NAME

ezstream − source client for Icecast with external de-/encoder support

SYNOPSIS

ezstream [**−hqVv**] [**−c** *configfile*]

DESCRIPTION

The **ezstream** utility is a source client for the Icecast media streaming server. In its basic mode of operation, it streams media files and data from standard input "as-is" — such as Ogg Vorbis, Ogg Theora and MP3 — to a server. It can also use various external decoders and encoders to reencode from one format to another, and stream the result. The only requirement is that the external programs support writing to or reading from standard input, and can be used from the command line.

Command line parameters

−c *configfile*
Use the XML configuration in `configfile`. (Mandatory.)

−h Print a summary of available command line parameters with short descriptions and exit.

−q Be more quiet. Suppress the output that external programs send to standard error.

−V Print the **ezstream** version number and exit.

−v Produce more verbose output from **ezstream**. Use twice for even more verbose output.

When the **−q** and **−v** parameters are provided simultaneously, an additional line of information about the currently streamed file — playlist position, approximate playing time and bitrate — is displayed.

Runtime control

On POSIX systems, **ezstream** offers limited runtime control via signals. By sending a signal to the ezstream process, e.g. with the `kill`(1) utility, a certain action will be triggered.

SIGHUP
Rereads the playlist file after the track that is currently streamed. If the playlist is not to be shuffled, **ezstream** attempts to find the previously streamed file and continue with the one following it, or restarts from the beginning of the list otherwise.

SIGUSR1
Skips the currently playing track and moves on to the next in playlist mode, or restarts the current track when streaming a single file.

SIGUSR2
Triggers rereading of metadata for the stream by running the program or script specified in `<metadata_progname/>` (see below.) This is the only meaningful signal when streaming from standard input.

CONFIGURATION FILE SYNTAX

The **ezstream** utility uses a simple XML configuration file format. It has a tree-like structure and is made up of *XML elements*. Of all the possible XML features, only regular elements that contain text or other elements, and comments, appear in an **ezstream** configuration file.

Each element in the configuration file consists of a *start tag*, its content and an *end tag*. For example:

```
<filename>playlist.m3u</filename>
<!-- XML comments look like this. -->
```

XML CONFIGURATION

In this section, each available element is listed and described. Note that for this purpose, elements are introduced in their short, i.e. empty form. In the configuration file, they need to be used as *start tag + content + end tag*, like in the introductory example shown above.

Root element

<ezstream />

(Mandatory.) The configuration file's root element. It contains all other configuration elements.

Global configuration elements

Each of the global configuration elements have the `<ezstream/>` element as their parent.

<url />

(Mandatory.) Specifies the location and mountpoint of the Icecast server, to which the stream will be sent. The content must be of the form `http://server:port/mountpoint` For example:

```
<url>http://example.com:8000/stream.ogg</url>
```

<sourcepassword />

(Mandatory.) Sets the source password for authentication with the Icecast server.

<format />

(Mandatory.) This element has two different meanings, depending on whether reencoding is enabled or not. It specifies the *output format* of the stream if reencoding is enabled. Otherwise, it specifies the *input format* of **all** input files. Recognized and supported values for output stream formats are **VORBIS**, **MP3** and **THEORA**. Other values will be ignored and cause **`ezstream`** to simply pass through the data, which may or may not work.

<filename />

(Mandatory.) Set the path and name of a single media file, a playlist, the name of an external program (see below), or the keyword `stdin` for streaming from standard input. Playlists are recognized by their filename extension and end with either *.m3u* or *.txt*.

A playlist consists of filenames, one entry per line. Comments in playlists are introduced by a '#' sign at the beginning of a line and ignored by **`ezstream`**.

<playlist_program />

(Optional.) Set to **1** (one) to indicate that the file in `<filename/>` is actually an executable program or script. If set to **0** (zero), `<filename/>` content is assumed to be a media file, playlist file or the keyword `stdin` (the default).

See the **SCRIPTING** section for details on how the playlist program must behave.

