

tinytable

Table of contents

| | | |
|----------|---|-----------|
| 1 | Tiny Tables | 2 |
| 1.1 | Output formats | 2 |
| 1.2 | Themes | 3 |
| 1.3 | Alignment | 3 |
| 1.4 | Width | 4 |
| 1.5 | Line breaks and text wrapping | 4 |
| 1.6 | Captions and cross-references | 5 |
| 2 | Style | 6 |
| 2.1 | Colors, lines, space, font, spans, etc. | 6 |
| 2.2 | Cells, rows, columns | 7 |
| 2.3 | Spanning cells | 8 |
| 2.4 | Headers | 9 |
| 3 | HTML customization | 10 |
| 4 | LaTeX / PDF customization | 10 |
| 4.1 | tabularray keys | 10 |

`tinytable` is a small but powerful R package to draw HTML, LaTeX, PDF, Markdown, and Typst tables. The interface is minimalist, but it gives users direct and convenient access to powerful frameworks to create endlessly customizable tables.

This tutorial introduces the main functions of the package. It is available in two versions:

- [PDF](#)
- [HTML](#)

1 Tiny Tables

```
# library(tinytable)
library(data.table)

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

tt(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 |

1.1 Output formats

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

```
tt(x, output = "html")
tt(x, output = "latex")
tt(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.

1.2 Themes

`tinytable` offers a few basic themes out of the box: “default”, “striped”, “grid”, “void.” Those themes can be applied with the `theme` argument of the `tt()` function. As we will see below, it is easy to go much beyond those basic settings to customize your own tables. Here we only illustrate a few of the simplest settings:

```
tt(x, theme = "striped")
```

| 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 |

```
tt(x, theme = "grid")
```

| 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 |

```
tt(x, theme = "void")
```

| 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 |

1.3 Alignment

To align columns, we use a single string, where each letter represents a column:

```
tt(x, align = "ccrrl")
```

| 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 |

1.4 Width

The `width` argument accepts a number between 0 and 1, indicating what proportion of the linewidth the table should cover:

```
tt(x, width = 0.5)
```

| 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 |

```
tt(x, width = 1)
```

| 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 |

1.5 Line breaks and text wrapping

When the `width` argument is specified and a cell includes long text, the text is automatically wrapped to match the table.

```
d <- data.frame(
  a = "Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque",
  b = "dicta sunt explicabo. Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui placeat aut qui vel illum, qui dolorem eum fugiat qui voluptatem qui aperiam, dolorum nostrum et ipsam vel qui laudantium ut qui illo esse aliquam reprehenderit sed, totam rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicabo."
)
tt(d, width = 3/4)
```

| a | b |
|---|---|
| Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae | dicta sunt explicabo. Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos |

Manual line breaks work slightly different in LaTeX (PDF) or HTML. This table shows the two strategies. For HTML, we insert a `
` tag. For LaTeX, we wrap the string in curly braces {}, and then insert two (escaped) backslashes: `\\`

```
d <- data.table(
  `LaTeX line break` = "{Sed ut \\ \\ perspiciatis unde}",
  `HTML line break` = "dicta sunt<br> explicabo. Nemo"
)
tt(d, width = 1)
```

| LaTeX line break | HTML line break |
|-----------------------------|--------------------------------|
| Sed ut perspiciatis unde | dicta sunt explicabo. Nemo |

1.6 Captions and cross-references

```
tt(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

2 Style

`tinytable` exports three styling functions.

1. `style_tt()` is a general interface to frequently used style choices which works for both HTML and LaTeX (PDF): colors, font style and size, row and column spans, etc.
2. `style_tabularray()` is a specialized interface which allows users to use the [extraordinarily powerful `tabularray` package](#) to customize LaTeX tables.
3. `style_bootstrap()` is a specialized interface which allows users to use the [powerful `Bootstrap framework`](#) to customize HTML tables.

2.1 Colors, lines, space, font, spans, etc.

These functions can be used to customize rows, columns, or individual cells. They control many features, including:

- 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 `style_*()` functions can modify individual cells, or entire columns and rows. The portion of the table that is styled is determined by the `i` (rows) and `j` (columns) arguments.

2.2 Cells, rows, columns

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

```
tt(x) |>
  style_tt(
    i = 2,
    j = 3,
    background = "black",
    color = "white")
```

| 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:

```
tt(x) |>
  style_tt(
    i = 2:3,
    j = c(1, 3, 4),
    italic = TRUE,
    color = "red")
```

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

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

```
tt(x) |> style_tt(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 |

We can style entire rows by omitting the `j` argument:

```
tt(x) |> style_tt(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 |

We can style entire columns by omitting the `i` argument:

```
tt(x) |> style_tt(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 |

2.3 Spanning cells

Sometimes, it can be useful to make a cell stretch across multiple columns, for example when we want to insert a label. To achieve this, we can use the `colspan` argument. Here, we make the 2nd cell of the 2nd row stretch across three columns:


```
tt(x)|> style_tt(
  i = 2, j = 2,
  colspan = 3,
  align = "c",
  color = "white",
  background = "black")
```

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

Here is the original table for comparison:

```
tt(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 |

2.4 Headers

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

```
k <- x
colnames(k) <- NULL
tt(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 |

3 HTML customization

Warning: The HTML customization options described in this section are not available for LaTeX (or PDF) documents. Please refer to the web documentation to view this tutorial.

