

Package ‘jsTree’

December 13, 2020

Title Create Interactive Trees with the 'jQuery' 'jsTree' Plugin

Version 1.2

Date 2020-12-10

Maintainer Jonathan Sidi <yonicd@gmail.com>

Description Create and customize interactive trees using the 'jQuery' 'jsTree' <<https://www.jstree.com/>> plugin library and the 'htmlwidgets' package. These trees can be used directly from the R console, from 'RStudio', in Shiny apps and R Markdown documents.

Depends R (>= 2.10)

Imports htmlwidgets,jsonlite,htmltools,data.table

License MIT + file LICENSE

Encoding UTF-8

LazyData true

URL <https://github.com/yonicd/jsTree>

BugReports <https://github.com/yonicd/jsTree/issues>

RoxygenNote 7.1.1

Suggests testthat, covr,knitr,rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Jonathan Sidi [aut, cre],
Kohleth Chia [ctb]

Repository CRAN

Date/Publication 2020-12-13 00:10:06 UTC

R topics documented:

jsTree	2
jsTree-shiny	4
states	5
state_bird	5

jsTree

*Htmlwidget for the jsTree Javascript library***Description**

Htmlwidget for the jsTree Javascript library

Usage

```
jsTree(
  obj,
  sep = "/",
  sep_fixed = TRUE,
  core = NULL,
  tooltips = NULL,
  nodestate = NULL,
  ...,
  width = NULL,
  height = NULL,
  elementId = NULL,
  file = tempfile(pattern = "jstree-", fileext = ".html"),
  browse = TRUE
)
```

Arguments

<code>obj</code>	character, vector of directory tree
<code>sep</code>	character, separator for 'obj' which defines the hierarchy, Default: '/'.
<code>sep_fixed</code>	boolean, to treat the sep character(s) as fixed when seperating, Default: TRUE.
<code>core</code>	list, additional parameters to pass to core of jsTree, default: NULL
<code>tooltips</code>	character, named vector of tooltips for elements in the tree, Default: NULL
<code>nodestate</code>	boolean, vector the length of obj that initializes tree open to true values, Default: NULL
<code>...</code>	parameters that are passed to the vcs package (see details)
<code>width, height</code>	Must be a valid CSS unit (like '100%', '400px', 'auto') or a number, which will be coerced to a string and have 'px' appended.
<code>elementId</code>	The input slot that will be used to access the element.
<code>file</code>	character, html filename to save output to, Default: <code>tempfile(pattern = 'jstree-', fileext = '.html')</code> .
<code>browse</code>	whether to open a browser to view the html, Default: TRUE

Details

valid core objects can be found in the jsTree javascript library api [homepage](#).

All objects that are children of 'jstree.defaults.core' are valid inputs, except 'jstree.defaults.core.data' which is constructed internally by the R function call. The R list object is translated internally into a valid javascript object.

parameters in ... that can be passed on to the vcs package are:

remote_repo a character object that defines the remote user/repository

remote_branch character object that defines the branch of remote_repo (usually 'master')

vcs character object that defines for vcs which version control system to attach (github, bitbucket, svn)

preview.search character object that defines a search term to initialize to in the preview pane search-box

if **remote_repo** is given a preview pane of a selected file from the tree will appear to the right of the tree

preview.search is only relevant for vcs in (github,bitbucket) where file previewing is available

For more information on the vcs package go to <https://github.com/yonicd/vcs>

Examples

```
if(interactive()){

data(states)
data(state_bird)

#collapse columns to text (with sep "/")
nested_string <- apply(states,1,paste,collapse='/')
jsTree(nested_string)

#pass additional parameters to core
jsTree(nested_string,core=list(multiple=FALSE))

# Add tooltips to state names with the state bird
jsTree(nested_string,tooltips = state_bird)

#initialize tree with checked boxes for certain fields
nodestate1 <- states$variable=='Area'
jsTree(nested_string,nodestate=nodestate1)

nodestate2 <- states$variable=='Area'&grepl('^M',states$state.name)
jsTree(nested_string,nodestate=nodestate2)

nodestate3 <- states$variable %in% c('Murder') & states$value >= 10
nodestate4 <- states$variable %in% c('HS.Grad') & states$value <= 55
jsTree(nested_string,nodestate=nodestate3|nodestate4)

#change the order of the hierarchy
nested_string2 <- apply(states[,c(4,1,2,3,5)],1,paste,collapse='/')
```

```
jsTree(nested_string2)

#use jsTree to visualize folder structure

jsTree(list.files(full.names = TRUE,recursive = FALSE))

## Not run:
# This may be too long for example if running from ~.
jsTree(list.files(full.names = TRUE,recursive = TRUE))

## End(Not run)
}
```

jsTree-shiny

Shiny bindings for jsTree

Description

Output and render functions for using jsTree within Shiny applications and interactive Rmd documents.

Usage

```
jsTreeOutput(outputId, width = "100%", height = "400px")

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

Arguments

outputId	output variable to read from
width, height	Must be a valid CSS unit (like '100%', '400px', 'auto') or a number, which will be coerced to a string and have 'px' appended.
expr	An expression that generates a jsTree
env	The environment in which to evaluate expr.
quoted	Is expr a quoted expression (with quote())? This is useful if you want to save an expression in a variable.

states	<i>State data</i>
--------	-------------------

Description

state dataset melted into a data.frame

Usage

```
states
```

Format

A data frame with 400 rows and 5 variables:

```
state.region factor State Region  
state.division factor State Sub Region  
state.name character State Name  
variable factor Characteristic  
value double Characteristic Value
```

state_bird	<i>Character vector of state birds</i>
------------	----------------------------------------

Description

Character vector of state birds

Usage

```
state_bird
```

Format

A character vector of length 50

Source

<https://state.1keydata.com/state-birds.php>

Index

* **datasets**

state_bird, [5](#)

states, [5](#)

jsTree, [2](#)

jsTree-shiny, [4](#)

jsTreeOutput (jsTree-shiny), [4](#)

renderJsTree (jsTree-shiny), [4](#)

state_bird, [5](#)

states, [5](#)