

EasyPlayer

COLLABORATORS

	<i>TITLE :</i> EasyPlayer		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY		February 14, 2023	

REVISION HISTORY

<i>NUMBER</i>	<i>DATE</i>	<i>DESCRIPTION</i>	<i>NAME</i>

Contents

1	EasyPlayer	1
1.1	main	1
1.2	introduction	2
1.3	requirements	2
1.4	install	3
1.5	playlist	3
1.6	tapedeck	4
1.7	preferences	4
1.8	interfaces	5
1.9	examples	8
1.10	predefined	9
1.11	menus	10
1.12	keys	10
1.13	shell	11
1.14	workbench	12
1.15	author	12
1.16	thanks	12
1.17	history	13

Chapter 1

EasyPlayer

1.1 main

EasyPlayer (EYP) 1.01

© Copyright 1997-1999 by Ralph Debusmann

A user-friendly front-end for music players without one.

. Introduction	.
. Requirements	.
. Installation	.
. Playlist	.
. Tapedeck	.
. Preferences	.
. Interfaces	.
. Examples	.
. Predefined	.
. Pulldown menus	.
. Keyboard shortcuts	.
. Shell support	.
. Workbench support	.
. Author	.
. Thanks	.
. History	.

1.2 introduction

-- Introduction

EasyPlayer (EYP for short) is a multi-format music player.

"Why bother" I hear you say, we Amigans already have plenty of great multi-format music players like "DeliTracker", "EaglePlayer", "HippoPlayer", "APlayer" or even "SongPlayer" and "AmigaAMP". Why yet another one?

Because EYP is different. While all the music players mentioned above use more ("HippoPlayer") or less ("DeliTracker") integrated replay routines, EYP has not one replay routine built-in but is designed to cooperate efficiently with shell-based music players like "mpega" for MP3 or "SIDPLAY" for SID-tunes.

The approach EYP takes has both advantages and disadvantages. For one, integrated music players like "HippoPlayer" tend to be quicker. Also, EYP misses some rather essential features like volume control or a pause facility. But on the other hand, there are a couple of advantages that come to its rescue:

more freedom

You can configure EYP to front-end whatever (shell-based) player program you like for any music module format there is. You are free to use PowerPC-player programs for CPU-intensive module format for instance.

more formats

Does any of the above players support MIDI? Or SIDPLAY? With EYP you simply define "GMPlay" as your MIDI-player and "SIDPLAY" for SID-tunes and off you go.

more future

You can incorporate new music formats and corresponding players anytime to keep up with new trends. You do not have to wait for to wait for anybody to write a "DeliPlayer" or else - just plug in an existing (shell-based) player program by yourself.

more consistency

EYP provides a single consistent, easy-to-use, mouse or keyboard-driven front-end for every music player without a GUI. No more lame, ad-hoc, non font-sensitive BlitzBasic-GUIs from Aminet mus/* required.

1.3 requirements

-- Requirements

- o an Amiga running AmigaOS 2.04 (V37) or higher

- o reqtools.library V38 or higher
- o some (shell-based) music player programs like "mpega" or "GMPlay"

1.4 install

-- Installation

The installation of EasyPlayer proceeds in two steps:

1. Copy the "EYP"-directory and its contents to some place on your HD.
2. Configure

 interfaces
 for the music players that you intend to
use EYP as a front-end for. In practice, this amounts to just changing
the "Path"-field for each
 predefined interface
 (= the path to the
module player for the interface). The default paths are prefixed with
"PLEASE_CHANGE_PATH:".

1.5 playlist

-- Playlist

The playlist is a list of music modules. It can be manipulated in a number of ways:

argument passing

An initial playlist can be passed over to EasyPlayer by making use of either

 shell-
 or
 Workbench
-argument passing.

playlist buttons

You can manipulate the playlist by using the playlist buttons directly below the list. These are the icons' meanings (from left to right):

add module(s)
remove module
move module to top
move module up
move module down
move module to bottom
shuffle (randomize) modules

Advanced users might wish to have a look at the corresponding
 keyboard shortcuts

open, append, save

A playlist can be opened, appended or saved by using items from
 the "Project"-
 menu

1.6 tapedeck

-- Tapedeck

The playlist-entries can be made to play, stop etc. by the tapedeck
 buttons (below the playlist buttons). Their icons' meanings from left to
 right are:

play previous module
 add and play module(s)
 set subsong number
 play next module
 stop module (= send "CTRL_C" to player program)
 eject module(s) (stop module and clear playlist)

To play a module from the playlist just double click on its name.

Note that by "previous module" and "next module" not the module previous
 or next to the module highlighted is meant but the one previous or next
 to the one playing.

