Package: roreviewapi (via r-universe)

June 1, 2026

Title Plumber API to report package structure and function

Version 0.2.1.002

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 airtabler, BiocManager, callr, commonmark, DBI, fs, gert, gh, ghql, gitcreds, here, httr2, jsonlite, later, logger, magrittr, methods, openssl, pkgcheck, plumber, rappdirs, remotes, RSQLite, srr, tictoc, xml2

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

VignetteBuilder knitr

Remotes bergant/airtabler, ropensci-review-tools/pkgcheck, ropensci-review-tools/pkgmatch, ropensci-review-tools/srr

Encoding UTF-8

Language en-GB

LazyData true

NeedsCompilation yes

Roxygen list(markdown = TRUE)

SystemRequirements gh, dos2unix

Config/testthat/edition 3

Config/ropensci/maintainer staff

Config/roxygen2/version 8.0.0

Config/pak/sysreqs cmake git libglpk-dev make libgit2-dev libicu-dev libsodium-dev libuv1-dev libxml2-dev libssl-dev libx11-dev zlib1g-dev

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

Date/Publication 2026-05-27 07:21:49 UTC

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

RemoteRef HEAD

RemoteSha 4b1b0cb6628dcafde4a0fc8c921887a311c9d075

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(), issue_is_stats(), 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()

cran_to_list *Convert names of CRAN packages to markdown-formatted list.*

Description

This function is exported because it needs to be called in the main plumber endpoint function.

Usage

```
cran_to_list(matches, n = 5L)
```

Arguments

matches	A **pkgmatch** data.frame object with columns of ("package", "version", "rank").
n	Number of matches to return in list.

See Also

Other pkgmatch: pkgmatch_repo(), ros_to_list()

deactivate_search *Deactivate a volunteer search and delete all associated data*

Description

Sets active = 0 first as a guard against concurrent clicks, then deletes all recipient rows followed by the search row itself.

Usage

```
deactivate_search(repo, issue_id)
```

Arguments

repo	GitHub review repository in org/repo format.
issue_id	Integer issue number in the review repository.

Value

Named list with deactivated (logical) and issue_ref.

See Also

Other email: handle_click(), list_searches(), send_search()

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(), get_subdir_from_url(), post_to_issue(), url_is_r_pkg()

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/sub-directory" appended

Value

Branch as single string.

See Also

Other github: `dl_gh_repo()`, `get_subdir_from_url()`, `post_to_issue()`, `url_is_r_pkg()`

---

get_subdir_from_url *Return sub-directory from URL if present. This function is also intended to test whether packages are in sub-directories (issue #64)*

Description

Return sub-directory from URL if present. This function is also intended to test whether packages are in sub-directories (issue #64)

Usage

```
get_subdir_from_url(repourl)
```

Arguments

`repourl`	Potentially with "/tree/branch_name/sub-directory" appended

See Also

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

`handle_click` *Handle a volunteer link click*

Description

Looks up the token, checks whether the parent search is still active, guards against double-clicks, records the click timestamp, and triggers a Postmark notification (Phase 2).

Usage

```
handle_click(token)
```

Arguments

`token`	64-character hex token from the recipient's unique link.

Value

Named list with `status` (integer HTTP status code) and `body` (character HTML string).

See Also

Other email: `deactivate_search()`, `list_searches()`, `send_search()`

`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(), issue_is_stats(), push_to_gh_pages(), readme_has_peer_review_badge
srr_counts(), srr_counts_from_report(), srr_counts_summary(), stats_badge()

`issue_is_stats` *Determine whether a GitHub issue is a Stats submission*

Description

Determine whether a GitHub issue is a Stats submission

Usage

```
issue_is_stats(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 submission type.

Value

`TRUE` if the submission type is "Stats", otherwise `FALSE`.

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(), stats_badge()

`list_searches` *List all volunteer searches with recipient and click counts*

Description

List all volunteer searches with recipient and click counts

Usage

```
list_searches()
```

Value

Data frame with columns `search_id`, `created_at`, `notify_email`, `active`, `total`, `clicked`.

See Also

Other email: deactivate_search(), handle_click(), send_search()

pkgmatch_repo *Body of main 'pkgmatch' response*

Description

Body of main 'pkgmatch' response

Usage

```
pkgmatch_repo(repourl, repo, issue_id, n_top = 5L, post_to_issue = TRUE)
```

Arguments

repourl	The URL for the repo being matched, 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.
n_top	Return this number of top-matches packages.
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, otherwise URL of the issue comment to which response body has been posted.

See Also

Other pkgmatch: cran_to_list(), ros_to_list()

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(), get_subdir_from_url(), url_is_r_pkg()

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()`, `issue_is_stats()`, `readme_has_peer_review_bad`
`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()`, `issue_is_stats()`, `push_to_gh_pages()`, `srr_counts()`, `srr_counts_from_report()`, `srr_counts_summary()`, `stats_badge()`

`ros_to_list` *Convert names of rOpenSci packages to markdown-formatted list.*

Description

This function is exported because it needs to be called in the main plumber endpoint function.

Usage

```
ros_to_list(matches, n = 5L)
```

Arguments

`matches`	A **pkgmatch** `data.frame` object with columns of ("package", "version", "rank").
`n`	Number of matches to return in list.

See Also

Other pkgmatch: `cran_to_list()`, `pkgmatch_repo()`

send_search *Send a batch of editor search emails*

Description

Fetches current editor email addresses via `get_editor_emails()`, inserts a new search record and one recipient row per address into the database, then dispatches emails via Postmark. The notify address is read from the AirTable cache written by the internal 'notify_email_refresh' function. The base URL for click links is read from the `ROREVIEWAPI_BASE_URL` environment variable. The stats/standard distinction is determined by calling `issue_is_stats()` on the supplied `repo` and `issue_id`.

Usage

```
send_search(
  repourl,
  repo,
  issue_id,
  subject = "Seeking editors for rOpenSci software submission",
  fetcher = NULL,
  stats_checker = roreviewapi::issue_is_stats
)
```

Arguments

`repourl`	URL of the package repository; included in the outgoing emails.
`repo`	GitHub review repository in `org/repo` format.
`issue_id`	Integer issue number in the review repository.
`subject`	Subject line for the outgoing emails.
`fetcher`	Function used to fetch editor emails; injectable for testing. Must accept `(airtable_base_id, stats)` and return a character vector.
`stats_checker`	Function used to determine submission type; injectable for testing. Must accept `(repo, issue_id)` and return a logical.

Value

Named list with `search_id` (integer) and `sent` (integer).

See Also

Other email: `deactivate_search()`, `handle_click()`, `list_searches()`

`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(), issue_is_stats(), push_to_gh_pages(), readme_has_peer_review_badge(), 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(), issue_is_stats(), push_to_gh_pages(), readme_has_peer_review_badge(), 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()`, `issue_is_stats()`, `push_to_gh_pages()`, `readme_has_peer_review_badge()`, `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(), issue_is_stats(), push_to_gh_pages(), readme_has_peer_review_badge(), 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()`

url_is_r_pkg *Check whether a given GitHub URL contains an R package.*

Description

Check whether a given GitHub URL contains an R package.

Usage

```
url_is_r_pkg(repourl)
```

Arguments

`repourl`	Potentially with "/tree/branch_name/sub-directory" appended

Value

`TRUE` if 'repourl' is an R package, otherwise `FALSE`

See Also

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

Index