

Package ‘chk’

January 15, 2020

Title Check User-Supplied Function Arguments

Version 0.3.0

Description For developers to check user-supplied function arguments. It is designed to be simple, fast and customizable. Error messages follow the tidyverse style guide.

License MIT + file LICENSE

URL <https://github.com/poissonconsulting/chk>

BugReports <https://github.com/poissonconsulting/chk/issues>

Depends R (>= 3.3)

Imports lifecycle,
methods,
rlang,
tools,
utils

Suggests covr,
knitr,
rmarkdown,
testthat

VignetteBuilder knitr

RdMacros lifecycle

Encoding UTF-8

Language en-US

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.0.2

R topics documented:

abort_chk	3
cc	3
chkor	4
chk_all	5
chk_all_equal	6
chk_all_equivalent	7

chk_all_identical	8
chk_atomic	9
chk_date	10
chk_datetime	11
chk_dir	12
chk_environment	13
chk_equal	14
chk_equivalent	15
chk_ext	16
chk_false	17
chk_file	18
chk_flag	19
chk_function	20
chk_gt	21
chk_gte	22
chk_identical	23
chk_lgl	24
chk_list	25
chk_lt	26
chk_lte	27
chk_match	28
chk_named	29
chk_not_any_na	30
chk_not_empty	31
chk_not_null	32
chk_null	33
chk_number	34
chk_numeric	35
chk_range	36
chk_s3_class	37
chk_s4_class	38
chk_scalar	39
chk_setequal	40
chk_sorted	41
chk_string	42
chk_subset	43
chk_superset	44
chk_true	45
chk_unique	46
chk_unused	47
chk_used	48
chk_vector	49
chk_whole_number	50
chk_whole_numeric	51
deparse_backtick	52
err	53
message_chk	54
p	55
vld	55

 abort_chk

Abort Check

Description

A wrapper on `err()` that sets the subclass to be 'chk_error'.

Usage

```
abort_chk(..., n = NULL, tidy = TRUE)
```

Arguments

...	Multiple objects that are converted to a string using <code>paste0(..., collapse = '')</code> .
n	The value of n for converting <code>sprintf</code> -like types.
tidy	A flag specifying whether capitalize the first character and add a missing period.

Details

It is exported to allow users to easily construct their own `chk_` functions.

Value

Throws an error of class 'chk_error'.

See Also

[err\(\)](#)

Examples

```
try(abort_chk("x must be NULL"))
try(abort_chk("`x` must be NULL"))
try(abort_chk("there %r %n problem value%s", n = 1))
try(abort_chk("there %r %n problem value%s", n = 1.5))
```

 cc

Concatenate with Commas

Description

Concatenates object values into a string with each value separated by a comma and the last value separated by a conjunction.

Usage

```

cc(
  x,
  conj = ", ",
  sep = ", ",
  brac = if (is.character(x) || is.factor(x)) "" else "",
  ellipsis = 10L,
  chk = TRUE
)

```

Arguments

x	The object to concatenate.
conj	A string of the conjunction to separate the last value by.
sep	A string of the separator.
brac	A string to brace the values by.
ellipsis	A numeric scalar of the maximum number of values to display before using an ellipsis.
chk	A flag specifying whether to check the other parameters.

Details

By default, if x has more than 10 values an ellipsis is used to ensure only 10 values are displayed (including the ellipsis).

Value

A string.

Examples

```

cc(1:2)
cc(1:2, conj = " or")
cc(3:1, brac = "'")
cc(1:11)
cc(as.character(1:2))

```

chkor

Check OR

Description

Check OR

Usage

```
chkor(...)
```

Arguments

... Multiple chk_ functions.

Value

An informative error if the test fails.

Examples

```
chkor()  
chkor(chk_flag(TRUE))  
try(chkor(chk_flag(1)))  
try(chkor(chk_flag(1), chk_flag(2)))  
chkor(chk_flag(1), chk_flag(TRUE))
```

chk_all	<i>Check All</i>
---------	------------------

Description

Checks all elements using
`all(vapply(x, chk_fun, TRUE, ...))`

Usage

```
chk_all(x, chk_fun, ..., x_name = NULL)  
  
vld_all(x, vld_fun, ...)
```

Arguments

x	The object to check.
chk_fun	A <code>chk_</code> function.
...	Additional arguments.
x_name	A string of the name of object x or NULL.
vld_fun	A <code>vld_</code> function.

Value

The `chk_` function throws an informative error if the test fails.

The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_all`: Validate All

See Also

Other `chk_all`s: [chk_all_equal\(\)](#), [chk_all_equivalent\(\)](#), [chk_all_identical\(\)](#)

Examples

```
# chk_all
chk_all(TRUE, chk_lgl)
# FIXME try(chk_all(1, chk_lgl))
chk_all(c(TRUE, NA), chk_lgl)

# vld_all
vld_all(c(TRUE, NA), vld_lgl)
```

chk_all_equal	<i>Check All Equal</i>
---------------	------------------------

Description

Checks all elements in `x` equal using

```
length(x) < 2L || all(vapply(x, vld_equal, TRUE, y = x[[1]], tolerance = tolerance))
```

Usage

```
chk_all_equal(x, tolerance = sqrt(.Machine$double.eps), x_name = NULL)
```

```
vld_all_equal(x, tolerance = sqrt(.Machine$double.eps))
```

Arguments

<code>x</code>	The object to check.
<code>tolerance</code>	A non-negative numeric scalar.
<code>x_name</code>	A string of the name of object <code>x</code> or <code>NULL</code> .

Value

The `chk_` function throws an informative error if the test fails.

The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_all_equal`: Validate All Equal

See Also

Other `chk_`alls: [chk_all_equivalent\(\)](#), [chk_all_identical\(\)](#), [chk_all\(\)](#)

Examples

```
# chk_all_equal
chk_all_equal(c(1, 1.0000001))
try(chk_all_equal(c(1, 1.000001)))
chk_all_equal(list(c(x = 1), c(x = 1)))
try(chk_all_equal(list(c(x = 1), c(y = 1))))

# vld_all_equal
vld_all_equal(c(1, 1L))
```

chk_all_equivalent	<i>Check All Equivalent</i>
--------------------	-----------------------------

Description

Checks all elements in `x` equivalent using

```
length(x) < 2L || all(vapply(x, vld_equivalent, TRUE, y = x[[1]], tolerance = tolerance))
```

Usage

```
chk_all_equivalent(x, tolerance = sqrt(.Machine$double.eps), x_name = NULL)
```

```
vld_all_equivalent(x, tolerance = sqrt(.Machine$double.eps))
```

Arguments

<code>x</code>	The object to check.
<code>tolerance</code>	A non-negative numeric scalar.
<code>x_name</code>	A string of the name of object <code>x</code> or <code>NULL</code> .

Value

The `chk_` function throws an informative error if the test fails.

