

# Package ‘rhandsontable’

November 3, 2016

**Type** Package

**Title** Interface to the 'Handsontable.js' Library

**Version** 0.3.4

**Maintainer** Jonathan Owen <jonathanro@gmail.com>

**Description** An R interface to the 'Handsontable' JavaScript library, which is a minimalist Excel-like data grid editor. See <<https://handsontable.com/>> for details.

**License** MIT + file LICENSE

**URL** <http://jrowen.github.io/rhandsontable/>

**BugReports** <https://github.com/jrowen/rhandsontable/issues>

**Imports** jsonlite, htmlwidgets (>= 0.3.3), magrittr, methods

**Suggests** knitr, rmarkdown, shiny (>= 0.13), miniUI (>= 0.1.1), rstudioapi (>= 0.6), htmltools

**VignetteBuilder** knitr

**RoxygenNote** 5.0.1

**NeedsCompilation** no

**Author** Jonathan Owen [aut, cre, cph],

Jeff Allen [ctb],

Yihui Xie [ctb],

Enzo Martoglio [ctb],

Warpechowski Marcin [ctb, cph] (Handsontable.js library),

Handsoncode sp. z o.o. [ctb, cph] (Handsontable.js library),

Aisch Gregor [ctb, cph] (Chroma.js library),

F<c3><b6>retagsplatsen [ctb, cph] (Numbro.js library),

Draper Adam [ctb, cph] (Numeral.js library),

Wood Tim [ctb, cph] (Moment.js library),

Chernev Iskren [ctb, cph] (Moment.js library),

Moment.js contributors [ctb, cph] (Moment.js library),

Bushell David [ctb, cph] (Pikaday.js library),

jQuery Foundation [ctb, cph] (jQuery.js library),

Splunk Inc [ctb, cph] (Sparkline.js library),

Russell Kent [ctb, cph] (Sparkline.js library),

Rohan Jon [ctb, cph] (ZeroClipboard library),

Greene James [ctb, cph] (ZeroClipboard library)

**Repository** CRAN

**Date/Publication** 2016-11-03 21:10:31

## R topics documented:

rhandsontable-package . . . . .	2
editAddin . . . . .	3
hot_cell . . . . .	3
hot_col . . . . .	4
hot_cols . . . . .	5
hot_context_menu . . . . .	6
hot_heatmap . . . . .	7
hot_rows . . . . .	8
hot_table . . . . .	8
hot_to_r . . . . .	9
hot_validate_character . . . . .	10
hot_validate_numeric . . . . .	11
renderRHandsontable . . . . .	12
rhandsontable . . . . .	12
rhandsontable-exports . . . . .	13
rHandsontableOutput . . . . .	14
<b>Index</b>	<b>15</b>

---

rhandsontable-package *rhandsontable*

---

### Description

R interface for creating tables using Handsontable, url<http://handsontable.com/>

### Details

For full documentation on the package, visit <http://jrowen.github.io/rhandsontable/>

---

editAddin	<i>Edit a Data Frame.</i>
-----------	---------------------------

---

### Description

Interactively edit a `data.frame` or `data.table`. The resulting code will be emitted as a call to reload the data from a temp RDS file.

### Usage

```
editAddin()
```

### Details

This addin can be used to interactively edit. The intended way to use this is as follows:

1. Highlight a symbol naming a `data.frame` or `data.table` in your R session, e.g. `mtcars`. 2. Execute this addin, to interactively edit it.

When you're done, the code performing this operation will be emitted at the cursor position.

This function borrows heavily from <https://github.com/rstudio/addinexamples/blob/master/R/subsetAddin.R>

---

hot_cell	<i>Handsontable widget</i>
----------	----------------------------

---

### Description

Configure single cell. See [Handsontable.js](#) for details.

### Usage

```
hot_cell(hot, row, col, comment = NULL)
```

### Arguments

hot	rhandsontable object
row	numeric row index
col	numeric column index
comment	character comment to add to cell

### See Also

[hot\\_cols](#), [hot\\_rows](#)

**Examples**

```
library(rhandsontable)
DF = data.frame(val = 1:10, bool = TRUE, big = LETTERS[1:10],
               small = letters[1:10],
               dt = seq(from = Sys.Date(), by = "days", length.out = 10),
               stringsAsFactors = FALSE)

rhandsontable(DF, readOnly = TRUE) %>%
  hot_cell(1, 1, "Test comment")
```

---

hot_col	<i>Handsontable widget</i>
---------	----------------------------

---

**Description**

Configure single column.

