Package 'lintr'

November 8, 2018

Title A 'Linter' for R Code

Version 1.0.3

URL https://github.com/jimhester/lintr

BugReports https://github.com/jimhester/lintr/issues

Description Checks adherence to a given style, syntax errors and possible semantic issues. Supports on the fly checking of R code edited with 'RStudio IDE', 'Emacs', 'Vim', 'Sublime Text' and 'Atom'.

Depends R (>= 3.1.1)

Imports rex, crayon, codetools, stringdist, testthat, digest, igraph, rstudioapi (>= 0.2), httr, jsonlite, knitr, stats, utils

Suggests rmarkdown, mockery

License MIT + file LICENSE

LazyData true

VignetteBuilder knitr

RoxygenNote 6.0.1

NeedsCompilation no

Author Jim Hester [aut, cre]

Maintainer Jim Hester <james.f.hester@gmail.com>

Repository CRAN

Date/Publication 2018-11-08 17:10:10 UTC

R topics documented:

`absolute_paths_linter` *linters*

Description

Available linters

Usage

```
absolute_paths_linter(source_file)

assignment_linter(source_file)

closed_curly_linter(allow_single_line = FALSE)

commas_linter(source_file)

commented_code_linter(source_file)

infix_spaces_linter(source_file)

line_length_linter(length)

no_tab_linter(source_file)

object_usage_linter(source_file)

camel_case_linter(source_file)

snake_case_linter(source_file)

multiple_dots_linter(source_file)

object_length_linter(length = 20L)

open_curly_linter(allow_single_line = FALSE)
```

```
single_quotes_linter(source_file)

spaces_inside_linter(source_file)

spaces_left_parentheses_linter(source_file)

trailing_blank_lines_linter(source_file)

trailing_whitespace_linter(source_file)
```

Arguments

source_file	returned by get_source_expressions
allow_single_line	if true allow a open and closed curly pair on the same line.
length	the length cutoff to use for the given linter.

Functions

- absolute_paths_linter: checks that no absolute paths are used.
- assignment_linter: checks that '<-' is always used for assignment
- closed_curly_linter: check that closed curly braces should always be on their own line unless they follow an else.
- commas_linter: check that all commas are followed by spaces, but do not have spaces before them.
- commented_code_linter: checks that there is no commented code outside roxygen blocks
- infix_spaces_linter: check that all infix operators have spaces around them.
- line_length_linter: check the line length of both comments and code is less than length.
- no_tab_linter: check that only spaces are used, never tabs.
- object_usage_linter: checks that closures have the proper usage using checkUsage. Note this runs eval on the code, so do not use with untrusted code.
- camel_case_linter: check that objects are not in camelCase.
- snake_case_linter: check that objects are not in snake_case.
- multiple_dots_linter: check that objects do not have.multiple.dots.
- object_length_linter: check that objects do are not very long.not have.multiple.dots.
- open_curly_linter: check that opening curly braces are never on their own line and are always followed by a newline.
- single_quotes_linter: checks that only single quotes are used to delimit string contestants.
- spaces_inside_linter: check that parentheses and square brackets do not have spaces directly inside them.
- spaces_left_parentheses_linter: check that all left parentheses have a space before them unless they are in a function call.
- trailing_blank_lines_linter: check there are no trailing blank lines.
- trailing_whitespace_linter: check there are no trailing whitespace characters.

clear_cache *Clear the lintr cache*

Description

Clear the lintr cache

Usage

```
clear_cache(file = NULL, path = NULL)
```

Arguments

file	filename whose cache to clear. If you pass NULL it will delete all of the caches.
path	directory to store caches. Reads option 'lintr.cache_directory' as the default.

default_linters *Default linters to use*

Description

Default linters to use

Usage

```
default_linters
```

Format

An object of class list of length 18.

default_settings *Default lintr settings*

Description

Default lintr settings

Usage

```
default_settings
```

Format

An object of class `list` of length 9.

See Also

`read_settings`, `default_linters`

`exclude` *Exclude lines or files from linting*

Description

Exclude lines or files from linting

Usage

```
exclude(lints, exclusions = settings$exclusions, ...)
```

Arguments

`lints`	that need to be filtered.
`exclusions`	manually specified exclusions
`...`	additional arguments passed to `parse_exclusions`

Details

Exclusions can be specified in three different ways.

