Package 'GitStats'

November 12, 2024

Title Standardized Git Repository Data

Version 2.1.2

Description Obtain standardized data from multiple 'Git' services, including 'GitHub' and 'GitLab'. Designed to be 'Git' service-agnostic, this package assists teams with activities spread across various 'Git' platforms by providing a unified way to access repository data.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

URL https://r-world-devs.github.io/GitStats/, https://github.com/r-world-devs/GitStats

BugReports https://github.com/r-world-devs/GitStats/issues

Imports cli, dplyr, glue, httr2, lubridate (>= 1.8.0), magrittr, rlang (>= 1.1.0), R6, purrr (>= 1.0.0), stringr

Suggests spelling, knitr, mockery, rmarkdown, testthat (>= 3.0.0), withr

Config/testthat/edition 3

VignetteBuilder knitr

Language en-US

NeedsCompilation no

Author Maciej Banas [aut, cre], Kamil Koziej [aut], Karolina Marcinkowska [aut], Matt Secrest [aut]

Maintainer Maciej Banas <banasmaciek@gmail.com>

Repository CRAN

Date/Publication 2024-11-12 16:20:02 UTC

Contents

`create_gitstats` *Create a* `GitStats` *object*

Description

Create a `GitStats` object

Usage

```
create_gitstats()
```

Value

A `GitStats` object.

Examples

```
my_gitstats <- create_gitstats()
```

get_commits *Get data on commits*

Description

List all commits from all repositories for an organization or a vector of repositories.

Usage

```
get_commits(
  gitstats_object,
  since = NULL,
  until = Sys.Date() + lubridate::days(1),
  cache = TRUE,
  verbose = is_verbose(gitstats_object),
  progress = verbose
)
```

Arguments

gitstats_object	A GitStats object.
since	A starting date.
until	An end date.
cache	A logical, if set to TRUE GitStats will retrieve the last result from its storage.
verbose	A logical, TRUE by default. If FALSE messages and printing output is switched off.
progress	A logical, by default set to verbose value. If FALSE no cli progress bar will be displayed.

Value

A data.frame.

Examples

```
## Not run:
my_gitstats <- create_gitstats() %>%
  set_github_host(
    token = Sys.getenv("GITHUB_PAT"),
    repos = c("openpharma/DataFakeR", "openpharma/visR")
  ) %>%
  set_gitlab_host(
    token = Sys.getenv("GITLAB_PAT_PUBLIC"),
    orgs = "mbtests"
  )
 get_commits(my_gitstats, since = "2018-01-01")
```

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

`get_commits_stats` *Get commits statistics*

Description

Prepare statistics from the pulled commits data.

Usage

```
get_commits_stats(gitstats_object, time_interval = c("month", "day", "week"))
```

Arguments

`gitstats_object`	A GitStats class object.
`time_interval`	A character, specifying time interval to show statistics.

Details

To make function work, you need first to get commits data with `GitStats`. See examples section.

Value

A table of `commits_stats` class.

Examples

```
## Not run:
 my_gitstats <- create_gitstats() %>%
   set_github_host(
     token = Sys.getenv("GITHUB_PAT"),
     repos = c("r-world-devs/GitStats", "openpharma/visR")
   )
 get_commits(my_gitstats, since = "2022-01-01")
 get_commits_stats(my_gitstats, time_interval = "week")

## End(Not run)
```

`get_files_content` *Get content of files*

Description

Pull text files content for a given scope (orgs, repos or whole git hosts).

Usage

```
get_files_content(
  gitstats_object,
  file_path = NULL,
  use_files_structure = TRUE,
  only_text_files = TRUE,
  cache = TRUE,
  verbose = is_verbose(gitstats_object),
  progress = verbose
)
```

Arguments

`gitstats_object`	A GitStats object.
`file_path`	Optional. A standardized path to file(s) in repositories. May be a character vector if multiple files are to be pulled. If set to `NULL` and `use_files_structure` parameter is set to `TRUE`, `GitStats` will try to pull data from `files_structure` (see below).
`use_files_structure`	Logical. If `TRUE` and `file_path` is set to `NULL`, will iterate over `files_structure` pulled by `get_files_structure()` function and kept in storage. If there is no `files_structure` in storage, an error will be returned. If `file_path` is defined, it will override `use_files_structure` parameter.
`only_text_files`	A logical, `TRUE` by default. If set to `FALSE`, apart from files with text content shows in table output also non-text files with `NA` value for text content.
`cache`	A logical, if set to `TRUE` GitStats will retrieve the last result from its storage.
`verbose`	A logical, `TRUE` by default. If `FALSE` messages and printing output is switched off.
`progress`	A logical, by default set to `verbose` value. If `FALSE` no `cli` progress bar will be displayed.

