Package 'mockthat'

December 9, 2020

Title Function Mocking for Unit Testing in R

Version 0.2.4

Description With the deprecation of mocking capabilities shipped with 'testthat' as of 'edition 3' it is left to third-party packages to replace this functionality, which in some test-scenarios is essential in order to run unit tests in limited environments (such as no Internet connection). Mocking in this setting means temporarily substituting a function with a stub that acts in some sense like the original function (for example by serving a HTTP response that has been cached as a file). The only exported function 'with_mock()' is modeled after the eponymous 'testthat' function with the intention of providing a drop-in replacement.

License MIT + file LICENSE

URL https://nbenn.github.io/mockthat/

BugReports https://github.com/nbenn/mockthat/issues

Depends R (>= 3.3.0)

Imports utils

Suggests testthat, curl, jsonlite, datasets, withr

Encoding UTF-8

Language en-US

LazyData true

RoxygenNote 7.1.1

NeedsCompilation no

Author Nicolas Bennett [aut, cre]

Maintainer Nicolas Bennett <nicolas.bennett@stat.math.ethz.ch>

Repository CRAN

Date/Publication 2020-12-09 10:20:11 UTC

R topics documented:

`with_mock` *Mock functions in a package.*

Description

Mocking allows you to temporary replace the implementation of functions within a package, which 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.

Up until recently, such capability was offered via `testthat::with_mock()`, but with release of version 3.0.0 and introduction of edition 3, this was deprecated from testthat, leaving it to third party packages to replace this feature. This mocking implementation is powered by `utils::assignInNamespace()` and therefore, caveats outlined in the corresponding documentation apply here too.

Usage

```
with_mock(..., mock_env = pkg_env(), eval_env = parent.frame())

local_mock(
  ...,
  mock_env = pkg_env(),
  eval_env = parent.frame(),
  local_env = eval_env
)

mock(expr, env = parent.frame())

mock_call(x, call_no = mock_n_called(x))

mock_args(x, arg = NULL, call_no = mock_n_called(x))

mock_n_called(x)
```

Arguments

`...`	Named parameters redefine mocked functions, unnamed parameters will be evaluated after mocking the functions.
`mock_env`	The environment in which to patch the functions, defaults to the top-level environment. A string is interpreted as package name.
`eval_env`	Environment in which expressions passed as `...` are evaluated, defaults to `parent.frame()`.
`local_env`	Passed to `withr::defer()` as `envir` argument
`expr`	Expression to be used as body of the function to be mocked.
`env`	Environment used as ancestor to the mock function environment.
`x`	Object of class `mock_fun` to be queried for call and argument information.
`call_no`	The call number of interest (in case the function was called multiple times).
`arg`	String-valued argument name to be retrieved.

Value

The result of the last unnamed argument passed as `...` (evaluated in the environment passed as `eval_env`) in the case of `local_mock()` and a list of functions or `mock_fun` objects (invisibly) in the case of `local_mock()`.

Examples

```
url <- "https://eu.httpbin.org/get?foo=123"
mok <- function(...) "mocked request"

with_mock(
  `curl::curl_fetch_memory` = mok,
  curl::curl_fetch_memory(url)
)

dl_fun <- function(x) curl::curl_fetch_memory(x)

with_mock(
  `curl::curl_fetch_memory` = mok,
  dl_fun(url)
)

json <- function(...) '["mocked request"]'

with_mock(
  `curl::curl` = json,
  jsonlite::fromJSON(url)
)

with_mock(
  `curl::curl` = '["mocked request"]',
  jsonlite::fromJSON(url)
)

with_mock(
  `curl::curl` = quote({
    x <- "mocked request"
    paste0('["', x, '"]')
  }),
  jsonlite::fromJSON(url)
)

with_mock(
  `curl::curl` = json,
  parse_and_simplify = function(txt, ...) gsub('\\[?\\"\\]?', "", txt),
  jsonlite::fromJSON(url),
  mock_env = "jsonlite"
)

mk <- mock("mocked request")
dl <- function(x) curl::curl(x)
```

```
with_mock(`curl::curl` = mk, dl(url))

mock_call(mk)
mock_args(mk)

mk <- local_mock(`curl::curl` = "mocked request")
dl(url)

mock_args(mk, "url")
```

Index