The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_all_equivalent`: Validate All Equivalent

See Also

Other `chk_`alls: [chk_all_equal\(\)](#), [chk_all_identical\(\)](#), [chk_all\(\)](#)

Examples

```
# chk_all_equivalent
chk_all_equivalent(c(1, 1.00000001))
try(chk_all_equivalent(c(1, 1.0000001)))
chk_all_equivalent(list(c(x = 1), c(x = 1)))
chk_all_equivalent(list(c(x = 1), c(y = 1)))

# vld_all_equivalent
vld_all_equivalent(c(x = 1, y = 1))
```

chk_all_identical *Check All Identical*

Description

Checks all elements in x identical using

```
length(x) < 2L || all(vapply(x, vld_identical, TRUE, y = x[[1]]))
```

Good: c(1, 1.00000001), list(1, 1)

Bad: c(1, 1.0000001), list(1, NA)

Usage

```
chk_all_identical(x, x_name = NULL)
```

```
vld_all_identical(x)
```

Arguments

x	The object to check.
x_name	A string of the name of object x or NULL.

Value

The chk_ function throws an informative error if the test fails.

The vld_ function returns a flag indicating whether the test was met.

Functions

- vld_all_identical: Validate All Identical

See Also

Other chk_all: [chk_all_equal\(\)](#), [chk_all_equivalent\(\)](#), [chk_all\(\)](#)

Examples

```
# chk_all_identical
chk_all_identical(c(1, 1))
try(chk_all_identical(c(1, 1.1)))

# vld_all_identical
vld_all_identical(c(1, 1))
```

chk_atomic	<i>Check Atomic</i>
------------	---------------------

Description

Checks if atomic using
`is.atomic(x)`

Usage

```
chk_atomic(x, x_name = NULL)

vld_atomic(x)
```

Arguments

x	The object to check.
x_name	A string of the name of object x or NULL.

Value

The `chk_` function throws an informative error if the test fails.
The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_atomic`: Validate Atomic

See Also

Other `chk_is`: [chk_environment\(\)](#), [chk_function\(\)](#), [chk_list\(\)](#), [chk_numeric\(\)](#), [chk_s3_class\(\)](#), [chk_s4_class\(\)](#), [chk_vector\(\)](#), [chk_whole_numeric\(\)](#)

Examples

```
# chk_atomic
chk_atomic(1)
try(chk_atomic(list(1)))

# vld_atomic
vld_atomic(1)
vld_atomic(matrix(1:3))
```

```
vld_atomic(character(0))  
vld_atomic(list(1))  
vld_atomic(NULL)
```

chk_date

Check Date

Description

Checks non-missing Date scalar using
`inherits(x, "Date") && length(x) == 1L && !anyNA(x)`

Usage

```
chk_date(x, x_name = NULL)  
  
vld_date(x)
```

Arguments

`x` The object to check.
`x_name` A string of the name of object `x` or `NULL`.

Value

The `chk_` functions throw an informative error if the test fails.
The `vld_` functions return a flag indicating whether the test was met.

Functions

- `vld_date`: Validate Date

See Also

Other `chk_` scalars: [chk_datetime\(\)](#), [chk_number\(\)](#), [chk_scalar\(\)](#), [chk_string\(\)](#), [chk_whole_number\(\)](#)

Examples

```
# chk_date  
chk_date(Sys.Date())  
try(chk_date(1))  
  
# vld_date  
vld_date(Sys.Date())  
vld_date(Sys.time())  
vld_date(1)
```

chk_datetime	<i>Check DateTime</i>
--------------	-----------------------

Description

Checks if non-missing POSIXct scalar using
`inherits(x, "POSIXct") && length(x) == 1L && !anyNA(x)`

Usage

```
chk_datetime(x, x_name = NULL)
vld_datetime(x, x_name = NULL)
```

Arguments

x	The object to check.
x_name	A string of the name of object x or NULL.

Value

The `chk_` functions throw an informative error if the test fails.
The `vld_` functions return a flag indicating whether the test was met.

Functions

- `vld_datetime`: Validate DateTime

See Also

Other `chk_` scalars: [chk_date\(\)](#), [chk_number\(\)](#), [chk_scalar\(\)](#), [chk_string\(\)](#), [chk_whole_number\(\)](#)

Examples

```
# chk_datetime
chk_datetime(as.POSIXct("2001-01-02"))
try(chk_datetime(1))

# vld_datetime
vld_datetime(as.POSIXct("2001-01-02"))
vld_datetime(Sys.time())
vld_datetime(1)
vld_datetime("2001-01-02")
vld_datetime(c(Sys.time(), Sys.time()))
```

chk_dir	<i>Check Directory Exists</i>
---------	-------------------------------

Description

Checks if directory exists using
`vld_string(x) && dir.exists(x)`

Usage

```
chk_dir(x, x_name = NULL)

vld_dir(x)
```

Arguments

x	The object to check.
x_name	A string of the name of object x or NULL.

Value

The `chk_` function throws an informative error if the test fails.
The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_dir`: Validate Directory Exists

See Also

Other `chk_files`: [chk_ext\(\)](#), [chk_file\(\)](#)

Examples

```
# chk_dir
chk_dir(tempdir())
try(chk_dir(tempfile()))

# vld_dir
vld_dir(1)
vld_dir(tempdir())
vld_dir(tempfile())
```

chk_environment	<i>Check Environment</i>
-----------------	--------------------------

Description

Checks if environment using
`is.environment(x)`

Usage

```
chk_environment(x, x_name = NULL)
vld_environment(x)
```

Arguments

x	The object to check.
x_name	A string of the name of object x or NULL.

Value

The `chk_` function throws an informative error if the test fails.

The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_environment`: Validate Environment

See Also

Other `chk_` is: [chk_atomic\(\)](#), [chk_function\(\)](#), [chk_list\(\)](#), [chk_numeric\(\)](#), [chk_s3_class\(\)](#), [chk_s4_class\(\)](#), [chk_vector\(\)](#), [chk_whole_numeric\(\)](#)

Examples

```
# chk_environment
chk_environment(.GlobalEnv)
try(chk_environment(1))

# vld_environment
vld_environment(1)
vld_environment(list(1))
vld_environment(.GlobalEnv)
vld_environment(environment())
```

`chk_equal`*Check Equal*

Description

Checks if is equal (identical within tolerance) to y using
`vld_true(all.equal(x,y,tolerance))`

Usage

```
chk_equal(x, y, tolerance = sqrt(.Machine$double.eps), x_name = NULL)

vld_equal(x, y, tolerance = sqrt(.Machine$double.eps))
```

Arguments

<code>x</code>	The object to check.
<code>y</code>	An object to check against.
<code>tolerance</code>	A non-negative numeric scalar.
<code>x_name</code>	A string of the name of object x or NULL.

Value

The `chk_` function throws an informative error if the test fails.
The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_equal`: Validate Equal

See Also

Other `chk_` equals: [chk_equivalent\(\)](#), [chk_identical\(\)](#)

Examples

```
# chk_equal
chk_equal(1, 1.00000001)
try(chk_equal(1, 1.0000001))
chk_equal(1, 1L)
chk_equal(c(x = 1), c(x = 1L))
try(chk_equal(c(x = 1), c(y = 1L)))

vld_equal(1, 1.00000001)
```

chk_equivalent	<i>Check Equivalent</i>
----------------	-------------------------

Description

Checks if is equivalent (equal ignoring attributes) to y using
vld_true(all.equal(x,y,tolerance,check.attributes = FALSE))

Usage

```
chk_equivalent(x, y, tolerance = sqrt(.Machine$double.eps), x_name = NULL)
vld_equivalent(x, y, tolerance = sqrt(.Machine$double.eps))
```

Arguments

x	The object to check.
y	An object to check against.
tolerance	A non-negative numeric scalar.
x_name	A string of the name of object x or NULL.

