Package ‘plume’

December 9, 2023

Type Package
Title A Simple Author Handler for Scientific Writing
Version 0.2.1
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)
URL https://github.com/arnaudgallou/plume,
https://arnaudgallou.github.io/plume/
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), lifecycle (>= 1.0.3), purrr (>= 1.0.0), R6, readr (>= 1.0.0), rlang (>= 1.0.0), stringr, tibble (>= 3.0.0), tidy (>= 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>
Description

Helper function returning the 14 contributor roles of the Contributor Roles Taxonomy (CRedit). This function is the default argument of the roles and role_cols parameters in plume classes and plm_template(), respectively.

Usage

credit_roles(oxford_spelling = TRUE)

Arguments

oxford_spelling

Should the suffix -ize/-ization be used?

Value

A named vector.

Examples

credit_roles()
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 10 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
- **affiliation_1,affiliation_2** authors’ affiliations
- **supervision** authors that supervised the project
- **writing,rédaction** authors involved in the writing
- **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
`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$new(encycopedists)
aut$get_author_list() |> enumerate()
```

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:**
deep Whether to make a deep clone.

---

**orcid**

**ORCID icon**

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

**Usage**

```r
orcid(size = 16, bw = FALSE)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>size</td>
<td>Size (in pixels) of the icon.</td>
</tr>
<tr>
<td>bw</td>
<td>Should the black and white version of the icon be used?</td>
</tr>
</tbody>
</table>

**Value**

A plume icon.

**Examples**

```r
aut <- Plume$new(encycopedists, 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**

```r
plm_template(minimal = TRUE, role_cols = credit_roles(), credit_roles = FALSE)
```
Arguments

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

role_cols A vector of names defining role columns to create. If the vector contains key-value pairs, columns will be named after the keys.

credit_roles [Deprecated]
It is now recommended to use role_cols = credit_roles() to use the Contributor Roles Taxonomy.

Value
An empty tibble.

Examples

plm_template()

plm_template(role_cols = paste0("role_", 1:5))

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::StatusSetterPlume -> 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:
Plume$new(
  data,
  names = NULL,
  symbols = NULL,
  roles = credit_roles(),
  credit_roles = FALSE,
  initials_given_name = FALSE,
  family_name_first = FALSE,
  interword_spacing = TRUE,
  orcid_icon = orcid(),
  by = NULL
)

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.
  roles  A vector of key-value pairs defining roles where keys identify role columns and values
         describe the actual roles to use.
  credit_roles [Deprecated]
            It is now recommended to use roles = credit_roles() to use the Contributor Roles Tax-
            onomy.
  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.
  by  A character string defining the default variable used to assign specific metadata to authors
       in all set_() methods. By default, uses authors’ id.

Returns:  A Plume object.

Method get_author_list(): Get author list.

Usage:
Plume$get_author_list(suffix = NULL, format = deprecated())

Arguments:

  suffix  A character string defining the format of symbols suffixing author names. See details.
  format [Deprecated]
            Please use the parameter suffix instead.

Details:  suffix 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:
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:
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:
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:**

```r
Plume$get_contributions(
  roles_first = TRUE,
  by_author = FALSE,
  alphabetical_order = FALSE,
  dotted_initials = TRUE,
  literal_names = FALSE,
  divider = " ": ",",
  sep = ", ",
  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 Separator used to separate roles or authors.
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:**

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

**Arguments:**

deep Whether to make a deep clone.

**Examples**

```r
# Create a Plume instance
aut <- Plume$new(encyclopedists)

# Set the desired corresponding authors, using authors’ id by default
```
aut$set_corresponding_author(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$new(encyclopedists, roles = c(
  supervision = "supervised the project",
  writing = "contributed to the Encyclopédie"
))
aut$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$\text{get\_plume()}
• PlumeHandler$\text{get\_roles()}
• PlumeHandler$\text{clone()}

**Method new():**
*Usage:*
PlumeHandler$new(
  data,
  names,
  roles,
  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 get\_roles():**
*Usage:*
PlumeHandler$get\_roles()

**Method clone():** The objects of this class are cloneable with this method.
*Usage:*
PlumeHandler$\text{clone}(\text{deep = FALSE})

*Arguments:*
deepl Whether to make a deep clone.

---

<table>
<thead>
<tr>
<th>PlumeQuarto</th>
<th>PlumeQuarto class</th>
</tr>
</thead>
</table>

**Description**

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

**Super classes**

plume::NameHandler -> plume::PlumeHandler -> plume::StatusSetter -> plume::StatusSetterPlumeQuarto -> 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,
  roles = credit_roles(),
  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.

roles A vector of key-value pairs defining roles where keys identify columns and values describe the actual roles to use.

credit_roles [Deprecated]

It is now recommended to use `roles = credit_roles()` to use the Contributor Roles Taxonomy.

initials_given_name Should the initials of given names be used?

by A character string defining the default variable used to assign specific metadata to authors 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.
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.
set_default_names

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)
```

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, contributor_rank, 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.

Value

A named list.
Examples

```r
# 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

**StatusSetter class**

**Description**

Internal class that manages authors’ status.

