

Package ‘conditions’

January 18, 2017

Type Package

Title Standardized Conditions for R

Version 0.1

Author Michel Lang <michellang@gmail.com>

Maintainer Michel Lang <michellang@gmail.com>

Description Implements specialized conditions, i.e., typed errors, warnings and messages. Offers a set of standardized conditions (value error, deprecated warning, io message, ...) in the fashion of Python's built-in exceptions.

License BSD_2_clause + file LICENSE

URL <https://github.com/mlg/conditions>

BugReports <https://github.com/mlg/conditions/issues>

NeedsCompilation yes

ByteCompile yes

Encoding UTF-8

Depends R (>= 3.0.0)

Suggests testthat

RoxygenNote 5.0.1

Repository CRAN

Date/Publication 2017-01-18 18:07:33

R topics documented:

assertion_message	2
as_message	5

Index	9
--------------	----------

assertion_message	<i>Generate a custom condition</i>
-------------------	------------------------------------

Description

condition creates a custom condition. The functions condition_message, condition_warning and condition_error are specialized to create conditions of type “message”, “warning” or “error”, respectively. Furthermore, the constructors for some standardized conditions are predefined (see details).

Usage

```
assertion_message(message, call = sys.call(-1L), attach = NULL)
deprecated_message(message, call = sys.call(-1L), attach = NULL)
dimension_message(message, call = sys.call(-1L), attach = NULL)
future_message(message, call = sys.call(-1L), attach = NULL)
index_message(message, call = sys.call(-1L), attach = NULL)
io_message(message, call = sys.call(-1L), attach = NULL)
length_message(message, call = sys.call(-1L), attach = NULL)
library_message(message, call = sys.call(-1L), attach = NULL)
lookup_message(message, call = sys.call(-1L), attach = NULL)
missing_message(message, call = sys.call(-1L), attach = NULL)
name_message(message, call = sys.call(-1L), attach = NULL)
runtime_message(message, call = sys.call(-1L), attach = NULL)
type_message(message, call = sys.call(-1L), attach = NULL)
value_message(message, call = sys.call(-1L), attach = NULL)
assertion_warning(message, call = sys.call(-1L), attach = NULL)
deprecated_warning(message, call = sys.call(-1L), attach = NULL)
dimension_warning(message, call = sys.call(-1L), attach = NULL)
future_warning(message, call = sys.call(-1L), attach = NULL)
```

```
index_warning(message, call = sys.call(-1L), attach = NULL)
io_warning(message, call = sys.call(-1L), attach = NULL)
length_warning(message, call = sys.call(-1L), attach = NULL)
library_warning(message, call = sys.call(-1L), attach = NULL)
lookup_warning(message, call = sys.call(-1L), attach = NULL)
missing_warning(message, call = sys.call(-1L), attach = NULL)
name_warning(message, call = sys.call(-1L), attach = NULL)
runtime_warning(message, call = sys.call(-1L), attach = NULL)
type_warning(message, call = sys.call(-1L), attach = NULL)
value_warning(message, call = sys.call(-1L), attach = NULL)
assertion_error(message, call = sys.call(-1L), attach = NULL)
deprecated_error(message, call = sys.call(-1L), attach = NULL)
dimension_error(message, call = sys.call(-1L), attach = NULL)
future_error(message, call = sys.call(-1L), attach = NULL)
index_error(message, call = sys.call(-1L), attach = NULL)
io_error(message, call = sys.call(-1L), attach = NULL)
length_error(message, call = sys.call(-1L), attach = NULL)
library_error(message, call = sys.call(-1L), attach = NULL)
lookup_error(message, call = sys.call(-1L), attach = NULL)
missing_error(message, call = sys.call(-1L), attach = NULL)
name_error(message, call = sys.call(-1L), attach = NULL)
runtime_error(message, call = sys.call(-1L), attach = NULL)
type_error(message, call = sys.call(-1L), attach = NULL)
value_error(message, call = sys.call(-1L), attach = NULL)
```

```

condition(type, class = character(0L), message, call = sys.call(-1L))
condition_error(class, message, call = sys.call(-1L), attach = NULL)
condition_warning(class, message, call = sys.call(-1L), attach = NULL)
condition_message(class, message, call = sys.call(-1L), attach = NULL)

