

# File I

## Implementation

### 1 l3backend-basics Implementation

```
1 <*package>
```

Whilst there is a reasonable amount of code overlap between backends, it is much clearer to have the blocks more-or-less separated than run in together and DocStripped out in parts. As such, most of the following is set up on a per-backend basis, though there is some common code (again given in blocks not interspersed with other material).

All the file identifiers are up-front so that they come out in the right place in the files.

```
2 \ProvidesExplFile
3 <*dvipdfmx>
4   {l3backend-dvipdfmx.def}{2022-08-23}{ }
5   {L3 backend support: dvipdfmx}
6 </dvipdfmx>
7 <*dvips>
8   {l3backend-dvips.def}{2022-08-23}{ }
9   {L3 backend support: dvips}
10 </dvips>
11 <*dvisvgm>
12   {l3backend-dvisvgm.def}{2022-08-23}{ }
13   {L3 backend support: dvisvgm}
14 </dvisvgm>
15 <*luatex>
16   {l3backend-luatex.def}{2022-08-23}{ }
17   {L3 backend support: PDF output (LuaTeX)}
18 </luatex>
19 <*pdftex>
20   {l3backend-pdftex.def}{2022-08-23}{ }
21   {L3 backend support: PDF output (pdfTeX)}
22 </pdftex>
23 <*xetex>
24   {l3backend-xetex.def}{2022-08-23}{ }
25   {L3 backend support: XeTeX}
26 </xetex>
```

Check if the loaded kernel is at least enough to load this file. The kernel date has to be at least equal to \ExplBackendFileDate or later. If \\_kernel\_dependency\\_version\\_check:Nn doesn't exist we're loading in an older kernel, so it's an error anyway. With time, this test should vanish and only the dependency check should remain.

```
27 \cs_if_exist:NTF \_kernel_dependency_version_check:nn
28   {
29     \_kernel_dependency_version_check:nn {2021-02-18}
30 <dvipdfmx>    {l3backend-dvipdfmx.def}
31 <dvips>       {l3backend-dvips.def}
32 <dvisvgm>     {l3backend-dvisvgm.def}
33 <luatex>      {l3backend-luatex.def}
34 <pdftex>      {l3backend-pdftex.def}
35 <xetex>       {l3backend-xetex.def}
```

```

36 }
37 {
38   \cs_if_exist_use:cF { @latex@error } { \errmessage }
39   {
40     Mismatched-LaTeX-support-files-detected. \MessageBreak
41     Loading~aborted!
42   }
43   { \use:c { @ehd } }
44   \tex_endinput:D
45 }

```

The order of the backend code here is such that we get somewhat logical outcomes in terms of code sharing whilst keeping things readable. (Trying to mix all of the code by concept is almost unmanageable.) The key parts which are shared are

- Color support is either dvips-like or LuaTeX/pdfTeX-like.
- LuaTeX/pdfTeX and dvipdfmx/X<sub>Y</sub>TeX share drawing routines.
- X<sub>Y</sub>TeX is the same as dvipdfmx other than image size extraction so takes most of the same code.

`__kernel_backend_literal:e` The one shared function for all backends is access to the basic `\special` primitive: it has slightly odd expansion behaviour so a wrapper is provided.

```

__kernel_backend_literal:n
__kernel_backend_literal:x
46 \cs_new_eq:NN __kernel_backend_literal:e \tex_special:D
47 \cs_new_protected:Npn __kernel_backend_literal:n #1
48   { __kernel_backend_literal:e { \exp_not:n {#1} } }
49 \cs_generate_variant:Nn __kernel_backend_literal:n { x }

```

(End definition for `__kernel_backend_literal:e`.)

`\_kernel_backend_first_shipout:n` We need to write at first shipout in a few places. As we want to use the most up-to-date method,

```

50 \cs_if_exist:NTF \@ifl@t@r
51   {
52     \@ifl@t@r \fmtversion { 2020-10-01 }
53     {
54       \cs_new_protected:Npn __kernel_backend_first_shipout:n #1
55         { \hook_gput_code:nnn { shipout / firstpage } { l3backend } {#1} }
56     }
57     { \cs_new_eq:NN __kernel_backend_first_shipout:n \AtBeginDvi }
58   }
59   { \cs_new_eq:NN __kernel_backend_first_shipout:n \use:n }

```

(End definition for `__kernel_backend_first_shipout:n`.)

## 1.1 dvips backend

```

60 <*dvips>

```

`__kernel_backend_literal_postscript:n` Literal PostScript can be included using a few low-level formats. Here, we use the form with no positioning: this is overall more convenient as a wrapper. Note that this does require that where position is important, an appropriate wrapper is included.

```

61 \cs_new_protected:Npn __kernel_backend_literal_postscript:n #1
62   { __kernel_backend_literal:n { ps:: #1 } }
63 \cs_generate_variant:Nn __kernel_backend_literal_postscript:n { x }

```

(End definition for `\_kernel_backend_literal_postscript:n`.)

`\_kernel_backend_postscript:n` PostScript data that does have positioning, and also applying a shift to `SDict` (which is not done automatically by `ps:` or `ps::`, in contrast to `!` or `"`).

`\_kernel_backend_postscript:x`

```

64 \cs_new_protected:Npn \_kernel_backend_postscript:n #1
65   { \_kernel_backend_literal:n { ps: SDict ~ begin ~ #1 ~ end } }
66 \cs_generate_variant:Nn \_kernel_backend_postscript:n { x }

```

(End definition for `\_kernel_backend_postscript:n`.)

PostScript for the header: a small saving but makes the code clearer. This is held until the start of shipout such that a document with no actual output does not write anything.

```

67 \bool_if:NT \g__kernel_backend_header_bool
68   {
69     \_kernel_backend_first_shipout:n
70     { \_kernel_backend_literal:n { header = l3backend-dvips.pro } }
71   }

```

`\_kernel_backend_align_begin:`

`\_kernel_backend_align_end:`

In `dvips` there is no built-in saving of the current position, and so some additional PostScript is required to set up the transformation matrix and also to restore it afterwards. Notice the use of the stack to save the current position “up front” and to move back to it at the end of the process. Notice that the `[begin]`/`[end]` pair here mean that we can use a run of PostScript statements in separate lines: not *required* but does make the code and output more clear.

```

72 \cs_new_protected:Npn \_kernel_backend_align_begin:
73   {
74     \_kernel_backend_literal:n { ps::[begin] }
75     \_kernel_backend_literal_postscript:n { currentpoint }
76     \_kernel_backend_literal_postscript:n { currentpoint-translate }
77   }
78 \cs_new_protected:Npn \_kernel_backend_align_end:
79   {
80     \_kernel_backend_literal_postscript:n { neg-exch-neg-exch-translate }
81     \_kernel_backend_literal:n { ps::[end] }
82   }

```

(End definition for `\_kernel_backend_align_begin:` and `\_kernel_backend_align_end:.`)

`\_kernel_backend_scope_begin:`

`\_kernel_backend_scope_end:`

Saving/restoring scope for general operations needs to be done with `dvips` positioning (try without to see this!). Thus we need the `ps:` version of the special here. As only the graphics state is ever altered within this pairing, we use the lower-cost `g`-versions.

```

83 \cs_new_protected:Npn \_kernel_backend_scope_begin:
84   { \_kernel_backend_literal:n { ps:gsave } }
85 \cs_new_protected:Npn \_kernel_backend_scope_end:
86   { \_kernel_backend_literal:n { ps:grestore } }

```

(End definition for `\_kernel_backend_scope_begin:` and `\_kernel_backend_scope_end:.`)

```

87 </dvips>

```

## 1.2 LuaTeX and pdfTeX backends

88 `<*luatex | pdftex>`

Both LuaTeX and pdfTeX write PDFs directly rather than via an intermediate file. Although there are similarities, the move of LuaTeX to have more code in Lua means we create two independent files using shared DocStrip code.

This is equivalent to `\special{pdf:}` but the engine can track it. Without the `direct` keyword everything is kept in sync: the transformation matrix is set to the current point automatically. Note that this is still inside the text (BT ...ET block).

```
89 \cs_new_protected:Npn \__kernel_backend_literal_pdf:n #1
90 {
91   <*luatex>
92   \tex_pdfextension:D literal
93   </luatex>
94   <*pdftex>
95   \tex_pdfliteral:D
96   </pdftex>
97   { \exp_not:n {#1} }
98 }
99 \cs_generate_variant:Nn \__kernel_backend_literal_pdf:n { x }
```

(End definition for `\__kernel_backend_literal_pdf:n`.)

Page literals are pretty simple. To avoid an expansion, we write out by hand.

```
100 \cs_new_protected:Npn \__kernel_backend_literal_page:n #1
101 {
102   <*luatex>
103   \tex_pdfextension:D literal ~
104   </luatex>
105   <*pdftex>
106   \tex_pdfliteral:D
107   </pdftex>
108   page { \exp_not:n {#1} }
109 }
```

(End definition for `\__kernel_backend_literal_page:n`.)

Higher-level interfaces for saving and restoring the graphic state.

```
110 \cs_new_protected:Npn \__kernel_backend_scope_begin:
111 {
112   <*luatex>
113   \tex_pdfextension:D save \scan_stop:
114   </luatex>
115   <*pdftex>
116   \tex_pdfsave:D
117   </pdftex>
118 }
119 \cs_new_protected:Npn \__kernel_backend_scope_end:
120 {
121   <*luatex>
122   \tex_pdfextension:D restore \scan_stop:
123   </luatex>
124   <*pdftex>
125   \tex_pdfrestore:D
```

```

126 </pdftex>
127 }

```

(End definition for `\_kernel_backend_scope_begin:` and `\_kernel_backend_scope_end:.`)

`\_kernel_backend_matrix:n` Here the appropriate function is set up to insert an affine matrix into the PDF. With pdfTeX and LuaTeX in direct PDF output mode there is a primitive for this, which only needs the rotation/scaling/skew part.

```

128 \cs_new_protected:Npn \_kernel_backend_matrix:n #1
129 {
130   <*luatex>
131   \tex_pdfextension:D setmatrix
132 </luatex>
133 <*pdftex>
134   \tex_pdfsetmatrix:D
135 </pdftex>
136   { \exp_not:n {#1} }
137 }
138 \cs_generate_variant:Nn \_kernel_backend_matrix:n { x }

```

(End definition for `\_kernel_backend_matrix:n.`)

```

139 </luatex | pdftex>

```

### 1.3 dvipdfmx backend

```

140 <*dvipdfmx | xetex>

```

The dvipdfmx shares code with the PDF mode one (using the common section to this file) but also with XeTeX. The latter is close to identical to dvipdfmx and so all of the code here is extracted for both backends, with some `clean up` for XeTeX as required. Undocumented but equivalent to pdfTeX's `literal` keyword. It's similar to be not the same as the documented `contents` keyword as that adds a q/Q pair.

```

\_kernel_backend_literal_pdf:n
\_kernel_backend_literal_pdf:x

```

```

141 \cs_new_protected:Npn \_kernel_backend_literal_pdf:n #1
142 { \_kernel_backend_literal:n { pdf:literal~ #1 } }
143 \cs_generate_variant:Nn \_kernel_backend_literal_pdf:n { x }

```

(End definition for `\_kernel_backend_literal_pdf:n.`)

`\_kernel_backend_literal_page:n` Whilst the manual says this is like `literal direct` in pdfTeX, it closes the BT block!

```

144 \cs_new_protected:Npn \_kernel_backend_literal_page:n #1
145 { \_kernel_backend_literal:n { pdf:literal~direct~ #1 } }

```

(End definition for `\_kernel_backend_literal_page:n.`)

`\_kernel_backend_scope_begin:` Scoping is done using the backend-specific specials. We use the versions originally from xdvipdfmx (`x:`) as these are well-tested “in the wild”.

```

146 \cs_new_protected:Npn \_kernel_backend_scope_begin:
147 { \_kernel_backend_literal:n { x:gsave } }
148 \cs_new_protected:Npn \_kernel_backend_scope_end:
149 { \_kernel_backend_literal:n { x:grestore } }

```

(End definition for `\_kernel_backend_scope_begin:` and `\_kernel_backend_scope_end:.`)

```

150 </dvipdfmx | xetex>

```

## 1.4 dvisvgm backend

151 `\dvisvgm`

`\_kernel_backend_literal_svg:n`  
`\_kernel_backend_literal_svg:x`

Unlike the other backends, the requirements for making SVG files mean that we can’t conveniently transform all operations to the current point. That makes life a bit more tricky later as that needs to be accounted for. A new line is added after each call to help to keep the output readable for debugging.

152 `\cs_new_protected:Npn \_kernel_backend_literal_svg:n #1`  
 153 `{ \_kernel_backend_literal:n { dvisvgm:raw~ #1 { ?nl } } }`  
 154 `\cs_generate_variant:Nn \_kernel_backend_literal_svg:n { x }`

(End definition for `\_kernel_backend_literal_svg:n`.)

`\g__kernel_backend_scope_int`  
`\l__kernel_backend_scope_int`

In SVG, we need to track scope nesting as properties attach to scopes; that requires a pair of int registers.

155 `\int_new:N \g__kernel_backend_scope_int`  
 156 `\int_new:N \l__kernel_backend_scope_int`

(End definition for `\g__kernel_backend_scope_int` and `\l__kernel_backend_scope_int`.)

`\_kernel_backend_scope_begin:`  
`\_kernel_backend_scope_end:`  
`\_kernel_backend_scope_begin:n`  
`\_kernel_backend_scope_begin:x`  
`\_kernel_backend_scope:n`  
`\_kernel_backend_scope:x`

In SVG, the need to attach concepts to a scope means we need to be sure we will close all of the open scopes. That is easiest done if we only need an outer “wrapper” **begin/end** pair, and within that we apply operations as a simple scoped statements. To keep down the non-productive groups, we also have a **begin** version that does take an argument.

157 `\cs_new_protected:Npn \_kernel_backend_scope_begin:`  
 158 `{`  
 159 `\_kernel_backend_literal_svg:n { <g> }`  
 160 `\int_set_eq:NN`  
 161 `\l__kernel_backend_scope_int`  
 162 `\g__kernel_backend_scope_int`  
 163 `\group_begin:`  
 164 `\int_gset:Nn \g__kernel_backend_scope_int { 1 }`  
 165 `}`  
 166 `\cs_new_protected:Npn \_kernel_backend_scope_end:`  
 167 `{`  
 168 `\prg_replicate:nn`  
 169 `{ \g__kernel_backend_scope_int }`  
 170 `{ \_kernel_backend_literal_svg:n { </g> } }`  
 171 `\group_end:`  
 172 `\int_gset_eq:NN`  
 173 `\g__kernel_backend_scope_int`  
 174 `\l__kernel_backend_scope_int`  
 175 `}`  
 176 `\cs_new_protected:Npn \_kernel_backend_scope_begin:n #1`  
 177 `{`  
 178 `\_kernel_backend_literal_svg:n { <g ~ #1 > }`  
 179 `\int_set_eq:NN`  
 180 `\l__kernel_backend_scope_int`  
 181 `\g__kernel_backend_scope_int`  
 182 `\group_begin:`  
 183 `\int_gset:Nn \g__kernel_backend_scope_int { 1 }`  
 184 `}`  
 185 `\cs_generate_variant:Nn \_kernel_backend_scope_begin:n { x }`

```

186 \cs_new_protected:Npn \__kernel_backend_scope:n #1
187 {
188   \__kernel_backend_literal_svg:n { <g ~ #1 > }
189   \int_gincr:N \g__kernel_backend_scope_int
190 }
191 \cs_generate_variant:Nn \__kernel_backend_scope:n { x }

(End definition for \__kernel_backend_scope_begin: and others.)

192 </dvisvgm>
193 </package>

```

## 2 l3backend-box Implementation

```

194 <*package>
195 <@@=box>

```

### 2.1 dvips backend

```

196 <*dvips>

```

`\__box_backend_clip:N` The `dvips` backend scales all absolute dimensions based on the output resolution selected and any TeX magnification. Thus for any operation involving absolute lengths there is a correction to make. See `normalscale` from `special.pro` for the variables, noting that here everything is saved on the stack rather than as a separate variable. Once all of that is done, the actual clipping is trivial.

```

197 \cs_new_protected:Npn \__box_backend_clip:N #1
198 {
199   \__kernel_backend_scope_begin:
200   \__kernel_backend_align_begin:
201   \__kernel_backend_literal_postscript:n { matrix~currentmatrix }
202   \__kernel_backend_literal_postscript:n
203     { Resolution~72~div~VResolution~72~div~scale }
204   \__kernel_backend_literal_postscript:n { DVImag~dup~scale }
205   \__kernel_backend_literal_postscript:x
206     {
207       0 ~
208       \dim_to_decimal_in_bp:n { \box_dp:N #1 } ~
209       \dim_to_decimal_in_bp:n { \box_wd:N #1 } ~
210       \dim_to_decimal_in_bp:n { -\box_ht:N #1 - \box_dp:N #1 } ~
211       rectclip
212     }
213   \__kernel_backend_literal_postscript:n { setmatrix }
214   \__kernel_backend_align_end:
215   \hbox_overlap_right:n { \box_use:N #1 }
216   \__kernel_backend_scope_end:
217   \skip_horizontal:n { \box_wd:N #1 }
218 }

```

(End definition for `\__box_backend_clip:N`.)

`\__box_backend_rotate:Nn` Rotating using `dvips` does not require that the box dimensions are altered and has a very convenient built-in operation. Zero rotation must be written as 0 not -0 so there is a quick test.

```

219 \cs_new_protected:Npn \__box_backend_rotate:Nn #1#2
220 { \exp_args:NNf \__box_backend_rotate_aux:Nn #1 { \fp_eval:n {#2} } }
221 \cs_new_protected:Npn \__box_backend_rotate_aux:Nn #1#2
222 {
223   \__kernel_backend_scope_begin:
224   \__kernel_backend_align_begin:
225   \__kernel_backend_literal_postscript:x
226   {
227     \fp_compare:nNnTF {#2} = \c_zero_fp
228     { 0 }
229     { \fp_eval:n { round ( -(#2) , 5 ) } } ~
230     rotate
231   }
232   \__kernel_backend_align_end:
233   \box_use:N #1
234   \__kernel_backend_scope_end:
235 }

```

(End definition for \\_\_box\_backend\_rotate:Nn and \\_\_box\_backend\_rotate\_aux:Nn.)

\\_\_box\_backend\_scale:Nnn The dvips backend once again has a dedicated operation we can use here.

```

236 \cs_new_protected:Npn \__box_backend_scale:Nnn #1#2#3
237 {
238   \__kernel_backend_scope_begin:
239   \__kernel_backend_align_begin:
240   \__kernel_backend_literal_postscript:x
241   {
242     \fp_eval:n { round ( #2 , 5 ) } ~
243     \fp_eval:n { round ( #3 , 5 ) } ~
244     scale
245   }
246   \__kernel_backend_align_end:
247   \hbox_overlap_right:n { \box_use:N #1 }
248   \__kernel_backend_scope_end:
249 }

```

(End definition for \\_\_box\_backend\_scale:Nnn.)

250 </dvips>

## 2.2 LuaTeX and pdfTeX backends

251 <\*luatex | pdftex>

\\_\_box\_backend\_clip:N The general method is to save the current location, define a clipping path equivalent to the bounding box, then insert the content at the current position and in a zero width box. The “real” width is then made up using a horizontal skip before tidying up. There are other approaches that can be taken (for example using XForm objects), but the logic here shares as much code as possible and uses the same conversions (and so same rounding errors) in all cases.

```

252 \cs_new_protected:Npn \__box_backend_clip:N #1
253 {
254   \__kernel_backend_scope_begin:
255   \__kernel_backend_literal_pdf:x
256   {

```



```

257      0~
258      \dim_to_decimal_in_bp:n { -\box_dp:N #1 } ~
259      \dim_to_decimal_in_bp:n { \box_wd:N #1 } ~
260      \dim_to_decimal_in_bp:n { \box_ht:N #1 + \box_dp:N #1 } ~
261      re~W~n
262    }
263    \hbox_overlap_right:n { \box_use:N #1 }
264    \__kernel_backend_scope_end:
265    \skip_horizontal:n { \box_wd:N #1 }
266  }

```

(End definition for \\_\_box\_backend\_clip:N.)

\\_\_box\_backend\_rotate:Nn Rotations are set using an affine transformation matrix which therefore requires sine/cosine values not the angle itself. We store the rounded values to avoid rounding twice. There are also a couple of comparisons to ensure that -0 is not written to the output, as this avoids any issues with problematic display programs. Note that numbers are compared to 0 after rounding.

```

267 \cs_new_protected:Npn \__box_backend_rotate:Nn #1#2
268 { \exp_args:Nnf \__box_backend_rotate_aux:Nn #1 { \fp_eval:n {#2} } }
269 \cs_new_protected:Npn \__box_backend_rotate_aux:Nn #1#2
270 {
271   \__kernel_backend_scope_begin:
272   \box_set_wd:Nn #1 { 0pt }
273   \fp_set:Nn \l__box_backend_cos_fp { round ( cosd ( #2 ) , 5 ) }
274   \fp_compare:nNnT \l__box_backend_cos_fp = \c_zero_fp
275   { \fp_zero:N \l__box_backend_cos_fp }
276   \fp_set:Nn \l__box_backend_sin_fp { round ( sind ( #2 ) , 5 ) }
277   \__kernel_backend_matrix:x
278   {
279     \fp_use:N \l__box_backend_cos_fp \c_space_tl
280     \fp_compare:nNnTF \l__box_backend_sin_fp = \c_zero_fp
281     { 0~0 }
282     {
283       \fp_use:N \l__box_backend_sin_fp
284       \c_space_tl
285       \fp_eval:n { -\l__box_backend_sin_fp }
286     }
287     \c_space_tl
288     \fp_use:N \l__box_backend_cos_fp
289   }
290   \box_use:N #1
291   \__kernel_backend_scope_end:
292 }
293 \fp_new:N \l__box_backend_cos_fp
294 \fp_new:N \l__box_backend_sin_fp

```

(End definition for \\_\_box\_backend\_rotate:Nn and others.)

\\_\_box\_backend\_scale:Nnn The same idea as for rotation but without the complexity of signs and cosines.

```

295 \cs_new_protected:Npn \__box_backend_scale:Nnn #1#2#3
296 {
297   \__kernel_backend_scope_begin:
298   \__kernel_backend_matrix:x

```

```

299     {
300       \fp_eval:n { round ( #2 , 5 ) } ~
301       0~0~
302       \fp_eval:n { round ( #3 , 5 ) }
303     }
304     \hbox_overlap_right:n { \box_use:N #1 }
305     \__kernel_backend_scope_end:
306   }

```

(End definition for \\_\_box\_backend\_scale:Nnn.)

```

307 </luatex | pdftex>

```

## 2.3 dvipdfmx/X<sub>Y</sub>TeX backend

```

308 <*dvipdfmx | xetex>

```

\\_\_box\_backend\_clip:N The code here is identical to that for LuaTeX/pdfTeX: unlike rotation and scaling, there is no higher-level support in the backend for clipping.

```

309 \cs_new_protected:Npn \__box_backend_clip:N #1
310 {
311   \__kernel_backend_scope_begin:
312   \__kernel_backend_literal_pdf:x
313   {
314     0~
315     \dim_to_decimal_in_bp:n { -\box_dp:N #1 } ~
316     \dim_to_decimal_in_bp:n { \box_wd:N #1 } ~
317     \dim_to_decimal_in_bp:n { \box_ht:N #1 + \box_dp:N #1 } ~
318     re~W~n
319   }
320   \hbox_overlap_right:n { \box_use:N #1 }
321   \__kernel_backend_scope_end:
322   \skip_horizontal:n { \box_wd:N #1 }
323 }

```

(End definition for \\_\_box\_backend\_clip:N.)

\\_\_box\_backend\_rotate:Nn Rotating in dvipdfmx/X<sub>Y</sub>TeX can be implemented using either PDF or backend-specific code. The former approach however is not “aware” of the content of boxes: this means that any embedded links would not be adjusted by the rotation. As such, the backend-native approach is preferred: the code therefore is similar (though not identical) to the dvips version (notice the rotation angle here is positive). As for dvips, zero rotation is written as 0 not -0.

```

324 \cs_new_protected:Npn \__box_backend_rotate:Nn #1#2
325 { \exp_args:Nnf \__box_backend_rotate_aux:Nn #1 { \fp_eval:n {#2} } }
326 \cs_new_protected:Npn \__box_backend_rotate_aux:Nn #1#2
327 {
328   \__kernel_backend_scope_begin:
329   \__kernel_backend_literal:x
330   {
331     x:rotate~
332     \fp_compare:nNnTF {#2} = \c_zero_fp
333     { 0 }
334     { \fp_eval:n { round ( #2 , 5 ) } }
335   }

```

```

336     \box_use:N #1
337     \__kernel_backend_scope_end:
338 }

```

(End definition for `\__box_backend_rotate:Nn` and `\__box_backend_rotate_aux:Nn`.)

`\__box_backend_scale:Nnn` Much the same idea for scaling: use the higher-level backend operation to allow for box content.

```

339 \cs_new_protected:Npn \__box_backend_scale:Nnn #1#2#3
340 {
341     \__kernel_backend_scope_begin:
342     \__kernel_backend_literal:x
343     {
344         x:scale~
345         \fp_eval:n { round ( #2 , 5 ) } ~
346         \fp_eval:n { round ( #3 , 5 ) }
347     }
348     \hbox_overlap_right:n { \box_use:N #1 }
349     \__kernel_backend_scope_end:
350 }

```

(End definition for `\__box_backend_scale:Nnn`.)

```

351 </dviPDFmx | xetex>

```

## 2.4 dvisvgm backend

```

352 <*dvisvgm>

```

`\__box_backend_clip:N`  
`\g__kernel_clip_path_int`

Clipping in SVG is more involved than with other backends. The first issue is that the clipping path must be defined separately from where it is used, so we need to track how many paths have applied. The naming here uses `l3cp` as the namespace with a number following. Rather than use a rectangular operation, we define the path manually as this allows it to have a depth: easier than the alternative approach of shifting content up and down using scopes to allow for the depth of the  $\text{\TeX}$  box and keep the reference point the same!

```

353 \cs_new_protected:Npn \__box_backend_clip:N #1
354 {
355     \int_gincr:N \g__kernel_clip_path_int
356     \__kernel_backend_literal_svg:x
357     { < clipPath-id = " l3cp \int_use:N \g__kernel_clip_path_int " > }
358     \__kernel_backend_literal_svg:x
359     {
360         <
361         path ~ d =
362         "
363         M ~ 0 ~
364         \dim_to_decimal:n { -\box_dp:N #1 } ~
365         L ~ \dim_to_decimal:n { \box_wd:N #1 } ~
366         \dim_to_decimal:n { -\box_dp:N #1 } ~
367         L ~ \dim_to_decimal:n { \box_wd:N #1 } ~
368         \dim_to_decimal:n { \box_ht:N #1 + \box_dp:N #1 } ~
369         L ~ 0 ~
370         \dim_to_decimal:n { \box_ht:N #1 + \box_dp:N #1 } ~
371         Z

```

```

372         "
373     />
374 }
375 \__kernel_backend_literal_svg:n
376 { < /clipPath > }

```

In general the SVG set up does not try to transform coordinates to the current point. For clipping we need to do that, so have a transformation here to get us to the right place, and a matching one just before the  $\text{\TeX}$  box is inserted to get things back on track. The clip path needs to come between those two such that if lines up with the current point, as does the  $\text{\TeX}$  box.

```

377 \__kernel_backend_scope_begin:n
378 {
379     transform =
380     "
381         translate ( { ?x } , { ?y } ) ~
382         scale ( 1 , -1 )
383     "
384 }
385 \__kernel_backend_scope:x
386 {
387     clip-path =
388     "url ( \c_hash_str l3cp \int_use:N \g__kernel_clip_path_int ) "
389 }
390 \__kernel_backend_scope:n
391 {
392     transform =
393     "
394         scale ( -1 , 1 ) ~
395         translate ( { ?x } , { ?y } ) ~
396         scale ( -1 , -1 )
397     "
398 }
399 \box_use:N #1
400 \__kernel_backend_scope_end:
401 }
402 \int_new:N \g__kernel_clip_path_int

```

(End definition for  $\backslash\_box\_backend\_clip:N$  and  $\backslash g\_kernel\_clip\_path\_int$ .)

$\backslash\_box\_backend\_rotate:Nn$  Rotation has a dedicated operation which includes a centre-of-rotation optional pair. That can be picked up from the backend syntax, so there is no need to worry about the transformation matrix.

```

403 \cs_new_protected:Npn \__box_backend_rotate:Nn #1#2
404 {
405     \__kernel_backend_scope_begin:x
406     {
407         transform =
408         "
409             rotate
410             ( \fp_eval:n { round ( -(#2) , 5 ) } , ~ { ?x } , ~ { ?y } )
411         "
412     }
413     \box_use:N #1

```

```

414     \__kernel_backend_scope_end:
415 }

```

(End definition for \\_\_box\_backend\_rotate:Nn.)

\\_\_box\_backend\_scale:Nnn

In contrast to rotation, we have to account for the current position in this case. That is done using a couple of translations in addition to the scaling (which is therefore done backward with a flip).

```

416 \cs_new_protected:Npn \__box_backend_scale:Nnn #1#2#3
417 {
418     \__kernel_backend_scope_begin:x
419     {
420         transform =
421         "
422             translate ( { ?x } , { ?y } ) ~
423             scale
424             (
425                 \fp_eval:n { round ( -#2 , 5 ) } ,
426                 \fp_eval:n { round ( -#3 , 5 ) }
427             ) ~
428             translate ( { ?x } , { ?y } ) ~
429             scale ( -1 )
430         "
431     }
432     \hbox_overlap_right:n { \box_use:N #1 }
433     \__kernel_backend_scope_end:
434 }

```

(End definition for \\_\_box\_backend\_scale:Nnn.)

435 </dvisvgm>

436 </package>

### 3 l3backend-color Implementation

437 <\*package>

438 <@@=color>

Color support is split into parts: collecting data from L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>, the color stack, general color, separations, and color for drawings. We have different approaches in each backend, and have some choices to make about dvipdfmx/X<sub>Y</sub>L<sub>T</sub>E<sub>X</sub> in particular. Whilst it is in some ways convenient to use the same approach in multiple backends, the fact that dvipdfmx/X<sub>Y</sub>L<sub>T</sub>E<sub>X</sub> is PDF-based means it (largely) sticks closer to direct PDF output.

#### 3.1 Collecting information from L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>

##### 3.1.1 dvips-style

439 <\*dvisvgm | dvipdfmx | dvips | xetex>

\\_\_color\_backend\_pickup:N

Allow for L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> color. Here, the possible input values are limited: dvips-style colors can be taken as-is. The x-type expansion is there to cover the case where xcolor is in use.

\\_\_color\_backend\_pickup:w

```

440 \cs_new_protected:Npn \__color_backend_pickup:N #1
441 {
442     \exp_args:NV \tl_if_head_is_space:nTF \current@color

```

```

443 {
444   \tl_set:Nn #1 { { gray } { 0 } }
445   \msg_warning:nxx { color } { unhandled }
446   { \current@color }
447 }
448 {
449   \exp_last_unbraced:Nx \__color_backend_pickup:w
450   { \current@color } \s__color_stop #1
451 }
452 }
453 \cs_new_protected:Npn \__color_backend_pickup:w #1 ~ #2 \s__color_stop #3
454 { \tl_set:Nn #3 { {#1} {#2} } }
(End definition for \__color_backend_pickup:N and \__color_backend_pickup:w.)
455 </dvisvgm | dvipdfmx | dvips | xetex>

```

### 3.1.2 LuaTeX and pdfTeX

```

456 <*luatex | pdftex>

```

\\_\_color\_backend\_pickup:N Same ideas, but with a different backend-dependent format.

```

\__color_backend_pickup:w
457 \cs_new_protected:Npn \__color_backend_pickup:N #1
458 {
459   \exp_last_unbraced:Nx \__color_backend_pickup:w
460   { \current@color } ~ 0 ~ 0 ~ 0 \s__color_stop #1
461 }
462 \cs_new_protected:Npn \__color_backend_pickup:w
463 #1 ~ #2 ~ #3 ~ #4 ~ #5 ~ #6 \s__color_stop #7
464 {
465   \str_if_eq:nnTF {#2} { g }
466   { \tl_set:Nn #7 { { gray } {#1} } }
467   {
468     \str_if_eq:nnTF {#4} { rg }
469     { \tl_set:Nn #7 { { rgb } { #1 ~ #2 ~ #3 } } }
470     {
471       \str_if_eq:nnTF {#5} { k }
472       { \tl_set:Nn #7 { { cmyk } { #1 ~ #2 ~ #3 ~ #4 } } }
473       {
474         \tl_set:Nn #1 { { gray } { 0 } }
475         \msg_warning:nxx { color } { unhandled }
476         { \current@color }
477       }
478     }
479   }
480 }
(End definition for \__color_backend_pickup:N and \__color_backend_pickup:w.)
481 </luatex | pdftex>

```

## 3.2 The color stack

For PDF-based engines, we have a color stack available inside the specials. This is used for concepts beyond color itself: it is needed to manage the graphics state generally. Although dvipdfmx/X<sub>Y</sub>TeX have multiple color stacks in recent releases, the way these

interact with the original single stack and with other graphic state operations means that currently it is not feasible to use the multiple stacks.

### 3.2.1 Common code

```
482 < *luatex | pdftex >
```

`\l__color_backend_stack_int` For tracking which stack is in use where multiple stacks are used: currently just pdfTeX/LuaTeX but at some future stage may also cover dvipdfmx/X<sub>Y</sub>TeX.

```
483 \int_new:N \l__color_backend_stack_int
```

(End definition for `\l__color_backend_stack_int`.)

```
484 < /luatex | pdftex >
```

### 3.2.2 LuaTeX and pdfTeX

```
485 < *luatex | pdftex >
```

`\__kernel_color_backend_stack_init:Nnn`

```
486 \cs_new_protected:Npn \__kernel_color_backend_stack_init:Nnn #1#2#3
487 {
488   \int_const:Nn #1
489   {
490     < *luatex >
491     \tex_pdffeedback:D colorstackinit ~
492     < /luatex >
493     < *pdftex >
494     \tex_pdfcolorstackinit:D
495     < /pdftex >
496     \tl_if_blank:nF {#2} { #2 ~ }
497     {#3}
498   }
499 }
```

(End definition for `\__kernel_color_backend_stack_init:Nnn`.)

`\__kernel_color_backend_stack_push:nn`

`\__kernel_color_backend_stack_pop:n`

```
500 \cs_new_protected:Npn \__kernel_color_backend_stack_push:nn #1#2
501 {
502   < *luatex >
503   \tex_pdfextension:D colorstack ~
504   < /luatex >
505   < *pdftex >
506   \tex_pdfcolorstack:D
507   < /pdftex >
508   \int_eval:n {#1} ~ push ~ {#2}
509 }
510 \cs_new_protected:Npn \__kernel_color_backend_stack_pop:n #1
511 {
512   < *luatex >
513   \tex_pdfextension:D colorstack ~
514   < /luatex >
515   < *pdftex >
516   \tex_pdfcolorstack:D
517   < /pdftex >
```

```

518     \int_eval:n {#1} ~ pop \scan_stop:
519   }

```

(End definition for `\__kernel_color_backend_stack_push:nn` and `\__kernel_color_backend_stack_pop:n`.)

```

520 </luatex | pdftex>

```

### 3.3 General color

#### 3.3.1 dvips-style

```

521 <*dvips | dvisvgm>

```

Push the data to the stack. In the case of `dvips` also saves the drawing color in raw PostScript. The spot model is for handling data in classical format.

```

522 \cs_new_protected:Npn \__color_backend_select_cmyk:n #1
523   { \__color_backend_select:n { cmyk ~ #1 } }
524 \cs_new_protected:Npn \__color_backend_select_gray:n #1
525   { \__color_backend_select:n { gray ~ #1 } }
526 \cs_new_protected:Npn \__color_backend_select_named:n #1
527   { \__color_backend_select:n { ~ #1 } }
528 \cs_new_protected:Npn \__color_backend_select_rgb:n #1
529   { \__color_backend_select:n { rgb ~ #1 } }
530 \cs_new_protected:Npn \__color_backend_select:n #1
531   {
532     \__kernel_backend_literal:n { color~push~ #1 }
533   <*dvips>
534     \__kernel_backend_postscript:n { /color.sc ~ { } ~ def }
535   </dvips>
536   }
537 \cs_new_protected:Npn \__color_backend_reset:
538   { \__kernel_backend_literal:n { color~pop } }

```

(End definition for `\__color_backend_select_cmyk:n` and others. This function is documented on page ??.)

```

539 </dvips | dvisvgm>

```

#### 3.3.2 LuaTeX and pdfTeX

```

540 <*luatex | pdftex>

```

```

\l__color_backend_fill_tl
\l__color_backend_stroke_tl
541 \tl_new:N \l__color_backend_fill_tl
542 \tl_new:N \l__color_backend_stroke_tl

```

(End definition for `\l__color_backend_fill_tl` and `\l__color_backend_stroke_tl`.)

Store the values then pass to the stack.

```

543 \cs_new_protected:Npn \__color_backend_select_cmyk:n #1
544   { \__color_backend_select:nn { #1 ~ k } { #1 ~ K } }
545 \cs_new_protected:Npn \__color_backend_select_gray:n #1
546   { \__color_backend_select:nn { #1 ~ g } { #1 ~ G } }
547 \cs_new_protected:Npn \__color_backend_select_rgb:n #1
548   { \__color_backend_select:nn { #1 ~ rg } { #1 ~ RG } }
549 \cs_new_protected:Npn \__color_backend_select:nn #1#2

```



```

550 {
551   \tl_set:Nn \l__color_backend_fill_tl {#1}
552   \tl_set:Nn \l__color_backend_stroke_tl {#2}
553   \__kernel_color_backend_stack_push:nn \l__color_backend_stack_int { #1 ~ #2 }
554 }
555 \cs_new_protected:Npn \__color_backend_reset:
556 { \__kernel_color_backend_stack_pop:n \l__color_backend_stack_int }

```

(End definition for \\_\_color\_backend\_select\_cmyk:n and others.)

```
557 </luatex | pdftex>
```

### 3.3.3 dvipdfmx/XqTeX

These backends have the most possible approaches: it recognises both dvips-based color specials and its own format, plus one can include PDF statements directly. Recent releases also have a color stack approach similar to pdfTeX. Of the stack methods, the dedicated the most versatile is the latter as it can cover all of the use cases we have. However, at present this interacts problematically with any color on the original stack. We therefore stick to a single-stack approach here.

```
558 <*dvipdfmx | xetex>
```

```

\__color_backend_select:n Using the single stack is relatively easy as there is only one route.
  \_color_backend_select_cmyk:n 559 \cs_new_protected:Npn \__color_backend_select:n #1
  \_color_backend_select_gray:n 560 { \__kernel_backend_literal:n { pdf : bc ~ [ #1 ] } }
  \_color_backend_select_rgb:n 561 \cs_new_eq:NN \__color_backend_select_cmyk:n \__color_backend_select:n
\__color_backend_reset: 562 \cs_new_eq:NN \__color_backend_select_gray:n \__color_backend_select:n
563 \cs_new_eq:NN \__color_backend_select_rgb:n \__color_backend_select:n
564 \cs_new_protected:Npn \__color_backend_reset:
565 { \__kernel_backend_literal:n { pdf : ec } }

```

(End definition for \\_\_color\_backend\_select:n and others.)

