Package ‘assertive.datetimes’

February 2, 2017

Type Package
Title Assertions to Check Properties of Dates and Times
Version 0.0-2
Date 2016-05-08
Author Richard Cotton [aut, cre]
Maintainer Richard Cotton <richierocks@gmail.com>
Description A set of predicates and assertions for checking the properties of
dates and times. This is mainly for use by other package developers who
want to include run-time testing features in their own packages. End-users
will usually want to use assertive directly.

URL https://bitbucket.org/richierocks/assertive.datetimes

BugReports https://bitbucket.org/richierocks/assertive.datetimes/issues

Depends R (>= 3.0.0)
Imports assertive.base (>= 0.0-2), assertive.types
Suggests testthat
License GPL (>= 3)

LazyLoad yes

LazyData yes

Acknowledgments Development of this package was partially funded by
the Proteomics Core at Weill Cornell Medical College in Qatar
<http://qatar-weill.cornell.edu>. The Core is supported by
‘Biomedical Research Program’ funds, a program funded by Qatar
Foundation.

RoxygenNote 5.0.1

‘is-date-string.R’ ‘workaround.R’ ‘is-time.R’

NeedsCompilation no
Repository CRAN
Date/Publication 2016-05-10 10:30:32
R topics documented:

- assert_all_are_after .................................................. 2
- assert_all_are_date_strings ............................................. 4

Index

assert_all_are_after  Is the input in the past/future?

Description

Checks to see if the input is a time in the past/future, or before/after some time point.

Usage

assert_all_are_after(x, y, na_ignore = FALSE,
  severity = getOption("assertive.severity", "stop"))

assert_any_are_after(x, y, na_ignore = FALSE,
  severity = getOption("assertive.severity", "stop"))

assert_all_are_before(x, y, na_ignore = FALSE,
  severity = getOption("assertive.severity", "stop"))

assert_any_are_before(x, y, na_ignore = FALSE,
  severity = getOption("assertive.severity", "stop"))

assert_all_are_in_future(x, na_ignore = FALSE,
  severity = getOption("assertive.severity", "stop"))

assert_any_are_in_future(x, na_ignore = FALSE,
  severity = getOption("assertive.severity", "stop"))

assert_all_are_in_past(x, na_ignore = FALSE,
  severity = getOption("assertive.severity", "stop"))

assert_any_are_in_past(x, na_ignore = FALSE,
  severity = getOption("assertive.severity", "stop"))

is_after(x, y, .xname = get_name_in_parent(x),
  .yname = get_name_in_parent(y))

is_before(x, y, .xname = get_name_in_parent(x),
  .yname = get_name_in_parent(y))

is_in_future(x, .xname = get_name_in_parent(x))

is_in_past(x, .xname = get_name_in_parent(x))
Arguments

x  Date or POSIXt input to check.
y  Another date-time object to compare against.
na_ignore  A logical value. If FALSE, NA values cause an error; otherwise they do not. Like na.rm in many stats package functions, except that the position of the failing values does not change.
severity  How severe should the consequences of the assertion be? Either "stop", "warning", "message", or "none".
.xname  Not intended to be used directly.
.yname  Not intended to be used directly.

Details

The current time is determined by Sys.time, and the input is coerced to POSIXct format if necessary.

Value

The is_* function return TRUE if the input is a time in the future/past. The assert_* functions return nothing but throw an error if the corresponding is_* function returns FALSE.

Note

Note that the print method for POSIXct objects means that the cause attribute (in the event of failures) is not shown. You can still access it via, e.g., cause(is_in_past(x)).

See Also

Sys.time.

Examples

x <- Sys.time() + c(-1, 100)
is_in_past(x)
is_in_future(x)

# more generally, compare against any date-time
is_before(x, as.POSIXct("1999-12-31"))
is_after(x, as.POSIXct("0001-01-01"))
assert_all_are_date_strings

Does the character vector contain dates?

Description
Checks that the input contains dates or times.

Usage

```r
assert_all_are_date_strings(x, format = "%F %T", na_ignore = FALSE,
                           severity = getOption("assertive.severity", "stop"))
assert_any_are_date_strings(x, format = "%F %T", na_ignore = FALSE,
                           severity = getOption("assertive.severity", "stop"))
is_date_string(x, format = "%F %T", .xname = get_name_in_parent(x))
```

Arguments

- **x**: Input to check.
- **format**: Expected format of the dates. See `strptime`.
- **na_ignore**: A logical value. If FALSE, NA values cause an error; otherwise they do not. Like `na.rm` in many stats package functions, except that the position of the failing values does not change.
- **severity**: How severe should the consequences of the assertion be? Either "stop", "warning", "message", or "none".
- **.xname**: Not intended to be used directly.

Value
A logical vector that is TRUE when the input contains valid dates or times.

See Also

- `strptime` for specifying formats, and the `lubridate` package for automatic guessing of date formats (and other date manipulation functions).

Examples

```r
x <- c("1999-12-31 23:59:59", "wednesday", NA)
is_date_string(x)
assert_all_are_date_strings("01-Aug-1979", format = "%d%b%Y")  # My DOB!
```
Index

assert_all_are_after, 2
assert_all_are_before
  (assert_all_are_after), 2
assert_all_are_date_strings, 4
assert_all_are_in_future
  (assert_all_are_after), 2
assert_all_are_in_past
  (assert_all_are_after), 2
assert_any_are_after
  (assert_all_are_after), 2
assert_any_are_before
  (assert_all_are_after), 2
assert_any_are_date_strings
  (assert_all_are_date_strings), 4
assert_any_are_in_future
  (assert_all_are_after), 2
assert_any_are_in_past
  (assert_all_are_after), 2
is_after (assert_all_are_after), 2
is_before (assert_all_are_after), 2
is_date_string
  (assert_all_are_date_strings), 4
is_in_future (assert_all_are_after), 2
is_in_past (assert_all_are_after), 2

strptime, 4
Sys.time, 3