SpeechRecSDK for docomo Developer support

開発ガイド

（第 1.0.2 版）

エヌ・ティ・ティ アイティ株式会社

CopyRight© 2015 エヌ・ティ・ティ アイティ株式会社

商標
・SpeechRec はエヌ・ティ・ティアイティ株式会社の登録商標です。
・iPhone、iPad、Mac、Apple、iTunes、MacOS、iOS は、米国 Apple Inc.の米国およびその他の国における登録商標または商標です。
・Android は Google Inc.の商標または登録商標です。
・その他、本文中の製品名は各社の商標または登録商標です。

注意
1.エヌ・ティ・ティアイティ株式会社からの書面による許諾を得ずに、いかなる方法においても本書の内容の一部または全部を無断で複製、複写、転載、翻訳、頒布することを禁じます。
2.本書の内容は予告なく変更する場合があります。
3.本書の商品性、特定目的に対する適合性に関して、エヌ・ティ・ティアイティ株式会社は保証いたしません。
4.本書の内容については万全を期しておりますが、万一記載内容の誤りなどにお気づきの点がございましたら、エヌ・ティ・ティアイティ株式会社までご連絡ください。
5.運用した結果の影響については、4 項にかかわらず、エヌ・ティ・ティアイティ株式会社は一切の責任を負いかねますのでご了承ください。

改版履歴

版数	日付	変更点
第 1.0.0 版	2014/3/7	新規作成
第 1.0.1 版	2014/10/2	文中の Xcode のバージョンをすべて Xcode 6.0.1 に変更 認識結果 XML の説明を追加 Android 4.4 (KitKat)での不具合対応
第 1.0.2 版	2015/5/11	iOS のインタフェースを Android に合わせて修正 エラー一覧を整備 開発ガイド for Android, for iOS を統合 区間検出ライブラリの改良

目次

1

はじめに

1.1　本書の目的

本書ではおもに SpeechRecSDK のライブラリ仕様、およびサンプルソースの利用方法を記述しています。

1.2　対象となる読者

本書は SpeechRecSDK を利用する開発者向けの資料です。

以下のような技術者の方を前提として記述しています。

- 開発対象となるプラットフォーム、開発ツール、ソフトウェア開発の知識を有する方。
- 開発言語に関する知識を有する方。
- 音声認識の基本的な知識を有する方。

1.3　用語の定義

用語	説明
ユーザ	SpeechRecSDKを利用して作成した音声認識アプリケーションを利用する利用者
音声認識アプリケーション	音声認識を利用するアプリケーション
SpeechRec	NTT アイティの音声認識システムの総称
SpeechRec SDK	SpeechRec ラインアップのひとつで、お客様独自の音声認識アプリケーションを作成するための開発キット
SpeechRec Server(SRS)	音声認識エンジンを制御するサーバ
SpeechRec Client(SRC)	音声認識アプリケーションが、音声認識を利用するためのインタフェースを提供するライブラリ
音声認識サービス	SRC を通じて音声認識を行うサービスクラス
区間検出制御ライブラリ	区間検出ライブラリを制御するラッパーライブラリ。音声認識サービスと連動し、取得した音声の始端・終端の検出、雑音抑圧を行う
区間検出ライブラリ	区間検出、雑音抑圧の機能を有するライブラリ
エンコード制御ライブラリ	エンコードライブラリを制御するラッパーライブラリ。音声認識サービスと連動し、取得した音声の符号化を行う
エンコードライブラリ	任意のコーデックにて音声データの符号化を行うライブラリ

API キー	docomo Developer support から払い出される API キー
SBM モード	音声区間検出を利用する際に下記の何れかを指定する 0: 雑音が大きい環境 1: 雑音が比較的小さい環境
中継サーバ	WebSocket プロトコルを用いたプロキシサーバで、音声認識クライアントと音声認識サーバの中継サーバの役割をなす。
認証サーバ	音声認識サーバへの接続時に別途払い出された API キーを用いて認証を行うサーバ

2

SpeechRecSDK の構成

SpeechRecSDK のファイル構成は OS によって異なります。

SpeechRecSDK for Android のファイル構成は以下の通りです。

フォルダ/ファイル	概要
SpeechRecSDK/	
Docs/	
SpeechRecSDK 製品仕様書.pdf	SpeechRec SDK の製品仕様書
SpeechRecSDK 開発ガイド.pdf	アプリ開発時に必要となるライブラリの配置方法や API の使用方法など
Libs/	
android-websockets.jar	WebSocket ライブラリ
sbm.jar	区間検出制御ライブラリ
speechrec-audio-codec.jar	エンコード制御ライブラリ

speechrec-client.jar	音声認識クライアントライブラリ(SRC)
speechrec-service.jar	音声認識サービスライブラリ
armeabi-v7a/	
libsbm.so	区間検出ライブラリ
libaudiocodec.so	エンコードライブラリ
Assets/	
divide/model_140319	区間検出音声モデル
Sample/	
SpeechRecForAndroid	サンプルプロジェクト
*	

SpeechRecSDK for iOS のファイル構成は以下の通りです。

フォルダ/ファイル	概要
SpeechRecSDK/	
Docs/	
SpeechRecSDK for docomo Developer support 製品仕様書.pdf	SpeechRec SDK の製品仕様書
SpeechRecSDK for docomo Developer support 開発ガイド.pdf	アプリ開発時に必要となるライブラリの配置方法や API の使用方法など
Libs/	
libSRClientHelper.a	SpeechRec Client (SRC) Cocoa Touch Static Library

Include/	
SRClientHelper.h	インクルードヘッダファイル
SRClientDataClasses.h	認識結果データクラス用ヘッダファイル
Resources/	
model_140319.dat	音声区間検出で使用する音声モデル
Sample/	

3

音声認識の流れ

3.1　システム構成

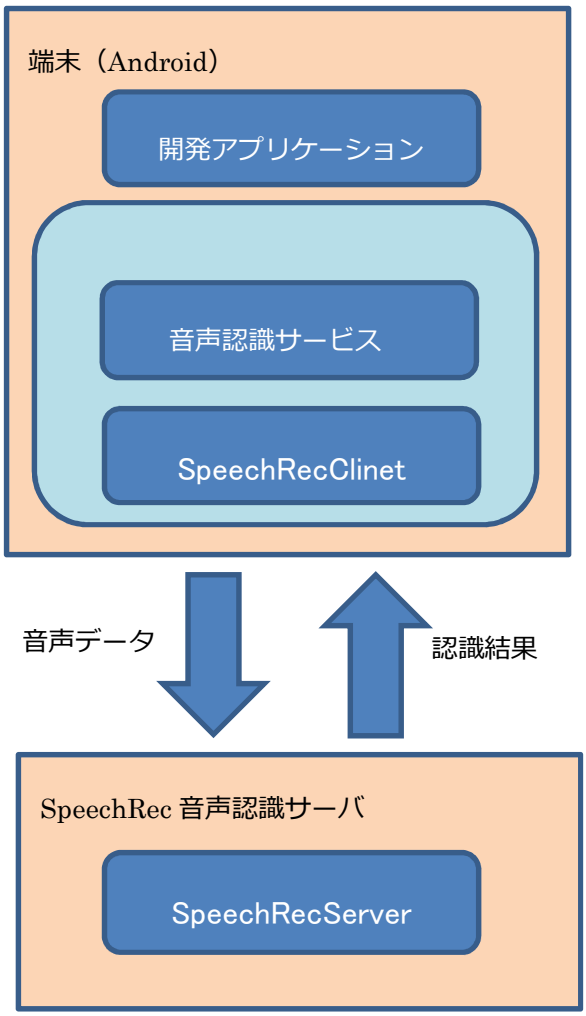
SpeechRec 音声認識システムはユーザが利用する端末と、音声認識処理を行う SpeechRec 音声認識サーバとから構成されます。

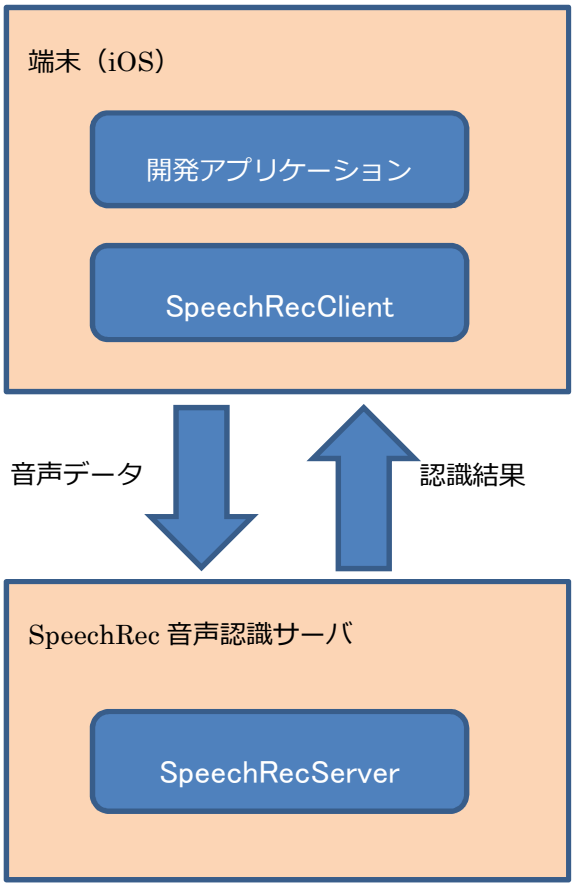
SpeechRecSDK として提供される物には、音声認識を利用する為のインタフェースを提供するライブラリ SpeechRecClient(SRC)、及び SRC を通じて音声認識を行う音声認識サービスが含まれています。

