智能语音V2.X 语音合成RESTful API使用文档

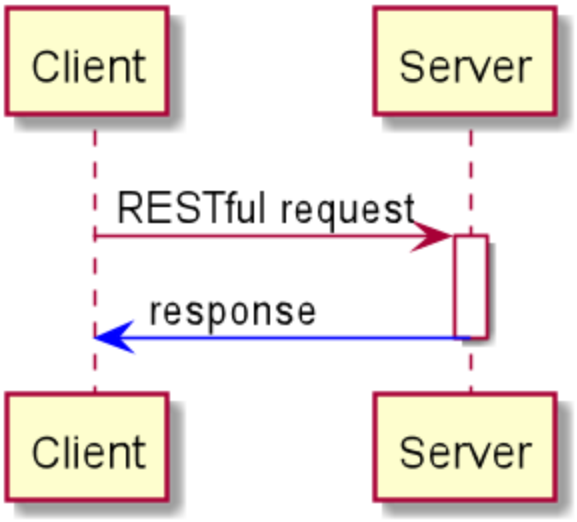
功能介绍

语音合成RESTful API支持HTTP **GET**和**POST**两种方法的请求，将待合成的文本上传到服务端，服务端返回文本的语音合成结果，开发者需要保证在语音合成结果返回之前连接不被中断。

- 支持设置合成音频的格式：pcm，wav，mp3；
- 支持设置不同的发音人：通用xiaoyun（支持采样率8KHz/16KHz）、xiaogang（支持采样率8KHz/16KHz），其他发音人请根据模型选择；
- 支持设置语速、语调、音量。

交互流程

客户端向服务端发送带有文本内容的HTTP GET方法 或 POST方法的请求，服务端返回带有合成语音数据的HTTP响应。



请求参数

语音合成需要设置的请求参数如下表所示。如果使用HTTP GET方法的请求，需要将这些参数设置到HTTP的URL请求参数中；如果使用HTTP POST方法的请求，需要将这些参数设置到HTTP的请求体（Body）中。

名称	类型	是否必需	描述
appkey	String	是	应用appkey， 专有云为 default
text	String	是	待合成的文本， 需要为UTF-8编码。 使用GET方法， 需要再采用RFC 3986规范进行urlencode编码，比如加号 `+` 编码为 `%2B`； 使用POST方法不需要urlencode编码。

token	String	否	服务鉴权Token， 专有云为 default。 若不设置token参数， 需要在HTTP Headers中设置”X- NLS- Token”字段来指定Tok en。
format	String	否	音频编码格式， 支持的格式：pcm、 wav、mp3， 默认是pcm
sample_rate	Integer	否	音频采样率， 支持16000Hz、 8000Hz， 默认是16000Hz
voice	String	否	发音人， 默认是xiaoyun， 支持发音人： xiaoyun、xiaogang、 ruoxi
volume	Integer	否	音量，范围是0~100， 默认50
speech_rate	Integer	否	语速， 范围是-500~500， 默认是0
pitch_rate	Integer	否	语调， 范围是-500~500， 可选，默认是0

GET方法上传文本

一个完整的语音合成RESTful API GET方法的请求包含以下要素：

1.URL

阿里巴巴智能语音

协议	URL	方法
HTTP	http://gateway所在IP:8101/stream/v1/tts	GET

2.请求参数

见上述请求参数表格。

如上URL和请求参数组成的完整请求链接如下所示，在浏览器中打开该链接可直接获取语音合成的结果：

```
# 在浏览器中打开该链接，可直接获取语音合成结果。
# text的内容为"今天是周一，天气挺好的。"
http://gateway所在IP:8101/stream/v1/tts?appkey=default&token=default&text=%E4%BB%8A%E5%A4%A9%E6%98%AF%E5%91%A8%E4%B8%80%EF%BC%8C%E5%A4%A9%E6%B0%94%E6%8C%BA%E5%A5%BD%E7%9A%84%E3%80%82&format=wav&sample_rate=16000
```

3.HTTP GET 请求头部

名称	类型	是否必需	描述
X-NLS-Token	String	否	服务鉴权Token， 专有云为 default。 若请求参数中没有设置**token**参数， 则需要在这里设置该字段。

注意：

- 服务鉴权Token参数既可以在请求参数token中设置，也可以在HTTP Headers的”X-NLS-Token”字段设置，**推荐使用请求参数token**。
- 参数**text**必须采用**UTF-8**编码，在采用RFC 3986规范进行urlencode编码，比如加号 `+` 编码为 `%2B`，星号 `*` 编码为 `%2A`，`%7E` 编码为 `~`。

POST方法上传文本

一个完整的语音合成RESTful API POST请求包含以下要素：

1.URL

协议	URL	方法
HTTP	http://gateway所在IP:8101/stream/v1/tts	POST

2.HTTP POST 请求头部

名称	类型	是否必需	描述
X-NLS-Token	String	否	服务鉴权Token，专有云为 default。若Body请求参数中没有设置**token**参数，则需要在这里设置该字段。
Content-Type	String	是	必须为“application/json”，表明HTTP Body的内容为JSON格式字符串
Content-Length	long	否	HTTP Body中内容的长度

