Package 'profvis'

September 20, 2024

Title Interactive Visualizations for Profiling R Code

Version 0.4.0

Description Interactive visualizations for profiling R code.

License MIT + file LICENSE

URL <https://profvis.r-lib.org>, <https://github.com/r-lib/profvis>

BugReports <https://github.com/r-lib/profvis/issues>

Depends R (>= 4.0)

Imports htmlwidgets (>= 0.3.2), rlang (>= 1.1.0), vctrs

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

VignetteBuilder knitr

Config/Needs/website tidyverse/tidytemplate, rmarkdown

Config/testthat/edition 3

Encoding UTF-8

RoxygenNote 7.3.2

NeedsCompilation yes

Author Hadley Wickham [aut, cre],
Winston Chang [aut],
Javier Luraschi [aut],
Timothy Mastny [aut],
Posit Software, PBC [cph, fnd],
jQuery Foundation [cph] (jQuery library),
jQuery contributors [ctb, cph] (jQuery library; authors listed in
inst/htmlwidgets/lib/jquery/AUTHORS.txt),
Mike Bostock [ctb, cph] (D3 library),
D3 contributors [ctb] (D3 library),
Ivan Sagalaev [ctb, cph] (highlight.js library)

Maintainer Hadley Wickham <hadley@posit.co>

Repository CRAN

Date/Publication 2024-09-20 11:50:10 UTC

Contents

---

  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\ 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	Orientation of the split bar: either "h" (the default) for horizontal or "v" for vertical.
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,
  timing = NULL,
  width = NULL,
  height = NULL,
  split = c("h", "v"),
  torture = 0,
  simplify = TRUE,
  rerun = FALSE
)
```

Arguments

expr	Expression to profile. The expression will be turned into the body of a zero-argument anonymous function which is then called repeatedly as needed. This means that if you create variables inside of expr they will not be available outside of it.
	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.
	Not compatible with prof_input.

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.
timing	The type of timing to use. Either "elapsed" (the default) for wall clock time, or "cpu" for CPU time. Wall clock time includes time spent waiting for other processes (e.g. waiting for a web page to download) so is generally more useful.
	If NULL, the default, will use elapsed time where possible, i.e. on Windows or on R 4.4.0 or greater.
width	Width of the htmlwidget.
height	Height of the htmlwidget
split	Orientation of the split bar: either "h" (the default) for horizontal or "v" for vertical.
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. Equivalent to the filter.callframes argument to [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 and renders functions for use in Shiny*

Description

Widget output and renders functions for use in Shiny

Usage

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

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

Arguments

outputId	Output variable for profile visualization.
width	Width of the htmlwidget.
height	Height of the htmlwidget
expr	An expression that returns a profvis object.

| env | The environment in which to evaluate `expr`. |
| quoted | Is expr a quoted expression (with [quote()](#))? |

---

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 [shiny::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](#).

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)
  shinyApp(
    fluidPage(
      plotOutput("plot"),
      actionButton("new", "New plot"),
      profvis_ui("profiler")
```

```
        ),
        function(input, output, session) {
          callModule(profvis_server, "profiler")

          output$plot <- renderPlot({
            input$new
            boxplot(mpg ~ cyl, data = mtcars)
          })
        }
      )
    }
```

Index