

# Package ‘conditionz’

April 24, 2019

**Title** Control How Many Times Conditions are Thrown

**Description** Provides ability to control how many times in function calls conditions are thrown (shown to the user). Includes control of warnings and messages.

**Version** 0.1.0

**License** MIT + file LICENSE

**URL** <https://github.com/ropenscilabs/conditionz>

**BugReports** <http://github.com/ropenscilabs/conditionz/issues>

**Encoding** UTF-8

**Depends** R(>= 3.2.1)

**Imports** R6, uuid

**Suggests** testthat

**RoxygenNote** 6.1.1

**X-schema.org-applicationCategory** Utilities

**X-schema.org-keywords** condition, condition-control, warning, message

**X-schema.org-isPartOf** <https://ropensci.org>

**NeedsCompilation** no

**Author** Scott Chamberlain [aut, cre] (<<https://orcid.org/0000-0003-1444-9135>>)

**Maintainer** Scott Chamberlain <[myrmecocystus@gmail.com](mailto:myrmecocystus@gmail.com)>

**Repository** CRAN

**Date/Publication** 2019-04-24 12:20:07 UTC

## R topics documented:

conditionz-package . . . . .	2
ConditionKeeper . . . . .	2
handle_conditions . . . . .	3

<b>Index</b>	<b>5</b>
--------------	----------

conditionz-package     *condition control*

---

### Description

condition control

### Author(s)

Scott Chamberlain [myrmecocystus@gmail.com](mailto:myrmecocystus@gmail.com)

---

ConditionKeeper     *ConditionKeeper*

---

### Description

ConditionKeeper

### Arguments

times            (integer) number of times to throw condition. required. default: 1  
condition        (character) which condition, one of "message" (default) or "warning"

### Details

#### Methods

- `add(x)` - add a condition to internal storage
- `remove()` - remove the first condition from internal storage; returns that condition so you know what you removed
- `purge()` - removes all conditions
- `thrown_already(x)` - (return: logical) has the condition been thrown already?
- `not_thrown_yet(x)` - (return: logical) has the condition NOT been thrown yet?
- `thrown_times(x)` - (return: numeric) number of times the condition has been thrown
- `thrown_enough(x)` - (return: logical) has the condition been thrown enough? "enough" being: thrown number of times equal to what you specified in the `times` parameter
- `get_id()` - get the internal ID for the ConditionKeeper object
- `handle_conditions(expr)` - pass a code block or function and handle conditions within it

### See Also

[handle\\_conditions\(\)](#)

**Examples**

```
x <- ConditionKeeper$new(times = 4)
x
x$get_id()
x$add("one")
x$add("two")
x
x$thrown_already("one")
x$thrown_already("bears")
x$not_thrown_yet("bears")

x$add("two")
x$add("two")
x$add("two")
x$thrown_times("two")
x$thrown_enough("two")
x$thrown_enough("one")

foo <- function(x) {
  message("you gave: ", x)
  return(x)
}
foo('a')
x$handle_conditions(foo('a'))

x <- ConditionKeeper$new(times = 4, condition = "warning")
x
x$add("one")
x$add("two")
x
```

---

handle\_conditions      *Handle conditions*

---

**Description**

Handle conditions

**Usage**

```
handle_conditions(expr, condition = "message", times = 1)

handle_messages(expr, times = 1)

handle_warnings(expr, times = 1)
```

**Arguments**

expr            an expression  
condition      (character) one of "message" or "warning"  
times          (integer) max. times a condition should be thrown. default: 1

**Details**

Uses [ConditionKeeper](#) internally

**Value**

whatever the expr returns

**Examples**

```
foo <- function(x) {  
  message("you gave: ", x)  
  return(x)  
}  
  
foo('a')  
capture_message(foo('a'))  
handle_conditions(foo('a'))  
suppressMessages(handle_conditions(foo('a')))  
handle_conditions(foo('a'), "message")  
  
bar <- function(x) {  
  for (i in x) message("you gave: ", i)  
  return(x)  
}  
bar(1:5)  
handle_conditions(bar(1:5))  
  
handle_messages(foo('a'))  
  
hello <- function(x) {  
  warning("you gave: ", x)  
  return(x)  
}  
handle_warnings(hello('a'))  
  
# code block  
handle_warnings({  
  as.numeric(letters[1:3])  
  as.numeric(letters[4:6])  
  as.numeric(letters[7:9])  
})
```

# Index

\*Topic **datasets**

ConditionKeeper, [2](#)

\*Topic **package**

conditionz-package, [2](#)

ConditionKeeper, [2](#), [4](#)

conditionz (conditionz-package), [2](#)

conditionz-package, [2](#)

handle\_conditions, [3](#)

handle\_conditions(), [2](#)

handle\_messages (handle\_conditions), [3](#)

handle\_warnings (handle\_conditions), [3](#)