```

Arguments

message	[character(1)] Information about the condition.
call	[call NULL] Call stack.
attach	[ANY] Object to attach to the condition. Can be accessed via <code>cond\$attached</code> in a tryCatch (see example).
type	[character(1)] Should be one of “error”, “warning” or “message”.
class	[character] Class for the condition. The functions <code>condition_error</code> , <code>condition_warning</code> and <code>condition_message</code> automatically append the respective type with an underscore (see example).

Details

The standardized conditions include:

- “**assertion**”: Assertion (on user input) failed.
- “**deprecated**”: Feature is deprecated.
- “**dimension**”: Wrong dimension.
- “**future**”: Feature is subject to change in the future.
- “**index**”: Subscript out of range.
- “**io**”: File/directory not found or accessible.
- “**length**”: Wrong length.
- “**library**”: Required package not installed.
- “**lookup**”: Named subelement does not exist.
- “**missing**”: Missing values.
- “**name**”: Failed lookup of a global variable.
- “**runtime**”: Something else which does not fit in any other category went wrong.
- “**type**”: Unexpected type/class.
- “**value**”: Inappropriate value.

Value

condition .

Examples

```
# A simple IO error:
e = condition_error("io", "Failed to load file")
print(e)
class(e)

# To signal the condition, use message/warning/stop.
## Not run:
message(e)
warning(e)
stop(e)

## End(Not run)

# These are equivalent (except the call):
w1 = condition("warning", "dimension_warning", "foo")
w2 = condition_warning("dimension", "foo")
w3 = tryCatch(dimension_warning("foo"), condition = function(e) e)

# Attach and retrieve additional information
f = function(x) {
  if(!is.numeric(x))
    assertion_error(" must be numeric", attach = x)
  x^2
}
f(1:10)

tryCatch(f(letters), assertion_error = function(e) {
  message("x must be numeric, but is ", typeof(e$attached))
})
```

as_message

Convert Conditions

Description

These functions are intended to change the class of conditions. In combination with `tryCatch`, unspecified conditions can easily be casted to a more specific type. See `condition` for a short explanation about the predefined condition classes.

Usage

```
as_message(class, message = NULL)
```

```
as_warning(class, message = NULL)
```

```
as_error(class, message = NULL)
as_assertion_message(message = NULL)
as_deprecated_message(message = NULL)
as_dimension_message(message = NULL)
as_future_message(message = NULL)
as_index_message(message = NULL)
as_io_message(message = NULL)
as_length_message(message = NULL)
as_library_message(message = NULL)
as_lookup_message(message = NULL)
as_missing_message(message = NULL)
as_name_message(message = NULL)
as_runtime_message(message = NULL)
as_type_message(message = NULL)
as_value_message(message = NULL)
as_assertion_warning(message = NULL)
as_deprecated_warning(message = NULL)
as_dimension_warning(message = NULL)
as_future_warning(message = NULL)
as_index_warning(message = NULL)
as_io_warning(message = NULL)
as_length_warning(message = NULL)
as_library_warning(message = NULL)
as_lookup_warning(message = NULL)
```

```
as_missing_warning(message = NULL)
as_name_warning(message = NULL)
as_runtime_warning(message = NULL)
as_type_warning(message = NULL)
as_value_warning(message = NULL)
as_assertion_error(message = NULL)
as_deprecated_error(message = NULL)
as_dimension_error(message = NULL)
as_future_error(message = NULL)
as_index_error(message = NULL)
as_io_error(message = NULL)
as_length_error(message = NULL)
as_library_error(message = NULL)
as_lookup_error(message = NULL)
as_missing_error(message = NULL)
as_name_error(message = NULL)
as_runtime_error(message = NULL)
as_type_error(message = NULL)
as_value_error(message = NULL)
```

Arguments

class	[character] New classes for the condition.
message	[character] If provided, overwrites the original condition message.