\\_color\_backend\_select\_named:n For classical named colors, the only value we should get is Black.

```

566 \cs_new_protected:Npn \__color_backend_select_named:n #1
567 {
568   \str_if_eq:nnTF {#1} { Black }
569     { \_color_backend_select_gray:n { 0 } }
570     { \msg_error:nnn { color } { unknown-named-color } {#1} }
571 }
572 \msg_new:nnn { color } { unknown-named-color }
573 { Named~color~'#1'~is~not~known. }

```

(End definition for \\_\_color\_backend\_select\_named:n.)

```
574 </dvipdfmx | xetex>
```

## 3.4 Separations

Here, life gets interesting and we need essentially one approach per backend.

```
575 <*dvipdfmx | luatex | pdftex | xetex | dvips>
```

But we start with some functionality needed for both PostScript and PDF based backends.

```

\g_color_backend_colorant_prop
576 \prop_new:N \g__color_backend_colorant_prop
(End definition for \g_color_backend_colorant_prop.)

\__color_backend_devicen_colorants:n
\__color_backend_devicen_colorants:w
577 \cs_new:Npx \__color_backend_devicen_colorants:n #1
578 {
579   \exp_not:N \tl_if_blank:nF {#1}
580   {
581     \c_space_tl
582     << ~
583     /Colorants ~
584     << ~
585     \exp_not:N \__color_backend_devicen_colorants:w #1 ~
586     \exp_not:N \q_recursion_tail \c_space_tl
587     \exp_not:N \q_recursion_stop
588     >> ~
589     >>
590   }
591 }
592 \cs_new:Npn \__color_backend_devicen_colorants:w #1 ~
593 {
594   \quark_if_recursion_tail_stop:n {#1}
595   \prop_if_in:NnT \g__color_backend_colorant_prop {#1}
596   {
597     #1 ~
598     \prop_item:Nn \g__color_backend_colorant_prop {#1} ~
599   }
600   \__color_backend_devicen_colorants:w
601 }
(End definition for \__color_backend_devicen_colorants:n and \__color_backend_devicen_colorants:w.)

602 </dvipdfmx | luatex | pdftex | xetex | dvips>
603 <*dvips>

\__color_backend_select_separation:nn
\__color_backend_select_devicen:nn
604 \cs_new_protected:Npn \__color_backend_select_separation:nn #1#2
605 { \__color_backend_select:n { separation ~ #1 ~ #2 } }
606 \cs_new_eq:NN \__color_backend_select_devicen:nn \__color_backend_select_separation:nn
(End definition for \__color_backend_select_separation:nn and \__color_backend_select_devicen:nn.)

\__color_backend_select_iccbased:nn No support.
607 \cs_new_protected:Npn \__color_backend_select_iccbased:nn #1#2 { }
(End definition for \__color_backend_select_iccbased:nn.)

\__color_backend_separation_init:nnnnn
\__color_backend_separation_init:nxxnn
\__color_backend_separation_init_aux:nnnnnn
\__color_backend_separation_init_/DeviceCMYK:nnnn
\__color_backend_separation_init_/DeviceGray:nnnn
\__color_backend_separation_init_/DeviceRGB:nnnn
\__color_backend_separation_init_Device:Nn
\__color_backend_separation_init:nnnn
\__color_backend_separation_init_count:n
\__color_backend_separation_init_count:w
\__color_backend_separation_init:nnnn
\__color_backend_separation_init:w
\__color_backend_separation_init:n
\__color_backend_separation_init:nw
\__color_backend_separation_init_CIELAB:nnnn

```

Initialising here means creating a small header set up plus massaging some data. This comes about as we have to deal with PDF-focussed data, which makes most sense “higher-up”. The approach is based on ideas from <https://tex.stackexchange.com/q/560093> plus using the PostScript manual for other aspects.

```

610 \bool_if:NT \g__kernel_backend_header_bool
611 {
612   \exp_args:Nx \__kernel_backend_first_shipout:n
613   {
614     \exp_not:N \__color_backend_separation_init_aux:nnnnnn
615     { \exp_not:N \int_use:N \g__color_model_int }
616     {#1} {#2} {#3} {#4} {#5}
617   }
618   \prop_gput:Nxx \exp_not:N \g__color_backend_colorant_prop
619   { / \exp_not:N \str_convert_pdfname:n {#1} }
620   {
621     << ~
622     /setcolorspace ~ {} ~
623     >> ~ begin ~
624     color \exp_not:N \int_use:N \g__color_model_int \c_space_tl
625     end
626   }
627 }
628 }
629 \cs_generate_variant:Nn \__color_backend_separation_init:nnnnn { nxx }
630 \cs_new_protected:Npn \__color_backend_separation_init_aux:nnnnnn #1#2#3#4#5#6
631 {
632   \__kernel_backend_literal:e
633   {
634     !
635     TeXDict ~ begin ~
636     /color #1
637     {
638       [ ~
639       /Separation ~ ( \str_convert_pdfname:n {#2} ) ~
640       [ ~ #3 ~ ] ~
641       {
642         \cs_if_exist_use:cF { __color_backend_separation_init_ #3 :nnn }
643         { \__color_backend_separation_init:nnn }
644         {#4} {#5} {#6}
645       }
646       ] ~ setcolorspace
647     } ~ def ~
648     end
649   }
650 }
651 \cs_new:cpn { __color_backend_separation_init_ /DeviceCMYK :nnn } #1#2#3
652 { \__color_backend_separation_init_Device:Nn 4 {#3} }
653 \cs_new:cpn { __color_backend_separation_init_ /DeviceGray :nnn } #1#2#3
654 { \__color_backend_separation_init_Device:Nn 1 {#3} }
655 \cs_new:cpn { __color_backend_separation_init_ /DeviceRGB :nnn } #1#2#3
656 { \__color_backend_separation_init_Device:Nn 2 {#3} }
657 \cs_new:Npn \__color_backend_separation_init_Device:Nn #1#2
658 {
659   #2 ~
660   \prg_replicate:nn {#1}
661   { #1 ~ index ~ mul ~ #1 ~ 1 ~ roll ~ }
662   \int_eval:n { #1 + 1 } ~ -1 ~ roll ~ pop
663 }

```

For the generic case, we cannot use `/FunctionType 2` unfortunately, so we have to code that idea up in PostScript. Here, we will therefore assume that a range is *always* given. First, we count values in each argument: at the backend level, we can assume there are always well-behaved with spaces present.

```

664 \cs_new:Npn \__color_backend_separation_init:nnn #1#2#3
665 {
666   \exp_args:Ne \__color_backend_separation_init:nnnn
667   { \__color_backend_separation_init_count:n {#2} }
668   {#1} {#2} {#3}
669 }
670 \cs_new:Npn \__color_backend_separation_init_count:n #1
671 { \int_eval:n { 0 \__color_backend_separation_init_count:w #1 ~ \s__color_stop } }
672 \cs_new:Npn \__color_backend_separation_init_count:w #1 ~ #2 \s__color_stop
673 {
674   +1
675   \tl_if_blank:nF {#2}
676   { \__color_backend_separation_init_count:w #2 \s__color_stop }
677 }

```

Now we implement the algorithm. In the terms in the PostScript manual, we have  $\mathbf{N} = 1$  and  $\mathbf{Domain} = [0 \ 1]$ , with  $\mathbf{Range}$  as  $\#2$ ,  $\mathbf{C0}$  as  $\#3$  and  $\mathbf{C1}$  as  $\#4$ , with the number of output components in  $\#1$ . So all we have to do is implement  $y_i = \mathbf{C0}_i + x(\mathbf{C1}_i - \mathbf{C0}_i)$  with lots of stack manipulation, then check the ranges. That's done by adding everything to the stack first, then using the fact we know all of the offsets. As manipulating the stack is tricky, we start by re-formatting the  $\mathbf{C0}$  and  $\mathbf{C1}$  arrays to be interleaved, and add a 0 to each pair: this is used to keep the stack of constant length while we are doing the first pass of mathematics. We then working through that list, calculating from the last to the first value before tidying up by removing all of the input values. We do that by first copying all of the final  $y$  values to the end of the stack, then rolling everything so we can pop the now-unneeded material.

```

678 \cs_new:Npn \__color_backend_separation_init:nnnn #1#2#3#4
679 {
680   \__color_backend_separation_init:w #3 ~ \s__color_stop #4 ~ \s__color_stop
681   \prg_replicate:nn {#1}
682   {
683     pop ~ 1 ~ index ~ neg ~ 1 ~ index ~ add ~
684     \int_eval:n { 3 * #1 } ~ index ~ mul ~
685     2 ~ index ~ add ~
686     \int_eval:n { 3 * #1 } ~ #1 ~ roll ~
687   }
688   \int_step_function:nnnN {#1} { -1 } { 1 }
689   \__color_backend_separation_init:n
690   \int_eval:n { 4 * #1 + 1 } ~ #1 ~ roll ~
691   \prg_replicate:nn { 3 * #1 + 1 } { pop ~ }
692   \tl_if_blank:nF {#2}
693   { \__color_backend_separation_init:nw {#1} #2 ~ \s__color_stop }
694 }
695 \cs_new:Npn \__color_backend_separation_init:w
696 #1 ~ #2 \s__color_stop #3 ~ #4 \s__color_stop
697 {
698   #1 ~ #3 ~ 0 ~
699   \tl_if_blank:nF {#2}
700   { \__color_backend_separation_init:w #2 \s__color_stop #4 \s__color_stop }

```

```

701 }
702 \cs_new:Npn \__color_backend_separation_init:n #1
703 { \int_eval:n { #1 * 2 } ~ index ~ }

```

Finally, we deal with the range limit if required. This is handled by splitting the range into pairs. It's then just a question of doing the comparisons, this time dropping everything except the desired result.

```

704 \cs_new:Npn \__color_backend_separation_init:nw #1#2 ~ #3 ~ #4 \s_color_stop
705 {
706   #2 ~ #3 ~
707   2 ~ index ~ 2 ~ index ~ lt ~
708   { ~ pop ~ exch ~ pop ~ } ~
709   { ~
710     2 ~ index ~ 1 ~ index ~ gt ~
711     { ~ exch ~ pop ~ exch ~ pop ~ } ~
712     { ~ pop ~ pop ~ } ~
713     ifelse ~
714   }
715   ifelse ~
716   #1 ~ 1 ~ roll ~
717   \tl_if_blank:nF {#4}
718   { \__color_backend_separation_init:nw {#1} #4 \s_color_stop }
719 }

```

CIELAB support uses the detail from the PostScript reference, page 227; other than that block of PostScript, this is the same as for PDF-based routes.

```

720 \cs_new_protected:Npn \__color_backend_separation_init_CIELAB:nnn #1#2#3
721 {
722   \__color_backend_separation_init:nxxxnn
723   {#2}
724   {
725     /CIEBasedABC ~
726     << ~
727     /RangeABC ~ [ ~ \c__color_model_range_CIELAB_tl \c_space_tl ] ~
728     /DecodeABC ~
729     [ ~
730       { ~ 16 ~ add ~ 116 ~ div ~ } ~ bind ~
731       { ~ 500 ~ div ~ } ~ bind ~
732       { ~ 200 ~ div ~ } ~ bind ~
733     ] ~
734     /MatrixABC ~ [ ~ 1 ~ 1 ~ 1 ~ 1 ~ 0 ~ 0 ~ 0 ~ 0 ~ -1 ~ ] ~
735     /DecodeLMN ~
736     [ ~
737     { ~
738       dup ~ 6 ~ 29 ~ div ~ ge ~
739       { ~ dup ~ dup ~ mul ~ mul ~ ~ } ~
740       { ~ 4 ~ 29 ~ div ~ sub ~ 108 ~ 841 ~ div ~ mul ~ } ~
741       ifelse ~
742       0.9505 ~ mul ~
743     } ~ bind ~
744     { ~
745       dup ~ 6 ~ 29 ~ div ~ ge ~
746       { ~ dup ~ dup ~ mul ~ mul ~ ~ } ~
747       { ~ 4 ~ 29 ~ div ~ sub ~ 108 ~ 841 ~ div ~ mul ~ } ~
748       ifelse ~

```

```

749         } ~ bind ~
750         { ~
751             dup ~ 6 ~ 29 ~ div ~ ge ~
752             { ~ dup ~ dup ~ mul ~ mul ~ } ~
753             { ~ 4 ~ 29 ~ div ~ sub ~ 108 ~ 841 ~ div ~ mul ~ } ~
754             ifelse ~
755             1.0890 ~ mul ~
756         } ~ bind
757     ] ~
758     /WhitePoint ~
759     [ ~ \tl_use:c { c__color_model_whitepoint_CIELAB_ #1 _tl } ~ ] ~
760     >>
761 }
762 { \c__color_model_range_CIELAB_tl }
763 { 100 ~ 0 ~ 0 }
764 {#3}
765 }

```

(End definition for `\__color_backend_separation_init:nnnnn` and others.)

`\__color_backend_devicen_init:nnn` Trivial as almost all of the work occurs in the shared code.

```

766 \cs_new_protected:Npn \__color_backend_devicen_init:nnn #1#2#3
767 {
768     \__kernel_backend_literal:e
769     {
770         !
771         TeXDict ~ begin ~
772         /color \int_use:N \g__color_model_int
773         {
774             [ ~
775                 /DeviceN ~
776                 [ ~ #1 ~ ] ~
777                 #2 ~
778                 { ~ #3 ~ } ~
779                 \__color_backend_devicen_colorants:n {#1}
780             ] ~ setcolorspace
781         } ~ def ~
782     end
783 }
784 }

```

(End definition for `\__color_backend_devicen_init:nnn`.)

`\__color_backend_iccbased_init:nnn` No support at present.

```

785 \cs_new_protected:Npn \__color_backend_iccbased_init:nnn #1#2#3 { }

```

(End definition for `\__color_backend_iccbased_init:nnn`.)

```

786 </dvips>
787 <*dvisvgm>

```

`\__color_backend_select_separation:nn` No support at present.

```

\__color_backend_select_devicen:nn
788 \cs_new_protected:Npn \__color_backend_select_separation:nn #1#2 { }
789 \cs_new_eq:NN \__color_backend_select_devicen:nn \__color_backend_select_separation:nn

```

(End definition for `\_color_backend_select_separation:nn` and `\_color_backend_select_devicen:nn`.)

No support at present.

```
790 \cs_new_protected:Npn \_color_backend_separation_init:nnnnn #1#2#3#4#5 { }
791 \cs_new_protected:Npn \_color_backend_separation_init_CIELAB:nnnnnn #1#2#3 { }
```

(End definition for `\_color_backend_separation_init:nnnnn` and `\_color_backend_separation_init_CIELAB:nnnn`.)

As detailed in <https://www.w3.org/TR/css-color-4/#at-profile>, we can apply a color profile using CSS. As we have a local file, we use a relative URL.

```
792 \cs_new_protected:Npn \_color_backend_select_iccbased:nn #1#2
793 {
794   \_kernel_backend_literal_svg:x
795   {
796     <style>
797       @color-profile ~
798       \str_if_eq:nnTF {#2} { cmyk }
799       { device-cmyk }
800       { --color \int_use:N \g__color_model_int }
801       \c_space_tl
802       {
803         src: (" #1 ")
804       }
805     </style>
806   }
807 }
```

(End definition for `\_color_backend_select_iccbased:nn`.)

```
808 </dvisvgm>
```

```
809 <*dviPDFmx | luatex | pdftex | xetex>
```

```
\_color_backend_select_separation:nn
\_color_backend_select_devicen:nn
\_color_backend_select_iccbased:nn
810 <*dviPDFmx | xetex>
811 \cs_new_protected:Npn \_color_backend_select_separation:nn #1#2
812 { \_kernel_backend_literal:x { pdf : bc ~ \pdf_object_ref:n {#1} ~ [ #2 ] } }
813 </dviPDFmx | xetex>
814 <*luatex | pdftex>
815 \cs_new_protected:Npn \_color_backend_select_separation:nn #1#2
816 { \_color_backend_select:nn { /#1 ~ cs ~ #2 ~ scn } { /#1 ~ CS ~ #2 ~ SCN } }
817 </luatex | pdftex>
818 \cs_new_eq:NN \_color_backend_select_devicen:nn \_color_backend_select_separation:nn
819 \cs_new_eq:NN \_color_backend_select_iccbased:nn \_color_backend_select_separation:nn
```

(End definition for `\_color_backend_select_separation:nn`, `\_color_backend_select_devicen:nn`, and `\_color_backend_select_iccbased:nn`.)

Resource initiation comes up a few times. For `dviPDFmx/XgTeX`, we skip this as at present it's handled by the backend.

```
820 \cs_new_protected:Npn \_color_backend_init_resource:n #1
821 {
822   <*luatex | pdftex>
823   \bool_lazy_and:nnT
824   { \cs_if_exist_p:N \pdfmanagement_if_active_p: }
```

```

825     { \pdfmanagement_if_active_p: }
826     {
827         \use:x
828         {
829             \pdfmanagement_add:nnn
830             { Page / Resources / ColorSpace }
831             { #1 }
832             { \pdf_object_ref_last: }
833         }
834     }
835 </luatex | pdftex>
836 }

```

(End definition for \\_color\_backend\_init\_resource:n.)

\\_color\_backend\_separation\_init:nnnnn  
\\_color\_backend\_separation\_init:nn  
\\_color\_backend\_separation\_init\_CIELAB:nnn

Initialising the PDF structures needs two parts: creating an object containing the “real” name of the Separation, then adding a reference to that to each page. We use a separate object for the tint transformation following the model in the PDF reference. The object here for the color needs to be named as that way it’s accessible to dvipdfmx/X<sub>q</sub>TeX.

```

837 \cs_new_protected:Npn \_color_backend_separation_init:nnnnn #1#2#3#4#5
838 {
839     \pdf_object_unnamed_write:nx { dict }
840     {
841         /FunctionType ~ 2
842         /Domain ~ [0 ~ 1]
843         \tl_if_blank:nF {#3} { /Range ~ [#3] }
844         /C0 ~ [#4] ~
845         /C1 ~ [#5] /N ~ 1
846     }
847     \exp_args:Nx \_color_backend_separation_init:nn
848     { \str_convert_pdfname:n {#1} } {#2}
849     \_color_backend_init_resource:n { color \int_use:N \g__color_model_int }
850 }
851 \cs_new_protected:Npn \_color_backend_separation_init:nn #1#2
852 {
853     \use:x
854     {
855         \pdf_object_new:nn { color \int_use:N \g__color_model_int } { array }
856         \pdf_object_write:nn { color \int_use:N \g__color_model_int }
857         { /Separation /#1 ~ #2 ~ \pdf_object_ref_last: }
858     }
859     \prop_gput:Nnx \g__color_backend_colorant_prop { /#1 }
860     { \pdf_object_ref_last: }
861 }

```

For CIELAB colors, we need one object per document for the illuminant, plus initialisation of the color space referencing that object.

```

862 \cs_new_protected:Npn \_color_backend_separation_init_CIELAB:nnn #1#2#3
863 {
864     \pdf_object_if_exist:nF { __color_illuminant_CIELAB_ #1 }
865     {
866         \pdf_object_new:nn { __color_illuminant_CIELAB_ #1 } { array }
867         \pdf_object_write:nx { __color_illuminant_CIELAB_ #1 }
868         {

```



```

869         /Lab ~
870         <<
871         /WhitePoint ~
872         [ \tl_use:c { c__color_model_whitepoint_CIELAB_ #1 _tl } ]
873         /Range ~ [ \c__color_model_range_CIELAB_tl ]
874         >>
875     }
876 }
877 \__color_backend_separation_init:nnnnn
878 {#2}
879 { \pdf_object_ref:n { __color_illuminant_CIELAB_ #1 } }
880 { \c__color_model_range_CIELAB_tl }
881 { 100 ~ 0 ~ 0 }
882 {#3}
883 }

```

(End definition for \\_\_color\_backend\_separation\_init:nnnnn, \\_\_color\_backend\_separation\_init:nn, and \\_\_color\_backend\_separation\_init\_CIELAB:nnn.)

\\_\_color\_backend\_devicen\_init:nnn  
\\_\_color\_backend\_devicen\_init:w

Similar to the Separations case, but with an arbitrary function for the alternative space work.

```

884 \cs_new_protected:Npn \__color_backend_devicen_init:nnn #1#2#3
885 {
886     \pdf_object_unnamed_write:nx { stream }
887     {
888         {
889             /FunctionType ~ 4 ~
890             /Domain ~
891             [ ~
892                 \prg_replicate:nn
893                 { 0 \__color_backend_devicen_init:w #1 ~ \s__color_stop }
894                 { 0 ~ 1 ~ }
895             ] ~
896             /Range ~
897             [ ~
898                 \str_case:nn {#2}
899                 {
900                     { /DeviceCMYK } { 0 ~ 1 ~ 0 ~ 1 ~ 0 ~ 1 ~ 0 ~ 1 }
901                     { /DeviceGray } { 0 ~ 1 }
902                     { /DeviceRGB } { 0 ~ 1 ~ 0 ~ 1 ~ 0 ~ 1 }
903                 } ~
904             ]
905         }
906         { {#3} }
907     }
908     \use:x
909     {
910         \pdf_object_new:nn { color \int_use:N \g__color_model_int } { array }
911         \pdf_object_write:nn { color \int_use:N \g__color_model_int }
912         {
913             /DeviceN ~
914             [ ~ #1 ~ ] ~
915             #2 ~
916             \pdf_object_ref_last:

```

```

917         \_color_backend_devicen_colorants:n {#1}
918     }
919 }
920 \_color_backend_init_resource:n { color \int_use:N \g_color_model_int }
921 }
922 \cs_new:Npn \_color_backend_devicen_init:w #1 ~ #2 \s_color_stop
923 {
924     + 1
925     \tl_if_blank:nF {#2}
926     { \_color_backend_devicen_init:w #2 \s_color_stop }
927 }

```

(End definition for \\_color\_backend\_devicen\_init:nnn and \\_color\_backend\_devicen\_init:w.)

\\_color\_backend\_iccbased\_init:nnn Lots of data to save here: we only want to do that once per file, so track it by name.

```

928 \cs_new_protected:Npn \_color_backend_iccbased_init:nnn #1#2#3
929 {
930     \pdf_object_if_exist:nF { __color_icc_ #1 }
931     {
932         \pdf_object_new:nn { __color_icc_ #1 } { fstream }
933         \pdf_object_write:nx { __color_icc_ #1 }
934         {
935             {
936                 /N ~ \exp_not:n { #2 } ~
937                 \tl_if_empty:nF { #3 } { /Range~[ #3 ] }
938             }
939             {#1}
940         }
941     }
942     \pdf_object_unnamed_write:nx { array }
943     { /ICCBased ~ \pdf_object_ref:n { __color_icc_ #1 } }
944     \_color_backend_init_resource:n { color \int_use:N \g_color_model_int }
945 }

```

(End definition for \\_color\_backend\_iccbased\_init:nnn.)

\\_color\_backend\_iccbased\_device:nnn This is very similar to setting up a color space: the only part we add to the page resources differently.

```

946 \cs_new_protected:Npn \_color_backend_iccbased_device:nnn #1#2#3
947 {
948     \pdf_object_if_exist:nF { __color_icc_ #1 }
949     {
950         \pdf_object_new:nn { __color_icc_ #1 } { fstream }
951         \pdf_object_write:nn { __color_icc_ #1 }
952         {
953             { /N ~ #3 }
954             {#1}
955         }
956     }
957     \pdf_object_unnamed_write:nx { array }
958     { /ICCBased ~ \pdf_object_ref:n { __color_icc_ #1 } }
959     \_color_backend_init_resource:n { Default #2 }
960 }

```

(End definition for \\_color\_backend\_iccbased\_device:nnn.)

```

961 </dvipdfmx | luatex | pdftex | xetex>

```

### 3.5 Fill and stroke color

Here, dvipdfmx/X<sub>Y</sub>TeX we write direct PDF specials for the fill, and only use the stack for the stroke color (see above for comments on why we cannot use multiple stacks with these backends). LuaTeX and pdfTeX have mutiple stacks that can deal with fill and stroke. For dvips we have to manage fill and stroke color ourselves. We also handle dvisvgm independently, as there we can create SVG directly.

962 <\*dvipdfmx | xetex>

```

    \_color_backend_fill:n
\_color_backend_fill_cmyk:n
\_color_backend_fill_gray:n
\_color_backend_fill_rgb:n
    \_color_backend_stroke:n
        \_color_backend_stroke_cmyk:n
        \_color_backend_stroke_gray:n
        \_color_backend_stroke_rgb:n
963 \cs_new_protected:Npn \_color_backend_fill:n #1
964 { \_kernel_backend_literal:n { pdf : bc ~ fill ~ [ #1 ] } }
965 \cs_new_eq:NN \_color_backend_fill_cmyk:n \_color_backend_fill:n
966 \cs_new_eq:NN \_color_backend_fill_gray:n \_color_backend_fill:n
967 \cs_new_eq:NN \_color_backend_fill_rgb:n \_color_backend_fill:n
968 \cs_new_protected:Npn \_color_backend_stroke:n #1
969 { \_kernel_backend_literal:n { pdf : bc ~ stroke ~ [ #1 ] } }
970 \cs_new_eq:NN \_color_backend_stroke_cmyk:n \_color_backend_stroke:n
971 \cs_new_eq:NN \_color_backend_stroke_gray:n \_color_backend_stroke:n
972 \cs_new_eq:NN \_color_backend_stroke_rgb:n \_color_backend_stroke:n

```

(End definition for \\_color\_backend\_fill:n and others.)

```

\_color_backend_fill_separation:nn
\_color_backend_stroke_separation:nn
    \_color_backend_fill_devicen:nn
    \_color_backend_stroke_devicen:nn
973 \cs_new_protected:Npn \_color_backend_fill_separation:nn #1#2
974 {
975     \_kernel_backend_literal:x
976     { pdf : bc ~ fill ~ \pdf_object_ref:n {#1} ~ [ #2 ] }
977 }
978 \cs_new_protected:Npn \_color_backend_stroke_separation:nn #1#2
979 {
980     \_kernel_backend_literal:x
981     { pdf : bc ~ stroke ~ \pdf_object_ref:n {#1} ~ [ #2 ] }
982 }
983 \cs_new_eq:NN \_color_backend_fill_devicen:nn \_color_backend_fill_separation:nn
984 \cs_new_eq:NN \_color_backend_stroke_devicen:nn \_color_backend_stroke_separation:nn

```

(End definition for \\_color\_backend\_fill\_separation:nn and others.)

```

\_color_backend_fill_reset:
    \_color_backend_stroke_reset:
985 \cs_new_eq:NN \_color_backend_fill_reset: \_color_backend_reset:
986 \cs_new_eq:NN \_color_backend_stroke_reset: \_color_backend_reset:

```

(End definition for \\_color\_backend\_fill\_reset: and \\_color\_backend\_stroke\_reset:.)

987 </dvipdfmx | xetex>

988 <\*luatex | pdftex>

\\_color\_backend\_fill\_cmyk:n Drawing (fill/stroke) color is handled in dvipdfmx/X<sub>Y</sub>TeX in the same way as LuaTeX/pdfTeX. We use the same approach as earlier, except the color stack is not involved so the generic direct PDF operation is used. There is no worry about the nature of strokes: everything is handled automatically.

```

989 \cs_new_protected:Npn \_color_backend_fill_cmyk:n #1
990 { \_color_backend_fill:n { #1 ~ k } }

```

```

    \_color_backend_stroke_cmyk:n
    \_color_backend_stroke_gray:n
    \_color_backend_stroke_rgb:n
\_color_backend_stroke:n

```

```

991 \cs_new_protected:Npn \__color_backend_fill_gray:n #1
992 { \__color_backend_fill:n { #1 ~ g } }
993 \cs_new_protected:Npn \__color_backend_fill_rgb:n #1
994 { \__color_backend_fill:n { #1 ~ rg } }
995 \cs_new_protected:Npn \__color_backend_fill:n #1
996 {
997   \tl_set:Nn \l__color_backend_fill_tl {#1}
998   \__kernel_color_backend_stack_push:nn \l__color_backend_stack_int
999   { #1 ~ \l__color_backend_stroke_tl }
1000 }
1001 \cs_new_protected:Npn \__color_backend_stroke_cmyk:n #1
1002 { \__color_backend_stroke:n { #1 ~ K } }
1003 \cs_new_protected:Npn \__color_backend_stroke_gray:n #1
1004 { \__color_backend_stroke:n { #1 ~ G } }
1005 \cs_new_protected:Npn \__color_backend_stroke_rgb:n #1
1006 { \__color_backend_stroke:n { #1 ~ RG } }
1007 \cs_new_protected:Npn \__color_backend_stroke:n #1
1008 {
1009   \tl_set:Nn \l__color_backend_stroke_tl {#1}
1010   \__kernel_color_backend_stack_push:nn \l__color_backend_stack_int
1011   { \l__color_backend_fill_tl \c_space_tl #1 }
1012 }

```

(End definition for \\_\_color\_backend\_fill\_cmyk:n and others.)

```

\__color_backend_fill_separation:nn
\__color_backend_stroke_separation:nn
\__color_backend_fill_devicen:nn
\__color_backend_stroke_devicen:nn
1013 \cs_new_protected:Npn \__color_backend_fill_separation:nn #1#2
1014 { \__color_backend_fill:n { /#1 ~ cs ~ #2 ~ scn } }
1015 \cs_new_protected:Npn \__color_backend_stroke_separation:nn #1#2
1016 { \__color_backend_stroke:n { /#1 ~ CS ~ #2 ~ SCN } }
1017 \cs_new_eq:NN \__color_backend_fill_devicen:nn \__color_backend_fill_separation:nn
1018 \cs_new_eq:NN \__color_backend_stroke_devicen:nn \__color_backend_stroke_separation:nn

```

(End definition for \\_\_color\_backend\_fill\_separation:nn and others.)

```

\__color_backend_fill_reset:
\__color_backend_stroke_reset:
1019 \cs_new_eq:NN \__color_backend_fill_reset: \__color_backend_reset:
1020 \cs_new_eq:NN \__color_backend_stroke_reset: \__color_backend_reset:

```

(End definition for \\_\_color\_backend\_fill\_reset: and \\_\_color\_backend\_stroke\_reset:.)

1021 </luatex | pdftex>

1022 <\*dvips>

\\_\_color\_backend\_fill\_cmyk:n Fill color here is the same as general color *except* we skip the stroke part.

```

\__color_backend_fill_gray:n
\__color_backend_fill_rgb:n
\__color_backend_fill:n
\__color_backend_stroke_cmyk:n
\__color_backend_stroke_gray:n
\__color_backend_stroke_rgb:n
1023 \cs_new_protected:Npn \__color_backend_fill_cmyk:n #1
1024 { \__color_backend_fill:n { cmyk ~ #1 } }
1025 \cs_new_protected:Npn \__color_backend_fill_gray:n #1
1026 { \__color_backend_fill:n { gray ~ #1 } }
1027 \cs_new_protected:Npn \__color_backend_fill_rgb:n #1
1028 { \__color_backend_fill:n { rgb ~ #1 } }
1029 \cs_new_protected:Npn \__color_backend_fill:n #1
1030 {
1031   \__kernel_backend_literal:n { color~push~ #1 }
1032 }

```

```

1033 \cs_new_protected:Npn \__color_backend_stroke_cmyk:n #1
1034 { \__kernel_backend_postscript:n { /color.sc { #1 ~ setcmykcolor } def } }
1035 \cs_new_protected:Npn \__color_backend_stroke_gray:n #1
1036 { \__kernel_backend_postscript:n { /color.sc { #1 ~ setgray } def } }
1037 \cs_new_protected:Npn \__color_backend_stroke_rgb:n #1
1038 { \__kernel_backend_postscript:n { /color.sc { #1 ~ setrgbcolor } def } }

```

(End definition for \\_\_color\_backend\_fill\_cmyk:n and others.)

```

\__color_backend_fill_separation:nn
\__color_backend_stroke_separation:nn
\__color_backend_fill_devicen:nn
\__color_backend_stroke_devicen:nn
1039 \cs_new_protected:Npn \__color_backend_fill_separation:nn #1#2
1040 { \__color_backend_fill:n { separation ~ #1 ~ #2 } }
1041 \cs_new_protected:Npn \__color_backend_stroke_separation:nn #1#2
1042 { \__kernel_backend_postscript:n { /color.sc { separation ~ #1 ~ #2 } def } }
1043 \cs_new_eq:NN \__color_backend_fill_devicen:nn \__color_backend_fill_separation:nn
1044 \cs_new_eq:NN \__color_backend_stroke_devicen:nn \__color_backend_stroke_separation:nn

```

(End definition for \\_\_color\_backend\_fill\_separation:nn and others.)

```

\__color_backend_fill_reset:
\__color_backend_stroke_reset:
1045 \cs_new_eq:NN \__color_backend_fill_reset: \__color_backend_reset:
1046 \cs_new_protected:Npn \__color_backend_stroke_reset: { }
(End definition for \__color_backend_fill_reset: and \__color_backend_stroke_reset:.)
1047 </dvips>
1048 <*dvisvgm>

```

\\_\_color\_backend\_fill\_cmyk:n Fill color here is the same as general color *except* we skip the stroke part.

```

\__color_backend_fill_gray:n
\__color_backend_fill_rgb:n
\__color_backend_fill:n
1049 \cs_new_protected:Npn \__color_backend_fill_cmyk:n #1
1050 { \__color_backend_fill:n { cmyk ~ #1 } }
1051 \cs_new_protected:Npn \__color_backend_fill_gray:n #1
1052 { \__color_backend_fill:n { gray ~ #1 } }
1053 \cs_new_protected:Npn \__color_backend_fill_rgb:n #1
1054 { \__color_backend_fill:n { rgb ~ #1 } }
1055 \cs_new_protected:Npn \__color_backend_fill:n #1
1056 {
1057   \__kernel_backend_literal:n { color~push~ #1 }
1058 }

```

(End definition for \\_\_color\_backend\_fill\_cmyk:n and others.)

\\_\_color\_backend\_stroke\_cmyk:n For drawings in SVG, we use scopes for all stroke colors. That requires using RGB values, which luckily are easy to convert here (cmyk to RGB is a fixed function).

```

\__color_backend_stroke_cmyk:w
\__color_backend_stroke_gray:n
\__color_backend_stroke_gray_aux:n
\__color_backend_stroke_rgb:n
\__color_backend_stroke_rgb:w
\__color_backend:nnn
1059 \cs_new_protected:Npn \__color_backend_stroke_cmyk:n #1
1060 { \__color_backend_cmyk:w #1 \s__color_stop }
1061 \cs_new_protected:Npn \__color_backend_stroke_cmyk:w
1062 #1 ~ #2 ~ #3 ~ #4 \s__color_stop
1063 {
1064   \use:x
1065   {
1066     \__color_backend:nnn
1067     { \fp_eval:n { -100 * ( 1 - min ( 1 , #1 + #4 ) ) } }
1068     { \fp_eval:n { -100 * ( 1 - min ( 1 , #2 + #4 ) ) } }
1069     { \fp_eval:n { -100 * ( 1 - min ( 1 , #3 + #4 ) ) } }

```

```

1070     }
1071 }
1072 \cs_new_protected:Npn \__color_backend_stroke_gray:n #1
1073 {
1074   \use:x
1075   {
1076     \__color_backend_stroke_gray_aux:n
1077     { \fp_eval:n { 100 * (#1) } }
1078   }
1079 }
1080 \cs_new_protected:Npn \__color_backend_stroke_gray_aux:n #1
1081 { \__color_backend:nnn {#1} {#1} {#1} }
1082 \cs_new_protected:Npn \__color_backend_stroke_rgb:n #1
1083 { \__color_backend_rgb:w #1 \s__color_stop }
1084 \cs_new_protected:Npn \__color_backend_stroke_rgb:w
1085 #1 ~ #2 ~ #3 \s__color_stop
1086 {
1087   \use:x
1088   {
1089     \__color_backend:nnn
1090     { \fp_eval:n { 100 * (#1) } }
1091     { \fp_eval:n { 100 * (#2) } }
1092     { \fp_eval:n { 100 * (#3) } }
1093   }
1094 }
1095 \cs_new_protected:Npx \__color_backend:nnn #1#2#3
1096 {
1097   \__kernel_backend_scope:n
1098   {
1099     stroke =
1100     "
1101     rgb
1102     (
1103       #1 \c_percent_str ,
1104       #2 \c_percent_str ,
1105       #3 \c_percent_str
1106     )
1107     "
1108   }
1109 }

```

(End definition for \\_\_color\_backend\_stroke\_cmyk:n and others.)

At present, these are no-ops.

```

\__color_backend_fill_separation:nn
\__color_backend_stroke_separation:nn
\__color_backend_fill_devicen:nn
\__color_backend_stroke_devicen:nn
1110 \cs_new_protected:Npn \__color_backend_fill_separation:nn #1#2 { }
1111 \cs_new_protected:Npn \__color_backend_stroke_separation:nn #1#2 { }
1112 \cs_new_eq:NN \__color_backend_fill_devicen:nn \__color_backend_fill_separation:nn
1113 \cs_new_eq:NN \__color_backend_stroke_devicen:nn \__color_backend_stroke_separation:nn

```

(End definition for \\_\_color\_backend\_fill\_separation:nn and others.)

```

\__color_backend_fill_reset:
\__color_backend_stroke_reset:
1114 \cs_new_eq:NN \__color_backend_fill_reset: \__color_backend_reset:
1115 \cs_new_protected:Npn \__color_backend_stroke_reset: { }

```

(End definition for `\_color_backend_fill_reset:` and `\_color_backend_stroke_reset:.`)

`\_color_backend_devicen_init:nnn` No support at present.

`\_color_backend_iccbased_init:nnn`   
1116 `\cs_new_protected:Npn \_color_backend_devicen_init:nnn #1#2#3 { }`  
1117 `\cs_new_protected:Npn \_color_backend_iccbased_init:nnn #1#2#3 { }`

(End definition for `\_color_backend_devicen_init:nnn` and `\_color_backend_iccbased_init:nnn.`)

1118 `</dvisvgm>`

1119 `</package>`

## 4 l3backend-draw Implementation

1120 `<*package>`

1121 `<@@=draw>`

### 4.1 dvips backend

1122 `<*dvips>`

`\_draw_backend_literal:n` The same as literal PostScript: same arguments about positioning apply her.

`\_draw_backend_literal:x`   
1123 `\cs_new_eq:NN \_draw_backend_literal:n \_kernel_backend_literal_postscript:n`  
1124 `\cs_generate_variant:Nn \_draw_backend_literal:n { x }`

(End definition for `\_draw_backend_literal:n.`)

`\_draw_backend_begin:` The `ps::[begin]` special here deals with positioning but allows us to continue on to a  
`\_draw_backend_end:` matching `ps::[end]`: contrast with `ps:`, which positions but where we can't split material between separate calls. The `@beginspecial/@endspecial` pair are from `special.pro` and correct the scale and *y*-axis direction. In contrast to `pgf`, we don't save the current point: discussion with Tom Rokici suggested a better way to handle the necessary translations (see `\_draw_backend_box_use:Nnnnn`). (Note that `@beginspecial/@endspecial` forms a backend scope.) The `[begin]/[end]` lines are handled differently from the rest as they are conceptually different: not really drawing literals but instructions to `dvips` itself.

