13. Acoustic Tools

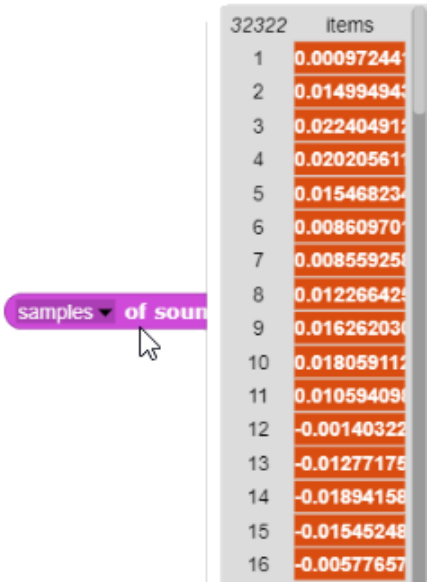
Glen Bull, Joe Watts, and Joe Garofalo

In previous modules, waveforms have been displayed in the TuneScope display in the top half of the TuneScope screen. The process of graphing the waveform has been automated to facilitate the initial exploration of waveforms.

The foundation for exploring the underpinnings of digitized sound has now been established. The process of recording a sound was described in the previous module.

This process converts the back-and-forth movements of a computer's microphone into an electrical signal. The amplitude of the varying electrical signal is repeatedly measured and converted into a series of digital numbers. This process is known as analog-to-digital (A/D) conversion. The process indirectly measures minute movements of the microphone, and converts that movement into a series of digital numbers.

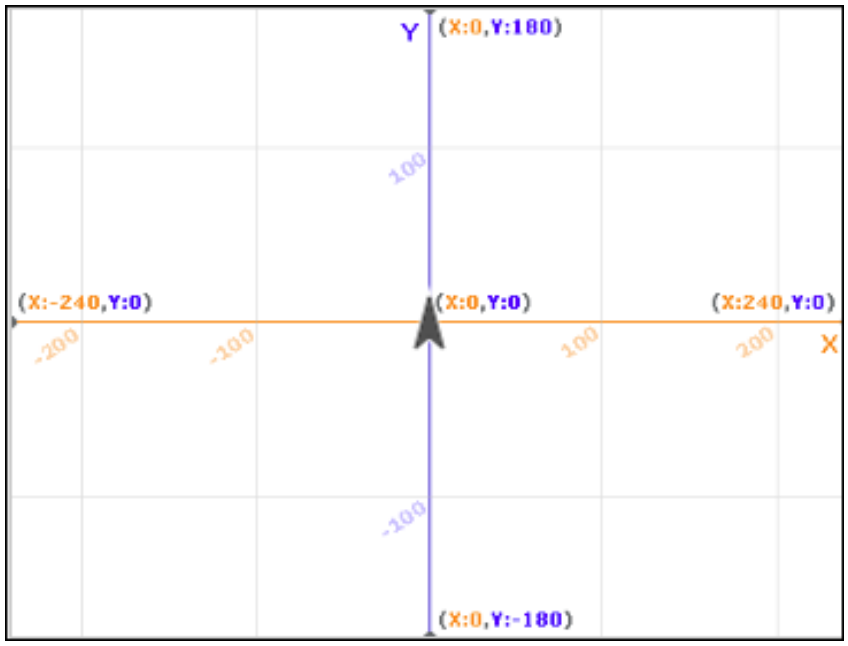
Even a short segment of digitized sound may consist of thousands of sound samples. If the digitization rate is 48,000 samples per second (for example), a single second of speech will result in a list of 48,000 numbers. These numbers represent the voltage levels generated by the back-and-forth movement of the diaphragm of the microphone used to capture the digitized sound.



In article "The Magic Number Seven, Plus or Minus Two" the psychologist George Miller reports that the average individual can only hold about seven numbers in short term memory at one time.

(This is one of the most cited scientific papers of all time.) The limits may vary by one or two, but under no circumstance is it possible for a person to process or understand a list of 48,000 numbers.

This list of numbers representing a second of digitized speech, however, is more comprehensible if it is presented as a graphical representation rather than in number form. The coordinate system for the default *Snap!* stage spans the range from -240 to +240 units on the horizontal (X) axis, and from -180 to +180 units on the vertical (Y) axis.



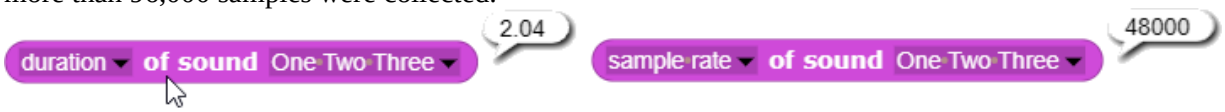
These coordinates provide the framework for graphing any type of data in *Snap!*, including a segment of sound.

Topic 13.1 Graphing Sound Samples

Prior to graphing the waveform, the turtle is positioned on the left side of the screen. The screen is cleared, and the turtle's pen is lowered.

Depending on the computer, the digitized sound samples may consist of numbers such as 0.27 and 0.42. To be clearly seen, the positive peaks of the waveform should ideally fall between 20 and 100 on the vertical axis of the stage. The sound samples can be multiplied by an appropriate number to increase the scale if necessary. In the example below, the samples of the sound utterance "One Two Three" have been multiplied by 100 and assigned to a variable named *Sound*.

