

Oyez User Documentatin (0.1.8)

Seth de l'Isle

18th July 2002

Contents

0.1	Meta Information	1
0.2	Versions and Contact Information	1
0.3	What is a streaming audio server?	2
0.4	Installation and Configuration	2
0.4.1	Summary	2
0.4.2	Dependencies	2
0.4.3	Download	2
0.4.4	Prepare the Base Directory	3
0.4.5	Editing the Configuration Files	3
0.4.6	Run Oyez	6
0.5	Creating Links to an Oyez Stream	6
0.5.1	Direct Links to an Oyez Stream	7
0.5.2	Playlists and MIME Types	7
0.5.3	.m3u playlists	8
0.5.4	.pls Playlists	8

0.1 Meta Information

This document describes Oyez, a streaming audio server written in Python. Oyez is a standalone server capable of playing playlists of Vorbis or MP3, or of accepting input from an external encoder such as Oggenc or Lame. It covers Oyez version 0.1.6.

0.2 Versions and Contact Information

Versions

Current versions of this document are available on the Oyez homepage <http://ubertechnique.com/seth/oyez>.

Contact Information

You can contact the author of this document via email, the address is szoth@ubertechnique.com. Information pertaining to Oyez is available at the Oyez home page. If you have questions this document fails to answer, please contact me directly, or join the mailing list.

Details on the oyez mailing list are available on the Oyez homepage. You will also find links to an archive of mailing list discussion there.

0.3 What is a streaming audio server?

A streaming audio server allows multiple clients to listen to the same audio data at the same time. A streaming audio server is necessary for broadcasting live audio for example. You don't need a streaming audio server if you have a collection of audio files and you want to make them available over a network; a web server such as Apache does that job very well. Another typical use of a streaming audio server is to handle a playlist. If you have static audio files, but you want to play them in a program like a radio station would, a streaming audio server is what you need.

0.4 Installation and Configuration

0.4.1 Summary

- Make sure you have Python1.5 or newer.
- Download the latest Oyez from Oyez Archive <http://ubertechnique.com/seth/oyez/archive>.
- Change to the directory where you would like to put Oyez's base directory.
- Unpack the Oyez tarball.
- Make a global config file to suit your tastes based on the file `conf/global.conf.example`, and one config file for each stream you would like to run based on `conf/stream.conf.example`.
- Run `oyez.py`.

0.4.2 Dependencies

Oyez is written in Python. It is tested against Python1.5.2 and Python 2.2.1, so any version after 1.5.2 should work. Earlier version of Python may work, but I don't intend to try to maintain support for versions earlier than 1.5.2. I test Oyez on Linux, but because it's written in Python I expect it to behave well on any UNIX platform. Oyez should run on Windows sometime soon, although I will only maintain minimal support for that platform. I will gladly accept patches to improve Oyez's behavior on Windows. I'm a Windowsphobe not a Windows-user-phobe.

0.4.3 Download

The Oyez home page is located at Oyez Home <http://ubertechnique.com/seth/oyez>. It has a notice about and a link to the current version of Oyez. Old versions are also available at Oyez Archive <http://ubertechnique.com/seth/oyez/archive>.

0.4.4 Prepare the Base Directory

Decide where you want Oyez's files to stay.

On UNIX Oyez does a chroot into its base directory, unless you tell it not to in the global config file. This means that Oyez doesn't have access to any part of your file system except the part that is contained in its base directory. So, if Oyez is installed in `/usr/local/oyez` and you give it a playlist that includes `/var/ogg/squiggly_squirrel.ogg`, Oyez will be unable to access your favorite Squiggly tune. If you put `squiggly_squirrel.ogg` in `/usr/local/oyez/ogg/` instead then Oyez will find it if the playlist specifies either `ogg/squiggly_squirrel.ogg` or `/ogg/squiggly_squirrel.ogg`.

On UNIX Oyez also changes user and group to the user and group specified in the global config file. If the global config file doesn't specify a user and group, then user *nobody* and group *nobody* are used. Make sure that all files Oyez will need to read after starting are readable by the user specified in the configuration file.

With these considerations in mind, do something like the following:

```
zcat Oyez-0.1.4.tgz | tar x -C /usr/local
mkdir /usr/local/oyez/ogg
mv $HOME/ogg/* /usr/local/oyez/ogg
chown -R oyez.oyez /usr/local/oyez/ogg
```

0.4.5 Editing the Configuration Files

Since version 0.1.6 Oyez reads multiple configuration files starting with a global configuration file which Oyez looks for at `/usr/local/oyez/conf/global.conf` by default. The file `conf/global.conf.example` contains an example global configuration file. The global configuration file provides an option to specify streamwise configuration files. Stream configuration files will be read from the same directory as the global configuration file, see the global configuration option *stream* described below. The syntax of the configuration file is fairly simple. White space at the beginning or end of a line is ignored. If the first non-whitespace character is a hash mark, the line is ignored as a comment. Empty lines are fine. Configuration options have short descriptive names containing no whitespace. Everything right of the configuration option name is interpreted as the value to which to set that configuration option. The following configuration options control Oyez's behavior:

address (stream)

