Package ‘assertive.data.us’

October 21, 2018

Type Package
Title Assertions to Check Properties of Strings
Version 0.0-2
Date 2018-10-21
Author Richard Cotton [aut, cre]
Maintainer Richard Cotton <richierocks@gmail.com>
Description A set of predicates and assertions for checking the properties of
US-specific complex data types. 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.data.us
BugReports https://bitbucket.org/richierocks/assertive.data.us/issues
Depends R (>= 3.0.0)
Imports assertive.base (>= 0.0-2), assertive.strings
Suggests testthat, devtools
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.
Collate 'imports.R' 'assert-is-data-us.R' 'is-data-us.R'
RoxygenNote 6.1.0
NeedsCompilation no
Repository CRAN
Date/Publication 2018-10-21 20:50:03 UTC
assert_all_are_us_social_security_numbers

Is the string a valid US SSN?

Description
Checks that the input contains US Social Security Number.

Usage
assert_all_are_us_social_security_numbers(x, na_ignore = FALSE, severity = getOption("assertive.severity", "stop"))
assert_any_are_us_social_security_numbers(x, na_ignore = FALSE, severity = getOption("assertive.severity", "stop"))
is_us_social_security_number(x)

Arguments
- **x**: Input to check.
- **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".

Value
is_us_social_security_number returns TRUE if the input string contains a valid US Social Security Number. The assert_* functions return nothing but throw an error when the is_* function returns FALSE.

Note
A valid SSN is considered to be 3 digits, then 2 digits then 4 digits possibly separated by a hyphen or space. The first block cannot be 666 or a begin with a nine, and no block can contain all zeroes. The function doesn't guarantee that the SSN actually exists.
assert_all_are_us_telephone_numbers

Examples

ssns <- c("123-45-6789", "666-45-6789", "123-00-6789")
is_us_social_security_number(ssns)

assert_all_are_us_telephone_numbers
Is the string a valid US telephone number?

Description

Checks that the input contains US/Canadian (NANPA) telephone numbers.

Usage

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

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

is_us_telephone_number(x)

Arguments

x                         Input to check.
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".

Value

is_us_telephone_number returns TRUE if the input string contains a valid US telephone number.
The assert_* functions return nothing but throw an error when the is_* function returns FALSE.

Note

A valid US phone number consists of an optional country code (either +1, 001 or just 1), followed
by a 3 digit NPA area code, where the first digit is between 2 and 9, and the second and third
digits don’t match. Next is a 3 digit exchange (NXX) code, where the first digit is between 2 and
9 and the second and third digits aren’t 11. Finally there is a four digit subscriber number (with no
restrictions). 7 digit numbers (without the NPA code) are not supported here. Canada, parts of the
Caribbean, and some Atlantic and Pacific islands also use the same numbering system.
assert_all_are_us_zip_codes

Is the string a valid US zip code?

Description
Checks that the input contains US zip codes.

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

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

is_us_zip_code(x)

Arguments
x Input to check.
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".

Value
is_us_zip_code returns TRUE if the input string contains a valid US zip code. The assert_* functions return nothing but throw an error when the is_* function returns FALSE.
Note

A valid zip code is considered to be 5 digits, or 5 digits then a hyphen then 4 digits. Unused area prefixes return FALSE, but the function doesn’t guarantee that the zip code actually exists. It should correctly return TRUE for genuine zip codes, and will weed out most badly formatted strings non-existent areas, but some non-existent codes may incorrectly return TRUE. If you need 100 up-to-date zip code base.

References


Examples

```r
zip_codes <- c(
  "Beverley Hills" = "90210",
  "The White House" = "20500",
  USPTO = "22313-1450", #5+4 style ok
  "No hyphen" = "223131450",
  "Bad area prefix" = "09901",
  Missing = NA
)
assertive::is_us_zip_code(zip_codes)
assertive::assert_any_are_us_zip_codes(zip_codes)
#The following code should throw an error.
assertive.base::dont_stop(assert_all_are_us_zip_codes(zip_codes))
```
Index

assert_all_are_us_social_security_numbers,
  2
assert_all_are_us_telephone_numbers, 3
assert_all_are_us_zip_codes, 4
assert_any_are_us_social_security_numbers
  (assert_all_are_us_social_security_numbers),
  2
assert_any_are_us_telephone_numbers
  (assert_all_are_us_telephone_numbers),
  3
assert_any_are_us_zip_codes
  (assert_all_are_us_zip_codes),
  4

is_us_social_security_number
  (assert_all_are_us_social_security_numbers),
  2
is_us_telephone_number
  (assert_all_are_us_telephone_numbers),
  3
is_us_zip_code
  (assert_all_are_us_zip_codes),
  4