CHAPTER 7

Working with Speech

In the previous chapter, you learned how to use micro:bit's audio capabilities to produce music using the music library. In addition to that, micro:bit provides a speech library to work with text-to-speech conversion that can be used to produce sound similar to the human voice by way of fine tuning various parameters.

Connecting a Speaker

You can use the same wiring diagram that you used in Chapter 6, "Working with Music," to connect a speaker to the micro:bit. Instead of connecting a speaker to the micro:bit pin 0 and GND, you can use the micro:bit's pins 0 and 1 to connect a speaker, as shown in Figure 7-1.

P. Seneviratne, *Beginning BBC micro:bit*, https://doi.org/10.1007/978-1-4842-3360-3_7

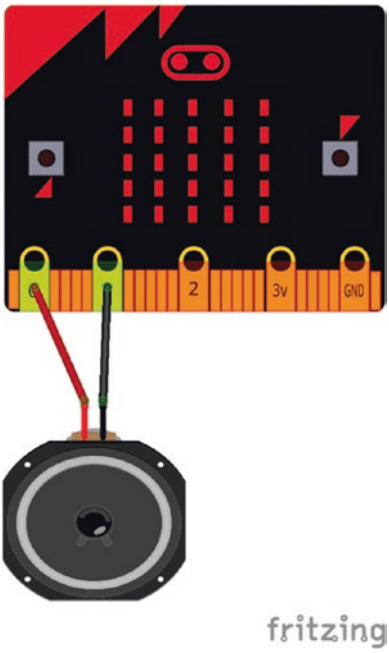

Figure 7-1. *Connecting a speaker to pins 0 and 1*

The speech library provides all the functionalities needed to work with speech and voice related projects. You can import the speech library by adding the `import speech` statement to the beginning of your program.

Let's start with simple code used to convert text to speech (see Listing 7-1). It converts the text `Hello, World` to speech and you can hear it from the speaker.

Listing 7-1. Text to Speech Conversion

```
from microbit import *
import speech

speech.say("Hello, World")
```

The `speech.say()` function converts English text to speech and plays it from the attached speaker. When you run this code with micro:bit, you can hear a voice similar to a robot, which is semi-accurate in English. The quality of the voice is not amazing, but it is quite usable. In addition, the `speech.say()` function provides some parameters that you can use to change the default voice.

Timbre

The character or quality of a musical sound or voice is known as its *timbre*. You can change the quality of the default voice by overriding some of the parameters that the speech synthesizer uses to produce it.

Pitch

The pitch defines how high or low the voice sounds. The acceptable values are 0 (high) to 255 (low). You can get a clue about the pitch by hearing the vocals of following singers.

- Highest pitch: Adam Lopez Costa at `https://www.youtube.com/results?search_query=Adam+Lopez+Costa`
- Lowest pitch: Barry White at `https://www.youtube.com/results?search_query=Barry+White`

The default pitch value is 64. Listing 7-2 shows a list of value categories that you can use to set the pitch of the voice.

Listing 7-2. Pitch Categories and Values

```
0-20 impractical
20-30 very high
30-40 high
40-50 high normal
```

```
50-70 normal
70-80 low normal
80-90 low
90-255 very low
```

Listing 7-3 shows the MicroPython code that produces a voice with different pitch categories. The code uses average values in each category.

Listing 7-3. Pitch Levels

```
from microbit import *
import speech

speech.say("Hello, World")#default pitch is 64
sleep(1000)
speech.say("Hello, World", pitch=10)# impractical
sleep(1000)

speech.say("Hello, World", pitch=25)# very high
sleep(1000)

speech.say("Hello, World", pitch=35)# high
sleep(1000)

speech.say("Hello, World", pitch=45)# high normal
sleep(1000)

speech.say("Hello, World", pitch=60)# normal
sleep(1000)

speech.say("Hello, World", pitch=75)# low normal
sleep(1000)

speech.say("Hello, World", pitch=85)# low
sleep(1000)

speech.say("Hello, World", pitch=170)# very low
```

Speed

Speed defines how quickly the device talks. The acceptable values are from 0 (impossible) to 255 (like bedtime story). The default value is 72. Listing 7-4 shows a list of categories and values to define the speed.

Listing 7-4. Speed Categories and Values

```
0-20 impractical
20-40 very fast
40-60 fast
60-70 fast conversational
70-75 normal conversational
75-90 narrative
90-100 slow
100-225 very slow
```

Listing 7-5 shows the code that speaks the text `Hello, World` in different speeds.

