Package: roreviewapi (via r-universe)

October 17, 2024

Title Plumber API to report package structure and function

Version 0.1.0.056

Description Plumber API to report package structure and function.

License GPL-3

URL https://docs.ropensci.org/roreviewapi,
https://github.com/ropensci-review-tools/roreviewapi

BugReports https://github.com/ropensci-review-tools/roreviewapi/issues

Depends R (>= 4.1.0)

Imports BiocManager, callr, commonmark, fs, gert, ghql, gitcreds,
here, httr, jsonlite, logger, magrittr, methods, pkgcheck,
plumber, rappdirs, remotes, srr, tictoc, xml2

Suggests bspm, knitr, rmarkdown, roxygen2, rvest, testthat (>= 3.0.0),
withr

VignetteBuilder knitr

Remotes ropensci-review-tools/pkgcheck, ropensci-review-tools/srr

Encoding UTF-8

Language en-GB

LazyData true

NeedsCompilation yes

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

SystemRequirements gh, dos2unix

Config/testthat/edition 3

Repository https://ropensci-review-tools.r-universe.dev

RemoteUrl https://github.com/ropensci-review-tools/roreviewapi

RemoteRef HEAD

RemoteSha 936ee13a7daa639e85435f1e83b58e1e61af475e

Contents

---

check_cache   *check_cache*

Description

Check whether a package has been cached, and if so, whether commits have been added to the github repo since cached version.

Usage

```
check_cache(org, repo, cache_dir = fs::path_temp())
```

Arguments

org	Github organization
repo	Github repository
cache_dir	Directory in which packages are cached

Value

FALSE If a package has previously been cached, and the github repo has not changed; otherwise TRUE.

Note

This function is not intended to be called directly, and is only exported to enable it to be used within the **plumber** API.

See Also

Other utils: pkgrep_install_deps(), stdout_stderr_cache(), symbol_crs(), symbol_tck()

---

check_issue_template   *Check template variables in GitHub issue*

Description

Check template variables in GitHub issue

Usage

```
check_issue_template(orgrepo, issue_num)
```

Arguments

orgrepo   GitHub organization and repo as single string separated by forward slash (org/repo).

issue_num   Number of issue from which to extract opening comment

Value

Comment as character string

See Also

Other ropensci: is_user_authorized(), push_to_gh_pages(), readme_has_peer_review_badge(), srr_counts(), srr_counts_from_report(), srr_counts_summary(), stats_badge()

---

collate_editor_check   *Collate list of checks to single concatenated character string*

Description

Collate list of checks to single concatenated character string

Usage

```
collate_editor_check(checks)
```

Arguments

checks   Result of pkgcheck::pgkcheck function

Value

Single character

Note

Exported only for access by plumber; not intended for general external usage.

See Also

Other main: [editor_check](), [serve_api]()

---

dl_gh_repo   *Download a GitHub repo to local cache*

Description

Download a GitHub repo to local cache

Usage

```
dl_gh_repo(u, branch = NULL)
```

Arguments

u   URL of GitHub repository

branch   Checkout specified (non-default) branch of repo.

Value

Path to locally cached '.zip' version of repository

See Also

Other github: [get_branch_from_url](), [post_to_issue]()

editor_check	*Body of main 'editorcheck' response*

Description

Body of main 'editorcheck' response

Usage

```
editor_check(repourl, repo, issue_id, post_to_issue = TRUE)
```

Arguments

repourl	The URL for the repo being checked, potentially including full path to non-default branch.
repo	The 'context.repo' parameter defining the repository from which the command was invoked, passed in 'org/repo' format.
issue_id	The id (number) of the issue from which the command was invoked.
post_to_issue	Integer value > 0 will post results back to issue (via 'gh' cli); otherwise just return character string with result.

Value

If !post_to_issue, a markdown-formatted response body from static package checks, otherwise URL of the issue comment to which response body has been posted.

See Also

Other main: collate_editor_check(), serve_api()

get_branch_from_url	*Get branch from a GitHub URL if non-default branch specified there*

Description

Get branch from a GitHub URL if non-default branch specified there

Usage

```
get_branch_from_url(repourl)
```

Arguments

repourl	Potentially with "/tree/branch_name" appended

Value

Branch as single string.

