
    "CbMethod": "xxx",
    "CbUrl": "xxx"
}
```

注：
1.task必须是json格式数据
res = client.CreateTask(task)

置顶任务

- 接口名

TopTaskByTaskID

- 调用方式

```
res = client.TopTaskByTaskID(TaskID = taskid)
```

删除任务

- 接口名

DelTaskByTaskID

- 调用方式

```
res = client.DelTaskByTaskID(TaskID = taskid)
```

查询任务列表

- 接口名

GetTaskList

- 调用方式

```
res = client.GetTaskList(StartDate=20170100,EndDate=20170112,Marker=0,Limit=50)
```

查询任务详情

- 接口名

GetTaskByTaskID

- 调用方式

```
res = client.GetTaskByTaskID(TaskID = taskid)
```

查询任务META列表

- 接口名

GetTaskMetaInfo

- 调用方式

```
res = client.GetTaskMetaInfo(StartDate=20170100,EndDate=20170112,Marker=0,Limit=50)
```

SDK for Java 使用指南

查询任务队列信息

更新任务队列状态

查询任务META信息

安装Java SDK

1.git 安装

```
git clone https://github.com/KscSDK/ksc-sdk-java.git
mvn clean install
```

2.工程使用sdk

建议使用Maven构建自己的项目，导入ksc-sdk-java,添加ksc-sdk-java-kvs的依赖。

新建maven项目

进入到pom文件下，进行如下配置

```
<dependencies>
    <dependency>
        <groupId>com.ksyun</groupId>
        <artifactId>ksc-sdk-java-kvs</artifactId>
        <version>0.3.8</version>
    </dependency>
</dependencies>
```

3.通过文件配置及管理密钥

本地文件配置:

```
 ~/.aws/credentials on Linux, OS X or unix
C:\Users\USERNAME\.aws\credentials on Windows
```

该文件包含下述内容：

```
[default]
aws_access_key_id = your_access_key_id
aws_secret_access_key = your_secret_access_key
```

如不通过本地文件读入ak和sk信息，则需要在执行的类文件中增加以下代码：

```
AWSCredentials credentials = new BasicAWSCredentials(aws_access_key_id, aws_secret_access_key);
```

SDK介绍和使用

在调用接口前，先初始化 KSCKVSJsonClient，如下用例：

```
KSCKVSJsonClient ksc = new KSCKVSJsonClient();
//可传入AWSCredentials参数 如:KSCKVSJsonClient ksc = new KSCKVSJsonClient(credentials);
ksc.setEndpoint("http://kvs.cn-beijing-6.api.ksyun.com/");
字符集：utf-8
```

注：接口的传递的参数及返回值请参考点播转码API

创建模板

- 接口名

Preset

- 调用方式

```
PresetRequest presetRequest = new PresetRequest();
String data = PresetSet("xxx");
presetRequest.setData(data);
KvsErrResult presetResult = ksc.Preset(presetRequest);
//请求内容生成接口
```

```
private static String PresetSet(String preset) throws JSONException {
    String presettype = "xxx";
    JSONObject data = new JSONObject();
    JSONObject param = new JSONObject();
    JSONObject video = new JSONObject();
    JSONObject audio = new JSONObject();
    JSONArray logos = new JSONArray();
    data.put("Preset", "xxx");
    data.put("PresetType", "xxx");
    data.put("Description", "xxx");
    video.put("vr", "13");
    video.put("vb", "780000");
    video.put("vcodec", "h264");
    for (int i = 0; i < 4; i++) {
        JSONArray logo = new JSONArray();
        for (int j = 0; j < 2; j++) {
            JSONObject tmp = new JSONObject();
            tmp.put("url", String.valueOf(i) + ":" + String.valueOf(j));
            tmp.put("ss", i + j);
            logo.put(tmp);
        }
        logos.put(logo);
    }
    audio.put("ar", "xxx");
    audio.put("ab", "xxx");
    audio.put("acodec", "aac");
    audio.put("an", 0);
    param.put("f", "xxx");
    param.put("VIDEO", video);
    param.put("AUDIO", audio);
    param.put("LOGOS", logos);
    data.put("Param", param);
    return data.toString();
}
```

更新模板

- 接口名

UpdatePreset

- 调用方式

```
UpdatePresetRequest presetRequest = new UpdatePresetRequest();
String data = PresetSet("xxx");
presetRequest.setData(data1);
KvsErrResult presetResult = ksc.UpdatePreset(presetRequest);
```

删除模板

- 接口名

DelPreset

- 调用方式

```
DeletePresetRequest deletePresetRequest = new DeletePresetRequest();
deletePresetRequest.setPreset("xxx");
KvsErrResult deletePresetResult = ksc.DelPreset(deletePresetRequest);
```

查询模板列表

- 接口名

GetPresetList

- 调用方式

```
GetListRequest gitlistrequest = new GetListRequest();
gitlistrequest.setWithDetail(1);
GetPresetListResult getpresetlistResult = ksc.GetPresetList(gitlistrequest);
```

查询模板详情

- 接口名

GetPresetDetail

- 调用方式

