

# 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}{2021-05-07}{i}
5   {L3 backend support: dvipdfmx}
6 </dvipdfmx>
7 <*dvips>
8   {l3backend-dvips.def}{2021-05-07}{i}
9   {L3 backend support: dvips}
10 </dvips>
11 <*dvisvgm>
12   {l3backend-dvisvgm.def}{2021-05-07}{i}
13   {L3 backend support: dvisvgm}
14 </dvisvgm>
15 <*luatex>
16   {l3backend-luatex.def}{2021-05-07}{i}
17   {L3 backend support: PDF output (LuaTeX)}
18 </luatex>
19 <*pdftex>
20   {l3backend-pdftex.def}{2021-05-07}{i}
21   {L3 backend support: PDF output (pdfTeX)}
22 </pdftex>
23 <*xetex>
24   {l3backend-xetex.def}{2021-05-07}{i}
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 { \cs_new_eq:NN \__kernel_backend_first_shipout:n \AtBeginDvi }
52 { \cs_new_eq:NN \__kernel_backend_first_shipout:n \use:n }

```

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

## 1.1 dvips backend

```

53 <*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.

```

54 \cs_new_protected:Npn \__kernel_backend_literal_postscript:n #1
55 { \__kernel_backend_literal:n { ps:: #1 } }
56 \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 `"`).

```

57 \cs_new_protected:Npn \__kernel_backend_postscript:n #1
58 { \__kernel_backend_literal:n { ps: SDict ~ begin ~ #1 ~ end } }
59 \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.

```
60 \bool_if:NT \g__kernel_backend_header_bool
61 {
62   \_kernel_backend_first_shipout:n
63   { \_kernel_backend_literal:n { header = l3backend-dvips.pro } }
64 }
```

`\_kernel_backend_align_begin:`

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.

```
65 \cs_new_protected:Npn \_kernel_backend_align_begin:
66 {
67   \_kernel_backend_literal:n { ps::[begin] }
68   \_kernel_backend_literal_postscript:n { currentpoint }
69   \_kernel_backend_literal_postscript:n { currentpoint-translate }
70 }
71 \cs_new_protected:Npn \_kernel_backend_align_end:
72 {
73   \_kernel_backend_literal_postscript:n { neg-exch-neg-exch-translate }
74   \_kernel_backend_literal:n { ps::[end] }
75 }
```

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

`\_kernel_backend_scope_begin:`

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.

```
76 \cs_new_protected:Npn \_kernel_backend_scope_begin:
77 { \_kernel_backend_literal:n { ps:gsave } }
78 \cs_new_protected:Npn \_kernel_backend_scope_end:
79 { \_kernel_backend_literal:n { ps:grestore } }
```

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

```
80 </dvips>
```

## 1.2 LuaTeX and pdfTeX backends

```
81 <*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.

`\_kernel_backend_literal_pdf:n`

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).

`\_kernel_backend_literal_pdf:x`

```
82 \cs_new_protected:Npn \_kernel_backend_literal_pdf:n #1
83 {
```

```

84 <*luatex>
85   \tex_pdfextension:D literal
86 </luatex>
87 <*pdftex>
88   \tex_pdfliteral:D
89 </pdftex>
90   { \exp_not:n {#1} }
91 }
92 \cs_generate_variant:Nn \__kernel_backend_literal_pdf:n { x }

```

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

`\__kernel_backend_literal_page:n` Page literals are pretty simple. To avoid an expansion, we write out by hand.

```

93 \cs_new_protected:Npn \__kernel_backend_literal_page:n #1
94 {
95 <*luatex>
96   \tex_pdfextension:D literal ~
97 </luatex>
98 <*pdftex>
99   \tex_pdfliteral:D
100 </pdftex>
101   page { \exp_not:n {#1} }
102 }

```

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

`\__kernel_backend_scope_begin:` Higher-level interfaces for saving and restoring the graphic state.

```

\__kernel_backend_scope_end:
103 \cs_new_protected:Npn \__kernel_backend_scope_begin:
104 {
105 <*luatex>
106   \tex_pdfextension:D save \scan_stop:
107 </luatex>
108 <*pdftex>
109   \tex_pdfsave:D
110 </pdftex>
111 }
112 \cs_new_protected:Npn \__kernel_backend_scope_end:
113 {
114 <*luatex>
115   \tex_pdfextension:D restore \scan_stop:
116 </luatex>
117 <*pdftex>
118   \tex_pdfrestore:D
119 </pdftex>
120 }

```

(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  
`\__kernel_backend_matrix:x` pdfTeX and LuaTeX in direct PDF output mode there is a primitive for this, which only needs the rotation/scaling/skew part.

```

121 \cs_new_protected:Npn \__kernel_backend_matrix:n #1
122 {
123 <*luatex>
124   \tex_pdfextension:D setmatrix

```

```

125 </luatex>
126 <*pdftex>
127   \tex_pdfsetmatrix:D
128 </pdftex>
129   { \exp_not:n {#1} }
130 }
131 \cs_generate_variant:Nn \_kernel_backend_matrix:n { x }

```

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

```
132 </luatex | pdftex>
```

### 1.3 dvipdfmx backend

```
133 <*dvipdfmx | xetex>
```

The `dvipdfmx` shares code with the PDF mode one (using the common section to this file) but also with `XqTeX`. 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 `XqTeX` 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
```

```

134 \cs_new_protected:Npn \_kernel_backend_literal_pdf:n #1
135   { \_kernel_backend_literal:n { pdf:literal~ #1 } }
136 \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!