1125 `\cs_new_protected:Npn \_draw_backend_begin:`  
1126 `{`  
1127 `\_kernel_backend_literal:n { ps::[begin] }`  
1128 `\_draw_backend_literal:n { @beginspecial }`  
1129 `}`  
1130 `\cs_new_protected:Npn \_draw_backend_end:`  
1131 `{`  
1132 `\_draw_backend_literal:n { @endspecial }`  
1133 `\_kernel_backend_literal:n { ps::[end] }`  
1134 `}`

(End definition for `\_draw_backend_begin:` and `\_draw_backend_end:.`)

`\_draw_backend_scope_begin:` Scope here may need to contain saved definitions, so the entire memory rather than just  
`\_draw_backend_scope_end:` the graphic state has to be sent to the stack.

1135 `\cs_new_protected:Npn \_draw_backend_scope_begin:`  
1136 `{ \_draw_backend_literal:n { save } }`  
1137 `\cs_new_protected:Npn \_draw_backend_scope_end:`  
1138 `{ \_draw_backend_literal:n { restore } }`

(End definition for `\_draw_backend_scope_begin:` and `\_draw_backend_scope_end:`.)

`\_draw_backend_moveto:nn` Path creation operations mainly resolve directly to PostScript primitive steps, with only the need to convert to bp. Notice that x-type expansion is included here to ensure that any variable values are forced to literals before any possible caching. There is no native rectangular path command (without also clipping, filling or stroking), so that task is done using a small amount of PostScript.

```

1139 \cs_new_protected:Npn \_draw_backend_moveto:nn #1#2
1140 {
1141   \_draw_backend_literal:x
1142   {
1143     \dim_to_decimal_in_bp:n {#1} ~
1144     \dim_to_decimal_in_bp:n {#2} ~ moveto
1145   }
1146 }
1147 \cs_new_protected:Npn \_draw_backend_lineto:nn #1#2
1148 {
1149   \_draw_backend_literal:x
1150   {
1151     \dim_to_decimal_in_bp:n {#1} ~
1152     \dim_to_decimal_in_bp:n {#2} ~ lineto
1153   }
1154 }
1155 \cs_new_protected:Npn \_draw_backend_rectangle:nnnn #1#2#3#4
1156 {
1157   \_draw_backend_literal:x
1158   {
1159     \dim_to_decimal_in_bp:n {#4} ~ \dim_to_decimal_in_bp:n {#3} ~
1160     \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~
1161     moveto~dup~0~rlineto~exch~0~exch~rlineto~neg~0~rlineto~closepath
1162   }
1163 }
1164 \cs_new_protected:Npn \_draw_backend_curveto:nnnnnn #1#2#3#4#5#6
1165 {
1166   \_draw_backend_literal:x
1167   {
1168     \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~
1169     \dim_to_decimal_in_bp:n {#3} ~ \dim_to_decimal_in_bp:n {#4} ~
1170     \dim_to_decimal_in_bp:n {#5} ~ \dim_to_decimal_in_bp:n {#6} ~
1171     curveto
1172   }
1173 }

```

(End definition for `\_draw_backend_moveto:nn` and others.)

`\_draw_backend_evenodd_rule:` The even-odd rule here can be implemented as a simply switch.

```

\g__draw_draw_eor_bool 1174 \cs_new_protected:Npn \_draw_backend_evenodd_rule:
1175 { \bool_gset_true:N \g__draw_draw_eor_bool }
1176 \cs_new_protected:Npn \_draw_backend_nonzero_rule:
1177 { \bool_gset_false:N \g__draw_draw_eor_bool }
1178 \bool_new:N \g__draw_draw_eor_bool

```

(End definition for `\_draw_backend_evenodd_rule:`, `\_draw_backend_nonzero_rule:`, and `\g__draw_draw_eor_bool`.)



`\__draw_backend_closepath:` Unlike PDF, PostScript doesn't track separate colors for strokes and other elements. It is  
`\__draw_backend_stroke:` also desirable to have the `clip` keyword after a stroke or fill. To achieve those outcomes,  
`\__draw_backend_closestroke:` there is some work to do. For color, the stroke color is simple but the fill one has to be  
`\__draw_backend_fill:` inserted by hand. For clipping, the required ordering is achieved using a  $\TeX$  switch.  
`\__draw_backend_fillstroke:` All of the operations end with a new path instruction as they do not terminate (again in  
`\__draw_backend_clip:` contrast to PDF).  
`\__draw_backend_discardpath:`  
`\g__draw_draw_clip_bool`

```

1179 \cs_new_protected:Npn \__draw_backend_closepath:
1180 { \__draw_backend_literal:n { closepath } }
1181 \cs_new_protected:Npn \__draw_backend_stroke:
1182 {
1183   \__draw_backend_literal:n { gsave }
1184   \__draw_backend_literal:n { color.sc }
1185   \__draw_backend_literal:n { stroke }
1186   \__draw_backend_literal:n { grestore }
1187   \bool_if:NT \g__draw_draw_clip_bool
1188   {
1189     \__draw_backend_literal:x
1190     {
1191       \bool_if:NT \g__draw_draw_eor_bool { eo }
1192       clip
1193     }
1194   }
1195   \__draw_backend_literal:n { newpath }
1196   \bool_gset_false:N \g__draw_draw_clip_bool
1197 }
1198 \cs_new_protected:Npn \__draw_backend_closestroke:
1199 {
1200   \__draw_backend_closepath:
1201   \__draw_backend_stroke:
1202 }
1203 \cs_new_protected:Npn \__draw_backend_fill:
1204 {
1205   \__draw_backend_literal:x
1206   {
1207     \bool_if:NT \g__draw_draw_eor_bool { eo }
1208     fill
1209   }
1210   \bool_if:NT \g__draw_draw_clip_bool
1211   {
1212     \__draw_backend_literal:x
1213     {
1214       \bool_if:NT \g__draw_draw_eor_bool { eo }
1215       clip
1216     }
1217   }
1218   \__draw_backend_literal:n { newpath }
1219   \bool_gset_false:N \g__draw_draw_clip_bool
1220 }
1221 \cs_new_protected:Npn \__draw_backend_fillstroke:
1222 {
1223   \__draw_backend_literal:x
1224   {
1225     \bool_if:NT \g__draw_draw_eor_bool { eo }

```

```

1226         fill
1227     }
1228     \__draw_backend_literal:n { gsave }
1229     \__draw_backend_literal:n { color.sc }
1230     \__draw_backend_literal:n { stroke }
1231     \__draw_backend_literal:n { grestore }
1232     \bool_if:NT \g__draw_draw_clip_bool
1233     {
1234         \__draw_backend_literal:x
1235         {
1236             \bool_if:NT \g__draw_draw_eor_bool { eo }
1237             clip
1238         }
1239     }
1240     \__draw_backend_literal:n { newpath }
1241     \bool_gset_false:N \g__draw_draw_clip_bool
1242 }
1243 \cs_new_protected:Npn \__draw_backend_clip:
1244 { \bool_gset_true:N \g__draw_draw_clip_bool }
1245 \bool_new:N \g__draw_draw_clip_bool
1246 \cs_new_protected:Npn \__draw_backend_discardpath:
1247 {
1248     \bool_if:NT \g__draw_draw_clip_bool
1249     {
1250         \__draw_backend_literal:x
1251         {
1252             \bool_if:NT \g__draw_draw_eor_bool { eo }
1253             clip
1254         }
1255     }
1256     \__draw_backend_literal:n { newpath }
1257     \bool_gset_false:N \g__draw_draw_clip_bool
1258 }

```

(End definition for \\_\_draw\_backend\_closepath: and others.)

Converting paths to output is again a case of mapping directly to PostScript operations.

```

\__draw_backend_dash_pattern:nn
\__draw_backend_dash:n
\__draw_backend_linewidth:n
\__draw_backend_miterlimit:n
\__draw_backend_cap_butt:
\__draw_backend_cap_round:
\__draw_backend_cap_rectangle:
\__draw_backend_join_miter:
\__draw_backend_join_round:
\__draw_backend_join_bevel:
1259 \cs_new_protected:Npn \__draw_backend_dash_pattern:nn #1#2
1260 {
1261     \__draw_backend_literal:x
1262     {
1263         [
1264             \exp_args:Nf \use:n
1265             { \clist_map_function:nN {#1} \__draw_backend_dash:n }
1266         ] ~
1267         \dim_to_decimal_in_bp:n {#2} ~ setdash
1268     }
1269 }
1270 \cs_new:Npn \__draw_backend_dash:n #1
1271 { ~ \dim_to_decimal_in_bp:n {#1} }
1272 \cs_new_protected:Npn \__draw_backend_linewidth:n #1
1273 {
1274     \__draw_backend_literal:x
1275     { \dim_to_decimal_in_bp:n {#1} ~ setlinewidth }

```

```

1276 }
1277 \cs_new_protected:Npn \__draw_backend_miterlimit:n #1
1278 { \__draw_backend_literal:n { #1 ~ setmiterlimit } }
1279 \cs_new_protected:Npn \__draw_backend_cap_butt:
1280 { \__draw_backend_literal:n { 0 ~ setlinecap } }
1281 \cs_new_protected:Npn \__draw_backend_cap_round:
1282 { \__draw_backend_literal:n { 1 ~ setlinecap } }
1283 \cs_new_protected:Npn \__draw_backend_cap_rectangle:
1284 { \__draw_backend_literal:n { 2 ~ setlinecap } }
1285 \cs_new_protected:Npn \__draw_backend_join_miter:
1286 { \__draw_backend_literal:n { 0 ~ setlinejoin } }
1287 \cs_new_protected:Npn \__draw_backend_join_round:
1288 { \__draw_backend_literal:n { 1 ~ setlinejoin } }
1289 \cs_new_protected:Npn \__draw_backend_join_bevel:
1290 { \__draw_backend_literal:n { 2 ~ setlinejoin } }

```

(End definition for `\__draw_backend_dash_pattern:nn` and others.)

`\__draw_backend_cm:nnnn`

In `dvips`, keeping the transformations in line with the engine is unfortunately not possible for scaling and rotations: even if we decompose the matrix into those operations, there is still no backend tracking (cf. `dvipdfmx/XYTeX`). Thus we take the shortest path available and simply dump the matrix as given.

```

1291 \cs_new_protected:Npn \__draw_backend_cm:nnnn #1#2#3#4
1292 {
1293   \__draw_backend_literal:n
1294   { [ #1 ~ #2 ~ #3 ~ #4 ~ 0 ~ 0 ] ~ concat }
1295 }

```

(End definition for `\__draw_backend_cm:nnnn`.)

`\__draw_backend_box_use:Nnnnn`

Inside a picture `@beginspecial/@endspecial` are active, which is normally a good thing but means that the position and scaling would be off if the box was inserted directly. To deal with that, there are a number of possible approaches. The implementation here was suggested by Tom Rokici (author of `dvips`). We end the current special placement, then set the current point with a literal `[begin]`. As for general literals, we then use the stack to store the current point and move to it. To insert the required transformation, we have to flip the  $y$ -axis, once before and once after it. Then we get back to the `TeX` reference point to insert our content. The clean up has to happen in the right places, hence the `[begin]/[end]` pair around `restore`. Finally, we can return to “normal” drawing mode. Notice that the set up here is very similar to that in `\__draw_align_currentpoint_...`, but the ordering of saving and restoring is different (intermixed).

```

1296 \cs_new_protected:Npn \__draw_backend_box_use:Nnnnn #1#2#3#4#5
1297 {
1298   \__draw_backend_literal:n { @endspecial }
1299   \__draw_backend_literal:n { [end] }
1300   \__draw_backend_literal:n { [begin] }
1301   \__draw_backend_literal:n { save }
1302   \__draw_backend_literal:n { currentpoint }
1303   \__draw_backend_literal:n { currentpoint~translate }
1304   \__draw_backend_cm:nnnn { 1 } { 0 } { 0 } { -1 }
1305   \__draw_backend_cm:nnnn { #2 } { #3 } { #4 } { #5 }
1306   \__draw_backend_cm:nnnn { 1 } { 0 } { 0 } { -1 }
1307   \__draw_backend_literal:n { neg~exch~neg~exch~translate }

```

```

1308     \__draw_backend_literal:n { [end] }
1309     \hbox_overlap_right:n { \box_use:N #1 }
1310     \__draw_backend_literal:n { [begin] }
1311     \__draw_backend_literal:n { restore }
1312     \__draw_backend_literal:n { [end] }
1313     \__draw_backend_literal:n { [begin] }
1314     \__draw_backend_literal:n { @beginspecial }
1315 }

```

(End definition for \\_\_draw\_backend\_box\_use:Nnnnn.)

```

1316 </dvips>

```

## 4.2 LuaTeX, pdfTeX, dvipdfmx and XeTeX

LuaTeX, pdfTeX, dvipdfmx and XeTeX directly produce PDF output and understand a shared set of specials for drawing commands.

```

1317 <*dvipdfmx | luatex | pdftex | xetex>

```

### 4.2.1 Drawing

\\_\_draw\_backend\_literal:n Pass data through using a dedicated interface.

```

\__draw_backend_literal:x
1318 \cs_new_eq:NN \__draw_backend_literal:n \__kernel_backend_literal_pdf:n
1319 \cs_generate_variant:Nn \__draw_backend_literal:n { x }

```

(End definition for \\_\_draw\_backend\_literal:n.)

\\_\_draw\_backend\_begin: No special requirements here, so simply set up a drawing scope.

```

\__draw_backend_end:
1320 \cs_new_protected:Npn \__draw_backend_begin:
1321 { \__draw_backend_scope_begin: }
1322 \cs_new_protected:Npn \__draw_backend_end:
1323 { \__draw_backend_scope_end: }

```

(End definition for \\_\_draw\_backend\_begin: and \\_\_draw\_backend\_end:.)

\\_\_draw\_backend\_scope\_begin: Use the backend-level scope mechanisms.

```

\__draw_backend_scope_end:
1324 \cs_new_eq:NN \__draw_backend_scope_begin: \__kernel_backend_scope_begin:
1325 \cs_new_eq:NN \__draw_backend_scope_end: \__kernel_backend_scope_end:

```

(End definition for \\_\_draw\_backend\_scope\_begin: and \\_\_draw\_backend\_scope\_end:.)

\\_\_draw\_backend\_moveto:nn Path creation operations all resolve directly to PDF primitive steps, with only the need to convert to bp.

```

\__draw_backend_lineto:nn
\__draw_backend_moveto:nnnnnn
\__draw_backend_rectangle:nnnn
1326 \cs_new_protected:Npn \__draw_backend_moveto:nn #1#2
1327 {
1328     \__draw_backend_literal:x
1329     { \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~ m }
1330 }
1331 \cs_new_protected:Npn \__draw_backend_lineto:nn #1#2
1332 {
1333     \__draw_backend_literal:x
1334     { \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~ l }
1335 }
1336 \cs_new_protected:Npn \__draw_backend_curveto:nnnnnn #1#2#3#4#5#6
1337 {

```

```

1338   \_draw_backend_literal:x
1339   {
1340     \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~
1341     \dim_to_decimal_in_bp:n {#3} ~ \dim_to_decimal_in_bp:n {#4} ~
1342     \dim_to_decimal_in_bp:n {#5} ~ \dim_to_decimal_in_bp:n {#6} ~
1343     c
1344   }
1345 }
1346 \cs_new_protected:Npn \_draw_backend_rectangle:nnnn #1#2#3#4
1347 {
1348   \_draw_backend_literal:x
1349   {
1350     \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~
1351     \dim_to_decimal_in_bp:n {#3} ~ \dim_to_decimal_in_bp:n {#4} ~
1352     re
1353   }
1354 }

```

(End definition for `\_draw_backend_moveto:nn` and others.)

`\_draw_backend_evenodd_rule:` The even-odd rule here can be implemented as a simply switch.

```

1355 \cs_new_protected:Npn \_draw_backend_evenodd_rule:
1356 { \bool_gset_true:N \g__draw_draw_eor_bool }
1357 \cs_new_protected:Npn \_draw_backend_nonzero_rule:
1358 { \bool_gset_false:N \g__draw_draw_eor_bool }
1359 \bool_new:N \g__draw_draw_eor_bool

```

(End definition for `\_draw_backend_evenodd_rule:`, `\_draw_backend_nonzero_rule:`, and `\g__draw_draw_eor_bool`.)

`\_draw_backend_closepath:` Converting paths to output is again a case of mapping directly to PDF operations.

```

1360 \cs_new_protected:Npn \_draw_backend_closepath:
1361 { \_draw_backend_literal:n { h } }
1362 \cs_new_protected:Npn \_draw_backend_stroke:
1363 { \_draw_backend_literal:n { S } }
1364 \cs_new_protected:Npn \_draw_backend_closestroke:
1365 { \_draw_backend_literal:n { s } }
1366 \cs_new_protected:Npn \_draw_backend_fill:
1367 {
1368   \_draw_backend_literal:x
1369   { f \bool_if:NT \g__draw_draw_eor_bool * }
1370 }
1371 \cs_new_protected:Npn \_draw_backend_fillstroke:
1372 {
1373   \_draw_backend_literal:x
1374   { B \bool_if:NT \g__draw_draw_eor_bool * }
1375 }
1376 \cs_new_protected:Npn \_draw_backend_clip:
1377 {
1378   \_draw_backend_literal:x
1379   { W \bool_if:NT \g__draw_draw_eor_bool * }
1380 }
1381 \cs_new_protected:Npn \_draw_backend_discardpath:
1382 { \_draw_backend_literal:n { n } }

```

(End definition for `\_draw_backend_closepath:` and others.)

Converting paths to output is again a case of mapping directly to PDF operations.

```

\__draw_backend_dash_pattern:nn
\__draw_backend_dash:n
\__draw_backend_linewidth:n
\__draw_backend_miterlimit:n
\__draw_backend_cap_but:
\__draw_backend_cap_round:
\__draw_backend_cap_rectangle:
\__draw_backend_join_miter:
\__draw_backend_join_round:
\__draw_backend_join_bevel:
1383 \cs_new_protected:Npn \__draw_backend_dash_pattern:nn #1#2
1384 {
1385   \__draw_backend_literal:x
1386   {
1387     [
1388       \exp_args:Nf \use:n
1389       { \clist_map_function:nN {#1} \__draw_backend_dash:n }
1390     ] ~
1391     \dim_to_decimal_in_bp:n {#2} ~ d
1392   }
1393 }
1394 \cs_new:Npn \__draw_backend_dash:n #1
1395 { ~ \dim_to_decimal_in_bp:n {#1} }
1396 \cs_new_protected:Npn \__draw_backend_linewidth:n #1
1397 {
1398   \__draw_backend_literal:x
1399   { \dim_to_decimal_in_bp:n {#1} ~ w }
1400 }
1401 \cs_new_protected:Npn \__draw_backend_miterlimit:n #1
1402 { \__draw_backend_literal:x { #1 ~ M } }
1403 \cs_new_protected:Npn \__draw_backend_cap_but:
1404 { \__draw_backend_literal:n { 0 ~ J } }
1405 \cs_new_protected:Npn \__draw_backend_cap_round:
1406 { \__draw_backend_literal:n { 1 ~ J } }
1407 \cs_new_protected:Npn \__draw_backend_cap_rectangle:
1408 { \__draw_backend_literal:n { 2 ~ J } }
1409 \cs_new_protected:Npn \__draw_backend_join_miter:
1410 { \__draw_backend_literal:n { 0 ~ j } }
1411 \cs_new_protected:Npn \__draw_backend_join_round:
1412 { \__draw_backend_literal:n { 1 ~ j } }
1413 \cs_new_protected:Npn \__draw_backend_join_bevel:
1414 { \__draw_backend_literal:n { 2 ~ j } }

```

(End definition for `\_draw_backend_dash_pattern:nn` and others.)

```

\__draw_backend_cm:nnnn
\__draw_backend_cm_aux:nnnn

```

Another split here between LuaTeX/pdfTeX and dvipdfmx/X<sub>Y</sub>TeX. In the former, we have a direct method to maintain alignment: the backend can use a matrix itself. For dvipdfmx/X<sub>Y</sub>TeX, we can to decompose the matrix into rotations and a scaling, then use those operations as they are handled by the backend. (There is backend support for matrix operations in dvipdfmx/X<sub>Y</sub>TeX, but as a matched pair so not suitable for the “stand alone” transformation set up here.) The specials used here are from xdvipdfmx originally: they are well-tested, but probably equivalent to the pdf<sub>+</sub> versions!

```

1415 \cs_new_protected:Npn \__draw_backend_cm:nnnn #1#2#3#4
1416 {
1417   <*luatex | pdftex>
1418   \__kernel_backend_matrix:n { #1 ~ #2 ~ #3 ~ #4 }
1419   </luatex | pdftex>
1420   <*dvipdfmx | xetex>
1421   \__draw_backend_cm_decompose:nnnnN {#1} {#2} {#3} {#4}
1422   \__draw_backend_cm_aux:nnnn
1423   </dvipdfmx | xetex>

```

```

1424 }
1425 <*dvipdfmx | xetex>
1426 \cs_new_protected:Npn \__draw_backend_cm_aux:nnnn #1#2#3#4
1427 {
1428   \__kernel_backend_literal:x
1429   {
1430     x:rotate~
1431     \fp_compare:nNnTF {#1} = \c_zero_fp
1432     { 0 }
1433     { \fp_eval:n { round ( -#1 , 5 ) } }
1434   }
1435   \__kernel_backend_literal:x
1436   {
1437     x:scale~
1438     \fp_eval:n { round ( #2 , 5 ) } ~
1439     \fp_eval:n { round ( #3 , 5 ) }
1440   }
1441   \__kernel_backend_literal:x
1442   {
1443     x:rotate~
1444     \fp_compare:nNnTF {#4} = \c_zero_fp
1445     { 0 }
1446     { \fp_eval:n { round ( -#4 , 5 ) } }
1447   }
1448 }
1449 </dvipdfmx | xetex>

```

(End definition for \\_\_draw\_backend\_cm:nnnn and \\_\_draw\_backend\_cm\_aux:nnnn.)

```

\__draw_backend_cm_decompose:nnnnN
\__draw_backend_cm_decompose_auxi:nnnnN
\__draw_backend_cm_decompose_auxii:nnnnN
\__draw_backend_cm_decompose_auxiii:nnnnN

```

Internally, transformations for drawing are tracked as a matrix. Not all engines provide a way of dealing with this: if we use a raw matrix, the engine loses track of positions (for example for hyperlinks), and this is not desirable. They do, however, allow us to track rotations and scalings. Luckily, we can decompose any (two-dimensional) matrix into two rotations and a single scaling:

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} = \begin{bmatrix} \cos \beta & \sin \beta \\ -\sin \beta & \cos \beta \end{bmatrix} \begin{bmatrix} w_1 & 0 \\ 0 & w_2 \end{bmatrix} \begin{bmatrix} \cos \gamma & \sin \gamma \\ -\sin \gamma & \cos \gamma \end{bmatrix}$$

The parent matrix can be converted to

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} = \begin{bmatrix} E & H \\ -H & E \end{bmatrix} + \begin{bmatrix} F & G \\ G & -F \end{bmatrix}$$

From these, we can find that

$$\begin{aligned} \frac{w_1 + w_2}{2} &= \sqrt{E^2 + H^2} \\ \frac{w_1 - w_2}{2} &= \sqrt{F^2 + G^2} \\ \gamma - \beta &= \tan^{-1}(G/F) \\ \gamma + \beta &= \tan^{-1}(H/E) \end{aligned}$$

at which point we just have to do various pieces of re-arrangement to get all of the values. (See J. Blinn, *IEEE Comput. Graph. Appl.*, 1996, **16**, 82–88.) There is one wrinkle: the

PostScript (and PDF) way of specifying a transformation matrix exchanges where one would normally expect  $B$  and  $C$  to be.

```

1450 <*dvipdfmx | xetex>
1451 \cs_new_protected:Npn \__draw_backend_cm_decompose:nnnnN #1#2#3#4#5
1452 {
1453   \use:x
1454   {
1455     \__draw_backend_cm_decompose_auxi:nnnnN
1456     { \fp_eval:n { (#1 + #4) / 2 } }
1457     { \fp_eval:n { (#1 - #4) / 2 } }
1458     { \fp_eval:n { (#3 + #2) / 2 } }
1459     { \fp_eval:n { (#3 - #2) / 2 } }
1460   }
1461   #5
1462 }
1463 \cs_new_protected:Npn \__draw_backend_cm_decompose_auxi:nnnnN #1#2#3#4#5
1464 {
1465   \use:x
1466   {
1467     \__draw_backend_cm_decompose_auxii:nnnnN
1468     { \fp_eval:n { 2 * sqrt ( #1 * #1 + #4 * #4 ) } }
1469     { \fp_eval:n { 2 * sqrt ( #2 * #2 + #3 * #3 ) } }
1470     { \fp_eval:n { atand ( #3 , #2 ) } }
1471     { \fp_eval:n { atand ( #4 , #1 ) } }
1472   }
1473   #5
1474 }
1475 \cs_new_protected:Npn \__draw_backend_cm_decompose_auxii:nnnnN #1#2#3#4#5
1476 {
1477   \use:x
1478   {
1479     \__draw_backend_cm_decompose_auxiii:nnnnN
1480     { \fp_eval:n { ( #4 - #3 ) / 2 } }
1481     { \fp_eval:n { ( #1 + #2 ) / 2 } }
1482     { \fp_eval:n { ( #1 - #2 ) / 2 } }
1483     { \fp_eval:n { ( #4 + #3 ) / 2 } }
1484   }
1485   #5
1486 }
1487 \cs_new_protected:Npn \__draw_backend_cm_decompose_auxiii:nnnnN #1#2#3#4#5
1488 {
1489   \fp_compare:nNnTF { abs ( #2 ) } > { abs ( #3 ) }
1490   { #5 {#1} {#2} {#3} {#4} }
1491   { #5 {#1} {#3} {#2} {#4} }
1492 }
1493 </dvipdfmx | xetex>

```

(End definition for `\__draw_backend_cm_decompose:nnnnN` and others.)

`\__draw_backend_box_use:Nnnnn` Inserting a  $\text{\TeX}$  box transformed to the requested position and using the current matrix is done using a mixture of  $\text{\TeX}$  and low-level manipulation. The offset can be handled by  $\text{\TeX}$ , so only any rotation/skew/scaling component needs to be done using the matrix operation. As this operation can never be cached, the scope is set directly not using the `draw` version.



```

1494 \cs_new_protected:Npn \__draw_backend_box_use:Nnnnn #1#2#3#4#5
1495 {
1496   \__kernel_backend_scope_begin:
1497   <*luatex | pdftex>
1498   \__draw_backend_cm:nnnn {#2} {#3} {#4} {#5}
1499   </luatex | pdftex>
1500   <*dvipdfmx | xetex>
1501   \__kernel_backend_literal:n
1502   { pdf:btrans~matrix~ #2 ~ #3 ~ #4 ~ #5 ~ 0 ~ 0 }
1503   </dvipdfmx | xetex>
1504   \hbox_overlap_right:n { \box_use:N #1 }
1505   <*dvipdfmx | xetex>
1506   \__kernel_backend_literal:n { pdf:etrans }
1507   </dvipdfmx | xetex>
1508   \__kernel_backend_scope_end:
1509 }

```

(End definition for \\_\_draw\_backend\_box\_use:Nnnnn.)

```

1510 </dvipdfmx | luatex | pdftex | xetex>

```

### 4.3 dvisvgm backend

```

1511 <*dvisvgm>

```

\\_\_draw\_backend\_literal:n The same as the more general literal call.

```

\__draw_backend_literal:x
1512 \cs_new_eq:NN \__draw_backend_literal:n \__kernel_backend_literal_svg:n
1513 \cs_generate_variant:Nn \__draw_backend_literal:n { x }

```

(End definition for \\_\_draw\_backend\_literal:n.)

\\_\_draw\_backend\_scope\_begin: Use the backend-level scope mechanisms.

```

\__draw_backend_scope_end:
1514 \cs_new_eq:NN \__draw_backend_scope_begin: \__kernel_backend_scope_begin:
1515 \cs_new_eq:NN \__draw_backend_scope_end: \__kernel_backend_scope_end:

```

(End definition for \\_\_draw\_backend\_scope\_begin: and \\_\_draw\_backend\_scope\_end:.)

\\_\_draw\_backend\_begin: A drawing needs to be set up such that the co-ordinate system is translated. That is done inside a scope, which as described below

```

1516 \cs_new_protected:Npn \__draw_backend_begin:
1517 {
1518   \__kernel_backend_scope_begin:
1519   \__kernel_backend_scope:n { transform="translate({?x},{?y})~scale(1,-1)" }
1520 }
1521 \cs_new_eq:NN \__draw_backend_end: \__kernel_backend_scope_end:

```

(End definition for \\_\_draw\_backend\_begin: and \\_\_draw\_backend\_end:.)

\\_\_draw\_backend\_moveto:nn Once again, some work is needed to get path constructs correct. Rather than write the values as they are given, the entire path needs to be collected up before being output in one go. For that we use a dedicated storage routine, which adds spaces as required. Since paths should be fully expanded there is no need to worry about the internal x-type expansion.

```

\__draw_backend_lineto:nn
\__draw_backend_rectangle:nnnn
\__draw_backend_curveto:nnnnnn
\__draw_backend_add_to_path:n
\g__draw_backend_path_tl
1522 \cs_new_protected:Npn \__draw_backend_moveto:nn #1#2
1523 {

```

```

1524     \__draw_backend_add_to_path:n
1525     { M ~ \dim_to_decimal:n {#1} ~ \dim_to_decimal:n {#2} }
1526   }
1527 \cs_new_protected:Npn \__draw_backend_lineto:nn #1#2
1528 {
1529   \__draw_backend_add_to_path:n
1530   { L ~ \dim_to_decimal:n {#1} ~ \dim_to_decimal:n {#2} }
1531 }
1532 \cs_new_protected:Npn \__draw_backend_rectangle:nnnn #1#2#3#4
1533 {
1534   \__draw_backend_add_to_path:n
1535   {
1536     M ~ \dim_to_decimal:n {#1} ~ \dim_to_decimal:n {#2}
1537     h ~ \dim_to_decimal:n {#3} ~
1538     v ~ \dim_to_decimal:n {#4} ~
1539     h ~ \dim_to_decimal:n { -#3 } ~
1540     Z
1541   }
1542 }
1543 \cs_new_protected:Npn \__draw_backend_curveto:nnnnnn #1#2#3#4#5#6
1544 {
1545   \__draw_backend_add_to_path:n
1546   {
1547     C ~
1548     \dim_to_decimal:n {#1} ~ \dim_to_decimal:n {#2} ~
1549     \dim_to_decimal:n {#3} ~ \dim_to_decimal:n {#4} ~
1550     \dim_to_decimal:n {#5} ~ \dim_to_decimal:n {#6}
1551   }
1552 }
1553 \cs_new_protected:Npn \__draw_backend_add_to_path:n #1
1554 {
1555   \tl_gset:Nx \g__draw_backend_path_tl
1556   {
1557     \g__draw_backend_path_tl
1558     \tl_if_empty:NF \g__draw_backend_path_tl { \c_space_tl }
1559     #1
1560   }
1561 }
1562 \tl_new:N \g__draw_backend_path_tl

```

(End definition for \\_\_draw\_backend\_moveto:nn and others.)

```

\__draw_backend_evenodd_rule: The fill rules here have to be handled as scopes.
\__draw_backend_nonzero_rule: 1563 \cs_new_protected:Npn \__draw_backend_evenodd_rule:
1564   { \__kernel_backend_scope:n { fill-rule="evenodd" } }
1565 \cs_new_protected:Npn \__draw_backend_nonzero_rule:
1566   { \__kernel_backend_scope:n { fill-rule="nonzero" } }

```

(End definition for \\_\_draw\_backend\_evenodd\_rule: and \\_\_draw\_backend\_nonzero\_rule:.)

```

\__draw_backend_path:n
\__draw_backend_closepath:
\__draw_backend_stroke:
\__draw_backend_closestroke:
\__draw_backend_fill:
\__draw_backend_fillstroke:
\__draw_backend_clip:
\__draw_backend_discardpath:
\g__draw_draw_clip_bool
\g__draw_draw_path_int

```

Setting fill and stroke effects and doing clipping all has to be done using scopes. This means setting up the various requirements in a shared auxiliary which deals with the bits and pieces. Clipping paths are reused for path drawing: not essential but avoids constructing them twice. Discarding a path needs a separate function as it's not quite the same.

```

1567 \cs_new_protected:Npn \__draw_backend_closepath:
1568 { \__draw_backend_add_to_path:n { Z } }
1569 \cs_new_protected:Npn \__draw_backend_path:n #1
1570 {
1571   \bool_if:NTF \g__draw_draw_clip_bool
1572   {
1573     \int_gincr:N \g__kernel_clip_path_int
1574     \__draw_backend_literal:x
1575     {
1576       < clipPath~id = " l3cp \int_use:N \g__kernel_clip_path_int " >
1577       { ?nl }
1578       <path~d=" \g__draw_backend_path_tl "/> { ?nl }
1579       < /clipPath > { ? nl }
1580       <
1581       use~xlink:href =
1582       "\c_hash_str l3path \int_use:N \g__draw_backend_path_int " ~
1583       #1
1584       />
1585     }
1586     \__kernel_backend_scope:x
1587     {
1588       clip-path =
1589       "url( \c_hash_str l3cp \int_use:N \g__kernel_clip_path_int )"
1590     }
1591   }
1592   {
1593     \__draw_backend_literal:x
1594     { <path ~ d=" \g__draw_backend_path_tl " ~ #1 /> }
1595   }
1596   \tl_gclear:N \g__draw_backend_path_tl
1597   \bool_gset_false:N \g__draw_draw_clip_bool
1598 }
1599 \int_new:N \g__draw_backend_path_int
1600 \cs_new_protected:Npn \__draw_backend_stroke:
1601 { \__draw_backend_path:n { style="fill:none" } }
1602 \cs_new_protected:Npn \__draw_backend_closestroke:
1603 {
1604   \__draw_backend_closepath:
1605   \__draw_backend_stroke:
1606 }
1607 \cs_new_protected:Npn \__draw_backend_fill:
1608 { \__draw_backend_path:n { style="stroke:none" } }
1609 \cs_new_protected:Npn \__draw_backend_fillstroke:
1610 { \__draw_backend_path:n { } }
1611 \cs_new_protected:Npn \__draw_backend_clip:
1612 { \bool_gset_true:N \g__draw_draw_clip_bool }
1613 \bool_new:N \g__draw_draw_clip_bool
1614 \cs_new_protected:Npn \__draw_backend_discardpath:
1615 {
1616   \bool_if:NT \g__draw_draw_clip_bool
1617   {
1618     \int_gincr:N \g__kernel_clip_path_int
1619     \__draw_backend_literal:x
1620     {

```

```

1621         < clipPath~id = " l3cp \int_use:N \g__kernel_clip_path_int " >
1622             { ?nl }
1623         <path~d=" \g__draw_backend_path_tl "/> { ?nl }
1624         < /clipPath >
1625     }
1626     \__kernel_backend_scope:x
1627     {
1628         clip-path =
1629         "url( \c_hash_str l3cp \int_use:N \g__kernel_clip_path_int)"
1630     }
1631 }
1632 \tl_gclear:N \g__draw_path_tl
1633 \bool_gset_false:N \g__draw_draw_clip_bool
1634 }

```

(End definition for \\_\_draw\_backend\_path:n and others.)

All of these ideas are properties of scopes in SVG. The only slight complexity is converting the dash array properly (doing any required maths).

```

\__draw_backend_dash_pattern:nn
\__draw_backend_dash:n
\__draw_backend_dash_aux:nn
\__draw_backend_linewidth:n
\__draw_backend_miterlimit:n
\__draw_backend_cap_butt:
\__draw_backend_cap_round:
\__draw_backend_cap_rectangle:
\__draw_backend_join_miter:
\__draw_backend_join_round:
\__draw_backend_join_bevel:
1635 \cs_new_protected:Npn \__draw_backend_dash_pattern:nn #1#2
1636 {
1637     \use:x
1638     {
1639         \__draw_backend_dash_aux:nn
1640         { \clist_map_function:nn {#1} \__draw_backend_dash:n }
1641         { \dim_to_decimal:n {#2} }
1642     }
1643 }
1644 \cs_new:Npn \__draw_backend_dash:n #1
1645 { , \dim_to_decimal_in_bp:n {#1} }
1646 \cs_new_protected:Npn \__draw_backend_dash_aux:nn #1#2
1647 {
1648     \__kernel_backend_scope:x
1649     {
1650         stroke-dasharray =
1651         "
1652         \tl_if_empty:nTF {#1}
1653         { none }
1654         { \use_none:n #1 }
1655         " ~
1656         stroke-offset=" #2 "
1657     }
1658 }
1659 \cs_new_protected:Npn \__draw_backend_linewidth:n #1
1660 { \__kernel_backend_scope:x { stroke-width=" \dim_to_decimal:n {#1} " } }
1661 \cs_new_protected:Npn \__draw_backend_miterlimit:n #1
1662 { \__kernel_backend_scope:x { stroke-miterlimit=" #1 " } }
1663 \cs_new_protected:Npn \__draw_backend_cap_butt:
1664 { \__kernel_backend_scope:n { stroke-linecap="butt" } }
1665 \cs_new_protected:Npn \__draw_backend_cap_round:
1666 { \__kernel_backend_scope:n { stroke-linecap="round" } }
1667 \cs_new_protected:Npn \__draw_backend_cap_rectangle:
1668 { \__kernel_backend_scope:n { stroke-linecap="square" } }
1669 \cs_new_protected:Npn \__draw_backend_join_miter:

```

```

1670 { \__kernel_backend_scope:n { stroke-linejoin="miter" } }
1671 \cs_new_protected:Npn \__draw_backend_join_round:
1672 { \__kernel_backend_scope:n { stroke-linejoin="round" } }
1673 \cs_new_protected:Npn \__draw_backend_join_bevel:
1674 { \__kernel_backend_scope:n { stroke-linejoin="bevel" } }

```

(End definition for \\_\_draw\_backend\_dash\_pattern:nn and others.)

\\_\_draw\_backend\_cm:nnnn The four arguments here are floats (the affine matrix), the last two are a displacement vector.

```

1675 \cs_new_protected:Npn \__draw_backend_cm:nnnn #1#2#3#4
1676 {
1677   \__kernel_backend_scope:n
1678   {
1679     transform =
1680     " matrix ( #1 , #2 , #3 , #4 , Opt , Opt ) "
1681   }
1682 }

```

(End definition for \\_\_draw\_backend\_cm:nnnn.)

\\_\_draw\_backend\_box\_use:Nnnnn No special savings can be made here: simply displace the box inside a scope. As there is nothing to re-box, just make the box passed of zero size.

```

1683 \cs_new_protected:Npn \__draw_backend_box_use:Nnnnn #1#2#3#4#5
1684 {
1685   \__kernel_backend_scope_begin:
1686   \__draw_backend_cm:nnnn {#2} {#3} {#4} {#5}
1687   \__kernel_backend_literal_svg:n
1688   {
1689     < g~
1690     stroke="none"~
1691     transform="scale(-1,1)~translate({?x},{?y})~scale(-1,-1)"
1692     >
1693   }
1694   \box_set_wd:Nn #1 { Opt }
1695   \box_set_ht:Nn #1 { Opt }
1696   \box_set_dp:Nn #1 { Opt }
1697   \box_use:N #1
1698   \__kernel_backend_literal_svg:n { </g> }
1699   \__kernel_backend_scope_end:
1700 }

```

(End definition for \\_\_draw\_backend\_box\_use:Nnnnn.)

```

1701 </divsvgm>
1702 </package>

```

## 5 l3backend-graphics Implementation

```

1703 <*package>
1704 <@@=graphics>

```

\\_\_graphics\_backend\_loaded:n To deal with file load ordering. Plain users are on their own.

```

1705 \cs_new_protected:Npn \__graphics_backend_loaded:n #1

```

```

1706 {
1707   \cs_if_exist:NTF \hook_gput_code:nnn
1708   {
1709     \hook_gput_code:nnn
1710     { file / l3graphics.sty / after }
1711     { backend }
1712     {#1}
1713   }
1714   {#1}
1715 }

```

(End definition for `\__graphics_backend_loaded:n`.)

