

Package: ghactions4r (via r-universe)

October 25, 2026

Title Use GitHub Actions Reusable Workflows for R Packages

Version 1.0.0

Description Provides functions to allow users to set up github action workflows related to R packages.

License CC0

Imports cli, httr, jsonlite, lifecycle, rlang, usethis

Suggests dplyr, knitr, rmarkdown, rvest, testthat (>= 3.0.0), visNetwork

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Config/pak/sysreqs cmake git make libgit2-dev libuv1-dev libssl-dev libx11-dev

Repository <https://noaa-fisheries-integrated-toolbox.r-universe.dev>

Date/Publication 2026-09-25 22:15:21 UTC

RemoteUrl <https://github.com/nmfs-ost/ghactions4r>

RemoteRef HEAD

RemoteSha 9b1a2e5f20be8c45ddcc5df20bb275590e24a265

Contents

rm_dollar_sign	2
use_build_pkgdown	3
use_calc_cov_summaries	4
use_create_cov_badge	5
use_doc_and_style_r	6
use_r_cmd_check	9
use_spell_check	10
use_update_pkgdown	11

Index**14**

rm_dollar_sign	<i>Replace dollar sign operator in a file</i>
----------------	---

Description

The dollar sign is convenient to write, but allows for partial matching, which we don't often want. This function takes a file and changes all dollar signs to double brackets with names in quotations instead. Note that this function will incorrectly convert text enclosed in backticks that includes a dollar sign.

Usage

```
rm_dollar_sign(file, out_file = file, allow_recursive = TRUE, max_loops = 5)
```

Arguments

file	Filename either with full path or relative to working directory.
out_file	The name or path of a new file to write to. This is by default the same as the original file. Set to NULL to avoid writing a new file.
allow_recursive	Should dollar sign references of list in list be replaced? If this is FALSE, then only the first reference will be replaced. For example, F\$first\$second would become F[["first"]]\$second when allow_recursive is FALSE, but would become F[["first"]][["second"]] if TRUE.
max_loops	How many times should dollar signs be looked for recursively, if allow_recursive is TRUE? Defaults to 5. This is meant to prevent an infinite loop in case the function does not work properly.

Value

A character vector of the modified text. As a side effect, produces a text file (name specified in out_file) written out from R using writeLines().

Author(s)

Kathryn Doering

Examples

```
test_text <- c(
  "x$my_name <- y$test",
  "x[['my_name']]",
  "no_replace_here <- 44",
  "x$names<-new_assign;x$`22`",
  "x$names <- new_assign; x$`22`",
  "x$`$$weirdcharacters`<-222",
```

```

"x$nameinbacktick",
"x$mylist$my_col$YetAnotherCol",
"x$mylist$my_col$`1_somename`"
)
writeLines(test_text, "test_rm_dollar_sign.txt")
new_text <- rm_dollar_sign(
  file = "test_rm_dollar_sign.txt",
  out_file = NULL
)
new_text
file.remove("test_rm_dollar_sign.txt")

```

use_build_pkgdown	<i>use workflow in current pkg to check pkgdown site builds.</i>
-------------------	--

Description

This workflow will try to build the pkgdown site, and store it as an artifact to be retrieved from the GitHub actions run. It will fail if the pkgdown site cannot be built. This is a companion to `use_update_pkgdown()`, as the `update_pkgdown` the site, and therefore can be used to test if the build is working in cases where you do not want to deploy as well.

Usage

```

use_build_pkgdown(
  workflow_name = "call-build-pkgdown.yml",
  build_trigger = "pull_request",
  additional_args = NULL,
  tag_ghactions4r = NULL
)

```

Arguments

<code>workflow_name</code>	What to name the github action workflow locally. Must have the file extension <code>.yml</code>
<code>build_trigger</code>	Select the event that triggers the workflow . Options are: <code>pull_request</code> Run when a pull request is opened, reopened, or updated (default) <code>push_to_main</code> Run on pushing commits to main <code>push_to_all_branches</code> Run on pushing commits to any branch <code>manually</code> Run manually with the <code>workflow_dispatch</code> trigger <code>weekly</code> Run on the default branch (usually main) once a week Multiple build triggers are allowed; specify them as a vector. Note that invalid build triggers will be silently removed as long as one build trigger is specified correctly.

