

The `cellprops` package

CSS-like cell and table properties*

Julien “_FrnchFrfgg_” RIVAUD[†]

Released 2018/06/12

1 `cellprops` documentation

This package reworks the internals of `tabular`, `array`, and similar constructs, and adds a `\cellprops` command accepting CSS-like selectors and properties.

It depends on `mdwtab`, `xcolor` and of course `expl3` and `xparse`.

1.1 Example

The code:

```
\[
\cellprops{
  td { padding: 1ex; }
  td:nth-child(4) { color: red; }
  tr:nth-child(2) { color: red; }
  tr:nth-child(1) td:nth-child(4) { color: inherit; }
  td:nth-child(even) { background-color: black!5!white; }
  tr:nth-child(2n+1) { background-color: yellow!30!white; }
  tr:nth-child(odd) td:nth-child(even) {
    background-color: black!20!yellow!20!white;
  }
}
\begin{array}{Tlccc}
\hline
This is text & A_2 & A_3 & A_4 \\
\hline
B1 & This is maths & B_3 & \\
C1 & C_2 & X & Y \\
D1 & D_2 & DX & v \\
E & F & \int_a^b f(t) dt & v \\
\hline
\end{array}\]
```

*This file describes v1.1a, last revised 2018/06/12.

[†]E-mail: frnchfrgg@free.fr

produces:

This is text	A_2	A_3	A_4
B1	<i>This is maths</i>	B_3	
C1	C_2	X	Y
D1	D_2	DX	v
E	F	$\int_a^b f(t)dt$	v

1.2 Usage guide

```

<usage>:      '\cellprops{' [ <selectors> '{' <properties> '}' ]* '}'
<selectors>:  <selector> [, <selectors> ]
<selector>:   [<environment> ' ' ] <element1>
<element1>:   'table' | 'tr' [<pseudo-class>] [ ' ' <element2> ] | <element2>
<element2>:   'td' [<pseudo-class>]
<pseudo-class>: ':nth-child(' <nth> ')'
<nth>:        <number> | 'odd' | 'even' | <number> 'n+' <number>
<properties>: [ <property> ';' ]*
<property>:   'padding: ' <dimension> [ ' ' <dimension> ]* |
               'padding-top: ' <dimension> |
               'padding-left: ' <dimension> |
               'padding-bottom: ' <dimension> |
               'padding-right: ' <dimension> |
               'min-height: ' <dimension> |
               'min-depth: ' <dimension> |
               'min-width: ' <dimension> |
               'text-align: ' ( 'left' | 'right' | 'center' ) |
               'math-mode: ' ( 'text' | 'math' | 'auto' ) |
               'color: ' ( <xcolor-expression> | 'inherit' ) |
               'background-color: ' ( <xcolor-expression> | 'transparent' )

```

Most of these properties are straight-forward. Some explanations:
Here are the supported column types:

- **n**: The most basic cell type, hbox, honoring all properties.
- **l**, **c**, and **r**: Same as **n** but with forced **text-align**.
- **M<coltype>**: Same as **<coltype>** but enforces **math-mode: math**. The net effect is that **Mc** will create a centered column whose contents are in non-display math mode.
- **T<coltype>**: Same as **<coltype>** but enforces **math-mode: text**.

- `p<width>`, `m<width>` and `b<width>`: parbox cell with the corresponding vertical alignment (`\vtop`, `\vcenter` or `\vbox`).
- `*{<count>}{<coltypes>}`: same as in `array` or `mdwtab`.
- You can try to use constructs of `array` or `mdwtab`, but they might alter the function of `cellprops`. Most should be fine though.

The intended usage is to use `n`-type columns and set the properties with CSS, but L^AT_EX-like columns in the preamble are often less verbose.

Details for some properties:

- `math-mode`: `auto` means that the cell will be in math mode in environments `array`, `matrix`, ..., and in text mode in environments like `tabular`, ...
- `background-color` is only painted on the cell, and `transparent` actually means `inherit` except that if all values encountered are `inherit/transparent` no background is painted at all. That means that (currently) you cannot paint a row in some color and rely on transparency to have it bleed through a cell background.
- There are no columns in the CSS object model so you have to use `td:nth-child()` to select a column. Currently, cells spanning several columns actually increase the child count by the number of column they span, so that `nth-child` can still be used to select columns. This is not consistent with the CSS specification.
- Any `:nth-child(An+B)` or `:nth-child(An)` or `:nth-child(B)` is supported, with arbitrary A and B . The performance will slowly degrade the more different A are active (but if in a T_EX group, they will become inactive again when leaving the group). A *big difference* with the CSS specification is that currently n is not enforced non-negative. In particular `:nth-child(2n+8)` *will* match for the second element. This also prevents tricks like `:nth-child(-n+3)`. I am investigating how to handle those without slowing down the general case.

