Package ‘plume’

October 6, 2023

Type  Package
Title  A Simple Author Handler for Scientific Writing
Version  0.1.0
Description  Handles and formats author information in scientific writing in ‘R Markdown’ and ‘Quarto’. ‘plume’ provides easy-to-use and flexible tools for injecting author metadata in ‘YAML’ headers as well as generating author and contribution lists (among others) as strings from tabular data.
License  GPL (>= 3)
BugReports  https://github.com/arnaudgallou/plume/issues
Depends  R (>= 4.1.0)
Imports  dplyr (>= 1.0.0), glue (>= 1.3.2), jsonlite (>= 1.6.0), knitr (>= 1.40), purrr (>= 1.0.0), R6, readr (>= 1.0.0), rlang (>= 1.0.0), stringr, tibble (>= 3.0.0), tidyr (>= 1.1.0), tidyselect (>= 1.0.0), vctrs (>= 0.3.0), yaml (>= 2.3.5)
Suggests  covr, fontawesome, gt, rmarkdown, testthat, waldo (>= 0.3.0), withr
VignetteBuilder  knitr
Config/Needs/website  arnaudgallou/cygne
Config/testthat/edition  3
Encoding  UTF-8
Language  en-GB
LazyData  true
RoxygenNote  7.2.3
NeedsCompilation  no
Author  Arnaud Gallou [aut, cre, cph] (<https://orcid.org/0000-0002-1002-4247>), Hadley Wickham [cph] (Author of included stringb functions.)
encyclopedists

Maintainer Arnaud Gallou <arangacas@gmail.com>
Repository CRAN
Date/Publication 2023-10-06 08:40:02 UTC

**R topics documented:**

encyclopedists ......................................................... 2  
enumerate .............................................................. 3  
everyone ............................................................... 4  
NameHandler ........................................................... 4  
orcid ................................................................. 5  
plm_template ........................................................... 6  
Plume ................................................................. 6  
PlumeHandler ......................................................... 10  
PlumeQuarto ........................................................... 11  
sequential .............................................................. 13  
set_default_names .................................................... 14  
StatusSetter .......................................................... 15  
StatusSetterQuarto ................................................... 16

Index 18

<table>
<thead>
<tr>
<th>encyclopedists</th>
<th>Famous encyclopedists</th>
</tr>
</thead>
</table>

**Description**

Data on four famous authors of the Encyclopédie (originally "Encyclopédie, ou dictionnaire raisonné des sciences, des arts et des métiers") published in France in the second half of the 18th century. The data set is available in English (encyclopedists) and French (encyclopedists_fr).

**Usage**

encyclopedists

encyclopedists_fr

**Format**

A tibble with 4 rows and 12 variables:

given_name, prénom authors’ given names  
family_name, nom authors’ family names  
email, courriel authors’ email addresses  
phone, téléphone authors’ phone numbers  
orcid authors’ ORCID
**enumerate**

affiliation authors’ affiliations
role_n authors’ roles written as nouns
role_v authors’ roles written as active verbs
note special notes about authors

**Examples**

encyclopedists
encyclopedists_fr

---

| enumerate | **Enumerate vector elements** |

**Description**

Wrapper around `glueCollapse()` using `sep = ", "` and `last = " and "` as default arguments.

**Usage**

```r
enumerate(x, sep = ", ", last = " and ")
```

**Arguments**

- `x` A character vector.
- `sep` Separator used to separate the terms.
- `last` Separator used to separate the last two items if `x` has at least 2 items.

**Value**

A character string with the same class as `x`.

**Examples**

```r
aut <- Plume$new(encyclopedists)
aut$get_author_list(format = "^a^") |> enumerate()
```
everyone

Select all authors or exclude some from a selection

Description

Selection helpers to use in conjunction with set_*() methods:

- **everyone()** select all authors.
- **everyone_but()** select all but specific authors.

Usage

everyone()

everyone_but(...)

