

# Package ‘shiny.info’

March 23, 2020

**Type** Package

**Version** 0.2.0

**Title** 'shiny' Info

**Description**

Displays simple diagnostic information of the 'shiny' project in the user interface of the app.

**License** MIT + file LICENSE

**BugReports** <https://github.com/Appsilon/shiny.info/issues>

**Encoding** UTF-8

**RoxygenNote** 6.1.1

**Imports** git2r (>= 0.22.1), glue, shiny

**Suggests** testthat, lintr, covr

**NeedsCompilation** no

**Author** Paweł Przytuła [aut],  
Dominik Krzemiński [aut],  
Jakub Nowicki [aut, cre],  
Appsilon Sp. z o.o. [cph]

**Maintainer** Jakub Nowicki <kuba@appsilon.com>

**Repository** CRAN

**Date/Publication** 2020-03-23 10:30:02 UTC

## R topics documented:

.loader_dots . . . . .	2
.loader_spinner . . . . .	2
busy . . . . .	3
display . . . . .	3
get_args . . . . .	4
git_info . . . . .	4
infoPanel . . . . .	5
info_panel . . . . .	5
info_value . . . . .	6

inspect_btn_server . . . . .	6
inspect_btn_ui . . . . .	7
powered_by . . . . .	7
render_info_value . . . . .	8
render_session_info . . . . .	9
toggle_info . . . . .	10
version . . . . .	11

<b>Index</b>	<b>12</b>
--------------	-----------

---

<code>.loader_dots</code>	<i>CSS/HTML code for a dots loader</i>
---------------------------	--

---

### **Description**

CSS/HTML code for a dots loader

### **Usage**

`.loader_dots`

### **Format**

An object of class `list` of length 2.

---

<code>.loader_spinner</code>	<i>CSS/HTML code for a spinner loader</i>
------------------------------	---

---

### **Description**

CSS/HTML code for a spinner loader

### **Usage**

`.loader_spinner`

### **Format**

An object of class `list` of length 2.

---

busy	<i>Busy or not</i>
------	--------------------

---

### Description

Displays a spinner indicating if there are any calculations running on the server side.

### Usage

```
busy(loader = "dots", position = "top right")
```

### Arguments

loader	character or list. If character given, then it selects from one of a few available loaders (look Details for more). By using list you can create your custom loader. List needs to have structure <code>loader &lt;-list(css&lt;-"CSS CODE",html &lt;-div("CODE TO DISPLAY"))</code> .
position	character with position of the parameter. Default "top right".

### Details

Currently available loaders: "dots", "spinner".

Solution inspired by: <https://colinfay.me/watch-r-shiny/>.

### Value

shiny tag List with js script, style of spinner and display div

---

display	<i>Display message on the top-right corner</i>
---------	--

---

### Description

Display message on the top-right corner

### Usage

```
display(message, position = "top right", type = "message")
```

### Arguments

message	character with any message you want
position	character with position of the parameter. Default "top right".
type	character with display type to specify the id. Default to "message"

**Value**

div which wraps your message to display it in the position corner of your shiny app.

---

get_args	<i>Auxiliary function that returns list of arguments for parent calling function</i>
----------	--

---

**Description**

Auxiliary function that returns list of arguments for parent calling function

**Usage**

```
get_args(fun, depth = -9)
```

**Arguments**

fun	string; for what function to look for
depth	negative numeric; how layers above to check the arguments

**Value**

list of function arguments

---

git_info	<i>Git information</i>
----------	------------------------

---

**Description**

Displays git information from the repository of the current working space.

**Usage**

```
git_info(position = "top right")
```

**Arguments**

position	character with position of the parameter. Default "top right".
----------	--

---

infoPanel	<i>Wrapper for info_panel function</i>
-----------	--

---

**Description**

Wrapper for info\_panel function

**Usage**

```
infoPanel(..., position = "top right")
```

**Arguments**

...	calls with info elements
position	character with position of the parameter. Default "top right".

**Value**

div which wraps your all info boxes to display it in the position corner of your shiny app.

---

info_panel	<i>Creates a panel for all info boxes so they do not overlap</i>
------------	--

---

**Description**

Creates a panel for all info boxes so they do not overlap

**Usage**

```
info_panel(..., position = "top right")
```

**Arguments**

...	calls with info elements
position	character with position of the parameter. Default "top right".

**Value**

div which wraps your all info boxes to display it in the position corner of your shiny app.

info\_value                      *UI output for info value*

---

**Description**

UI output for info value

**Usage**

```
info_value(id, position = "top right")
```

**Arguments**

id                      output id to render  
position                character with position of the parameter. Default "top right".

**Value**

div which wraps rendered value to display it in the position corner of your shiny app.

---

inspect\_btn\_server            *Inspect button server*

---

**Description**

Helps to easlily stop app at any moment for debugging or checking state.

**Usage**

```
inspect_btn_server(input)
```

**Arguments**

input                      Shiny server input.

**Value**

observeEvent for Inspect button

---

inspect_btn_ui	<i>Inspect button UI</i>
----------------	--------------------------

---

**Description**

Helps to easlily stop app at any moment for debugging or checking state.

