Package ‘ensurer’

August 29, 2016

Type Package
Title Ensure Values at Runtime
Version 1.1
Author Stefan Milton Bache
Maintainer Stefan Milton Bache <stefan@stefanbache.dk>
Description Add simple runtime contracts to R values. These ensure that values
fulfil certain conditions and will raise appropriate errors if they do not.
License MIT + file LICENSE
Suggests magrittr, testthat, knitr
VignetteBuilder knitr
ByteCompile Yes

URL https://github.com/smbache/ensurer
BugReports https://github.com/smbache/ensurer/issues
NeedsCompilation no
Repository CRAN
Date/Publication 2015-04-21 23:45:48

R topics documented:

check_that .......................................................... 2
ensurer ............................................................ 4
print.ensurer .................................................. 5

Index
check_that

Ensure Certain Conditions for a Value at Runtime.

Description

Use ensure_that (imperative form) to ensure conditions for a value "on the fly". The present tense form, ensures_that is used to make reusable "contracts" (functions) which can subsequently be applied to values; see examples below. It is also possible to check (rather than ensure) whether conditions are satisfied; the check_that function works like ensure_that but will return TRUE or FALSE.

Usage

check_that(. , ...)

check(. , ...)

ensure_that(. , ... , fail_with = function(e) stop(e) , err_desc = "")

ensure(. , ... , fail_with = function(e) stop(e) , err_desc = "")

ensures_that(... , fail_with = function(e) stop(e) , err_desc = "")

ensures(... , fail_with = function(e) stop(e) , err_desc = "")

Arguments

. The value which is to be ensured.

... conditions which must pass for the ensuring contract to be fulfilled. Any named argument will treated as values available when evaluating the conditions. To reference the value itself use the dot-placeholder, `.var`. See 'Details' for some special named arguments.

fail_with Either a unary function (accepting a simpleError) or a static value.

err_desc A character string with an additional error description.

Details

It is possible to specify custom error message for specific conditions to make them more readable and user-friendly. To do this use a formula condition ~ err.message, where err.message is a single character value.

Existing contracts can be added as a condition argument, which will add the conditions from the existing contract to the new contract (along with any assigned values). To do this use (unary) `+` to indicate that an argument is a contract. See example below.

It is important to note that a condition is only satisfied if it evaluates to TRUE (tested with `isTRUE`), i.e. a vector with several TRUEs will fail, so be sure to use all or any in such a case.

The functions ensure ensures, and check are short-hand aliases for their *_that counterparts.
Value

ensures_that returns an ensuring function; ensure_that returns the value itself on success. 
check_that returns TRUE on success, and FALSE otherwise.

Examples

```r
## Not run:

ensure_that(1:10, is.integer)

# Examples below will use the magrittr pipe
library(magrittr)

# Create a contract which can ensure that a matrix is square.
ensure_square <- ensures_that(ncol(.) == nrow(.))

# apply it.
A <-
  diag(4) %>%
  ensure_square

# Without the pipe operator:
A <- ensure_square(diag(4))

# Ensure on the fly (this will pass the test)
A <-
  matrix(runif(16), 4, 4) %>%
  ensure_that(ncol(.) == nrow(.), all(. <= 1))

# This will raise an error
A <-
  matrix(NA, 4, 4) %>%
  ensure_that(. %>% anyNA %>% not)

# Tweak failure:
A <-
  1:10 %>%
  ensure_that(all(. < 5), err_desc = "Number tests!")

# A default value for failure situations:
A <-
  1:10 %>%
  ensure_that(all(. < 5), fail_with = NA)

# Suppose you had an email function:
email_err <- function(e) {email(e$message); stop(e)}

A <-
  1:10 %>%
  ensure_that(all(. < 5), fail_with = email_err)

# Two similar contracts, one extending the other.
```
ensurer - Ensure Values at Runtime

Description

ensurer is a utility package for R that provides a simple and light-weight mechanism for ensuring certain aspects of values at runtime.

Details

R does not provide any mechanism for type-safety and since it is not a compiled language, the risk of having unexpected results is there at runtime. R functions often accept different types for the same input and/or have different return types for different situations.

As an example, a query to a database or the scraping of a website might not return valid data, where "validity" can refer to a number of conditions. It might be a positive or certain number of records; that all cases are complete; that some column is weekly increasing; or simply that the result is a data.frame.

If one does not deal with these ambiguities and risks appropriately, some resulting errors may be hard to track down or may even go unnoticed. It is desirable to get an error immediately when a value does not have the correct type or does not satisfy certain criteria.

"Ensuring values" is here meant as a "contract", or a set of conditions, such that if a value does not comply an error is raised instantly. An ensuring contract (a function) is created with ensures_that (ideal for multiple use or readability with complex contracts).

It is also possible to ensure properties on the fly using ensure_that (ideal for simple, one-time contracts).

Using the magrittr pipe %>%, greatly improves semantics of the functionality provided by this package, but it is not necessary.

This package is not meant as a substitute for unit testing, and great packages for this already exist, e.g. testthat by Hadley Wickham. The ensurer package is meant as a simple and ideal tool for scripts or programs where runtime conditions may break the functionality, and where errors should be raised as soon and clear as possible.

For a more thorough introduction, see vignette(ensurer).

Author(s)

Stefan Holst Milton Bache <stefan@stefanbache.dk>
Description
Print method for ensurer contracts

Usage
## S3 method for class 'ensurer'
print(x, ...)

Arguments

- `x` a function made with `ensures_that`
- `...` not used.

Value

- `x`
Index

check (check_that), 2
check_that, 2

ensure (check_that), 2
ensure_that, 4
ensure_that (check_that), 2
ensurer, 4
ensurer-package (ensurer), 4
ensures (check_that), 2
ensures_that, 4
ensures_that (check_that), 2

print.ensurer, 5