Package 'checkr'

November 1, 2018

Title Check the Properties of Common R Objects

Version 0.4.0

Description Expressive, assertive, pipe-friendly functions
to check the properties of common R objects.
In the case of failure the functions issue informative error messages.

License MIT + file LICENSE

URL https://github.com/poissonconsulting/checkr

BugReports https://github.com/poissonconsulting/checkr/issues

Imports err

Suggests assertthat, checkmate, covr, datasets, magrittr, dplyr,
testthat, knitr, rmarkdown

Encoding UTF-8

LazyData true

RoxygenNote 6.1.0

VignetteBuilder knitr

Language en-CA

NeedsCompilation no

Author Joe Thorley [aut, cre] (<https://orcid.org/0000-0002-7683-4592>)

Maintainer Joe Thorley <joe@poissonconsulting.ca>

Repository CRAN

Date/Publication 2018-11-01 17:30:02 UTC

R topics documented:

`checkor` *Check OR*

Description

Checks that at least one check passes.

Usage

```
checkor(..., error = TRUE)
```

Arguments

`...`	The checks to check.
`error`	A flag indicating whether to throw an informative error or immediately generate an informative message if all checks fails.

Value

An invisible flag indicating whether at least one check passes (if it doesn't throw an error).

Examples

```
checkor(check_null(NULL), check_null(1), error = FALSE)
checkor(check_null(1), check_null(1), error = FALSE)
checkor(check_null(1), check_null(2), error = FALSE)
```

check_attributes *Check Attributes*

Description

Checks an objects attributes.

Usage

```
check_attributes(x, values = NULL, exclusive = FALSE, order = FALSE,
  x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
values	An optional vector or named list specifying the values.
exclusive	A flag indicating whether other elements are not permitted.
order	A flag indicating whether the elements have to occur in the same order as values.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_list

Examples

```
x <- 1
attributes(x) <- list(y = 2L)
check_attributes(x, values = list(y = 3:4), error = FALSE)
```

check_character *Check Character*

Description

Checks if x is a character vector with no attributes including names.

Usage

```
check_character(x, coerce = FALSE, x_name = substitute(x),
  error = TRUE)
```

Arguments

x	The object to check.
coerce	A flag indicating whether to coerce a factor to a character vector and drop attributes including names.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_int

Examples

```
check_character("1", error = FALSE)
check_character(1:2, error = FALSE)
```

check_chr *Check String*

Description

Checks if object is a string (non-missing character scalar with no attributes including names).

Usage

```
check_chr(x, coerce = FALSE, x_name = substitute(x), error = TRUE)

check_string(x, coerce = FALSE, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
coerce	A flag indicating whether to coerce a factor scalar to a string and drop attributes including names.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_scalar

Examples

```
check_chr(1, error = FALSE)
check_chr("1", error = FALSE)
check_chr(c("1", "2"), error = FALSE)
```

---

check_classes *Check Classes*

Description

Checks that an object inherits from one or more classes.

Usage

```
check_classes(x, classes = character(0), exclusive = FALSE,
  order = FALSE, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
classes	A character vector of the classes x should inherit from.
exclusive	A flag indicating whether other classes are not permitted.
order	A flag indicating whether the object classes have to occur in the same order as classes.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Details

The classes of an object can be returned using the 'class()' function.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_inherits

Examples

```
check_classes(list())
check_classes(list(), "list")
check_classes(list(), "numeric", error = FALSE)
```

---

check_colnames *Check Colnames*

Description

Checks the column names of a data frame as returned by the 'colnames()' function. The function can check the order of the columns and whether other columns are permitted.

Usage

```
check_colnames(x, colnames = character(0), exclusive = FALSE,
  order = FALSE, x_name = substitute(x), error = TRUE)
```

Arguments

x	The data to check.
colnames	A character vector of the column names.
exclusive	A flag indicating whether other columns are not permitted.
order	A flag indicating whether the columns have to occur in the same order as colnames.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_missing_colnames and check_data

Examples