**Usage**

```
hot_col(hot, col, type = NULL, format = NULL, source = NULL,
        strict = NULL, readOnly = NULL, validator = NULL, allowInvalid = NULL,
        halign = NULL, valign = NULL, renderer = NULL, copyable = NULL,
        dateFormat = NULL, default = NULL, language = NULL, ...)
```

**Arguments**

hot	rhandsontable object
col	vector of column names or indices
type	character specify the data type. Options include: numeric, date, checkbox, select, dropdown, autocomplete, password, and handsontable (not implemented yet)
format	characer specifying column format. See Cell Types at <a href="#">Handsontable.js</a> for the formatting options for each data type. Numeric columns are formatted using <a href="#">Numeral.js</a> .
source	a vector of choices for select, dropdown and autocomplete column types
strict	logical specifying whether values not in the source vector will be accepted
readOnly	logical making the table read-only
validator	character defining a Javascript function to be used to validate user input. See <code>hot_validate_numeric</code> and <code>hot_validate_character</code> for pre-build validators.
allowInvalid	logical specifying whether invalid data will be accepted. Invalid data cells will be color red.
halign	character defining the horizontal alignment. Possible values are <code>htLeft</code> , <code>htCenter</code> , <code>htRight</code> and <code>htJustify</code>

valign	character defining the vertical alignment. Possible values are htTop, htMiddle, htBottom
renderer	character defining a Javascript function to be used to format column cells. Can be used to implement conditional formatting.
copyable	logical defining whether data in a cell can be copied using Ctrl + C
dateFormat	character defining the date format. See <a href="#">Moment.js</a> for details.
default	default column value for new rows (NA if not specified; shiny only)
language	locale passed to <a href="#">Numbro.js</a> ; default is 'en-US'.
...	passed to handsontable

**See Also**

[hot\\_cols](#), [hot\\_rows](#), [hot\\_cell](#)

**Examples**

```
library(rhandsontable)
DF = data.frame(val = 1:10, bool = TRUE, big = LETTERS[1:10],
               small = letters[1:10],
               dt = seq(from = Sys.Date(), by = "days", length.out = 10),
               stringsAsFactors = FALSE)

rhandsontable(DF, rowHeaders = NULL) %>%
  hot_col(col = "big", type = "dropdown", source = LETTERS) %>%
  hot_col(col = "small", type = "autocomplete", source = letters,
         strict = FALSE)
```

---

hot_cols	<i>Handsontable widget</i>
----------	----------------------------

---

**Description**

Configure multiple columns.

**Usage**

```
hot_cols(hot, colWidths = NULL, columnSorting = NULL,
         manualColumnMove = NULL, manualColumnResize = NULL,
         fixedColumnsLeft = NULL, ...)
```

**Arguments**

hot	rhandsontable object
colWidths	a scalar or numeric vector of column widths
columnSorting	logical enabling row sorting. Sorting only alters the table presentation and the original dataset row order is maintained. The sorting will be done when a user click on column name

```

manualColumnMove      logical enabling column drag-and-drop reordering
manualColumnResize    logical enabling column width resizing
fixedColumnsLeft      a numeric vector indicating which columns should be frozen on the left
...                   passed to hot_col

```

**See Also**

[hot\\_col](#), [hot\\_rows](#), [hot\\_cell](#)

**Examples**

```

library(rhandsontable)
DF = data.frame(val = 1:10, bool = TRUE, big = LETTERS[1:10],
               small = letters[1:10],
               dt = seq(from = Sys.Date(), by = "days", length.out = 10),
               stringsAsFactors = FALSE)

rhandsontable(DF) %>%
  hot_cols(columnSorting = TRUE)

```

---

hot_context_menu	<i>Handsontable widget</i>
------------------	----------------------------

---

**Description**

Configure the options for the right-click context menu. See [Context Menu](#) and [jquery contextMenu](#) for details.

**Usage**

```

hot_context_menu(hot, allowRowEdit = TRUE, allowColEdit = TRUE,
  allowReadOnly = FALSE, allowComments = FALSE,
  allowCustomBorders = FALSE, customOpts = NULL, ...)

```

**Arguments**

hot	rhandsontable object
allowRowEdit	logical enabling row editing
allowColEdit	logical enabling column editing. Note that Handsontable does not support column add/remove when column types are defined (i.e. useTypes == TRUE in rhandsontable).
allowReadOnly	logical enabling read-only toggle
allowComments	logical enabling comments

```

allowCustomBorders    logical enabling custom borders
customOpts            list
...                   ignored

```

### Examples

```

library(rhandsontable)
DF = data.frame(val = 1:10, bool = TRUE, big = LETTERS[1:10],
               small = letters[1:10],
               dt = seq(from = Sys.Date(), by = "days", length.out = 10),
               stringsAsFactors = FALSE)

rhandsontable(DF) %>%
  hot_context_menu(allowRowEdit = FALSE, allowColEdit = FALSE)

```

---

hot_heatmap	<i>Handsontable widget</i>
-------------	----------------------------

---

### Description

Add heatmap to table. See [Heatmaps for values in a column](#) for details.

