`tbl.typ`: a `tbl`-like preprocessor for Typst and `tablex`

Version 0.0.2

Max Rees

2023

Contents

1. Introduction

Typst [1] is "a new markup-based typesetting system that is powerful and easy to learn." While Typst provides a built-in `table()` function, it does not currently support more advanced features such as row spans and column spans, fine-grain control of borders, or complex cell alignments. Pg Biel's `tablex` project [2] provides many of these features. However, it remains the case that writing a table using either `table()` or `tablex()` can require rather verbose syntax.

The `tbl.typ` project is an effort to allow the expression of rich tables in Typst using a more terse syntax. This syntax comes from a UNIX heritage: the `tbl` preprocessor which designed for use with the traditional TROFF typesetting system [3] [4] [5]. Important differences between the syntax of traditional `tbl` and `tbl.typ` are noted later in this document. The goal of this project is to support many traditional `tbl` features in a sensible manner (i.e. not pixel-for-pixel or bug compatible). Some of these features are unique to `tbl.typ` and are not easily reproduced in either `table()` or `tablex()` alone.

2. Usage

1. Make sure you are using Typst version 0.5.0.
2. Put `tablex.typ` version 0.0.2 and `tbl.typ` in your `TYPST_ROOT`.
3. Add the following code to the top of your `.typ` file:

```
#import "/tbl.typ"
#show: tbl.template
```

The basic format of a table when using `tbl.typ` is the following:

````
```tbl
Format specifications .
Data
```
````

The two main components of this syntax are:

- *Format specifications*. This describes the layout of the table in terms of the number and style of columns for each row.

  The last line of the format specifications must end in a period (`.`). This is the separator between the two sections.

- *Data*. This is the content that will fill each cell of the table. Generally every input line in this section corresponds to a row in the table, though there are exceptions noted later. Cells are separated by the `tab` option which defaults to a TAB character.

3. Region options

In addition to the overall table syntax itself, you may specify *region options* that control the parsing and styling of the table as a whole using a "show-everything" rule prior to the tables you would like to control. For example:

```
#show: tbl.template.with(
  allbox: true,
  tab: "|",
)
```

You must provide at least one of these rules somewhere in your document before your first table (even if no options are specified); otherwise the table(s) will not be rendered.

The following options are recognized:

`align`	How to align the table as a whole. *Default:* `left`
`auto-lines`, `allbox`	Like `box`, but also draw a line between every cell if `true`. This is the same option from `tablex`. *Default:* `false` *cf. Example 11, 12, 13, 14.*
`box`, `frame`	If `true`, draw a line around the entire table. *Default:* `false` *cf. Example 1, 2, 3, 4, 5.*
`breakable`, `nokeep`	If `true`, the table can span multiple pages if necessary. *Default:* `false`
`center`, `centre`	Aliases for a `align` value of `center`.
`decimalpoint`	The string used to separate the integral part of a number from the fractional part. Used in `N`-classified columns. *Default:* `"."`
`doublebox`, `doubleframe`	Like `box`, but also draw a second line around the entire table if `true`. *Default:* `false` *cf. Example 15.*

`font`

The font for the table. Can be overridden later by the `f(...)` column modifier.

Default: `"Times"`
n.b. all tables in this document are formatted with the New Computer Modern font.

`header-rows`

The number of rows at the beginning of the table to consider part of the "header" for the purposes of `repeat-header`. This option is also controlled by `.TH` rows in the table data.

Default: `1`

`leading`

The vertical spacing / leading to apply to table cells. Can be overridden later by the `v(...)` column modifier.

`macros`

A dictionary of (name, function) pairs that can be used with column modifier `m(...)`.

Default: `(:)`

`mode`

- `"content"`: all table cells are evaluated as `[content blocks]`.
- `"math"`: all table cells are evaluated as `$inline equations$`.

Default: `"content"`
cf. Example 16.

`pad`

This is the padding used for each cell, for use with the Typst `pad` element function. The `left` and `right` keys can be overridden using a numeric column modifier.

Default: `(x: 0.75em, y: 3pt)`
cf. Example 16.

`repeat-header`

If `breakable` is `true` and this option is `true`, then the table header controlled by `header-rows` will be re-displayed on each subsequent page. This option is also controlled by `.TH` rows in the table data.

