Documentation

search

Search

- *rocket_launch*

  Get started

  - Installation
    add
  - Fundamentals
    add
  - First steps
    add
- *code*

  Develop

  - Concepts
    add
  - API reference
    remove
    - PAGE ELEMENTS

    - Write and magic
      add
    - Text elements
      add
    - Data elements
      add
    - Chart elements
      add
    - Input widgets
      add
    - Media elements
      remove
      - st.audio
      - st.image
      - st.logo
      - st.video
    - Layouts and containers
      add
    - Chat elements
      add
    - Status elements
      add
    - Third-party components*open_in_new*
    - APPLICATION LOGIC

    - Navigation and pages
      add
    - Execution flow
      add
    - Caching and state

st.audio

Streamlit Version Version 1.41.0

Display an audio player.

Function signature[source]

st.audio(data, format="audio/wav", start_time=0, *, sample_rate=None, end_time=None, loop=False, autoplay=False)

Parameters

Function signature[source]

st.audio(data, format="audio/wav", start_time=0, *, sample_rate=None, end_time=None, loop=False, autoplay=False)

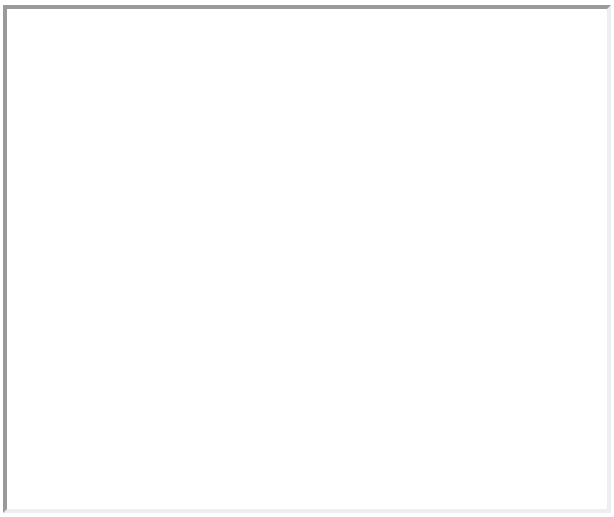
data (str, Path, bytes, BytesIO, numpy.ndarray, or file)	The audio to play. This can be one of the following: • A URL (string) for a hosted audio file. • A path to a local audio file. The path can be a `str` or `Path` object. Paths can be absolute or relative to the working directory (where you execute `streamlit run`). • Raw audio data. Raw data formats must include all necessary file headers to match the file format specified via `format`. If `data` is a NumPy array, it must either be a 1D array of the waveform or a 2D array of shape (C, S) where C is the number of channels and S is the number of samples. See the default channel order at http://msdn.microsoft.com/en-us/library/windows/hardware/dn653308(v=vs.85).aspx
format (str)	The MIME type for the audio file. This defaults to `"audio/wav"`. For more information, see https://tools.ietf.org/html/rfc4281.
start_time (int, float, timedelta, str, or None)	The time from which the element should start playing. This can be one of the following: • `None` (default): The element plays from the beginning. • An `int` or `float` specifying the time in seconds. `float` values are rounded down to whole seconds. • A string specifying the time in a format supported by Pandas' Timedelta constructor, e.g. `"2 minute"`, `"20s"`, or `"1m14s"`. • A `timedelta` object from Python's built-in datetime library, e.g. `timedelta(seconds=70)`.
sample_rate (int or None)	The sample rate of the audio data in samples per second. This is only required if `data` is a NumPy array.
end_time (int, float, timedelta, str, or None)	The time at which the element should stop playing. This can be one of the following: • `None` (default): The element plays through to the end. • An `int` or `float` specifying the time in seconds. `float` values are rounded down to whole seconds. • A string specifying the time in a format supported by Pandas' Timedelta constructor, e.g. `"2 minute"`, `"20s"`, or `"1m14s"`. • A `timedelta` object from Python's built-in datetime library, e.g. `timedelta(seconds=70)`.
loop (bool)	Whether the audio should loop playback.
autoplay (bool)	Whether the audio file should start playing automatically. This is `False` by default. Browsers will not autoplay audio files if the user has not interacted with the page by clicking somewhere.

Examples

To display an audio player for a local file, specify the file's string path and format.

```
import streamlit as st

st.audio("cat-purr.mp3", format="audio/mpeg", loop=True)
```



Built with Streamlit 🎈
Fullscreen *open_in_new*

You can also pass `bytes` or `numpy.ndarray` objects to `st.audio`.

```
import streamlit as st
import numpy as np

audio_file = open("myaudio.ogg", "rb")
audio_bytes = audio_file.read()

st.audio(audio_bytes, format="audio/ogg")

sample_rate = 44100  # 44100 samples per second
seconds = 2  # Note duration of 2 seconds
frequency_la = 440  # Our played note will be 440 Hz
# Generate array with seconds*sample_rate steps, ranging between 0 and seconds
t = np.linspace(0, seconds, seconds * sample_rate, False)
# Generate a 440 Hz sine wave
note_la = np.sin(frequency_la * t * 2 * np.pi)

st.audio(note_la, sample_rate=sample_rate)
```

Built with Streamlit 🎈
Fullscreen *open_in_new*

←Previous: Media elementsNext: st.image→

forum

Still have questions?

Our forums are full of helpful information and Streamlit experts.

 Cookie policy

forum Ask AI