Package ‘shiny.react’

December 25, 2022

Title Tools for Using React in Shiny
Version 0.3.0
Description A toolbox for defining React component wrappers which can be used seamlessly in Shiny apps.
License LGPL (>= 3)
Encoding UTF-8
RoxygenNote 7.2.3
VignetteBuilder knitr
Imports glue, htmltools, jsonlite, logger, purrr, rlang, shiny
Suggests knitr, leaflet, lintr (>= 3.0.0), mockery (>= 0.4.2), rcmdcheck, rmarkdown, styler, testthat
NeedsCompilation no
Author Kamil Żyła [aut, cre], Jakub Sobolewski [aut], Marek Rogala [aut], Appsilon Sp. z o.o. [cph]
Maintainer Kamil Żyła <opensource+kamil@appsilon.com>
Repository CRAN
Date/Publication 2022-12-25 07:20:07 UTC

R topics documented:

asProps ................................................................. 2
enableReactDebugMode ........................................... 2
JS ................................................................. 3
ReactContext .................................................. 3
reactDependency ............................................ 4
reactElement .................................................. 4
**asProps**

*Parse arguments as props*

**Description**

Converts arguments to a list which can be passed as the props argument to `reactElement()`. Unnamed arguments become children and named arguments become attributes for the element.

**Usage**

```r
asProps(...) 
```

**Arguments**

...  

Arguments to prepare for passing as props to a 'React' component

**Value**

A list of the arguments structured suitably for `reactElement()`.

**See Also**

`reactElement`

---

**enableReactDebugMode**

*Enable 'React' debug mode*

**Description**

Sets the `shiny.react_DEBUG` option to TRUE. In debug mode, 'shiny.react' will load a dev version of 'React', which is useful for debugging. It will also set the logging level to DEBUG. Call this function before running the app to enable the debugging mode.

**Usage**

```r
enableReactDebugMode() 
```

**Value**

Nothing. This function is called for its side effects.
**JS**  
*Mark character strings as literal JavaScript code*

**Description**
Copied verbatim from the htmlwidgets package to avoid adding a dependency just for this single function.

**Usage**

```r
JS(...)  
```

**Arguments**

```r  
...  
```
Character vectors as the JavaScript source code (all arguments will be pasted into one character string).

**Value**

The input character vector marked with a special class.

---

**ReactContext**  
*React context*

**Description**
Render children with React.

**Usage**

```r
ReactContext(...)  
```

**Arguments**

```r  
...  
```
Children to render.

**Examples**

```r  
if (interactive()) shinyApp(  
    ui = shiny.react:::ReactContext(  
      "This text is rendered by React"  
    ),  
    server = function(input, output) {}  
)  
```
**reactDependency**  
'React' library dependency

**Description**

'React' library dependency

**Usage**

```r
code
reactDependency(useCdn = FALSE)
```

**Arguments**

- **useCdn**: If TRUE, 'React' will be loaded from a CDN instead of being served locally.

**Value**

An `htmlDependency` object which can be used to attach the 'React' library.

**reactElement**  
Create a 'React' element

**Description**

Creates a `shiny.tag` which can be rendered just like other 'Shiny' tags as well as passed in props to other 'React' elements. Typically returned from a wrapper ("component") function, which parses its arguments with `asProps()` and fills in the other arguments.

**Usage**

```r
code
reactElement(module, name, props, deps = NULL)
```

**Arguments**

- **module**: JavaScript module to import the component from.
- **name**: Name of the component.
- **props**: Props to pass to the component.
- **deps**: HTML dependencies to attach.

**Value**

A `shiny.tag` object representing the 'React' element.

**See Also**

`asProps`
**Examples**

```r
Component <- function(...) reactElement(
  module = "@/module", name = "Component", props = asProps(...)
)
```

---

**Description**

Creates a 'Shiny' output which can be used analogously to `shiny::uiOutput()` but preserves 'React' state on re-renders.

**Usage**

```r
reactOutput(outputId)
```

**Arguments**

- `outputId`  
  Id that can be used to render React on the server

**Value**

A `shiny.tag` object which can be placed in the UI.

**See Also**

- `renderReact`

**Examples**

```r
if (interactive()) {
  colors <- list("Gold", "Lavender", "Salmon")

  shinyApp(
    ui = bootstrapPage(
      reactOutput("ui"),
      selectInput("color", label = "Background color", choices = colors)
    ),
    server = function(input, output) {
      output$ui <- renderReact(
        shiny.react:::Box(
          style = list(backgroundColor = input$color),
          shiny.react:::Pinger()
        )
      )
    }
  )
}
```
renderReact  
Render 'React'

Description
Renders HTML and/or 'React' in outputs created with `reactOutput()` (analogously to `shiny::renderUI()`).

Usage
`renderReact(expr, env = parent.frame(), quoted = FALSE)`

Arguments
- `expr`  Expression returning the HTML / 'React' to render.
- `env`  Environment in which to evaluate `expr`.
- `quoted`  Is `expr` a quoted expression?

Value
A function which can be assigned to an output in a Shiny server function.

See Also
- `reactOutput`

setInput  
Set input

Description
Creates a handler which can be used for `onChange` and similar props of 'React' components to set the value of a 'Shiny' input to one of the arguments passed to the handler.

Usage
`setInput(inputId, argIdx = 1)`
shinyReactDependency

**Arguments**

- **inputId**  
  'Shiny' input ID to set the value on.

- **argIdx**  
  Index of the argument to use as value.

**Value**

A ReactData object which can be passed as a prop to 'React' components.

---

**triggerEvent**  
*Trigger event*

**Description**

Creates a handler which can be used for `onClick` and similar props of 'React' components to trigger an event in 'Shiny'.

**Usage**

```javascript
triggerEvent(inputId)
```

**Arguments**

- **inputId**  
  'Shiny' input ID to trigger the event on.

**Value**

A ReactData object which can be passed as a prop to 'React' components.
updateReactInput  

Update 'React' input

Description

Updates inputs created with the help of InputAdapter function (part of the JavaScript interface). Analogous to shiny::updateX() family of functions, but generic.

Usage

updateReactInput(session = shiny::getDefaultReactiveDomain(), inputId, ...)

Arguments

session  
Session object passed to function given to shinyServer.

inputId  
Id of the input object.

...  
Props to modify.

Details

If you’re creating a wrapper package for a 'React' library, you’ll probably want to provide a dedicated update function for each input to imitate 'Shiny' interface.

Value

Nothing. This function is called for its side effects.
Index

asProps, 2, 4
enableReactDebugMode, 2
JS, 3
ReactContext, 3
reactDependency, 4
reactElement, 2, 4
reactOutput, 5, 6
renderReact, 5, 6
setInput, 6
shinyReactDependency, 7
triggerEvent, 7
updateReactInput, 8