

# Package ‘rtabulator’

September 23, 2024

**Type** Package

**Title** R Bindings for 'Tabulator JS'

**Version** 0.1.2

**Date** 2024-09-22

**Maintainer** Stefan Kuethe <crazycapivara@gmail.com>

**Description** Provides R bindings for 'Tabulator JS' <<https://tabulator.info/>>. Makes it a breeze to create highly customizable interactive tables in 'rmarkdown' documents and 'shiny' applications. It includes filtering, grouping, editing, input validation, history recording, column formatters, packaged themes and more.

**URL** <https://github.com/eodaGmbH/rtabulator>  
<https://eodagmbh.github.io/rtabulator/>

**BugReports** <https://github.com/eodaGmbH/rtabulator/issues>

**License** MIT + file LICENSE

**Encoding** UTF-8

**RoxygenNote** 7.3.2

**Imports** glue, htmltools, htmlwidgets, purrr, readr, shiny

**Config/Needs/website** rmarkdown

**NeedsCompilation** no

**Author** Stefan Kuethe [aut, cre, cph],  
Nico Friess [aut],  
Oli Folkerd [cph] (Author of included tabulator.js library)

**Repository** CRAN

**Date/Publication** 2024-09-23 13:40:04 UTC

## Contents

|                                |   |
|--------------------------------|---|
| add_row . . . . .              | 2 |
| delete_selected_rows . . . . . | 3 |
| for_each_col . . . . .         | 4 |

|                                       |           |
|---------------------------------------|-----------|
| redo . . . . .                        | 4         |
| rtabulator-shiny . . . . .            | 5         |
| set_calculation . . . . .             | 6         |
| set_column_defaults . . . . .         | 6         |
| set_editor . . . . .                  | 8         |
| set_formatter_color . . . . .         | 9         |
| set_formatter_datetime . . . . .      | 10        |
| set_formatter_html . . . . .          | 11        |
| set_formatter_image . . . . .         | 12        |
| set_formatter_link . . . . .          | 13        |
| set_formatter_money . . . . .         | 14        |
| set_formatter_plaintext . . . . .     | 15        |
| set_formatter_progress . . . . .      | 16        |
| set_formatter_star . . . . .          | 17        |
| set_formatter_textarea . . . . .      | 18        |
| set_formatter_tick_cross . . . . .    | 19        |
| set_formatter_toggle_switch . . . . . | 19        |
| set_formatter_traffic_light . . . . . | 21        |
| set_header_filter . . . . .           | 22        |
| set_multi_column_header . . . . .     | 23        |
| set_options_group_by . . . . .        | 23        |
| set_options_pagination . . . . .      | 24        |
| set_tooltip . . . . .                 | 25        |
| spreadsheet_def . . . . .             | 26        |
| tabulator . . . . .                   | 27        |
| tabulatorContext . . . . .            | 28        |
| tabulator_options . . . . .           | 29        |
| titanic . . . . .                     | 31        |
| trigger_download . . . . .            | 32        |
| trigger_get_data . . . . .            | 33        |
| trigger_get_sheet_data . . . . .      | 33        |
| undo . . . . .                        | 34        |
| <b>Index</b>                          | <b>35</b> |

---

add\_row

*Add row to table*

---

### Description

Add row to table

### Usage

add\_row(ctx, row = NULL)

### Arguments

`ctx` A `tabulatorContext()` object.  
`row` (list): row data or NULL to add an empty row

### Value

A `tabulatorContext()` object

### Examples

```
tabulatorContext("table") |>  
  add_row()
```

---

`delete_selected_rows` *Delete selected rows from table*

---

### Description

Delete selected rows from table

### Usage

```
delete_selected_rows(ctx)
```

### Arguments

`ctx` A `tabulatorContext()` object.

### Value

A `tabulatorContext()` object

### Examples

```
tabulatorContext("table") |>  
  delete_selected_rows()
```

---

 for\_each\_col

*Apply a column setter function to multiple columns*


---

### Description

Apply a column setter function to multiple columns

### Usage

```
for_each_col(widget, columns = NULL, .f, ...)
```

### Arguments

|         |  |
|---------|--|
| widget  | A <a href="#">tabulator()</a> HTML widget.   |
| columns | (character vector): The columns the column setter function (.f) is applied to. If set to NULL, it is applied to all columns. |
| .f      | (function): The column setter function that updates the column settings.   |
| ...     | Arguments that are passed to .f.   |

### Value

The updated [tabulator\(\)](#) HTML widget

### Examples

```
numeric_cols <- c("Sepal_Length", "Sepal_Width", "Petal_Length", "Petal_Width")
tabulator(iris) |>
  for_each_col(numeric_cols, .f = set_header_filter, type = "number", func = "<=")
```

---

 redo

*Redo changes*


---

### Description

Redo changes

### Usage

```
redo(ctx)
```

### Arguments

|     |  |
|-----|--|
| ctx | A <a href="#">tabulatorContext()</a> object. |
|-----|--|

**Value**

A `tabulatorContext()` object

**Examples**

```
tabulatorContext("table") |>
  redo()
```

---

rtabulator-shiny      *Shiny bindings for rtabulator*

---

**Description**

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

**Usage**

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

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

**Arguments**

|                            |  |
|----------------------------|--|
| <code>outputId</code>      | output variable to read from   |
| <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>expr</code>          | An expression that generates a rtabulator  |
| <code>env</code>           | The environment in which to evaluate <code>expr</code> .   |
| <code>quoted</code>        | Is <code>expr</code> a quoted expression (with <code>quote()</code> )? This is useful if you want to save an expression in a variable. |

**Value**

Components for use inside a Shiny app.

