

京东AI语音合成iOS、iPad OS与Mac OS SDK用户接入手册

1. 简介

京东AI语音合成iOS_SDK是集成京东AI语音合成相关基础服务，运行在iOS、iPad OS与Mac OS设备上的一体化开放工具包。

1-1. 基础功能

主要支持的功能: 基本功能: 语音数据处理，网络通讯，音频实时回调，实时播放合成音频等。

2. 使用说明

2-1. 兼容性

类别	兼容范围
系统	支持iOS8.0及以上系统，包含iPhone、iPad与Mac（Mac OS 10.15及以上系统）设备
架构	armv7、arm64、i386、x86_64
网络	支持移动网络、WIFI等网络环境
外设	设备有声音输出
开发环境	建议使用最新版本Xcode进行开发

2-2. 资源占用描述

SDK类型	JDSpeechSynthesizer.framework	大小
语音合成头文件	JDSTTSDefines.h JDSTTSParameters.h JDSpeechSynthesizer.h JDSpeechSynthesizerDelegate.h JDSpeechError.h	20K
语音合成静态库	JDSpeechSynthesizer	13M

支持BITCODE，开启和关闭BITCODE对于最终编译大小影响不大，编译开启了BITCODE选项。由于 BITCODE 开启会导致二进制文件体积增大，这部分会在 AppStore 发布时进行进一步编译优化，并不会引起最终文件的体积变化。虽然ipa的包会有增大，但实际在AppStore的优化下，用户下载的应用大小会远小于ipa文件的大小。

3. 集成指南

3-1. Demo工程

强烈建议在使用iOS SDK之前，运行并试用Demo工程相关功能，参考Demo工程的调用和配置的方式。

1. 双击使用Xcode打开JDTTSSpeechTest/JDSpeechTest.xcodeproj
2. 确保网络通畅，运行工程

3-2. 开发前准备

对于开发者使用京东语AI音云服务，登录NeuHub京东人工智能开放平台，获取有效的AppKey和SecretKey。（通过NeuHub平台调用，无需设置Appid，使用特殊服务调用，请发邮件到jdai-speech@jd.com申请AppId，用于语音SDK鉴权使用。）

3-3. 添加头文件和静态库

只需要引入如下头文件：

```
JDSTTSDefines.h JDSTTSParameters.h JDSpeechSynthesizer.h JDSpeechSynthesizerDelegate.h JDSpeechError.h
```

SDK提供的是静态库，开发者只需要将库文件拖入工程目录即可，可参考demo程序的引用。

4. 语音合成 SDK 接口说明

详细接口代码请参考SDK头文件：

```
JDSTTSDefines.h JDSTTSParameters.h JDSpeechSynthesizer.h JDSpeechSynthesizerDelegate.h JDSpeechError.h
```

SDK版本号为JDSpeechSynthesizerVersion，在JDSpeechSynthesizer.h中定义。

4-1. JDSpeechSynthesizer

JDSpeechSynthesizer为语音合成实现接口。

Inherits from	NSObject
Declared in	JDSpeechSynthesizer.h

delegate

```
设置识别的委托对象
/* OBJECTIV-C */
@property (nonatomic, weak) id<JDSpeechSynthesizerDelegate> delegate;
```

+ sharedInstance

```
返回合成对象的单例
/* OBJECTIV-C */
+ (instancetype)sharedInstance

Return Value
合成对象
```

+ destroy

```
销毁合成对象
/* OBJECTIV-C */
+ (BOOL)destroy

Return Value
成功返回YES, 失败返回NO.
```

- setParameter:forKey:

设置合成参数

```
/* OBJECTIV-C */  
-(BOOL) setParameter:(id) value forKey:(NSString*)key;
```

Parameters

value: 参数取值

key: 合成参数

参数设置详细请参考JDSTTSDefines.h与JDSTTSParameters.h。

Return Value

设置成功返回YES，失败返回NO。

– parameterForKey:

获取合成参数

```
/* OBJECTIV-C */  
-(id) parameterForKey:(NSString *)key;  
参数请参考JDSTTSParameters.h。
```

Parameters

key: 参数key

Return Value

参数值

– startSpeaking:

开始合成 (播放)

调用此函数进行合成，如果发生错误会回调错误onCompleted，当多次连续合成请注意内部缓冲限制

```
/* OBJECTIV-C */  
- (void)startSpeaking:(NSString *)text
```

Parameters

text: 合成的文本，最大的字节数为1k

– synthesize:

开始合成 (不播放)

调用此函数进行合成，如果发生错误会回调错误onCompleted，当多次连续合成请注意内部缓冲限制

```
/* OBJECTIV-C */  
- (void) synthesize:(NSString *)text;
```

Parameters

text: 合成的文本，最大的字节数为1k

– pauseSpeaking

暂停播放

暂停播放之后，合成不会暂停，仍会继续，如果发生错误则会回调错误onCompleted

```
/* OBJECTIV-C */  
- (void) pauseSpeaking;
```

