

北京百度网讯科技有限公司

(版权所有,翻版必究)

**百度语音合成SDK iOS离在线融合版**

**开发手册**

**V1.3**

目录

目录 2

1 概念解释 3

2 简介 3

2.1 兼容性 3

2.2 开发包说明 3

2.3 总体框图 4

3 集成指南 4

3.1 添加BDSSpeechSynthesizer到工程 4

3.2 添加授权文件 5

3.3 添加语音合成资源文件 5

3.4 引入 BDSSpeechSynthesizer 的头文件 5

3.5 引入静态库文件 5

4 语音合成 6

4.1 合成前准备 6

4.1.1 设置合成代理 6

4.1.2 设置合成参数 6

4.2 合成同时播放 7

4.2.1 状态监听 8

4.3 获取合成数据自行播放 8

4.3.1 获取合成数据 8

4.4 线程安全相关 9

5 日志 9

FAQ 12

# 概念解释

语音合成是实现人机语音交互，建立一个有听和讲能力的交互系统所必需的关键技术。随着语音技术的发展，百度自主研发了语音合成系统（TTS），功能是接受用户发送的文本，生成语音发送给用户。

对本文中将提到的概念约定如下：

* **语音合成**：将文本合成为语音，即声音文件。
* **合成引擎：**将文本合成为语音的核心模块。
* **TTS**：Text To Speech，即“从文本到语音”。
* **BDSSpeechSynthesizer**：语音合成SDK简称，详见下条。
* **语音合成SDK**：即本开发包，文中简称为BDSSpeechSynthesizer。BDSSpeechSynthesizer是一个封装了语音合成、音频播放功能的语音合成解决方案。借助BDSSpeechSynthesizer可以快速地在应用程序中集成语音合成功能。

# 简介

百度语音合成客户端iOS离在线融合版SDK（以下简称BDSSpeechSynthesizer）是一种面向iOS移动设备的语音合成解决方案，以Cocoa Touch Static Library形式发布。支持在线优先等合模式。目前版本已支持SDK内部直接播放合成语音。离线合成支持语速、音调、音量、引擎优化级别设置、支持男女声、后续版本将等更多的合成参数；在线合成支持男女声、语速、音调、音量、音频码率设置、支持从SDK获取语音数据、成进度提示。

本版本优先使用在线语音合成服务合成，以获得更好的合成效果。如在线合成服务不可用，如网络连接异常，蜂窝信号差等，将会使用离线合成引擎合成文本，保证功能可用，并按规则定期侦测在线语音合成服务，如在线服务可用，下次语音合成将使用在线服务合成。

## 兼容性

系统：支持iOS 6.0及以上。

机型：iPhone和iPad皆可。

架构：支持i386、x86\_64、armv7、arm64。

## 开发包说明

表1开发包说明表

|  |  |
| --- | --- |
| 文件(夹)名 | 说明 |
| Doc/ Baidu\_Combined\_TTS\_SDK\_iOS\_Manual.pdf | 本文档 |
| BDSSpeechSynthesizer\_SDK | 语音合成SDK Lib库，支持simulator和iOS设备 |
| BDSSpeechSynthesizerSample | 开发示例(xcode project) |
| OfflineTTSDatFiles/ Chinese\_Speech\_Female.dat  OfflineTTSDatFiles/ English\_Speech\_Female.dat | 语音合成资源文件 (speech data file，女声) |
| OfflineTTSDatFiles/ Chinese\_Speech\_Male.dat  OfflineTTSDatFiles/ English\_Speech\_Male.dat | 语音合成资源文件(speech data file，男声) |
| OfflineTTSDatFiles/ Chinese\_Text.dat  OfflineTTSDatFiles/ English\_Text.dat | 语音合成资源文件(text data file) |
| OfflineTTSDatFiles/ offline\_engine\_tmp\_license.dat | 授权文件 |

## 总体框图



图1语音合成系统架构图

# 集成指南

## 添加BDSSpeechSynthesizer到工程

BDSSpeechSynthesizer使用了一些系统的framework， 需要添加到工程里面。

添加方式:右键点击 Xcode 中的工程文件,在出现的界面中,选中 TARGETS 下的应用,在 出现的界面中选中 Build Phases->Link Binary With Libraries,点击界面中的“+”图标,在弹出的界面中选择上下图中的一个frame即可, 逐个添加后，添加完成效果图如图 8 所示(libetts\_device\_simulator.a 将在随后添加)。

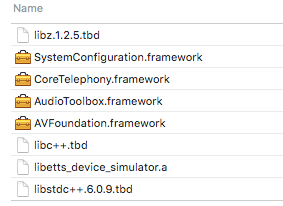


图8 添加framework

## 添加授权文件

将开发包中的OfflineTTSDatFiles目录下的offline\_engine\_tmp\_license.dat文件添加到工程, 或者安装app后部署到app对应的Documents目录下。