1.7 preferences

-- Preferences

Here you can customize EasyPlayer (a bit):

Module path
 Default module path

Playlist path
 Default playlist path

The preferences can be opened or saved by using the "Preferences"-
 menu
 items. The default preferences filename is "EYP.prefs". Upon ↔
 startup, the
 preferences are searched for in the EYP-program directory ("PROGDIR:"),
 i.e. the directory from which EYP has been launched.

1.8 interfaces

-- Interfaces

"Interfaces" define the way EasyPlayer acts upon encountering a music module of a certain format.

The interfaces window ("EYP ints") serves to administrate the interfaces. The buttons in the row directly below the list of interfaces have the following meaning:

- New: add a new interface
- Edit: edit an interface (same as double click on list item)
- Copy: duplicate an interface
- *: (in-)activate interface (star declares interface inactive)
- Del: delete interface
- Clear: delete all interfaces

The buttons to the right of the list of interfaces do this:

- First: move interface to top
- Up: move interface up
- Down: move interface down
- Last: move interface to bottom
- a-z: sort interfaces in ascending order
- z-a: sort interfaces in descending order

Note: The order of the list of interfaces can be crucial: In search of an interface for a music module the list is traversed from top to bottom. If interfaces A and B both could recognize a module, and A comes before B in the list of interfaces, A will be used (and not B).

A list of interfaces can be opened, appended or saved by using the "Interfaces"-

- menu
- items. The default interfaces filename is "EYP.ints" in "PROGDIR:". This file EYP attempts to read upon startup.

Any number of interfaces can be edited (defined) in parallel.

-- Interface definition

Each interface is made up of a definition of the module format that it recognizes ("Recognition") and a corresponding definition of the player to invoke ("Invocation"). I will describe both in the following. Please note: For an interface to work, just the following fields must be filled in: "Path", and either "Header" ("Offset") or "Pattern".

-- Invocation

Name

Name of the interface. A star "*" at the beginning of an interface name declares it inactive.

Path

Full path of the music player program. The program should be shell-based and stop after being sent a "CTRL_C"-signal.

Misc args

Miscellaneous shell-arguments

Song arg

Player program subsong shell-argument

Help arg

Help argument and console definition. Hints:

Use the "WAIT"-option to keep the console from closing right after it has been opened.

"Control" + "Backslash" can be used to close console windows.

Enclose the console definition in quotes if its window title contains spaces.

A minimal but sensible definition for "Help arg" goes like this:
>CON://///WAIT.

A ">" must be placed in front of the "Help arg"-console definition.

Console

Console definition when the player program is launched to play a module. There are four differences to note compared to "Help arg"-console definition:

The "WAIT"-option must not be used.

The console definition must not be enclosed in quotes.

Minimal definition: CON: for a console, NIL: for no output.

Do not place a ">" in front of the "Console" definition.

Put quotes around modname

A few simpler player programs do not support quoted filenames. Uncheck this gadget if this is the case.

Force filter off

If checked EYP will turn off the filter a couple of times in a second while playing a module.

This is the order of the composition of the command to invoke the interface module player program:

```
<Path> <Misc Args> <Module name> <Song arg>
```

The composed command that is sent to the module player can be made visible through the debug mode.

-- Recognition

Header

A string representing the music module header of the format

Offset
 Offset of the header (in bytes from begin of module). If no header is given, the offset isn't taken into account.

Pattern
 AmigaDOS-pattern to recognize modules of this format. Might come in useful if you want to recognize XPK- or PowerPacker-crunched modules.

Subsong support
 Check if the format supports subsongs

Def song offset
 Offset of default subsong ordinal in module (in bytes from begin of module)

Len
 Length of default subsong ordinal (1-4 bytes, 0 means that there is no default subsong ordinal to be found in the module)

Add to value
 Add a number to the default song value. Some module formats count their subsongs from 0 to N-1 instead of from 1 to N. Add 1 in these cases. If modules of the format to be defined do not have a default song, also add 1. (e.g. (Octa)MED, see also examples).

Num songs offset
 Offset of the number of subsongs ordinal (in bytes from begin of module)

Len
 Length of number of subsongs ordinal (1-4 bytes, 0 means that there is no number of subsongs ordinal to be found in the module)

Add to value
 Add a number to the value read for the number of songs in the module. (Octa)MED-modules for example state N-1 as the number of songs although there are N subsongs. Add 1 in this case (see also examples).

Use the debug mode for debugging interfaces with subsong-support.

Note that an EYP-interface can recognize a module both by looking at the module's filename and the module header. Module recognition proceeds as follows: The interface first checks if the module filename agrees with the "Pattern" and, if it doesn't, whether the file header agrees with the "Header". If none of the two does, EasyPlayer goes on to the next interface in the list of interfaces and checks whether this one can recognize the module. A module cannot be recognized if no (activated) interface was able to recognize it.