Value

The `chk_` function throws an informative error if the test fails.
The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_equivalent`: Validate Equivalent

See Also

Other `chk_equals`: [chk_equal\(\)](#), [chk_identical\(\)](#)

Examples

```
# chk_equivalent
chk_equivalent(1, 1.00000001)
try(chk_equivalent(1, 1.0000001))
chk_equivalent(1, 1L)
chk_equivalent(c(x = 1), c(y = 1))

vld_equivalent(c(x = 1), c(y = 1L))
```

`chk_ext`*Check File Extension*

Description

Checks extension using

```
vld_string(x) && vld_subset(tools::file_ext(x), ext)
```

The user may want to use `toupper()` or `tolower()` to ensure the case matches.

Usage

```
chk_ext(x, ext, x_name = NULL)
```

```
vld_ext(x, ext)
```

Arguments

<code>x</code>	The object to check.
<code>ext</code>	A character vector of the permitted file extensions (without the <code>.</code>).
<code>x_name</code>	A string of the name of object <code>x</code> or <code>NULL</code> .

Value

The `chk_` function throws an informative error if the test fails.

The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_ext`: Validate File Extension

See Also

Other `chk_files`: `chk_dir()`, `chk_file()`

Examples

```
# chk_ext
try(chk_ext("file1.pdf", "png"))

# vld_ext
vld_ext("oeu.pdf", "pdf")
vld_ext(toupper("oeu.pdf"), "PDF")
```

chk_false	<i>Check FALSE</i>
-----------	--------------------

Description

Check if FALSE using

```
is.logical(x) && length(x) == 1L && !anyNA(x) && !x
```

Usage

```
chk_false(x, x_name = NULL)
```

```
vld_false(x)
```

Arguments

x	The object to check.
x_name	A string of the name of object x or NULL.

Value

The `chk_` function throws an informative error if the test fails.

The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_false`: Validate FALSE

See Also

Other `chk_logical`: [chk_flag\(\)](#), [chk_lgl\(\)](#), [chk_true\(\)](#)

Examples

```
# chk_false
chk_false(FALSE)
try(chk_false(0))

# vld_false
vld_false(TRUE)
vld_false(FALSE)
vld_false(NA)
vld_false(0)
vld_false(c(FALSE, FALSE))
```

chk_file	<i>Check File Exists</i>
----------	--------------------------

Description

Checks if file exists using

```
vld_string(x) && file.exists(x) && !dir.exists(x)
```

Usage

```
chk_file(x, x_name = NULL)
```

```
vld_file(x)
```

Arguments

x	The object to check.
x_name	A string of the name of object x or NULL.

Value

The `chk_` functions throw an informative error if the test fails.

The `vld_` functions return a flag indicating whether the test was met.

Functions

- `vld_file`: Validate File Exists

See Also

Other `chk_files`: [chk_dir\(\)](#), [chk_ext\(\)](#)

Examples

```
# chk_file
try(chk_file(tempfile()))

# vld_file
vld_file(tempfile())
```

`chk_flag`*Check Flag*

Description

Checks if non-missing logical scalar using
`is.logical(x) && length(x) == 1L && !anyNA(x)`

Good: TRUE, FALSE, NA.

Bad: `logical(0)`, `c(TRUE,TRUE)`, "TRUE", 1, NA_real_.

Usage

```
chk_flag(x, x_name = NULL)
```

```
vld_flag(x)
```

Arguments

<code>x</code>	The object to check.
<code>x_name</code>	A string of the name of object <code>x</code> or NULL.

Value

The `chk_` function throws an informative error if the test fails.

The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_flag`: Validate Flag

See Also

Other `chk_logical`: [chk_false\(\)](#), [chk_lgl\(\)](#), [chk_true\(\)](#)

Examples

```
# chk_flag
chk_flag(TRUE)
try(vld_flag(1))

# vld_flag
vld_flag(TRUE)
vld_flag(1)
```

`chk_function`*Check Function*

Description

Checks if is a function using

```
is.function(x) && (is.null(formals) || length(formals(x)) == formals)
```

Usage

```
chk_function(x, formals = NULL, x_name = NULL)
```

```
vld_function(x, formals = NULL)
```

Arguments

<code>x</code>	The object to check.
<code>formals</code>	A count of the number of formal arguments.
<code>x_name</code>	A string of the name of object <code>x</code> or <code>NULL</code> .

Value

The `chk_function` throws an informative error if the test fails.

The `vld_function` returns a flag indicating whether the test was met.

Functions

- `vld_function`: Validate Function

See Also

Other `chk_is`: [chk_atomic\(\)](#), [chk_environment\(\)](#), [chk_list\(\)](#), [chk_numeric\(\)](#), [chk_s3_class\(\)](#), [chk_s4_class\(\)](#), [chk_vector\(\)](#), [chk_whole_numeric\(\)](#)

Examples

```
# chk_function
chk_function(mean)
try(chk_function(1))

# vld_function
vld_function(mean)
vld_function(function(x) x)
vld_function(1)
vld_function(list(1))
```

chk_gt	<i>Check Greater Than</i>
--------	---------------------------

Description

Checks if all non-missing values are greater than value using
`all(x[!is.na(x)] > value)`

Usage

```
chk_gt(x, value = 0, x_name = NULL)
```

```
vld_gt(x, value = 0)
```

Arguments

x	The object to check.
value	A non-missing scalar of a value.
x_name	A string of the name of object x or NULL.

Value

The `chk_` function throws an informative error if the test fails.

The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_gt`: Validate Greater Than

See Also

Other `chk_ranges`: [chk_gte\(\)](#), [chk_lte\(\)](#), [chk_lt\(\)](#), [chk_range\(\)](#)

Examples

```
# chk_gt
chk_gt(0.1)
try(chk_gt(c(0.1, -0.2)))

# vld_gt
vld_gt(numeric(0))
vld_gt(0)
vld_gt(0.1)
vld_gt(c(0.1, 0.2, NA))
vld_gt(c(0.1, -0.2))
vld_gt(c(-0.1, 0.2), value = -1)
vld_gt("b", value = "a")
```

`chk_gte`*Check Greater Than or Equal To*

Description

Checks if all non-missing values are greater than or equal to `y` using `all(x[!is.na(x)] >= value)`

Usage

```
chk_gte(x, value = 0, x_name = NULL)
```

```
vld_gte(x, value = 0)
```

Arguments

<code>x</code>	The object to check.
<code>value</code>	A non-missing scalar of a value.
<code>x_name</code>	A string of the name of object <code>x</code> or <code>NULL</code> .

Value

The `chk_` function throws an informative error if the test fails.