Value

A data.frame.

Examples

```
## Not run:
 my_gitstats <- create_gitstats() %>%
  set_github_host(
    token = Sys.getenv("GITHUB_PAT"),
    orgs = c("r-world-devs")
  ) %>%
  set_gitlab_host(
    token = Sys.getenv("GITLAB_PAT_PUBLIC"),
    orgs = "mbtests"
  )
 get_files_content(
   gitstats_obj = my_gitstats,
   file_path = c("LICENSE", "DESCRIPTION")
 )

 # example with files structure
 files_structure <- get_files_structure(
   gitstats_obj = my_gitstats,
   pattern = "\\.Rmd",
   depth = 2L
 )
 # get_files_content() will make use of pulled earlier files structure
 files_content <- get_files_content(
   gitstats_obj = my_gitstats
 )

## End(Not run)
```

---

get_files_structure *Get structure of files*

Description

Pulls file structure for a given repository.

Usage

```
get_files_structure(
  gitstats_object,
  pattern = NULL,
  depth = Inf,
  cache = TRUE,
  verbose = is_verbose(gitstats_object),
  progress = verbose
)
```

Arguments

gitstats_object	A GitStats object.
pattern	An optional regular expression. If defined, it pulls file structure for a repository matching this pattern.
depth	An optional integer. Defines level of directories to retrieve files from. E.g. if set to `0`, it will pull files only from root, if `1`, will take data from `root` directory and directories visible in `root` directory. If left with no argument, will pull files from all directories.
cache	A logical, if set to `TRUE` GitStats will retrieve the last result from its storage.
verbose	A logical, `TRUE` by default. If `FALSE` messages and printing output is switched off.
progress	A logical, by default set to `verbose` value. If `FALSE` no `cli` progress bar will be displayed.

Value

A list of vectors.

Examples

```
## Not run:
 my_gitstats <- create_gitstats() %>%
  set_github_host(
    token = Sys.getenv("GITHUB_PAT"),
    orgs = c("r-world-devs")
  ) %>%
  set_gitlab_host(
    token = Sys.getenv("GITLAB_PAT_PUBLIC"),
    orgs = "mbtests"
  )
 get_files_structure(
   gitstats_obj = my_gitstats,
   pattern = "\\.md"
 )

## End(Not run)
```

---

get_release_logs *Get release logs*

Description

Pull release logs from repositories.

Usage

```
get_release_logs(
  gitstats_object,
  since = NULL,
  until = Sys.Date() + lubridate::days(1),
  cache = TRUE,
  verbose = is_verbose(gitstats_object),
  progress = verbose
)
```

Arguments

gitstats_object	A GitStats object.
since	A starting date.
until	An end date.
cache	A logical, if set to TRUE GitStats will retrieve the last result from its storage.
verbose	A logical, TRUE by default. If FALSE messages and printing output is switched off.
progress	A logical, by default set to verbose value. If FALSE no cli progress bar will be displayed.

Value

A data.frame.

Examples

```
## Not run:
 my_gitstats <- create_gitstats() %>%
  set_github_host(
    token = Sys.getenv("GITHUB_PAT"),
    orgs = c("r-world-devs", "openpharma")
  )
  get_release_logs(my_gistats, since = "2024-01-01")

## End(Not run)
```

---

get_repos *Get data on repositories*

Description

Pulls data on all repositories for an organization, individual user or those with a given text in code blobs (with_code parameter) or a file (with_files parameter) and parse it into table format.

Usage

```
get_repos(
  gitstats_object,
  add_contributors = TRUE,
  with_code = NULL,
  in_files = NULL,
  with_files = NULL,
  cache = TRUE,
  verbose = is_verbose(gitstats_object),
  progress = verbose
)
```

Arguments

gitstats_object	A GitStats object.
add_contributors	A logical parameter to decide whether to add information about repositories' contributors to the repositories output (table). If set to FALSE it makes function run faster as, in the case of org search mode, it reaches only GraphQL endpoint with a query on repositories, and in the case of code search mode it reaches only repositories REST API endpoint. However, the pitfall is that the result does not convey information on contributors. When set to TRUE (by default), GitStats iterates additionally over pulled repositories and reaches to the contributors APIs, which makes it slower, but gives additional information.
with_code	A character vector, if defined, GitStats will pull repositories with specified code phrases in code blobs.
in_files	A character vector of file names. Works when with_code is set - then it searches code blobs only in files passed to in_files parameter.
with_files	A character vector, if defined, GitStats will pull repositories with specified files.
cache	A logical, if set to TRUE GitStats will retrieve the last result from its storage.
verbose	A logical, TRUE by default. If FALSE messages and printing output is switched off.
progress	A logical, by default set to verbose value. If FALSE no cli progress bar will be displayed.

