Package ‘plu’

June 9, 2020

Type Package
Title Pluralize Phrases
Version 0.1.0
Description Converts English phrases to singular or plural form based on the length of an associated vector. Contains helper functions to create natural language lists from vectors and to include the length of a vector in natural language.
License MIT + file LICENSE
URL https://github.com/rossellhayes/plu
BugReports https://github.com/rossellhayes/plu/issues
Depends R (>= 2.10)
Suggests covr, nombre, testthat
Encoding UTF-8
Language en-US
LazyData true
RoxygenNote 7.1.0
NeedsCompilation no
Author Alexander Rossell Hayes [aut, cre, cph], Kevin Atkinson [dct]
Maintainer Alexander Rossell Hayes <alexander@rossellhayes.com>
Repository CRAN
Date/Publication 2020-06-09 10:50:07 UTC

R topics documented:

plu_ral ................................................................. 2
plu_ralize ............................................................ 4
plu_stick .............................................................. 5

Index 8
**plu_ral**  
_Pluralize a phrase based on the length of a vector_

**Description**

Pluralize a phrase based on the length of a vector

**Usage**

```r
call = plu_ral(x, vector = integer(2), n_fn = NULL, ...,
               n = length(vector), pl = abs(n) != 1,
               irregulars = c("moderate", "conservative", "liberal", "none", "easter"),
               replace_n = TRUE)
```

```r
call = ral(x, vector = integer(2), n_fn = NULL, ...,
           n = length(vector), pl = abs(n) != 1,
           irregulars = c("moderate", "conservative", "liberal", "none", "easter"),
           replace_n = TRUE)
```

**Arguments**

- `x`  
  An English word or phrase to be pluralized. See details for special sequences which are handled differently.

- `vector`  
  A vector whose length determines `n`. Defaults to length 2.

- `n_fn`  
  A function to apply to the output of the special sequence "n". See examples. Defaults to `identity`, which returns `n` unchanged.

- `...`  
  Additional arguments passed to the function `n_fn`.

- `n`  
  The number which will determine the plurality of `x`. Defaults to `length(n)`. If specified, overrides `vector`.

- `pl`  
  A logical value indicating whether to use the plural form (if `TRUE`) or the singular form (if `FALSE`) of `x`. Defaults to `FALSE` when `n` is 1 or -1 and `TRUE` for all other values. If specified, overrides `n`. 
irregulars  What level of irregularity to use in pluralization. "moderate" uses the most common pluralization. "conservative" uses the most common irregular plural if one exists, even if a regular plural is more common. "liberal" uses a regular plural if it exists, even if an irregular plural is more common. "none" attempts to apply regular noun pluralization rules to all words. Defaults to "moderate". The default can be changed by setting options(plu.irregulars). See examples in plu::ralize() for more details.

replace_n  A logical indicating whether to use special handling for "n". See details. Defaults to TRUE.

Details

Certain strings in x are treated specially.

- By default, "a" and "an" are deleted in the plural ("a word" to "words").
- The string "n" will be replaced with the length of vector or the number in n.
  - This output can be modified with n_fn.
- Strings between braces separated by a pipe will be treated as a custom plural ("{a|some} word" to "a word", "some words").
  - Three strings separated by pipes will be treated as a singular, dual, and plural form ("{the|both|all} word" to "the word" (1), "both words" (2), "all words" (3+)).
- Any other string between braces will be treated as invariant ("attorney {general}" to "attorneys general").

Value

The character vector x altered to match the number of n

See Also

plu::ralize() to convert an English word to its plural form.

Examples

plu::ral("apple", pl = FALSE)
plu::ral("apple", pl = TRUE)

plu::ral("apple", n = 1)
plu::ral("apple", n = 2)
plu::ral("apple", n = 0)
plu::ral("apple", n = -1)
plu::ral("apple", n = 0.5)

mon <- c("apple")
tue <- c("pear", "pear")

plu::ral("apple", mon)
plu::ral("pear", tue)
plu_ralize  
Pluralize a word

Description
Pluralize a word

Usage
plu_ralize(
  x,
  irregulars = getOption("plu.irregulars", c("moderate", "conservative", "liberal", "none", "easter"))
)
ralize(
  x,
  irregulars = getOption("plu.irregulars", c("moderate", "conservative", "liberal", "none", "easter"))
)

Arguments
x  A character vector of English words to be pluralized
irregulars: What level of irregularity to use in pluralization. "moderate" uses the most common pluralization. "conservative" uses the most common irregular plural if one exists, even if a regular plural is more common. "liberal" uses a regular plural if it exists, even if an irregular plural is more common. "none" attempts to apply regular noun pluralization rules to all words. Defaults to "moderate". The default can be changed by setting options(plu.irregulars). See examples.

