Package 'subtools'

March 24, 2026

Type Package

Title Read and Manipulate Video Subtitles

Version 1.1.0

Date 2026-03-16

Maintainer Francois Keck <francois.keck@gmail.com>

Description A collection of functions to read, write and manipulate video subtitles. Supported formats include 'srt', 'subrip', 'sub', 'subviewer', 'microdvd', 'ssa', 'ass', 'substation', 'vtt', and 'webvtt'.

Language en-GB

License GPL-3

Encoding UTF-8

Imports methods, utils, jsonlite, tibble, dplyr, readr, hms, tidytext, rlang

Suggests testthat (>= 3.0.0), knitr, rmarkdown, ggplot2

VignetteBuilder knitr

Config/testthat/edition 3

Config/testthat/parallel false

Config/testthat/start-first *assert*, *utils*, *read*

RoxygenNote 7.3.3

X-schema.org-applicationCategory Text Processing

X-schema.org-keywords subtitles, video, text, srt, vtt, ssa, natural language processing

NeedsCompilation no

Author Francois Keck [aut, cre, cph] (ORCID: <https://orcid.org/0000-0002-3323-4167>), Bob Rudis [ctb] (ORCID: <https://orcid.org/0000-0001-5670-2640>), Alban Sagouis [ctb] (ORCID: <https://orcid.org/0000-0002-3827-1063>)

Repository CRAN

Date/Publication 2026-03-24 10:40:02 UTC

Contents

`as_subtitle` *Convert an R object to a* `subtitles` *object*

Description

Convert an R object to a `subtitles` object

Usage

```
as_subtitle(
  x,
  format = "auto",
  clean.tags = TRUE,
  metadata = data.frame(),
  frame.rate = NA,
  encoding = "auto",
  ...
)

## Default S3 method:
as_subtitle(
  x,
  format = "auto",
  clean.tags = TRUE,
  metadata = data.frame(),
  frame.rate = NA,
  encoding = "auto",
  ...
)
```

```
## S3 method for class 'character'
as_subtitle(
  x,
  format = "auto",
  clean.tags = TRUE,
  metadata = data.frame(),
  frame.rate = NA,
  encoding = "auto",
  ...
)
```

Arguments

x	an R object that can be coerced into a subtitles object
format	a character string specifying the format of the subtitles. Four formats can be read: "subrip", "substation", "microdvd", "subviewer" (v.2) and "webvtt". Default is "auto" which tries to detect automatically the format of the file from its extension.
clean.tags	logical. If "TRUE" (default), formatting tags are deleted from subtitles using clean_tags.
metadata	a one-row dataframe or tibble, or any object that can be coerced into a one-row tibble by link[tibble]{as_tibble}.
frame.rate	a numeric value giving the frame rate in frames per second. Only relevant for MicroDVD format. If NA (default), the function tries to extract the frame.rate from the file. If it fails, the frame rate is set at 24p (23.976).
encoding	the name of the encoding to be used. Default is "auto" and uses guess_encoding to detect encoding.
...	passed on to downstream methods.

Value

An object of class subtitles (see subtitles).

Examples

```
as_subtitle(
  c("WEBVTT",
    "X-TIMESTAMP-MAP=MPEGTS:181083,LOCAL:00:00:00.000",
    "",
    "3",
    "00:00:21.199 --> 00:00:22.333", ">> FEMALE SPEAKER:",
    "Don't stay up too late.",
    "",
    ""
  ), format = "webvtt"
)
```

bind_subtitles *Bind subtitles*

Description

Bind subtitles or/and multisubtitles objects.

Usage

```
bind_subtitles(..., collapse = TRUE, sequential = TRUE)
```

Arguments

...	subtitles or multisubtitles objects to be bound.
collapse	logical. If TRUE, subtitles are combined in a single subtitles object.
sequential	logical. If TRUE (default) timecodes are recalculated to follow concatenation.

Value

A subtitles object if collapse = TRUE (default). A multisubtitles object if collapse = FALSE.

