Package ‘shinyTree’

October 14, 2022

Type Package
Title jsTree Bindings for Shiny
Version 0.2.7
Date 2019-05-27
Description Exposes bindings to jsTree -- a JavaScript library that supports interactive trees -- to enable a rich, editable trees in Shiny.
License MIT + file LICENSE
Depends R (>= 2.15.1), methods
Imports shiny (>= 0.9.0), htmlwidgets, jsonlite, stringr, promises
Suggests testthat

BugReports https://github.com/trestletech/shinyTree/issues
RoxygenNote 6.1.1
Encoding UTF-8

NeedsCompilation no

Author Trestle Technology, LLC [aut],
Jeff Allen [aut],
Institut de Radioprotection et de Sûreté Nucléaire [cph],
Ivan Bozhanov [ctb, cph] (jsTree),
The Dojo Foundation [ctb, cph] (require.js),
jQuery Foundation, Inc. [ctb, cph],
Mike Schaffer [cre],
Timm Danker [ctb],
Michael Bell [ctb]

Maintainer Mike Schaffer <mschaff@gmail.com>

Repository CRAN

Date/Publication 2019-05-27 12:40:02 UTC
R topics documented:

- `get_selected` ........................................... 2
- `renderEmptyTree` ..................................... 2
- `renderTree` ............................................. 3
- `renderTreeAsync` ...................................... 3
- `shinyTree` ............................................... 4
- `updateTree` ............................................. 5

Index 6

**get_selected**  
*Get the selected nodes from a tree*

**Description**

Extract the nodes from the tree that are selected in a more convenient format. You can choose which format you prefer.

**Usage**

```r
get_selected(tree, format = c("names", "slices", "classid"))
```

**Arguments**

- `tree` The input$tree shinyTree you want to inspect.
- `format` In which format you want the output. Use `names` to get a simple list of the names (with attributes describing the node’s ancestry), or `slices` to get a list of lists, each of which is a slice of the list used to get down to the selected node.

**renderEmptyTree**  
*Render an empty ShinyTree*

**Description**

Renders a tree with no defined nodes.

**Usage**

```r
renderEmptyTree()
```

**See Also**

`shinyTree`
### renderTree

**Render a ShinyTree**

**Description**

Should return a list from the given expression which will be converted into a `shinyTree`.

**Usage**

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

**Arguments**

- `expr`: The expression to be evaluated which should produce a list.
- `env`: The environment in which `expr` should be evaluated.
- `quoted`: Is `expr` a quoted expression (with `quote()`)? This is useful if you want to save an expression in a variable.

**See Also**

- `shinyTree`

### renderTreeAsync

**Render an asynchronous ShinyTree**

**Description**

Should return a list from the given expression which will be converted into a `shinyTree`.

**Usage**

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

**Arguments**

- `expr`: The expression to be evaluated which should produce a list.
- `env`: The environment in which `expr` should be evaluated.
- `quoted`: Is `expr` a quoted expression (with `quote()`)? This is useful if you want to save an expression in a variable.

**See Also**

- `shinyTree`
shinyTree

Create a Shiny Tree

Description

This creates a spot in your Shiny UI for a shinyTree which can then be filled in using `renderTree`

Usage

```r
shinyTree(outputId, checkbox = FALSE, search = FALSE,
searchtime = 250, dragAndDrop = FALSE, types = NULL,
theme = "default", themeIcons = TRUE, themeDots = TRUE,
sort = FALSE, unique = FALSE, wholerow = FALSE, stripes = FALSE,
multiple = TRUE, animation = 200, contextmenu = FALSE)
```

Arguments

- `outputId` The ID associated with this element
- `checkbox` If TRUE, will enable checkboxes next to each node to make the selection of multiple nodes in the tree easier.
- `search` If TRUE, will enable search functionality in the tree by adding a search box above the produced tree. Alternatively, you can set the parameter to the ID of the text input you wish to use as the search field.
- `searchtime` Determines the reaction time of the search algorithm. Default is 250ms.
- `dragAndDrop` If TRUE, will allow the user to rearrange the nodes in the tree.
- `types` enables jstree types functionality when sent proper json (please see the types example)
- `theme` jsTree theme, one of `default`, `default-dark`, or `proton`.
- `themeIcons` If TRUE, will show theme icons for each item.
- `themeDots` If TRUE, will include level dots.
- `sort` If TRUE, will sort the nodes in alphabetical/numerical order.
- `unique` If TRUE, will ensure that no node name exists more than once.
- `wholerow` If TRUE, will highlight the whole selected row.
- `stripes` If TRUE, the tree background is striped.
- `multiple` If TRUE, multiple nodes can be selected.
- `animation` The open / close animation duration in milliseconds. Set this to FALSE to disable the animation (default is 200).
- `contextmenu` If TRUE, will enable a contextmenu to create/rename/delete/cut/copy/paste nodes.

See Also

`renderTree`
updateTree

---

updateTree  

*Update the tree with new data*

---

**Description**

Extract the nodes from the tree that are selected in a more convenient format. You can choose which format you prefer.

**Usage**

```r
updateTree(session, treeId, data = NULL)
```

**Arguments**

- `session`  
The current session variable.
- `treeId`  
The identifier for the shinyTree object
- `data`  
JSON data or nested list representing the new tree structure.
Index

get_selected, 2
renderEmptyTree, 2
renderTree, 3, 4
renderTreeAsync, 3
shinyTree, 2, 3, 4
updateTree, 5