Default: `false`

`stroke`, `linesize`

How to draw all lines in the table.

Default: `1pt`
cf. Example 16.

`tab` The string delimiter that separates different cells within a given row of the table data.

Default: `"\t"` (a TAB character)
cf. Example 15. Most tables in this document use `"|"` *(a vertical bar) for readability purposes, though this should not be confused with the column classifier of the same name.*

---

4. Format specifications

The format specifications section controls the layout and style of cells within rows and columns of the table.

Each comma or new line of format specification begins a new *row definition*. Within each row definition, encountering a *column classifier* character denotes a new column in the table. The classifier may be followed by any number of *column modifiers*, some of which may have required arguments enclosed in parentheses.

The total number of columns in the table is determined by the row definition with the largest number of columns specified. Any row definitions that have fewer columns than this maximum are assumed to have however many `L` columns at the end to complete the row.

The last row definition in the format specifications determines the layout of that row and all rows for the rest of the table.

Spaces and tabs between any column classifiers or column modifiers are ignored. Column classifier letters and column modifier letters can be given as either uppercase (preferred for column classifiers) or lowercase (preferred for column modifiers). For example:

```
L Rb
Cr n I.
```

This specifies:

- Row 1:
  - Column 1 is left-aligned (`L`)
  - Column 2 is right-aligned (`R`) and bold (`b`)
  - Column 3 is not specified, but will be assumed to be left-aligned
- Row 2 (**and all subsequent rows**):
  - Column 1 is centered (`C`)
  - Column 2 is right-aligned (`r`)
  - Column 3 is numerically-aligned (`n`) and italic (`I`)

4.1. Column classifiers

The following column classifiers are recognized:

`L`	**L**eft align.
`R`	**R**ight align.
`C`	**C**enter align.
`N`	**N**umerically align: all cells with this classifier in the current column are centered with respect to an *alignment point*, which is determined according to the following rules: • One position after the leftmost occurrence of the *non-printing input token* `\&`, if any is present. • Otherwise, the rightmost occurrence of the `decimalpoint` string that immediately precedes a digit. • Otherwise, the rightmost digit. • Otherwise, the content is instead centered with respect to the column as a whole. The alignment point is centered horizontally with respect to the column as a whole. *cf. Example 3, 4, 8, 9, 10, 11, 15.*
`A`	**A**lphabetically align: in the current column the widest cell with this classifier is centered and the rest with this classifier are left-aligned with respect to that widest cell. These are sometimes called *sub-columns* because they appear to be indented relative to `L`-classified cells. *cf. Example 10.*
`S`	This cell is column-spanned by the previous cell to the left in the current row. *The corresponding table data entries should be empty.* *cf. Example 4, 5, 11, 13, 15.*
`^` (caret)	This cell is row-spanned by the corresponding cell in the previous row above. *The corresponding table data entries should be empty.* *cf. Example 1.*

`_` (underscore), `-` (hyphen)	This cell contains a vertically-centered horizontal rule. *The corresponding table data entries should be empty.*
`=` (equals sign)	Same as `_`, but draw a double horizontal rule instead. *The corresponding table data entries should be empty.*
`|` (vertical bar)	This classifier does not actually begin a new column, but rather indicates the location of a vertical line. If placed at the beginning of a row definition, the line is drawn to the left of the first cell in that row. Otherwise, it is drawn to the right of the current cell in that row. *cf. Example 1, 3, 4, 5, 8.*

4.2. Column modifiers

The following column modifiers are recognized:

`b`	**B**old text using the Typst `strong` element function.
`d`	**D**own — set the vertical alignment to `bottom`.
`e`	**E**qualize the width of all columns with this modifier to the maximum width among those columns. This overrides modifier `x`.
`f(...)`	**F**ont name to use is given in parentheses. `f(B)` is an alias for the `b` modifier. `f(I)` is an alias for the `i` modifier. `f(BI)` is an alias for providing both of the above modifiers. *cf. Example 13.*
`i`	**I**talicize text using the Typst `emph` element function.
`m(...)`	**M**acro (function) to apply to each corresponding cell. The macros must be scoped using the `macros` region option. The macro currently only receives a single argument: the content of the cell. A future version may also pass the position of the cell in terms of row number and column number.
`o(...)`	Fill c**o**lor for the cell is given in parentheses. *cf. Example 12.*

`p(...)`

Point size of the font is modified according to the argument in parentheses.

