

Package: monarch (via r-universe)

September 23, 2026

Type Package

Title Manage rOpenSci Social Contacts

Version 0.1.0

Description Tools to find and keep track of contact info and social media handles of rOpenSci community members.

License GPL (>= 3)

Encoding UTF-8

LazyData true

Depends R (>= 4.2.0)

Imports cli, dplyr, gh (>= 1.6.0), glue, httr2, memoise, purrr, readr, rlang, rtot, stringr, tidyr, usethis, yaml

RoxygenNote 7.3.3

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

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

Repository <https://ropensci.r-universe.dev>

Date/Publication 2026-09-23 20:40:55 UTC

RemoteUrl <https://github.com/ropensci-org/monarch>

RemoteRef main

RemoteSha c5a7d4e53da4fea0fcdb20c0c38d6ce6787a36f7

Contents

add_handles	2
cache_check	4
cocoon_fetch	4
cocoon_open	5
cocoon_remove	5
cocoon_update	6
fetch	7

fmt_masto	7
gh_search	8
local_eg	8
masto2user	9
socials_fetch	9
socials_gh	10
socials_masto	11
socials_ro	12
socials_update	13
Index	14

add_handles	<i>Find and insert missing social media handles</i>
-------------	---

Description

Find and insert missing social media handles

Usage

```
add_handles(  
  df,  
  primary = "github",  
  which_cols = c("github", "name", "mastodon", "bluesky", "linkedin"),  
  pkg_col = "pkg",  
  owner_col = "owner",  
  prefix = "",  
  force = FALSE  
)
```

Arguments

df	Data frame with at least 1 column appended with ‘_name’ or ‘_github’.
primary	Character. Either "github" or "name", which ever column is the primary column by which social handles should be fetched.
which_cols	Character vector. Social columns to return, any of "github", "name", "mastodon", "linkedin", "bluesky".
pkg_col	Character. Name of the "pkg" column. Optional but recommended for fetching by name when ‘primary = "name"’.
owner_col	Character. Name of the column containing repository owners for packages (‘pkg_col’). Optional but recommended for fetching by name when ‘primary = "name"’.
prefix	Character. Optional prefix on column names (e.g. "maintainer_" for "maintainer_github", "maintainer_name", etc.)
force	Logical. Whether to force an update of all handles (not just those missing).

Value

Data frame with added names and social media handles.

Examples

```
d <- data.frame(author_name = "Steffi LaZerte")
add_handles(d, primary = "name", prefix = "author_")

d <- data.frame(
  author_name = "Steffi LaZerte",
  pkg = "weathercan",
  owner = "ropensci",
  author_github = "steffilazerte"
)

add_handles(
  d,
  primary = "name",
  prefix = "author_",
  pkg_col = "pkg",
  owner_col = "owner",
  which_cols = "github"
)

add_handles(
  d,
  primary = "name",
  prefix = "author_",
  pkg_col = "pkg",
  owner_col = "owner"
)

d <- data.frame(github = "steffilazerte")
add_handles(d)

# If all complete, do not overwrite unless force == TRUE
d <- data.frame(
  github = "steffilazerte",
  name = "test",
  mastodon = "test",
  linkedin = "test"
)
add_handles(d)
add_handles(d, force = TRUE)

# Use name for LinkedIn (always)
d <- data.frame(github = "steffilazerte", name = "test", mastodon = "test")
add_handles(d)

d <- data.frame(
  author_name = c("Steffi LaZerte", "Yanina Bellini Saibene"),
  author_github = c("steffilazerte2", NA)
)
```

```
# Keep original github
add_handles(d, primary = "name", prefix = "author_")
# Get stored github
add_handles(d, primary = "name", prefix = "author_", force = TRUE)
```

cache_check

Check Cached Dir

Description

Checks for the presence of a cache dir. If interactive and not present, asks user if it should be created. If non-interactive and not present, it is created automatically.

Usage

```
cache_check()
```

Value

Logical. TRUE if it exists, FALSE if not.

Examples

```
cache_check()
```

cocoon_fetch

Fetch all details on a social contact

Description

Fetch all details on a social contact

Usage

```
cocoon_fetch(value, type = "github")
```

Arguments

value	Character. Value to search by
type	Character. Type of 'value' (e.g., "github" for github handle)

Value

Data frame of socials

Examples

```
cocoon_fetch("steffilazerte")
cocoon_fetch("steffi LaZerte", type = "name")
```

`cocoon_open`*Load socials data from local file*

Description

Reads your local socials data for perusal or modification.

Usage

```
cocoon_open()
```

Value

Socials data frame

Examples

```
cocoon_open()
```

`cocoon_remove`*Remove Socials by GitHub handle*

Description

Remove Socials by GitHub handle

Usage

```
cocoon_remove(github)
```

Arguments

`github` Character. Github username

Value

Socials data frame

Examples

```
cocoon_fetch("steffilazerte")
cocoon_remove("steffilazerte")
cocoon_fetch("steffilazerte")
```

cocoon_update	<i>Save/Update socials data to local file</i>
---------------	---

Description

Save/Update your local copy of the contact information.