Once these setup and scaling activities are completed, a loop can be used to plot the sound samples. The utterance is slightly more than 2 seconds long. At a sample rate of 48,000 samples per second, more than 96,000 samples were collected.

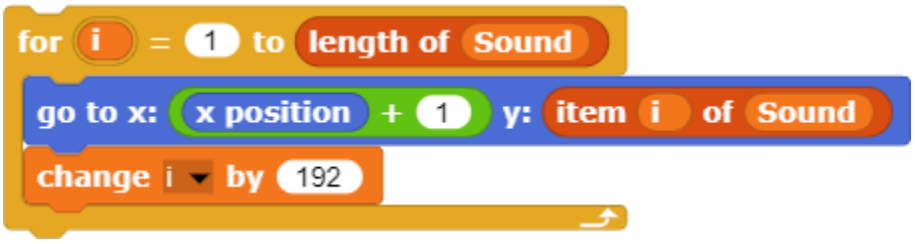


Dividing 48 into 48,000 yields the result of 1000. Consequently, if every 48th point is plotted, each point will represent a one millisecond time increment (since there are 1000 milliseconds in a second).

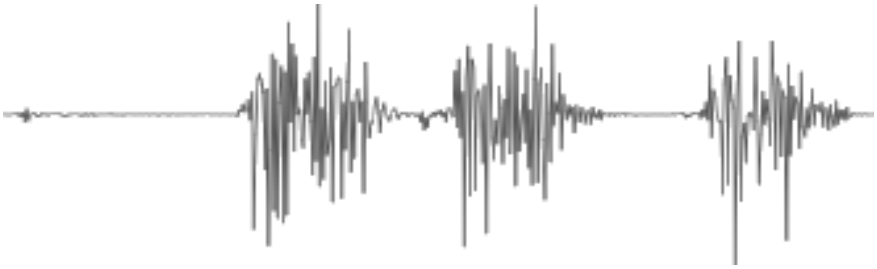
The default width of the stage is 480 steps. If each step in the graph represents a one millisecond time increment, it is still only possible to graph approximately a half-second of the utterance (0.48 seconds).

This time scale can be adjusted by calibrating the number of sound samples that are graphed. If every 96th sound sample is graphed, each point will represent two milliseconds, and it becomes possible to graph nearly a full second of sound on the stage. If every 192nd sound sample is graphed, each point will represent four milliseconds, and it becomes possible to graph nearly two seconds of sound on the stage.

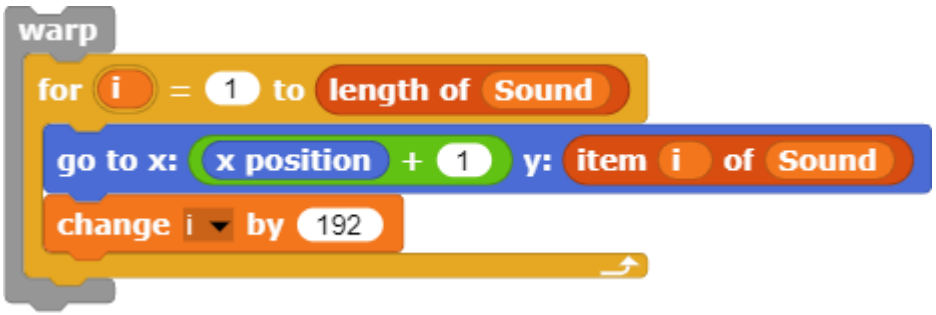
Since the recording of the words "one, two, three" is approximately two seconds long, graphing every 192nd point results in a time scale that makes it possible to display the entire utterance on the stage. This can be accomplished with a For loop that graphs every 192nd sound sample in the utterance. The default increment for a **For** loop is 1; in this instance, the increment has been increased to every 192nd point. After each point is plotted, the turtle moves over one step. By this means, the amplitude of each sound sample is graphed.



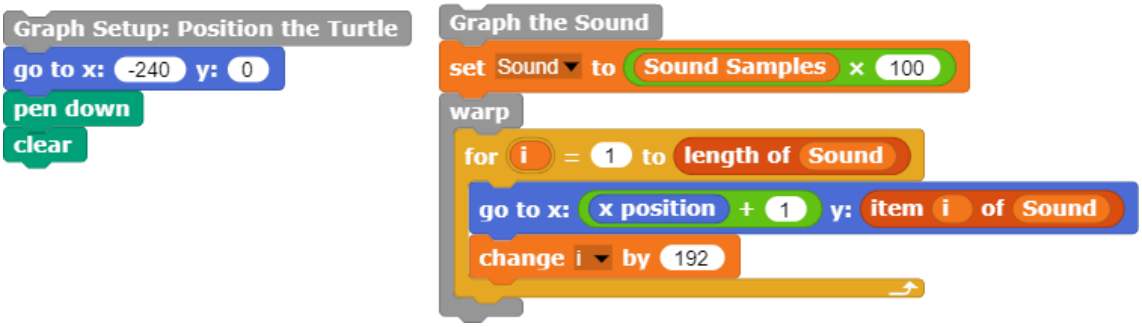
The graph shown in the illustration below results. The words "one", "two", and "three" can be clearly seen. The up and down variations in amplitude correspond to minute variations in the back-and-forth movements of the diaphragm of the microphone that captured the sound.