If the argument begins with a `+` or `-`, then the argument is added or subtracted respectively with respect to the current size.

The argument may be suffixed by a unit. If no unit is specified, `pt` is assumed. Valid units are:

- `pt`, `p`: points.
- `mm`: millimeters.
- `cm`, `c`: centimeters.
- `in`, `i`: inches.
- `em`, `m`: `1em` corresponds to the current font size.
- `en`, `n`: one *en* equals half of an em.
- `P`: six *picas* equals one inch.
- `M`: 100 of these equals one em.

cf. Example 8, 13, 15.

`t`

Top — set the vertical alignment to `top`.

cf. Example 13.

`u`

"Stagger" the affected cells so that they appear **between** the current row and the previous one above.

cf. Example 7.

`v(...)`

Vertical spacing (leading) is modified according to the argument in parentheses.

The length argument provided is in the same format as `p(...)`, with a default unit of `pt` and `+` / `-` relative adjustments supported.

`w(...)`

Width of the column is guaranteed to be at least as big as the argument in parentheses, which acts as a *minimum width*.

The length argument provided supports the same units as `p(...)`, with a default unit of `en`. However, relative adjustments are **not** supported.

This overrides modifier `x`.

cf. Example 13, 14.

`x`	**E**xpand the width of the column to `1fr`, which will consume all of the remaining horizontal space on the page or in the current container. Applying this modifier to multiple columns will divide that remaining space evenly between them. This overrides modifiers `e` and `w(...)`.
`z`	The corresponding cell is treated as if it has **z**ero width for the purpose of determining the width of its column. *cf. Example 1.*
Number	A number given as a column modifier is interpreted as a `en` length which is used as a *column separation*. This is the distance that separates the end of the current cell's content from the beginning of the next cell's content. If there is a vertical line between the two cells, then it will appear centered on this separation distance. The default column separation is controlled by the sum of the `left` and `right` keys of the `pad` option. When not specified, this defaults to `0.75em + 0.75em`, which traditional TROFF calls `3n`. *cf. Example 14, 15.*

5. Data

Each input line following the terminating `.` of the format specifications creates a new row of data in the table, with each cell separated by the `tab` string.

If a row provides fewer entries than there are columns in the table at that point, then the remaining columns are assumed to be empty. It is an error to provide more entries in a row than there are columns.

5.1. Special input lines

Some input lines do not represent table rows at all:

- A line consisting of only `_` (underscore) draws a horizontal line at that position in the table. This is only useful if `auto-lines` is `false`.

  cf. Example 3, 4, 5, 15, 16.

  Similarly, `=` (equals sign) in TROFF would draw a double horizontal line, but this is not currently supported.

- A line consisting of only `.TH` (period + capital T + capital H) is an *end-of-header* marker. All rows of data that precede it are considered part of the table's header for the purposes of the `header-rows` option. It also sets `repeat-header` to `true`. This is only useful if `breakable` is also `true` and the table spans multiple pages.
- A line consisting of only `.T&` (period + capital T + ampersand) in TROFF marks the beginning of a new set of format specifications to be terminated by `.` and more table data to follow, but this is not currently supported.
- Lines that begin with `.\"` (period + backslash + double quote) are treated as comments and completely ignored.
- Other lines that begin with `.` (period) in TROFF were used as *commands* (*requests* or *macro invocations*), but this cannot be supported for obvious reasons. Any such line is rejected. To have the first cell in a row begin with a period, use a Typst escape (e.g. `\.`).
- Lines that end with `\` (backslash) indicate that the table entry for the current cell continues on the next input line.

  cf. Example 2.

5.2. Table entries

The string representing the cell content is called the *table entry*. Each table entry is evaluated by the Typst `eval` function. By default, they will be evaluated as Typst markup, but you can change the `mode` region option to evaluate them as equations instead.

Any leading or trailing spaces or tabs within a table entry (so long as `tab` is neither) are ignored. The Examples section takes advantage of this in order to improve legibility, but note that making the input look pretty is **not** a requirement: see Example 6.

