# jack_playfile

## Name

`jack_playfile` — play audio files with JACK

## Synopsis

*jack_playfile* [OPTION]... FILE...

## DESCRIPTION

*jack_playfile* is a simple audio file player for JACK.

Main features:

- Plays most RIFF, AIFF PCM wave files, also plays FLAC, Ogg Vorbis, Opus and MP3 files
- High quality resampling to match JACK sample rate (can be turned off)
- Keyboard control while playing (can be turned off)
- Fast and frame accurate seeking
- Supports multichannel files
- Adapts to a broad range of file formats and JACK settings
- Versatile command line options and live control
- Survives JACK restart (resume playing at the same position)

Please note:

- Opus files are limited to 8 channels by libopusfile.
- FLAC files are limited to 8 channels per stream by spec.
- MP3 supports a maximum of two channels (no "MP3 Surround" support).

Best results are achieved when playing back 32-bit float RIFF wave files at JACK sample rate (without resampling).

*jack_playfile* can read and play local audio files only. There is currently no support for playlists. No DSP other than optional resampling is applied to the audio signal. To meter the signal, control volume and apply equalization look out for corresponding JACK clients to chain with *jack_playfile*. This plugin collection is a good starting point: https://github.com/x42/x42-plugins

The term "frame" refers to samples of multiple channels i.e. one frame includes one sample of every channel.

**THIS PROGRAM COMES WITHOUT ANY WARRANTY**

/!\ *jack_playfile* is still work in progress. It can happen that output is of unexpected nature (i.e. noise, unaligned buffers etc.) in special circumstances due to bugs in the code. If you find such cases that can be reproduced, please consider reporting a bug (see *BUGS*).

*jack_playfile* version is 0.83

# OPTIONS

**-h, --help** (w/o argument)

> Display help and quit.

**-v, --version** (w/o argument)

> Display version and quit.

**-n, --name** (string)

> JACK client name. **Default:** "jack_playfile"

**-s, --sname** (string)

> JACK server name to start *jack_playfile* in a specific JACK server. **Default:** "default"

**-N, --noconnect** (w/o argument)

> Don't connect to JACK playback ports. **Default:** *jack_playfile* tries to connect all available channels to all available physical output/playback ports 1:1.

**-E, --noreconnect** (w/o argument)

> Don't wait for JACK to reconnect as a client. **Default:** If JACK goes down, *jack_playfile* waits for the server to come back and then continues operation. If *jack_playfile* was playing when JACK went down, it will continue right at the position where it was before JACK went down. If JACK settings were changed between a restart, *jack_playfile* tries to adapt to the new settings as good as possible.

**-D, --nocontrol**:: (w/o argument)

> Disable keyboard control. **Default:** *jack_playfile* accepts keyboard input while playing. For a detailed overview on available control actions, see *KEYBOARD SHORTCUTS* below or hit *F1* or *h* while *jack_playfile* is started and control is enabled.

**-R, --noresampling** (w/o argument)

> Disable resampling. If resampling is disabled, the samples read from given file are treated in the JACK sample rate domain without any modification. **Default:** The samples read from given file are

resampled to match JACK sample rate. This makes files play at the expected pitch and tempo. Best results are achieved when the file sample rate matches JACK's.

**-p, --paused** (w/o argument)

Start paused. **Default:** Start playing after successful file open and connection to JACK.

**-m, --muted** (w/o argument)

Start muted. **Default:** Not muted, i.e. hear sound.

**-l, --loop** (w/o argument)

Enable loop. If end of track is reached (given offset+count), start over at offset. **Default:** disabled. What happens when the end of a track is reached depends on other conditions.

**-e, --pae** (w/o argument)

Pause at end: If end of track is reached (given offset+count), don't quit but pause playback instead. If loop is disabled, the position will correspond to end. If loop is enabled, the position will be set to start. While paused at end, play, toggle play and forward seeks are blocked i.e. not executed. **Default:** Off. If the end of a track is reached and loop is not enabled, *jack_playfile* is done and will quit.

**-j, --transport** (w/o argument)

JACK transport: If enabled, *jack_playfile* will follow JACK transport signals "Stopped" and "Rolling", mapped to paused and playing. In the opposite direction, *jack_playfile* sends transport signals to JACK when the play status changes, i.e. spacebar was hit to toggle play. No positional information from JACK transport is processed or sent by *jack_playfile* to JACK. The position of *jack_playfile* is independent, i.e. it can loop while the JACK transport position increments linearly. **Default:** Off.

**-f, --frames** (w/o argument)

Show time as frames. A number of (multichannel) frames in native file sample rate. Note: The sample rate for Opus and MP3 files is always 48000. This is a consequence of how *jack_playfile* treats these formats, not a limitation of the formats. **Default:** Show time as seconds.