additional_args

A named list of additional command line arguments to be passed to the workflow. The names of the list represent the platforms (windows, macos, or ubuntu), and the values are character vectors of arguments. These additional arguments are executed after the step that checks out the repository and before the step that sets up R.

tag_ghactions4r

Tag of ghactions4r workflow to pin to for the reusable workflow. This ensures compatibility with a specific version of ghactions4r functions. The tradeoff is that fixes will not happen automatically; instead, the user will have to update the pinned version of the caller workflow used. See the options by looking at the [ghactions4r tags](#). Another option is to pin to commit hashes, but the user will need to do this by manually editing the caller workflow after creation. See [github documentation on security](#) to understand how this is more secure. Again, the tradeoff is that the workflow will not be automatically updated.

Examples

```
## Not run:
use_build_pkgdown(
  additional_args = list(
    ubuntu = c(
      "sudo apt-get update",
      "sudo apt-get install -y libcurl4-openssl-dev",
      "sudo add-apt-repository ppa:ubuntu-toolchain-r/test",
      "sudo apt-get install --only-upgrade libstdc++6"
    ),
    macos = c("brew install curl")
  )
)

## End(Not run)
```

use_calc_cov_summaries

Workflow for calculating coverage and summarizing using octocov toolkit

Description

This workflow calculates coverage using the covr package, then creates summaries to post as a GitHub action summary or as pull request comment. All data remains in the GitHub repository.

Usage

```
use_calc_cov_summaries(
  workflow_name = "call-calc-cov-summaries.yml",
  build_trigger = "pull_request",
```

```

    use_public_rspm = TRUE,
    depends_on_quarto = FALSE,
    tag_ghactions4r = NULL
  )

```

Arguments

- workflow_name** What to name the github action workflow locally. Must have the file extension `.yml`
- build_trigger** Select the **event that triggers the workflow**. Options are:
- `pull_request` Run when a pull request is opened, reopened, or updated (default)
 - `push_to_main` Run on pushing commits to main
 - `push_to_all_branches` Run on pushing commits to any branch
 - `manually` run manually with the `workflow_dispatch` trigger
- Multiple build triggers are allowed; specify them as a vector. Note that invalid build triggers will be silently removed as long as one build trigger is specified correctly.
- use_public_rspm** Use posit package manager instead of CRAN to install dependencies? The advantage here is that dependencies are precompiled, so install should be much quicker. In rare situations (like packages with TMB dependencies), using `use_public_rspm = FALSE` may be a better option. Note a setting only needs to be specified in the yml if `use_public_rspm` is `FALSE`, so there will be no setting added if `use_public_rspm` is `TRUE`.
- depends_on_quarto** Adds an option that installs quarto as a dependency.
- tag_ghactions4r** Tag of ghactions4r workflow to pin to for the reusable workflow. This ensures compatibility with a specific version of ghactions4r functions. The tradeoff is that fixes will not happen automatically; instead, the user will have to update the pinned version of the caller workflow used. See the options by looking at the **ghactions4r tags**. Another option is to pin to commit hashes, but the user will need to do this by manually editing the caller workflow after creation. See **github documentation on security** to understand how this is more secure. Again, the tradeoff is that the workflow will not be automatically updated.

`use_create_cov_badge` *workflow for adding a code coverage badge*

Description

This workflow will create a new code coverage badge with the latest overall coverage value from the main branch on every push to main. The coverage badges are pushed to a branch called badges within the repository, which can be referenced in the readme file on main to display the current code coverage.