### Usage

```
hot_heatmap(hot, cols, color_scale = c("#ED6D47", "#17F556"),
            renderer = NULL)
```

### Arguments

```

hot            rhandsontable object
cols          numeric vector of columns to include in the heatmap. If missing all columns are
              used.
color_scale   character vector that includes the lower and upper colors
renderer      character defining a Javascript function to be used to determine the cell colors.
              If missing, rhandsontable:::renderer_heatmap is used.

```

### Examples

```

MAT = matrix(rnorm(50), nrow = 10, dimnames = list(LETTERS[1:10],
           letters[1:5]))

rhandsontable(MAT) %>%
  hot_heatmap()

```

---

hot_rows	<i>Handsontable widget</i>
----------	----------------------------

---

**Description**

Configure rows. See [Handsontable.js](#) for details.

**Usage**

```
hot_rows(hot, rowHeights = NULL, fixedRowsTop = NULL)
```

**Arguments**

hot	rhandsontable object
rowHeights	a scalar or numeric vector of row heights
fixedRowsTop	a numeric vector indicating which rows should be frozen on the top

**See Also**

[hot\\_cols](#), [hot\\_cell](#)

**Examples**

```
library(rhandsontable)
MAT = matrix(rnorm(50), nrow = 10, dimnames = list(LETTERS[1:10],
  letters[1:5]))

rhandsontable(MAT, width = 300, height = 150) %>%
hot_cols(colWidths = 100, fixedColumnsLeft = 1) %>%
  hot_rows(rowHeights = 50, fixedRowsTop = 1)
```

---

hot_table	<i>Handsontable widget</i>
-----------	----------------------------

---

**Description**

Configure table. See [Handsontable.js](#) for details.

**Usage**

```
hot_table(hot, contextMenu = TRUE, stretchH = "none",
  customBorders = NULL, highlightRow = NULL, highlightCol = NULL,
  enableComments = FALSE, overflow = NULL, rowHeaderWidth = NULL, ...)
```



**Arguments**

hot	rhandsontable object
contextMenu	logical enabling the right-click menu
stretchH	character describing column stretching. Options are 'all', 'right', and 'none'. See <a href="#">Column stretching</a> for details.
customBorders	json object. See <a href="#">Custom borders</a> for details.
highlightRow	logical enabling row highlighting for the selected cell
highlightCol	logical enabling column highlighting for the selected cell
enableComments	logical enabling comments in the table
overflow	character setting the css overflow behavior. Options are auto (default), hidden and visible
rowHeaderWidth	numeric width (in px) for the rowHeader column
...	passed to <a href="#">Handsontable.js</a> constructor

**See Also**

[rhandsontable](#)

**Examples**

```
library(rhandsontable)
DF = data.frame(val = 1:10, bool = TRUE, big = LETTERS[1:10],
               small = letters[1:10],
               dt = seq(from = Sys.Date(), by = "days", length.out = 10),
               stringsAsFactors = FALSE)

rhandsontable(DF) %>%
hot_table(highlightCol = TRUE, highlightRow = TRUE)
```

---

hot\_to\_r

*Handsontable widget*

---

**Description**

Convert handsontable data to R object. Can be used in a shiny app to convert the input json to an R dataset.

**Usage**

```
hot_to_r(...)
```

**Arguments**

... passed to `rhandsontable:::toR`

**See Also**

[rHandsontableOutput](#)

---

hot\_validate\_character

*Handsontable widget*

---

**Description**

Add numeric validation to a column

**Usage**

```
hot_validate_character(hot, cols, choices, allowInvalid = FALSE)
```

**Arguments**

hot	rhandsontable object
cols	vector of column names or indices
choices	a vector of acceptable numeric choices. It will be evaluated after min and max if specified.
allowInvalid	logical specifying whether invalid data will be accepted. Invalid data cells will be color red.