```
data <- data.frame(x = 1, y = 2, z = 0)
check_colnames(data, c("y", "x"), error = FALSE)
check_colnames(data, c("y", "x"), exclusive = TRUE, error = FALSE)
check_colnames(data, c("y", "x"), order = TRUE, error = FALSE)
check_colnames(data, c("a"), error = FALSE)
```

---

check_count *Check Count*

Description

Checks if an object is a count (non-negative integer or if coerce = TRUE non-negative numeric whole number).

Usage

```
check_count(x, coerce = FALSE, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
coerce	A flag indicating whether to coerce a non-negative numeric (dbl) whole number to a count and drop attributes (including names).
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_scalar

Examples

```
check_count(-1L, error = FALSE)
check_count(1L, error = FALSE)
check_count(1, error = FALSE)
check_count(1, coerce = TRUE, error = FALSE)
check_count(1.01, coerce = TRUE, error = FALSE)
```

check_data *Check Data*

Description

Checks whether an object is a data frame. Can also check the number of rows, the names and order and values of the columns as well as whether particular columns form a unique key.

Usage

```
check_data(x, values = NULL, nrow = NA, exclusive = FALSE,
  order = FALSE, key = character(0), x_name = substitute(x),
  error = TRUE)
```

Arguments

x	The object to check.
values	NULL (default) or a character vector specifying the column names or a named list specifying the column names and values.
nrow	A flag indicating whether x should have rows (versus no rows) or a missing value indicating no requirements or a count or count range of the number of rows.
exclusive	A flag indicating whether other columns are not permitted.
order	A flag indicating whether the columns have to occur in the same order as values.
key	A character vector of the columns that represent a unique key.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_colnames, check_nrow and check_key

Examples

```
z <- data.frame(
 Count = c(0L, 3L, 3L, 0L, NA),
  Longitude = c(0, 0, 90, 90, 180),
  Latitude = c(0, 90, 90.2, 100, -180),
  Type = factor(c("Good", "Bad", "Bad", "Bad", "Bad"), levels = c("Good", "Bad")),
  Extra = TRUE,
  Comments = c("In Greenwich", "Somewhere else", "I'm lost",
    "I didn't see any", "Help"),
```

```
  stringsAsFactors = FALSE)

check_data(z, values = list(
  Count = 1,
  Extra = NA,
  Latitude = c(45, 90)
  ), exclusive = TRUE, order = TRUE, nrow = 10L, key = "Longitude", error = FALSE)
```

---

check_date *Check Date*

Description

Checks if x is a date (non-missing unnamed Date scalar).

Usage

```
check_date(x, coerce = FALSE, x_name = substitute(x), error = TRUE)

check_day(x, coerce = FALSE, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
coerce	A flag indicating whether to coerce a date time (POSIXt scalar) to a Date and remove names.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_datetime

Examples

```
check_date(Sys.Date(), error = FALSE)
check_date(Sys.time(), error = FALSE)
check_date(Sys.time(), coerce = TRUE, error = FALSE)
```

check_dbl *Check Dbl*

Description

Checks if x is a dbl (non-missing numeric scalar with no attributes including names).

Usage

```
check_dbl(x, coerce = FALSE, x_name = substitute(x), error = TRUE)

check_number(x, coerce = FALSE, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
coerce	A flag indicating whether to coerce a scalar integer to a dbl and drop attributes including names.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_scalar

Examples

```
check_dbl(1, error = FALSE)
check_dbl(1L, error = FALSE)
check_dbl(c(1,2), error = FALSE)
```

check_dttm *Check Date Time*

Description

Checks if x is a datetime (non-missing unnamed POSIXct scalar).

Usage

```
check_dttm(x, coerce = FALSE, tzone = "UTC", x_name = substitute(x),
  error = TRUE)

check_datetime(x, coerce = FALSE, tzone = "", x_name = substitute(x),
  error = TRUE)
```

Arguments

x	The object to check.
coerce	A flag indicating whether to coerce a date to a dttm (using the time zone tzone) and remove names.
tzone	A string of the time zone where "" is the current time zone.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_scalar

Examples

```
check_dttm(Sys.Date(), error = FALSE)
check_dttm(Sys.time(), error = FALSE)
```

check_environment *Check Environment*

Description

Checks if x is an environment.