Value

A data.frame.

Examples

```
## Not run:
my_gitstats <- create_gitstats() %>%
  set_github_host(
```

```
    token = Sys.getenv("GITHUB_PAT"),
    orgs = c("r-world-devs", "openpharma")
  ) %>%
  set_gitlab_host(
    token = Sys.getenv("GITLAB_PAT_PUBLIC"),
    orgs = "mbtests"
  )
get_repos(my_gitstats)
get_repos(my_gitstats, add_contributors = FALSE)
get_repos(my_gitstats, with_code = "Shiny", in_files = "renv.lock")
get_repos(my_gitstats, with_files = "DESCRIPTION")

## End(Not run)
```

`get_repos_urls` *Get repository URLS*

Description

Pulls a vector of repositories URLs (web or API): either all for an organization or those with a given text in code blobs (`with_code` parameter) or a file (`with_files` parameter).

Usage

```
get_repos_urls(
  gitstats_object,
  type = "web",
  with_code = NULL,
  in_files = NULL,
  with_files = NULL,
  cache = TRUE,
  verbose = is_verbose(gitstats_object),
  progress = verbose
)
```

Arguments

`gitstats_object`	A GitStats object.
`type`	A character, choose if `api` or `web` (`html`) URLs should be returned.
`with_code`	A character vector, if defined, GitStats will pull repositories with specified code phrases in code blobs.
`in_files`	A character vector of file names. Works when `with_code` is set - then it searches code blobs only in files passed to `in_files` parameter.
`with_files`	A character vector, if defined, GitStats will pull repositories with specified files.
`cache`	A logical, if set to `TRUE` GitStats will retrieve the last result from its storage.

verbose — A logical, TRUE by default. If FALSE messages and printing output is switched off.

progress — A logical, by default set to verbose value. If FALSE no cli progress bar will be displayed.

Value

A character vector.

Examples

```
## Not run:
my_gitstats <- create_gitstats() %>%
  set_github_host(
    token = Sys.getenv("GITHUB_PAT"),
    orgs = c("r-world-devs", "openpharma")
  ) %>%
  set_gitlab_host(
    token = Sys.getenv("GITLAB_PAT_PUBLIC"),
    orgs = "mbtests"
  )
get_repos_urls(my_gitstats, type = "api")
get_repos_urls(my_gitstats, with_files = c("DESCRIPTION", "LICENSE"))

## End(Not run)
```

---

get_R_package_usage *Get data on package usage across repositories*

Description

Wrapper over searching repositories by code blobs related to loading package (library(package) and require(package) in all files) or using it as a dependency (package in DESCRIPTION and NAMESPACE files).

Usage

```
get_R_package_usage(
  gitstats_object,
  packages,
  only_loading = FALSE,
  split_output = FALSE,
  cache = TRUE,
  verbose = is_verbose(gitstats_object)
)
```

Arguments

gitstats_object	A GitStats object.
packages	A character vector, names of R packages to look for.
only_loading	A boolean, if TRUE function will check only if package is loaded in repositories, not used as dependencies.
split_output	Optional, a boolean. If TRUE will return a list of tables, where every element of the list stands for the package passed to packages parameter. If FALSE, will return only one table with name of the package stored in first column.
cache	A logical, if set to TRUE GitStats will retrieve the last result from its storage.
verbose	A logical, TRUE by default. If FALSE messages and printing output is switched off.

Value

A tibble or list of tibbles depending on split_output parameter.

Examples

```
## Not run:
 my_gitstats <- create_gitstats() %>%
  set_github_host(
    token = Sys.getenv("GITHUB_PAT"),
    orgs = c("r-world-devs", "openpharma")
  )

 get_R_package_usage(
   gitstats_object = my_gitstats,
   packages = c("purrr", "shiny"),
   split_output = TRUE
 )

## End(Not run)
```

---

get_storage *Get data from* GitStats *storage*

Description

Retrieves whole or particular data (see storage parameter) pulled earlier with GitStats.

Usage

```
get_storage(gitstats_object, storage = NULL)
```

Arguments

gitstats_object	A GitStats object.
storage	A character, type of the data you want to get from storage: commits, repositories, release_logs, users, files, files_structure, R_package_usage or release_logs.

Value

A list of tibbles (if storage set to NULL) or a tibble (if storage defined).

