Package 'rcmdcheck'

October 14, 2022

Title Run 'R CMD check' from 'R' and Capture Results

Version 1.4.0

Description Run 'R CMD check' from 'R' and capture the results of the individual checks. Supports running checks in the background, timeouts, pretty printing and comparing check results.

License MIT + file LICENSE

URL https://r-lib.github.io/rcmdcheck/, https://github.com/r-Lib/rcmdcheck#readme

BugReports https://github.com/r-Lib/rcmdcheck/issues

Imports callr (>= 3.1.1.9000), cli (>= 3.0.0), curl, desc (>= 1.2.0), digest, pkgbuild, prettyunits, R6, rprojroot, sessioninfo (>= 1.1.1), utils, withr, xopen

Suggests covr, knitr, mockery, processx, ps, rmarkdown, svglite, testthat, webfakes

Encoding UTF-8

RoxygenNote 7.1.2

Config/testthat/edition 3

NeedsCompilation no

Author Gábor Csárdi [cre, aut]

Maintainer Gábor Csárdi <csardi.gabor@gmail.com>

Repository CRAN

Date/Publication 2021-09-27 15:10:02 UTC

R topics documented:

`check_details` *Query R CMD check results and parameters*

Description

Query R CMD check results and parameters

Usage

```
check_details(check)
```

Arguments

`check`	A check result.

Value

A named list with elements:

- `package`: package name.
- `version`: package version.
- `rversion`: R version.
- `notes`: character vector of check NOTEs, each NOTE is an element.
- `warnings`: character vector of check WARNINGs, each WARNING is an element.
- `errors`: character vector of check ERRORs, each ERROR is an element. A check timeout adds an ERROR to this vector.
- `platform`: check platform
- `checkdir`: check directory.
- `install_out`: the output of the installation, contents of the `00install.out` file. A single string.
- `description`: the contents of the DESCRIPTION file of the package. A single string.
- `session_info`: the output of `sessioninfo::session_info()`, from the R session performing the checks.
- `checkdir`: the path to the check directory, if it hasn't been cleaned up yet, or `NA`. The check directory is automatically cleaned up, when the check object is deleted (garbage collected).

- `cran`: whether it is a CRAN packaged package.
- `bioc`: whether it is a BioConductor package.

`compare_checks` *Compare a set of check results to another check result*

Description

Compare a set of check results to another check result

Usage

```
compare_checks(old, new)
```

Arguments

`old`	A check result, or a list of check results.
`new`	A check result.

Value

An `rcmdcheck_comparison` object with fields:

- `package`: the name of the package, string,
- `versions`: package versions, length two character,
- `status`: comparison status, see below,
- `old`: list of `rcmdcheck` objects the old check(s),
- `new`: `rcmdcheck` object, the new check,
- `cmp`:

See Also

Other check comparisons: `compare_to_cran()`

compare_to_cran *Compare a check result to CRAN check results*

Description

Compare a check result to CRAN check results

Usage

```
compare_to_cran(check, flavours = cran_check_flavours(check$package))
```

Arguments

check	A check result.
flavours	CRAN check flavour(s) to use. By default all platforms are used.

Value

An rmdcheck_comparison object.

See Also

Other check comparisons: compare_checks()

cran_check_flavours *Download and show all CRAN check flavour platforms*

Description

If the package argument is NULL, then all current platforms are downloaded. If the package argument is specified, then all flavours used for the latest package checks for that package, are downloaded and returned.

Usage

```
cran_check_flavours(package = NULL)
```

Arguments

package	CRAN package name or NULL.

Value

Character vector of platform ids.

Examples

```
## Not run:
cran_check_flavours()
cran_check_flavours("simplegraph")

## End(Not run)
```

`cran_check_results` *Download and parse R CMD check results from CRAN*

Description

Download and parse R CMD check results from CRAN

Usage

```
cran_check_results(
  package,
  flavours = cran_check_flavours(package),
  quiet = FALSE
)
```

Arguments

`package`	Name of a single package to download the checks for.
`flavours`	CRAN check flavours to use. Defaults to all flavours that were used to check the package.
`quiet`	Whether to omit the download progress bars.