---

|                 |                        |
|-----------------|------------------------|
| set_calculation | <i>Set calculation</i> |
|-----------------|------------------------|

---

**Description**

Set calculation

**Usage**

```
set_calculation(
  widget,
  column,
  func = c("avg", "max", "min", "sum", "count", "unique"),
  precision = 2,
  pos = c("top", "bottom")
)
```

**Arguments**

|           |  |
|-----------|--|
| widget    | A <a href="#">tabulator()</a> HTML widget.   |
| column    | (character): The column the func is applied to.                                    |
| func      | (character): The calculation function to be applied to the values of the column.   |
| precision | (integer) The number of decimals to display. Set to FALSE to display all decimals. |
| pos       | (character): Position at which calculated values are displayed.                    |

**Value**

The updated [tabulator\(\)](#) HTML widget

**Examples**

```
tabulator(iris) |>
  set_calculation("Sepal_Length", "avg")
```

---

|                     |                            |
|---------------------|----------------------------|
| set_column_defaults | <i>Set column defaults</i> |
|---------------------|----------------------------|

---

**Description**

Set column defaults

## Usage

```
set_column_defaults(  
  widget,  
  editor = FALSE,  
  header_filter = FALSE,  
  header_sort = TRUE,  
  tooltip = TRUE,  
  width = NULL,  
  ...  
)
```

## Arguments

|               |  |
|---------------|--|
| widget        | A <a href="#">tabulator()</a> HTML widget.   |
| editor        | (character, bool): One of "input" or "number". If set to FALSE cells are not editable. |
| header_filter | (character, bool): One of "input" or "number". Set to FALSE to disable header filters. |
| header_sort   | (bool): Whether to enable header sorting.  |
| tooltip       | (bool): Whether to show tooltips displaying the cell value.                            |
| width         | (integer): Fixed width of columns.   |
| ...           | Additional settings.   |

## Value

The updated [tabulator\(\)](#) HTML widget

## See Also

<https://tabulator.info/docs/6.2/columns#defaults>

## Examples

```
tabulator(iris, theme = "simple") |>  
  set_column_defaults(  
    header_filter = TRUE,  
    header_sort = FALSE,  
    tooltip = TRUE  
  )
```

---

 set\_editor

*Set editor*


---

### Description

Set editor

### Usage

```
set_editor(
  widget,
  column,
  editor = c("input", "textarea", "number", "range", "tickCross", "star", "progress",
    "date", "time", "datetime", "list"),
  validator = NULL,
  ...
)
```

### Arguments

|           |  |
|-----------|--|
| widget    | A <a href="#">tabulator()</a> HTML widget.                         |
| column    | The name of the column the formatter is applied to.                |
| editor    | (character): The editor type.                                      |
| validator | (character vector): One or more validators to validate user input. |
| ...       | Optional editor parameters depending on the selected editor.       |

### Value

The updated [tabulator\(\)](#) HTML widget

### See Also

- <https://tabulator.info/docs/6.2/edit> for available editors
- <https://tabulator.info/docs/6.2/validate> for available validators

### Examples

```
setup <- tabulator_options(edit_trigger_event = "click")

tabulator(iris[, c(2, 5)], setup) |>
  set_editor(
    column = "Species",
    editor = "list",
    values_lookup = "active",
    clearable = TRUE,
    autocomplete = TRUE
  ) |>
```



```
set_editor(  
  column = "Sepal_Width",  
  editor = "number",  
  min = 0,  
  max = 10,  
  step = 0.1,  
  validator = c("min:0", "max:10")  
)
```

---

set\_formatter\_color    *Set color formatter*

---

## Description

Set color formatter

## Usage

```
set_formatter_color(widget, column)
```

## Arguments

widget            A [tabulator\(\)](#) HTML widget.  
column            The name of the column the formatter is applied to.

## Value

The updated [tabulator\(\)](#) HTML widget

## Examples

```
data <- data.frame(  
  Label = c("R", "G", "B"),  
  Color = c("red", "green", "blue")  
)  
  
tabulator(data, width = 200) |>  
  set_formatter_color("Color")
```

---

set\_formatter\_datetime  
*Set datetime formatter*

---

## Description

Set datetime formatter

## Usage

```
set_formatter_datetime(  
  widget,  
  column,  
  input_format = "yyyy-MM-dd hh:ss:mm",  
  output_format = "yy/MM/dd",  
  invalid_placeholder = "(invalid datetime)",  
  timezone = NA,  
  hoz_align = "left"  
)
```

## Arguments

|                     |  |
|---------------------|--|
| widget              | A <a href="#">tabulator()</a> HTML widget.                                 |
| column              | The name of the column the formatter is applied to.                        |
| input_format        | (character): The datetime input format.                                    |
| output_format       | (character): The datetime output format.                                   |
| invalid_placeholder | (character): The value to be displayed if an invalid datetime is provided. |
| timezone            | (character): The timezone of the datetime.                                 |
| hoz_align           | (character): The horizontal alignment of the column.                       |

## Details

To use this formatter, you need to include the [luxon](#) HTML dependency with `tabulator(..., luxon = TRUE)`.

