

Package ‘waiter’

September 13, 2020

Title Loading Screen for 'Shiny'

Version 0.1.3

Date 2020-09-12

Description

Full screen and partial loading screens for 'Shiny' with spinners, progress bars, and notifications.

License MIT + file LICENSE

URL <https://waiter.john-coene.com/>,
<https://github.com/JohnCoene/waiter>

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

Encoding UTF-8

LazyData true

Imports R6, shiny, crayon, magrittr, htmltools, rstudioapi

RoxygenNote 7.1.1.9000

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author John Coene [aut, cre],
Jinhwan Kim [ctb]

Maintainer John Coene <jcoenep@gmail.com>

Repository CRAN

Date/Publication 2020-09-13 10:50:03 UTC

R topics documented:

butler	2
butlerClass	3
garcon	5
hostess	8
hostessLoader	13

preview_spinner	15
spinners	16
transparent	19
use_steward	20
waiter	20
waiterClass	23
waiterTheme	25
waitress	25
waitressClass	26

Index	32
--------------	-----------

butler	<i>Butler</i>
--------	---------------

Description

Programmatically show and hide loading bar.

Usage

```
use_butler()

show_butler()

hide_butler()

config_butler(
    thickness = 5,
    colors = list(`0` = "red", `.3` = "blue", `1` = "green"),
    shadow_blur = 5,
    shadow_color = "rgba(0, 0, 0, .5)"
)
```

Arguments

thickness	Thickness of the bar.
colors	List of gradient color stops used to draw the progress bar.
shadow_blur	Shadow blur size.
shadow_color	Shadow color.

Functions

- `use_butler`: butler dependencies to include anywhere in your UI but ideally at the top.
- `show_butler`: Show a butler.
- `hide_butler`: Hide butler.
- `config_butler`: Configure the butler.

Class

Arguments passed to `config_butler` are passed to the initialisation method `new`.

- Butler: initialise a Butler.

Examples

```
library(shiny)

ui <- fluidPage(
  use_butler(),
  br(),
  actionButton("show", "show butler"),
  actionButton("hide", "hide butler")
)

server <- function(input, output){

  observeEvent(input$show,{
    show_butler()
  })

  observeEvent(input$hide,{
    hide_butler()
  })

}

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

butlerClass

Butler R6 Class

Description

Create a butler.

Details

Create an object to show a loading bar to display at the top of the application.

Methods

Public methods:

- `Butler$new()`
- `Butler$show()`
- `Butler$print()`

- [Butler\\$hide\(\)](#)
- [Butler\\$clone\(\)](#)

Method new():

Usage:

```
Butler$new(  
  thickness = 5,  
  colors = list(`0` = "red", `3` = "blue", `1` = "green"),  
  shadow_blur = 5,  
  shadow_color = "rgba(0, 0, 0, .5)"  
)
```

Arguments:

`thickness` Thickness of the bar.

`colors` List of gradient color stops used to draw the progress bar.

`shadow_blur` Shadow blur size.

`shadow_color` Shadow color.

Details: Create a butler.

Examples:

```
\dontrun{Butler$new()}
```

Method show():

Usage:

```
Butler$show()
```

Details: Show the butler.

Examples:

```
\dontrun{Butler$new()$show()}
```

Method print():

Usage:

```
Butler$print()
```

Details: print the butler

Method hide():

Usage:

```
Butler$hide()
```

Details: Hide the butler.

Examples:

```
\dontrun{Butler$new()$show()$hide()}
```

Method clone(): The objects of this class are cloneable with this method.

Usage:

```
Butler$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Examples

```
## -----  
## Method `Butler$new`  
## -----  
  
## Not run: Butler$new()  
  
## -----  
## Method `Butler$show`  
## -----  
  
## Not run: Butler$new()$show()  
  
## -----  
## Method `Butler$hide`  
## -----  
  
## Not run: Butler$new()$show()$hide()
```

garcon

Garcon

Description

Create a garcon to animate images on the waiter.