## 5.1 dvips backend

```

1716 <dvips>

```

`\l_graphics_search_ext_seq`

```

1717 \__graphics_backend_loaded:n
1718 { \seq_set_from_clist:Nn \l_graphics_search_ext_seq { .eps , .ps } }

```

(End definition for `\l_graphics_search_ext_seq`. This variable is documented on page ??.)

`\__graphics_backend_getbb_eps:n`  
`\__graphics_backend_getbb_ps:n`

Simply use the generic function.

```

1719 \__graphics_backend_loaded:n
1720 {
1721   \cs_new_eq:NN \__graphics_backend_getbb_eps:n \__graphics_read_bb:n
1722   \cs_new_eq:NN \__graphics_backend_getbb_ps:n \__graphics_read_bb:n
1723 }

```

(End definition for `\__graphics_backend_getbb_eps:n` and `\__graphics_backend_getbb_ps:n`.)

`\__graphics_backend_include_eps:n`  
`\__graphics_backend_include_ps:n`

The special syntax is relatively clear here: remember we need PostScript sizes here.

```

1724 \cs_new_protected:Npn \__graphics_backend_include_eps:n #1
1725 {
1726   \__kernel_backend_literal:x
1727   {
1728     PSfile = #1 \c_space_tl
1729     llx = \dim_to_decimal_in_bp:n \l__graphics_llx_dim \c_space_tl
1730     lly = \dim_to_decimal_in_bp:n \l__graphics_lly_dim \c_space_tl
1731     urx = \dim_to_decimal_in_bp:n \l__graphics_urx_dim \c_space_tl
1732     ury = \dim_to_decimal_in_bp:n \l__graphics_ury_dim
1733   }
1734 }
1735 \cs_new_eq:NN \__graphics_backend_include_ps:n \__graphics_backend_include_eps:n

```

(End definition for `\__graphics_backend_include_eps:n` and `\__graphics_backend_include_ps:n`.)

`\__graphics_backend_get_pagecount:n`

```

1736 \__graphics_backend_loaded:n
1737 { \cs_new_eq:NN \__graphics_backend_get_pagecount:n \__graphics_get_pagecount:n }

```

(End definition for `\__graphics_backend_get_pagecount:n`.)

```

1738 </dvips>

```

## 5.2 LuaTeX and pdfTeX backends

1739 `\*luatex | pdftex`

`\l_graphics_search_ext_seq`

```
1740 \__graphics_backend_loaded:n
1741 {
1742   \seq_set_from_clist:Nn
1743   \l_graphics_search_ext_seq
1744   { .pdf , .eps , .ps , .png , .jpg , .jpeg }
1745 }
```

(End definition for `\l_graphics_search_ext_seq`. This variable is documented on page ??.)

`\l_graphics_graphics_attr_tl`

In PDF mode, additional attributes of an graphic (such as page number) are needed both to obtain the bounding box and when inserting the graphic: this occurs as the graphic dictionary approach means they are read as part of the bounding box operation. As such, it is easier to track additional attributes using a dedicated `tl` rather than build up the same data twice.

```
1746 \tl_new:N \l__graphics_graphics_attr_tl
```

(End definition for `\l__graphics_graphics_attr_tl`.)

`\_graphics_backend_getbb_jpg:n`  
`\_graphics_backend_getbb_jpeg:n`  
`\_graphics_backend_getbb_pdf:n`  
`\_graphics_backend_getbb_png:n`  
`\_graphics_backend_getbb_auxi:n`  
`\_graphics_backend_getbb_auxii:n`  
`\_graphics_backend_getbb_auxiii:n`  
`\_graphics_backend_dequote:w`

Getting the bounding box here requires us to box up the graphic and measure it. To deal with the difference in feature support in bitmap and vector graphics but keeping the common parts, there is a little work to do in terms of auxiliaries. The key here is to notice that we need two forms of the attributes: a “short” set to allow us to track for caching, and the full form to pass to the primitive.

```
1747 \cs_new_protected:Npn \__graphics_backend_getbb_jpg:n #1
1748 {
1749   \int_zero:N \l__graphics_page_int
1750   \tl_clear:N \l__graphics_pagebox_tl
1751   \tl_set:Nx \l__graphics_graphics_attr_tl
1752   {
1753     \tl_if_empty:NF \l__graphics_decodearray_str
1754     { :D \l__graphics_decodearray_str }
1755     \bool_if:NT \l__graphics_interpolate_bool
1756     { :I }
1757   }
1758   \tl_clear:N \l__graphics_graphics_attr_tl
1759   \__graphics_backend_getbb_auxi:n {#1}
1760 }
1761 \cs_new_eq:NN \__graphics_backend_getbb_jpeg:n \__graphics_backend_getbb_jpg:n
1762 \cs_new_eq:NN \__graphics_backend_getbb_png:n \__graphics_backend_getbb_jpg:n
1763 \cs_new_protected:Npn \__graphics_backend_getbb_pdf:n #1
1764 {
1765   \tl_clear:N \l__graphics_decodearray_str
1766   \bool_set_false:N \l__graphics_interpolate_bool
1767   \tl_set:Nx \l__graphics_graphics_attr_tl
1768   {
1769     : \l__graphics_pagebox_tl
1770     \int_compare:nNnT \l__graphics_page_int > 1
1771     { :P \int_use:N \l__graphics_page_int }
1772   }
```

```

1773   \__graphics_backend_getbb_auxi:n {#1}
1774 }
1775 \cs_new_protected:Npn \__graphics_backend_getbb_auxi:n #1
1776 {
1777   \__graphics_bb_restore:xF { #1 \l__graphics_graphics_attr_tl }
1778   { \__graphics_backend_getbb_auxii:n {#1} }
1779 }

```

Measuring the graphic is done by boxing up: for PDF graphics we could use `\tex_pdfximagebbox:D`, but if doesn't work for other types. As the box always starts at (0,0) there is no need to worry about the lower-left position. Quotes need to be *removed* as LuaTeX does not like them here.

```

1780 \cs_new_protected:Npn \__graphics_backend_getbb_auxii:n #1
1781 {
1782   \exp_args:Ne \__graphics_backend_getbb_auxiii:n
1783   { \__graphics_backend_dequote:w #1 " #1 " \s__graphics_stop }
1784   \int_const:cn { c__graphics_ #1 \l__graphics_graphics_attr_tl _int }
1785   { \tex_the:D \tex_pdflastximage:D }
1786   \__graphics_bb_save:x { #1 \l__graphics_graphics_attr_tl }
1787 }
1788 \cs_new_protected:Npn \__graphics_backend_getbb_auxiii:n #1
1789 {
1790   \tex_immediate:D \tex_pdfximage:D
1791   \bool_lazy_or:nnT
1792   { \l__graphics_interpolate_bool }
1793   { ! \tl_if_empty_p:N \l__graphics_decodearray_str }
1794   {
1795     attr ~
1796     {
1797       \tl_if_empty:NF \l__graphics_decodearray_str
1798       { /Decode~[ \l__graphics_decodearray_str ] }
1799       \bool_if:NT \l__graphics_interpolate_bool
1800       { /Interpolate~true }
1801     }
1802   }
1803   \int_compare:nNnT \l__graphics_page_int > 0
1804   { page ~ \int_use:N \l__graphics_page_int }
1805   \tl_if_empty:NF \l__graphics_pagebox_tl
1806   { \l__graphics_pagebox_tl }
1807   {#1}
1808   \hbox_set:Nn \l__graphics_internal_box
1809   { \tex_pdfrefximage:D \tex_pdflastximage:D }
1810   \dim_set:Nn \l__graphics_urx_dim { \box_wd:N \l__graphics_internal_box }
1811   \dim_set:Nn \l__graphics_ury_dim { \box_ht:N \l__graphics_internal_box }
1812 }
1813 \cs_new:Npn \__graphics_backend_dequote:w #1 " #2 " #3 \s__graphics_stop {#2}

```

(End definition for `\__graphics_backend_getbb_jpg:n` and others.)

```

\__graphics_backend_include_jpg:n
\__graphics_backend_include_jpeg:n
\__graphics_backend_include_pdf:n
\__graphics_backend_include_png:n

```

Images are already loaded for the measurement part of the code, so inclusion is straightforward, with only any attributes to worry about. The latter carry through from determination of the bounding box.

```

1814 \cs_new_protected:Npn \__graphics_backend_include_jpg:n #1
1815 {

```



```

1816 \tex_pdfrefximage:D
1817 \int_use:c { c__graphics_ #1 \l__graphics_graphics_attr_tl _int }
1818 }
1819 \cs_new_eq:NN \__graphics_backend_include_jpeg:n \__graphics_backend_include_jpg:n
1820 \cs_new_eq:NN \__graphics_backend_include_pdf:n \__graphics_backend_include_jpg:n
1821 \cs_new_eq:NN \__graphics_backend_include_png:n \__graphics_backend_include_jpg:n

```

(End definition for \\_\_graphics\_backend\_include\_jpg:n and others.)

```

\__graphics_backend_getbb_eps:n EPS graphics may be included in LuaTeX/pdfTeX by conversion to PDF: this requires
\__graphics_backend_getbb_ps:n restricted shell escape. Modelled on the epstopdf LATEX 2ε package, but simplified, con-
\__graphics_backend_getbb_eps:n version takes place here if we have shell access.
\__graphics_backend_include_eps:n
\__graphics_backend_include_ps:n
\l__graphics_backend_dir_str
\l__graphics_backend_name_str
\l__graphics_backend_ext_str
1822 \sys_if_shell:T
1823 {
1824   \str_new:N \l__graphics_backend_dir_str
1825   \str_new:N \l__graphics_backend_name_str
1826   \str_new:N \l__graphics_backend_ext_str
1827   \cs_new_protected:Npn \__graphics_backend_getbb_eps:n #1
1828   {
1829     \file_parse_full_name:nNNN {#1}
1830     \l__graphics_backend_dir_str
1831     \l__graphics_backend_name_str
1832     \l__graphics_backend_ext_str
1833     \exp_args:Nx \__graphics_backend_getbb_eps:nn
1834     {
1835       \exp_args:Ne \__kernel_file_name_quote:n
1836       {
1837         \l__graphics_backend_name_str
1838         - \str_tail:N \l__graphics_backend_ext_str
1839         -converted-to.pdf
1840       }
1841     }
1842     {#1}
1843   }
1844   \cs_new_eq:NN \__graphics_backend_getbb_ps:n \__graphics_backend_getbb_eps:n
1845   \cs_new_protected:Npn \__graphics_backend_getbb_eps:nn #1#2
1846   {
1847     \file_compare_timestamp:nNnT {#2} > {#1}
1848     {
1849       \sys_shell_now:n
1850       { repstopdf ~ #2 ~ #1 }
1851     }
1852     \tl_set:Nn \l__graphics_final_name_str {#1}
1853     \__graphics_backend_getbb_pdf:n {#1}
1854   }
1855   \cs_new_protected:Npn \__graphics_backend_include_eps:n #1
1856   {
1857     \file_parse_full_name:nNNN {#1}
1858     \l__graphics_backend_dir_str \l__graphics_backend_name_str \l__graphics_backend_ext_str
1859     \exp_args:Nx \__graphics_backend_include_pdf:n
1860     {
1861       \exp_args:Ne \__kernel_file_name_quote:n
1862       {
1863         \l__graphics_backend_name_str

```

```

1864             - \str_tail:N \l__graphics_backend_ext_str
1865             -converted-to.pdf
1866         }
1867     }
1868 }
1869 \cs_new_eq:NN \__graphics_backend_include_ps:n \__graphics_backend_include_eps:n
1870 }

```

(End definition for \\_\_graphics\_backend\_getbb\_eps:n and others.)

\\_\_graphics\_backend\_get\_pagecount:n Simply load and store.

```

1871 \cs_new_protected:Npn \__graphics_backend_get_pagecount:n #1
1872 {
1873     \tex_pdfximage:D {#1}
1874     \int_const:cn { c__graphics_ #1 _pages_int }
1875     { \int_use:N \tex_pdflastximagepages:D }
1876 }

```

(End definition for \\_\_graphics\_backend\_get\_pagecount:n.)

```

1877 </luatex | pdftex>

```

### 5.3 dvipdfmx backend

```

1878 <*dvipdfmx | xetex>

```

\l\_graphics\_search\_ext\_seq

```

1879 \__graphics_backend_loaded:n
1880 {
1881     \seq_set_from_clist:Nn \l_graphics_search_ext_seq
1882     { .pdf , .eps , .ps , .png , .jpg ., jpeg , .bmp }
1883 }

```

(End definition for \l\_graphics\_search\_ext\_seq. This variable is documented on page ??.)

\\_\_graphics\_backend\_getbb\_eps:n Simply use the generic functions: only for dvipdfmx in the extraction cases.

```

\__graphics_backend_getbb_ps:n
\__graphics_backend_getbb_jpg:n
\__graphics_backend_getbb_jpeg:n
\__graphics_backend_getbb_pdf:n
\__graphics_backend_getbb_png:n
\__graphics_backend_getbb_bmp:n
1884 \__graphics_backend_loaded:n
1885 {
1886     \cs_new_eq:NN \__graphics_backend_getbb_eps:n \__graphics_read_bb:n
1887     \cs_new_eq:NN \__graphics_backend_getbb_ps:n \__graphics_read_bb:n
1888 }
1889 <*dvipdfmx>
1890 \cs_new_protected:Npn \__graphics_backend_getbb_jpg:n #1
1891 {
1892     \int_zero:N \l__graphics_page_int
1893     \tl_clear:N \l__graphics_pagebox_tl
1894     \__graphics_extract_bb:n {#1}
1895 }
1896 \cs_new_eq:NN \__graphics_backend_getbb_jpeg:n \__graphics_backend_getbb_jpg:n
1897 \cs_new_eq:NN \__graphics_backend_getbb_png:n \__graphics_backend_getbb_jpg:n
1898 \cs_new_eq:NN \__graphics_backend_getbb_bmp:n \__graphics_backend_getbb_jpg:n
1899 \cs_new_protected:Npn \__graphics_backend_getbb_pdf:n #1
1900 {
1901     \tl_clear:N \l__graphics_decodearray_str
1902     \bool_set_false:N \l__graphics_interpolate_bool

```

```

1903     \__graphics_extract_bb:n {#1}
1904   }
1905 </dvipdfmx>

```

(End definition for \\_\_graphics\_backend\_getbb\_eps:n and others.)

\g\_\_graphics\_track\_int Used to track the object number associated with each graphic.

```

1906 \int_new:N \g__graphics_track_int

```

(End definition for \g\_\_graphics\_track\_int.)

The special syntax depends on the file type. There is a difference in how PDF graphics are best handled between dvipdfmx and Xe<sub>La</sub>TeX: for the latter it is better to use the primitive route. The relevant code for that is included later in this file.

```

\__graphics_backend_include_eps:n
\__graphics_backend_include_ps:n
\__graphics_backend_include_jpg:n
\__graphics_backend_include_jpeg:n
\__graphics_backend_include_pdf:n
\__graphics_backend_include_png:n
\__graphics_backend_include_bmp:n
\__graphics_backend_include_auxi:nn
\__graphics_backend_include_auxii:nnn
\__graphics_backend_include_auxii:xnn
\__graphics_backend_include_auxiii:nnn
1907 \cs_new_protected:Npn \__graphics_backend_include_eps:n #1
1908 {
1909   \__kernel_backend_literal:x
1910   {
1911     PSfile = #1 \c_space_tl
1912     llx = \dim_to_decimal_in_bp:n \l__graphics_llx_dim \c_space_tl
1913     lly = \dim_to_decimal_in_bp:n \l__graphics_lly_dim \c_space_tl
1914     urx = \dim_to_decimal_in_bp:n \l__graphics_urx_dim \c_space_tl
1915     ury = \dim_to_decimal_in_bp:n \l__graphics_ury_dim
1916   }
1917 }
1918 \cs_new_eq:NN \__graphics_backend_include_ps:n \__graphics_backend_include_eps:n
1919 \cs_new_protected:Npn \__graphics_backend_include_jpg:n #1
1920 { \__graphics_backend_include_auxi:nn {#1} { image } }
1921 \cs_new_eq:NN \__graphics_backend_include_jpeg:n \__graphics_backend_include_jpg:n
1922 \cs_new_eq:NN \__graphics_backend_include_png:n \__graphics_backend_include_jpg:n
1923 \cs_new_eq:NN \__graphics_backend_include_bmp:n \__graphics_backend_include_jpg:n
1924 <*dvipdfmx>
1925 \cs_new_protected:Npn \__graphics_backend_include_pdf:n #1
1926 { \__graphics_backend_include_auxi:nn {#1} { epdf } }
1927 </dvipdfmx>

```

Graphic inclusion is set up to use the fact that each image is stored in the PDF as an XObject. This means that we can include repeated images only once and refer to them. To allow that, track the nature of each image: much the same as for the direct PDF mode case.

```

1928 \cs_new_protected:Npn \__graphics_backend_include_auxi:nn #1#2
1929 {
1930   \__graphics_backend_include_auxii:xnn
1931   {
1932     \tl_if_empty:NF \l__graphics_pagebox_tl
1933     { : \l__graphics_pagebox_tl }
1934     \int_compare:nNnT \l__graphics_page_int > 1
1935     { :P \int_use:N \l__graphics_page_int }
1936     \tl_if_empty:NF \l__graphics_decodearray_str
1937     { :D \l__graphics_decodearray_str }
1938     \bool_if:NT \l__graphics_interpolate_bool
1939     { :I }
1940   }
1941   {#1} {#2}

```

```

1942 }
1943 \cs_new_protected:Npn \__graphics_backend_include_auxii:nnn #1#2#3
1944 {
1945   \int_if_exist:cTF { c__graphics_ #2#1 _int }
1946   {
1947     \__kernel_backend_literal:x
1948     { pdf:useobj~@graphic \int_use:c { c__graphics_ #2#1 _int } }
1949   }
1950   { \__graphics_backend_include_auxiii:nnn {#2} {#1} {#3} }
1951 }
1952 \cs_generate_variant:Nn \__graphics_backend_include_auxii:nnn { x }

```

Inclusion using the specials is relatively straight-forward, but there is one wrinkle. To get the `pagebox` correct for PDF graphics in all cases, it is necessary to provide both that information and the `bbox` argument: odd things happen otherwise!

```

1953 \cs_new_protected:Npn \__graphics_backend_include_auxiii:nnn #1#2#3
1954 {
1955   \int_gincr:N \g__graphics_track_int
1956   \int_const:cn { c__graphics_ #1#2 _int } { \g__graphics_track_int }
1957   \__kernel_backend_literal:x
1958   {
1959     pdf:#3~
1960     @graphic \int_use:c { c__graphics_ #1#2 _int } ~
1961     \int_compare:nNnT \l__graphics_page_int > 1
1962     { page ~ \int_use:N \l__graphics_page_int \c_space_tl }
1963     \tl_if_empty:NF \l__graphics_pagebox_tl
1964     {
1965       pagebox ~ \l__graphics_pagebox_tl \c_space_tl
1966       bbox ~
1967         \dim_to_decimal_in_bp:n \l__graphics_llx_dim \c_space_tl
1968         \dim_to_decimal_in_bp:n \l__graphics_lly_dim \c_space_tl
1969         \dim_to_decimal_in_bp:n \l__graphics_urx_dim \c_space_tl
1970         \dim_to_decimal_in_bp:n \l__graphics_ury_dim \c_space_tl
1971     }
1972     (#1)
1973     \bool_lazy_or:nnT
1974     { \l__graphics_interpolate_bool }
1975     { ! \tl_if_empty_p:N \l__graphics_decodearray_str }
1976     {
1977       <<
1978       \tl_if_empty:NF \l__graphics_decodearray_str
1979       { /Decode~[ \l__graphics_decodearray_str ] }
1980       \bool_if:NT \l__graphics_interpolate_bool
1981       { /Interpolate~true> }
1982       >>
1983     }
1984   }
1985 }

```

(End definition for `\__graphics_backend_include_eps:n` and others.)

`\__graphics_backend_get_pagecount:n`

```

1986 {*dvipdfmx}
1987 \__graphics_backend_loaded:n
1988 { \cs_new_eq:NN \__graphics_backend_get_pagecount:n \__graphics_get_pagecount:n }

```

1989  $\langle$ /dvi<sub>pdf</sub>mx $\rangle$

(End definition for `\_graphics_backend_get_pagecount:n`.)

1990  $\langle$ /dvi<sub>pdf</sub>mx | xetex $\rangle$

## 5.4 X<sub>Y</sub>T<sub>E</sub>X backend

1991  $\langle$ \*xetex $\rangle$

For X<sub>Y</sub>T<sub>E</sub>X, there are two primitives that allow us to obtain the bounding box without needing `extractbb`. The only complexity is passing the various minor variations to a common core process. The X<sub>Y</sub>T<sub>E</sub>X primitive omits the text box from the page box specification, so there is also some “trimming” to do here.

```

1992 \cs_new_protected:Npn \_graphics_backend_getbb_jpg:n #1
1993 {
1994   \int_zero:N \l__graphics_page_int
1995   \tl_clear:N \l__graphics_pagebox_tl
1996   \_graphics_backend_getbb_auxi:nN {#1} \tex_XeTeXpicfile:D
1997 }
1998 \cs_new_eq:NN \_graphics_backend_getbb_jpeg:n \_graphics_backend_getbb_jpg:n
1999 \cs_new_eq:NN \_graphics_backend_getbb_png:n \_graphics_backend_getbb_jpg:n
2000 \cs_new_eq:NN \_graphics_backend_getbb_bmp:n \_graphics_backend_getbb_jpg:n
2001 \cs_new_protected:Npn \_graphics_backend_getbb_pdf:n #1
2002 {
2003   \tl_clear:N \l__graphics_decodearray_str
2004   \bool_set_false:N \l__graphics_interpolate_bool
2005   \_graphics_backend_getbb_auxi:nN {#1} \tex_XeTeXpdffile:D
2006 }
2007 \cs_new_protected:Npn \_graphics_backend_getbb_auxi:nN #1#2
2008 {
2009   \int_compare:nNnTF \l__graphics_page_int > 1
2010     { \_graphics_backend_getbb_auxii:VnN \l__graphics_page_int {#1} #2 }
2011     { \_graphics_backend_getbb_auxiii:nNnn {#1} #2 { :P 1 } { page 1 } }
2012 }
2013 \cs_new_protected:Npn \_graphics_backend_getbb_auxii:nnN #1#2#3
2014 { \_graphics_backend_getbb_auxiii:nNnn {#2} #3 { :P #1 } { page #1 } }
2015 \cs_generate_variant:Nn \_graphics_backend_getbb_auxii:nnN { V }
2016 \cs_new_protected:Npn \_graphics_backend_getbb_auxiii:nNnn #1#2#3#4
2017 {
2018   \tl_if_empty:NTF \l__graphics_pagebox_tl
2019     { \_graphics_backend_getbb_auxiv:VnNnn \l__graphics_pagebox_tl }
2020     { \_graphics_backend_getbb_auxv:nNnn }
2021     {#1} #2 {#3} {#4}
2022 }
2023 \cs_new_protected:Npn \_graphics_backend_getbb_auxiv:nnNnn #1#2#3#4#5
2024 {
2025   \use:x
2026   {
2027     \_graphics_backend_getbb_auxv:nNnn {#2} #3 { : #1 #4 }
2028     {
2029       #5
2030       \tl_if_blank:nF {#1}
2031         { \c_space_tl \_graphics_backend_getbb_pagebox:w #1 }
2032     }
2033   }

```

```

2033     }
2034   }
2035   \cs_generate_variant:Nn \__graphics_backend_getbb_auxiv:nnNnn { V }
2036   \cs_new_protected:Npn \__graphics_backend_getbb_auxv:nNnn #1#2#3#4
2037   {
2038     \__graphics_bb_restore:nF {#1#3}
2039     { \__graphics_backend_getbb_auxvi:nNnn {#1} #2 {#3} {#4} }
2040   }
2041   \cs_new_protected:Npn \__graphics_backend_getbb_auxvi:nNnn #1#2#3#4
2042   {
2043     \hbox_set:Nn \l__graphics_internal_box { #2 #1 ~ #4 }
2044     \dim_set:Nn \l__graphics_urx_dim { \box_wd:N \l__graphics_internal_box }
2045     \dim_set:Nn \l__graphics_ury_dim { \box_ht:N \l__graphics_internal_box }
2046     \__graphics_bb_save:n {#1#3}
2047   }
2048   \cs_new:Npn \__graphics_backend_getbb_pagebox:w #1 box {#1}

```

(End definition for \\_\_graphics\_backend\_getbb\_jpg:n and others.)

\\_\_graphics\_backend\_include\_pdf:n

For PDF graphics, properly supporting the **pagebox** concept in X<sub>Y</sub>TeX is best done using the `\tex_XeTeXpdffile:D` primitive. The syntax here is the same as for the graphic measurement part, although we know at this stage that there must be some valid setting for `\l__graphics_pagebox_tl`.

```

2049   \cs_new_protected:Npn \__graphics_backend_include_pdf:n #1
2050   {
2051     \tex_XeTeXpdffile:D #1 ~
2052     \int_compare:nNnT \l__graphics_page_int > 0
2053     { page ~ \int_use:N \l__graphics_page_int \c_space_tl }
2054     \exp_after:wN \__graphics_backend_getbb_pagebox:w \l__graphics_pagebox_tl
2055   }

```

(End definition for \\_\_graphics\_backend\_include\_pdf:n.)

\\_\_graphics\_backend\_get\_pagecount:n

Very little to do here other than cover the case of a non-PDF file.

```

2056   \cs_new_protected:Npn \__graphics_backend_get_pagecount:n #1
2057   {
2058     \int_const:cn { c__graphics_#1 _pages_int }
2059     {
2060       \int_max:nn
2061       { \int_use:N \tex_XeTeXpdfpagecount:D #1 ~ }
2062       { 1 }
2063     }
2064   }

```

(End definition for \\_\_graphics\_backend\_get\_pagecount:n.)

2065 </xetex>

## 5.5 dvisvgm backend

2066 <\*dvisvgm>

\l\_graphics\_search\_ext\_seq

```

2067   \__graphics_backend_loaded:n
2068   {

```

```

2069 \seq_set_from_clist:Nn
2070 \l_graphics_search_ext_seq
2071 { .svg , .pdf , .eps , .ps , .png , .jpg , .jpeg }
2072 }

```

(End definition for `\l_graphics_search_ext_seq`. This variable is documented on page ??.)

```

\__graphics_backend_getbb_svg:n
\__graphics_backend_getbb_svg_auxi:nNn
\__graphics_backend_getbb_svg_auxii:w
\__graphics_backend_getbb_svg_auxiii:Nw
\__graphics_backend_getbb_svg_auxiv:Nw
\__graphics_backend_getbb_svg_auxv:Nw
\__graphics_backend_getbb_svg_auxvi:Nn
\__graphics_backend_getbb_svg_auxvii:w

```

This is relatively similar to reading bounding boxes for `.eps` files. Life is though made more tricky as we cannot pick a single line for the data. So we have to loop until we collect up both height and width. To do that, we can use a marker value. We also have to allow for the default units of the lengths: they are big points and may be omitted.

```

2073 \cs_new_protected:Npn \__graphics_backend_getbb_svg:n #1
2074 {
2075   \__graphics_bb_restore:nF {#1}
2076   {
2077     \ior_open:Nn \l__graphics_internal_ior {#1}
2078     \ior_if_eof:NTF \l__graphics_internal_ior
2079       { \msg_error:nnn { graphics } { graphic-not-found } {#1} }
2080       {
2081         \dim_zero:N \l__graphics_llx_dim
2082         \dim_zero:N \l__graphics_lly_dim
2083         \dim_set:Nn \l__graphics_urx_dim { -\c_max_dim }
2084         \dim_set:Nn \l__graphics_ury_dim { -\c_max_dim }
2085         \ior_str_map_inline:Nn \l__graphics_internal_ior
2086         {
2087           \dim_compare:nNnT \l__graphics_urx_dim = { -\c_max_dim }
2088           {
2089             \__graphics_backend_getbb_svg_auxi:nNn
2090             { width } \l__graphics_urx_dim {##1}
2091           }
2092           \dim_compare:nNnT \l__graphics_ury_dim = { -\c_max_dim }
2093           {
2094             \__graphics_backend_getbb_svg_auxi:nNn
2095             { height } \l__graphics_ury_dim {##1}
2096           }
2097           \bool_lazy_and:nnF
2098             { \dim_compare_p:nNn \l__graphics_urx_dim = { -\c_max_dim } }
2099             { \dim_compare_p:nNn \l__graphics_ury_dim = { -\c_max_dim } }
2100             { \ior_map_break: }
2101         }
2102         \__graphics_bb_save:n {#1}
2103       }
2104     \ior_close:N \l__graphics_internal_ior
2105   }
2106 }
2107 \cs_new_protected:Npn \__graphics_backend_getbb_svg_auxi:nNn #1#2#3
2108 {
2109   \use:x
2110   {
2111     \cs_set_protected:Npn \__graphics_backend_getbb_svg_auxii:w
2112       ###1 \tl_to_str:n {#1} = ####2 \tl_to_str:n {#1} = ####3
2113     \s__graphics_stop
2114   }
2115 }

```

```

2116         \tl_if_blank:nF {##2}
2117         {
2118             \peek_remove_spaces:n
2119             {
2120                 \peek_meaning:NTF ' % '
2121                 { \__graphics_backend_getbb_svg_auxiii:Nw #2 }
2122                 {
2123                     \peek_meaning:NTF " % "
2124                     { \__graphics_backend_getbb_svg_auxiv:Nw #2 }
2125                     { \__graphics_backend_getbb_svg_auxv:Nw #2 }
2126                 }
2127             }
2128             ##2 \s__graphics_stop
2129         }
2130     }
2131     \use:x
2132     {
2133         \__graphics_backend_getbb_svg_auxii:w #3
2134         \tl_to_str:n {#1} = \tl_to_str:n {#1} =
2135         \s__graphics_stop
2136     }
2137 }
2138 \cs_new_protected:Npn \__graphics_backend_getbb_svg_auxii:w { }
2139 \cs_new_protected:Npn \__graphics_backend_getbb_svg_auxiii:Nw #1 ' #2 ' #3 \s__graphics_stop
2140 { \__graphics_backend_getbb_svg_auxvi:Nn #1 {#2} }
2141 \cs_new_protected:Npn \__graphics_backend_getbb_svg_auxiv:Nw #1 " #2 " #3 \s__graphics_stop
2142 { \__graphics_backend_getbb_svg_auxvi:Nn #1 {#2} }
2143 \cs_new_protected:Npn \__graphics_backend_getbb_svg_auxv:Nw #1 #2 ~ #3 \s__graphics_stop
2144 { \__graphics_backend_getbb_svg_auxvi:Nn #1 {#2} }
2145 \cs_new_protected:Npn \__graphics_backend_getbb_svg_auxvi:Nn #1#2
2146 {
2147     \tex_afterassignment:D \__graphics_backend_getbb_svg_auxvii:w
2148     \l__graphics_internal_dim #2 bp \scan_stop:
2149     \dim_set_eq:NN #1 \l__graphics_internal_dim
2150 }
2151 \cs_new_protected:Npn \__graphics_backend_getbb_svg_auxvii:w #1 \scan_stop: { }

```

(End definition for \\_\_graphics\_backend\_getbb\_svg:n and others.)

\\_\_graphics\_backend\_getbb\_eps:n

Simply use the generic function.

\\_\_graphics\_backend\_getbb\_ps:n

```

2152 \__graphics_backend_loaded:n
2153 {
2154     \cs_new_eq:NN \__graphics_backend_getbb_eps:n \__graphics_read_bb:n
2155     \cs_new_eq:NN \__graphics_backend_getbb_ps:n \__graphics_read_bb:n
2156 }

```

(End definition for \\_\_graphics\_backend\_getbb\_eps:n and \\_\_graphics\_backend\_getbb\_ps:n.)

\\_\_graphics\_backend\_getbb\_png:n

These can be included by extracting the bounding box data.

\\_\_graphics\_backend\_getbb\_jpg:n

```

2157 \cs_new_protected:Npn \__graphics_backend_getbb_jpg:n #1
2158 {
2159     \int_zero:N \l__graphics_page_int
2160     \tl_clear:N \l__graphics_pagebox_tl
2161     \__graphics_extract_bb:n {#1}
2162 }

```

\\_\_graphics\_backend\_getbb\_jpeg:n



```

2163 \cs_new_eq:NN \__graphics_backend_getbb_jpeg:n \__graphics_backend_getbb_jpg:n
2164 \cs_new_eq:NN \__graphics_backend_getbb_png:n \__graphics_backend_getbb_jpg:n

```

(End definition for \\_\_graphics\_backend\_getbb\_png:n, \\_\_graphics\_backend\_getbb\_jpg:n, and \\_\_graphics\_backend\_getbb\_jpeg:n.)

\\_\_graphics\_backend\_getbb\_pdf:n Same as for dvipdfmx: use the generic function

```

2165 \cs_new_protected:Npn \__graphics_backend_getbb_pdf:n #1
2166 {
2167   \tl_clear:N \l__graphics_decodearray_str
2168   \bool_set_false:N \l__graphics_interpolate_bool
2169   \__graphics_extract_bb:n {#1}
2170 }

```

(End definition for \\_\_graphics\_backend\_getbb\_pdf:n.)

\\_\_graphics\_backend\_include\_eps:n \\_\_graphics\_backend\_include\_ps:n \\_\_graphics\_backend\_include\_pdf:n \\_\_graphics\_backend\_include\_nn The special syntax is relatively clear here: remember we need PostScript sizes here. (This is the same as the dvips code.)

```

2171 \cs_new_protected:Npn \__graphics_backend_include_eps:n #1
2172 { \__graphics_backend_include:nn { PSfile } {#1} }
2173 \cs_new_eq:NN \__graphics_backend_include_ps:n \__graphics_backend_include_eps:n
2174 \cs_new_protected:Npn \__graphics_backend_include_pdf:n #1
2175 { \__graphics_backend_include:nn { pdffile } {#1} }
2176 \cs_new_protected:Npn \__graphics_backend_include:nn #1#2
2177 {
2178   \__kernel_backend_literal:x
2179   {
2180     #1 = #2 \c_space_tl
2181     llx = \dim_to_decimal_in_bp:n \l__graphics_llx_dim \c_space_tl
2182     lly = \dim_to_decimal_in_bp:n \l__graphics_lly_dim \c_space_tl
2183     urx = \dim_to_decimal_in_bp:n \l__graphics_urx_dim \c_space_tl
2184     ury = \dim_to_decimal_in_bp:n \l__graphics_ury_dim
2185   }
2186 }

```

(End definition for \\_\_graphics\_backend\_include\_eps:n and others.)

\\_\_graphics\_backend\_include\_svg:n \\_\_graphics\_backend\_include\_png:n \\_\_graphics\_backend\_include\_jpg:n \\_\_graphics\_backend\_include\_jpeg:n \\_\_graphics\_backend\_include\_dequote:w The backend here has built-in support for basic graphic inclusion (see dvisvgm.def for a more complex approach, needed if clipping, etc., is covered at the graphic backend level). We have to deal with the fact that the image reference point is at the top, so there is a need for a veritcal shift to put it in the right place. The other issue is that #1 must be quote-corrected. The dvisvgm:img operation quotes the file name, but if it is already quoted (contains spaces) then we have an issue: we simply strip off any quotes as a result.

```

2187 \cs_new_protected:Npn \__graphics_backend_include_svg:n #1
2188 {
2189   \box_move_up:nn { \l__graphics_ury_dim }
2190   {
2191     \hbox:n
2192     {
2193       \__kernel_backend_literal:x
2194       {
2195         dvisvgm:img~
2196         \dim_to_decimal:n { \l__graphics_urx_dim } ~
2197         \dim_to_decimal:n { \l__graphics_ury_dim } ~

```

```

2198         \_graphics_backend_include_dequote:w #1 " #1 " \s__graphics_stop
2199     }
2200 }
2201 }
2202 }
2203 \cs_new_eq:NN \_graphics_backend_include_png:n \_graphics_backend_include_svg:n
2204 \cs_new_eq:NN \_graphics_backend_include_jpeg:n \_graphics_backend_include_svg:n
2205 \cs_new_eq:NN \_graphics_backend_include_jpg:n \_graphics_backend_include_svg:n
2206 \cs_new:Npn \_graphics_backend_include_dequote:w #1 " #2 " #3 \s__graphics_stop
2207     {#2}

```

(End definition for \\_graphics\_backend\_include\_svg:n and others.)

\\_graphics\_backend\_get\_pagecount:n

```

2208 \_graphics_backend_loaded:n
2209 { \cs_new_eq:NN \_graphics_backend_get_pagecount:n \_graphics_get_pagecount:n }

```

(End definition for \\_graphics\_backend\_get\_pagecount:n.)

```

2210 </dvisvgm>
2211 </package>

```

## 6 l3backend-pdf Implementation

```

2212 <*package>
2213 <@@=pdf>

```

Setting up PDF resources is a complex area with only limited documentation in the engine manuals. The following code builds heavily on existing ideas from `hyperref` work by Sebastian Rahtz and Heiko Oberdiek, and significant contributions by Alexander Grahn, in addition to the specific code referenced a various points.

### 6.1 Shared code

A very small number of items that belong at the backend level but which are common to all backends.

\l\_\_pdf\_internal\_box

```

2214 \box_new:N \l__pdf_internal_box

```

(End definition for \l\_\_pdf\_internal\_box.)

### 6.2 dvips backend

```

2215 <*dvips>

```

\\_pdf\_backend\_pdfmark:n Used often enough it should be a separate function.

```

\_pdf_backend_pdfmark:x
2216 \cs_new_protected:Npn \_pdf_backend_pdfmark:n #1
2217 { \_kernel_backend_postscript:n { mark #1 ~ pdfmark } }
2218 \cs_generate_variant:Nn \_pdf_backend_pdfmark:n { x }

```

(End definition for \\_pdf\_backend\_pdfmark:n.)

## 6.2.1 Catalogue entries

```

\__pdf_backend_catalog_gput:nn
\__pdf_backend_info_gput:nn
2219 \cs_new_protected:Npn \__pdf_backend_catalog_gput:nn #1#2
2220 { \__pdf_backend_pdfmark:n { { Catalog } << /#1 ~ #2 >> /PUT } }
2221 \cs_new_protected:Npn \__pdf_backend_info_gput:nn #1#2
2222 { \__pdf_backend_pdfmark:n { /#1 ~ #2 /DOCINFO } }

(End definition for \__pdf_backend_catalog_gput:nn and \__pdf_backend_info_gput:nn.)

```

## 6.2.2 Objects

For tracking objects to allow finalisation.

```

\g__pdf_backend_object_int
\g__pdf_backend_object_prop
2223 \int_new:N \g__pdf_backend_object_int
2224 \prop_new:N \g__pdf_backend_object_prop

(End definition for \g__pdf_backend_object_int and \g__pdf_backend_object_prop.)