Listing 7-5. Speak with Different Speeds

```
from microbit import *
import speech

speech.say("Hello, World")#default speed is 72
sleep(1000)
speech.say("Hello, World", speed=10) # impractical
sleep(1000)
speech.say("Hello, World", speed=30) # very fast
sleep(1000)
speech.say("Hello, World", speed=50) # fast
sleep(1000)
speech.say("Hello, World", speed=65) # fast conversational
sleep(1000)
```

```
speech.say("Hello, World", speed=73) # normal conversational
sleep(1000)
speech.say("Hello, World", speed=83) # narrative
sleep(1000)
speech.say("Hello, World", speed=95) # slow
sleep(1000)
speech.say("Hello, World", speed=175) # very slow
sleep(1000)
```

Mouth

Mouth defines how tight-lipped or overtly enunciating the voice sounds (0 = tight-lipped, 255 = Foghorn Leghorn).

- Tight-lipped: The most extreme example of this is a ventriloquist, which is a person who changes his or her voice so that it appears that the voice is coming from elsewhere.
- Overtly enunciating: A good example of this is Foghorn Leghorn, who was a cartoon character that has appeared in the Looney Tunes and Merrie Melodies cartoons of Warner Bros. (See https://www.youtube.com/results?search_query=Foghorn+Leghorn.)

Listing 7-6 shows some sample code with the `mouth` parameter.

Listing 7-6. Controlling the Mouth Parameter

```
from microbit import *
import speech

speech.say("Hello, World", mouth=200)
```

Throat

Throat defines how relaxed or tense the tone of voice is (0 = falling apart, 255 = totally chilled).

Listing 7-7 shows some sample code with the `throat` parameter.

Listing 7-7. Controlling the Throat Parameter

```
from microbit import *
import speech

speech.say("Hello, World", throat=100)
```

Example: Creating a Robotic Voice

The default voice produced by a speech synthesizer can be tuned with the parameters just discussed (pitch, speed, mouth, and throat) to produce a robotic voice.

Listing 7-8 shows some sample code that can be used to produce a voice similar to a robot. The `speech.say()` function combines all the given parameters to produce the voice for the given text.

Listing 7-8. Voice of a Robot

```
from microbit import *
import speech

speech.say("I am a baker bot", speed=120, pitch=100,
throat=100, mouth=200)
```

Punctuation

Punctuation makes a voice more realistic. With a speech library, you can use five types of punctuation to alter the delivery of speech. They are as follows:

- *Hyphen*: Creates a short pause in the speech.

  ```
  speech.say("I am a baker bot - crazy cooking",
  speed=120, pitch=100, throat=100, mouth=200)
  ```

- *Comma:* Adds a pause of approximately double that of the hyphen.

  ```
  speech.say("I am a baker bot, crazy cooking",
  speed=120, pitch=100, throat=100, mouth=200)
  ```

- *Full stop:* Creates a pause and causes the pitch to fall.

  ```
  speech.say("I am a baker bot - crazy cooking.",
  speed=120, pitch=100, throat=100, mouth=200)
  ```

- *Question mark:* Creates a pause and causes the pitch to rise.

  ```
  speech.say("I am a baker bot. Who are you?", speed=120,
  pitch=100, throat=100, mouth=200)
  ```

Phonemes

Phonemes can be used to translate English words into the correct sounds. They are the building blocks of language. The `speech.pronounce()` function allows you to translate any phoneme into the correct voice in English.

An example, the word `Hello` can be written with phonemes as `/HEHLOW`. Listing 7-9 shows the MicroPython code used to produce the voice using phonemes.

Listing 7-9. Phonemes

```
from microbit import *
import speech

speech.pronounce("/HEHLOW")  # "Hello"
```

You can convert any English text to a string of phonemes using the `speech.translate()` function (see Listing 7-10). Then you can fine tune the phonemes to produce a more natural voice.

Listing 7-10. Translate the Text to Phonemes

```
from microbit import *
import speech

print(speech.translate("Hello"))
```

