

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 v1.2, 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`.
- `>{<prefix>}` and `<{<suffix>}`: 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(A n +B)` or `:nth-child(A n)` 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(2 n +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 Compatibility

This package has been tested compatible with `diagbox`, `spreadtab`, `colcell`. Compatibility with `longtable` has been specifically taken care of, provided `cellprops` is loaded afterwards.

1.4 TODO

Add `border` properties. Test with different table-oriented packages, and add a test suite with compatibility tests. 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_if_exist:cTF { l__cellprops_property_value_#1_tl } {
24     \tl_set_eq:Nc #2 { l__cellprops_property_value_#1_tl }
25   }{
26     \tl_clear:N #2
27   }
28 }

```

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

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

```

29 \cs_set_protected:Nn \_cellprops\_define\_properties:nn {
30   \clist_map_inline:nn {#2} {
31     \cs_set:cpn { \_cellprops\_property\_type\_##1:nn } {#1}
32   }
33 }

```

(End definition for `_cellprops_property_type_<name>:nn` and `_cellprops_define_properties:nn`.)

`_cellprops_use_setter: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.

```

34 \cs_set_protected:Nn \_cellprops\_use\_setter:nn {
35   \use:x {
36     \use:c {\_cellprops\_property\_type\_#1:nn} {#1} {#2}
37   }
38 }

```

(End definition for `_cellprops_use_setter:nn`.)

```

39 \cs_set_protected:Nn \_cellprops\_parse\_properties:Nn {
40   \tl_clear:N #1
41   \seq_set_split:Nnn \l_tmpa_seq {;} {#2}
42   \seq_map_inline:Nn \l_tmpa_seq {
43     \tl_if_empty:nF {##1} {
44       \exp_args:NNV \seq_set_split:Nnn \l_tmpb_seq \c_colon_str {##1}
45       \int_compare:nNnTF {\seq_count:N \l_tmpb_seq} = \c_two {
46         \seq_get_left:NN \l_tmpb_seq \l_tmpa_tl
47         \exp_args:NNV \str_set:Nn \l_tmpa_str \l_tmpa_tl
48         \seq_get_right:NN \l_tmpb_seq \l_tmpa_tl
49         \cs_if_exist:cTF { \_cellprops\_property\_type\_ \l_tmpa_str :nn } {
50           \tl_put_right:Nx #1 {
51             \exp_args:Nc \exp_args:NVV
52             { \_cellprops\_property\_type\_ \l_tmpa_str :nn }
53             \l_tmpa_str \l_tmpa_tl
54           }
55         }{
56           % TODO: ERROR-no property with that name
57         }
58       }{
59         % TODO: ERROR-too many : or none at all
60       }
61     }
62   }
63 }
64
65 \cs_set:Nn \_cellprops\_fourdim\_setter:nn {
66   \_cellprops\_fourdim\_setter\_aux:w
67   {#1}#2~{\q_no_value}~{\q_no_value}~{\q_no_value}~\q_stop
68 }
69 \cs_set:Npn \_cellprops\_fourdim\_setter\_aux:w #1#2~#3~#4~#5~#6\q_stop {
70   \_cellprops\_generic\_setter:nnn \exp_not:n {#1-top} {#2}
71   \quark_if_no_value:nTF {#3} {
72     \_cellprops\_generic\_setter:nnn \exp_not:n {#1-right} {#2}
73     \_cellprops\_generic\_setter:nnn \exp_not:n {#1-left} {#2}

```

