Package ‘mwshiny’

October 13, 2022

Type Package

Title 'Shiny' for Multiple Windows

Version 2.1.0

Date 2020-06-05

Maintainer Hannah De los Santos <hdelossantos653@gmail.com>

Description A simple function, mwsApp(), that runs a 'shiny' app spanning multiple, connected windows. This uses all standard 'shiny' conventions, and depends only on the 'shiny' package.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Depends shiny (>= 1.2.0)

Imports htmltools (>= 0.3.6)

Suggests knitr, rmarkdown, ggplot2 (>= 3.1.0), visNetwork (>= 2.0.5), htmlwidgets (>= 1.3), datasets

VignetteBuilder knitr

RoxygenNote 6.1.1

NeedsCompilation no

Author Hannah De los Santos [aut, cre],
John Erickson [aut],
Joe Cheng [ctb],
Nicholas Thomson [ctb],
Kristin Bennett [aut]

Repository CRAN

Date/Publication 2020-06-05 22:00:02 UTC

R topics documented:

mwsApp *Runs Shiny app in multiple specified windows.*

Description

Runs Shiny app in multiple specified windows.

Usage

```
mwsApp(ui_win = list(), serv_calc = list(), serv_out = list())
```

Arguments

ui_win	named list of shiny UI pages. The name of each entry in the UI page list corresponds to its window title. No windows can be named 'WindowSelector', titles must be uniquely named, and titles cannot have spaces.
serv_calc	a named list of functions that calculate variables derived from user input, to be used in rendering output. Each function is of the form function(calc, session), where calc is a named list containing the traditional Shiny input and user-created reactive values, and session is the traditional Shiny server session value. All calculated variables that are needed to render output should be added, named, to the calc list. When using reactive functions such as observeEvent(), each should be contained in a separate function, and variables dependent on these reactions should be added to calc. Note that these functions follow all Shiny conventions (reactive values must be accessed in a reactive context, etc.).
serv_out	a named list of functions that render output. Each function is of the form function(calc, session), where calc is a named list containing the traditional Shiny input and reactive values that have calculated values derived from input, and session is the traditional Shiny server session value. It returns the results of a Shiny render function. The name of each function corresponds to its output label. Note that these functions follow all Shiny conventions (reactive values must be accessed in a reactive context, etc.).

Value

Shiny app object (i.e., it runs the app)

Examples

```
if(interactive()){
# Run a simple 2-window app, initially bringing up the window selector window:
ui_win <- list()
ui_win[["clickinput"]] <- fluidPage(numericInput(inputId = "click", label = "a", value = 1))
ui_win[["clickoutput"]] <- fluidPage(plotOutput("clickplot"))
serv_out <- list()
serv_out[["clickplot"]] <- function(calc, session){
  renderPlot({
```

```
        plot(1:calc$click,1:calc$click)
    })
}
mwsApp(ui_win, list(), serv_out)
}
```

Index