端末に向かってユーザが発話をすると、その音声がインターネットを経由して音声認識サーバに送信されます。

音声認識サーバでは、送信された音声を処理し、認識結果を SpeechRecClinet に返却します。

SpeechRecSDK を利用して作成したアプリケーションは音声認識サービスから結果を取得し、ユーザに対してその結果などを表示します。


端末（Android）
開発アプリケーション
音声認識サービス
SpeechRecClinet
音声データ
認識結果
SpeechRec 音声認識サーバ
SpeechRecServer



3.2　動作の流れ

基本的な動作の流れは以下のようになります。

動作	概要
初期化	音声認識サービスを利用するための初期化処理を行います。
オプションの設定	音声区間を検出するためのモードを設定します。
認識の開始	認識の開始要求を行います。 認識の開始要求を受けると、音声認識サービスはマイクデ

	バイスから音声の取得を始めます。 また、SpeechRec 音声認識サーバへのセッションを確立し、発話開始を検出すると音声の送信を始めます。
発話の終了/中止	認識の開始後、発話の中止、または停止ができます。 発話を終了した場合、認識結果が返却されます。 発話を中止した場合には、認識結果を取得せずに復帰します。 発話の終了条件は、 ・音声区間検出機能が発話区間の終端を検出した場合 ・ユーザ（アプリケーション）が明示的に終了メソッドを呼び出した場合 のいずれかです。 発話の中止条件は ・音声区間検出機能が、一定時間以上発話区間の始端を検出できない場合 ・音声区間検出機能が、一定時間以上発話区間の終端を検出できない場合 ・ユーザ（アプリケーション）が明示的に中止メソッドを呼び出した場合 のいずれかです。
認識結果の取得	発話が終了すると認識結果を取得することができます。
終了	音声認識サービスの終期化処理を行います。

4

開発環境の準備[Android 編]

4.1　ビルド環境の準備

SpeechRec SDK 開発では、以下の開発環境を準備して下さい。

4.1.1　JDK (Java Development Kit)

動作環境を参考に、下記のサイトから JDK をダウンロードし、インストールして下さい。

http://www.oracle.com/technetwork/java/javase/downloads/index.html

4.1.2　ADT Bundle

動作環境を参考に、下記のサイトから ADT（Android Development Tools）Bundle をダウンロードし、インストールして下さい。

http://developer.android.com/sdk/index.html

4.1.3　Pleiades プラグイン

Eclipse でのメニューや表示されるメッセージを日本語化したい場合は、下記のサイトから Pleiades プラグインをダウンロードし、インストールして下さい。

http://mergedoc.sourceforge.jp/

4.2　ライブラリの配置

ライブラリファイルをアプリケーションに組み込むための方法について説明します。

1. Eclipse で新規プロジェクトを作成し、プロジェクトフォルダ配下にライブラリファイル「jar ファイル」、「so ファイル」を配置して下さい。

   libs¥android-websockets.jar

   libs¥sbm.jar

   libs¥speechrec-audio-codec.jar

   libs¥speechrec-client.jar

   libs¥speechrec-service.jar

   libs¥armeabi-v7a¥libaudiocodec.so

   libs¥armeabi-v7a¥libsbm.so

2. プロジェクトのビルドパス設定（ライブラリ）（「プロパティ」-「Java のビルドパス」-「ライブラリ」タブ）に上記で配置した「jar ファイル」を追加して下さい。

3. Android Manifest の「許可」タブ（Android Manifest Permissions）で、Uses Permission として以下の 2 つを定義して下さい。

   - android.permission.INTERNET
   - android.permission.RECORD_AUDIO

4.3　assets ファイルの配置

assets ファイルをアプリケーションに組み込むための方法について説明します。

1. プロジェクトフォルダ配下に assets ファイル「model_140319」を配置して下さい。

   assets¥divide¥model_140319

5

開発環境の準備（iOS 編）

5.1　ビルド環境の準備

SpeechRec SDK 開発では、以下の開発環境を準備して下さい。

5.1.1　Xcode

Mac App Store より，Xcode をインストールして下さい。

以下手順は Xcode6 を基準としています。

5.2　プロジェクトの設定

5.2.1　ライブラリの配置

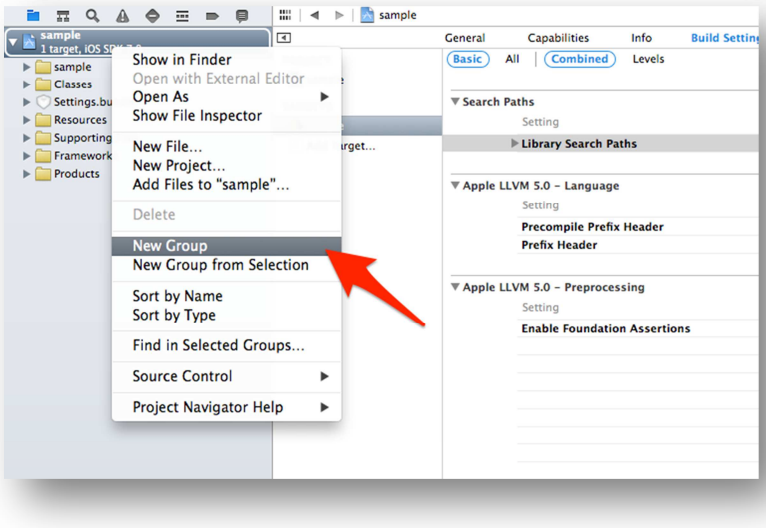
ライブラリファイル等の必要なファイルを，アプリケーションに組み込みます。

配置が必要なファイルは以下の通りです。

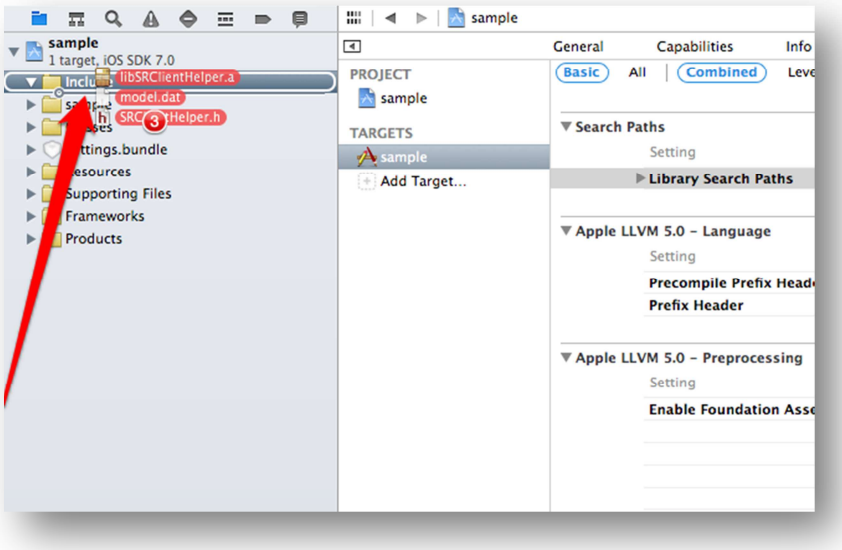
ファイル名	説明
libSRClientHelper.a	SpeechRec Client (SRC) Cocoa Touch Static Library
model_140319.dat	音声区間検出で使用する音声モデル
SRClientHelper.h	SpeechRec Client (SRC) ヘッダファイル
SRClientDataClasses.h	認識結果データクラス用ヘッダファイル

配置方法を以下に記します。

4.　任意のグループを作成し，以下のファイルをグループにドラッグします。



5. 作成したグループ（ここでは「Include」を作成）に，ファイルをドラッグ＆ドロップします。



6. 「Choose options for adding these files」というダイアログが表示されるので，以下のように設定をして「OK」をクリックしてください。

・Copy items into destination group's folder(if needed)