1.3 TODO

Add `border` properties. Test with different table-oriented packages. Improve the documentation, and test more L^AT_EX table constructs and preamble column types. Enforce $n \geq 0$ in `:nth-child` selectors to match the CSS specification.

2 cellprops implementation

```

1 <*package>
2 <@@=cellprops>
3 \ProvidesExplPackage
4   {\ExplFileName}{\ExplFileDate}{\ExplFileVersion}{\ExplFileDescription}
5
6 \RequirePackage{xparse}
7 \RequirePackage{xcolor}

```

2.1 Loading and fixing mdwtab

There is a bug in the command `\colpop` of `mdwtab`: instead of just popping one name in the stack of column sets currently used, it empties it completely because one

`\expandafter` is missing. This is proof that not many package authors really use this API as recommended by Mark WOODING... We thus load `mdwtab` and fix `\colpop`.

```

8 \RequirePackage{mdwtab}
9 \cs_set_nopar:Npn \tab@pop #1 { \tl_set:Nx #1 { \tl_tail:N #1 } }
10
11

```

2.2 Parsing CSS properties

Properties are parsed once at setting time, by expandable parsers that leave definitions in the input stream. All these resulting definitions are saved in a token list that will be expanded when we need the values. The goal is to have multiple token lists for multiple contexts, yet not to do the full parsing dance once per cell.

We first define a generic setter which just uses `\l__cellprops_property_value_<name>_tl` to store the value of the property. We define getters, one that leaves the value in the stream, and one saving the value in a token list.

```

12 \cs_set:Nn \__cellprops_generic_setter:nnn {
13   \exp_not:N \tl_set:Nn
14   \exp_not:c { l__cellprops_property_value_#2_tl }
15   {#1 {#3}}
16 }
17
18 \cs_set_nopar:Nn \__cellprops_get_property:n {
19   \tl_use:c { l__cellprops_property_value_#1_tl }
20 }
21
22 \cs_set_protected_nopar:Nn \__cellprops_get_property:nN {
23   \tl_set_eq:Nc #2 { l__cellprops_property_value_#1_tl }
24 }

```

(End definition for `\l__cellprops_property_value_<name>_tl` and others.)

The control sequence `__cellprops_property_type_<name>:nn` holds the setter for the property *<name>*. It can be set by the following helper:

```

25 \cs_set_protected:Nn \__cellprops_define_properties:nn {
26   \clist_map_inline:nn {#2} {
27     \cs_set:cpn { __cellprops_property_type_##1:nn } {#1}
28   }
29 }

```

(End definition for `__cellprops_property_type_<name>:nn` and `__cellprops_define_properties:nn`.)

Sometimes we need to use a setter right away rather than save its action somewhere. The following helper does that with an x-expansion.

```

30 \cs_set_protected:Nn \__cellprops_use_setter:nn {
31   \use:x {
32     \use:c { __cellprops_property_type_#1:nn } {#1} {#2}
33   }
34 }