Usage

```
check_environment(x, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

Examples

```
check_environment(1, error = FALSE)
check_environment(.GlobalEnv, error = FALSE)
```

check_flag_na *Check Flag or NA*

Description

Checks if x is a flag or NA (missing logical scalar).

Usage

```
check_flag_na(x, coerce = TRUE, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
coerce	A flag indicating whether to drop attributes including names.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Details

Useful when using flag to pass one of three options.

Value

An invisible copy of x (if it doesn't throw an error).

---

check_function *Check Function*

Description

Checks if x is a function.

Usage

```
check_function(x, nargs = NA, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
nargs	A count of the number of arguments or count range of the minimum and maximum number of arguments.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

Examples

```
check_function(character, error = FALSE)
check_function(character, nargs = 0L, error = FALSE)
```

check_grepl *Check Matches Regular Expression*

Description

Checks whether all the elements of an object match a regular expression.

Usage

```
check_grepl(x, pattern = ".*", regex = pattern,
  x_name = substitute(x), error = TRUE)

check_regex(x, regex = ".*", x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
pattern	A string of the regular expression.
regex	A string of the regular expression (deprecated for pattern).
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_nchar and check_pattern

Examples

```
check_grepl("foo", "fo")
check_grepl("foo", "fo$", error = FALSE)
```

check_homogenous *Check Homogenous*

Description

Checks whether the elements of x are all of the same class. It works on vectors, matrices and arrays which, by definition will always be homogenous and lists and data frames which may or may not be homogenous.

Usage

```
check_homogenous(x, strict = FALSE, recursive = FALSE,
  x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
strict	A flag indicating whether all the objects must have identical classes or just share one or more classes.
recursive	A flag indicating whether the check should be applied recursively.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_vector, check_list and check_data

Examples

```
check_homogenous(1:2)
check_homogenous(list(1,2))
check_homogenous(list(1,TRUE), error = FALSE)
```

check_inherits *Check Inherits*

Description

Checks if an object inherits from a class.

Usage

```
check_inherits(x, class, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
class	A string of the class x should inherit from.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_classes

Examples

```
check_inherits(list(), "list")
check_inherits(list(), "numeric", error = FALSE)
```

check_int *Check Int*

Description

Checks if x is a int (non-missing integer scalar with no attributes including names).

Usage

```
check_int(x, coerce = FALSE, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
coerce	A flag indicating whether to coerce a numeric (dbl) whole number to an int and drop attributes including names.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_scalar

Examples

```
check_int(1, error = FALSE)
check_int(1L, error = FALSE)
check_int(1:2, error = FALSE)
```

---

check_integer *Check Integer*

Description

Checks if x is an integer vector with no attributes including names.

Usage

```
check_integer(x, coerce = FALSE, x_name = substitute(x),
  error = TRUE)
```

Arguments

x	The object to check.
coerce	A flag indicating whether to coerce a numeric (dbl) whole number vector to an integer vector and drop attributes including names.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_int

Examples

```
check_integer(1, error = FALSE)
check_integer(1L, error = FALSE)
check_integer(1:2, error = FALSE)
```

---

check_intersection *Check Atomic Vector Intersection*

Description

Checks that all the elements in atomic vector x intersect with those in atomic vector y.

Usage

```
check_intersection(x, y, all_y = FALSE, x_name = substitute(x),
  y_name = substitute(y), error = TRUE)
```

Arguments

x	The object to check.
y	The second atomic vector.
all_y	A flag indicating whether all the elements in y should have a match in x.
x_name	A string of the name of the object x.
y_name	A string of the name of the object y.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_join

Examples

```
x1 <- 1:3
x2 <- 1:4
check_intersection(x1, x2)
check_intersection(x2, x1, error = FALSE)
```

check_join *Check Join*

Description

Checks that the columns in data frame x form a many-to-one join with the corresponding columns in y, ie, the join is a unique key in y and all the rows in x have a match in y.

Usage