Value

A list of `rcmdcheck` objects.

`parse_check` *Parse* `R CMD check` *results from a file or string*

Description

At most one of `file` or `text` can be given. If both are `NULL`, then the current working directory is checked for a `00check.log` file.

Usage

```
parse_check(file = NULL, text = NULL, ...)
```

Arguments

file	The 00check.log file, or a directory that contains that file. It can also be a connection object.
text	The contents of a 00check.log file.
...	Other arguments passed onto the constructor. Used for testing.

Value

An rcmdcheck object, the check results.

See Also

parse_check_url

---

parse_check_url *Shorthand to parse R CMD check results from a URL*

Description

Shorthand to parse R CMD check results from a URL

Usage

```
parse_check_url(url, quiet = FALSE)
```

Arguments

url	URL to parse the results from. Note that it should not contain HTML markup, just the text output.
quiet	Passed to download.file.

Value

An rcmdcheck object, the check results.

See Also

parse_check

print.rcmdcheck *Print R CMD check results*

Description

Print R CMD check results

Usage

```
## S3 method for class 'rcmdcheck'
print(
  x,
  header = TRUE,
  test_output = getOption("rcmdcheck.test_output", FALSE),
  ...
)
```

Arguments

x	Check result object to print.
header	Whether to print a header.
test_output	if TRUE, include the test output in the results, if there are no test failures. If some tests fail, then only the failures are printed.
...	Additional arguments, currently ignored.

print.rcmdcheck_comparison *Print R CMD check result comparisons*

Description

See compare_checks() and compare_to_cran().

Usage

```
## S3 method for class 'rcmdcheck_comparison'
print(x, header = TRUE, ...)
```

Arguments

x	R CMD check result comparison object.
header	Whether to print the header. You can suppress the header if you want to use the printout as part of another object's printout.
...	Additional arguments, currently ignored.

rcmdcheck *Run R CMD check from R and Capture Results*

Description

Run R CMD check from R programmatically, and capture the results of the individual checks.

Runs `R CMD check` as an external command, and parses its output and returns the check failures.

Usage

```
rcmdcheck(
  path = ".",
  quiet = FALSE,
  args = character(),
  build_args = character(),
  check_dir = NULL,
  libpath = .libPaths(),
  repos = getOption("repos"),
  timeout = Inf,
  error_on = Sys.getenv("RCMDCHECK_ERROR_ON", c("never", "error", "warning",
    "note")[1]),
  env = character()
)
```

Arguments

`path`	Path to a package tarball or a directory.
`quiet`	Whether to print check output during checking.
`args`	Character vector of arguments to pass to `R CMD check`. Pass each argument as a single element of this character vector (do not use spaces to delimit arguments like you would in the shell). For example, to skip running of examples and tests, use `args = c("--no-examples", "--no-tests")` and not `args = "--no-examples --no-tests"`. (Note that instead of the `--output` option you should use the `check_dir` argument, because `--output` cannot deal with spaces and other special characters on Windows.)
`build_args`	Character vector of arguments to pass to `R CMD build`. Pass each argument as a single element of this character vector (do not use spaces to delimit arguments like you would in the shell). For example, `build_args = c("--force", "--keep-empty-dirs")` is a correct usage and `build_args = "--force --keep-empty-dirs"` is incorrect.
`check_dir`	Path to a directory where the check is performed. If this is not `NULL`, then the a temporary directory is used, that is cleaned up when the returned object is garbage collected.
`libpath`	The library path to set for the check. The default uses the current library path.

`repos`	The `repos` option to set for the check. This is needed for cyclic dependency checks if you use the `--as-cran` argument. The default uses the current value.
`timeout`	Timeout for the check, in seconds, or as a base::difftime object. If it is not finished before this, it will be killed. `Inf` means no timeout. If the check is timed out, that is added as an extra error to the result object.
`error_on`	Whether to throw an error on `R CMD check` failures. Note that the check is always completed (unless a timeout happens), and the error is only thrown after completion. If `"never"`, then no errors are thrown. If `"error"`, then only `ERROR` failures generate errors. If `"warning"`, then `WARNING` failures generate errors as well. If `"note"`, then any check failure generated an error. Its default can be modified with the `RCMDCHECK_ERROR_ON` environment variable. If that is not set, then `"never"` is used.
`env`	A named character vector, extra environment variables to set in the check process.

