Package 'checkr'

January 29, 2018

Title Check Object Classes, Values, Names and Dimensions

Version 0.0.2

Date 2018-01-26

Description Checks the classes, values, names and dimensions of
scalar, vectors, lists and data frames.
Issues an informative error (or warning) if checks fail.
Otherwise it returns the original object allowing it to be used in pipes.

License MIT + file LICENSE

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

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

Depends R (>= 3.1.0)

Suggests covr, magrittr, dplyr, testthat

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

NeedsCompilation no

Author Joe Thorley [aut, cre]

Maintainer Joe Thorley <joe@poissonconsulting.ca>

Repository CRAN

Date/Publication 2018-01-29 10:15:11 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_classes *Check Classes*

Description

Check 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 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.

Value

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

Examples

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

check_colnames *Check Colnames*

Description

Check Colnames

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 permitted.
order	A flag indicating whether the columns have to occur in the same order as 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).

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 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 (real) whole number to a count.
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_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

Check Data

Usage

```
check_data(x, values, nrow = c(0L, 2147483647L), exclusive = FALSE,
  order = FALSE, key = character(0), x_name = substitute(x),
  error = TRUE)
```

Arguments

x	The object to check.
values	An optional character vector specifying the column names or a named list specifying the column names and values.
nrow	A count or count range of the number of rows.
exclusive	A flag indicating whether other columns are 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).

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 Date scalar).

Usage

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

check_datetime *Check Date Time*

Description

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

Usage

```
check_datetime(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_datetime(Sys.Date(), error = FALSE)
check_datetime(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` *Check Flag*

Description

Checks if x is a flag (non-missing logical scalar).

Usage

```
check_flag(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_flag(1, error = FALSE)
check_flag(FALSE, error = FALSE)
check_flag(c(FALSE, TRUE), error = FALSE)
```

check_function *Check Function*

Description

Checks if x is a function.

Usage

```
check_function(x, nargs = c(0L, .Machine$integer.max),
  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_inherits *Check Inherits*

Description

Check Inherits

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).

Examples

```
check_inherits(list(), "list")
check_inherits(list(), "numeric", 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.

Usage

```
check_join(x, y, by = NULL, 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.
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).

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.

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.

Details

By default all the columns are checked.

Value

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

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

Check Length

Usage

```
check_length(x, length = c(1L, .Machine$integer.max),
  x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
length	A count of the length or count range of the minimum and maximum length.
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_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

Check Levels

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 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).

Examples

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

check_list *Check List*

Description

Check List

Usage

```
check_list(x, values, length = c(0L, .Machine$integer.max), unique = FALSE,
  sorted = FALSE, named = NA, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
values	An optional vector or named list specifying the values.
length	A count of the length or count range of the minimum and maximum length.
unique	A flag indicating whether the values must be unique.
sorted	A flag indicating whether the list must be sorted.
named	A flag (or NA) indicating whether the list must be named or unnamed (or doesn't matter).
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_missing_colnames *Check Missing Colnames*

Description

Check Missing Colnames

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).

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

Check Missing Names

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).

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

Check Named

Usage

```
check_named(x, unique = FALSE, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
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).

Examples

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

check_names *Check Names*

Description

Check Names

Usage

```
check_names(x, names = character(0), exclusive = FALSE, order = FALSE,
  unique = FALSE, 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 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.
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

```
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_ncol *Check Number of Columns*

Description

Check Number of Columns

Usage

```
check_ncol(x, ncol = c(1L, 2147483647L), x_name = substitute(x),
  error = TRUE)
```

Arguments

x	The data to check.
ncol	A count of the number of columns or a count range of the minimum and maximum 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).

Examples

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

`check_nlevels` *Check nlevels*

Description

Check nlevels

Usage

```
check_nlevels(x, nlevels = c(1L, .Machine$integer.max),
  x_name = substitute(x), error = TRUE)
```

Arguments

`x`	The data to check.
`nlevels`	A count of the number of rows or a count range of the minimum and maximum 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).

Examples

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

check_nrow *Check nrow*

Description

Check nrow

Usage

```
check_nrow(x, nrow = c(1L, 2147483647L), x_name = substitute(x),
  error = TRUE)
```

Arguments

x	The data to check.
nrow	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).

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

Check 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_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).

Examples

```
check_pattern("b ", "b", error = FALSE)
check_pattern("b ", "^b$", error = FALSE)
```

check_probability *Check Probability*

Description

Checks if x is a probability (non-missing numeric scalar between 0 and 1 inclusive).

Usage

```
check_probability(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 numeric.
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_probability(1, error = FALSE)
check_probability(1.1, error = FALSE)
check_probability(c(0, 1), error = FALSE)
```

check_scalar *Check Scalar*

Description

Check Scalar

Usage

```
check_scalar(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_sorted` *Check Sorted*

Description

Checks whether object x is sorted.

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.

Details

Uses `!is.unsorted(x, na.rm = TRUE)` to test.

Value

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

Examples

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

check_string *Check String*

Description

Checks if object is a string (non-missing character scalar or if `coerce = TRUE` a non-missing factor scalar).

Usage

```
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.
`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_string(1, error = FALSE)
check_string("1", error = FALSE)
check_string(c("1", "2"), error = FALSE)
```

check_tz *Check TimeZone*

Description

Check TimeZone

Usage

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

Arguments

x	The object to check.
tz	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.

Value

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

Examples

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

---

check_unique *Check Unique*

Description

Check 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).

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

Check 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).

Examples

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

check_vector *Check Vector*

Description

Check Vector

Usage

```
check_vector(x, values, length = c(0L, .Machine$integer.max),
  unique = FALSE, sorted = FALSE, named = NA, x_name = substitute(x),
  error = TRUE)
```

Arguments

x	The object to check.
values	An optional vector specifying the values.
length	A count of the length or count range of the minimum and maximum length.
unique	A flag indicating whether the values must be unique.
sorted	A flag indicating whether the vector must be sorted.
named	A flag (or NA) indicating whether the vector must be named or unnamed (or doesn't matter).
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)
```

Index