The graph can be displayed much faster if the procedure is placed in a **Warp** block. The **Warp** code block suspends all background operations until the plotting loop is completed.

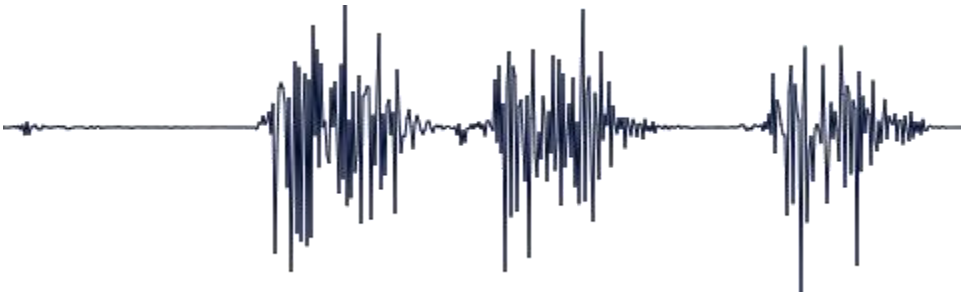


The script to graph the sound has two components: (1) a graph setup procedure that positions the turtle and clears the screen, and (2) a loop that moves through the table of sound samples and graphs these values on the stage.



These two components can be used to build the code blocks **Graph Setup: Position the Turtle** and **Graph the Sound**. These two blocks, in turn, can be used to create the custom code block, **Plot Sound Samples**, that plots the values of the sound samples to create a graph of the sound wave.

The **Plot Sound Samples** code block accepts the samples of a recorded sound as an input and plots the results.



Displaying the sound samples graphically makes it possible to gain a better understanding of the characteristics of the waveform.

Essential Knowledge

- CS Principle 3.1 Variables and Assignments
- CS Principle 3.3 Mathematical Expressions
- CS Principle 3.8 Iteration
- CS Principle 3.10 Lists
- CS Principle 3.13 Developing Procedures

Application of Essential Knowledge

The list of thousands of sound samples is impossible for a human to interpret. When the same sound samples for the utterance “one two three” is plotted, each of the individual words is readily discerned. If the same list of numbers is sent to a speaker cone, the back-and-forth movements of the speaker cone are heard as a spoken phrase. Thus, the same list of numbers serves as the basis for both a graph and a reproduced utterance.

Exploration 13.1 Graphing Sound Samples

Use the **Plot Sound Samples** code block to graph different types of recorded sounds: speech, music, noise, etc. How does the display of the different types of sounds differ? Change the time scale of the graph displayed on the stage.

Topic 13.2 Time Scale

A sense of the time scale is often useful. For example, if 20 cycles of a waveform are displayed, determination of the frequency of the waveform requires calculation of the duration of the sound segment, which makes knowledge of the time scale a prerequisite for obtaining this information. Other characteristics of sound such as onset of the sound (the rate at which the amplitude of the waveform increases at the beginning of an utterance) require a known time scale.

In the previous section, every 192nd sound sample was graphed. Since 48,000 samples per second were acquired when the sound was digitized, each sound sample plotted represents a 4-millisecond time increment. Calculation of the timeline involves two steps:

1. First, the absolute position of the turtle on the *X-axis* (i.e., the horizontal axis) must be determined. The **X-Position** code block can be used to obtain this value. Since the coordinates of the default stage range from -240 on the left to +240 on the right, the position of the turtle ranging from 1 to 480 can be determined by adding 240 to the turtle's current location. The result of this calculation can be assigned to a variable named *Time (seconds)*.

2. Since each sound sample represents a time increment of 4 milliseconds (when every 192nd sound sample is graphed, given a sampling rate of 48,000 samples per second), this result must be multiplied by four to obtain the location in milliseconds. For simplicity, this result is rounded to the nearest millisecond.

   To obtain time in seconds, this result can be divided by 1000 (since there are 1000 milliseconds in a second). In subsequent calculations below, results will be displayed in milliseconds.

When this loop is executed, positioning the turtle (recast in the role of a time cursor) slightly to the right of the center of the stage yields a value of 1000 milliseconds (i.e., one second). Since the duration of the utterance "One, Two, Three" is about two seconds in length, this result confirms that the calculation of the time scale is accurate.

This code can be encapsulated in a custom code block that uses the turtle as a cursor to indicate the time in milliseconds based on its position on the horizontal axis.

The ability to use the turtle as a marker that indicates the time scale is useful in a number of different analyses of the acoustic waveform.

Essential Knowledge

- CS Principle 3.1 Variables and Assignments
- CS Principle 3.3 Mathematical Expressions
- CS Principle 3.13 Developing Procedures

Application of Essential Knowledge

The graph of an acoustic waveform represents the amplitude of the sound sample (i.e., the value of the number representing each sample) over time. Therefore, a knowledge of the time base (i.e., the time increment represented by each sample) is crucial to interpretation of the graph.

