Package 'shinyToastify'

October 14, 2022

Type Package

Title Pretty Notifications for 'Shiny'

Version 2.0.0

Description This is a wrapper of the 'React' library 'React-Toastify'. It allows to show some notifications (toasts) in 'Shiny' applications. There are options for the style, the position, the transition effect, and more.

License GPL-3

Encoding UTF-8

URL https://github.com/stla/shinyToastify

BugReports https://github.com/stla/shinyToastify/issues

Imports htmltools, reactR, shiny, utils, fontawesome

RoxygenNote 7.1.1

NeedsCompilation no

Author Stéphane Laurent [aut, cre],
Fadi Khadra [cph] ('React-Toastify' library
(https://fkhadra.github.io/react-toastify/introduction))

Maintainer Stéphane Laurent <laurent_step@outlook.fr>

Repository CRAN

Date/Publication 2021-07-31 11:00:01 UTC

R topics documented:

shinyToastifyExample *Run a Shiny Toastify example*

Description

A function to run examples of Shiny apps using the `shinyToastify` package.

Usage

```
shinyToastifyExample(example, display.mode = "showcase", ...)
```

Arguments

`example`	example name
`display.mode`	the display mode to use when running the example; see `runApp`
`...`	arguments passed to `runApp`

Value

No return value, just launches a Shiny app.

Examples

```
if(interactive()){
  shinyToastifyExample("bodyClassName")
}
if(interactive()){
  shinyToastifyExample("className")
}
if(interactive()){
  shinyToastifyExample("progressClassName")
}
if(interactive()){
  shinyToastifyExample("TypeAndTransition")
}
```

shinyToastifyExamples *Shiny Toastify examples*

Description

List of examples.

Usage

```
shinyToastifyExamples()
```

Value

No return value, just prints a message listing the example names.

Examples

```
shinyToastifyExamples()
if(interactive()){
  shinyToastifyExample("TypeAndTransition")
}
```

---

showToast *Show a toast*

Description

Show a toast in a Shiny application.

Usage

```
showToast(
  session,
  input,
  id = NULL,
  text,
  type = "default",
  position = "top-right",
  transition = "slide",
  autoClose = 5000,
  hideProgressBar = FALSE,
  newestOnTop = FALSE,
  closeOnClick = TRUE,
  rtl = FALSE,
  pauseOnFocusLoss = TRUE,
  draggable = TRUE,
  draggableDirection = "x",
  draggablePercent = 80,
  pauseOnHover = TRUE,
  className = NULL,
  toastClassName = NULL,
  bodyClassName = NULL,
  progressClassName = NULL,
  style = NULL,
  Rcallback = function() {     NULL },
  JScallback = NULL
)
```

Arguments

session	the Shiny session object
input	the Shiny input object
id	an id for the toast or NULL for automatic id; see details for the use of this id
text	the text displayed in the toast; this can be a character string, an html element created with the HTML function, or a shiny.tag object such as tags$span(style = "color:lime;", "Message")
type	toast type, one of "info", "success", "warning", "error", "default" or "dark"
position	toast position, one of "top-left", "top-right", "top-center", "bottom-left", "bottom-right" or "bottom-center"
transition	the transition effect, one of "slide", "zoom", "flip" or "bounce"
autoClose	either a number, the time in ms to close the toast, or FALSE to close the toast manually
hideProgressBar	Boolean, whether to hide the progress bar
newestOnTop	Boolean, whether to display newest toast on top
closeOnClick	Boolean, whether to dismiss the toast on click
rtl	Boolean, right to left
pauseOnFocusLoss	Boolean, whether to pause the toast on focus loss
draggable	Boolean, ability to drag the toast to remove it
draggableDirection	"x" or "y"
draggablePercent	the percentage of the width of the toast needed to remove it by dragging
pauseOnHover	Boolean, whether to pause the toast on hover
className	name of a CSS class applied to the container
toastClassName	name of a CSS class applied on the toast wrapper
bodyClassName	name of a CSS class applied on the toast body
progressClassName	name of a CSS class applied on the progress bar
style	inline style applied to the container, e.g. list(boxShadow = "rgba(0, 0, 0, 0.56) 0px 22px 30px 4px")
Rcallback	a R function without arguments to be executed whenever the toast is closed; alternatively, use the id argument (see details)
JScallback	some JavaScript code given as a string to be executed whenever the toast is closed, e.g. 'alert("The toast is closed")'

Details

Usage of the `id` **argument.** If you provide a string to the `id` argument, say `"mytoast"`, the application will send the event `input[["mytoast_closed"]]` to the server whenever the toast closes; therefore you can listen to this event with an observer to perform an action whenever the toast closes.