There are a few important caveats:

- The `eval` function does not have access to anything other than the Typst standard library. This means it is not currently possible to reference variables or functions within a table entry.
- Numerically-aligned cells are split on the alignment point and then evaluated as two separate pieces of content. This may cause unexpected syntax errors if you have Typst markup that spans the alignment point.
- The `tab` string cannot be used **within** a table entry, except by using Typst hexadecimal escape sequences (provided that `tab` is not any of `\`, `u`, `{`, `}`, a letter, or a digit).
- Any occurrences of the string `\&` (backslash-ampersand; known as the *non-printing input token*) in the table entry are removed.

5.3. Special table entries

If a table entry consists of any of the following strings alone (ignoring any spaces or tabs), then they gain a special meaning:

- `_` (a single underscore): Draw a horizontal line through the middle of this otherwise empty cell. The line touches any adjacent vertical lines that are present.

  cf. Example 5, 8, 14.

- `\_` (backslash + underscore): Like `_` above, but the line does **not** touch any adjacent vertical lines, subject to the current column separation.

  cf. Example 14.

- `=` (equals sign): Like `_` above, but draw a double horizontal line.

  cf. Example 14.

- `\=` (backslash + equals sign): Like `=` above, but subject to column separation like `\_` above.

- `\^` (backslash + caret): This cell is row-spanned by the corresponding cell in the previous row above. This is similar to the `^` column classifier, but can be used at an arbitrary point in the table.

  cf. Example 4.

- `\Rx` (backslash + capital R + any character `x`): the single character `x` is repeated enough to fill the cell but does **not** touch any adjacent vertical lines, subject to the current column separation.

  cf. Example 14.

5.4. Text blocks

A table entry can also span multiple input lines by writing it as a *text block*. This consists of beginning the entry with `T{` (capital T + open brace), followed immediately by the end of that input line. All following input lines are collected as part of the text block until a input line that begins with `T}` (capital T + close brace) is encountered. The rest of that input line can provide the remaining entries for that row of the table.

If the cell is subject to the `w(...)` column modifier, then the text block is constrained to the specified width.

Otherwise, a constraining width W is calculated according to the following formula:

$$W = L \times \frac{C}{N - 1}$$

where L is the maximum width of the table based on the container it is in, or the width of the page minus the margins if there is no container; C is the number of columns this text block spans horizontally; and N is the total number of columns in the table.

cf. Example 13, 14.

6. Differences from traditional tbl

- The following features are unique to tbl.typ:
  - **Region options:** align, font, header-rows, leading, macros, mode, pad, repeat-header
  - **Column modifiers:** o(...)
- Region options must be specified using a "show-everything" rule; they cannot be provided within the raw block itself.
- The nospaces option is always in effect and cannot be disabled.
- The nowarn option is not supported. Typst currently does not support displaying text to standard output or error, except by the use of the assert and panic functions. As such, tbl.typ will halt compilation if any issue is detected.
- The linesize option is expected to be a Typst color, length, or stroke; a dimensionless number does not work.
- The tab option may be a multi-character string.
- The alignment point of numerically-centered cells that are in the same column as left-centered or right-centered cells is always centered with respect to the column as a whole (as if the classifier was C), rather than with respect to the widest L or R entry.
- All column modifiers that expect an argument must provide that argument in parentheses.
- The d and t column modifiers adjust the vertical alignment for all table cells, not just those that are vertically spanned. As a result, the default is more consistently middle alignment (or horizon in Typst parlance).
- Nothing special needs to be done to use equations within table entries, though numerically-aligned columns may behave unexpectedly until the delim option is implemented.
- An empty entry in the table data must be given even if the cell is spanned or contains a horizontal line.

7. Known issues

- The following region options are not currently supported:
  - `delim` (GH#1)
  - `expand` (GH#2)
- The following column classifiers are not currently supported:
  - `||` (double vertical line)
- The `x` (e**x**pand) column modifier does not currently constrain the width of text blocks like it should. (GH#7)
- `.T&` in the table data is not currently supported. (GH#4)
- Within text blocks, `.\"` comments are not removed, and other TROFF commands are not rejected. (GH#6)
- A table data row consisting of only `=` (double horizontal line) is not currently supported.

