# Package ‘dashPivottable’

August 12, 2020

**Title**  Interactive React-Based Pivot Tables for Dash

**Version**  0.0.2-1

**Description**  Pivot tables are useful for interactive presentation of summary statistics computed for data contained in another table. The 'dashPivottable' package wraps 'react-pivottable', making it easy to add drag-and-drop tables into your Dash for R applications.

**Depends**  R (>= 3.0.2)

**Imports**

**Suggests**  dash, dashHtmlComponents, dashTable, jsonlite

**License**  MIT + file LICENSE

**Copyright**  Plotly Technologies, Inc.

**URL**  https://github.com/plotly/dash-pivottable

**BugReports**  https://github.com/plotly/dash-pivottable/issues

**Encoding**  UTF-8

**LazyData**  true

**KeepSource**  true

**NeedsCompilation**  no

**Author**  Chris Parmer [aut],
           Nicolas Kruchten [aut],
           Xing Han Lu [trl],
           Ryan Patrick Kyle [cre] (<https://orcid.org/0000-0001-5829-9867>),
           Plotly Technologies, Inc. [cph]

**Maintainer**  Ryan Patrick Kyle <ryan@plotly.com>

**Repository**  CRAN

**Date/Publication**  2020-08-12 09:30:02 UTC

## R topics documented:

- dashPivottable-package ............................................. 2
- dashPivotTable .................................................. 2
- tips ........................................................... 5
dashPivotTable

**Description**

Pivot tables are useful for interactive presentation of summary statistics computed for data contained in another table. The 'dashPivotTable' package wraps 'react-pivottable', making it easy to add drag-and-drop tables into your 'dash' applications.

**Author(s)**

**Maintainer:** Ryan Patrick Kyle <ryan@plotly.com>

**dashPivotTable**

*PivotTable component*

**Description**

Pivot tables are useful for interactive presentation of summary statistics computed for data contained in another table. This function provides a convenient Dash interface to the 'react-pivottable' component, which makes it easy to embed pivot tables into Dash for R applications. Within React, the interactive component provided by 'react-pivottable' is 'PivotTableUI', but output rendering is delegated to the non-interactive 'PivotTable' component, which accepts a subset of its properties. 'PivotTable' in turn delegates to a specific renderer component, such as the default 'TableRenderer', which accepts a subset of the same properties. Finally, most renderers will create non-React PivotData objects to handle the actual computations, which also accept a subset of the same properties as the rest of the stack. The arguments for this function correspond to properties of the component; a full list is provided below. 'react-pivottable' was developed by Nicolas Kruchten; source for this component is available from https://github.com/plotly/react-pivottable.

**Usage**

dashPivotTable(id=NULL, data=NULL, hiddenAttributes=NULL, hiddenFromAggregators=NULL, hiddenFromDragDrop=NULL, menuLimit=NULL, unusedOrientationCutoff=NULL, cols=NULL, colOrder=NULL, rows=NULL, rowOrder=NULL, aggregatorName=NULL, vals=NULL, valueFilter=NULL, rendererName=NULL)
**Arguments**

- **id** Character. The ID used to identify this component in Dash callbacks
- **data** Unnamed list. Data to be summarized
- **hiddenAttributes** Unnamed list. Specifies attribute names to omit from the UI
- **hiddenFromAggregators** Unnamed list. Specifies attribute names to omit from the aggregator arguments dropdowns
- **hiddenFromDragDrop** Unnamed list. Specifies attribute names to omit from the drag and drop portion of the UI
- **menuLimit** Numeric. Maximum number of values to list in the double-click menu
- **unusedOrientationCutoff** Numeric. If the attributes’ names’ combined length in characters exceeds this value then the unused attributes area will be shown vertically to the left of the UI instead of horizontally above it. 0 therefore means ‘always vertical’, and infinity means ‘always horizontal’.
- **cols** Unnamed list. Specifies which columns are currently in the column area
- **colOrder** Character. The order in which column data is provided to the renderer, must be one of "key_a_to_z", "value_a_to_z", "value_z_to_a", ordering by value orders by column total
- **rows** Unnamed list. Specifies which rows are currently inside the row area.
- **rowOrder** Character. The order in which row data is provided to the renderer, must be one of "key_a_to_z", "value_a_to_z", "value_z_to_a", ordering by value orders by row total
- **aggregatorName** Character. Specifies which aggregator is currently selected. e.g. Count, Sum, Average, etc.
- **vals** Unnamed list. Values to use for the aggregator.
- **valueFilter** Named list. Value filter for each attribute name.
- **rendererName** Character. Specifies which renderer is currently selected. e.g. Table, Line Chart, Scatter Chart, etc.