```

74     }{
75         \__cellprops_generic_setter:nnn \exp_not:n {#1-right} {#3}
76         \quark_if_no_value:nTF {#5} {
77             \__cellprops_generic_setter:nnn \exp_not:n {#1-left} {#3}
78         }{
79             \__cellprops_generic_setter:nnn \exp_not:n {#1-left} {#5}
80         }
81     }
82     \quark_if_no_value:nTF {#4} {
83         \__cellprops_generic_setter:nnn \exp_not:n {#1-bottom} {#2}
84     }{
85         \__cellprops_generic_setter:nnn \exp_not:n {#1-bottom} {#4}
86     }
87 }
88
89 \tl_const:Nn \c__cellprops_inherit_color_tl { \q_nil }
90
91 \cs_new_nopar:Nn \__cellprops_color_setter:nn {
92     \str_if_eq:nnTF {#2} {inherit} {
93         \__cellprops_generic_setter:nnn \exp_not:n {#1} {\c__cellprops_inherit_color_tl}
94     }{
95         \str_case:x:nnF { \str_range:nnn {#2} {1} {4} } {
96             {rgb()} {
97                 \__cellprops_generic_setter:nnn \use:n {#1} {
98                     \exp_not:n {\color[RGB]} {\str_range:nnn {#2} {5} {-2}}
99                 }
100             }{hsl()} {
101                 \__cellprops_generic_setter:nnn \use:n {#1} {
102                     \exp_not:n {\color[Hsb]} {\str_range:nnn {#2} {5} {-2}}
103                 }
104             }{
105                 \__cellprops_generic_setter:nnn \exp_not:n {#1} {
106                     \color{#2}
107                 }
108             }
109         }
110     }
111 \cs_new_nopar:Nn \__cellprops_bgcolor_setter:nn {
112     \str_if_eq:nnTF {#2} {transparent} {
113         \__cellprops_color_setter:nn {#1} {inherit}
114     }{
115         \__cellprops_color_setter:nn {#1} {#2}
116     }
117 }
118
119
120 \__cellprops_define_properties:nn {
121     \__cellprops_generic_setter:nnn \exp_not:n
122 }{
123     padding-top,
124     padding-right,
125     padding-bottom,
126     padding-left,
127     min-height,

```

```

128     min-depth,
129     min-width,
130 }
131
132 \__cellprops_define_properties:nn {
133     \__cellprops_fourdim_setter:nn
134 }{
135     padding
136 }
137
138 \__cellprops_define_properties:nn {
139     \__cellprops_generic_setter:nnn \tl_to_str:n
140 }{
141     text-align,
142     math-mode,
143 }
144
145 \__cellprops_define_properties:nn {
146     \__cellprops_color_setter:nn
147 }{
148     color,
149 }
150
151 \__cellprops_define_properties:nn {
152     \__cellprops_bgcolor_setter:nn
153 }{
154     background-color,
155 }
156
157 \NewDocumentCommand \cellprops { m } {
158     \__cellprops_parse_css:n {#1}
159 }
160
161 \cs_new_protected:Nn \__cellprops_parse_css:n {
162     \__cellprops_parse_css:w #1 \q_mark {\q_nil} \q_stop
163 }
164
165 \tl_new:N \l__cellprops_parse_tmp_tl
166 \NewDocumentCommand \__cellprops_parse_css:w { lmu{\q_stop} } {
167     \quark_if_nil:nF {#2} {
168         \__cellprops_parse_properties:Nn \l__cellprops_parse_tmp_tl {#2}
169         \clist_map_inline:nn {#1} {
170             \__cellprops_parse_css_addprops:nV {##1} \l__cellprops_parse_tmp_tl
171         }
172         \__cellprops_parse_css:w #3 \q_stop
173     }
174 }
175
176
177 \seq_new:N \l__cellprops_parse_selector_seq
178 \tl_new:N \l__cellprops_parse_desc_tl
179
180 \str_const:Nn \c__cellprops_parse_nthchild_str { :nth-child( }
181 \prop_new:N \c__cellprops_parse_replace_prop

```

```

182 \prop_put:Nnn \c__cellprops_parse_replace_prop { :first-child } { :nth-child(1) }
183
184 \cs_new_protected:Nn \__cellprops_parse_selector:Nn {
185   \str_set:Nx \l_tmpa_str {#2}

```

Replace some aliases with their meaning:

```

186   \prop_map_inline:Nn \c__cellprops_parse_replace_prop {
187     \use:x {
188       \exp_not:n { \tl_replace_all:Nnn \l_tmpa_str }
189       { \tl_to_str:n { ##1 } } { \tl_to_str:n { ##2 } }
190     }
191   }

```

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

```

192   \tl_replace_all:Nnn \l_tmpa_str {~} {\q_stop}
193   \exp_args:NNVV
194   \seq_set_split:Nnn \l_tmpa_seq \c__cellprops_parse_nthchild_str \l_tmpa_str
195   \seq_pop_left:NN \l_tmpa_seq \l_tmpa_tl

```

And replace them back.

```

196   \tl_replace_all:Nnn \l_tmpa_tl {\q_stop} {~}
197   \seq_clear:N \l__cellprops_parse_selector_seq
198   \seq_put_right:NV \l__cellprops_parse_selector_seq \l_tmpa_tl
199   \seq_map_inline:Nn \l_tmpa_seq {
200     \tl_set:Nn \l_tmpa_tl { ##1 }

```

And replace them back again.

```

201     \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:Nnn` 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.

```

202     \tl_replace_once:Nnn \l_tmpa_tl { ) } { \q_stop\prg_do_nothing: }
203     \seq_set_split:NnV \l_tmpa_seq { \q_stop } \l_tmpa_tl
204     \seq_pop_right:NN \l_tmpa_seq \l__cellprops_parse_desc_tl
205     \tl_replace_once:Nnn \l__cellprops_parse_desc_tl { \prg_do_nothing: } {}
206     \seq_get_left:NNT \l_tmpa_seq \l_tmpa_tl {
207       \exp_args:NNV \__cellprops_parse_nth:Nn \l_tmpa_tl \l_tmpa_tl
208       \tl_put_left:Nn \l__cellprops_parse_desc_tl { ) }
209       \tl_put_left:NV \l__cellprops_parse_desc_tl \l_tmpa_tl
210     }
211     \seq_put_right:NV \l__cellprops_parse_selector_seq \l__cellprops_parse_desc_tl
212   }
213   \tl_set:Nx #1 {
214     \exp_args:NNV \seq_use:Nn
215     \l__cellprops_parse_selector_seq \c__cellprops_parse_nthchild_str
216   }
217 }
218
219 \str_const:Nn \c__cellprops_parse_n_str {n}
220 \seq_new:N \l__cellprops_used_nth_factors_seq
221 \cs_new_protected:Nn \__cellprops_parse_nth:Nn {
222   \str_case:nnF {#2} {