Value

No return value, called for side effect.

Examples

```
library(shiny)
library(shinyToastify)

ui <- fluidPage(
  useShinyToastify(),
  br(),
  actionButton("btn", "Show toast", class = "btn-primary btn-lg")
)

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

  toastTransitions <- c(
    "Zoom", "Bounce", "Flip", "Slide"
  )

  observeEvent(input[["btn"]], {

    toastTransition <- toastTransitions[1L + (input[["btn"]] %% 4L)]

    html <- HTML(
      '<span style="font-size: 30px; font-family: cursive;">',
      paste0(toastTransition, " transition"),
      '</span>',
    )

    showToast(
      session,
      input,
      text = html,
      type = "success",
      transition = tolower(toastTransition),
      autoClose = 3000,
      style = list(
        border = "4px solid crimson",
        boxShadow = "rgba(0, 0, 0, 0.56) 0px 22px 30px 4px"
      )
    )

  })
```

```
}

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

`toastUpdate` *Update a toast*

Description

Update a toast in a Shiny application. Run `shinyToastifyExample("toastUpdate")` for an example.

Usage

```
toastUpdate(
  session,
  toastId,
  text,
  type = "default",
  position = "top-right",
  transition = "slide",
  autoClose = 5000,
  hideProgressBar = FALSE,
  closeOnClick = TRUE,
  rtl = FALSE,
  pauseOnFocusLoss = TRUE,
  draggable = TRUE,
  draggableDirection = "x",
  draggablePercent = 80,
  pauseOnHover = TRUE,
  className = NULL,
  toastClassName = NULL,
  bodyClassName = NULL,
  progressClassName = NULL,
  style = NULL,
  JScallback = NULL
)
```

Arguments

`session`	the Shiny `session` object
`toastId`	the id of the toast to be updated (`id` argument)
`text`	the text displayed in the toast; this can be a character string, an html element created with the `HTML` function, or a `shiny.tag` object such as `tags$span(style = "color:lime;", "Message")`

type	toast type, one of "info", "success", "warning", "error", "default" or "dark"
position	toast position, one of "top-left", "top-right", "top-center", "bottom-left", "bottom-right" or "bottom-center"
transition	the transition effect, one of "slide", "zoom", "flip" or "bounce"
autoClose	either a number, the time in ms to close the toast, or FALSE to close the toast manually
hideProgressBar	Boolean, whether to hide the progress bar
closeOnClick	Boolean, whether to dismiss the toast on click
rtl	Boolean, right to left
pauseOnFocusLoss	Boolean, whether to pause the toast on focus loss
draggable	Boolean, ability to drag the toast to remove it
draggableDirection	"x" or "y"
draggablePercent	the percentage of the width of the toast needed to remove it by dragging
pauseOnHover	Boolean, whether to pause the toast on hover
className	name of a CSS class applied to the container
toastClassName	name of a CSS class applied on the toast wrapper
bodyClassName	name of a CSS class applied on the toast body
progressClassName	name of a CSS class applied on the progress bar
style	inline style applied to the container, e.g. list(boxShadow = "rgba(0, 0, 0, 0.56) 0px 22px 30px 4px")
JScallback	some JavaScript code given as a string to be executed whenever the toast is closed; it will have an effect only if the position argument differs from the one of the toast to be updated

Value

No return value, called for side effect.

`useShinyToastify` *Use Shiny toastify*

Description

This function must be called once in your Shiny ui to allow to use showToast.

Usage

```
useShinyToastify()
```

Value

An object of class `shiny.tag.list`.

Index