Package 'resmush'

December 18, 2024

Title Optimize and Compress Image Files with 'reSmush.it'

Version 0.2.1

Description Compress local and online images using the 'reSmush.it' API service <https://resmush.it/>.

License MIT + file LICENSE

URL https://dieghernan.github.io/resmush/, https://github.com/dieghernan/resmush

BugReports https://github.com/dieghernan/resmush/issues

Depends R (>= 3.6.0)

Imports cli, curl, httr2 (>= 1.0.0), tools, utils

Suggests grid, knitr, png, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/Needs/website dieghernan/gitdevr, xfun, dplyr, tibble, devtools, remotes

Config/testthat/edition 3

Config/testthat/parallel true

Encoding UTF-8

RoxygenNote 7.3.2

X-schema.org-keywords r, compress-images, optimize-images, resmushit, api, r-package, cran, cran-r

NeedsCompilation no

Author Diego Hernangómez [aut, cre, cph] (<https://orcid.org/0000-0001-8457-4658>)

Maintainer Diego Hernangómez <diego.hernangomezherrero@gmail.com>

Repository CRAN

Date/Publication 2024-12-18 10:40:02 UTC

Contents

`resmush_dir` *Optimize files of several directories*

Description

Optimize all the local files of a directory (or list of directories) using the reSmush.it API.

Usage

```
resmush_dir(
  dir,
  ext = "\\.(png|jpe?g|bmp|gif|tif)$",
  suffix = "_resmush",
  overwrite = FALSE,
  progress = TRUE,
  report = TRUE,
  recursive = FALSE,
  ...
)
```

Arguments

`dir`	Character or vector of characters representing paths of local directories.
`ext`	`regex` indicating the extensions of the files to be optimized. The default value would capture all the extensions admitted by the API.
`suffix`	Character, defaults to `"_resmush"`. By default, a new file with the suffix is created in the same directory (i.e., optimized `example.png` would be `example_resmush.png`). Values `""`, `NA` and `NULL` would be the same than `overwrite = TRUE`.
`overwrite`	Logical. Should the files in `dir` be overwritten? If `TRUE` `suffix` would be ignored.
`progress`	Logical. Display a progress bar when needed.
`report`	Logical. Display a summary report of the process in the console. See also **Value**.
`recursive`	Logical. Should the `dir` file search recursive? See also `list.files()`.
`...`	Arguments passed on to `resmush_file` `qlty` Only affects `jpg` files. Integer between 0 and 100 indicating the optimization level. For optimal results use vales above 90. `exif_preserve` Logical. Should the Exif information (if any) deleted? Default is to remove it (i.e. `exif_preserve = FALSE`).

Value

Writes on disk the optimized file if the API call is successful in the directories specified in `dir`.

In all cases, a (invisible) data frame with a summary of the process is returned as well.

See Also

reSmush.it API docs.

See `resmush_clean_dir()` to clean a directory of previous runs.

Other functions for optimizing: `resmush_file()`, `resmush_url()`

Examples

```
# Get example dir and copy
example_dir <- system.file("extimg", package = "resmush")
temp_dir <- tempdir()
file.copy(example_dir, temp_dir, recursive = TRUE)

# Dest folder

dest_folder <- file.path(tempdir(), "extimg")


# Non-recursive
resmush_dir(dest_folder)
resmush_clean_dir(dest_folder)

# Recursive
summary <- resmush_dir(dest_folder, recursive = TRUE)

# Same info in the invisible df
summary[, -c(1, 2)]

# Display with png
if (require("png", quietly = TRUE)) {
  a_png <- grepl("png$", summary$dest_img)
  my_png <- png::readPNG(summary[a_png, ]$dest_img[2])
  grid::grid.raster(my_png)
}

# Clean up example
unlink(dest_folder, force = TRUE, recursive = TRUE)
```

`resmush_file` *Optimize a local file*

Description

Optimize local images using the reSmush.it API.

