

tinytable tutorial

Table of contents

1	Output formats	2
2	Style	2
2.1	Cells	3
2.2	Rows and columns	4
3	Headers	5
4	Captions and cross-references	5
5	Customization: HTML with Bootstrap	6
6	Customization: LaTeX with tabulararray	6

tinytable is easy to use:

```
# library(tinytable)
pkgload::load_all()
```

i Loading tinytable

```
x <- mtcars[1:4, 1:5]

tinytable(x)
```

mpg	cyl	disp	hp	drat
21	6	160	110	3.9
21	6	160	110	3.9
22.8	4	108	93	3.85
21.4	6	258	110	3.08

1 Output formats

`tinytable` can produce tables in HTML, Markdown, or LaTeX (PDF) format. To choose, we use the `output` argument:

```
tinytable(x, output = "html")
tinytable(x, output = "latex")
tinytable(x, output = "markdown")
```

When calling `tinytable` from a Quarto or Rmarkdown document, `tinytable` detects the output format automatically and generates an HTML or LaTeX table as appropriate. This means that we do not need to explicitly specify the `output` format.

2 Style

The `style()` function allows us to apply visual styles to our table. This includes customizing features such as:

- Text color
- Background color
- Widths
- Heights
- Alignment
- Text Wrapping
- Column and Row Spacing
- Cell Merging
- Multi-row or column spans
- Border Styling
- Font Styling
- Header Customization

The main arguments of the `style()` function are rather self-explanatory:

- `i` row numbers: integer vector or `NULL` to style all rows.
- `j` column numbers: integer vector or `NULL` to style all columns.
- `color`: text color
- `background`: background color
- `bold`: bold text
- `italic`: bold text
- `align`: horizontal alignment

In addition, `style()` accepts two more arguments which allow unlimited possibilities for customizing every possible aspect of your tables in HTML or LaTeX (PDF):

- `latex = latexOptions()` for `tabulararray`
- `html = htmlOptions()` for Bootstrap

We discuss these extra arguments near the end of this page.

2.1 Cells

To style individual cells, we use the `style_cell()` function. The first two arguments (`i` and `j`) identify the positions of the cells of interest, by row and column numbers respectively. To style a cell in the 2nd row and 3rd column, we can do:

```
tinytable(x) |>
  style(
    i = 2,
    j = 3,
    background = "black",
    color = "white",
    bold = TRUE,
    italic = TRUE)
```

mpg	cyl	disp	hp	drat
21	6	<i>160</i>	110	3.9
21	6	160	110	3.9
22.8	4	108	93	3.85
21.4	6	258	110	3.08

The `i` and `j` accept vectors of integers to modify several cells at once:

```
tinytable(x) |> style(2, c(1, 3), background = "olive")
```

mpg	cyl	disp	hp	drat
21	6	160	110	3.9
21	6	160	110	3.9
22.8	4	108	93	3.85
21.4	6	258	110	3.08

We can style all cells in a table by omitting the `i` and `j` arguments:

```
tinytable(x) |> style(background = "black", color = "white", bold = TRUE)
```

mpg	cyl	disp	hp	drat
21	6	160	110	3.9
21	6	160	110	3.9
22.8	4	108	93	3.85
21.4	6	258	110	3.08

2.2 Rows and columns

We can style entire rows by omitting the `j` argument, or style entire columns by omitting the `i` argument:

```
tinytable(x) |> style(i = 1:2, color = "blue")
```

mpg	cyl	disp	hp	drat
21	6	160	110	3.9
21	6	160	110	3.9
22.8	4	108	93	3.85
21.4	6	258	110	3.08

```
tinytable(x) |> style(j = c(2, 4), bold = TRUE)
```

mpg	cyl	disp	hp	drat
21	6	160	110	3.9
21	6	160	110	3.9
22.8	4	108	93	3.85
21.4	6	258	110	3.08

3 Headers

The header can be omitted from the table by deleting the column names in the `x` data frame:

```
k <- x
colnames(k) <- NULL
tinytable(k)
```

21	6	160	110	3.9
21	6	160	110	3.9
22.8	4	108	93	3.85
21.4	6	258	110	3.08

4 Captions and cross-references

```
tinytable(x, caption = "Data about cars.")
```

Table 1: Data about cars.

mpg	cyl	disp	hp	drat
21	6	160	110	3.9
21	6	160	110	3.9
22.8	4	108	93	3.85
21.4	6	258	110	3.08

TODO: Cross-references

5 Customization: HTML with Bootstrap

```
knitr::opts_chunk$set(eval = knitr::is_html_output())
```

```
tinytable(x, html = htmlOptions(class = "table table-dark"))
```

6 Customization: LaTeX with tabularray

```
knitr::opts_chunk$set(eval = knitr::is_latex_output())
```

```
tinytable(x, latex = latexOptions(theme = "void", hlines = ""))
```

```
knitr::opts_chunk$set(eval = TRUE)
```