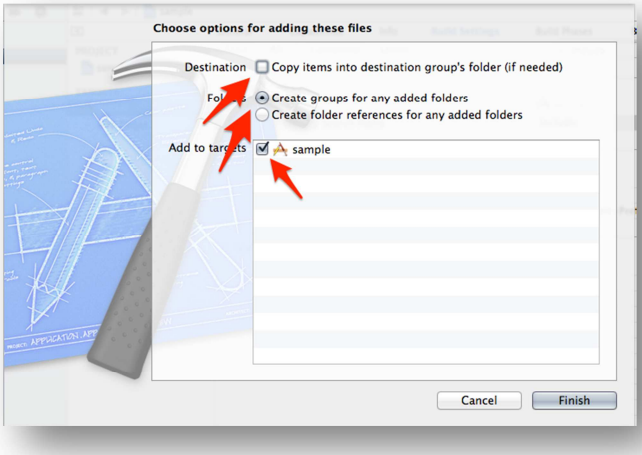
→チェックをオフ

・Folders

→「Create groups for any added folders」を選択

・Add to targets

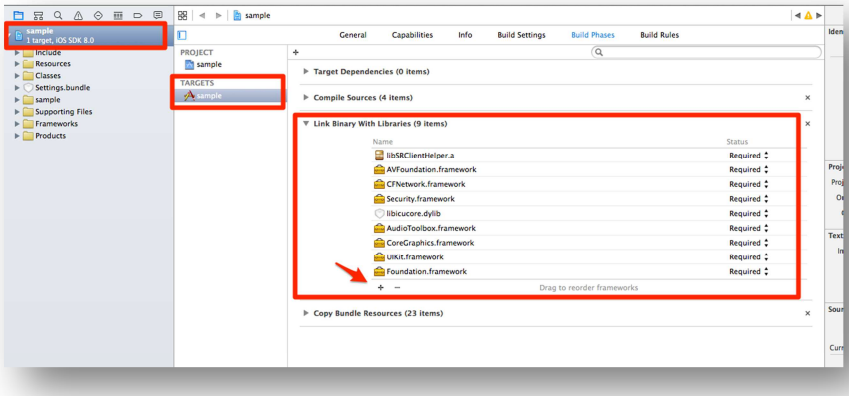
→対象のターゲットにチェックする



5.2.2　フレームワークの設定

アプリケーションのターゲットを選択し，「Build Phases」-「Link Binary With Libraries」にて，以下のアプリケーションの動作に必要なフレームワーク・ライブラリを「+」ボタンを押して追加します。

・libSRClientHelper.a
・AVFoundation.framework
・CFNetwork.framework
・Security.framework
・libicucore.dylib
・AudioToolbox.framework
・CoreGraphics.framework
・UIKit.framework
・Foundation.framework



6

API の利用方法[Android 編]

6.1　処理概要

SpeechRec SDK で提供する音声認識サービスは、利用者が開発するアプリケーションからサービスへの接続により操作可能です。

6.1.1　初期化

1. DivideFileManager クラスを使用して、区間検出音声モデルを assets フォルダから端末内の任意のディレクトリに展開します。
2. Connect()メソッドを呼び出し、音声認識サービスに接続します。
3. 接続に成功した場合、onServiceConnected() がコールバックされます。このコールバック後、startRecognition()メソッドが呼び出し可能となります。

以下は、サンプルアプリでの初期化の例です。

```
// Divide関連ファイルを端末内に展開
```

```
DivideFileManager divideFileManager =
    new DivideFileManager(this);
if (!divideFileManager.isExtracted()) {
  try {
    divideFileManager.extract();
  } catch (IOException e) {
    …….
  }
}
// 音声認識サービスと接続
helper.connect(this, this);
```

6.1.2　オプションの設定

1. 設定値格納用の Bunlde を生成します。
2. Bundle に SBM モード、API キーを追加します。
3. Bundle に区間検出音声モデルのファイルパスを追加します。

以下は、サンプルアプリでの SBM モード、API キー設定の例です。[1]

```
public void onClickStart(final View view) {
  int sbm_mode = radioHigh.isChecked() ? 0 : 1;
  Intent intent = new Intent(this, RecognitionActivity.class);
  intent.putExtra(RecognitionActivity.KEY_SBM_MODE, sbm_mode);
  // 別途発行されるAPIキーを設定してください(以下の値はダミーです)
  intent.putExtra(RecognitionActivity.KEY_API_KEY, "123456789");
```

[1] サンプルアプリでは、インテント発行時に SBM モード、API キーを Intent の extras に設定しています。

```
  startActivityForResult(intent, RECOGNIZE_ACTIVITY_REQUEST_ID);
}
```

以下は、サンプルアプリでのオプション設定の例です。

```
bundle = new Bundle();
// Bundleにインテントの値（SBMモード、APIキー）を追加
Intent intent = getIntent();
bundle.putAll(intent.getExtras());
// Bundleに区間検出モデルファイルを追加
bundle.putString(KEY_VAD_MODEL,
                 divideFileManager.getDivideModelPath());
```

6.1.3　認識の開始

1. 設定値を格納したBundleを引数にstartRecognition()メソッドを呼び出し、音声認識処理を開始します。
2. 認識が開始されると、一定間隔（20msec）で音声録音通知onRecord()がコルバックされます。録音した音声データ（16kHz、16bitリニアPCM形式）が引数で渡されますので、アプリケーション側では音声データをファイルとして保存したり、レベルメーターの表示に利用可能です。音声データを利用しない場合は、onRecord()メソッドをオーバーライドする必要はありません。

以下は、サンプルアプリでの認識開始の例です。[2]

```
@Override
public void onServiceConnected() {
  // 音声認識処理を開始
```

[2] 接続成功時にコールバックされるonServiceConnected()メソッド内で音声認識を開始しています。

```
  helper.startRecognition(bundle);
}
```

6.1.4　発話の終了/中止

1. stopRecognition()メソッドを呼び出し、音声認識処理を停止し認識完了通知待ち状態にします。但し、音声区間検出が有効の場合、終端を検知すると自動的に stopRecognition()がコールされますので、アプリケーション側から stopRecognition()を呼ぶ必要はありません。
2. 録音が終了すると、音声録音終了通知 onStopRecording()がコールバックされます。
3. 認識処理を中止したい場合は、close()メソッドを呼び出すことで認識処理は中止可能です。

以下は、サンプルアプリでの発話終了の例です。[3]

```
public void onClickFinishButton(View view) {
  // 音声認識処理を終了
  helper.stopRecognition();
}
```

6.1.5　認識結果の取得

1. stopRecognition()メソッドを呼び出すと、認識結果通知 onResult()がコールバックされます。

[3] 終端を検知すると自動的に stopRecognition()がコールされるので、ここではマイクアイコンに割り当てた終了ボタンクリック時の呼び出しになります。

2. 認識結果は Nbest クラスに格納されて引数で渡されますので、Nbest クラスのメソッドを使い、認識結果を取得します。

以下は、サンプルアプリでの認識結果取得の例です。

```
@Override
public void onResult(Nbest result) {
  List<Sentence> sentenceList = result.getSentenceList();
  if (resultList == null) {
    resultList = new LinkedList<StringBuilder>();
    for (int i = 0; i < sentenceList.size(); i++) {
      resultList.add(new StringBuilder());
    }
  } else {
    while (resultList.size() > sentenceList.size()) {
      resultList.removeLast();
    }
  }
  for (int i = 0; i < resultList.size(); i++) {
    StringBuilder sb = resultList.get(i);
    Sentence sentence = sentenceList.get(i);
    for (Word word : sentence.getWordList()) {
      String label = word.getLabel();
      // ラベルがnullは無視
      if (label == null) {
        continue;
      }
      // セミコロン以降、空白のみは無視
      label = label.replaceAll(";.*", "").replaceAll("[   ]", "");
      // ラベルが空文字列は無視
      if (label.length() == 0) {
```

```
        continue;
      }
      // 音声認識結果に追加
      if ((separator != null) && (sb.length() > 0)) {
        sb.append(separator);
      }
      sb.append(label);
    }
  }
}
```

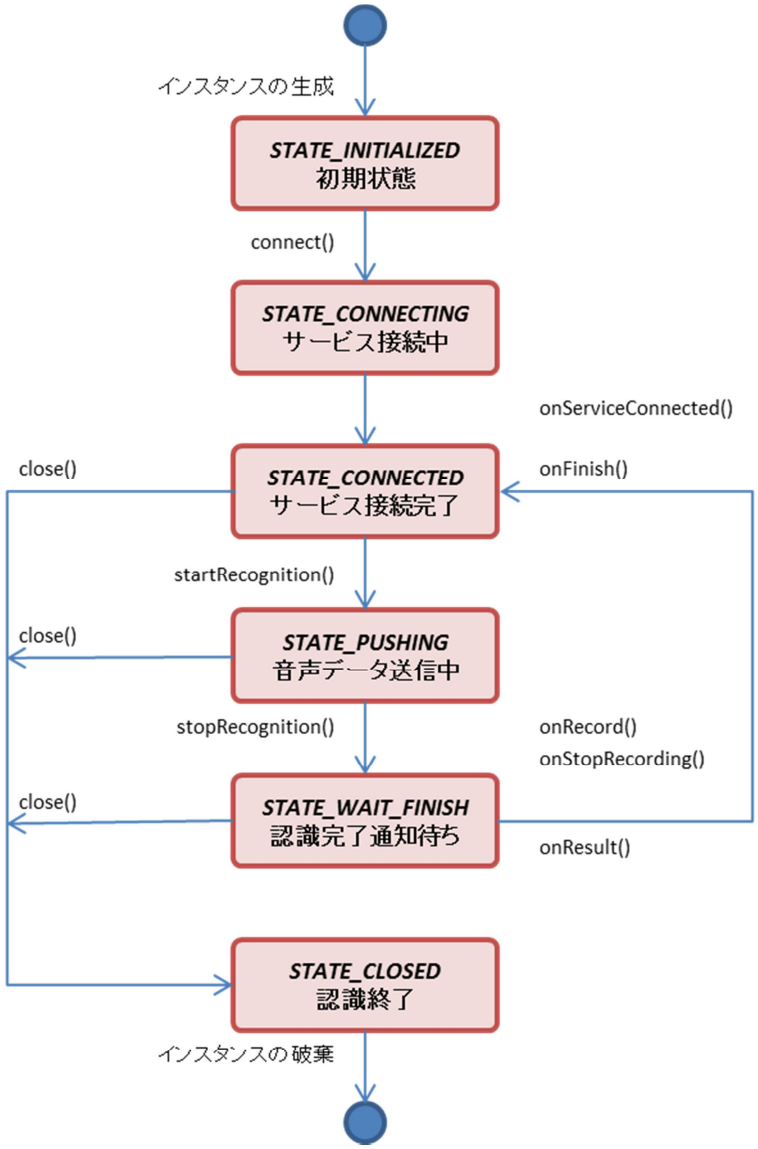
6.1.6　終了

1. 音声認識が完了すると、音声認識完了通知 onFinish()がコールバックされます。このコールバック後、再び startRecognition()メソッドが呼び出し可能となります。
2. 再び認識を開始しない場合は、close()メソッドを呼び出し音声認識サービスとの接続を切断します。close()メソッドは、connect()成功後であればどの時点でも呼び出し可能です。

