

6 テキスト読み上げ処理

6.1 テキスト読み上げの概要

TextToSpeech クラスは Android1.6 から実装された機能で、音声合成を利用してテキストを読み上げる機能です。今回の実習では、温度、湿度、不快指数を読み上げるプログラムを作成します。

Android 端末には英語・ドイツ語・スペイン語・フランス語・イタリア語対応の音声合成エンジン「Pico TTS」が標準装備されています。日本語を使用したい場合は、株式会社 KDDI 研究所の無料日本語音声合成エンジンの「N2 TTS」を入れておけば多くの読み上げアプリで使用できますが今回の実習では使用しません。

TextToSpeech クラスの利用手順としては、次のとおりです。

- ①オブジェクトの生成(初期化)
- ②言語の設定
- ③読み上げをする文章を指定して speak メソッドを呼び出す。

6.2 入力用アクティビティ (DICalculatorActivity) の追記・修正

```
public class DICalculatorActivity extends Activity
{
    . . .
    private Intent intent;//インテント
    private TextToSpeech tts;//テキスト読み上げ
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);//レイアウトの表示
        tts = new TextToSpeech(getApplicationContext(),
            new SampleInitListener());//テキスト読み上げ
        tts.setLanguage(Locale.JAPANESE);//テキスト読み上げ言語
        . . .
    }
    class calculateClickListener implements OnClickListener
    {
        public void onClick(View view)
        {
            if(view == button)
            {
                calculate();//不快指数の計算
                TextToSpeech();//テキスト読み上げ
                . . .
            }else if(view == buttonShowNextActivity)
            {
                calculate();//不快指数の計算
                TextToSpeech();//テキスト読み上げ
                . . .
            }
        }
    }
    class calculateDialogClickListener
        implements DialogInterface.OnClickListener
    {

```

```

        . . .
    }
}
//不快指数の計算
void calculate()
{
    . . .
}
//テキスト読み上げ
void TextToSpeech()
{
    String strTemperature = texttemperature.getText().toString();
    String strHumidity = texthumidity.getText().toString();
    String strDi = String.valueOf(di);
    if(strTemperature != null)
    {
        tts.speak("temperature is " + strTemperature,
                  TextToSpeech.QUEUE_FLUSH, null);
    }
    if(strHumidity != null)
    {
        tts.speak("humidity is " + strHumidity,
                  TextToSpeech.QUEUE_ADD, null);
    }
    if(strDi != null)
    {
        tts.speak("discomfort index is " + strDi,
                  TextToSpeech.QUEUE_ADD, null);
    }
}
}
class SampleInitListener implements OnInitListener
{
    public void onInit(int status) {}
}
}

```

6.3 オブジェクトの生成(初期化)

```

private TextToSpeech tts;//テキスト読み上げ

tts = new TextToSpeech(getApplicationContext(),
                        new SampleInitListener());//テキスト読み上げ

class SampleInitListener implements OnInitListener
{
    public void onInit(int status) {}
}

```

コンストラクタにコンテキストと初期化完了リスナ(TextToSpeech.OnInitListener)を指定します。

TextToSpeech(Context context, TextToSpeech.OnInitListener listener)

onInit メソッドをオーバーライドして初期化完了のイベントを受け取ります。

void onInit(int status)

status は成功の場合 : TextToSpeech.SUCCESS が、失敗の場合 : TextToSpeech.ERROR が

渡されます。

6.4 言語（ロケール）の設定

```
tts.setLanguage(Locale.JAPANESE); // テキスト読み上げ言語
```

初期化が成功した場合、次にロケールを設定します。

isLanguageAvailable メソッドで任意のロケールが利用可能か確認して、setLanguage メソッドでロケールを設定します。

```
int isLanguageAvailable(Locale locale)
```

戻り値が TextToSpeech.LANGUAGEAVAILABLE 以上の値であれば利用可能ですので、

```
int setLanguage(Locale locale)
```

で設定します。こちらの戻り値も isLanguageAvailable と同様に利用可能かどうかの値が返ってきます。

ロケール 【 locale 】

ソフトウェアに内蔵される、言語や国・地域ごとに異なる単位、記号、日付、通貨などの表記規則の集合。または単に、利用する言語や国・地域の指定。

6.5 テキストの読み上げ

```
tts.speak("temperature is " + strTemperature,
        TextToSpeech.QUEUE_FLUSH, null);

tts.speak("humidity is " + strHumidity, TextToSpeech.QUEUE_ADD, null);

tts.speak("discomfort index is " + strDi, TextToSpeech.QUEUE_ADD, null);
```

実際にテキストの読み上げを行うには下記のメソッドを呼び出します。

```
int speak(String text, int queueMode, HashMap<String, String> params)
```

キューモードは以下の通りです。

TextToSpeech.QUEUE_ADD: 再生キューへエントリを追加

TextToSpeech.QUEUE_FLUSH: 再生待ちのエントリをドロップしてエントリを実行

パラメータは実行後に呼びされるメソッドの引数に渡す値などが設定できますが不要であれば null を指定することも可能です。