```

(End definition for _cellprops_use_setter:nn.)

```

35 \cs_set_protected:Nn \_cellprops\_parse\_properties:Nn {
36   \tl_clear:N #1
37   \seq_set_split:Nnn \l\_tmpa\_seq {;} {#2}
38   \seq_map_inline:Nn \l\_tmpa\_seq {
39     \tl_if_empty:nF {##1} {
40       \exp_args:NNV \seq_set_split:Nnn \l\_tmpb\_seq \c\_colon\_str {##1}
41       \int_compare:nNnTF {\seq_count:N \l\_tmpb\_seq} = \c\_two {
42         \seq_get_left:NN \l\_tmpb\_seq \l\_tmpa\_tl
43         \exp_args:NNV \str_set:Nn \l\_tmpa\_str \l\_tmpa\_tl
44         \seq_get_right:NN \l\_tmpb\_seq \l\_tmpa\_tl
45         \cs_if_exist:cTF { \_cellprops\_property\_type\_l\_tmpa\_str :nn } {
46           \tl\_put\_right:Nx #1 {
47             \exp_args:Nc \exp\_args:NVV
48             { \_cellprops\_property\_type\_l\_tmpa\_str :nn }
49             \l\_tmpa\_str \l\_tmpa\_tl
50           }
51         }{
52           % TODO: ERROR-no property with that name
53         }
54       }{
55         % TODO: ERROR-too many : or none at all
56       }
57     }
58   }
59 }
60
61 \cs_set:Nn \_cellprops\_fourdim\_setter:nn {
62   \_cellprops\_fourdim\_setter\_aux:w
63   {#1}#2~{\q\_no\_value}~{\q\_no\_value}~{\q\_no\_value}~\q\_stop
64 }
65 \cs_set:Npn \_cellprops\_fourdim\_setter\_aux:w #1#2~#3~#4~#5~#6\q\_stop {
66   \_cellprops\_generic\_setter:nnn \exp\_not:n {#1-top} {#2}
67   \quark\_if\_no\_value:nTF {#3} {
68     \_cellprops\_generic\_setter:nnn \exp\_not:n {#1-right} {#2}
69     \_cellprops\_generic\_setter:nnn \exp\_not:n {#1-left} {#2}
70   }{
71     \_cellprops\_generic\_setter:nnn \exp\_not:n {#1-right} {#3}
72     \quark\_if\_no\_value:nTF {#5} {
73       \_cellprops\_generic\_setter:nnn \exp\_not:n {#1-left} {#3}
74     }{
75       \_cellprops\_generic\_setter:nnn \exp\_not:n {#1-left} {#5}
76     }
77   }
78   \quark\_if\_no\_value:nTF {#4} {
79     \_cellprops\_generic\_setter:nnn \exp\_not:n {#1-bottom} {#2}
80   }{
81     \_cellprops\_generic\_setter:nnn \exp\_not:n {#1-bottom} {#4}
82   }
83 }
84
85 \tl\_const:Nn \c\_cellprops\_inherit\_color\_tl { \q\_nil }
86
87 \cs\_new\_nopar:Nn \_cellprops\_color\_setter:nn {

```

```

88 \str_if_eq:nnTF {#2} {inherit} {
89 \__cellprops_generic_setter:nnn \exp_not:n {#1} {\c__cellprops_inherit_color_tl}
90 }{
91 \str_case_x:nnF { \str_range:nnn {#2} {1} {4} } {
92 {rgb} {
93 \__cellprops_generic_setter:nnn \use:n {#1} {
94 \exp_not:n {\color[RGB]} {\str_range:nnn {#2} {5} {-2}}
95 }}
96 {hsl} {
97 \__cellprops_generic_setter:nnn \use:n {#1} {
98 \exp_not:n {\color[Hsb]} {\str_range:nnn {#2} {5} {-2}}
99 }}
100 }{
101 \__cellprops_generic_setter:nnn \exp_not:n {#1} {
102 \color{#2}
103 }
104 }
105 }
106 }
107 \cs_new_nopar:Nn \__cellprops_bgcolor_setter:nn {
108 \str_if_eq:nnTF {#2} {transparent} {
109 \__cellprops_color_setter:nn {#1} {inherit}
110 }{
111 \__cellprops_color_setter:nn {#1} {#2}
112 }
113 }
114
115
116 \__cellprops_define_properties:nn {
117 \__cellprops_generic_setter:nnn \exp_not:n
118 }{
119 padding-top,
120 padding-right,
121 padding-bottom,
122 padding-left,
123 min-height,
124 min-depth,
125 min-width,
126 }
127
128 \__cellprops_define_properties:nn {
129 \__cellprops_fourdim_setter:nn
130 }{
131 padding
132 }
133
134 \__cellprops_define_properties:nn {
135 \__cellprops_generic_setter:nnn \tl_to_str:n
136 }{
137 text-align,
138 math-mode,
139 }
140
141 \__cellprops_define_properties:nn {

```

```

142   \__cellprops_color_setter:nn
143 }{
144   color,
145 }
146
147 \__cellprops_define_properties:nn {
148   \__cellprops_bgcolor_setter:nn
149 }{
150   background-color,
151 }
152
153 \NewDocumentCommand \cellprops { m } {
154   \__cellprops_parse_css:n {#1}
155 }
156
157 \cs_new_protected:Nn \__cellprops_parse_css:n {
158   \__cellprops_parse_css:w #1 \q_mark {\q_nil} \q_stop
159 }
160
161 \tl_new:N \l__cellprops_parse_tmp_tl
162 \NewDocumentCommand \__cellprops_parse_css:w { lmu{\q_stop} } {
163   \quark_if_nil:nF {#2} {
164     \__cellprops_parse_properties:Nn \l__cellprops_parse_tmp_tl {#2}
165     \clist_map_inline:nn {#1} {
166       \__cellprops_parse_css_addprops:nV {##1} \l__cellprops_parse_tmp_tl
167     }
168     \__cellprops_parse_css:w #3 \q_stop
169   }
170 }
171
172
173 \seq_new:N \l__cellprops_parse_selector_seq
174 \tl_new:N \l__cellprops_parse_desc_tl
175
176 \str_const:Nn \c__cellprops_parse_nthchild_str { :nth-child( }
177 \prop_new:N \c__cellprops_parse_replace_prop
178 \prop_put:Nnn \c__cellprops_parse_replace_prop { :first-child } { :nth-child(1) }
179
180 \cs_new_protected:Nn \__cellprops_parse_selector:Nn {
181   \str_set:Nx \l_tmpa_str {#2}