## 添加语音合成资源文件

将开发包中的OfflineTTSDatFiles目录下的Chinese\_Text.dat，Chinese\_Speech\_Female.dat/ Chinese\_Speech\_Male.dat文件添加到工程或者安装app后部署到指定目录(代码中启动合成引擎时需要指定该资源文件的访问路径)。

## 引入 BDSSpeechSynthesizer 的头文件

首先将 BDSSpeechSynthesizer 提供的头文件拷贝到工程目录下,在 XCode 中添加此文件,引入 BDSSpeechSynthesizer提供的头文件。

添加如下头文件：

#import "BDSSpeechSynthesizer.h"

## 引入静态库文件

BDSSpeechSynthesizer提供了支持模拟器 6.0 及更新版本、真机armv7，arm64及更新架构所使用的静态库文件,存放在开发包lib目录下。

引入静态库文件的具体方式为:将libetts\_device\_simulator.a 采用添加文件方式添加到工程的Framework 目录下。

说明: libetts\_device\_simulator.a 是一个通用的库文件，支持armv7、arm64、i386、x86\_64, 避免开发者在 build 不同 target 时频繁替换.a 文件的问题

# 语音合成

## 合成前准备

### 设置合成代理

设置实现BDSSpeechSynthesizerDelegate协议的代理，用于接收合成过程中的相关异步消息：

// 设置合成器代理

[[BDSSpeechSynthesizer sharedInstance] setSynthesizerDelegate: self];

### 设置合成参数

如使用在线合成，必须设置3.1和3.2节从开放云平台获取的apikey和secretkey。可以选择配置在线服务的超时时间。如使用离线合成，必须设置离线合成引擎所需的音库文件路径，和相关的授权信息。

// 在线相关设置

[[BDSSpeechSynthesizer sharedInstance] setApiKey:@"your apikey" withSecretKey:@"your secretkey"];

[[BDSSpeechSynthesizer sharedInstance] setSynthesizerParam:[NSNumber numberWithFloat:10.0]

forKey: BDS\_SYNTHESIZER\_PARAM\_ONLINE\_REQUEST\_TIMEOUT

];

// 离线相关设置

NSError\* ret=[[BDSSpeechSynthesizer sharedInstance] loadOfflineEngine: @"text data file path"

speechDataPath:@”speech data file path”

licenseFilePath:@ "license file path"

withAppCode: @"your-appcode"

];

if( ret != nil) {/\*错误\*/}

其他合成相关参数设置，代码如下（完整参数列表详见头文件BDSSpeechSynthesizerParams.h）：

// 合成参数设置

[[BDSSpeechSynthesizer sharedInstance] setSynthesizerParam:

[NSNumbernumberWithInt:BDS\_SYNTHESIZER\_SPEAKER\_FEMALE]

forKey:BDS\_SYNTHESIZER\_PARAM\_SPEAKER

];

[[BDSSpeechSynthesizer sharedInstance] setSynthesizerParam:[NSNumber numberWithInt:5]

forKey:BDS\_SYNTHESIZER\_PARAM\_VOLUME

];

[[BDSSpeechSynthesizer sharedInstance] setSynthesizerParam: [NSNumber numberWithInt:5]

forKey:BDS\_SYNTHESIZER \_PARAM\_SPEED

];

[[BDSSpeechSynthesizer sharedInstance] setSynthesizerParam: [NSNumber numberWithInt:5]

forKey:BDS\_SYNTHESIZER \_PARAM\_PITCH

];

[[BDSSpeechSynthesizer sharedInstance] setSynthesizerParam:

[NSNumber numberWithInt: BDS\_SYNTHESIZER\_AUDIO\_ENCODE\_MP3\_16K]

forKey:BDS\_SYNTHESIZER\_PARAM\_AUDIO\_ENCODING

];

注：目前暂不支持BDS\_SYNTHESIZER\_PARAM\_LANGUAGE。

参数默认值列表如下：

|  |  |  |
| --- | --- | --- |
| 参数名 | 默认值 | 备注 |
| BDS\_SYNTHESIZER\_PARAM\_SPEAKER | BDS\_SYNTHESIZER\_SPEAKER\_FEMALE | 女声 |
| BDS\_SYNTHESIZER \_PARAM\_VOLUME | 5 | 中级音量 |
| BDS\_SYNTHESIZER \_PARAM\_SPEED | 5 | 中速 |
| BDS\_SYNTHESIZER \_PARAM\_PITCH | 5 | 中调 |
| BDS\_SYNTHESIZER\_PARAM\_AUDIO\_ENCODING | BDS\_SYNTHESIZER\_AUDIO\_ENCODE\_MP3\_16K | mp3压缩的16k |

## 合成同时播放

NSError\* speakerr;

NSInteger sentenceID = [[BDSpeechSynthesizer sharedInstance] speakSentence: @"百度一下" withError:&speakerr];

if (sentenceID == -1) {/\*错误\*/}

也可以同时合成多句：

NSMutableArray\* sentenceIDs = [[NSMutableArray alloc] init];