Usage

```
use_garcon()
```

Methods

Public methods:

- [Garcon\\$new\(\)](#)
- [Garcon\\$set\(\)](#)
- [Garcon\\$inc\(\)](#)
- [Garcon\\$reset\(\)](#)
- [Garcon\\$destroy\(\)](#)
- [Garcon\\$print\(\)](#)
- [Garcon\\$stop\(\)](#)
- [Garcon\\$close\(\)](#)
- [Garcon\\$clone\(\)](#)

Method `new()`:

Usage:

```
Garçon$new(
  image,
  bg_color = "#FFFFFF",
  opacity = 0.5,
  direction = c("bt", "tb", "lr", "rl"),
  filter = NULL
)
```

Arguments:

image The CSS id of the image tag.

bg_color Background overlay color in hexadecimal or RGB.

opacity Overlay transparency.

direction Animation direction. Possible values: lr (left to right), rl (right to left), bt (bottom to top), tb (top to bottom).

filter Filter to apply, options are blur, grayscale, sepia, hue-rotate, invert, opacity.

Details: Initialise the garçon.

Examples:

```
\dontrun{Garçon$new("img")$set(30)}
```

Method set():*Usage:*

```
Garçon$set(value)
```

Arguments:

value Percentage to set to.

Details: Value to set the garçon to.

Examples:

```
\dontrun{Garçon$new("img")$set(30)}
```

Method inc():*Usage:*

```
Garçon$inc(value)
```

Arguments:

value Percentage to increase to.

Details: Value to increase the garçon to.

Examples:

```
\dontrun{Garçon$new("img")$inc(30)}
```

Method reset():*Usage:*

```
Garçon$reset(value)
```

Arguments:

value Percentage to set to.

Details: Reset the garçon to.

Examples:

```
\dontrun{Garçon$new("img")$set(30)$reset()}
```

Method destroy():

Usage:

```
Garçon$destroy()
```

Details: Kill the garçon to.

Examples:

```
\dontrun{Garçon$new("img")$set(30)$destroy()}
```

Method print():

Usage:

```
Garçon$print()
```

Details: print the garçon

Method stop():

Usage:

```
Garçon$stop()
```

Details: Stop the garçon.

Examples:

```
\dontrun{Garçon$new("img")$set(30)$stop()}
```

Method close():

Usage:

```
Garçon$close()
```

Details: Close the garçon.

Examples:

```
\dontrun{Garçon$new("img")$set(30)$close()}
```

Method clone(): The objects of this class are cloneable with this method.

Usage:

```
Garçon$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

Examples

```

## -----
## Method `Garcon$new`
## -----

## Not run: Garcon$new("img")$set(30)

## -----
## Method `Garcon$set`
## -----

## Not run: Garcon$new("img")$set(30)

## -----
## Method `Garcon$inc`
## -----

## Not run: Garcon$new("img")$inc(30)

## -----
## Method `Garcon$reset`
## -----

## Not run: Garcon$new("img")$set(30)$reset()

## -----
## Method `Garcon$destroy`
## -----

## Not run: Garcon$new("img")$set(30)$destroy()

## -----
## Method `Garcon$stop`
## -----

## Not run: Garcon$new("img")$set(30)$stop()

## -----
## Method `Garcon$close`
## -----

## Not run: Garcon$new("img")$set(30)$close()

```

hostess

Hostess

Description

Add hostess dependencies.

Usage

```
use_hostess()

hostess_init(id = "hostess")

hostess_set(id = "hostess", value)
```

Arguments

id	Id of hostess (valid CSS).
value	Value to set, between 0 and 100.

Methods**Public methods:**

- [Hostess\\$new\(\)](#)
- [Hostess\\$start\(\)](#)
- [Hostess\\$print\(\)](#)
- [Hostess\\$set\(\)](#)
- [Hostess\\$inc\(\)](#)
- [Hostess\\$get_loader\(\)](#)
- [Hostess\\$set_loader\(\)](#)
- [Hostess\\$notify\(\)](#)
- [Hostess\\$clone\(\)](#)

Method new():*Usage:*

```
Hostess$new(id = NULL, min = 0, max = 100, n = 1)
```

Arguments:

id Id used in `hostess_loader` if you generate the loader with the `loader` method you may leave this NULL.

min, max Minimum and maximum representing the starting and ending points of the progress bar.

n Number of loaders to generate.

Details: Create a hostess.*Examples:*

```
\dontrun{Hostess$new()}
```

Method start():*Usage:*

```
Hostess$start()
```

Details: Start the hostess**Method print():**

Usage:

```
Hostess$print()
```

Details: Print the hostess

Method set():*Usage:*

```
Hostess$set(value)
```

Arguments:

value Value to set, between 0 and 100.

Details: Set the hostess loading bar.

Examples:

```
\dontrun{Hostess$new()$set(20)}
```

Method inc():*Usage:*

```
Hostess$inc(value)
```

Arguments:

value Value to set, between 0 and 100.

Details: Increase the hostess loading bar.

Examples:

```
\dontrun{Hostess$new()$inc(10)}
```

Method get_loader():*Usage:*

```
Hostess$get_loader(
  preset = NULL,
  text_color = "#FFFFFF",
  center_page = FALSE,
  class = "",
  min = NULL,
  max = NULL,
  svg = NULL,
  progress_type = c("stroke", "fill"),
  fill_direction = c("btt", "ttb", "ltr", "rtl"),
  stroke_direction = c("normal", "reverse"),
  fill_color = NULL,
  stroke_color = NULL,
  ...
)
```

Arguments:

preset A loading bar preset, see section below.

text_color The color of the loading text.

`center_page` By default the hostess is centered in the middle of the screen, ideal when using it with waiter full screen, set to FALSE to prevent that.

`class` CSS class.

`min`, `max` Minimum and maximum representing the starting and ending points of the progress bar.

`svg` Either an svg path e.g.: `M10 10L90 10` or the path to a .svg file. Note that if passing the latter it must be made available to Shiny by placing it either in the www folder or using `shiny::addResourcePath()`.

`progress_type` The progress type, either `stroke` or `fill`. The former traces the path of the bar while the latter fills it progressively.

`fill_direction`, `stroke_direction` The direction which the progress bar should take. Whether `fill_direction` or `stroke_direction` is used depends on `progress_type`.

`fill_color`, `stroke_color` The color to use for the progress bar. Whether `fill_color` or `stroke_color` is used depends on `progress_type`.

... Any other other advanced options to pass to the loader see the [official documentation](#).

`value` Value to set, between 0 and 100.

`with_waiter` Deprecate in favour of `center_page`.

Details: Create a hostess loading bar.

Examples:

```
\dontrun{Hostess$new()$get_loader()}
```

Method `set_loader()`:

Usage:

```
Hostess$set_loader(loader)
```

Arguments:

`loader` Loader as defined by `hostess_loader()`.

Details: Set a hostess loader as defined by `hostess_loader()`.

Examples:

```
\dontrun{
  loader <- hostess_loader()
  Hostess$new()$set_loader(loader)
}
```

Method `notify()`:

Usage:

```
Hostess$notify(
  html = NULL,
  background_color = "transparent",
  text_color = "black",
  position = c("br", "tr", "bl", "tl")
)
```

Arguments:

`html` Additional HTML content of the tag or a character string.

background_color Background color of the notification.
 text_color Color of text of html.
 position Position of the notification on the screen. Where br is the bottom-right, tr is the top-right, bl is bottom-left, and tl is the top-left.

Details: Use the hostess as a notification. It is hidden when set tpo 100.

Examples:

```
\dontrun{Hostess$new()$notify()}
```

Method clone(): The objects of this class are cloneable with this method.

Usage:

```
Hostess$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----
## Method `Hostess$new`
## -----

## Not run: Hostess$new()

## -----
## Method `Hostess$set`
## -----

## Not run: Hostess$new()$set(20)

## -----
## Method `Hostess$inc`
## -----

## Not run: Hostess$new()$inc(10)

## -----
## Method `Hostess$get_loader`
## -----

## Not run: Hostess$new()$get_loader()

## -----
## Method `Hostess$set_loader`
## -----

## Not run:
loader <- hostess_loader()
Hostess$new()$set_loader(loader)
```

```
## End(Not run)

## -----
## Method `Hostess$notify`
## -----

## Not run: Hostess$new()$notify()
```

hostessLoader	<i>Loader</i>
---------------	---------------

Description

Customise the Hostess loading bar.

Usage