```

Replace some aliases with their meaning:

```

182   \prop_map_inline:Nn \c__cellprops_parse_replace_prop {
183     \use:x {
184       \exp_not:n { \tl_replace_all:Nnn \l_tmpa_str }
185       { \tl_to_str:n { ##1 } } { \tl_to_str:n { ##2 } }
186     }
187   }

```

Replace all spaces by \q_stop to defeat the space removal feature of \seq_set_split:Nnn.

```

188   \tl_replace_all:Nnn \l_tmpa_str {~} {\q_stop}
189   \exp_args:NNVV
190   \seq_set_split:Nnn \l_tmpa_seq \c__cellprops_parse_nthchild_str \l_tmpa_str
191   \seq_pop_left:NN \l_tmpa_seq \l_tmpa_tl

```

And replace them back.

```

192 \tl_replace_all:Nnn \l_tmpa_tl {\q_stop} {~}
193 \seq_clear:N \l__cellprops_parse_selector_seq
194 \seq_put_right:NV \l__cellprops_parse_selector_seq \l_tmpa_tl
195 \seq_map_inline:Nn \l_tmpa_seq {
196 \tl_set:Nn \l_tmpa_tl { ##1 }

```

And replace them back again.

```

197 \tl_replace_all:Nnn \l_tmpa_tl {\q_stop} {~}

```

Now replace the first closing parenthesis by `\q_stop\prg_do_nothing:` to use `\q_stop` as a delimiter for `\seq_set_split:NnV` and thus split at most once. Note that here the space trimming feature is desired for the left part, but not just at the right of the parenthesis, so `\prg_do_nothing:` will act as a guard and will be removed afterwards.

```

198 \tl_replace_once:Nnn \l_tmpa_tl { ) } { \q_stop\prg_do_nothing: }
199 \seq_set_split:NnV \l_tmpa_seq { \q_stop } \l_tmpa_tl
200 \seq_pop_right:NN \l_tmpa_seq \l__cellprops_parse_desc_tl
201 \tl_replace_once:Nnn \l__cellprops_parse_desc_tl { \prg_do_nothing: } {}
202 \seq_get_left:NNT \l_tmpa_seq \l_tmpa_tl {
203 \exp_args:NNV \__cellprops_parse_nth:Nn \l_tmpa_tl \l_tmpa_tl
204 \tl_put_left:Nn \l__cellprops_parse_desc_tl { ) }
205 \tl_put_left:NV \l__cellprops_parse_desc_tl \l_tmpa_tl
206 }
207 \seq_put_right:NV \l__cellprops_parse_selector_seq \l__cellprops_parse_desc_tl
208 }
209 \tl_set:Nx #1 {
210 \exp_args:NNV \seq_use:Nn
211 \l__cellprops_parse_selector_seq \c__cellprops_parse_nthchild_str
212 }
213 }
214
215 \str_const:Nn \c__cellprops_parse_n_str {n}
216 \seq_new:N \l__cellprops_used_nth_factors_seq
217 \cs_new_protected:Nn \__cellprops_parse_nth:Nn {
218 \str_case:nnF {#2} {
219 {even} { \str_set:Nn \l_tmpa_str {2n} }
220 {odd} { \str_set:Nn \l_tmpa_str {2n+1} }
221 }{
222 \str_set:Nn \l_tmpa_str {#2}
223 }
224 \exp_args:NNVV
225 \seq_set_split:Nnn \l_tmpa_seq \c__cellprops_parse_n_str \l_tmpa_str
226 \seq_pop_right:NN \l_tmpa_seq \l_tmpa_tl
227 \int_set:Nn \l_tmpb_int { 0\l_tmpa_tl }
228 \seq_get_left:NNTF \l_tmpa_seq \l_tmpa_tl {
229 \int_set:Nn \l_tmpa_int { 0\l_tmpa_tl }
230 }{
231 \int_zero:N \l_tmpa_int
232 }
233 \int_compare:nNnTF \l_tmpa_int = { 0 } {
234 \tl_set:Nx #1 { \int_use:N \l_tmpb_int }
235 }{
236 \int_set:Nn \l_tmpb_int { \int_mod:nn {\l_tmpb_int} {\l_tmpa_int} }
237 \tl_set:Nx #1 {