Usage

```
use_create_cov_badge(
  workflow_name = "call-create-cov-badge.yml",
  build_trigger = "push_to_main",
  use_public_rspm = TRUE,
  depends_on_quarto = FALSE,
  tag_ghactions4r = NULL
)
```

Arguments

- | | |
|-------------------|---|
| workflow_name | What to name the github action workflow locally. Must have the file extension .yml |
| build_trigger | Select the event that triggers the workflow . Options are:
push_to_main Run on pushing commits to main (default)
weekly Run on the default branch (usually main) once a week
manually run manually with the workflow_dispatch trigger
Multiple build triggers are allowed; specify them as a vector. Note that invalid build triggers will be silently removed as long as one build trigger is specified correctly. |
| use_public_rspm | Use posit package manager instead of CRAN to install dependencies? The advantage here is that dependencies are precompiled, so install should be much quicker. In rare situations (like packages with TMB dependencies), using use_public_rspm = FALSE may be a better option. Note a setting only needs to be specified in the yml if use_public_rspm is FALSE, so there will be no setting added if use_public_rspm is TRUE. |
| depends_on_quarto | Adds an option that installs quarto as a dependency. |
| tag_ghactions4r | Tag of ghactions4r workflow to pin to for the reusable workflow. This ensures compatibility with a specific version of ghactions4r functions. The tradeoff is that fixes will not happen automatically; instead, the user will have to update the pinned version of the caller workflow used. See the options by looking at the ghactions4r tags . Another option is to pin to commit hashes, but the user will need to do this by manually editing the caller workflow after creation. See github documentation on security to understand how this is more secure. Again, the tradeoff is that the workflow will not be automatically updated. |

use_doc_and_style_r	<i>Use workflow in current pkg to automate documenting and styling</i>
---------------------	--

Description

Style your R package components automatically by running devtools::document(), styler::style_pkg() or air, and usethis::use_tidy_description().

Usage

```
use_doc_and_style_r(
  workflow_name = "call-doc-and-style-r.yml",
  use_rm_dollar_sign = FALSE,
  how_to_commit = c("pull_request", "directly"),
  build_trigger = "push_to_main",
  use_air = FALSE,
  use_pat = FALSE,
  pat_name = "PAT",
  tag_ghactions4r = NULL
)
```

Arguments

workflow_name	What to name the github action workflow locally. Must have the file extension .yml
use_rm_dollar_sign	in addition to devtools::document and styler::style_pkg, should ghactions4r::rm_dollar_sign be run? Defaults to FALSE.
how_to_commit	Where should changes made to style and documentation be committed? Options are 1) in a pull request to the branch ("pull_request") where the workflow started; or 2) directly to the branch ("directly") where the workflow started.
build_trigger	Select the event that triggers the workflow . Options are: push_to_main Run on pushing commits to main (default) pull_request Run when a pull request is opened, reopened, or updated pr_comment Run when a comment containing the command \doc-and-style is made on a pull request by people with write permissions on the repository manually Run manually with the workflow_dispatch trigger weekly Run on the default branch (usually main) once a week Multiple build triggers are allowed; specify them as a vector. Note that invalid build triggers will be silently removed as long as one build trigger is specified correctly.
use_air	Use air instead of styler to style files? Defaults to FALSE.
use_pat	Should a personal access token (PAT) stored as a GitHub secret be used? This is only necessary if you want the pull request generated by the doc and style workflow to be able to start other GitHub action runs.
pat_name	Name of the personal access token (PAT), as stored in secrets. Only used if use_pat = TRUE. After generating the personal access token , See how to add it as a secret .
tag_ghactions4r	Tag of ghactions4r workflow to pin to for the reusable workflow. This ensures compatibility with a specific version of ghactions4r functions. The tradeoff is that fixes will not happen automatically; instead, the user will have to update the pinned version of the caller workflow used. See the options by looking at the ghactions4r tags . Another option is to pin to commit hashes, but the user

will need to do this by manually editing the caller workflow after creation. See [github documentation on security](#) to understand how this is more secure. Again, the tradeoff is that the workflow will not be automatically updated.

Details

The pull request created from this workflow can trigger GitHub Action runs by using a personal access token (instead of the standard GITHUB_TOKEN) to provide the right permissions. The PAT is first [generated on a user account](#), then [added to the repository or organization as a secret](#). Using a PAT can cause recursive runs, so should be used with caution. See [github documentation on triggering a workflow from a workflow](#) for more information.