Specify a fully qualified domain name (FQDN) or IP address, or the star character (*). Specifying an FQDN here has the same effect as specifying the IP address that FQDN points to. If you write a star character instead, Oyez will bind to all available IP addresses. If you specify a specific IP address Oyez will bind only to that IP address.

port (stream)

This option tells Oyez which port to bind to. The default is port 9090. This means that clients will connect to a URL like `Port Example http://myserver.somedomain:`

9090 to connect to your stream. Pick a number between 1 and 65535. Be sure you are the root user if you are starting Oyez after configuring it to listen on a port below 1024.

stream (global)

This option may be specified multiple times in the global configuration file. The argument of the stream option will be a stream name. For each stream option found in the global configuration file Oyez will read the file <stream name>.conf from the same directory from whence Oyez read the global configuration file.

send_timeout (global)

Defaulting to 30 seconds, this configuration option takes an integer number of seconds as an argument. Clients that have not been sent any data for this amount of time will be disconnected.

logfile (global or stream)

This option allows you to specify the location of the Oyez log file. The logging is poor, but you get to say where it goes anyway. The default log configuration is a unified file for all streams named "oyez.log" in the Oyez base directory.

home (global)

This option specifies Oyez's base directory. Oyez will change directories to this location right after reading its configuration file. On UNIX, Oyez's default behavior is to chroot into its home directory. See the section entitled Prepare the base directory for more discussion of the chroot call.

chroot (global)

When set to 1 Oyez will chroot into its base directory(specified with the configuration option *home*). When set to 0 Oyez will have access to the full file system at run-time. The default value is 1.

user (global)

This should be a valid user name on your system. After starting, Oyez will run with the permissions of this user.

group (global)

This should be a valid group name on your system. After starting, Oyez will run with the permission of this group.

read_size (global or stream)

Defaults to 1024. This value in bytes is used rather than an arbitrary value for read system calls in the Oyez source. Might be interesting for performance tuning, but I doubt it.

bitrate (stream)

This configuration item is never necessary for Vorbis streams, but is required for MP3 playlist based streams. With ogg files, which are usually variable bitrate Oyez has to determine how fast to stream based on the contents of the file (by using the *granule_pos* value from the ogg header). With MP3 on the other hand, Oyez is just going to have to believe what you tell it. This means that when you make an MP3 playlist, you'd better be sure that all the files are the same bitrate; specify that bitrate here. You can mix files of radically different bitrates in a Vorbis playlist.

source (stream)

Oyez uses this configuration option to decide whether to look for a playlist from which to stream or whether to look for a configuration option specifying an external program to use as a data source. The values that make sense to Oyez are *playlist*, and *external*. Remember that all other values will be ignored causing Oyez to use its default value, "*playlist*."

playlist (stream)

If the *source* configuration option is set to the value *playlist*. This configuration option will be used to determine where the Oyez playlist is located in the file system. If Oyez is configured to chroot, this path will be relative to the base directory specified by the *home* configuration option.

external_command_line (stream)

If Oyez is configured to accept data using an external encoder, this will be the command line Oyez runs to start that encoder. The command line should send an Ogg/Vorbis or MP3 stream to standard out. Oyez will depend on the external command to provide data at the appropriate rate. This is reasonable for a live source since most encoders still can't encode audio from the future. Just don't expect good results from a command like:

```
cat bulwinkles_buddy.ogg
```

I hope to add a timing feature for external sources soon. After all the aliens with whom I communicate via */dev/urandom* *do* have technology allowing them to record audio from the future.

file_type (stream)

If Oyez is running a playlist based stream this option is used to determine whether the playlist refers to mp3 or ogg files. Note that file extension are completely ignored, so you can name your audio files whatever you want and Oyez should behave well. Valid values for this option are *mp3* and *ogg*.

external_data_type (stream)

If Oyez is configured to stream data from an external encoder Oyez will use this variable to determine what kind of audio data that external encoder is producing. Valid values or this option are *mp3* and *ogg*.

random (stream)

If Oyez is streaming from a playlist and this configuration option is set to 1 Oyez will randomize its playlist on startup, and every time it gets to the end of the playlist. If this variable is set to 0 audio files are played in the order specified in the playlist. The default value of this option is 1.

max_clients (stream)

Use this configuration option to set the maximum number of clients that can connect to a particular stream. The default value is 50.

0.4.6 Run Oyez

The Oyez executable is `oyez.py`. Oyez takes a single command line option `-d`, allowing you to specify a non-default location to look for the configuration directory. For example if you'd like to use `/etc/oyez/global.conf` as your global configuration file, you can start oyez like this:

```
./oyez.py -d /etc/oyez
```

Oyez will fork, detach itself from the tty and run in the background. You will need to be root if Oyez is configured to chroot or if you wish to use ports below 1024. Oyez writes logging info to the file `oyez.log` in its configuration directory.

