GNU xhippo

A GTK-based front-end and playlist manager for sound players
version 3.2

Adam Sampson

1 Introduction

GNU `xhippo` is a generic playlist manager program for a variety of UNIX sound players. It's been shown to work with `mpg123`, `madplay`, `bplay`, `s3mod`, `tracker`, `xmp`, `mtvp`, `splay`, `ogg123` and `timidity`, and should work with more-or-less anything that can take a filename on the command line. It is capable of automatically deciding which player to use depending on a file's extension; the defaults are set in a config file. It uses textual playlist files, which are easily generated with the `find` or `locate` utilities. The interface of `xhippo` is very loosely modelled on the shareware 'HippoPlayer' player for the Amiga.

`xhippo` was developed under GNU/Linux, but it contains nothing Linux-specific, so it will probably work on any UNIX-like system where gcc, glib and GTK are available. (Additionally, it supports GNU long options where '`getopt_long`' is available.) If you're using NetBSD or FreeBSD, a port is available—see See Chapter 2 [Installation], page 2. `xhippo` will optionally integrate with the GNOME desktop, supporting GNOME themes and drag-and-drop.

`xhippo` comes with NO WARRANTY, to the extent permitted by law. You may redistribute copies of GNU `xhippo` under the terms of the GNU General Public License. For more information about these matters, see the file named '`COPYING`'.

If you've installed a previous version of `xhippo`, read the '`ChangeLog`' for information on what's changed recently.

If you are using `fvwm2`, you may like to read '`README.fvwm2`'. `xhippo` uses GNU gettext for internationalisation; you can pick the language you want by setting your '`LANG`' environment variable. If your C library's gettext support doesn't work, you can give the '`-with-included-gettext`' option to the '`configure`' script to make it use the copy of gettext included in the package.

You may wish to read '`contrib/README`'; it contains information about the programs submitted by other `xhippo` users that can be found in the contrib directory.

If you've downloaded `xhippo`, please send me some mail to tell me what you think of it. Suggestions for improvements will be gratefully received.

2 Installation

Before reading this section, if you're using NetBSD, FreeBSD or OpenBSD, there's an easier way of doing this, as an `xhippo` port is standard. Simply

```
cd /usr/ports/audio/xhippo
make install
```

and `xhippo` will automatically be downloaded and built for you. (However, this may not use the latest version, so you may want to consider installing it from the source yourself, in which case you should follow the instructions below.)

`xhippo` uses `GTK+`, and requires `GTK+ 1.0` or higher; it needs `gtk-config` in your path in order to build. It uses GNU `automake` and `autoconf`, so it will automatically detect some features of your system that can affect `xhippo`'s performance. If you have `libid3` installed (available from `http://id3lib.sourceforge.net/`), `xhippo` will use it to read ID3 tags (if you don't have it installed, `xhippo` will use its own simple implementation which only understands ID3v1).

To compile, change to the source directory and do

```
./configure
make
```

If you want GNOME support, do '`./configure -with-gnome`' instead of '`./configure`'. If you would also like the GNOME menubar, do '`./configure -with-gnome -with-gnome-menubar`'. If you encounter problems finding GTK while building, do '`./configure -help`' to find out how to specify where GTK files are stored. If you want to install into a different place, do '`./configure -prefix=/usr/local/xhippo`' or wherever. Several other options are available; try '`./configure -help`' for more information.

To install the program, do

```
make install
```

The archive includes xhippo.xpm; this is a small transparent xpm image that's suitable for use as an icon in your window manager. It's taken from the `HippoPlayer` NewIcon on the Amiga. I'll try to find a better xpm; I don't like this one very much. It's not installed anywhere, so you should probably copy it in to '`/usr/local/share/pixmaps/`' or wherever your window manager looks for icons.