**See Also**

[hot\\_validate\\_numeric](#)

**Examples**

```
library(rhandsontable)
DF = data.frame(val = 1:10, bool = TRUE, big = LETTERS[1:10],
               small = letters[1:10],
               dt = seq(from = Sys.Date(), by = "days", length.out = 10),
               stringsAsFactors = FALSE)

rhandsontable(DF) %>%
  hot_validate_character(col = "big", choices = LETTERS[1:10])
```

---

hot\_validate\_numeric *Handsontable widget*

---

## Description

Add numeric validation to a column

## Usage

```
hot_validate_numeric(hot, cols, min = NULL, max = NULL, choices = NULL,  
  exclude = NULL, allowInvalid = FALSE)
```

## Arguments

hot	rhandsontable object
cols	vector of column names or indices
min	minimum value to accept
max	maximum value to accept
choices	a vector of acceptable numeric choices. It will be evaluated after min and max if specified.
exclude	a vector of unacceptable numeric values
allowInvalid	logical specifying whether invalid data will be accepted. Invalid data cells will be color red.

## See Also

[hot\\_validate\\_character](#)

## Examples

```
library(rhandsontable)  
MAT = matrix(rnorm(50), nrow = 10, dimnames = list(LETTERS[1:10],  
  letters[1:5]))  
  
rhandsontable(MAT * 10) %>%  
  hot_validate_numeric(col = 1, min = -50, max = 50, exclude = 40)  
  
rhandsontable(MAT * 10) %>%  
  hot_validate_numeric(col = 1, choices = c(10, 20, 40))
```

---

renderRHandsontable *Handsontable widget*

---

### Description

Shiny bindings for rhandsontable

### Usage

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

### Arguments

expr	an expression that generates an rhandsontable.
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.

### See Also

[rHandsontableOutput](#), [hot\\_to\\_r](#)

---

rhandsontable *Handsontable widget*

---

### Description

Create a [Handsontable.js](#) widget.

### Usage

```
rhandsontable(data, colHeaders, rowHeaders, comments = NULL,
  useTypes = TRUE, readOnly = NULL, selectCallback = FALSE,
  width = NULL, height = NULL, digits = 4, debug = NULL, ...)
```

### Arguments

data	a data.table, data.frame or matrix
colHeaders	a vector of column names. If missing colnames will be used. Setting to NULL will omit.
rowHeaders	a vector of row names. If missing rownames will be used. Setting to NULL will omit.
comments	matrix or data.frame of comments; NA values are ignored

useTypes	logical specifying whether column classes should be mapped to equivalent Javascript types. Note that Handsontable does not support column add/remove when column types are defined (i.e. useTypes == TRUE in rhandsontable).
readOnly	logical specifying whether the table is editable
selectCallback	logical enabling the afterSelect event to return data. This can be used with shiny to tie updates to a selected table cell.
width	numeric table width
height	numeric table height
digits	numeric passed to jsonlite::toJSON
debug	numeric Javascript log level
...	passed to hot_table and to the params property of the widget

### Details

For full documentation on the package, visit <http://jrowen.github.io/rhandsontable/>

### See Also

[hot\\_table](#), [hot\\_cols](#), [hot\\_rows](#), [hot\\_cell](#)

### Examples

```
library(rhandsontable)
DF = data.frame(val = 1:10, bool = TRUE, big = LETTERS[1:10],
               small = letters[1:10],
               dt = seq(from = Sys.Date(), by = "days", length.out = 10),
               stringsAsFactors = FALSE)

rhandsontable(DF, rowHeaders = NULL)
```

---

rhandsontable-exports *rhandsontable* exported operators

---

### Description

The following functions are imported and then re-exported from the rhandsontable package to enable use of the magrittr pipe operator with no additional library calls

---

`rHandsontableOutput` *Handsontable widget*

---

**Description**

Shiny bindings for rhandsontable

**Usage**

```
rHandsontableOutput(outputId, width = "100%", height = "100%")
```

**Arguments**

`outputId` output variable to read from  
`width, height` must be a valid CSS unit in pixels or a number, which will be coerced to a string and have "px" appended.

**See Also**

[renderRHandsontable](#)

# Index

`%>%` (rhandsontable-exports), 13

`editAddin`, 3

`hot_cell`, 3, 5, 6, 8, 13

`hot_col`, 4, 6

`hot_cols`, 3, 5, 5, 8, 13

`hot_context_menu`, 6

`hot_heatmap`, 7

`hot_rows`, 3, 5, 6, 8, 13

`hot_table`, 8, 13

`hot_to_r`, 9, 12

`hot_validate_character`, 10, 11

`hot_validate_numeric`, 10, 11

`renderRHandsontable`, 12, 14

`rhandsontable`, 9, 12

`rhandsontable-exports`, 13

`rhandsontable-package`, 2

`rHandsontableOutput`, 10, 12, 14