Examples

```
f1 <- system.file("extdata", "ex_subrip.srt", package = "subtools")
s1 <- read_subtitles(f1, metadata = tibble::tibble(Season = 1, Episode = 2))
f2 <- system.file("extdata", "ex_substation.ass", package = "subtools")
s2 <- read_subtitles(f2, metadata = tibble::tibble(Season = 2))
bind_subtitles(s1, s2)
bind_subtitles(s1, s2, collapse = FALSE)
```

clean_tags *Clean subtitles*

Description

Functions to clean subtitles. clean_tags cleans formatting tags. clean_captions cleans close captions, i.e all text enclosed in parentheses or squared brackets. clean_patterns provides a more general and flexible cleaning based on regular expressions.

Usage

```
clean_tags(x, format = "srt", clean.empty = TRUE)

clean_captions(x, clean.empty = TRUE)

clean_patterns(x, pattern, clean.empty = TRUE)
```

Arguments

x	a subtitles or multisubtitles object.
format	the original format of the subtitles objects.
clean.empty	logical. Should empty remaining lines ("") deleted after cleaning.
pattern	a character string containing a regular expression to be matched and cleaned.

Value

A subtitles or multisubtitles object.

---

get_mkv_info *Get informations about subtitles embedded in MKV files*

Description

This function uses mkvmerge to extract tracks data from an MKV file. You must have mkvmerge installed on your computer.

Usage

```
get_mkv_info(file, mkvmerge.exec = "mkvmerge", print.info = TRUE)
```

Arguments

file	a character string giving the path to the MKV file.
mkvmerge.exec	a character string giving the path to the mkvmerge executable.
print.info	print basic informations about subtitle tracks. Default is TRUE.

Value

A list with complete data about the MKV is invisibly returned. If the MKV has at least 1 subtitles track and print.info is TRUE, basic informations are printed. Otherwise it returns a warning.

References

https://mkvtoolnix.download/downloads.html

get_raw_text *Get subtitles text*

Description

This function extracts the raw text content of subtitles objects as a character string.

Usage

```
get_raw_text(x, collapse = " ")
```

Arguments

x	an object of class subtitles or multisubtitles.
collapse	a character string to separate the subtitles lines.

Value

A character string.

Examples

```
f <- system.file("extdata", "ex_subrip.srt", package = "subtools")
s <- read_subtitles(f)
get_raw_text(s)
cat(get_raw_text(s, collapse = "\n"))
```

get_subtitles_info *Get basic informations for subtitle objects*

Description

Get basic informations for subtitle objects

Usage

```
get_subtitles_info(x)
```

Arguments

x	a subtitles or multisubtitles object.

Value

Called for its side effects (printing to the console). Returns NULL invisibly.

Examples

```
s <- read_subtitles(
  system.file("extdata", "ex_subrip.srt", package = "subtools")
)
get_subtitles_info(s)
```

---

move_subtitles *Move subtitles Move subtitles forward or backward in* subtitles.

Description

Move subtitles Move subtitles forward or backward in subtitles.

Usage

```
move_subtitles(x, lag)
```

Arguments

x	subtitles or multisubtitles objects to be moved.
lag	numeric. Number of seconds the subtitles should be moved, forward if positive and backward if negative.

Value

A subtitles or multisubtitles object.

Examples

```
f1 <- system.file("extdata", "ex_subrip.srt", package = "subtools")
s1 <- read_subtitles(f1, metadata = tibble::tibble(Season = 1, Episode = 2))
s2 <- move_subtitles(s1, 2.7)
```

---

print.multisubtitles *Print method for multisubtitles*

Description

Print method for multisubtitles

Usage

```
## S3 method for class 'multisubtitles'
print(x, printlen = 10L, ...)
```

Arguments

x	a multisubtitles object.
printlen	the maximum number of subtitles to print.
...	further arguments passed to or from other methods.

Value

Called for its side effects (printing to the console). Returns NULL invisibly.

Examples

```
f <- system.file("extdata", "ex_subrip.srt", package = "subtools")
s <- read_subtitles(f)
bind_subtitles(s, s, collapse = FALSE)
```

---

read_subtitles *Read subtitles*

Description

Reads subtitles from a file.