```

Tracking objects is similar to dvipdfmx.

```

\__pdf_backend_object_new:nn
\__pdf_backend_object_ref:n
2225 \cs_new_protected:Npn \__pdf_backend_object_new:nn #1#2
2226 {
2227   \int_gincr:N \g__pdf_backend_object_int
2228   \int_const:cn
2229   { c__pdf_backend_object_ \tl_to_str:n {#1} _int }
2230   { \g__pdf_backend_object_int }
2231   \prop_gput:Nnn \g__pdf_backend_object_prop {#1} {#2}
2232 }
2233 \cs_new:Npn \__pdf_backend_object_ref:n #1
2234 { { pdf.obj \int_use:c { c__pdf_backend_object_ \tl_to_str:n {#1} _int } } }

(End definition for \__pdf_backend_object_new:nn and \__pdf_backend_object_ref:n.)

```

This is where we choose the actual type: some work to get things right.

```

\__pdf_backend_object_write:nn
\__pdf_backend_object_write:nx
\__pdf_backend_object_write_array:nn
\__pdf_backend_object_write_dict:nn
\__pdf_backend_object_write_fstream:nn
\__pdf_backend_object_write_stream:nn
\__pdf_backend_object_write_stream:nnn
2235 \cs_new_protected:Npn \__pdf_backend_object_write:nn #1#2
2236 {
2237   \__pdf_backend_pdfmark:x
2238   {
2239     /objdef ~ \__pdf_backend_object_ref:n {#1}
2240     /type
2241     \str_case_e:nn
2242     { \prop_item:Nn \g__pdf_backend_object_prop {#1} }
2243     {
2244       { array } { /array }
2245       { dict } { /dict }
2246       { fstream } { /stream }
2247       { stream } { /stream }
2248     }
2249     /OBJ
2250   }
2251   \use:c
2252   { \__pdf_backend_object_write_ \prop_item:Nn \g__pdf_backend_object_prop {#1} :nn }
2253   { \__pdf_backend_object_ref:n {#1} } {#2}
2254 }
2255 \cs_generate_variant:Nn \__pdf_backend_object_write:nn { nx }
2256 \cs_new_protected:Npn \__pdf_backend_object_write_array:nn #1#2

```

```

2257 {
2258   \__pdf_backend_pdfmark:x
2259   { #1 ~0~ [ ~ \exp_not:n {#2} ~ ] ~ /PUTINTERVAL }
2260 }
2261 \cs_new_protected:Npn \__pdf_backend_object_write_dict:nn #1#2
2262 {
2263   \__pdf_backend_pdfmark:x
2264   { #1 << \exp_not:n {#2} >> /PUT }
2265 }
2266 \cs_new_protected:Npn \__pdf_backend_object_write_fstream:nn #1#2
2267 {
2268   \exp_args:Nx
2269   \__pdf_backend_object_write_fstream:nnn {#1} #2
2270 }
2271 \cs_new_protected:Npn \__pdf_backend_object_write_fstream:nnn #1#2#3
2272 {
2273   \__kernel_backend_postscript:n
2274   {
2275     SDict ~ begin ~
2276     mark ~ #1 ~ << #2 >> /PUT ~ pdfmark ~
2277     mark ~ #1 ~ ( #3 )~ ( r )~ file ~ /PUT ~ pdfmark ~
2278     end
2279   }
2280 }
2281 \cs_new_protected:Npn \__pdf_backend_object_write_stream:nn #1#2
2282 {
2283   \exp_args:Nx
2284   \__pdf_backend_object_write_stream:nnn {#1} #2
2285 }
2286 \cs_new_protected:Npn \__pdf_backend_object_write_stream:nnn #1#2#3
2287 {
2288   \__kernel_backend_postscript:n
2289   {
2290     mark ~ #1 ~ ( #3 ) /PUT ~ pdfmark ~
2291     mark ~ #1 ~ << #2 >> /PUT ~ pdfmark
2292   }
2293 }

```

(End definition for \\_\_pdf\_backend\_object\_write:nn and others.)

\\_\_pdf\_backend\_object\_now:nn No anonymous objects, so things are done manually.

```

\__pdf_backend_object_now:nx
2294 \cs_new_protected:Npn \__pdf_backend_object_now:nn #1#2
2295 {
2296   \int_gincr:N \g__pdf_backend_object_int
2297   \__pdf_backend_pdfmark:x
2298   {
2299     /_objdef ~ { pdf.obj \int_use:N \g__pdf_backend_object_int }
2300     /type
2301     \str_case:nn
2302     {#1}
2303     {
2304       { array } { /array }
2305       { dict } { /dict }
2306       { fstream } { /stream }

```

```

2307         { stream } { /stream }
2308     }
2309     /OBJ
2310 }
2311 \exp_args:Nnx \use:c { __pdf_backend_object_write_ #1 :nn }
2312 { { pdf.obj \int_use:N \g__pdf_backend_object_int } } {#2}
2313 }
2314 \cs_generate_variant:Nn \__pdf_backend_object_now:nn { nx }

```

(End definition for \\_\_pdf\_backend\_object\_now:nn.)

\\_\_pdf\_backend\_object\_last: Much like the annotation version.

```

2315 \cs_new:Npn \__pdf_backend_object_last:
2316 { { pdf.obj \int_use:N \g__pdf_backend_object_int } }

```

(End definition for \\_\_pdf\_backend\_object\_last:.)

\\_pdf\_backend\_pageobject\_ref:n Page references are easy in dvips.

```

2317 \cs_new:Npn \_pdf_backend_pageobject_ref:n #1
2318 { { Page #1 } }

```

(End definition for \\_pdf\_backend\_pageobject\_ref:n.)

### 6.2.3 Annotations

In dvips, annotations have to be constructed manually. As such, we need the object code above for some definitions.

\l\_\_pdf\_backend\_content\_box The content of an annotation.

```

2319 \box_new:N \l__pdf_backend_content_box

```

(End definition for \l\_\_pdf\_backend\_content\_box.)

\l\_\_pdf\_backend\_model\_box For creating model sizing for links.

```

2320 \box_new:N \l__pdf_backend_model_box

```

(End definition for \l\_\_pdf\_backend\_model\_box.)

\g\_pdf\_backend\_annotation\_int Needed as objects which are not annotations could be created.

```

2321 \int_new:N \g__pdf_backend_annotation_int

```

(End definition for \g\_\_pdf\_backend\_annotation\_int.)

\\_pdf\_backend\_annotation:nnnn Annotations are objects, but we track them separately. Notably, they are not in the object data lists. Here, to get the co-ordinates of the annotation, we need to have the data collected at the PostScript level. That requires a bit of box trickery (effectively a L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> picture of zero size). Once the data is collected, use it to set up the annotation border.

```

2322 \cs_new_protected:Npn \_pdf_backend_annotation:nnnn #1#2#3#4
2323 {
2324     \exp_args:Nf \_pdf_backend_annotation_aux:nnnn
2325     { \dim_eval:n {#1} } {#2} {#3} {#4}
2326 }
2327 \cs_new_protected:Npn \_pdf_backend_annotation_aux:nnnn #1#2#3#4
2328 {

```

```

2329 \box_move_down:nn {#3}
2330 { \hbox:n { \__kernel_backend_postscript:n { pdf.save.ll } } }
2331 \box_move_up:nn {#2}
2332 {
2333   \hbox:n
2334   {
2335     \__kernel_kern:n {#1}
2336     \__kernel_backend_postscript:n { pdf.save.ur }
2337     \__kernel_kern:n { -#1 }
2338   }
2339 }
2340 \int_gincr:N \g__pdf_backend_object_int
2341 \int_gset_eq:NN \g__pdf_backend_annotation_int \g__pdf_backend_object_int
2342 \__pdf_backend_pdfmark:x
2343 {
2344   /objdef { pdf.obj \int_use:N \g__pdf_backend_object_int }
2345   pdf.rect
2346   #4 ~
2347   /ANN
2348 }
2349 }

```

(End definition for \\_\_pdf\_backend\_annotation:nnnn.)

\\_\_pdf\_backend\_annotation\_last: Provide the last annotation we created: could get tricky of course if other packages are loaded.

```

2350 \cs_new:Npn \__pdf_backend_annotation_last:
2351 { { pdf.obj \int_use:N \g__pdf_backend_annotation_int } }

```

(End definition for \\_\_pdf\_backend\_annotation\_last:.)

\g\_\_pdf\_backend\_link\_int To track annotations which are links.

```

2352 \int_new:N \g__pdf_backend_link_int

```

(End definition for \g\_\_pdf\_backend\_link\_int.)

\g\_\_pdf\_backend\_link\_dict\_tl To pass information to the end-of-link function.

```

2353 \tl_new:N \g__pdf_backend_link_dict_tl

```

(End definition for \g\_\_pdf\_backend\_link\_dict\_tl.)

\g\_\_pdf\_backend\_link\_sf\_int Needed to save/restore space factor, which is needed to deal with the face we need a box.

```

2354 \int_new:N \g__pdf_backend_link_sf_int

```

(End definition for \g\_\_pdf\_backend\_link\_sf\_int.)

\g\_\_pdf\_backend\_link\_math\_bool Needed to save/restore math mode.

```

2355 \bool_new:N \g__pdf_backend_link_math_bool

```

(End definition for \g\_\_pdf\_backend\_link\_math\_bool.)

\g\_\_pdf\_backend\_link\_bool Track link formation: we cannot nest at all.

```

2356 \bool_new:N \g__pdf_backend_link_bool

```

(End definition for \g\_\_pdf\_backend\_link\_bool.)

`\l__pdf_breaklink_pdfmark_tl` Swappable content for link breaking.

```

2357 \tl_new:N \l__pdf_breaklink_pdfmark_tl
2358 \tl_set:Nn \l__pdf_breaklink_pdfmark_tl { pdfmark }

(End definition for \l__pdf_breaklink_pdfmark_tl.)

```

`\__pdf_breaklink_postscript:n` To allow dropping material unless link breaking is active.

```

2359 \cs_new_protected:Npn \__pdf_breaklink_postscript:n #1 { }

(End definition for \__pdf_breaklink_postscript:n.)

```

`\__pdf_breaklink_usebox:N` Swappable box unpacking or use.

```

2360 \cs_new_eq:NN \__pdf_breaklink_usebox:N \box_use:N

(End definition for \__pdf_breaklink_usebox:N.)

```

`\__pdf_backend_link_begin_goto:nnw` Links are crated like annotations but with dedicated code to allow for adjusting the size of the rectangle. In contrast to `hyperref`, we grab the link content as a box which can then unbox: this allows the same interface as for pdfTeX.

`\__pdf_backend_link_begin_user:nnw` Notice that the link setup here uses `/Action` not `/A`. That is because Distiller *requires* this trigger word, rather than a “raw” PDF dictionary key (Ghostscript can handle either form).

`\__pdf_backend_link:nw` Taking the idea of `evenboxes` from `hypdvips`, we implement a minimum box height and depth for link placement. This means that “underlining” with a hyperlink will generally give an even appearance. However, to ensure that the full content is always above the link border, we do not allow this to be negative (contrast `hypdvips` approach).

`\__pdf_backend_link_aux:nw` The result should be similar to pdfTeX in the vast majority of foreseeable cases.

`\__pdf_backend_link_end:` The object number for a link is saved separately from the rest of the dictionary as this allows us to insert it just once, at either an unbroken link or only in the first line of a broken one. That makes the code clearer but also avoids a low-level PostScript error with the code as taken from `hypdvips`.

`\__pdf_backend_link_end_aux:` Getting the outer dimensions of the text area may be better using a two-pass approach and `\tex_savepos:D`. That plus generic mode are still to re-examine.

`\__pdf_backend_link_minima:`

`\__pdf_backend_link_outerbox:n`

`\__pdf_backend_link_sf_save:`

`\__pdf_backend_link_sf_restore:`

`pdf.linkdp.pad`

`pdf.linkht.pad`

`pdf.llx`

`pdf.lly`

`pdf.ury`

`pdf.link.dict`

`pdf.outerbox`

`pdf.baselineskip`

```

2361 \cs_new_protected:Npn \__pdf_backend_link_begin_goto:nnw #1#2
2362 {
2363   \__pdf_backend_link_begin:nw
2364   { #1 /Subtype /Link /Action << /S /GoTo /D ( #2 ) >> }
2365 }
2366 \cs_new_protected:Npn \__pdf_backend_link_begin_user:nnw #1#2
2367 { \__pdf_backend_link_begin:nw {#1#2} }
2368 \cs_new_protected:Npn \__pdf_backend_link_begin:nw #1
2369 {
2370   \bool_if:NF \g__pdf_backend_link_bool
2371   { \__pdf_backend_link_begin_aux:nw {#1} }
2372 }

```

The definition of `pdf.link.dict` here is needed as there is code in the PostScript headers for breaking links, and that can only work with this available.

```

2373 \cs_new_protected:Npn \__pdf_backend_link_begin_aux:nw #1
2374 {
2375   \bool_gset_true:N \g__pdf_backend_link_bool
2376   \__kernel_backend_postscript:n
2377   { /pdf.link.dict ( #1 ) def }

```

```

2378 \tl_gset:Nn \g__pdf_backend_link_dict_tl {#1}
2379 \__pdf_backend_link_sf_save:
2380 \mode_if_math:TF
2381 { \bool_gset_true:N \g__pdf_backend_link_math_bool }
2382 { \bool_gset_false:N \g__pdf_backend_link_math_bool }
2383 \hbox_set:Nw \l__pdf_backend_content_box
2384 \__pdf_backend_link_sf_restore:
2385 \bool_if:NT \g__pdf_backend_link_math_bool
2386 { \c_math_toggle_token }
2387 }
2388 \cs_new_protected:Npn \__pdf_backend_link_end:
2389 {
2390   \bool_if:NT \g__pdf_backend_link_bool
2391   { \__pdf_backend_link_end_aux: }
2392 }
2393 \cs_new_protected:Npn \__pdf_backend_link_end_aux:
2394 {
2395   \bool_if:NT \g__pdf_backend_link_math_bool
2396   { \c_math_toggle_token }
2397   \__pdf_backend_link_sf_save:
2398   \hbox_set_end:
2399   \__pdf_backend_link_minima:
2400   \hbox_set:Nn \l__pdf_backend_model_box { Gg }
2401   \exp_args:Nx \__pdf_backend_link_outerbox:n
2402   {
2403     \int_if_odd:nTF { \value { page } }
2404     { \oddsidemargin }
2405     { \evensidemargin }
2406   }
2407   \box_move_down:nn { \box_dp:N \l__pdf_backend_content_box }
2408   { \hbox:n { \__kernel_backend_postscript:n { pdf.save.linkll } } }
2409   \__pdf_breaklink_postscript:n { pdf.bordertracking.begin }
2410   \__pdf_breaklink_usebox:N \l__pdf_backend_content_box
2411   \__pdf_breaklink_postscript:n { pdf.bordertracking.end }
2412   \box_move_up:nn { \box_ht:N \l__pdf_backend_content_box }
2413   {
2414     \hbox:n
2415     { \__kernel_backend_postscript:n { pdf.save.linkur } }
2416   }
2417   \int_gincr:N \g__pdf_backend_object_int
2418   \int_gset_eq:NN \g__pdf_backend_link_int \g__pdf_backend_object_int
2419   \__kernel_backend_postscript:x
2420   {
2421     mark
2422     /_objdef { pdf.obj \int_use:N \g__pdf_backend_link_int }
2423     \g__pdf_backend_link_dict_tl \c_space_tl
2424     pdf.rect
2425     /ANN ~ \l__pdf_breaklink_pdfmark_tl
2426   }
2427   \__pdf_backend_link_sf_restore:
2428   \bool_gset_false:N \g__pdf_backend_link_bool
2429 }
2430 \cs_new_protected:Npn \__pdf_backend_link_minima:
2431 {

```



```

2432 \hbox_set:Nn \l__pdf_backend_model_box { Gg }
2433 \__kernel_backend_postscript:x
2434 {
2435   /pdf.linkdp.pad ~
2436   \dim_to_decimal:n
2437   {
2438     \dim_max:nn
2439     {
2440       \box_dp:N \l__pdf_backend_model_box
2441       - \box_dp:N \l__pdf_backend_content_box
2442     }
2443     { Opt }
2444   } ~
2445   pdf.pt.dvi ~ def
2446 /pdf.linkht.pad ~
2447 \dim_to_decimal:n
2448 {
2449   \dim_max:nn
2450   {
2451     \box_ht:N \l__pdf_backend_model_box
2452     - \box_ht:N \l__pdf_backend_content_box
2453   }
2454   { Opt }
2455 } ~
2456 pdf.pt.dvi ~ def
2457 }
2458 }
2459 \cs_new_protected:Npn \__pdf_backend_link_outerbox:n #1
2460 {
2461   \__kernel_backend_postscript:x
2462   {
2463     /pdf.outerbox
2464     [
2465       \dim_to_decimal:n {#1} ~
2466       \dim_to_decimal:n { -\box_dp:N \l__pdf_backend_model_box } ~
2467       \dim_to_decimal:n { #1 + \textwidth } ~
2468       \dim_to_decimal:n { \box_ht:N \l__pdf_backend_model_box }
2469     ]
2470     [ exch { pdf.pt.dvi } forall ] def
2471 /pdf.baselineskip ~
2472 \dim_to_decimal:n { \tex_baselineskip:D } ~ dup ~ 0 ~ gt
2473 { pdf.pt.dvi ~ def }
2474 { pop ~ pop }
2475 ifelse
2476 }
2477 }
2478 \cs_new_protected:Npn \__pdf_backend_link_sf_save:
2479 {
2480   \int_gset:Nn \g__pdf_backend_link_sf_int
2481   {
2482     \mode_if_horizontal:TF
2483     { \tex_spacefactor:D }
2484     { 0 }
2485   }

```

```

2486 }
2487 \cs_new_protected:Npn \__pdf_backend_link_sf_restore:
2488 {
2489   \mode_if_horizontal:T
2490   {
2491     \int_compare:nNnT \g__pdf_backend_link_sf_int > { 0 }
2492     { \int_set_eq:NN \tex_spacefactor:D \g__pdf_backend_link_sf_int }
2493   }
2494 }

```

(End definition for \\_\_pdf\_backend\_link\_begin\_goto:nw and others. These functions are documented on page ??.)

**\@makecol@hook** Hooks to allow link breaking: something will be needed in format mode at some stage. At present this code is disabled as there is an open question about the name of the hook: to be resolved at the L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> end.

```

2495 \use_none:n
2496 {
2497   \cs_if_exist:NT \@makecol@hook
2498   {
2499     \tl_put_right:Nn \@makecol@hook
2500     {
2501       \box_if_empty:NF \@cclv
2502       {
2503         \vbox_set:Nn \@cclv
2504         {
2505           \__kernel_backend_postscript:n
2506           {
2507             pdf.globaldict /pdf.brokenlink.rect ~ known
2508             { pdf.bordertracking.continue }
2509             if
2510           }
2511           \vbox_unpack_drop:N \@cclv
2512           \__kernel_backend_postscript:n
2513           { pdf.bordertracking.endpage }
2514         }
2515       }
2516     }
2517     \tl_set:Nn \l__pdf_breaklink_pdfmark_tl { pdf.pdfmark }
2518     \cs_set_eq:NN \__pdf_breaklink_postscript:n \__kernel_backend_postscript:n
2519     \cs_set_eq:NN \__pdf_breaklink_usebox:N \hbox_unpack:N
2520   }
2521 }

```

(End definition for \@makecol@hook. This function is documented on page ??.)

**\\_\_pdf\_backend\_link\_last:** The same as annotations, but with a custom integer.

```

2522 \cs_new:Npn \__pdf_backend_link_last:
2523 { { pdf.obj \int_use:N \g__pdf_backend_link_int } }

```

(End definition for \\_\_pdf\_backend\_link\_last:.)

**\\_\_pdf\_backend\_link\_margin:n** Convert to big points and pass to PostScript.

```

2524 \cs_new_protected:Npn \__pdf_backend_link_margin:n #1
2525 {

```

```

2526   \__kernel_backend_postscript:x
2527   {
2528     /pdf.linkmargin { \dim_to_decimal:n {#1} ~ pdf.pt.dvi } def
2529   }
2530 }

```

(End definition for \\_\_pdf\_backend\_link\_margin:n.)

```

\__pdf_backend_destination:nn
\__pdf_backend_destination:nnnn
\__pdf_backend_destination_aux:nnnn

```

Here, we need to turn the zoom into a scale. We also need to know where the current anchor point actually is: worked out in PostScript. For the rectangle version, we have a bit more PostScript: we need two points. `fitr` without rule spec doesn't work, so it falls back to `/Fit` here.

```

2531 \cs_new_protected:Npn \__pdf_backend_destination:nn #1#2
2532 {
2533   \__kernel_backend_postscript:n { pdf.dest.anchor }
2534   \__pdf_backend_pdfmark:x
2535   {
2536     /View
2537     [
2538       \str_case:nnF {#2}
2539       {
2540         { xyz } { /XYZ ~ pdf.dest.point ~ null }
2541         { fit } { /Fit }
2542         { fitb } { /FitB }
2543         { fitbh } { /FitBH ~ pdf.dest.y }
2544         { fitbv } { /FitBV ~ pdf.dest.x }
2545         { fith } { /FitH ~ pdf.dest.y }
2546         { fitv } { /FitV ~ pdf.dest.x }
2547         { fitr } { /Fit }
2548       }
2549       {
2550         /XYZ ~ pdf.dest.point ~ \fp_eval:n { (#2) / 100 }
2551       }
2552     ]
2553     /Dest ( \exp_not:n {#1} ) cvn
2554     /DEST
2555   }
2556 }
2557 \cs_new_protected:Npn \__pdf_backend_destination:nnnn #1#2#3#4
2558 {
2559   \exp_args:Ne \__pdf_backend_destination_aux:nnnn
2560   { \dim_eval:n {#2} } {#1} {#3} {#4}
2561 }
2562 \cs_new_protected:Npn \__pdf_backend_destination_aux:nnnn #1#2#3#4
2563 {
2564   \vbox_to_zero:n
2565   {
2566     \__kernel_kern:n {#4}
2567     \hbox:n { \__kernel_backend_postscript:n { pdf.save.ll } }
2568     \tex_vss:D
2569   }
2570   \__kernel_kern:n {#1}
2571   \vbox_to_zero:n
2572   {

```

```

2573     \__kernel_kern:n { -#3 }
2574     \hbox:n { \__kernel_backend_postscript:n { pdf.save.ur } }
2575     \tex_vss:D
2576   }
2577   \__kernel_kern:n { -#1 }
2578   \__pdf_backend_pdfmark:n
2579   {
2580     /View
2581     [
2582       /FitR ~
2583       pdf.llx ~ pdf.lly ~ pdf.dest2device ~
2584       pdf.urx ~ pdf.ury ~ pdf.dest2device
2585     ]
2586     /Dest ( #2 ) cvn
2587     /DEST
2588   }
2589 }

(End definition for \__pdf_backend_destination:nn, \__pdf_backend_destination:nnnn, and \__-
pdf_backend_destination_aux:nnnn.)

```

#### 6.2.4 Structure

Doable for the usual ps2pdf method.

```

\__pdf_backend_compresslevel:n
\__pdf_backend_compress_objects:n
2590 \cs_new_protected:Npn \__pdf_backend_compresslevel:n #1
2591 {
2592   \int_compare:nNnT {#1} = 0
2593   {
2594     \__kernel_backend_literal_postscript:n
2595     {
2596       /setdistillerparams ~ where
2597       { pop << /CompressPages ~ false >> setdistillerparams }
2598       if
2599     }
2600   }
2601 }
2602 \cs_new_protected:Npn \__pdf_backend_compress_objects:n #1
2603 {
2604   \bool_if:nF {#1}
2605   {
2606     \__kernel_backend_literal_postscript:n
2607     {
2608       /setdistillerparams ~ where
2609       { pop << /CompressStreams ~ false >> setdistillerparams }
2610       if
2611     }
2612   }
2613 }

(End definition for \__pdf_backend_compresslevel:n and \__pdf_backend_compress_objects:n.)

```

```

\__pdf_backend_version_major_gset:n
\__pdf_backend_version_minor_gset:n
2614 \cs_new_protected:Npn \__pdf_backend_version_major_gset:n #1
2615 {
2616   \cs_gset:Npx \__pdf_backend_version_major: { \int_eval:n {#1} }

```

```

2617 }
2618 \cs_new_protected:Npn \__pdf_backend_version_minor_gset:n #1
2619 {
2620   \cs_gset:Npx \__pdf_backend_version_minor: { \int_eval:n {#1} }
2621 }

```

(End definition for \\_\_pdf\_backend\_version\_major\_gset:n and \\_\_pdf\_backend\_version\_minor\_gset:n.)

\\_\_pdf\_backend\_version\_major: Data not available!

```

\__pdf_backend_version_minor:
2622 \cs_new:Npn \__pdf_backend_version_major: { -1 }
2623 \cs_new:Npn \__pdf_backend_version_minor: { -1 }

```

(End definition for \\_\_pdf\_backend\_version\_major: and \\_\_pdf\_backend\_version\_minor:.)

### 6.2.5 Marked content

\\_\_pdf\_backend\_bdc:nn Simple wrappers.

```

\__pdf_backend_emc:
2624 \cs_new_protected:Npn \__pdf_backend_bdc:nn #1#2
2625 { \__pdf_backend_pdfmark:n { /#1 ~ #2 /BDC } }
2626 \cs_new_protected:Npn \__pdf_backend_emc:
2627 { \__pdf_backend_pdfmark:n { /EMC } }

```

(End definition for \\_\_pdf\_backend\_bdc:nn and \\_\_pdf\_backend\_emc:.)

2628 </dvips>

## 6.3 LuaTeX and pdfTeX backend

2629 <\*luatex | pdftex>

### 6.3.1 Annotations

\\_\_pdf\_backend\_annotation:nnnn Simply pass the raw data through, just dealing with evaluation of dimensions.

```

2630 \cs_new_protected:Npn \__pdf_backend_annotation:nnnn #1#2#3#4
2631 {
2632   <*luatex>
2633   \tex_pdfextension:D annot ~
2634   </luatex>
2635   <*pdftex>
2636   \tex_pdfannot:D
2637   </pdftex>
2638   width ~ \dim_eval:n {#1} ~
2639   height ~ \dim_eval:n {#2} ~
2640   depth ~ \dim_eval:n {#3} ~
2641   {#4}
2642 }

```

(End definition for \\_\_pdf\_backend\_annotation:nnnn.)

\\_\_pdf\_backend\_annotation\_last: A tiny amount of extra data gets added here; we use x-type expansion to get the space in the right place and form. The “extra” space in the LuaTeX version is *required* as it is consumed in finding the end of the keyword.

```

2643 \cs_new:Npx \__pdf_backend_annotation_last:
2644 {
2645   \exp_not:N \int_value:w
2646   <*luatex>

```

```

2647 \exp_not:N \tex_pdffeedback:D lastannot ~
2648 </luatex>
2649 <*pdfTeX>
2650 \exp_not:N \tex_pdflastannot:D
2651 </pdfTeX>
2652 \c_space_tl 0 ~ R
2653 }

```

(End definition for \\_pdf\_backend\_annotation\_last:.)

```

\_pdf_backend_link_begin_goto:nnw Links are all created using the same internals.
\_pdf_backend_link_begin_user:nnw
\_pdf_backend_link_begin:nnnw
\_pdf_backend_link_end:
2654 \cs_new_protected:Npn \_pdf_backend_link_begin_goto:nnw #1#2
2655 { \_pdf_backend_link_begin:nnnw {#1} { goto~name } {#2} }
2656 \cs_new_protected:Npn \_pdf_backend_link_begin_user:nnw #1#2
2657 { \_pdf_backend_link_begin:nnnw {#1} { user } {#2} }
2658 \cs_new_protected:Npn \_pdf_backend_link_begin:nnnw #1#2#3
2659 {
2660 <*luatex>
2661 \tex_pdfextension:D startlink ~
2662 </luatex>
2663 <*pdfTeX>
2664 \tex_pdfstartlink:D
2665 </pdfTeX>
2666 attr {#1}
2667 #2 {#3}
2668 }
2669 \cs_new_protected:Npn \_pdf_backend_link_end:
2670 {
2671 <*luatex>
2672 \tex_pdfextension:D endlink \scan_stop:
2673 </luatex>
2674 <*pdfTeX>
2675 \tex_pdfendlink:D
2676 </pdfTeX>
2677 }

```

(End definition for \\_pdf\_backend\_link\_begin\_goto:nnw and others.)

```

\_pdf_backend_link_last: Formatted for direct use.
2678 \cs_new:Npx \_pdf_backend_link_last:
2679 {
2680 \exp_not:N \int_value:w
2681 <*luatex>
2682 \exp_not:N \tex_pdffeedback:D lastlink ~
2683 </luatex>
2684 <*pdfTeX>
2685 \exp_not:N \tex_pdflastlink:D
2686 </pdfTeX>
2687 \c_space_tl 0 ~ R
2688 }

```

(End definition for \\_pdf\_backend\_link\_last:.)

`\_pdf_backend_link_margin:n` A simple task: pass the data to the primitive.

```

2689 \cs_new_protected:Npn \_pdf_backend_link_margin:n #1
2690 {
2691   <*luatex>
2692     \tex_pdfvariable:D linkmargin
2693   </luatex>
2694   <*pdftex>
2695     \tex_pdflinkmargin:D
2696   </pdftex>
2697     \dim_eval:n {#1} \scan_stop:
2698 }

```

(End definition for `\_pdf_backend_link_margin:n`.)

`\_pdf_backend_destination:nn` A simple task: pass the data to the primitive. The `\scan_stop:` deals with the danger of an unterminated keyword. The zoom given here is a percentage, but we need to pass it as *per mille*. The rectangle version is also easy as everything is build in.

`\_pdf_backend_destination:nnnn`

```

2699 \cs_new_protected:Npn \_pdf_backend_destination:nn #1#2
2700 {
2701   <*luatex>
2702     \tex_pdfextension:D dest ~
2703   </luatex>
2704   <*pdftex>
2705     \tex_pdfdest:D
2706   </pdftex>
2707     name {#1}
2708     \str_case:nnF {#2}
2709     {
2710       { xyz } { xyz }
2711       { fit } { fit }
2712       { fitb } { fitb }
2713       { fitbh } { fitbh }
2714       { fitbv } { fitbv }
2715       { fith } { fith }
2716       { fitv } { fitv }
2717       { fitr } { fitr }
2718     }
2719     { xyz ~ zoom \fp_eval:n { #2 * 10 } }
2720     \scan_stop:
2721 }
2722 \cs_new_protected:Npn \_pdf_backend_destination:nnnn #1#2#3#4
2723 {
2724   <*luatex>
2725     \tex_pdfextension:D dest ~
2726   </luatex>
2727   <*pdftex>
2728     \tex_pdfdest:D
2729   </pdftex>
2730     name {#1}
2731     fitr ~
2732     width \dim_eval:n {#2} ~
2733     height \dim_eval:n {#3} ~
2734     depth \dim_eval:n {#4} \scan_stop:
2735 }

```

(End definition for `\_pdf_backend_destination:nn` and `\_pdf_backend_destination:nnnn`.)

### 6.3.2 Catalogue entries

```

\__pdf_backend_catalog_gput:nn
\__pdf_backend_info_gput:nn
2736 \cs_new_protected:Npn \__pdf_backend_catalog_gput:nn #1#2
2737 {
2738   \*luatex
2739     \tex_pdfextension:D catalog
2740   \*pdfTeX
2741     \tex_pdfcatalog:D
2742   \*pdfTeX
2743     { / #1 ~ #2 }
2744 }
2745
2746 \cs_new_protected:Npn \__pdf_backend_info_gput:nn #1#2
2747 {
2748   \*luatex
2749     \tex_pdfextension:D info
2750   \*pdfTeX
2751     \tex_pdfinfo:D
2752   \*pdfTeX
2753     { / #1 ~ #2 }
2754 }
2755

```

(End definition for `\_pdf_backend_catalog_gput:nn` and `\_pdf_backend_info_gput:nn`.)

### 6.3.3 Objects

`\g__pdf_backend_object_prop` For tracking objects to allow finalisation.

```

2756 \prop_new:N \g__pdf_backend_object_prop

```

(End definition for `\g__pdf_backend_object_prop`.)

`\__pdf_backend_object_new:nn` Declaring objects means reserving at the PDF level plus starting tracking.

```

\__pdf_backend_object_ref:n
2757 \cs_new_protected:Npn \__pdf_backend_object_new:nn #1#2
2758 {
2759   \*luatex
2760     \tex_pdfextension:D obj ~
2761   \*pdfTeX
2762     \tex_pdfobj:D
2763   \*pdfTeX
2764     {
2765       reserveobjnum ~
2766       \int_const:cn
2767       { c__pdf_backend_object_ \tl_to_str:n {#1} _int }
2768     }
2769   \*luatex
2770     { \tex_pdffeedback:D lastobj }
2771   \*pdfTeX
2772     { \tex_pdflastobj:D }
2773   \*pdfTeX
2774     \prop_gput:Nnn \g__pdf_backend_object_prop {#1} {#2}

```



```

2775 }
2776 \cs_new:Npn \__pdf_backend_object_ref:n #1
2777 { \int_use:c { c__pdf_backend_object_ \tl_to_str:n {#1} _int } ~ 0 ~ R }

```

(End definition for \\_\_pdf\_backend\_object\_new:nn and \\_\_pdf\_backend\_object\_ref:n.)

\\_\_pdf\_backend\_object\_write:nn

Writing the data needs a little information about the structure of the object.

\\_\_pdf\_backend\_object\_write:nx

\\_\_pdf\_exp\_not\_i:nn

\\_\_pdf\_exp\_not\_ii:nn

```

2778 \cs_new_protected:Npn \__pdf_backend_object_write:nn #1#2
2779 {
2780 <*luatex>
2781   \tex_immediate:D \tex_pdfextension:D obj ~
2782 </luatex>
2783 <*pdfTeX>
2784   \tex_immediate:D \tex_pdfobj:D
2785 </pdfTeX>
2786   useobjnum ~
2787   \int_use:c
2788   { c__pdf_backend_object_ \tl_to_str:n {#1} _int }
2789   \str_case_e:nn
2790   { \prop_item:Nn \g__pdf_backend_object_prop {#1} }
2791   {
2792     { array } { { [ ~ \exp_not:n {#2} ~ ] } }
2793     { dict } { { << ~ \exp_not:n {#2} ~ >> } }
2794     { fstream }
2795     {
2796       stream ~ attr ~ { \__pdf_exp_not_i:nn #2 } ~
2797       file ~ { \__pdf_exp_not_ii:nn #2 }
2798     }
2799     { stream }
2800     {
2801       stream ~ attr ~ { \__pdf_exp_not_i:nn #2 } ~
2802       { \__pdf_exp_not_ii:nn #2 }
2803     }
2804   }
2805 }
2806 \cs_generate_variant:Nn \__pdf_backend_object_write:nn { nx }
2807 \cs_new:Npn \__pdf_exp_not_i:nn #1#2 { \exp_not:n {#1} }
2808 \cs_new:Npn \__pdf_exp_not_ii:nn #1#2 { \exp_not:n {#2} }

```

(End definition for \\_\_pdf\_backend\_object\_write:nn, \\_\_pdf\_exp\_not\_i:nn, and \\_\_pdf\_exp\_not\_ii:nn.)

\\_\_pdf\_backend\_object\_now:nn

Much like writing, but direct creation.

\\_\_pdf\_backend\_object\_now:nx

```

2809 \cs_new_protected:Npn \__pdf_backend_object_now:nn #1#2
2810 {
2811 <*luatex>
2812   \tex_immediate:D \tex_pdfextension:D obj ~
2813 </luatex>
2814 <*pdfTeX>
2815   \tex_immediate:D \tex_pdfobj:D
2816 </pdfTeX>
2817   \str_case:nn
2818   {#1}
2819   {
2820     { array } { { [ ~ \exp_not:n {#2} ~ ] } }

```

```

2821         { dict } { { << ~ \exp_not:n {#2} ~ >> } }
2822         { fstream }
2823         {
2824             stream ~ attr ~ { \__pdf_exp_not_i:nn #2 } ~
2825             file ~ { \__pdf_exp_not_ii:nn #2 }
2826         }
2827         { stream }
2828         {
2829             stream ~ attr ~ { \__pdf_exp_not_i:nn #2 } ~
2830             { \__pdf_exp_not_ii:nn #2 }
2831         }
2832     }
2833 }
2834 \cs_generate_variant:Nn \__pdf_backend_object_now:nn { nx }
(End definition for \__pdf_backend_object_now:nn.)

```

`\__pdf_backend_object_last:` Much like annotation.

```

2835 \cs_new:Npx \__pdf_backend_object_last:
2836 {
2837     \exp_not:N \int_value:w
2838     <*luatex>
2839     \exp_not:N \tex_pdffeedback:D lastobj ~
2840     </luatex>
2841     <*pdftex>
2842     \exp_not:N \tex_pdflastobj:D
2843     </pdftex>
2844     \c_space_tl 0 ~ R
2845 }
(End definition for \__pdf_backend_object_last:.)

```

`\__pdf_backend_pageobject_ref:n` The usual wrapper situation; the three spaces here are essential.

```

2846 \cs_new:Npx \__pdf_backend_pageobject_ref:n #1
2847 {
2848     \exp_not:N \int_value:w
2849     <*luatex>
2850     \exp_not:N \tex_pdffeedback:D pageref
2851     </luatex>
2852     <*pdftex>
2853     \exp_not:N \tex_pdfpageref:D
2854     </pdftex>
2855     \c_space_tl #1 \c_space_tl \c_space_tl \c_space_tl 0 ~ R
2856 }
(End definition for \__pdf_backend_pageobject_ref:n.)

```

### 6.3.4 Structure

`\__pdf_backend_compresslevel:n` Simply pass data to the engine.

```

\__pdf_backend_compress_objects:n 2857 \cs_new_protected:Npn \__pdf_backend_compresslevel:n #1
\__pdf_backend_objcompresslevel:n 2858 {
2859     \tex_global:D
2860     <*luatex>
2861     \tex_pdfvariable:D compresslevel

```

```

2862 </luatex>
2863 <*pdftex>
2864     \tex_pdfcompresslevel:D
2865 </pdftex>
2866     \int_value:w \int_eval:n {#1} \scan_stop:
2867 }
2868 \cs_new_protected:Npn \__pdf_backend_compress_objects:n #1
2869 {
2870     \bool_if:nTF {#1}
2871     { \__pdf_backend_objcompresslevel:n { 2 } }
2872     { \__pdf_backend_objcompresslevel:n { 0 } }
2873 }
2874 \cs_new_protected:Npn \__pdf_backend_objcompresslevel:n #1
2875 {
2876     \tex_global:D
2877 <*luatex>
2878     \tex_pdfvariable:D objcompresslevel
2879 </luatex>
2880 <*pdftex>
2881     \tex_pdfobjcompresslevel:D
2882 </pdftex>
2883     #1 \scan_stop:
2884 }

(End definition for \__pdf_backend_compresslevel:n, \__pdf_backend_compress_objects:n, and \__-
pdf_backend_objcompresslevel:n.)

