Package ‘reactable.extras’

October 29, 2024

Type Package

Title Extra Features for 'reactable' Package

Version 0.2.1

Description Enhanced functionality for 'reactable' in 'shiny' applications, offering interactive and dynamic data table capabilities with ease.
With 'reactable.extras', easily integrate a range of functions and components to enrich your 'shiny' apps and facilitate user-friendly data exploration.

License LGPL-3

URL https://appsilon.github.io/reactable.extras/,
https://github.com/Appsilon/reactable.extras

BugReports https://github.com/Appsilon/reactable.extras/issues

Depends R (>= 4.1.0)

Imports checkmate, dplyr, htmltools, purrr (>= 1.0.0), reactable (>= 0.4.0), rjson, rlang, shiny

Suggests covr, lintr, mockery, rcmdcheck, shinytest2, spelling, testthat (>= 3.0.0)

Config/testthat/edition 3

Encoding UTF-8

Language en-US

RoxygenNote 7.2.3

NeedsCompilation no

Author Recle Vibal [aut, cre],
Ivan Hrychaniuk [aut],
Andres Quintero [aut],
Pedro Silva [aut],
Eduardo Almeida [ctb],
Appsilon Sp. z o.o. [cph]

Maintainer Recle Vibal <opensource+recle@appsilon.com>

Repository CRAN

Date/Publication 2024-10-29 14:50:02 UTC

Contents

button_extra *Button input for reactable column cell*

Description

Button input for reactable column cell

Usage

```
button_extra(id, key = NULL, ...)
```

Arguments

`id`	id of the button input
`key`	alternative unique id for server side processing
`...`	parameters of button, only `class` is supported for now

Value

Custom JS button renderer for reactable

Examples

```
reactable::colDef(cell = button_extra("click", class = "table-button"))
```

`checkbox_extra` *Checkbox input for reactable column cell*

Description

Checkbox input for reactable column cell

Usage

```
checkbox_extra(id, key = NULL, ...)
```

Arguments

`id`	id of the checkbox input
`key`	alternative unique id for server side processing
`...`	parameters of checkbox, only `class` is supported for now

Value

Custom JS checkbox renderer for reactable

Examples

```
reactable::colDef(cell = checkbox_extra("check", class = "table-check"))
```

`date_extra` *Date input for reactable column cell*

Description

Date input for reactable column cell

Usage

```
date_extra(id, key = NULL, ...)
```

Arguments

`id`	id of the date input
`key`	alternative unique id for server side processing
`...`	parameters of date input, only `class` is supported for now

Value

Custom JS date input renderer for reactable

Examples

```
reactable::colDef(cell = date_extra("date", class = "table-date"))
```

dropdown_extra *Select input for reactable column cell*

Description

Select input for reactable column cell

Usage

```
dropdown_extra(id, choices, key = NULL, ...)
```

Arguments

`id`	id of the select input
`choices`	vector of choices
`key`	alternative unique id for server side processing
`...`	parameters of date input, only `class` is supported for now

Value

Custom JS dropdown renderer for reactable

Examples

```
reactable::colDef(
  cell = dropdown_extra("dropdown",
  choices = letters[1:5],
  class = "table-dropdown"
  )
 )
```

reactable-extras-server

Create reactable UI with server-side processing

Description

Create reactable UI with server-side processing

Usage

```
reactable_extras_ui(id, width = "auto", height = "auto")

reactable_extras_server(id, data, total_pages = 4, sortable = TRUE, ...)
```

Arguments

`id`	element id
`width, height`	CSS unit (`"100%"`, `"400px"`, or `"auto"`), numeric for number of pixels
`data`	passed to `reactable::reactable()`
`total_pages`	number of pages
`sortable`	allow sorting by columns
`...`	other arguments to be passed to `reactable::reactable()`

Details

Arguments passed to `reactable::reactable()` must not contain `pagination` or `showPagination`. These are set to `FALSE`. Pagination will be handled on the server-side.

Value

`reactable_extras_ui()` returns a custom UI for a server-side processed reactable

Examples

```
if (interactive()) {
  library(shiny)
  library(reactable)
  library(reactable.extras)

  shinyApp(
    reactable_extras_ui("big_data"),
    function(input, output, server) {
      reactable_extras_server(
        "big_data",
        data = mtcars,
        columns = list(
          mpg = reactable::colDef(name = "Miles per Gallon"),
          cyl = reactable::colDef(name = "Cylinders"),
          disp = reactable::colDef(name = "Displacement")
        )
      )
    }
  )
}
```

`reactable_extras_dependency` *Reactable.extras JS and CSS dependencies*

Description

Reactable.extras JS and CSS dependencies

Usage

```
reactable_extras_dependency()
```

Value

HTML dependency object.

`text_extra` *Text input for reactable column cell*

Description

Text input for reactable column cell

Usage

```
text_extra(id, key = NULL, ...)
```

Arguments

`id`	id of the text input
`key`	alternative unique id for server side processing
`...`	parameters of text input, only `class` is supported for now

Value

Custom JS text input renderer for reactable

Examples

```
reactable::colDef(cell = text_extra("text", class = "table-text"))
```

tooltip_extra *Tool-tip for table headers*

Description

Tool-tip for table headers

Usage

```
tooltip_extra(content, theme = "light")
```

Arguments

content	The content to be displayed in the tool-tip
theme	The theme of the tool-tip, either "light", "light-border", "material" or "translucent"

Value

Custom JS tool-tip renderer for reactable

Examples

```
reactable::colDef(header = tooltip_extra("This is my tool-tip", theme = "material"))
```

Index