Introduction to audio encoding

An audio encoding refers to the manner in which audio data is stored and transmitted. The documentation below describes how such encodings work. For guidelines on choosing the best encoding for your application, see Best Practices (https://cloud.google.com/speech-to-text/docs/best-practices).

Digital audio encoding is a complex topic, and you generally don't need to know the details to process audio within the Speech API. The concepts provided here are meant only as a general overview. Some of this background information may be useful for understanding how the API works, and how audio should be formulated and processed in your applications.

Audio formats vs encodings

Note that an audio format is not equivalent to an audio encoding. A popular file format like `.WAV` for example, defines the format of the header of an audio file, but is not itself an audio encoding. `.WAV` audio files often, but not always, use a linear PCM encoding; don't assume a `.WAV` file has any particular encoding until you inspect its header.

`FLAC`, however, is both a file format and an encoding, which sometimes leads to some confusion. Within the Speech-to-Text API, `FLAC` is the only encoding that requires audio data to include a header; all other audio encodings specify headerless audio data. When we refer to `FLAC` within the Speech-to-Text API, we are always referring to the codec. When we refer to a FLAC file format, we will use the format "a `.FLAC` file."

You are not required to specify the encoding and sample rate for WAV or FLAC files. If omitted, Cloud Speech-to-Text automatically determines the encoding and sample rate for WAV or FLAC files based on the file header. If you specify an encoding or sample rate value that does not match the value in the file header, then Cloud Speech-to-Text returns an error.

Supported audio encodings

The Speech-to-Text API supports a number of different encodings. The following table lists supported audio codecs:

Codec	Name	Lossless	Usage Notes
`MP3`	MPEG Audio Layer III	No	Only available as beta. See [`RecognitionConfig`](https://cloud.google.com/speech-to-text/docs/reference/rest/v1p1beta1/RecognitionConfig#) (https://cloud.google.com/speech-to-text/docs/reference/rest/v1p1beta1/RecognitionConfig# reference for details.
`FLAC`	Free Lossless Audio Codec	Yes	16-bit or 24-bit required for streams
`LINEAR16`	Linear PCM	Yes	16-bit linear pulse-code modulation (PCM) encoding
`MULAW`	μ-law	No	8-bit PCM encoding
`AMR`	Adaptive Multi-Rate Narrowband	No	Sample rate must be 8000 Hz
`AMR_WB`	Adaptive Multi-Rate Wideband	No	Sample rate must be 16000 Hz
`OGG_OPUS`	Opus encoded audio frames in an Ogg container	No	Sample rate must be one of 8000 Hz, 12000 Hz, 16000 Hz
`SPEEX_WITH_HEADER_BYTE`	Speex wideband	No	Sample rate must be 16000 Hz

Note: `FLAC` is both an audio codec and an audio file format. To transcribe audio files using `FLAC` encoding, you must provide them in the `.FLAC` file format, which includes a header containing metadata.

Note: Cloud Speech-to-Text supports `WAV` files with `LINEAR16` or `MULAW` encoded audio.

For more information on Cloud Speech-to-Text audio codecs, consult the AudioEncoding (https://cloud.google.com/speech-to-text/docs/reference/rest/v1/RecognitionConfig#AudioEncoding) reference documentation.

If you have a choice when encoding the source material, use a lossless encoding such as `FLAC` or `LINEAR16` for better speech recognition. For guidelines on selecting the appropriate codec for your task, see Best Practices (https://cloud.google.com/speech-to-text/docs/best-practices).

Why encode?

Audio is made up of waveforms, consisting of the interposition of waves of different frequencies and amplitudes. To represent these waveforms within digital media, the waveforms need to be *sampled* at rates that can (at least) represent sounds of the highest frequency which you wish to replicate, and they also need to store enough *bit depth* to represent the proper amplitude (loudness and softness) of the waveforms across the sound sample.

The ability of a sound processing device to recreate frequencies is known as its *frequency response* and the ability of it to create proper loudness and softness is known as its *dynamic range*. Together these terms are often referred to as a sound device's *fidelity*. An encoding, in its simplest form, is a means with which to reconstruct sound using these two basic principles, as well as being able to store and transport such data efficiently.

Sampling rates

Sound exists as an analog waveform. A segment of digital audio approximates this analog wave by sampling the amplitude of this analog wave at a fast enough rate to mimic the wave's intrinsic frequencies. A digital audio segment's *sample rate* specifies the number of samples to take from an audio's source material (per second); a high sample rate increases the ability of digital audio to faithfully represent high frequencies.

As a consequence of the Nyquist-Shannon theorem (https://en.wikipedia.org/wiki/Nyquist%E2%80%93Shannon_sampling_theorem), you generally need to sample at least twice the highest frequency of any sound wave you wish to capture digitally. To represent audio within the range of human hearing (20-20000 Hz), for example, a digital audio format must sample at least 40000 times per second (which is part of the reason why CD audio uses a sample rate of 44100 Hz).

