Package: openmpt (via r-universe)

November 5, 2024

Title Open ModPlug Tracker port

Version 0.0.1.0004

Description Reads, renders and plays music tracker files using the libopenmpt library.

License GPL (>= 3)

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

Imports av

LinkingTo cpp11

SystemRequirements openmpt: libopenmpt-devel (rpm) or libopenmpt-dev (deb), portaudio-devel (rpm) or portaudio19-dev (deb).

URL https://pepijn-devries.github.io/openmpt/, https://github.com/pepijn-devries/openmpt

BugReports https://github.com/pepijn-devries/openmpt/issues

Config/pak/sysreqs libavfilter-dev libopenmpt-dev portaudio19-dev

Repository https://pepijn-devries.r-universe.dev

RemoteUrl https://github.com/pepijn-devries/openmpt

RemoteRef master

RemoteSha 4f5177fec116f5527d5801e9d9ca5afe02ab7537

Contents

`channel_mute_status` *Control the volume of a module*

Description

Functions to control the global volume of a module, or that of specific channels in the module.

Usage

```
channel_mute_status(mod, channel, ...)

channel_mute_status(mod, channel, ...) <- value

channel_volume(mod, channel, ...)

channel_volume(mod, channel, ...) <- value

global_volume(mod, ...)

global_volume(mod, ...) <- value
```

Arguments

`mod`	A tracker module object of class `openmpt`.
`channel`	Channel index (`integer` starting at `0`) for which to request or control the volume.
`...`	Ignored
`value`	Replacement value. In case of 'status' functions a `logical` value, in case of 'volume' functions a `numeric` value.

Value

Returns the volume (status), or the updated object in case of an assign operation (`<-`).

Examples

```
mod <- demo_mod()

channel_mute_status(mod, 0L)

## Mute the first channel in the module
channel_mute_status(mod, 0L) <- TRUE

## Second channel volume at 50%
channel_volume(mod, 1L) <- 0.5

## global volume at 90%
global_volume(mod, 1L) <- 0.9
```

`convert_mod` *Convert a ModPlug Tracker module to an audio file*

Description

Renders ModPlug Tracker music to an audio file and encodes it to a desired output format (e.g. .mp3, .ogg, etc) using `av::av_audio_convert()`.

Usage

```
convert_mod(
  mod,
  file,
  start_order = 0L,
  start_row = 0L,
  sample_rate = 44100L,
  verbose = FALSE,
  ...
)
```

Arguments

`mod`	A tracker module object of class `openmpt`
`file`	Output audio file where the rendered audio is stored. The file name extension is used to determine the type of encoding to be applied.
`start_order`	Starting position (`integer` index starting at 0) in the pattern sequence table.
`start_row`	Starting row (`integer` index starting at 0) of the pattern table.
`sample_rate`	Output sample rate in Hz (samples per seconds).
`verbose`	Passed on to `av::av_audio_convert()`.
`...`	Ignored

Value

Returns NULL invisibly

Examples

```
mod <- demo_mod()

destination <- tempfile(fileext = ".mp3")

convert_mod(mod, destination)
```

---

demo_mod *Loads demo module included with the package*

Description

Reads the file cyberrid.mod. It is a ProTracker file create by Jester. It is redistributed under the Attribution Non-commercial Share Alike license. The music was part of an Amiga computer demo named 'Extension' and was originally released in 1993.

Usage

```
demo_mod()
```

Value

Returns the demo module tracker object of class openmpt

Examples

```
mod <- demo_mod()
```

---

get_duration_seconds *Get ModPlug Tracker module duration*

Description

Get the duration of the song from a openmpt class module object in seconds.

Usage

```
get_duration_seconds(mod, ...)
```

Arguments

mod	A tracker module object of class openmpt.
...	Ignored

Value

Returns a numeric value indicating the song duration in seconds.

Examples

```
mod <- demo_mod()
get_duration_seconds(mod)
```

---

get_metadata *Get ModPlug Tracker module meta data*

Description

Get meta data of a tracker module such as its "type", "title" and "tracker". Use get_metadata_keys() to get the available keys for a module object.

Usage

```
get_metadata(mod, key = "title", ...)

get_metadata_keys(mod, ...)
```

Arguments

mod	A tracker module object of class openmpt.
key	A key as listed by get_metadata_keys().
...	Ignored

Value

A list of available keys in case of get_metadata_keys(), the requested information in case of get_metadata().

