Package: reactlog (via r-universe)

November 3, 2026

Title Reactivity Visualizer for 'shiny'

Version 1.1.1.9000

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.

License MIT + file LICENSE

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

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

Depends R (>= 3.0.2)

Imports jsonlite (>= 0.9.16), rmarkdown

Suggests fontawesome (>= 0.3.0), htmltools, knitr, shiny (>= 1.5.0), testthat

VignetteBuilder knitr

Config/roxygen2/version 8.1.1

Encoding UTF-8

Language en-US

Roxygen list(markdown = TRUE)

Config/pak/sysreqs cmake make libuv1-dev

Repository https://posit-dev-shinycoreci.r-universe.dev

Date/Publication 2026-10-22 20:23:22 UTC

RemoteUrl https://github.com/rstudio/reactlog

RemoteRef HEAD

RemoteSha 213dda2dc4a7fd9b4497a91fc889fc65d388e0dc

Contents

reactlog_module_ui 2
reactlog_read 3
reactlog_render 4
reactlog_write 5
setReactLog 6

Index **7**

`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 (requireNamespace("shiny", quietly = TRUE)) {
  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_read` *Read reactlog*

Description

Read a reactlog written by `reactlog_write()`, so it can be passed to `reactlog_show()` or `reactlog_render()`.

Usage

```
reactlog_read(file)
```

Arguments

file	JSON file written by reactlog_write().

Value

The reactlog: a list of log entries with a version attribute.

---

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 = shiny::reactlog(), time = TRUE, ...)
```

Arguments

log	Log produced by shiny to be processed. Defaults to shiny::reactlog(), the log recorded in the current R session.
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.
...	Arguments passed on to reactlog_render(), such as session_token.

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
reactlog_show()

## 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

See Also

reactlog_read() to read the file back in.

Examples

```
## Not run:
reactlog_write(shiny::reactlog(), "reactlog.json")
log <- reactlog_read("reactlog.json")
reactlog_show(log)

## End(Not run)
```

---

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

reactlog_disable *(*setReactLog*)*, 6
reactlog_enable *(*setReactLog*)*, 6
reactlog_enable(), *5*
reactlog_module_server
*(*reactlog_module_ui*)*, 2
reactlog_module_ui, 2
reactlog_read, 3
reactlog_read(), *6*
reactlog_render, 4
reactlog_render(), *3*
reactlog_show *(*reactlog_render*)*, 4
reactlog_show(), *3*, *6*
reactlog_write, 5
reactlog_write(), *3*, *4*

setReactLog, 6
shiny::actionButton(), *2*
shiny::moduleServer(), *2*
shiny::reactlog(), *4*
shiny::reactlogShow(), *5*
shiny::runApp(), *4*