Arguments

... One or more unquoted expressions separated by commas. Expressions matching values in the column defined by the by parameter of set_*() methods are used to set a given status to authors. Matching of values is case-insensitive.

Examples

aut <- Plume$new(encyclopedists)

aut$set_corresponding_authors(everyone())
aut$get_plume() |> dplyr::select(1:3, corresponding)

aut$set_corresponding_authors(
  everyone_but(jean),
  by = "given_name"
)
aut$get_plume() |> dplyr::select(1:3, corresponding)

NameHandler

**NameHandler class**

Description

Internal class that handles the names of a plume object.
Methods

Public methods:

• NameHandler$new()
• NameHandler$clone()

Method new():

Usage:
NameHandler$new(names)

Method clone(): The objects of this class are cloneable with this method.

Usage:
NameHandler$clone(deep = FALSE)

Arguments:
dee p Whether to make a deep clone.

Description

Helper function to control the size and colour of the ORCID icon.

Usage

orcid(size = 16, bw = FALSE)

Arguments

size Size (in pixels) of the icon.
bw Should the black and white version of the icon be used?

Value

A plume icon.

Examples

aut <- Plume$new(encyclopedists, orcid_icon = orcid(bw = TRUE))
plm_template

Create a table template for plume classes

Description

This helper function allows you to generate an empty tibble that you can use as a template to supply author data.

Usage

plm_template(minimal = TRUE, credit_roles = FALSE)

Arguments

minimal

If TRUE, returns an empty tibble with the following columns: given_name, family_name, email, orcid, affiliation, role and note. Otherwise the function returns a template with all columns that can be supplied to plume classes that are not PlumeQuarto-specific.

credit_roles

Should the Contributor Roles Taxonomy be used?

Value

An empty tibble.

Examples

plm_template()

Plume

Plume class

Description

Class that generates author lists and other author-related information as character strings.

Super classes

plume::NameHandler -> plume::PlumeHandler -> plume::StatusSetter -> Plume
Methods

Public methods:

- `Plume$new()`
- `Plume$get_author_list()`
- `Plume$get_affiliations()`
- `Plume$get_notes()`
- `Plume$get_orcids()`
- `Plume$get_contact_details()`
- `Plume$get_contributions()`
- `Plume$clone()`

**Method** `new()`: Create a Plume object.

**Usage:**

```r
Plume$new(
  data,
  names = NULL,
  symbols = NULL,
  credit_roles = FALSE,
  initials_given_name = FALSE,
  family_name_first = FALSE,
  interword_spacing = TRUE,
  orcid_icon = orcid()
)
```

**Arguments:**

data A data frame containing author-related data.

names A vector of key-value pairs specifying custom names to use, where keys are default names and values their respective replacements.

symbols A list of key-value pairs defining the symbols to use to link authors and their metadata. Valid keys are "affiliation", "corresponding" and "note". By default, uses digits for affiliations, "*" for corresponding authors and "†", "‡", "§", "¶", "#" for notes. Set a key to `NULL` to use numerals.

credit_roles Should the Contributor Roles Taxonomy be used? See vignette("using-credit-roles") for details.

initials_given_name Should the initials of given names be used?

family_name_first Should literal names show family names first?

interword_spacing Should literal names use spacing? This parameter is only useful for people writing in languages that don’t separate words with a space such as Chinese or Japanese.

orcid_icon The ORCID icon, as defined by `orcid()`, to be used.

**Returns:** A Plume object.

**Method** `get_author_list()`: Get author list.

**Usage:**

```r
Plume$get_author_list(format = NULL)
```

**Arguments:**

format
format: A character string defining the format of symbols suffixing author names. See details.

**Details:** format lets you choose which symbol categories to suffix authors with, using the following keys:
- `a` for affiliations
- `c` for corresponding authors
- `n` for notes
- `o` for ORCIDs

The order of the keys determines the order of symbol types. E.g. "ac" shows affiliation ids first and corresponding author mark second, when "ca" shows corresponding author mark first and affiliation ids second. Use "," to separate and "^" to superscript symbols. Use NULL or an empty string to list author names without suffixes.