```
hostess_loader(
  id = "hostess",
  preset = NULL,
  text_color = "#FFFFFF",
  center_page = FALSE,
  class = "",
  min = 0,
  max = 100,
  svg = NULL,
  progress_type = c("stroke", "fill"),
  fill_direction = c("btt", "ttb", "ltr", "rtl"),
  stroke_direction = c("normal", "reverse"),
  fill_color = NULL,
  stroke_color = NULL,
  ...,
  with_waiter
)

hostess_gradient(angle = 0, duration = 1, colors = c("red", "white", "blue"))

hostess_bubble(
  color_background = "#697682",
  color_bubble = "#f7fff7",
  count = 25,
  duration = 1
)

hostess_stripe(color1 = "#697682", color2 = "#f7fff7", duration = 1)
```

Arguments

id	Id of hostess (valid CSS).
preset	A loading bar preset, see section below.
text_color	The color of the loading text.
center_page	By default the hostess is <i>not</i> centered in the middle of the screen, centering in the middle of the page is however ideal when using it with waiter full screen, for the latter set to TRUE.
class	CSS class.
min, max	Minimum and maximum representing the starting and ending points of the progress bar.
svg	Either an svg path e.g.: M10 10L90 10 or the path to a .svg file. Note that if passing the latter it must be made available to Shiny by placing it either in the www folder or using <code>shiny::addResourcePath()</code> .
progress_type	The progress type, either stroke or fill. Ther former traces the path of the svg while the latter fills it progressively.
fill_direction, stroke_direction	The direction which the progress bar should take. Wether fill_direction or stroke_direction is used depends on progress_type.
fill_color, stroke_color	The color to use for the progress bar. Wether fill_color or stroke_color is used depends on progress_type.
...	Any other other advanced options to pass to the loaded see the official documentation .
with_waiter	Deprecate in favour of center_page.
angle	Angle of gradient.
duration	Duration of the loop.
colors	Color vectors composing the gradient.
color_background	The background of the color.
color_bubble	The color of the bubbles contour.
count	The number of bubbles.
color1, color2	Colors of stripes.

Presets

- line
- fan
- circle
- bubble
- rainbow
- energy
- stripe
- text

Examples

```
library(shiny)
library(waiter)

# diagonal line
path <- "M10 10L90 30"

ui <- fluidPage(
  use_waiter(),
  use_hostess(),
  actionButton("draw", "redraw"),
  plotOutput("plot")
)

server <- function(input, output) {

  dataset <- reactive({
    input$draw

    hostess <- Hostess$new(min = 0, max = 10)
    hostess$set_loader <- hostess_loader(
      progress_type = "stroke",
      stroke_color = hostess_stripe()
    )
    waiter <- Waiter$new(
      "plot",
      hostess$loader()
    )

    waiter$show()

    for(i in 1:10){
      Sys.sleep(.2)
      hostess$inc(1)
    }

    runif(100)

  })

  output$plot <- renderPlot(plot(dataset()))

}

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

Description

Allows previewing spinners in web browser or RStudio Viewer.

Usage

```
preview_spinner(spinner, bg_color = "black")
```

Arguments

spinner	A waiter link{spinner}.
bg_color	Background color.

Examples

```
if(interactive()) preview_spinner(spin_1())
```

spinners

Spinners

Description

Spinkit spinners to use with [show_waiter](#).

Usage

```
spin_rotating_plane()  
spin_fading_circles()  
spin_folding_cube()  
spin_double_bounce()  
spin_wave()  
spin_wandering_cubes()  
spin_pulse()  
spin_chasing_dots()  
spin_three_bounce()  
spin_circle()  
spin_rotate()
```


spin_solar()
spin_orbit()
spin_squares()
spin_cube_grid()
spin_circles()
spin_orbiter()
spin_pixel()
spin_flower()
spin_dual_ring()
spin_heart()
spin_ellipsis()
spin_facebook()
spin_hourglass()
spin_ring()
spin_ripple()
spin_terminal()
spin_loader()
spin_throbber()
spin_refresh()
spin_heartbeat()
spin_gauge()
spin_3k()
spin_wobblebar()
spin_atebits()

```
spin_whirly()
spin_flowers()
spin_dots()
spin_3circles()
spin_plus()
spin_pulsar()
spin_hexdots()
spin_inner_circles()
spin_pong()
spin_timer()
spin_ball()
spin_dual_circle()
spin_seven_circle()
spin_clock()
spin_pushing_shapes()
spin_fill()
spin_rhombus()
spin_balance()
spin_square_circle()
spin_circle_square()
spin_puzzle()
spin_half()
spin_loaders(id = 1, color = "white", style = NULL)
spin_1()
```

```
spin_2()  
spin_3()  
spin_4()  
spin_5()  
spin_6()  
spin_google()  
browse_waiters()
```

Arguments

id	The spinner identifier, an integer between 1, and 42.
color	Desired color of spinner.
style	CSS style to apply to spinner.