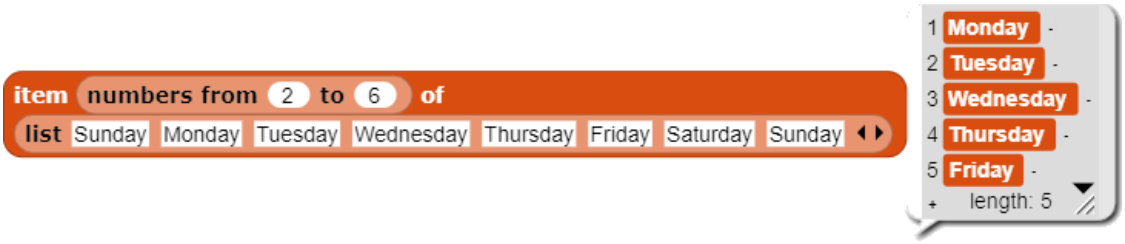
Exploration 13.2 Time Scale

Use the turtle as a time marker to determine the start of the graph of the waveform of the word "One." Then determine the time in milliseconds at which the word ends. How long is the word "One"? How does this duration compare with the length of the words "Two" and "Three"?

Topic 13.3 Sound Segments

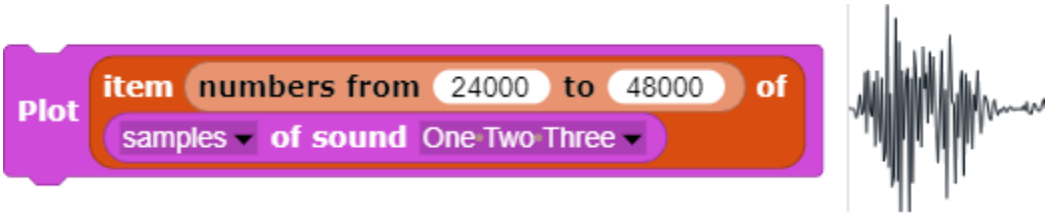
There are times when it may be useful to isolate a portion of an utterance or a specific sound segment for analysis. The **Numbers from __ to __** code block can be used to isolate a portion of a list. For example, if only the workdays in a list of the seven days of the week are desired:

The list of workdays can be obtained by requesting Items 2 through 6 (Mon through Fri) in the list:



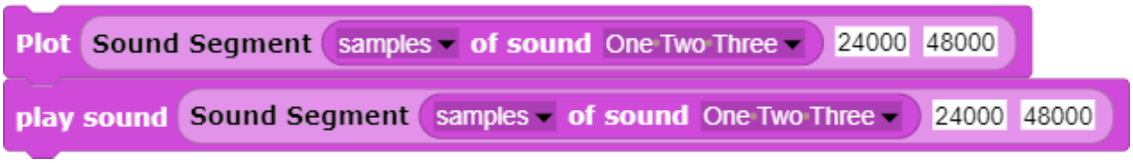
The same method can be used to obtain a portion of a sound recording. If there are 48,0000 sound samples in a second (for sound digitized at that rate), then Items 24,000 to 48,000 will yield the portion of the utterance from 0.5 seconds to 1.0 seconds.

In fact, plotting *Items 24,000 through 48,000* of the utterance "One, Two, Three" yields the word "One" just as we would expect (since this word begins about a half-second into the recording).

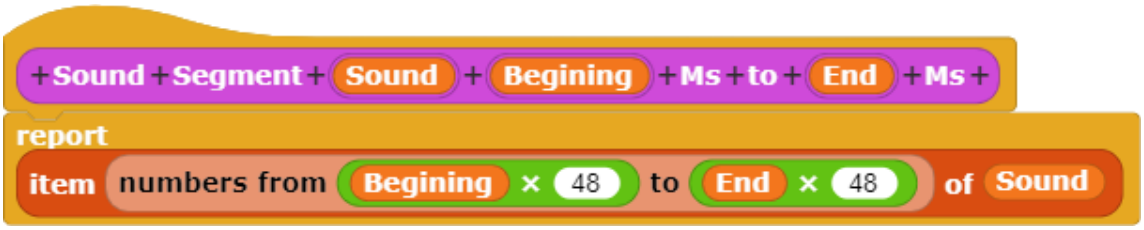


This capability can be incorporated into a custom **Sound Segment** code block.

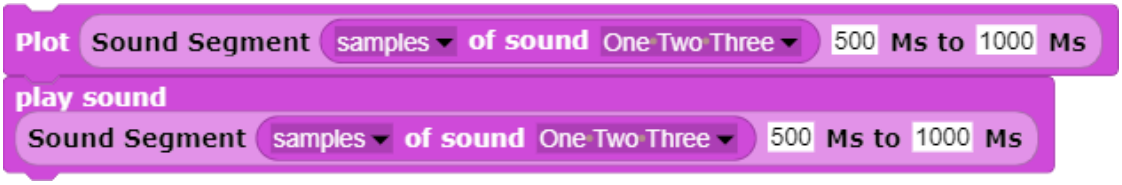
One benefit of creating a custom **Sound Segment** code block is that this code block can not only be used to graph a sound segment but also to play the sound segment to verify that the portion of the utterance obtained is the desired segment.



It may be more convenient to enter the start and end of the sound segment in milliseconds rather than in sound samples. This conversion can be accomplished by multiplying milliseconds by 48 (since 48 times 1000 milliseconds yields 48,000 sound samples). The addition of the label *Ms* after the input slots to indicate the units of measurement employed improves clarity.



The **Sound Segment** code block can now be used with the start and end of the desired sound segment indicated in milliseconds.



The time cursor developed in the previous section can be used to indentify the beginning and end of a desired sound segment. Once these values are obtained, they can be used as inputs to the Sound Segment code block to isolate portions of the utterance.