3.HTTP POST 请求体

HTTP POST请求体传入的是请求参数组成的JSON格式的字符串，因此在HTTP POST请求头部中的Content-Type必须设置为”application/json”。示例如下：

```
{
  "appkey":"default",
  "text":"今天是周一，天气挺好的。",
  "token":"default",
  "format":"wav"
}
```

注意：

- 服务鉴权Token参数既可以在Body中的请求参数token中设置，也可以在HTTP Headers的”X-NLS-Token”字段设置，**推荐使用Body参数token**。
- 使用POST方法的请求，Body中的请求参数**text**必须采用**UTF-8**编码，但是不进行urlencode编码，注意与GET方法请求的区分。

响应结果

使用HTTP GET方法和使用HTTP POST方法请求的响应是相同的，响应的结果都包含在HTTP的Body中。响应结果的成功或失败通过HTTP Header的**Content-Type**字段来区分：

- 成功响应
  - HTTP Headers的Content-Type字段内容为**`audio/mpeg`**，表示合成成功，合成的语音数据在Body中。
  - HTTP Header的X-NLS-RequestId字段内容为请求任务的task_id，方便调试排查。
  - Body内容为合成音频的二进制数据。
- 失败响应
  - HTTP Headers没有Content-Type字段，或者Content-Type字段内容为**`application/json`**，表示合成失败，错误信息在Body中。
  - HTTP Header的X-NLS-RequestId字段内容为请求任务的task_id，方便调试排查。
  - Body内容为错误信息，JSON格式的字符串。如下所示：

```
{
    "task_id":"8f95d0b9b6e948bc98e8d0ce64b0cf57",
    "result":"",
    "status":40000000,
    "message":"Gateway:CLIENT_ERROR:in post data, json format illegal"
}
```

失败响应时的错误信息字段如下表所示：

名称	类型	描述
task_id	String	32位请求任务id
result	String	服务结果
status	Integer	服务状态码
message	String	服务状态描述

服务状态码

服务状态码	服务状态描述	解决办法
20000000	请求成功	
40000000	默认的客户端错误码	查看错误消息或提交工单
40000001	身份认证失败	检查使用的令牌是否正确， 是否过期
40000002	无效的消息	检查发送的消息是否符合要求
40000003	无效的参数	检查参数值设置是否合理
40000004	空闲超时	确认是否长时间没有发送数据掉 服务端
40000005	请求数量过多	检查是否超过了并发连接数或者 每秒钟请求数
50000000	默认的服务端错误	如果偶现可以忽略， 重复出现请提交工单
50000001	内部GRPC调用错误	如果偶现可以忽略， 重复出现请提交工单

示例代码

以下各语言的Demo中使用的appkey和token均为default。
链接地址为 http://gateway所在IP:8101/stream/v1/tts， host为"gateway所在IP:8101"

Java Demo

依赖：

```
<dependency>
    <groupId>com.squareup.okhttp3</groupId>
    <artifactId>okhttp</artifactId>
    <version>3.9.1</version>
</dependency>
<!-- http://mvnrepository.com/artifact/com.alibaba/fastjson -->
```

```
<dependency>
    <groupId>com.alibaba</groupId>
    <artifactId>fastjson</artifactId>
    <version>1.2.42</version>
</dependency>
```