Each user who wants to use `xhippo` should create a '`.xhippo`' directory in their home directory. `xhippo` will look for the '`config`' and '`gtkrc`' files there, and will save its window state into the '`winstate`' file there if configured to. Playlists should be kept in a '`.xhippo/playlists/`' directory.

`xhippo` finds your home directory by looking for the '`HOME`' environment variable. If this is not set by default, you should add a line of the form

```
export HOME=`pwd`
```

or your shell's equivalent to your profile script.

3 Invocation

To use `xhippo`, you need to give it at least one playlist. You can either load a playlist by specifying it on the command line or in the config file, or you can build a playlist by dropping files from a file manager into the `xhippo` window or using the "Add Song" option on the popup menu.

Playlists are files containing names of files to play, one per line. This is compatible with X11Amp/XMMS's playlist format, so if you have an X11Amp playlist called '`Nice`', you could do '`xhippo $HOME/.x11amp/Nice`' to use it. (GQmpeg can also import `xhippo` playlists.) Alternately, you can generate them with the `find` command; for instance, if you keep your .mp3 files in your '`$HOME/sound`' directory, you could do

```
find $HOME/sound -name *.mp3 | sort >$HOME/.xhippo/playlists/mp3
xhippo $HOME/.xhippo/playlists/mp3
```

to make a playlist and play it. (With a little trickery, `xhippo` can be persuaded to automatically build playlists from a directory on startup; see the example config file for more information.)

Playlists can also include other playlists by name; to do this, put a line of the form

```
!foo
```

in the playlist. `xhippo` will then try to load the file '`foo`' as a playlist, inserting its entries into the list at that point. If the '`-i`' command-line option or '`readid3`' config-file option are enabled, `xhippo` will try to find ID3 tags in the listed files and will put them in the list rather than the filenames if found.

To start playing automatically once a playlist is loaded, use the '`-a`' option anywhere on the command line (or the '`autostart`' command in the config file).

You can specify multiple playlists on the command line. Alternatively, you can specify the '`-f`' option to make `xhippo` treat command-line arguments as files to be added to the playlist rather than playlists to load (for instance, '`xhippo -f *.ogg`'), or '`-D`' to make `xhippo` treat command-line arguments as directories to be searched for playable files.

'`xhippo -h`' or '`xhippo -help`' will give you some simple help instructions.

4 GUI

(If you built with GNOME, you will see a menu bar at the top of the window; this replicates the buttons described below.)

The status line shows a little information about the player; it shows the number of playlist entries upon startup, and what player is being used to play the current song (and the PID of the player process, if you use '`-p`' or '`showpid:1`' in the config file). To start a song, click on it in the list, or click "Next" to pick either a random song (see the '`mustplayall`' config file option below to find out how to control this), or the next song in the playlist, depending on the setting of the "Random" checkbox. Clicking on "Prev" will play the previous song (if the "Random" checkbox is enabled, the last random song picked). To restart the current song from the beginning, click "Restart". To stop, click "Stop".

`xhippo` supports a number of keyboard accelerators: *r* for Restart, *s* or keypad */* for Stop, *p* or keypad *+* for Pause, *n*, keypad *** or keypad *-* for Next, *b* for Prev, *`* for Mini, *h* for Random, *a* for Add File, *d* for Add Directory, *l* for Load Playlist, *v* for Save Playlist, *o* for Sort By Name, *w* for Sort By Swapped Name, *t* for Sort By Mtime, *c* for Clear Playlist, *0* to *9* for user-defined menu entries and *q* for quit.

When the end of a song is reached, `xhippo` will pick either a random one or the next one from the list depending on whether the "Random" checkbox is set or not. Optionally, `xhippo` can scroll the list so that the randomly-picked song is at the top of the visible section; to enable this, use the '`-s`' command-line option, or the '`scroll:1`' config-file option below. To quit, use your window manager's close button, pick Quit from the popup menu or send `xhippo` a 'SIGINT' *Ctrl-C*.

