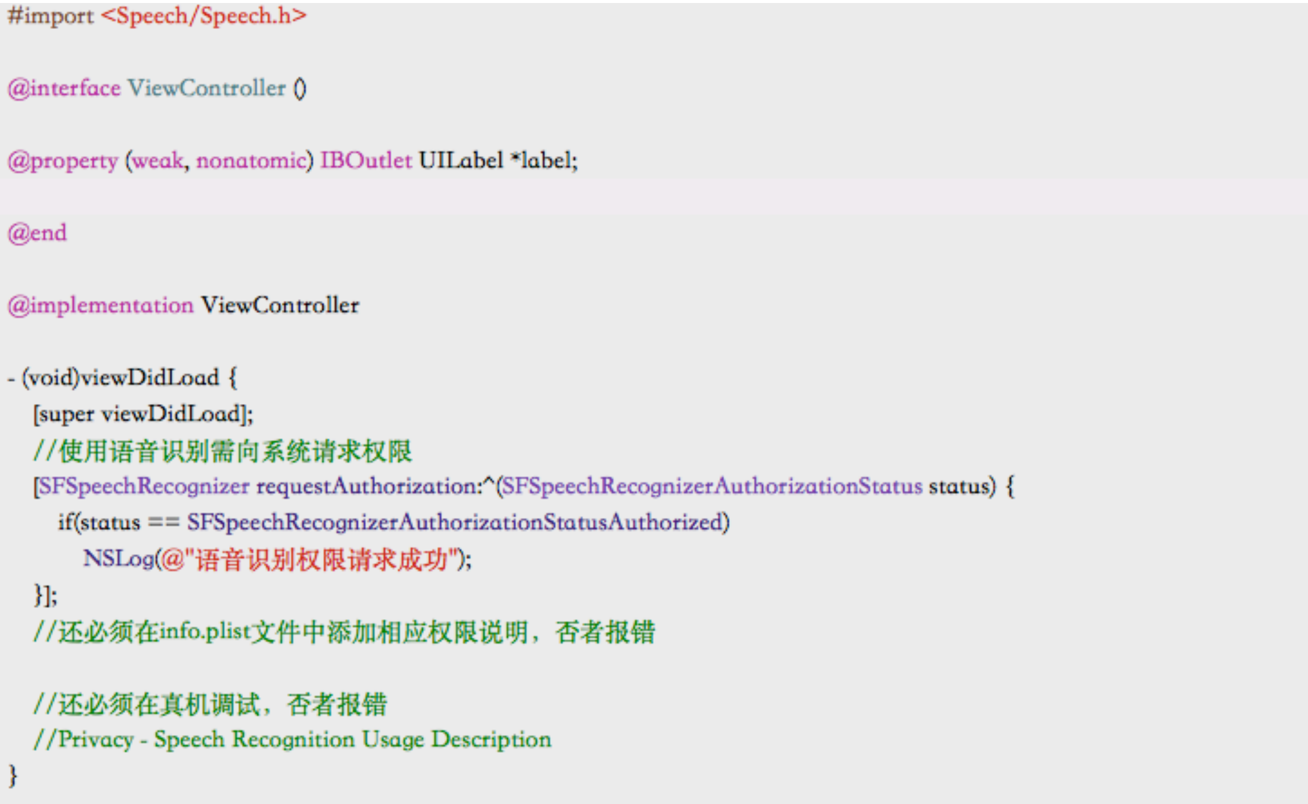
IOS10-语音识别

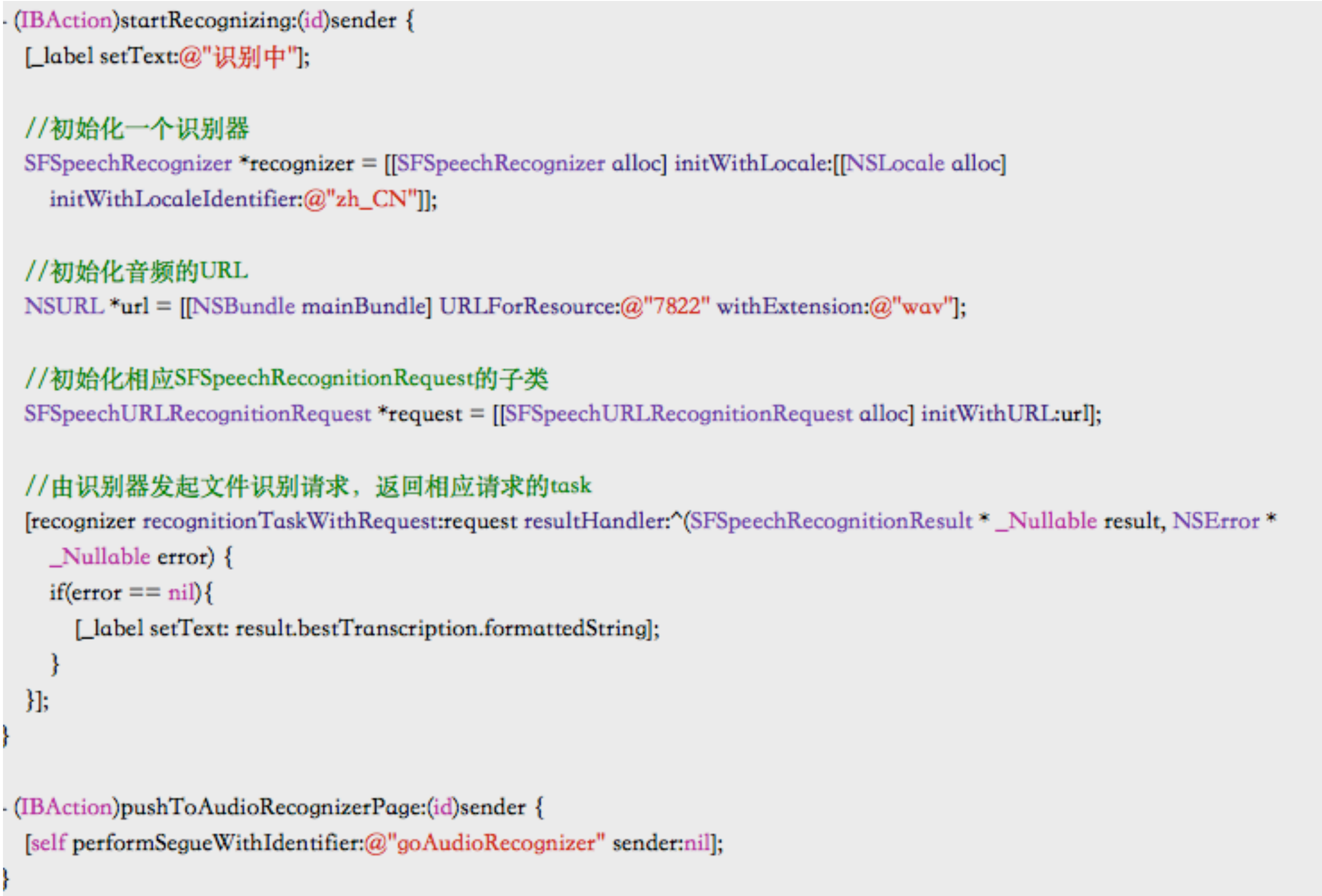
1. 简述：iOS10 开放了一组原生语音识别 API，Sppech，这个框架支持持续的语音识别、对语音文件以及对语音流的识别，并且支持多语言的听写。在这之前，iOS 开发用到语音识别一般是借助与第三方如：讯飞语音。而 Speech 已经可以满足很多 App 功能需求了。

2. 实例：

   (1) 对语音文件的识别<speech/speech.h>

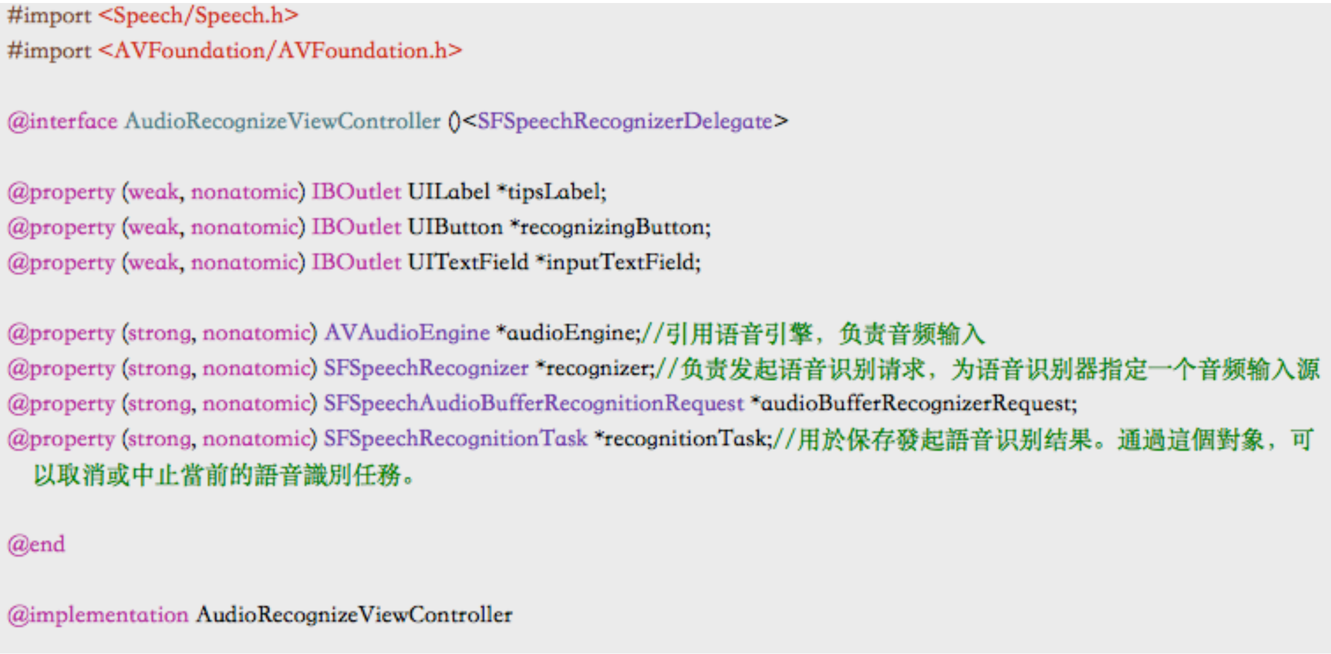


```
#import <Speech/Speech.h>

@interface ViewController ()

@property (weak, nonatomic) IBOutlet UILabel *label;

@end

@implementation ViewController

- (void)viewDidLoad {
    [super viewDidLoad];
    //使用语音识别需向系统请求权限