Details

See rcmdcheck_process if you need to run `R CMD check` in a background process.

Value

An S3 object (list) with fields `errors`, `warnings` and `notes`. These are all character vectors containing the output for the failed check.

Turning off package checks

Sometimes it is useful to programmatically turn off some checks that may report check NOTEs. rcmdcehck provides two ways to do this.

First, you may declare in `DESCRIPTION` that you don't want to see NOTEs that are accepted on CRAN, with this entry:

```
Config/rcmdcheck/ignore-inconsequential-notes: true
```

Currently, this will make rcmdcheck ignore the following notes:

- report large package sizes (`_R_CHECK_PKG_SIZES_ = FALSE`),
- check cross-references in Rd files (`_R_CHECK_RD_XREFS_ = FALSE`),
- NOTE if package requires GNU make (`_R_CHECK_CRAN_INCOMING_NOTE_GNU_MAKE_ = FALSE`),
- report marked non-ASCII strings in datasets (`_R_CHECK_PACKAGE_DATASETS_SUPPRESS_NOTES_ = TRUE`).

The second way is more flexible, and lets you turn off individual checks via setting environment variables. You may provide a `tools/check.env` *environment file* in your package with the list of environment variable settings that rcmdcheck will automatically use when checking the package. See Startup for the format of this file.

Here is an example `tools/check.env` file:

```
# Report if package size is larger than 10 megabytes
_R_CHECK_PKG_SIZES_THRESHOLD_=10

# Do not check Rd cross references
_R_CHECK_RD_XREFS_=false

# Do not report if package requires GNU make
_R_CHECK_CRAN_INCOMING_NOTE_GNU_MAKE_=false

# Do not check non-ASCII strings in datasets
_R_CHECK_PACKAGE_DATASETS_SUPPRESS_NOTES_=true
```

See the "R internals" manual and the R source code for the environment variables that control the checks.

Note that `Config/rcmdcheck/ignore-inconsequential-notes` and the `check.env` file are only supported by rcmdcheck, and running `R CMD check` from a shell (or GUI) will not use them.

`rcmdcheck-config` *rcmdcheck configuration*

Description

Options take precedence over environment variables. E.g. if both the `RCMDCHECK_NUM_COLORS` environment variables and the `rcmdcheck.num_colors` option are set, then the latter is used.

Details

rcmdcheck uses the cli package for much of its output, so you can configure the output via cli, see cli::cli-config.

Package configration is defined in the `DESCRIPTION` file of the checked package. E.g.:

```
Config/build/clean-inst-doc: FALSE
```

Environment variables

- `R_PROFILE_USER`: standard R environment variable to configure the path to the user level R profile. See base::R_PROFILE_USER.
- `RCMDCHECK_BASE_URL`: URL to the root of the CRAN check web page. You can use this to select an alternative CRAN mirror. Defaults to `https://cran.r-project.org/web/checks/`.
- `RCMDCHECK_DETAILS_URL`: URL to the root of the CRAN check output page. Defaults to `https://www.r-project.org/nosvn/R.check/`.
- `RCMDCHECK_ERROR_ON`: the default value for the `error_on` argument of `rcmdcheck()`.
- `RCMDCHECK_FLAVOURS_URL` URL to the CRAN check flavours page. You can use this to select an alternative CRAN mirror. Defaults to `https://cran.r-project.org/web/checks/check_flavors.html`.

