Package 'reactlog'

September 12, 2020

Title Reactivity Visualizer for 'shiny'

Version 1.1.0

Description Building interactive web applications with R is incredibly easy with 'shiny'. Behind the scenes, 'shiny' builds a reactive graph that can quickly become intertwined and difficult to debug. 'reactlog' (Schloerke 2019) <doi:10.5281/zenodo.2591517> provides a visual insight into that black box of 'shiny' reactivity by constructing a directed dependency graph of the application's reactive state at any time point in a reactive recording.

Depends R (>= 3.0.2)

Imports jsonlite (>= 0.9.16),

Suggests shiny (>= 1.5.0), knitr, rmarkdown, htmltools, testthat

License GPL-3 | file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

URL https://rstudio.github.io/reactlog/,
https://github.com/rstudio/reactlog,
https://community.rstudio.com/tags/reactlog

BugReports https://github.com/rstudio/reactlog/issues

VignetteBuilder knitr

Language en-US

NeedsCompilation no

Author Barret Schloerke [aut, cre] (<https://orcid.org/0000-0001-9986-114X>),
Joe Cheng [ctb],
RStudio [cph, fnd]

Maintainer Barret Schloerke <barret@rstudio.com>

Repository CRAN

Date/Publication 2020-09-12 13:20:03 UTC

R topics documented:

`reactlog_module_ui` *Reactlog Shiny Module*

Description

Displays an iframe of the reactlog in the given application.

Usage

```
reactlog_module_ui(include_refresh = TRUE, id = "reactlog_module")

reactlog_module_server(
  id = "reactlog_module",
  width = "100%",
  height = 600,
  ...
)
```

Arguments

`include_refresh`	should the iframe refresh button be included?
`id`	**shiny** module id to use
`width, height`	HTML attributes to be applied to the reactlog iframe
`...`	parameters passed to `shiny::actionButton()`

Details

State will not be preserved between refreshes. To open the reactlog at a particular step, be sure to mark your time points with `Cmd+Shift+F3` (Windows: `Ctrl+Shift+F3`)

See Also

`shiny::moduleServer()`

Examples

```
if (!require("shiny")) {
  message("`shiny` required to run example")
  return()
}

library(shiny)
# Enable reactlog
reactlog_enable()

# Define UI for app that draws a histogram ----
ui <- fluidPage(
  tags$h1("Pythagorean theorem"),
  numericInput("a", "A", 3),
  numericInput("b", "B", 4),
  "C:", verbatimTextOutput("c"),
### start ui module
  reactlog_module_ui()
### end ui module
)

server <- function(input, output, session) {
  a2 <- reactive({a <- input$a; req(a); a * a}, label = "a^2")
  b2 <- reactive({b <- input$b; req(b); b * b}, label = "b^2")
  c2 <- reactive({a2() + b2()}, label = "c^2")
  c_val <- reactive({sqrt(c2())}, label = "c")

  output$c <- renderText({
    c_val()
  })

### start server module
  reactlog_module_server()
### end server module

}

if (interactive()) {
  shinyApp(ui = ui, server = server)
}
```

`reactlog_render` *Reactive Log Visualizer*

Description

Provides an interactive browser-based tool for visualizing reactive dependencies and execution in your application.

Usage

```
reactlog_render(log, session_token = NULL, time = TRUE)

reactlog_show(log, time = TRUE, ...)
```

Arguments

log	Log produced by shiny to be processed
session_token	token to be used to subset which session is displayed. Defaults to all sessions.
time	A boolean that specifies whether or not to display the time that each reactive takes to calculate a result.
...	Future parameter expansion. Currently ignored

Details

To use the reactive log visualizer, start with a fresh R session and run the command `reactlog_enable()`; then launch your application in the usual way (e.g. using `shiny::runApp()`). At any time you can hit

`Ctrl+F3`

(or for Mac users,

`Cmd+F3`

) in your web browser to launch the reactive log visualization.

The reactive log visualization only includes reactive activity up until the time the report was loaded. If you want to see more recent activity, refresh the browser.

Note that Shiny does not distinguish between reactive dependencies that "belong" to one Shiny user session versus another, so the visualization will include all reactive activity that has taken place in the process, not just for a particular application or session.

As an alternative to pressing

`Ctrl/Cmd+F3`

–for example, if you are using reactives outside of the context of a Shiny application–you can run the `shiny::reactlogShow()` function, which will generate the reactive log visualization as a static HTML file and launch it in your default browser. In this case, refreshing your browser will not load new activity into the report; you will need to call `shiny::reactlogShow()` explicitly.

For security and performance reasons, do not enable the reactlog in production environments. When the option is enabled, it's possible for any user of your app to see at least some of the source code of your reactive expressions and observers.

See Also

`shiny::reactlogShow()` and `reactlog_enable()`

Examples

```
## Not run:
library(shiny)
library(reactlog)

# tell shiny to log reactivity
reactlog_enable()

# run a shiny app
app <- system.file("examples/01_hello", package = "shiny")
runApp(app)

# once app has closed, display reactlog
shiny::reactlogShow()

## End(Not run)
```

`reactlog_write` *Write reactlog*

Description

Write the reactlog to a file. If a session token is provided, all reactive interactions will be subsetted to either the global session or the session provided.

Usage

```
reactlog_write(log, file = stdout(), session_token = NULL)
```

Arguments

`log`	produced by shiny to be written
`file`	location of output file. If a `NULL` file is given, the json representation is return.
`session_token`	Session token identifier to be used when subsetting the complete reactlog

---

`setReactLog` *Enable or disable the reactlog*

Description

Before the reactlog can be visualized, it needs to be enabled. For security and performance reasons, you should not enable the reactlog in a shiny app in production.

Usage

```
reactlog_enable()

reactlog_disable(warn = TRUE)
```

Arguments

warn	Should a warning message be shown?

See Also

reactlog_show()

Index