

Package ‘sever’

July 23, 2025

Title Customise 'Shiny' Disconnected Screens and Error Messages

Version 0.0.7

Date 2021-07-14

Description Customise 'Shiny' disconnected screens as well as sanitize error messages to make them clearer and friendlier to the user.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.1.1

Imports cli, shiny, htmltools

URL <https://sever.john-coene.com/>

BugReports <https://github.com/JohnCoene/sever/issues>

NeedsCompilation no

Author John Coene [aut, cre]

Maintainer John Coene <jcoenep@gmail.com>

Repository CRAN

Date/Publication 2021-07-14 18:50:02 UTC

Contents

chisel	2
chisel_theme	3
cleave	3
cleave_theme	4
dependencies	5
reconnect	6
reload_button	7
rupture	8
rupture_default	9
sever	10
sever_default	11
using_golem	11

Index**12**

chisel

*Chisel***Description**

Customise silent error messages: messages raised by `shiny::validate()` and `shiny::need()`. If used in combination with `cleave()` make sure you are setting `silent_errors` to `FALSE`.

Usage

```
chisel(  
  html = NULL,  
  color = NULL,  
  bg_color = NULL,  
  duration = NULL,  
  center_vertical = NULL,  
  center_horizontal = NULL,  
  ids = NULL,  
  session = shiny::getDefaultReactiveDomain()  
)
```

Arguments

html	Html content to display instead of error messages, if NULL the original message is displayed.
color	Color of error message text.
bg_color	Background color of error message overlay. If NULL will be transparent.
duration	Duration of animation showing the error message, set to 0 to have none.
center_vertical	Whether to center the message vertically and horizontally, a boolean.
center_horizontal	Whether to center the message vertically and horizontally, a boolean.
ids	Ids of elements to apply the cleave to. If NULL applies to all error messages.
session	A valid shiny session.

Value

None

chisel_theme	<i>Chisel Theme</i>
--------------	---------------------

Description

Define a theme to apply to all subsequent `chisel()`.

Usage

```
chisel_theme(  
  color = "darkgrey",  
  bg_color = NULL,  
  duration = 0.1,  
  center_vertical = TRUE,  
  center_horizontal = TRUE  
)
```

Arguments

color	Color of error message text.
bg_color	Background color of error message overlay. If NULL will be transparent.
duration	Duration of animation showing the error message, set to 0 to have none.
center_vertical	Whether to center the message vertically and horizontally, a boolean.
center_horizontal	Whether to center the message vertically and horizontally, a boolean.

Value

None

cleave	<i>Cleave</i>
--------	---------------

Description

Customise hard error messages.

Usage

```

cleave(
  html = NULL,
  color = NULL,
  bg_color = NULL,
  duration = NULL,
  center_vertical = NULL,
  center_horizontal = NULL,
  ids = NULL,
  silent_errors = FALSE,
  session = shiny::getDefaultReactiveDomain()
)

```

Arguments

html	Html content to display instead of error messages, if NULL the original message is displayed.
color	Color of error message text.
bg_color	Background color of error message overlay. If NULL will be transparent.
duration	Duration of animation showing the error message, set to 0 to have none.
center_vertical, center_horizontal	Whether to center the message vertically and horizontally, a boolean.
ids	Ids of elements to apply the cleave to. If NULL applies to all error messages.
silent_errors	Set to TRUE to also cleave silent errors: errors that are raised by shiny::validate() . Alternatively you might want to use chisel() to specifically stylise those messages.
session	A valid shiny session.

Value

None

See Also

[chisel\(\)](#) for soft error messages.

cleave_theme

Cleave Theme

Description

Define a theme to apply to all subsequent [cleave\(\)](#).

Usage

```
cleave_theme(  
  color = "red",  
  bg_color = NULL,  
  duration = 0.1,  
  center_vertical = TRUE,  
  center_horizontal = TRUE  
)
```

Arguments

color	Color of error message text.
bg_color	Background color of error message overlay. If NULL will be transparent.
duration	Duration of animation showing the error message, set to 0 to have none.
center_vertical	Whether to center the message vertically and horizontally, a boolean.
center_horizontal	Whether to center the message vertically and horizontally, a boolean.

Value

None

dependencies

Dependencies

Description

Import dependencies, place this in your shiny UI.

Usage

```
useSever()  
  
use_sever()
```

Value

[shiny::tags](#) containing the necessary dependencies.

reconnect	<i>Reload</i>
-----------	---------------

Description

Create a button to reload/reconnect to shiny.

Usage

```
reconnect_button(  
  text = "reconnect",  
  class = c("default", "danger", "info", "success", "warning")  
)  
  
reconnect_link(  
  text = "reconnect",  
  class = c("default", "danger", "info", "success", "warning")  
)  
  
f7_reconnect_button(text = "reconnect", color = "#000")
```

Arguments

text	The text to use on the button.
class	The class to apply to the button.
color	Color of button.

Value

A button or link in the form of [shiny::tags](#).