## Value

The updated [tabulator\(\)](#) HTML widget

**Examples**

```
data <- data.frame(
  Person = c("Franz", "Ferdinand"),
  Birthday = c(
    "2024-12-06 22:00:10",
    "2023-06-07 14:12:45"
  )
)

tabulator(data, luxon = TRUE) |>
  set_formatter_datetime("Birthday", output_format = "dd.MM.yyyy")
```

---

set\_formatter\_html      *Set HTML formatter*

---

**Description**

Set HTML formatter

**Usage**

```
set_formatter_html(widget, column, hoz_align = c("left", "center", "right"))
```

**Arguments**

|           |  |
|-----------|--|
| widget    | A <a href="#">tabulator()</a> HTML widget.           |
| column    | The name of the column the formatter is applied to.  |
| hoz_align | (character): The horizontal alignment of the column. |

**Value**

The updated [tabulator\(\)](#) HTML widget

**Examples**

```
data <- data.frame(
  id = c(1, 2, 3, 4, 5, 6, 7),
  text_style = c(
    "<i>Italic</i>",
    "<b>Bold</b>",
    "<span style='color: green'>Green</span>",
    "<del>Deleted</del>",
    "<small>Small</small>",
    "H<sub>2</sub>0",
    "x<sup>2</sup>"
  )
)

tabulator(data, width = 400) |>
  set_formatter_html("text_style")
```

---

set\_formatter\_image    *Set image formatter*

---

### Description

Set image formatter

### Usage

```
set_formatter_image(
  widget,
  column,
  height = "50px",
  width = "50px",
  url_prefix = NULL,
  url_suffix = NULL,
  hoz_align = "center"
)
```

### Arguments

|            |  |
|------------|--|
| widget     | A <a href="#">tabulator()</a> HTML widget.   |
| column     | The name of the column the formatter is applied to.  |
| height     | (character): A CSS value for the height of the image.  |
| width      | (character): A CSS value for the width of the image.   |
| url_prefix | (character): String to add to the start of the cell value when generating the image src url. |
| url_suffix | (character): String to add to the end of the cell value when generating the image src url.   |
| hoz_align  | (character): The horizontal alignment of the column.   |

### Value

The updated [tabulator\(\)](#) HTML widget

### Examples

```
image_url <- "https://picsum.photos/id/"
image_size <- 100

image_data <- data.frame(
  image = c(88, 98, 102, 201),
  label = c("Pic No. 88", "Pic No. 98", "Pic No. 102", "Pic No 201")
)

tabulator(image_data, tabulator_options(height = "400px")) |>
```

```

set_formatter_image(
    "image",
    height = image_size,
    width = image_size,
    url_prefix = image_url,
    url_suffix = glue::glue("/{image_size}"),
    hoz_align = "left"
)

```

---

set\_formatter\_link     *Set link formatter*

---

### Description

Set link formatter

### Usage

```

set_formatter_link(
  widget,
  column,
  label_field = NULL,
  url_prefix = NULL,
  url = NULL,
  target = "_blank",
  hoz_align = "left"
)

```

### Arguments

|             |   |
|-------------|---|
| widget      | A <a href="#">tabulator()</a> HTML widget.  |
| column      | The name of the column the formatter is applied to.   |
| label_field | (character): Column to be used as label for the link.   |
| url_prefix  | (character): Prefix to add to the URL value.  |
| url         | (JavaScript function): A JavaScript function that return the URL value. The cell is passed to the function as its first argument. Use <a href="#">JS</a> to pass JS code. |
| target      | (character): Target attribute of the anchor tag.  |
| hoz_align   | (character): The horizontal alignment of the column.  |

### Value

The updated [tabulator\(\)](#) HTML widget

**Examples**

```

data <- data.frame(
  label = c("This is a link", "This is another link"),
  link = c("https://eoda.de", "https://eoda.de/data-science"),
  link2 = c("eoda.de", "github.com"),
  link3 = c("guinan", "data-science")
)

js_func <- "(cell) => `https://eoda.de/${cell.getValue()}`"

tabulator(data) |>
  set_formatter_link("link", label_field = "label") |>
  set_formatter_link("link2", url_prefix = "https://") |>
  set_formatter_link("link3", url = htmlwidgets::JS(js_func), label_field = "label")

```

---

set\_formatter\_money    *Set money formatter*

---

**Description**

Set money formatter

**Usage**

```

set_formatter_money(
  widget,
  column,
  decimal = c(",", "."),
  thousand = c(".", ","),
  symbol = "$",
  symbol_after = "p",
  negative_sign = "-",
  precision = FALSE,
  hoz_align = "left"
)