```

`\__pdf_backend_version_major_gset:n`  
`\__pdf_backend_version_minor_gset:n`

The availability of the primitive is not universal, so we have to test at load time.

```

2885 \cs_new_protected:Npx \__pdf_backend_version_major_gset:n #1
2886 {
2887 <*luatex>
2888     \int_compare:nNnT \tex_luatexversion:D > { 106 }
2889     {
2890         \exp_not:N \tex_global:D \tex_pdfvariable:D majorversion
2891         \exp_not:N \int_eval:n {#1} \scan_stop:
2892     }
2893 </luatex>
2894 <*pdftex>
2895     \cs_if_exist:NT \tex_pdfmajorversion:D
2896     {
2897         \exp_not:N \tex_global:D \tex_pdfmajorversion:D
2898         \exp_not:N \int_eval:n {#1} \scan_stop:
2899     }
2900 </pdftex>
2901 }
2902 \cs_new_protected:Npn \__pdf_backend_version_minor_gset:n #1
2903 {
2904     \tex_global:D
2905 <*luatex>
2906     \tex_pdfvariable:D minorversion
2907 </luatex>
2908 <*pdftex>
2909     \tex_pdfminorversion:D
2910 </pdftex>

```

```

2911         \int_eval:n {#1} \scan_stop:
2912     }

```

(End definition for `\__pdf_backend_version_major_gset:n` and `\__pdf_backend_version_minor_gset:n`.)

`\__pdf_backend_version_major:` As above.

```

\__pdf_backend_version_minor:
2913 \cs_new:Npx \__pdf_backend_version_major:
2914 {
2915   \*luatex
2916   \int_compare:nNnTF \tex_luatexversion:D > { 106 }
2917   { \exp_not:N \tex_the:D \tex_pdfvariable:D majorversion }
2918   { 1 }
2919 }
2920 \*pdfTeX
2921 \cs_if_exist:NTF \tex_pdfmajorversion:D
2922 { \exp_not:N \tex_the:D \tex_pdfmajorversion:D }
2923 { 1 }
2924 }
2925 }
2926 \cs_new:Npn \__pdf_backend_version_minor:
2927 {
2928   \tex_the:D
2929   \*luatex
2930   \tex_pdfvariable:D minorversion
2931   \*luatex
2932   \*pdfTeX
2933   \tex_pdfminorversion:D
2934   \*pdfTeX
2935 }

```

(End definition for `\__pdf_backend_version_major:` and `\__pdf_backend_version_minor:.`)

### 6.3.5 Marked content

`\__pdf_backend_bdc:nn` Simple wrappers. May need refinement: see <https://chat.stackexchange.com/transcript/message/49970158#49970158>.

```

\__pdf_backend_emc:
2936 \cs_new_protected:Npn \__pdf_backend_bdc:nn #1#2
2937 { \__kernel_backend_literal_page:n { /#1 ~ #2 ~ BDC } }
2938 \cs_new_protected:Npn \__pdf_backend_emc:
2939 { \__kernel_backend_literal_page:n { EMC } }

```

(End definition for `\__pdf_backend_bdc:nn` and `\__pdf_backend_emc:.`)

```

2940 \*luatex | pdfTeX

```

## 6.4 dvipdfmx backend

```

2941 \*dvipdfmx | xetex

```

`\__pdf_backend:n` A generic function for the backend PDF specials: used where we can.

```

\__pdf_backend:x
2942 \cs_new_protected:Npx \__pdf_backend:n #1
2943 { \__kernel_backend_literal:n { pdf: #1 } }
2944 \cs_generate_variant:Nn \__pdf_backend:n { x }

```

(End definition for `\__pdf_backend:n`.)

### 6.4.1 Catalogue entries

```

    \_pdf_backend_catalog_gput:nn
\_pdf_backend_info_gput:nn
2945 \cs_new_protected:Npn \_pdf_backend_catalog_gput:nn #1#2
2946 { \_pdf_backend:n { put ~ @catalog << /#1 ~ #2 >> } }
2947 \cs_new_protected:Npn \_pdf_backend_info_gput:nn #1#2
2948 { \_pdf_backend:n { docinfo << /#1 ~ #2 >> } }

(End definition for \_pdf_backend_catalog_gput:nn and \_pdf_backend_info_gput:nn.)

```

### 6.4.2 Objects

For tracking objects to allow finalisation.

```

2949 \int_new:N \g__pdf_backend_object_int
2950 \prop_new:N \g__pdf_backend_object_prop

(End definition for \g__pdf_backend_object_int and \g__pdf_backend_object_prop.)

```

Objects are tracked at the macro level, but we don't have to do anything at this stage.

```

\_pdf_backend_object_new:nn
\_pdf_backend_object_ref:n
2951 \cs_new_protected:Npn \_pdf_backend_object_new:nn #1#2
2952 {
2953   \int_gincr:N \g__pdf_backend_object_int
2954   \int_const:cn
2955     { c__pdf_backend_object_ \tl_to_str:n {#1} _int }
2956     { \g__pdf_backend_object_int }
2957   \prop_gput:Nnn \g__pdf_backend_object_prop {#1} {#2}
2958 }
2959 \cs_new:Npn \_pdf_backend_object_ref:n #1
2960 { @pdf.obj \int_use:c { c__pdf_backend_object_ \tl_to_str:n {#1} _int } }

(End definition for \_pdf_backend_object_new:nn and \_pdf_backend_object_ref:n.)

```

This is where we choose the actual type.

```

\_pdf_backend_object_write:nn
\_pdf_backend_object_write:nx
\_pdf_backend_object_write:nnn
\_pdf_backend_object_write_array:nn
\_pdf_backend_object_write_dict:nn
\_pdf_backend_object_write_fstream:nn
\_pdf_backend_object_write_stream:nn
\_pdf_backend_object_write_stream:nnnn
2961 \cs_new_protected:Npn \_pdf_backend_object_write:nn #1#2
2962 {
2963   \exp_args:Nx \_pdf_backend_object_write:nnn
2964     { \prop_item:Nn \g__pdf_backend_object_prop {#1} } {#1} {#2}
2965 }
2966 \cs_generate_variant:Nn \_pdf_backend_object_write:nn { nx }
2967 \cs_new_protected:Npn \_pdf_backend_object_write:nnn #1#2#3
2968 {
2969   \use:c { __pdf_backend_object_write_ #1 :nn }
2970   { \_pdf_backend_object_ref:n {#2} } {#3}
2971 }
2972 \cs_new_protected:Npn \_pdf_backend_object_write_array:nn #1#2
2973 {
2974   \_pdf_backend:x
2975     { obj ~ #1 ~ [ ~ \exp_not:n {#2} ~ ] }
2976 }
2977 \cs_new_protected:Npn \_pdf_backend_object_write_dict:nn #1#2
2978 {
2979   \_pdf_backend:x
2980     { obj ~ #1 ~ << ~ \exp_not:n {#2} ~ >> }
2981 }
2982 \cs_new_protected:Npn \_pdf_backend_object_write_fstream:nn #1#2

```

```

2983 { \_pdf_backend_object_write_stream:nnnn { f } {#1} #2 }
2984 \cs_new_protected:Npn \_pdf_backend_object_write_stream:nn #1#2
2985 { \_pdf_backend_object_write_stream:nnnn { } {#1} #2 }
2986 \cs_new_protected:Npn \_pdf_backend_object_write_stream:nnnn #1#2#3#4
2987 {
2988   \_pdf_backend:x
2989   {
2990     #1 stream ~ #2 ~
2991     ( \exp_not:n {#4} ) ~ << \exp_not:n {#3} >>
2992   }
2993 }

```

(End definition for \\_pdf\_backend\_object\_write:nn and others.)

\\_pdf\_backend\_object\_now:nn No anonymous objects with dvipdfmx so we have to give an object name.  
\\_pdf\_backend\_object\_now:nx

```

2994 \cs_new_protected:Npn \_pdf_backend_object_now:nn #1#2
2995 {
2996   \int_gincr:N \g__pdf_backend_object_int
2997   \exp_args:Nnx \use:c { \_pdf_backend_object_write_ #1 :nn }
2998   { @pdf.obj \int_use:N \g__pdf_backend_object_int }
2999   {#2}
3000 }
3001 \cs_generate_variant:Nn \_pdf_backend_object_now:nn { nx }

```

(End definition for \\_pdf\_backend\_object\_now:nn.)

\\_pdf\_backend\_object\_last:

```

3002 \cs_new:Npn \_pdf_backend_object_last:
3003 { @pdf.obj \int_use:N \g__pdf_backend_object_int }

```

(End definition for \\_pdf\_backend\_object\_last:.)

\\_pdf\_backend\_pageobject\_ref:n Page references are easy in dvipdfmx/X<sub>Y</sub>TeX.

```

3004 \cs_new:Npn \_pdf_backend_pageobject_ref:n #1
3005 { @page #1 }

```

(End definition for \\_pdf\_backend\_pageobject\_ref:n.)

### 6.4.3 Annotations

\g\_\_pdf\_backend\_annotation\_int Needed as objects which are not annotations could be created.

```

3006 \int_new:N \g__pdf_backend_annotation_int

```

(End definition for \g\_\_pdf\_backend\_annotation\_int.)

\\_pdf\_backend\_annotation:nnnn Simply pass the raw data through, just dealing with evaluation of dimensions.

```

3007 \cs_new_protected:Npn \_pdf_backend_annotation:nnnn #1#2#3#4
3008 {
3009   \int_gincr:N \g__pdf_backend_object_int
3010   \int_gset_eq:NN \g__pdf_backend_annotation_int \g__pdf_backend_object_int
3011   \_pdf_backend:x
3012   {
3013     ann ~ @pdf.obj \int_use:N \g__pdf_backend_object_int \c_space_tl
3014     width ~ \dim_eval:n {#1} ~
3015     height ~ \dim_eval:n {#2} ~

```

```

3016         depth ~ \dim_eval:n {#3} ~
3017         << /Type /Annot #4 >>
3018     }
3019 }

```

(End definition for \\_pdf\_backend\_annotation:nnnn.)

\\_pdf\_backend\_annotation\_last:

```

3020 \cs_new:Npn \_pdf_backend_annotation_last:
3021 { @pdf.obj \int_use:N \g_pdf_backend_annotation_int }

```

(End definition for \\_pdf\_backend\_annotation\_last:.)

\g\_pdf\_backend\_link\_int To track annotations which are links.

```

3022 \int_new:N \g_pdf_backend_link_int

```

(End definition for \g\_pdf\_backend\_link\_int.)

\\_pdf\_backend\_link\_begin\_goto:nnw All created using the same internals.

\\_pdf\_backend\_link\_begin\_user:nnw

\\_pdf\_backend\_link\_begin:n

\\_pdf\_backend\_link\_end:

```

3023 \cs_new_protected:Npn \_pdf_backend_link_begin_goto:nnw #1#2
3024 { \_pdf_backend_link_begin:n { #1 /Subtype /Link /A << /S /GoTo /D ( #2 ) >> } }
3025 \cs_new_protected:Npn \_pdf_backend_link_begin_user:nnw #1#2
3026 { \_pdf_backend_link_begin:n {#1#2} }
3027 \cs_new_protected:Npx \_pdf_backend_link_begin:n #1
3028 {
3029     \exp_not:N \int_gincr:N \exp_not:N \g_pdf_backend_link_int
3030     \_pdf_backend:x
3031     {
3032         bann ~
3033         @pdf.lnk
3034         \exp_not:N \int_use:N \exp_not:N \g_pdf_backend_link_int
3035         \c_space_tl
3036         <<
3037         /Type /Annot
3038         #1
3039         >>
3040     }
3041 }
3042 \cs_new_protected:Npn \_pdf_backend_link_end:
3043 { \_pdf_backend:n { eann } }

```

(End definition for \\_pdf\_backend\_link\_begin\_goto:nnw and others.)

\\_pdf\_backend\_link\_last: Available using the backend mechanism with a suitably-recent version.

```

3044 \cs_new:Npn \_pdf_backend_link_last:
3045 { @pdf.lnk \int_use:N \g_pdf_backend_link_int }

```

(End definition for \\_pdf\_backend\_link\_last:.)

\\_pdf\_backend\_link\_margin:n Pass to dvipdfmx.

```

3046 \cs_new_protected:Npn \_pdf_backend_link_margin:n #1
3047 { \_kernel_backend_literal:x { dvipdfmx:config~ \dim_eval:n {#1} } }

```

(End definition for \\_pdf\_backend\_link\_margin:n.)

`\_pdf_backend_destination:nn`  
`\_pdf_backend_destination:nnnn`  
`\_pdf_backend_destination_aux:nnnn`

Here, we need to turn the zoom into a scale. The method for `FitR` is from Alexander Grahn: the idea is to avoid needing to do any calculations in  $\text{\TeX}$  by using the backend data for `@xpos` and `@ypos`. `/FitR` without rule spec doesn't work, so it falls back to `/Fit` here.

```

3048 \cs_new_protected:Npn \_pdf_backend_destination:nn #1#2
3049 {
3050   \_pdf_backend:x
3051   {
3052     dest ~ ( \exp_not:n {#1} )
3053     [
3054       @thispage
3055       \str_case:nnF {#2}
3056       {
3057         { xyz } { /XYZ ~ @xpos ~ @ypos ~ null }
3058         { fit } { /Fit }
3059         { fitb } { /FitB }
3060         { fitbh } { /FitBH }
3061         { fitbv } { /FitBV ~ @xpos }
3062         { fith } { /FitH ~ @ypos }
3063         { fitv } { /FitV ~ @xpos }
3064         { fitr } { /Fit }
3065       }
3066       { /XYZ ~ @xpos ~ @ypos ~ \fp_eval:n { (#2) / 100 } }
3067     ]
3068   }
3069 }
3070 \cs_new_protected:Npn \_pdf_backend_destination:nnnn #1#2#3#4
3071 {
3072   \exp_args:Ne \_pdf_backend_destination_aux:nnnn
3073   { \dim_eval:n {#2} } {#1} {#3} {#4}
3074 }
3075 \cs_new_protected:Npn \_pdf_backend_destination_aux:nnnn #1#2#3#4
3076 {
3077   \vbox_to_zero:n
3078   {
3079     \_kernel_kern:n {#4}
3080     \hbox:n
3081     {
3082       \_pdf_backend:n { obj ~ @pdf_ #2 _llx ~ @xpos }
3083       \_pdf_backend:n { obj ~ @pdf_ #2 _lly ~ @ypos }
3084     }
3085     \tex_vss:D
3086   }
3087   \_kernel_kern:n {#1}
3088   \vbox_to_zero:n
3089   {
3090     \_kernel_kern:n { -#3 }
3091     \hbox:n
3092     {
3093       \_pdf_backend:n
3094       {
3095         dest ~ (#2)
3096         [
3097           @thispage

```



```

3098             /FitR ~
3099             @pdf_ #2 _llx ~ @pdf_ #2 _lly ~
3100             @xpos ~ @ypos
3101         ]
3102     }
3103 }
3104 \tex_vss:D
3105 }
3106 \__kernel_kern:n { -#1 }
3107 }

```

(End definition for \\_\_pdf\_backend\_destination:nn, \\_\_pdf\_backend\_destination:nnnn, and \\_\_pdf\_backend\_destination\_aux:nnnn.)

#### 6.4.4 Structure

\\_\_pdf\_backend\_compresslevel:n  
\\_\_pdf\_backend\_compress\_objects:n

Pass data to the backend: these are a one-shot.

```

3108 \cs_new_protected:Npn \__pdf_backend_compresslevel:n #1
3109 { \__kernel_backend_literal:x { dvipdfmx:config~z~ \int_eval:n {#1} } }
3110 \cs_new_protected:Npn \__pdf_backend_compress_objects:n #1
3111 {
3112     \bool_if:nF {#1}
3113     { \__kernel_backend_literal:n { dvipdfmx:config~C~0x40 } }
3114 }

```

(End definition for \\_\_pdf\_backend\_compresslevel:n and \\_\_pdf\_backend\_compress\_objects:n.)

\\_\_pdf\_backend\_version\_major\_gset:n  
\\_\_pdf\_backend\_version\_minor\_gset:n

We start with the assumption that the default is active.

```

3115 \cs_new_protected:Npn \__pdf_backend_version_major_gset:n #1
3116 {
3117     \cs_gset:Npx \__pdf_backend_version_major: { \int_eval:n {#1} }
3118     \__kernel_backend_literal:x { pdf:majorversion~ \__pdf_backend_version_major: }
3119 }
3120 \cs_new_protected:Npn \__pdf_backend_version_minor_gset:n #1
3121 {
3122     \cs_gset:Npx \__pdf_backend_version_minor: { \int_eval:n {#1} }
3123     \__kernel_backend_literal:x { pdf:minorversion~ \__pdf_backend_version_minor: }
3124 }

```

(End definition for \\_\_pdf\_backend\_version\_major\_gset:n and \\_\_pdf\_backend\_version\_minor\_gset:n.)

\\_\_pdf\_backend\_version\_major:  
\\_\_pdf\_backend\_version\_minor:

We start with the assumption that the default is active.

```

3125 \cs_new:Npn \__pdf_backend_version_major: { 1 }
3126 \cs_new:Npn \__pdf_backend_version_minor: { 5 }

```

(End definition for \\_\_pdf\_backend\_version\_major: and \\_\_pdf\_backend\_version\_minor:.)

#### 6.4.5 Marked content

\\_\_pdf\_backend\_bdc:nn  
\\_\_pdf\_backend\_emc:

Simple wrappers. May need refinement: see <https://chat.stackexchange.com/transcript/message/49970158#49970158>.

```

3127 \cs_new_protected:Npn \__pdf_backend_bdc:nn #1#2
3128 { \__kernel_backend_literal_page:n { /#1 ~ #2 ~ BDC } }
3129 \cs_new_protected:Npn \__pdf_backend_emc:
3130 { \__kernel_backend_literal_page:n { EMC } }

```

(End definition for `\_pdf_backend_bdc:nn` and `\_pdf_backend_emc:.`)

3131 `</dvipdfmx | xetex>`

## 6.5 dvisvgm backend

3132 `<*dvisvgm>`

### 6.5.1 Annotations

`\_pdf_backend_annotation:nnnn`

3133 `\cs_new_protected:Npn \_pdf_backend_annotation:nnnn #1#2#3#4 { }`

(End definition for `\_pdf_backend_annotation:nnnn`.)

`\_pdf_backend_annotation_last:`

3134 `\cs_new:Npn \_pdf_backend_annotation_last: { }`

(End definition for `\_pdf_backend_annotation_last:.`)

`\_pdf_backend_link_begin_goto:nnw`

`\_pdf_backend_link_begin_user:nnw`

`\_pdf_backend_link_begin:nnnw`

`\_pdf_backend_link_end:`

3135 `\cs_new_protected:Npn \_pdf_backend_link_begin_goto:nnw #1#2 { }`

3136 `\cs_new_protected:Npn \_pdf_backend_link_begin_user:nnw #1#2 { }`

3137 `\cs_new_protected:Npn \_pdf_backend_link_begin:nnnw #1#2#3 { }`

3138 `\cs_new_protected:Npn \_pdf_backend_link_end: { }`

(End definition for `\_pdf_backend_link_begin_goto:nnw` and others.)

`\_pdf_backend_link_last:`

3139 `\cs_new:Npx \_pdf_backend_link_last: { }`

(End definition for `\_pdf_backend_link_last:.`)

`\_pdf_backend_link_margin:n`

A simple task: pass the data to the primitive.

3140 `\cs_new_protected:Npn \_pdf_backend_link_margin:n #1 { }`

(End definition for `\_pdf_backend_link_margin:n`.)

`\_pdf_backend_destination:nn`

`\_pdf_backend_destination:nnnn`

3141 `\cs_new_protected:Npn \_pdf_backend_destination:nn #1#2 { }`

3142 `\cs_new_protected:Npn \_pdf_backend_destination:nnnn #1#2#3#4 { }`

(End definition for `\_pdf_backend_destination:nn` and `\_pdf_backend_destination:nnnn`.)

### 6.5.2 Catalogue entries

`\_pdf_backend_catalog_gput:nn`

No-op.

`\_pdf_backend_info_gput:nn`

3143 `\cs_new_protected:Npn \_pdf_backend_catalog_gput:nn #1#2 { }`

3144 `\cs_new_protected:Npn \_pdf_backend_info_gput:nn #1#2 { }`

(End definition for `\_pdf_backend_catalog_gput:nn` and `\_pdf_backend_info_gput:nn`.)

### 6.5.3 Objects

All no-ops here.

```

\__pdf_backend_object_new:nn
\__pdf_backend_object_ref:n      3145 \cs_new_protected:Npn \__pdf_backend_object_new:nn #1#2 { }
    \_pdf_backend_object_write:nn 3146 \cs_new:Npn \__pdf_backend_object_ref:n #1 { }
    \_pdf_backend_object_write:nx 3147 \cs_new_protected:Npn \__pdf_backend_object_write:nn #1#2 { }
\__pdf_backend_object_now:nn     3148 \cs_new_protected:Npn \__pdf_backend_object_write:nx #1#2 { }
\__pdf_backend_object_now:nx     3149 \cs_new_protected:Npn \__pdf_backend_object_now:nn #1#2 { }
\__pdf_backend_object_last:      3150 \cs_new_protected:Npn \__pdf_backend_object_now:nx #1#2 { }
    \_pdf_backend_pageobject_ref:n 3151 \cs_new:Npn \__pdf_backend_object_last: { }
    3152 \cs_new:Npn \__pdf_backend_pageobject_ref:n #1 { }

```

(End definition for \\_\_pdf\_backend\_object\_new:nn and others.)

### 6.5.4 Structure

These are all no-ops.

```

\_pdf_backend_compresslevel:n    3153 \cs_new_protected:Npn \__pdf_backend_compresslevel:n #1 { }
\_pdf_backend_compress_objects:n 3154 \cs_new_protected:Npn \__pdf_backend_compress_objects:n #1 { }

```

(End definition for \\_\_pdf\_backend\_compresslevel:n and \\_\_pdf\_backend\_compress\_objects:n.)

Data not available!

```

\_pdf_backend_version_major_gset:n 3155 \cs_new_protected:Npn \__pdf_backend_version_major_gset:n #1 { }
\_pdf_backend_version_minor_gset:n 3156 \cs_new_protected:Npn \__pdf_backend_version_minor_gset:n #1 { }

```

(End definition for \\_\_pdf\_backend\_version\_major\_gset:n and \\_\_pdf\_backend\_version\_minor\_gset:n.)

Data not available!

```

\_pdf_backend_version_major:      3157 \cs_new:Npn \__pdf_backend_version_major: { -1 }
\_pdf_backend_version_minor:      3158 \cs_new:Npn \__pdf_backend_version_minor: { -1 }

```

(End definition for \\_\_pdf\_backend\_version\_major: and \\_\_pdf\_backend\_version\_minor:.)

More no-ops.

```

\__pdf_backend_bdc:nn            3159 \cs_new_protected:Npn \__pdf_backend_bdc:nn #1#2 { }
\__pdf_backend_emc:              3160 \cs_new_protected:Npn \__pdf_backend_emc: { }

```

(End definition for \\_\_pdf\_backend\_bdc:nn and \\_\_pdf\_backend\_emc:.)

3161  $\langle$ /divisvgn $\rangle$

3162  $\langle$ /package $\rangle$

## 7 l3backend-opacity Implementation

```

3163  $\langle$ *package $\rangle$ 
3164  $\langle$ @@=opacity $\rangle$ 

```

Although opacity is not color, it needs to be managed in a somewhat similar way: using a dedicated stack if possible. Depending on the backend, that may not be possible. There is also the need to cover fill/stroke setting as well as more general running opacity. It is easiest to describe the value used in terms of opacity, although commonly this is referred to as transparency.

3165  $\langle$ \*dvips $\rangle$

`\__opacity_backend_select:n
 \__opacity_backend_select_aux:n
 \__opacity_backend_fill:n
\__opacity_backend_stroke:n
 \__opacity_backend:nnn
 \__opacity_backend:xnn`

No stack so set values directly. The need to deal with Distiller and Ghostscript separately means we use a common auxiliary: the two systems require different PostScript for transparency. This is of course not quite as efficient as doing one test for setting all transparency, but it keeps things clearer here. Thanks to Alex Grahn for the detail on testing for GhostScript.

```

3166 \cs_new_protected:Npn \__opacity_backend_select:n #1
3167 {
3168   \exp_args:Nx \__opacity_backend_select_aux:n
3169   { \fp_eval:n { min(max(0,#1),1) } }
3170 }
3171 \cs_new_protected:Npn \__opacity_backend_select_aux:n #1
3172 {
3173   \__opacity_backend:nnn {#1} { fill } { ca }
3174   \__opacity_backend:nnn {#1} { stroke } { CA }
3175 }
3176 \cs_new_protected:Npn \__opacity_backend_fill:n #1
3177 {
3178   \__opacity_backend:xnn
3179   { \fp_eval:n { min(max(0,#1),1) } }
3180   { fill }
3181   { ca }
3182 }
3183 \cs_new_protected:Npn \__opacity_backend_stroke:n #1
3184 {
3185   \__opacity_backend:xnn
3186   { \fp_eval:n { min(max(0,#1),1) } }
3187   { stroke }
3188   { CA }
3189 }
3190 \cs_new_protected:Npn \__opacity_backend:nnn #1#2#3
3191 {
3192   \__kernel_backend_postscript:n
3193   {
3194     product ~ (Ghostscript) ~ search
3195     {
3196       pop ~ pop ~ pop ~
3197       #1 ~ .set #2 constantalpha
3198     }
3199     {
3200       pop ~
3201       mark ~
3202       /#3 ~ #1
3203       /SetTransparency ~
3204       pdfmark
3205     }
3206     ifelse
3207   }
3208 }
3209 \cs_generate_variant:Nn \__opacity_backend:nnn { x }

```

(End definition for `\__opacity_backend_select:n` and others.)

```

3210 </dvips>
3211 <*dvipdfmx | luatex | pdftex | xetex>

```

\c\_opacity\_backend\_stack\_int Set up a stack, where that is applicable.

```

3212 \bool_lazy_and:nnT
3213 { \cs_if_exist_p:N \pdfmanagement_if_active_p: }
3214 { \pdfmanagement_if_active_p:}
3215 {
3216 <*luatex | pdftex>
3217   \__kernel_color_backend_stack_init:Nnn \c__opacity_backend_stack_int
3218   { page ~ direct } { /opacity 1 ~ gs }
3219 </luatex | pdftex>
3220   \pdfmanagement_add:nnn { Page / Resources / ExtGState }
3221   { opacity 1 } { << /ca ~ 1 /CA ~ 1 >> }
3222 }

```

(End definition for \c\_\_opacity\_backend\_stack\_int.)

\l\_\_opacity\_backend\_fill\_tl We use tl here for speed: at the backend, this should be reasonable.

```

\l__opacity_backend_stroke_tl
3223 \tl_new:N \l__opacity_backend_fill_tl
3224 \tl_new:N \l__opacity_backend_stroke_tl

```

(End definition for \l\_\_opacity\_backend\_fill\_tl and \l\_\_opacity\_backend\_stroke\_tl.)

\\_\_opacity\_backend\_select:n Other than the need to evaluate the opacity as an fp, much the same as color.

```

\__opacity_backend_select_aux:n
\__opacity_backend_reset:
3225 \cs_new_protected:Npn \__opacity_backend_select:n #1
3226 {
3227   \exp_args:Nx \__opacity_backend_select_aux:n
3228   { \fp_eval:n { min(max(0,#1),1) } }
3229 }
3230 \cs_new_protected:Npn \__opacity_backend_select_aux:n #1
3231 {
3232   \tl_set:Nn \l__opacity_backend_fill_tl {#1}
3233   \tl_set:Nn \l__opacity_backend_stroke_tl {#1}
3234   \pdfmanagement_add:nnn { Page / Resources / ExtGState }
3235   { opacity #1 }
3236   { << /ca ~ #1 /CA ~ #1 >> }
3237 <*dvipdfmx | xetex>
3238   \__kernel_backend_literal_pdf:n
3239 </dvipdfmx | xetex>
3240 <*luatex | pdftex>
3241   \__kernel_color_backend_stack_push:nn \c__opacity_backend_stack_int
3242 </luatex | pdftex>
3243   { /opacity #1 ~ gs }
3244   \group_insert_after:N \__opacity_backend_reset:
3245 }
3246 \bool_lazy_and:nnF
3247 { \cs_if_exist_p:N \pdfmanagement_if_active_p: }
3248 { \pdfmanagement_if_active_p:}
3249 {
3250   \cs_gset_protected:Npn \__opacity_backend_select_aux:n #1 { }
3251 }
3252 \cs_new_protected:Npn \__opacity_backend_reset:
3253 {
3254 <*dvipdfmx | xetex>
3255   \__kernel_backend_literal_pdf:n
3256   { /opacity1 ~ gs }

```

```

3257 </dvipdfmx | xetex>
3258 <*luatex | pdftex>
3259     \__kernel_color_backend_stack_pop:n \c__opacity_backend_stack_int
3260 </luatex | pdftex>
3261 }

```

(End definition for \\_\_opacity\_backend\_select:n, \\_\_opacity\_backend\_select\_aux:n, and \\_\_opacity\_backend\_reset:.)

\\_\_opacity\_backend\_fill:n For separate fill and stroke, we need to work out if we need to do more work or if we can  
 \\_\_opacity\_backend\_stroke:n stick to a single setting.

```

\__opacity_backend_fillstroke:nn 3262 \cs_new_protected:Npn \__opacity_backend_fill:n #1
\__opacity_backend_fillstroke:xx 3263 {
3264     \__opacity_backend_fill_stroke:xx
3265     { \fp_eval:n { min(max(0,#1),1) } }
3266     \l__opacity_backend_stroke_tl
3267 }
3268 \cs_new_protected:Npn \__opacity_backend_stroke:n #1
3269 {
3270     \__opacity_backend_fill_stroke:xx
3271     \l__opacity_backend_fill_tl
3272     { \fp_eval:n { min(max(0,#1),1) } }
3273 }
3274 \cs_new_protected:Npn \__opacity_backend_fill_stroke:nn #1#2
3275 {
3276     \str_if_eq:nnTF {#1} {#2}
3277     { \__opacity_backend_select_aux:n {#1} }
3278     {
3279         \tl_set:Nn \l__opacity_backend_fill_tl {#1}
3280         \tl_set:Nn \l__opacity_backend_stroke_tl {#2}
3281         \pdfmanagement_add:nnn { Page / Resources / ExtGState }
3282         { opacity.fill #1 }
3283         { << /ca ~ #1 >> }
3284         \pdfmanagement_add:nnn { Page / Resources / ExtGState }
3285         { opacity.stroke #1 }
3286         { << /CA ~ #2 >> }
3287 <*dvipdfmx | xetex>
3288     \__kernel_backend_literal_pdf:n
3289 </dvipdfmx | xetex>
3290 <*luatex | pdftex>
3291     \__kernel_color_backend_stack_push:nn \c__opacity_backend_stack_int
3292 </luatex | pdftex>
3293     { /opacity.fill #1 ~ gs /opacity.stroke #2 ~ gs }
3294     \group_insert_after:N \__opacity_backend_reset:
3295 }
3296 }
3297 \cs_generate_variant:Nn \__opacity_backend_fill_stroke:nn { xx }

```

(End definition for \\_\_opacity\_backend\_fill:n, \\_\_opacity\_backend\_stroke:n, and \\_\_opacity\_backend\_fillstroke:nn.)

```

3298 </dvipdfmx | luatex | pdftex | xetex>
3299 <*dvisvgm>

```

<code> \__opacity_backend_select:n \__opacity_backend_fill:n \__opacity_backend_stroke:n \__opacity_backend:nn </code>	<p>Once again, we use a scope here. There is a general opacity function for SVG, but that is of course not set up using the stack.</p> <pre> 3300 \cs_new_protected:Npn \__opacity_backend_select:n #1 3301 { \__opacity_backend:nn {#1} { } } 3302 \cs_new_protected:Npn \__opacity_backend_fill:n #1 3303 { \__opacity_backend:nn {#1} { fill- } } 3304 \cs_new_protected:Npn \__opacity_backend_stroke:n #1 3305 { \__opacity_backend:nn { {#1} } { stroke- } } 3306 \cs_new_protected:Npn \__opacity_backend:nn #1#2 3307 { \__kernel_backend_scope:x { #2 opacity = " \fp_eval:n { min(max(0,#1),1) } " } } </pre> <p><i>(End definition for \__opacity_backend_select:n and others.)</i></p> <pre> 3308 &lt;/dvisvgm&gt; 3309 &lt;/package&gt; </pre>
--	---

## 8 l3backend-header Implementation

<code> 3310 &lt;*dvips &amp; header&gt; </code>	<p>Empty definition for color at the top level.</p> <pre> 3311 /color.sc { } def </pre> <p><i>(End definition for color.sc. This function is documented on page ??.)</i></p>
<p>TeXcolorseparation separation</p>	<p>Support for separation/spot colors: this strange naming is so things work with the color stack.</p> <pre> 3312 TeXDict begin 3313 /TeXcolorseparation { setcolor } def 3314 end </pre> <p><i>(End definition for TeXcolorseparation and separation. These functions are documented on page ??.)</i></p>
<p>pdf.globaldict</p>	<p>A small global dictionary for backend use.</p> <pre> 3315 true setglobal 3316 /pdf.globaldict 4 dict def 3317 false setglobal </pre> <p><i>(End definition for pdf.globaldict. This function is documented on page ??.)</i></p>
<p>pdf.cvs pdf.dvi.pt pdf.pt.dvi pdf.rect.ht</p>	<p>Small utilities for PostScript manipulations. Conversion to DVI dimensions is done here to allow for Resolution. The total height of a rectangle (an array) needs a little maths, in contrast to simply extracting a value.</p> <pre> 3318 /pdf.cvs { 65534 string cvs } def 3319 /pdf.dvi.pt { 72.27 mul Resolution div } def 3320 /pdf.pt.dvi { 72.27 div Resolution mul } def 3321 /pdf.rect.ht { dup 1 get neg exch 3 get add } def </pre> <p><i>(End definition for pdf.cvs and others. These functions are documented on page ??.)</i></p>
<p>pdf.linkmargin pdf.linkdp.pad pdf.linkht.pad</p>	<p>Settings which are defined up-front in SDict.</p> <pre> 3322 /pdf.linkmargin { 1 pdf.pt.dvi } def 3323 /pdf.linkdp.pad { 0 } def 3324 /pdf.linkht.pad { 0 } def </pre>

(End definition for `pdf.linkmargin`, `pdf.linkdp.pad`, and `pdf.linkht.pad`. These functions are documented on page ??.)

`pdf.rect` Functions for marking the limits of an annotation/link, plus drawing the border. We  
`pdf.save.ll` separate links for generic annotations to support adding a margin and setting a minimal  
`pdf.save.ur` size.

```
pdf.save.linkll 3325 /pdf.rect
pdf.save.linkur 3326 { /Rect [ pdf.llx pdf.lly pdf.urx pdf.ury ] } def
pdf.llx         3327 /pdf.save.ll
pdf.lly         3328 {
pdf.urx         3329     currentpoint
pdf.ury         3330     /pdf.lly exch def
                3331     /pdf.llx exch def
                3332 }
                3333 def
pdf.save.ur     3334 /pdf.save.ur
                3335 {
                3336     currentpoint
                3337     /pdf.ury exch def
                3338     /pdf.urx exch def
                3339 }
                3340 def
pdf.save.linkll 3341 /pdf.save.linkll
                3342 {
                3343     currentpoint
                3344     pdf.linkmargin add
                3345     pdf.linkdp.pad add
                3346     /pdf.lly exch def
                3347     pdf.linkmargin sub
                3348     /pdf.llx exch def
                3349 }
                3350 def
pdf.save.linkur 3351 /pdf.save.linkur
                3352 {
                3353     currentpoint
                3354     pdf.linkmargin sub
                3355     pdf.linkht.pad sub
                3356     /pdf.ury exch def
                3357     pdf.linkmargin add
                3358     /pdf.urx exch def
                3359 }
                3360 def
```

(End definition for `pdf.rect` and others. These functions are documented on page ??.)

`pdf.dest.anchor` For finding the anchor point of a destination link. We make the use case a separate  
`pdf.dest.x` function as it comes up a lot, and as this makes it easier to adjust if we need additional  
`pdf.dest.y` effects. We also need a more complex approach to convert a co-ordinate pair correctly  
`pdf.dest.point` when defining a rectangle: this can otherwise be out when using a landscape page.  
`pdf.dest2device` (Thanks to Alexander Grahn for the approach here.)

```
pdf.dev.x 3361 /pdf.dest.anchor
pdf.dev.y 3362 {
pdf.tmpa 3363     currentpoint exch
pdf.tmpb 3364     pdf.dvi.pt 72 add
pdf.tmpc
pdf.tmpd
```



```

3365     /pdf.dest.x exch def
3366     pdf.dvi.pt
3367     vsize 72 sub exch sub
3368     /pdf.dest.y exch def
3369   }
3370   def
3371 /pdf.dest.point
3372   { pdf.dest.x pdf.dest.y } def
3373 /pdf.dest2device
3374   {
3375     /pdf.dest.y exch def
3376     /pdf.dest.x exch def
3377     matrix currentmatrix
3378     matrix defaultmatrix
3379     matrix invertmatrix
3380     matrix concatmatrix
3381     cvx exec
3382     /pdf.dev.y exch def
3383     /pdf.dev.x exch def
3384     /pdf.tmpd exch def
3385     /pdf.tmpc exch def
3386     /pdf.tmpb exch def
3387     /pdf.tmpa exch def
3388     pdf.dest.x pdf.tmpa mul
3389     pdf.dest.y pdf.tmpc mul add
3390     pdf.dev.x add
3391     pdf.dest.x pdf.tmpb mul
3392     pdf.dest.y pdf.tmpd mul add
3393     pdf.dev.y add
3394   }
3395   def

```

(End definition for pdf.dest.anchor and others. These functions are documented on page ??.)

pdf.bordertracking	To know where a breakable link can go, we need to track the boundary rectangle. That
pdf.bordertracking.begin	can be done by hooking into a and x operations: those names have to be retained. The
pdf.bordertracking.end	boundary is stored at the end of the operation. Special effort is needed at the start and
pdf.leftboundary	end of pages (or rather galleys), such that everything works properly.
pdf.rightboundary	
pdf.brokenlink.rect	3396 /pdf.bordertracking false def
pdf.brokenlink.skip	3397 /pdf.bordertracking.begin
pdf.brokenlink.dict	3398 {
pdf.bordertracking.endpage	3399   SDict /pdf.bordertracking true put
pdf.bordertracking.continue	3400   SDict /pdf.leftboundary undef
pdf.originx	3401   SDict /pdf.rightboundary undef
pdf.originy	3402   /a where
	3403     {
	3404       /a
	3405       {
	3406         currentpoint pop
	3407         SDict /pdf.rightboundary known dup
	3408         {
	3409           SDict /pdf.rightboundary get 2 index lt
	3410           { not }
	3411         if

```

3412     }
3413     if
3414     { pop }
3415     { SDict exch /pdf.rightboundary exch put }
3416     ifelse
3417     moveto
3418     currentpoint pop
3419     SDict /pdf.leftboundary known dup
3420     {
3421         SDict /pdf.leftboundary get 2 index gt
3422         { not }
3423         if
3424     }
3425     if
3426     { pop }
3427     { SDict exch /pdf.leftboundary exch put }
3428     ifelse
3429 }
3430 put
3431 }
3432 if
3433 }
3434 def
3435 /pdf.bordertracking.end
3436 {
3437     /a where { /a { moveto } put } if
3438     /x where { /x { 0 exch rmoveto } put } if
3439     SDict /pdf.leftboundary known
3440     { pdf.outerbox 0 pdf.leftboundary put }
3441     if
3442     SDict /pdf.rightboundary known
3443     { pdf.outerbox 2 pdf.rightboundary put }
3444     if
3445     SDict /pdf.bordertracking false put
3446 }
3447 def
3448 /pdf.bordertracking.endpage
3449 {
3450 pdf.bordertracking
3451 {
3452     pdf.bordertracking.end
3453     true setglobal
3454     pdf.globaldict
3455     /pdf.brokenlink.rect [ pdf.outerbox aload pop ] put
3456     pdf.globaldict
3457     /pdf.brokenlink.skip pdf.baselineskip put
3458     pdf.globaldict
3459     /pdf.brokenlink.dict
3460     pdf.link.dict pdf.cvs put
3461     false setglobal
3462     mark pdf.link.dict cvx exec /Rect
3463     [
3464         pdf.llx
3465         pdf.lly

```

```

3466         pdf.outerbox 2 get pdf.linkmargin add
3467         currentpoint exch pop
3468         pdf.outerbox pdf.rect.ht sub pdf.linkmargin sub
3469     ]
3470     /ANN pdf.pdfmark
3471 }
3472 if
3473 }
3474 def
3475 /pdf.bordertracking.continue
3476 {
3477     /pdf.link.dict pdf.globaldict
3478     /pdf.brokenlink.dict get def
3479     /pdf.outerbox pdf.globaldict
3480     /pdf.brokenlink.rect get def
3481     /pdf.baselineskip pdf.globaldict
3482     /pdf.brokenlink.skip get def
3483     pdf.globaldict dup dup
3484     /pdf.brokenlink.dict undef
3485     /pdf.brokenlink.skip undef
3486     /pdf.brokenlink.rect undef
3487     currentpoint
3488     /pdf.originy exch def
3489     /pdf.originx exch def
3490     /a where
3491     {
3492         /a
3493         {
3494             moveto
3495             SDict
3496             begin
3497                 currentpoint pdf.originy ne exch
3498                 pdf.originx ne or
3499                 {
3500                     pdf.save.linkll
3501                     /pdf.lly
3502                     pdf.lly pdf.outerbox 1 get sub def
3503                     pdf.bordertracking.begin
3504                 }
3505                 if
3506                 end
3507             }
3508             put
3509         }
3510     if
3511     /x where
3512     {
3513         /x
3514         {
3515             0 exch rmoveto
3516             SDict
3517             begin
3518                 currentpoint
3519                 pdf.originy ne exch pdf.originx ne or

```

```

3520         {
3521             pdf.save.linkll
3522             /pdf.lly
3523             pdf.lly pdf.outerbox 1 get sub def
3524             pdf.bordertracking.begin
3525         }
3526     if
3527     end
3528 }
3529 put
3530 }
3531 if
3532 }
3533 def

```

(End definition for pdf.bordertracking and others. These functions are documented on page ??.)