Usage

```
read_subtitles(
  file,
  format = "auto",
  clean.tags = TRUE,
  metadata = data.frame(),
  frame.rate = NA,
  encoding = "auto"
)
```

Arguments

file	the name of the file which the subtitles are to be read from. If it does not contain an absolute path, the file name is relative to the current working directory.
format	a character string specifying the format of the subtitles. Four formats can be read: "subrip", "substation", "microdvd", "subviewer" (v.2) and "webvtt". Default is "auto" which tries to detect automatically the format of the file from its extension.
clean.tags	logical. If "TRUE" (default), formatting tags are deleted from subtitles using clean_tags.
metadata	a one-row dataframe or tibble, or any object that can be coerced into a one-row tibble by link[tibble]{as_tibble}.

frame.rate	a numeric value giving the frame rate in frames per second. Only relevant for MicroDVD format. If NA (default), the function tries to extract the frame.rate from the file. If it fails, the frame rate is set at 24p (23.976).
encoding	the name of the encoding to be used. Default is "auto" and uses guess_encoding to detect encoding.

Details

The support of WebVTT is basic and experimental.

Value

An object of class subtitles (see subtitles).

Examples

```
# read a SubRip file
f <- system.file("extdata", "ex_subrip.srt", package = "subtools")
f <- system.file("extdata", "ex_webvtt.vtt", package = "subtools")
read_subtitles(f)
```

---

read_subtitles_mkv *Extract subtitles embedded in MKV files*

Description

This function uses mkvextract to extract subtitles from an MKV file. You must have mkvmerge and mkvextract installed on your computer.

Usage

```
read_subtitles_mkv(
  file,
  id = 2,
  mkvextract.exec = "mkvextract",
  mkvmerge.exec = "mkvmerge"
)
```

Arguments

file	a character string giving the path to the MKV file.
id	An integer giving the ID of the tracks to be extracted. Can be a vector of length > 1 to extract several tracks. If NA (default), the default subtitle tracks will be extracted
mkvextract.exec	a character string giving the path to the mkvextract executable.
mkvmerge.exec	character string giving the path to the mkvmerge executable.

Details

The function `get_mkv_info` is a simple way to identify the ID of subtitles tracks from a MKV file.

Not all the subtitle formats supported by `mkvextract` can be read by `subtools`. See `read_subtitles` for the list of formats currently supported.

Value

An object of class `subtitles` (see `subtitles`). If several tracks are requested (via `id`), an object of class `multisubtitles`; i.e. a list of `subtitles` objects.

References

https://mkvtoolnix.download/downloads.html

---

read_subtitles_season *Read series subtitles*

Description

These functions read one or several subtitles files organized in directories. They are designed to import subtitles of series with multiple episodes.

Usage

```
read_subtitles_season(
  dir,
  format = "auto",
  bind = TRUE,
  detect.meta = TRUE,
  quietly = FALSE,
  ...
)

read_subtitles_serie(
  dir,
  format = "auto",
  bind = TRUE,
  detect.meta = TRUE,
  quietly = FALSE,
  ...
)

read_subtitles_multiseries(
  dir,
  format = "auto",
  bind = TRUE,
  detect.meta = TRUE,
```

```
  quietly = FALSE,
  ...
)
```

Arguments

dir	the name of the directory which the subtitles are to be read from (see Details).
format	a character string specifying the format of the subtitles (default is "auto", see read_subtitles for details).
bind	a logical. If TRUE (default), subtitles are bound with bind_subtitles
detect.meta	a logical. If TRUE (default), the function tries to automatically detect metadata (Serie, Season and Episode) from file names. This will override user metadata with the same name.
quietly	a logical. If FALSE (default), a message indicating the number of imported files is printed.
...	further arguments to be passed to read_subtitles.

Details

These functions read subtitles files at different levels from a 3-levels directory (see the tree below). The function read_subtitles_multiseries reads everything recursively from "Series_Collection". The function read_subtitles_serie reads everything recursively from a serie folder (e.g. "Serie_A"). The function read_subtitles_season reads everything from a season folder (e.g. "Season_1"). To read a specific episode file (e.g. "Episode_1.srt), use read_subtitles.

```
Series_Collection
|-- Serie_A
|   |-- Season_1
|   |   |-- Episode_1.srt
|-- Serie_B
|   |-- Season_1
|   |   |-- Episode_1.srt
|   |-- Season_2
|   |   |-- Episode_1.srt
|   |   |-- Episode_2.srt
```

Value

If bind is set on TRUE a subtitles object, otherwise an object of class multisubtitles; i.e. a list of subtitles objects.

subtitles *Create a* subtitles *object*

Description

A subtitles is a special form of tibble.

