Package: ytdlpr (via r-universe)

September 5, 2024

Title R wrapper for yt-dlp, focused on extracting and processing subtitles

Version 0.0.1.9003

Description Get subtitles from YouTube and parse them.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

Imports cli, dplyr, fs, lubridate, purrr, readr, rlang, stringr, tibble, yyjsonr

Repository https://giocomai.r-universe.dev

RemoteUrl https://github.com/giocomai/ytdlpr

RemoteRef HEAD

RemoteSha 8e3708abf52a5d6feb0f68c7c70930264bd9a720

Contents

`extract_captions_from_json` *Extract available languages in captions*

Description

Used internally

Usage

```
extract_captions_from_json(json_l, x)
```

Arguments

`json_l`	A list object, resulting from a parsed .info.json file
`x`	Type of data to extract. In practice, either "subtitles" or "automatic_captions"

Value

A list, with names of available languages

Examples

```
## Not run:
extract_captions_from_json(
  yyjsonr::read_json_file("path_to_json_file"),
  "automatic_captions"
)

## End(Not run)
```

`yt_concatenate` *Concatenated trimmed video files in a single video clip*

Description

Concatenated trimmed video files in a single video clip

Usage

```
yt_concatenate(
  trimmed_df,
  sort_by_date = FALSE,
  info_df = NULL,
  destination_filename = "concatenated",
  destination_folder = "0_concatenated_video",
  destination_path = NULL,
  overwrite = FALSE,
  yt_base_folder = NULL
)
```

Arguments

`trimmed_df`	A data frame, typically generated as output of `yt_trim()` or `yt_trim_with_text()`.
`sort_by_date`	Defaults to FALSE. If TRUE, retrieves (and, if necessary, downloads) info files for all video clips, and reads them with `yt_read_info_json()` in order to get information about the published time.
`info_df`	Defaults to NULL. If given, a data frame typically generated with `yt_read_info_json()`.
`destination_folder`	Defaults to `0_trimmed_video`: trimmed video files will be stored inside a folder called `0_trimmed_video` inside your base folder as defined by `yt_base_folder` or options. If you want an absolute path, use the argument `destination_path` instead.
`destination_path`	Defaults to NULL. Location where trimmed video files will be stored. If given, takes precedence over `destination_folder`.
`overwrite`	Defaults to FALSE. If TRUE, overwrites concatenated file if it already exists.
`yt_base_folder`	Base folder, defaults to NULL. Can be set with `yt_set_base_folder()`

Value

The path of the generated video clip.

`yt_extract_id` *Extract YouTube identifier from an URL.*

Description

Extract YouTube identifier from an URL.

Usage

```
yt_extract_id(yt_id)
```

Arguments

yt_id	Url or unique identifier of a YouTube video.

Value

A character vector of YouTube identifiers.

Examples

```
yt_extract_id("https://youtu.be/WXPBOfRtXQE?feature=shared")

# if already an identifier, just returns it:
yt_extract_id("WXPBOfRtXQE")
```

---

yt_filter *Filter subtitles and link back the original source*

Description

Filter subtitles and link back the original source

Usage

```
yt_filter(
  subtitles_df,
  pattern,
  ignore_case = TRUE,
  regex = TRUE,
  playlist = NULL,
  sub_lang = NULL,
  sub_format = "vtt",
  lag = -3,
  yt_base_folder = NULL
)
```

Arguments

subtitles_df	Defaults to NULL. If given must be a data frame, typically generated with yt_get_local_subtitles() |> yt_read_vtt().
pattern	A character string.
ignore_case	Defaults to TRUE.
regex	Defaults to TRUE.
playlist	Playlist, either as full url from Youtube or as id.
sub_lang	Defaults to NULL. If not given, all local subtitles are returned. If given, only subtitles in the given sub_lang are returned.

sub_format	Defaults to "vtt". File extension of the subtitle.
lag	Defaults to -3. Refers to the number of seconds before or after the start time as recorded in the subtitles. Minus three or four seems to generally be a good fit.
yt_base_folder	Base folder, defaults to NULL. Can be set with yt_set_base_folder()

Value

A data frame, including only lines where the given pattern is found.

Examples

```
## Not run:
yt_get_subtitles_playlist(
  playlist = "https://www.youtube.com/playlist?list=PLbyvawxScNbuSi7sJaJbHNyyx3iYJeW3P"
)

subtitles_df <- yt_get_local_subtitles(
  playlist = "https://www.youtube.com/playlist?list=PLbyvawxScNbuSi7sJaJbHNyyx3iYJeW3P"
) |>
  yt_read_vtt()

yt_filter(
  pattern = "rover",
  subtitles_df = subtitles_df
)

## End(Not run)
```

---

yt_get *Retrieve video or write_subs from a playlist or by id*

Description

If write_subs or auto_subs is set to TRUE, it checks that the subtitles in the language set with sub_lang are available. In all other cases, by defaults, it skips ' downloads if *any* previous file associated with a given video identifier has ' been downloaded. Set check_previous to FALSE to always download files.