0.5 Creating Links to an Oyez Stream

Players capable of listening to network audio streams each behave a little differently. There are three ways to provide access to an HTTP audio stream such as Oyez's, `.m3u` playlists, `.pls` playlists, and direct links to the stream. I am told that `.m3u` playlists are preferred.

0.5.1 Direct Links to an Oyez Stream

You can create a direct link to an Oyez stream in HTML. For example if your running Oyez on a server with address `examplehost.exampledomain` and Oyez has bound to the its default port, 9090:

```
<a href="http://examplehost.exampledomain:9090">
```

Some players look for a file extension in the stream URL to decide what kind of media they are trying to decode. If the client is using Xmms to connect to a Vorbis stream this is definitely the case (at least with Xmms 1.2.5). It is advisable to put a phony file name in the stream URL; in any case it doesn't hurt:

```
< href="http://examplehost.exampledomain:9090/squirrel_call.ogg">
```

A browser following this link will connect to `examplehost.exampledomain` on port 9090 and request the file `squirrel_call.ogg`. Oyez completely ignores the content of HTTP requests, and responds to any connection by simply spewing data. Note that this is a security advantage in addition to making the server design simpler. The disadvantage is that you can't serve multiple streams from the same port like you can in Icecast, but come on, there are 65535 ports to use; don't try to tell me you've run out.

0.5.2 Playlists and MIME Types

Configuring your web server to serve .m3u and .pls playlist files

When a client application requests a file from a web server, the server first must determine the file's MIME type, so that it can send this information to the client in an http header. Servers generally guess the file's MIME type based on its file extension. So a file named "rocky_owns.html," will be assumed to be an HTML file, and the server will send the MIME Type, `text/html`. Most web servers use `text/plain` as a default MIME Type when they can't guess the file type based on the file extension. You will quite likely have to configure your web server to recognize `.m3u` and `.pls` files and send the appropriate MIME Type for each. If you click on a link to a playlist file on a typical modern browser and see the contents of the playlist displayed in the browser window, then the web server probably sent the default MIME Type `text/plain` and is not configured for your playlist type. Server configuration naturally depends on what server you're using. The MIME Type for `.m3u` files is `audio/x-mpegurl` while that for `.pls` files is `audio/x-scpls`. I think "scpls" is meant to stand for "Shoutcast playlist," but if that's not the case it's still a good mnemonic.

Configuring your browser to handle .m3u and .pls playlist files

When you click on a link to a playlist file in you browser, the web server will send the file along with a MIME type for that file. For your browser to know how to handle the playlist it needs to be configured to handle that MIME type. When, under Windows or MacOS, you install a Media Player, the installation process will often set MIME types for you, I would comment on the MIME types wars that various players have been

waging, but I don't want to have to change my diapers; I don't use those OS's anyway. In any case, most browsers will allow you to configure MIME Types manually.

For example in Mozilla (Netscape's good son), you follow the cascading menus to Edit->Preferences. In the Preferences Dialog follow the hierarchy browser to Navigator->Helper Applications. The mime type for .m3u files is *audio/x-mpegurl* while that for .pls files is *audio/x-scpls*. If the MIME type you're interested in is already listed but improperly configured, select it and click the Edit button. If the MIME type isn't listed, click the New Type button. A dialog box will appear. Fill it in somewhat as follows for .m3u files:

```
Description of Type: m3u audio playlist
File Extensions: .m3u
MIME Type: audio/x-mpegurl
Application to Use: /usr/bin/xmms
```

For a .pls file it would look like this:

```
Description of Type: Shoutcast Playlist
File Extensions: .pls
MIME Type: audio/x-scpls
Application to Use: /usr/bin/xmms
```

0.5.3 .m3u playlists

An .m3u playlist providing access to an http audio stream need only contain a single line, the URL of the stream itself. Having multiple continual streams in a playlist may seem a little strange, but you can do it, to specify a static intro file for example. Here's example of a .m3u playlist file:

```
http://exampleserver.exampledomain/welcome.mp3
http://exampleserver.exampledomain:9090/flying_rodent.ogg
http://exampleserver.exampledomain/buzz_off.ogg
```

See the section entitled "Direct Links to an Oyez Stream," for further discussion of URLs linking directly to an Oyez stream.

0.5.4 .pls Playlists

.pls files are used in the same way as .m3u files, they just have a more complex format. Here is enough of an example to be able to figure out how to make your own .pls files, compare it with the .m3u example above:

```
[playlist]
numberofentries=3
File1=http://exampleserver.exampledomain/welcome.ogg
Title1=Frozen Arctic Ground Squirrels Sing the Blues (Intro)
Length1=-1
```

File2=http://exampleserver.exampledomain:9091/ground_squirrel.mp3
Title2=Frozen Arctic Ground Squirrels Sing the Blues
Length2=-1
File3=http://exampleserver.exampledomain/our_bad.ogg
Length3=-1

I guess Length<number>=-1 means unspecified/continuous. I've never streamed static files like this.