Essential Knowledge

- CS Principle 3.3 Mathematical Expressions
- CS Principle 3.10 Lists
- CS Principle 3.13 Developing Procedures

Application of Essential Knowledge

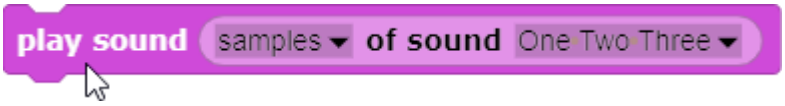
The ability to isolate a specific segment of the sound is an essential tool for interpretation of the graph. Pointers to the beginning and end points of the segment in the list serve as an index for isolating a portion of the sound.

Exploration 13.3 Sound Segments

Use the **Sound Segment** code block to identify and isolate consonants and the vowels in the words “Two” and “Three”.

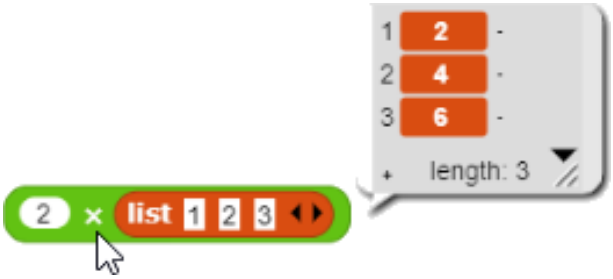
Topic 13.4 Amplification

When a digitized sound is played, the numbers in the list of sound samples control the displacement of the speaker cone. The larger the number, the greater the displacement. Greater displacement of the speaker cone, in turn, results in a sound that is perceived as louder.

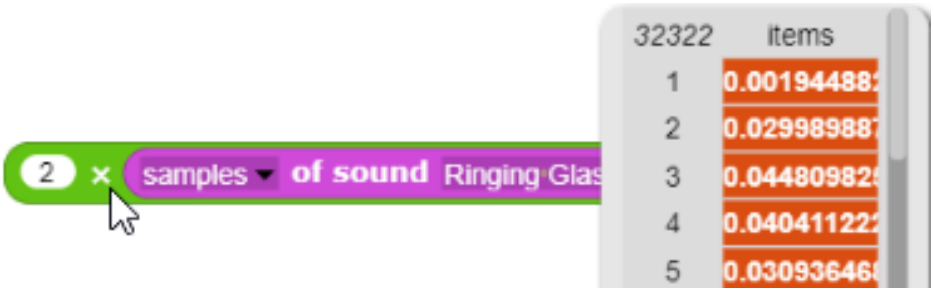


Therefore, the loudness of the sound can be increased by increasing the amplitude of the sound samples. The *multiplication operator* is used to multiply the value of a number.

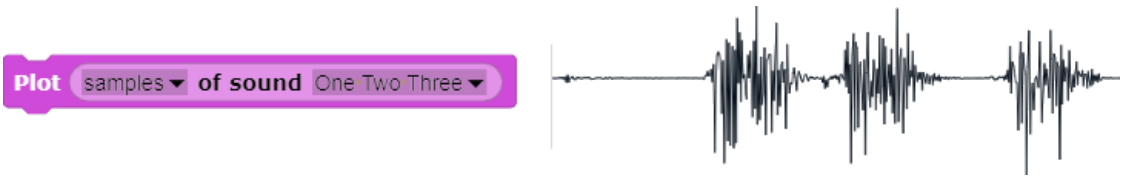
The multiplication operator can be used to multiply all of the numbers in a list.



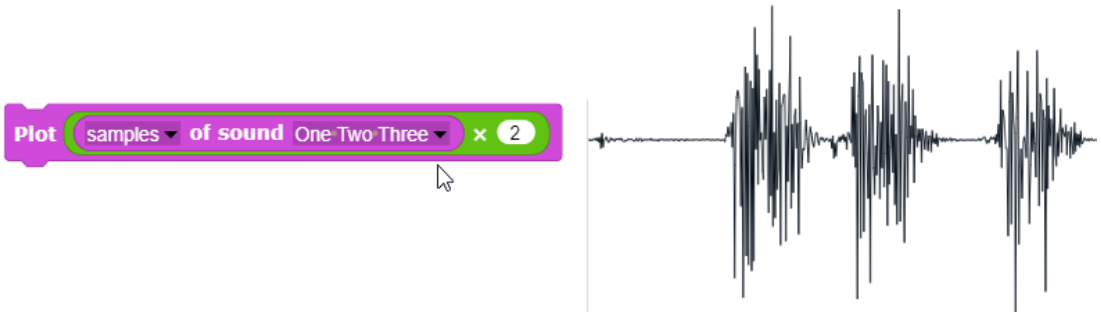
For example, each item in the list of sound samples can be doubled by multiplying by two.



If the graph of the waveform results in the following plot,

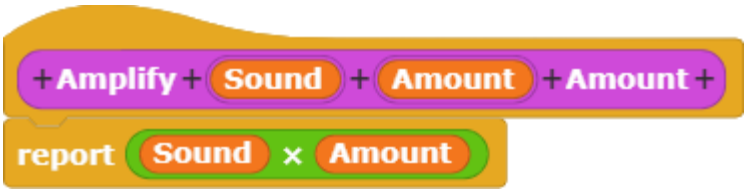


then multiplying the values in the list of sound samples by two results in a graph that is double the amplitude of the previous one.