If using the PR_comment build trigger, note that the command to run the workflow

Examples

```
# set up running the doc and style workflow on each push to main, opening a
# pull request to main when changes are found.
## Not run:
use_doc_and_style_r()

## End(Not run)
# the same as
## Not run:
use_doc_and_style_r(
  how_to_commit = "pull_request",
  build_trigger = "push_to_main"
)

## End(Not run)
# Set up running doc and style on each pull request, committing directly to
# the pull request branch
## Not run:
use_doc_and_style_r(
  how_to_commit = "directly",
  build_trigger = "pull_request"
)

## End(Not run)
# Set up the workflow to use a personal access token (PAT)
## Not run:
use_doc_and_style_r(use_pat = TRUE, pat_name = "PAT")

## End(Not run)

# set up to run doc and style on a pull request, where an owner or member of
# the repository adds a comment on the pull request with the command \doc-and-style
## Not run:
use_doc_and_style_r(build_trigger = "pr_comment")

## End(Not run)
```

use_r_cmd_check	<i>Use workflow to run r cmd check on Linux, Mac, and Windows GitHub Actions</i>
-----------------	--

Description

Use workflow to run r cmd check on Linux, Mac, and Windows GitHub Actions

Usage

```
use_r_cmd_check(
  workflow_name = "call-r-cmd-check.yml",
  build_trigger = "push_to_main",
  use_full_build_matrix = FALSE,
  depends_on_tmb = FALSE,
  depends_on_quarto = FALSE,
  additional_args = NULL,
  tag_ghactions4r = NULL
)
```

Arguments

workflow_name	What to name the github action workflow locally. Must have the file extension .yml
build_trigger	Select the event that triggers the workflow . Options are: push_to_main Run on pushing commits to main (default) push_to_all_branches Run on pushing commits to any branch pull_request Run when a pull request is opened, reopened, or updated manually Run manually with the workflow_dispatch trigger weekly Run on the default branch (usually main) once a week Multiple build triggers are allowed; specify them as a vector. Note that invalid build triggers will be silently removed as long as one build trigger is specified correctly.
use_full_build_matrix	Run R cmd check with two older versions of R in addition to the three runs that use the release version.
depends_on_tmb	Adds an option that install Matrix from source for windows and Mac builds to solved a nuanced issue for packages dependent on TMB. See this google groups thread for more information.
depends_on_quarto	Adds an option that installs quarto as a dependency.
additional_args	A named list of additional command line arguments to be passed to the workflow. The names of the list represent the platforms (windows, macos, or ubuntu), and the values are character vectors of arguments. These additional arguments

are executed after the step that checks out the repository and before the step that sets up R.

tag_ghactions4r

Tag of ghactions4r workflow to pin to for the reusable workflow. This ensures compatibility with a specific version of ghactions4r functions. The tradeoff is that fixes will not happen automatically; instead, the user will have to update the pinned version of the caller workflow used. See the options by looking at the [ghactions4r tags](#). Another option is to pin to commit hashes, but the user will need to do this by manually editing the caller workflow after creation. See [github documentation on security](#) to understand how this is more secure. Again, the tradeoff is that the workflow will not be automatically updated.

Examples

```
## Not run:
use_r_cmd_check(
  additional_args = list(
    ubuntu = c(
      "sudo apt-get update",
      "sudo apt-get install -y libcurl4-openssl-dev",
      "sudo add-apt-repository ppa:ubuntu-toolchain-r/test",
      "sudo apt-get install --only-upgrade libstdc++6"
    ),
    macos = c("brew install curl")
  )
)

## End(Not run)
```

use_spell_check	<i>use workflow to run spelling::spell_check_package()</i>
-----------------	--

Description

This workflow will run the `spell_check_package()` function from the `spelling` package on the current package, and will fail if any spelling errors are found. It will also run the `spell_check_files()` on specified file types if `spell_check_additional_files` is `TRUE`.

