Package 'goodpractice'

May 2, 2018

Title Advice on R Package Building

Version 1.0.2

Description Give advice about good practices when building R packages. Advice includes functions and syntax to avoid, package structure, code complexity, code formatting, etc.

License MIT + file LICENSE

LazyData true

URL https://github.com/mangothecat/goodpractice

BugReports https://github.com/mangothecat/goodpractice/issues

RoxygenNote 6.0.1

Suggests testthat, knitr, rmarkdown

Imports clisymbols, covr, crayon, cyclocomp (>= 1.1.0), desc, jsonlite, lintr, praise, rcmdcheck, rstudioapi, tools, utils, whoami, withr, xml2, xmlparsedata (>= 1.0.1)

Collate 'api.R' 'customization.R' 'lists.R' 'chk_covr.R' 'chk_cyclocomp.R' 'chk_description.R' 'chk_lintr.R' 'chk_namespace.R' 'chk_rcmdcheck.R' 'chk_tnf.R' 'gp.R' 'my_linters.R' 'package.R' 'prep_covr.R' 'prep_cyclocomp.R' 'prep_description.R' 'prep_expressions.R' 'prep_lintr.R' 'prep_namespace.R' 'prep_rcmdcheck.R' 'print.R' 'rstudio_markers.R' 'utils.R'

VignetteBuilder knitr

NeedsCompilation no

Author Gabor Csardi [aut], Hannah Frick [aut, cre]

Maintainer Hannah Frick <hfrick@mango-solutions.com>

Repository CRAN

Date/Publication 2018-05-02 19:03:36 UTC

R topics documented:

`goodpractice-package` *goodpractice: Advice on R Package Building*

Description

Give advice about good practices when building R packages. Advice includes functions and syntax to avoid, package structure, code complexity, code formatting, etc.

Author(s)

Maintainer: Hannah Frick <hfrick@mango-solutions.com>

Authors:

- Gabor Csardi <csardi.gabor@gmail.com>

See Also

Useful links:

- https://github.com/mangothecat/goodpractice
- Report bugs at https://github.com/mangothecat/goodpractice/issues

all_checks *List the names of all checks*

Description

List the names of all checks

Usage

```
all_checks()
```

Value

Character vector of checks

checks *List all checks performed*

Description

List all checks performed

Usage

```
checks(gp)
```

Arguments

gp	gp output.

Value

Character vector of check names.

See Also

Other API: failed_checks, results

Examples

```
path <- system.file("bad1", package = "goodpractice")
# run a subset of all checks available
g <- gp(path, checks = all_checks()[3:16])
checks(g)
```

export_json *Export failed checks to JSON*

Description

Export failed checks to JSON

Usage

```
export_json(gp, file, pretty = FALSE)
```

Arguments

gp	gp output.
file	Output connection or file.
pretty	Whether to pretty-print the JSON.

failed_checks *Names of the failed checks*

Description

Names of the failed checks

Usage

```
failed_checks(gp)
```

Arguments

gp	gp output.

Value

Names of the failed checks.

See Also

Other API: checks, results

Examples

```
path <- system.file("bad1", package = "goodpractice")
# run a subset of all checks available
g <- gp(path, checks = all_checks()[3:16])
failed_checks(g)
```

failed_positions *Positions of check failures in the source code*

Description

Note that not all checks refer to the source code. For these the result will be NULL.

Usage

```
failed_positions(gp)
```

Arguments

gp	gp output.

Details

For the ones that do, the results is a list, one for each failure. Since the same check can fail multiple times. A single failure is a list with entries: filename, line_number, column_number, ranges. ranges is a list of pairs of start and end positions for each line involved in the check.

Value

A list of lists of positions. See details below.

gp *Run good practice checks*

Description

To see the results, just print it to the screen.

Usage

```
gp(path = ".", checks = all_checks(), extra_preps = NULL,
  extra_checks = NULL, quiet = TRUE)
```

Arguments

path	Path to a package root.
checks	Character vector, the checks to run. Defaults to all checks. Use all_checks to list all checks.
extra_preps	Custom preparation functions. See make_prep on creating preparation functions.
extra_checks	Custom checks. See make_check on creating checks.
quiet	Whether to suppress output from the preparation functions. Note that not all preparation functions produce output, even if this option is set to FALSE.

Value

A goodpractice object that you can query with a simple API. See `results` to start.

Examples

```
path <- system.file("bad1", package = "goodpractice")
# run a subset of all checks available
g <- gp(path, checks = all_checks()[3:16])
g
```

`make_check` *Create a check function*

Description

Create a check function

Usage

```
make_check(description, check, gp, ...)
```

Arguments

`description`	A description of the check.
`check`	A function that takes the `state` as an argument.
`gp`	A short description of what is good practice.
`...`	Further arguments.

Examples

```
# make a preparation function
url_prep <- make_prep(
  name = "desc",
  func = function(path, quiet) desc::description$new(path)
)
# and the corresponding check function
url_chk <- make_check(
  description = "URL field in DESCRIPTION",
  tags = character(),
  preps = "desc",
  gp = "have a URL field in DESCRIPTION",
  check = function(state) state$desc$has_fields("URL")
)
# use together in gp()
bad1 <- system.file("bad1", package = "goodpractice")
res <- gp(bad1, checks = "no_description_depends",
          extra_preps = list("desc" = url_prep),
          extra_checks = list("url" = url_chk))
```

make_prep *Create a preparation function*

Description

Create a preparation function

Usage

```
make_prep(name, func)
```

Arguments

name	Name of the preparation function.
func	A function that takes two arguments: The path to the root directory of the package, and a logical argument: quiet. If quiet is true, the preparation function may print out diagnostic messages.

Examples

```
# make a preparation function
url_prep <- make_prep(
  name = "desc",
  func = function(path, quiet) desc::description$new(path)
)
# and the corresponding check function
url_chk <- make_check(
  description = "URL field in DESCRIPTION",
  tags = character(),
  preps = "desc",
  gp = "have a URL field in DESCRIPTION",
  check = function(state) state$desc$has_fields("URL")
)
# use together in gp()
bad1 <- system.file("bad1", package = "goodpractice")
res <- gp(bad1, checks = "no_description_depends",
          extra_preps = list("desc" = url_prep),
          extra_checks = list("url" = url_chk))
```

results *Return all check results in a data frame*

Description

Return all check results in a data frame

Usage

```
results(gp)
```

Arguments

gp	gp output.

Value

Data frame, with columns:

check	The name of the check.
result	Logical, whether it has failed or not.

See Also

Other API: checks, failed_checks

Examples

```
path <- system.file("bad1", package = "goodpractice")
# run a subset of all checks available
g <- gp(path, checks = all_checks()[3:16])
results(g)
```

Index