以下は、サンプルアプリでの終了の例です。

```
@Override
protected void onDestroy() {
  // 音声認識サービスとの接続を切断
  helper.close();
  super.onDestroy();
}
```

6.2　状態遷移


インスタンスの生成
STATE_INITIALIZED
初期状態
connect()
STATE_CONNECTING
サービス接続中
onServiceConnected()
close()
STATE_CONNECTED
サービス接続完了
onFinish()
startRecognition()
close()
STATE_PUSHING
音声データ送信中
stopRecognition()
onRecord()
onStopRecording()
close()
STATE_WAIT_FINISH
認識完了通知待ち
onResult()
STATE_CLOSED
認識終了
インスタンスの破棄

7

API の利用方法（iOS 編）

7.1　処理概要

SpeechRec SDK で提供する音声認識サービスは、利用者が開発するアプリケーションからサービスへの接続により操作可能です。

7.1.1　初期化

4. API キーを設定します。
5. initWithDevice メソッドにてインスタンスの作成を行います。

7.1.2　認識の開始

3. start メソッドを呼び出し、音声認識処理を開始します。
4. マイクデバイスの初期化を行い，音声データの取得が開始されます。
5. SpeechRec 音声認識サーバに接続し，認証を行います。Speechrec 音声認識

サーバとのセッションが確立されると,デリゲートメソッド(srcDidReady)が通知されます。

6. マイクデバイスから取得された音声データは，順次音声区間検出が行われ，音声区間の始端が検出されると SpeechRec 音声認識サーバへ音声データの送信を行います。

7. 認識が開始されると，デリゲートメソッド（srcDidRecord）が通知されます。録音した音声データ（16kHz、16bit リニア PCM 形式）が引数で渡されますので、アプリケーション側では音声データをファイルとして保存したり,レベルメーターの表示に利用可能です。音声データを利用しない場合は、デリゲートメソッド（srcDidRecord）を定義する必要はありません。

7.1.3　発話の終了/中止

4. stop メソッドを呼び出すと，音声区間の終端が検出される前に音声認識処理を停止し，認識結果通知待ち状態にすることができます。通常は終端を検知するとデリゲートメソッド（srcDidSentenceEnd）が通知され自動的に認識結果通知待ち状態にシフトしますので，アプリケーション側から stop メソッドを呼ぶ必要はありません。

5. 認識処理を中止したい場合は、cancel メソッドを呼び出すことで認識処理は中止可能です。

7.1.4　認識結果の取得

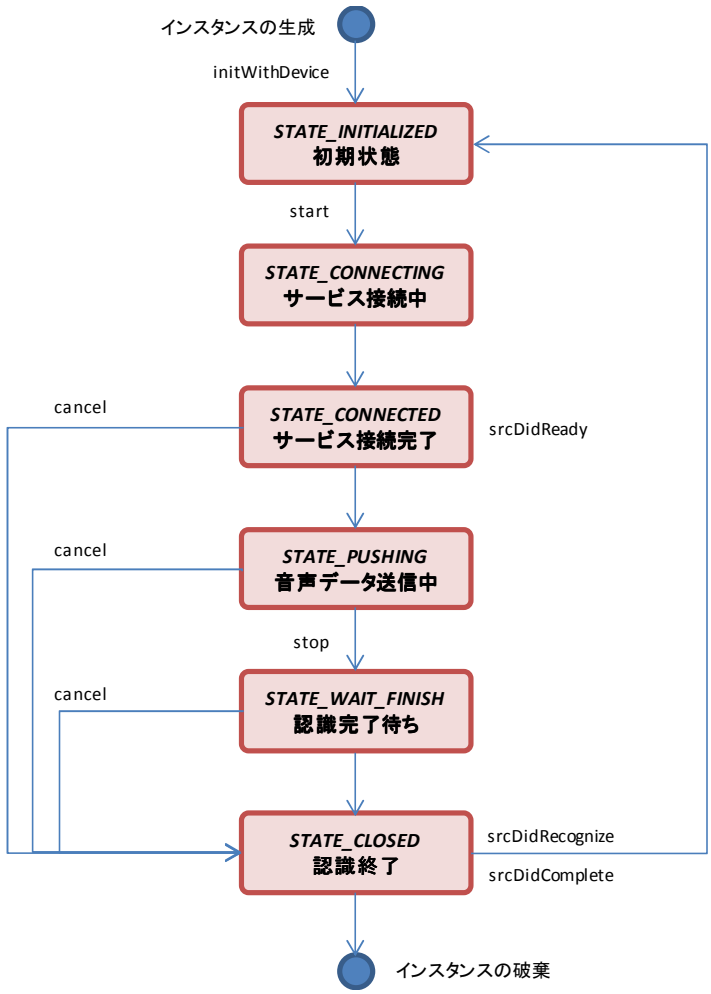
3. デリゲートメソッド(srcDidRecognize)により,認識結果が通知されます。

4. 認識結果は Nbest クラスに格納された状態で引数で渡されます。取得用メソッドを使って結果を取得できます。

7.1.5　終了

3.　音声認識が完了すると，デリゲートメソッド（srcDidComplete）が通知されます。この通知後、再び start メソッドの呼び出しが可能となります。

7.2　状態遷移



8

API リファレンス[Android 編]

8.1　音声認識サービスヘルパークラス

SpeechRecServiceHelper

音声認識サービス（SpeechRecService クラス）の仕様を簡略化するヘルパークラスです。

8.1.1　音声認識サービスとの接続

音声認識サービスに接続します。

```
void connect(Context context,
             VoiceRecognitionEventListener callback)
```

パラメータ	context	アプリケーションコンテキスト

	callback	イベントリスナ
例外	NullPointerException	引数に null が指定された場合
	IllegalStateException	既に接続済みか close()メソッド呼び出し後だった場合
解説	接続に成功した場合、 VoiceRecognitionEventListener.onServiceConnected() がコールバックされます	

8.1.2 音声認識の開始

音声認識を開始します。

```
void startRecognition(Bundle parameter)
```

パラメータ	parameter	音声認識処理のパラメータを格納した Bundle
例外	IllegalStateException	接続済みでない、既に開始済み、close()メソッド呼び出し後のいずれかの場合
解説	音声区間検出が無効の場合、 SpeechRecServiceHelper.stopRecognition() を呼び手動で終了しなければなりません。音声区間検出が有効の場合、自動で認識か完了し VoiceRecognitionEventListener.onFinish(Exception)がコールバックされます。 この場合でも SpeechRecServiceHelper.stopRecognition() を呼び手動で終了することができます。	

8.1.3　音声認識の停止

音声認識を停止します。

```
void stopRecognition()
```

パラメータ	-
例外	IllegalStateException　音声認識を開始していない場合
解説	音声認識を停止し、音声認識完了通知待ち状態にします。 本メソッドは音声認識の開始後に呼び出すことができます。

8.1.4　音声認識サービスとの接続の切断

音声認識サービスとの接続を切断します。

```
void close()
```

パラメータ	-
例外	-
解説	本メソッドはconnect()成功後であればどの時点でも 呼び出し可能です。音声認識を中止したい場合は、stopRecognition()メソッドをコールせず、本メソッドをコールする事で音声認識は中止されます。

