:mod:`ossaudiodev` --- Access to OSS-compatible audio devices

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 1); *backlink*

Unknown interpreted text role "mod".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 4)

Unknown directive type "module".

```
.. module:: ossaudiodev
   :platform: Linux, FreeBSD
   :synopsis: Access to OSS-compatible audio devices.
   :deprecated:
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 9)

Unknown directive type "deprecated".

```
.. deprecated:: 3.11
   The :mod:`ossaudiodev` module is deprecated (see :pep:`594` for details).
```

This module allows you to access the OSS (Open Sound System) audio interface. OSS is available for a wide range of open-source and commercial Unices, and is the standard audio interface for Linux and recent versions of FreeBSD.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 46)

Unknown directive type "versionchanged".

```
.. versionchanged:: 3.3
   Operations in this module now raise :exc:`OSError` where :exc:`IOError`
   was raised.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 51)

Unknown directive type "seealso".

```
.. seealso::

   `Open Sound System Programmer's Guide <http://www.opensound.com/pguide/oss.pdf>`_
      the official documentation for the OSS C API

   The module defines a large number of constants supplied by the OSS device
   driver; see ``<sys/soundcard.h>`` on either Linux or FreeBSD for a listing.
```

:mod:`ossaudiodev` defines the following variables and functions:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 59); *backlink*

Unknown interpreted text role "mod".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 62)

Unknown directive type "exception".

```
.. exception:: OSSAudioError

   This exception is raised on certain errors.  The argument is a string describing
   what went wrong.

   (If :mod:`ossaudiodev` receives an error from a system call such as
   :c:func:`open`, :c:func:`write`, or :c:func:`ioctl`, it raises :exc:`OSError`.
   Errors detected directly by :mod:`ossaudiodev` result in :exc:`OSSAudioError`.)

   (For backwards compatibility, the exception class is also available as
   ``ossaudiodev.error``.)
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 75)

Unknown directive type "function".

```
.. function:: open(mode)
              open(device, mode)

   Open an audio device and return an OSS audio device object.  This object
   supports many file-like methods, such as :meth:`read`, :meth:`write`, and
   :meth:`fileno` (although there are subtle differences between conventional Unix
   read/write semantics and those of OSS audio devices).  It also supports a number
   of audio-specific methods; see below for the complete list of methods.

   *device* is the audio device filename to use.  If it is not specified, this
   module first looks in the environment variable :envvar:`AUDIODEV` for a device
   to use.  If not found, it falls back to :file:`/dev/dsp`.

   *mode* is one of ``'r'`` for read-only (record) access, ``'w'`` for
   write-only (playback) access and ``'rw'`` for both. Since many sound cards
   only allow one process to have the recorder or player open at a time, it is a
   good idea to open the device only for the activity needed.  Further, some
   sound cards are half-duplex: they can be opened for reading or writing, but
   not both at once.

   Note the unusual calling syntax: the *first* argument is optional, and the
   second is required.  This is a historical artifact for compatibility with the
   older :mod:`linuxaudiodev` module which :mod:`ossaudiodev` supersedes.

   .. XXX it might also be motivated
      by my unfounded-but-still-possibly-true belief that the default
      audio device varies unpredictably across operating systems.  -GW
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 104)

Unknown directive type "function".

```
.. function:: openmixer([device])

   Open a mixer device and return an OSS mixer device object.   *device* is the
   mixer device filename to use.  If it is not specified, this module first looks
   in the environment variable :envvar:`MIXERDEV` for a device to use.  If not
   found, it falls back to :file:`/dev/mixer`.
```

Audio Device Objects

Before you can write to or read from an audio device, you must call three methods in the correct order:

1. :meth:`setfmt` to set the output format

   System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 120); *backlink*

   Unknown interpreted text role "meth".

2. :meth:`channels` to set the number of channels

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]ossaudiodev.rst`, line 122); *backlink*

Unknown interpreted text role "meth".

3. :meth:`speed` to set the sample rate

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]ossaudiodev.rst`, line 124); *backlink*

Unknown interpreted text role "meth".

Alternately, you can use the :meth:`setparameters` method to set all three audio parameters at once. This is more convenient, but may not be as flexible in all cases.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]ossaudiodev.rst`, line 126); *backlink*

Unknown interpreted text role "meth".