```



```

238         \int_use:N \l_tmpa_int \exp_not:V \c__cellprops_parse_n_str
239         + \int_use:N \l_tmpb_int }
240     \seq_put_right:Nx
241     \l__cellprops_used_nth_factors_seq { \int_use:N \l_tmpa_int }
242 }
243 }
244
245 \cs_new_protected:Npn \__cellprops_parse_css_addprops:nV #1 #2 {
246     \__cellprops_parse_selector:Nn \l_tmpa_tl {#1}
247     \tl_set:Nx \l_tmpa_tl { l__cellprops_property_group_\l_tmpa_tl _tl }
248     \tl_if_exist:cF { \l_tmpa_tl } { \tl_clear:c { \l_tmpa_tl } }
249     \tl_put_right:cV { \l_tmpa_tl } #2
250 }
251
252 \cs_set_protected:Nn \__cellprops_recall_properties:n {
253     \tl_if_exist:cT { l__cellprops_property_group_#1_tl } {
254         \tl_use:c { l__cellprops_property_group_#1_tl }
255     }
256     \clist_map_inline:nn { \@currenvir } {
257         \tl_if_exist:cT { l__cellprops_property_group_##1~#1_tl } {
258             \tl_use:c { l__cellprops_property_group_##1~#1_tl }
259         }
260     }
261 }
262
263 \dim_new:N \l__cellprops_colsep_dim
264 \ExplSyntaxOff
265 \cellprops{
266     td {
267         padding: Opt \csname l__cellprops_colsep_dim\endcsname;
268         min-height: Opt;
269         min-depth: Opt;
270         min-width: Opt;
271         text-align: left;
272         math-mode: auto;
273         color: inherit;
274         background-color: transparent;
275     }
276     tr {
277         color: inherit;
278         background-color: transparent;
279     }
280     table {
281         padding: Opt; % No change at load time
282         color: inherit;
283         background-color: transparent;
284     }
285 }
286 \ExplSyntaxOn
287
288 \int_new:N \g__cellprops_row_int
289 \int_new:N \g__cellprops_col_int
290 \bool_new:N \g__cellprops_inrow_bool
291 \bool_gset_false:N \g__cellprops_inrow_bool

```

```

292
293 \box_new:N \l__cellprops_cell_box
294 \skip_new:N \l__cellprops_left_skip
295 \skip_new:N \l__cellprops_right_skip
296 \dim_new:N \g__cellprops_ht_dim
297 \dim_new:N \g__cellprops_dp_dim
298
299 \tl_new:N \l__cellprops_restore_tl
300
301 \dim_new:N \l__cellprops_tablepadding_top_dim
302 \dim_new:N \l__cellprops_tablepadding_bottom_dim
303 \tl_new:N \l__cellprops_color_tl
304 \tl_new:N \l__cellprops_bgcolor_tl
305
306 % To count rows and columns
307 \cs_set_eq:NN \__cellprops_orig_array: \@array
308 \cs_set_protected_nopar:Npn \@array[#1]#2 {
309     \tl_set:Nx \l__cellprops_restore_tl {
310         \bool_if:NTF \g__cellprops_inrow_bool {
311             \exp_not:n {\bool_gset_true:N \g__cellprops_inrow_bool}
312         }{
313             \exp_not:n {\bool_gset_false:N \g__cellprops_inrow_bool}
314         }
315         \exp_not:n { \int_gset:Nn \g__cellprops_row_int }
316         { \int_use:N \g__cellprops_row_int }
317         \exp_not:n { \int_gset:Nn \g__cellprops_col_int }
318         { \int_use:N \g__cellprops_col_int }
319         \exp_not:n { \dim_gset:Nn \g__cellprops_ht_dim }
320         { \dim_use:N \g__cellprops_ht_dim }
321         \exp_not:n { \dim_gset:Nn \g__cellprops_dp_dim }
322         { \dim_use:N \g__cellprops_dp_dim }
323     }
324     \int_gzero:N \g__cellprops_row_int
325     \bool_gset_false:N \g__cellprops_inrow_bool
326     \cs_set_eq:NN \__cellprops_orig_tab@initread: \tab@initread
327     \cs_set_protected_nopar:Npn \tab@initread {
328         \cs_set_eq:NN \tab@initread \__cellprops_orig_tab@initread:
329         \tab@initread
330         \tl_put_left:Nn \tab@tabtext {\int_gincr:N \g__cellprops_col_int}
331     }

```

Zero `\col@sep` but remember its value for the default padding.

```

332 \dim_set_eq:NN \l__cellprops_colsep_dim \col@sep
333 \dim_zero:N \col@sep