Bit depths

Bit depth affects the dynamic range of a given audio sample. A higher bit depth allows you to represent more precise amplitudes. If you have lots of loud and soft sounds within the same audio sample, you will need more bit depth to represent those sounds correctly.

Higher bit depths also reduce the signal to noise ratio within audio samples. CD musical audio is provided using 16 bits of bit depth. DVD Audio uses 24 bits of bit depth, while most telephony equipment uses 8 bits of bit depth. (Certain compression techniques can compensate for smaller bit depths, but they tend to be lossy.)

Uncompressed audio

Most digital audio processing uses these two techniques — sampling rate and bit depth — to store audio data in a straightforward manner. One of the most popular digital audio techniques (popularized in use of the Compact Disc) is known as Pulse Code Modulation (https://en.wikipedia.org/wiki/Pulse-code_modulation) (or PCM). Audio is sampled at set intervals, and the amplitude of the sampled wave at that point is stored as a digital value using the sample's bit depth.

Linear PCM (which indicates that the amplitude response is linearly uniform across the sample) is the standard used within CDs, and within the `LINEAR16` encoding of the Speech-to-Text API. Both encodings produce an uncompressed stream of bytes corresponding directly to audio data, and both standards contain 16 bits of depth. Linear PCM uses a sample rate of 44,100 Hz within CDs, which is appropriate for the recomposition of music; however a sample rate of 16000 Hz is more appropriate for recomposing speech.

Linear PCM (`LINEAR16`) is an example of *uncompressed audio* in that the digital data is stored exactly as the above standards imply. Reading a one-channel stream of bytes encoded using Linear PCM, you could count off every 16 bits (2 bytes), for example, to get another amplitude value of the waveform. Almost all devices can manipulate such digital data natively — you can even crop Linear PCM audio files using a text editor — but (obviously) uncompressed audio is not the most efficient way to transport or store digital audio. For that reason, most audio uses digital compressions techniques.

Compressed audio

Audio data, like all data, is often compressed to make it easier to store and to transport. Compression within audio encoding may be either *lossless* or *lossy*. Lossless compression can be unpacked to restore the digital data to its original form. Lossy compression necessarily removes some such information during compression and decompression, and is parameterized to indicate how much tolerance to give to the compression technique to remove data.

Lossless compression

Lossless compression compresses digital audio data using complex rearrangements of the stored data, but results in no degradation in quality of the original digital sample. With lossless compression, when unpacking the data into its original digital form, no information will be lost.

So why do lossless compression techniques sometimes have optimization parameters? These parameters often trade file size for decompression time. For example, `FLAC` uses a compression level parameter from 0 (fastest) to 8 (smallest file size). Higher level FLAC compression won't lose any information in comparison to lower level compression. Instead, the compression algorithm will just need to expend more computational energy when constructing or deconstructing original digital audio.

The Speech-to-Text API supports two lossless encodings: `FLAC` and `LINEAR16`. Technically, `LINEAR16` isn't "lossless compression" because no compression is involved in the first place. If file size or data transmission is important to you, choose `FLAC` as your audio encoding choice.

Lossy compression

Lossy compression, on the other hand, compresses audio data by eliminating or reducing certain types of information during the construction of the compressed data. The Speech-to-Text API supports several lossy formats, though you should avoid them if you have control over the audio, because data loss may affect recognition accuracy.

The popular MP3 codec is an example of a lossy encoding technique. All MP3 compression techniques remove audio from outside a normal human's audio range, and adjust the amount of compression by adjusting the MP3 codec's effective *bit rate*, or amount of bits per second to store the audio date.

For example, a stereo CD using Linear PCM of 16 bits has an effective bit rate of:

```
44100 * 2 channels * 16 bits = 1411200 bits per second (bps) = 1411 kbps
```

MP3 compression removes such digital data using bit rates such as 320 kbps, 128 kbps, or 96 kbps, for example, with resulting degradation in audio quality. MP3 also supports variable bit rates, which can compress the audio further. Both techniques lose information and can affect quality. Most people can tell the difference between 96 kbps or 128 kbps encoded MP3 music, for example.

Other forms of compression will parameterize some other constraint.

MULAW (https://en.wikipedia.org/wiki/%CE%9C-law_algorithm) is an 8 bit PCM encoding, where the sample's amplitude is modulated logarithmically rather than linearly. As a result, uLaw reduces the effective dynamic range of the audio thus compressed. Though uLaw was introduced to specifically optimize encoding of speech in contrast to other types of audio, 16 bit `LINEAR16` (uncompressed PCM) is still far superior to 8 bit uLaw compressed audio.

AMR (https://en.wikipedia.org/wiki/Adaptive_Multi-Rate_audio_codec) and AMR_WB (https://en.wikipedia.org/wiki/Adaptive_Multi-Rate_Wideband) modulate the encoded audio sample by introducing a variable bit rate on the source audio sample.

Although the Speech-to-Text API supports several lossy formats, you should avoid them if you have control over the source audio. Although the removal of such data through lossy compression may not noticeably affect audio as heard by the human ear, loss of such data to a speech recognition engine may significantly degrade accuracy.