The following table lists the phonemes understood by the synthesizer (Source: http://microbit-micropython.readthedocs.io/en/latest/speech.html).

```
SIMPLE VOWELS                          VOICED CONSONANTS
IY            f(ee)t                   R          (r)ed
IH            p(i)n                    L          a(ll)ow
EH            b(e)g                    W          a(w)ay
AE            S(a)m                    W          (wh)ale
AA            p(o)t                    Y          (y)ou
AH            b(u)dget                 M          (S)am
AO            t(al)k                   N          ma(n)
OH            c(o)ne                   NX         so(ng)
UH            b(oo)k                   B          (b)ad
UX            l(oo)t                   D          (d)og
ER            b(ir)d                   G          a(g)ain
AX            gall(o)n                 J          (j)u(dg)e
IX            dig(i)t                  Z          (z)oo
                                       ZH         plea(s)ure
DIPHTHONGS                             V          se(v)en
EY            m(a)de                   DH         (th)en
AY            h(igh)
OY            b(oy)
AW            h(ow)                    UNVOICED CONSONANTS
OW            sl(ow)                   S           (S)am
UW            cr(ew)                   SH          fi(sh)
                                       F           (f)ish
                                       TH          (th)in
SPECIAL PHONEMES                       P           (p)oke
UL            sett(le) (=AXL)          T           (t)alk
UM            astron(om)y (=AXM)       K           (c)ake
UN            functi(on) (=AXN)        CH          spee(ch)
Q             kitt-en (glottal stop)   /H          a(h)ead
```

Here is a list of non-standard symbols:

```
YX            diphthong ending (weaker version of Y)
WX            diphthong ending (weaker version of W)
RX            R after a vowel (smooth version of R)
LX            L after a vowel (smooth version of L)
/X            H before a non-front vowel or consonant - as in (wh)o
DX            T as in pi(t)y (weaker version of T)
```

Here is a list of some seldom used phoneme combinations:

```
PHONEME         YOU PROBABLY WANT:     UNLESS IT SPLITS SYLLABLES LIKE:
COMBINATION
GS              GZ e.g. ba(gs)         bu(gs)pray
BS              BZ e.g. slo(bz)        o(bsc)ene
DS              DZ e.g. su(ds)         Hu(ds)son
PZ              PS e.g. sla(ps)        -----
TZ              TS e.g. cur(ts)y       -----
KZ              KS e.g. fi(x)          -----
NG              NXG e.g. singing       i(ng)rate
NK              NXK e.g. bank          Su(nk)ist
```

Using lmtool

lmtool (see http://www.speech.cs.cmu.edu/tools/lmtool-new.html) provides an easy way to convert English text to phonemes. Use the following steps to convert a text file to a Pronunciation Dictionary file using lmtool.

1. Using a text editor, create a file with your sentence (or many sentences) in English. You should include at least two words in your file, otherwise the compilation will fail. Then, save the file on your computer (see Figure 7-2).

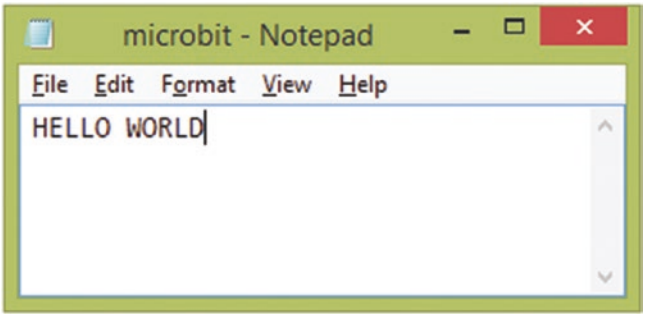

Figure 7-2. *Source file with text*

2. Browse and locate the saved file by clicking the Choose File button (see Figure 7-3).
3. Click the Compile Knowledge Base button (see Figure 7-3).

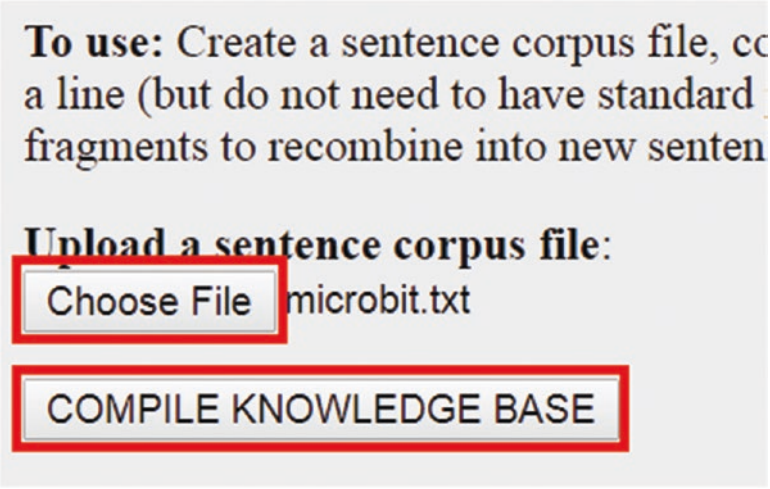

Figure 7-3. *Uploading and compiling a text file*

4. On the results page, click the file name with the .dic extension (see Figure 7-4). This is the Pronunciation Dictionary file.