– resumeSpeaking

恢复播放

```
/* OBJECTIV-C */  
- (void) resumeSpeaking;
```

– stop

```
停止播放并停止合成，任务完成之后调用或者是出错了需要调用
/* OBJECTIV-C */
- (void) stop;
```

isSpeaking

```
是否正在播放
@property (nonatomic, readonly) BOOL isSpeaking;
```

4-2. JDSpeechSynthesizerDelegate

JDSpeechSynthesizerDelegate为语音合成回调接口。

Conforms to	NSObject
Declared in	JDSpeechSynthesizerDelegate.h

– onCompleted:

```
结束回调，required method
当整个合成结束之后会回调此函数，如果进行连续合成，收到此回调之后，异步地进行下一次合成请求。
/* OBJECTIV-C */
- (void) onCompleted:(JDSpeechError*) error;

Parameters
error 错误码与错误描述，详见JDSpeechError.h.
```

– onBegin

```
开始合成回调
/* OBJECTIV-C */
- (void) onBegin;
```

– onSynthesizeData:timeStamp:

```
合成数据回调
/* OBJECTIV-C */
- (void) onSynthesizeData:(NSData*) data timeStamp:(NSString *) timeStamp;

Parameters
data: 合成音频数据，PCM格式
timeStamp: 音频数据对应的音素时间序列，如果没有设置timeStamp参数，为nil
```

– onSynthesizeProgress:message:

```
合成进度回调
/* OBJECTIV-C */
- (void) onSynthesizeProgress:(float) progress message:(NSString *)msg;

Parameters
progress: 缓冲进度，0-100
msg:      附件信息，此版本为nil
```

– onSpeakBegin

```
开始播放回调
/* OBJECTIV-C */
- (void) onSpeakBegin;
```

– onSpeakProgress:message:

```
播放进度回调
/* OBJECTIV-C */
- (void) onSpeakProgress:(float) progress message:(NSString *)msg;

Parameters
progress: 当前播放进度, 0-100
msg:      附件信息, 此版本为nil
```

– onSpeakEnd

```
结束播放回调
/* OBJECTIV-C */
- (void) onSpeakEnd;
```

– onBufferStart:

```
缓冲开始回调
/* OBJECTIV-C */
- (void) onBufferStart:(NSString *)msg;

Parameters
msg: 附件信息, 此版本为nil
```

– onBufferEnd:

```
缓冲技术回调
/* OBJECTIV-C */
- (void) onBufferEnd:(NSString *)msg;

Parameters
msg: 附件信息, 此版本为nil
```

4-3. JDSpeechError

JDSpeechError为语音错误描述类。

Inherits from	NSObject
Declared in	JDSpeechError.h

errorCode

```
错误码
/* OBJECTIV-C */
@property(nonatomic,assign) NSInteger errorCode;
```

errorDesc

```
错误描述
/* OBJECTIV-C */
@property(nonatomic,retain) NSString* errorDesc;
```

+ initWithError:err:

初始化

```
/* OBJECTIV-C */  
+ (instancetype) initWithError:(NSInteger) errorCode err:(NSString *)errorDesc;
```

Parameters

errorCode: 错误码
errorDesc: 错误描述

Return Value

@return JDSpeechError对象

5. 示例 demo

```
// 实例化  
if (_speechSynthesizer == nil) {  
    _speechSynthesizer = [JDSpeechSynthesizer sharedInstance];  
    [self printLogs:[NSString stringWithFormat:@"TTS版本: %@", JDSpeechSynthesizerVersion]];  
    // 注册监听回调  
    _speechSynthesizer.delegate = self;  
}  
// 设置参数  
NSString *appKeyVal = [_speechSynthesizer parameterForKey:JDS_TTS_APP_KEY];  
if (![instance.appIDVal isEqualToString:appKeyVal]) {  
    [_speechSynthesizer setParameter:instance.appKeyVal forKey:JDS_TTS_APP_KEY];  
}  
NSString *secretKeyVal = [_speechSynthesizer parameterForKey:JDS_TTS_SECRET_KEY];  
if (![instance.secretKeyVal isEqualToString:secretKeyVal]) {  
    [_speechSynthesizer setParameter:instance.secretKeyVal forKey:JDS_TTS_SECRET_KEY];  
}  
NSString *serverURLVal = [_speechSynthesizer parameterForKey:JDS_TTS_SERVER_URL];  
if (![instance.serverURLVal isEqualToString:serverURLVal]) {  
    [_speechSynthesizer setParameter:instance.serverURLVal forKey:JDS_TTS_SERVER_URL];  
}  
// 启动合成  
if (index == 0) {  
    [self.speechSynthesizer startSpeaking:self.ttsTextView.text];  
} else {  
    [self.speechSynthesizer synthesize:self.ttsTextView.text];  
}
```