示例代码：

```
import com.alibaba.fastjson.JSONObject;
import okhttp3.MediaType;
import okhttp3.OkHttpClient;
import okhttp3.Request;
import okhttp3.RequestBody;
import okhttp3.Response;

import java.io.File;
import java.io.FileOutputStream;
import java.io.UnsupportedEncodingException;
import java.net.URLEncoder;

public class SpeechSynthesizerRESTfulDemo {
    private String accessToken;
    private String appkey;
    private String url;

    public SpeechSynthesizerRESTfulDemo(String appkey, String token, String ur
l) {
        this.appkey = appkey;
        this.accessToken = token;
        this.url = url;
    }

    /**
     * HTTP GET 请求
     */
    public void processGETRequet(String text, String audioSaveFile, String for
mat, int sampleRate) {
        /**
         * 设置HTTP GET请求
         * 1.使用HTTP协议
         * 2.语音识别服务域名：gateway所在IP:8101
         * 3.语音识别接口请求路径：/stream/v1/tts
```

```
         * 4.设置必须请求参数：appkey、token、text、format、sample_rate
         * 5.设置可选请求参数：voice、volume、speech_rate、pitch_rate
         */
        url = url + "?appkey=" + appkey;
        url = url + "&token=" + accessToken;
        url = url + "&text=" + text;
        url = url + "&format=" + format;
        url = url + "&sample_rate=" + String.valueOf(sampleRate);

        // voice 发音人，可选，默认是xiaoyun
        // url = url + "&voice=" + "xiaoyun";
        // volume 音量，范围是0~100，可选，默认50
        // url = url + "&volume=" + String.valueOf(50);
        // speech_rate 语速，范围是-500~500，可选，默认是0
        // url = url + "&speech_rate=" + String.valueOf(0);
        // pitch_rate 语调，范围是-500~500，可选，默认是0
        // url = url + "&pitch_rate=" + String.valueOf(0);

        System.out.println("URL: " + url);

        /**
         * 发送HTTP GET请求，处理服务端的响应
         */
        Request request = new Request.Builder()
                .url(this.url)
                .get()
                .build();

        try {
            OkHttpClient client = new OkHttpClient();
            Response response = client.newCall(request).execute();
            String contentType = response.header("Content-Type");
            if ("audio/mpeg".equals(contentType)) {
                File f = new File(audioSaveFile);
                FileOutputStream fout = new FileOutputStream(f);
                fout.write(response.body().bytes());
                fout.close();
                System.out.println("The GET request succeed!");
            }
            else {
                // ContentType 为 null 或者为 "application/json"
                String errorMessage = response.body().string();
                System.out.println("The GET request failed: " + errorMessag
e);
            }
```

```
            response.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    /**
     * HTTP POST 请求
     */
    public void processPOSTRequest(String text, String audioSaveFile, String f
ormat, int sampleRate) {
        /**
         * 设置HTTP POST请求
         * 1.使用HTTP协议
         * 2.语音合成服务域名：gateway所在IP:8101
         * 3.语音合成接口请求路径：/stream/v1/tts
         * 4.设置必须请求参数：appkey、token、text、format、sample_rate
         * 5.设置可选请求参数：voice、volume、speech_rate、pitch_rate
         */
        JSONObject taskObject = new JSONObject();
        taskObject.put("appkey", appkey);
        taskObject.put("token", accessToken);
        taskObject.put("text", text);
        taskObject.put("format", format);
        taskObject.put("sample_rate", sampleRate);

        // voice 发音人，可选，默认是xiaoyun
        // taskObject.put("voice", "xiaoyun");
        // volume 音量，范围是0~100，可选，默认50
        // taskObject.put("volume", 50);
        // speech_rate 语速，范围是-500~500，可选，默认是0
        // taskObject.put("speech_rate", 0);
        // pitch_rate 语调，范围是-500~500，可选，默认是0
        // taskObject.put("pitch_rate", 0);

        String bodyContent = taskObject.toJSONString();
        System.out.println("POST Body Content: " + bodyContent);
        RequestBody reqBody = RequestBody.create(MediaType.parse("application/
json"), bodyContent);
        Request request = new Request.Builder()
                .url(this.url)
                .header("Content-Type", "application/json")
                .post(reqBody)
                .build();
```

```
        try {
            OkHttpClient client = new OkHttpClient();
            Response response = client.newCall(request).execute();
            String contentType = response.header("Content-Type");
            if ("audio/mpeg".equals(contentType)) {
                File f = new File(audioSaveFile);
                FileOutputStream fout = new FileOutputStream(f);
                fout.write(response.body().bytes());
                fout.close();
                System.out.println("The POST request succeed!");
            }
            else {
                // ContentType 为 null 或者为 "application/json"
                String errorMessage = response.body().string();
                System.out.println("The POST request failed: " + errorMessag
e);
            }

            response.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    public static void main(String[] args) {
        if (args.length < 1) {
            System.err.println("SpeechSynthesizerRESTfulDemo need params: <ga
teway所在IP>");
            System.exit(-1);
        }

        String ip = args[0];
        String token = "default";
        String appkey = "default";
        String port = "8101";

        String url = "http://" + ip + ":" + port + "/stream/v1/tts";

        SpeechSynthesizerRESTfulDemo demo = new SpeechSynthesizerRESTfulDemo
(appkey, token, url);

        String text = "今天是周一，天气挺好的。";
        // 采用RFC 3986规范进行urlencode编码
        String textUrlEncode = text;
```

```
        try {
            textUrlEncode = URLEncoder.encode(textUrlEncode, "UTF-8")
                    .replace("+", "%20")
                    .replace("*", "%2A")
                    .replace("%7E", "~");
        } catch (UnsupportedEncodingException e) {
            e.printStackTrace();
        }
        System.out.println(textUrlEncode);

        String audioSaveFile = "syAudio.wav";
        String format = "wav";
        int sampleRate = 16000;

        demo.processGETRequet(textUrlEncode, audioSaveFile, format, sampleRat
e);

        //demo.processPOSTRequest(text, audioSaveFile, format, sampleRate);
    }
}
```

C++ Demo

C++ Demo使用了第三方函数库curl处理HTTP的请求和响应，使用jsoncpp处理POST请求Body内容JSON格式字符串的设置。Demo相关文件在nls-cpp-example/nls-cpp-restful目录下：