The audio device objects returned by :func:`.open` define the following methods and (read-only) attributes:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]ossaudiodev.rst`, line 130); *backlink*

Unknown interpreted text role "func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]ossaudiodev.rst`, line 134)

Unknown directive type "method".

```
.. method:: oss_audio_device.close()

   Explicitly close the audio device.  When you are done writing to or reading from
   an audio device, you should explicitly close it.  A closed device cannot be used
   again.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]ossaudiodev.rst`, line 141)

Unknown directive type "method".

```
.. method:: oss_audio_device.fileno()

   Return the file descriptor associated with the device.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]ossaudiodev.rst`, line 146)

Unknown directive type "method".

```
.. method:: oss_audio_device.read(size)

   Read *size* bytes from the audio input and return them as a Python string.
   Unlike most Unix device drivers, OSS audio devices in blocking mode (the
   default) will block :func:`read` until the entire requested amount of data is
   available.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]ossaudiodev.rst`, line 154)

Unknown directive type "method".

```
.. method:: oss_audio_device.write(data)

   Write a :term:`bytes-like object` *data* to the audio device and return the
   number of bytes written.  If the audio device is in blocking mode (the
   default), the entire data is always written (again, this is different from
   usual Unix device semantics).  If the device is in non-blocking mode, some
   data may not be written---see :meth:`writeall`.

   .. versionchanged:: 3.5
      Writable :term:`bytes-like object` is now accepted.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 166)

Unknown directive type "method".

```
.. method:: oss_audio_device.writeall(data)

   Write a :term:`bytes-like object` *data* to the audio device: waits until
   the audio device is able to accept data, writes as much data as it will
   accept, and repeats until *data* has been completely written. If the device
   is in blocking mode (the default), this has the same effect as
   :meth:`write`; :meth:`writeall` is only useful in non-blocking mode.  Has
   no return value, since the amount of data written is always equal to the
   amount of data supplied.

   .. versionchanged:: 3.5
      Writable :term:`bytes-like object` is now accepted.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 180)

Unknown directive type "versionchanged".

```
.. versionchanged:: 3.2
   Audio device objects also support the context management protocol, i.e. they can
   be used in a :keyword:`with` statement.
```

The following methods each map to exactly one :c:func:`ioctl` system call. The correspondence is obvious: for example, :meth:`setfmt` corresponds to the `SNDCTL_DSP_SETFMT` ioctl, and :meth:`sync` to `SNDCTL_DSP_SYNC` (this can be useful when consulting the OSS documentation). If the underlying :c:func:`ioctl` fails, they all raise :exc:`OSError`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 185); *backlink*

Unknown interpreted text role "c:func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 185); *backlink*

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 185); *backlink*

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 185); *backlink*

Unknown interpreted text role "c:func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 185); *backlink*

Unknown interpreted text role "exc".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst, line 192)

Unknown directive type "method".

```
.. method:: oss_audio_device.nonblock()

   Put the device into non-blocking mode.  Once in non-blocking mode, there is no
   way to return it to blocking mode.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst, line 198)

Unknown directive type "method".

```
.. method:: oss_audio_device.getfmts()

   Return a bitmask of the audio output formats supported by the soundcard.  Some
   of the formats supported by OSS are:

   +-------------------------+---------------------------------------------+
   | Format                  | Description                                 |
   +=========================+=============================================+
   | :const:`AFMT_MU_LAW`    | a logarithmic encoding (used by Sun ``.au`` |
   |                         | files and :file:`/dev/audio`)               |
   +-------------------------+---------------------------------------------+
   | :const:`AFMT_A_LAW`     | a logarithmic encoding                      |
   +-------------------------+---------------------------------------------+
   | :const:`AFMT_IMA_ADPCM` | a 4:1 compressed format defined by the      |
   |                         | Interactive Multimedia Association          |
   +-------------------------+---------------------------------------------+
   | :const:`AFMT_U8`        | Unsigned, 8-bit audio                       |
   +-------------------------+---------------------------------------------+
   | :const:`AFMT_S16_LE`    | Signed, 16-bit audio, little-endian byte    |
   |                         | order (as used by Intel processors)         |
   +-------------------------+---------------------------------------------+
   | :const:`AFMT_S16_BE`    | Signed, 16-bit audio, big-endian byte order |
   |                         | (as used by 68k, PowerPC, Sparc)            |
   +-------------------------+---------------------------------------------+
   | :const:`AFMT_S8`        | Signed, 8 bit audio                         |
   +-------------------------+---------------------------------------------+
   | :const:`AFMT_U16_LE`    | Unsigned, 16-bit little-endian audio        |
   +-------------------------+---------------------------------------------+
   | :const:`AFMT_U16_BE`    | Unsigned, 16-bit big-endian audio           |
   +-------------------------+---------------------------------------------+

   Consult the OSS documentation for a full list of audio formats, and note that
   most devices support only a subset of these formats.  Some older devices only
   support :const:`AFMT_U8`; the most common format used today is
   :const:`AFMT_S16_LE`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst, line 235)

Unknown directive type "method".