Note further that you do not have to "Use" an interface definition before you can try it on modules from the playlist - any changes are propagated through and are thus active at once. Still "Cancel" can be used to undo the changes made. The same principle applies to the list of interfaces and the preferences window.

1.9 examples

-- Examples

This section is about some example interface definitions that shall serve as a kind of tutorial. I will define three interfaces and comment on some definitions.

-- mpega

Invocation

Name: mpega

Path: Work:Music/mpega

You will probably need to change this (-;

Misc args: -n -i

Song arg: (none)

MP-files never have subsongs

Help arg: >"CON:0/0/500/500/EYP mpega help/CLOSE/WAIT"

This defines a console window at the top left corner of the screen with width 500 and height 400 pixels, and to be equipped with a close-gadget ("CLOSE"-option).

Console: CON:0/0/400/200/EYP mpega console/CLOSE

Put quotes around modname: Checked

Force filter off: Unchecked

Recognition

Header: (none)

Offset: 0

Pattern: (#?.mp2|#?.mp3)

A list of filename pattern is given here to recognize modules to be played with "mpega".

Subsong support: Unchecked

-- SIDPLAY

Invocation

Name: SIDPLAY

Path: Work:Music/SIDPLAY

Misc args: (none)

Song arg: -o

SIDPLAY (and the PSID-module format) have subsong-support.

Help arg: >"CON:0/0/500/300/EYP SIDPLAY help/CLOSE/WAIT"

Console: CON:0/0/500/200/EYP SIDPLAY console/CLOSE

Put quotes around modname: Checked

Force filter off: Unchecked

Recognition

Header: PSID

Offset: 0

Pattern: (none)

SIDPLAY-modules can be solely recognized by their file header (if uncrunched)

Subsongs: Checked

Def song offset: 16

```

    The default song ordinal can be found at offset 16...
Len: 2
    ... and is 2 bytes long.
Add to value: 0
Num songs offset: 14
    The number of songs within a SIDPLAY-module can be
    found at offset 14 ...
Len: 2
    ... and is 2 bytes long, too.
Add to value: 0

```

```
-- OctaMEDPlayer
```

```
Invocation
```

```

Name: OctaMEDPlayer
Path: Work:Music/OctaMEDPlayer
Misc args: (none)
Song arg: (none)
    OctaMEDPlayer has no "Song arg" but just expects the song number
    to come after the filename.
Help arg: ? >"CON:0/0/500/200/EYP OctaMEDPlayer help/CLOSE/WAIT"
Console: KCON:0/0/400/100/EYP OctaMEDPlayer console/CLOSE
Put quotes around modname: Checked
Force filter off: Unchecked

```

```
Recognition
```

```

Header: MMD
Offset: 0
Pattern: (none)
Subsongs: Checked
Def song offset: 0
Len: 0
    No default song offset ordinal
Add to value: 1
Num songs offset: 50
    The number of songs within a OctaMED-module can be found at offset
    50 ...
Len: 2
    ... and is 2 bytes long.
Add to value: 1

```

1.10 predefined

```
-- Predefined Interfaces
```

The current version of EasyPlayer comes with support for six different module formats, namely AHX, MPEG-Audio, MIDI, (Octa-)MED, ProTracker and SIDPLAY. Corresponding Interfaces are included within the EasyPlayer distribution. To make EasyPlayer play the modules, you will need shell-based player programs corresponding to the different module formats. Recommended are:

```

for AHX: PlayTHX by Dexter/Abyss, Kyzer/CSG
Aminet: mus/play/PlayTHX.lha

```

```

for MPEG-Audio: mpega by Stephane Tavenard

```

Aminet: mus/play/mpega.lha

for MIDI: GMPlay by Alessandro de Luca and Christian Buchner
Aminet: mus/midi/GMPlay13.lha

for (Octa-)MED: OctaMEDPlayer by Teijo Kinnunen
Aminet: mus/play/OctaMEDPlayer7.lha

for ProTracker: ModPlay (from ModUtils) by Ben Hutchings
Aminet: mus/misc/ModUtils1_27.lha

for SIDPLAY: SIDPLAY by Phill Wooller
Aminet: mus/play/sidplay.lha
or alternatively (less CPU-intensive)
Clips by Dave Dustin
Aminet: mus/play/Clips.lha

If your favorite music format is not yet included in the list above, do not despair but plug in an appropriate player by yourself. It is easy.

