Package ‘rintrojs’
January 11, 2024

Type Package
Title Wrapper for the 'Intro.js' Library
Version 0.3.4
Description A wrapper for the 'Intro.js' library (For more info: <https://introjs.com/>).
          This package makes it easy to include step-by-step introductions, and clickable hints in a 'Shiny' application. It supports both static introductions in the UI, and programmatic introductions from the server-side.
License AGPL-3
Imports shiny, jsonlite
Depends R (>= 3.0.0)
RoxygenNote 7.2.3
BugReports https://github.com/carlganz/rintrojs/issues
Suggests testthat, covr
Config/potools/style base
NeedsCompilation no
Author Carl Ganz [aut, cre] (<https://orcid.org/0000-0002-2973-3545>), Afshin Mehrabani [ctb, cph] (intro.js in javascript/introjs)
Maintainer Carl Ganz <carlganz@gmail.com>
Repository CRAN
Date/Publication 2024-01-11 21:30:02 UTC

R topics documented:

introBox ................................................................. 2
introjs ................................................................. 3
introjsUI ............................................................... 4
readCallback ......................................................... 5

Index 7
introBox

Generate intro elements in UI

Description

Wrap introBox around elements you want to include in introduction. Use data.step to order the boxes and data.intro to specify the comment in the introduction.

Usage

introBox(
    ..., 
    data.step, 
    data.intro, 
    data.hint, 
    data.position = c("bottom", "auto", "top", "left", "right", "bottom", 
                      "bottom-left_aligned", "bottom-middle-aligned", "bottom-right-aligned", "auto")
)

Arguments

... Elements in introduction element
data.step a number indicating its spot in the order in the intro
data.intro text for introduction
data.hint text for clickable hints
data.position position of intro

See Also

introjsUI() introjs()

Examples

## Not run:
library(rintrojs)
library(shiny)
ui <- shinyUI(fluidPage(
    introjsUI(), # must include in UI
    mainPanel(
        introBox(
            tableOutput("mtcars"),
            data.step = 1,
            data.intro = "This is the table"
        ),
        introBox(
            actionButton("btn","Intro"),
            data.step = 2,
data.intro = "This is the button"
})
})
server <- shinyServer(function(input, output, session) {
  output$mtcars <- renderTable({
    head(mtcars)
  })
  observeEvent(input$btn,
    introjs(session))
})
# Run the application
shinyApp(ui = ui, server = server)

## End(Not run)

---

### introjs

**Initiate intro.js**

**Description**

Initiates an introduction via the intro.js library

**Usage**

```r
introjs(session, options = list(), events = list())
```

```r
hintjs(session, options = list(), events = list())
```

**Arguments**

- **session**: the Shiny session object (from the server function of the Shiny app)
- **options**: List of options to be passed to intro.js
- **events**: List of text that are the body of a Javascript function. Must wrap in `I()`

**Note**

For documentation on intro.js options and events, see [https://introjs.com/docs/](https://introjs.com/docs/).

**See Also**

`introjsUI()` `introBox()`
Examples

```r
## Not run:
library(introjs)
library(shiny)
ui <- shinyUI(fluidPage(
  introjsUI(), # must include in UI
  mainPanel(
    introBox(
      tableOutput("mtcars"),
      data.step = 1,
      data.intro = "This is the table"
    ),
    introBox(
      actionButton("btn","Intro"),
      data.step = 2,
      data.intro = "This is the button",
      data.hint = "Here is clue"
    )
  )
))
server <- shinyServer(function(input, output, session) {
  hintjs(session, options = list("hintButtonLabel"="That was a hint"))
  output$mtcars <- renderTable({
    head(mtcars)
  })
  observeEvent(input$btn,
    introjs(session, options = list("nextLabel"="Onwards and Upwards"),
      events = list("oncomplete"=I('alert("It is over")')))})
})
# Run the application
shinyApp(ui = ui, server = server)
## End(Not run)
```

---

introjsUI

*Set up Shiny app to use intro.js*

Description

This function must be called from a Shiny app’s UI in order to use the package.

Usage

`introjsUI(includeOnly = FALSE, cdn = FALSE, version = "3.2.1")`

Arguments

- `includeOnly`: Only include intro.js files. For users who will write their own javascript
**readCallback**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cdn</td>
<td>Indicate whether to include intro.js files from CDN</td>
</tr>
<tr>
<td>version</td>
<td>Specify intro.js version to use from cdn</td>
</tr>
</tbody>
</table>

**Examples**

```r
## Not run:
library(rintrojs)
library(shiny)

shinyApp(
  ui = fluidPage(
    introjsUI(), # must include in UI
    actionButton("btn", "Click me")
  ),
  server = function(input, output, session) {
    observeEvent(input$btn, {
      intro <- data.frame(element="#btn",
        intro="In Codd we trust")
      introjs(session, options = list(steps= intro))
    })
  }
)
## End(Not run)
```

**readCallback**  
*Read a JS callback function into rintrojs*

**Description**

Reads a JS callback function into rintrojs

**Usage**

```r
readCallback(funname = c("switchTabs"))
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>funname</td>
<td>The name of the function you want to use. Options include:</td>
</tr>
</tbody>
</table>

**switchTabs** This function is intended to be passed to IntroJs’s *onBeforeChange* method. It will switch the currently active tab in your Shiny app to be the one containing the next element in your tour (this function is called by IntroJs right before it moves to the next element). Try running `shiny::runApp(system.file("examples/switchTabs.R", package = "rintrojs"))` to see an example.

**Value**

A string containing the body of a callback function
Examples

```r
## Not run:
introjs(session, events = list(onbeforechange = readCallback("switchTabs")))

## End(Not run)
```
Index

hintjs (introjs), 3
I(), 3
introBox, 2
introBox(), 3
introjs, 3
introjs(), 2
introjsUI, 4
introjsUI(), 2, 3

readCallback, 5