```
check_join(x, y, by = NULL, all_y = FALSE, x_name = substitute(x),
  y_name = substitute(y), error = TRUE)
```

Arguments

x	The object to check.
y	The parent data frame.
by	A character vector or named character vector of the columns to join by.
all_y	A flag indicating whether all the rows in y should have a match in x.
x_name	A string of the name of the object x.
y_name	A string of the name of the object y.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_data

Examples

```
data1 <- data.frame(x = 1:2)
data2 <- data.frame(x = 3:5, y = 2L)
check_join(data1, data2, error = FALSE)
check_join(data1, data2, by = c(x = "y"), error = FALSE)
```

check_key *Check Key*

Description

Checks that columns in a data frame represent a unique key. By default all the columns are checked.

Usage

```
check_key(x, key = names(x), x_name = substitute(x), error = TRUE)
```

Arguments

x	The data to check.
key	A character vector of the column names representing the key.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_data

Examples

```
data <- data.frame(x = 1:1, y = 1:2)
check_key(data, "x", error = FALSE)
check_key(data, c("y", "x"), error = FALSE)
```

check_length *Check Length*

Description

Checks whether the number of elements in an object is an exact number, within a range or 0 vs positive.

Usage

```
check_length(x, length = TRUE, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
length	A flag indicating whether x should have elements (versus no elements) or a missing value indicating no requirements or a count or count range of the number of elements or a count vector of the permitted number of elements.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_vector, check_list and check_data

Examples

```
check_length(2)
check_length(character(0), length = 0)
check_length(NULL, error = FALSE)
check_length(list(), error = FALSE)
```

---

check_length1 *Check Length One*

Description

Checks whether x is an object of length 1.

Usage

```
check_length1(x, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

Examples

```
check_length1(2)
check_length1(1:2, error = FALSE)
check_length1(NULL, error = FALSE)
check_length1(list(), error = FALSE)
```

`check_levels` *Check Levels*

Description

Checks the levels in a factor including the order and whether other levels are permitted.

Usage

```
check_levels(x, levels, exclusive = TRUE, order = TRUE,
  x_name = substitute(x), error = TRUE)
```

Arguments

`x`	The object to check.
`levels`	A character vector of the levels.
`exclusive`	A flag indicating whether other levels are not permitted.
`order`	A flag indicating whether the object levels have to occur in the same order as names. To check whether x is an ordered factor use `check_vector(x, ordered(1))`.
`x_name`	A string of the name of the object.
`error`	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

`check_nlevels` and `check_vector`

Examples

```
check_levels(1, c("x", "y"), error = FALSE)
check_levels(factor(1), c("x", "y"), error = FALSE)
```

check_lgl *Check Flag*

Description

Checks if x is a flag (non-missing logical scalar with no attributes including names).

Usage

```
check_lgl(x, coerce = FALSE, x_name = substitute(x), error = TRUE)

check_flag(x, coerce = FALSE, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
coerce	A flag indicating whether to drop attributes including names.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_scalar

Examples

```
check_lgl(1, error = FALSE)
check_lgl(FALSE, error = FALSE)
check_lgl(c(FALSE, TRUE), error = FALSE)
```

check_list *Check List*

Description

Checks whether an object is a list and optionally the names and values of its elements.

Usage

```
check_list(x, values = NULL, length = NA, unique = FALSE,
  named = NA, exclusive = FALSE, order = FALSE,
  x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
values	An optional vector or named list specifying the values.
length	A flag indicating whether x should have elements (versus no elements) or a missing value indicating no requirements or a count or count range of the number of elements or a count vector of the permitted number of elements.
unique	A flag indicating whether the values must be unique.
named	A flag indicating whether the list must be named or unnamed or a regular expression that must match all the names or count or count range of the number of characters in the names or NA if it doesn't matter if the list is named.
exclusive	A flag indicating whether other elements are not permitted.
order	A flag indicating whether the elements have to occur in the same order as values.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_length and check_unique

Examples

