`tinytable` tutorial

Table of contents

This tutorial is available in two versions:

- PDF
- HTML

1 `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

2 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.

3 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 tabularray
- `html = htmlOptions()` for Bootstrap

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

3.1 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	***160***	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**

3.2 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

4 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

5 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

6 HTML customization

Warning: The `html` argument is only used for HTML tables, and it is ignored when `tinytable` creates LaTeX tables for PDF files. You can skip this section if you are reading the PDF version of this tutorial.

6.1 Themes

The Bootstrap framework provides a number of built-in themes to style tables. To use them, we call `htmlOptions()` with the `class` argument, and we specify the Bootstrap class. A list of available Bootstrap classes can be found here: https://getbootstrap.com/docs/5.3/content/tables/

For example, to produce a "dark" table, we use the `table-dark` class:

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

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 also combine several Bootstrap classes. Here, we get a "striped" table with the "warning" color:

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

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

6.2 CSS

For manual customization, we can also specify our own CSS class.

7 LaTeX / PDF customization

Warning: The `latex` argument is only used for LaTeX/PDF tables, and it is ignored when `tinytable` creates HTML tables for the web. You can skip this section if you are reading the HTML version of this tutorial.

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

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