Usage

```
cocoon_update(socials, type = NULL, value = NULL)
```

Arguments

socials	Data frame. Data frame of previously fetched/loaded social contact information. Data frame with three columns 'type', 'value' and 'github' to indicate the type and value of the contact information and the GitHub username identifying the individual.
type	Character. Type of 'value' (e.g., "github" for github handle)
value	Character. Value to search by

Value

Nothing

Examples

```
socials_fetch("steffilazerte") |>
  cocoon_update()

# Or directly
cocoon_update("steffilazerte")

# Or add parts
cocoon_update("steffilazerte", type = "name", value = "Stefanie LaZerte")
```

fetch	<i>Fetch a value</i>
-------	----------------------

Description

Fetches a single value from your local socials data frame.

Usage

```
fetch(values, type = "mastodon", n = 1)
```

Arguments

values	Character. Values of social data to match by.
type	Character. Type of social data to fetch.
n	Numeric. Maximum number of matches per value to return.

Value

Character string of the value of the type of data requested

Examples

```
fetch("Steffi LaZerte", type = "linkedin")
fetch(c("Steffi LaZerte", NA, "Yanina Bellini Saibene"))
fetch("steffilazerte")
fetch("Maëlle Salmon", type = "github")
```

fmt_masto	<i>Convert a mastodon user link to handle</i>
-----------	---

Description

Convert a mastodon user link to handle

Usage

```
fmt_masto(x)
```

Arguments

x	Character. Link to user's profile
---	-----------------------------------

Value

Character user handle @user@instance

Examples

```
fmt_masto("https://fosstodon.org/@steffilazerte")
fmt_masto("steffi lazerte")
fmt_masto("@steffilazerte@fosstodon.org")
fmt_masto(NA)
fmt_masto(c("https://fosstodon.org/@steffilazerte", "https://hackyderm.io/@ropensci"))
fmt_masto("none")
```

gh_search

Find GH username from repository and full name

Description

Look up users of a repository and match to a name. Try with and without initials.

Usage

```
gh_search(name, pkg = NULL, owner = "ropensci", open_browser = interactive())
```

Arguments

name	Character. Full or partial name of the person for whom you want to fetch the GitHub username.
pkg	Character. (Optional) Repository name (package name).
owner	Character. (Optional) Owner of the repository.
open_browser	Logical. Whether to open the profile page in a browser for confirmation.

Value

Accepted user name

Examples

```
gh_search(name = "Steffi E. LaZerte", pkg = "weathercan")
gh_search(name = "Steffi", pkg = "weathercan")
```

local_eg

Detect if running examples locally

Description

Skip examples on CI or R-Universe

Usage

```
local_eg()
```

masto2user	<i>Convert a mastodon user link to handle</i>
------------	---

Description

Convert a mastodon user link to handle

Usage

```
masto2user(x)
```

Arguments

x Character. Link to user's profile

Value

Character user handle @user@instance

Examples

```
masto2user("https://fosstodon.org/@steffilazerte")
masto2user("steffi")
masto2user("@steffilazerte@fosstodon.org")
masto2user(NA)
```

socials_fetch	<i>Fetch contact information from social media</i>
---------------	--

Description

This is a wrapper around the three 'socials_XXX()' functions. 'socials_gh()' starts (ideally) with a GitHub username to fetch contact info from GitHub, followed by 'socials_ro()' which checks rOpenSci author pages and finally, if Mastodon handles have not been found 'socials_masto()'.

Usage

```
socials_fetch(
  github = NULL,
  name = NULL,
  pkg = NULL,
  owner = "ropensci",
  which_cols = c("github", "name", "mastodon"),
  force_masto = FALSE,
  quiet = FALSE
)
```

Arguments

github	Character. Github username
name	Character. Alternatively, a full name (can be slow if ‘pkg’ not also provided).
pkg	Character. Repository name for an rOpenSci package (in the ‘ropensci’ organization)
owner	Character. (Optional) Owner of the repository.
which_cols	Character vector. Social handles to include defaults to "github", "name", "mastodon".
force_masto	Logical. Whether to force a re-fetching of Mastodon handles.
quiet	Logical. Whether to suppress progress messages.

Details

This workflow generally assumes that the person has a GitHub username. You can supply name, which will trigger a search for the GitHub username for you. This will search all of the rOpenSci organization for users with that name which can take a while. However, if you supply a name and a ‘pkg’ (repository in rOpenSci), it can be much faster.

Value

Data frame of contact information

Examples

```
socials_fetch("steffilazerte")
socials_fetch(name = "Steffi LaZerte", pkg = "weathercan")
socials_fetch(name = "Bart Vanhoorne", pkg = "worrms")
socials_fetch(name = "Maelle Salmon", pkg = "babelquarto", owner = "ropensci-review-tools")
```

socials_gh

Fetch contact information from GitHub

Description

Using a GitHub username fetch the contact information from their profile.