**Usage**

```
inspect_btn_ui(label = "Inspect", position = "top right")
```

**Arguments**

label	label for debug button. Default "Inspect"
position	character with position of the parameter. Default "top right".

**Value**

div with "shinyinfo\_inspect\_btn".

---

powered_by	<i>Powered by</i>
------------	-------------------

---

**Description**

Displays information about authors of the shiny app.

**Usage**

```
powered_by(company_name, link = "#", position = "top right")
```

**Arguments**

company_name	character with the creator of the app
link	link to the creator's website
position	character with position of the parameter. Default "top right".

**Value**

div with "powered by".

---

render\_info\_value      *Server render function for info value*

---

### Description

Server render function for info value

### Usage

```
render_info_value(expr, env = parent.frame(), quoted = FALSE,
  sep = " ", add_name = TRUE)
```

### Arguments

expr	value to render
env	The environment in which to evaluate expr. Default parent.frame()
quoted	Is expr a quoted expression (with quote())? This is useful if you want to save an expression in a variable. Default FALSE
sep	A separator passed to cat to be appended after each element.
add_name	Should expression name be added. Default TRUE

### Details

If you want to use it with `toggle_info()`, you have to add `outputOptions(output, [info value id], suspendWhenHidden = FALSE)` to force rendering when the value is hidden.

### Value

Shiny render function to be saved as an element of output.

### Examples

```
if(interactive()){
  library(shiny)
  library(shiny.info)

  ui <- fluidPage(
    info_value("value_to_display")
  )

  server <- function(input, output, session) {
    test_reactive <- reactiveVal("some value")
    output$value_to_display <- render_info_value(expr = test_reactive())
    # next line is required to work with toggle_info()
    outputOptions(output, "value_to_display", suspendWhenHidden = FALSE)
  }
}
```



---

render_session_info	<i>Server render function for rendering informations about user session (app URL, username and pixel ratio)</i>
---------------------	---

---

## Description

Server render function for rendering informations about user session (app URL, username and pixel ratio)

## Usage

```
render_session_info()
```

## Details

Use it with `info_value` UI function. If you want to use it with `toggle_info()`, you have to add `outputOptions(output, [session info id], suspendWhenHidden = FALSE)` to force rendering when the value is hidden.

## Value

Shiny render function to be saved as an element of output.

## Examples

```
if(interactive()){
  library(shiny)
  library(shiny.info)

  ui <- fluidPage(
    info_value("session_info")
  )

  server <- function(input, output, session) {
    output$session_info <- render_session_info()
    # next line is required to work with toggle_info()
    outputOptions(output, "session_info", suspendWhenHidden = FALSE)
  }
}
```

---

`toggle_info`*Toggle showing info with keyboard shortcut*

---

**Description**

Toggle showing info with keyboard shortcut

**Usage**

```
toggle_info(shortcut = "Ctrl+Shift+K", hidden_on_start = TRUE)
```

**Arguments**

`shortcut` keys that trigger showing info. Shortcut can include special keys: Ctrl, Alt, Shift. Keys should be separated with '+' sign. Default Ctrl+Shift+K

`hidden_on_start` should info panels be hidden on start of the application? Default TRUE.

**Details**

`toggle_info()` should be added in the header of the application in ui.R, since it adds a script with toggle functionality. If you want to use it with `info_value`, you have to add `outputOptions(output, [info value id], suspendWhenHidden = FALSE)` to force rendering when the value is hidden.

**Value**

JS script that adds toggle functionality.

**Examples**

```
if (interactive()){
  library(shiny)
  library(shiny.info)

  ui <- fluidPage(
    toggle_info(),
    shiny.info::display("test message"),
    shiny.info::info_value("test_input_value", "bottom right"),
    textInput(inputId = "test_input", label = NULL)
  )

  server <- function(input, output, session) {
    output$test_input_value <- shiny.info::render_info_value(input$test_input)
    outputOptions(output, "test_input_value", suspendWhenHidden = FALSE)
  }

  shinyApp(ui = ui, server = server)
}
```

---

version	<i>Display version of the app</i>
---------	-----------------------------------

---

**Description**

Displays the version of the app by default using VERSION global variable.

**Usage**

```
version(ver = NULL, position = "top right")
```

**Arguments**

ver	(default NULL) custom version number
position	character with position of the parameter. Default "top right".

**Examples**

```
version() # if VERSION global variable exists  
version("1.2.1") # with custom version number
```

# Index

## \*Topic **datasets**

.loader\_dots, [2](#)

.loader\_spinner, [2](#)

.loader\_dots, [2](#)

.loader\_spinner, [2](#)

busy, [3](#)

display, [3](#)

get\_args, [4](#)

git\_info, [4](#)

info\_panel, [5](#)

info\_value, [6](#)

infoPanel, [5](#)

inspect\_btn\_server, [6](#)

inspect\_btn\_ui, [7](#)

powered\_by, [7](#)

render\_info\_value, [8](#)

render\_session\_info, [9](#)

toggle\_info, [10](#)

version, [11](#)