8.2　音声認識イベントリスナー

VoiceRecognitionEventListener

音声認識サービスヘルパーから音声認識のイベントを受け取るインターフェースです。

8.2.1　音声認識サービス接続通知

音声認識サービスクラスに接続完了した場合に通知されます。

```
void onServiceConnected()
```

パラメータ	-
例外	-
解説	このコールバック後、startRecognition()メソッドが呼び出し可能となります。

8.2.2　音声録音通知

音声を録音した場合に通知されます。

```
void onRecord(short[] samples)
```

パラメータ	samples	16bit, 16kHz, リニア PCM 形式の音声データ
例外	-	

解説	一定間隔(20msec)でコールバックされます。

8.2.3　音声録音終了通知

音声録音が終了した場合に通知されます。

```
void onStopRecording()
```

パラメータ	-
例外	-
解説	マイクデバイスからの音声データ取得スレッドが終了した契機で通知されます。

8.2.4　認識結果通知

音声認識サーバからの認識結果通知です。

```
void onResult(Nbest result)
```

パラメータ	result　　音声認識結果
例外	-
解説	認識結果通知は音声認識の開始から、音声認識完了通知を受信するまでの間に、複数回通知されます。

8.2.5　音声認識完了通知

音声認識サーバからの音声認識完了通知です。

```
void onFinish(SpeechRecException e)
```

パラメータ	e	音声認識でエラーが発生した場合は例外オブジェクトが、成功した場合はnullが渡されます。
例外	-	
解説	このコールバック後、再びstartRecognition()が呼び出し可能となります。	

8.3　候補クラス

Nbest

音声認識結果のうち、候補の内容取得機能を提供します。

音声認識結果クラスは以下のクラス群で構成されます。

- Nbest　（候補クラス）
  - Sentence　（文候補クラス）
    - Word　（単語クラス）

8.3.1　文候補クラスのリスト取得

文候補クラスのインスタンスのリストを取得します。

```
List<Sentence> getSentenceList()
```

パラメータ	-
例外	-
解説	取得可能なリスト数は候補のセンテンス数となります。 取得可能なセンテンス数は音声認識サーバの設定に依存します。

8.4　文候補クラス

Sentence

音声認識結果のうち、文候補の内容取得機能を提供します。

8.4.1　単語クラスのリスト取得

単語クラスのインスタンスのリストを取得します。

```
List<Word> getWordList()
```

パラメータ	-
例外	-
解説	取得可能なリスト数は文候補の単語（形態素）数となります。

8.5　単語クラス

Word

音声認識結果のうち、単語の内容取得機能を提供します。

8.5.1　単語表記取得

単語表記を取得します。

```
String getLabel()
```

パラメータ	-
例外	-
解説	単語表記は以下のフォーマットで表現しています。 「表記;読み;発音;品詞情報」

8.5.2　単語の開始位置取得

単語の開始位置（秒）を取得します。

```
double getStartPoint()
```

パラメータ	-
例外	-
解説	取得値は、SpeechRecServiceHelper#startRecognition()呼び出し後に、最初に設定した音声データ先頭位置を基準としていま

	す。

8.5.3　単語の終了位置取得

単語の終了位置（秒）を取得します。

```
double getEndPoint()
```

パラメータ	-
例外	-
解説	取得値は、SpeechRecServiceHelper#startRecognition()呼び出し後に、最初に設定した音声データ先頭位置を基準としています。

8.6　DIVIDE ファイル管理クラス

DivideFileManager

区間検出機能で使用するファイルを端末内に展開するクラスです。

8.6.1　コンストラクタ

インスタンスを生成します。

```
DivideFileManager(Context context)
```

パラメータ	context　　アプリケーションコンテキスト
例外	-
解説	インスタンスを生成します。

8.6.2　音声モデルパス取得

端末内に展開された音声モデルファイルのパスを取得します。

```
String getDivideModelPath()
```

パラメータ	-
例外	-
解説	端末内に展開された音声モデルファイルの絶対パスをします。

8.6.3　ファイル展開状態取得

端末内にファイルが展開された否かの状態を取得します。

```
boolean isExtracted()
```

パラメータ	-
例外	-
解説	ファイルが展開済みの場合 true、展開されていない場合 false を返します。

8.6.4　ファイル展開

区間検出で使用するファイルを端末内に展開します。

```
void extract() throws IOException
```

パラメータ	
例外	`IOException`　　ファイルの展開に失敗した場合
解説	assetsフォルダに置かれたDIVIDE関連フィアルを端末内に展開します。

9

API リファレンス（iOS 編）

9.1　SRClientHelper クラス

9.1.1　概要

SRClientHelper クラスは，SpeechRecClient の機能を提供するヘルパークラスです。

9.1.2　インスタンスメソッド

initWithDevice

マイク デバイスを有効にしてオブジェクトを初期化します。

```
- (id)initWithDevice:(NSDictionary*)settings
```

パラメータ	settings	初期設定値を保持する NSDictionary オブジェクト
復帰値	初期化済の SRClientHelper オブジェクトを返します。 エラー時には nil を返します。	
宣言	SRClientHelper.h	

delegate

delegate オブジェクトを返します。

```
- (id<SRClientHelperDelegate>)delegate
```

宣言	SRClientHelper.h

setDelegate

delegate オブジェクトを設定します。

```
- (void)setDelegate:(id<SRClientHelperDelegate>)delegate
```

パラメータ	delegate	delegate に設定する SRClientHelperDelegate プロトコルに準拠したオブジェクト
宣言	SRClientHelper.h	

start

イベントハンドル可能な音声認識を開始します。

```
- (void)start
```

宣言	SRClientHelper.h

stop

音声区間検出を終了し，認識結果待ち状態にします。

```
- (void)stop
```

宣言	SRClientHelper.h

cancel

音声認識をキャンセルします。

```
- (void)cancel
```

宣言	SRClientHelper.h

9.2　SRClientHelperDelegate プロトコル

9.2.1　概要

SRClientHelperDelegate プロトコルは，SRClientHelper クラスからの非同期イベントを受け取るためのメソッドが定義されています。

9.2.2　インスタンスメソッド

srcDidReady

SpeechRec 音声認識サーバに接続され，音声認識の準備ができた際に通知されます。

```
- (void)srcDidReady
```

宣言	SRClientHelper.h

srcDidSentenceEnd

音声区間の終端が検出され，認識結果待ち状態になった際に通知されます。

```
- (void)srcDidSentenceEnd
```

宣言	SRClientHelper.h

srcDidRecognize:

音声認識結果を通知します。

```
- (void)srcDidRecognize:(NSData*)data
```

パラメータ	data	認識結果の XML を保持する NSData オブジェクト
宣言	SRClientHelper.h	

srcDidComplete:

音声認識サービスの認識処理が完了した際に通知されます。

```
- (void)srcDidComplete:(NSError*)error
```

パラメータ	error	エラーを伴って終了した場合のみ，エラー情報を保持する NSError オブジェクトが格納される エラーを伴わない場合は nil が設定される
宣言	SRClientHelper.h	
備考	NSError オブジェクトのプロパティにアクセスすることで，以下の情報が取得できます ・code - エラーコード ・domain - エラータイプ ・localizedDescription - エラーメッセージ ・localizedFailureReason	

	- エラーの詳細（未設定のケースもあります） ※詳細は「12.4　エラー一覧」を参照してください

srcDidRecord:

iOS 端末のマイクから音声が入力された場合に都度通知されます

```
- (void)srcDidRecord:(NSData*)pcmData
```

パラメータ	pcmData　16bit, 16kHz, リニア PCM 形式の音声データ
宣言	SRClientHelper.h

9.3　SRNbest クラス

9.3.1　概要

音声認識結果のうち、候補の内容取得機能を提供します。

音声認識結果クラスは以下のクラス群で構成されます。

Nbest　　（候補クラス）

Sentence　　（文候補クラス）

Word　　（単語クラス）

9.3.2　インスタンスメソッド

getNbestStringArray

認識候補を文字列配列で取得します。

```
- (NSArray*)getNbestStringArray:(BOOL)signageOnly
```

パラメータ	signageOnly	表記のみを取り出すかどうか YES: 表記のみを取り出す NO: 表記のほかに，読み・発音・品詞情報も合わせて取り出す（デリミタは「;」）
復帰値	認識候補が格納された NSArray オブジェクトを返します。 エラー時には nil を返します。	
宣言	SRClientDataClasses.h	

serialize

認識候補を XML 形式で取得します。

```
- (NSString*)serialize
```

復帰値	認識候補が格納された XML 形式の NString オブジェクトを返します。 エラー時には nil を返します。
宣言	SRClientDataClasses.h

9.4　SRSentence クラス

9.4.1　概要

音声認識結果のうち、文候補の内容取得機能を提供します。

9.4.2　インスタンスメソッド

getSentenceStringArray

分候補を文字列配列で取得します。

```
- (NSArray*)getSEntenceStringArray:(BOOL)signageOnly
```

パラメータ	signageOnly	表記のみを取り出すかどうか YES: 表記のみを取り出す NO: 表記のほかに，読み・発音・品詞情報も合わせて取り出す（デリミタは「;」）
復帰値	分候補が格納された NSArray オブジェクトを返します。 エラー時には nil を返します。	
宣言	SRClientDataClasses.h	

serialize

文候補を XML 形式で取得します。

```
- (NSString*)serialize
```

復帰値	文候補が格納された XML 形式の NString オブジェクトを返します。 エラー時には nil を返します。
宣言	SRClientDataClasses.h

9.5 SRWord クラス

9.5.1 概要

音声認識結果のうち、単語の内容取得機能を提供します。

9.5.2 インスタンスメソッド

serialize

単語を XML 形式で取得します。

```
- (NSString*)serialize
```

復帰値	単語が格納された XML 形式の NString オブジェクトを返します。 エラー時には nil を返します。
宣言	SRClientDataClasses.h

10

サンプルアプリケーション[Android 編]