Usage

```
yt_get(
  yt_id = NULL,
  playlist = NULL,
  check_previous = TRUE,
  sub_lang = "en",
  sub_format = "vtt",
  subs = FALSE,
```

```
  auto_subs = FALSE,
  video = FALSE,
  description = FALSE,
  info_json = FALSE,
  comments = FALSE,
  thumbnail = FALSE,
  min_sleep_interval = 1,
  max_sleep_interval = 8,
  sleep_subtitles = 2,
  custom_options = "",
  yt_base_folder = NULL
)
```

Arguments

yt_id	YouTube identifier of a video or full url to a video. Normally a vector, but if a data frame is given, yt_id column is taken as input (useful for pipe-based workflows).
playlist	Playlist, either as full url from Youtube or as id.
check_previous	Defaults to TRUE. If FALSE, input is always downloaded. If TRUE, and subs is TRUE, it checks that the requested language is locally available. If subtitles is set to FALSE, and only one of the write options is enabled, it checks for the local availabilty of relevant files. For some options, the presence of *any* local file associated with a given id prevents further downloads associated with it.
sub_lang	Defaults to "en". If more than one, can be given as comma separated two letter codes, or a as vector.
sub_format	Defaults to "vtt". Other formats not yet supported.
subs	Defaults to FALSE. "Write subtitle file".
auto_subs	Defaults to FALSE "Write automatically generated subtitle file".
video	Defaults to FALSE. Download the video files.
description	Defaults to FALSE. "Write video description to a .description file"
info_json	Defaults to FALSE. "Write video metadata to a .info.json file"
comments	Defaults to FALSE. "Retrieve video comments to be placed in the infojson."
thumbnail	Defaults to FALSE. "Write thumbnail image to disk"
min_sleep_interval	"Number of seconds to sleep before each download. This is the minimum time to sleep when used along with –max-sleep-interval"
max_sleep_interval	"Maximum number of seconds to sleep. Can only be used along with –min-sleep-interval"
sleep_subtitles	"Number of seconds to sleep before each subtitle download"
custom_options	Defaults to an empty string. If given, it should correspond to parameters exactly as they would be used on command line. For a full list, see the original documentation.
yt_base_folder	Base folder, defaults to NULL. Can be set with yt_set_base_folder()

Details

Argument definition are quoted from the original yt-dlp project.

Value

A data frame, with details about locally available subtitles.

Examples

```
## Not run:
yt_get(
  playlist = "https://www.youtube.com/playlist?list=PLbyvawxScNbtMcDKmT2dRAfjmSFwOt1Vj"
)

yt_get(
  yt_id = "https://youtu.be/WXPBOfRtXQE",
  subtitles = TRUE,
  video = TRUE,
  description = TRUE,
  info_json = TRUE
)

## End(Not run)
```

---

yt_get_available_subtitles

Checks with clips have subtitles or automatic captions based on info json file

---

Description

Checks with clips have subtitles or automatic captions based on info json file

Usage

```
yt_get_available_subtitles(
  info_json_df = NULL,
  sub_lang = "en",
  automatic_captions = TRUE,
  subtitles = TRUE,
  yt_id = NULL,
  playlist = NULL
)
```

Arguments

info_json_df	Defaults to NULL. Generally created with yt_read_info_json(). If not given, either yt_id or playlist must be given. If given, both yt_id and playlist are ignored.
sub_lang	Defaults to "en", subtitles language.
automatic_captions	Defaults to TRUE. If TRUE, checks if subtitles are available as "automatic captions".
subtitles	Defaults to TRUE. If TRUE, checks if subtitles are available as (manually added or approved) "subtitles".
yt_id	YouTube video identifier. Ignored if info_json_df given.
playlist	YouTube list. Ignored if info_json_df given.

Value

A data frame with three columns, yt_id, sub_lang, and sub_type (sub_type can either be automatic_captions or subtitles).

Examples

```
## Not run:
yt_get_available_subtitles(
  playlist = "https://www.youtube.com/playlist?list=PLbyvawxScNbtMcDKmT2dRAfjmSFwOt1Vj"
)

## End(Not run)
```

---

yt_get_base_folder *Retrieves base folder where all retrieved files will be stored for the current session*

Description

Retrieves base folder where all retrieved files will be stored for the current session

Usage

```
yt_get_base_folder(path = NULL)
```

Arguments

path	Defaults to NULL. If not set, checks if a path has been previously set, and if not, defaults to current folder.