```

137 \cs_new_protected:Npn \_kernel_backend_literal_page:n #1
138   { \_kernel_backend_literal:n { pdf:literal~direct~ #1 } }

```

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

```
\_kernel_backend_scope_begin:
\_kernel_backend_scope_end:
```

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

```

139 \cs_new_protected:Npn \_kernel_backend_scope_begin:
140   { \_kernel_backend_literal:n { x:gsave } }
141 \cs_new_protected:Npn \_kernel_backend_scope_end:
142   { \_kernel_backend_literal:n { x:grestore } }

```

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

```
143 <@@=sys>
```

```
\c__kernel_sys_dvipdfmx_version_int
```

A short excursion into the `sys` module to set up the backend version information.

```

144 \group_begin:
145   \cs_set:Npn \_sys_tmp:w #1 Version ~ #2 ~ #3 \q_stop {#2}
146   \sys_get_shell:nnNTF { extractbb---version }
147     { \char_set_catcode_space:n { ‘\ } }
148     \l__sys_internal_tl
149     {
150       \int_const:Nn \c__kernel_sys_dvipdfmx_version_int
151         {
152           \exp_after:wN \_sys_tmp:w \l__sys_internal_tl
153             \q_stop
154         }

```

```

155     }
156     { \int_const:Nn \c__kernel_sys_dvipdfmx_version_int { 0 } }
157 \group_end:

```

(End definition for \c\_\_kernel\_sys\_dvipdfmx\_version\_int.)

```

158 <@@=)
159 </dvipdfmx | xetex>

```

## 1.4 dvisvgm backend

```

160 <*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.

```

161 \cs_new_protected:Npn \__kernel_backend_literal_svg:n #1
162 { \__kernel_backend_literal:n { dvisvgm:raw~ #1 { ?nl } } }
163 \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.

```

164 \int_new:N \g__kernel_backend_scope_int
165 \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.

```

166 \cs_new_protected:Npn \__kernel_backend_scope_begin:
167 {
168   \__kernel_backend_literal_svg:n { <g> }
169   \int_set_eq:NN
170     \l__kernel_backend_scope_int
171     \g__kernel_backend_scope_int
172   \group_begin:
173     \int_gset:Nn \g__kernel_backend_scope_int { 1 }
174 }
175 \cs_new_protected:Npn \__kernel_backend_scope_end:
176 {
177   \prg_replicate:nn
178     { \g__kernel_backend_scope_int }
179     { \__kernel_backend_literal_svg:n { </g> } }
180   \group_end:
181   \int_set_eq:NN
182     \g__kernel_backend_scope_int
183     \l__kernel_backend_scope_int
184 }
185 \cs_new_protected:Npn \__kernel_backend_scope_begin:n #1

```

```

186 {
187   \__kernel_backend_literal_svg:n { <g ~ #1 > }
188   \int_set_eq:NN
189     \l__kernel_backend_scope_int
190     \g__kernel_backend_scope_int
191   \group_begin:
192     \int_gset:Nn \g__kernel_backend_scope_int { 1 }
193 }
194 \cs_generate_variant:Nn \__kernel_backend_scope_begin:n { x }
195 \cs_new_protected:Npn \__kernel_backend_scope:n #1
196 {
197   \__kernel_backend_literal_svg:n { <g ~ #1 > }
198   \int_gincr:N \g__kernel_backend_scope_int
199 }
200 \cs_generate_variant:Nn \__kernel_backend_scope:n { x }

```

(End definition for \\_\_kernel\_backend\_scope\_begin: and others.)

```

201 </divisvgn>
202 </package>

```

## 2 l3backend-box Implementation

```

203 <*package>
204 <@@=box>

```

### 2.1 dvips backend

```

205 <*dvips>

```

\\_\_box\_backend\_clip:N The dvips backend scales all absolute dimensions based on the output resolution selected and any T<sub>E</sub>X 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.

```

206 \cs_new_protected:Npn \__box_backend_clip:N #1
207 {
208   \__kernel_backend_scope_begin:
209   \__kernel_backend_align_begin:
210   \__kernel_backend_literal_postscript:n { matrix-currentmatrix }
211   \__kernel_backend_literal_postscript:n
212     { Resolution~72~div~VResolution~72~div~scale }
213   \__kernel_backend_literal_postscript:n { DVImag-dup~scale }
214   \__kernel_backend_literal_postscript:x
215     {
216       0 ~
217       \dim_to_decimal_in_bp:n { \box_dp:N #1 } ~
218       \dim_to_decimal_in_bp:n { \box_wd:N #1 } ~
219       \dim_to_decimal_in_bp:n { -\box_ht:N #1 - \box_dp:N #1 } ~
220       rectclip
221     }
222   \__kernel_backend_literal_postscript:n { setmatrix }
223   \__kernel_backend_align_end:
224   \hbox_overlap_right:n { \box_use:N #1 }