The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_gte`: Validate Greater Than or Equal To

See Also

Other `chk_ranges`: [chk_gt\(\)](#), [chk_lte\(\)](#), [chk_lt\(\)](#), [chk_range\(\)](#)

Examples

```
# chk_gte
chk_gte(0)
try(chk_gte(-0.1))

# vld_gte
vld_gte(numeric(0))
vld_gte(0)
vld_gte(-0.1)
vld_gte(c(0.1, 0.2, NA))
vld_gte(c(0.1, 0.2, NA), value = 1)
```

chk_identical	<i>Check Identical</i>
---------------	------------------------

Description

Checks if `x` is identical to `y` using
`identical(x,y)`

Usage

```
chk_identical(x, y, x_name = NULL)
vld_identical(x, y)
```

Arguments

<code>x</code>	The object to check.
<code>y</code>	An object to check against.
<code>x_name</code>	A string of the name of object <code>x</code> or <code>NULL</code> .

Value

The `chk_` function throws an informative error if the test fails.
The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_identical`: Validate Identical

See Also

Other `chk_equals`: [chk_equal\(\)](#), [chk_equivalent\(\)](#)

Examples

```
# chk_identical
chk_identical(1, 1)
try(chk_identical(1, 1L))
chk_identical(c(1, 1), c(1, 1))
try(chk_identical(1, c(1, 1)))

vld_identical(1, 1)
```

`chk_lgl`*Check Logical Scalar*

Description

Checks if logical scalar using
`is.logical(x) && length(x) == 1L`

Usage

```
chk_lgl(x, x_name = NULL)
vld_lgl(x)
```

Arguments

<code>x</code>	The object to check.
<code>x_name</code>	A string of the name of object <code>x</code> or <code>NULL</code> .

Value

The `chk_` function throws an informative error if the test fails.
The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_lgl`: Validate Logical Scalar

See Also

Other `chk_logical`: [chk_false\(\)](#), [chk_flag\(\)](#), [chk_true\(\)](#)

Examples

```
# chk_lgl
chk_lgl(NA)
try(chk_lgl(1))

# vld_lgl
vld_lgl(TRUE)
vld_lgl(FALSE)
vld_lgl(NA)
vld_lgl(1)
vld_lgl(c(TRUE, TRUE))
```

`chk_list`*Check List*

Description

Checks if is a list using
`is.list(x)`

Usage

```
chk_list(x, x_name = NULL)

vld_list(x)
```

Arguments

<code>x</code>	The object to check.
<code>x_name</code>	A string of the name of object <code>x</code> or <code>NULL</code> .

Value

The `chk_` function throws an informative error if the test fails.
The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_list`: Validate List

See Also

Other `chk_is`: [chk_atomic\(\)](#), [chk_environment\(\)](#), [chk_function\(\)](#), [chk_numeric\(\)](#), [chk_s3_class\(\)](#),
[chk_s4_class\(\)](#), [chk_vector\(\)](#), [chk_whole_numeric\(\)](#)

Examples

```
# chk_list
chk_list(list())
try(chk_list(1))

# vld_list
vld_list(list())
vld_list(list(x = 1))
vld_list(mtcars)
vld_list(1)
vld_list(NULL)
```

chk_lt	<i>Check Less Than</i>
--------	------------------------

Description

Checks if all non-missing values are less than value using
`all(x[!is.na(x)] < value)`

Usage

```
chk_lt(x, value = 0, x_name = NULL)
vld_lt(x, value = 0)
```

Arguments

x	The object to check.
value	A non-missing scalar of a value.
x_name	A string of the name of object x or NULL.

Value

The `chk_` function throws an informative error if the test fails.

The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_lt`: Validate Less Than

See Also

Other `chk_ranges`: [chk_gte\(\)](#), [chk_gt\(\)](#), [chk_lte\(\)](#), [chk_range\(\)](#)

Examples

```
# chk_lt
chk_lt(-0.1)
try(chk_lt(c(-0.1, 0.2)))

# vld_lt
vld_lt(numeric(0))
vld_lt(0)
vld_lt(-0.1)
vld_lt(c(-0.1, -0.2, NA))
vld_lt(c(-0.1, 0.2))
vld_lt(c(-0.1, 0.2), value = 1)
vld_lt("a", value = "b")
```

chk_lte	<i>Check Less Than or Equal To</i>
---------	------------------------------------

Description

Checks if all non-missing values are less than or equal to y using
`all(x[!is.na(x)] <= value)`

Usage

```
chk_lte(x, value = 0, x_name = NULL)
```

```
vld_lte(x, value = 0)
```

Arguments

x	The object to check.
value	A non-missing scalar of a value.
x_name	A string of the name of object x or NULL.

Value

The `chk_` function throws an informative error if the test fails.

The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_lte`: Validate Less Than or Equal To

See Also

Other `chk_` ranges: [chk_gte\(\)](#), [chk_gt\(\)](#), [chk_lt\(\)](#), [chk_range\(\)](#)

Examples

```
# chk_lte
chk_lte(0)
try(chk_lte(0.1))

# vld_lte
vld_lte(numeric(0))
vld_lte(0)
vld_lte(0.1)
vld_lte(c(-0.1, -0.2, NA))
vld_lte(c(-0.1, -0.2, NA), value = -1)
```

`chk_match`*Check Matches*

Description

Checks if all values match regular expression using
`all(grepl(regex, x[!is.na(x)]))`

Usage

```
chk_match(x, regex = ".+", x_name = NULL)
```

```
vld_match(x, regex = ".+")
```

Arguments

<code>x</code>	The object to check.
<code>regex</code>	A string of a regular expression.
<code>x_name</code>	A string of the name of object <code>x</code> or <code>NULL</code> .

Value

The `chk_` function throws an informative error if the test fails.

The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_match`: Validate Matches

See Also

Other `chk_misc`: [chk_named\(\)](#), [chk_unique\(\)](#)

Examples

```
# chk_match
chk_match("1")
try(chk_match("1", regex = "2"))

# vld_match
vld_match("1")
vld_match("a", regex = "a")
vld_match("")
vld_match("1", regex = "2")
vld_match(NA_character_, regex = ".*")
```

chk_named	<i>Check Named</i>
-----------	--------------------

Description

Checks if is named using

```
!is.null(names(x))
```

Usage

```
chk_named(x, x_name = NULL)
```

```
vld_named(x)
```

Arguments

x	The object to check.
x_name	A string of the name of object x or NULL.

Value

The `chk_` function throws an informative error if the test fails.

The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_named`: Validate Named

See Also

Other `chk_misc`: [chk_match\(\)](#), [chk_unique\(\)](#)

Examples

```
# chk_named
chk_named(c(x = 1))
try(chk_named(list(1)))

# vld_named
vld_named(c(x = 1))
vld_named(list(x = 1))
vld_named(c(x = 1)[-1])
vld_named(list(x = 1)[-1])
vld_named(1)
vld_named(list(1))
```

chk_not_any_na *Check Not Any Missing Values*

Description

Checks if not any missing values using

`!anyNA(x)`

Good: 1, 1:2, "1", `logical(0)`.

Bad: NA, `c(1, NA)`.

Usage

`chk_not_any_na(x, x_name = NULL)`

`vld_not_any_na(x)`

Arguments

x The object to check.

x_name A string of the name of object x or NULL.

Value

The `chk_` function throws an informative error if the test fails.

