Package: tippy (via r-universe)

September 4, 2024

Title Add Tooltips to 'R markdown' Documents or 'Shiny' Apps

Version 1.0.0

Description 'Htmlwidget' of 'Tippyjs' to add tooltips to 'Shiny' apps
and 'R markdown' documents.

Depends R (>= 3.4.0)

License MIT + file LICENSE

Encoding UTF-8

Imports htmlwidgets, htmltools, shiny, jsonlite

RoxygenNote 7.1.2

URL `https://tippy.john-coene.com/`

BugReports `https://github.com/JohnCoene/tippy/issues`

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

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

RemoteRef HEAD

RemoteSha 0d89b2a5cf67c7ba2868a4777125326ef462ec76

Contents

deprecated *Use tippy*

Description

__Deprecated.__ Dynamically use tippy.

Usage

```
use_tippy()

tippy_class(class, ...)

with_tippy(element, tooltip, ...)
```

Arguments

`class`	Class of elements to apply tooltip to.
`...`	Any option from the official documentation.
`element`	Shiny, or htmltools element, or character string.
`tooltip`	Content of tooltip.

Functions

- `use_tippy` Includes tippy.js in header.
- `call_tippy` call `tippy` on specific target element(s), place after elements to be targeted.

See Also

official documentation

Examples

```
library(shiny)

if(FALSE){
shinyApp(
  ui = fluidPage(
    p("Some text", class = "tooltip"),
    p("Some text", class = "tooltip"),
    p("Some text", class = "tooltip"),
    p("Some text", class = "tooltip"),
    p("Some text", class = "tooltip"),
    p("Some text", class = "tooltip"),
    tippy("tooltip", content = "Hi!") # all elements with class
 ),
 server = function(input, output) {}
)
}
```

methods *Methods*

Description

Call tippy methods on tooltips.

Usage

```
tippy_disable(
  selectors = NULL,
  .session = shiny::getDefaultReactiveDomain(),
  .is_tag = FALSE
)

tippy_enable(
  selectors = NULL,
  .session = shiny::getDefaultReactiveDomain(),
  .is_tag = FALSE
)

tippy_destroy(
  selectors = NULL,
  .session = shiny::getDefaultReactiveDomain(),
  .is_tag = FALSE
)

tippy_show(
  selectors = NULL,
  .session = shiny::getDefaultReactiveDomain(),
  .is_tag = FALSE
)

tippy_hide(
  selectors = NULL,
  .session = shiny::getDefaultReactiveDomain(),
  .is_tag = FALSE
)
```

Arguments

`selectors`	Selectors of tooltips to apply the method. If ‘NULL‘ then the method is applied to _all tooltips._
`.session`	A valid shiny session.
`.is_tag`	Whether ‘selectors‘ are tags, e.g.: ‘<p>‘.

Examples

```
library(shiny)
library(tippy)

ui <- fluidPage(
  useTippy(),
  h5("plot tooltip", class = "tip"),
  p("Another tooltip", class = "tip"),
  tippyThis(
    ".tip",
    "This is indeed a tooltip"
  ),
  actionButton("disable", "Disable"),
  actionButton("enable", "Enable")
)

server <- function(input, output) {
  observeEvent(input$disable, {
    tippy_disable()
  });

  observeEvent(input$enable, {
    tippy_enable()
  })
}

if(FALSE) {
 shinyApp(ui, server)
}
```

---

tippy *Add a Tooltip to an Element*

Description

Add tooltips to an element.

Usage

```
tippy(element, content, ...)
```

Arguments

element	Shiny, or htmltools element, or character string.
content	Content of the tooltip.
...	Any other options from the official documentation.

See Also

official documentation

Examples

```
if(FALSE){
 tippy("Hover me!", content = "Hi, I'm the tooltip!")
 tippy(h3("Hello"), content = "World")
}
```

`tippyDefault` *Default*

Description

Set tippy defaults that will be shared by all other tooltips.

Usage

```
tippyDefault(...)
```

Arguments

`...`	Any other options from the official documentation.

`tippyTheme` *Theme*

Description

Convenience function to create a theme.

Usage

```
tippyTheme(
  name,
  box_bg = "black",
  box_color = "white",
  box_font_size = "12px",
  arrow_color = box_bg,
  content_bg = box_bg,
  content_color = box_color,
  content_font_size = box_font_size
)
```

Arguments

`name`	Name of theme.
`box_bg`, `box_color`, `content_bg`, `content_color`	Background and text color respectively.
`box_font_size`, `content_font_size`	Size of the font.
`arrow_color`	Color of the arrow (if enabled).

Examples

```
library(shiny)

ui <- fluidPage(
  useTippy(),
  tippyTheme("myTheme", box_bg = "darkred"),
  h5("plot tooltip", id = "plotTip"),
  tippyThis(
    "plotTip",
    "Hello world",
    theme = "myTheme"
  )
)

server <- function(input, output) {}

if(FALSE){
  shinyApp(ui, server)
}
```

`tippy_global_theme` *Theme*

Description

Set a global theme. Every tooltip will subsequently use this theme.

Usage

```
tippy_global_theme(name)
```

Arguments

`name`	Name of the theme to set.

tippy_plot *Use a plot as tooltip*

Description

Use a plot as tooltip

Usage

```
tippy_plot(
  selector,
  plotId,
  ...,
  .width = 200,
  .height = 200,
  .hidePlot = FALSE,
  .session = shiny::getDefaultReactiveDomain()
)
```

Arguments

selector	A CSS selector, e.g.: '#id' or '.class'. If it is a bare selector (e.g.: 'sth') then it is assumed to be an id and processed as '#sth'. Set '.is_tag' to 'TRUE' to disable that.
plotId	Id of plot to use.
...	Any other options from the official documentation.
.width, .height	Dimensions of plot.
.hidePlot	Whether to hide the original plot.
.session	A valid shiny session.

Examples

```
library(shiny)

ui <- fluidPage(
  useTippy(),
  plotOutput("plot"),
  h5("plot tooltip", id = "plotTip")
)

server <- function(input, output) {
  output$plot <- renderPlot({
    on.exit({
      tippy_plot(
        "plotTip",
        "plot",
        theme = "light",
        .hidePlot = TRUE,
     .width = 600,
     .height = 200,
        maxWidth = 700
      )
    })
    plot(cars)
  })
}

if(FALSE){
  shinyApp(ui, server)
}
```

---

tippy_this *Bind Tooltip to a Selector*

Description

Binds a tooltip to a valid CSS selector.

Usage

```
tippy_this(selector, content = NULL, ...)

tippyThis(selector, content = NULL, ..., .is_tag = FALSE)
```

Arguments

`selector`	A CSS selector, e.g.: ‘#id‘ or ‘.class‘. If it is a bare selector (e.g.: ‘sth‘) then it is assumed to be an id and processed as ‘#sth‘. Set ‘.is_tag‘ to ‘TRUE‘ to disable that.
`content`	Content of the tooltip.
`...`	Any other options from the official documentation.
`.is_tag`	Whether the selector is a tag, e.g.: ‘<p>‘.

Examples

```
library(shiny)

ui <- fluidPage(
 tippy(h3("Hello"), "World"),
 h4("World", id = "theId"),
 tippyThis("theId", "A tooltip")
)

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

```
if(FALSE){
 shinyApp(ui, server)
}
```

useTippy *Dependencies*

Description

Include dependencies, place anywhere in the shiny UI.

Usage

```
useTippy()
```

Index