If the sound samples that have been increased in amplitude are played, the sound is perceived as louder.

This method can be used to create a custom **Amplify** code block to control the amplification of the sound.



If the amount of amplification is greater than one, the amplitude of the sound is increased.

If the amount of amplification is less than one, the amplitude of the sound is decreased.

Essential Knowledge

- CS Principle 3.3 Mathematical Expressions
- CS Principle 3.10 Lists
- CS Principle 3.13 Developing Procedures

Application of Essential Knowledge

The amplitude of the waveform corresponds to the attribute of the sound perceived as loudness. Increasing or decreasing the amplitude by multiplying or dividing all the numbers in the list provides a way of deterring the relationship between a physical attribute of the sound and the perceived loudness.

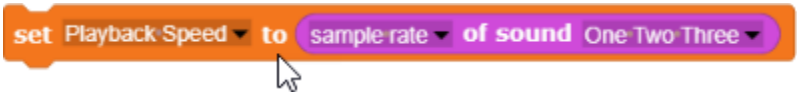
Exploration 13.4 Amplification

Use the **Amplify** code block to increase and decrease the amplitude of a digitized sound. Graph each result and play the resulting sound. How does the amplitude of the graph of the waveform correspond to the perceived loudness of the sound?

Topic 13.5 Adjusting Frequency

Previously, we adjusted the *amplitude* of a sound (perceived as loudness) by multiplying each number in the list of sound samples. Similarly, we can adjust the *frequency* (perceived as pitch) of the recording by varying the playback speed.

This can be accomplished by first setting a variable, *Playback Speed*, to the sampling rate. The default rate at which sound samples in TuneScope are recorded is 48,000 samples per second. Therefore, in most cases the sample rate will be 48,000 samples per second.



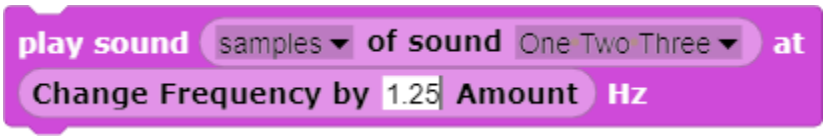
When the sound is played back at the same speed at which it was recorded, a normal pitch is heard. In other words, the recorded frequency and the frequency played back are the same.

However, increasing the playback speed increases the frequency played back in comparison with the frequency at which the sound was recorded.

Similarly, decreasing the playback speed decreases the frequency.

This method can be used as the basis for development of a custom code block to adjust the frequency of a sound.

This code block can then be used to adjust the frequency of a recorded sound when it is played back.



Essential Knowledge

- CS Principle 3.1 Variables and Assignments
- CS Principle 3.3 Mathematical Expressions
- CS Principle 3.10 Lists
- CS Principle 3.13 Developing Procedures

Application of Essential Knowledge

The concept known as *sampling rate* refers to the rate at which sound samples were recorded. The sampling rate determines the time base of the recorded sound. The same time base used to record the sound must be used in playback in order to reproduce the sound with fidelity. If the time base is increased or decreased in playback, the frequency of the sound will change, affecting the perceived pitch of the reproduced sound.

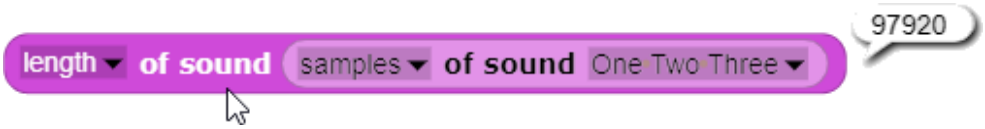
Exploration 13.5 Adjusting Frequency

Use the **Change Frequency** code block to increase and decrease the frequency of a digitized sound when it is played back. How does the frequency of the sound correspond to its perceived pitch?

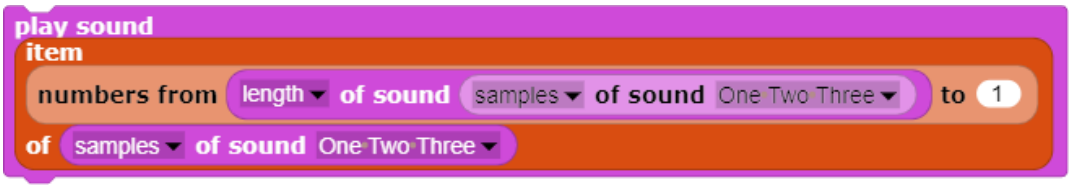
Topic 13.6 Reversing a Sound

If the items in a list are displayed from the end of the list to the beginning, the numbers are reversed. In the example below, the numbers “4, 5, and 6” are reversed.

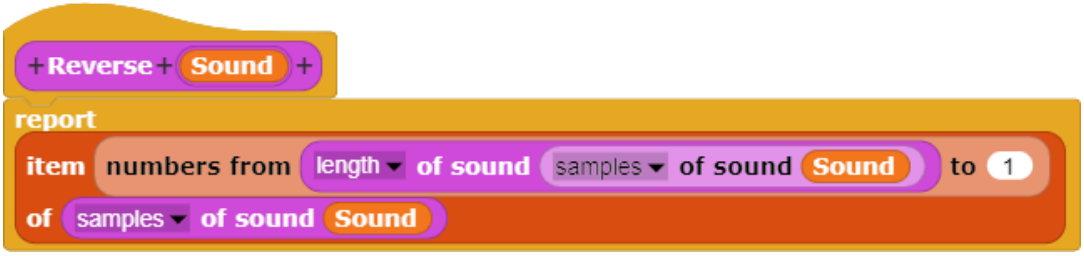
The **Length of Sound** code block can be used to identify the total number of sound samples in a list. In this example, there are a total of 97,920 sound samples in the list.



