Package: cicerone (via r-universe)

October 18, 2024

Title Provide Tours of 'Shiny' Applications

Version 1.0.5.9000

Date 2021-01-10

Description Provide step by step guided tours of 'Shiny' applications.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports R6, shiny, htmltools, assertthat

RoxygenNote 7.2.3

Roxygen list(markdown = TRUE)

URL https://cicerone.john-coene.com/

BugReports https://github.com/JohnCoene/cicerone/issues

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

RemoteUrl https://github.com/johncoene/cicerone

RemoteRef HEAD

RemoteSha da81fedad5af915baf4eb747dc081281583a219d

Contents

`Cicerone` *Define Steps*

Description

Define cicerone steps.

Position

- left
- right
- left-center
- left-bottom
- top
- top-center
- top-right
- right
- right-center
- right-bottom
- bottom
- bottom-center
- mid-center

Methods

Public methods:

- `Cicerone$new()`
- `Cicerone$step()`
- `Cicerone$init()`
- `Cicerone$reset()`
- `Cicerone$start()`
- `Cicerone$move_forward()`
- `Cicerone$move_backward()`
- `Cicerone$highlight()`
- `Cicerone$get_highlighted_el()`
- `Cicerone$get_previous_el()`
- `Cicerone$has_next_step()`
- `Cicerone$get_next()`
- `Cicerone$get_previous()`
- `Cicerone$clone()`

Method `new()`**:**

Usage:

```
Cicerone$new(
  animate = TRUE,
  opacity = 0.75,
  padding = 10,
  allow_close = TRUE,
  overlay_click_next = FALSE,
  done_btn_text = "Done",
  close_btn_text = "Close",
  stage_background = "#ffffff",
  next_btn_text = "Next",
  prev_btn_text = "Previous",
  show_btns = TRUE,
  keyboard_control = TRUE,
  id = NULL,
  mathjax = FALSE
)
```

Arguments:

`animate` Whether to animate or not.

`opacity` Background opacity (0 means only popovers and without overlay).

`padding` Distance of element from around the edges.

`allow_close` Whether the click on overlay should close or not.

`overlay_click_next` Whether the click on overlay should move next.

`done_btn_text` Text on the final button.

`close_btn_text` Text on the close button for this step.

`stage_background` Background color for the staged behind highlighted element.

`next_btn_text` Next button text for this step.

`prev_btn_text` Previous button text for this step.

`show_btns` Do not show control buttons in footer.

`keyboard_control` Allow controlling through keyboard (escape to close, arrow keys to move).

`id` A unique identifier, useful if you are using more than one cicerone.

`mathjax` Whether to use MathJax in the steps.

Details: Create a new `Cicerone` object.

Returns: A Cicerone object.

Method `step()`**:**

Usage:

```
Cicerone$step(
  el,
  title = NULL,
  description = NULL,
  position = NULL,
  class = NULL,
```

```
  show_btns = NULL,
  close_btn_text = NULL,
  next_btn_text = NULL,
  prev_btn_text = NULL,
  tab = NULL,
  tab_id = NULL,
  is_id = NULL,
  on_highlighted = NULL,
  on_highlight_started = NULL,
  on_next = NULL
)
```

Arguments:

`el` Id of element to be highlighted.

`title` Title on the popover.

`description` Body of the popover.

`position` Where to position the popover. See positions section.

`class` className to wrap this specific step popover in addition to the general className in Driver options.

`show_btns` Whether to show control buttons.

`close_btn_text` Text on the close button.

`next_btn_text` Next button text.

`prev_btn_text` Previous button text.

`tab` The name of the tab to set.

`tab_id` The id of the tabs to activate in order to highlight `tab_id`.

`is_id` **Deprecated** Whether the selector passed to `el` is an HTML id, set to `FALSE` to use other selectors, e.g.: `.class`.

`on_highlighted` A JavaScript function to run when the step is highlighted, generally a callback function. This is effectively a string that is evaluated JavaScript-side.

`on_highlight_started` A JavaScript function to run when the step is just aobut to be highlighted, generally a callback function. This is effectively a string that is evaluated JavaScript-side.

`on_next` A JavaScript function to run when the next button is clicked (or its event triggered), generally a callback function. This is effectively a string that is evaluated JavaScript-side.

Details: Add a step.

Method `init()`**:**

Usage:

```
Cicerone$init(session = NULL, run_once = FALSE)
```

Arguments:

`session` A valid Shiny session if `NULL` the function attempts to get the session with `shiny::getDefaultReactiveDomai`

`run_once` Whether to only run the guide once. If `TRUE` any subsequent calls of the method will not run the guide.

Details: Initialise Cicerone.

Method `reset()`**:**

Usage:

```
Cicerone$reset(session = NULL)
```

Arguments:

`session` A valid Shiny session if `NULL` the function attempts to get the session with `shiny::getDefaultReactiveDomai`

Details: Reset Cicerone.