    [SFSpeechRecognizer requestAuthorization:^(SFSpeechRecognizerAuthorizationStatus status) {
        if(status == SFSpeechRecognizerAuthorizationStatusAuthorized)
            NSLog(@"语音识别权限请求成功");
    }];
    //还必须在info.plist文件中添加相应权限说明，否者报错

    //还必须在真机调试，否者报错
    //Privacy - Speech Recognition Usage Description
}
```

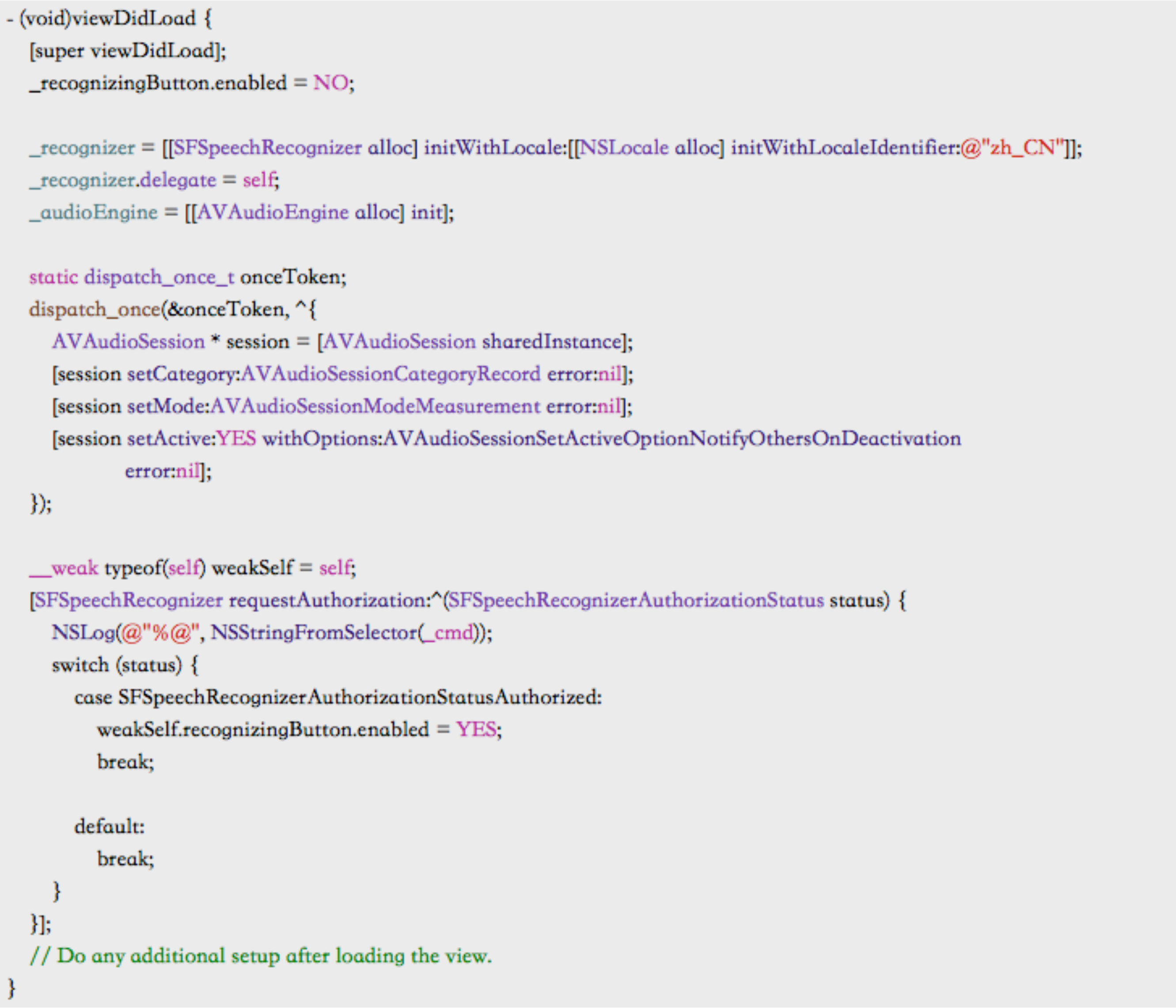


```
- (IBAction)startRecognizing:(id)sender {
  [_label setText:@"识别中"];

  //初始化一个识别器
  SFSpeechRecognizer *recognizer = [[SFSpeechRecognizer alloc] initWithLocale:[[NSLocale alloc]