```
check_list(list())
check_list(list(x1 = 2, x2 = 1:2), values = list(x1 = 1, x2 = 1L))
```

---

check_logical *Check Logical*

Description

Checks if x is a logical vector with no attributes including names.

Usage

```
check_logical(x, coerce = FALSE, x_name = substitute(x),
  error = TRUE)
```

Arguments

x	The object to check.
coerce	A flag indicating whether to drop attributes including names.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_scalar

Examples

```
check_logical(1, error = FALSE)
check_logical(FALSE, error = FALSE)
check_logical(c(FALSE, TRUE), error = FALSE)
```

---

check_missing_colnames

Check Missing Colnames

Description

Checks whether specific colnames are missing from a data frame.

Usage

```
check_missing_colnames(x, colnames, x_name = substitute(x),
  error = TRUE)
```

Arguments

x	The data to check.
colnames	A character vector of the column names that must be missing from x.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_colnames and check_data

Examples

```
data <- data.frame(x = 1, y = 2, z = 0)
check_missing_colnames(data, c("y", "x", "a"), error = FALSE)
check_missing_colnames(data, "a", error = FALSE)
```

---

check_missing_names *Check Missing Names*

Description

Checks whether specific names are missing from an object.

Usage

```
check_missing_names(x, names, x_name = substitute(x), error = TRUE)
```

Arguments

x	The named object to check.
names	A character vector of the names that must be missing from x.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_names

Examples

```
vec <- c(x = 1, y = 2, z = 0)
check_missing_names(vec, c("y", "x", "a"), error = FALSE)
check_missing_names(vec, "a", error = FALSE)
```

check_named *Check Named*

Description

Checks whether an object is named.

Usage

```
check_named(x, nchar = c(0L, chk_max_int()), pattern = ".*",
  regex = pattern, unique = FALSE, x_name = substitute(x),
  error = TRUE)
```

Arguments

x	The object to check.
nchar	A count or count range of the number of characters.
pattern	A string of the regular expression that must match all names.
regex	A string of the regular expression that must match all names.
unique	A flag indicating whether the names must be unique.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_unnamed, check_names and check_missing_names

Examples

```
check_named(2, error = FALSE)
x <- 1
names(x) <- "y"
check_named(x, error = FALSE)
```

check_names *Check Names*

Description

Checks the names of an object as returned by the 'names()' function. The function can check the order of the names and whether other names are permitted.

Usage

```
check_names(x, names = character(0), exclusive = FALSE,
  order = FALSE, unique = FALSE, complete = TRUE,
  x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
names	A character vector of the names.
exclusive	A flag indicating whether other names are not permitted.
order	A flag indicating whether the object names have to occur in the same order as names.
unique	A flag indicating whether all the object names have to be unique.
complete	A flag indicating whether all the possible names have to be represented.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_named

Examples

```
vec <- c(x = 1, y = 2, z = 0)
check_names(vec, c("y", "x"), error = FALSE)
check_names(vec, c("y", "x"), exclusive = TRUE, error = FALSE)
check_names(vec, c("y", "x"), order = TRUE, error = FALSE)
check_names(vec, c("a"), error = FALSE)
```

check_nchar *Check Number of Characters*

Description

Checks the number of characters in the elements of an object.

Usage

```
check_nchar(x, nchar = TRUE, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
nchar	A flag indicating whether x should have characters or a missing value indicating no requirements or a count or count range of the number of characters.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_pattern and check_regex

Examples

```
check_nchar(c("foo", "bar"), nchar = 3)
```

check_ncol *Check Number of Columns*

Description

Checks the number of columns of a data frame.

Usage

```
check_ncol(x, ncol = TRUE, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
ncol	A flag indicating whether x should have columns (versus no columns) or a missing value indicating no requirements or a count or count range of the number of columns.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_nrow and check_data

Examples

```
check_ncol(data.frame(x = 1), error = FALSE)
check_ncol(data.frame(x = 1:2), ncol = 1, error = FALSE)
```

---

check_neg_dbl *Check Negative Dbl*

Description

Checks if x is a negative dbl (non-missing numeric scalar with no attributes including names).