**-a, --absolute** (w/o argument)

Show absolute time. The frames and seconds indication relate to absolute position 0 of audio samples in file. **Default:** Show relative time. Frames and seconds indication relate to given offset of audio samples in file (offset=relative position 0).

**-r, --remaining**:: (w/o argument)

Show remaining time. How many frames or seconds until the end of the track is reached (offset+count). **Default:** Show elapsed time. How many frames or seconds away from start (offset).

**-k, --noclock** (w/o argument)

Disable clock display. This can save some resources. **Default:** Enabled. The display is updated approximately with every JACK cycle.

**-o, --offset** (integer)

Set frame offset: The first n (multichannel) frames (given number) in the file will be ignored. The frame offset relates to native file sample rate (not JACK's). The offset is relative frame/time position 0 and will be used for seeking to start and looping. Offsets beyond the end of available frames in the file will be set to 0. **Default:** 0 (At first audio sample in file).

**-c, --count** (integer)

Frame count: A number of (multichannel) frames to play from given offset position. The frame count relates to native file sample rate (not JACK's). Counts resulting in positions beyond the end of available frames in the file will be set to default. **Default:** All available frames, full length of track (respecting given offset).

**-O, --choffset** (integer)

Set channel offset: The first n channels (given number) in the file will be ignored. Offsets beyond the total file channel count result in *jack_playfile* skipping the file, i.e. nothing will be set to a sane value and nothing will be played. **Default:** 0 (At first channel in file).

**-C, --chcount** (integer)

Channel count: How many channels to read from file, counting from offset (--choffset). The channel count defines how many output ports *jack_playfile* will expose to JACK. If the file can deliver less channels than offset+count, the remaining JACK output channels will be filled with zero to match given --chcount. This can be handy if files with different channel counts are played in a row (no port recreation or reconnection involved). A fixed channel count i.e. --chcount 4 will ensure that even if the first file initializes *jack_playfile* with only one JACK output channel (mono file), a following 4-channel file can still play all its 4 channels through the available JACK output ports. **Default:** All available channels (respecting given offset). This means the file channel count (-offset) sets the number of JACK output ports *jack_playfile* will have.

**-L, --libs** (w/o argument)

Show license and library information (see *LIBRARIES AND DEPENDENCIES*)

Count and offset relate to the sample rate and duration (frame count) indicated when *jack_playfile* starts up. For the audio formats Opus and MP3, frame offsets and counts always relate to a fixed sample rate of 48k.

To play multiple files, each file can be appended to the command line, i.e.:

```
$ jack_playfile a.wav b.ogg c.flac
```

Please note that files containing spaces or special characters should be enclosed in ″ and if ″ is part of the filename, it needs escaping like \″.

Of course all the shell filename expansion mechanisms can be used, so that

```
$ jack_playfile *.wav 0??.ogg
```

will play all files matching the patterns.

If multiple files are available as arguments, they will be played in a row without recreating or reconnecting JACK ports. Using keyboard control < and > will browse through the list of files. If a file can't be played, the next file will be tried until there is a valid file or no more files left to try.

## KEYBOARD SHORTCUTS

- Start refers to the relative start given with --offset which is 0 by default. Relative start is always 0.
- End refers to relative end which is always equal to --count.
- Default Values are marked with "*"

**h, f1**

  Help (this screen)

