Package 'tinsel'

October 14, 2022

Title Transform Functions using Decorators

Version 0.0.1

Description Instead of nesting function calls, annotate and transform functions using ``#." comments.

License MIT + file LICENSE

URL https://github.com/nteetor/tinsel

BugReports https://github.com/nteetor/tinsel/issues

Encoding UTF-8

RoxygenNote 5.0.1

Depends R (>= 3.3.1)

Suggests testthat, rstudioapi

NeedsCompilation no

Author Nathan Teetor [aut, cre]

Maintainer Nathan Teetor <nathanteetor@gmail.com>

Repository CRAN

Date/Publication 2016-11-17 08:27:54

R topics documented:

tinsel-package *tinsel: Transform Functions using Decorators*

Description

tinsel provides a decorator syntax for R allowing decoration and transformation of functions using `#.` comments.

Details

To the package in action save the code snippet below to a file, run `source_decoratees` on the file, and then call `tmbg()` or `cats(5)`.

```
# emphasize text
emph <- function(f, style = '**') {
  function(...) {
    if (length(style) == 1) {
      paste(style, f(...), style)
    } else {
      paste(style[1], f(...), style[2])
    }
  }
}

#. emph
tmbg <- function() {
  'tmbg are okay'
}

#. emph(c('<b>', '</b>'))
cats <- function(n) {
  paste(rep('cats', n), collapse = ' ')
}
```

The call you make to `tmbg` is equivalent to `emph(tmbg)`. The second example, `cats(5)`, illustrates passing arguments to the decorator function.

While the above examples are small hopefully you begin to see how decorators may be used to transform or ensure function output without modifying the function's code by hand.

decorators *Get Function Decorators or Original Function*

Description

Get the decorators of a function or the original decoratee function from a decorated function object.

Usage

```
decorators(f)

original(f)
```

Arguments

f	A decorated function.

Examples

```
source_decoratees(tinsel_example('attributes.R'))

# sourced from the 'attributes.R' example file
selector1

# get a list of decorators wrapping a function
decorators(selector1)

# get the original decoratee function of the
# decorated `selector1` function
original(selector1)
```

is.decorated *Decorated Functions*

Description

Returns TRUE if the function f is decorated, otherwise FALSE.

Usage

```
is.decorated(f)
```

Arguments

f	A function.

Value

TRUE or FALSE.

Examples

```
source_decoratees(tinsel_example('timer.R'))

# sourced from the timer.R example file
is.decorated(waldo)
is.decorated(jack)

# it's a function, but not decorated
is.decorated(mean)

# far from the mark
is.decorated(3030)
```

---

print.decorated *Print a Decorated Function*

Description

The print.decorated function naively prints x as a function. In reality, the function printed may be the final of any number of decorators to a decoratee. To get the original function or the decorators wrapping it use original and decorators.

Usage

```
## S3 method for class 'decorated'
print(x, ...)
```

Arguments

x	A decorated function.
...	Additional arguments for next print method.

Examples

```
source_decoratees(tinsel_example('tags.R'))

print(html_paragraph)
print(html_bold)
```

`source_decoratees` *Read and Parse Decoratees from a File*

Description

Given a file, `source_decoratees` reads and parses decorated functions (decoratees) into the calling environment.

Usage

```
source_decoratees(file)
```

Arguments

`file`	A character string specifying a file path.

Details

Malformed decoratees are ignored and a message will alert the user a function has been skipped. However, an error is raised if a decorator is undefined.

If you are working within RStudio the "Source Active File Decoratees" addin effectively allows you to bind `source_decoratees` to a keyboard shorcut. The addin is found under **Tools > Addins**.

Examples

```
# source example files
source_decoratees(tinsel_example('attributes.R'))
source_decoratees(tinsel_example('tags.R'))

# the important thing is to look at the contents
# of the example files, note the use of the special
# "#." comment
writeLines(readLines(tinsel_example('attributes.R')))
writeLines(readLines(tinsel_example('tags.R')))

# the decorator functions are not sourced,
exists('attribute')  # FALSE
exists('html_wrap')  # FALSE

# only decorated functions are sourced
print(selector1)
selector1(mtcars, 'mpg')

# format with bold tags
html_bold('make this bold')

# format with paragraph tags
html_paragraph("I'll make my report as if I told a story...")
```

Index