Package ‘verbaliseR’

July 22, 2025

Title Make your Text Mighty Fine

Version 0.1

Description Turn R analysis outputs into full sentences, by writing vectors into in-sentence lists, pluralising words conditionally, spelling out numbers if they are at the start of sentences, writing out dates in full following US or UK style, and managing capitalisations in tidy data.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.1.2

URL https://github.com/cararthompson/verbaliseR

BugReports https://github.com/cararthompson/verbaliseR/issues

Maintainer Cara Thompson <cara.r.thompson@gmail.com>

Imports stringr

NeedsCompilation no

Author Cara Thompson [aut, cre] (ORCID:
<https://orcid.org/0000-0002-6472-5073>)

Repository CRAN

Date/Publication 2022-09-21 08:50:02 UTC

Contents

listify — *Turn vectors into lists with any specified linking word*

Description

Turn vectors into lists with any specified linking word

Usage

```
listify(items, linking_word = "and", oxford_comma = FALSE)
```

Arguments

`items` A vector of items to turn into a list phrase (e.g. c("a", "b", "c")).

`linking_word` Defaults to "and". Can be anything.

`oxford_comma` `logical`. Defaults to `FALSE`. If TRUE, an oxford comma is added (e.g. "a, b, and c").

Value

A string in the form of a list (e.g. "a, b and c")

Examples

```
listify(c("a", "b", "c"), "or")
```

num_to_text — *Spell out numbers if they are smaller than ten*

Description

Spell out numbers if they are smaller than ten

Usage

```
num_to_text(
  number,
  sentence_start = FALSE,
  zero_or_no = "no",
  uk_or_us = "UK",
  big_mark = ","
)
```

Arguments

`number` Whole number as `numeric` or `integer`, to be turned into text. Numbers 1-10 are always written out in full, regardless of their place in the sentence. Number 11-999 are written out in full if they are at the beginning of a sentence. Numbers greater than 1000 are returned as numerals.

`sentence_start` Logical. If `TRUE`, numbers below 100 are written out in full, and their first letter is capitalised.

`zero_or_no` Specify what to print when the number is 0. Defaults to "no". Can be any string.

`uk_or_us` Defaults to UK which adds an "and" between "hundred" and other numbers (e.g. "One hundred and five"). If "US" is chosen, the "and" is removed (e.g. "One hundred five").

`big_mark` Defaults to "," (e.g. "1,999").

Value

A string

Examples

```
num_to_text(3)
num_to_text(333, sentence_start = TRUE)
```

`pluralise` *Pluralise words if their accompanying number is not 1*

Description

Pluralise words if their accompanying number is not 1

Usage

```
pluralise(
  word,
  count,
  plural = "s",
  add_or_swap = "add",
  include_number = TRUE,
  sentence_start = FALSE,
  zero_or_no = "no",
  uk_or_us = "UK",
  big_mark = ","
)
```

Arguments

`word` A word which should be returned as plural if `count` is not equal to 1.

`count` A number to apply to `word`

`plural` How to make the plural; defaults to an "s" which is added at the end of the word. Can be anything. See `add_or_swap`.

`add_or_swap` Choose between `add` (add the plural form (e.g. "s") onto the end; e.g. `house` becomes `houses`) and `swap` (swap for the plural form; e.g. `mouse` becomes `mice`)

`include_number` Logical. If `TRUE`, the number will be turned into text, as per `num_to_text()` (if it is a whole number,)

`sentence_start` Logical. Defaults to `FALSE`, which results in only numbers 1-10 being written out in full. If `TRUE`, numbers 11-999 are written out in full if included. (If `include_number` is `FALSE`, the first letter of `word` is capitalised.)

`zero_or_no` Prefered string to use where count == 0. Defaults to "no". Can be anything.

`uk_or_us` Only used if `include_number == TRUE`. Defaults to UK which adds an "and" between "hundred" and other numbers (e.g. "One hundred and five"). If "US" is chosen, the "and" is removed (e.g. "One hundred five").

`big_mark` Passed to `num_to_text`. Defaults to "," (e.g. "1,999")

Value

A word which is pluralised or not based on the value of `count`

Examples

```
pluralise("penguin", 3)
pluralise("bateau", 1234, "x")
pluralise("sheep", 333, "sheep", add_or_swap = TRUE, sentence_start =  TRUE)
```

`prettify_date` *Render ordinal dates in UK or US style*

Description

Render ordinal dates in UK or US style

Usage

```
prettify_date(
  date_to_format = Sys.Date(),
  uk_or_us = "UK",
  formal_or_informal = "informal"
)
```

Arguments

`date_to_format` The date to use. It must be either be of class `Date` or a string written as "YYYY-MM-DD" or "YYYY/MM/DD")

`uk_or_us` Defaults to "UK", which results in outputs like "12th September 2022"; if "US", the output resembles "September 12th, 2022".

`formal_or_informal`
Defaults to "informal", so the ordinals are included (e.g. "st", "nd", "rd", "th"). If "formal" is chosen, the ordinals are omitted (e.g. "12 September 2022").

Value

A string (e.g. "12th September 2022")

Examples

```
prettify_date(Sys.Date(), "UK", "informal")
```

`restore_capitals` *Restore sustom capitalisation in a string*

Description

Restore sustom capitalisation in a string

Usage

```
restore_capitals(x, items_to_capitalise)
```

Arguments

`x` A string in which capitalisation needs to be restored

`items_to_capitalise`
Whole words or acronyms in which capitalisation must be retained; special characters can be included (e.g. "R2-D2")

Value

A string with restored capitals

Examples

```
x <- "Should i tell c-3po the french call him z-6po?"
restore_capitals(x, c("I", "C-3PO", "French", "Z-6PO"))
```

Index