

Package ‘assertive.models’

August 29, 2016

Type Package

Title Assertions to Check Properties of Models

Version 0.0-1

Date 2015-10-06

Author Richard Cotton [aut, cre]

Maintainer Richard Cotton <richierocks@gmail.com>

Description A set of predicates and assertions for checking the properties of models. 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.models>

BugReports <https://bitbucket.org/richierocks/assertive.models/issues>

Depends R (>= 3.0.0)

Imports assertive.base (>= 0.0-2), stats

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-has-terms.R' 'assert-is-empty-model.R' 'has-terms.R' 'is-empty-model.R'

NeedsCompilation no

Repository CRAN

Date/Publication 2015-10-06 14:04:11

R topics documented:

assert_has_terms	2
assert_is_empty_model	3

Index	4
--------------	----------

assert_has_terms	<i>Does the input have terms?</i>
------------------	-----------------------------------

Description

Checks to see if the input has a terms component or attribute.

Usage

```
assert_has_terms(x, severity = getOption("assertive.severity", "stop"))
```

```
has_terms(x, .xname = get_name_in_parent(x))
```

Arguments

x	Input to check.
severity	How severe should the consequences of the assertion be? Either "stop", "warning", "message", or "none".
.xname	Not intended to be used directly.

Value

has_terms returns TRUE if the input has an element or an attribute named terms. assert_has_terms returns nothing but throws an error if has_terms is not TRUE.

See Also

[terms.](#)

Examples

```
assert_has_terms(lm(uptake ~ conc, CO2))
```

assert_is_empty_model *Is the input the empty model?*

Description

Checks to see if the input is the empty model.

Usage

```
assert_is_empty_model(x, severity = getOption("assertive.severity", "stop"))
```

```
assert_is_non_empty_model(x, severity = getOption("assertive.severity",  
"stop"))
```

```
is_empty_model(x, .xname = get_name_in_parent(x))
```

```
is_non_empty_model(x, .xname = get_name_in_parent(x))
```

Arguments

x	Input to check.
severity	How severe should the consequences of the assertion be? Either "stop", "warning", "message", or "none".
.xname	Not intended to be used directly.

Value

is_[non_]empty_model returns TRUE if the input is an [non] empty model. (has_terms is used to determine that a variable is a model object.) The model is considered empty if there are no factors and no intercept. The assert_* functions return nothing but throw an error if the corresponding is_* function returns FALSE.

See Also

[is.empty.model](#) and `is_empty`.

Examples

```
assert_is_empty_model(lm(uptake ~ 0, CO2))  
assert_is_non_empty_model(lm(uptake ~ conc, CO2))  
assert_is_non_empty_model(lm(uptake ~ 1, CO2))
```

Index

`assert_has_terms`, [2](#)
`assert_is_empty_model`, [3](#)
`assert_is_non_empty_model`
 (`assert_is_empty_model`), [3](#)

`has_terms` (`assert_has_terms`), [2](#)

`is.empty.model`, [3](#)
`is_empty_model` (`assert_is_empty_model`),
 [3](#)
`is_non_empty_model`
 (`assert_is_empty_model`), [3](#)

`terms`, [2](#)