```



```

223     {even} { \str_set:Nn \l_tmpa_str {2n} }
224     {odd} { \str_set:Nn \l_tmpa_str {2n+1} }
225   }{
226     \str_set:Nn \l_tmpa_str {#2}
227   }
228   \exp_args:NNVV
229     \seq_set_split:Nnn \l_tmpa_seq \c__cellprops_parse_n_str \l_tmpa_str
230   \seq_pop_right:NN \l_tmpa_seq \l_tmpa_tl
231   \int_set:Nn \l_tmpb_int { 0\l_tmpa_tl }
232   \seq_get_left:NNTF \l_tmpa_seq \l_tmpa_tl {
233     \int_set:Nn \l_tmpa_int { 0\l_tmpa_tl }
234   }{
235     \int_zero:N \l_tmpa_int
236   }
237   \int_compare:nNnTF \l_tmpa_int = { 0 } {
238     \tl_set:Nx #1 { \int_use:N \l_tmpb_int }
239   }{
240     \int_set:Nn \l_tmpb_int { \int_mod:nn {\l_tmpb_int} {\l_tmpa_int} }
241     \tl_set:Nx #1 {
242       \int_use:N \l_tmpa_int \exp_not:V \c__cellprops_parse_n_str
243       + \int_use:N \l_tmpb_int }
244     \seq_put_right:Nx
245       \l__cellprops_used_nth_factors_seq { \int_use:N \l_tmpa_int }
246   }
247 }
248
249 \cs_new_protected:Npn \__cellprops_parse_css_addprops:nV #1 #2 {
250   \__cellprops_parse_selector:Nn \l_tmpa_tl {#1}
251   \tl_set:Nx \l_tmpa_tl { l__cellprops_property_group_\l_tmpa_tl _tl }
252   \tl_if_exist:cF { \l_tmpa_tl } { \tl_clear:c { \l_tmpa_tl } }
253   \tl_put_right:cV { \l_tmpa_tl } #2
254 }
255
256 \cs_set_protected:Nn \__cellprops_recall_properties:n {
257   \tl_if_exist:cT { l__cellprops_property_group_#1_tl } {
258     \tl_use:c { l__cellprops_property_group_#1_tl }
259   }
260   \clist_map_inline:nn { \@currenvir } {
261     \tl_if_exist:cT { l__cellprops_property_group_##1~#1_tl } {
262       \tl_use:c { l__cellprops_property_group_##1~#1_tl }
263     }
264   }
265 }
266
267 \dim_new:N \l__cellprops_colsep_dim
268 \ExplSyntaxOff
269 \cellprops{
270   td {
271     padding: Opt \csname l__cellprops_colsep_dim\endcsname;
272     min-height: Opt;
273     min-depth: Opt;
274     min-width: Opt;
275     text-align: left;
276     math-mode: auto;

```

```

277         color: inherit;
278         background-color: transparent;
279     }
280     tr {
281         color: inherit;
282         background-color: transparent;
283     }
284     table {
285         padding: Opt; % No change at load time
286         color: inherit;
287         background-color: transparent;
288     }
289 }
290 \ExplSyntaxOn
291
292 \int_new:N \g__cellprops_row_int
293 \int_new:N \g__cellprops_col_int
294 \bool_new:N \g__cellprops_inrow_bool
295 \bool_gset_false:N \g__cellprops_inrow_bool
296
297 \box_new:N \l__cellprops_cell_box
298 \skip_new:N \l__cellprops_left_skip
299 \skip_new:N \l__cellprops_right_skip
300 \dim_new:N \g__cellprops_ht_dim
301 \dim_new:N \g__cellprops_dp_dim
302
303 \tl_new:N \l__cellprops_restore_tl
304
305 \dim_new:N \l__cellprops_tablepadding_top_dim
306 \dim_new:N \l__cellprops_tablepadding_bottom_dim
307 \tl_new:N \l__cellprops_color_tl
308 \tl_new:N \l__cellprops_bgcolor_tl
309
310 % To count rows and columns
311 \cs_new_protected:Nn \__cellprops_array_init: {
312     \tl_set:Nx \l__cellprops_restore_tl {
313         \bool_if:NTF \g__cellprops_inrow_bool {
314             \exp_not:n {\bool_gset_true:N \g__cellprops_inrow_bool}
315         }{
316             \exp_not:n {\bool_gset_false:N \g__cellprops_inrow_bool}
317         }
318         \exp_not:n { \int_gset:Nn \g__cellprops_row_int }
319         { \int_use:N \g__cellprops_row_int }
320         \exp_not:n { \int_gset:Nn \g__cellprops_col_int }
321         { \int_use:N \g__cellprops_col_int }
322         \exp_not:n { \dim_gset:Nn \g__cellprops_ht_dim }
323         { \dim_use:N \g__cellprops_ht_dim }
324         \exp_not:n { \dim_gset:Nn \g__cellprops_dp_dim }
325         { \dim_use:N \g__cellprops_dp_dim }
326     }
327     \int_gzero:N \g__cellprops_row_int
328     \bool_gset_false:N \g__cellprops_inrow_bool
329     \cs_set_eq:NN \__cellprops_orig_tab@readpreamble:n \tab@readpreamble
330     \cs_set_eq:NN \tab@readpreamble \__cellprops_readpreamble:n