Details

Much of the CSS used is to provide those spinners. One can greatly reduce the load on the browser by only sourcing the CSS for the spinners required. You can find out which CSS kits are required to load by using the spinner in the R console as shown in the example. This prints the kit and instructions to only source the required file.

Value

An object of class spinner.

Examples

```
spin_rotating_plane()
```

transparent	<i>Transparency</i>
-------------	---------------------

Description

A convenience function to create a waiter with transparent background.

Usage

```
transparent(alpha = 0)
```

Arguments

alpha Alpha channel where 0 is completely transparent and 1 is opaque.

Examples

```
transparent()
```

use_steward	<i>Steward</i>
-------------	----------------

Description

A colorful steward to work with the [waiter](#).

Usage

```
use_steward(
  colors = c("#ee7752", "#e73c7e", "#23a6d5", "#23d5ab"),
  speed = 30,
  angle = -45
)
```

Arguments

colors Color palette forming gradient.
 speed Seconds it takes to loop over colors.
 angle Degrees at which colors slide.

waiter	<i>Waiter</i>
--------	---------------

Description

Programmatically show and hide loading screens.

Usage

```
use_waiter(spinner = 1:7, include_js = TRUE)

waiter_use(spinner = 1:7, include_js = TRUE)

show_waiter(
  html = "",
  color = "#333e48",
  logo = "",
  id = NULL,
  hide_on_render = FALSE
)

waiter_show(
  id = NULL,
  html = spin_1(),
  color = "#333e48",
  logo = "",
  hide_on_render = !is.null(id)
)

show_waiter_on_load(html = "", color = "#333e48", logo = "")

waiter_show_on_load(html = spin_1(), color = "#333e48", logo = "")

hide_waiter_on_drawn(id)

waiter_hide_on_render(id)

waiter_on_busy(html = spin_1(), color = "#333e48", logo = "")

hide_waiter(id = NULL)

waiter_hide(id = NULL)

update_waiter(html = "", id = NULL)

waiter_update(id = NULL, html = NULL)
```

Arguments

<code>spinners</code>	Spinners to include. By default all the CSS files for all spinners are included you can customise this only that which you need in order to reduce the amount of CSS that needs to be loaded and improve page loading speed. There are 7 spinner kits. The spinner kit required for the spinner you use is printed in the R console when using the spinner. You can specify a single spinner kit e.g.: 1 or multiple spinner kits as a vector e.g.: <code>c(1, 3, 6)</code> .
<code>include_js</code>	Deprecated argument, no longer needed.

html	HTML content of waiter, generally a spinner, see spinners .
color	Background color of loading screen.
logo	Logo to display.
id	Id of element to hide or element on which to show waiter over.
hide_on_render	Set to TRUE to automatically hide the waiter when the plot in id is drawn. Note the latter will only work with shiny plots, tables, htmlwidgets, etc. but will not work with arbitrary elements.

Functions

- `use_waiter` and `waiter_use`: waiter dependencies to include anywhere in your UI but ideally at the top.
- `waiter_show_on_load`: Show a waiter on page load, before the session is even loaded, include in UI *after* `use_waiter`.
- `waiter_show`: Show waiting screen.
- `waiter_hide`: Hide any waiting screen.
- `waiter_on_busy`: Automatically shows the waiting screen when the server is busy, and hides it when it goes back to idle.
- `waiter_update`: Update the content `html` of the waiting screen.
- `waiter_hide_on_render`: Hide any waiting screen when the output is drawn, useful for outputs that take a long time to draw, *use in ui*.