<p>pdf.breaklink</p> <p>pdf.breaklink.write</p> <p>pdf.count</p> <p>pdf.currentrect</p>	<p>Dealing with link breaking itself has multiple stage. The first step is to find the Rect entry in the dictionary, looping over key-value pairs. The first line is handled first, adjusting the rectangle to stay inside the text area. The second phase is a loop over the height of the bulk of the link area, done on the basis of a number of baselines. Finally, the end of the link area is tidied up, again from the boundary of the text area.</p>
---	--

```

3534 /pdf.breaklink
3535 {
3536     pop
3537     counttomark 2 mod 0 eq
3538     {
3539         counttomark /pdf.count exch def
3540         {
3541             pdf.count 0 eq { exit } if
3542             counttomark 2 roll
3543             1 index /Rect eq
3544             {
3545                 dup 4 array copy
3546                 dup dup
3547                 1 get
3548                 pdf.outerbox pdf.rect.ht
3549                 pdf.linkmargin 2 mul add sub
3550                 3 exch put
3551                 dup
3552                 pdf.outerbox 2 get
3553                 pdf.linkmargin add
3554                 2 exch put
3555                 dup dup
3556                 3 get
3557                 pdf.outerbox pdf.rect.ht
3558                 pdf.linkmargin 2 mul add add
3559                 1 exch put
3560             } /pdf.currentrect exch def
3561             pdf.breaklink.write
3562             {
3563                 pdf.currentrect
3564                 dup
3565                 pdf.outerbox 0 get

```

```

3566         pdf.linkmargin sub
3567         0 exch put
3568     dup
3569         pdf.outerbox 2 get
3570         pdf.linkmargin add
3571         2 exch put
3572     dup dup
3573         1 get
3574         pdf.baselineskip add
3575         1 exch put
3576     dup dup
3577         3 get
3578         pdf.baselineskip add
3579         3 exch put
3580     /pdf.currentrect exch def
3581     pdf.breaklink.write
3582 }
3583     1 index 3 get
3584     pdf.linkmargin 2 mul add
3585     pdf.outerbox pdf.rect.ht add
3586     2 index 1 get sub
3587     pdf.baselineskip div round cvi 1 sub
3588     exch
3589     repeat
3590     pdf.currentrect
3591     dup
3592         pdf.outerbox 0 get
3593         pdf.linkmargin sub
3594         0 exch put
3595     dup dup
3596         1 get
3597         pdf.baselineskip add
3598         1 exch put
3599     dup dup
3600         3 get
3601         pdf.baselineskip add
3602         3 exch put
3603     dup 2 index 2 get 2 exch put
3604     /pdf.currentrect exch def
3605     pdf.breaklink.write
3606     SDict /pdf.pdfmark.good false put
3607     exit
3608 }
3609 { pdf.count 2 sub /pdf.count exch def }
3610 ifelse
3611 }
3612 loop
3613 }
3614 if
3615 /ANN
3616 }
3617 def
3618 /pdf.breaklink.write
3619 {

```

```

3620     counttomark 1 sub
3621     index /_objdef eq
3622     {
3623         counttomark -2 roll
3624         dup wcheck
3625         {
3626             readonly
3627             counttomark 2 roll
3628         }
3629         { pop pop }
3630         ifelse
3631     }
3632     if
3633     counttomark 1 add copy
3634     pop pdf.currentrect
3635     /ANN pdfmark
3636 }
3637 def

```

(End definition for pdf.breaklink and others. These functions are documented on page ??.)

pdf.pdfmark The business end of breaking links starts by hooking into pdfmarks. Unlike hypdvips, pdf.pdfmark.good we avoid altering any links we have not created by using a copy of the core pdfmarks pdf.outerbox function. Only mark types which are known are altered. At present, this is purely ANN pdf.baselineskip marks, which are measured relative to the size of the baseline skip. If they are more than pdf.pdfmark.dict one apparent line high, breaking is applied.

```

3638 /pdf.pdfmark
3639 {
3640     SDict /pdf.pdfmark.good true put
3641     dup /ANN eq
3642     {
3643         pdf.pdfmark.store
3644         pdf.pdfmark.dict
3645         begin
3646             Subtype /Link eq
3647             currentdict /Rect known and
3648             SDict /pdf.outerbox known and
3649             SDict /pdf.baselineskip known and
3650             {
3651                 Rect 3 get
3652                 pdf.linkmargin 2 mul add
3653                 pdf.outerbox pdf.rect.ht add
3654                 Rect 1 get sub
3655                 pdf.baselineskip div round cvi 0 gt
3656                 { pdf.breaklink }
3657                 if
3658             }
3659             if
3660             end
3661             SDict /pdf.outerbox undef
3662             SDict /pdf.baselineskip undef
3663             currentdict /pdf.pdfmark.dict undef
3664         }
3665     if

```

```

3666     pdf.pdfmark.good
3667     { pdfmark }
3668     { cleartomark }
3669     ifelse
3670   }
3671   def
3672 /pdf.pdfmark.store
3673 {
3674   /pdf.pdfmark.dict 65534 dict def
3675   counttomark 1 add copy
3676   pop
3677   {
3678     dup mark eq
3679     {
3680       pop
3681       exit
3682     }
3683     {
3684       pdf.pdfmark.dict
3685       begin def end
3686     }
3687     ifelse
3688   }
3689   loop
3690 }
3691 def

```

*(End definition for pdf.pdfmark and others. These functions are documented on page ??.)*

```

3692 </dvips & header>

```

# Index

The italic numbers denote the pages where the corresponding entry is described, numbers underlined point to the definition, all others indicate the places where it is used.

<b>A</b>	
<code>\AtBeginDvi</code> . . . . .	57
<b>B</b>	
bool commands:	
<code>\bool_gset_false:N</code> . . . . .	1177, 1196, 1219, 1241, 1257, 1358, 1597, 1633, 2382, 2428
<code>\bool_gset_true:N</code> . . . . .	1175, 1244, 1356, 1612, 2375, 2381
<code>\bool_if:NTF</code> . . . . .	67, 610, 1187, 1191, 1207, 1210, 1214, 1225, 1232, 1236, 1248, 1252, 1369, 1374, 1379, 1571, 1616, 1755, 1799, 1938, 1980, 2370, 2385, 2390, 2395
<code>\bool_if:nTF</code> . . . . .	2604, 2870, 3112
<code>\bool_lazy_and:nnTF</code> . . . . .	823, 2097, 3212, 3246
<code>\bool_lazy_or:nnTF</code> . . . . .	1791, 1973
<code>\bool_new:N</code> . . . . .	1178, 1245, 1359, 1613, 2355, 2356
<code>\bool_set_false:N</code> . . . . .	1766, 1902, 2004, 2168
box commands:	
<code>\box_dp:N</code> . . . . .	208, 210, 258, 260, 315, 317, 364, 366, 368, 370, 2407, 2440, 2441, 2466
<code>\box_ht:N</code> . . . . .	210, 260, 317, 368, 370, 1811, 2045, 2412, 2451, 2452, 2468
<code>\box_if_empty:NTF</code> . . . . .	2501
<code>\box_move_down:nn</code> . . . . .	2329, 2407
<code>\box_move_up:nn</code> . . . . .	2189, 2331, 2412
<code>\box_new:N</code> . . . . .	2214, 2319, 2320
<code>\box_set_dp:Nn</code> . . . . .	1696
<code>\box_set_ht:Nn</code> . . . . .	1695
<code>\box_set_wd:Nn</code> . . . . .	272, 1694
<code>\box_use:N</code> . . . . .	215, 233, 247, 263, 290, 304, 320, 336, 348, 399, 413, 432, 1309, 1504, 1697, 2360
<code>\box_wd:N</code> . . . . .	209, 217, 259, 265, 316, 322, 365, 367, 1810, 2044
box internal commands:	
<code>\__box_backend_clip:N</code> . . . . .	197, 197, 252, 252, 309, 309, 353, 353
<code>\l_box_backend_cos_fp</code> . . . . .	267
<code>\__box_backend_rotate:Nn</code> . . . . .	219, 219, 267, 267, 324, 324, 403, 403
<code>\__box_backend_rotate_aux:Nn</code> . . . . .	219, 220, 221, 267, 268, 269, 324, 325, 326
<code>\__box_backend_scale:Nnn</code> . . . . .	236, 236, 295, 295, 339, 339, 416, 416
<code>\l__box_backend_sin_fp</code> . . . . .	267
<b>C</b>	
clist commands:	
<code>\clist_map_function:nN</code> . . . . .	1265, 1389, 1640
color internal commands:	
<code>\__color_backend:nnn</code> . . . . .	1059, 1066, 1081, 1089, 1095
<code>\__color_backend_cmyk:w</code> . . . . .	1060
<code>\g_color_backend_colorant_prop</code> . . . . .	576, 595, 598, 618, 859
<code>\__color_backend_devicen_colorants:n</code> . . . . .	577, 577, 779, 917
<code>\__color_backend_devicen_colorants:w</code> . . . . .	577, 585, 592, 600
<code>\__color_backend_devicen_init:nnn</code> . . . . .	766, 766, 884, 884, 1116, 1116
<code>\__color_backend_devicen_init:w</code> . . . . .	884, 893, 922, 926
<code>\__color_backend_fill:n</code> . . . . .	963, 963, 965, 966, 967, 989, 990, 992, 994, 995, 1014, 1023, 1024, 1026, 1028, 1029, 1040, 1049, 1050, 1052, 1054, 1055
<code>\__color_backend_fill_cmyk:n</code> . . . . .	963, 965, 989, 989, 1023, 1023, 1049, 1049
<code>\__color_backend_fill_devicen:nn</code> . . . . .	973, 983, 1013, 1017, 1039, 1043, 1110, 1112
<code>\__color_backend_fill_gray:n</code> . . . . .	963, 966, 989, 991, 1023, 1025, 1049, 1051
<code>\__color_backend_fill_reset:</code> . . . . .	985, 985, 1019, 1019, 1045, 1045, 1114, 1114
<code>\__color_backend_fill_rgb:n</code> . . . . .	963, 967, 989, 993, 1023, 1027, 1049, 1053
<code>\__color_backend_fill_separation:nn</code> . . . . .	973, 973, 983, 1013, 1013, 1017, 1039, 1039, 1043, 1110, 1110, 1112
<code>\l_color_backend_fill_tl</code> . . . . .	541, 551, 997, 1011
<code>\__color_backend_iccbased_device:nnn</code> . . . . .	946, 946



```

\__color_backend_iccbased_-
  init:nnn ..... 785, 785, 928, 928, 1116, 1117
\__color_backend_init_resource:n
  ..... 820, 820, 849, 920, 944, 959
\__color_backend_pickup:N .....
  ..... 440, 440, 457, 457
\__color_backend_pickup:w .....
  ..... 440, 449, 453, 457, 459, 462
\__color_backend_reset: .....
  ..... 522, 537, 543, 555, 559,
  564, 985, 986, 1019, 1020, 1045, 1114
\__color_backend_rgb:w ..... 1083
\__color_backend_select:n .....
  ..... 522, 523, 525, 527,
  529, 530, 559, 559, 561, 562, 563, 605
\__color_backend_select:nn .....
  ..... 543, 544, 546, 548, 549, 816
\__color_backend_select_cmyk:n ..
  ..... 522, 522, 543, 543, 559, 561
\__color_backend_select_devicen:nn
  ..... 604, 606, 788, 789, 810, 818
\__color_backend_select_gray:n ..
  .... 522, 524, 543, 545, 559, 562, 569
\__color_backend_select_iccbased:nn
  ..... 607, 607, 792, 792, 810, 819
\__color_backend_select_named:n .
  ..... 522, 526, 566, 566
\__color_backend_select_rgb:n ...
  ..... 522, 528, 543, 547, 559, 563
\__color_backend_select_separation:nn
  ..... 604, 604, 606,
  788, 788, 789, 810, 811, 815, 818, 819
\__color_backend_separation_-
  init:n ..... 608, 689, 702
\__color_backend_separation_-
  init:nn ..... 837, 847, 851
\__color_backend_separation_-
  init:nnn ..... 608, 643, 664
\__color_backend_separation_-
  init:nnnn ..... 608, 666, 678
\__color_backend_separation_-
  init:nnnnn ..... 608,
  608, 629, 722, 790, 790, 837, 837, 877
\__color_backend_separation_-
  init:nw ..... 608, 693, 704, 718
\__color_backend_separation_-
  init:w ..... 608, 680, 695, 700
\__color_backend_separation_-
  init_/DeviceCMYK:nnn ..... 608
\__color_backend_separation_-
  init_/DeviceGray:nnn ..... 608
\__color_backend_separation_-
  init_/DeviceRGB:nnn ..... 608

\__color_backend_separation_-
  init_aux:nnnnnn ..... 608, 614, 630
\__color_backend_separation_-
  init_CIELAB:nnn .....
  ..... 608, 720, 790, 837, 862
\__color_backend_separation_-
  init_CIELAB:nnnnnn ..... 791
\__color_backend_separation_-
  init_count:n ..... 608, 667, 670
\__color_backend_separation_-
  init_count:w ... 608, 671, 672, 676
\__color_backend_separation_-
  init_Device:Nn .....
  ..... 608, 652, 654, 656, 657
\l__color_backend_stack_int ....
  ..... 483, 553, 556, 998, 1010
\__color_backend_stroke:n .....
  ..... 963, 968, 970, 971,
  972, 989, 1002, 1004, 1006, 1007, 1016
\__color_backend_stroke_cmyk:n ..
  ..... 963,
  970, 989, 1001, 1023, 1033, 1059, 1059
\__color_backend_stroke_cmyk:w ..
  ..... 1059, 1061
\__color_backend_stroke_devicen:nn
  ..... 973,
  984, 1013, 1018, 1039, 1044, 1110, 1113
\__color_backend_stroke_gray:n ..
  ..... 963,
  971, 989, 1003, 1023, 1035, 1059, 1072
\__color_backend_stroke_gray_-
  aux:n ..... 1059, 1076, 1080
\__color_backend_stroke_reset: ..
  ..... 985,
  986, 1019, 1020, 1045, 1046, 1114, 1115
\__color_backend_stroke_rgb:n ...
  ..... 963,
  972, 989, 1005, 1023, 1037, 1059, 1082
\__color_backend_stroke_rgb:w ...
  ..... 1059, 1084
\__color_backend_stroke_separation:nn
  .. 973, 978, 984, 1013, 1015, 1018,
  1039, 1041, 1044, 1110, 1111, 1113
\l__color_backend_stroke_tl ....
  ..... 541, 552, 999, 1009
\g__color_model_int 615, 624, 772,
  800, 849, 855, 856, 910, 911, 920, 944
\c__color_model_range_CIELAB_tl .
  ..... 727, 762, 873, 880
color.sc ..... 522, 3311
cs commands:
  \cs_generate_variant:Nn .....
  49, 63, 66, 99, 138, 143, 154, 185,
  191, 629, 1124, 1319, 1513, 1952,

```

2015, 2035, 2218, 2255, 2314, 2806,	1360, 1362, 1364, 1366, 1371, 1376,
2834, 2944, 2966, 3001, 3209, 3297	1381, 1383, 1396, 1401, 1403, 1405,
\cs_gset:Npx .. 2616, 2620, 3117, 3122	1407, 1409, 1411, 1413, 1415, 1426,
\cs_gset_protected:Npn ..... 3250	1451, 1463, 1475, 1487, 1494, 1516,
\cs_if_exist:NTF ..... 27, 50, 1707, 2497, 2895, 2921	1522, 1527, 1532, 1543, 1553, 1563,
\cs_if_exist_p:N ..... 824, 3213, 3247	1565, 1567, 1569, 1600, 1602, 1607,
\cs_if_exist_use:NTF ..... 38, 642	1609, 1611, 1614, 1635, 1646, 1659,
\cs_new:Npn ..... 592, 651, 653, 655, 657, 664,	1661, 1663, 1665, 1667, 1669, 1671,
670, 672, 678, 695, 702, 704, 922,	1673, 1675, 1683, 1705, 1724, 1747,
1270, 1394, 1644, 1813, 2048, 2206,	1763, 1775, 1780, 1788, 1814, 1827,
2233, 2315, 2317, 2350, 2522, 2622,	1845, 1855, 1871, 1890, 1899, 1907,
2623, 2776, 2807, 2808, 2926, 2959,	1919, 1925, 1928, 1943, 1953, 1992,
3002, 3004, 3020, 3044, 3125, 3126,	2001, 2007, 2013, 2016, 2023, 2036,
3134, 3146, 3151, 3152, 3157, 3158	2041, 2049, 2056, 2073, 2107, 2138,
\cs_new:Npx ..... 577, 2643, 2678, 2835, 2846, 2913, 3139	2139, 2141, 2143, 2145, 2151, 2157,
\cs_new_eq:NN ..... 46, 57, 59, 561,	2165, 2171, 2174, 2176, 2187, 2216,
562, 563, 606, 789, 818, 819, 965,	2219, 2221, 2225, 2235, 2256, 2261,
966, 967, 970, 971, 972, 983, 984,	2266, 2271, 2281, 2286, 2294, 2322,
985, 986, 1017, 1018, 1019, 1020,	2327, 2359, 2361, 2366, 2368, 2373,
1043, 1044, 1045, 1112, 1113, 1114,	2388, 2393, 2430, 2459, 2478, 2487,
1123, 1318, 1324, 1325, 1512, 1514,	2524, 2531, 2557, 2562, 2590, 2602,
1515, 1521, 1721, 1722, 1735, 1737,	2614, 2618, 2624, 2626, 2630, 2654,
1761, 1762, 1819, 1820, 1821, 1844,	2656, 2658, 2669, 2689, 2699, 2722,
1869, 1886, 1887, 1896, 1897, 1898,	2736, 2746, 2757, 2778, 2809, 2857,
1918, 1921, 1922, 1923, 1988, 1998,	2868, 2874, 2902, 2936, 2938, 2945,
1999, 2000, 2154, 2155, 2163, 2164,	2947, 2951, 2961, 2967, 2972, 2977,
2173, 2203, 2204, 2205, 2209, 2360	2982, 2984, 2986, 2994, 3007, 3023,
\cs_new_protected:Npn ..... 47, 54, 61, 64, 72, 78, 83, 85, 89,	3025, 3042, 3046, 3048, 3070, 3075,
100, 110, 119, 128, 141, 144, 146,	3108, 3110, 3115, 3120, 3127, 3129,
148, 152, 157, 166, 176, 186, 197,	3133, 3135, 3136, 3137, 3138, 3140,
219, 221, 236, 252, 267, 269, 295,	3141, 3142, 3143, 3144, 3145, 3147,
309, 324, 326, 339, 353, 403, 416,	3148, 3149, 3150, 3153, 3154, 3155,
440, 453, 457, 462, 486, 500, 510,	3156, 3159, 3160, 3166, 3171, 3176,
522, 524, 526, 528, 530, 537, 543,	3183, 3190, 3225, 3230, 3252, 3262,
545, 547, 549, 555, 559, 564, 566,	3268, 3274, 3300, 3302, 3304, 3306
604, 607, 630, 720, 766, 785, 788,	\cs_new_protected:Npx ..... 608, 1095, 2885, 2942, 3027
790, 791, 792, 811, 815, 820, 837,	\cs_set_eq:NN ..... 2518, 2519
851, 862, 884, 928, 946, 963, 968,	\cs_set_protected:Npn ..... 2111
973, 978, 989, 991, 993, 995, 1001,	
1003, 1005, 1007, 1013, 1015, 1023,	
1025, 1027, 1029, 1033, 1035, 1037,	
1039, 1041, 1046, 1049, 1051, 1053,	
1055, 1059, 1061, 1072, 1080, 1082,	
1084, 1110, 1111, 1115, 1116, 1117,	
1125, 1130, 1135, 1137, 1139, 1147,	
1155, 1164, 1174, 1176, 1179, 1181,	
1198, 1203, 1221, 1243, 1246, 1259,	
1272, 1277, 1279, 1281, 1283, 1285,	
1287, 1289, 1291, 1296, 1320, 1322,	
1326, 1331, 1336, 1346, 1355, 1357,	

  

D

dim commands:

\dim_compare:nNnTF ..... 2087, 2092
\dim_compare_p:nNn ..... 2098, 2099
\dim_eval:n ..... 2325, 2560,
2638, 2639, 2640, 2697, 2732, 2733,
2734, 3014, 3015, 3016, 3047, 3073
\dim_max:nn ..... 2438, 2449
\dim_set:Nn ..... 1810, 1811, 2044, 2045, 2083, 2084
\dim_set_eq:NN ..... 2149
\dim_to_decimal:n .. 364, 365, 366,
367, 368, 370, 1525, 1530, 1536,
1537, 1538, 1539, 1548, 1549, 1550,

1641, 1660, 2196, 2197, 2436, 2447,  
 2465, 2466, 2467, 2468, 2472, 2528  
 \dim\_to\_decimal\_in\_bp:n .....  
 .... 208, 209, 210, 258, 259, 260,  
 315, 316, 317, 1143, 1144, 1151,  
 1152, 1159, 1160, 1168, 1169, 1170,  
 1267, 1271, 1275, 1329, 1334, 1340,  
 1341, 1342, 1350, 1351, 1391, 1395,  
 1399, 1645, 1729, 1730, 1731, 1732,  
 1912, 1913, 1914, 1915, 1967, 1968,  
 1969, 1970, 2181, 2182, 2183, 2184  
 \dim\_zero:N ..... 2081, 2082  
 \c\_max\_dim .....  
 .. 2083, 2084, 2087, 2092, 2098, 2099  
 draw internal commands:  
 \\_\_draw\_align\_currentpoint... .. 35  
 \\_\_draw\_backend\_add\_to\_path:n ...  
 ..... 1522,  
 1524, 1529, 1534, 1545, 1553, 1568  
 \\_\_draw\_backend\_begin: .....  
 .. 1125, 1125, 1320, 1320, 1516, 1516  
 \\_\_draw\_backend\_box\_use:Nnnnn ...  
 31, 1296, 1296, 1494, 1494, 1683, 1683  
 \\_\_draw\_backend\_cap\_but: .....  
 .. 1259, 1279, 1383, 1403, 1635, 1663  
 \\_\_draw\_backend\_cap\_rectangle: ..  
 .. 1259, 1283, 1383, 1407, 1635, 1667  
 \\_\_draw\_backend\_cap\_round: .....  
 .. 1259, 1281, 1383, 1405, 1635, 1665  
 \\_\_draw\_backend\_clip: .....  
 .. 1179, 1243, 1360, 1376, 1567, 1611  
 \\_\_draw\_backend\_closepath: .....  
 ..... 1179, 1179,  
 1200, 1360, 1360, 1567, 1567, 1604  
 \\_\_draw\_backend\_closestroke: ...  
 .. 1179, 1198, 1360, 1364, 1567, 1602  
 \\_\_draw\_backend\_cm:nnnn .....  
 .... 1291, 1291, 1304, 1305, 1306,  
 1415, 1415, 1498, 1675, 1675, 1686  
 \\_\_draw\_backend\_cm\_aux:nnnn .....  
 ..... 1415, 1422, 1426  
 \\_\_draw\_backend\_cm\_decompose:nnnnN  
 ..... 1421, 1450, 1451  
 \\_\_draw\_backend\_cm\_decompose\_-  
 auxi:nnnnN ..... 1450, 1455, 1463  
 \\_\_draw\_backend\_cm\_decompose\_-  
 auxii:nnnnN ..... 1450, 1467, 1475  
 \\_\_draw\_backend\_cm\_decompose\_-  
 auxiii:nnnnN ..... 1450, 1479, 1487  
 \\_\_draw\_backend\_curveto:nnnnnn ..  
 .. 1139, 1164, 1326, 1336, 1522, 1543  
 \\_\_draw\_backend\_dash:n .....  
 ..... 1259, 1265, 1270,  
 1383, 1389, 1394, 1635, 1640, 1644  
 \\_\_draw\_backend\_dash\_aux:nn .....  
 ..... 1635, 1639, 1646  
 \\_\_draw\_backend\_dash\_pattern:nn ..  
 .. 1259, 1259, 1383, 1383, 1635, 1635  
 \\_\_draw\_backend\_discardpath: ...  
 .. 1179, 1246, 1360, 1381, 1567, 1614  
 \\_\_draw\_backend\_end: .....  
 .. 1125, 1130, 1320, 1322, 1516, 1521  
 \\_\_draw\_backend\_evenodd\_rule: ...  
 .. 1174, 1174, 1355, 1355, 1563, 1563  
 \\_\_draw\_backend\_fill: .....  
 .. 1179, 1203, 1360, 1366, 1567, 1607  
 \\_\_draw\_backend\_fillstroke: .....  
 .. 1179, 1221, 1360, 1371, 1567, 1609  
 \\_\_draw\_backend\_join\_bevel: .....  
 .. 1259, 1289, 1383, 1413, 1635, 1673  
 \\_\_draw\_backend\_join\_miter: .....  
 .. 1259, 1285, 1383, 1409, 1635, 1669  
 \\_\_draw\_backend\_join\_round: .....  
 .. 1259, 1287, 1383, 1411, 1635, 1671  
 \\_\_draw\_backend\_lineto:nn .....  
 .. 1139, 1147, 1326, 1331, 1522, 1527  
 \\_\_draw\_backend\_linewidth:n .....  
 .. 1259, 1272, 1383, 1396, 1635, 1659  
 \\_\_draw\_backend\_literal:n .....  
 ..... 1123, 1123, 1124, 1128,  
 1132, 1136, 1138, 1141, 1149, 1157,  
 1166, 1180, 1183, 1184, 1185, 1186,  
 1189, 1195, 1205, 1212, 1218, 1223,  
 1228, 1229, 1230, 1231, 1234, 1240,  
 1250, 1256, 1261, 1274, 1278, 1280,  
 1282, 1284, 1286, 1288, 1290, 1293,  
 1298, 1299, 1300, 1301, 1302, 1303,  
 1307, 1308, 1310, 1311, 1312, 1313,  
 1314, 1318, 1318, 1319, 1328, 1333,  
 1338, 1348, 1361, 1363, 1365, 1368,  
 1373, 1378, 1382, 1385, 1398, 1402,  
 1404, 1406, 1408, 1410, 1412, 1414,  
 1512, 1512, 1513, 1574, 1593, 1619  
 \\_\_draw\_backend\_miterlimit:n ...  
 .. 1259, 1277, 1383, 1401, 1635, 1661  
 \\_\_draw\_backend\_moveto:nn .....  
 .. 1139, 1139, 1326, 1326, 1522, 1522  
 \\_\_draw\_backend\_nonzero\_rule: ...  
 .. 1174, 1176, 1355, 1357, 1563, 1565  
 \\_\_draw\_backend\_path:n .....  
 ..... 1567, 1569, 1601, 1608, 1610  
 \g\_\_draw\_backend\_path\_int 1582, 1599  
 \g\_\_draw\_backend\_path\_tl .....  
 ..... 1522, 1578, 1594, 1596, 1623  
 \\_\_draw\_backend\_rectangle:nnnn ..  
 .. 1139, 1155, 1326, 1346, 1522, 1532  
 \\_\_draw\_backend\_scope\_begin: 1135,  
 1135, 1321, 1324, 1324, 1514, 1514

<code>\__draw_backend_scope_end:</code>	<a href="#">1135</a> , <a href="#">1137</a> , <a href="#">1323</a> , <a href="#">1324</a> , <a href="#">1325</a> , <a href="#">1514</a> , <a href="#">1515</a>
<code>\__draw_backend_stroke:</code>	<a href="#">1179</a> , <a href="#">1181</a> , <a href="#">1201</a> , <a href="#">1360</a> , <a href="#">1362</a> , <a href="#">1567</a> , <a href="#">1600</a> , <a href="#">1605</a>
<code>\g_draw_draw_clip_bool</code>	<a href="#">1179</a> , <a href="#">1567</a>
<code>\g__draw_draw_eor_bool</code>	<a href="#">1174</a> , <a href="#">1191</a> , <a href="#">1207</a> , <a href="#">1214</a> , <a href="#">1225</a> , <a href="#">1236</a> , <a href="#">1252</a> , <a href="#">1355</a> , <a href="#">1369</a> , <a href="#">1374</a> , <a href="#">1379</a>
<code>\g__draw_draw_path_int</code>	<a href="#">1567</a>
<code>\g__draw_path_tl</code>	<a href="#">1632</a>
<b>E</b>	
<code>\errmessage</code>	<a href="#">38</a>
<code>\evensidemargin</code>	<a href="#">2405</a>
exp commands:	
<code>\exp_after:wN</code>	<a href="#">2054</a>
<code>\exp_args:Ne</code>	<a href="#">666</a> , <a href="#">1782</a> , <a href="#">1835</a> , <a href="#">1861</a> , <a href="#">2559</a> , <a href="#">3072</a>
<code>\exp_args:Nf</code>	<a href="#">1264</a> , <a href="#">1388</a> , <a href="#">2324</a>
<code>\exp_args:NNf</code>	<a href="#">220</a> , <a href="#">268</a> , <a href="#">325</a>
<code>\exp_args:Nnx</code>	<a href="#">2311</a> , <a href="#">2997</a>
<code>\exp_args:NV</code>	<a href="#">442</a>
<code>\exp_args:Nx</code>	<a href="#">612</a> , <a href="#">847</a> , <a href="#">1833</a> , <a href="#">1859</a> , <a href="#">2268</a> , <a href="#">2283</a> , <a href="#">2401</a> , <a href="#">2963</a> , <a href="#">3168</a> , <a href="#">3227</a>
<code>\exp_last_unbraced:Nx</code>	<a href="#">449</a> , <a href="#">459</a>
<code>\exp_not:N</code>	<a href="#">579</a> , <a href="#">585</a> , <a href="#">586</a> , <a href="#">587</a> , <a href="#">614</a> , <a href="#">615</a> , <a href="#">618</a> , <a href="#">619</a> , <a href="#">624</a> , <a href="#">2645</a> , <a href="#">2647</a> , <a href="#">2650</a> , <a href="#">2680</a> , <a href="#">2682</a> , <a href="#">2685</a> , <a href="#">2837</a> , <a href="#">2839</a> , <a href="#">2842</a> , <a href="#">2848</a> , <a href="#">2850</a> , <a href="#">2853</a> , <a href="#">2890</a> , <a href="#">2891</a> , <a href="#">2897</a> , <a href="#">2898</a> , <a href="#">2917</a> , <a href="#">2922</a> , <a href="#">3029</a> , <a href="#">3034</a>
<code>\exp_not:n</code>	<a href="#">48</a> , <a href="#">97</a> , <a href="#">108</a> , <a href="#">136</a> , <a href="#">936</a> , <a href="#">2259</a> , <a href="#">2264</a> , <a href="#">2553</a> , <a href="#">2792</a> , <a href="#">2793</a> , <a href="#">2807</a> , <a href="#">2808</a> , <a href="#">2820</a> , <a href="#">2821</a> , <a href="#">2975</a> , <a href="#">2980</a> , <a href="#">2991</a> , <a href="#">3052</a>
<code>\ExplBackendFileDate</code>	<a href="#">1</a>
<b>F</b>	
file commands:	
<code>\file_compare_timestamp:nNnTF</code>	<a href="#">1847</a>
<code>\file_parse_full_name:nNNN</code>	<a href="#">1829</a> , <a href="#">1857</a>
<code>\fmtversion</code>	<a href="#">52</a>
fp commands:	
<code>\fp_compare:nNnTF</code>	<a href="#">227</a> , <a href="#">274</a> , <a href="#">280</a> , <a href="#">332</a> , <a href="#">1431</a> , <a href="#">1444</a> , <a href="#">1489</a>
<code>\fp_eval:n</code>	<a href="#">220</a> , <a href="#">229</a> , <a href="#">242</a> , <a href="#">243</a> , <a href="#">268</a> , <a href="#">285</a> , <a href="#">300</a> , <a href="#">302</a> , <a href="#">325</a> , <a href="#">334</a> , <a href="#">345</a> , <a href="#">346</a> , <a href="#">410</a> , <a href="#">425</a> , <a href="#">426</a> , <a href="#">1067</a> , <a href="#">1068</a> , <a href="#">1069</a> , <a href="#">1077</a> , <a href="#">1090</a> , <a href="#">1091</a> , <a href="#">1092</a> , <a href="#">1433</a> , <a href="#">1438</a> , <a href="#">1439</a> , <a href="#">1446</a> , <a href="#">1456</a> , <a href="#">1457</a> , <a href="#">1458</a> , <a href="#">1459</a> , <a href="#">1468</a> , <a href="#">1469</a> , <a href="#">1470</a> , <a href="#">1471</a> , <a href="#">1480</a> , <a href="#">1481</a> , <a href="#">1482</a> , <a href="#">1483</a> , <a href="#">2550</a> , <a href="#">2719</a> , <a href="#">3066</a> , <a href="#">3169</a> , <a href="#">3179</a> , <a href="#">3186</a> , <a href="#">3228</a> , <a href="#">3265</a> , <a href="#">3272</a> , <a href="#">3307</a>
<code>\fp_new:N</code>	<a href="#">293</a> , <a href="#">294</a>
<code>\fp_set:Nn</code>	<a href="#">273</a> , <a href="#">276</a>
<code>\fp_use:N</code>	<a href="#">279</a> , <a href="#">283</a> , <a href="#">288</a>
<code>\fp_zero:N</code>	<a href="#">275</a>
<code>\c_zero_fp</code>	<a href="#">227</a> , <a href="#">274</a> , <a href="#">280</a> , <a href="#">332</a> , <a href="#">1431</a> , <a href="#">1444</a>
<b>G</b>	
graphics commands:	
<code>\l_graphics_search_ext_seq</code>	<a href="#">1717</a> , <a href="#">1740</a> , <a href="#">1879</a> , <a href="#">2067</a>
graphics internal commands:	
<code>\__graphics_backend_dequote:w</code>	<a href="#">1747</a> , <a href="#">1783</a> , <a href="#">1813</a>
<code>\l__graphics_backend_dir_str</code>	<a href="#">1822</a>
<code>\l__graphics_backend_ext_str</code>	<a href="#">1822</a>
<code>\__graphics_backend_get_pagecount:n</code>	<a href="#">1736</a> , <a href="#">1737</a> , <a href="#">1871</a> , <a href="#">1871</a> , <a href="#">1986</a> , <a href="#">1988</a> , <a href="#">2056</a> , <a href="#">2056</a> , <a href="#">2208</a> , <a href="#">2209</a>
<code>\__graphics_backend_getbb_auxi:n</code>	<a href="#">1747</a> , <a href="#">1759</a> , <a href="#">1773</a> , <a href="#">1775</a>
<code>\__graphics_backend_getbb_-auxi:nN</code>	<a href="#">1992</a> , <a href="#">1996</a> , <a href="#">2005</a> , <a href="#">2007</a>
<code>\__graphics_backend_getbb_-auxii:n</code>	<a href="#">1747</a> , <a href="#">1778</a> , <a href="#">1780</a>
<code>\__graphics_backend_getbb_-auxii:nnN</code>	<a href="#">1992</a> , <a href="#">2010</a> , <a href="#">2013</a> , <a href="#">2015</a>
<code>\__graphics_backend_getbb_-auxiii:n</code>	<a href="#">1747</a> , <a href="#">1782</a> , <a href="#">1788</a>
<code>\__graphics_backend_getbb_-auxiii:nNnn</code>	<a href="#">1992</a> , <a href="#">2011</a> , <a href="#">2014</a> , <a href="#">2016</a>
<code>\__graphics_backend_getbb_-auxiv:nnNnn</code>	<a href="#">1992</a> , <a href="#">2019</a> , <a href="#">2023</a> , <a href="#">2035</a>
<code>\__graphics_backend_getbb_-auxv:nNnn</code>	<a href="#">1992</a> , <a href="#">2020</a> , <a href="#">2027</a> , <a href="#">2036</a>
<code>\__graphics_backend_getbb_-auxvi:nNnn</code>	<a href="#">2039</a> , <a href="#">2041</a>
<code>\__graphics_backend_getbb_bmp:n</code>	<a href="#">1884</a> , <a href="#">1898</a> , <a href="#">1992</a> , <a href="#">2000</a>
<code>\__graphics_backend_getbb_eps:n</code>	<a href="#">1719</a> , <a href="#">1721</a> , <a href="#">1822</a> , <a href="#">1827</a> , <a href="#">1844</a> , <a href="#">1884</a> , <a href="#">1886</a> , <a href="#">2152</a> , <a href="#">2154</a>
<code>\__graphics_backend_getbb_eps:nm</code>	<a href="#">1822</a>
<code>\__graphics_backend_getbb_eps:nn</code>	<a href="#">1833</a> , <a href="#">1845</a>
<code>\__graphics_backend_getbb_jpeg:n</code>	<a href="#">1747</a> , <a href="#">1761</a> , <a href="#">1884</a> , <a href="#">1896</a> , <a href="#">1992</a> , <a href="#">1998</a> , <a href="#">2157</a> , <a href="#">2163</a>
<code>\__graphics_backend_getbb_jpg:n</code>	<a href="#">1747</a> , <a href="#">1747</a> , <a href="#">1761</a> , <a href="#">1762</a> , <a href="#">1884</a> , <a href="#">1890</a> , <a href="#">1896</a> , <a href="#">1897</a> , <a href="#">1898</a> , <a href="#">1992</a> , <a href="#">1992</a> , <a href="#">1998</a> , <a href="#">1999</a> , <a href="#">2000</a> , <a href="#">2157</a> , <a href="#">2157</a> , <a href="#">2163</a> , <a href="#">2164</a>
<code>\__graphics_backend_getbb_-pagebox:w</code>	<a href="#">1992</a> , <a href="#">2031</a> , <a href="#">2048</a> , <a href="#">2054</a>
<code>\__graphics_backend_getbb_pdf:n</code>	<a href="#">1747</a> , <a href="#">1763</a> , <a href="#">1853</a> , <a href="#">1884</a> , <a href="#">1899</a> , <a href="#">1992</a> , <a href="#">2001</a> , <a href="#">2165</a> , <a href="#">2165</a>