```

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

```
331 \dim_set_eq:NN \l__cellprops_colsep_dim \col@sep
332 \dim_zero:N \col@sep
```

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

```
333 \dim_zero:N \tab@extrasep
334 \group_begin:
335   \__cellprops_recall_properties:n {table}
336   \dim_gset:Nn \g_tmpa_dim { \__cellprops_get_property:n {padding-top} }
337   \dim_gset:Nn \g_tmpb_dim { \__cellprops_get_property:n {padding-bottom} }
338   \__cellprops_update_colors:
339   \tl_gset_eq:NN \g_tmpa_tl \l__cellprops_color_tl
340   \tl_gset_eq:NN \g_tmpb_tl \l__cellprops_bgcolor_tl
341 \group_end:
342 \dim_set_eq:NN \l__cellprops_tablepadding_top_dim \g_tmpa_dim
343 \dim_set_eq:NN \l__cellprops_tablepadding_bottom_dim \g_tmpb_dim
344 \tl_set_eq:NN \l__cellprops_color_tl \g_tmpa_tl
345 \tl_set_eq:NN \l__cellprops_bgcolor_tl \g_tmpb_tl
346 \__cellprops_recall_properties:n {tr}
347 }
348
349 \cs_set_nopar:Nn \__cellprops_array_startcontent: {
350   \hlx{s[\l__cellprops_tablepadding_top_dim]}
351 }
352
353 \cs_set_protected_nopar:Nn \__cellprops_readpreamble:n {
354   \cs_set_eq:NN \tab@readpreamble \__cellprops_orig_tab@readpreamble:n
355   \tl_put_left:Nn \tab@multicol {\__cellprops_startrow:}
356   \tl_put_left:Nn \tab@tabtext {\int_gincr:N \g__cellprops_col_int}
357   \tab@readpreamble{#1}
358   \exp_args:Nx \tab@preamble
359   { \exp_not:N\__cellprops_startrow: \the\tab@preamble \exp_not:N\__cellprops_endrow: }
360 }
361
362
363 \cs_set_eq:NN \__cellprops_orig_array:w \@array
364 \cs_set_protected_nopar:Npn \@array[#1]#2 {
365   \__cellprops_array_init:
366   \__cellprops_orig_array:w [#1]{#2}
367   \__cellprops_array_startcontent:
368 }
369
370 \cs_set_eq:NN \__cellprops_orig_LTmkpream:n \@mkpream
371 \cs_set_protected_nopar:Npn \@mkpream#1 {
372   \group_end:
373   \__cellprops_array_init:
374   \group_begin:
375   \__cellprops_orig_LTmkpream:n {#1}
376 }
377
378 \cs_set_eq:NN \__cellprops_orig_LTarray:w \LT@array
379 \cs_set_protected_nopar:Npn \LT@array [#1]#2 {
380   \__cellprops_orig_LTarray:w [#1]{#2}
```

```

381     \__cellprops_array_startcontent:
382 }
383