```

```

225     \__kernel_backend_scope_end:
226     \skip_horizontal:n { \box_wd:N #1 }
227 }

```

(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.

```

228 \cs_new_protected:Npn \__box_backend_rotate:Nn #1#2
229 { \exp_args:Nmf \__box_backend_rotate_aux:Nn #1 { \fp_eval:n {#2} } }
230 \cs_new_protected:Npn \__box_backend_rotate_aux:Nn #1#2
231 {
232     \__kernel_backend_scope_begin:
233     \__kernel_backend_align_begin:
234     \__kernel_backend_literal_postscript:x
235     {
236         \fp_compare:nNnTF {#2} = \c_zero_fp
237         { 0 }
238         { \fp_eval:n { round ( -(#2) , 5 ) } } ~
239         rotate
240     }
241     \__kernel_backend_align_end:
242     \box_use:N #1
243     \__kernel_backend_scope_end:
244 }

```

(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.

```

245 \cs_new_protected:Npn \__box_backend_scale:Nnn #1#2#3
246 {
247     \__kernel_backend_scope_begin:
248     \__kernel_backend_align_begin:
249     \__kernel_backend_literal_postscript:x
250     {
251         \fp_eval:n { round ( #2 , 5 ) } ~
252         \fp_eval:n { round ( #3 , 5 ) } ~
253         scale
254     }
255     \__kernel_backend_align_end:
256     \hbox_overlap_right:n { \box_use:N #1 }
257     \__kernel_backend_scope_end:
258 }

```

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

```

259 </dvips>

```

## 2.2 LuaTeX and pdfTeX backends

```

260 <*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.

```

261 \cs_new_protected:Npn \__box_backend_clip:N #1
262 {
263   \__kernel_backend_scope_begin:
264   \__kernel_backend_literal_pdf:x
265   {
266     0~
267     \dim_to_decimal_in_bp:n { -\box_dp:N #1 } ~
268     \dim_to_decimal_in_bp:n { \box_wd:N #1 } ~
269     \dim_to_decimal_in_bp:n { \box_ht:N #1 + \box_dp:N #1 } ~
270     re~W~n
271   }
272   \hbox_overlap_right:n { \box_use:N #1 }
273   \__kernel_backend_scope_end:
274   \skip_horizontal:n { \box_wd:N #1 }
275 }

```

(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.

```

276 \cs_new_protected:Npn \__box_backend_rotate:Nn #1#2
277 { \exp_args:Nnf \__box_backend_rotate_aux:Nn #1 { \fp_eval:n {#2} } }
278 \cs_new_protected:Npn \__box_backend_rotate_aux:Nn #1#2
279 {
280   \__kernel_backend_scope_begin:
281   \box_set_wd:Nn #1 { Opt }
282   \fp_set:Nn \l__box_backend_cos_fp { round ( cosd ( #2 ) , 5 ) }
283   \fp_compare:nNnTF \l__box_backend_cos_fp = \c_zero_fp
284     { \fp_zero:N \l__box_backend_cos_fp }
285   \fp_set:Nn \l__box_backend_sin_fp { round ( sind ( #2 ) , 5 ) }
286   \__kernel_backend_matrix:x
287   {
288     \fp_use:N \l__box_backend_cos_fp \c_space_tl
289     \fp_compare:nNnTF \l__box_backend_sin_fp = \c_zero_fp
290       { 0~0 }
291       {
292         \fp_use:N \l__box_backend_sin_fp
293         \c_space_tl
294         \fp_eval:n { -\l__box_backend_sin_fp }
295       }
296     \c_space_tl
297     \fp_use:N \l__box_backend_cos_fp
298   }
299   \box_use:N #1
300   \__kernel_backend_scope_end:
301 }

```

```

302 \fp_new:N \l__box_backend_cos_fp
303 \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.

```

304 \cs_new_protected:Npn \__box_backend_scale:Nnn #1#2#3
305 {
306   \__kernel_backend_scope_begin:
307   \__kernel_backend_matrix:x
308   {
309     \fp_eval:n { round ( #2 , 5 ) } ~
310     0~0~
311     \fp_eval:n { round ( #3 , 5 ) }
312   }
313   \hbox_overlap_right:n { \box_use:N #1 }
314   \__kernel_backend_scope_end:
315 }

```

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

```

316 </luatex | pdftex>

```

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

```

317 <*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.

```

318 \cs_new_protected:Npn \__box_backend_clip:N #1
319 {
320   \__kernel_backend_scope_begin:
321   \__kernel_backend_literal_pdf:x
322   {
323     0~
324     \dim_to_decimal_in_bp:n { -\box_dp:N #1 } ~
325     \dim_to_decimal_in_bp:n { \box_wd:N #1 } ~
326     \dim_to_decimal_in_bp:n { \box_ht:N #1 + \box_dp:N #1 } ~
327     re~W~n
328   }
329   \hbox_overlap_right:n { \box_use:N #1 }
330   \__kernel_backend_scope_end:
331   \skip_horizontal:n { \box_wd:N #1 }
332 }

```

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

`\__box_backend_rotate:Nn`  
`\__box_backend_rotate_aux: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.

