

Package ‘withr’

August 29, 2016

Encoding UTF-8

Title Run Code 'With' Temporarily Modified Global State

Version 1.0.2

Description A set of functions to run code 'with' safely and temporarily modified global state. Many of these functions were originally a part of the 'devtools' package, this provides a simple package with limited dependencies to provide access to these functions.

URL <http://github.com/jimhester/withr>

BugReports <http://github.com/jimhester/withr/issues>

Depends R (>= 3.0.2)

License GPL (>= 2)

LazyData true

Imports stats, graphics

Suggests testthat

Collate 'with_R.R' 'collate.R' 'dir.R' 'env.R' 'libpaths.R' 'locale.R' 'makevars.R' 'options.R' 'par.R' 'path.R' 'wrap.R' 'sink.R' 'with.R'

RoxygenNote 5.0.1

NeedsCompilation no

Author Jim Hester [aut, cre],
Kirill Müller [aut],
Hadley Wickham [aut],
Winston Chang [aut],
RStudio [cph]

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

Repository CRAN

Date/Publication 2016-06-20 17:32:02

R topics documented:

withr	2
with_collate	3
with_dir	4
with_envvar	5
with_libpaths	5
with_locale	6
with_makevars	7
with_options	8
with_par	8
with_path	9
with_sink	10
with_temp_libpaths	11
Index	12

withr	<i>Execute code in temporarily altered environment</i>
-------	--------------------------------------------------------

Description

All functions prefixed by `with_` work as follows. First, a particular aspect of the global environment is modified (see below for a list). Then, custom code (passed via the `code` argument) is executed. Upon completion or error, the global environment is restored to the previous state.

Arguments pattern

<code>new</code>	[various]	Values for setting
<code>code</code>	[any]	Code to execute in the temporary environment
<code>...</code>		Further arguments

Usage pattern

```
with_...(new, code, ...)
```

withr functions

- `with_collate`: collation order
- `with_dir`: working directory
- `with_envvar`: environment variables
- `with_libpaths`: library paths, replacing current libpaths
- `with_locale`: any locale setting

- `with_makevars`: Makevars variables
- `with_options`: options
- `with_par`: graphics parameters
- `with_path`: PATH environment variable
- `with_sink`: output redirection

Creating new "with" functions

All `with_` functions are created by a helper function, `with_`. This function accepts two arguments: a setter function and an optional resetter function. The setter function is expected to change the global state and return an "undo instruction". This undo instruction is then passed to the resetter function, which changes back the global state. In many cases, the setter function can be used naturally as resetter.

Examples

```
getwd()
with_dir(tempdir(), getwd())
getwd()

Sys.getenv("HADLEY")
with_envvar(c("HADLEY" = 2), Sys.getenv("HADLEY"))
Sys.getenv("HADLEY")

with_envvar(c("A" = 1),
  with_envvar(c("A" = 2), action = "suffix", Sys.getenv("A"))
)
```

with_collate	<i>Collation Order</i>
--------------	------------------------

Description

Temporarily change collation order by changing the value of the LC_COLLATE locale.

Usage

```
with_collate(new, code)
```

Arguments

new	[character(1)] New collation order
code	[any] Code to execute in the temporary environment

Value

[any]
The results of the evaluation of the code argument.

See Also

[withr](#) for examples

with_dir	<i>Working directory</i>
----------	--------------------------

Description

Temporarily change the current working directory.

Usage

```
with_dir(new, code)
```

Arguments

new	[character(1)] New working directory
code	[any] Code to execute in the temporary environment

Value

[any]
The results of the evaluation of the code argument.

See Also

[withr](#) for examples

[setwd](#)

with_envvar	<i>Environment variables</i>
-------------	------------------------------

Description

Temporarily change system environment variables.

Usage

```
with_envvar(new, code, action = "replace")
```

Arguments

new	[named character] New environment variables
code	[any] Code to execute in the temporary environment
action	should new values "replace", "prefix" or "suffix" existing variables with the same name.

Details

if NA is used those environment variables will be unset. If there are any duplicated variable names only the last one is used.

Value

[any]
The results of the evaluation of the code argument.

See Also

[withr](#) for examples

[Sys.setenv](#)

with_libpaths	<i>Library paths</i>
---------------	----------------------

Description

Temporarily change library paths.

Usage

```
with_libpaths(new, code, action = "replace")
```

Arguments

new	[character] New library paths
code	[any] Code to execute in the temporary environment
action	[character(1)] should new values "replace", "prefix" or "suffix" existing paths.

Value

[any]
The results of the evaluation of the code argument.