Usage

```
use_spell_check(
  workflow_name = "call-spell-check.yml",
  build_trigger = "pull_request",
  spell_check_additional_files = FALSE,
  spell_check_report_level = c("error", "warning"),
  tag_ghactions4r = NULL
)
```

Arguments

- workflow_name** What to name the github action workflow locally. Must have the file extension `.yaml`
- build_trigger** Select the **event that triggers the workflow**. Options are:
- `pull_request` Run when a pull request is opened, reopened, or updated (default)
 - `push_to_main` Run on pushing commits to main
 - `push_to_all_branches` Run on pushing commits to any branch
 - `manually` Run manually with the `workflow_dispatch` trigger
 - `weekly` Run on the default branch (usually main) once a week
- Multiple build triggers are allowed; specify them as a vector. Note that invalid build triggers will be silently removed as long as one build trigger is specified correctly.
- spell_check_additional_files**
Logical. Should the workflow run `spell_check_files()` on specified files in the package? Defaults to `FALSE`.
- spell_check_report_level**
Character. The level of the report to generate for the additional spell check. Options are "warning" or "error". Defaults to "error". If set to "error", the workflow will fail if any spelling errors are found. If set to "warning", the workflow will pass even if spelling errors are found.
- tag_ghactions4r**
Tag of ghactions4r workflow to pin to for the reusable workflow. This ensures compatibility with a specific version of ghactions4r functions. The tradeoff is that fixes will not happen automatically; instead, the user will have to update the pinned version of the caller workflow used. See the options by looking at the **ghactions4r tags**. Another option is to pin to commit hashes, but the user will need to do this by manually editing the caller workflow after creation. See **github documentation on security** to understand how this is more secure. Again, the tradeoff is that the workflow will not be automatically updated.

<code>use_update_pkgdown</code>	<i>Creates a workflow in current R package to update an existing pkgdown GitHub pages site</i>
---------------------------------	--

Description

For more information, see **the setup details vignette** before running this function. This workflow will assume the website is built from a branch called `gh-pages`.

Usage

```
use_update_pkgdown(
  workflow_name = "call-update-pkgdown.yml",
  build_trigger = "push_to_main",
  additional_args = NULL,
  tag_ghactions4r = NULL
)
```

Arguments

- | | |
|-----------------|---|
| workflow_name | What to name the github action workflow locally. Must have the file extension .yml |
| build_trigger | <p>Select the event that triggers the workflow. Options are:</p> <p>push_to_main Run on pushing commits to main (default)</p> <p>weekly Run on the default branch (usually main) once a week</p> <p>manually run manually with the workflow_dispatch trigger</p> <p>Multiple build triggers are allowed; specify them as a vector. Note that invalid build triggers will be silently removed as long as one build trigger is specified correctly.</p> |
| additional_args | <p>A named list of additional command line arguments to be passed to the workflow. The names of the list represent the platforms (windows, macos, or ubuntu), and the values are character vectors of arguments. These additional arguments are executed after the step that checks out the repository and before the step that sets up R.</p> |
| tag_ghactions4r | <p>Tag of ghactions4r workflow to pin to for the reusable workflow. This ensures compatibility with a specific version of ghactions4r functions. The tradeoff is that fixes will not happen automatically; instead, the user will have to update the pinned version of the caller workflow used. See the options by looking at the ghactions4r tags. Another option is to pin to commit hashes, but the user will need to do this by manually editing the caller workflow after creation. See github documentation on security to understand how this is more secure. Again, the tradeoff is that the workflow will not be automatically updated.</p> |

Examples

```
## Not run:
use_update_pkgdown(
  additional_args = list(
    ubuntu = c(
      "sudo apt-get update",
      "sudo apt-get install -y libcurl4-openssl-dev",
      "sudo add-apt-repository ppa:ubuntu-toolchain-r/test",
      "sudo apt-get install --only-upgrade libstdc++6"
    ),
    macos = c("brew install curl")
  )
)
```

use_update_pkgdown

13

)

End(Not run)

Index

`rm_dollar_sign`, [2](#)

`use_build_pkgdown`, [3](#)

`use_calc_cov_summaries`, [4](#)

`use_create_cov_badge`, [5](#)

`use_doc_and_style_r`, [6](#)

`use_r_cmd_check`, [9](#)

`use_spell_check`, [10](#)

`use_update_pkgdown`, [11](#)