<shuffle />

(Optional.) Set to **1** (one) to randomly shuffle the entries of the playlist specified in `<filename/>`. Files are played sequentially if set to **0** (zero) or when the `<shuffle/>` element is absent. This option will be ignored if `<playlist_program/>` is set to 1 (one.)

<metadata_progname />

(Optional.) Set the path and name of an executable program or script that should be used by **`ezstream`** to set the metadata of the stream. The program is automatically queried when a new track is streamed, or whenever the **SIGUSR2** signal is received.

If the `<metadata_progname/>` element is present in the configuration, no attempts will be made to read metadata from files that are being streamed. If this behavior is not desired, it should be removed or commented out in the configuration file.

See the **SCRIPTING** section for details on how the metadata program must behave.

<metadata_format />
(Optional.) Set the format of the string that should be used for the '`@M@`' placeholder when setting metadata with an external program or script via `<metadata_progname/>`.

See the **METADATA** section for details on how metadata is handled by **`ezstream`**.

<stream_once />
Set to **1** (one) in order to stream the content of `<filename/>` only once, and to **0** (zero) for continuous streaming (the default) .

<reconnect_tries />
Set how many attempts should be made to reconnect to the Icecast server in case the connection is interrupted. The default is to try indefinitely, which is equal to setting this configuration option to **0** (zero) .

<svrinfoname />
(Optional.) Set the name of the broadcast. Informational only.

<svrinfourl />
(Optional.) Set the URL of the web site associated with the broadcast. Informational only.

<svrinfogenre />
(Optional.) Set the genre of the broadcast. Informational only, used for YP.

<svrinfodescription />
(Optional.) Set the description of the broadcast. Informational only, used for YP.

<svrinfobitrate />
(Optional.) Set the bitrate of the broadcast. This setting is also purely informational and only used for YP. The value is set by the user and not **`ezstream`**, and should match the bitrate of the stream.

<svrinfoquality />
(Optional.) Set the quality setting of an Ogg Vorbis broadcast. Informational only and needs to be set by the user, used for YP.

<svrinfochannels />
(Optional.) Set the number of audio channels in the broadcast, e.g. **1** (one) for mono or **2** for stereo. Informational only and needs to be set by the user, used for YP.

<svrinfosamplerate />
(Optional.) Set the sample rate of the broadcast. Informational only and needs to be set by the user, used for YP.

<svrinfopublic />
(Optional.) Set to **1** (one) if the broadcast may be listed in a public YP directory. If set to **0** (zero) , the Icecast server will not submit this stream to a YP directory, which is also the default if the `<svrinfopublic/>` element is absent.

<reencode />
(Optional.) Element that contains child elements, which specify if and how reencoding should be done.

Reencoding settings

Each of the reencoding configuration elements have the `<reencode/>` element as their parent.

<enable />

Set to **1** (one) to enable reencoding. If set to **0** (zero), no reencoding will be done, which is also the default if the `<enable/>` element is absent.

<encdec />

Element that contains child elements, which specify how to decode and encode a certain media file format for streaming. Each format is described by a separate `<encdec/>` element.

Decoder/Encoder settings

Each of the decoder/encoder configuration elements have the `<encdec/>` element as their parent.

<format />

This element is used by **ezstream** to find the appropriate encoder for the output stream format specified in the `<format/>` element inside the global configuration. For consistency reasons, it is recommended that this element is always supplied, even for currently unsupported output formats, with content such as **VORBIS**, **MP3**, **THEORA**, **FLAC**, et cetera.

<match />

Set the filename extension used to identify a given media file format. This allows **ezstream** to find the appropriate decoder for a given file. Should be set to *.mp3* for MP3, *.flac* for FLAC, *.ogg* for Ogg Vorbis, and so on.

<decode />

Set the command to decode the specified media file format to raw data and send it to standard output. During runtime, the placeholder '`@T@`' is replaced with the name of the media file, as it is specified in the `<filename/>` element or contained in a playlist file. It should always be enclosed in quotes, to prevent problems with filenames that contain whitespaces.