```

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.

```

384 \cs_new_protected_nopar:Nn \__cellprops_update_color:Nn {
385     \__cellprops_get_property:nN {#2} \l_tmpa_tl
386     \exp_args:NV \tl_if_eq:NNF \l_tmpa_tl \c__cellprops_inherit_color_tl {
387         \tl_set_eq:NN #1 \l_tmpa_tl
388     }
389 }
390
391 \cs_set_protected_nopar:Nn \__cellprops_update_colors: {
392     \__cellprops_update_color:Nn \l__cellprops_color_tl {color}
393     \__cellprops_update_color:Nn \l__cellprops_bgcolor_tl {background-color}
394 }
395
396 \cs_set_nopar:Nn \__cellprops_end_array:n {
397     \crrc
398     \hlx{s[\l__cellprops_tablepadding_bottom_dim]}
399     #1
400     \tl_use:N \l__cellprops_restore_tl
401 }
402
403 \cs_set_eq:NN \__cellprops_orig_endarray: \endarray
404 \cs_set_nopar:Npn \endarray {
405     \__cellprops_end_array:n { \__cellprops_orig_endarray: }
406 }
407
408 \cs_set_eq:NN \__cellprops_orig_endLT: \endlongtable
409 \cs_set_nopar:Npn \endlongtable {
410     \__cellprops_end_array:n { \__cellprops_orig_endLT: }
411 }
412
413 \cs_set_protected_nopar:Nn \__cellprops_startrow: {
414     \bool_if:NF \g__cellprops_inrow_bool {
415         \bool_gset_true:N \g__cellprops_inrow_bool
416         \int_gincr:N \g__cellprops_row_int
417         \int_gset_eq:NN \g__cellprops_col_int \c_one
418         \dim_gzero:N \g__cellprops_ht_dim
419         \dim_gzero:N \g__cellprops_dp_dim
420     }
421 }
422
423 \cs_set_protected_nopar:Npn \__cellprops_endrow: {
424     \bool_if:NT \g__cellprops_inrow_bool {
425         \bool_gset_false:N \g__cellprops_inrow_bool
426         \__cellprops_fix_valign_end:
427     }
428 }
429

```

```

430 \cs_set_eq:NN \__cellprops_orig_tab@arraycr:n \tab@arraycr
431 \cs_set_nopar:Npn \tab@arraycr { \__cellprops_endrow:\__cellprops_orig_tab@arraycr:n }
432
433 \cs_set_eq:NN \__cellprops_orig_multicolumn:w \multicolumn
434 \cs_set:Npn \multicolumn#1#2#3 {
435   \__cellprops_orig_multicolumn:w {#1}{#2}{
436     #3
437     \int_gadd:Nn \g__cellprops_col_int {#1}
438   }
439 }
440
441 \cs_new_protected_nopar:Nn \__cellprops_fix_valign_end: {
442   \exp_args:NV \tl_if_eq:nnTF \tab@hlstate {b} {
443     \dim_gadd:Nn \tab@endheight {
444       \g__cellprops_dp_dim - \box_dp:N \@arstrutbox
445     }
446   }{
447     \int_compare:nNnT \g__cellprops_row_int = \c_one {
448       \dim_gadd:Nn \tab@endheight {
449         \g__cellprops_ht_dim - \box_ht:N \@arstrutbox
450       }
451     }
452   }
453 }
454
455 \colpush{tabular}
456
457 \coldef n{\tabcoltype{
458   \__cellprops_begincell:n{
459 }{
460   \__cellprops_endcell:
461 }}
462 \coldef l{\tabcoltype{
463   \__cellprops_begincell:n
464   {\__cellprops_use_setter:nn {text-align} {left}}
465 }{
466   \__cellprops_endcell:
467 }}
468 \coldef c{\tabcoltype{
469   \__cellprops_begincell:n
470   {\__cellprops_use_setter:nn {text-align} {center}}
471 }{
472   \__cellprops_endcell:
473 }}
474 \coldef r{\tabcoltype{
475   \__cellprops_begincell:n
476   {\__cellprops_use_setter:nn {text-align} {right}}
477 }{
478   \__cellprops_endcell:
479 }}
480 \coldef M#1{\__cellprops_MTcol:nn {math}{#1}}
481 \coldef T#1{\__cellprops_MTcol:nn {text}{#1}}
482 \cs_new_protected_nopar:Nn \__cellprops_MTcol:nn {
483   % TODO: error if align not l, c, or r