**Super classes**

`plume::NameHandler` \rightarrow `plume::PlumeHandler` \rightarrow `StatusSetter`

**Methods**

**Public methods:**

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

**Method `new()`:**

*Usage:*

`StatusSetter$new(..., by)`

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

*Usage:*

`StatusSetter$set_corresponding_authors(..., .by, by = deprecated())`

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

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

[Deprecated]
Please use the .by parameter instead.

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.

Description
Internal class extending StatusSetter for Plume.

Super classes
plume::NameHandler -> plume::PlumeHandler -> plume::StatusSetter -> StatusSetterPlume

Methods

Public methods:
• StatusSetterPlume$set_main_contributors()
• StatusSetterPlume$clone()

Method set_main_contributors(): Force one or more contributors’ names to appear first in the contribution list.

Usage:
StatusSetterPlume$set_main_contributors(..., .roles = NULL, .by)

Arguments:
... One or more unquoted expressions separated by commas. Expressions matching values in the column defined by by/.by determine main contributors. Expressions can be named after any role to set different main contributors to different roles at once, in which case the .roles parameter is ignored. Matching of values is case-insensitive.

.roles Roles to assign main contributors to. If .roles is a named vector, only the names will be used.

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

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

Usage:
StatusSetterPlumeQuarto$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.

---

StatusSetterPlumeQuarto

StatusSetterPlumeQuarto class

Description

Internal class extending StatusSetter for PlumeQuarto.

Super classes

plume::NameHandler -> plume::PlumeHandler -> plume::StatusSetter -> StatusSetterPlumeQuarto

Methods

Public methods:

• StatusSetterPlumeQuarto$set_cofirst_authors()
• StatusSetterPlumeQuarto$set_equal_contributor()
• StatusSetterPlumeQuarto$set_deceased()
• StatusSetterPlumeQuarto$clone()

Method set_cofirst_authors(): Set co-first authors.

Usage:
StatusSetterPlumeQuarto$set_cofirst_authors(..., .by)

Arguments:

... One or more unquoted expressions separated by commas. Expressions matching values in the column defined by by/.by determine co-first authors. Matching of values is case-insensitive.
.by Variable used to specify which authors contributed equally to the work. By default, uses authors’ id.

Returns: The class instance.

Method set_equal_contributor(): [Deprecated]

This method has been deprecated in favour of set_cofirst_authors().

Usage:
StatusSetterPlumeQuarto$set_equal_contributor(..., .by, by = deprecated())

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

\texttt{by} Variable used to specify which authors are equal contributors. By default, uses authors’ id. \texttt{by} \textbf{[Deprecated]} Please use the \texttt{.by} parameter instead.

\textit{Returns:} The class instance.

\textbf{Method} \texttt{set_deceased()}: Set deceased authors.

\textit{Usage:}
\texttt{StatusSetterPlumeQuarto\$set_deceased(..., .by, by = deprecated())}

\textit{Arguments:}

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

\texttt{by} Variable used to specify whether an author is deceased or not. By default, uses authors’ id. \texttt{by} \textbf{[Deprecated]} Please use the \texttt{.by} parameter instead.

\textit{Returns:} The class instance.

\textbf{Method} \texttt{clone()}: The objects of this class are cloneable with this method.

\textit{Usage:}
\texttt{StatusSetterPlumeQuarto\$clone(deep = FALSE)}

\textit{Arguments:}

\texttt{deep} Whether to make a deep clone.
Index

* datasets
  - encyclopedists, 3

credit_roles, 2

encyclopedists, 3
encyclopedists_fr (encyclopedists), 3
enumerate, 4

 glue, 9
 glue_collapse(), 4

NameHandler, 4

orcid, 5
orcid(), 7

plm_template, 5
plm_template(), 2
Plume, 6
plume::NameHandler, 6, 10, 11, 15–17
plume::PlumeHandler, 6, 11, 15–17
plume::StatusSetter, 6, 11, 16, 17
plume::StatusSetterPlume, 6
plume::StatusSetterPlumeQuarto, 11
PlumeHandler, 10
PlumeQuarto, 11, 14

sequential, 13
set_default_names, 14
StatusSetter, 15
StatusSetterPlume, 16
StatusSetterPlumeQuarto, 17

tibble, 5