8. Version history

- **Version 0.0.2:** Saturday 10 June 2023
  - *Breaking changes*
    - Region option `tbl-align` has been renamed to `align`. The former is now an undocumented alias for the latter. **This alias will be removed in the next release**.
    - `tablex.typ` is now pulled from `TYPST_ROOT` rather than relative to the current working directory.
  - *New features*
    - New region option: `mode`.
    - New column classifier: `A`.
    - New special table entry: `\Rx`.
    - Line continuations in the table data are now supported.
  - *Bugs fixed*
    - Fix order of operations for column width measurement, especially for class `N` columns. It is no longer necessary to include spurious `e` modifiers.
    - `w(...)` column modifier now places a definitive lower bound on the width of the column. (GH#5)
    - `pad` region option now accepts underspecified input. (GH#3)
    - Fix width of horizontally-spanned cells.
  - *Improvements*
    - Clarify error message for malformed text block close.
    - Clean up and refactor implementation.
    - Add test suite based on existing examples from `README`.
  - *Documentation*
    - Fix `README` compilation with Typst version 0.4.0.
    - Align columns in code for example tables to improve legibility.
    - Annotate a short example table format specification.
    - Document behavior when fewer table entries are provided than expected columns for a particular row.
    - Fix width of renderings for example tables.
    - Clarify lack of `nospaces` and `nowarn` region options.
    - Expand usage instructions.
    - Document more differences and extensions.
- **Version 0.0.1:** Friday 19 May 2023
  - *Initial release*

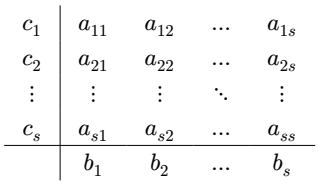
9. Examples

The following examples are formatted with these region options:

```
#show: tbl.template.with(box: true, tab: "|")
```

Example 1: adapted from [4]

```
```tbl
Lz     S   | Rt
Lt  |  Cb  | ^
^   |  Rz    S.
left|      | r
l   |center|
    |     right
```
```

left r
l **center**
right

Example 2: adapted from [5, p. 41]

```
```tbl
        C                C                    C
        L                L                    N.
Fact            |Location            |Statistic
Largest state   |Alaska              |\
591,004 sq. mi.
Smallest state  |Rhode Island        |\
1,212 sq. mi.
Longest river   |Mississippi-Missouri|3,710 mi.
Highest mountain|Mount McKinley, AK  |20,320 ft.
Lowest point    |Death Valley, CA    |-- 282 ft.
```
```

Fact	Location	Statistic
Largest state	Alaska	591,004 sq. mi.
Smallest state	Rhode Island	1,212 sq. mi.
Longest river	Mississippi-Missouri	3,710 mi.
Highest mountain	Mount McKinley, AK	20,320 ft.
Lowest point	Death Valley, CA	– 282 ft.

Example 3: adapted from [4]

```
```tbl
      R | L
      R   N.
software|version
-
     AFL|2.39b
    Mutt|1.8.0
    Ruby|1.8.7.374
TeX Live|2015
```
```

software	version
AFL	2.39b
Mutt	1.8.0
Ruby	1.8.7.374
TeX Live	2015

Example 4: adapted from [5, p. 43]

```
```tbl
Cf(Courier New)  S       S   S
C              | C       S   S
C              | C       S   S
C              | C     | C | C
C              | C     | C | C
L              | N     | N | N.
Composition of Foods

_
Food           |Percent by Weight
\^             |_
\^             |Protein|Fat|Carbo-
\^             |\^     |\^ |hydrate
_
Apples         |  .4   | .5|13.0
Halibut        |18.4   |5.2|...
Lima beans     | 7.5   | .8|22.0
Milk           | 3.3   |4.0| 5.0
Mushrooms      | 3.5   | .4| 6.0
Rye bread      | 9.0   | .6|52.7
```
```

Composition of Foods

Food	Percent by Weight		
	Protein	Fat	Carbo-hydrate
Apples	.4	.5	13.0
Halibut	18.4	5.2	…
Lima beans	7.5	.8	22.0
Milk	3.3	4.0	5.0
Mushrooms	3.5	.4	6.0
Rye bread	9.0	.6	52.7