Usage

```
check_neg_dbl(x, coerce = FALSE, x_name = substitute(x),
  error = TRUE)
```

Arguments

x	The object to check.
coerce	A flag indicating whether to coerce a scalar integer to a dbl and drop attributes including names.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_dbl

Examples

```
check_neg_dbl(1, error = FALSE)
check_neg_dbl(0L, error = FALSE)
check_neg_dbl(0, error = FALSE)
```

---

check_neg_int *Check Negative Int*

Description

Checks if x is a negative int (non-missing integer scalar with no attributes including names).

Usage

```
check_neg_int(x, coerce = FALSE, x_name = substitute(x),
  error = TRUE)
```

Arguments

x	The object to check.
coerce	A flag indicating whether to coerce a numeric (dbl) whole number to an int and drop attributes including names.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_int

Examples

```
check_neg_int(0L, error = FALSE)
check_neg_int(-1L, error = FALSE)
```

check_nlevels *Check nlevels*

Description

Checks the number of levels of an object.

Usage

```
check_nlevels(x, nlevels = TRUE, x_name = substitute(x),
  error = TRUE)
```

Arguments

x	The data to check.
nlevels	A flag indicating whether x should have elements (versus no elements) or a missing value indicating no requirements or a count or count range of the number of elements.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_levels and check_vector

Examples

```
check_nlevels(factor(1), error = FALSE)
check_nlevels(factor(1), nlevels = 2, error = FALSE)
```

check_noneg_dbl *Check Non-Negative Dbl*

Description

Checks if x is a non-negative dbl (non-missing numeric scalar with no attributes including names).

Usage

```
check_noneg_dbl(x, coerce = FALSE, x_name = substitute(x),
  error = TRUE)
```

Arguments

x	The object to check.
coerce	A flag indicating whether to coerce a scalar integer to a dbl and drop attributes including names.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_dbl

Examples

```
check_noneg_dbl(1, error = FALSE)
check_noneg_dbl(0L, error = FALSE)
check_noneg_dbl(0, error = FALSE)
```

---

check_noneg_int *Check Non-Negative Int*

Description

Checks if x is a count (non-missing non-negative integer scalar with no attributes including names).

Usage

```
check_noneg_int(x, coerce = FALSE, x_name = substitute(x),
  error = TRUE)
```

Arguments

x	The object to check.
coerce	A flag indicating whether to coerce a non-negative numeric (dbl) whole number to a count and drop attributes (including names).
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_int

Examples

```
check_noneg_int(1, error = FALSE)
check_noneg_int(0L, error = FALSE)
check_noneg_int(1L, error = FALSE)
```

---

check_no_attributes *Check No Attributes*

Description

Checks an object has no attributes.

Usage

```
check_no_attributes(x, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_attributes

Examples

```
x <- 1
attributes(x) <- list(y = 2L)
check_no_attributes(x, error = FALSE)
```

check_nrow *Check Number of Rows*

Description

Checks the number of rows of a data frame.

Usage

```
check_nrow(x, nrow = TRUE, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
nrow	A flag indicating whether x should have rows (versus no rows) or a missing value indicating no requirements or a count or count range of the number of rows.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_ncol and check_data

Examples

```
check_nrow(data.frame(x = 1), error = FALSE)
check_nrow(data.frame(x = integer(0)), error = FALSE)
check_nrow(data.frame(x = 1:2), nrow = 1, error = FALSE)
```

check_null *Check NULL*

Description

Checks whether an object is NULL.

Usage

```
check_null(x, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

Examples

```
check_null(1, error = FALSE)
check_null(NULL, error = FALSE)
```

---

check_numeric *Check Numeric*

Description

Checks if x is an numeric (double) vector with no attributes including names.

Usage

```
check_numeric(x, coerce = FALSE, x_name = substitute(x),
  error = TRUE)

check_double(x, coerce = FALSE, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
coerce	A flag indicating whether to coerce a integer vector to an double vector and drop attributes including names.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_number

Examples

```
check_numeric(1, error = FALSE)
check_numeric(1L, error = FALSE)
check_numeric(1:2, error = FALSE)
```

---

check_pattern *Check Pattern*

Description

Checks whether all or some of the elements of x match pattern using grepl.

