Audio Signal Processing with API

-R. Naga Leela Krishna

Audio signals are the representation for sound waves, whether it is an analog or digital source, being computed and generated to be utilized as data for application.

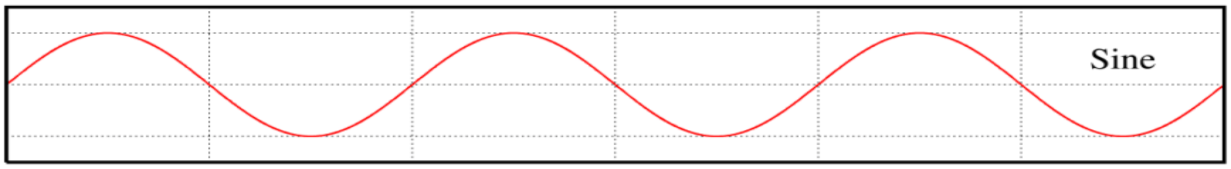
It is actually a subtopic of Digital Signal Processing, which is the main topic of translating data into a computer form to be used, where data can vary from speech, imaging, telecommunication, seismology, biomedical engineering, etc.

Examples in applying Audio Signal Processing:

- storage
- data compression
- speech processing
- transmission
- noise cancellation
- acoustic fingerprinting
- enhancement (equalization, filtering, level compression, echo and reverb removal or addition, etc.)

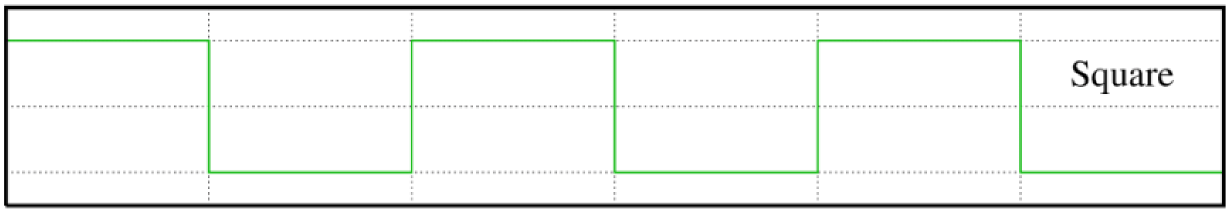
Audo signals are the representation of sound. There are two different types of signals: Analog and Digital Signals.

- Analog is represented with a sign wave, described with Period, Amplitude, Phase and Frequency.



With this representation of the data, it can be translated and interpreted as digital data to be processed with the computer now. This is most represented as a sine wave. Analog is like the human voice where you can control your volume, speed, and articulation, etc.

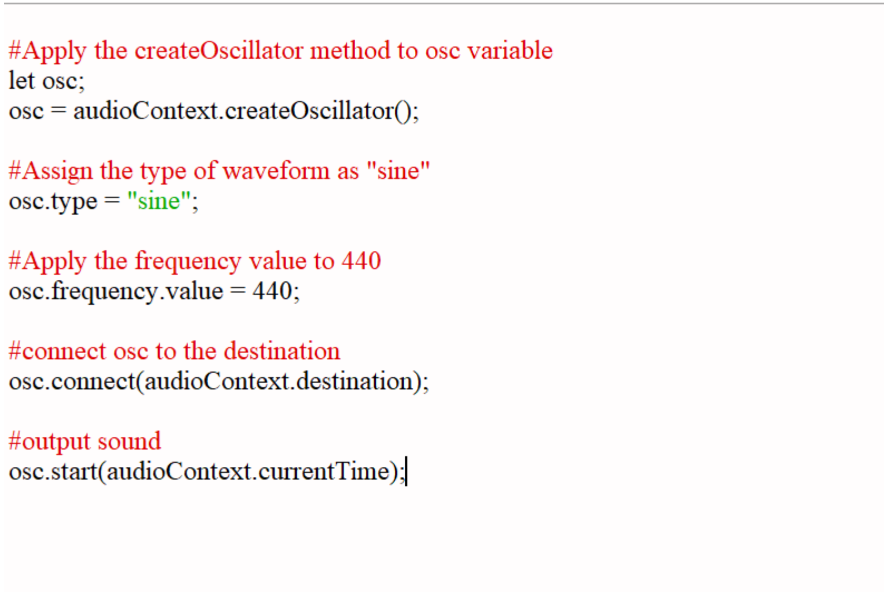
- Digital is represented with a square wave. It carries data in binary form. It's described with bit rate and bit interval.



The square wave is more straight-forward with no curves and slopes. This means that the sound would be cutoff with no steady progression. Digital is like picking a note on an electric guitar immediately (unless you control your volume with the volume knob).

We use Web Audio API, which retrieves and manipulates audio data, as an example with processing audio signals.

Here is a code structure that describes the audio signal processing using Web API:



```
#Apply the createOscillator method to osc variable
let osc;
osc = audioContext.createOscillator();

#Assign the type of waveform as "sine"
osc.type = "sine";

#Apply the frequency value to 440
osc.frequency.value = 440;

#connect osc to the destination
osc.connect(audioContext.destination);

#output sound
osc.start(audioContext.currentTime);
```

With this example, we are creating an audio signal, which is the oscillator and giving the waveform type. We are manipulating the oscillator by giving a frequency value of 440.

We could also adjust other parts to the audio, like the amplitude which would adjust the gain or volume.