10.1　サンプルアプリ概要

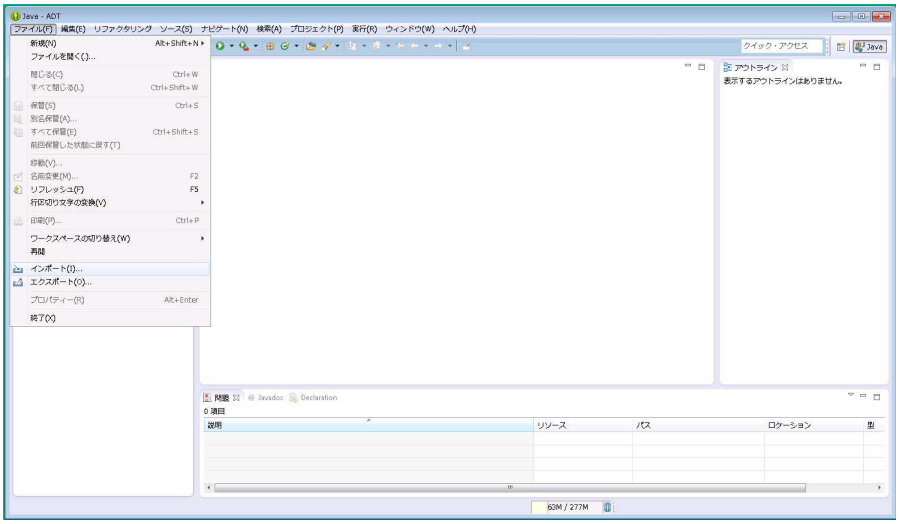
SpeechRec SDK には、音声認識サービスを利用した Android のサンプルアプリが含まれています。

10.2　サンプルアプリビルド方法

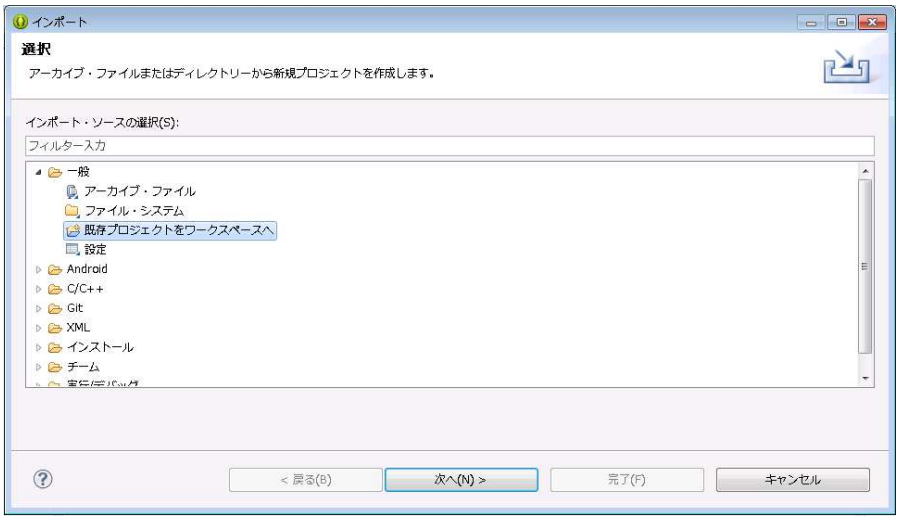
サンプルアプリのビルド方法は次の通りです。

以下、ビルド手順は Eclipse v4.2 での説明となります。

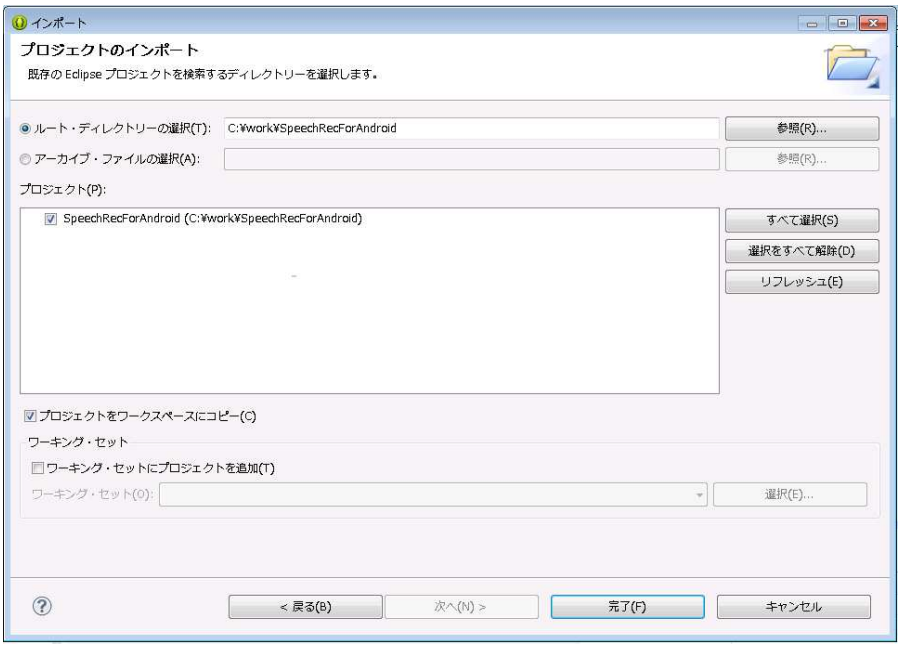
1. Eclipse を起動します。
2. ファイルメニューから「インポート」を選択します。



3. インポート画面の「インポートソースの選択」で「一般」内の「既存プロジェクトをワークスペースへ」を選択し、「次へ」を選択します。



4. インポート画面の「ルートディレクトリの選択」でサンプルアプリのディレクトリを選択し、「完了」を選択します。



5. パッケージビューにインポートしたプロジェクトが追加されます。正常にインポートされない場合には、プロジェクトを削除し、再インポートしてください。

10.3　サンプルアプリ操作方法

サンプルアプリの操作方法は次の通りです。

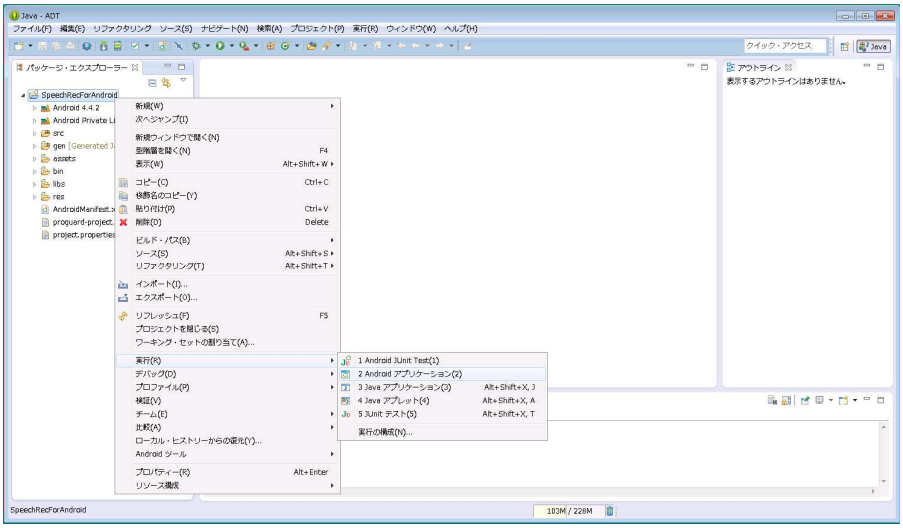
1. Eclipse で Android アプリケーションプロジェクトを読み込みます。

   「11.2　プロジェクトを開く」を参照ください。

2. サンプルアプリをインストールする端末を USB 接続し、接続が確立されたことを確認してください。

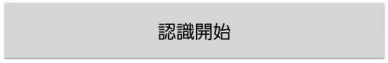
   事前に端末用 USB ドライバと ADB USB ドライバをインストールしておいてください。

3. Eclipse メニューから「実行」を選択してください。サンプルアプリがインストールされ、起動します。



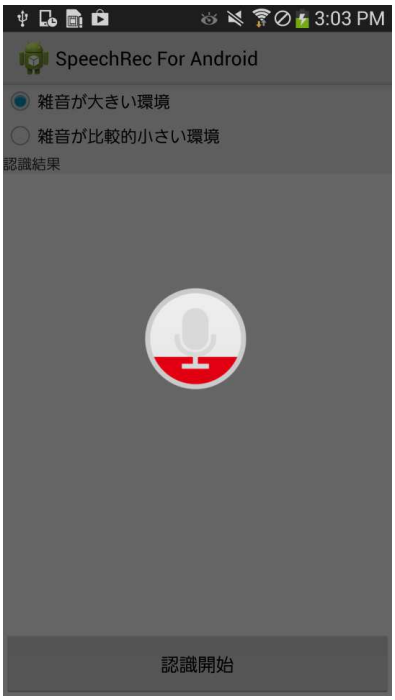
4. 周囲の騒音状況に応じて、「雑音が大きい環境」、「雑音が比較的小さい環境」のいずれかを選択してください。