```

**Arguments**

|              |  |
|--------------|--|
| widget       | A <a href="#">tabulator()</a> HTML widget.   |
| column       | The name of the column the formatter is applied to.  |
| decimal      | (character): Symbol to represent the decimal point.  |
| thousand     | (character, bool): Symbol to represent the thousands separator. Set to FALSE to disable the separator. |
| symbol       | (character): The currency symbol.  |
| symbol_after | (bool): Whether to put the symbol after the number.  |

|               |  |
|---------------|--|
| negative_sign | (character, bool): The sign to show in front of the number. Set to TRUE causes negative numbers to be enclosed in brackets (123.45), which is the standard style for negative numbers in accounting. |
| precision     | (integer, bool): The number of decimals to display. Set to FALSE to display all decimals that are provided.  |
| hoz_align     | (character): The horizontal alignment of the column.   |

### Value

The updated `tabulator()` HTML widget

### Examples

```
data <- data.frame(  
  Account_Number = c(  
    123456,  
    345667,  
    234567,  
    234566  
  ),  
  Account_Balance = c(100, -50, 200.30, -21.5)  
)  
  
tabulator(data) |>  
  set_formatter_money(  
    "Account_Balance",  
    symbol = "\u20AC",  
    symbol_after = FALSE,  
    hoz_align = "right"  
  )
```

---

set\_formatter\_plaintext

*Set plain text formatter*

---

### Description

Set plain text formatter

### Usage

```
set_formatter_plaintext(widget, column, hoz_align = "left")
```

### Arguments

|           |  |
|-----------|--|
| widget    | A <code>tabulator()</code> HTML widget.              |
| column    | The name of the column the formatter is applied to.  |
| hoz_align | (character): The horizontal alignment of the column. |

**Value**

The updated `tabulator()` HTML widget

**Examples**

```
tabulator(iris) |>
  set_formatter_plaintext("Species", hoz_align = "right")
```

---

```
set_formatter_progress
```

*Set progress formatter*

---

**Description**

Set progress formatter

**Usage**

```
set_formatter_progress(
  widget,
  column,
  min = NA,
  max = NA,
  color = c("yellow", "orange", "red"),
  legend = NA,
  legend_color = "#000000",
  legend_align = c("center", "left", "right", "justify"),
  hoz_align = "left"
)
```

**Arguments**

|                           |  |
|---------------------------|--|
| <code>widget</code>       | A <code>tabulator()</code> HTML widget.  |
| <code>column</code>       | The name of the column the formatter is applied to.  |
| <code>min</code>          | (numeric): The minimum value for progress bar. If set to NA, the minimum value of the column is used.  |
| <code>max</code>          | (numeric): The maximum value for progress bar. If set to NA, the maximum value of the column is used.  |
| <code>color</code>        | (character): Either a single color or a vector of colors   |
| <code>legend</code>       | (character, TRUE, JavaScript function): If set to TRUE, the value of the cell is displayed. Set to NA to display no value at all. Use <b>JS</b> to pass a JavaScript function as legend. In this case, the cell value is passed to the function as its first argument. |
| <code>legend_color</code> | (character): The text color of the legend.   |
| <code>legend_align</code> | (character): The text alignment of the legend.   |
| <code>hoz_align</code>    | (character): The horizontal alignment of the column.   |



**Value**

The updated `tabulator()` HTML widget

**Examples**

```
data <- data.frame(
  id = 1:6,
  value = c(10, 0, 100, 20, 40, 60),
  value2 = c(10, 0, 100, 20, 40, 60),
  value3 = c(10, 0, 100, 20, 40, 60)
)

js_func <- htmlwidgets::JS("(cellValue) => `${cellValue}%`")

tabulator(data) |>
  set_formatter_progress("value") |>
  set_formatter_progress("value2", legend = TRUE, legend_align = "left") |>
  set_formatter_progress("value3", legend = js_func, legend_align = "right")
```

---

set\_formatter\_star      *Set star rating formatter*

---

**Description**

Set star rating formatter

**Usage**

```
set_formatter_star(widget, column, number_of_stars = NA, hoz_align = "center")
```

**Arguments**

|                 |   |
|-----------------|---|
| widget          | A <code>tabulator()</code> HTML widget.   |
| column          | The name of the column the formatter is applied to.   |
| number_of_stars | The maximum number of stars to be displayed. If set to NA, the maximum value of the column is used. |
| hoz_align       | (character): The horizontal alignment of the column.  |

**Value**

The updated `tabulator()` HTML widget

## Examples

```
data <- data.frame(
  Passengers = c("Hans", "Franz", "Ferdinand", "Julia"),
  PassengerClass = c(1, 2, 1, 3)
)

tabulator(data, width = 200) |>
  set_formatter_star("PassengerClass", number_of_stars = max(data$PassengerClass))
```

---

set\_formatter\_textarea

*Set text area formatter*

---

## Description

Set text area formatter

## Usage

```
set_formatter_textarea(widget, column, hoz_align = "left")
```

## Arguments

|           |  |
|-----------|--|
| widget    | A <a href="#">tabulator()</a> HTML widget.           |
| column    | The name of the column the formatter is applied to.  |
| hoz_align | (character): The horizontal alignment of the column. |

