Package 'profvis'

May 2, 2023

Title Interactive Visualizations for Profiling R Code

Version 0.3.8

Description Interactive visualizations for profiling R code.

Depends R (>= 3.0)

Imports htmlwidgets (>= 0.3.2), purrr, rlang (>= 0.4.9), stringr, vctrs

License GPL-3 | file LICENSE

Suggests knitr, ggplot2, rmarkdown, testthat (>= 3.0.0), devtools, shiny, htmltools

RoxygenNote 7.2.3

URL https://rstudio.github.io/profvis/

Encoding UTF-8

Config/testthat/edition 3

NeedsCompilation yes

Author Winston Chang [aut, cre],
Javier Luraschi [aut],
Timothy Mastny [aut],
RStudio [cph],
jQuery Foundation [cph] (jQuery library),
jQuery contributors [ctb, cph] (jQuery library; authors listed in inst/www/shared/jquery-AUTHORS.txt),
Mike Bostock [ctb, cph] (D3 library),
D3 contributors [ctb] (D3 library),
Ivan Sagalaev [ctb, cph] (highlight.js library)

Maintainer Winston Chang <winston@rstudio.com>

Repository CRAN

Date/Publication 2023-05-02 18:00:02 UTC

R topics documented:

`parse_rprof` *Parse Rprof output file for use with profvis*

Description

Parse Rprof output file for use with profvis

Usage

```
parse_rprof(path = "Rprof.out", expr_source = NULL)
```

Arguments

`path`	Path to the `Rprof` output file.
`expr_source`	If any source refs in the profiling output have an empty filename, that means they refer to code executed at the R console. This code can be captured and passed (as a string) as the `expr_source` argument.

`pause` *Pause an R process*

Description

This function pauses an R process for some amount of time. It differs from `Sys.sleep` in that time spent in `pause` will show up in profiler data. Another difference is that `pause` uses up 100% of a CPU, whereas `Sys.sleep` does not.

Usage

```
pause(seconds)
```

Arguments

`seconds`	Number of seconds to pause.

Examples

```
# Wait for 0.5 seconds
pause(0.5)
```

`print.profvis` *Print a profvis object*

Description

Print a profvis object

Usage

```
## S3 method for class 'profvis'
print(x, ..., width = NULL, height = NULL, split = NULL, aggregate = NULL)
```

Arguments

`x`	The object to print.
`...`	Further arguments to passed on to other print methods.
`width`	Width of the htmlwidget.
`height`	Height of the htmlwidget
`split`	Direction of split. Either `"v"` (the default) for vertical, or `"h"` for horizontal. This is the orientation of the split bar.
`aggregate`	If 'TRUE', the profiled stacks are aggregated by name. This makes it easier to see the big picture. Set your own global default for this argument with 'options(profvis.aggregate =)'.

`profvis` *Profile an R expression and visualize profiling data*

Description

This function will run an R expression with profiling, and then return an htmlwidget for interactively exploring the profiling data.

Usage

```
profvis(
  expr = NULL,
  interval = 0.01,
  prof_output = NULL,
  prof_input = NULL,
  width = NULL,
  height = NULL,
  split = c("h", "v"),
  torture = 0,
  simplify = TRUE,
  rerun = FALSE
)
```