Examples

```
## Not run:
 my_gitstats <- create_gitstats() %>%
  set_github_host(
    token = Sys.getenv("GITHUB_PAT"),
    orgs = c("r-world-devs", "openpharma")
  )
  get_release_logs(my_gistats, since = "2024-01-01")
  get_repos(my_gitstats)

  release_logs <- get_storage(
    gitstats_object = my_gitstats,
    storage = "release_logs"
  )

## End(Not run)
```

---

get_users *Get users data*

Description

Get users data

Usage

```
get_users(
  gitstats_object,
  logins,
  cache = TRUE,
  verbose = is_verbose(gitstats_object)
)
```

Arguments

gitstats_object	A GitStats object.
logins	A character vector of logins.
cache	A logical, if set to TRUE GitStats will retrieve the last result from its storage.
verbose	A logical, TRUE by default. If FALSE messages and printing output is switched off.

Value

A data.frame.

Examples

```
## Not run:
 my_gitstats <- create_gitstats() %>%
  set_github_host(
    token = Sys.getenv("GITHUB_PAT"),
    orgs = c("r-world-devs")
  ) %>%
  set_gitlab_host(
    token = Sys.getenv("GITLAB_PAT_PUBLIC"),
    orgs = "mbtests"
  )
 get_users(my_gitstats, c("maciekabanas", "marcinkowskak"))

## End(Not run)
```

---

is_verbose *Is verbose mode switched on*

Description

Is verbose mode switched on

Usage

```
is_verbose(gitstats_object)
```

Arguments

gitstats_object	A GitStats object.

`set_github_host` *Set GitHub host*

Description

Set GitHub host

Usage

```
set_github_host(
  gitstats_object,
  host = NULL,
  token = NULL,
  orgs = NULL,
  repos = NULL,
  verbose = is_verbose(gitstats_object)
)
```

Arguments

`gitstats_object`	A GitStats object.
`host`	A character, optional, URL name of the host. If not passed, a public host will be used.
`token`	A token.
`orgs`	An optional character vector of organisations. If you pass it, `repos` parameter should stay `NULL`.
`repos`	An optional character vector of repositories full names (organization and repository name, e.g. "r-world-devs/GitStats"). If you pass it, `orgs` parameter should stay `NULL`.
`verbose`	A logical, `TRUE` by default. If `FALSE` messages and printing output is switched off.

Details

If you do not define `orgs` and `repos`, `GitStats` will be set to scan whole Git platform (such as enterprise version of GitHub or GitLab), unless it is a public platform. In case of a public one (like GitHub) you need to define `orgs` or `repos` as scanning through all organizations may take large amount of time.

Value

A `GitStats` object with added information on host.

Examples

```
## Not run:
my_gitstats <- create_gitstats() %>%
  set_github_host(
    orgs = c("r-world-devs", "openpharma", "pharmaverse")
  )

## End(Not run)
```

`set_gitlab_host` *Set GitLab host*

Description

Set GitLab host

Usage

```
set_gitlab_host(
  gitstats_object,
  host = NULL,
  token = NULL,
  orgs = NULL,
  repos = NULL,
  verbose = is_verbose(gitstats_object)
)
```

Arguments

`gitstats_object`	A GitStats object.
`host`	A character, optional, URL name of the host. If not passed, a public host will be used.
`token`	A token.
`orgs`	An optional character vector of organisations. If you pass it, `repos` parameter should stay `NULL`.
`repos`	An optional character vector of repositories full names (organization and repository name, e.g. "r-world-devs/GitStats"). If you pass it, `orgs` parameter should stay `NULL`.
`verbose`	A logical, `TRUE` by default. If `FALSE` messages and printing output is switched off.

Details

If you do not define `orgs` and `repos`, `GitStats` will be set to scan whole Git platform (such as enterprise version of GitHub or GitLab), unless it is a public platform. In case of a public one (like GitHub) you need to define `orgs` or `repos` as scanning through all organizations may take large amount of time.

Value

A GitStats object with added information on host.

Examples

```
## Not run:
my_gitstats <- create_gitstats() %>%
  set_gitlab_host(
    token = Sys.getenv("GITLAB_PAT_PUBLIC"),
    orgs = "mbtests"
  )

## End(Not run)
```

---

show_orgs *Show organizations set in* GitStats

Description

Retrieves organizations set or pulled by GitStats. Especially helpful when user is scanning whole git platform and wants to have a glimpse at organizations.

Usage

```
show_orgs(gitstats_object)
```

Arguments

gitstats_object	A GitStats object.

Value

A vector of organizations.

---

verbose_off *Switch off verbose mode*

Description

Stop printing messages and output.

Usage

```
verbose_off(gitstats_object)
```

Arguments

`gitstats_object`
: A GitStats object.

Value

A GitStats object.

---

verbose_on *Switch on verbose mode*

Description

Print all messages and output.

Usage

```
verbose_on(gitstats_object)
```

Arguments

`gitstats_object`
: A GitStats object.

Value

A GitStats object.

Index