4 LaTeX / PDF customization

```
inner <- "  
  hlines = {white},  
  vlines = {white},  
  cell{1,6}{odd} = {teal7},  
  cell{1,6}{even} = {green7},  
  cell{2,4}{1,4} = {red7},  
  cell{3,5}{1,4} = {purple7},  
  cell{2}{2} = {r=4,c=2}{c,azure7},  
  "  
mtcars[1:5, 1:4] |>  
  tt(theme = "void") |>  
  style_tabularray(inner = inner)
```

| mpg | cyl | disp | hp |
|------|-----|------|-----|
| 21 | 6 | | 110 |
| 21 | | | 110 |
| 22.8 | | | 93 |
| 21.4 | | | 110 |
| 18.7 | 8 | 360 | 175 |

4.1 tabularray keys

Inner specifications:

| Key | Description and Values | Initial Value |
|-----------------------|---|---------------|
| <code>rulesep</code> | space between two hlines or vlines | 2pt |
| <code>stretch</code> | stretch ratio for struts added to cell text | 1 |
| <code>abovesep</code> | set vertical space above every row | 2pt |
| <code>belowsep</code> | set vertical space below every row | 2pt |

| Key | Description and Values | Initial Value |
|-----------------|---|----------------|
| rowsep | set vertical space above and below every row | 2pt |
| leftsep | set horizontal space to the left of every column | 6pt |
| rightsep | set horizontal space to the right of every column | 6pt |
| colsep | set horizontal space to both sides of every column | 6pt |
| hspan | horizontal span algorithm: default , even , or minimal | default |
| vspan | vertical span algorithm: default or even | default |
| baseline | set the baseline of the table | m |

Outer specifications:

| Key | Description and Values | Initial Value |
|-----------------|--|---------------|
| baseline | set the baseline of the table | m |
| long | change the table to a long table | None |
| tall | change the table to a tall table | None |
| expand | you need this key to use verb commands | None |

Cells:

| Key | Description and Values | Initial Value |
|---------------|--|---------------|
| halign | horizontal alignment: l (left), c (center), r (right) or j (justify) | j |
| valign | vertical alignment: t (top), m (middle), b (bottom), h (head) or f (foot) | t |
| wd | width dimension | None |
| bg | background color name | None |
| fg | foreground color name | None |
| font | font commands | None |
| mode | set cell mode: math , imath , dmath or text | None |
| cmd | execute command for the cell text | None |
| preto | prepend text to the cell | None |
| appto | append text to the cell | None |
| r | number of rows the cell spans | 1 |
| c | number of columns the cell spans | 1 |

Rows:

| Key | Description and Values | Initial Value |
|-----------------|--|---------------|
| halign | horizontal alignment: l (left), c (center), r (right) or j (justify) | j |
| valign | vertical alignment: t (top), m (middle), b (bottom), h (head) or f (foot) | t |
| ht | height dimension | None |
| bg | background color name | None |
| fg | foreground color name | None |
| font | font commands | None |
| mode | set mode for row cells: math , imath , dmath or text | None |
| cmd | execute command for every cell text | None |
| abovesep | set vertical space above the row | 2pt |
| belowsep | set vertical space below the row | 2pt |
| rowsep | set vertical space above and below the row | 2pt |
| preto | prepend text to every cell (like > specifier in rowspec) | None |
| appto | append text to every cell (like < specifier in rowspec) | None |

Columns:

| Key | Description and Values | Initial Value |
|-----------------|--|---------------|
| halign | horizontal alignment: l (left), c (center), r (right) or j (justify) | j |
| valign | vertical alignment: t (top), m (middle), b (bottom), h (head) or f (foot) | t |
| wd | width dimension | None |
| co | coefficient for the extendable column (X column) | None |
| bg | background color name | None |
| fg | foreground color name | None |
| font | font commands | None |
| mode | set mode for column cells: math , imath , dmath or text | None |
| cmd | execute command for every cell text | None |
| leftsep | set horizontal space to the left of the column | 6pt |
| rightsep | set horizontal space to the right of the column | 6pt |
| colsep | set horizontal space to both sides of the column | 6pt |
| preto | prepend text to every cell (like > specifier in colspec) | None |
| appto | append text to every cell (like < specifier in colspec) | None |

hlines:

| Key | Description and Values | Initial Value |
|-----------------|--|---------------|
| dash | dash style: solid , dashed or dotted | solid |
| text | replace hline with text (like ! specifier in rowspec) | None |
| wd | rule width dimension | 0.4pt |
| fg | rule color name | None |
| leftpos | crossing or trimming position at the left side | 1 |
| rightpos | crossing or trimming position at the right side | 1 |
| endpos | adjust leftpos/rightpos for only the leftmost/rightmost column | false |

vlines:

| Key | Description and Values | Initial Value |
|-----------------|---|---------------|
| dash | dash style: solid , dashed or dotted | solid |
| text | replace vline with text (like ! specifier in colspec) | None |
| wd | rule width dimension | 0.4pt |
| fg | rule color name | None |
| abovepos | crossing or trimming position at the above side | 0 |
| belowpos | crossing or trimming position at the below side | 0 |