Package 'fuzzr'

May 8, 2026

Type Package

Title Fuzz-Test R Functions

Version 0.2.2

Description Test function arguments with a wide array of inputs, and produce reports summarizing messages, warnings, errors, and returned values.

License MIT + file LICENSE

URL https://github.com/mdlincoln/fuzzr

BugReports https://github.com/mdlincoln/fuzzr/issues

Imports assertthat, progress, purrr

Suggests knitr, rmarkdown, testthat

VignetteBuilder knitr

LazyData TRUE

RoxygenNote 6.0.1

NeedsCompilation no

Author Matthew Lincoln [aut, cre]

Maintainer Matthew Lincoln <matthew.d.lincoln@gmail.com>

Repository CRAN

Date/Publication 2018-05-08 16:45:18 UTC

Contents

as.data.frame.fuzz_results

Summarize fuzz test results as a data frame

Description

Summarize fuzz test results as a data frame

Usage

```
## S3 method for class 'fuzz_results'
as.data.frame(x, ..., delim = "; ")
```

Arguments

x	Object returned by fuzz_function.
...	Additional arguments to be passed to or from methods.
delim	The delimiter to use for fields like messages or warnings in which there may be multiple results.

Value

A data frame with the following columns:

fuzz_input The name of the fuzz test performed.

output Delimited outputs to the command line from the process, if applicable.

messages Delimited messages, if applicable.

warnings Delimited warnings, if applicable.

errors Error returned, if applicable.

value_classes Delimited classes of the object returned by the function, if applicable

results_index Index of x from which the summary was produced.

fuzzr

Fuzz-Test R Functions

Description

Test function arguments with a wide array of inputs, and produce reports summarizing messages, warnings, errors, and returned values.

`fuzz_function` *Fuzz-test a function*

Description

Evaluate how a function responds to unexpected or non-standard inputs.

Usage

```
fuzz_function(fun, arg_name, ..., tests = test_all(), check_args = TRUE,
  progress = interactive())

p_fuzz_function(fun, .l, check_args = TRUE, progress = interactive())
```

Arguments

`fun`	A function.
`arg_name`	Quoted name of the argument to fuzz test.
`...`	Other non-dynamic arguments to pass to `fun`. These will be repeated for every one of the `tests`.
`tests`	Which fuzz tests to run. Accepts a named list of inputs, defaulting to `test_all`.
`check_args`	Check if `arg_name` and any arguments passed as `...` are accepted by `fun`. Set to `FALSE` if you need to pass arguments to a function that accepts arguments via `....`
`progress`	Show a progress bar while running tests?
`.l`	A named list of tests.

Details

`fuzz_function` provides a simple interface to fuzz test a single argument of a function by passing the function, name of the argument, static values of other required arguments, and a named list of test values.

`p_fuzz_function` takes a nested list of arguments paired with lists of tests to run on each argument, and will evaluate every combination of argument and provided test.

Value

A `fuzz_results` object.

Note

The user will be asked to confirm before proceeding if the combinations of potential tests exceeds 500,000.

See Also

fuzz_results and as.data.frame.fuzz_results to access fuzz test results.

Examples

```
# Evaluate the 'formula' argument of lm, passing additional required variables
fr <- fuzz_function(lm, "formula", data = iris)

# When evaluating a function that takes ..., set check_args to FALSE
fr <- fuzz_function(paste, "x", check_args = FALSE)

# Pass tests to multiple arguments via a named list
test_args <- list(
   data = test_df(),
   subset = test_all(),
   # Specify custom tests with a new named list
   formula = list(all_vars = Sepal.Length ~ ., one_var = mpg ~ .))
fr <- p_fuzz_function(lm, test_args)
```

---

fuzz_results *Access individual fuzz test results*

Description

Access individual fuzz test results

Usage

```
fuzz_value(fr, index = NULL, ...)

fuzz_call(fr, index = NULL, ...)
```

Arguments

fr	fuzz_results object
index	The test index (by position) to access. Same as the results_index in the data frame returned by as.data.frame.fuzz_results.
...	Additional arguments must be named regex patterns that will be used to match against test names. The names of the patterns must match the function argument name(s) whose test names you wish to match.

Functions

- fuzz_value: Access the object returned by the fuzz test
- fuzz_call: Access the call used for the fuzz test

test_all *Fuzz test inputs*

Description

Each test_all returns a named list that concatenates all the available tests specified below.

Usage

```
test_all()

test_char()

test_int()

test_dbl()

test_lgl()

test_fctr()

test_date()

test_raw()

test_df()

test_null()
```

Functions

- test_char: Character vectors
  - char_empty: character(0)
  - char_single: "a"
  - char_single_blank: ""
  - char_multiple: c("a", "b", "c")
  - char_multiple_blank: c("a", "b", "c", "")
  - char_with_na: c("a", "b", NA)
  - char_single_na: NA_character_
  - char_all_na: c(NA_character_, NA_character_, NA_character_)
- test_int: Integer vectors
  - int_empty: integer(0)
  - int_single: 1L
  - int_multiple: 1:3
  - int_with_na: c(1L, 2L, NA)

  - int_single_na: NA_integer_
  - int_all_na: c(NA_integer_, NA_integer_, NA_integer_)

- test_dbl: Double vectors
  - dbl_empty: numeric(0)
  - dbl_single: 1.5
  - dbl_mutliple: c(1.5, 2.5, 3.5)
  - dbl_with_na: c(1.5, 2.5, NA)
  - dbl_single_na: NA_real_
  - dbl_all_na: c(NA_real_, NA_real_, NA_real_)
- test_lgl: Logical vectors
  - lgl_empty: logical(0)
  - lgl_single: TRUE
  - lgl_mutliple: c(TRUE, FALSE, FALSE)
  - lgl_with_na: c(TRUE, NA, FALSE)
  - lgl_single_na: NA
  - lgl_all_na: c(NA, NA, NA)
- test_fctr: Factor vectors
  - fctr_empty: structure(integer(0), .Label = character(0), class = "factor")
  - fctr_single: structure(1L, .Label = "a", class = "factor")
  - fctr_multiple: structure(1:3, .Label = c("a", "b", "c"), class = "factor")
  - fctr_with_na: structure(c(1L, 2L, NA), .Label = c("a", "b"), class = "factor")
  - fctr_missing_levels: structure(1:3, .Label = c("a", "b", "c", "d"), class = "factor")
  - fctr_single_na: structure(NA_integer_, .Label = character(0), class = "factor")
  - fctr_all_na: structure(c(NA_integer_, NA_integer_, NA_integer_), .Label = character(0), class = "factor")
- test_date: Date vectors
  - date_single: as.Date("2001-01-01")
  - date_multiple: as.Date(c("2001-01-01", "1950-05-05"))
  - date_with_na: as.Date(c("2001-01-01", NA, "1950-05-05"))
  - date_single_na: as.Date(NA_integer_, origin = "1971-01-01")
  - date_all_na: as.Date(rep(NA_integer_, 3), origin = "1971-01-01")
- test_raw: Raw vectors
  - raw_empty: raw(0)
  - raw_char: as.raw(0x62),
  - raw_na: charToRaw(NA_character_)
- test_df: Data frames
  - df_complete: datasets::iris
  - df_empty: data.frame(NULL)
  - df_one_row: datasets::iris[1,]
  - df_one_col: datasets::iris[,1]

  - df_with_na: iris with several NAs added to each column.
- test_null: Null value
  - null_value: NULL

Index