The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_not_any_na`: Validate Not Any Missing Values

See Also

Other `chk_miscellaneous`: [chk_not_empty\(\)](#), [chk_sorted\(\)](#)

Examples

```
# chk_not_any_na
chk_not_any_na(1)
try(chk_not_any_na(NA))

# vld_not_any_na
vld_not_any_na(1)
vld_not_any_na(1:2)
vld_not_any_na(NA_real_)
vld_not_any_na(integer(0))
vld_not_any_na(c(NA, 1))
vld_not_any_na(TRUE)
```

chk_not_empty	<i>Check Not Empty</i>
---------------	------------------------

Description

Checks if not empty using

`length(x) != 0L`

Good: 1, 1:2, NA, `matrix(1:3)`, `list(1)`, `data.frame(x = 1)`.

Bad: NULL, `logical(0)`, `list()`, `data.frame()`.

Usage

```
chk_not_empty(x, x_name = NULL)
```

```
vld_not_empty(x)
```

Arguments

x	The object to check.
x_name	A string of the name of object x or NULL.

Value

The `chk_` function throws an informative error if the test fails.

The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_not_empty`: Validate Not Empty

See Also

Other `chk_miscellaneous`: [chk_not_any_na\(\)](#), [chk_sorted\(\)](#)

Examples

```
# chk_not_empty
chk_not_empty(1)
try(chk_not_empty(numeric(0)))

# vld_not_empty
vld_not_empty(1)
vld_not_empty(matrix(1:3))
vld_not_empty(character(0))
vld_not_empty(list(1))
vld_not_empty(NULL)
vld_not_empty(list())
```

chk_not_null	<i>Check not NULL</i>
--------------	-----------------------

Description

Checks if not NULL using
`!is.null(x)`

Usage

```
chk_not_null(x, x_name = NULL)
```

```
vld_not_null(x)
```

Arguments

<code>x</code>	The object to check.
<code>x_name</code>	A string of the name of object <code>x</code> or NULL.

Value

The `chk_` function throws an informative error if the test fails.

The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_not_null`: Validate Not NULL

See Also

Other `chk_nulls`: [chk_null\(\)](#)

Examples

```
# chk_not_null
try(chk_not_null(NULL))
chk_not_null(1)

# vld_not_null
vld_not_null(1)
vld_not_null(NULL)
```

chk_null	<i>Check NULL</i>
----------	-------------------

Description

Checks if NULL using
`is.null(x)`

Usage

```
chk_null(x, x_name = NULL)
```

```
vld_null(x)
```

Arguments

x	The object to check.
x_name	A string of the name of object x or NULL.

Value

The `chk_` functions throw an informative error if the test fails.

The `vld_` functions return a flag indicating whether the test was met.

Functions

- `vld_null`: Validate NULL

See Also

Other `chk_nulls`: [chk_not_null\(\)](#)

Examples

```
# chk_null
try(chk_null(1))
chk_null(NULL)

# vld_null
vld_null(NULL)
vld_null(1)
```

chk_number	<i>Check Number</i>
------------	---------------------

Description

Checks if non-missing numeric scalar using
`is.numeric(x) && length(x) == 1L && !anyNA(x)`

Good: 1, 2L, log(10), -Inf

Bad: "a", 1:3, NA_real_

Usage

```
chk_number(x, x_name = NULL)
```

```
vld_number(x)
```

Arguments

x	The object to check.
x_name	A string of the name of object x or NULL.

Value

The `chk_` function throws an informative error if the test fails.

The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_number`: Validate Number

See Also

Other `chk_scalars`: [chk_datetime\(\)](#), [chk_date\(\)](#), [chk_scalar\(\)](#), [chk_string\(\)](#), [chk_whole_number\(\)](#)

Examples

```
# chk_number
chk_number(1.1)
try(chk_number(TRUE))

# vld_number
vld_number(1.1)
```

chk_numeric	<i>Check Numeric</i>
-------------	----------------------

Description

Checks if numeric using

`is.numeric(x)`

Good: 1, 1:2, NA_real_, integer(0), matrix(1:3).

Bad: TRUE, "1", NA, NULL.

Usage

```
chk_numeric(x, x_name = NULL)
```

```
vld_numeric(x)
```

Arguments

x The object to check.

x_name A string of the name of object x or NULL.

Value

The `chk_` function throws an informative error if the test fails.

The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_numeric`: Validate Numeric

See Also

Other `chk_is`: [chk_atomic\(\)](#), [chk_environment\(\)](#), [chk_function\(\)](#), [chk_list\(\)](#), [chk_s3_class\(\)](#), [chk_s4_class\(\)](#), [chk_vector\(\)](#), [chk_whole_numeric\(\)](#)

Examples

```
# chk_numeric
chk_numeric(1)
try(chk_numeric("1"))

# vld_numeric
vld_numeric(1)
vld_numeric(1:2)
vld_numeric(NA_real_)
vld_numeric(integer(0))
vld_numeric("1")
vld_numeric(TRUE)
```

chk_range	<i>Checks range of non-missing values</i>
-----------	---

Description

Checks all non-missing values fall within range using

```
all(x[!is.na(x)] >= range[1] & x[!is.na(x)] <= range[2])
```

Usage

```
chk_range(x, range = c(0, 1), x_name = NULL)
```

```
vld_range(x, range = c(0, 1))
```

Arguments

x	The object to check.
range	A non-missing sorted vector of length 2 of the lower and upper permitted values.
x_name	A string of the name of object x or NULL.

Value

The `chk_` functions throw an informative error if the test fails.

The `vld_` functions return a flag indicating whether the test was met.

Functions

- `vld_range`: Validate Range

See Also

Other `chk_` ranges: [chk_gte\(\)](#), [chk_gt\(\)](#), [chk_lte\(\)](#), [chk_lt\(\)](#)

Examples

```
# chk_range
chk_range(0)
try(chk_range(-0.1))

# vld_range
vld_range(numeric(0))
vld_range(0)
vld_range(-0.1)
vld_range(c(0.1, 0.2, NA))
vld_range(c(0.1, 0.2, NA), range = c(0, 1))
```

chk_s3_class	<i>Check Type</i>
--------------	-------------------

Description

Checks inherits from S3 class using
`!isS4(x) && inherits(x, class)`

Usage

```
chk_s3_class(x, class, x_name = NULL)
vld_s3_class(x, class)
```

Arguments

x	The object to check.
class	A string specifying the class.
x_name	A string of the name of object x or NULL.

Value

The `chk_` functions throw an informative error if the test fails.
The `vld_` functions return a flag indicating whether the test was met.

Functions

- `vld_s3_class`: Validate Inherits from S3 Class

See Also

Other `chk_` is: [chk_atomic\(\)](#), [chk_environment\(\)](#), [chk_function\(\)](#), [chk_list\(\)](#), [chk_numeric\(\)](#), [chk_s4_class\(\)](#), [chk_vector\(\)](#), [chk_whole_numeric\(\)](#)

Examples

```
# chk_s3_class
chk_s3_class(1, "numeric")
try(chk_s3_class(getClass("MethodDefinition"), "classRepresentation"))

# vld_s3_class
vld_s3_class(numeric(0), "numeric")
vld_s3_class(getClass("MethodDefinition"), "classRepresentation")
```

chk_s4_class *Check Inherits from S4 Class*

Description

Checks inherits from S4 class using
`isS4(x) && methods::is(x, class)`

Usage

```
chk_s4_class(x, class, x_name = NULL)
vld_s4_class(x, class)
```

Arguments

x	The object to check.
class	A string specifying the class.
x_name	A string of the name of object x or NULL.