Value

The given path (or, if left to NULL, the path previously set) is returned invisibly.

Examples

```
yt_set_base_folder(path = fs::path(
  fs::path_home_r(),
  "R",
  "ytdlpr"
))
yt_get_base_folder()
```

`yt_get_local` *Lists locally available files that can be attributed to a video id*

Description

Lists locally available files that can be attributed to a video id

Usage

```
yt_get_local(
  yt_id = NULL,
  playlist = NULL,
  file_extension = NULL,
  yt_base_folder = NULL
)
```

Arguments

`yt_id`	Url or unique identifier of a YouTube video.
`playlist`	Playlist, either as full url from Youtube or as id.
`file_extension`	Defaults to NULL. Only file names with the given extension are returned.
`yt_base_folder`	Base folder, defaults to NULL. Can be set with `yt_set_base_folder()`

Value

A data frame, with two columns, `yt_id` and `path`.

Examples

```
## Not run:
yt_get_local()

yt_get_local(
  yt_id = "WXPBOfRtXQE",
  file_extension = "webm"
)

## End(Not run)
```

`yt_get_local_subtitles` *Check for the availability of local subtitles and return details in a data frame*

Description

Check for the availability of local subtitles and return details in a data frame

Usage

```
yt_get_local_subtitles(
  yt_id = NULL,
  playlist = NULL,
  sub_lang = NULL,
  sub_format = "vtt",
  yt_base_folder = NULL
)
```

Arguments

`yt_id`	Url or unique identifier of a YouTube video.
`playlist`	Playlist, either as full url from Youtube or as id.
`sub_lang`	Defaults to NULL. If not given, all local subtitles are returned. If given, only subtitles in the given sub_lang are returned.
`sub_format`	Defaults to "vtt". File extension of the subtitle.
`yt_base_folder`	Base folder, defaults to NULL. Can be set with `yt_set_base_folder()`

Value

A data frame (a tibble) with details on locally available subtitle files.

Examples

```
## Not run:
yt_get_local_subtitles()

## End(Not run)
```

yt_get_playlist_folder *Get local path to folder where files for the given playlist will be stored*

Description

Get local path to folder where files for the given playlist will be stored

Usage

```
yt_get_playlist_folder(playlist, yt_base_folder = NULL)
```

Arguments

playlist	Playlist, either as full url from Youtube or as id.
yt_base_folder	Base folder, defaults to NULL. Can be set with yt_set_base_folder()

Value

Path to playlist folder.

Examples

```
## Not run:
yt_get_playlist_folder(
  playlist = "https://www.youtube.com/playlist?list=PLbyvawxScNbtMcDKmT2dRAfjmSFwOt1Vj"
)

## End(Not run)
```

yt_get_playlist_id *Get all Youtube identifiers for a given playlist*

Description

Get all Youtube identifiers for a given playlist

Usage

```
yt_get_playlist_id(playlist, update = FALSE, yt_base_folder = NULL)
```

Arguments

playlist	The full url of a Youtube playlist.
update	Defaults to FALSE. If FALSE, data is returned immediately if previously stored. If TRUE, it checks again the playlist on Youtube to see if new content has been added.
yt_base_folder	Base folder, defaults to NULL. Can be set with yt_set_base_folder()

Value

A data frame (a tibble) with a single column named yt_id.

Examples

```
## Not run:
yt_get_playlist_id(
  playlist = "https://www.youtube.com/playlist?list=PLbyvawxScNbtMcDKmT2dRAfjmSFwOt1Vj"
)

## End(Not run)
```

---

yt_read_info_json *Read info json files, and extract key data in a data frame*

Description

Read info json files, and extract key data in a data frame

Usage

```
yt_read_info_json(path)
```

Arguments

path	Path to one or more subtitle file in the json format, such as those downloaded by yt_get(info_json =? TRUE). If a data frame is used as input, a column named "path" in that data frame will be used as source.

Value

A data frame

Examples

```
## Not run:
yt_get(
  playlist = "https://www.youtube.com/playlist?list=PLbyvawxScNbuSi7sJaJbHNyyx3iYJeW3P",
  info_json = TRUE
) |>
  yt_read_info_json()

## End(Not run)
```

`yt_read_vtt` *Read vtt subtitles*

Description

Read vtt subtitles

Usage

```
yt_read_vtt(path)
```

Arguments

`path`	Path to one or more subtitle file in the vtt format. If a data frame is used as input, a column named "path" in that data frame will be used as source.