```

333 \cs_new_protected:Npn \__box_backend_rotate:Nn #1#2
334 { \exp_args:Nnf \__box_backend_rotate_aux:Nn #1 { \fp_eval:n {#2} } }

```

```

335 \cs_new_protected:Npn \__box_backend_rotate_aux:Nn #1#2
336 {
337   \__kernel_backend_scope_begin:
338   \__kernel_backend_literal:x
339   {
340     x:rotate~
341     \fp_compare:nNnTF {#2} = \c_zero_fp
342     { 0 }
343     { \fp_eval:n { round ( #2 , 5 ) } } }
344   }
345   \box_use:N #1
346   \__kernel_backend_scope_end:
347 }

```

(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.

```

348 \cs_new_protected:Npn \__box_backend_scale:Nnn #1#2#3
349 {
350   \__kernel_backend_scope_begin:
351   \__kernel_backend_literal:x
352   {
353     x:scale~
354     \fp_eval:n { round ( #2 , 5 ) } ~
355     \fp_eval:n { round ( #3 , 5 ) }
356   }
357   \hbox_overlap_right:n { \box_use:N #1 }
358   \__kernel_backend_scope_end:
359 }

```

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

```
360 </dviPDFmx | xetex>
```

## 2.4 dvisvgm backend

```
361 <*dvisvgm>
```

`\__box_backend_clip:N`  
`\g__box_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  $\TeX$  box and keep the reference point the same!

```

362 \cs_new_protected:Npn \__box_backend_clip:N #1
363 {
364   \int_gincr:N \g__box_clip_path_int
365   \__kernel_backend_literal_svg:x
366   { < clipPath-id = " l3cp \int_use:N \g__box_clip_path_int " > }
367   \__kernel_backend_literal_svg:x
368   {
369     <
370     path ~ d =

```

```

371         "
372         M ~ 0 ~
373             \dim_to_decimal:n { -\box_dp:N #1 } ~
374         L ~ \dim_to_decimal:n { \box_wd:N #1 } ~
375             \dim_to_decimal:n { -\box_dp:N #1 } ~
376         L ~ \dim_to_decimal:n { \box_wd:N #1 } ~
377             \dim_to_decimal:n { \box_ht:N #1 + \box_dp:N #1 } ~
378         L ~ 0 ~
379             \dim_to_decimal:n { \box_ht:N #1 + \box_dp:N #1 } ~
380         Z
381     "
382     />
383 }
384 \__kernel_backend_literal_svg:n
385 { < /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  $\text{T}_{\text{E}}\text{X}$  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  $\text{T}_{\text{E}}\text{X}$  box.

```

386 \__kernel_backend_scope_begin:n
387 {
388     transform =
389     "
390         translate ( { ?x } , { ?y } ) ~
391         scale ( 1 , -1 )
392     "
393 }
394 \__kernel_backend_scope:x
395 {
396     clip-path =
397     "url ( \c_hash_str l3cp \int_use:N \g__box_clip_path_int ) "
398 }
399 \__kernel_backend_scope:n
400 {
401     transform =
402     "
403         scale ( -1 , 1 ) ~
404         translate ( { ?x } , { ?y } ) ~
405         scale ( -1 , -1 )
406     "
407 }
408 \box_use:N #1
409 \__kernel_backend_scope_end:
410 }
411 \int_new:N \g__box_clip_path_int

```

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

$\backslash\_box\_backend\_rotate:Nn$  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.

```

412 \cs_new_protected:Npn \__box_backend_rotate:Nn #1#2

```

```

413 {
414   \__kernel_backend_scope_begin:x
415   {
416     transform =
417     "
418       rotate
419       ( \fp_eval:n { round ( -(#2) , 5 ) } , ~ { ?x } , ~ { ?y } )
420     "
421   }
422   \box_use:N #1
423   \__kernel_backend_scope_end:
424 }

```

(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).

```

425 \cs_new_protected:Npn \__box_backend_scale:Nnn #1#2#3
426 {
427   \__kernel_backend_scope_begin:x
428   {
429     transform =
430     "
431       translate ( { ?x } , { ?y } ) ~
432       scale
433       (
434         \fp_eval:n { round ( -#2 , 5 ) } ,
435         \fp_eval:n { round ( -#3 , 5 ) }
436       ) ~
437       translate ( { ?x } , { ?y } ) ~
438       scale ( -1 )
439     "
440   }
441   \hbox_overlap_right:n { \box_use:N #1 }
442   \__kernel_backend_scope_end:
443 }

```

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

```

444 </divisvgn>
445 </package>

```

### 3 l3backend-color Implementation

```

446 <*package>
447 <@@=color>

```

Color support is split into parts: collecting data from L<sup>A</sup>T<sub>E</sub>X<sub>2 $\epsilon$</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>A</sub>T<sub>E</sub>X 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>A</sub>T<sub>E</sub>X 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

```
448 <*dvisvgn | 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 mainly be taken as-is with the exception spot ones (here we need a model and a tint).  
`\_color_backend_pickup:w` The x-type expansion is there to cover the case where xcolor is in use.

