Package 'typetracer'

June 26, 2023

Title Trace Function Parameter Types

Version 0.2.2

Description The 'R' language includes a set of defined types, but the language itself is ``absurdly dynamic'' (Turcotte & Vitek (2019) <doi:10.1145/3340670.3342426>), and lacks any way to specify which types are expected by any expression. The 'typetracer' package enables code to be traced to extract detailed information on the properties of parameters passed to 'R' functions. 'typetracer' can trace individual functions or entire packages.

License MIT + file LICENSE

URL https://github.com/mpadge/typetracer, https://mpadge.github.io/typetracer/

BugReports https://github.com/mpadge/typetracer/issues

Imports brio, checkmate, methods, rlang, tibble, withr

Suggests knitr, rematch, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

Language en-GB

RoxygenNote 7.2.3

NeedsCompilation yes

Author Mark Padgham [aut, cre] (<https://orcid.org/0000-0003-2172-5265>), Filip Krikava [ctb] (Author of original 'injectr' code on which this package builds; https://github.com/PRL-PRG/injectr), covr authors [cph] (Original authors of sections of code from 'covr' package included here in modified form.)

Maintainer Mark Padgham <mark.padgham@email.com>

Repository CRAN

Date/Publication 2023-06-26 11:10:03 UTC

R topics documented:

`clear_traces` *Clear previous traces*

Description

Traces are by default appended to previous traces. This function can be used to clean those previous ones, to enable subsequent calls to generate new traces that are not appended to previous ones.

Usage

```
clear_traces()
```

Value

(Invisibly) A single logical value indicating whether or not traces were successfully cleared.

Examples

```
f <- function (x, y, z, ...) {
    x * x + y * y
}
inject_tracer (f)
val <- f (1:2, 3:4 + 0., a = "blah")
x <- load_traces ()
print (x)

# Then call 'clear_traces' to remove them:
clear_traces ()
# Trying to load again wil then indicate 'No traces found':
x <- load_traces ()
# Traces should also always be "uninjected":
uninject_tracer (f)
```

`inject_tracer` *Inject parameter tracer into one function*

Description

Inject parameter tracer into one function

Usage

```
inject_tracer(f, trace_lists = FALSE)
```

Arguments

`f`	A function (that is, an object of class "function", and not a character string).
`trace_lists`	If `TRUE`, trace into any nested list parameters (including `data.frame`-type objects), and return type information on each list component. The parameter names for these list-components are then specified in "dollar-notation", for example 'Orange$age'.

Value

Nothing (will error on fail).

Note

The tracer is defined in the internal `typetracer_header()` function. This uses an `options` variable defined on package load for the current `tempdir`, defining a single location where all traced values are dumped. This is done via `options` to allow both multi-threaded function calls and calls via **callr** to be traced.

Examples

```
f <- function (x, y, z, ...) {
    x * x + y * y
}
inject_tracer (f)
val <- f (1:2, 3:4 + 0., a = "blah")
x <- load_traces ()

# Traces should always be "uninjected":
uninject_tracer (f)
# Traces may also be removed:
clear_traces ()
```

load_traces *Load traces of parameter types*

Description

Load traces of parameter types

Usage

```
load_traces(files = FALSE, quiet = FALSE)
```

Arguments

files	If TRUE, return paths to all temporary files holding trace data.
quiet	If FALSE, issue message when no traces found.

Value

A 'data.frame' of traces, including names of functions and parameters, and values of each parameter traced in both unevaluated and evaluated forms.

Examples

```
f <- function (x, y, z, ...) {
    x * x + y * y
}
inject_tracer (f)
val <- f (1:2, 3:4 + 0., a = "blah")
x <- load_traces ()
print (x)

# Traces should always be "uninjected":
uninject_tracer (f)
# Traces may also be removed:
clear_traces ()
```

trace_package *Trace all parameters for all functions in a specified package*

Description

Trace all parameters for all functions in a specified package

Usage

```
trace_package(
  package = NULL,
  pkg_dir = NULL,
  functions = NULL,
  types = c("examples", "tests"),
  trace_lists = FALSE
)
```

Arguments

package	Name of package to be traced (as character value).
pkg_dir	For "types" including "tests", a local directory to the source code of the package. (This is needed because installed versions do not generally include tests.)
functions	Optional character vector of names of functions to trace. Defaults to tracing all functions.
types	The types of code to be run to generate traces: one or both values of "examples" or "tests" (as for `tools::testInstalledPackage`). Note that only tests run via the **testthat** package can be traced.
trace_lists	If `TRUE`, trace into any nested list parameters (including `data.frame`-type objects), and return type information on each list component. The parameter names for these list-components are then specified in "dollar-notation", for example 'Orange$age'.

Value

A `data.frame` of data on every parameter of every function as specified in code provided in package examples.

Examples

```
## Not run:
res <- trace_package ("rematch")
res <- trace_package (pkg_dir = "/<path>/<to>/<local>/<pacakge>")

## End(Not run)
```

---

uninject_tracer *Remove parameter tracer from one function*

Description

This function removes traces previous injected into functions with the inject_tracer function.

Usage

```
uninject_tracer(f)
```

Arguments

f	A function (that is, an object of class "function", and not a character string).

Value

Logical value indicating whether or not tracer was able to be removed ("uninjected").

Examples

```
f <- function (x, y, z, ...) {
    x * x + y * y
}
inject_tracer (f)
val <- f (1:2, 3:4 + 0., a = "blah")
x <- load_traces ()

# Traces should always be "uninjected":
uninject_tracer (f)
# Traces may also be removed:
clear_traces ()
```

Index