Package 'wyz.code.testthat'

October 6, 2021

Type Package

Title Wizardry Code Offensive Programming Test Generation

Version 1.1.20

Author Fabien Gelineau <neonira@gmail.com>

Maintainer Fabien Gelineau <neonira@gmail.com>

Description Allows to generate automatically 'testthat' code files from offensive programming test cases. Generated test files are complete and ready to run. Using 'wyz.code.testthat' you will earn a lot of time, reduce the number of errors in test case production, be able to test immediately generated files without any need to view or modify them, and enter a zero time latency between code implementation and industrial testing. As with 'testthat', you may complete provided test cases according to your needs to push testing further, but this need is nearly void when using 'wyz.code.offensiveProgramming'.

Encoding UTF-8

License GPL-3

Depends R (>= 4.0)

Imports methods, data.table (>= 1.11.8), tidyr, wyz.code.offensiveProgramming (>= 1.1.22), R6 (>= 2.4.0)

Suggests testthat, knitr, rmarkdown

RoxygenNote 6.1.1

VignetteBuilder knitr

URL https://neonira.github.io/offensiveProgrammingBook_v1.2.2/

NeedsCompilation no

Repository CRAN

Date/Publication 2021-10-06 06:50:02 UTC

R topics documented:

`generateAllUnitTestsFromObject`
Generate All Unit Tests From Object

Description

Generate `testthat` code form an instrumented offensive programming object.

Usage

```
generateAllUnitTestsFromObject(object_o_1, sourceFile_s_1,
                               sourcePackage_s_1, targetFolder_s_1,
                               overwriteFile_b_1 = TRUE, verbose_b_1 = FALSE)
```

Arguments

`object_o_1`	the instrumented offensive programming object to consider
`sourceFile_s_1`	the source package file related to the offensive programming object
`sourcePackage_s_1`	the package name that holds the offensive programming object
`targetFolder_s_1`	the folder where to write produced `testthat` code files
`overwriteFile_b_1`	A boolean value, either `TRUE` or *FALSE* to switch on or off the overwritting of the output file. Beware, as it is on by default!
`verbose_b_1`	A boolean value, either `TRUE` or `FALSE` to switch on or off the processing verbosity!

Details

Generates automatically, all the `testthat` files with compliant `testthat` code, from the offensive programming object.

Value

A list with two fields named `class` and `filenames`. Former provides the class of the analyzed and mined offensive programming object, the latter gives back all produced `testthat` file names.

Note

This is rather easy to use function that implements a quite complex algorithm based on meta-programmation to generate `testthat` code.

Author(s)

Fabien Gelineau <neonira@gmail.com>

Maintainer: Fabien Gelineau <neonira@gmail.com>

References

See EvaluationMode for more information.

Refer to test_file from package testthat.

Refer to runTestCase from package wyz.code.offensiveProgramming.

Examples

```
library(data.table)
library(wyz.code.offensiveProgramming)
library(wyz.code.testthat)

source_file <- 'code-samples/both-defs/good/full/AdditionTCFIG1.R'
source_package <- 'wyz.code.offensiveProgramming'
source(system.file(source_file, package = source_package))

object <- AdditionTCFIG1()
g <- gautfo(object, source_file, source_package, tempdir())
print(g)
# $class
# [1] "AdditionTCFIG1"
#
# $filenames
#                                                 filename overwritten
# 1:       /tmp/RtmpKLCrXA/test_AdditionTCFIG1-addDouble.R        TRUE
# 2:      /tmp/RtmpKLCrXA/test_AdditionTCFIG1-addInteger.R        TRUE
# 3:    /tmp/RtmpKLCrXA/test_AdditionTCFIG1-divideByZero.R        TRUE
# 4: /tmp/RtmpKLCrXA/test_AdditionTCFIG1-generateWarning.R        TRUE
# 5:   /tmp/RtmpKLCrXA/test_AdditionTCFIG1-generateError.R        TRUE
# 6:  /tmp/RtmpKLCrXA/test_AdditionTCFIG1-addMultiDouble.R        TRUE
# 7: /tmp/RtmpKLCrXA/test_AdditionTCFIG1-addMultiInteger.R        TRUE
```

---

generateUnitTestFile *Generate Testthat Unit Test File*

Description

Generate one testthat unit test file from an instrumented offensive programming object. One test file per function is created. It holds all the related tests, organized by evaluation mode. See EvaluationMode for more information.

Usage

```
generateUnitTestFile(filename_s_1, content_s,
                     overwrite_b_1 = FALSE,
                     verbose_b_1 = TRUE)
```

Arguments

filename_s_1	the name of the unit test file to generate
content_s	the content of the unit test file to generate
overwrite_b_1	A boolean value, either TRUE or FALSE to switch on or off the overwritting of the output file.
verbose_b_1	A boolean value, either TRUE or FALSE to switch on or off the output message from this function.

Value

Returns the file name.

It might be different from the one provided, as testthat requires to follow some file naming conventions that are enforced by this function.

Note

This function is provided for convenience. It may ease your own way to generate testthat code.

To put this package in action, simply use gautfo that offers a much more straightforward way to get results.

Author(s)

Fabien Gelineau <neonira@gmail.com>

Maintainer: Fabien Gelineau <neonira@gmail.com>

References

Refer to test_file from package testthat.

Refer to runTestCase from package wyz.code.offensiveProgramming.

Examples

```
##---- typical example ----
fn <- file.path(tempdir(), 'myFile')
fr <- generateUnitTestFile(fn, c("# a comment", "x <- 2"))
cat(paste(readLines(fr$filename), collapse  = '\n'))
# a comment
# x <- 2
```

opTestthatInformation *Package Functions Information*

Description

list package functions

Usage

```
opTestthatInformation()
```

Value

A data.table with following columns

name	the object name
category	the category of the object describe by function name. Could be CLASS, FUNCTION or DATA.
nature	either INTERNAL or EXPORTED.
stratum	the stratum the object belongs to. Values are CORE, LAYER_1, LAYER_2, LAYER_3.
phasing	main usage phase of the object. Values are DESIGN, BUILD, TEST, RUN, MAINTAIN, EVOLVE and TRANSVERSAL.
intent	main global intent of the object. Values are PARTS_BUILDING, PARTS_ASSEMBLY, QUALITY_CONTROL, FEEDBACK, STATISTICS, CONTENT_GENERATION and UTILITIES.

Author(s)

Fabien Gelineau <neonira@gmail.com>

Maintainer: Fabien Gelineau <neonira@gmail.com>

Examples

```
opTestthatInformation()

opTestthatInformation()[stratum == 'CORE']
```

testthatFactory *A factory to produce* testthat *code*

Description

Produces testthat code from an instrumented offensive programming object.

Usage

```
testthatFactory()
```

Value

Returns a function that manages dispatch depending on targeted offensive programming object test case evaluation mode value.

Note

This function is provided for convenience. It may ease your own way to generate testthat code from an instrumented offensive programming object.

To put this package in action, simply use gautfo that offers a much more straightforward way to get results.

Author(s)

Fabien Gelineau <neonira@gmail.com>

Maintainer: Fabien Gelineau <neonira@gmail.com>

References

See EvaluationMode for more information.

Refer to test_file from package testthat.

Refer to runTestCase from package wyz.code.offensiveProgramming.

Examples

```
##---- typical example ----
rv <- testthatFactory()(c(call('isTRUE', TRUE), call('isFALSE', FALSE)), 'correct')
print(rv)
# [[1]]
# expect_true(isTRUE(TRUE))
#
# [[2]]
# expect_true(isFALSE(FALSE))
```

Index