```
.. method:: oss_audio_device.setfmt(format)

   Try to set the current audio format to *format*---see :meth:`getfmts` for a
   list.  Returns the audio format that the device was set to, which may not be the
   requested format.  May also be used to return the current audio format---do this
   by passing an "audio format" of :const:`AFMT_QUERY`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst, line 243)

Unknown directive type "method".

```
.. method:: oss_audio_device.channels(nchannels)
```

```
   Set the number of output channels to *nchannels*.  A value of 1 indicates
   monophonic sound, 2 stereophonic.  Some devices may have more than 2 channels,
   and some high-end devices may not support mono. Returns the number of channels
   the device was set to.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 251)

Unknown directive type "method".

```
.. method:: oss_audio_device.speed(samplerate)

   Try to set the audio sampling rate to *samplerate* samples per second.  Returns
   the rate actually set.  Most sound devices don't support arbitrary sampling
   rates.  Common rates are:

   +-------+-------------------------------------------+
   | Rate  | Description                               |
   +=======+===========================================+
   | 8000  | default rate for :file:`/dev/audio`       |
   +-------+-------------------------------------------+
   | 11025 | speech recording                          |
   +-------+-------------------------------------------+
   | 22050 |                                           |
   +-------+-------------------------------------------+
   | 44100 | CD quality audio (at 16 bits/sample and 2 |
   |       | channels)                                 |
   +-------+-------------------------------------------+
   | 96000 | DVD quality audio (at 24 bits/sample)     |
   +-------+-------------------------------------------+
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 273)

Unknown directive type "method".

```
.. method:: oss_audio_device.sync()

   Wait until the sound device has played every byte in its buffer.  (This happens
   implicitly when the device is closed.)  The OSS documentation recommends closing
   and re-opening the device rather than using :meth:`sync`.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 280)

Unknown directive type "method".

```
.. method:: oss_audio_device.reset()

   Immediately stop playing or recording and return the device to a state where it
   can accept commands.  The OSS documentation recommends closing and re-opening
   the device after calling :meth:`reset`.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 287)

Unknown directive type "method".

```
.. method:: oss_audio_device.post()

   Tell the driver that there is likely to be a pause in the output, making it
   possible for the device to handle the pause more intelligently.  You might use
   this after playing a spot sound effect, before waiting for user input, or before
   doing disk I/O.
```

The following convenience methods combine several ioctls, or one ioctl and some simple calculations.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst, line 298)

Unknown directive type "method".

```
.. method:: oss_audio_device.setparameters(format, nchannels, samplerate[, strict=False])

   Set the key audio sampling parameters---sample format, number of channels, and
   sampling rate---in one method call.  *format*,  *nchannels*, and *samplerate*
   should be as specified in the :meth:`setfmt`, :meth:`channels`, and
   :meth:`speed`  methods.  If *strict* is true, :meth:`setparameters` checks to
   see if each parameter was actually set to the requested value, and raises
   :exc:`OSSAudioError` if not.  Returns a tuple (*format*, *nchannels*,
   *samplerate*) indicating the parameter values that were actually set by the
   device driver (i.e., the same as the return values of :meth:`setfmt`,
   :meth:`channels`, and :meth:`speed`).

   For example,  ::

      (fmt, channels, rate) = dsp.setparameters(fmt, channels, rate)

   is equivalent to  ::

      fmt = dsp.setfmt(fmt)
      channels = dsp.channels(channels)
      rate = dsp.rate(rate)
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst, line 321)

Unknown directive type "method".

```
.. method:: oss_audio_device.bufsize()

   Returns the size of the hardware buffer, in samples.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst, line 326)

Unknown directive type "method".

```
.. method:: oss_audio_device.obufcount()

   Returns the number of samples that are in the hardware buffer yet to be played.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst, line 331)

Unknown directive type "method".

```
.. method:: oss_audio_device.obuffree()

   Returns the number of samples that could be queued into the hardware buffer to
   be played without blocking.
```

Audio device objects also support several read-only attributes:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst, line 339)

Unknown directive type "attribute".