目录说明：

- CMakeLists.txt demo工程的CMakeList文件
- demo

文件名	描述
restfulTtsDemo.cpp	语音合成RESTful API Demo

- include

目录名	描述
curl	curl库头文件目录

json	jsoncpp库头文件目录

- lib包含curl、jsoncpp动态库。根据平台不同，可以选择linux版本（glibc:2.5及以上，Gcc4，Gcc5）、windows版本（VS2013、VS2015）。
- readme.txt 说明
- release.log 更新记录
- version 版本号
- build.sh demo编译脚本

注意：

1. Linux环境下，运行环境最低要求：Glibc 2.5及以上， Gcc4、Gcc5。
2. token值有效期默认24小时，超时请重新获取，否则会导致请求识别。
3. Windows下需要用户自己搭建demo工程。

运行编译:

```
1. 请确认本地系统以安装Cmake，最低版本2.4
2. cd path/to/sdk/lib
3. tar -zxvpf linux.tar.gz
4. cd path/to/sdk
5. 执行[./build.sh]编译demo
6. 编译完毕，进入demo目录，执行[./restfulTtsDemo <gateway所在IP>]

如果不支持cmake，可以尝试手动编译：
1: cd path/to/sdk/lib
2: tar -zxvpf linux.tar.gz
3: cd path/to/sdk/demo
4: g++ -o restfulTtsDemo restfulTtsDemo.cpp -I path/to/sdk/include -L path/to/
sdk/lib/linux -ljsoncpp -lssl -lcrypto -lcurl -D_GLIBCXX_USE_CXX11_ABI=0
5: export LD_LIBRARY_PATH=path/to/sdk/lib/linux/
6: ./restfulTtsDemo <gateway所在IP>
Windows平台需要用户自己搭建工程。
```