**space**

  Toggle play/pause

**enter**

  Play

**(<) arrow left**

  Seek one step backward

**(>) arrow right**

  Seek one step forward

**(^) arrow up**

  Increment seek step size

**(v) arrow down**

  Decrement seek step size

**home**

  Seek to start

**0**

  Seek to start and pause

**backspace**

  Seek to start and play

**end**

    Seek to end

**< less than**

    Load previous file

**> greater than**

    Load next file

**m**

    Toggle mute on/off*

**l**

    Toggle loop on/off*

**p**

    Toggle pause at end on/off*

**j**

    Toggle JACK transport on/off*

**c**

    Toggle clock display on*/off

**, comma**

    Toggle clock seconds*/frames

**. period**

    Toggle clock absolute*/relative

**- dash**

    Toggle clock elapsed*/remaining

**q**

    Quit

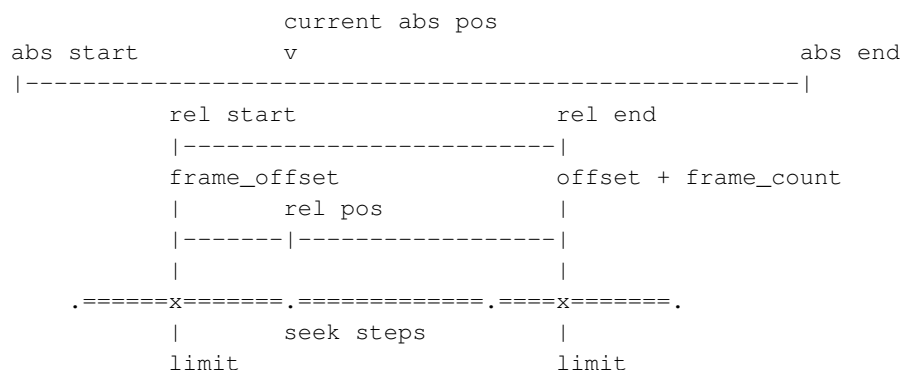If the clock is set to seconds, changing the seek step size will use the following grid:

- 0.001, 0.010, 0.100, 1, 10*, 60, 600, 3600 (seconds)

If the clock is set to frames, changing the seek step size will use the following grid:

- 1*, 10, 100, 1000, 10k, 100k, 1000k, 10M, 100M (frames)

## TIMELINE

The relation of absolute and relative start and end using offset and count, limited seek steps:

```
                    current abs pos
abs start           v                                   abs end
|-------------------------------------------------------|
          rel start                   rel end
          |-------------------------|
          frame_offset               offset + frame_count
          |         rel pos          |
          |-------|-----------------|
          |                         |
     .======x=======.=============.====x=======.
          |         seek steps     |
          limit                     limit
```

## EXAMPLES

• Play RIFF wave file:

```
$ jack_playfile audio.wav
```

Example output of *jack_playfile*:

```
file:         audio.wav  (#1/1)
size:         57274264 bytes (57.27 MB)
format:       Microsoft WAV format (little endian)
              Signed 16 bit data (0x00010002)
duration:     00:05:24.684 (14318555 frames)
sample rate: 44100
channels:     2
data rate:   176400.0 bytes/s (0.18 MB/s)
frame_count set to 14318555 (all available frames)
playing frames offset, count, end: 0 14318555 14318555
playing channels offset, count, end, file: 0 2 2
JACK sample rate: 48000
JACK period size: 128 frames
JACK cycles per second: 375.00
JACK output data rate: 384000.0 bytes/s (0.38 MB/s)
total byte out_to_in ratio: 2.176871
resampler out_to_in ratio: 1.088435
autoconnect: jack_playfile-01:output_1 -> firewire_pcm:000a9200d6012385_MainOut 1L_out
autoconnect: jack_playfile-01:output_2 -> firewire_pcm:000a9200d6012385_MainOut 2R_out
>  playing       S rel    10          4.3  (00:00:04.321)
```

(the last line is being updated in an interval)

Note on ratios:

- byte_out_to_in_ratio: Bytes delivered to JACK divided by bytes read from file. For lossy compressed formats (Ogg, Opus, MP3), the total file size is used for calculation.

- resampler out_to_in ratio: JACK sample rate divided by file sample rate.

- data_rate: Bytes to read from file per second to satisfy constant flow to JACK output. For lossy compressed formats (Ogg, Opus, MP3), the total file size is used for calculation.

Legend (example prompt):

```
|| paused   JMLP  S rel 0.001       943.1  (00:15:43.070)
^          ^^^^  ^ ^   ^      ^      ^      ^ ^               ^
1          2345  6 7   8      9      10     9 11              12

 1): status playing '>', paused '||' or seeking '...'
 2): JACK transport on/off 'J' or ' '
 3): mute on/off 'M' or ' '
 4): loop on/off 'L' or ' '
 5): pause at end on/off 'P' or ' '
 6): time and seek in seconds 'S' or frames 'F'
 7): time indication 'rel' to frame_offset or 'abs'
 8): seek step size in seconds or frames
 9): time elapsed ' ' or remaining '-'
10): time in seconds or frames
11): time in HMS.millis
12): keyboard input indication (i.e. seek)
```

- Play Opus file, starting at an offset of 480000 frames (10 seconds), playing 48000 frames (1 second), showing remaining absolute time, pause at end and loop:

  ```
  $ jack_playfile -o 480000 -c 48000 -r -a --pae -l audio.opus
  ```
- Play a short snippet of all wave files in a directory, using only first channel

  ```
  $ jack_playfile --offset 5000 --count 10000 --chcount 1 samples/*.wav
  ```

## ERROR MESSAGES

*jack_playfile* does not automatically start a JACK default server if there is none running. If *jack_playfile* is started with the option --noreconnect, this will lead to the following message:

```
Cannot connect to server socket err = No such file or directory
Cannot connect to server request channel
jack server is not running or cannot be started
jack_client_open() failed, status = 0x11
Unable to connect to JACK server
```

Simply start JACK before using *jack_playfile*. If --noreconnect is not present, *jack_playfile* will wait until JACK is reachable:

```
waiting for connection to JACK server...
```

To find out how to start JACK, see *jackd* manpage and tutorials on http://jackaudio.org. There is an excellent graphical JACK control program called *qjackctl*, http://qjackctl.sourceforge.net/.