```
.. attribute:: oss_audio_device.closed

   Boolean indicating whether the device has been closed.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 344)

Unknown directive type "attribute".

```
.. attribute:: oss_audio_device.name

   String containing the name of the device file.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 349)

Unknown directive type "attribute".

```
.. attribute:: oss_audio_device.mode

   The I/O mode for the file, either ``"r"``, ``"rw"``, or ``"w"``.
```

Mixer Device Objects

The mixer object provides two file-like methods:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 362)

Unknown directive type "method".

```
.. method:: oss_mixer_device.close()

   This method closes the open mixer device file.  Any further attempts to use the
   mixer after this file is closed will raise an :exc:`OSError`.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 368)

Unknown directive type "method".

```
.. method:: oss_mixer_device.fileno()

   Returns the file handle number of the open mixer device file.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 372)

Unknown directive type "versionchanged".

```
.. versionchanged:: 3.2
   Mixer objects also support the context management protocol.
```

The remaining methods are specific to audio mixing:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 379)

Unknown directive type "method".

```
.. method:: oss_mixer_device.controls()

   This method returns a bitmask specifying the available mixer controls ("Control"
   being a specific mixable "channel", such as :const:`SOUND_MIXER_PCM` or
   :const:`SOUND_MIXER_SYNTH`).  This bitmask indicates a subset of all available
   mixer controls---the :const:`SOUND_MIXER_\*` constants defined at module level.
   To determine if, for example, the current mixer object supports a PCM mixer, use
   the following Python code::
      mixer=ossaudiodev.openmixer()
      if mixer.controls() & (1 << ossaudiodev.SOUND_MIXER_PCM):
          # PCM is supported
          ... code ...

   For most purposes, the :const:`SOUND_MIXER_VOLUME` (master volume) and
   :const:`SOUND_MIXER_PCM` controls should suffice---but code that uses the mixer
   should be flexible when it comes to choosing mixer controls.  On the Gravis
   Ultrasound, for example, :const:`SOUND_MIXER_VOLUME` does not exist.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 399)

Unknown directive type "method".

```
.. method:: oss_mixer_device.stereocontrols()

   Returns a bitmask indicating stereo mixer controls.  If a bit is set, the
   corresponding control is stereo; if it is unset, the control is either
   monophonic or not supported by the mixer (use in combination with
   :meth:`controls` to determine which).

   See the code example for the :meth:`controls` function for an example of getting
   data from a bitmask.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 410)

Unknown directive type "method".

```
.. method:: oss_mixer_device.reccontrols()

   Returns a bitmask specifying the mixer controls that may be used to record.  See
   the code example for :meth:`controls` for an example of reading from a bitmask.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 416)

Unknown directive type "method".

```
.. method:: oss_mixer_device.get(control)

   Returns the volume of a given mixer control.  The returned volume is a 2-tuple
   ``(left_volume,right_volume)``.  Volumes are specified as numbers from 0
   (silent) to 100 (full volume).  If the control is monophonic, a 2-tuple is still
   returned, but both volumes are the same.

   Raises :exc:`OSSAudioError` if an invalid control is specified, or
   :exc:`OSError` if an unsupported control is specified.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 427)

Unknown directive type "method".

```
.. method:: oss_mixer_device.set(control, (left, right))

   Sets the volume for a given mixer control to ``(left,right)``. ``left`` and
   ``right`` must be ints and between 0 (silent) and 100 (full volume).  On
   success, the new volume is returned as a 2-tuple. Note that this may not be
   exactly the same as the volume specified, because of the limited resolution of
   some soundcard's mixers.

   Raises :exc:`OSSAudioError` if an invalid mixer control was specified, or if the
   specified volumes were out-of-range.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 439)

Unknown directive type "method".

```
.. method:: oss_mixer_device.get_recsrc()

   This method returns a bitmask indicating which control(s) are currently being
   used as a recording source.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]ossaudiodev.rst`, line 445)

Unknown directive type "method".

```
.. method:: oss_mixer_device.set_recsrc(bitmask)

   Call this function to specify a recording source.  Returns a bitmask indicating
   the new recording source (or sources) if successful; raises :exc:`OSError` if an
   invalid source was specified.  To set the current recording source to the
   microphone input::

      mixer.setrecsrc (1 << ossaudiodev.SOUND_MIXER_MIC)
```