**Returns:** A character vector.

**Method** `get_affiliations()`: Get authors’ affiliations.

**Usage:**
```r
Plume$get_affiliations(sep = "", superscript = TRUE)
```

**Arguments:**
- `sep` Separator used to separate affiliation ids and affiliations.
- `superscript` Should affiliation ids be superscripted?

**Returns:** A character vector.

**Method** `get_notes()`: Get authors’ notes.

**Usage:**
```r
Plume$get_notes(sep = "", superscript = TRUE)
```

**Arguments:**
- `sep` Separator used to separate note ids and notes.
- `superscript` Should note ids be superscripted?

**Returns:** A character vector.

**Method** `get_orcids()`: Get authors’ ORCID.

**Usage:**
```r
Plume$get_orcids(compact = FALSE, icon = TRUE, sep = "")
```

**Arguments:**
- `compact` Should links only display the 16-digit identifier?
- `icon` Should the ORCID icon be shown?
- `sep` Separator used to separate authors and their respective ORCID.

**Returns:** A character vector.

**Method** `get_contact_details()`: Get the contact details of corresponding authors.

**Usage:**
Plume$get_contact_details(
    format = "\{(details) \{(name)\}\}",
    email = TRUE,
    phone = FALSE,
    fax = FALSE,
    url = FALSE,
    sep = ", "
)

Arguments:
format A glue specification that uses the variables name and/or details.
email, phone, fax, url Arguments equal to TRUE are evaluated and passed to the variable
details. By default, only email is set to TRUE.
sep Separator used to separate details items.

Returns: A character vector.

Method get_contributions(): Get authors' contributions.

Usage:
Plume$get_contributions(
    roles_first = TRUE,
    by_author = FALSE,
    alphabetical_order = FALSE,
    dotted_initials = TRUE,
    literal_names = FALSE,
    divider = ": ",
    sep_last = " and "
)

Arguments:
roles_first If TRUE, displays roles first and authors second. If FALSE, roles follow authors.
by_author Should roles be grouped by author?
alphabetical_order Should authors be listed in alphabetical order? By default, lists authors
in the order they are defined in the data.
dotted_initials Should initials be dot-separated?
literal_names Should literal names be used?
divider Separator used to separate roles and authors. Uses ": " by default.
sep_last Separator used to separate the last two roles or authors if more than one item is
associated to a role or author.

Returns: A character vector.

Method clone(): The objects of this class are cloneable with this method.

Usage:
Plume$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.
PlumeHandler

Examples

# Create a Plume instance
aut <- Plume$new(encyclopedists, names = c(role = "role_n"))

# Set the desired corresponding authors, using
# authors' id by default
aut$set_corresponding_authors(1, 4)

# Getting authors suffixed by affiliation ids
# and the corresponding author mark:
aut$get_author_list("^a,c^")

# Or maybe with the corresponding author mark
# coming before affiliation ids:
aut$get_author_list("^c,a^")

# Getting more author metadata
aut$get_affiliations()

aut$get_contributions()

# Use `symbols` to change the default symbols.
# E.g. to use letters as affiliation ids:
aut <- Plume$new(encyclopedists,
symbols = list(affiliation = letters))

aut$get_author_list("^a^")

aut$get_affiliations()

# It is also possible to output contributions in the
# active voice
aut <- Plume$new(encyclopedists, names = c(role = "role_v"))
aut$get_contributions(roles_first = FALSE, divider = " ")

---

PlumeHandler

PlumeHandler class

Description

Internal class processing and shaping tabular data into a plume object.

Super class

plume::NameHandler -> PlumeHandler
Methods

Public methods:

- PlumeHandler$new()
- PlumeHandler$print()
- PlumeHandler$get_plume()
- PlumeHandler$clone()

Method new():

Usage:
PlumeHandler$new(
  data,
  names,
  credit_roles,
  initials_given_name,
  family_name_first = FALSE,
  interword_spacing = TRUE
)

Method print():

Usage:
PlumeHandler$print()

Method get_plume():

Usage:
PlumeHandler$get_plume()

Method clone(): The objects of this class are cloneable with this method.