示例代码：

```
#ifdef _WIN32
#include <Windows.h>
#endif
```

```
#include <iostream>
#include <string>
#include <map>
#include <fstream>
#include <sstream>

#include "curl/curl.h"
#include "json/json.h"

using namespace std;

#ifdef _WIN32
string GBKToUTF8(const string &strGBK) {
    string strOutUTF8 = "";
    WCHAR * str1;

    int n = MultiByteToWideChar(CP_ACP, 0, strGBK.c_str(), -1, NULL, 0);

    str1 = new WCHAR[n];

    MultiByteToWideChar(CP_ACP, 0, strGBK.c_str(), -1, str1, n);

    n = WideCharToMultiByte(CP_UTF8, 0, str1, -1, NULL, 0, NULL, NULL);

    char * str2 = new char[n];
    WideCharToMultiByte(CP_UTF8, 0, str1, -1, str2, n, NULL, NULL);
    strOutUTF8 = str2;

    delete[] str1;
    str1 = NULL;
    delete[] str2;
    str2 = NULL;

    return strOutUTF8;
}
#endif


void stringReplace(string& src, const string& s1, const string& s2) {
    string::size_type pos = 0;
    while ((pos = src.find(s1, pos)) != string::npos) {
        src.replace(pos, s1.length(), s2);
        pos += s2.length();
    }
```

```
}

string urlEncode(const string& src) {
    CURL* curl = curl_easy_init();
    char* output = curl_easy_escape(curl, src.c_str(), src.size());
    string result(output);

    curl_free(output);
    curl_easy_cleanup(curl);

    return result;
}

size_t responseHeadersCallback(void* ptr, size_t size, size_t nmemb, void* use
rdata)
{
    map<string, string> *headers = (map<string, string>*)userdata;
    string line((char*)ptr);
    string::size_type pos = line.find(':');
    if (pos != line.npos)
    {
        string name = line.substr(0, pos);
        string value = line.substr(pos + 2);
        size_t p = 0;
        if ((p = value.rfind('\r')) != value.npos) {
            value = value.substr(0, p);
        }

        headers->insert(make_pair(name, value));
    }
    return size * nmemb;
}

size_t responseBodyCallback(void* ptr, size_t size, size_t nmemb, void* userDa
ta) {

    size_t len = size * nmemb;
    char* pBuf = (char*)ptr;
    string* bodyContent = (string*)userData;
    (*bodyContent).append(string(pBuf, pBuf + len));

    return len;
}

int processGETRequest(string url, string appKey, string token, string text,
```

```
string audioSaveFile, string format, int sampleRate) {
CURL* curl = NULL;
CURLcode res;

curl = curl_easy_init();
if (curl == NULL) {
    return -1;
}

/**
* 设置HTTP URL请求参数
*/
ostringstream oss;
oss << url;
oss << "?appkey=" << appKey;
oss << "&token=" << token;
oss << "&text=" << text;
oss << "&format=" << format;
oss << "&sample_rate=" << sampleRate;

// voice 发音人，可选，默认是xiaoyun
// oss << "&voice=" << "xiaoyun";
// volume 音量，范围是0~100，可选，默认50
// oss << "&volume=" << 50;
// speech_rate 语速，范围是-500~500，可选，默认是0
// oss << "&speech_rate=" << 0;
// pitch_rate 语调，范围是-500~500，可选，默认是0
// oss << "&pitch_rate=" << 0;

string request = oss.str();
cout << request << endl;

curl_easy_setopt(curl, CURLOPT_URL, request.c_str());

/**
* 设置获取响应的HTTP Headers回调函数
*/
map<string, string> responseHeaders;
curl_easy_setopt(curl, CURLOPT_HEADERFUNCTION, responseHeadersCallback);
curl_easy_setopt(curl, CURLOPT_HEADERDATA, &responseHeaders);

/**
* 设置获取响应的HTTP Body回调函数
*/
string bodyContent = "";
```

```
    curl_easy_setopt(curl, CURLOPT_WRITEFUNCTION, responseBodyCallback);
    curl_easy_setopt(curl, CURLOPT_WRITEDATA, &bodyContent);

    /**
    * 发送HTTP GET请求
    */
    res = curl_easy_perform(curl);

    /**
    * 释放资源
    */
    curl_easy_cleanup(curl);

    if (res != CURLE_OK) {
        cerr << "curl_easy_perform failed: " << curl_easy_strerror(res) << e
ndl;
        return -1;
    }

    /**
    * 处理服务端返回的响应
    */
    map<string, string>::iterator it = responseHeaders.find("Content-Type");
    if (it != responseHeaders.end() && it->second.compare("audio/mpeg") == 0)
{
        ofstream fs;
        fs.open(audioSaveFile.c_str(), ios::out | ios::binary);
        if (!fs.is_open()) {
            cout << "The audio save file can not open!";
            return -1;
        }
        fs.write(bodyContent.c_str(), bodyContent.size());
        fs.close();

        cout << "The GET request succeed!" << endl;
    }
    else {
        cout << "The GET request failed: " + bodyContent << endl;
        return -1;
    }

    return 0;
}

int processPOSTRequest(string url, string appKey, string token, string text,
```

```
    string audioSaveFile, string format, int sampleRate) {
    CURL* curl = NULL;
    CURLcode res;

    curl = curl_easy_init();
    if (curl == NULL) {
        return -1;
    }

    /**
    * 设置HTTP POST URL
    */
    curl_easy_setopt(curl, CURLOPT_URL, url.c_str());
    curl_easy_setopt(curl, CURLOPT_POST, 1L);

    /**
    * 设置HTTP POST请求头部
    */
    struct curl_slist* headers = NULL;
    // Content-Type
    headers = curl_slist_append(headers, "Content-Type:application/json");

    curl_easy_setopt(curl, CURLOPT_HTTPHEADER, headers);

    /**
    * 设置HTTP POST请求体
    */
    Json::Value root;
    Json::FastWriter writer;
    root["appkey"] = appKey;
    root["token"] = token;
    root["text"] = text;
    root["format"] = format;
    root["sample_rate"] = sampleRate;

    // voice 发音人，可选，默认是xiaoyun
    // root["voice"] = "xiaoyun";
    // volume 音量，范围是0~100，可选，默认50
    // root["volume"] = 50;
    // speech_rate 语速，范围是-500~500，可选，默认是0
    // root["speech_rate"] = 0;
    // pitch_rate 语调，范围是-500~500，可选，默认是0
    // root["pitch_rate"] = 0;

    string task = writer.write(root);
```

```
    cout << "POST request Body: " << task << endl;

    curl_easy_setopt(curl, CURLOPT_POSTFIELDS, task.c_str());
    curl_easy_setopt(curl, CURLOPT_POSTFIELDSIZE, task.length());

    /**
    * 设置获取响应的HTTP Headers回调函数
    */
    map<string, string> responseHeaders;
    curl_easy_setopt(curl, CURLOPT_HEADERFUNCTION, responseHeadersCallback);
    curl_easy_setopt(curl, CURLOPT_HEADERDATA, &responseHeaders);

    /**
    * 设置获取响应的HTTP Body回调函数
    */
    string bodyContent = "";
    curl_easy_setopt(curl, CURLOPT_WRITEFUNCTION, responseBodyCallback);
    curl_easy_setopt(curl, CURLOPT_WRITEDATA, &bodyContent);

    /**
    * 发送HTTP POST请求
    */
    res = curl_easy_perform(curl);

    /**
    * 释放资源
    */
    curl_slist_free_all(headers);
    curl_easy_cleanup(curl);

    if (res != CURLE_OK) {
        cerr << "curl_easy_perform failed: " << curl_easy_strerror(res) << e
ndl;
        return -1;
    }

    /**
    * 处理服务端返回的响应
    */
    map<string, string>::iterator it = responseHeaders.find("Content-Type");
    if (it != responseHeaders.end() && it->second.compare("audio/mpeg") == 0)
{
        ofstream fs;
        fs.open(audioSaveFile.c_str(), ios::out | ios::binary);
        if (!fs.is_open()) {
```

```
            cout << "The audio save file can not open!";
            return -1;
        }
        fs.write(bodyContent.c_str(), bodyContent.size());
        fs.close();

        cout << "The POST request succeed!" << endl;
    }
    else {
        cout << "The POST request failed: " + bodyContent << endl;
        return -1;
    }

    return 0;
}

int main(int argc, char* argv[]) {
    if (argc < 2) {
        cerr << "params is not valid. Usage: ./demo <gateway所在IP>" << endl;
        return -1;
    }

    string ip = argv[1];
    string token = "default";
    string appKey = "default";
    string port = "8101";

    string url = "http://" + ip + ":" + port + "/stream/v1/tts";

    string text = "今天是周一，天气挺好的。";
#ifdef _WIN32
    text = GBKToUTF8(text);
#endif

    string textUrlEncode = urlEncode(text);
    stringReplace(textUrlEncode, "+", "%20");
    stringReplace(textUrlEncode, "*", "%2A");
    stringReplace(textUrlEncode, "%7E", "~");

    string audioSaveFile = "syAudio.wav";
    string format = "wav";
    int sampleRate = 16000;

    // 全局只初始化一次
    curl_global_init(CURL_GLOBAL_ALL);
```

```
    processGETRequest(url, appKey, token, textUrlEncode, audioSaveFile, forma
t, sampleRate);

    // processPOSTRequest(url, appKey, token, text, audioSaveFile, format, sa
mpleRate);

    curl_global_cleanup();

    return 0;

}
```

