

Package ‘shiny.react’

July 23, 2025

Title Tools for Using React in Shiny

Version 0.4.0

URL <https://appsilon.github.io/shiny.react/>,
<https://github.com/Appsilon/shiny.react>

Description

A toolbox for defining React component wrappers which can be used seamlessly in Shiny apps.

License LGPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.1

VignetteBuilder knitr

Imports glue, htmltools, jsonlite, logger, methods, purrr, rlang,
shiny, stringi

Suggests chromote, covr, knitr, leaflet, lintr (>= 3.0.0), rcmdcheck,
rmarkdown, shinytest2, styler, testthat, withr

NeedsCompilation no

Author Jakub Sobolewski [aut, cre],
Kamil Żyła [aut],
Marek Rogala [aut],
Appsilon Sp. z o.o. [cph]

Maintainer Jakub Sobolewski <opensource+jakub.sobolewski@appsilon.com>

Repository CRAN

Date/Publication 2024-05-20 12:30:02 UTC

Contents

asProps	2
enableReactDebugMode	2
JS	3
ReactContext	3
reactDependency	4

reactElement	4
reactOutput	5
renderReact	6
setInput	6
shinyReactDependency	8
triggerEvent	8
updateReactInput	9
Index	10

asProps	<i>Parse arguments as props</i>
---------	---------------------------------

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

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	<i>Enable 'React' debug mode</i>
----------------------	----------------------------------

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

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

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

Arguments

... 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

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

Arguments

... Children to render.

Examples

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

reactDependency	<i>'React' library dependency</i>
-----------------	-----------------------------------

Description

'React' library dependency

Usage

```
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	<i>Create a 'React' element</i>
--------------	---------------------------------

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

```
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

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

reactOutput

'React' output

Description

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

Usage

```
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

```
# This example uses some unexported test components. The components are not exported,
# as shiny.react is designed to only provide the machinery for building React-based packages.
# See shiny.fluent for a large number of examples.
```

```
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	<i>Render 'React'</i>
-------------	-----------------------

Description

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

Usage

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

Arguments

<code>expr</code>	Expression returning the HTML / 'React' to render.
<code>env</code>	Environment in which to evaluate <code>expr</code> .
<code>quoted</code>	Is <code>expr</code> a quoted expression?

Value

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

See Also

[reactOutput](#)

setInput	<i>Set input</i>
----------	------------------

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, jsAccessor)

## S4 method for signature 'character,missing'
setInput(inputId)

## S4 method for signature 'character,numeric'
setInput(inputId, jsAccessor)

## S4 method for signature 'character,character'
setInput(inputId, jsAccessor)
```

Arguments

inputId	'Shiny' input ID to set the value on.
jsAccessor	Index (numeric 0-based index) or accessor (JavaScript string) of the argument to use as value.

Details

The argument jsAccessor can be empty (assumes jsAccessor = 0) or take one of the following types:

- A valid JavaScript accessor string to be applied to the object (example: jsAccessor = "[0].target.checked").
- A valid JavaScript 0-based index.

As an example, calling setInput("some_index", 1) is equivalent to setInput("some_index", "[1]")

Value

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

Methods (by class)

- setInput(inputId = character, jsAccessor = missing): Uses as index jsAccessor = 0
- setInput(inputId = character, jsAccessor = numeric): Gets the value via index (see examples).
- setInput(inputId = character, jsAccessor = character): Gets value via accessor (see examples).

Examples

```
# Same as `setInput("some_id", 0)`
setInput("some_id")

# Equivalent to `(...args) => Shiny.setInputValue('some_id', args[1])` in JS.
setInput("some_id", 1)
```

```
# Same as `setInput("some_id", 1)`.  
setInput("some_id", "[1]")  
  
# Equivalent to `(...args) => Shiny.setInputValue('some_id', args[0].target.value)` in JS.  
setInput("some_id", "[0].target.value")
```

shinyReactDependency *'shiny.react' JavaScript dependency*

Description

'shiny.react' JavaScript dependency

Usage

shinyReactDependency()

Value

An htmlDependency object which can be used attach the JavaScript code required by 'shiny.react'.

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

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	<i>Update 'React' input</i>
------------------	-----------------------------

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

<code>session</code>	Session object passed to function given to <code>shinyServer</code> .
<code>inputId</code>	Id of the input object.
<code>...</code>	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](#)

`setInput`, `character`, `character-method`
(`setInput`), [6](#)

`setInput`, `character`, `missing-method`
(`setInput`), [6](#)

`setInput`, `character`, `numeric-method`
(`setInput`), [6](#)

`shinyReactDependency`, [8](#)

`triggerEvent`, [8](#)

`updateReactInput`, [9](#)