```

Also ignore `*extrasep` dimensions that are not part of cellprop interface and should be replaced by CSS equivalents.

```

334 \dim_zero:N \tab@extrasep
335 \group_begin:
336     \__cellprops_recall_properties:n {table}
337     \dim_gset:Nn \g_tmpa_dim { \__cellprops_get_property:n {padding-top} }
338     \dim_gset:Nn \g_tmpb_dim { \__cellprops_get_property:n {padding-bottom} }
339     \__cellprops_update_colors:
340     \tl_gset_eq:NN \g_tmpa_tl \l__cellprops_color_tl
341     \tl_gset_eq:NN \g_tmpb_tl \l__cellprops_bgcolor_tl

```

```

342 \group_end:
343 \dim_set_eq:NN \l__cellprops_tablepadding_top_dim \g_tmpa_dim
344 \dim_set_eq:NN \l__cellprops_tablepadding_bottom_dim \g_tmpb_dim
345 \tl_set_eq:NN \l__cellprops_color_tl \g_tmpa_tl
346 \tl_set_eq:NN \l__cellprops_bgcolor_tl \g_tmpb_tl
347 \__cellprops_recall_properties:n {tr}
348 \__cellprops_orig_array:[{#1}]{#2}
349 \hlx{s[\l__cellprops_tablepadding_top_dim]}
350 }

```

The color inheritance is handled with `\l__cellprops_inherit_color_tl`, `\l__cellprops_color_tl` and `\l__cellprops_bgcolor_tl`. The role of `__cellprops_update_color:Nn` is to set the inherit fallback to the already existing value of #1 then set #1 to the CSS value, which can be the inherit variable.

```

351 \cs_new_protected_nopar:Nn \__cellprops_update_color:Nn {
352   \__cellprops_get_property:nN {#2} \l_tmpc_tl
353   \exp_args:NV \tl_if_eq:NNF \l_tmpc_tl \c__cellprops_inherit_color_tl {
354     \tl_set_eq:NN #1 \l_tmpc_tl
355   }
356 }
357
358 \cs_set_protected_nopar:Nn \__cellprops_update_colors: {
359   \__cellprops_update_color:Nn \l__cellprops_color_tl {color}
360   \__cellprops_update_color:Nn \l__cellprops_bgcolor_tl {background-color}
361 }
362
363 \cs_set_eq:NN \__cellprops_orig_endarray: \endarray
364 \cs_set_nopar:Npn \endarray {
365   \crrc
366   \hlx{s[\l__cellprops_tablepadding_bottom_dim]}
367   \__cellprops_orig_endarray:
368   \tl_use:N \l__cellprops_restore_tl
369 }
370
371 \cs_set_eq:NN \__cellprops_orig_tab@startrow: \tab@startrow
372 \cs_set_nopar:Npn \tab@startrow {
373   \__cellprops_orig_tab@startrow:
374   \bool_if:NF \g__cellprops_inrow_bool {
375     \bool_gset_true:N \g__cellprops_inrow_bool
376     \int_gincr:N \g__cellprops_row_int
377     \int_gset_eq:NN \g__cellprops_col_int \c_one
378     \dim_gzero:N \g__cellprops_ht_dim
379     \dim_gzero:N \g__cellprops_dp_dim
380   }
381 }
382
383 \cs_set_eq:NN \__cellprops_orig_tab@arraycr: \tab@arraycr
384 \cs_set_protected_nopar:Npn \tab@arraycr {
385   \bool_if:NT \g__cellprops_inrow_bool {
386     \bool_gset_false:N \g__cellprops_inrow_bool
387     \__cellprops_fix_valign_end:
388   }
389   \__cellprops_orig_tab@arraycr:
390 }

```

```

391
392
393
394 \cs_set_eq:NN \__cellprops_orig_multicolumn:w \multicolumn
395 \cs_set:Npn \multicolumn#1#2#3 {
396   \__cellprops_orig_multicolumn:w {#1}{#2}{
397     #3
398     \int_gadd:Nn \g__cellprops_col_int {#1}
399   }
400 }
401
402
403 \cs_new_protected_nopar:Nn \__cellprops_fix_valign_end: {
404   \exp_args:NV \tl_if_eq:nnTF \tab@hlstate {b} {
405     \dim_gadd:Nn \tab@endheight {
406       \g__cellprops_dp_dim - \box_dp:N \@arstrutbox
407     }
408   }{
409     \int_compare:nNnT \g__cellprops_row_int = \c_one {
410       \dim_gadd:Nn \tab@endheight {
411         \g__cellprops_ht_dim - \box_ht:N \@arstrutbox
412       }
413     }
414   }
415 }
416
417 \colpush{tabular}
418
419 \coldef n{\tabcoltype{
420   \__cellprops_begincell:n{
421   }{
422     \__cellprops_endcell:
423   }}
424 \coldef l{\tabcoltype{
425   \__cellprops_begincell:n
426   {\__cellprops_use_setter:nn {text-align} {left}}
427   }{
428     \__cellprops_endcell:
429   }}
430 \coldef c{\tabcoltype{
431   \__cellprops_begincell:n
432   {\__cellprops_use_setter:nn {text-align} {center}}
433   }{
434     \__cellprops_endcell:
435   }}
436 \coldef r{\tabcoltype{
437   \__cellprops_begincell:n
438   {\__cellprops_use_setter:nn {text-align} {right}}
439   }{
440     \__cellprops_endcell:
441   }}
442 \coldef M#1{\__cellprops_MTcol:nn {math}{#1}}
443 \coldef T#1{\__cellprops_MTcol:nn {text}{#1}}
444 \cs_new_protected_nopar:Nn \__cellprops_MTcol:nn {