By playing the list of sound samples from the **Length of Sound** (97,920) to Item 1 in the list, the sound is played in reverse.



This method can be used to create a custom code block, **Reverse**, to reverse the sound.



The **Reverse** code block reverses a sound, starting with the end of the list of sound samples and continuing to the first item in the list of sound samples.

Essential Knowledge

- CS Principle 3.10 Lists
- CS Principle 3.13 Developing Procedures

Application of Essential Knowledge

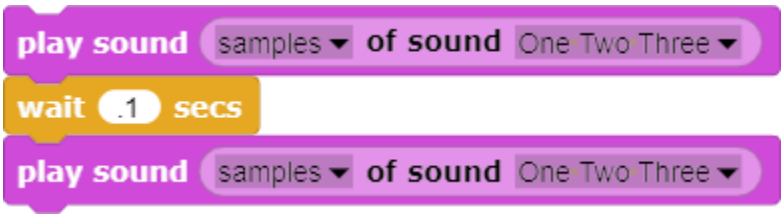
All of the examples described above involve *list processing* – i.e., processing a list in one manner or another. In this instance, the process involves reversing the order of the numbers in the list, which causes the reproduced sound to be played in reverse order.

Exploration 13.6 Reversing a Sound

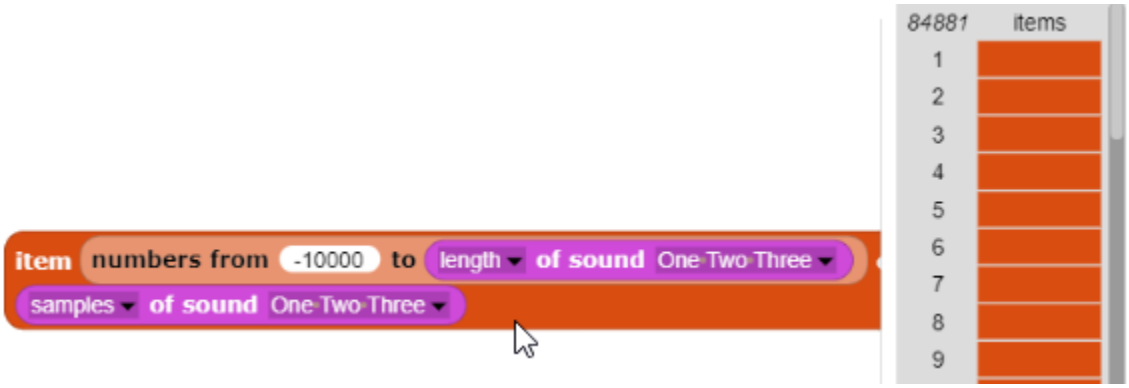
Use the **Reverse** code block to play a list of digitized sound samples in reverse. Play several vowels in reverse to examine the effect that this has upon perception. Are vowels still recognizable when they are reversed? Then play several types of consonants in reverse. What effect does this have upon perception?

Topic 13.7 Echo

If a sound begins to play and then is played a second time with a slight delay, the result is perceived as an echo.

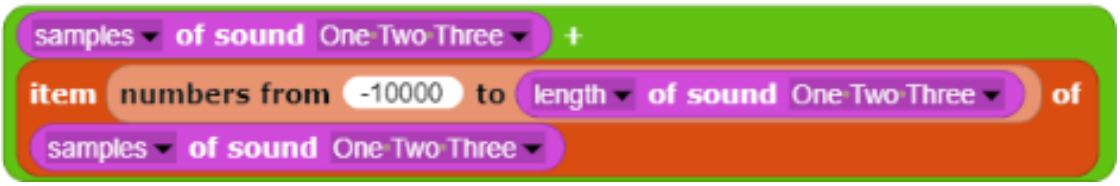


Musicians often use this method to create reverberation. A permanent list that combines one list of sound samples with a second, delayed list of samples can be created by placing a series of empty cells at the beginning of the second list. If the items from -10,000 to the **Length of the List** are displayed, the first 10,000 items in the list will be empty.

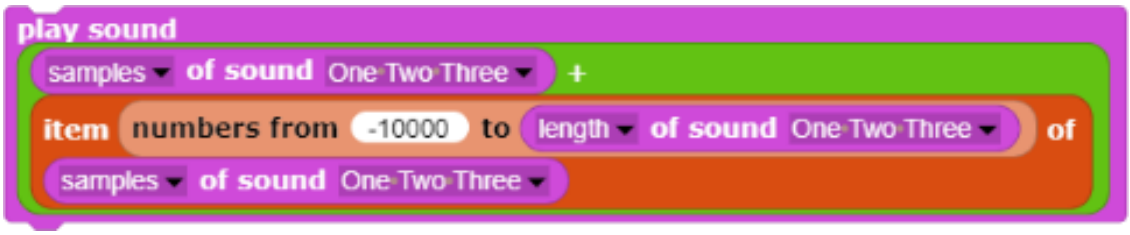


When the empty cells in the list of sound samples are played back, the sound will be silent during this portion of the list. This, in effect, delays the playback for that amount of time.