In a nutshell:

- Starting JACK in realtime mode from a terminal with ALSA backend (i.e. onboard audio), using first available audio card

  ```
  $ jackd -R -dalsa -r48000 -p512 -n3 -dhw:0
  ```

This can fail for several reasons:

- *jackd* is not installed ⟶ check repository for "jackd" or similar and install
- The default JACK server is already running ⟶ no need to start again
- The device at hw:0 is already in use by another audio server, i.e. *pulseaudio* ⟶ try to stop pulse or try another card (i.e. hw:1)
- You don't have permissions to run *jackd* because of security limits (rtprio, memlock) ⟶ check /etc/security/limits.d/audio.conf, check that user is part of group "audio", eventually log out and login to make group changes take effect.
- Other reason

If *jackd* is installed, it's possible to start JACK with a dummy backend where no physical audio devices are involved:

- Starting JACK with dummy backend, server name "virtual"

  ```
  $ jackd --name virtual -ddummy -r4800 -p128
  ```
- Telling jack_playfile to use JACK server "virtual"

  ```
  $ jack_playfile --sname virtual audio.ogg
  ```

If you have trouble starting *jackd* on your host, please consult JACK FAQ at http://jackaudio.org/faq/ and join IRC #jack on freenode. There is a mailinglist too.

*jack_playfile* returns 0 on regular program exit, or 1 if there was an error.

## PROGRAM STATUSES

- Initializing (JACK, first/next/prev file)
- Paused (‖)

- Playing (>)
- Seeking (. . .)
- Shutting down

## PROGRAM LIFE CYCLE

*jack_playfile* procedure:

0) Initializing, starting up with given parameters

1) Trying to open given file with several decoders, quit on fail

2) Check if JACK libraries are available on host, quit on fail

3) Eventually wait for JACK server to become available

4) Register JACK client, register ports, optionally connect ports, quit on fail

5) Start operation based on playback settings (paused, muted etc.)

6) Eventually stop operation if JACK away

7) Eventually resume operation if JACK available

8) Eventually play next or previous file in args list

9) Release resources and quit nicely if all done or quit was requested

During all operation *jack_playfile* tries to prevent to cause JACK X-runs or *jack_playfile* internal buffer underflows. It's very likely that underruns happen inside *jack_playfile* though (not enough data available to play in buffer), i.e. while seeking, during startup or shutdown. *jack_playfile* relies on constant fast file read access. Files can be copied to a RAM disk (i.e. /dev/shm/) before playing to prevent physical disk access on non-SSD disks.

## LIBRARIES AND DEPENDENCIES

Major audio libraries *jack_playfile* depends on:

- JACK audio connection kit - http://jackaudio.org/ - *jack_playfile* works exclusively with JACK as audio backend. JACK is available for Linux, Windows and OSX.
- libsndfile - http://www.mega-nerd.com/libsndfile/ - This is the main library to read audio files.
- libzita-resampler - http://kokkinizita.linuxaudio.org/linuxaudio/ - High quality resampler.
- libopus, libopusfile - http://www.opus-codec.org/ - RFC 6716, incorporates SILK & CELT codecs.
- libvorbisfile - http://xiph.org/vorbis/ - Fast seeking in Ogg Vorbis files
- libmpg123 - http://www.mpg123.org/ - (optional due to patent foo)

Libraries abstracted by libsndfile:

- libFLAC - http://xiph.org/flac/
- libvorbis, libvorbisenc - http://xiph.org/vorbis/
- libogg - http://xiph.org/ogg/

## RESOURCES

Github: https://github.com/7890/jack_tools in subdirectory jack_playfile

## BUGS

Please report any bugs as issues to the github repository. Patches and pull requests are welcome.

## SEE ALSO

**jackd**(1) **jack_capture**(1) **sndfile-info**(1) **zresample**(1) **flac**(1) **oggenc**(1) **opusenc**(1) **mpg123**(1) **sox**(1) **patchage**(1)

## AUTHORS

Thomas Brand <tom@trellis.ch (mailto:tom@trellis.ch)>

Last Update: Sat Aug 15 17:30:25 CEST 2015

# COPYING