    initWithLocaleIdentifier:@"zh_CN"]];

  //初始化音频的URL
  NSURL *url = [[NSBundle mainBundle] URLForResource:@"7822" withExtension:@"wav"];

  //初始化相应SFSpeechRecognitionRequest的子类
  SFSpeechURLRecognitionRequest *request = [[SFSpeechURLRecognitionRequest alloc] initWithURL:url];

  //由识别器发起文件识别请求，返回相应请求的task
  [recognizer recognitionTaskWithRequest:request resultHandler:^(SFSpeechRecognitionResult * _Nullable result, NSError *
    _Nullable error) {
    if(error == nil){
      [_label setText: result.bestTranscription.formattedString];
    }
  }];
}

- (IBAction)pushToAudioRecognizerPage:(id)sender {
  [self performSegueWithIdentifier:@"goAudioRecognizer" sender:nil];
}
```

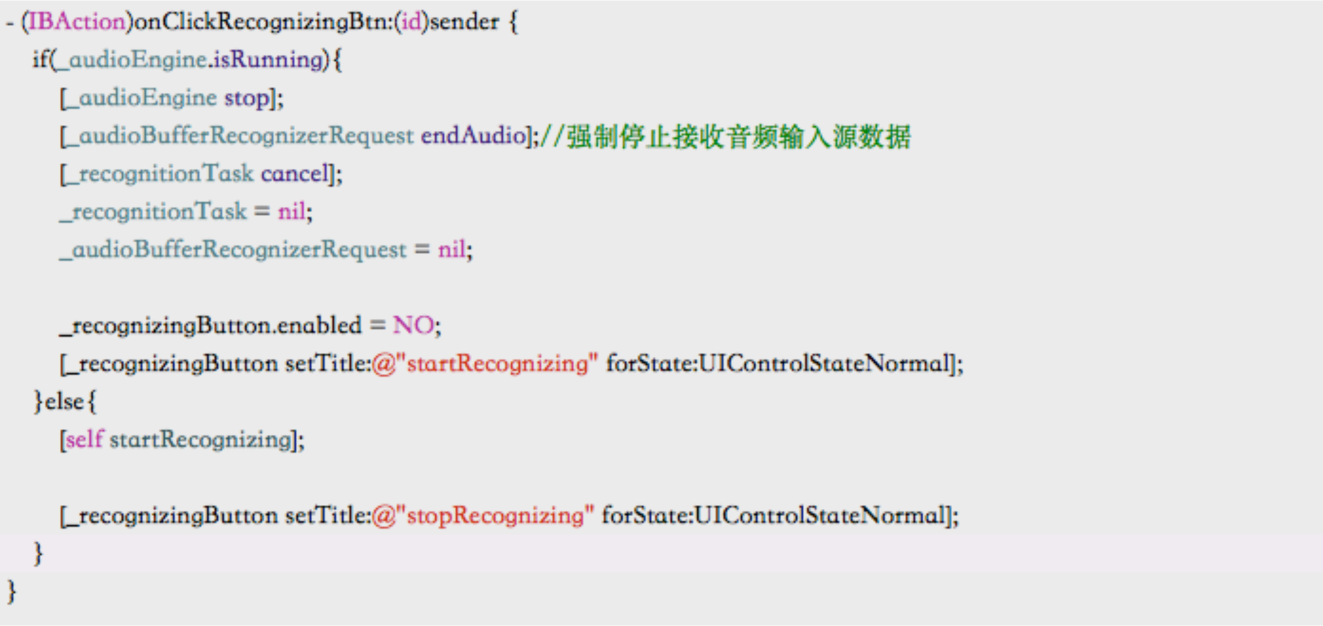
(2) 对语音流的识别



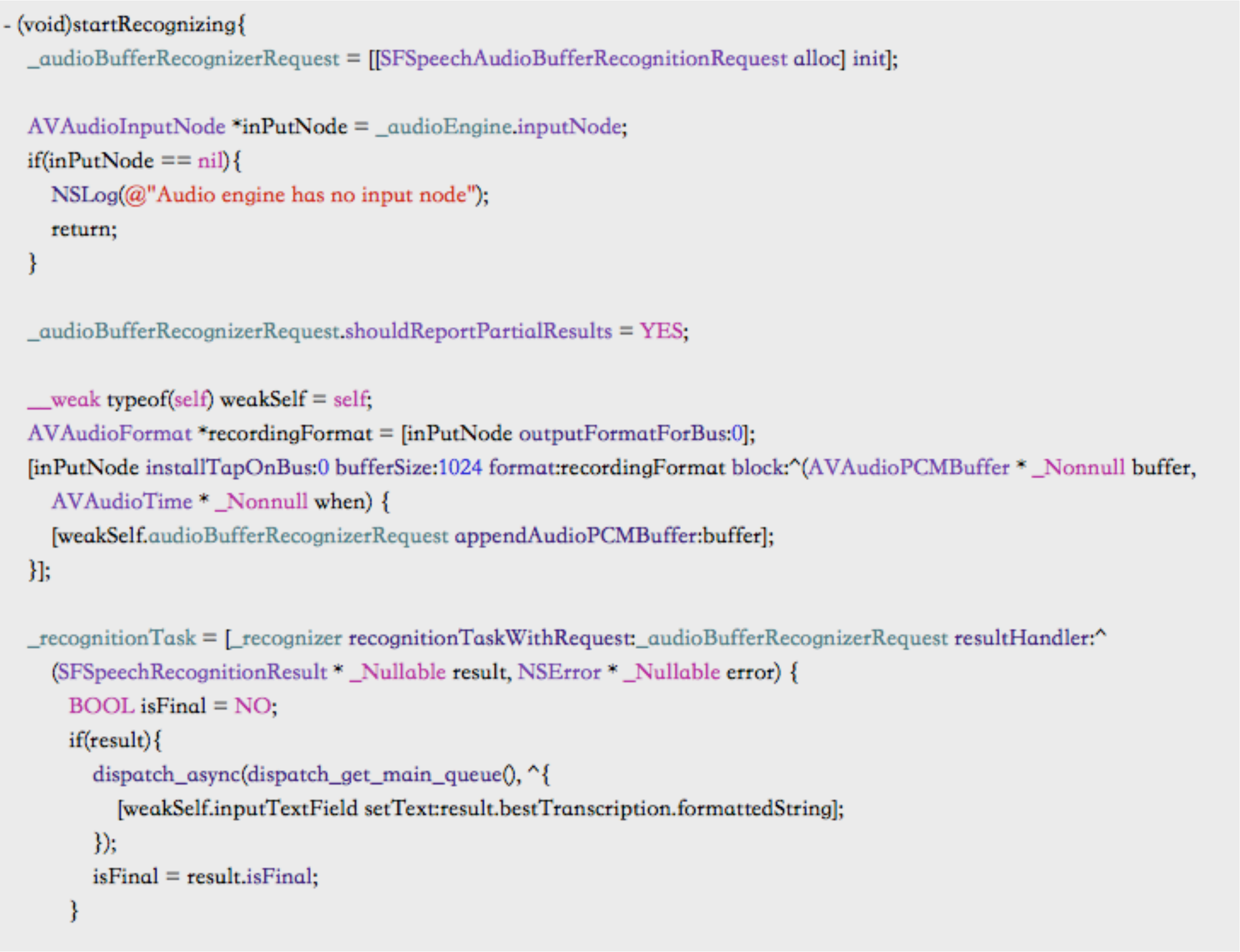
```
#import <Speech/Speech.h>
#import <AVFoundation/AVFoundation.h>

@interface AudioRecognizeViewController ()<SFSpeechRecognizerDelegate>

@property (weak, nonatomic) IBOutlet UILabel *tipsLabel;
@property (weak, nonatomic) IBOutlet UIButton *recognizingButton;
@property (weak, nonatomic) IBOutlet UITextField *inputTextField;