Functions

- `reconnect_button` - Returns a button.
- `reconnect_link` - Returns a link.
- `f7_reconnect_button` - A reconnect button for shinyMobile.

reload_button	<i>Reload</i>
---------------	---------------

Description

Create a button to reload/reconnect to shiny.

Usage

```
reload_button(  
  text = "reload",  
  class = c("default", "danger", "info", "success", "warning")  
)  
  
reload_link(  
  text = "reload",  
  class = c("default", "danger", "info", "success", "warning")  
)  
  
f7_reload_button(text = "reload", color = "#000")
```

Arguments

text	The text to use on the button.
class	The class to apply to the button.
color	Color of button.

Value

A button or link in the form of [shiny::tags](#).

Functions

- reload_button - Returns a button.
- reload_link - Returns a link.
- f7_reload_button - A reload button for shinyMobile.

rupture

*Rupture***Description**

Displays a disconnected screen after `ms` milliseconds of inactivity.

Usage

```
rupture(
  html = rupture_default(),
  color = "#fff",
  opacity = 1,
  bg_color = "#333e48",
  bg_image = NULL,
  ms = 1000 * 60 * 15,
  session = shiny::getDefaultReactiveDomain(),
  box = FALSE
)
```

Arguments

<code>html</code>	Shiny tags to use as content for the disconnected screen, generally <code>shiny::tagList()</code> .
<code>opacity</code>	Opacity of background.
<code>bg_color</code> , <code>color</code>	Background color, color of text.
<code>bg_image</code>	Background image to use.
<code>ms</code>	Milliseconds before showing the disconnected screen, defaults to $1000 * 60 * 15$ which is 15 minutes (same as shinyapps.io).
<code>session</code>	A valid shiny session.
<code>box</code>	Set to TRUE to enclose the html in a box.

Value

None

Examples

```
library(shiny)

ui <- fluidPage(
  useSever(),
  h1("rupture")
)

server <- function(input, output){
  rupture(
```



```
      tagList(  
        h1("Whoops"),  
        reconnect_button()  
      )  
    )  
  }  
  
  if(interactive())  
    shinyApp(ui, server)
```

rupture_default

Default Rupture Screen

Description

The default rupture screen for convenience.

Usage

```
rupture_default(  
  title = "Idle",  
  subtitle = "Your session is disconnected",  
  button = "Reconnect",  
  button_class = "default"  
)
```

Arguments

title	Title and subtitle to display.
subtitle	Title and subtitle to display.
button	Text to display on button, passed to reload_button() .
button_class	Class of button, passed to reload_button() .

Value

shiny::tags.

sever

Sever

Description

Customise the Shiny disconnected screen.

Usage

```
sever(
  html = sever_default(),
  color = "#fff",
  opacity = 1,
  bg_color = "#333e48",
  bg_image = NULL,
  session = shiny::getDefaultReactiveDomain(),
  box = FALSE
)
```

Arguments

html	Shiny tags to use as content for the disconnected screen, generally <code>shiny::tagList()</code> .
opacity	Opacity of background.
bg_color, color	Background color, color of text.
bg_image	Background image to use.
session	A valid shiny session.
box	Set to TRUE to enclose the html in a box.

Value

None

Examples

```
library(shiny)

ui <- fluidPage(
  useSever(),
  h1("sever")
)

server <- function(input, output){
  sever(
    tagList(
      h1("Whoops"),
      reload_button()
    )
  )
}
```

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

`sever_default`*Default Sever Screen*

Description

The default sever screen for convenience.

Usage

```
sever_default(  
  title = "Whoops!",  
  subtitle = "You have been disconnected",  
  button = "Reload",  
  button_class = "default"  
)
```

Arguments

`title, subtitle` Title and subtitle to display.
`button` Text to display on button, passed to [reload_button\(\)](#).
`button_class` Class of button, passed to [reload_button\(\)](#).

Value

`shiny::tags`.

`using_golem`*Uses Golem*

Description

Checks if uses golem.

Usage

```
uses_golem()  
  
runs_golem()
```

Index

chisel, [2](#)
chisel(), [3](#), [4](#)
chisel_theme, [3](#)
cleave, [3](#)
cleave(), [2](#), [4](#)
cleave_theme, [4](#)

dependencies, [5](#)

f7_reconnect_button(reconnect), [6](#)
f7_reload_button(reload_button), [7](#)

reconnect, [6](#)
reconnect_button(reconnect), [6](#)
reconnect_link(reconnect), [6](#)
reload_button, [7](#)
reload_button(), [9](#), [11](#)
reload_link(reload_button), [7](#)
runs_golem(using_golem), [11](#)
rupture, [8](#)
rupture_default, [9](#)

sever, [10](#)
sever_default, [11](#)
shiny::need(), [2](#)
shiny::tagList(), [8](#), [10](#)
shiny::tags, [5–7](#)
shiny::validate(), [2](#), [4](#)

use_sever(dependencies), [5](#)
uses_golem(using_golem), [11](#)
useSever(dependencies), [5](#)
using_golem, [11](#)