Usage:
PlumeHandler$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.

---

**PlumeQuarto**

**PlumeQuarto class**

Description

Class that pushes or updates author metadata in the YAML header of Quarto files.

Super classes

plume::NameHandler -> plume::PlumeHandler -> plume::StatusSetter -> plume::StatusSetterQuarto -> PlumeQuarto
Methods

Public methods:

- `PlumeQuarto$new()`
- `PlumeQuarto$to_yaml()`
- `PlumeQuarto$clone()`

**Method** `new()`: Create a `PlumeQuarto` object.

**Usage:**

```r
PlumeQuarto$new(
  data,
  file,
  names = NULL,
  credit_roles = FALSE,
  initials_given_name = FALSE,
  by = NULL
)
```

**Arguments:**

- `data` A data frame containing author-related data.
- `file` A `.qmd` file to insert author data into.
- `names` A vector of key-value pairs specifying custom names to use, where keys are default names and values their respective replacements.
- `credit_roles` Should the Contributor Roles Taxonomy be used? See vignette("using-credit-roles") for details.
- `initials_given_name` Should the initials of given names be used?
- `by` A character string defining the default variable used to assign authors' status in all set_*() methods. By default, uses authors' id.

**Returns:** A `PlumeQuarto` object.

**Method** `to_yaml()`: Push or update author information in a YAML header. The generated YAML complies with Quarto's `author` and `affiliations` schemas.

**Usage:**

```r
PlumeQuarto$to_yaml()
```

**Details:** If missing, `to_yaml()` pushes author information into a YAML header. If already existing, the function replaces old author and affiliations values with the ones provided in the input data.

**Returns:** The input file invisibly.

**Method** `clone()`: The objects of this class are cloneable with this method.

**Usage:**

```r
PlumeQuarto$clone(deep = FALSE)
```

**Arguments:**

- `deep` Whether to make a deep clone.
### Examples

```r
# Create a simple temporary file with a YAML header
# containing a title
tmp_file <- withr::local_tempfile(
  lines = "---
title: Encyclopédie
---",
  fileext = "qmd"
)

# View the temporary file
cat(readr::read_file(tmp_file))

# Create a PlumeQuarto instance using the temporary file
# you've just created
aut <- PlumeQuarto$new(
  encyclopedists,
  file = tmp_file,
  names = c(role = "role_n")
)

# And push author data to the YAML header
aut$to_yml()

cat(readr::read_file(tmp_file))

# Pushing again with new data updates the YAML
# header accordingly
aut <- PlumeQuarto$new(
  dplyr::slice(encyclopedists, 2),
  file = tmp_file,
  names = c(role = "role_n")
)
aut$to_yml()

cat(readr::read_file(tmp_file))
```

---

**sequential**  
*Control the sequencing behaviour of character vectors*

---

**Description**

Modifier function used to generate logical sequences of characters.

**Usage**

`sequential(x)`
Arguments

x  A character vector.

Value

A character vector with parent S3 class sequential.

Examples

```r
aut <- Plume$new(
  tibble::tibble(
    given_name = "X",
    family_name = "Y",
    affiliation = 1:60
  ),
  symbols = list(affiliation = sequential(letters))
)

aut$get_affiliations(sep = " ", superscript = FALSE) |> 
cat(sep = "\n")
```

---

**set_default_names**  
Set new default names to a plume subclass

Description

This helper function allows you to set new default names to a plume subclass, e.g. to set default names to a language other than English.

Usage

```r
set_default_names(..., .plume_quarto = FALSE)
```

Arguments

...  Key-value pairs where keys are default names and values their respective replacements.

.plume_quarto  Are you setting new names for PlumeQuarto?

Details

Available names are:

- id, initials, literal_name, corresponding, given_name, family_name, email, phone, fax, url, affiliation, role, note, orcid.