```

```

484 \exp_args:Nx \tabcoltype {
485   \exp_not:N \__cellprops_begincell:n {
486     \exp_not:n {\__cellprops_use_setter:nn {math-mode} {#1} }
487     \exp_not:n {\__cellprops_use_setter:nn {text-align}} {
488       \str_case:nn {#2} {
489         {l} {left}
490         {c} {center}
491         {r} {right}
492       }
493     }
494   }
495 }{
496   \__cellprops_endcell:
497 }
498 }
499
500 \coldef p#1{\tabcoltype{
501   \__cellprops_begin_par_cell:nn \vtop {#1}
502 }{
503   \__cellprops_end_par_cell:n {}
504 }}
505 \coldef m#1{\tabcoltype{
506   \__cellprops_begin_par_cell:nn {\c_math_toggle_token\vcenter} {#1}
507 }{
508   \__cellprops_end_par_cell:n{\c_math_toggle_token}
509 }}
510 \coldef b#1{\tabcoltype{
511   \__cellprops_begin_par_cell:nn \vbox {#1}
512 }{
513   \__cellprops_end_par_cell:n {}
514 }}
515
516
517 \colpop
518   Handle various :nth-child() forms.
519 \cs_new_protected_nopar:Nn \__cellprops_seq_nthchild:Nn {
520   \seq_clear:N #1
521   \seq_map_inline:Nn \l__cellprops_used_nth_factors_seq {
522     \seq_put_right:Nx #1 {
523       ##1 n + \int_eval:n{\int_mod:nn{#2}{##1}}
524     }
525   }
526   \seq_put_right:Nx #1 { \int_eval:n{#2} }
527 }
528 \cs_new_protected_nopar:Nn \__cellprops_begincell:n {
529   \__cellprops_begin_raw_cell:n {
530     \__cellprops_seq_nthchild:Nn \l_tmpa_seq { \__cellprops_row_int }
531     \__cellprops_seq_nthchild:Nn \l_tmpb_seq { \__cellprops_col_int }
532     \seq_map_inline:Nn \l_tmpa_seq {
533       \__cellprops_recall_properties:n {tr:nth-child(##1)}
534     }
535     \__cellprops_update_colors:
536     \__cellprops_recall_properties:n {td}

```

```

537     \_cellprops\_recall\_properties:n {tr~td}
538     \seq\_map\_inline:Nn \l\_tmpb\_seq {
539         \_cellprops\_recall\_properties:n {td:nth-child(##1)}
540     }
541     \seq\_map\_inline:Nn \l\_tmpa\_seq {
542         \_cellprops\_recall\_properties:n {tr:nth-child(##1)~td}
543     }
544     \seq\_map\_inline:Nn \l\_tmpa\_seq {
545         \seq\_map\_inline:Nn \l\_tmpb\_seq {
546             \_cellprops\_recall\_properties:n {tr:nth-child(##1)~
547                 td:nth-child(###1)}
548         }
549     }
550     \_cellprops\_update\_colors:
551     #1
552     \hbox\_set:Nw \l\_cellprops\_cell\_box
553     \str\_case\_x:nnF {\_cellprops\_get\_property:n {math-mode}} {
554         { text } { \tab@btext }
555         { math } { \tab@bmaths }
556     }{% any other treated as |auto|
557         \tab@bgroup
558     }
559 }
560 }
561
562 \cs\_new\_protected\_nopar:Nn \_cellprops\_endcell: {
563     \str\_case\_x:nnF {\_cellprops\_get\_property:n {math-mode}} {
564         { text } { \tab@etext }
565         { math } { \tab@emaths }
566     }{% any other treated as |auto|
567         \tab@egroup
568     }
569     \hbox\_set\_end:
570     \_cellprops\_end\_raw\_cell:
571 }
572
573 \cs\_new\_protected\_nopar:Nn \_cellprops\_begin\_par\_cell:nn {
574     \savenotes
575     \_cellprops\_begin\_raw\_cell:n{
576         \hbox\_set:Nw \l\_cellprops\_cell\_box
577         #1
578         \bgroup
579         \hsize#2\relax
580         \@arrayparboxrestore
581         \global\@minipagetrue
582         \everypar{
583             \global\@minipagefalse
584             \everypar{}
585         }
586         \_cellprops\_recall\_properties:n {td~p}
587     }
588 }
589 \cs\_new\_protected\_nopar:Nn \_cellprops\_end\_par\_cell:n {
590     \ifhmode\@maybe@unskip\par\fi

