Package: mockr (via r-universe)

October 9, 2024

Title Mocking in R

Version 0.2.0.9002

Date 2022-12-30

Description Provides a means to mock a package function, i.e., temporarily substitute it for testing. Designed as a drop-in replacement for the now deprecated 'testthat::with_mock()' and 'testthat::local_mock()'.

License GPL-3

URL https://krlmlr.github.io/mockr/, https://github.com/krlmlr/mockr

BugReports https://github.com/krlmlr/mockr/issues

Imports rlang, withr

Suggests covr, fs, knitr, pkgload, rmarkdown, testthat, usethis

VignetteBuilder knitr

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

Repository https://mrc-ide.r-universe.dev

RemoteUrl https://github.com/mrc-ide/mockr

RemoteRef main

RemoteSha 61ff4551e57e5e22ac95006f8b7b38ccf5f0a91a

Contents

get_mock_env *Get environment for mocking*

Description

Called by default from `with_mock()` to determine the environment where to update mocked functions. This function is exported to help troubleshooting.

Usage

```
get_mock_env(.parent = parent.frame())
```

Arguments

`.parent`	`[environment]` the environment in which to evaluate the expressions, defaults to `parent.frame()`. Usually doesn't need to be changed.

Details

This function works differently depending on `testthat::is_testing()`.

Outside testthat, `topenv(.parent)` is returned. This was the default for mockr < 0.1.0 and works for many cases.

In testthat, `asNamespace("<package>")` for the tested package is returned. The tested package is determined via `testthat::testing_package()`. If this is empty (e.g. if a `test_that()` block is run in interactive mode), this function looks in the search path for packages loaded by `pkgload::load_all()`.

local_mock *Mock functions in a package*

Description

`local_mock()` temporarily substitutes implementations of package functions. This is useful for testing code that relies on functions that are slow, have unintended side effects or access resources that may not be available when testing.

`with_mock()` substitutes, runs code locally, and restores in one go.

Usage

```
local_mock(
  ...,
  .parent = parent.frame(),
  .env = get_mock_env(.parent),
  .defer_env = parent.frame()
)

with_mock(..., .parent = parent.frame(), .env = get_mock_env(.parent))
```

Arguments

`...`	`[any]` Named arguments redefine mocked functions. An unnamed argument containing code in braces (`{}`) should be provided to `with_mock()`, it will be evaluated after mocking the functions. Use `:=` to mock functions that start with a dot to avoid potential collision with current or future arguments to `with_mock()` or `local_mock()`. Passing more than one unnamed argument to `with_mock()`, or code that is not inside braces, gives a warning.
`.parent`	`[environment]` the environment in which to evaluate the expressions, defaults to `parent.frame()`. Usually doesn't need to be changed.
`.env`	`[environment]` the environment in which to patch the functions, defaults to `topenv()`. Usually doesn't need to be changed.
`.defer_env`	`[environment]` Attach exit handlers to this environment. Typically, this should be either the current environment or a parent frame (accessed through `parent.frame()`). This argument is passed on as `envir` to `withr::defer()`.

Details

This works by adding a shadow environment as a parent of the environment in which the expressions are evaluated. Everything happens at the R level, but only functions in your own package can be mocked. Otherwise, the implementation is modeled after the original version in the `testthat` package, which is now deprecated.

Value

`local_mock()` returns `NULL`, invisibly.

`with_mock()` returns the result of the last unnamed argument. Visibility is preserved.

References

Suraj Gupta (2012): How R Searches And Finds Stuff

Examples

```
some_func <- function() stop("oops")
some_other_func <- function() some_func()
my_env <- environment()

tester_func <- function() {
  # The default for .env works well most of the time,
  # unfortunately not in examples
  local_mock(some_func = function() 42, .env = my_env)
  some_other_func()
}
try(some_other_func())
tester_func()

tester_func_with <- function() {
  with_mock(
    some_func = function() 42,
    .env = my_env,
    {
      some_other_func()
    }
  )
}
tester_func_with()
```

Index