```
449 \cs_new_protected:Npn \_color_backend_pickup:N #1 { }
450 \cs_if_exist:cT { ver@color.sty }
451 {
452   \cs_set_protected:Npn \_color_backend_pickup:N #1
453   {
454     \exp_args:NW \tl_if_head_is_space:nTF \current@color
455     {
456       \tl_set:Nx #1
457       {
458         { \exp_after:wN \use:n \current@color }
459         { 1 }
460       }
461     }
462     {
463       \exp_last_unbraced:Nx \_color_backend_pickup:w
464       { \current@color } \s_color_stop #1
465     }
466   }
467   \cs_new_protected:Npn \_color_backend_pickup:w #1 ~ #2 \s_color_stop #3
468   { \tl_set:Nn #3 { {#1} {#2} } }
469 }
```

(End definition for `\_color_backend_pickup:N` and `\_color_backend_pickup:w`.)

```
470 </dvisvgn | dvipdfmx | dvips | xetex>
```

### 3.1.2 Lua<sub>T</sub><sub>E</sub>X and pdf<sub>T</sub><sub>E</sub>X

```
471 <*luatex | pdftex>
```

`\_color_backend_pickup:N` The current color in driver-dependent format: pick up the package-mode data if available. We end up converting back and forward in this route as we store our color data in dvips format. The `\current@color` needs to be x-expanded before `\_color_backend_pickup:w` breaks it apart, because for instance xcolor sets it to be instructions to generate a color

```
472 \cs_new_protected:Npn \_color_backend_pickup:N #1 { }
473 \cs_if_exist:cT { ver@color.sty }
474 {
475   \cs_set_protected:Npn \_color_backend_pickup:N #1
476   {
477     \exp_last_unbraced:Nx \_color_backend_pickup:w
478     { \current@color } ~ 0 ~ 0 ~ 0 \s_color_stop #1
479   }
480   \cs_new_protected:Npn \_color_backend_pickup:w
481   #1 ~ #2 ~ #3 ~ #4 ~ #5 ~ #6 \s_color_stop #7
482   {
483     \str_if_eq:nnTF {#2} { g }
```

```

484     { \tl_set:Nn #7 { { gray } {#1} } }
485     {
486       \str_if_eq:nnTF {#4} { rg }
487       { \tl_set:Nn #7 { { rgb } { #1 ~ #2 ~ #3 } } }
488       {
489         \str_if_eq:nnTF {#5} { k }
490         { \tl_set:Nn #7 { { cmyk } { #1 ~ #2 ~ #3 ~ #4 } } }
491         {
492           \str_if_eq:nnTF {#2} { cs }
493           {
494             \tl_set:Nx #7 { { \use:n #1 } { #5 } }
495           }
496           {
497             \tl_set:Nn #7 { { gray } { 0 } }
498           }
499         }
500       }
501     }
502   }
503 }

```

(End definition for `\_color_backend_pickup:N` and `\_color_backend_pickup:w`.)

```
504 </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. The exact form depends on the engine, and for `dvipdfmx/XYTeX` the backend version.

### 3.2.1 Common code

```
505 <*dvipdfmx | luatex | pdftex | xetex>
```

`\_color_backend_stack_int` pdfTeX, LuaTeX and recent (x)dvipdfmx have multiple stacks available, and to track which one is in use a variable is required.

```
506 \int_new:N \_color_backend_stack_int
```

(End definition for `\_color_backend_stack_int`.)

```
507 </dvipdfmx | luatex | pdftex | xetex>
```

### 3.2.2 dvipdfmx/X<sub>Y</sub>TeX

```
508 <*dvipdfmx | xetex>
```

`\_kernel_color_backend_stack_init:Nnn` In (x)dvipdfmx, the base color stack is not set up, so we have to force that, as well as providing a mechanism more generally.

```
\g__color_backend_stack_int
\_color_backend_main_stack_int
```

```

509 \int_compare:nNnTF \_kernel_sys_dvipdfmx_version_int < { 20201111 }
510 { \cs_new_protected:Npn \_kernel_color_backend_stack_init:Nnn #1#2#3 { } }
511 {
512   \int_new:N \g__color_backend_stack_int
513   \cs_new_protected:Npx \_kernel_color_backend_stack_init:Nnn #1#2#3
514   {
515     \int_gincr:N \exp_not:N \g__color_backend_stack_int

```

```

516     \int_const:Nn #1 { \exp_not:N \g__color_backend_stack_int }
517     \use:x
518     {
519         \__kernel_backend_first_shipout:n
520         {
521             \__kernel_backend_literal:n
522             {
523                 pdfcolorstackinit ~
524                 \exp_not:N \int_use:N \exp_not:N \g__color_backend_stack_int
525                 \c_space_tl
526                 \exp_not:N \tl_if_blank:nF {#2} { #2 ~ }
527                 (#3)
528             }
529         }
530     }
531 }
532 \cs_if_exist:cTF { main@pdfcolorstack }
533 {
534     \int_set:Nn \l__color_backend_stack_int
535     { \int_use:c { main@pdfcolorstack } }
536 }
537 {
538     \__kernel_color_backend_stack_init:Nnn \c__color_backend_main_stack_int
539     { page ~ direct } { 0 ~ g ~ 0 ~ G }
540     \int_set_eq:NN \l__color_backend_stack_int
541     \c__color_backend_main_stack_int
542     \int_const:cn { main@pdfcolorstack } { \c__color_backend_main_stack_int }
543 }