Examples

```
library(shiny)

ui <- fluidPage(
  use_waiter(), # dependencies
  waiter_show_on_load(spin_fading_circles()), # shows before anything else
  actionButton("show", "Show loading for 5 seconds")
)

server <- function(input, output, session){
  waiter_hide() # will hide *on_load waiter

  observeEvent(input$show, {
    waiter_show(
      html = tagList(
        spin_fading_circles(),
        "Loading ..."
      )
    )
    Sys.sleep(3)
    waiter_hide()
  })
}

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

waiterClass

*Waiter R6 Class***Description**

Create a waiter to then show, hide or update its content.

Details

Create an object to show a waiting screen on either the entire application or just a portion of the app by specifying the id. Then show, then hide or meanwhile update the content of the waiter.

Methods**Public methods:**

- `Waiter$new()`
- `Waiter$show()`
- `Waiter$hide()`
- `Waiter$update()`
- `Waiter$print()`
- `Waiter$clone()`

Method new():*Usage:*

```
Waiter$new(
  id = NULL,
  html = NULL,
  color = NULL,
  logo = NULL,
  hide_on_render = !is.null(id),
  hide_on_error = !is.null(id),
  hide_on_silent_error = !is.null(id)
)
```

Arguments:

`id` Id, or vector of ids, of element on which to overlay the waiter, if NULL the waiter is applied to the entire body.

`html` HTML content of waiter, generally a spinner, see [spinners](#) or a list of the latter.

`color` Background color of loading screen.

`logo` Logo to display.

`hide_on_render` Set to TRUE to automatically hide the waiter when the element in `id` is drawn. Note the latter will work with shiny plots, tables, htmlwidgets, etc. but will not work with arbitrary elements.

`hide_on_error`, `hide_on_silent_error` Whether to hide the waiter when the underlying element throws an error. Silent error are thrown by [req](#) and [validate](#).

Details: Create a waiter.

Examples:

```
\dontrun{Waiter$new()}
```

Method show():

Usage:

```
Waiter$show()
```

Details: Show the waiter.

Method hide():

Usage:

```
Waiter$hide()
```

Details: Hide the waiter.

Method update():

Usage:

```
Waiter$update(html = NULL)
```

Arguments:

html HTML content of waiter, generally a spinner, see [spinners](#).

Details: Update the waiter's html content.

Method print():

Usage:

```
Waiter$print()
```

Details: print the waiter

Method clone(): The objects of this class are cloneable with this method.

Usage:

```
Waiter$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----
## Method `Waiter$new`
## -----

## Not run: Waiter$new()
```

waiterTheme	<i>Define a Theme</i>
-------------	-----------------------

Description

Define a theme to be used by all waiter loading screens. These can be overridden in individual loading screens.

Usage

```
waiter_set_theme(html = spin_1(), color = "#333e48", logo = "")
waiter_get_theme()
waiter_unset_theme()
```

Arguments

html	HTML content of waiter, generally a spinner, see spinners .
color	Background color of loading screen.
logo	Logo to display.

waitress	<i>Waitress</i>
----------	-----------------

Description

Programatically show and hide loading bars.

Usage

```
use_waitress(color = "#697682", percent_color = "#333333")

call_waitress(
  selector = NULL,
  theme = c("line", "overlay", "overlay-radius", "overlay-opacity", "overlay-percent"),
  min = 0,
  max = 100,
  infinite = FALSE
)

browse_waitresses()
```

Arguments

color, percent_color	Color of waitress and color of percent text shown when theme is set to overlay-percent.
selector	Element selector to apply the waitress to, if NULL then the waitress is applied to the whole screen.
theme	A valid theme, see function usage.
min, max	Minimum and maximum representing the starting and ending points of the progress bar.
infinite	Set to TRUE to create a never ending loading bar, ideal when you cannot compute increments or assess the time it might take before the loading bar should be removed.

Details

You can pipe the methods with `$. Waitress$new()` and `call_waitress()` are equivalent.

Examples

```
library(shiny)

ui <- fluidPage(
  use_waitress("red"), # dependencies
  sliderInput("set", "percentage", 1, 100, step = 5, value = 1)
)

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

  w <- Waitress$
  new()$ # call a waitress
  start() # start waitress

  observeEvent(input$set, {
    w$set(input$set) # set at percentage
  })
}

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

waitressClass

Waitress R6 Class

Description

Create a waitress (progress bar) and programmatically set or increase its percentage, then hide it when done.