Method `start()`**:**

Usage:

```
Cicerone$start(step = 1, session = NULL)
```

Arguments:

`step` The step index at which to start.

`session` A valid Shiny session if `NULL` the function attempts to get the session with `shiny::getDefaultReactiveDomai`

Details: Start Cicerone.

Method `move_forward()`**:**

Usage:

```
Cicerone$move_forward(session = NULL)
```

Arguments:

`session` A valid Shiny session if `NULL` the function attempts to get the session with `shiny::getDefaultReactiveDomai`

Details: Move Cicerone one step.

Method `move_backward()`**:**

Usage:

```
Cicerone$move_backward(session = NULL)
```

Arguments:

`session` A valid Shiny session if `NULL` the function attempts to get the session with `shiny::getDefaultReactiveDomai`

Details: Move Cicerone one step backward.

Method `highlight()`**:**

Usage:

```
Cicerone$highlight(el, session = NULL)
```

Arguments:

`el` Id of element to highlight

`session` A valid Shiny session if `NULL` the function attempts to get the session with `shiny::getDefaultReactiveDomai`

Details: Highlight a specific step.

Method `get_highlighted_el()`**:**

Usage:

```
Cicerone$get_highlighted_el(session = NULL)
```

Arguments:

`session` A valid Shiny session if `NULL` the function attempts to get the session with `shiny::getDefaultReactiveDomai`

Details: Retrieve the id of the currently highlighted element.

Method `get_previous_el()`**:**

Usage:

```
Cicerone$get_previous_el(session = NULL)
```

Arguments:

`session` A valid Shiny session if `NULL` the function attempts to get the session with `shiny::getDefaultReactiveDomai`

Details: Retrieve the id of the previously highlighted element.

Method `has_next_step()`**:**

Usage:

```
Cicerone$has_next_step(session = NULL)
```

Arguments:

`session` A valid Shiny session if `NULL` the function attempts to get the session with `shiny::getDefaultReactiveDomai`

Details: Retrieve whether there is a next step.

Method `get_next()`**:**

Usage:

```
Cicerone$get_next(session = NULL)
```

Arguments:

`session` A valid Shiny session if `NULL` the function attempts to get the session with `shiny::getDefaultReactiveDomai`

Details: Retrieve data that was fired when the user hit the "next" button.

Method `get_previous()`**:**

Usage:

```
Cicerone$get_previous(session = NULL)
```

Arguments:

`session` A valid Shiny session if `NULL` the function attempts to get the session with `shiny::getDefaultReactiveDomai`

Details: Retrieve data that was fired when the user hit the "previous" button.

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

Usage:

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

Arguments:

`deep` Whether to make a deep clone.

highlight *Highlight & Initialise*

Description

Initialise and highlight an element.

Usage

```
highlight(
  el,
  id,
  title = NULL,
  description = NULL,
  position = NULL,
  class = NULL,
  show_btns = NULL,
  close_btn_text = NULL,
  next_btn_text = NULL,
  prev_btn_text = NULL,
  session = NULL
)

initialise(
  id,
  animate = TRUE,
  opacity = 0.75,
  padding = 10,
  allow_close = TRUE,
  overlay_click_next = FALSE,
  done_btn_text = "Done",
  close_btn_text = "Close",
  stage_background = "#ffffff",
  next_btn_text = "Next",
  prev_btn_text = "Previous",
  show_btns = TRUE,
  keyboard_control = TRUE,
  session = NULL
)
```

Arguments

el	Id of element to be highlighted.
id	Unique identifier of cicerone.
title	Title on the popover.
description	Body of the popover.

position	Where to position the popover. See positions section.
class	className to wrap this specific step popover in addition to the general className in Driver options.
show_btns	Do not show control buttons in footer.
close_btn_text	Text on the close button for this step.
next_btn_text	Next button text for this step.
prev_btn_text	Previous button text for this step.
session	A valid Shiny session if NULL the function attempts to get the session with shiny::getDefaultReactiveDomain().
animate	Whether to animate or not.
opacity	Background opacity (0 means only popovers and without overlay).
padding	Distance of element from around the edges.
allow_close	Whether the click on overlay should close or not.
overlay_click_next	Whether the click on overlay should move next.
done_btn_text	Text on the final button.
stage_background	Background color for the staged behind highlighted element.
keyboard_control	Allow controlling through keyboard (escape to close, arrow keys to move).

Position

- left
- right
- left-center
- left-bottom
- top
- top-center
- top-right
- right
- right-center
- right-bottom
- bottom
- bottom-center
- mid-center

use_cicerone *Dependencies*

Description

Include cicerone dependencies in your Shiny UI.

Usage

```
use_cicerone()
```

Examples

```
library(shiny)

ui <- fluidPage(
 use_cicerone()
)

server <- function(input, output){}

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

Index