Chirps, Zips, and Pings! Sound!

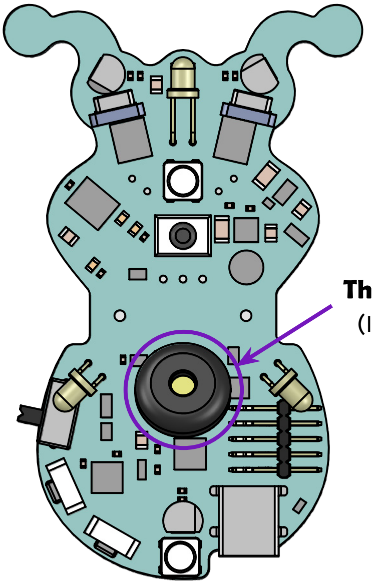
Ringo can make simple sounds with a piece of hardware called a piezo. The piezo is similar to speaker, and can be used to create tones.

Causing Ringo to emit a tone from his piezo is super easy. Just use the `PlayChirp()` function. The PlayChirp function takes two arguments. Arguments are bits of information you pass into a function when it is run. You did this with the pixel and motor functions in the previous example and didn't even realize it. :-)

Let's look at an example:

```
PlayChirp(Frequency, Amplitude);
```

You pass a frequency, which is the pitch of the tone to be played in hertz. Amplitude is a variable that controls the overall tonal quality and loudness. An amplitude of about 50 results in the loudest tone for most pitches. Normally you'll pass the amplitude as 50, but you can experiment with this number a bit to see what happens. Usually, amplitudes greater than 50 will result in more current being used (shorter battery life) without really making the piezo much louder.



This is Ringo's Piezo Element
(It is tucked under the battery)

1
SKILL LEVEL

Chirps, Zips, and Pings! Sound!

Let’s look at a working example. Try entering the following code into your main loop() function and see what happens.

```
void loop(){
  PlayChirp(2000, 50);     //start playing a tone at 2000 hertz
  delay(1000);             //wait 1 second
  PlayChirp(4000, 50);     //start playing a tone at 4000 hertz
  delay(1000);             //wait 1 second
  OffChirp();              //stop playing tone
  delay(2000);             //wait 2 seconds
}
```

Run the code on Ringo and see (hear?) what happens. You should hear a mid pitch tone for one second that then transitions to a higher pitch tone for one second, then you should hear silence for two seconds, then the process should repeat.

The line PlayChirp(2000, 50); causes the tone to begin. It should be noted that Ringo can do other things while a tone is playing. As you already know, the delay(1000); causes Ringo to wait for 1000 milliseconds (which is one second) before executing the next line PlayChrip(4000, 50); which causes Ringo to now begin playing the tone at 4000 hertz.

The OffChirp(); is important because this is what is required to make Ringo stop playing a tone. You can also make him stop playing a tone by either setting the frequency, or the amplitude, or both, to zero.

```
OffChirp();                //turn off chirp the easy way
PlayChirp(0, 50);          //set Frequency to zero. Ringo stops playing tone
PlayChirp(1000, 0);        //set Amplitude to zero. Ringo stops playing tone
PlayChirp(0, 0);   //set Frequency & Amplitude to zero. Ringo stops playing tone
```

Chirps, Zips, and Pings! Sound!

Playing Note Pitches:

If you want Ringo to play a song, you can tell him to play specific notes. Look at the tabs across the top of the Arduino IDE (or click the little down arrow at the end of the tabs if you can't see them all) and select the tab called "Pitches.h". This file "defines" each piano key note to the corresponding pitch in hertz. The "middle C" key of the keyboard is defined as "NOTE_C4". Let's make Ringo play middle C. Go back to the main tab in the Arduino IDE (the one on the far left) and try this code.

Like this:

```
void loop(){
  PlayChirp(NOTE_C6, 50);          //start playing the pitch for middle C
  delay(333);                      //wait about 1/3 of a second
  PlayChirp(NOTE_E6, 50);          //start playing the pitch for E
  delay(333);                      //wait about 1/3 of a second
  PlayChirp(NOTE_G6, 50);          //start playing the pitch of G
  delay(333);                      //wait about 1/3 of a second
  OffChirp();                      //stop making sound
  delay(1000);                     //wait 1 second before looping
}
```

Example Sketch: Ringo_Guide_Chirp_01

Ringo should be playing the C-Major chord every second. If you work with the notes and timings it is possible to make Ringo play simple songs.

Playing Sweeps:

You can "sweep" between tones to create interesting effects. Like this:

```
//PlaySweep(Start_Note, End_Note, Dwell_Time);  //dwell time is in microseconds

void loop(){
  PlaySweep(1000, 4000, 330);
  delay(500);                  //wait 1/2 second
  PlaySweep(4000, 1000, 330);
  delay(500);                  //wait 1/2 second before looping
}
```

Example Sketch: Ringo_Guide_Chirp_02