Methods**Public methods:**

- `Waitress$new()`
- `Waitress$start()`
- `Waitress$notify()`
- `Waitress$set()`
- `Waitress$auto()`
- `Waitress$increase()`
- `Waitress$inc()`
- `Waitress$hide()`
- `Waitress$close()`
- `Waitress$print()`
- `Waitress$clone()`

Method new():*Usage:*

```
Waitress$new(
  selector = NULL,
  theme = c("line", "overlay", "overlay-radius", "overlay-opacity", "overlay-percent"),
  min = 0,
  max = 100,
  infinite = FALSE,
  hide_on_render = FALSE
)
```

Arguments:

`selector` Element selector to apply the waitress to, if NULL then the waitress is applied to the whole screen.

`theme` A valid theme, see function usage.

`min`, `max` Minimum and maximum representing the starting and ending points of the progress bar.

`infinite` Set to TRUE to create a never ending loading bar, ideal when you cannot compute increments or assess the time it might take before the loading bar should be removed.

`hide_on_render` Set to TRUE to automatically hide the waitress when the element in `id` is rendered. Note the latter will work with shiny plots, tables, htmlwidgets, etc. but will not work with arbitrary elements.

`color`, `percent_color` Color of waitress and color of percent text shown when theme is set to `overlay-percent`.

Details: Create a waitress.*Examples:*

```
\dontrun{Waitress$new("#plot")}
```

Method start():*Usage:*

```
Waitress$start(  
  html = NULL,  
  background_color = "transparent",  
  text_color = "black"  
)
```

Arguments:

html HTML content to show over the waitress, accepts htmltools and shiny tags.

background_color The background color of the html.

text_color The color of the html content.

Details: Start the waitress.

Examples:

```
\dontrun{Waitress$new("#plot")$start()}
```

Method notify():

Usage:

```
Waitress$notify(  
  html = NULL,  
  background_color = "white",  
  text_color = "black",  
  position = c("br", "tr", "bl", "tl")  
)
```

Arguments:

html HTML content to show over the waitress, accepts htmltools and shiny tags.

background_color The background color of the html.

text_color The color of the html content.

position Position of the notification on the screen. Where br is the bottom-right, tr is the top-right, bl is bottom-left, and tl is the top-left.

Details: Show the waitress as a notification.

Examples:

```
\dontrun{Waitress$new()$notify()}
```

Method set():

Usage:

```
Waitress$set(value)
```

Arguments:

value Value to set waitress to.

Details: Set the waitress to a specific percentage.

Examples:

```
\dontrun{Waitress$new("#plot")$set(20)}
```

Method auto():

Usage:

Waitress\$auto(value, ms)

Arguments:

value Value to set waitress to.

ms Number of Milliseconds

Details: Automatically start and end the waitress.

Examples:

```
\dontrun{Waitress$new("#plot")$auto(20, 2000)}
```

Method increase():

Usage:

```
Waitress$increase(percent)
```

Arguments:

percent Percentage to increase waitress to.

Details: Deprecated in favour of inc method. Increase the waitress by a percentage.

Method inc():

Usage:

```
Waitress$inc(value)
```

Arguments:

value Value to increase waitress to.

Details: Increase the waitress by a percentage.

Examples:

```
\dontrun{Waitress$new("#plot")$inc(30)}
```

Method hide():

Usage:

```
Waitress$hide()
```

Details: Deprecated in favour of close method. Hide the waitress.

Method close():

Usage:

```
Waitress$close()
```

Details: Close the waitress.

Examples:

```
\dontrun{Waitress$new("#plot")$close()}
```

Method print():

Usage:

```
Waitress$print()
```

Details: Print the waitress.

Examples:

```
\dontrun{Waitress$new("#plot")$hide()}
```

Method clone(): The objects of this class are cloneable with this method.

Usage:

```
Waitress$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----
## Method `Waitress$new`
## -----

## Not run: Waitress$new("#plot")

## -----
## Method `Waitress$start`
## -----

## Not run: Waitress$new("#plot")$start()

## -----
## Method `Waitress$notify`
## -----

## Not run: Waitress$new()$notify()

## -----
## Method `Waitress$set`
## -----

## Not run: Waitress$new("#plot")$set(20)

## -----
## Method `Waitress$auto`
## -----

## Not run: Waitress$new("#plot")$auto(20, 2000)

## -----
## Method `Waitress$inc`
## -----

## Not run: Waitress$new("#plot")$inc(30)

## -----
## Method `Waitress$close`
## -----
```

```
## Not run: Waitress$new("#plot")$close()

## -----
## Method `Waitress$print`
## -----

## Not run: Waitress$new("#plot")$hide()
```