Metadata placeholders can be used in the `<decode/>` element as well, for combined de-/encoder programs that produce streamable data. See the **METADATA** section for details on how metadata is handled by **ezstream**.

For example, to decode Ogg Vorbis files using the **oggdec** utility:

```
<decode>oggdec -R -o - "@T@"</decode>
```

<encode />

Set the command to encode raw data, received from standard input, to the specified stream format.

Metadata placeholders can be used in the `<encode/>` element. For details about using metadata in **ezstream**, see below in the **METADATA** section.

For example, to encode an Ogg Vorbis stream using the quality setting 1.5 with the **oggenc** utility:

```
<encode>oggenc -r -q 1.5 -t "@M@" -</encode>
```

SCRIPTING

The **ezstream** utility provides hooks for externally controlled playlist and metadata management. This is done by running external programs or scripts that need to behave in ways explained here.

Common Rules

- The program must be an executable file.
- The program must write one line to standard output and exit.
- The program must not require arbitary command line options to function. A wrapper script must be used if there is no other way.

Playlist Programs

- The program must return only filenames, with one filename per execution.
- The program should not return an empty line unless **ezstream** is supposed to know that the end of the playlist has been reached. This is significant when the `<stream_once/>` option is enabled.

Metadata Programs

- The program must not return anything (just a newline character is okay) if it is called by **ezstream** with a command line parameter that the program does not support.
- When called without command line parameters, the program should return a complete string that should be used for metadata.
- When called with the command line parameter "`artist`", the program should return only the artist information of the metadata. (Optional.)
- When called with the command line parameter "`title`", the program should return only the title information of the metadata. (Optional.)

METADATA

The main tool for handling metadata with **ezstream** is placeholders in decoder and encoder commands that are replaced with real content during runtime. The tricky part about is that one placeholders has to be handled differently depending on where the metadata comes from. This section will explain each possible scenario.

Metadata Placeholders

@T@
Replaced with the media file name. Required in `<decode/>` and is available in `<metadata_format/>`.

@M@
Replaced with a metadata string. See below for a detailed explanation. Available in `<decode/>` and `<encode/>`.

@a@
Replaced with the artist information. Available in `<decode/>`, `<encode/>` and `<metadata_format/>`.

@t@
Replaced with the title information. Available in `<decode/>`, `<encode/>` and `<metadata_format/>`.

@s@
Replaced with the string returned by `<metadata_progname/>` when called without any command line parameters. Available only in `<metadata_format/>`.

The @M@ Placeholder

While all other placeholders are simply replaced with whatever data they are associated with, '`@M@`' is context-sensitive. The logic used by **ezstream** is the following:

```
If ('@M@ is present')
    If ('<metadata_progname/>' AND '<metadata_format/>')
        Replace with format string result.
    Else
        If (NOT '<metadata_progname/>' AND '@t@ is present')
            Replace with empty string.
        else
            Replace with generated metadata string.
```

```
                Endif
            Endif
        Endif
```

The generated metadata string for '`@M@`' is of the format ''*Artist - Title*'', if both artist and title information is available. If one of the two is missing, the available one is displayed without a leading or trailing dash, e.g. just ''*Artist*''. If neither artist nor title are available, the name of the media file — without its file extension — is used.

Metadata Caveats

It is possible to generate strange results with odd combinations of placeholders, external metadata programs and updates during runtime via **SIGUSR2**. If things start to become just confusing, simplify.

Metadata updates during runtime are done with a relatively broken feature of libshout. Additional metadata information that is already present in the stream sent via **`ezstream`** is usually destroyed and replaced with the new data. It is not possible to properly discern between artist and title information, which means that anything set with the **SIGUSR2** feature will continue to end up entirely in the "*Title*" field of a stream.

FILES

`ezstream-0.4.0-win32/examples`	Directory containing example configuration files for various uses of **`ezstream`**, as well as example playlist and metadata scripts.

AUTHORS

`ezstream` was written by:

Ed Zaleski <oddsock@oddsock.org>
Moritz Grimm <gtgbr@gmx.net>

This manual was written by Moritz Grimm.