@property (strong, nonatomic) AVAudioEngine *audioEngine;//引用语音引擎，负责音频输入
@property (strong, nonatomic) SFSpeechRecognizer *recognizer;//负责发起语音识别请求，为语音识别器指定一个音频输入源
@property (strong, nonatomic) SFSpeechAudioBufferRecognitionRequest *audioBufferRecognizerRequest;
@property (strong, nonatomic) SFSpeechRecognitionTask *recognitionTask;//用於保存發起語音识别结果。通過這個對象，可
   以取消或中止當前的語音識別任務。

@end

@implementation AudioRecognizeViewController
```



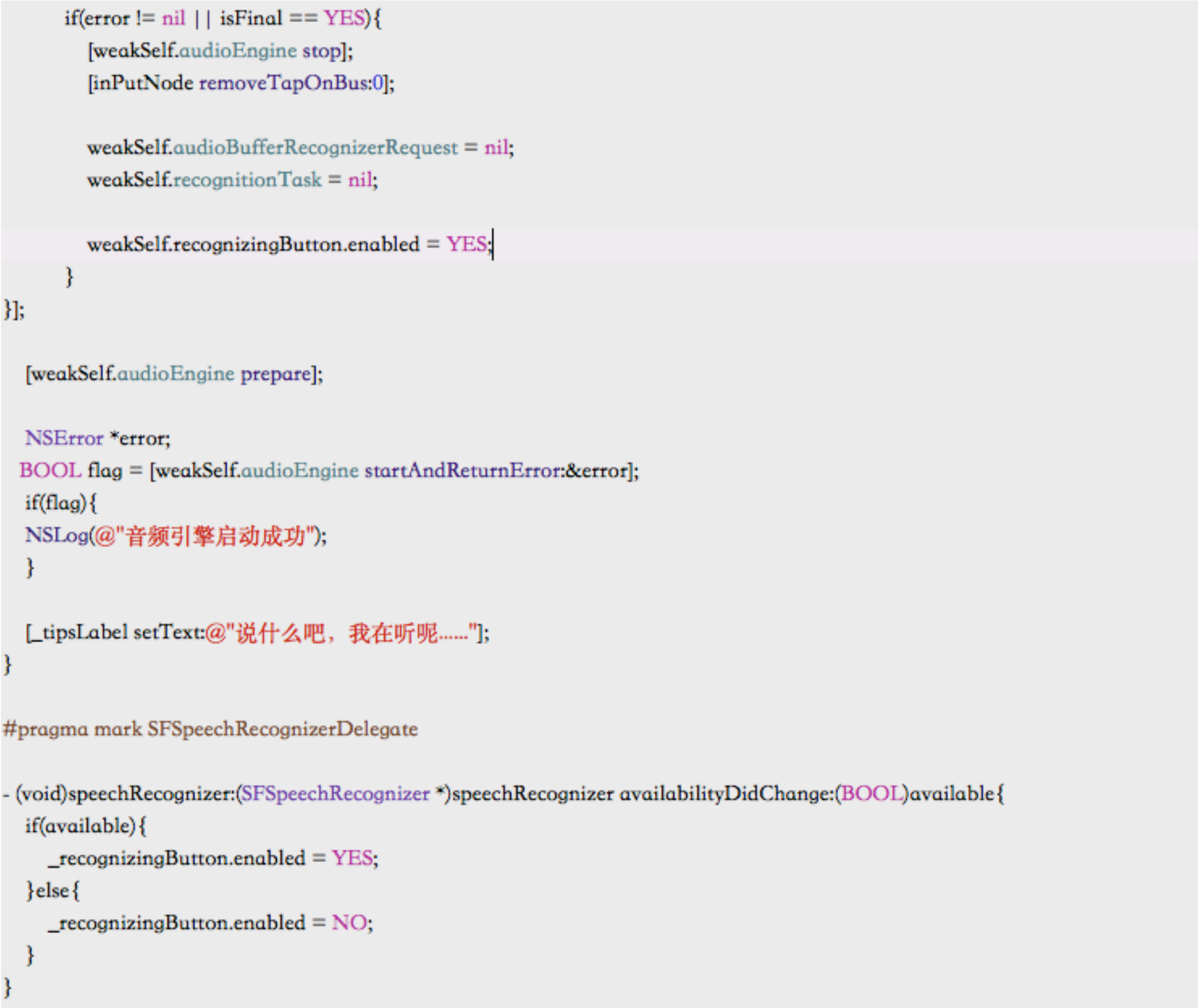
```
- (void)viewDidLoad {
    [super viewDidLoad];
    _recognizingButton.enabled = NO;

    _recognizer = [[SFSpeechRecognizer alloc] initWithLocale:[[NSLocale alloc] initWithLocaleIdentifier:@"zh_CN"]];
    _recognizer.delegate = self;
    _audioEngine = [[AVAudioEngine alloc] init];

    static dispatch_once_t onceToken;
    dispatch_once(&onceToken, ^{
        AVAudioSession * session = [AVAudioSession sharedInstance];
        [session setCategory:AVAudioSessionCategoryRecord error:nil];
        [session setMode:AVAudioSessionModeMeasurement error:nil];
        [session setActive:YES withOptions:AVAudioSessionSetActiveOptionNotifyOthersOnDeactivation
                    error:nil];
    });

    __weak typeof(self) weakSelf = self;
    [SFSpeechRecognizer requestAuthorization:^(SFSpeechRecognizerAuthorizationStatus status) {
        NSLog(@"%@", NSStringFromSelector(_cmd));
        switch (status) {
            case SFSpeechRecognizerAuthorizationStatusAuthorized:
                weakSelf.recognizingButton.enabled = YES;
                break;

            default:
                break;
        }
    }];
    // Do any additional setup after loading the view.