5. 「認識開始」をタップして、音声認識を開始します。

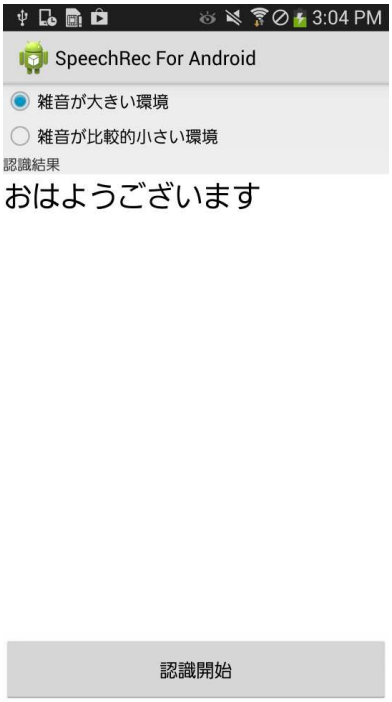


6. 文章や単語をお話しください。

   発話を行うと、音量に応じてレベルメーターが表示されます。



7. 認識結果が表示されます。



8. 再度、音声認識を行う場合は、手順 4 から始めてください。
9. バックキー押下で画面が終了します。

※ご利用上の注意

- ネットワーク経由で音声認識サーバに接続して認識するためパケット通信を行います。
- 会話をするように自然にお話ください。
- 話し方や周囲の騒音状況によっては、お客様が意図しない認識結果となる場合があります。
- 認識できる言語は日本語のみです。
- 音声認識を中止する場合は、バックキー押下で音声認識をキャンセルします。

11

サンプルアプリケーション（iOS 編）

11.1　概要

SpeechRec SDK には、音声認識サービスを利用した iOS のサンプルアプリが含まれています。

11.2　プロジェクトを開く

サンプルアプリのプロジェクトをオープンします。

以下手順は Xcode6 を基準としています。

6. Xcode を起動します。
7. 「File」メニューから「Open...」を選択します。
8. 「sample」フォルダ内の「sample.xcodeproj」を選択し「Open」をクリックしてプロジェクトを開きます。

11.3　ファイル構成

11.3.1　「Include」グループ

ファイル名	説明
libSRClientHelper.a	SpeechRec Client (SRC) Cocoa Touch Static Library
model_140319.dat	音声区間検出で使用する音声モデル
SRClientHelper.h	SpeechRec Client (SRC) ヘッダファイル
SRClientDataClasses.h	認識結果データクラス用ヘッダファイル

11.3.2　「Resource」グループ

ファイル名	説明
speak_now_x.png	マイクアイコンイメージ（レベル 0～16）
none.png	マイクアイコンイメージ（オフライン）
recognizing.png	マイクアイコンイメージ（認識結果待ち）

11.3.3　「Classes」グループ

ファイル名	説明
SRMicPressureLevelButton.h	マイクボタンカスタムクラスヘッダファイル
SRMicPressureLevelButton.m	マイクボタンカスタムクラス実装ファイル

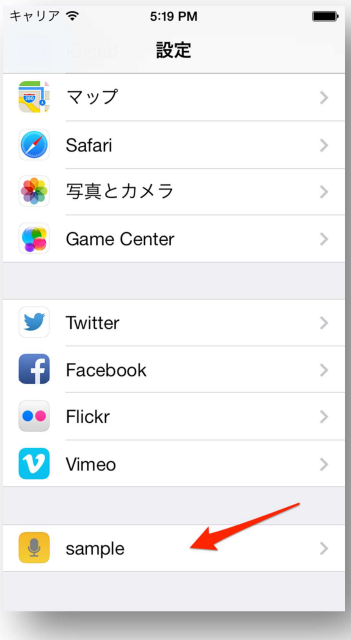
11.4　サンプルアプリ操作方法

11.4.1　アプリケーションの設定

iOS の設定画面より，以下の設定が可能です。

・API キー

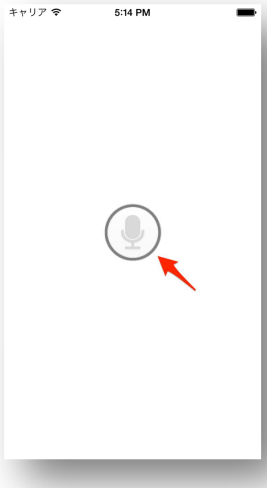
・SBM モード



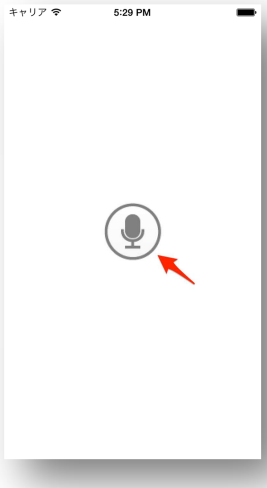
11.4.2　アプリケーションの画面

アプリケーションを起動すると，以下の画面になります。

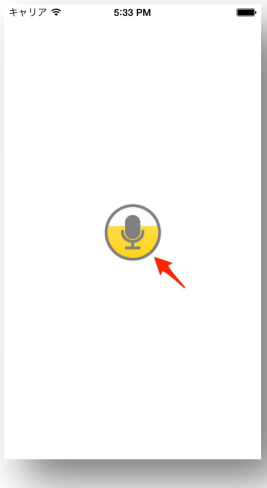
10. マイクアイコンをタップすると，音声認識を開始します。



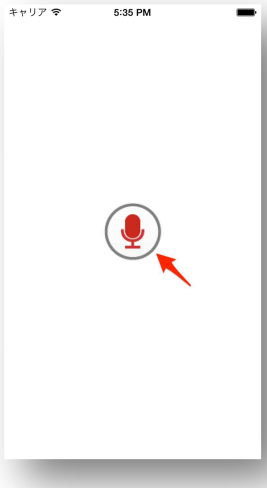
11. マイクアイコンの色が濃くなると音声認識が可能です。
    文章や単語をお話しください。



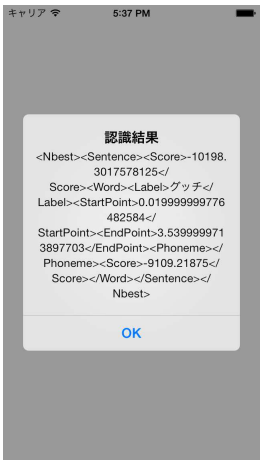
12. 声大きさに応じてレベルメーターが表示されます。



13. 3.の状態でマイクアイコンをタップするか，音声区間の終端が検出されると，マイクアイコンが赤くなり，認識中となります。



14. 認識処理が終了すると，認識結果が表示されます。



15. 1.に戻り繰り返し認識させることが可能です。

11.4.3　ご利用上の注意

- ネットワーク経由で音声認識サーバに接続して認識するためパケット通信を行います。
- 会話をするように自然にお話ください。
- 話し方や周囲の騒音状況によっては、お客様が意図しない認識結果となる場合があります。
- 認識できる言語は日本語のみです。

12

参考

12.1　認識結果 XML

認識結果は XML で構成されます。

XML は入れ子形式で記述されます。入れ子のレベルは下表のとおりで、例えばレベル 2 のタグはレベル 1 の中に記述されることを表します。

タグ名とその意味は以下のようになります。

レベル	タグ名	概要
1	Notification	ルートタグ。
2	Kind	通知種別。システム固有値がセットされます。
2	Target	通知対象。

3	Kind	通知種別。システム固有値がセットされます。
3	Id	通知種別。システム固有値がセットされます。
2	ArgumentList	通知結果。
3	Double	システム通知。システム固有値がセットされます。
3	Nbest	認識結果ブロック。
4	Sentence	認識結果センテンスブロック。認識結果が複数候補返却された場合には、センテンスブロックが複数記述されます。
5	Score	認識結果センテンススコア。サーバの設定により、音声認識エンジン固有の値が記述される場合と、正規化された値が記述される場合とがあります。 エンジン固有値の場合には、値が小さいほど認識結果が妥当であることを表します。 正規化された値の場合には、値が大きいほど認識結果が妥当であることを表します。 正規化された値は0から1までの間の数値になります。
5	Word	認識結果センテンスを構成する単語ブロック。センテンスを構成する単語の数分だけ繰り返し記述されます。
6	Label	単語情報。表記、読み、音素表現、品詞の情報をセミコロンで連結した形式で記述します。

		単語の読みが複数ある場合には、読みカラム内に読みがカンマで複数記述されます。
6	StartPoint	当該ラベルの開始時間を秒で表した数値。起点（ゼロ秒）は入力音声の先頭。
6	EndPoint	当該ラベルの開始時間を秒で表した数値。起点（ゼロ秒）は入力音声の先頭。

12.1.1　XML タグ基本構成

```
<Notification>
    <Kind>XXX</Kind>
    <Target>
        <Kind>XXX</Kind>
        <Id>XXX</Id>
    </Target>
    <ArgumentList>
        <Double>XXX</Double>
        <Nbest>
            <Sentence>
                <Score>XXX</Score>
                <Word>
                    <Label>XXX</Label>
                    <StartPoint>XXX</StartPoint>
                    <EndPoint>XXX</EndPoint>
                </Word>
            </Sentence>
        </Nbest>
    </ArgumentList>
</Notification>
```