## Value

The updated [tabulator\(\)](#) HTML widget

## Examples

```
data <- data.frame(
  id = c(1, 2),
  txt = c(
    "This\nis\nsome\ntext\nwith\nmultiple\nline\nbreaks",
    "- R\n- Python\n- Julia"
  )
)

tabulator(data, width = 200) |>
  set_formatter_textarea("txt")
```

---

set\_formatter\_tick\_cross  
*Set tick cross formatter*

---

**Description**

Set tick cross formatter

**Usage**

```
set_formatter_tick_cross(widget, column, hoz_align = "center")
```

**Arguments**

|           |  |
|-----------|--|
| widget    | A <a href="#">tabulator()</a> HTML widget.           |
| column    | The name of the column the formatter is applied to.  |
| hoz_align | (character): The horizontal alignment of the column. |

**Value**

The updated [tabulator\(\)](#) HTML widget

**Examples**

```
data <- data.frame(  
  Artist = c("Art Blackey", "Nirvana", "Bob Marley"),  
  Grunge = c(0, 1, 0),  
  Jazz = c(1, 0, 0),  
  Reggae = c(0, 0, 1)  
)  
  
tabulator(data, width = 400) |>  
  set_formatter_tick_cross("Grunge") |>  
  set_formatter_tick_cross("Jazz") |>  
  set_formatter_tick_cross("Reggae")
```

---

set\_formatter\_toggle\_switch  
*Set toggle switch formatter*

---

**Description**

Set toggle switch formatter

## Usage

```
set_formatter_toggle_switch(  
  widget,  
  column,  
  size = 20,  
  on_value = "on",  
  off_value = "off",  
  on_truthy = FALSE,  
  on_color = "green",  
  off_color = "red",  
  clickable = TRUE  
)
```

## Arguments

|           |  |
|-----------|--|
| widget    | A <a href="#">tabulator()</a> HTML widget.                                   |
| column    | The name of the column the formatter is applied to.                          |
| size      | (numeric): The size of the switch in pixels.                                 |
| on_value  | (character): The value of the cell for the switch to be on.                  |
| off_value | (character) The value of the cell for the switch to be off.                  |
| on_truthy | (bool): Whether to show the switch as on if the value of the cell is truthy. |
| on_color  | (character): The color of the switch if it is on.                            |
| off_color | (character): The color of the switch if it is off.                           |
| clickable | (bool): Enable switch functionality to toggle the cell value on click.       |

## Value

The updated [tabulator\(\)](#) HTML widget

## Examples

```
data <- data.frame(  
  Language = c("R", "Python", "Julia"),  
  Available = c("yes", "yes", "no")  
)  
  
tabulator(data) |>  
  set_formatter_toggle_switch("Available", on_value = "yes", off_value = "no")
```

---

```
set_formatter_traffic_light
  Set traffic light formatter
```

---

## Description

Set traffic light formatter

## Usage

```
set_formatter_traffic_light(  
  widget,  
  column,  
  min = NA,  
  max = NA,  
  color = c("green", "orange", "red"),  
  hoz_align = "center"  
)
```

## Arguments

|           |   |
|-----------|---|
| widget    | A <a href="#">tabulator()</a> HTML widget.  |
| column    | The name of the column the formatter is applied to.   |
| min       | (numeric): The minimum value for progress bar. If set to NA, the minimum value of the column is used. |
| max       | (numeric): The maximum value for progress bar. If set to NA, the maximum value of the column is used. |
| color     | (character): Either a single color or a vector of colors  |
| hoz_align | (character): The horizontal alignment of the column.  |

## Value

The updated [tabulator\(\)](#) HTML widget

## Examples

```
data <- data.frame(  
  label = 1:10,  
  value = 1:10  
)  
  
tabulator(data, width = 200) |>  
  set_formatter_traffic_light("value") |>  
  set_formatter_plaintext("label", hoz_align = "center")
```

---

set\_header\_filter      *Set header filter*

---

## Description

Set header filter

## Usage

```
set_header_filter(  
  widget,  
  column,  
  type = c("input", "number", "list", "tickCross"),  
  func = c("like", "=", ">", ">=", "<", "<="),  
  values_lookup = TRUE,  
  clearable = TRUE,  
  placeholder = NULL  
)
```

## Arguments

|               |  |
|---------------|--|
| widget        | A <a href="#">tabulator()</a> HTML widget.                       |
| column        | The name of the column the formatter is applied to.              |
| type          | (character): The type of the filter.                             |
| func          | (character): The filter function.                                |
| values_lookup | (bool): Whether to use unique column values for the list filter. |
| clearable     | (bool): Whether to display a cross to clear the filter.          |
| placeholder   | (character): Text that is displayed when no filter is set.       |

## Value

The updated [tabulator\(\)](#) HTML widget

## Examples

```
data <- data.frame(  
  age = c(10, 20, 40),  
  first_name = c("Franz", "Ferdinand", "Julia"),  
  last_name = c("Young", "Bowie", "Blackey")  
)  
  
tabulator(data) |>  
  set_header_filter("age", type = NULL, func = "<=", placeholder = "max age") |>  
  set_header_filter("first_name", placeholder = "Fran") |>  
  set_header_filter("last_name", type = "list")
```

---

```
set_multi_column_header
```

*Set multi column header*

---

**Description**

Set multi column header

**Usage**

```
set_multi_column_header(widget, multi_columns)
```

**Arguments**

widget            A [tabulator\(\)](#) HTML widget.  
multi\_columns    (list): Multi column definitions.