See Also

Other github: [dl_gh_repo](), [post_to_issue]()

---

is_user_authorized	*Check whether a user, identified from GitHub API token, is authorized to call endpoints.*

Description

This function is used only in the **plumber** endpoints, to prevent them being called by unauthorized users.

Usage

```
is_user_authorized(secret = NULL)
```

Arguments

secret   Environment variable PKGCHECK_TOKEN sent from bot.

Value

Logical value indicating whether or not a user is authorized.

See Also

Other ropensci: [check_issue_template](), [push_to_gh_pages](), [readme_has_peer_review_badge](), [srr_counts](), [srr_counts_from_report](), [srr_counts_summary](), [stats_badge]()

---

pkgrep_install_deps	*Install all system and package dependencies of an R package*

Description

Install all system and package dependencies of an R package

Usage

```
pkgrep_install_deps(path, repo, issue_id)
```

Arguments

path	Path to local file or directory
repo	The 'context.repo' parameter defining the repository from which the command was invoked, passed in 'org/repo' format.
issue_id	The id (number) of the issue from which the command was invoked.

Value

Hopefully a character vector of length zero, otherwise a message detailing any R packages unable to be installed.

See Also

Other utils: `check_cache()`, `stdout_stderr_cache()`, `symbol_crs()`, `symbol_tck()`

---

post_to_issue   *Post review checks to GitHub issue*

Description

Post review checks to GitHub issue

Usage

```
post_to_issue(cmt, repo, issue_id)
```

Arguments

cmt	Single character string with comment to post.
repo	The repository to post to (obtained directly from bot).
issue_id	The number of the issue to post to.

Value

URL of the comment within the nominated issue

See Also

Other github: `dl_gh_repo()`, `get_branch_from_url()`

---

push_to_gh_pages	*Push static* html *files to* gh-pages *branch of this repo to serve via GitHub pages.*

Description

Push static html files to gh-pages branch of this repo to serve via GitHub pages.

Usage

```
push_to_gh_pages(check)
```

Arguments

check	Return result of [editor_check](#) function.

Value

Vector of two paths containing URLs of the srr and network files.

See Also

Other ropensci: [check_issue_template()](#), [is_user_authorized()](#), [readme_has_peer_review_badge()](#), [srr_counts()](#), [srr_counts_from_report()](#), [srr_counts_summary()](#), [stats_badge()](#)

---

readme_badge	*Check whether README.md features an rOpenSci software-review badge*

Description

Check whether README.md features an rOpenSci software-review badge

Usage

```
readme_badge(repourl, repo = NULL, issue_id = NULL, post_to_issue = TRUE)
```

Arguments

repourl	The URL for the repo being checked, potentially including full path to non-default branch.
repo	The 'context.repo' parameter defining the repository from which the command was invoked, passed in 'org/repo' format.
issue_id	The id (number) of the issue from which the command was invoked.
post_to_issue	Integer value > 0 will post results back to issue (via 'gh' cli); otherwise just return character string with result.

Value

A string, empty if the badge was found.

---

readme_has_peer_review_badge
Check whether 'README.md' has a "peer reviewed" badge

Description

Check whether 'README.md' has a "peer reviewed" badge

Usage

```
readme_has_peer_review_badge(path = getwd(), issue_id = NULL)
```

Arguments

path	Local path to package directory.
issue_id	The id (number) of the issue from which the command was invoked.

Value

A string, empty if the badge was found.

See Also

Other ropensci: check_issue_template(), is_user_authorized(), push_to_gh_pages(), srr_counts(), srr_counts_from_report(), srr_counts_summary(), stats_badge()

---

serve_api   *serve plumber API to report on packages*

Description

The API exposes the single POST points of report to download software from the given URL and return a textual analysis of its structure and functionality.

Usage

```
serve_api(port = 8000L, cache_dir = NULL, os = "", os_release = "")
```

Arguments

port	Port for API to be exposed on
cache_dir	Directory where previously downloaded repositories are cached
os	Name of operating system, passed to **remotes** package to install system dependencies.
os_release	Name of operating system release, passed to **remotes** package to install system dependencies.

Value

Nothing; calling this starts a blocking process.