**Value**

named list of JSON elements corresponding to React.js properties and their values

**Examples**

```
# Input data for dashPivottable may be passed in the "list-of-lists"
# format -- scroll down to see an example which demonstrates how
# to pass a data.frame into dashPivottable directly.
if (interactive() && require(dash)) {
  library(dash)
  library(dashPivottable)
  library(dashHtmlComponents)
```
app <- Dash$new()
app$title("Summary statistics for tips data")

app$layout(
  htmlDiv(
    list(
      dashPivotTable(
        id = "table",
        data = tips,
        cols = list("Day of Week"),
        colOrder = "key_a_to_z",
        rows = list("Party Size"),
        rowOrder = "key_a_to_z",
        rendererName = "Grouped Column Chart",
        aggregatorName = "Average",
        vals = list("Total Bill"),
        valueFilter = list("Day of Week"=list("Thursday"=FALSE))
      ),
      htmlDiv(
        id = "output"
      )
    )
  )
)

app$callback(output = output(id="output", property="children"),
             params = list(input(id="table", property="cols"),
                           input(id="table", property="rows"),
                           input(id="table", property="rowOrder"),
                           input(id="table", property="colOrder"),
                           input(id="table", property="aggregatorName"),
                           input(id="table", property="rendererName")),
             function(cols, rows, row_order, col_order, aggregator, renderer) {
              return(
                list(
                  htmlP(cols, id="columns"),
                  htmlP(rows, id="rows"),
                  htmlP(row_order, id="row_order"),
                  htmlP(col_order, id="col_order"),
                  htmlP(aggregator, id="aggregator"),
                  htmlP(renderer, id="renderer")
                )
              )
            } )

app$run_server(debug=TRUE)

# This example illustrates the use of `df_to_list` to format a data.frame
# for use with dashPivotTable
library(dashTable)
app <- Dash$new()
app$title("Summary statistics for iris data")

app$layout(
  htmlDiv(
    list(
      dashPivotTable(
        id = "table",
        data = df_to_list(Loblolly),
        cols = list("Seed"),
        colOrder = "key_a_to_z",
        rows = list("age"),
        rowOrder = "key_a_to_z",
        rendererName = "Grouped Column Chart",
        aggregatorName = "Average",
        vals = list("height")
      ),
      htmlDiv(
        id = "output"
      )
    )
  )
)

app$callback(output = output(id="output", property="children"),
params = list(input(id="table", property="cols"),
              input(id="table", property="rows"),
              input(id="table", property="rowOrder"),
              input(id="table", property="colOrder"),
              input(id="table", property="aggregatorName"),
              input(id="table", property="rendererName")),
function(cols, rows, row_order, col_order, aggregator, renderer) {
  return(
    list(
      htmlP(cols, id="columns"),
      htmlP(rows, id="rows"),
      htmlP(row_order, id="row_order"),
      htmlP(col_order, id="col_order"),
      htmlP(aggregator, id="aggregator"),
      htmlP(renderer, id="renderer")
    )
  )
})

app$run_server(debug=TRUE)

---
tips

*Tips received by a single server over a two month period*
Description

In 1990, a server recorded data on all tips received during a two and a half month period working in a single restaurant. The restaurant was part of a national chain and was located in a suburban shopping center.

Usage

tips

Format

A data frame with 244 rows and 7 variables:

- **total_bill**: total bill (cost of the meal), including tax, in US dollars
- **tip**: amount of gratuity received, in US dollars
- **sex**: sex of person paying (0 = male, 1 = female)
- **smoker**: was at least one member of the party a smoker? (0 = no, 1 = yes)
- **day**: 3 = Thursday, 4 = Friday, 5 = Saturday, 6 = Sunday
- **time**: 0 = day, 1 = night
- **size**: party size

Source

Index

* datasets
  tips, 5

dashPivotTable, 2
dashPivottable
  (dashPivottable-package), 2
dashPivottable-package, 2
tips, 5