Package ‘onetime’

November 22, 2022

Type Package

Title Run Code Only Once

Version 0.1.0

Author David Hugh-Jones [aut, cre]

Maintainer David Hugh-Jones <davidhughjones@gmail.com>

Description Allows code to be run only once on a given computer, using lockfiles. Typical use cases include startup messages shown only when a package is loaded for the very first time.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.2.2

Imports rappdirs, filelock

URL https://github.com/hughjonesd/onetime, https://hughjonesd.github.io/onetime/

BugReports https://github.com/hughjonesd/onetime/issues

Suggests callr, covr, devtools, mockr, rlang, testthat (>= 2.1.0), withr

NeedsCompilation no

Repository CRAN

Date/Publication 2022-11-22 14:00:05 UTC

R topics documented:

`check_ok_to_store` *Check if the package has permission to store files on the user's computer*

Description

The onetime package works by storing lockfiles in `rappdirs::user_config_dir()`. It won't do so unless permission has been granted. Before using `onetime` functions, package authors should call `check_ok_to_store(ask = TRUE)` in an interactive session, in functions which are called directly from the command line.

Usage

```
check_ok_to_store(
  ask = FALSE,
  message = "The onetime package requests to store files in '%s'.",
  confirm_prompt = "Is this OK? [Yn] ",
  confirm_answers = c("Y", "y", "Yes", "yes", "YES"),
  default_answer = "Y"
)
```

Arguments

`ask`	`TRUE` to ask the user for permission.
`message`	Message to display to the user.
`confirm_prompt`	Character string. Question to prompt the user to hide the message in future.
`confirm_answers`	Character vector. Answers which will cause the message to be hidden in future.
`default_answer`	Character string. Default answer if user simply presses return.

Details

If your package is not used interactively, a workaround is to call `set_ok_to_store()`. This grants permission and prints an informative message. Package owners should *only* call this if they cannot ask explicitly.

`onetime_message_confirm()` is an exception: by default it doesn't require global permission to store files, since the user accepting "Don't show this again" is considered sufficient.

`ask = TRUE` asks the user, if he or she has not already given permission, and if the session is `interactive()`.

Remaining parameters are passed to `onetime_message_confirm()` in this case, and ignored otherwise. A `"%s"` in `message` will be replaced by the onetime storage directory.

Value

`TRUE` if:

- We already have permission;
- `ask` is `TRUE`, we are in an interactive session and the user gives us permission;
- `options("onetime.dir")` is set to a non-`NULL` value.

Otherwise `FALSE`.

Examples

```
check_ok_to_store()
```

`onetime` *Run code only once*

Description

Onetime allows package authors to run code only once (ever) for a given user. It does so by writing a file, typically to a folder in the user's configuration directory as given by `rappdirs::user_config_dir()`. The user can set an alternative filepath using `options("onetime.dir")`.

Details

It is package authors' responsibility to check for permission to store lockfiles. This may have been done already by another package if onetime was already installed. You can ask permission interactively on the command line by calling `check_ok_to_store()` with `ask = TRUE`.

Core functions include:

- `onetime_do()` runs arbitrary code only once.
- `onetime_warning()` and friends print a warning or message only once.
- `onetime_message_confirm()` prints a message and asks "Show this message again?"
- `onetime_rlang_warn()` and `onetime_rlang_inform()` print messages using functions from the rlang package.
- `onetime_only()` returns a function that runs only once.
- `check_ok_to_store()` and `set_ok_to_store()` check for or grant permission to store lockfiles on the user's computer.

Example:

```
library(onetime)
ids  <- paste0("onetime-readme-", sample(1e9, 4))


for (i in 1:5) {
  cat("Loop ", i, " of 5\n")
  onetime_do(cat("This command will only be run once.\n"), id = ids[1])
  onetime_warning("This warning will only be shown once.", id = ids[2])
  onetime_message("This message will only be shown once.", id = ids[3])
}
```

```
## Loop  1  of 5
## This command will only be run once.

## Warning: This warning will only be shown once.

## This message will only be shown once.

## Loop  2  of 5
## Loop  3  of 5
## Loop  4  of 5
## Loop  5  of 5
```

```
# Meanwhile, in a separate process:
library(callr)
result <- callr::r(function (ids) {
  onetime::onetime_message("This message with an existing ID will not be shown.", id = ids[1])
  onetime::onetime_message("This message with a new ID will be shown.", id = ids[4])
}, show = TRUE, args = list(ids = ids))
```

```
## This message with an existing ID will not be shown.
## This message with a new ID will be shown.
```

`onetime-rlang` *Print a warning or message only once using* `rlang` *functions*

Description

If you use these you will need to add `"rlang"` to your package dependencies.

