Package: gargoyle (via r-universe)

June 30, 2024

Title An Event-Based Mechanism for 'Shiny'

Version 0.0.1

Description An event-Based framework for building 'Shiny' apps.
Instead of relying on standard 'Shiny' reactive objects, this
package allow to relying on a lighter set of triggers, so that
reactive contexts can be invalidated with more control.

License MIT + file LICENSE

URL https://github.com/ColinFay/gargoyle

BugReports https://github.com/ColinFay/gargoyle/issues

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

Imports shiny, attempt

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Repository https://colinfay.r-universe.dev

RemoteUrl https://github.com/colinfay/gargoyle

RemoteRef HEAD

RemoteSha d15eaf13398cbb741e2f66f1f30bce52b9c005a0

Contents

`get_gargoyle_logs` *Handle logs*

Description

Get / Clear the logs of all the time the ‘trigger()‘ functions are launched.

Usage

```
get_gargoyle_logs()

clear_gargoyle_logs()
```

Value

A data.frame of the logs.

Examples

```
if (interactive()){
  get_gargoyle_logs()
  clear_gargoyle_logs()
}
```

`init` *Initiate, trigger, event*

Description

Initiate, trigger, event

Usage

```
init(..., session = getDefaultReactiveDomain())

trigger(..., session = getDefaultReactiveDomain())

watch(name, session = getDefaultReactiveDomain())
```

Arguments

`session`	The shiny session object
`name, ...`	The name(s) of the events

Value

The ‘session‘ object invisibly. These functions are mainly used for side-effects.

Examples

```
if (interactive()){
  library(shiny)
  library(gargoyle)
  options("gargoyle.talkative" = TRUE)
  ui <- function(request){
    tagList(
      h4('Go'),
      actionButton("y", "y"),
      h4('Output of z$v'),
      tableOutput("evt")
    )
  }

  server <- function(input, output, session){

    # Initiating the flags
    init("airquality", "iris", "renderiris")

    # Creating a new env to store values, instead of
    # a reactive structure
    z <- new.env()

    observeEvent( input$y , {
      z$v <- mtcars
      # Triggering the flag
      trigger("airquality")
    })

    on("airquality", {
      # Triggering the flag
      z$v <- airquality
      trigger("iris")
    })

    on("iris", {
      # Triggering the flag
      z$v <- iris
      trigger("renderiris")
    })

    output$evt <- renderTable({
      # This part will only render when the renderiris
      # flag is triggered
      watch("renderiris")
      head(z$v)
    })

  }

  shinyApp(ui, server)
}
```

`on` *React on an event*

Description

React on an event

Usage

```
on(name, expr, session = getDefaultReactiveDomain())
```

Arguments

`name`	the name of the event to react to
`expr`	the expression to run when the event is triggered.
`session`	The shiny session object

Value

An observeEvent object. This object will rarely be used, ‘on‘ is mainly called for side-effects.

Index