Usage

```
resmush_file(
  file,
  suffix = "_resmush",
  overwrite = FALSE,
  progress = TRUE,
  report = TRUE,
  qlty = 92,
  exif_preserve = FALSE
)
```

Arguments

`file`	Path or paths to local files. **reSmush** can optimize the following image files: • `png` • `jpg/jpeg` • `gif` • `bmp` • `tiff`
`suffix`	Character, defaults to `"_resmush"`. By default, a new file with the suffix is created in the same directory than `file`. (i.e., optimized `example.png` would be `example_resmush.png`). Values `""`, `NA` and `NULL` would be the same than `overwrite = TRUE`.
`overwrite`	Logical. Should the file in `file` be overwritten? If `TRUE` `suffix` would be ignored.
`progress`	Logical. Display a progress bar when needed.
`report`	Logical. Display a summary report of the process in the console. See also **Value**.
`qlty`	Only affects `jpg` files. Integer between 0 and 100 indicating the optimization level. For optimal results use vales above 90.
`exif_preserve`	Logical. Should the Exif information (if any) deleted? Default is to remove it (i.e. `exif_preserve = FALSE`).

Value

Writes on disk the optimized file if the API call is successful in the same directory than `file`.

With the option `report = TRUE` a summary report is displayed in the console. In all cases, a (invisible) data frame with a summary of the process used for generate the report is returned.

See Also

reSmush.it API docs.

See `resmush_clean_dir()` to clean a directory of previous runs.

Other functions for optimizing: `resmush_dir()`, `resmush_url()`

Examples

```
png_file <- system.file("extimg/example.png", package = "resmush")

# For the example, copy to a temporary file
tmp_png <- tempfile(fileext = ".png")

file.copy(png_file, tmp_png, overwrite = TRUE)

resmush_file(tmp_png)

# Several paths
jpg_file <- system.file("extimg/example.jpg", package = "resmush")
tmp_jpg <- tempfile(fileext = ".jpg")

file.copy(jpg_file, tmp_jpg, overwrite = TRUE)

# Output summary in console
summary <- resmush_file(c(tmp_png, tmp_jpg))

# Similar info in an (invisible) data frame as a result
summary



# Display with png
if (require("png", quietly = TRUE)) {
  my_png <- png::readPNG(summary$dest_img[1])
  grid::grid.raster(my_png)
}

# With parameters
resmush_file(tmp_jpg)
resmush_file(tmp_jpg, qlty = 10)
```

`resmush_url` *Optimize an online file*

Description

Optimize and download an online image using the reSmush.it API.

Usage

```
resmush_url(
  url,
  outfile = file.path(tempdir(), basename(url)),
  overwrite = FALSE,
  progress = TRUE,
  report = TRUE,
  qlty = 92,
  exif_preserve = FALSE
)
```

Arguments

`url`	url or a vector of urls pointing to hosted image files. **reSmush** can optimize the following image files: • `png` • `jpg/jpeg` • `gif` • `bmp` • `tiff`
`outfile`	Path or paths where the optimized files would be store in your disk. By default, temporary files (see `tempfile()`) with the same `basename()` than the file provided in `url` would be created. It should be of the same length than `url` parameter.
`overwrite`	Logical. Should `outfile` be overwritten (if already exists)? If `FALSE` and `outfile` exists it would create a copy with a numerical suffix (i.e. `<outfile>.png`, `<outfile>_01.png`, etc.).
`progress`	Logical. Display a progress bar when needed.
`report`	Logical. Display a summary report of the process in the console. See also **Value**.
`qlty`	Only affects `jpg` files. Integer between 0 and 100 indicating the optimization level. For optimal results use vales above 90.
`exif_preserve`	Logical. Should the Exif information (if any) deleted? Default is to remove it (i.e. `exif_preserve = FALSE`).

Value

Writes on disk the optimized file if the API call is successful. In all cases, a (invisible) data frame with a summary of the process is returned as well.

If any value of the vector `outfile` is duplicated, `resmush_url()` would rename the output with a suffix `_01`. `_02`, etc.

See Also

reSmush.it API docs.

Other functions for optimizing: `resmush_dir()`, `resmush_file()`

Examples

```
# Base url
base_url <- "https://raw.githubusercontent.com/dieghernan/resmush/main/inst/"

png_url <- paste0(base_url, "/extimg/example.png")
resmush_url(png_url)

# Several urls
jpg_url <- paste0(base_url, "/extimg/example.jpg")


summary <- resmush_url(c(png_url, jpg_url))

# Returns an (invisible) data frame with a summary of the process
summary

# Display with png
if (require("png", quietly = TRUE)) {
  my_png <- png::readPNG(summary$dest_img[1])
  grid::grid.raster(my_png)
}

# Use with jpg and parameters
resmush_url(jpg_url)
resmush_url(jpg_url, qlty = 10)
```

Index