}
```



```
- (IBAction)onClickRecognizingBtn:(id)sender {
    if(_audioEngine.isRunning){
        [_audioEngine stop];
        [_audioBufferRecognizerRequest endAudio];//强制停止接收音频输入源数据
        [_recognitionTask cancel];
        _recognitionTask = nil;
        _audioBufferRecognizerRequest = nil;

        _recognizingButton.enabled = NO;
        [_recognizingButton setTitle:@"startRecognizing" forState:UIControlStateNormal];
    }else{
        [self startRecognizing];

        [_recognizingButton setTitle:@"stopRecognizing" forState:UIControlStateNormal];
    }
}
```



```
- (void)startRecognizing{
  _audioBufferRecognizerRequest = [[SFSpeechAudioBufferRecognitionRequest alloc] init];

  AVAudioInputNode *inPutNode = _audioEngine.inputNode;
  if(inPutNode == nil){
    NSLog(@"Audio engine has no input node");
    return;
  }

  _audioBufferRecognizerRequest.shouldReportPartialResults = YES;

  __weak typeof(self) weakSelf = self;
  AVAudioFormat *recordingFormat = [inPutNode outputFormatForBus:0];
  [inPutNode installTapOnBus:0 bufferSize:1024 format:recordingFormat block:^(AVAudioPCMBuffer * _Nonnull buffer,
    AVAudioTime * _Nonnull when) {
    [weakSelf.audioBufferRecognizerRequest appendAudioPCMBuffer:buffer];
  }];

  _recognitionTask = [_recognizer recognitionTaskWithRequest:_audioBufferRecognizerRequest resultHandler:^
    (SFSpeechRecognitionResult * _Nullable result, NSError * _Nullable error) {
      BOOL isFinal = NO;
      if(result){
        dispatch_async(dispatch_get_main_queue(), ^{
          [weakSelf.inputTextField setText:result.bestTranscription.formattedString];
        });
        isFinal = result.isFinal;
      }
```



```
        if(error != nil || isFinal == YES){
            [weakSelf.audioEngine stop];
            [inPutNode removeTapOnBus:0];

            weakSelf.audioBufferRecognizerRequest = nil;
            weakSelf.recognitionTask = nil;

            weakSelf.recognizingButton.enabled = YES;
        }
}];

   [weakSelf.audioEngine prepare];

   NSError *error;
  BOOL flag = [weakSelf.audioEngine startAndReturnError:&error];
   if(flag){
   NSLog(@"音频引擎启动成功");
   }

   [_tipsLabel setText:@"说什么吧，我在听呢……"];
}

#pragma mark SFSpeechRecognizerDelegate

- (void)speechRecognizer:(SFSpeechRecognizer *)speechRecognizer availabilityDidChange:(BOOL)available{
   if(available){
      _recognizingButton.enabled = YES;
   }else{
      _recognizingButton.enabled = NO;
   }
}
```