

Package ‘cicerone’

August 4, 2020

Title Provide Tours of 'Shiny' Applications

Version 1.0.3

Date 2020-08-03

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

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports R6, shiny, assertthat

RoxygenNote 7.1.1.9000

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

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

NeedsCompilation no

Author John Coene [aut, cre] (<<https://orcid.org/0000-0002-6637-4107>>)

Maintainer John Coene <jcoenep@gmail.com>

Repository CRAN

Date/Publication 2020-08-04 14:10:08 UTC

R topics documented:

| | |
|------------------------|----------|
| Cicerone | 2 |
| highlight | 6 |
| use_cicerone | 8 |
| Index | 9 |

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  
)
```

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.

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 = TRUE
  )

```

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

Details: Add a step.

Method `init()`:*Usage:*

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

Arguments:

`session` A valid Shiny session if `NULL` the function attempts to get the session with [shiny::getDefaultReactiveDomain](#)

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::getDefaultReactiveDomain](#)

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::getDefaultReactiveDomain](#)

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::getDefaultReactiveDomain`

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::getDefaultReactiveDomain`

Details: Move Cicerone one step backward.

Method `highlight()`:

Usage:

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

Arguments:

`el` Id of element to highlight

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

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

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::getDefaultReactiveDomain`

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::getDefaultReactiveDomain`

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::getDefaultReactiveDomain`

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::getDefaultReactiveDomain`

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::getDefaultReactiveDomain`

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

| | |
|---------------------------------|--|
| <code>el</code> | Id of element to be highlighted. |
| <code>id</code> | Unique identifier of cicerone. |
| <code>title</code> | Title on the popover. |
| <code>description</code> | Body of the popover. |
| <code>position</code> | Where to position the popover. See positions section. |
| <code>class</code> | <code>className</code> to wrap this specific step popover in addition to the general class-Name in Driver options. |
| <code>show_btns</code> | Do not show control buttons in footer. |
| <code>close_btn_text</code> | Text on the close button for this step. |
| <code>next_btn_text</code> | Next button text for this step. |
| <code>prev_btn_text</code> | Previous button text for this step. |
| <code>session</code> | A valid Shiny session if NULL the function attempts to get the session with <code>shiny::getDefaultReactiveDomain()</code> . |
| <code>animate</code> | Whether to animate or not. |
| <code>opacity</code> | Background opacity (0 means only popovers and without overlay). |
| <code>padding</code> | Distance of element from around the edges. |
| <code>allow_close</code> | Whether the click on overlay should close or not. |
| <code>overlay_click_next</code> | Whether the click on overlay should move next. |
| <code>done_btn_text</code> | 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

Cicerone, [2](#)

highlight, [6](#)

initialise (highlight), [6](#)

shiny::getDefaultReactiveDomain(), [4-7](#)

use_cicerone, [8](#)