Usage

```
onetime_rlang_warn(
  ...,
  id = calling_package(),
  path = default_lockfile_dir(),
  expiry = NULL,
  without_permission = "warn"
)

onetime_rlang_inform(
  ...,
  id = calling_package(),
  path = default_lockfile_dir(),
  expiry = NULL,
  without_permission = "warn"
)
```

Arguments

`...`	Passed to `rlang::warn()` or `rlang::inform()`.
`id`	Unique ID string. By default, name of the calling package.
`path`	Directory to store lockfiles.
`expiry`	`difftime()` or e.g. `lubridate::duration()` object. After this length of time, code will be run again.
`without_permission`	Character string. What to do if the user hasn't given permission to store files? `"warn"` runs the action with an extra warning; `"run"` runs the action; `"pass"` does nothing and returns the default; `"stop"` throws an error; `"ask"` asks for permission, after running the action but before recording it on disk.

Value

`TRUE` if the message/warning was shown, `FALSE` otherwise.

Examples

```
oo <- options(onetime.dir = tempdir(check = TRUE))
id <- sample(10000L, 1)

for (n in 1:3) {
  onetime_rlang_warn(c("rlang-style warning", i = "Extra info"), id = id)
}

onetime_reset(id = id)
options(oo)
```

`onetime_been_done` *Check if a onetime call has already been made*

Description

Check if a onetime call has already been made

Usage

```
onetime_been_done(
  id = calling_package(),
  path = default_lockfile_dir(),
  expiry = NULL
)
```

Arguments

`id`	Unique ID string. By default, name of the calling package.
`path`	Directory to store lockfiles.
`expiry`	`difftime()` or e.g. `lubridate::duration()` object. After this length of time, code will be run again.

Value

`TRUE` if the call has been recorded (within the `expiry` time, if given).

Examples

```
oo <- options(onetime.dir = tempdir(check = TRUE))
id <- sample(10000L, 1)

onetime_been_done(id = id)
onetime_message("Creating an ID",  id = id)
onetime_been_done(id = id)

onetime_reset(id = id)
options(oo)
```

`onetime_do` *Run code only once*

Description

This function runs an expression just once. It then creates a lockfile recording a unique ID which will prevent the expression being run again.

Usage

```
onetime_do(
  expr,
  id = calling_package(),
  path = default_lockfile_dir(),
  expiry = NULL,
  default = NULL,
  without_permission = c("warn", "run", "stop", "pass", "ask")
)
```

Arguments

`expr`	The code to evaluate. An R statement or `expression()` object.
`id`	Unique ID string. By default, name of the calling package.
`path`	Directory to store lockfiles.
`expiry`	`difftime()` or e.g. `lubridate::duration()` object. After this length of time, code will be run again.
`default`	Value to return if `expr` was not executed.
`without_permission`	Character string. What to do if the user hasn't given permission to store files? `"warn"` runs the action with an extra warning; `"run"` runs the action; `"pass"` does nothing and returns the default; `"stop"` throws an error; `"ask"` asks for permission, after running the action but before recording it on disk.

Details

Calls are identified by `id`. If you use the same value of `id` across different calls to `onetime_do()` and similar functions, only the first call will get made.

By default, `id` is just the name of the calling package. This is for the common use case of a single call within a package (e.g. at first startup). If you want to use multiple calls, or if the calling code is not within a package, then you *must* set `id` explicitly. If you are working in a large project with many contributors, it is *strongly recommended to set* `id` *explicitly*.

The default `path`, where lockfiles are stored, is within `rappdirs::user_config_dir()` unless overridden by `options("onetime.dir")`. If the lockfile cannot be written (e.g. because the user has not given permission to store files on his or her computer), then the call will still be run, so it may be run repeatedly. Conversely, if the call gives an error, the lockfile is still written.

Value

The value of `expr`, invisibly; or `default` if `expr` was not run because it had been run already.

Examples

```
oo <- options(onetime.dir = tempdir(check = TRUE))
id <- sample(10000L, 1L)

for (n in 1:3) {
  onetime_do(print("printed once"), id = id)
}

onetime_reset(id = id)
options(oo)
```

onetime_message_confirm *Print a message, and ask for confirmation to hide it in future*

Description

This uses `readline()` to ask the user if the message should be shown again in future. In a non-interactive session, it does nothing.

Usage

```
onetime_message_confirm(
  message,
  id = calling_package(),
  path = default_lockfile_dir(),
  expiry = NULL,
  confirm_prompt = "Show this message again? [yN] ",
  confirm_answers = c("N", "n", "No", "no"),
  default_answer = "N",
  require_permission = FALSE,
  without_permission = "warn"
)
```

Arguments

`message`	Message to display to the user.
`id`	Unique ID string. By default, name of the calling package.
`path`	Directory to store lockfiles.
`expiry`	`difftime()` or e.g. `lubridate::duration()` object. After this length of time, code will be run again.
`confirm_prompt`	Character string. Question to prompt the user to hide the message in future.
`confirm_answers`	Character vector. Answers which will cause the message to be hidden in future.
`default_answer`	Character string. Default answer if user simply presses return.
`require_permission`	Logical. Ask permission to store files on the user's computer, if this hasn't been granted? Setting this to `FALSE` overrides `without_permission`.
`without_permission`	Character string. What to do if the user hasn't given permission to store files? `"warn"` runs the action with an extra warning; `"run"` runs the action; `"pass"` does nothing and returns the default; `"stop"` throws an error; `"ask"` asks for permission, after running the action but before recording it on disk.