Example 5: adapted from [5, p. 42]

```
```tbl
C                S               S
C              | C             | C
L              | L             | N.
Major New York Bridges

_
Bridge         |Designer       |Length
_
Brooklyn       |J . A . Roebling|1595
Manhattan      |G . Lindenthal  |1470
Williamsburg   |L . L . Buck    |1600
_
Queensborough  |Palmer &        |1182
               |Hornbostel
_
               |                |1380
Triborough     |O . H . Ammann  |_
               |                |383
_
Bronx Whitestone|O . H . Ammann |2300
Throgs Neck    |O . H . Ammann  |1800
_
George Washington|O . H . Ammann|3500
```
```

Major New York Bridges

Bridge	Designer	Length
Brooklyn	J . A . Roebling	1595
Manhattan	G . Lindenthal	1470
Williamsburg	L . L . Buck	1600
Queensborough	Palmer & Hornbostel	1182
Triborough	O . H . Ammann	1380
		383
Bronx Whitestone	O . H . Ammann	2300
Throgs Neck	O . H . Ammann	1800
George Washington	O . H . Ammann	3500

The following examples are formatted with these region options:

```
#show: tbl.template.with(tab: "|")
```

Example 6: adapted from [4]

````
```tbl
rBclB, rcIl.
r|center|l
ri|ce|le
right|c|left
```
````

r	center	**l**
ri	*ce*	le
right	*c*	left

Example 7: adapted from [3]

````
```tbl
Cf(BI) Cf(BI)            Cf(B)
C      C                 Cu.
n     |n*_#sym.times;_*n|difference
1     |1
2     |4                |3
3     |9                |5
4     |16               |7
5     |25               |9
6     |36               |11
```
````

n	$n \times n$	**difference**
1	1	3
2	4	5
3	9	7
4	16	9
5	25	11
6	36	

Example 8: adapted from [5, p. 42]

````
```tbl
C         C
N p(-2) | N   |.
        |Stack
        |_
       1|46
        |_
       2|23
        |_
       3|15
        |_
       4|6.5
        |_
       5|2.1
        |_
```
````

	Stack
1	46
2	23
3	15
4	6.5
5	2.1

Example 9: adapted from [5, p. 37]

````
```tbl
N.
13
4.2
26.4.12
26.4. 12
26.4 .12
abc
abc\&
43\&3.22
749.12
```
````

N.
13
4.2
26.4.12
26.4. 12
26.4 .12
abc
abc
433.22
749.12

Example 10: adapted from [3]

````
```tbl
Cb            S
L             N
A             N.
Daily energy intake (in MJ)
Macronutrients
Carbohydrates|4.5
Fats         |2.25
Protein      |3
.T&
L             N
A             N.
Mineral
Pu-239       |14.6
_
.T&
L             N.
Total        |\~24.4
```
````

Daily energy intake (in MJ)	
Macronutrients	
Carbohydrates	4.5
Fats	2.25
Protein	3
Mineral	
Pu-239	14.6
Total	~24.4

The following examples are formatted with these region options:

```
#show: tbl.template.with(allbox: true, tab: "|")
```

Example 11: adapted from [5, p. 41]

````
```tbl
C    S      S
C    C      C
N    N      N.
AT&T Common Stock
Year|Price |Dividend
1984|15-20 |\$1.20
   5|19-25 |1.20
   6|21-28 |1.20
   7|20-36 |1.20
   8|24-30 |1.20
   9|29-37 |.30*
```
````

AT&T Common Stock		
Year	Price	Dividend
1984	15-20	$1.20
5	19-25	1.20
6	21-28	1.20
7	20-36	1.20
8	24-30	1.20
9	29-37	.30*

Example 12

```
```tbl
C b o(luma(85%))
C   o(luma(95%)) C.
Grade           |Points
A               |$ >= 510$
B               |$ >= 450$
C               |$ >= 390$
D               |$ >= 330$
```
```

Grade	Points
A	≥ 510
B	≥ 450
C	≥ 390
D	≥ 330

Example 13: adapted from [5, p. 44]

```
```tbl
Cf(I)       S               S
C           Cw(1in)         Cw(1in)
Ltp(9)      Ltp(9)          Ltp(9).
New York Area Rocks
Era         |Formation      |Age (years)
Precambrian|Reading Prong   |>1 billion
Paleozoic   |Manhattan Prong|400 million
Mesozoic    |T{
  #set text(hyphenate: true, overhang: true)