Usage

```
check_pattern(x, pattern, all = TRUE, x_name = substitute(x),
  error = TRUE)
```

Arguments

x	The object to check.
pattern	A string of the regular expression.
all	A flag indicating whether all or some of the element must match pattern.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_nchar and check_regex

check_pos_dbl *Check Positive Dbl*

Description

Checks if x is a positive dbl (non-missing numeric scalar with no attributes including names).

Usage

```
check_pos_dbl(x, coerce = FALSE, x_name = substitute(x),
  error = TRUE)
```

Arguments

x	The object to check.
coerce	A flag indicating whether to coerce a scalar integer to a dbl and drop attributes including names.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_dbl

Examples

```
check_pos_dbl(1, error = FALSE)
check_pos_dbl(0L, error = FALSE)
check_pos_dbl(0, error = FALSE)
```

check_pos_int *Check Positive Int*

Description

Checks if x is a positive int (non-missing integer scalar with no attributes including names).

Usage

```
check_pos_int(x, coerce = FALSE, x_name = substitute(x),
  error = TRUE)
```

Arguments

x	The object to check.
coerce	A flag indicating whether to coerce a numeric (dbl) whole number to an int and drop attributes including names.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_int

Examples

```
check_pos_int(0, error = FALSE)
check_pos_int(1L, error = FALSE)
check_pos_int(1:2, error = FALSE)
```

---

check_prob *Check Probability*

Description

Checks if x is a probability (non-missing dbl between 0 and 1 inclusive with no attributes including names).

Usage

```
check_prob(x, coerce = FALSE, x_name = substitute(x), error = TRUE)

check_probability(x, coerce = FALSE, x_name = substitute(x),
  error = TRUE)

check_prop(x, coerce = FALSE, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
coerce	A flag indicating whether to coerce an integer to a dbl and drop attributes including names.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_scalar

Examples

```
check_prob(1, error = FALSE)
check_prob(1.1, error = FALSE)
check_prob(c(0, 1), error = FALSE)
```

---

check_props *Check Proportions*

Description

Checks if x is proportions vector - non-missing dbls between 0 and 1 that sum to 1.

Usage

```
check_props(x, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

check_rbind *Check Row Bind*

Description

Checks whether a data frame has the same columns of the same classes as a second data frame which means they can be rbinded without a problem.

Usage

```
check_rbind(x, y, exclusive = TRUE, order = FALSE,
  x_name = substitute(x), y_name = substitute(y), error = TRUE)
```

Arguments

x	The first data frame.
y	The second data frame.
exclusive	A flag indicating whether other columns are not permitted.
order	A flag indicating whether the columns have to occur in the same order.
x_name	A string of the name of the object x.
y_name	A string of the name of the object y.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_join

Examples

```
check_rbind(datasets::mtcars, datasets::mtcars)
```

check_scalar *Check Scalar*

Description

Checks whether an object is an atomic vector with one element.

Usage

```
check_scalar(x, values = NULL, named = FALSE, attributes = named,
  only = FALSE, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
values	NULL or a vector specifying the values.
named	A flag indicating whether the scalar must be named or unnamed or NA if it doesn't matter if the scalar is named.
attributes	A flag indicating whether the scalar must or must not have attributes or NA if it doesn't matter if the scalar is named.
only	A flag indicating whether only the actual values are permitted. It only affects values with two or less non-missing elements.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_vector

Examples

```
check_scalar(1)
check_scalar(c(1,2), error = FALSE)
check_scalar(1, c(2,3), error = FALSE)
```

check_sorted *Check Sorted*

Description

Checks whether object x is sorted using `!is.unsorted(x, na.rm = TRUE)`.

Usage

```
check_sorted(x, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_vector and check_list

Examples

```
check_sorted(1:2, error = FALSE)
check_sorted(2:1, error = FALSE)
```

check_tzone *Check TimeZone*

Description

Checks an objects tzone attribute.