Value

The `chk_` function throws an informative error if the test fails.
The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_s4_class`: Validate Inherits from S4 Class

See Also

Other `chk_`is: [chk_atomic\(\)](#), [chk_environment\(\)](#), [chk_function\(\)](#), [chk_list\(\)](#), [chk_numeric\(\)](#),
[chk_s3_class\(\)](#), [chk_vector\(\)](#), [chk_whole_numeric\(\)](#)

Examples

```
# chk_s4_class
try(chk_s4_class(1, "numeric"))
chk_s4_class(getClass("MethodDefinition"), "classRepresentation")

# vld_s4_class
vld_s4_class(numeric(0), "numeric")
vld_s4_class(getClass("MethodDefinition"), "classRepresentation")
```

chk_scalar	<i>Check Scalar</i>
------------	---------------------

Description

Checks if is a vector using
`length(x) == 1L`

Usage

```
chk_scalar(x, x_name = NULL)
```

```
vld_scalar(x)
```

Arguments

x	The object to check.
x_name	A string of the name of object x or NULL.

Value

The `chk_` function throws an informative error if the test fails.

The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_scalar`: Validate Scalar

See Also

Other `chk_` scalars: [chk_datetime\(\)](#), [chk_date\(\)](#), [chk_number\(\)](#), [chk_string\(\)](#), [chk_whole_number\(\)](#)

Examples

```
# chk_scalar
chk_scalar(1)
chk_scalar(list(1))
try(chk_scalar(1:2))

# vld_scalar
vld_scalar(1)
```

chk_setequal	<i>Check Set Equal</i>
--------------	------------------------

Description

Checks if equal set using
setequal(x, values)

Usage

```
chk_setequal(x, values, x_name = NULL)
vld_setequal(x, values)
```

Arguments

x	The object to check.
values	A vector of the permitted values.
x_name	A string of the name of object x or NULL.

Value

The `chk_` function throws an informative error if the test fails.
The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_setequal`: Validate Set Equal

See Also

Other `chk_set`: [chk_subset\(\)](#), [chk_superset\(\)](#)

Examples

```
# chk_setequal
chk_setequal(1:2, 2:1)
try(chk_setequal(1, 1:2))

# vld_setequal
vld_setequal(1, 1)
vld_setequal(1:2, 2:1)
vld_setequal(1, 2:1)
vld_setequal(1:2, 2)
```

chk_sorted	<i>Check Sorted</i>
------------	---------------------

Description

Checks if is sorted using
`is.unsorted(x)`

Usage

```
chk_sorted(x, x_name = NULL)
```

```
vld_sorted(x)
```

Arguments

x	The object to check.
x_name	A string of the name of object x or NULL.

Value

The `chk_` function throws an informative error if the test fails.

The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_sorted`: Validate Sorted

See Also

Other `chk_miscellaneous`: [chk_not_any_na\(\)](#), [chk_not_empty\(\)](#)

Examples

```
# chk_sorted
chk_sorted(1:2)
try(chk_sorted(2:1))

# vld_sorted
vld_sorted(1:2)
vld_sorted(2:1)
```

`chk_string`*Check String*

Description

Checks if string

```
is.character(x) && length(x) == 1L && !anyNA(x)
```

Usage

```
chk_string(x, x_name = NULL)
```

```
vld_string(x, x_name = NULL)
```

Arguments

`x` The object to check.

`x_name` A string of the name of object `x` or `NULL`.

Value

The `chk_` functions throw an informative error if the test fails.

The `vld_` functions return a flag indicating whether the test was met.

Functions

- `vld_string`: Validate String

See Also

Other `chk_` scalars: [chk_datetime\(\)](#), [chk_date\(\)](#), [chk_number\(\)](#), [chk_scalar\(\)](#), [chk_whole_number\(\)](#)

Examples

```
# chk_string
chk_string("1")
try(chk_string(1))

# vld_string
vld_string("1")
vld_string("")
vld_string(1)
vld_string(NA_character_)
vld_string(c("1", "1"))
```

chk_subset	<i>Check Subset</i>
------------	---------------------

Description

Checks if all values in values using
`all(x %in% values)`

Usage

```
chk_subset(x, values, x_name = NULL)
vld_subset(x, values)
```

Arguments

x	The object to check.
values	A vector of the permitted values.
x_name	A string of the name of object x or NULL.

Value

The `chk_` functions throw an informative error if the test fails.
The `vld_` functions return a flag indicating whether the test was met.

Functions

- `vld_subset`: Validate Subset

See Also

Other `chk_set`: [chk_setequal\(\)](#), [chk_superset\(\)](#)

Examples

```
# chk_subset
chk_subset(1, 1:10)
try(chk_subset(11, 1:10))

# vld_subset
vld_subset(numeric(0), 1:10)
vld_subset(1, 1:10)
vld_subset(11, 1:10)
```

chk_superset	<i>Check Superset</i>
--------------	-----------------------

Description

Checks if includes all values using
`all(values %in% x)`

Usage

```
chk_superset(x, values, x_name = NULL)
vld_superset(x, values)
```

Arguments

x	The object to check.
values	A vector of the permitted values.
x_name	A string of the name of object x or NULL.

Value

The `chk_` function throws an informative error if the test fails.
The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_superset`: Validates Superset

See Also

Other `chk_set`: [chk_setequal\(\)](#), [chk_subset\(\)](#)

Examples

```
# chk_superset
chk_superset(1:3, 1)
try(chk_superset(1:3, 4))

# vld_superset
vld_superset(1:3, 1)
vld_superset(1:3, 4)
vld_superset(integer(0), integer(0))
```

chk_true	<i>Check TRUE</i>
----------	-------------------

Description

Checks if TRUE using
`is.logical(x) && length(x) == 1L && !anyNA(x) && x`

Usage

```
chk_true(x, x_name = NULL)
vld_true(x)
```

Arguments

x	The object to check.
x_name	A string of the name of object x or NULL.

Value

The `chk_` functions throw an informative error if the test fails.
The `vld_` functions return a flag indicating whether the test was met.

Functions

- `vld_true`: Validate TRUE

See Also

Other `chk_logical`: [chk_false\(\)](#), [chk_flag\(\)](#), [chk_lgl\(\)](#)

Examples

```
# chk_true
chk_true(TRUE)
try(chk_true(1))

# vld_true
vld_true(TRUE)
vld_true(FALSE)
vld_true(NA)
vld_true(0)
vld_true(c(TRUE, TRUE))
```

`chk_unique`*Check Unique*

Description

Checks if unique using

```
!anyDuplicated(x, incomparables = incomparables)
```

Usage

```
chk_unique(x, incomparables = FALSE, x_name = NULL)
```

```
vld_unique(x, incomparables = FALSE)
```

Arguments

<code>x</code>	The object to check.
<code>incomparables</code>	A vector of values that cannot be compared. FALSE means that all values can be compared.
<code>x_name</code>	A string of the name of object x or NULL.