Value

function . Returns a function which takes a condition as first argument, creates a new condition of the respective type and signals the created condition with `message`, `warning` or `stop`, respectively.

Examples

```
# Turn the warning of sqrt() into a value message
message(tryCatch(sqrt(-1), warning = as_value_message()))

## Not run:
# Turn the warning of sqrt() into a value error
tryCatch(sqrt(-1), warning = as_value_error())

# Or, alternatively with a custom message:
tryCatch(sqrt(-1), warning = as_value_error("sqrt of negative value"))

## End(Not run)
```


Index

as_assertion_error (as_message), 5
as_assertion_message (as_message), 5
as_assertion_warning (as_message), 5
as_deprecated_error (as_message), 5
as_deprecated_message (as_message), 5
as_deprecated_warning (as_message), 5
as_dimension_error (as_message), 5
as_dimension_message (as_message), 5
as_dimension_warning (as_message), 5
as_error (as_message), 5
as_future_error (as_message), 5
as_future_message (as_message), 5
as_future_warning (as_message), 5
as_index_error (as_message), 5
as_index_message (as_message), 5
as_index_warning (as_message), 5
as_io_error (as_message), 5
as_io_message (as_message), 5
as_io_warning (as_message), 5
as_length_error (as_message), 5
as_length_message (as_message), 5
as_length_warning (as_message), 5
as_library_error (as_message), 5
as_library_message (as_message), 5
as_library_warning (as_message), 5
as_lookup_error (as_message), 5
as_lookup_message (as_message), 5
as_lookup_warning (as_message), 5
as_message, 5
as_missing_error (as_message), 5
as_missing_message (as_message), 5
as_missing_warning (as_message), 5
as_name_error (as_message), 5
as_name_message (as_message), 5
as_name_warning (as_message), 5
as_runtime_error (as_message), 5
as_runtime_message (as_message), 5
as_runtime_warning (as_message), 5
as_type_error (as_message), 5
as_type_message (as_message), 5
as_type_warning (as_message), 5
as_value_error (as_message), 5
as_value_message (as_message), 5
as_value_warning (as_message), 5
as_warning (as_message), 5
assertion_error (assertion_message), 2
assertion_message, 2
assertion_warning (assertion_message), 2
condition, 5
condition (assertion_message), 2
condition_error (assertion_message), 2
condition_message (assertion_message), 2
condition_warning (assertion_message), 2
deprecated_error (assertion_message), 2
deprecated_message (assertion_message),
2
deprecated_warning (assertion_message),
2
dimension_error (assertion_message), 2
dimension_message (assertion_message), 2
dimension_warning (assertion_message), 2
future_error (assertion_message), 2
future_message (assertion_message), 2
future_warning (assertion_message), 2
index_error (assertion_message), 2
index_message (assertion_message), 2
index_warning (assertion_message), 2
io_error (assertion_message), 2
io_message (assertion_message), 2
io_warning (assertion_message), 2
length_error (assertion_message), 2
length_message (assertion_message), 2
length_warning (assertion_message), 2
library_error (assertion_message), 2
library_message (assertion_message), 2

library_warning (assertion_message), 2
lookup_error (assertion_message), 2
lookup_message (assertion_message), 2
lookup_warning (assertion_message), 2

message, 8
missing_error (assertion_message), 2
missing_message (assertion_message), 2
missing_warning (assertion_message), 2

name_error (assertion_message), 2
name_message (assertion_message), 2
name_warning (assertion_message), 2

runtime_error (assertion_message), 2
runtime_message (assertion_message), 2
runtime_warning (assertion_message), 2

stop, 8

tryCatch, 4, 5
type_error (assertion_message), 2
type_message (assertion_message), 2
type_warning (assertion_message), 2

value_error (assertion_message), 2
value_message (assertion_message), 2
value_warning (assertion_message), 2

warning, 8