If you check the "Mini" checkbox, the list of files will disappear, making the window smaller; unchecking it will make it reappear. You can make `xhippo` start up in this "minified" state by using the '`-t`' switch or the '`startmini:1`' option in your config file.

You can drop 'file:' URLs (such as files from `gmc` or `xftree`) onto the `xhippo` window to add songs to the playlist (if you drop a directory, it will search the directory for files to add). Other URLs (such as 'http:') are not supported, as there's no simple mechanism that all players understand to stream a file from a network connection.

Right-clicking on the playlist or the status bar will bring up a popup menu, which allows you to bring up an information window for a song showing the song's size, location and the date it was last modified, move songs up and down within the playlist, remove songs from the playlist, add songs or directories to the playlist, sort the playlist by song name, swapped song name (the part after the first '`-`' in the name) or song mtime, or load and save playlists. The default directory for loading and saving playlists is '`$HOME/.xhippo/playlists`'. Left-clicking on the status bar will bring up the information window for the song that is currently playing.

If you use the '`-w`' switch or the '`savewinstate`' config file option, `xhippo` will save its window position and size to your '`$HOME/.xhippo/winstate`' file when you close the window, and will reload it on startup.

5 Config file

`xhippo` searches for its config file as '`/usr/local/etc/xhippo.config`' (or wherever you specified with the '`-prefix`' option to `configure`), '`$HOME/.xhippo/config`' and '`xhippo.config`' (in the current directory); all that are present will be read.

Most config-file options have a command-line equivalent; these support both traditional ('`-x`') and GNU-style long ('`--extended`') options. The long options have the same name as the config-file options; '`-option`' is equivalent to '`option:1`' in the config file (i.e. it forces the option on). The command-line options override the config file. Invoke `xhippo` as '`xhippo -help`' for more information.

A config file line starting with a '`#`' will be ignored.

Lines have the format '`command:arg1:arg2...`'. Arguments can be of several types: booleans, integers, strings and sort types. Booleans represent on/off conditions; '`yes`', '`on`', '`true`' or any non-zero integer will enable the attribute, and any other value will disable it. For sort types, '`none`' (or any other unrecognised value) means don't sort, '`name`' (or, for backwards compatibility, any non-zero numeric value) means sort by name, '`swapped`' means sort by swapped name, '`mtime`' means sort by mtime.

The following configuration commands are accepted:

`type:extension:command[:options]`
: When `xhippo` encounters a file whose name ends in *.extension*, it will use '`command` *file*' to play it. The extension is case-insensitive. '`options`' is optional and controls how the player will be started by `xhippo`; if it contains '`g`' then the player will be started in its own process group (necessary to properly kill some multithreaded players); if it contains '`i`' then the player will be started with stdin connected to `/dev/null`. Examples: '`type:mp3:mpg123 -b 1024`', '`type:ogg:ogg123:i`'

`autostart:boolean`
: If enabled, `xhippo` will play a random song on startup. This is equivalent to '`-a`' on the command line.

`scroll:boolean`
: If enabled, `xhippo` will scroll the list when a random item is selected so that the selected song is visible. This is equivalent to '`-s`' on the command line.

`mustplayall:boolean`
: If enabled, `xhippo` will always pick an item that it hasn't played before from the playlist when asked to pick a random entry. This continues until it has played all the entries, at which point it will just pick a random one as before. This is equivalent to '`-m`' on the command line.

`readid3:boolean`
: If enabled, `xhippo` will try to read ID3 tags from the files listed in the playlists and will use them as the playlist entries if found. This slows down `xhippo` startup considerably, so it's disabled by default. This is equivalent to '`-i`' on the command line.

`exec:command`
: *command* will be executed as a shell command (using '`/bin/sh`') before any further config commands are parsed. For an example of why I included this, look at the example config file ('`xhippo.config`').

`load:playlist`
: *playlist* will be loaded as a playlist file. This is equivalent to including *playlist* on the command line.

