

Package ‘concatenate’

August 29, 2016

Title Human-Friendly Text from Unknown Strings

Version 1.0.0

Description Simple functions for joining strings. Construct human-friendly messages whose elements aren't known in advance, like in stop, warning, or message, from clean code.

Depends R (>= 3.1.0)

Imports methods

License GPL (>= 3.2)

Encoding UTF-8

LazyData true

URL <https://github.com/jamesdunham/concatenate>

BugReports <https://github.com/jamesdunham/concatenate/issues>

Suggests testthat

RoxygenNote 5.0.1

NeedsCompilation no

Author James Dunham [aut, cre]

Maintainer James Dunham <james@jamesdunham.io>

Repository CRAN

Date/Publication 2016-05-08 15:20:17

R topics documented:

cc	2
cn	3
concatenate	4
infix	4

Index	5
--------------	----------

cc *Comma Concatenation*

Description

cc collapses text into a comma-separated list (the colloquial kind of list). `cc_or` and `cc_and` insert "or" and "and" before the last element.

Usage

```
cc(...)  
  
## S4 method for signature 'data.frame'  
cc(...)  
  
cc_or(..., oxford = FALSE)  
  
cc_and(..., oxford = FALSE)
```

Arguments

<code>...</code>	Character vectors or a <code>data.frame</code> .
<code>oxford</code>	Whether to use the Oxford comma.

Details

The `data.frame` method is dispatched when the first argument in `...` is a `data.frame`. It operates row-wise. If there are subsequent arguments to `cc` they are be ignored.

Value

A length-one character vector in which each element in `...` is separated by a comma (and a space).

See Also

[cn](#) for `cc` with (grammatical) number awareness (like `ngettext`) and substitution (like `sprintf`)

Examples

```
cc("hello", "world")  
  
a <- "one thing"  
b <- "another"  
cc_or(a, b)  
  
a <- "this"  
b <- c("that", "the other")  
cc_and(a, b)
```

cn *Number-aware Strings with Substitution*

Description

cn combines grammatical number awareness as in [ngettext](#) with [sprintf](#)-like substitution for comma-concatenated text.

Usage

```
cn(object, singular, plural = singular)

cn_and(object, singular, plural = singular)

cn_or(object, singular, plural = singular)

## S4 method for signature 'data.frame'
cn(object, singular, plural = singular)

## S4 method for signature 'data.frame'
cn_and(object, singular, plural = singular)

## S4 method for signature 'data.frame'
cn_or(object, singular, plural = singular)
```

Arguments

object	An n-vector, or <code>data.frame</code> with n rows.
singular	The string to return if n = 1.
plural	The string to return if n is in 0, 2, 3, 4, ...

Details

Like [ngettext](#), this function returns one string to be used with a singular referent and another with a plural referent. `cn` chooses between the two based on the length of its first argument, `object`, or if `object` is a `data.frame`, its row count.

Two substitutions are made `sprintf`-style. "%n" is replaced with the number of `object`, and "%c" is replaced with the comma-concatenated values of `object`, as in [cc](#).

`cn_and` uses [cc_and](#) instead of `cc`; `cn_or` uses [cc_or](#).

See Also

[cc](#)

concatenate	<i>concatenate: Comma Concatenation</i>
-------------	---

Description

Each function in concatenate returns a comma-separated string. (A length-one character vector.) They can be used to construct human-friendly messages whose elements aren't known in advance, like calls to `message`, `warning` or `stop`, from clean code.

Details

The workhorse function is `cc`. `cn` combines it with grammatical number awareness, as in `ngettext`, and `sprintf`-like substitution.

infix	<i>Binary Infix Concatenation</i>
-------	-----------------------------------

Description

`%+%`: binary infix operator for strings.
`% + %`: like `%+%` but with a space between its inputs.
`%,%,%or%,%and%`: infix versions of `cc`, `cc_or`, `cc_and`.

Usage

```
x %+% y
x % + % y
x %, % y
x %or% y
x %and% y
```

Arguments

`x`, `y` Character vectors.

Examples

```
v <- "important value"
v %+% "!"

message("Two" % + % "words")
```

Index

`% + %`(infix), 4
`%+%`(infix), 4
`%,%`(infix), 4
`%and%`(infix), 4
`%or%`(infix), 4

`cc`, 2, 3, 4
`cc`, data.frame-method (`cc`), 2
`cc_and`, 3, 4
`cc_and`(`cc`), 2
`cc_or`, 3, 4
`cc_or`(`cc`), 2
`cn`, 2, 3, 4
`cn`, data.frame-method (`cn`), 3
`cn_and`(`cn`), 3
`cn_and`, data.frame-method (`cn`), 3
`cn_or`(`cn`), 3
`cn_or`, data.frame-method (`cn`), 3
`concatenate`, 4
`concatenate`-package (`concatenate`), 4

`infix`, 4

`ngettext`, 3, 4

`sprintf`, 3, 4