**Value**

The updated [tabulator\(\)](#) HTML widget

**Examples**

```
multi_columns <- list(  
  Sepal = c("Sepal_Length", "Sepal_Width"),  
  Petal = c("Petal_Length", "Petal_Width")  
)  
  
tabulator(iris) |>  
  set_multi_column_header(multi_columns)
```

---

```
set_options_group_by
```

*Set group by options*

---

**Description**

Set group by options

**Usage**

```
set_options_group_by(  
  widget,  
  group_by,  
  group_start_open = TRUE,  
  group_toggle_element = "header",  
  ...  
)
```

**Arguments**

|                      |   |
|----------------------|---|
| widget               | A <code>tabulator()</code> HTML widget.   |
| group_by             | (character vector): Field to group rows by. Pass multiple values for multi level grouping.  |
| group_start_open     | (bool vector): Open state of groups when grouping is enabled. The length of the vector depends on the number of levels set with <code>group_by</code> . |
| group_toggle_element | (character, bool): One of arrow or header. Set to FALSE to disable toggling at all.   |
| ...                  | Further options.  |

**Value**

The updated `tabulator()` HTML widget

**Examples**

```
tabulator(iris) |>
  set_options_group_by("Species", group_start_open = FALSE)
```

---

```
set_options_pagination
```

*Set pagination options*

---

**Description**

Set pagination options

**Usage**

```
set_options_pagination(
  widget,
  pagination = TRUE,
  pagination_size = 10,
  pagination_size_selector = FALSE,
  pagination_add_row = c("page", "table"),
  ...
)
```

**Arguments**

|                 |   |
|-----------------|---|
| widget          | A <code>tabulator()</code> HTML widget. |
| pagination      | (bool): Whether to enable pagination.   |
| pagination_size | (integer): Number of rows on each page. |



pagination\_size\_selector  
     (list): Add pagination size selector.  
 pagination\_add\_row  
     (character): Where to add rows to the table when pagination is enabled.  
 ...           Further options.

**Value**

The updated `tabulator()` HTML widget

**Examples**

```
tabulator(iris) |>
  set_options_pagination(pagination_size_selector = c(10, 20, 50))
```

---

|             |                    |
|-------------|--------------------|
| set_tooltip | <i>Set tooltip</i> |
|-------------|--------------------|

---

**Description**

Set tooltip

**Usage**

```
set_tooltip(widget, column)
```

**Arguments**

widget           A `tabulator()` HTML widget.  
 column           The name of the column the formatter is applied to.

**Value**

The updated `tabulator()` HTML widget

**Examples**

```
tabulator(airquality) |>
  set_formatter_traffic_light("Ozone") |>
  set_tooltip("Ozone")
```

---

|                 |   |
|-----------------|---|
| spreadsheet_def | <i>Spreadsheet definition for tabulator() HTML widget</i> |
|-----------------|---|

---

### Description

Spreadsheet definition for tabulator() HTML widget

### Usage

```
spreadsheet_def(title, key = NULL, data = NULL)
```

### Arguments

|       |  |
|-------|--|
| title | Name of the spreadsheet.   |
| key   | Unique key of the spreadsheet.   |
| data  | Initial data of the spreadsheet. Set to NULL to create an empty spreadsheet. |

### Value

list with spreadsheet options to be used as a sheet of the spreadsheet\_sheets parameter in [tabulator\\_options\(\)](#)

### Examples

```
setup <- tabulator_options(  
  spreadsheet = TRUE,  
  spreadsheet_sheets = list(  
    spreadsheet_def(  
      title = "First",  
      data = list(c(1, 2, 3))  
    ),  
    spreadsheet_def(  
      title = "Second",  
      data = list(c(4, 5, 6))  
    )  
  ),  
  spreadsheet_sheet_tabs = TRUE  
)  
  
tabulator(data = NULL, setup, theme = "midnight")
```

---

|           |  |
|-----------|--|
| tabulator | <i>HTML widget to work with tabular data</i> |
|-----------|--|

---

## Description

HTML widget to work with tabular data

## Usage

```
tabulator(
  data,
  options = tabulator_options(),
  editable = FALSE,
  luxon = FALSE,
  sheetjs = FALSE,
  theme = c("default", "midnight", "modern", "simple", "site", "bootstrap3",
    "bootstrap4", "bootstrap5", "bulma", "materialize", "semanticui"),
  col_select = NULL,
  width = NULL,
  height = NULL,
  element_id = NULL,
  ...
)
```