See Also

[withr](#) for examples

[.libPaths](#)

Other libpaths: [with_temp_libpaths](#)

with_locale	<i>Locale settings</i>
-------------	------------------------

Description

Temporarily change locale settings.

Usage

```
with_locale(new, code)
```

Arguments

new	[named character] New locale settings
code	[any] Code to execute in the temporary environment

Details

Setting the LC_ALL category is currently not implemented.

Value

[any]
The results of the evaluation of the code argument.

See Also[withr](#) for examples[Sys.setlocale](#)

with_makevars	<i>Makevars variables</i>
---------------	---------------------------

Description

Temporarily change contents of an existing Makevars file.

Usage

```
with_makevars(new, code, path = file.path("~", ".R", "Makevars"),
  assignment = c("=", ":", "?=", "+="))
```

Arguments

new	[named character] New variables and their values
code	[any] Code to execute in the temporary environment
path	[character(1)] location of existing Makevars file to modify.
assignment	[character(1)] assignment type to use.

Details

If no Makevars file exists or the fields in new do not exist in the existing Makevars file then the fields are added to the new file. Existing fields which are not included in new are appended unchanged. Fields which exist in Makevars and in new are modified to use the value in new.

Value

[any]
The results of the evaluation of the code argument.

See Also[withr](#) for examples

with_options	<i>Options</i>
--------------	----------------

Description

Temporarily change global options.

Usage

```
with_options(new, code)
```

Arguments

new	[named list] New options and their values
code	[any] Code to execute in the temporary environment

Value

[any]
The results of the evaluation of the code argument.

See Also

[withr](#) for examples
[options](#)

with_par	<i>Graphics parameters</i>
----------	----------------------------

Description

Temporarily change graphics parameters.

Usage

```
with_par(new, code, no.readonly = FALSE)
```

Arguments

new	[named list] New graphics parameters and their values
code	[any] Code to execute in the temporary environment
no.readonly	[logical(1)] see par documentation.

Value

[any]
The results of the evaluation of the code argument.

See Also

[withr](#) for examples

[par](#)

<code>with_path</code>	<i>PATH environment variable</i>
------------------------	----------------------------------

Description

Temporarily change the system search path.

Usage

```
with_path(new, code, action = "prefix")
```

Arguments

<code>new</code>	[character] New PATH entries
<code>code</code>	[any] Code to execute in the temporary environment
<code>action</code>	[character(1)] Should new values "replace", "prefix" or "suffix" existing paths

Value

[any]
The results of the evaluation of the code argument.

See Also

[withr](#) for examples

[Sys.setenv](#)

with_sink	<i>Output redirection</i>
-----------	---------------------------

Description

Temporarily divert output to a file via [sink](#). For sinks of type message, an error is raised if such a sink is already active.

Usage

```
with_output_sink(new, code, append = FALSE, split = FALSE)
```

```
with_message_sink(new, code, append = FALSE)
```

Arguments

new	[character(1) connection] A writable connection or a character string naming the file to write to. Passing NULL will throw an error.
code	[any] Code to execute in the temporary environment
append	logical. If TRUE, output will be appended to file; otherwise, it will overwrite the contents of file.
split	logical: if TRUE, output will be sent to the new sink and to the current output stream, like the Unix program tee.

Value

[any]
The results of the evaluation of the code argument.

See Also

[withr](#) for examples

[sink](#)

`with_temp_libpaths` *Library paths*

Description

Temporarily prepend a new temporary directory to the library paths.

Usage

`with_temp_libpaths(code)`

Arguments

`code` [any]
Code to execute in the temporary environment

Value

[any]
The results of the evaluation of the code argument.

See Also

[withr](#) for examples
[.libPaths](#)
Other libpaths: [with_libpaths](#)

Index

.libPaths, [6](#), [11](#)
connection, [10](#)
options, [8](#)
par, [8](#), [9](#)
setwd, [4](#)
sink, [10](#)
Sys.setenv, [5](#), [9](#)
Sys.setlocale, [7](#)

with_, [3](#)
with_collate, [2](#), [3](#)
with_dir, [2](#), [4](#)
with_envvar, [2](#), [5](#)
with_libpaths, [2](#), [5](#), [11](#)
with_locale, [2](#), [6](#)
with_makevars, [3](#), [7](#)
with_message_sink (with_sink), [10](#)
with_options, [3](#), [8](#)
with_output_sink (with_sink), [10](#)
with_par, [3](#), [8](#)
with_path, [3](#), [9](#)
with_sink, [3](#), [10](#)
with_temp_libpaths, [6](#), [11](#)
withr, [2](#), [4–11](#)
withr-package (withr), [2](#)