- RCMDCHECK_NUM_COLORS: the number of ANSI colors to use in the output. It can be used to override the number of colors detected or configured by the cli package. See cli::num_ansi_colors(). This configuration is only used for the output of rcmdcheck and it does not affect the examples and test cases (and other code) of the checked package. It not set, then the default of cli is uesed. The corresponding option is rcmdcheck.num_colors.
- RCMDCHECK_TIMESTAMP_LIMIT: lower limit is seconds, above which rcmdcheck adds time stamps to the individual check steps. It may be fractional. Defaults to 1/3 of a second. The corresponding option is rcmdcheck.timestamp_limit.
- RCMDCHECK_USE_RSTUDIO_PANDOC: Flag (true or false). If true, then rcmdcheck *always* puts RStudio's pandoc (if available) on the path. If false, then it *never* does that. If not set, or set to a different value, then pandoc is put on the path only if it is not already available. RStudio's pandoc is detected via an RSTUDIO_PANDOC environment variable.
- RCMDCHECK_LOAD_CHECK_ENV: you can use this environment variable suppress loading environment variables from the tools/check.env file. See rcmdcheck() for details.
- RSTUDIO_PANDOC: if set, rcmdcheck adds this environment variable to the PATH if pandoc is not on the PATH already. It is usually set in RStudio. See also the RCMDCHECK_USE_RSTUDIO_PANDOC environment variable.

Options

- rcmdcheck.num_colors: the number of ANSI colors to use in the output. It can be used to override the number of colors detected or configured by the cli package. See cli::num_ansi_colors(). This configuration is only used for the output of rcmdcheck and it does not affect the examples and test cases (and other code) of the checked package. It not set, then the default of cli is uesed. The corresponding environment variable is RCMDCHECK_NUM_COLORS.
- rcmdcheck.test_output: Flag (TRUE or FALSE), whether print.rcmdcheck() should print the full test output if there are no test failures. If some tests fail, then only the failures are printed, independently of this option.
- rcmdcheck.timestamp_limit: lower limit is seconds, above which rcmdcheck adds time stamps to the individual check steps. It may be fractional. Defaults to 1/3 of a second. The corresponding environment variable is RCMDCHECK_TIMESTAMP_LIMIT.

Package configuration:

- Config/build/clean-inst-doc: Flag (TRUE or FALSE) to specify if the inst/doc directory should be cleaned up when building a package directory. If not specified, then NULL is used. See the clean_doc option of pkgbuild::build() for more details.

---

rcmdcheck_process *Run an* R CMD check *process in the background*

Description

rcmdcheck_process is an R6 class, that extends the callr::rcmd_process class (which in turn extends processx::process.

Usage

```
cp <- rcmdcheck_process$new(path = ".", args = character(),
         build_args = character(), check_dir = NULL,
         libpath = .libPaths(), repos = getOption("repos"))

cp$parse_results()
```

Other methods are inherited from callr::rcmd_process and processx::process.

Note that you calling the `get_output_connection` and `get_error_connection` method on this is not a good idea, because then the stdout and/or stderr of the process will not be collected for `parse_results()`.

You can still use the `read_output_lines()` and `read_error_lines()` methods to read the standard output and error, `parse_results()` is not affected by that.

Arguments

- `cp`: A new rcmdcheck_process object.
- `path`: Path to a package tree or a package archive file. This is the package to check.
- `args`: Command line arguments to `R CMD check`.
- `build_args`: Command line arguments to `R CMD build`.
- `check_dir`: Directory for the results.
- `libpath`: The library path to set for the check.
- `repos`: The `repos` option to set for the check. This is needed for cyclic dependency checks if you use the `--as-cran` argument. The default uses the current value.
- `env`: A named character vector, extra environment variables to set in the check process.

Details

Most methods are inherited from callr::rcmd_process and processx::process.

`cp$parse_results()` parses the results, and returns an S3 object with fields `errors`, `warnnigs` and `notes`, just like `rcmdcheck()`. It is an error to call it before the process has finished. Use the `wait()` method to wait for the check to finish, or the `is_alive()` method to check if it is still running.

---

xopen.rcmdcheck *Open the check directory in a file browser window*

Description

Open the check directory in a file browser window

Usage

```
## S3 method for class 'rcmdcheck'
xopen(target, app = NULL, quiet = FALSE, ...)
```

Arguments

target	rcmdcheck() result.
app	Specify the app to open target with, and its arguments, in a character vector. Note that app names are platform dependent.
quiet	Whether to echo the command to the screen, before running it.
...	Additional arguments, not used currently.

Index