```

```

445 % TODO: error if align not l, c, or r
446 \exp_args:Nx \tabcoltype {
447   \exp_not:N \__cellprops_beginncell:n {
448     \exp_not:n {\__cellprops_use_setter:nn {math-mode} {#1} }
449     \exp_not:n {\__cellprops_use_setter:nn {text-align} } {
450       \str_case:nn {#2} {
451         {l} {left}
452         {c} {center}
453         {r} {right}
454       }
455     }
456   }
457 }{
458   \__cellprops_endcell:
459 }
460 }
461
462 \coldef p#1{\tabcoltype{
463   \__cellprops_begin_par_cell:nn \vtop {#1}
464 }{
465   \__cellprops_end_par_cell:n {}
466 }}
467 \coldef m#1{\tabcoltype{
468   \__cellprops_begin_par_cell:nn {\c_math_toggle_token\vcenter} {#1}
469 }{
470   \__cellprops_end_par_cell:n{\c_math_toggle_token}
471 }}
472 \coldef b#1{\tabcoltype{
473   \__cellprops_begin_par_cell:nn \vbox {#1}
474 }{
475   \__cellprops_end_par_cell:n {}
476 }}
477
478
479 \colpop
480   Handle various :nth-child() forms.
481 \cs_new_protected_nopar:Nn \__cellprops_seq_nthchild:Nn {
482   \seq_clear:N #1
483   \seq_map_inline:Nn \l__cellprops_used_nth_factors_seq {
484     \seq_put_right:Nx #1 {
485       ##1 n + \int_eval:n{\int_mod:nn{#2}{##1}}
486     }
487   }
488   \seq_put_right:Nx #1 { \int_eval:n{#2} }
489 }
490 \cs_new_protected_nopar:Nn \__cellprops_beginncell:n {
491   \__cellprops_begin_raw_cell:n {
492     \__cellprops_seq_nthchild:Nn \l_tmpa_seq { \g__cellprops_row_int }
493     \__cellprops_seq_nthchild:Nn \l_tmpb_seq { \g__cellprops_col_int }
494     \seq_map_inline:Nn \l_tmpa_seq {
495       \__cellprops_recall_properties:n {tr:nth-child(##1)}
496     }
497     \__cellprops_update_colors:

```

```

498     \_cellprops\_recall\_properties:n {td}
499     \_cellprops\_recall\_properties:n {tr~td}
500     \seq\_map\_inline:Nn \l\_tmpb\_seq {
501         \_cellprops\_recall\_properties:n {td:nth-child(##1)}
502     }
503     \seq\_map\_inline:Nn \l\_tmpa\_seq {
504         \_cellprops\_recall\_properties:n {tr:nth-child(##1)~td}
505     }
506     \seq\_map\_inline:Nn \l\_tmpa\_seq {
507         \seq\_map\_inline:Nn \l\_tmpb\_seq {
508             \_cellprops\_recall\_properties:n {tr:nth-child(##1)~
509                 td:nth-child(###1)}
510         }
511     }
512     \_cellprops\_update\_colors:
513     #1
514     \hbox\_set:Nw \l\_cellprops\_cell\_box
515     \str\_case\_x:nnF {\_cellprops\_get\_property:n {math-mode}} {
516         { text } { \tab@btext }
517         { math } { \tab@bmaths }
518     }{% any other treated as |auto|
519         \tab@bgroup
520     }
521 }
522 }
523
524 \cs\_new\_protected\_nopar:Nn \_cellprops\_endcell: {
525     \str\_case\_x:nnF {\_cellprops\_get\_property:n {math-mode}} {
526         { text } { \tab@etext }
527         { math } { \tab@emaths }
528     }{% any other treated as |auto|
529         \tab@egroup
530     }
531     \hbox\_set\_end:
532     \_cellprops\_end\_raw\_cell:
533 }
534
535 \cs\_new\_protected\_nopar:Nn \_cellprops\_begin\_par\_cell:nn {
536     \savenotes
537     \_cellprops\_begin\_raw\_cell:n{
538         \hbox\_set:Nw \l\_cellprops\_cell\_box
539         #1
540         \bgroup
541         \hsize#2\relax
542         \@arrayparboxrestore
543         \global\@minipagetrue
544         \everypar{
545             \global\@minipagefalse
546             \everypar{}
547         }
548         \_cellprops\_recall\_properties:n {td~p}
549     }
550 }
551 \cs\_new\_protected\_nopar:Nn \_cellprops\_end\_par\_cell:n {