## Arguments

|               |  |
|---------------|--|
| data          | (data.frame, character or list): In spreadsheet mode data needs to be a list or NULL for an empty spreadsheet.                                       |
| options       | List of setup options. Defaults to <a href="#">tabulator_options()</a> .   |
| editable      | Whether the table is editable.   |
| luxon         | Whether to add <a href="#">luxon</a> HTML dependency, which is required for <a href="#">set_formatter_datetime()</a> .                               |
| sheetjs       | Whether to add <a href="#">sheetjs</a> HTML dependency, which is required for <a href="#">trigger_download()</a> to support downloads of xlsx files. |
| theme         | Name of the theme to be applied to the table.  |
| col_select    | Passed to <a href="#">readr::read_csv()</a> if data is a file name.  |
| width, height | Output size.   |
| element_id    | Unique ID of the widget element.   |
| ...           | Named arguments that are appended to the options parameter.  |

## Details

Dots in column names are replaced by underscores.

**Value**

A HTML widget

**See Also**

[tabulatorOutput\(\)](#) for use within Shiny apps

**Examples**

```
setup <- tabulator_options(  
  spreadsheet = TRUE,  
  spreadsheet_sheets = list(  
    spreadsheet_def(  
      title = "First",  
      data = list(c(1, 2, 3))  
    ),  
    spreadsheet_def(  
      title = "Second",  
      data = list(c(4, 5, 6))  
    )  
  ),  
  spreadsheet_sheet_tabs = TRUE  
)  
  
tabulator(data = NULL, setup, theme = "midnight")
```

---

|                  |   |
|------------------|---|
| tabulatorContext | <i>Context object to update a tabulator() HTML widget</i> |
|------------------|---|

---

**Description**

Context object to update a tabulator() HTML widget

**Usage**

```
tabulatorContext(output_id, session = shiny::getDefaultReactiveDomain())
```

**Arguments**

|           |  |
|-----------|--|
| output_id | A tabulator output id set with <a href="#">tabulatorOutput()</a> . |
| session   | A shiny session object.  |

**Details**

Makes it possible to update your [tabulator\(\)](#) HTML widget in Shiny apps.

**Value**

A context object that can be used to interact with a [tabulator\(\)](#) HTML widget

## Examples

```
tabulatorContext("table") |>  
  trigger_download("csv")
```

---

|                   |  |
|-------------------|--|
| tabulator_options | <i>Setup options for tabulator() HTML widget</i> |
|-------------------|--|

---

## Description

Setup options for tabulator() HTML widget

## Usage

```
tabulator_options(  
  height = "311px",  
  history = FALSE,  
  columns = NULL,  
  layout = c("fitColumns", "fitData", "fitDataFill", "fitDataStretch", "fitDataTable"),  
  responsive_layout = FALSE,  
  movable_columns = FALSE,  
  header_visible = TRUE,  
  row_height = NULL,  
  add_row_pos = c("bottom", "top"),  
  movable_rows = FALSE,  
  resizable_rows = FALSE,  
  frozen_rows = NULL,  
  index = "id",  
  group_by = NULL,  
  group_start_open = TRUE,  
  group_toggle_element = NULL,  
  edit_trigger_event = c("dblclick", "click", "focus"),  
  selectable_rows = "highlight",  
  pagination = FALSE,  
  pagination_size = 10,  
  pagination_size_selector = FALSE,  
  pagination_add_row = c("page", "table"),  
  spreadsheet = FALSE,  
  spreadsheet_rows = NULL,  
  spreadsheet_columns = NULL,  
  spreadsheet_column_definition = NULL,  
  spreadsheet_sheets = NULL,  
  spreadsheet_sheet_tabs = NULL,  
  ...  
)
```

**Arguments**

|                          |  |
|--------------------------|--|
| height                   | (character) The height of the table in pixels.   |
| history                  | (bool): Enable user interaction history functionality.   |
| columns                  | (list): Column definitions. If set to NULL, definitions are auto generated from table data.  |
| layout                   | (character): Layout mode of the table columns.   |
| responsive_layout        | (bool): Automatically hide and show columns to fit the width of the Tabulator element.   |
| movable_columns          | (bool): Allow users to move and reorder rows.  |
| header_visible           | (bool): Whether header is visible.   |
| row_height               | (numeric): A fixed height of the rows.   |
| add_row_pos              | (character): The table position where new rows are added.  |
| movable_rows             | (bool): Allow users to move and reorder rows.  |
| resizable_rows           | (bool): Allow user to resize rows.   |
| frozen_rows              | (numeric): Number of frozen rows.  |
| index                    | (character): Field to be used as the unique index for each row.  |
| group_by                 | (character vector): Field to group rows by. Pass multiple values for multi level grouping.   |
| group_start_open         | (bool vector): Open state of groups when grouping is enabled. The length of the vector depends on the number of levels set with group_by.  |
| group_toggle_element     | (character, bool): One of arrow or header. Set to FALSE to disable toggling at all.  |
| edit_trigger_event       | (character): Event that triggers a cell edit.  |
| selectable_rows          | (character, bool, integer) Set to FALSE to disable row selection. If set to TRUE, you can select as many rows as you want. If set to "highlight", rows are just highlighted but do not change state when clicked. An integer value sets the maximum number of rows that can be selected. |
| pagination               | (bool): Whether to enable pagination.  |
| pagination_size          | (integer): Number of rows on each page.  |
| pagination_size_selector | (list): Add pagination size selector.  |
| pagination_add_row       | (character): Where to add rows to the table when pagination is enabled.  |
| spreadsheet              | (bool): Whether to enable spreadsheet mode.  |
| spreadsheet_rows         | (integer): Number of spreadsheet rows.   |