Value

The `chk_` functions throw an informative error if the test fails.

The `vld_` functions return a flag indicating whether the test was met.

Functions

- `vld_unique`: Validate Unique

See Also

Other `chk_misc`: [chk_match\(\)](#), [chk_named\(\)](#)

Examples

```
# chk_unique
chk_unique(c(NA, 2))
try(chk_unique(c(NA, NA, 2)))
chk_unique(c(NA, NA, 2), incomparables = NA)

# vld_unique
vld_unique(NULL)
vld_unique(numeric(0))
vld_unique(c(NA, 2))
vld_unique(c(NA, NA, 2))
vld_unique(c(NA, NA, 2), incomparables = NA)
```

`chk_unused`*Check ... Unused*

Description

Checks if ... is unused

```
length(list(...)) == 0L
```

Usage

```
chk_unused(...)
```

```
vld_unused(...)
```

Arguments

... Additional arguments.

Value

The `chk_` functions throw an informative error if the test fails.

The `vld_` functions return a flag indicating whether the test was met.

Functions

- `vld_unused`: Validate ... Unused

See Also

Other `chk_ellipsis`: [chk_used\(\)](#)

Examples

```
# chk_unused
fun <- function(x, ...) {
  chk_unused(...)
  x
}
fun(1)
try(fun(1, 2))

# vld_unused
fun <- function(x, ...) {
  vld_unused(...)
}
fun(1)
try(fun(1, 2))
```

`chk_used`*Check ... Used*

Description

Checks if is ... used using
`length(list(...)) != 0L`

Usage`chk_used(...)``vld_used(...)`**Arguments**

... Additional arguments.

Value

The `chk_` function throws an informative error if the test fails.

The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_used`: Validate ... Used

See Also

Other `chk_ellipsis`: [chk_unused\(\)](#)

Examples

```
# chk_used
fun <- function(x, ...) {
  chk_used(...)
  x
}
try(fun(1))
fun(1, 2)

# vld_used
fun <- function(x, ...) {
  vld_used(...)
}
fun(1)
fun(1, 2)
```

chk_vector	<i>Check Vector</i>
------------	---------------------

Description

Checks if is a vector using
`is.vector(x)`

Usage

```
chk_vector(x, x_name = NULL)
vld_vector(x)
```

Arguments

x	The object to check.
x_name	A string of the name of object x or NULL.

Value

The `chk_` function throws an informative error if the test fails.
The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_vector`: Validate Vector

See Also

Other `chk_is`: [chk_atomic\(\)](#), [chk_environment\(\)](#), [chk_function\(\)](#), [chk_list\(\)](#), [chk_numeric\(\)](#),
[chk_s3_class\(\)](#), [chk_s4_class\(\)](#), [chk_whole_numeric\(\)](#)

Examples

```
# chk_vector
chk_vector(1)
chk_vector(list())
try(chk_vector(matrix(1)))

# vld_vector
vld_vector(1)
```

chk_whole_number	<i>Check Whole Number</i>
------------------	---------------------------

Description

Checks if non-missing integer scalar or double equivalent using

```
vld_number(x) && (is.integer(x) || vld_true(all.equal(x, trunc(x))))
```

Good: 1, 2L, 1e10, -Inf

Bad: "a", 1:3, NA_integer_, log(10)

Usage

```
chk_whole_number(x, x_name = NULL)
```

```
vld_whole_number(x)
```

Arguments

x	The object to check.
x_name	A string of the name of object x or NULL.

Value

The `chk_` function throws an informative error if the test fails.

The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_whole_number`: Validate Whole Number

See Also

Other `chk_scalars`: [chk_datetime\(\)](#), [chk_date\(\)](#), [chk_number\(\)](#), [chk_scalar\(\)](#), [chk_string\(\)](#)

Examples

```
# chk_whole_number
chk_whole_number(2)
try(chk_whole_number(1.1))

# vld_whole_number
vld_whole_number(2)
```

chk_whole_numeric	<i>Check Whole Numeric</i>
-------------------	----------------------------

Description

Checks if integer vector or double equivalent using
`is.integer(x) || (is.double(x) && vld_true(all.equal(x, as.integer(x))))`

Usage

```
chk_whole_numeric(x, x_name = NULL)
```

```
vld_whole_numeric(x)
```

Arguments

x	The object to check.
x_name	A string of the name of object x or NULL.

Value

The `chk_` function throws an informative error if the test fails.

The `vld_` function returns a flag indicating whether the test was met.

Functions

- `vld_whole_numeric`: Validate Whole Numeric

See Also

Other `chk_is`: [chk_atomic\(\)](#), [chk_environment\(\)](#), [chk_function\(\)](#), [chk_list\(\)](#), [chk_numeric\(\)](#), [chk_s3_class\(\)](#), [chk_s4_class\(\)](#), [chk_vector\(\)](#)

Examples

```
# chk_whole_numeric
chk_whole_numeric(1)
try(chk_whole_numeric(1.1))

# vld_whole_numeric
vld_whole_numeric(1)
vld_whole_numeric(NA_real_)
vld_whole_numeric(1:2)
vld_whole_numeric(double(0))
vld_whole_numeric(TRUE)
vld_whole_numeric(1.5)
```

deparse_backtick *Deparse Backtick*

Description

deparse_backtick_chk is a wrapper on [deparse\(\)](#) and `backtick_chk`.

Usage

deparse_backtick(x)

deparse_backtick_chk(x)

backtick_chk(x)

unbacktick_chk(x)

Arguments

x A substituted object to deparse.

Details

It is exported to allow users to easily construct their own `chk_` functions.

Value

A string of the backticked substituted object.

Functions

- `deparse_backtick`: Deparse Backtick
Soft-deprecated
- `backtick_chk`: Backtick
- `unbacktick_chk`: Unbacktick

See Also

[deparse\(\)](#)

Examples

```
# deparse_backtick_chk
deparse_backtick_chk(2)
deparse_backtick_chk(2^2)
```

Description

The functions call `message_chk()` to process the message and then `rlang::abort()`, `rlang::warn()` and `rlang::inform()`, respectively.

Usage

```
err(..., n = NULL, tidy = TRUE, .subclass = NULL)
```

```
wrn(..., n = NULL, tidy = TRUE, .subclass = NULL)
```

```
msg(..., n = NULL, tidy = TRUE, .subclass = NULL)
```

Arguments

<code>...</code>	zero or more objects which can be coerced to character (and which are pasted together with no separator) or a single condition object.
<code>n</code>	The value of <code>n</code> for converting <code>sprintf</code> -like types.
<code>tidy</code>	A flag specifying whether capitalize the first character and add a missing period.
<code>.subclass</code>	This argument was renamed to <code>class</code> in <code>rlang</code> 0.4.2. It will be deprecated in the next major version. This is for consistency with our conventions for class constructors documented in https://adv-r.hadley.nz/s3.html#s3-subclassing .

Details

The user can set the subclass.