Details

By default, the message will be hidden if the user answers "n", "No", or "N", or just presses return to the prompt question.

Unlike other `onetime` functions, `onetime_message_confirm()` doesn't by default require permission to store files on the user's computer. The assumption is that saying "Don't show this message again" counts as granting permission (just for this one message). You can ask for broader permission by setting `require_permission = TRUE` and `without_permission = "ask"`.

Value

`NULL` if the message was not shown (shown already or non-interactive session). `TRUE` if the user confirmed (i.e. asked to hide the message). `FALSE` if the message was shown but the user did not confirm. Note that by default, `TRUE` is returned when the user answers "no" to "Show this message again?"

Examples

```
oo <- options(onetime.dir = tempdir(check = TRUE))
id <- sample(10000L, 1L)

onetime_message_confirm("A message to show one or more times", id = id)

onetime_reset(id = id)
options(oo)
```

`onetime_only` *Wrap a function to be called only once*

Description

This takes a function and returns the same function wrapped by `onetime_do()`. Use it for code which should run only once, but which may be called from multiple locations. This frees you from having to use the same `id` multiple times.

Usage

```
onetime_only(
  .f,
  id = calling_package(),
  path = default_lockfile_dir(),
  without_permission = "warn"
)
```

Arguments

`.f`	A function
`id`	Unique ID string. By default, name of the calling package.
`path`	Directory to store lockfiles.
`without_permission`	Character string. What to do if the user hasn't given permission to store files? `"warn"` runs the action with an extra warning; `"run"` runs the action; `"pass"` does nothing and returns the default; `"stop"` throws an error; `"ask"` asks for permission, after running the action but before recording it on disk.

Value

A wrapped function.

See Also

`onetime_do()`

Examples

```
oo <- options(onetime.dir = tempdir(check = TRUE))
id <- sample(10000L, 1)

sample_once <- onetime_only(sample, id = id)
sample_once(1:10)
sample_once(1:10)

onetime_reset(id)
options(oo)
```

`onetime_reset` *Reset a onetime call by ID*

Description

Reset a onetime call by ID

Usage

```
onetime_reset(id = calling_package(), path = default_lockfile_dir())
```

Arguments

`id`	Unique ID string. By default, name of the calling package.
`path`	Directory to store lockfiles.

Value

The result of `file.remove()`, invisibly.

Examples

```
oo <- options(onetime.dir = tempdir(check = TRUE))
id <- sample(10000L, 1)

onetime_message("will be shown",  id = id)
onetime_message("won't be shown", id = id)
onetime_reset(id = id)
onetime_message("will be shown",  id = id)

onetime_reset(id = id)
options(oo)
```

`onetime_warning` *Print a warning or message only once*

Description

These functions use `onetime_do()` to print a warning or message just once.

Usage

```
onetime_warning(
  ...,
  id = calling_package(),
  path = default_lockfile_dir(),
  expiry = NULL,
  without_permission = "warn"
)

onetime_message(
  ...,
  id = calling_package(),
  path = default_lockfile_dir(),
  expiry = NULL,
  without_permission = "warn"
)

onetime_startup_message(
  ...,
  id = calling_package(),
  path = default_lockfile_dir(),
  expiry = NULL,
  without_permission = "warn"
)
```

Arguments

`...`	Passed to `warning()`, `message()` or `packageStartupMessage()`.
`id`	Unique ID string. By default, name of the calling package.
`path`	Directory to store lockfiles.
`expiry`	`difftime()` or e.g. `lubridate::duration()` object. After this length of time, code will be run again.
`without_permission`	Character string. What to do if the user hasn't given permission to store files? `"warn"` runs the action with an extra warning; `"run"` runs the action; `"pass"` does nothing and returns the default; `"stop"` throws an error; `"ask"` asks for permission, after running the action but before recording it on disk.

Value

`TRUE` if the message/warning was shown, `FALSE` otherwise.

See Also

`onetime_do()`

Examples

```
oo <- options(onetime.dir = tempdir(check = TRUE))
id <- sample(10000L, 1)

for (n in 1:3) {
  onetime_warning("will be shown once", id = id)
}

onetime_reset(id = id)
options(oo)
```

`set_ok_to_store` *Grant or revoke permission to store lockfiles on the user's computer*

Description

End users may use this from the command line. Package authors should *only* call it if they cannot ask for permission interactively using `check_ok_to_store(ask = TRUE)`.

Usage

```
set_ok_to_store(ok = TRUE)
```

Arguments

`ok`	`TRUE` to grant permission to store lockfiles, `FALSE` to revoke it and unset `options("onetime.dir")`.

Value

TRUE if the operation succeeded.

Examples

```
## Not run:
set_ok_to_store()

## End(Not run)
```

Index