```

The backend automatically restores the stack color from the “classical” approach (`pdf:bcolor`) after a scope. That will be an issue for us, so we manually ensure that the one we are using is inserted.

```

544     \cs_gset_protected:Npn \__kernel_backend_scope_end:
545     {
546         \__kernel_backend_literal:n { x:grestore }
547         \__kernel_backend_literal:n
548         { pdfcolorstack ~ \g__color_backend_stack_int current }
549     }
550 }

```

(End definition for `\__kernel_color_backend_stack_init:Nnn`, `\g__color_backend_stack_int`, and `\c__color_backend_main_stack_int`.)

```

\__kernel_color_backend_stack_push:nn
\__kernel_color_backend_stack_push:nx
\__kernel_color_backend_stack_pop:n

```

Simple enough but needs a version check.

```

551 \int_compare:nNnF \c__kernel_sys_dvipdfmx_version_int < { 20201111 }
552 {
553     \cs_new_protected:Npn \__kernel_color_backend_stack_push:nn #1#2
554     {
555         \__kernel_backend_literal:x
556         {
557             pdfcolorstack ~
558             \int_eval:n {#1} ~
559             push ~ (#2)
560         }
561     }

```



```

562 \cs_generate_variant:Nn \__kernel_color_backend_stack_push:nn { nx }
563 \cs_new_protected:Npn \__kernel_color_backend_stack_pop:n #1
564 {
565   \__kernel_backend_literal:x
566   {
567     pdfcolorstack ~
568     \int_eval:n {#1} ~
569     pop
570   }
571 }
572 }

```

(End definition for \\_\_kernel\_color\_backend\_stack\_push:nn and \\_\_kernel\_color\_backend\_stack\_pop:n.)

```
573 </divpdfmx | xetex>
```

### 3.2.3 LuaTeX and pdfTeX

```
574 <*luatex | pdftex>
```

\\_\_kernel\_color\_backend\_stack\_init:Nnn

```

575 \cs_new_protected:Npn \__kernel_color_backend_stack_init:Nnn #1#2#3
576 {
577   \int_const:Nn #1
578   {
579     <*luatex>
580     \tex_pdffeedback:D colorstackinit ~
581     </luatex>
582     <*pdftex>
583     \tex_pdfcolorstackinit:D
584     </pdftex>
585     \tl_if_blank:nF {#2} { #2 ~ }
586     {#3}
587   }
588 }

```

(End definition for \\_\_kernel\_color\_backend\_stack\_init:Nnn.)

\\_\_kernel\_color\_backend\_stack\_push:nn

\\_\_kernel\_color\_backend\_stack\_push:nx

\\_\_kernel\_color\_backend\_stack\_pop:n

```

589 \cs_new_protected:Npn \__kernel_color_backend_stack_push:nn #1#2
590 {
591   <*luatex>
592   \tex_pdfextension:D colorstack ~
593   </luatex>
594   <*pdftex>
595   \tex_pdfcolorstack:D
596   </pdftex>
597   \int_eval:n {#1} ~ push ~ {#2}
598 }
599 \cs_generate_variant:Nn \__kernel_color_backend_stack_push:nn { nx }
600 \cs_new_protected:Npn \__kernel_color_backend_stack_pop:n #1
601 {
602   <*luatex>
603   \tex_pdfextension:D colorstack ~
604   </luatex>

```

```

605 <*pdftex>
606   \tex_pdfcolorstack:D
607 </pdftex>
608   \int_eval:n {#1} ~ pop \scan_stop:
609 }

```

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

```
610 </luatex | pdftex>
```

### 3.3 General color

#### 3.3.1 dvips-style

```
611 <*dvips | dvisvgm>
```

Push the data to the stack. In the case of `dvips` also saves the drawing color in raw PostScript.

```

\__color_backend_select_cmyk:n
\__color_backend_select_gray:n
\__color_backend_select_rgb:n
\__color_backend_select:n
\__color_backend_reset:
color.sc
612 \cs_new_protected:Npn \__color_backend_select_cmyk:n #1
613   { \__color_backend_select:n { cmyk ~ #1 } }
614 \cs_new_protected:Npn \__color_backend_select_gray:n #1
615   { \__color_backend_select:n { gray ~ #1 } }
616 \cs_new_protected:Npn \__color_backend_select_rgb:n #1
617   { \__color_backend_select:n { rgb ~ #1 } }
618 \cs_new_protected:Npn \__color_backend_select:n #1
619   {
620     \__kernel_backend_literal:n { color-push~ #1 }
621 <*dvips>
622     \__kernel_backend_postscript:n { /color.sc ~ { } ~ def }
623 </dvips>
624     \group_insert_after:N \__color_backend_reset:
625   }
626 \cs_new_protected:Npn \__color_backend_reset:
627   { \__kernel_backend_literal:n { color-pop } }

```

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

```
628 </dvips | dvisvgm>
```

#### 3.3.2 LuaTeX and pdfTeX

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

```

\l__color_backend_fill_tl
\l__color_backend_stroke_tl
630 \tl_new:N \l__color_backend_fill_tl
631 \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.