Value

The character vector x pluralized

Source

Irregular plurals list adapted from Automatically Generated Inflection Database (AGID)
Copyright 2000-2014 by Kevin Atkinson

Permission to use, copy, modify, distribute and sell this database, the associated scripts, the output created from the scripts and its documentation for any purpose is hereby granted without fee, provided that the above copyright notice appears in all copies and that both that copyright notice and this permission notice appear in supporting documentation. Kevin Atkinson makes no representations about the suitability of this array for any purpose. It is provided "as is" without express or implied warranty.

See Also

plu::ral() to pluralize an English phrase based on a condition

Examples

plu::ralize("word")
plu::ralize(c("group", "word"))

plu::ralize(c("formula", "person", "child"), irregulars = "conservative")
plu::ralize(c("formula", "person", "child"), irregulars = "moderate")
plu::ralize(c("formula", "person", "child"), irregulars = "liberal")
plu::ralize(c("formula", "person", "child"), irregulars = "none")
Usage

\[
\text{plu\_stick}(\ \\
\quad x, \ \\
\quad \text{fn} = \text{NULL}, \ \\
\quad \ldots, \ \\
\quad \text{max} = \text{Inf}, \ \\
\quad \text{fn\_overflow} = \text{FALSE}, \ \\
\quad \text{sep} = ", ", \ \\
\quad \text{conj} = " \ \text{and} \ \", \ \\
\quad \text{syndeton} = \text{c("last", \ "all", \ "none")}, \ \\
\quad \text{oxford} = \text{getOption("plu.oxford\_comma")})
\]

\[
\text{stick}(\ \\
\quad x, \ \\
\quad \text{fn} = \text{NULL}, \ \\
\quad \ldots, \ \\
\quad \text{max} = \text{Inf}, \ \\
\quad \text{fn\_overflow} = \text{FALSE}, \ \\
\quad \text{sep} = ", ", \ \\
\quad \text{conj} = " \ \text{and} \ \", \ \\
\quad \text{syndeton} = \text{c("last", \ "all", \ "none")}, \ \\
\quad \text{oxford} = \text{getOption("plu.oxford\_comma")})
\]

Arguments

- **x**: A character vector (or a vector coercible to character)
- **fn**: A function to apply to all items in the list
- **\ldots**: Additional arguments to **fn**
- **max**: The maximum number of items to list. Additional arguments are replaced with "n more". Defaults to Inf, which prints all items.
- **fn\_overflow**: Whether to apply **fn** to the overflow message when **x** contains more items than **max**. Defaults to FALSE.
- **sep**: The mark to place between list items. Defaults to ", ,".
- **conj**: A conjunction to place between list items. Defaults to " and ".
- **syndeton**: Whether to place the conjunction before the "last" list items, between "all" list items, or between "none". Defaults to "last".
- **oxford**: A logical value indicating whether to place **sep** before the last list item (x, y, and z) or not (x, y and z) in lists of length three or more where syndeton is "last". Defaults to TRUE if R’s locale is set to the United States and FALSE otherwise. The default can be changed by setting options(plu.oxford_comma).

Value

A character vector of length 1
See Also

\texttt{glue::glue_collapse()} for a generalized way to collapse vectors into a single string

Examples

\begin{verbatim}
ingredients <- c("sugar", "spice", "everything nice")
plu::stick(ingredients)

plu::stick(ingredients, fn = toupper)
plu::stick(names(formals(plu::stick)), fn = encodeString, quote = """)

plu::stick(ingredients, conj = "or")
plu::stick(ingredients, syndeton = "all")

plu::stick(ingredients, sep = "/", syndeton = "none")

creed <- c("snow", "rain", "heat", "gloom of night")
plu::stick(creed, conj = "nor", syndeton = "all")

dedication <- c("my parents", "Ayn Rand", "God")
plu::stick(dedication)
plu::stick(dedication, oxford = TRUE)
plu::stick(dedication, oxford = FALSE)
\end{verbatim}
Index

glue::glueCollapse(), 7
plu::ral(), 5
plu::ralize(), 3
plu_ral, 2
plu_ralize, 4
plu_stick, 5
ral (plu_ral), 2
ralize (plu_ralize), 4
stick (plu_stick), 5