`savewinstate:boolean`
: If enabled, `xhippo` will save its window position, size and state (whether it is minified or not) between sessions in the '`$HOME/.xhippo/winstate`' file. If it is zero, `xhippo` will allow your window manager to place it, will start with a "standard" (small) size, and won't start minified (unless the next option is specified). This is equivalent to '`-w`' on the command line.

`startmini:boolean`
: If enabled, `xhippo` will start up in the "minified" state, as if you'd clicked the "Mini" button (even if the winstate file says that the window wasn't tinified). This is equivalent to '`-t`' (for "tiny") on the command line.

`showpid:boolean`
: If enabled, `xhippo` will show the PID of its player process in the status line when not in mini mode. This is equivalent to '`-p`' on the command line.

`ordered:boolean`
: If enabled, `xhippo` will start with the "Random" checkbox turned off. This is equivalent to '`-o`' on the command line.

`stripextension:boolean`
: If enabled, `xhippo` will strip the extensions from the filenames displayed in the playlist. This is equivalent to '`-S`' on the command line.

`leftscroll:boolean`
: If enabled, `xhippo` will place the vertical scrollbar on the left side of the playlist. This looks better with NeXTStep-like themes. This is equivalent to '`-l`' on the command line.

`hideplayeroutput:boolean`
: If enabled, `xhippo` will redirect the output (stdout and stderr) of child player processes to '`/dev/null`'. This is equivalent to '`-q`' on the command line.

`sortonload:sorttype`
: *sorttype* specifies how `xhippo` should sort playlists when they are loaded. '`-O`' on the command line is equivalent to '`sortonload:name`'.

`playlistdir:dir`
: Use *dir* as the default directory for loading or saving playlists.

`demanglenames:boolean`
: If enabled, `xhippo` will replace '`_`' (underscores) and '`%20`'s in song names with spaces on the display. This is equivalent to '`-d`' on the command line.

`onetime:boolean`
: If enabled and either '`ordered`' or '`mustplayall`' are turned on, `xhippo` will stop when all the items in the playlist have been played. This is equivalent to '`-1`' on the command line.

`playlisttitle:boolean`
: If enabled, then `xhippo` will set the window title to include the name of the current playlist. This is equivalent to '`-T`' on the command line.

`titlebasename:boolean`
: If enabled, then `xhippo` will use the basename of the playlist name when setting the window title if '`playlisttitle`' is set. This is equivalent to '`-b`' on the command line.

`nocheckfiles:boolean`
: Normally, when a playlist is loaded, `xhippo` will check to see whether all the listed files exist and discard them if they don't. If enabled, then `xhippo` won't bother checking, which will make startup significantly faster on large playlists. This is equivalent to '`-c`' on the command line. Note that `xhippo` will read the information if it's needed at a later time, so if you sort the playlist by mtime then it'll need to scan all the files to get the mtimes.

`writeplaying:boolean`
: If enabled, then `xhippo` will write the name of the song that is currently playing to '`$HOME/.xhippo/current_song`'. (If the file cannot be written, `xhippo` will silently ignore it.) This is equivalent to '`-W`' on the command line.

`skippath:integer`
: Normally when displaying song names in the playlist, `xhippo` will use the basename of the file (i.e. it will strip off the path to the file). If *integer* is set to something other than zero, `xhippo` will only strip the first *integer* elements of the path; this could be useful if you sort your music collection into albums and want to display the album names in the playlist. This is equivalent to '`-k` *integer*' on the command line.

`usercommand:description:command`
: Define a user command. This will add an entry titled *description* to the context menu (and assign it a numerical accelerator starting from '0'); when the entry is picked, *command* will be run (with a single instance of '`%s`' in the command replaced by the full filename of the selected song, or the empty string if the menu is invoked while not over a song).