Index

browse_waiters (spinners), 16
browse_waitresses (waitress), 25
Butler (butlerClass), 3
butler, 2
butlerClass, 3

call_waitress (waitress), 25
config_butler (butler), 2

Garcon (garcon), 5
garcon, 5

hide_butler (butler), 2
hide_waiter (waiter), 20
hide_waiter_on_drawn (waiter), 20
Hostess (hostess), 8
hostess, 8
hostess_bubble (hostessLoader), 13
hostess_gradient (hostessLoader), 13
hostess_init (hostess), 8
hostess_loader (hostessLoader), 13
hostess_loader(), 11
hostess_set (hostess), 8
hostess_stripe (hostessLoader), 13
hostessLoader, 13

preview_spinner, 15

req, 23

shiny::addResourcePath(), 11, 14
show_butler (butler), 2
show_waiter, 16
show_waiter (waiter), 20
show_waiter_on_load (waiter), 20
spin_1 (spinners), 16
spin_2 (spinners), 16
spin_3 (spinners), 16
spin_3circles (spinners), 16
spin_3k (spinners), 16
spin_4 (spinners), 16
spin_5 (spinners), 16
spin_6 (spinners), 16
spin_atebits (spinners), 16
spin_balance (spinners), 16
spin_ball (spinners), 16
spin_chasing_dots (spinners), 16
spin_circle (spinners), 16
spin_circle_square (spinners), 16
spin_circles (spinners), 16
spin_clock (spinners), 16
spin_cube_grid (spinners), 16
spin_dots (spinners), 16
spin_double_bounce (spinners), 16
spin_dual_circle (spinners), 16
spin_dual_ring (spinners), 16
spin_ellipsis (spinners), 16
spin_facebook (spinners), 16
spin_fading_circles (spinners), 16
spin_fill (spinners), 16
spin_flower (spinners), 16
spin_flowers (spinners), 16
spin_folding_cube (spinners), 16
spin_gauge (spinners), 16
spin_google (spinners), 16
spin_half (spinners), 16
spin_heart (spinners), 16
spin_heartbeat (spinners), 16
spin_hexdots (spinners), 16
spin_hourglass (spinners), 16
spin_inner_circles (spinners), 16
spin_loader (spinners), 16
spin_loaders (spinners), 16
spin_orbit (spinners), 16
spin_orbiter (spinners), 16
spin_pixel (spinners), 16
spin_plus (spinners), 16
spin_pong (spinners), 16
spin_pulsar (spinners), 16
spin_pulse (spinners), 16

- spin_pushing_shapes (spinners), 16
- spin_puzzle (spinners), 16
- spin_refresh (spinners), 16
- spin_rhombus (spinners), 16
- spin_ring (spinners), 16
- spin_ripple (spinners), 16
- spin_rotate (spinners), 16
- spin_rotating_plane (spinners), 16
- spin_seven_circle (spinners), 16
- spin_solar (spinners), 16
- spin_square_circle (spinners), 16
- spin_squares (spinners), 16
- spin_terminal (spinners), 16
- spin_three_bounce (spinners), 16
- spin_throbber (spinners), 16
- spin_timer (spinners), 16
- spin_wandering_cubes (spinners), 16
- spin_wave (spinners), 16
- spin_whirly (spinners), 16
- spin_wobblebar (spinners), 16
- spinners, 16, 22–25

- transparent, 19

- update_waiter (waiter), 20
- use_butler (butler), 2
- use_garcon (garcon), 5
- use_hostess (hostess), 8
- use_steward, 20
- use_waiter (waiter), 20
- use_waitress (waitress), 25

- validate, 23

- Waiter (waiterClass), 23
- waiter, 20, 20
- waiter_get_theme (waiterTheme), 25
- waiter_hide (waiter), 20
- waiter_hide_on_render (waiter), 20
- waiter_on_busy (waiter), 20
- waiter_set_theme (waiterTheme), 25
- waiter_show (waiter), 20
- waiter_show_on_load (waiter), 20
- waiter_unset_theme (waiterTheme), 25
- waiter_update (waiter), 20
- waiter_use (waiter), 20
- waiterClass, 23
- waiterTheme, 25
- Waitress (waitressClass), 26

- waitress, 25
- waitressClass, 26