```

```

552 \ifhmode\@maybe@unskip\par\fi
553 \unskip
554 \egroup
555 #1
556 \hbox_set_end:
557 \__cellprops_end_raw_cell:
558 \spewnotes\hfil
559 }
560
561 \cs_new_protected_nopar:Nn \__cellprops_begin_raw_cell:n {
562   \group_begin:
563   % Additional init code
564   #1
565   % Install the cell color
566   \__cellprops_update_colors:
567   \tl_use:N \l__cellprops_color_tl
568 }
569
570 \cs_new_protected_nopar:Nn \__cellprops_end_raw_cell: {
571   % Here \l__cellprops_cell_box must contain the contents of the cell
572   %
573   % Handle vpadding and min-height/depth
574   \box_set_ht:Nn \l__cellprops_cell_box {
575     \dim_max:nn
576     {\box_ht:N \l__cellprops_cell_box}
577     {\__cellprops_get_property:n {min-height}}
578     + (\__cellprops_get_property:n {padding-top})
579   }
580   \box_set_dp:Nn \l__cellprops_cell_box {
581     \dim_max:nn
582     {\box_dp:N \l__cellprops_cell_box}
583     {\__cellprops_get_property:n {min-depth}}
584     + (\__cellprops_get_property:n {padding-bottom})
585   }
586   % To fix vertical alignment later
587   \dim_gset:Nn \g__cellprops_ht_dim {
588     \dim_max:nn
589     {\g__cellprops_ht_dim}
590     {\box_ht:N \l__cellprops_cell_box}
591   }
592   \dim_gset:Nn \g__cellprops_dp_dim {
593     \dim_max:nn
594     {\g__cellprops_dp_dim}
595     {\box_dp:N \l__cellprops_cell_box}
596   }
597   % Handle hpadding and halign
598   \skip_set:Nn \l_tmpa_skip {
599     \dim_max:nn
600     {0pt}
601     {\__cellprops_get_property:n {min-width}}
602     - \box_wd:N \l__cellprops_cell_box }
603   }
604   \skip_add:Nn \l_tmpa_skip {
605     Opt plus 1fil

```

```

606 }
607 \skip_set:Nn \l__cellprops_left_skip
608   {\__cellprops_get_property:n {padding-left}}
609 \skip_set:Nn \l__cellprops_right_skip
610   {\__cellprops_get_property:n {padding-right}}
611 \str_case:x:nnF {\__cellprops_get_property:n {text-align}} {
612   { right } {
613     \skip_add:Nn \l__cellprops_left_skip { \l_tmpa_skip }
614   }
615   { center } {
616     \skip_add:Nn \l__cellprops_left_skip { \l_tmpa_skip / 2 }
617     \skip_add:Nn \l__cellprops_right_skip { \l_tmpa_skip / 2 }
618   }
619 }{% any other treated as |left|
620   \skip_add:Nn \l__cellprops_right_skip { \l_tmpa_skip }
621 }
622 \kern\c_zero_dim
623 \tl_if_empty:NF \l__cellprops_bgcolor_tl {
624   \group_begin:
625   % Paint a background with leaders
626   \tl_use:N \l__cellprops_bgcolor_tl % install the color
627   \skip_set:Nn \l_tmpa_skip {
628     \l__cellprops_left_skip
629     + \box_wd:N \l__cellprops_cell_box
630     + \l__cellprops_right_skip
631   }
632   \leaders
633     \vrule
634     \skip_horizontal:N \l_tmpa_skip
635     \skip_horizontal:n {-\l_tmpa_skip}
636   \group_end:
637 }
638 \skip_horizontal:N \l__cellprops_left_skip
639 \box_use_clear:N \l__cellprops_cell_box
640 \skip_horizontal:N \l__cellprops_right_skip
641 \kern\c_zero_dim
642 \group_end:
643 }
644 \end{package}

```