  Newark Basin, incl.
  Stockton, Lockatong, and Brunswick
  formations; also Watchungs
  and Palisades.
T}                          |200 million
Cenozoic    |Coastal Plain  |T{
  #set text(hyphenate: true, overhang: true)
  #set par(justify: true)

  On Long Island 30,000 years;
  Cretaceous sediments redeposited
  by recent glaciation.
T}
```
```

New York Area Rocks		
Era	Formation	Age (years)
Precambrian	Reading Prong	>1 billion
Paleozoic	Manhattan Prong	400 million
Mesozoic	Newark Basin, incl. Stockton, Lockatong, and Brunswick formations; also Watchungs and Palisades.	200 million
Cenozoic	Coastal Plain	On Long Island 30,000 years; Cretaceous sediments redeposited by recent glaciation.

Example 14: adapted from [4]

```
```tbl
Le                Le7 Lw(10).
The fourth line  |_  |line 1
of this column   |=  |line 2
determines       |_ |line 3
the column width.|T{
  This text is too wide to fit into a column of width 17.
T}                  |line 4
T{
  No break here.
T}               |\R.|line 5
```
```

The fourth line		line 1
of this column		line 2
determines		line 3
the column width.	This text is too wide to fit into a column of width 17.	line 4
No break here.	.....	line 5

The following examples are formatted with these region options:

```
#show: tbl.template.with(doublebox: true, tab: " : ")
```

Example 15: adapted from [5, p. 45]

```
```tbl
C b       S       S         S         S
C p(-2)   S       S         S         S
C       | C     | C       | C       | C
C       | C     | C       | C       | C
R 2     | N 2   | N 2     | N 2     | N b.
Readability of Text
Line Width and Leading for 10-Point Type

_
Line    : Set   : 1-Point : 2-Point : 4-Point
Width   : Solid : Leading : Leading : Leading

_
9 Pica  : 93    : --6.0   : --5.3   : --7.1
14 Pica : 450   : --0.6   : --0.3   : --1.7
19 Pica : 5     : --5.1   : 0.0     : --2.0
31 Pica : 3     : --3.8   : --2.4   : --3.6
43 Pica : 5.1   : --90.00 : --5.9   : --8.8
```
```

Readability of Text

Line Width and Leading for 10-Point Type

Line Width	Set Solid	1-Point Leading	2-Point Leading	4-Point Leading
9 Pica	93	–6.0	–5.3	**–7.1**
14 Pica	450	–0.6	–0.3	**–1.7**
19 Pica	5	–5.1	0.0	**–2.0**
31 Pica	3	–3.8	–2.4	**–3.6**
43 Pica	5.1	–90.00	–5.9	**–8.8**

The following examples are formatted with these region options:

```
#show: tbl.template.with(
  tab: "|",
  pad: (bottom: 4pt),
  mode: "math",
  stroke: 0.1pt,
)
```

Example 16: adapted from Discord

````
```tbl
c      | c         c         c           c.
c_1    | a_(11)  | a_(12)  | dots.h    | a_(1 s)
c_2    | a_(21)  | a_(22)  | dots.h    | a_(2 s)
dots.v | dots.v  | dots.v  | dots.down | dots.v
c_s    | a_(s 1) | a_(s 2) | dots.h    | a_(s s)
_
       | b_1     | b_2     | dots.h    | b_s
```
````



10. References

[1] https://typst.app/

[2] Pg Biel, "Typst-tablex." https://github.com/PgBiel/typst-tablex

[3] https://man7.org/linux/man-pages/man1/tbl.1.html

[4] https://man.openbsd.org/tbl.7

[5] L. L. Cherry, and M. E. Lesk, "Tbl – a program to format tables," in *Unix Res. System*, A. G. Hume, and M. D. McIlroy, Eds., vol. 2, 10th ed., Murray Hill, New Jersey 07974: Holt Rinehart & Winston, pp. 35–51. [Online]. Available: https://9p.io/10thEdMan/tbl.pdf