```

\__color_backend_select_cmyk:n
\__color_backend_select_gray:n
\__color_backend_select_rgb:n
\__color_backend_select:nn
\__color_backend_reset:
632 \cs_new_protected:Npn \__color_backend_select_cmyk:n #1
633   { \__color_backend_select:nn { #1 ~ k } { #1 ~ K } }
634 \cs_new_protected:Npn \__color_backend_select_gray:n #1
635   { \__color_backend_select:nn { #1 ~ g } { #1 ~ G } }
636 \cs_new_protected:Npn \__color_backend_select_rgb:n #1

```

```

637 { \_color_backend_select:nn { #1 ~ rg } { #1 ~ RG } }
638 \cs_new_protected:Npn \_color_backend_select:nn #1#2
639 {
640   \tl_set:Nn \l__color_backend_fill_tl {#1}
641   \tl_set:Nn \l__color_backend_stroke_tl {#2}
642   \__kernel_color_backend_stack_push:nn \l__color_backend_stack_int { #1 ~ #2 }
643   \group_insert_after:N \_color_backend_reset:
644 }
645 \cs_new_protected:Npn \_color_backend_reset:
646 { \__kernel_color_backend_stack_pop:n \l__color_backend_stack_int }

```

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

```

647 </dviptdpmx | luatex | pdftex | xetex>

```

### 3.3.3 dvipdpmx/X<sub>g</sub>T<sub>E</sub>X

```

648 <*dvipdpmx | xetex>

```

These backends have the most possible approaches: it recognises both dvips-based color specials and it’s own format, plus one can include PDF statements directly. Recent releases also have a color stack approach similar to pdfT<sub>E</sub>X. Of the stack methods, the dedicated the most versatile is the latter as it can cover all of the use cases we have. Thus it is used in preference to the dvips-style interface or the “native” color specials (which have only one stack).

Push the data to the stack.

```

\_color_backend_select_cmyk:n
\_color_backend_select_gray:n
\_color_backend_select_rgb:n
\_color_backend_reset:
649 \int_compare:nNnT \c__kernel_sys_dvipdpmx_version_int < { 20201111 }
650 {
651   \cs_gset_protected:Npn \_color_backend_select_cmyk:n #1
652   {
653     \__kernel_backend_literal:n { pdf: bc ~ [#1] }
654     \group_insert_after:N \_color_backend_reset:
655   }
656   \cs_gset_eq:NN \_color_backend_select_gray:n \_color_backend_select_cmyk:n
657   \cs_gset_eq:NN \_color_backend_select_rgb:n \_color_backend_select_cmyk:n
658   \cs_gset_protected:Npn \_color_backend_reset:
659   { \__kernel_backend_literal:n { pdf: ec } }
660 }

```

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

```

661 </dviptdpmx | xetex>

```

## 3.4 Separations

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

```

662 <*dvips>

```

```

\_color_backend_select_separation:nn
\_color_backend_select_devicen:nn
663 \cs_new_protected:Npn \_color_backend_select_separation:nn #1#2
664 { \_color_backend_select:n { separation ~ #1 ~ #2 } }
665 \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_separation_init:nnnnn
\_color_backend_separation_init:nxxnn
\_color_backend_separation_init_aux:nnnnn
lor_backend_separation_init_/DeviceCMYK:nnn
lor_backend_separation_init_/DeviceGray:nnn
olor_backend_separation_init_/DeviceRGB:nnn
\_color_backend_separation_init_Device:Nn
\_color_backend_separation_init:nnn
\_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:nnn

```

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.

```

666 \cs_new_protected:Npx \_color_backend_separation_init:nnnnn #1#2#3#4#5
667 {
668   \bool_if:NT \g__kernel_backend_header_bool
669   {
670     \_kernel_backend_first_shipout:n
671     {
672       \exp_not:N \_color_backend_separation_init_aux:nnnnn
673       {#1} {#2} {#3} {#4} {#5}
674     }
675   }
676 }
677 \cs_generate_variant:Nn \_color_backend_separation_init:nnnnn { nxx }
678 \cs_new_protected:Npn \_color_backend_separation_init_aux:nnnnn #1#2#3#4#5
679 {
680   \_kernel_backend_literal:e
681   {
682     !
683     TeXDict ~ begin ~
684     /color \int_use:N \g__color_model_int
685     {
686       [ ~
687         /Separation ~ ( \str_convert_pdfname:n {#1} ) ~
688         [ ~ #2 ~ ] ~
689         {
690           \cs_if_exist_use:cF { __color_backend_separation_init_ #2 :nnn }
691           { \_color_backend_separation_init:nnn }
692           {#3} {#4} {#5}
693         }
694       ] ~ setcolorspace
695     } ~ def ~
696   end
697 }
698 }
699 \cs_new:cpn { __color_backend_separation_init_ /DeviceCMYK :nnn } #1#2#3
700 { \_color_backend_separation_init_Device:Nn 4 {#3} }
701 \cs_new:cpn { __color_backend_separation_init_ /DeviceGray :nnn } #1#2#3
702 { \_color_backend_separation_init_Device:Nn 1 {#3} }
703 \cs_new:cpn { __color_backend_separation_init_ /DeviceRGB :nnn } #1#2#3
704 { \_color_backend_separation_init_Device:Nn 2 {#3} }
705 \cs_new:Npn \_color_backend_separation_init_Device:Nn #1#2
706 {
707   #2 ~
708   \prg_replicate:nn {#1}
709   { #1 ~ index ~ mul ~ #1 ~ 1 ~ roll ~ }
710   \int_eval:n { #1 + 1 } ~ -1 ~ roll ~ pop
711 }

```

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.