See Also

Other main: `collate_editor_check()`, `editor_check()`

---

srr_counts	*Count number of 'srr' statistical standards complied with, and confirm whether than represents > 50% of all applicable standards.*

Description

Count number of 'srr' statistical standards complied with, and confirm whether than represents > 50% of all applicable standards.

Usage

```
srr_counts(repourl, repo, issue_id, post_to_issue = TRUE)
```

Arguments

repourl	The URL for the repo being checked, potentially including full path to non-default branch.
repo	The 'context.repo' parameter defining the repository from which the command was invoked, passed in 'org/repo' format.
issue_id	The id (number) of the issue from which the command was invoked.
post_to_issue	Integer value > 0 will post results back to issue (via 'gh' cli); otherwise just return character string with result.

Value

Vector of three numbers:

1. Number of standards complied with
2. Total number of applicable standards
3. Number complied with as proportion of total

See Also

Other ropensci: `check_issue_template()`, `is_user_authorized()`, `push_to_gh_pages()`, `readme_has_peer_review_b`
`srr_counts_from_report()`, `srr_counts_summary()`, `stats_badge()`

---

srr_counts_from_report

Extract final counts of 'srr' standards from the report

Description

Extract final counts of 'srr' standards from the report

Usage

```
srr_counts_from_report(srr_rep)
```

Arguments

srr_rep   An 'srr' report generated by the `srr::srr_report()` function.

Value

Character vector with markdown-formatted summary summary of numbers of standards complied with.

See Also

Other ropensci: `check_issue_template()`, `is_user_authorized()`, `push_to_gh_pages()`, `readme_has_peer_review_b`
`srr_counts()`, `srr_counts_summary()`, `stats_badge()`

---

srr_counts_summary   *Summarise counts of 'srr' standards from full 'srr' report*

Description

Summarise counts of 'srr' standards from full 'srr' report

Usage

```
srr_counts_summary(srr_rep, has_errors = FALSE)
```

Arguments

srr_rep   An 'srr' report generated by the `srr::srr_report()` function.

has_errors   This is `TRUE` when the 'srr' report has errors, in which case the final message issued here excludes statements about package being able to be submitted.

Value

Character vector with markdown-formatted summary summary of numbers of standards complied with.

See Also

Other ropensci: check_issue_template(), is_user_authorized(), push_to_gh_pages(), readme_has_peer_review_b
srr_counts(), srr_counts_from_report(), stats_badge()

---

stats_badge   *Get stats badge grade and standards version for a submission*

Description

Get stats badge grade and standards version for a submission

Usage

```
stats_badge(repo = "ropensci/software-review", issue_num = 258)
```

Arguments

repo   The submission repo

issue_num   GitHub issue number of submission

Value

A single character containing the label used directly for the issue badge

See Also

Other ropensci: check_issue_template(), is_user_authorized(), push_to_gh_pages(), readme_has_peer_review_b
srr_counts(), srr_counts_from_report(), srr_counts_summary()

stdout_stderr_cache   *Set up stdout & stderr cache files for r_bg process*

Description

Set up stdout & stderr cache files for r_bg process

Usage

```
stdout_stderr_cache(repourl)
```

Arguments

repourl   The URL of the repo being checked

Value

Vector of two strings holding respective local paths to stdout and stderr files for r_bg process controlling the main editor_check function.

Note

These files are needed for the **callr** r_bg process which controls the main editor_check call. See <https://github.com/r-lib/callr/issues/204>. The stdout and stderr pipes from the process are stored in the cache directory so they can be inspected via their own distinct endpoint calls.

See Also

Other utils: check_cache(), pkgrep_install_deps(), symbol_crs(), symbol_tck()

symbol_crs   *Cross symbol, exported for direct use in plumber API*

Description

Cross symbol, exported for direct use in plumber API

Usage

```
symbol_crs()
```

See Also

Other utils: check_cache(), pkgrep_install_deps(), stdout_stderr_cache(), symbol_tck()

symbol_tck *Tick symbol, exported for direct use in plumber API*

Description

Tick symbol, exported for direct use in plumber API

Usage

```
symbol_tck()
```

See Also

Other utils: `check_cache()`, `pkgrep_install_deps()`, `stdout_stderr_cache()`, `symbol_crs()`

Index