`spreadsheet_columns`  
 (integer): Number of spreadsheet columns.  
`spreadsheet_column_definition`  
 Column definition used for all columns in the sheet.  
`spreadsheet_sheets`  
 (list): List of sheet definitions.  
`spreadsheet_sheet_tabs`  
 (bool): Whether to show sheet tabs in the footer.  
 ... Further options.

**Value**

list with setup options to be used as options parameter in `tabulator()`

**See Also**

<https://tabulator.info/docs/6.2/options> for a complete list of available options

**Examples**

```

setup <- tabulator_options(
  group_by = "Species",
  pagination = TRUE,
  pagination_size = 10,
  pagination_size_selector = c(10, 20, 50),
  movable_columns = TRUE,
  movable_rows = TRUE
)
tabulator(iris, setup)

```

---

titanic

*Titanic data set*


---

**Description**

Titanic data set

**Usage**

```
titanic(col_select = NULL)
```

**Arguments**

`col_select` (character vector): Columns to select.

**Value**

data frame

## Examples

```
titanic(c("PassengerId", "Age", "Pclass", "Fare")) |>
  head()
```

---

|                  |                            |
|------------------|----------------------------|
| trigger_download | <i>Download table data</i> |
|------------------|----------------------------|

---

## Description

Download table data

## Usage

```
trigger_download(ctx, type = c("csv", "json", "xlsx"), file_name = NULL)
```

## Arguments

|           |  |
|-----------|--|
| ctx       | A <a href="#">tabulatorContext()</a> object.         |
| type      | File format.   |
| file_name | File name. Set to "data.{type}" if file_name = NULL. |

## Details

If you want to support xlsx downloads, you need to include the [sheetjs](#) HTML dependency with `tabulator(..., sheetjs = TRUE)`.

## Value

A [tabulatorContext\(\)](#) object

## Examples

```
tabulatorContext("table") |>
  trigger_download("csv", "table-data.csv")
```



---

|                  |                         |
|------------------|-------------------------|
| trigger_get_data | <i>Submit data to R</i> |
|------------------|-------------------------|

---

**Description**

Submit data to R

**Usage**

```
trigger_get_data(ctx)
```

**Arguments**

ctx           A `tabulatorContext()` object.

**Value**

A `tabulatorContext()` object

**Examples**

```
tabulatorContext("table") |>  
  trigger_get_data()
```

---

|                        |                               |
|------------------------|-------------------------------|
| trigger_get_sheet_data | <i>Submit sheet data to R</i> |
|------------------------|-------------------------------|

---

**Description**

Submit sheet data to R

**Usage**

```
trigger_get_sheet_data(ctx)
```

**Arguments**

ctx           A `tabulatorContext()` object.

**Value**

A `tabulatorContext()` object

**Examples**

```
tabulatorContext("table") |>  
  trigger_get_sheet_data()
```

---

undo

*Undo changes*

---

**Description**

Undo changes

**Usage**

```
undo(ctx)
```

**Arguments**

ctx            A `tabulatorContext()` object.

**Value**

A `tabulatorContext()` object

**Examples**

```
tabulatorContext("table") |>  
  undo()
```

# Index

`add_row`, 2

`delete_selected_rows`, 3

`for_each_col`, 4

JS, 13, 16

`readr::read_csv()`, 27

`redo`, 4

`renderTabulator (rtabulator-shiny)`, 5

`rtabulator-shiny`, 5

`set_calculation`, 6

`set_column_defaults`, 6

`set_editor`, 8

`set_formatter_color`, 9

`set_formatter_datetime`, 10

`set_formatter_datetime()`, 27

`set_formatter_html`, 11

`set_formatter_image`, 12

`set_formatter_link`, 13

`set_formatter_money`, 14

`set_formatter_plaintext`, 15

`set_formatter_progress`, 16

`set_formatter_star`, 17

`set_formatter_textarea`, 18

`set_formatter_tick_cross`, 19

`set_formatter_toggle_switch`, 19

`set_formatter_traffic_light`, 21

`set_header_filter`, 22

`set_multi_column_header`, 23

`set_options_group_by`, 23

`set_options_pagination`, 24

`set_tooltip`, 25

`spreadsheet_def`, 26

`tabulator`, 27

`tabulator()`, 4, 6–25, 28, 31

`tabulator_options`, 29

`tabulator_options()`, 26, 27

`tabulatorContext`, 28

`tabulatorContext()`, 3–5, 32–34

`tabulatorOutput (rtabulator-shiny)`, 5

`tabulatorOutput()`, 28

titanic, 31

`trigger_download`, 32

`trigger_download()`, 27

`trigger_get_data`, 33

`trigger_get_sheet_data`, 33

undo, 34