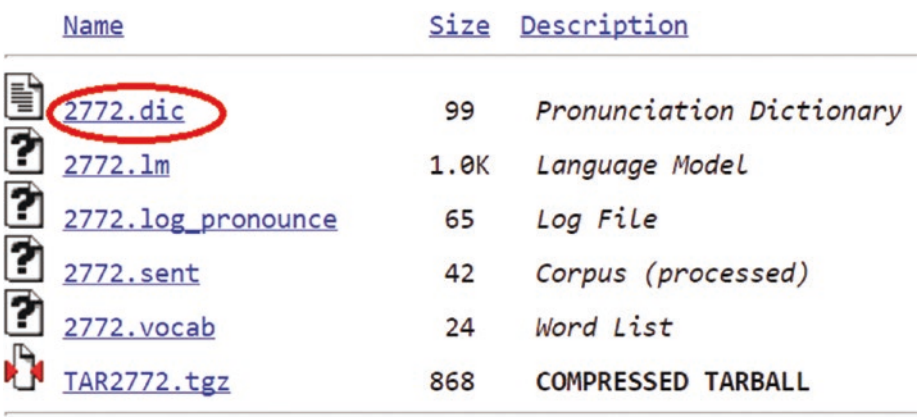

Figure 7-4. *Pronunciation Dictionary file*

5. The file contains phonemes for each word in the sentence (see Figure 7-5). As you can see, the tool suggests two phonemes for the word "Hello". Choose the most relevant phoneme for your micro:bit application.

```
HELLO    HH AH L OW
HELLO(2)         HH EH L OW
WORLD    W ER L D
```

Figure 7-5. *Phonemes for each word*

Stress Markers

Stress markers can be used to create a more expressive tone of voice. They range from 1-8. You can insert the required number after the vowel to create stress. For example, the lack of expression of `/HEHLOW` can be improved by inserting stress marker 3 followed by the vowel `EH`, as in `/HEH3LOW`. Listing 7-11 shows a list of stress markers.

Listing 7-11. Stress Markers

```
1- very emotional stress
2- very emphatic stress
3- rather strong stress
4- ordinary stress
5- tight stress
6- neutral (no pitch change) stress
7 - pitch-dropping stress
8- extreme pitch-dropping stress
```

Listing 7-12 shows the MicroPython code that produces a much improved voice for the word "Hello" by inserting a stress marker.

Listing 7-12. Stress Markers

```
from microbit import *
import speech

speech.pronounce("/HEH3LOW")  # "Hello"
```

Singing with Phonemes

The `speech.sing()` function can be used to sign phonemes. Listing 7-13 shows the lyrics for a happy birthday song.

Listing 7-13. Lyrics for the Happy Birthday Song

```
Happy Birthday to You
Happy Birthday to You
Happy Birthday Dear Micro Bit
Happy Birthday to You
```

First, you need to convert the text to phonemes, as shown in Listing 7-14. You can use the `speech.translate()` function or lmtool to convert the text into phonemes.

Listing 7-14. Phonemes for the Happy Birthday Song

```
HH AE P IY B ER TH D EY T UW Y UW
HH AE P IY B ER TH D EY T UW Y UW
HH AE P IY B ER TH D EY D IH R M AY K R OW B IH T
HH AE P IY B ER TH D EY T UW Y UW
```

Listing 7-15 shows the MicroPython code used to sing a happy birthday song with phonemes.

Listing 7-15. Sing a Song with Phonemes

```
from microbit import *
import speech

speech.sing("#115 /H AE P IY B ER TH D EY T UW Y UW",
speed=100)
speech.sing("#115 /H AE P IY B ER TH D EY T UW Y UW",
speed=100)
speech.sing("#115 /H AE P IY B ER TH D EY D IH R M AY K R OW B
IH T", speed=100) speech.sing("#115 /H AE P IY B ER TH D EY T
UW Y UW", speed=100)
```

You can change the value of the speed parameter to control the speed of the song. The pitch number 115 is used with a hash (#115) as an annotation. You can also add other parameters—such as pitch, mouth, and throat—to change the timbre (quality) of the voice.

Summary

In this chapter, you learned how to produce voices and songs using the micro:bit speech library. You learned how to emulate different voices by changing the characteristics of the voice.

The next chapter explains how to store and manipulate files with micro:bit's internal storage.