NSArray\* sentences = @[@"百度一下", @"今天天气怎么样", @"我想出去"]；

NSError\* speakerr;

for(NSString\* str in sentences){

NSInteger sentenceID = [[BDSpeechSynthesizer sharedInstance] speakSentence: str withError:&speakerr];

if (sentenceID == -1) {

/\*错误\*/

}

else{

[sentenceIDs addObject:[NSNumber numberWithInteger:sentenceID]];

}

}

使用多句合成播放时，句子的数量不受限，但每个句子的长度是受限的。SDK通过BDSSpeechSynthesizerDelegate中的 SynthesizeSentence-和SpeakSentence-parameters通知哪一句正在合成/播放。

### 状态监听

为了更好地实现用于界面,BDSSpeechSynthesizer 提供了BDSSpeechSynthesizerDelegate代理对象用于对合成器的状态进行通知。

完整的开发示例请参见开发包所附示例xcode project——TTSDemo。

## 获取合成数据自行播放

BDSSpeechSynthesizer还支持仅获取合成数据，开发者可以在获取数据后先本地缓存，在合适的时候再进行播放。参数设置同4.1.1。

NSError\* syntherr;

NSInteger sentenceID = [[BDSpeechSynthesizer sharedInstance] synthesizeSentence: @"百度一下" withError:&syntherr];

if (sentenceID == -1) {/\*错误\*/}

也可以同时合成多句：

NSMutableArray\* sentenceIDs = [[NSMutableArray alloc] init];

NSArray\* sentences = @[@"百度一下", @"今天天气怎么样", @"我想出去"]；

NSError\* syntherr;

for(NSString\* str in sentences){

NSInteger sentenceID = [[BDSpeechSynthesizer sharedInstance] synthesizeSentence:str withError:&syntherr];

if (sentenceID == -1) {

/\*错误\*/

}

else{

[sentenceIDs addObject:[NSNumber numberWithInteger:sentenceID]];

}

}

使用多句合放时，句子的数量不受限，但每个句子的长度是受限的。SDK通过BDSSpeechSynthesizerDelegate中的 SynthesizeSentence通知哪一句正在合成。

### 获取合成数据

通过实现BDSSpeechSynthesizerDelegate的synthesizerNewDataArrived方法获取数据，代码如下：

- (void)synthesizerNewDataArrived:(NSData \*)newData

DataFormat:(BDSAudioFormat)fmt

characterCount:(int)newLength

sentenceNumber:(NSInteger)SynthesizeSentence

{

[self logDebug:[NSString stringWithFormat:@"新的音频数据：%ld, (当前已经合成了句子%ld的 %d个字节)", (long)[newData length], (long)SynthesizeSentence, newLength]];

}

合成数据分多次返回，合成完成时向BDSSpeechSynthesizerDelegate发送synthesizerFinishWorkingSentence消息。

## 线程安全相关

BDSSpeechSynthesizer 默认运行在UI线程，为保证线程安全，在其运行期间不应该有其他函数阻塞UI线程。对BDSSpeechSynthesizerDelegate 中方法的回调也会在UI线程调用。.

如果需要从非UI线程调用BDSSpeechSynthesizer ，正确的做法是创建一个新的串行分发队列（serial dispatch queue），使用接口setSDKCallbackQueue 将其注册给BDSSpeechSynthesizer 。

BDSSpeechSynthesizer synthesizer = [[BDSSpeechSynthesizer alloc] initSynthesizerWithDelegate

:nil];

// 注：your-apiKey和your-secretKey需要换成在百度开发者中心注册应用得到的对应值

[synthesizer setApiKey:@"your-apiKey" withSecretKey:@"your-secretKey"];

// 注：第1个参数传入任意非空字符串即可

[synthesizer setSDKCallbackQueue:dispatch\_queue\_create("Q\_name", DISPATCH\_QUEUE\_SERIAL)];

至此，对BDSSpeechSynthesizerDelegate 中方法的回调都将移动到传入的分发队列中，更推荐的做法是所有对BDSSpeechSynthesizer 的接口的调用都移动到该分发队列中。可以通过接口getCurrentCallbackQueue 得到BDSSpeechSynthesizer 当前正在使用的分发队列。

dispatch\_sync([synthesizer getCurrentCallbackQueue], ^

{

[[BDSpeechSynthesizer sharedInstance] speakSentence: @"百度一下" withError:nil];

});

# 日志

为了方便调试，BDSSpeechSynthesizer提供了对日志级别进行设置的接口，通过如下代码可以设置日志级别。

[BDSEmbeddedSynthesizer setLogLevel: BDS\_PUBLIC\_LOG\_DEBUG];

提供6个级别的日志，从低到高依次为VERBOSE、DEBUG、INFO、WARN、ERROR、OFF，其中VERBOSE将输出最多日志，OFF将关闭所有日志，在模拟器中默认日志级别为DEBUG， iOS系统真机中默认日志级别为ERROR。

FAQ

1. 如何联系我们？

开发过程中有任何bug或反馈意见，请发送邮件至voice\_feedback@baidu.com

欢迎加入百度语音开发者交流Hi群：1412729 或 QQ群：369510575 获取更多技术支持