※Sentence ブロックは認識候補分、繰り返し記述。

※Word ブロックはセンテンスを構成する単語分、繰り返し記述。

12.1.2　認識結果サンプル

```
<Notification>
    <Kind>GrammarListener::EndPoint</Kind>
    <Target>
        <Kind>Grammar</Kind>
        <Id>00D2E410</Id>
    </Target>
    <ArgumentList>
        <Double>2.75</Double>
        <Nbest>
            <Sentence>
                <Score>-210880640</Score>
                <Word>
                    <Label>休業;キュウギョウ;キュウギョウ;名詞:動
作;</Label>
                    <StartPoint>0.019999999776482584</StartPoint>
                    <EndPoint>0.57999999284744264</EndPoint>
                </Word>
                <Word>
                    <Label>します;シマス;シマス;動詞節;</Label>
                    <StartPoint>0.57999999284744264</StartPoint>
                    <EndPoint>1.2200000381469727</EndPoint>
                </Word>
            </Sentence>
            <Sentence>
                <Score>-210726512</Score>
                <Word>
                    <Label>急病;キュウビョウ;キュウビョウ;名詞;</Label>
                    <StartPoint>0.019999999776482584</StartPoint>
                    <EndPoint>0.57999999284744264</EndPoint>
                </Word>
                <Word>
                    <Label>;、;、;pause;</Label>
                    <StartPoint>0.57999999284744264</StartPoint>
                    <EndPoint>0.68000001668930055</EndPoint>
                </Word>
                <Word>
                    <Label>します;シマス;シマス;動詞節;</Label>
                    <StartPoint>0.68000001668930055</StartPoint>
                    <EndPoint>1.2200000381469727</EndPoint>
                </Word>
            </Sentence>
            <Sentence>
                <Score>-210699632</Score>
                <Word>
                    <Label>急病;キュウビョウ;キュウビョウ;名詞;</Label>
                    <StartPoint>0.019999999776482584</StartPoint>
                    <EndPoint>0.57999999284744264</EndPoint>
                </Word>
                <Word>
```

```
                    <Label>します;シマス;シマス;動詞節;</Label>
                    <StartPoint>0.57999999284744264</StartPoint>
                    <EndPoint>1.2200000381469727</EndPoint>
                </Word>
            </Sentence>
            <Sentence>
                <Score>-210673520</Score>
                <Word>
                    <Label>急病;キュウビョウ;キュウビョウ;名詞;</Label>
                    <StartPoint>0.019999999776482584</StartPoint>
                    <EndPoint>0.57999999284744264</EndPoint>
                </Word>
                <Word>
                    <Label>に;二;二;格助詞:連用;</Label>
                    <StartPoint>0.57999999284744264</StartPoint>
                    <EndPoint>0.69000000715255738</EndPoint>
                </Word>
                <Word>
                    <Label>します;シマス;シマス;動詞節;</Label>
                    <StartPoint>0.69000000715255738</StartPoint>
                    <EndPoint>1.2200000381469727</EndPoint>
                </Word>
            </Sentence>
            <Sentence>
                <Score>-210668000</Score>
                <Word>
                    <Label> キ ュ ビ オ ス ; キ ュ ビ オ ス ; キ ュ ビ オ
ス;Katakana:Undef;</Label>
                    <StartPoint>0.019999999776482584</StartPoint>
                    <EndPoint>0.59999997377395631</EndPoint>
                </Word>
                <Word>
                    <Label>します;シマス;シマス;動詞節;</Label>
                    <StartPoint>0.59999997377395631</StartPoint>
                    <EndPoint>1.2200000381469727</EndPoint>
                </Word>
            </Sentence>
        </Nbest>
    </ArgumentList>
</Notification>
```

12.2　認識オプション（Android）

認識開始時に指定可能なオプションは以下の通りです。

項目		説明	必須
api_key	String	docomo Developer support から払い出される API キーを設定してください	○
sbm_mode	int	環境によって何れかを設定します 0: 雑音が大きい環境 1: 雑音が比較的小さい環境	×
white_list_text	JSON	条件に一致した認識結果のみを得たい場合は，JSON 形式でその条件を指定します 正規表現（regex キー）と文字列完全一致（match キー）での指定が可能です	×

12.3　設定オプション（iOS）

認識開始時に指定可能なオプションは以下の通りです。

項目		説明	必須
api_key	NSString	docomo Developer support から払い出される API キーを設定してください	○

sbm_mode	NSNumber (BOOL)	環境によって何れかを設定します 0: 雑音が大きい環境 1: 雑音が比較的小さい環境	×
white_list	NSDictionary	ホワイトリストを指定します。条件にマッチしたかどうかの情報と共に認識結果を返します。 Dictionary には以下のキー項目の指定が可能です match: マッチさせたい文字列を NSArray 型の配列で複数指定できます regex: 正規表現を NSArray 型の配列で複数指定できます	×

12.4　エラー一覧

SpeechRec SDK で出力されるエラー一覧を以下に示します。

コード	タイプ	メッセージ	原因・対処など
1005	SpeechRec::Client	音声区間検出モード値が不正な値です。	設定オプション「sbm_mode」は 0 もしくは 1 を指定してください。
1007	SpeechRec::Client	音声区間検出モデルが見つかりません。	「5.2.1　ライブラリの配置」を参照し，音声区間検出モデルを配置してください
1008	SpeechRec::Client	音声区間検出モデルが不正です。	もしくは「5.2.1　ライブラリの配置」を参照し, 正しい音声区間検出モデルを配置してください
1011	SpeechRec::Client	その他の認識オプションが不正です。	設定オプションの指定を確認してください。
2000	SpeechRec::Client	状態が不正です。	API の呼出し順序が誤っている可能性があります。「3.2　動作の流れ」を参照し，正しい順序で API を使用してください。
3000	SpeechRec::Client	音声区間の始端が検出できません。	一定時間音声の入力を確認できませんでした。
3001	SpeechRec::Client	音声区間の終端が検出できません。	一定時間内に音声区間の終わりを検出できませんでした。より短いセンテンスで入力してみてください。
3002	SpeechRec::Client	音声認識処理がタイムアウトしました。	一定時間内に音声認識サーバからの応答がありませんでした。 本エラーが頻発するようであれば, サービス提供事業者にお問い合わせください。

4000	SpeechRec::Client	受信したレスポンスが不正です。	本エラーが頻発するようであれば, サービス提供事業者にお問い合わせください。
4001	SpeechRec::Client	受信したレスポンスに期待されたノードが存在しません。	本エラーが頻発するようであれば, サービス提供事業者にお問い合わせください。
4002	SpeechRec::Client	受信したリクエストが不正です。	本エラーが頻発するようであれば, サービス提供事業者にお問い合わせください。
4003	SpeechRec::Client	受信したリクエストに期待されたノードが存在しません。	本エラーが頻発するようであれば, サービス提供事業者にお問い合わせください。
4004	SpeechRec::Client	音声認識サーバが異常な状態です。	本エラーが頻発するようであれば, サービス提供事業者にお問い合わせください。
5000	SpeechRec::Client	音声認識サーバに接続できません。	ネットワーク環境が原因の可能性がありますので，端末の設定を確認してください。
5001	SpeechRec::Client	認証に失敗しました。	API キーが誤っている可能性がありますので, 設定オプション「api_key」の設定を確認してください。 また, ネットワーク環境が原因の可能性もありすので，端末の設定を確認してください。
5002	SpeechRec::Client	通信で異常が発生しました。	ネットワーク環境が原因の可能性がありすので，端末の設定を確認してください。
5003	SpeechRec::Client	パケットの送信に失敗しました。	ネットワーク環境が原因の可能性がありすので，端末の設定を確認してください。
5004	SpeechRec::Client	リクエストがタイムアウトしました。	ネットワーク環境が原因の可能性がありすので，端末の設定を確認してください。
5005	SpeechRec::Client	レスポンスがタイムアウトしました。	ネットワーク環境が原因の可能性がありすので，端末の設定を確認してください。

5006	SpeechRec::Client	コネクションが切断されました。	音声認識サーバで受付可能なセッション数の上限を超えた可能性があります。少し時間をおいてから再度接続してください。 さらに, ネットワーク環境が原因の可能性もありすので，端末の設定を確認してください。
6000	SpeechRec::Client	音声キャプチャデバイスが開始できません。	ご使用の端末のマイクが利用可能か確認してください。
6001	SpeechRec::Client	音声区間検出処理でエラーが発生しました。	本エラーが頻発するようであれば, サービス提供事業者にお問い合わせください。
6002	SpeechRec::Client	エンコード処理でエラーが発生しました。	本エラーが頻発するようであれば, サービス提供事業者にお問い合わせください。
7000	SpeechRec::Client	内部エラーが発生しました。	本エラーが頻発するようであれば, サービス提供事業者にお問い合わせください。
-	-	-	タイプが「SpeechRec::Client」でない場合、音声認識サーバから通知されたエラー内容になりますので、サービス提供事業者にお問い合わせください。

SpeechRecSDK

開発ガイド

発行　　エヌ・ティ・ティ　アイティ株式会社

〒231-0032

神奈川県横浜市中区不老町二丁目 9 番地 1

http://www.ntt-it.co.jp/