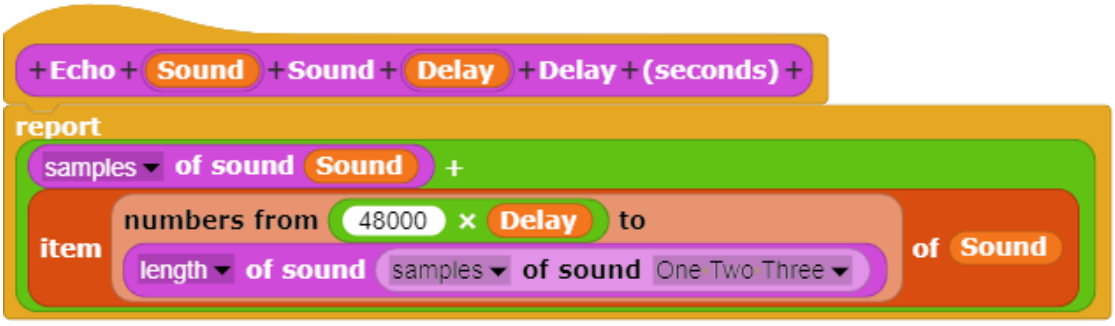
The list with the silence at the beginning can be combined with the original list by using the plus ("+") operator.



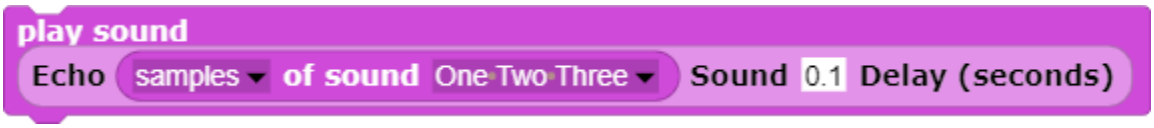
When the combined lists created in this manner are played back, there is a slight delay before the second list of sound samples begins, creating the effect of an echo. Increasing or decreasing the number of empty cells in the second list controls the length of the delay.



This technique can be used to create a custom **Echo** code block to create an echo with a specified delay that controls the timing of when the second sound begins playing.



The custom **Echo** code block can be used to specify the amount of echo that is incorporated into the playback.



In a live performance setting, such as an auditorium, the amount of delay is related to the amount of time that it takes for the sound to travel to the walls and reflective surfaces of the auditorium and return. These reverberations affect the quality of musical performances. Listener's perceptions of recorded performances are also affected by these factors.

Essential Knowledge

- CS Principle 3.3 Mathematical Expressions
- CS Principle 3.10 Lists
- CS Principle 3.13 Developing Procedures

Application of Essential Knowledge

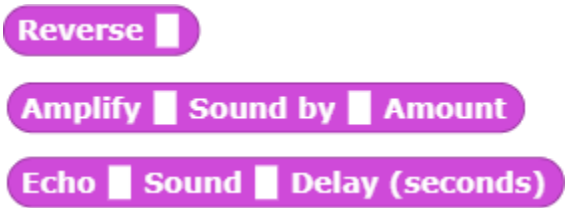
Reproduction of a sound a second time, with a slight delay between the first and the second reproduction, simulates the effect of a sound wave bouncing off the walls and ceiling of a room to create the effect of an echo. In an actual room, the reverberation results from the sound wave reflecting from many different surfaces rather than just one.

Exploration 13.7 Echo

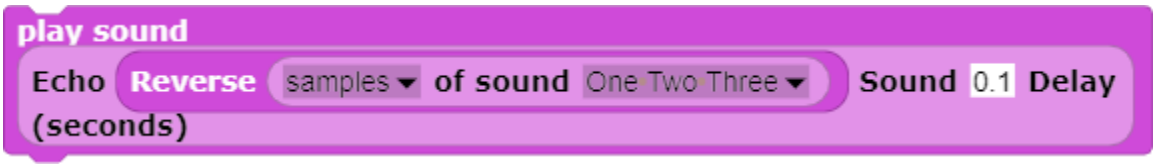
Use the **Echo** code block to explore the effect on different lists of digitized sound samples. Attempt to create the perception of a large auditorium by controlling the amount of echo employed in the playback.

Topic 13.8 Combining Sound Effects

The **Reverse**, **Amplify**, and **Echo** code blocks can be combined with one another.



For example, a reversed sound with echo added can be played by combining those two code blocks.



Exploration 13.8 Combining Sound Effects. Explore the effect of combining the **Reverse**, **Amplify**, and **Echo** code blocks in various combinations.

The acoustic tools developed for manipulating and digitizing sound samples can perform operations on the list of sound samples to increase or decrease the loudness, increase or decrease the pitch, play the sound backwards, add an echo, or isolate specific words or sounds within a longer recording.

Summary

The sound effects created through custom code blocks such as **Amplify** and **Echo** are illustrative of many other types of operations that can be performed to create other sound effects. For example, a high-pass filter removes some of the lower frequencies. Conversely, a low-pass filter removes some of the higher frequencies. These methods are the basis for many popular effects now used to create and record music. In various combinations, these methods are also the basis for creating and manipulating synthesized speech.