```

```

591 \unskip
592 \egroup
593 #1
594 \hbox_set_end:
595 \__cellprops_end_raw_cell:
596 \spewnotes\hfil
597 }
598
599 \cs_new_protected_nopar:Nn \__cellprops_begin_raw_cell:n {
600 \group_begin:
601 % Additional init code
602 #1
603 % Install the cell color
604 \__cellprops_update_colors:
605 \tl_use:N \l__cellprops_color_tl
606 }
607
608 \cs_new_protected_nopar:Nn \__cellprops_end_raw_cell: {
609 % Here \l__cellprops_cell_box must contain the contents of the cell
610 %
611 % Handle vpadding and min-height/depth
612 \box_set_ht:Nn \l__cellprops_cell_box {
613 \dim_max:nn
614 {\box_ht:N \l__cellprops_cell_box}
615 {\__cellprops_get_property:n {min-height}}
616 + (\__cellprops_get_property:n {padding-top})
617 }
618 \box_set_dp:Nn \l__cellprops_cell_box {
619 \dim_max:nn
620 {\box_dp:N \l__cellprops_cell_box}
621 {\__cellprops_get_property:n {min-depth}}
622 + (\__cellprops_get_property:n {padding-bottom})
623 }
624 % To fix vertical alignment later
625 \dim_gset:Nn \g__cellprops_ht_dim {
626 \dim_max:nn
627 {\g__cellprops_ht_dim}
628 {\box_ht:N \l__cellprops_cell_box}
629 }
630 \dim_gset:Nn \g__cellprops_dp_dim {
631 \dim_max:nn
632 {\g__cellprops_dp_dim}
633 {\box_dp:N \l__cellprops_cell_box}
634 }
635 % Handle hpadding and halign
636 \skip_set:Nn \l_tmpa_skip {
637 \dim_max:nn
638 {0pt}
639 { (\__cellprops_get_property:n {min-width})
640 - \box_wd:N \l__cellprops_cell_box }
641 }
642 \skip_add:Nn \l_tmpa_skip {
643 0pt plus 1fil
644 }

```



```

645 \skip_set:Nn \l__cellprops_left_skip
646   {\__cellprops_get_property:n {padding-left}}
647 \skip_set:Nn \l__cellprops_right_skip
648   {\__cellprops_get_property:n {padding-right}}
649 \str_case:x:nnF {\__cellprops_get_property:n {text-align}} {
650   { right } {
651     \skip_add:Nn \l__cellprops_left_skip { \l_tmpa_skip }
652   }
653   { center } {
654     \skip_add:Nn \l__cellprops_left_skip { \l_tmpa_skip / 2 }
655     \skip_add:Nn \l__cellprops_right_skip { \l_tmpa_skip / 2 }
656   }
657 }{% any other treated as |left|
658   \skip_add:Nn \l__cellprops_right_skip { \l_tmpa_skip }
659 }
660 \kern\c_zero_dim
661 \tl_if_empty:NF \l__cellprops_bgcolor_tl {
662   \group_begin:
663   % Paint a background with leaders
664   \tl_use:N \l__cellprops_bgcolor_tl % install the color
665   \skip_set:Nn \l_tmpa_skip {
666     \l__cellprops_left_skip
667     + \box_wd:N \l__cellprops_cell_box
668     + \l__cellprops_right_skip
669   }
670   \leaders
671     \vrule
672     \skip_horizontal:N \l_tmpa_skip
673     \skip_horizontal:n {-\l_tmpa_skip}
674   \group_end:
675 }
676 \skip_horizontal:N \l__cellprops_left_skip
677 \box_use_clear:N \l__cellprops_cell_box
678 \skip_horizontal:N \l__cellprops_right_skip
679 \kern\c_zero_dim
680 \group_end:
681 }
682 \end{package}

```