`commandlinesongs:boolean`
: If enabled, `xhippo` will treat command-line arguments as songs to add to the playlist rather than playlists to load. This is equivalent to '`-f`' on the command line; you can therefore do something like '`xhippo -f *.mp3`' to start `xhippo` listing all the '.mp3' files in the current directory.

`commandlinedirs:boolean`
: If enabled, `xhippo` will treat command-line arguments as directories to search for songs to add to the playlist. This is equivalent to '`-D`' on the command line.

`deletewhenplayed:boolean`
: If enabled, `xhippo` will remove songs from the playlist once they have been played. This is equivalent to '`-x`' on the command line.

`persistplaylist:boolean`
: If enabled, `xhippo` will attempt to load a playlist from '`$HOME/.xhippo/saved_playlist`' on startup (if no other files are given on the command line), and will save the current playlist to that file on exit. This is equivalent to '`-P`' on the command line.

`commandlineguess:boolean`
: If enabled, `xhippo` will attempt to guess what the command-line arguments are. If they have a known extension (one specified with '`type`' above) then they are assumed to be songs; if they are directories they are assumed to be directories; otherwise they are assumed to be playlists. You probably want this turned on unless you're in the habit of calling your playlist '`playlist.ogg`'. This is equivalent to '`-g`' on the command line.

`persistfrequently:boolean`
: If enabled and '`persistplaylist`' is also enabled, `xhippo` will save the current playlist whenever a new song is started. You may want this if you're in the habit of killing xhippo randomly.

6 gtkrc

To allow for customised GTK appearances, `xhippo` reads a standard gtkrc file in '`$HOME/.xhippo/gtkrc`'. An example gtkrc is included as '`xhippo.gtkrc`'. For more information about gtkrc files, consult the GTK documentation. If you're using GNOME, you can leave this file empty, because `xhippo` will inherit the GNOME themes like any other GNOME application.

7 Distribution

If you want an archive to give to somebody else, invoke 'make dist' in the xhippo source directory. This will produce the same 'xhippo-VERSION.tar.gz' file that I distribute. If you wish to mail me a modified version, do exactly the same (after removing the 'doc' directory); I can then diff it against my last release to see what you've changed.

8 Contact

`xhippo` is far from perfect. Please contact `bug-xhippo@gnu.org` if you discover any bugs, or have any suggestions.

`xhippo` was written by me, Adam Sampson, `azz@gnu.org`. My other software can always be found at `http://zenchaos.netpedia.net`; `xhippo` is now a GNU (`http://www.gnu.org`) application and is distributed from `ftp.gnu.org` or from mirror sites.

The original German translation was done by Volker Assmann, `volka@bigfoot.de`, who was also responsible for betatesting.

Hubert Feyrer first alerted me to the problems with GTK+-1.0 and 1.1 compatibility, and also maintains the NetBSD package at

`ftp://ftp.netbsd.org/pub/NetBSD/packages/pkgsrc/audio/xhippo/README.html`

(or '`/usr/ports/audio/xhippo`' on your NetBSD system).

Rod Taylor maintains the FreeBSD port (in '`/usr/ports/audio/xhippo`' on FreeBSD 3.2 and up).

Kevin Lo maintains the OpenBSD port at

`http://www.openbsd.org/cgi-bin/cvsweb/ports/audio/xhippo/`

Craig Knudsen provided a routine to read ID3 tags.

Joseph Turian suggested the idea of file inclusion in playlists.

Jeff Covey supplied a Perl script which provided the functionality of the current `"Load"` button, which encouraged me build the feature in.

Kevin Everets implemented the Pause button, the leftscroll option, translated the documentation to texinfo and provided patches for or suggested various other features.

Several other people who contributed are credited in the '`ChangeLog`' file and the '`contrib/README`' file.

Table of Contents

1 Introduction 1

2 Installation 2

3 Invocation 3

4 GUI 4

5 Config file 5

6 gtkrc 9

7 Distribution 10

8 Contact 11