Usage

```
check_tzone(x, tzone = "UTC", x_name = substitute(x), error = TRUE)

check_tz(x, tz = "UTC", x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
tzone	A string of the time zone.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.
tz	A string of the time zone.

Value

An invisible copy of x (if it doesn't throw an error).

Examples

```
check_tzone(Sys.Date(), error = FALSE)
x <- as.POSIXct("2000-01-02 03:04:55", tz = "Etc/GMT+8")
check_tzone(x, tzone = "PST8PDT", error = FALSE)
```

---

check_unique *Check Unique*

Description

Checks whether all elements of an object are unique.

Usage

```
check_unique(x, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_vector

Examples

```
check_unique(2, error = FALSE)
check_unique(c(2,2), error = FALSE)
check_unique(1:2, error = FALSE)
check_unique(character(0), error = FALSE)
check_unique(NULL, error = FALSE)
check_unique(list(), error = FALSE)
```

---

check_unnamed *Check Unnamed*

Description

Checks whether an objects is unnamed.

Usage

```
check_unnamed(x, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

See Also

check_named, check_names and check_missing_names

Examples

```
check_unnamed(2, error = FALSE)
x <- 1
names(x) <- "y"
check_unnamed(x, error = FALSE)
```

check_unused *Check Unused*

Description

Checks whether ... is unused. It can only be used in functions.

Usage

```
check_unused(..., x_name = "...", error = TRUE)
```

Arguments

...	The arguments to check.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

Examples

```
fun <- function(...) check_unused(..., error = FALSE)
fun()
fun(1)
```

check_vector *Check Atomic Vector*

Description

Check Atomic Vector

Usage

```
check_vector(x, values = NULL, length = NA, unique = FALSE,
  sorted = FALSE, named = NA, attributes = named, only = FALSE,
  x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
values	NULL or a vector specifying the values.
length	A flag indicating whether x should have elements (versus no elements) or a missing value indicating no requirements or a count or count range of the number of elements or a count vector of the permitted number of elements.
unique	A flag indicating whether the values must be unique.
sorted	A flag indicating whether the vector must be sorted.
named	A flag indicating whether the vector must be named or unnamed or NA if it doesn't matter.
attributes	A flag indicating whether the vector must or must not have attributes or NA if it doesn't matter.
only	A flag indicating whether only the actual values are permitted. It only affects values with two or less non-missing elements.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (if it doesn't throw an error).

Examples

```
check_vector(2:1, length = 3, sorted = TRUE, named = TRUE, error = FALSE)
check_vector(c("one", "two", "four"), values = c("one", "two", "two"), error = FALSE)
```

---

chk_deparse *Deparse*

Description

chk_deparse is a wrapper on deparse that sets a missing value to be "NA"

Usage

```
chk_deparse(x)
```

Arguments

x	A substituted object to deparse

Value

A string

See Also

deparse

Examples

```
chk_deparse(1^2)
```

`chk_fail` *Fail*

Description

Fail

Usage

```
chk_fail(..., error)
```

Arguments

`...`	The message.
`error`	A flag indicating whether to return an error (the default) or a warning.

`chk_max_dbl` *Max Double*

Description

Max Double

Usage

```
chk_max_dbl()
```

Value

An dbl of the maximum numeric value for the system.

Examples

```
chk_max_dbl()
```

chk_max_int *Max Int*

Description

Max Int

Usage

```
chk_max_int()
```

Value

An int of the maximum integer value for the system.

Examples

```
chk_max_int()
```

chk_min_dbl *Min Double*

Description

Min Double

Usage

```
chk_min_dbl()
```

Value

An dbl of the minimum numeric value for the system.

Examples

```
chk_min_dbl()
```

chk_min_int *Min Integer*

Description

Min Integer

Usage

```
chk_min_int()
```

Value

An int of the minimum integer value for the system.

Examples

```
chk_min_int()
```

chk_tiny_dbl *Tiny Positive Double*

Description

Tiny Positive Double

Usage

```
chk_tiny_dbl()
```

Value

An dbl of the tiniest positive numeric value for the system.

Examples

```
chk_tiny_dbl()
```

Index