Value

A data frame with

Examples

```
## Not run:
yt_get(
  playlist = "https://www.youtube.com/playlist?list=PLbyvawxScNbuSi7sJaJbHNyyx3iYJeW3P",
  auto_subs = TRUE
) |>
  yt_read_vtt()

## End(Not run)
```

yt_set_base_folder *Set for the current session base folder where all retrieved files will be stored*

Description

Set for the current session base folder where all retrieved files will be stored

Usage

```
yt_set_base_folder(path)
```

Arguments

path	Defaults to NULL. If not set, checks if a path has been previously set, and if not, defaults to current folder.

Value

The given path (or, if left to NULL, the path previously set) is returned invisibly.

Examples

```
yt_set_base_folder(path = fs::path(
  fs::path_home_r(),
  "R",
  "ytdlpr"
))
yt_get_base_folder()
```

yt_trim *Trim video files so that it shows only the part with relevant subtitles*

Description

Trim video files so that it shows only the part with relevant subtitles

Usage

```
yt_trim(
  subtitles_df,
  only_local = TRUE,
  lag = -3,
  duration = 5,
  check_previous = TRUE,
  video_file_extension = "webm|mp4|mkv",
```

```
  simulate = FALSE,
  destination_folder = "0_trimmed_video",
  destination_path = NULL,
  yt_base_folder = NULL
)
```

Arguments

subtitles_df	A data frame with subtitles, typically generated with a combination of yt_read_vtt() and yt_filter(). See examples.
only_local	Defaults to TRUE. If FALSE, downloads missing video files.
lag	Defaults to -3. Refers to the number of seconds before or after the start time as recorded in the subtitles. Minus three or four seems to generally be a good fit.
duration	Duration in seconds of the trimmed video. Defaults to 5.
check_previous	Defaults to TRUE. If a file with the same id, same starting time, and same duration has already been stored in the destination folder, skip it.
video_file_extension	Defaults to "webm|mp4|mkv". Can be set to explictly only rely on video files stored in a specific file format.
simulate	Defaults to FALSE. Similiarly to the same argument in yt-dlp, if set to TRUE nothing is actually done
destination_folder	Defaults to 0_trimmed_video: trimmed video files will be stored inside a folder called 0_trimmed_video inside your base folder as defined by yt_base_folder or options. If you want an absolute path, use the argument destination_path instead.
destination_path	Defaults to NULL. Location where trimmed video files will be stored. If given, takes precedence over destination_folder.
yt_base_folder	Base folder, defaults to NULL. Can be set with yt_set_base_folder()

Value

A data fram with details about the export and ffmpeg commmand. Mostly used for side effects (creates trimmed video files).

Examples

```
## Not run:
filtered_subs_df <- yt_get(
  yt_id = "-0pPBAiJaYk",
  subtitles = TRUE,
  video = TRUE
) |>
  yt_read_vtt() |>
  yt_filter(pattern = "community")

yt_trim(filtered_subs_df)
```

```
## End(Not run)
```

`yt_trim_with_text` *Trim video files, including text overlay with basic information*

Description

Arguments taken from FFMPEG's drawtext filter

Usage

```
yt_trim_with_text(
  subtitles_df,
  only_local = TRUE,
  font = "Mono",
  fontcolor = "white",
  fontsize = 32,
  box = 1,
  boxcolor = "black",
  boxopacity = 0.5,
  boxborderw = 5,
  position_x = 10,
  position_y = 10,
  yt_base_folder = NULL,
  ...
)
```

Arguments

`subtitles_df`	A data frame with subtitles, typically generated with a combination of `yt_read_vtt()` and `yt_filter()`. See examples.
`only_local`	Defaults to TRUE. If FALSE, downloads missing video files.
`font`	Defaults to "Mono".
`fontcolor`	Defaults to "white".
`fontsize`	Defaults to 32.
`box`	Defaults to 1.
`boxcolor`	Defaults to "black".
`boxopacity`	Defaults to 0.5
`boxborderw`	Defaults to 5
`position_x`	Defaults to 10
`position_y`	Defaults to 10
`yt_base_folder`	Base folder, defaults to NULL. Can be set with `yt_set_base_folder()`
`...`	Passed to `yt_trim()`

Value

Nothing, used for side effects.

Examples

```
## Not run:
yt_get(
  playlist = "https://www.youtube.com/playlist?list=PLbyvawxScNbuSi7sJaJbHNyyx3iYJeW3P",
  auto_subs = TRUE
) |> # download subtitles
  yt_read_vtt() |> # read them
  yt_filter(pattern = "rover") |> # keep only those with "rover" in the text
  dplyr::slice_sample(n = 2) |> # keep two, as this is only an example
 yt_trim_with_text(only_local = FALSE) # download video files and json files and trim video

## End(Not run)
```

Index