Arguments

`expr`	Expression to profile. Not compatible with `prof_input`. The expression is repeatedly evaluated until ‘Rprof()‘ produces an output. It can _be_ a quosure injected with [rlang::inject()] but it cannot _contain_ injected quosures.
`interval`	Interval for profiling samples, in seconds. Values less than 0.005 (5 ms) will probably not result in accurate timings
`prof_output`	Name of an Rprof output file or directory in which to save profiling data. If `NULL` (the default), a temporary file will be used and automatically removed when the function exits. For a directory, a random filename is used.
`prof_input`	The path to an `Rprof` data file. Not compatible with `expr` or `prof_output`.
`width`	Width of the htmlwidget.
`height`	Height of the htmlwidget
`split`	Direction of split. Either `"v"` (the default) for vertical, or `"h"` for horizontal. This is the orientation of the split bar.
`torture`	Triggers garbage collection after every `torture` memory allocation call. Note that memory allocation is only approximate due to the nature of the sampling profiler and garbage collection: when garbage collection triggers, memory allocations will be attributed to different lines of code. Using `torture = steps` helps prevent this, by making R trigger garbage collection after every `torture` memory allocation step.
`simplify`	Whether to simplify the profiles by removing intervening frames caused by lazy evaluation. This only has an effect on R 4.0. See the `filter.callframes` argument of `Rprof()`.
`rerun`	If ‘TRUE‘, ‘Rprof()‘ is run again with ‘expr‘ until a profile is actually produced. This is useful for the cases where ‘expr‘ returns too quickly, before R had time to sample a profile. Can also be a string containing a regexp to match profiles. In this case, ‘profvis()‘ reruns ‘expr‘ until the regexp matches the modal value of the profile stacks.

Details

An alternate way to use `profvis` is to separately capture the profiling data to a file using `Rprof()`, and then pass the path to the corresponding data file as the `prof_input` argument to `profvis()`.

See Also

`print.profvis` for printing options.

`Rprof` for more information about how the profiling data is collected.

Examples

```
# Only run these examples in interactive R sessions
if (interactive()) {

# Profile some code
profvis({
  dat <- data.frame(
    x = rnorm(5e4),
    y = rnorm(5e4)
  )

  plot(x ~ y, data = dat)
  m <- lm(x ~ y, data = dat)
  abline(m, col = "red")
})


# Save a profile to an HTML file
p <- profvis({
  dat <- data.frame(
    x = rnorm(5e4),
    y = rnorm(5e4)
  )

  plot(x ~ y, data = dat)
  m <- lm(x ~ y, data = dat)
  abline(m, col = "red")
})
htmlwidgets::saveWidget(p, "profile.html")

# Can open in browser from R
browseURL("profile.html")

}
```

---

profvisOutput *Widget output function for use in Shiny*

Description

Widget output function for use in Shiny

Usage

```
profvisOutput(outputId, width = "100%", height = "600px")
```

Arguments

`outputId`	Output variable for profile visualization.
`width`	Width of the htmlwidget.
`height`	Height of the htmlwidget

`profvis_ui` *Profvis UI for Shiny Apps*

Description

Use this Shiny module to inject Profvis controls into your Shiny app. The Profvis Shiny module injects UI that can be used to start and stop profiling, and either view the results in the Profvis UI or download the raw .Rprof data. It is highly recommended that this be used for testing and debugging only, and not included in production apps!

Usage

```
profvis_ui(id)

profvis_server(input, output, session, dir = ".")
```

Arguments

`id`	Output id from `profvis_server`.
`input, output, session`	Arguments provided by `callModule`.
`dir`	Output directory to save Rprof files.

Details

The usual way to use Profvis with Shiny is to simply call 'profvis(shiny::runApp())', but this may not always be possible or desirable: first, if you only want to profile a particular interaction in the Shiny app and not capture all the calculations involved in starting up the app and getting it into the correct state; and second, if you're trying to profile an application that's been deployed to a server.

For more details on how to invoke Shiny modules, see [this article](https://shiny.rstudio.com/articles/modules.html).

Examples

```
# In order to avoid "Hit <Return> to see next plot" prompts,
# run this example with `example(profvis_ui, ask=FALSE)`

if(interactive()) {
  library(shiny)
  library(ggplot2)
  shinyApp(
    fluidPage(
      plotOutput("plot"),
      actionButton("new", "New plot"),
      profvis_ui("profiler")
    ),
    function(input, output, session) {
      callModule(profvis_server, "profiler")

      output$plot <- renderPlot({
        input$new
        ggplot(diamonds, aes(carat, price)) + geom_point()
      })
    }
  )
}
```

---

renderProfvis *Widget render function for use in Shiny*

Description

Widget render function for use in Shiny

Usage

```
renderProfvis(expr, env = parent.frame(), quoted = FALSE)
```

Arguments

expr	An expression that returns a profvis object.
env	The environment in which to evaluate expr.
quoted	Is expr a quoted expression (with quote())?

Index