1. single line in the source file. default: `# nolint`
2. line range in the source file. default: `# nolint start`, `# nolint end`
3. exclusions parameter, a named list of the files and lines to exclude, or just the filenames if you want to exclude the entire file.

expect_lint *Lint expectation*

Description

Lint expectation

Usage

```
expect_lint(content, checks, ..., file = NULL)
```

Arguments

content	the file content to be linted
checks	a list of named vectors of checks to be performed. Performs different checks depending on the value of checks. • NULL check if the lint returns no lints. • unnamed-vector check if the lint's message matches the value. • named-vector check if the lint's field matches the named field. • list-vectors check if the given lint matches (use if more than one lint is returned for the content)
...	one or more linters to use for the check
file	if not NULL read content from a file rather than from content

expect_lint_free *Test that the package is lint free*

Description

This function is a thin wrapper around lint_package that simply tests there are no lints in the package. It can be used to ensure that your tests fail if the package contains lints.

Usage

```
expect_lint_free(...)
```

Arguments

...	arguments passed to lint_package

get_source_expressions *Parsed sourced file from a filename*

Description

This object is given as input to each linter

Usage

```
get_source_expressions(filename)
```

Arguments

filename	the file to be parsed.

Lint *Create a* Lint *object*

Description

Create a Lint object

Usage

```
Lint(filename, line_number = 1L, column_number = NULL, type = c("style",
  "warning", "error"), message = "", line = "", ranges = NULL,
  linter = NULL)
```

Arguments

filename	path to the source file that was linted.
line_number	line number where the lint occurred.
column_number	column the lint occurred.
type	type of lint.
message	message used to describe the lint error
line	code source where the lint occurred
ranges	ranges on the line that should be emphasized.
linter	name of linter that created the Lint object.

lintr *Lintr*

Description

Checks adherence to a given style, syntax errors and possible semantic issues. Supports on the fly checking of R code edited with Emacs, Vim and Sublime Text.

See Also

lint, lint_package, linters

lint_file *Lint a given file*

Description

Apply one or more linters to a file and return a list of lints found.

Usage

```
lint(filename, linters = NULL, cache = FALSE, ..., parse_settings = TRUE)
```

Arguments

filename	the given filename to lint.
linters	a list of linter functions to apply see linters for a full list of default and available linters.
cache	toggle caching of lint results, if passed a character vector uses that file as the cache.
...	additional arguments passed to exclude.
parse_settings	whether to try and parse the settings

lint_package *Lint a package*

Description

Apply one or more linters to all of the R files in a package.

Usage

```
lint_package(path = ".", relative_path = TRUE, ...)
```

Arguments

path	the path to the base directory of the package, if NULL, the base directory will be searched for by looking in the parent directories of the current directory.
relative_path	if TRUE, file paths are printed using their path relative to the package base directory. If FALSE, use the full absolute path.
...	additional arguments passed to lint

parse_exclusions *read a source file and parse all the excluded lines from it*

Description

read a source file and parse all the excluded lines from it

Usage

```
parse_exclusions(file, exclude = settings$exclude,
  exclude_start = settings$exclude_start,
  exclude_end = settings$exclude_end)
```

Arguments

file	R source file
exclude	regular expression used to mark lines to exclude
exclude_start	regular expression used to mark the start of an excluded range
exclude_end	regular expression used to mark the end of an excluded range

read_settings *Read lintr settings*

Description

Lintr searches for settings for a given file in the following order.

1. options defined as `linter.setting`.
2. `linter_file` in the same directory
3. `linter_file` in the project directory
4. `default_settings`

Usage

```
read_settings(filename)
```

Arguments

`filename`	source file to be linted

Details

The default linter_file name is `.lintr` but it can be changed with option `lintr.linter_file`. This file is a dcf file, see `read.dcf` for details.

with_defaults *Modify the list of default linters*

Description

Modify the list of default linters

Usage

```
with_defaults(..., default = default_linters)
```

Arguments

`...`	named arguments of linters to change. If the named linter already exists it is replaced by the new linter, if it does not exist it is added. If the value is `NULL` the linter is removed.
`default`	default linters to change

Examples

```
# change the default line length cutoff
with_defaults(line_length_linter = line_length_linter(120))

# you can also omit the argument name if you are just using different
#   arguments.
with_defaults(line_length_linter(120))

# enforce camelCase rather than snake_case
with_defaults(camel_case_linter = NULL,
              snake_case_linter)
```

Index