Python Demo

注意：

1. Python 2.x请使用httplib模块；Python 3.x请使用http.client模块。
2. 采用RFC 3986规范进行urlencode编码，Python 2.x请使用urllib模块的urllib.quote，Python 3.x请使用urllib.parse模块的urllib.parse.quote_plus。

```
# -*- coding: UTF-8 -*-

# Python 2.x 引入httplib模块
# import httplib

# Python 3.x 引入http.client模块
import http.client

# Python 2.x 引入urllib模块
# import urllib

# Python 3.x 引入urllib.parse模块
import urllib.parse

import json


def processGETRequest(host, appKey, token, text, audioSaveFile, format, sampl
eRate) :

    url = 'http://' + host + '/stream/v1/tts'
```

```
# 设置URL请求参数
url = url + '?appkey=' + appKey
url = url + '&token=' + token
url = url + '&text=' + text
url = url + '&format=' + format
url = url + '&sample_rate=' + str(sampleRate)

# voice 发音人，可选，默认是xiaoyun
# url = url + '&voice=' + 'xiaoyun'
# volume 音量，范围是0~100，可选，默认50
# url = url + '&volume=' + str(50)
# speech_rate 语速，范围是-500~500，可选，默认是0
# url = url + '&speech_rate=' + str(0)
# pitch_rate 语调，范围是-500~500，可选，默认是0
# url = url + '&pitch_rate=' + str(0)

print(url)

# Python 2.x 请使用httplib
# conn = httplib.HTTPConnection(host)

# Python 3.x 请使用http.client
conn = http.client.HTTPConnection(host)

conn.request(method='GET', url=url)

# 处理服务端返回的响应
response = conn.getresponse()
print('Response status and response reason:')
print(response.status ,response.reason)

contentType = response.getheader('Content-Type')
print(contentType)

body = response.read()

if 'audio/mpeg' == contentType :
    with open(audioSaveFile, mode='wb') as f:
        f.write(body)

    print('The GET request succeed!')
else :
    print('The GET request failed: ' + str(body))
```

```
    conn.close()


def processPOSTRequest(host, appKey, token, text, audioSaveFile, format, samp
leRate) :

    url = 'http://' + host + '/stream/v1/tts'

    # 设置HTTP Headers
    httpHeaders = {
        'Content-Type': 'application/json'
        }

    # 设置HTTP Body
    body = {'appkey': appKey, 'token': token, 'text': text, 'format': forma
t, 'sample_rate': sampleRate}
    body = json.dumps(body)

    print('The POST request body content: ' + body)

    # Python 2.x 请使用httplib
    # conn = httplib.HTTPConnection(host)

    # Python 3.x 请使用http.client
    conn = http.client.HTTPConnection(host)

    conn.request(method='POST', url=url, body=body, headers=httpHeaders)

    # 处理服务端返回的响应
    response = conn.getresponse()
    print('Response status and response reason:')
    print(response.status ,response.reason)

    contentType = response.getheader('Content-Type')
    print(contentType)

    body = response.read()

    if 'audio/mpeg' == contentType :
        with open(audioSaveFile, mode='wb') as f:
            f.write(body)

        print('The POST request succeed!')
    else :
        print('The POST request failed: ' + str(body))
```

```
    conn.close()



appKey = 'default'
token = 'default'

host = 'gateway所在IP:8101'


text = '今天是周一，天气挺好的。'

# 采用RFC 3986规范进行urlencode编码
textUrlencode = text
# Python 2.x请使用 urllib.quote
# textUrlencode = urllib.quote(textUrlencode, '')
# Python 3.x请使用urllib.parse.quote_plus
textUrlencode = urllib.parse.quote_plus(textUrlencode)
textUrlencode = textUrlencode.replace("+", "%20")
textUrlencode = textUrlencode.replace("*", "%2A")
textUrlencode = textUrlencode.replace("%7E", "~")

print('text: ' + textUrlencode)

audioSaveFile = 'syAudio.wav'
format = 'wav'
sampleRate = 16000


# GET 请求方式
processGETRequest(host, appKey, token, textUrlencode, audioSaveFile, format,
sampleRate)

# POST 请求方式
# processPOSTRequest(host, appKey, token, text, audioSaveFile, format, sample
Rate)
```