1.11 menus

-- Pulldown menus

```
"Project"  
  "Open ...": open a playlist  
  "Append ...": append a playlist to existing one  
  "Save As ...": save a playlist  
  "Snapshot": snapshot main window position and size  
  "Debug ...": toggle debug mode  
  "About ...": about EYP  
  "Quit": quit EYP  
  
"Preferences"  
  "Change ...": change preferences  
  "Open ...": open preferences file  
  "Save As ...": save preferences file  
  
"Interfaces"  
  "Change ...": change interfaces  
  "Open ...": open interfaces file  
  "Append ...": append interfaces file to existing one  
  "Save As ...": save interfaces file
```

1.12 keys

-- Keyboard shortcuts

Most keyboard shortcuts are made explicit by underscores (as with text-buttons), so that this section only dwells on the remaining, implicit keyboard shortcuts.

Keyboard shortcuts for playlist buttons:

```
add module(s): "a" or "+"
remove module: del or "-"
move module to top: "f"
move module up: "p"
move module down: "d"
move module to bottom: "l"
shuffle (randomize) modules "s", "r"
```

Keyboard shortcuts for tapedeck buttons:

```
play previous module: Shift + Cursor left
add and play module(s): Shift + Return
set subsong number: Cursor left (previous subsong) and
                    Cursor right (next subsong)
play next module: Shift + Cursor right
stop module: Space
eject module(s): Tab
```

Press "t" to toggle the Amiga's built-in low-pass filter
(playlist/tapedeck window)

Pressing "Return" without "Shift" is equivalent to double clicking a
module (i.e. start playing it).

"Cursor up" and "Cursor down" can be used to traverse lists (i.e. the
playlist and the list of interfaces). "Alt" + "Cursor up" brings you to
the top of a list, "Alt" + "Cursor down" to the bottom.

The "Esc"-key can be used to close the currently active window and hence
behaves like a click on the close-gadget. If a (Save)|Use|Cancel window is
active (preferences, list of interfaces, interface definition windows),
both pressing "Esc" and clicking the close-gadget amount to clicking the
"Use"-button.

Filerequester-buttons accompanying textfields (e.g. preferences window)
are "numbered" - the upmost one within a window can be opened by pressing
"1", the one below by "2" and so on.

1.13 shell

-- Shell support

This is EasyPlayer's commandline template:

```
MOD=MODULE/M,PLST=PLAYLIST
```

```
MOD=MODULE
```

Pass a list of modules to EYP upon shell-startup. An example:

```
EYP MOD="mod.4 thorus" MOD=Lightforce.mid MOD=IntoPersona
```

The MOD (= MODULE)-keyword may also be dropped:

```
EYP "mod.4 thorus" Lightforce.mid IntoPersona
```

PLST=PLAYLIST

Pass a playlist to EYP upon shell-startup. For example:

```
EYP PLST=C64Music
```

The PLST(= PLAYLIST)-keyword may not be dropped.

It is possible to pass over to EYP both a list of modules and a playlist. In this case, the playlist will be appended to the list of modules.

1.14 workbench

-- Workbench support

So far EasyPlayer supports multiple Workbench-arguments (i.e. modules) and can be used as the default tool for music modules.

1.15 author

-- Author

Send a postcard to:

```
Ralph Debusmann  
Paul-Marien-Strasse 6  
D-66111 Saarbruecken  
Germany
```

Send suggestions etc. to:

```
rade@coli.uni-sb.de
```

Get the latest version at:

```
http://www.coli.uni-sb.de/~rade/e.html
```

1.16 thanks

-- Thanks

EasyPlayer has been developed in AmigaE on an A4000/060, AmigaOS 3.0. I'd like to thank:

- o Wouter van Oortmerssen for AmigaE and the EasyGUI-interface builder
 - o Jason R. Hulance for enhancing EasyGUI
 - o Gregor Goldbach for the EBuild-"make"-clone
 - o Fabio Rotondo for his Amiga Foundation Classes and his
-

"NodeMaster", "StringNode" and "Reqtooler" classes in particular

- o Victor Ducedre for his "buttonbase" and "dclistview" plugins
- o Ali Graham for his "multitext" plugin
- o CygnusSoft, Schatztruhe and Olaf Barthel for CygnusEd
- o Cloanto for PersonalPaint
- o Morten Eriksen for PicCon

and

- o Sylvain Rougier and Pierre Carrette for BrowserII.

Without the work of these people EasyPlayer would have never made it.

1.17 history

-- History

Version 1.01 (26.12.1999)

- bugfix in easyrecognize.e (pattern matching now looks at the filename rather than entire path of the module)

Version 1.0 (25.12.1999)

- revised included interfaces a bit (added "PLEASE_CHANGE_PATH:" as a prefix to all paths)
- revised this guide a little bit more

Version 0.91 (28.9.1999)

- filter can be toggled by pressing "t"
- "Interface" definition part renamed to "Invocation", "Format" to "Recognition"
- removed path dependencies from included interface file "EYP.ints"
- revised this guide a bit

Version 0.9 (25.9.1999)

- (attempted) first Aminet release
-