Usage

```
socials_gh(github, quiet = FALSE)
```

Arguments

github	Character. Github username
quiet	Logical. Whether to suppress progress messages.

Value

Data frame of social contact information. Three columns 'type', 'value' and 'github' to indicate the type and value of the contact information and the GitHub username identifying the individual.

Examples

```
socials_gh("steffilazerte")
```

socials_masto	<i>Fetch Mastodon handle</i>
---------------	------------------------------

Description

Useful if neither GitHub nor the rOpenSci author pages contain a Mastodon handle. Because this can take a bit of time, if a mastodon handle is detected in the 'socials' data frame, 'socials_masto()' will skip the search, unless 'force = TRUE'. Useful if you think the handle is incorrect.

Usage

```
socials_masto(  
  socials = NULL,  
  names = NULL,  
  github = NULL,  
  force = FALSE,  
  open_browser = interactive(),  
  quiet = FALSE  
)
```

Arguments

socials	Data frame. Data frame of previously fetched/loaded social contact information. Data frame with three columns 'type', 'value' and 'github' to indicate the type and value of the contact information and the GitHub username identifying the individual.
names	Character. Names to fetch by.
github	Character. Github username
force	Logical. Whether to force a search even if the handle already exists.
open_browser	Logical. Whether to open the profile page in a browser for confirmation.
quiet	Logical. Whether to suppress progress messages.

Details

Unlike most other functions, 'socials_masto()' permits searching for a handle by name without providing a 'socials' data frame or a 'github' handle. In this case, the Mastodon handle is returned as a character string and not as a socials data frame.

Value

If ‘socials’ or ‘github’ are provided, a data frame of social contact information. Three columns ‘type’, ‘value’ and ‘github’ to indicate the type and value of the contact information and the GitHub username identifying the individual. Otherwise a character string of the mastodon handle.

Examples

```
socials_gh("steffilazerte") |>
  socials_masto(force = TRUE)

socials_masto(github = "steffilazerte")
socials_masto(names = "steffi")

socials_masto(names = "steffi", github = "steffilazerte")
```

socials_ro	<i>Fetch contact information from rOpenSci author pages</i>
------------	---

Description

Once initial contact information has been sourced via ‘socials_gh()’, we can add to this by looking for additional details on the rOpenSci author pages (<<https://ropensci.org/author>>).

Usage

```
socials_ro(socials = NULL, names = NULL, github = NULL, quiet = FALSE)
```

Arguments

socials	Data frame. Data frame of previously fetched/loaded social contact information. Data frame with three columns ‘type’, ‘value’ and ‘github’ to indicate the type and value of the contact information and the GitHub username identifying the individual.
names	Character. Optional, supply the full name as on RO author pages. (Doesn’t require the socials data frame)
github	Character. Github username
quiet	Logical. Whether to suppress progress messages.

Details

By default ‘socials_ro()’ will use the ‘name’ stored in the ‘socials’ data frame.

If you know this doesn’t match exactly that on the rOpenSci author pages, you can supply an alternative with ‘name’. Ideally however, you would add an additional name to the socials data frame to keep track of minor variations in names.

Value

Data frame of social contact information. Three columns 'type', 'value' and 'github' to indicate the type and value of the contact information and the GitHub username identifying the individual.

Examples

```
socials_gh("steffilazerte") |>
  socials_ro()

socials_ro(names = c("Stefanie LaZerte", "Steffi LaZerte"))
```

socials_update	<i>Update a socials data frame with new values</i>
----------------	--

Description

'socials_update()' takes an existing social data frame and adds or updates new values. Only certain data types are permitted to have more than one value (i.e. names and websites). In all other cases, if there is more than one handle or id value, you will be asked to choose one to keep.

Usage

```
socials_update(socials, socials_new = NULL, type = NULL, value = NULL)
```

Arguments

socials	Data frame. Data frame of previously fetched/loaded social contact information. Data frame with three columns 'type', 'value' and 'github' to indicate the type and value of the contact information and the GitHub username identifying the individual.
socials_new	Data frame. New data to add/update. If adding only a single value, use 'type' and 'value' instead.
type	Character. Type of data to add (to add a single value, alternative to 'socials_new')
value	Character. Value of data to add (to add a single value, alternative to 'socials_new').

Value

Socials data frame with new values

Examples

```
socials_fetch("steffilazerte") |>
  socials_update(type = "name", value = "Stefanie LaZerte") # Add an alias
```

Index

`add_handles`, [2](#)

`cache_check`, [4](#)

`cocoon_fetch`, [4](#)

`cocoon_open`, [5](#)

`cocoon_remove`, [5](#)

`cocoon_update`, [6](#)

`fetch`, [7](#)

`fmt_masto`, [7](#)

`gh_search`, [8](#)

`local_eg`, [8](#)

`masto2user`, [9](#)

`socials_fetch`, [9](#)

`socials_gh`, [10](#)

`socials_masto`, [11](#)

`socials_ro`, [12](#)

`socials_update`, [13](#)