Usage

```
subtitles(text, timecode.in, timecode.out, id, metadata = data.frame())
```

Arguments

text	a character vector of subtitles text content.
timecode.in	a character vector giving the time that the subtitles appear on the screen. The format must be "HH:MM:SS.mS".
timecode.out	a character vector giving the time that the subtitles disappear. The format must be "HH:MM:SS.mS".
id	a vector of IDs for subtitles. If not provided it is generated automatically from timecode.in order.
metadata	a one-row dataframe or tibble, or any object that can be coerced into a one-row tibble by link[tibble]{as_tibble}.

Value

a subtitles object i.e. a tibble with at least 4 columns containing IDs, timecodes and text of the subtitles and optionally metadata in extra columns.

unnest_tokens.subtitles *Split a column into tokens*

Description

This function extends unnest_tokens to subtitles objects. The main difference with the data.frame method is the possibility to perform timecode remapping according to the split of the input column.

This wrapper turns the function tidytext::unnest_tokens into an S3 generic. The default method (unnest_tokens.default) delegates to the original implementation. See "?unnest_tokens.subtitles" for the subtools specific documentation.

It simply calls the original ‘tidytext::unnest_tokens‘.

Usage

```
## S3 method for class 'subtitles'
unnest_tokens(
  tbl,
  output,
  input,
  token = "words",
  format = c("text", "man", "latex", "html", "xml"),
  to_lower = TRUE,
  drop = TRUE,
  collapse = NULL,
  ...,
  time.remapping = TRUE
)

unnest_tokens(
  tbl,
  output,
  input,
  token = "words",
  format = c("text", "man", "latex", "html", "xml"),
  to_lower = TRUE,
  drop = TRUE,
  collapse = NULL,
  ...
)

## Default S3 method:
unnest_tokens(
  tbl,
  output,
  input,
  token = "words",
  format = c("text", "man", "latex", "html", "xml"),
  to_lower = TRUE,
  drop = TRUE,
  collapse = NULL,
  ...
)
```

Arguments

tbl	A data frame
output	Output column to be created as string or symbol.
input	Input column that gets split as string or symbol. The output/input arguments are passed by expression and support quasiquotation; you can unquote strings and symbols.

token	Unit for tokenizing, or a custom tokenizing function. Built-in options are "words" (default), "characters", "character_shingles", "ngrams", "skip_ngrams", "sentences", "lines", "paragraphs", "regex", and "ptb" (Penn Treebank). If a function, should take a character vector and return a list of character vectors of the same length.
format	Either "text", "man", "latex", "html", or "xml". When the format is "text", this function uses the tokenizers package. If not "text", this uses the hunspell tokenizer, and can tokenize only by "word".
to_lower	Whether to convert tokens to lowercase.
drop	Whether original input column should get dropped. Ignored if the original input and new output column have the same name.
collapse	A character vector of variables to collapse text across, or NULL. For tokens like n-grams or sentences, text can be collapsed across rows within variables specified by collapse before tokenization. At tidytext 0.2.7, the default behavior for collapse = NULL changed to be more consistent. The new behavior is that text is *not* collapsed for NULL. Grouping data specifies variables to collapse across in the same way as collapse but you **cannot** use both the collapse argument and grouped data. Collapsing applies mostly to token options of "ngrams", "skip_ngrams", "sentences", "lines", "paragraphs", or "regex".
...	Extra arguments passed on to tokenizers, such as strip_punct for "words", n and k for "ngrams" and "skip_ngrams", and pattern for "regex".
time.remapping	a logical. If TRUE (default), subtitle timecodes are recalculated to take into account the split of the input column.

Value

A tibble.

A tibble.

Examples

```
f <- system.file("extdata", "ex_webvtt.vtt", package = "subtools")
s <- read_subtitles(f, metadata = data.frame(test = "Test"))

#require(tidytext)
unnest_tokens(s)
unnest_tokens(s, Word, Text_content, drop = FALSE)
unnest_tokens(s, Word, Text_content, token = "lines")
```

`write_subtitles` *Write subtitles*

Description

This function writes a `subtitles` object in a file.

Usage

```
write_subtitles(x, file, format = "srt", encoding = "UTF-8")
```

Arguments

`x`	an object of class `subtitles`.
`file`	a character string naming a file for writing.
`format`	a character string giving the file format. Not used (only SubRip format is currently implemented).
`encoding`	the name of the encoding to be used.

Value

Called for its side effects (writing to `file`). Returns `NULL` invisibly.

Index