Functions

- `err`: Error
- `wrn`: Warning
- `msg`: Message

Examples

```
# err
try(err("there %r %n problem value%s", n = 2))

# wrn
wrn("there %r %n problem value%s", n = 2)

# msg
msg("there %r %n problem value%s", n = 2)
```

 message_chk

Construct Tidyverse Style Message

Description

If `tidy = TRUE` constructs a tidyverse style message by

Usage

```
message_chk(..., n = NULL, tidy = TRUE)
```

Arguments

<code>...</code>	Multiple objects that are converted to a string using <code>paste0(..., collapse = '')</code> .
<code>n</code>	The value of <code>n</code> for converting <code>sprintf</code> -like types.
<code>tidy</code>	A flag specifying whether capitalize the first character and add a missing period.

Details

- Capitalizing the first character if possible.
- Adding a trailing `.` if missing.

Also if `n != NULL` replaces the recognized `sprintf`-like types.

Value

A string of the message.

`sprintf`-like types

The following recognized `sprintf`-like types can be used in a message:

```
n The value of n.
s " if n == 1 otherwise 's'
r 'is' if n == 1 otherwise 'are'
y 'y' if n == 1 otherwise 'ie'
```

Examples

```
message_chk("there %r %n", " problem director%y%s")
message_chk("there %r %n", " problem director%y%s", n = 1)
message_chk("There %r %n", " problem director%y%s.", n = 3)
```

p *Concatenate Strings*

Description

A wrapper on `base::paste()`.

Usage

```
p(..., sep = " ", collapse = NULL)
```

```
p0(..., collapse = NULL)
```

Arguments

... one or more R objects, to be converted to character vectors.
 sep a character string to separate the terms. Not `NA_character_`.
 collapse an optional character string to separate the results. Not `NA_character_`.

Value

A character vector.

Functions

- `p0`: A wrapper on `base::paste0()`

Examples

```
p("a", "b")
p(c("a", "b"), collapse = " ")
p0("a", "b")
p0(c("a", "b"), collapse = "")
```

vld *Validators*

Description

Each `chk_()` function has a corresponding `vld_()` function.

Arguments

x The object to check.
 y An object to check against.
 vld_fun A `vld_` function.
 tolerance A non-negative numeric scalar.
 ... Additional arguments.

Value

A flag indicating whether the object passed the test.

Index

abort_chk, 3

backtick_chk (deparse_backtick), 52

base::paste(), 55

base::paste0(), 55

cc, 3

chk_all, 5, 6–8

chk_all_equal, 5, 6, 7, 8

chk_all_equivalent, 5, 6, 7, 8

chk_all_identical, 5–7, 8

chk_atomic, 9, 13, 20, 25, 35, 37, 38, 49, 51

chk_date, 10, 11, 34, 39, 42, 50

chk_datetime, 10, 11, 34, 39, 42, 50

chk_dir, 12, 16, 18

chk_environment, 9, 13, 20, 25, 35, 37, 38, 49, 51

chk_equal, 14, 15, 23

chk_equivalent, 14, 15, 23

chk_ext, 12, 16, 18

chk_false, 17, 19, 24, 45

chk_file, 12, 16, 18

chk_flag, 17, 19, 24, 45

chk_function, 9, 13, 20, 25, 35, 37, 38, 49, 51

chk_gt, 21, 22, 26, 27, 36

chk_gte, 21, 22, 26, 27, 36

chk_identical, 14, 15, 23

chk_lgl, 17, 19, 24, 45

chk_list, 9, 13, 20, 25, 35, 37, 38, 49, 51

chk_lt, 21, 22, 26, 27, 36

chk_lte, 21, 22, 26, 27, 36

chk_match, 28, 29, 46

chk_named, 28, 29, 46

chk_not_any_na, 30, 31, 41

chk_not_empty, 30, 31, 41

chk_not_null, 32, 33

chk_null, 32, 33

chk_number, 10, 11, 34, 39, 42, 50

chk_numeric, 9, 13, 20, 25, 35, 37, 38, 49, 51

chk_range, 21, 22, 26, 27, 36

chk_s3_class, 9, 13, 20, 25, 35, 37, 38, 49, 51

chk_s4_class, 9, 13, 20, 25, 35, 37, 38, 49, 51

chk_scalar, 10, 11, 34, 39, 42, 50

chk_setequal, 40, 43, 44

chk_sorted, 30, 31, 41

chk_string, 10, 11, 34, 39, 42, 50

chk_subset, 40, 43, 44

chk_superset, 40, 43, 44

chk_true, 17, 19, 24, 45

chk_unique, 28, 29, 46

chk_unused, 47, 48

chk_used, 47, 48

chk_vector, 9, 13, 20, 25, 35, 37, 38, 49, 51

chk_whole_number, 10, 11, 34, 39, 42, 50

chk_whole_numeric, 9, 13, 20, 25, 35, 37, 38, 49, 51

chkor, 4

deparse(), 52

deparse_backtick, 52

deparse_backtick_chk
(deparse_backtick), 52

err, 53

err(), 3

message_chk, 54

message_chk(), 53

msg(err), 53

NA_character_, 55

p, 55

p0(p), 55

rlang::abort(), 53

rlang::inform(), 53

rlang::warn(), 53

tolower(), 16

toupper(), 16

unbacktick_chk (deparse_backtick), 52

vld, 55

vld_all(chk_all), 5

vld_all_equal(chk_all_equal), 6

vld_all_equivalent
(chk_all_equivalent), 7

vld_all_identical (chk_all_identical), 8
vld_atomic (chk_atomic), 9
vld_date (chk_date), 10
vld_datetime (chk_datetime), 11
vld_dir (chk_dir), 12
vld_environment (chk_environment), 13
vld_equal (chk_equal), 14
vld_equivalent (chk_equivalent), 15
vld_ext (chk_ext), 16
vld_false (chk_false), 17
vld_file (chk_file), 18
vld_flag (chk_flag), 19
vld_function (chk_function), 20
vld_gt (chk_gt), 21
vld_gte (chk_gte), 22
vld_identical (chk_identical), 23
vld_lgl (chk_lgl), 24
vld_list (chk_list), 25
vld_lt (chk_lt), 26
vld_lte (chk_lte), 27
vld_match (chk_match), 28
vld_named (chk_named), 29
vld_not_any_na (chk_not_any_na), 30
vld_not_empty (chk_not_empty), 31
vld_not_null (chk_not_null), 32
vld_null (chk_null), 33
vld_number (chk_number), 34
vld_numeric (chk_numeric), 35
vld_range (chk_range), 36
vld_s3_class (chk_s3_class), 37
vld_s4_class (chk_s4_class), 38
vld_scalar (chk_scalar), 39
vld_setequal (chk_setequal), 40
vld_sorted (chk_sorted), 41
vld_string (chk_string), 42
vld_subset (chk_subset), 43
vld_superset (chk_superset), 44
vld_true (chk_true), 45
vld_unique (chk_unique), 46
vld_unused (chk_unused), 47
vld_used (chk_used), 48
vld_vector (chk_vector), 49
vld_whole_number (chk_whole_number), 50
vld_whole_numeric (chk_whole_numeric),
51

wrn (err), 53