<code>\__graphics_backend_getbb_png:n .</code>	<a href="#">1747</a> , <a href="#">1762</a> ,
<code>.....</code>	<a href="#">1884</a> , <a href="#">1897</a> , <a href="#">1992</a> , <a href="#">1999</a> , <a href="#">2157</a> , <a href="#">2164</a>
<code>\__graphics_backend_getbb_ps:n .</code>	<a href="#">1719</a> , <a href="#">1722</a> ,
<code>.....</code>	<a href="#">1822</a> , <a href="#">1844</a> , <a href="#">1884</a> , <a href="#">1887</a> , <a href="#">2152</a> , <a href="#">2155</a>
<code>\__graphics_backend_getbb_svg:n .</code>	<a href="#">2073</a> , <a href="#">2073</a>
<code>\__graphics_backend_getbb_svg_-</code>	
<code>auxi:nNn . . .</code>	<a href="#">2073</a> , <a href="#">2089</a> , <a href="#">2094</a> , <a href="#">2107</a>
<code>\__graphics_backend_getbb_svg_-</code>	
<code>auxii:w . . . .</code>	<a href="#">2073</a> , <a href="#">2111</a> , <a href="#">2133</a> , <a href="#">2138</a>
<code>\__graphics_backend_getbb_svg_-</code>	
<code>auxiii:Nw . . . . .</code>	<a href="#">2073</a> , <a href="#">2121</a> , <a href="#">2139</a>
<code>\__graphics_backend_getbb_svg_-</code>	
<code>auxiv:Nw . . . . .</code>	<a href="#">2073</a> , <a href="#">2124</a> , <a href="#">2141</a>
<code>\__graphics_backend_getbb_svg_-</code>	
<code>auxv:Nw . . . . .</code>	<a href="#">2073</a> , <a href="#">2125</a> , <a href="#">2143</a>
<code>\__graphics_backend_getbb_svg_-</code>	
<code>auxvi:Nn . . . . .</code>	<a href="#">2073</a> , <a href="#">2140</a> , <a href="#">2142</a> , <a href="#">2144</a> , <a href="#">2145</a>
<code>\__graphics_backend_getbb_svg_-</code>	
<code>auxvii:w . . . . .</code>	<a href="#">2073</a> , <a href="#">2147</a> , <a href="#">2151</a>
<code>\__graphics_backend_include:nn . .</code>	<a href="#">2171</a> , <a href="#">2172</a> , <a href="#">2175</a> , <a href="#">2176</a>
<code>\__graphics_backend_include_-</code>	
<code>auxi:nn . . . .</code>	<a href="#">1907</a> , <a href="#">1920</a> , <a href="#">1926</a> , <a href="#">1928</a>
<code>\__graphics_backend_include_-</code>	
<code>auxii:nnn . . . .</code>	<a href="#">1907</a> , <a href="#">1930</a> , <a href="#">1943</a> , <a href="#">1952</a>
<code>\__graphics_backend_include_-</code>	
<code>auxiii:nnn . . . . .</code>	<a href="#">1907</a> , <a href="#">1950</a> , <a href="#">1953</a>
<code>\__graphics_backend_include_-</code>	
<code>bmp:n . . . . .</code>	<a href="#">1907</a> , <a href="#">1923</a>
<code>\__graphics_backend_include_-</code>	
<code>dequote:w . . . . .</code>	<a href="#">2187</a> , <a href="#">2198</a> , <a href="#">2206</a>
<code>\__graphics_backend_include_-</code>	
<code>eps:n . . . . .</code>	<a href="#">1724</a> ,
<code>.....</code>	<a href="#">1724</a> , <a href="#">1735</a> , <a href="#">1822</a> , <a href="#">1855</a> , <a href="#">1869</a> ,
<code>.....</code>	<a href="#">1907</a> , <a href="#">1907</a> , <a href="#">1918</a> , <a href="#">2171</a> , <a href="#">2171</a> , <a href="#">2173</a>
<code>\__graphics_backend_include_-</code>	
<code>jpeg:n . . . . .</code>	<a href="#">1814</a> , <a href="#">1819</a> , <a href="#">1921</a> , <a href="#">2187</a> , <a href="#">2204</a>
<code>\__graphics_backend_include_-</code>	
<code>jpg:n . . . . .</code>	<a href="#">1814</a> ,
<code>.....</code>	<a href="#">1814</a> , <a href="#">1819</a> , <a href="#">1820</a> , <a href="#">1821</a> , <a href="#">1907</a> ,
<code>.....</code>	<a href="#">1919</a> , <a href="#">1921</a> , <a href="#">1922</a> , <a href="#">1923</a> , <a href="#">2187</a> , <a href="#">2205</a>
<code>\__graphics_backend_include_-</code>	
<code>jpseg:n . . . . .</code>	<a href="#">1907</a>
<code>\__graphics_backend_include_-</code>	
<code>pdf:n . . . . .</code>	<a href="#">1814</a> , <a href="#">1820</a> , <a href="#">1859</a> ,
<code>.....</code>	<a href="#">1907</a> , <a href="#">1925</a> , <a href="#">2049</a> , <a href="#">2049</a> , <a href="#">2171</a> , <a href="#">2174</a>
<code>\__graphics_backend_include_-</code>	
<code>png:n . . . . .</code>	
<code>..</code>	<a href="#">1814</a> , <a href="#">1821</a> , <a href="#">1907</a> , <a href="#">1922</a> , <a href="#">2187</a> , <a href="#">2203</a>
<code>\__graphics_backend_include_ps:n</code>	<a href="#">1724</a> , <a href="#">1735</a> ,
<code>.....</code>	<a href="#">1822</a> , <a href="#">1869</a> , <a href="#">1907</a> , <a href="#">1918</a> , <a href="#">2171</a> , <a href="#">2173</a>
<code>\__graphics_backend_include_-</code>	
<code>svg:n . . . . .</code>	<a href="#">2187</a> , <a href="#">2187</a> , <a href="#">2203</a> , <a href="#">2204</a> , <a href="#">2205</a>
<code>\__graphics_backend_loaded:n . . .</code>	<a href="#">1705</a> , <a href="#">1705</a> , <a href="#">1717</a> , <a href="#">1719</a> , <a href="#">1736</a> , <a href="#">1740</a> ,
<code>.....</code>	<a href="#">1879</a> , <a href="#">1884</a> , <a href="#">1987</a> , <a href="#">2067</a> , <a href="#">2152</a> , <a href="#">2208</a>
<code>\l__graphics_backend_name_str .</code>	<a href="#">1822</a>
<code>\__graphics_bb_restore:nTF . . . . .</code>	<a href="#">1777</a> , <a href="#">2038</a> , <a href="#">2075</a>
<code>\__graphics_bb_save:n . . . . .</code>	<a href="#">1786</a> , <a href="#">2046</a> , <a href="#">2102</a>
<code>\l__graphics_decodearray_str . . .</code>	<a href="#">1753</a> , <a href="#">1754</a> ,
<code>.....</code>	<a href="#">1765</a> , <a href="#">1793</a> , <a href="#">1797</a> , <a href="#">1798</a> , <a href="#">1901</a> , <a href="#">1936</a> ,
<code>.....</code>	<a href="#">1937</a> , <a href="#">1975</a> , <a href="#">1978</a> , <a href="#">1979</a> , <a href="#">2003</a> , <a href="#">2167</a>
<code>\__graphics_extract_bb:n . . . . .</code>	<a href="#">1894</a> , <a href="#">1903</a> , <a href="#">2161</a> , <a href="#">2169</a>
<code>\l__graphics_final_name_str . . .</code>	<a href="#">1852</a>
<code>\__graphics_get_pagecount:n . . . .</code>	<a href="#">1737</a> , <a href="#">1988</a> , <a href="#">2209</a>
<code>\l__graphics_graphics_attr_tl . . .</code>	<a href="#">1746</a> , <a href="#">1751</a> ,
<code>.....</code>	<a href="#">1758</a> , <a href="#">1767</a> , <a href="#">1777</a> , <a href="#">1784</a> , <a href="#">1786</a> , <a href="#">1817</a>
<code>\l__graphics_internal_box . . . . .</code>	<a href="#">1808</a> , <a href="#">1810</a> , <a href="#">1811</a> , <a href="#">2043</a> , <a href="#">2044</a> , <a href="#">2045</a>
<code>\l__graphics_internal_dim . . . . .</code>	<a href="#">2148</a> , <a href="#">2149</a>
<code>\l__graphics_internal_ior . . . . .</code>	<a href="#">2077</a> , <a href="#">2078</a> , <a href="#">2085</a> , <a href="#">2104</a>
<code>\l__graphics_interpolate_bool . . .</code>	<a href="#">1755</a> , <a href="#">1766</a> , <a href="#">1792</a> , <a href="#">1799</a> ,
<code>.....</code>	<a href="#">1902</a> , <a href="#">1938</a> , <a href="#">1974</a> , <a href="#">1980</a> , <a href="#">2004</a> , <a href="#">2168</a>
<code>\l__graphics_llx_dim . . . . .</code>	<a href="#">1729</a> , <a href="#">1912</a> , <a href="#">1967</a> , <a href="#">2081</a> , <a href="#">2181</a>
<code>\l__graphics_lly_dim . . . . .</code>	<a href="#">1730</a> , <a href="#">1913</a> , <a href="#">1968</a> , <a href="#">2082</a> , <a href="#">2182</a>
<code>\l__graphics_page_int . . . . .</code>	<a href="#">1749</a> , <a href="#">1770</a> , <a href="#">1771</a> , <a href="#">1803</a> ,
<code>.....</code>	<a href="#">1804</a> , <a href="#">1892</a> , <a href="#">1934</a> , <a href="#">1935</a> , <a href="#">1961</a> , <a href="#">1962</a> ,
<code>.....</code>	<a href="#">1994</a> , <a href="#">2009</a> , <a href="#">2010</a> , <a href="#">2052</a> , <a href="#">2053</a> , <a href="#">2159</a>
<code>\l__graphics_pagebox_tl . . . . .</code>	<a href="#">54</a> , <a href="#">1750</a> , <a href="#">1769</a> ,
<code>.....</code>	<a href="#">1805</a> , <a href="#">1806</a> , <a href="#">1893</a> , <a href="#">1932</a> , <a href="#">1933</a> , <a href="#">1963</a> ,
<code>.....</code>	<a href="#">1965</a> , <a href="#">1995</a> , <a href="#">2018</a> , <a href="#">2019</a> , <a href="#">2054</a> , <a href="#">2160</a>
<code>\__graphics_read_bb:n . . . . .</code>	<a href="#">1721</a> , <a href="#">1722</a> , <a href="#">1886</a> , <a href="#">1887</a> , <a href="#">2154</a> , <a href="#">2155</a>
<code>\g__graphics_track_int . . . . .</code>	<a href="#">1906</a> , <a href="#">1955</a> , <a href="#">1956</a>
<code>\l__graphics_urx_dim . . . . .</code>	<a href="#">1731</a> , <a href="#">1810</a> , <a href="#">1914</a> , <a href="#">1969</a> , <a href="#">2044</a> ,
<code>.....</code>	<a href="#">2083</a> , <a href="#">2087</a> , <a href="#">2090</a> , <a href="#">2098</a> , <a href="#">2183</a> , <a href="#">2196</a>
<code>\l__graphics_ury_dim . . . . .</code>	<a href="#">1732</a> , <a href="#">1811</a> , <a href="#">1915</a> , <a href="#">1970</a> , <a href="#">2045</a> , <a href="#">2084</a> ,
<code>.....</code>	<a href="#">2092</a> , <a href="#">2095</a> , <a href="#">2099</a> , <a href="#">2184</a> , <a href="#">2189</a> , <a href="#">2197</a>
<code>group commands:</code>	
<code>\group_begin: . . . . .</code>	<a href="#">163</a> , <a href="#">182</a>

\group\_end: ..... 171  
 \group\_insert\_after:N ... 3244, 3294

## H

hbox commands:

\hbox:n ..... 2191, 2330, 2333,  
 2408, 2414, 2567, 2574, 3080, 3091  
 \hbox\_overlap\_right:n ..... 215,  
 247, 263, 304, 320, 348, 432, 1309, 1504  
 \hbox\_set:Nn .. 1808, 2043, 2400, 2432  
 \hbox\_set:Nw ..... 2383  
 \hbox\_set\_end: ..... 2398  
 \hbox\_unpack:N ..... 2519

hook commands:

\hook\_gput\_code:nnn .. 55, 1707, 1709

## I

int commands:

\int\_compare:nNnTF .....  
 ..... 1770, 1803, 1934, 1961,  
 2009, 2052, 2491, 2592, 2888, 2916  
 \int\_const:Nn ..... 488, 1784,  
 1874, 1956, 2058, 2228, 2766, 2954  
 \int\_eval:n 508, 518, 662, 671, 684,  
 686, 690, 703, 2616, 2620, 2866,  
 2891, 2898, 2911, 3109, 3117, 3122  
 \int\_gincr:N ..... 189,  
 355, 1573, 1618, 1955, 2227, 2296,  
 2340, 2417, 2953, 2996, 3009, 3029  
 \int\_gset:Nn ..... 164, 183, 2480  
 \int\_gset\_eq:NN 172, 2341, 2418, 3010  
 \int\_if\_exist:NTF ..... 1945  
 \int\_if\_odd:nTF ..... 2403  
 \int\_max:nn ..... 2060  
 \int\_new:N ..... 155,  
 156, 402, 483, 1599, 1906, 2223,  
 2321, 2352, 2354, 2949, 3006, 3022  
 \int\_set\_eq:NN ..... 160, 179, 2492  
 \int\_step\_function:nnnN ..... 688  
 \int\_use:N ..... 357,  
 388, 615, 624, 772, 800, 849, 855,  
 856, 910, 911, 920, 944, 1576, 1582,  
 1589, 1621, 1629, 1771, 1804, 1817,  
 1875, 1935, 1948, 1960, 1962, 2053,  
 2061, 2234, 2299, 2312, 2316, 2344,  
 2351, 2422, 2523, 2777, 2787, 2960,  
 2998, 3003, 3013, 3021, 3034, 3045  
 \int\_value:w .....  
 ..... 2645, 2680, 2837, 2848, 2866  
 \int\_zero:N ... 1749, 1892, 1994, 2159

ior commands:

\ior\_close:N ..... 2104  
 \ior\_if\_eof:NTF ..... 2078  
 \ior\_map\_break: ..... 2100

\ior\_open:Nn ..... 2077  
 \ior\_str\_map\_inline:Nn ..... 2085

## K

kernel internal commands:

\\_\_kernel\_backend\_align\_begin: ..  
 ..... 72, 72, 200, 224, 239  
 \\_\_kernel\_backend\_align\_end: ...  
 ..... 72, 78, 214, 232, 246  
 \\_\_kernel\_backend\_first\_shipout:n  
 ..... 50, 54, 57, 59, 69, 612  
 \g\_kernel\_backend\_header\_bool ..  
 ..... 67, 610  
 \\_\_kernel\_backend\_literal:n ....  
 ..... 46, 46, 47, 48,  
 49, 62, 65, 70, 74, 81, 84, 86, 142,  
 145, 147, 149, 153, 329, 342, 532,  
 538, 560, 565, 632, 768, 812, 964,  
 969, 975, 980, 1031, 1057, 1127,  
 1133, 1428, 1435, 1441, 1501, 1506,  
 1726, 1909, 1947, 1957, 2178, 2193,  
 2943, 3047, 3109, 3113, 3118, 3123  
 \\_\_kernel\_backend\_literal\_page:n  
 ..... 100,  
 100, 144, 144, 2937, 2939, 3128, 3130  
 \\_\_kernel\_backend\_literal\_pdf:n .  
 ..... 89, 89, 99, 141, 141,  
 143, 255, 312, 1318, 3238, 3255, 3288  
 \\_\_kernel\_backend\_literal\_-  
 postscript:n ..... 61,  
 61, 63, 75, 76, 80, 201, 202, 204,  
 205, 213, 225, 240, 1123, 2594, 2606  
 \\_\_kernel\_backend\_literal\_svg:n .  
 . 152, 152, 154, 159, 170, 178, 188,  
 356, 358, 375, 794, 1512, 1687, 1698  
 \\_\_kernel\_backend\_matrix:n .....  
 ..... 128, 128, 138, 277, 298, 1418  
 \\_\_kernel\_backend\_postscript:n ..  
 ..... 64, 64, 66,  
 534, 1034, 1036, 1038, 1042, 2217,  
 2273, 2288, 2330, 2336, 2376, 2408,  
 2415, 2419, 2433, 2461, 2505, 2512,  
 2518, 2526, 2533, 2567, 2574, 3192  
 \\_\_kernel\_backend\_scope:n .....  
 ..... 157, 186, 191, 385,  
 390, 1097, 1519, 1564, 1566, 1586,  
 1626, 1648, 1660, 1662, 1664, 1666,  
 1668, 1670, 1672, 1674, 1677, 3307  
 \\_\_kernel\_backend\_scope\_begin: ..  
 83, 83, 110, 110, 146, 146, 157, 157,  
 199, 223, 238, 254, 271, 297, 311,  
 328, 341, 1324, 1496, 1514, 1518, 1685  
 \\_\_kernel\_backend\_scope\_begin:n .  
 ..... 157, 176, 185, 377, 405, 418

`\__kernel_backend_scope_end:` ...  
     ... [83](#), [85](#), [110](#), [119](#),  
     [146](#), [148](#), [157](#), [166](#), [216](#), [234](#), [248](#),  
     [264](#), [291](#), [305](#), [321](#), [337](#), [349](#), [400](#),  
     [414](#), [433](#), [1325](#), [1508](#), [1515](#), [1521](#), [1699](#)  
`\g__kernel_backend_scope_int` ...  
     ... [155](#), [162](#), [164](#), [169](#), [173](#), [181](#), [183](#), [189](#)  
`\l__kernel_backend_scope_int` ...  
     ... [155](#), [161](#), [174](#), [180](#)  
`\g__kernel_clip_path_int` ...  
     ... [353](#), [1573](#), [1576](#), [1589](#), [1618](#), [1621](#), [1629](#)  
`\__kernel_color_backend_stack-`  
`init:Nnn` ... [486](#), [486](#), [3217](#)  
`\__kernel_color_backend_stack-`  
`pop:n` ... [500](#), [510](#), [556](#), [3259](#)  
`\__kernel_color_backend_stack-`  
`push:nn` ...  
     ... [500](#), [500](#), [553](#), [998](#), [1010](#), [3241](#), [3291](#)  
`\__kernel_dependency_version-`  
`check:Nn` ... [1](#)  
`\__kernel_dependency_version-`  
`check:nn` ... [27](#), [29](#)  
`\__kernel_file_name_quote:n` ...  
     ... [1835](#), [1861](#)  
`\__kernel_kern:n` ...  
     ... [2335](#), [2337](#), [2566](#), [2570](#),  
     [2573](#), [2577](#), [3079](#), [3087](#), [3090](#), [3106](#)

**M**

`\MessageBreak` ... [40](#)  
mode commands:  
`\mode_if_horizontal:TF` ... [2482](#), [2489](#)  
`\mode_if_math:TF` ... [2380](#)  
msg commands:  
`\msg_error:nnn` ... [570](#), [2079](#)  
`\msg_new:nnn` ... [572](#)  
`\msg_warning:nnn` ... [445](#), [475](#)

**O**

`\oddsidemargin` ... [2404](#)  
opacity internal commands:  
`\__opacity_backend:nn` ...  
     ... [3300](#), [3301](#), [3303](#), [3305](#), [3306](#)  
`\__opacity_backend:nnn` ... [3166](#),  
     [3173](#), [3174](#), [3178](#), [3185](#), [3190](#), [3209](#)  
`\__opacity_backend_fill:n` ...  
     ... [3166](#), [3176](#), [3262](#), [3262](#), [3300](#), [3302](#)  
`\__opacity_backend_fill_stroke:nn`  
     ... [3264](#), [3270](#), [3274](#), [3297](#)  
`\l__opacity_backend_fill_tl` ...  
     ... [3223](#), [3232](#), [3271](#), [3279](#)  
`\__opacity_backend_fillstroke:nn`  
     ... [3262](#)

`\__opacity_backend_reset:` ...  
     ... [3225](#), [3244](#), [3252](#), [3294](#)  
`\__opacity_backend_select:n` ...  
     ... [3166](#), [3166](#), [3225](#), [3225](#), [3300](#), [3300](#)  
`\__opacity_backend_select_aux:n` ...  
     ... [3166](#), [3168](#),  
     [3171](#), [3225](#), [3227](#), [3230](#), [3250](#), [3277](#)  
`\c__opacity_backend_stack_int` ...  
     ... [3212](#), [3241](#), [3259](#), [3291](#)  
`\__opacity_backend_stroke:n` ...  
     ... [3166](#), [3183](#), [3262](#), [3268](#), [3300](#), [3304](#)  
`\l__opacity_backend_stroke_tl` ...  
     ... [3223](#), [3233](#), [3266](#), [3280](#)

**P**

pdf commands:  
`\pdf_object_if_exist:nTF` [864](#), [930](#), [948](#)  
`\pdf_object_new:nn` ...  
     ... [855](#), [866](#), [910](#), [932](#), [950](#)  
`\pdf_object_ref:n` ...  
     ... [812](#), [879](#), [943](#), [958](#), [976](#), [981](#)  
`\pdf_object_ref_last:` ...  
     ... [832](#), [857](#), [860](#), [916](#)  
`\pdf_object_unnamed_write:nn` ...  
     ... [839](#), [886](#), [942](#), [957](#)  
`\pdf_object_write:nn` ...  
     ... [856](#), [867](#), [911](#), [933](#), [951](#)

pdf internal commands:  
`\__pdf_backend:n` . [2942](#), [2942](#), [2944](#),  
     [2946](#), [2948](#), [2974](#), [2979](#), [2988](#), [3011](#),  
     [3030](#), [3043](#), [3050](#), [3082](#), [3083](#), [3093](#)  
`\__pdf_backend_annotation:nnnn` ...  
     ... [2322](#), [2322](#),  
     [2630](#), [2630](#), [3007](#), [3007](#), [3133](#), [3133](#)  
`\__pdf_backend_annotation-`  
`aux:nnnn` ... [2324](#), [2327](#)  
`\g__pdf_backend_annotation_int` ...  
     ... [2321](#), [2341](#), [2351](#), [3006](#), [3010](#), [3021](#)  
`\__pdf_backend_annotation_last:` ...  
     ... [2350](#), [2350](#),  
     [2643](#), [2643](#), [3020](#), [3020](#), [3134](#), [3134](#)  
`\__pdf_backend_bdc:nn` [2624](#), [2624](#),  
     [2936](#), [2936](#), [3127](#), [3127](#), [3159](#), [3159](#)  
`\__pdf_backend_catalog_gput:nn` ...  
     ... [2219](#), [2219](#),  
     [2736](#), [2736](#), [2945](#), [2945](#), [3143](#), [3143](#)  
`\__pdf_backend_compress_objects:n`  
     ... [2590](#), [2602](#),  
     [2857](#), [2868](#), [3108](#), [3110](#), [3153](#), [3154](#)  
`\__pdf_backend_compresslevel:n` ...  
     ... [2590](#), [2590](#),  
     [2857](#), [2857](#), [3108](#), [3108](#), [3153](#), [3153](#)  
`\l__pdf_backend_content_box` [2319](#),  
     [2383](#), [2407](#), [2410](#), [2412](#), [2441](#), [2452](#)



\__pdf_backend_destination:nn ...	\g__pdf_backend_link_sf_int ....
..... 2531, 2531,	..... 2354, 2480, 2491, 2492
2699, 2699, 3048, 3048, 3141, 3141	\__pdf_backend_link_sf_restore: .
\__pdf_backend_destination:nnnn .	..... 2361, 2384, 2427, 2487
..... 2531, 2557,	\__pdf_backend_link_sf_save: ...
2699, 2722, 3048, 3070, 3141, 3142	..... 2361, 2379, 2397, 2478
\__pdf_backend_destination_-	\l__pdf_backend_model_box . 2320,
aux:nnnn .. 2531, 2559, 2562, 3048, 3072, 3075	2400, 2432, 2440, 2451, 2466, 2468
\__pdf_backend_emc: .. 2624, 2626,	\__pdf_backend_objcompresslevel:n
2936, 2938, 3127, 3129, 3159, 3160	..... 2857, 2871, 2872, 2874
\__pdf_backend_info_gput:nn ....	\g__pdf_backend_object_int ....
..... 2219, 2221,	..... 2223, 2227, 2230,
2736, 2746, 2945, 2947, 3143, 3144	2296, 2299, 2312, 2316, 2340, 2341,
\__pdf_backend_link:nw ..... 2361	2344, 2417, 2418, 2949, 2953, 2956,
\__pdf_backend_link_aux:nw ... 2361	2996, 2998, 3003, 3009, 3010, 3013
\__pdf_backend_link_begin:n ....	\__pdf_backend_object_last: ....
..... 3023, 3024, 3026, 3027	..... 2315, 2315,
\__pdf_backend_link_begin:nnnw ..	2835, 2835, 3002, 3002, 3145, 3151
.. 2654, 2655, 2657, 2658, 3135, 3137	\__pdf_backend_object_new:nn ...
\__pdf_backend_link_begin:nw ...	..... 2225, 2225,
..... 2363, 2367, 2368	2757, 2757, 2951, 2951, 3145, 3145
\__pdf_backend_link_begin_aux:nw	\__pdf_backend_object_now:nn ...
..... 2371, 2373	2294, 2294, 2314, 2809, 2809, 2834,
\__pdf_backend_link_begin_-	2994, 2994, 3001, 3145, 3149, 3150
goto:nnw ..... 2361, 2361,	\g__pdf_backend_object_prop ....
2654, 2654, 3023, 3023, 3135, 3135	..... 2223, 2231, 2242, 2252,
\__pdf_backend_link_begin_-	2756, 2774, 2790, 2949, 2957, 2964
user:nnw ..... 2361, 2366,	\__pdf_backend_object_ref:n ....
2654, 2656, 3023, 3025, 3135, 3136	... 2225, 2233, 2239, 2253, 2757,
\g__pdf_backend_link_bool ..... 2356, 2370, 2375, 2390, 2428	2776, 2951, 2959, 2970, 3145, 3146
\g__pdf_backend_link_dict_tl ...	\__pdf_backend_object_write:nn ..
..... 2353, 2378, 2423	2235, 2235, 2255, 2778, 2778, 2806,
\__pdf_backend_link_end: .....	2961, 2961, 2966, 3145, 3147, 3148
..... 2361, 2388,	\__pdf_backend_object_write:nnn .
2654, 2669, 3023, 3042, 3135, 3138	..... 2961, 2963, 2967
\__pdf_backend_link_end_aux: ...	\__pdf_backend_object_write_-
..... 2361, 2391, 2393	array:nn ... 2235, 2256, 2961, 2972
\g__pdf_backend_link_int .....	\__pdf_backend_object_write_-
..... 2352, 2418,	dict:nn .... 2235, 2261, 2961, 2977
2422, 2523, 3022, 3029, 3034, 3045	\__pdf_backend_object_write_-
\__pdf_backend_link_last: .....	fstream:nn . 2235, 2266, 2961, 2982
..... 2522, 2522,	\__pdf_backend_object_write_-
2678, 2678, 3044, 3044, 3139, 3139	fstream:nnn ..... 2269, 2271
\__pdf_backend_link_margin:n ...	\__pdf_backend_object_write_-
..... 2524, 2524,	stream:nn .. 2235, 2281, 2961, 2984
2689, 2689, 3046, 3046, 3140, 3140	\__pdf_backend_object_write_-
\g__pdf_backend_link_math_bool ..	stream:nnn ..... 2235, 2284, 2286
..... 2355, 2381, 2382, 2385, 2395	\__pdf_backend_object_write_-
\__pdf_backend_link_minima: ....	stream:nnnn . 2961, 2983, 2985, 2986
..... 2361, 2399, 2430	\__pdf_backend_pageobject_ref:n .
\__pdf_backend_link_outerbox:n ..	..... 2317, 2317,
..... 2361, 2401, 2459	2846, 2846, 3004, 3004, 3145, 3152
	\__pdf_backend_pdfmark:n .....
	..... 2216, 2216,



2218, 2220, 2222, 2237, 2258, 2263, 2297, 2342, 2534, 2578, 2625, 2627	pdf.lly ..... 2361, 3325
\_pdf_backend_version_major: ... ... 2616, 2622, 2622, 2913, 2913, 3117, 3118, 3125, 3125, 3157, 3157	pdf.originx ..... 3396
\_pdf_backend_version_major_- gset:n ..... 2614, 2614, 2885, 2885, 3115, 3115, 3155, 3155	pdf.originy ..... 3396
\_pdf_backend_version_minor: ... ... 2620, 2622, 2623, 2913, 2926, 3122, 3123, 3125, 3126, 3157, 3158	pdf.outerbox ..... 2361, 3638
\_pdf_backend_version_minor_- gset:n ..... 2614, 2618, 2885, 2902, 3115, 3120, 3155, 3156	pdf.pdfmark ..... 3638
\l_pdf_breaklink_pdfmark_tl ... ..... 2357, 2425, 2517	pdf.pdfmark.dict ..... 3638
\_pdf_breaklink_postscript:n ... ..... 2359, 2359, 2409, 2411, 2518	pdf.pdfmark.good ..... 3638
\_pdf_breaklink_usebox:N ..... ..... 2360, 2360, 2410, 2519	pdf.pt.dvi ..... 3318
\_pdf_exp_not_i:nn ..... .. 2778, 2796, 2801, 2807, 2824, 2829	pdf.rect ..... 3325
\_pdf_exp_not_ii:nn ..... .. 2778, 2797, 2802, 2808, 2825, 2830	pdf.rect.ht ..... 3318
\l_pdf_internal_box ..... 2214	pdf.rightboundary ..... 3396
pdf.baselineskip ..... 2361, 3638	pdf.save.linkll ..... 3325
pdf.bordertracking ..... 3396	pdf.save.linkur ..... 3325
pdf.bordertracking.begin ..... 3396	pdf.save.ll ..... 3325
pdf.bordertracking.continue ..... 3396	pdf.save.ur ..... 3325
pdf.bordertracking.end ..... 3396	pdf.tmpa ..... 3361
pdf.bordertracking.endpage ..... 3396	pdf.tmpb ..... 3361
pdf.breaklink ..... 3534	pdf.tmpc ..... 3361
pdf.breaklink.write ..... 3534	pdf.tmpd ..... 3361
pdf.brokenlink.dict ..... 3396	pdf.urx ..... 3325
pdf.brokenlink.rect ..... 3396	pdf.ury ..... 2361, 3325
pdf.brokenlink.skip ..... 3396	pdfmanagement commands:
pdf.count ..... 3534	\pdfmanagement_add:nnn ..... ..... 829, 3220, 3234, 3281, 3284
pdf.currentrect ..... 3534	\pdfmanagement_if_active_p: ... ..... 824, 825, 3213, 3214, 3247, 3248
pdf.cvs ..... 3318	peek commands:
pdf.dest.anchor ..... 3361	\peek_meaning:Ntf ..... 2120, 2123
pdf.dest.point ..... 3361	\peek_remove_spaces:n ..... 2118
pdf.dest.x ..... 3361	prg commands:
pdf.dest.y ..... 3361	\prg_replicate:nn ..... ..... 168, 660, 681, 691, 892
pdf.dest2device ..... 3361	prop commands:
pdf.dev.x ..... 3361	\prop_gput:Nnn ..... ..... 618, 859, 2231, 2774, 2957
pdf.dev.y ..... 3361	\prop_if_in:Nntf ..... 595
pdf.dvi.pt ..... 3318	\prop_item:Nn ..... ..... 598, 2242, 2252, 2790, 2964
pdf.globaldict ..... 3315	\prop_new:N .... 576, 2224, 2756, 2950
pdf.leftboundary ..... 3396	\ProvidesExplFile ..... 2
pdf.link.dict ..... 2361	
pdf.linkdp.pad ..... 2361, 3322	
pdf.linkht.pad ..... 2361, 3322	
pdf.linkmargin ..... 3322	
pdf.llx ..... 2361, 3325	

## Q

quark commands:	
\quark_if_recursion_tail_stop:n	594
\q_recursion_stop	587
\q_recursion_tail	586

## S

scan commands:	
\scan_stop:	113, 122, 518, 2148, 2151, 2672, 2697, 2720, 2734, 2866, 2883, 2891, 2898, 2911

scan internal commands:	503, 513, 2633, 2661, 2672, 2702, 2725, 2739, 2749, 2760, 2781, 2812
\ s _ c o l o r _ s t o p	450, 453, 460, 463, 671, 672, 676, 680, 693, 696, 700, 704, 718, 893, 922, 926, 1060, 1062, 1083, 1085
\ s _ g r a p h i c s _ s t o p	1783, 1813, 2113, 2128, 2135, 2139, 2141, 2143, 2198, 2206
separation	3312
seq commands:	
\ s e q _ s e t _ f r o m _ c l i s t : N n	1718, 1742, 1881, 2069
skip commands:	
\ s k i p _ h o r i z o n t a l : n	217, 265, 322
str commands:	
\ c _ h a s h _ s t r	388, 1582, 1589, 1629
\ c _ p e r c e n t _ s t r	1103, 1104, 1105
\ s t r _ c a s e : n n	898, 2301, 2817
\ s t r _ c a s e : n n T F	2538, 2708, 3055
\ s t r _ c a s e _ e : n n	2241, 2789
\ s t r _ c o n v e r t _ p d f n a m e : n	619, 639, 848
\ s t r _ i f _ e q : n n T F	465, 468, 471, 568, 798, 3276
\ s t r _ n e w : N	1824, 1825, 1826
\ s t r _ t a i l : N	1838, 1864
sys commands:	
\ s y s _ i f _ s h e l l : T F	1822
\ s y s _ s h e l l _ n o w : n	1849
<b>T</b>	
TeX and L <sup>A</sup> T <sub>E</sub> X 2 <sub>ε</sub> commands:	
\ @ c c l v	2501, 2503, 2511
\ @ i f l @ t @ r	50, 52
\ @ m a k e c o l @ h o o k	2495
\ c u r r e n t @ c o l o r	442, 446, 450, 460, 476
\ s p e c i a l	2
TeX commands:	
\ t e x _ a f t e r a s s i g n m e n t : D	2147
\ t e x _ b a s e l i n e s k i p : D	2472
\ t e x _ e n d i n p u t : D	44
\ t e x _ g l o b a l : D	2859, 2876, 2890, 2897, 2904
\ t e x _ i m m e d i a t e : D	1790, 2781, 2784, 2812, 2815
\ t e x _ l u a t e x v e r s i o n : D	2888, 2916
\ t e x _ p d f a n n o t : D	2636
\ t e x _ p d f c a t a l o g : D	2742
\ t e x _ p d f c o l o r s t a c k : D	506, 516
\ t e x _ p d f c o l o r s t a c k i n i t : D	494
\ t e x _ p d f c o m p r e s s l e v e l : D	2864
\ t e x _ p d f d e s t : D	2705, 2728
\ t e x _ p d f e n d l i n k : D	2675
\ t e x _ p d f e x t e n s i o n : D	92, 103, 113, 122, 131,
	503, 513, 2633, 2661, 2672, 2702, 2725, 2739, 2749, 2760, 2781, 2812
\ t e x _ p d f f e e d b a c k : D	491, 2647, 2682, 2769, 2839, 2850
\ t e x _ p d f i n f o : D	2752
\ t e x _ p d f l a s t a n n o t : D	2650
\ t e x _ p d f l a s t l i n k : D	2685
\ t e x _ p d f l a s t o b j : D	2772, 2842
\ t e x _ p d f l a s t x i m a g e : D	1785, 1809
\ t e x _ p d f l a s t x i m a g e p a g e s : D	1875
\ t e x _ p d f l i n k m a r g i n : D	2695
\ t e x _ p d f l i t e r a l : D	95, 106
\ t e x _ p d f m a j o r v e r s i o n : D	2895, 2897, 2921, 2922
\ t e x _ p d f m i n o r v e r s i o n : D	2909, 2933
\ t e x _ p d f o b j : D	2763, 2784, 2815
\ t e x _ p d f o b j c o m p r e s s l e v e l : D	2881
\ t e x _ p d f p a g e r e f : D	2853
\ t e x _ p d f r e f x i m a g e : D	1809, 1816
\ t e x _ p d f r e s t o r e : D	125
\ t e x _ p d f s a v e : D	116
\ t e x _ p d f s e t m a t r i x : D	134
\ t e x _ p d f s t a r t l i n k : D	2664
\ t e x _ p d f v a r i a b l e : D	2692, 2861, 2878, 2890, 2906, 2917, 2930
\ t e x _ p d f x i m a g e : D	1790, 1873
\ t e x _ s p a c e f a c t o r : D	2483, 2492
\ t e x _ s p e c i a l : D	46
\ t e x _ t h e : D	1785, 2917, 2922, 2928
\ t e x _ v s s : D	2568, 2575, 3085, 3104
\ t e x _ X e T e X p d f f i l e : D	2005, 2051
\ t e x _ X e T e X p d f p a g e c o u n t : D	2061
\ t e x _ X e T e X p i c f i l e : D	1996
TeXcolorseparation	3312
\ t e x t w i d t h	2467
tl commands:	
\ c _ s p a c e _ t l	279, 284, 287, 581, 586, 624, 727, 801, 1011, 1558, 1728, 1729, 1730, 1731, 1911, 1912, 1913, 1914, 1962, 1965, 1967, 1968, 1969, 1970, 2031, 2053, 2180, 2181, 2182, 2183, 2423, 2652, 2687, 2844, 2855, 3013, 3035
\ t l _ c l e a r : N	1750, 1758, 1765, 1893, 1901, 1995, 2003, 2160, 2167
\ t l _ g c l e a r : N	1596, 1632
\ t l _ g s e t : N n	1555, 2378
\ t l _ i f _ b l a n k : n T F	496, 579, 675, 692, 699, 717, 843, 925, 2030, 2116
\ t l _ i f _ e m p t y : N T F	1558, 1753, 1797, 1805, 1932, 1936, 1963, 1978, 2018
\ t l _ i f _ e m p t y : n T F	937, 1652
\ t l _ i f _ e m p t y _ p : N	1793, 1975
\ t l _ i f _ h e a d _ i s _ s p a c e : n T F	442