Using .plume_quarto = TRUE adds deceased, equal_contributor, number, dropping_particle and acknowledgements.
**Status Setter**

**Value**

A named list.

**Examples**

```
# Extending `Plume` with default names in French
PlumeFr <- R6::R6Class(
  classname = "PlumeFr",
  inherit = Plume,
  private = list(
    plume_names = set_default_names(
      initials = "initiales",
      literal_name = "nom_complet",
      corresponding = "correspondant",
      given_name = "prénom",
      family_name = "nom",
      email = "courriel",
      phone = "téléphone"
    )
  )
)

PlumeFr$new(encyclopedists_fr)
```

---

**StatusSetter**

**Status Setter class**

**Description**

Internal class that manages authors' status.

**Super classes**

`plume::NameHandler` -> `plume::PlumeHandler` -> `StatusSetter`

**Methods**

**Public methods:**

- `StatusSetter$new()`
- `StatusSetter$set_corresponding_authors()`
- `StatusSetter$clone()`

**Method new():**

*Usage:*

`StatusSetter$new(...)`

**Method set_corresponding_authors():** Set corresponding authors.
**StatusSetterQuarto**

*Usage:*

StatusSetter$set_corresponding_authors(..., by)

*Arguments:*

... One or more unquoted expressions separated by commas. Expressions matching values in the column defined by by determine corresponding authors. Matching of values is case-insensitive.

by Variable used to set corresponding authors. By default, uses authors’ id.

*Returns:* The class instance.

**Method clone():** The objects of this class are cloneable with this method.

*Usage:*

StatusSetter$clone(deep = FALSE)

*Arguments:*

depth Whether to make a deep clone.

---

**StatusSetterQuarto StatusSetterQuarto class**

**Description**

Internal class extending StatusSetter for PlumeQuarto.

**Super classes**

plume::NameHandler -> plume::PlumeHandler -> plume::StatusSetter -> StatusSetterQuarto

**Methods**

**Public methods:**

- StatusSetterQuarto$new()
- StatusSetterQuarto$set_equal_contributor()
- StatusSetterQuarto$set_deceased()
- StatusSetterQuarto$clone()

**Method new():**

*Usage:*

StatusSetterQuarto$new(..., by)

**Method set_equal_contributor():** Set equal contributors.

*Usage:*

StatusSetterQuarto$set_equal_contributor(..., by)

*Arguments:*

... One or more unquoted expressions separated by commas. Expressions matching values in the column defined by by determine equal contributors. Matching of values is case-insensitive.

by Variable used to specify which authors are equal contributors. By default, uses authors’ id.

Returns: The class instance.

Method set_deceased(): Set deceased authors.

Usage:
StatusSetterQuarto$set_deceased(..., by)

Arguments:
... One or more unquoted expressions separated by commas. Expressions matching values in the column defined by by determine deceased authors. Matching of values is case-insensitive.

by Variable used to specify whether an author is deceased or not. By default, uses authors’ id.

Returns: The class instance.

Method clone(): The objects of this class are cloneable with this method.

Usage:
StatusSetterQuarto$clone(deep = FALSE)

Arguments:

depth Whether to make a deep clone.
Index

* datasets
  encyclopedists, 2

encyclopedists, 2
encyclopedists_fr (encyclopedists), 2
enumerate, 3
everyone, 4
everyone(), 4
everyone_but (everyone), 4
everyone_but(), 4

glue, 9
glue_collapse(), 3

NameHandler, 4

orcid, 5
orcid(), 7

plm_template, 6
Plume, 6
plume::NameHandler, 6, 10, 11, 15, 16
plume::PlumeHandler, 6, 11, 15, 16
plume::StatusSetter, 6, 11, 16
plume::StatusSetterQuarto, 11
PlumeHandler, 10
PlumeQuarto, 11, 14

sequential, 13
set_default_names, 14
StatusSetter, 15
StatusSetterQuarto, 16

tibble, 6