6. 参数说明

参数设置中，详细的Key请参考JDSTTSParameters.h，部分默认值设置参考JDSTTSDefines.h。

参数	描述
JDS TTSENGINE_TYPE	引擎类型，在线与离线引擎，类型为NSInteger，默认为EJDS_TTSLocal
JDS TTSSERVER_URL	服务器地址，类型为NSString，默认为空，根据不同产品接口设置，参考NeuHub平台所购买的API接口
JDS TTSAPP_ID	Application Id 调用Neuhub平台无需配置，类型为NSString，默认为空
JDS TTSAPP_KEY	Application Key 类型为NSString，默认为空
JDS TTSSECRET_KEY	Secret Key 类型为NSString，默认为空
JDS TTSSAMPLE_RATE	设置语音合成后音频采样率，默认为24k，类型为NSInteger
JDS TTSSVOLUME_LEVEL	合成语音音量大小，默认值为2.0f，取值范围为[0.1, 10.0]，类型为float
JDS TTSSPEED_LEVEL	合成语音语速大小，默认值为1.0f，取值范围为[0.5, 2.0]，类型为float
JDS TTSLANGUAGE	合成语言，默认为普通话（0：普通话 1：广东话 2：英语），类型为NSInteger
JDS TTSTIM_TYPE	合成语音音色，默认为0，类型为NSInteger 普通版音色：0:桃桃(女声) 1:斌斌(男声) 3:婷婷(女声) 精品版音色：0:桃桃(女声) 4:郭靖 (男声) 6:莫妮卡(女声) 8:德洛丽丝(女声)
JDS TTS_AUE_TYPE	合成语音音频格式，默认为opus（0：wav 1：pcm 2：opus），类型为NSInteger
JDS TTSTIME_STAMP	是否获取音素序列时间戳，默认为不获取（0：不获取 1：获取），类型为NSInteger
JDS TTSRES_PATH	离线tts前端资源文件路径，类型为NSString，默认为空，JDS TTSENGINE_TYPE设置为离线的时候需要设置
JDS TTSVOICE_PATH	离线tts发音人文件路径，类型为NSString，默认为空，JDS TTSENGINE_TYPE设置为离线的时候需要设置

7. 错误码描述

错误码	描述	错误原因	备注
0	成功		
-2001	不支持的参数	用户相关，参数不支持	
-2002	服务器未完成初始化	服务器相关，服务器未完成初始化	
-2003	服务器忙	服务器相关，服务器忙	
-2004	文本太短	用户相关，文本太短，没有文本时返回	
-2005	文本太长	用户相关，文本太长，文本长度超过1k	
-2006	seq_id不正确	用户相关，seq_id 不正确，如值为 1 但是当前已有相同 session; 或值不为 1 但当前没有相同session	
-2007	引擎出错	服务器相关，TTS 引擎相关，TTS 引擎读取错误	
-2008	服务端opus压缩出错	服务器相关，服务端opus压缩出错	
-2009	鉴权失败	鉴权key非法	
-2010	播放器出错		
-2011	网络连接失败	检测网络或者权限问题	
-2012	网络读写超时	网络问题	
-2013	京东云网关错误	京东云网关系统错误信息	
-2014	未播放或者未合成的内部缓冲数量达到上限	内部缓冲限制数量为3	
-2100	无效的前端资源文件	无效的路径或者无效的文件	
-2101	无效的发音人模型文件	无效的路径或者无效的文件	

京东云网关系统错误信息：

错误码	描述	备注
10000	查询成功	
10001	错误的请求appkey	
10003	不存在相应的数据信息	
10004	URL上appkey参数不能为空	
10010	接口需要付费，请充值	
10020	系统繁忙，请稍后再试	
10030	调用网关失败，请与NeuHub联系	
10040	超过每天限量，请明天继续	
10041	URL上timestamp参数不能为空	
10042	URL上sign参数不能为空	
10043	超过QPS限额，请降低调用频率或与NeuHub联系	
10044	集群QPS超限额，请与NeuHub联系	
10045	timestamp参数无效，请检查timestamp距离当前时间是否超过5分钟	
10046	timestamp参数格式错误	
10047	请求签名sign无效，请检查签名信息	
10048	无接口权限，请下单购买	
10050	用户已被禁用	
10060	发布方设置调用权限，请联系发布方	
10090	文件大小超限，请上传小于5M的文件	
11010	发布方接口调用异常，请稍后再试	
11030	发布方接口返回格式有误	

8. 常见问题

集成使用过程中如有问题:

In place "Build Settings"/"Other Linker Flags", setting as "-Objc -all_load".

Demo project integrated with C++ objects, so pls change a file's suffix (any one named *.m) as mm.

In place "Build Settings"/"Compile sources As" , setting as "Objective-C++".