PHP Demo

阿里巴巴智能语音

```php
<?php

function processGETRequest($url, $appkey, $token, $text, $audioSaveFile, $format, $sampleRate) {

    $url = $url . "?appkey=" . $appkey;
    $url = $url . "&token=" . $token;
    $url = $url . "&text=" . $text;
    $url = $url . "&format=" . $format;
    $url = $url . "&sample_rate=" . strval($sampleRate);

    // voice 发音人，可选，默认是xiaoyun
    // $url = $url . "&voice=" . "xiaoyun";
    // volume 音量，范围是0~100，可选，默认50
    // $url = $url . "&volume=" . strval(50);
    // speech_rate 语速，范围是-500~500，可选，默认是0
    // $url = $url . "&speech_rate=" . strval(0);
    // pitch_rate 语调，范围是-500~500，可选，默认是0
    // $url = $url . "&pitch_rate=" . strval(0);

    print $url . "\n";

    $curl = curl_init();
    curl_setopt($curl, CURLOPT_RETURNTRANSFER, TRUE);

    /**
     * 设置HTTP GET URL
     */
    curl_setopt($curl, CURLOPT_URL, $url);

    /**
     * 设置返回的响应包含HTTP头部信息
     */
    curl_setopt($curl, CURLOPT_HEADER, TRUE);

    /**
     * 发送HTTP GET请求
     */
    curl_setopt($curl, CURLOPT_SSL_VERIFYPEER, FALSE);
    curl_setopt($curl, CURLOPT_SSL_VERIFYHOST, FALSE);
    $response = curl_exec($curl);

    if ($response == FALSE) {
        print "curl_exec failed!\n";
```

```
        curl_close($curl);
        return ;
    }

    /**
     * 处理服务端返回的响应
     */
    $headerSize = curl_getinfo($curl, CURLINFO_HEADER_SIZE);
    $headers = substr($response, 0, $headerSize);
    $bodyContent = substr($response, $headerSize);

    curl_close($curl);

    if (stripos($headers, "Content-Type: audio/mpeg") != FALSE || stripos($hea
ders, "Content-Type:audio/mpeg") != FALSE) {
        file_put_contents($audioSaveFile, $bodyContent);
        print "The GET request succeed!\n";
    }
    else {
        print "The GET request failed: " . $bodyContent . "\n";
    }
}

function processPOSTRequest($url, $appkey, $token, $text, $audioSaveFile, $for
mat, $sampleRate) {

    /**
     * 请求参数，以JSON格式字符串填入HTTP POST请求的Body中
     */
    $taskArr = array(
        "appkey" => $appkey,
        "token" => $token,
        "text" => $text,
        "format" => $format,
        "sample_rate" => $sampleRate
        // voice 发音人，可选，默认是xiaoyun
        // "voice" => "xiaoyun",
        // volume 音量，范围是0~100，可选，默认50
        // "volume" => 50,
        // speech_rate 语速，范围是-500~500，可选，默认是0
        // "speech_rate" => 0,
        // pitch_rate 语调，范围是-500~500，可选，默认是0
        // "pitch_rate" => 0
    );
```

```
$body = json_encode($taskArr);
print "The POST request body content: " . $body . "\n";

$curl = curl_init();
curl_setopt($curl, CURLOPT_RETURNTRANSFER, TRUE);

/**
 * 设置HTTP POST URL
 */
curl_setopt($curl, CURLOPT_URL, $url);
curl_setopt($curl, CURLOPT_POST, TRUE);

/**
 * 设置HTTP POST请求头部
 * */
$httpHeaders = array(
    "Content-Type: application/json"
);
curl_setopt($curl, CURLOPT_HTTPHEADER, $httpHeaders);

/**
 * 设置HTTP POST请求体
 */
curl_setopt($curl, CURLOPT_POSTFIELDS, $body);

/**
 * 设置返回的响应包含HTTP头部信息
 */
curl_setopt($curl, CURLOPT_HEADER, TRUE);

/**
 * 发送HTTP POST请求
 */
$response = curl_exec($curl);

if ($response == FALSE) {
    print "curl_exec failed!\n";
    curl_close($curl);
    return ;
}

/**
 * 处理服务端返回的响应
 */
$headerSize = curl_getinfo($curl, CURLINFO_HEADER_SIZE);
```

```
    $headers = substr($response, 0, $headerSize);
    $bodyContent = substr($response, $headerSize);

    curl_close($curl);

    if (stripos($headers, "Content-Type: audio/mpeg") != FALSE || stripos($hea
ders, "Content-Type:audio/mpeg") != FALSE) {
        file_put_contents($audioSaveFile, $bodyContent);
        print "The POST request succeed!\n";
    }
    else {
        print "The POST request failed: " . $bodyContent . "\n";
    }
}



$appkey = "default";
$token = "default";

$url = "http://gateway所在IP:8101/stream/v1/tts";

$text = "今天是周一，天气挺好的。";

$textUrlEncode = urlencode($text);
$textUrlEncode = preg_replace('/\+/', '%20', $textUrlEncode);
$textUrlEncode = preg_replace('/\*/', '%2A', $textUrlEncode);
$textUrlEncode = preg_replace('/%7E/', '~', $textUrlEncode);

$audioSaveFile = "syAudio.wav";
$format = "wav";
$sampleRate = 16000;

processGETRequest($url, $appkey, $token, $textUrlEncode, $audioSaveFile, $form
at, $sampleRate);

//processPOSTRequest($url, $appkey, $token, $text, $audioSaveFile, $format, $s
ampleRate);

?>
```