Examples

```
mod <- demo_mod()
get_metadata_keys(mod)

get_metadata(mod, "tracker")
```

openmpt_info *Retrieve information about the OpenMPT library*

Description

A wrapper for the `get` function in libopenmpt (see API documentation)

Usage

```
openmpt_info(key = "library_version", ...)
```

Arguments

`key`	A key `character` string indicating which information to retrieve. can be `"library_version"`, `"library_features"` and many others. See API documentation) for all possible values.
`...`	Ignored

Value

Returns a `character` string with the requested information.

Examples

```
openmpt_info("library_version")
openmpt_info("library_features")
openmpt_info("url")
```

pattern *Get a specific ModPlug pattern table*

Description

Collects a specific pattern table from a tracker module and presents it as a matrix of formatted `character` strings.

Usage

```
pattern(mod, pattern = 0L, width = 0L, pad = TRUE, ...)
```

Arguments

`mod`	A tracker module object of class `openmpt`.
`pattern`	The pattern index (starting at `0`) of the pattern to get.
`width`	The maximum number of characters the string should contain. `0` means no limit.
`pad`	If `TRUE`, the string will be resized to the exact length provided in the width parameter.
`...`	Ignored

Value

A `matrix` of pattern cells formatted as `character` strings. Each column represents an audio channel.

Examples

```
mod <- demo_mod()
pattern(mod)
```

`pitch_factor` *Control the pitch and tempo of a module*

Description

Functions to control the pitch and tempo of a module.

Usage

```
pitch_factor(mod, ...)

pitch_factor(mod, ...) <- value

tempo_factor(mod, ...)

tempo_factor(mod, ...) <- value
```

Arguments

`mod`	A tracker module object of class `openmpt`.
`...`	Ignored
`value`	Replacement value. A `numeric` factor with which to adjust the tempo or pitch of a module

Value

Returns current factor, or the updated object in case of an assign operation (`<-`).

Examples

```
mod <- demo_mod()

## Increase the module pitch with a factor 2
pitch_factor(mod) <- 2

## Increase the module tempo with a factor 2
tempo_factor(mod) <- 2
```

play *Play a ModPlug Tracker module*

Description

Renders a module tracker object of class openmpt and plays it instantaneously.

Usage

```
play(mod, sample_rate = 44100L, progress = "vu", ...)
```

Arguments

mod	A tracker module object of class openmpt.
sample_rate	Output sample rate when playing the module.
progress	Progress printed to console while playing. Should be one of "vu" (indicative volume meter), "time" (shows timer) or "none" (don't show progress). If your audio is stuttering you might want to set this to "none" to save processing speed.
...	Ignored

Value

Returns NULL invisibly.

Examples

```
## Not run:
mod <- demo_mod()
play(mod)

## End(Not run)
```

position_seconds *Get and set ModPlug Tracker module position*

Description

Get or set the position of the music player. rewind() moves the position to the start of the song.

Usage

```
position_seconds(mod, ...)

position_seconds(mod, ...) <- value

rewind(mod, ...)

set_position_order_row(mod, order, row, ...)
```

Arguments

mod	A tracker module object of class openmpt.
...	Ignored
value	Position in seconds to move the player to. The value is rounded to its nearest order and row position.
order	Index of the position in the pattern sequence table (starts at 0).
row	Index of the row in the current pattern table (starts at 0).

Value

Returns NULL invisibly, or the updated object in case of the assign operator (<-).

Examples

```
mod <- demo_mod()
position_seconds(mod)
position_seconds(mod) <- 10.2
set_position_order_row(mod, 1, 4)
rewind(mod)
```

---

print.openmpt *Implementation of basic S3 generics*

Description

Implementation of basic S3 generics such as print().

Usage

```
## S3 method for class 'openmpt'
print(x, ...)

## S3 method for class 'openmpt'
format(x, ...)
```

Arguments

x	Object to apply the method to.
...	Ignored.

Value

In case of print and format a formatted string with basic information about the module is returned.

read_mod *Read Open ModPlug module*

Description

Read any of the music tracker module file formats supported by libopenmpt: https://wiki.openmpt.org/Manual:_Module_formats.

Usage

```
read_mod(file, ...)
```

Arguments

file	File path or URL to read the file from. Binary connections are also supported.
...	Ignored

Value

A modplug class object. It is an external pointer, pointing to the module object in memory. It can be used for rendering audio.

Examples

```
## You can read from files
mod1 <- read_mod(system.file("cyberrid", "cyberrid.mod", package = "openmpt"))


## but also URLs
mod2 <- read_mod("https://api.modarchive.org/downloads.php?moduleid=41529#elektric_funk.mod")
```

Index