jack_playfile

jack_playfile(1)

Name

`jack_playfile` — play audio files with JACK

Synopsis

jack_playfile [OPTIONS] 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)
- Frame accurate and fast 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:

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

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

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 version is 0.81

OPTIONS

--help (w/o argument)

Display help and quit.

--version (w/o argument)

Display version and quit.

--name (string)

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

--sname (string)

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

--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.

--noreconnect (w/o argument)

Don't wait for JACK to re-connect as a client. **Default:** If JACK goes down, *jack_playfile* waits for the server to come back and continue operation. If *jack_playfile* was playing when JACK went down, it should 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.

--nocontrol

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.

--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.

--paused (w/o argument)

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

--muted (w/o argument)

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

--loop (w/o argument)

Enable loop. If end of track is reached (given offset+count), start over at offset. Default not enabled. What happens when end of track reached depends on other conditions.

--pae (w/o argument)

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

--frames (w/o argument)

Show time as frames. A number of (multichannel) frames in native file sample rate. **Default:** Show time as seconds.

--absolute (w/o argument)

Show absolute time. Frame and second indications relate to absolute 0 of audio samples in file. **Default:** Show relative time. Frame and second indications relate to given offset of audio samples in file (offset=relative 0).

--remaining

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).

--noclock (w/o argument)

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

--offset (integer)

Set frame offset: A number of (multichannel) frames to seek before start reading from file. The frame offset relates to native file sample rate (not JACKs). The offset is relative frame/time 0 and will be used for seeking to start and looping. **Default:** 0 (At start of audio samples).

--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 JACKs). **Default:** All available frames, full length of track (respecting given offset).

Counts and offset relate to the indicated sample rate and duration (frame count). For the audio formats opus and mp3, frame offsets and counts always relate to a fixed sample rate of 48k.

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 made 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

m

Toggle mute on/off*

l

Toggle loop on/off*

p

Toggle pause at end 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 clock set to seconds, changing the seek step size is using the following grid:

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

If clock set to frames, changing the seek step size is using 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
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 from/to/length: 0 14318555 14318555
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   MLP  S rel 0.001       943.1  (00:15:43.070)
^           ^^^  ^ ^   ^     ^     ^     ^ ^             ^
1           234  5 6   7     8     9     8 10            11
```

```
 1): status playing '>', paused '||' or seeking '...'
 2): mute on/off 'M' or ' '
 3): loop on/off 'L' or ' '
 4): pause at end on/off 'P' or ' '
```

```
 5): time and seek in seconds 'S' or frames 'F'
 6): time indication 'rel' to frame_offset or 'abs'
 7): seek step size in seconds or frames
 8): time elapsed ' ' or remaining '-'
 9): time in seconds or frames
10): time in HMS.millis
11): 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 --offset 480000 --count 48000 --remaining --absolute --pae --loop audio.opus

ERROR MESSAGES

jack_playfile does not automatically start a JACK default server if there is none running. If *jack_playfile* is started 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 given, *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 manager tool 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
```

That 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

- 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.

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

PROGRAM STATUSES

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

During all operation *jack_playfile* tries to prevent to produce JACK x-runs or *jack_playfile* internal buffer underflows. It's very likely that underruns happen inside *jack_playfile* though (not enough data to play in buffer), i.e. while seeking.

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.
- 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/
- libmpg123 - http://www.mpg123.de/ - (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

BUGS

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

SEE ALSO

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

AUTHORS

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

Last Update: Sat Aug 8 00:29:58 CEST 2015

COPYING