Node.js Demo

说明：request依赖安装，请在您的Demo文件所在目录执行如下命令：

```
npm install request --save
```

示例代码：

```
const request = require('request');
const fs = require('fs');

function processGETRequest(url, appkey, token, text, audioSaveFile, format, s
ampleRate) {

    /**
     * 设置URL请求参数
     */
    url = url + '?appkey=' + appkey;
    url = url + '&token=' + token;
    url = url + '&text=' + text;
    url = url + '&format=' + format;
    url = url + '&sample_rate=' + sampleRate;

    // voice 发音人，可选，默认是xiaoyun
    // url = url + "&voice=" + "xiaoyun";
    // volume 音量，范围是0~100，可选，默认50
    // url = url + "&volume=" + 50;
    // speech_rate 语速，范围是-500~500，可选，默认是0
    // url = url + "&speech_rate=" + 0;
    // pitch_rate 语调，范围是-500~500，可选，默认是0
    // url = url + "&pitch_rate=" + 0;

    console.log(url);

    /**
     * 设置HTTP GET请求
     * encoding必须设置为null，HTTP响应的Body为二进制Buffer类型
     */
    var options = {
        url: url,
        method: 'GET',
        encoding: null
    };
    request(options, function (error, response, body) {
        /**
         * 处理服务端的响应
```

```
         */
        if (error != null) {
            console.log(error);
        }
        else {
            var contentType = response.headers['content-type'];
            if (contentType === undefined || contentType != 'audio/mpeg') {
                console.log(body.toString());
                console.log('The GET request failed!');
            }
            else {
                fs.writeFileSync(audioSaveFile, body);
                console.log('The GET request is succeed!');
            }
        }
    });
}


function processPOSTRequest(url, appkeyValue, tokenValue, textValue, audioSav
eFile, formatValue, sampleRateValue) {

    /**
     * 请求参数，以JSON格式字符串填入HTTP POST 请求的Body中
    */
    var task = {
        appkey : appkeyValue,
        token : tokenValue,
        text : textValue,
        format : formatValue,
        sample_rate : sampleRateValue
        // voice 发音人，可选，默认是xiaoyun
        // voice : 'xiaoyun',
        // volume 音量，范围是0~100，可选，默认50
        // volume : 50,
        // speech_rate 语速，范围是-500~500，可选，默认是0
        // speech_rate : 0,
        // pitch_rate 语调，范围是-500~500，可选，默认是0
        // pitch_rate : 0
    };
    var bodyContent = JSON.stringify(task);
    console.log('The POST request body content: ' + bodyContent);

    /**
     * 设置HTTP POST请求头部
```

```
     */
    var httpHeaders = {
        'Content-type' : 'application/json'
    }

    /**
     * 设置HTTP POST请求
     * encoding必须设置为null，HTTP响应的Body为二进制Buffer类型
     */
    var options = {
        url: url,
        method: 'POST',
        headers: httpHeaders,
        body: bodyContent,
        encoding: null
    };

    request(options, function (error, response, body) {
        /**
         * 处理服务端的响应
         */
        if (error != null) {
            console.log(error);
        }
        else {
            var contentType = response.headers['content-type'];
            if (contentType === undefined || contentType != 'audio/mpeg') {
                console.log(body.toString());
                console.log('The POST request failed!');
            }
            else {
                fs.writeFileSync(audioSaveFile, body);
                console.log('The POST request is succeed!');
            }
        }
    });
}



var appkey = 'default';
var token = 'default';

var url = 'http://gateway所在IP:8101/stream/v1/tts';
```

```
var text = '今天是周一，天气挺好的。';

var textUrlEncode = encodeURIComponent(text)
                    .replace(/[!'()*]/g, function(c) {
                      return '%' + c.charCodeAt(0).toString(16);
                    });



console.log(textUrlEncode);

var audioSaveFile = 'syAudio.wav';
var format = 'wav';
var sampleRate = 16000;

processGETRequest(url, appkey, token, textUrlEncode, audioSaveFile, format, s
ampleRate);

// processPOSTRequest(url, appkey, token, text, audioSaveFile, format, sample
Rate);
```