```
GetPresetDetailRequest getPresetDetailRequest = new GetPresetDetailRequest();
getPresetDetailRequest.setPreset("xxx");
GetPresetDetailResult getPresetDetailResult = ksc.GetPresetDetail(getPresetDetailRequest);
```

创建任务

- 接口名

CreateTask

- 调用方式

```
CreateTaskRequest createTaskRequest = new CreateTaskRequest();
String data2 = setTask("xxx", "xxx", "xxx", "xxx");
createTaskRequest.setData(data2);
CreateTasklResult createTasklResult = ksc.CreateTask(createTaskRequest);
//请求内容生成接口
private static String setTask(String preset, String dst_bucket, String dst_object_key, String src_object_key)
        throws JSONException {
    JSONObject data = new JSONObject();
    data.put("Preset", preset);
    data.put("SrcInfo", TaskSrcInfo(dst_bucket, src_object_key));
    data.put("DstBucket", dst_bucket);
    data.put("DstObjectKey", dst_object_key);
    data.put("DstDir", "");
    data.put("IsTop", 0);
    data.put("DstAcl", "public-read");
    data.put("CbUrl", "");
    data.put("CbMethod", "POST");
    data.put("ExtParam", "");
    return data.toString();
}
```

置顶任务

- 接口名

TopTaskByTaskID

- 调用方式

```
TopTaskByTaskIDRequest topTaskByTaskIDRequest = new TopTaskByTaskIDRequest();
topTaskByTaskIDRequest.setTaskID("xxx");
KvsErrResult TopTaskByTaskIDResult = ksc.TopTaskByTaskID(topTaskByTaskIDRequest);
```

删除任务

- 接口名

DelTaskByTaskID

- 调用方式

```
DelTaskByTaskIDRequest delTaskByTaskIDRequest = new DelTaskByTaskIDRequest();
delTaskByTaskIDRequest.setTaskID("xxx");
KvsErrResult DelTaskByTaskIDResult = ksc.DelTaskByTaskID(delTaskByTaskIDRequest);
```

查询任务列表

- 接口名

GetTaskList

- 调用方式

```
GetTaskListRequest getTaskListRequest = new GetTaskListRequest();
getTaskListRequest.setStartDate(20200202);
getTaskListRequest.setEndDate(20200220);
getTaskListRequest.setTaskStatus("succ");
getTaskListRequest.setErrorCode("3255");
GetTaskListResult getTaskListResult = ksc.GetTaskList(getTaskListRequest);
```

查询任务详情

- 接口名

GetTaskByTaskID

- 调用方式

```
GetTaskByTaskIDRequest getTaskByTaskIDRequest = new GetTaskByTaskIDRequest();
getTaskByTaskIDRequest.setTaskID("xxx");
GetTaskByTaskIDResult getTaskByTaskIDResult = ksc.GetTaskByTaskID(getTaskByTaskIDRequest);
```

查询任务队列信息

- 接口名

QueryPipeline

- 调用方式

```
QueryPipelineRequest queryPipelineRequest = new QueryPipelineRequest();
queryPipelineRequest.setPipelineName("usual");
QueryPipelineResult queryPipelineResult = ksc.QueryPipeline(queryPipelineRequest);
System.out.println(queryPipelineResult);
```

更新任务队列状态

- 接口名

UpdatePipeline

- 调用方式

```
UpdatePipelineRequest updatePipelineRequest = new UpdatePipelineRequest();
updatePipelineRequest.setData(setPipeline("usual"));
KvsErrResult updatePipelineResult = ksc.UpdatePipeline(updatePipelineRequest);
System.out.println(updatePipelineResult); //设置更新任务队列请求参数 private static String setPipeline(String PipelineName) { JSONO
bject data = new JSONObject(); data.put("PipelineName", PipelineName); data.put("State", "Active"); return data.toString(); }
```

查询任务META列表

- 接口名

GetTaskMetaInfo

- 调用方式

```
GetTaskMetaRequest getTaskMetaInfoRequest = new GetTaskMetaRequest();
getTaskMetaInfoRequest.setTaskID("xxx"); //需要是avinfo模板类型的任务
getTaskMetaInfoRequest.setStartDate(20200202);
GetTaskMetaResult getTaskMetaResult = ksc.GetTaskMetaInfo(getTaskMetaInfoRequest);
System.out.println(getTaskMetaResult);
```

同步获取META信息接口

- 接口名

FetchMetaInfo

- 调用方式

```
JSONObject data = new JSONObject();
    data.put("SrcPath","bkt-bj-youpin-video-kscn/daren/b911aecedd47b97b8d3c1214c758d62e.mp4");
    System.out.println(data);
    FetchMetaInfoRequest fetchMetaInfoRequest = new FetchMetaInfoRequest();
    fetchMetaInfoRequest.setData(data.toString());
    FetchMetaInfoResult fetchMetaInfoResult = ksc.FetchMetaInfo(fetchMetaInfoRequest);
    System.out.println(fetchMetaInfoResult.getMetaInfo());
```