```

712 \cs_new:Npn \__color_backend_separation_init:nnn #1#2#3
713 {
714   \exp_args:Ne \__color_backend_separation_init:nnnn
715   { \__color_backend_separation_init_count:n {#2} }
716   {#1} {#2} {#3}
717 }
718 \cs_new:Npn \__color_backend_separation_init_count:n #1
719 { \int_eval:n { 0 \__color_backend_separation_init_count:w #1 ~ \s__color_stop } }
720 \cs_new:Npn \__color_backend_separation_init_count:w #1 ~ #2 \s__color_stop
721 {
722   +1
723   \tl_if_blank:nF {#2}
724   { \__color_backend_separation_init_count:w #2 \s__color_stop }
725 }

```

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.

```

726 \cs_new:Npn \__color_backend_separation_init:nnnn #1#2#3#4
727 {
728   \__color_backend_separation_init:w #3 ~ \s__color_stop #4 ~ \s__color_stop
729   \prg_replicate:nn {#1}
730   {
731     pop ~ 1 ~ index ~ neg ~ 1 ~ index ~ add ~
732     \int_eval:n { 3 * #1 } ~ index ~ mul ~
733     2 ~ index ~ add ~
734     \int_eval:n { 3 * #1 } ~ #1 ~ roll ~
735   }
736   \int_step_function:nnnN {#1} { -1 } { 1 }
737   \__color_backend_separation_init:n
738   \int_eval:n { 4 * #1 + 1 } ~ #1 ~ roll ~
739   \prg_replicate:nn { 3 * #1 + 1 } { pop ~ }
740   \tl_if_blank:nF {#2}
741   { \__color_backend_separation_init:nw {#1} #2 ~ \s__color_stop }
742 }
743 \cs_new:Npn \__color_backend_separation_init:w
744 #1 ~ #2 \s__color_stop #3 ~ #4 \s__color_stop
745 {
746   #1 ~ #3 ~ 0 ~
747   \tl_if_blank:nF {#2}
748   { \__color_backend_separation_init:w #2 \s__color_stop #4 \s__color_stop }
749 }
750 \cs_new:Npn \__color_backend_separation_init:n #1
751 { \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.

```

752 \cs_new:Npn \__color_backend_separation_init:nw #1#2 ~ #3 ~ #4 \s_color_stop
753 {
754   #2 ~ #3 ~
755   2 ~ index ~ 2 ~ index ~ lt ~
756   { ~ pop ~ exch ~ pop ~ } ~
757   { ~
758     2 ~ index ~ 1 ~ index ~ gt ~
759     { ~ exch ~ pop ~ exch ~ pop ~ } ~
760     { ~ pop ~ pop ~ } ~
761     ifelse ~
762   }
763   ifelse ~
764   #1 ~ 1 ~ roll ~
765   \tl_if_blank:nF {#4}
766   { \__color_backend_separation_init:nw {#1} #4 \s_color_stop }
767 }

```

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.

```

768 \cs_new_protected:Npn \__color_backend_separation_init_CIELAB:nnn #1#2#3
769 {
770   \__color_backend_separation_init:nxxxnn
771   {#2}
772   {
773     /CIEBasedABC ~
774     << ~
775     /RangeABC ~ [ ~ \c__color_model_range_CIELAB_tl \c_space_tl ] ~
776     /DecodeABC ~
777     [ ~
778     { ~ 16 ~ add ~ 116 ~ div ~ } ~ bind ~
779     { ~ 500 ~ div ~ } ~ bind ~
780     { ~ 200 ~ div ~ } ~ bind ~
781     ] ~
782     /MatrixABC ~ [ ~ 1 ~ 1 ~ 1 ~ 1 ~ 0 ~ 0 ~ 0 ~ 0 ~ -1 ~ ] ~
783     /DecodeLMN ~
784     [ ~
785     { ~
786     dup ~ 6 ~ 29 ~ div ~ ge ~
787     { ~ dup ~ dup ~ mul ~ mul ~ ~ } ~
788     { ~ 4 ~ 29 ~ div ~ sub ~ 108 ~ 841 ~ div ~ mul ~ } ~
789     ifelse ~
790     0.9505 ~ mul ~
791     } ~ bind ~
792     { ~
793     dup ~ 6 ~ 29 ~ div ~ ge ~
794     { ~ dup ~ dup ~ mul ~ mul ~ } ~
795     { ~ 4 ~ 29 ~ div ~ sub ~ 108 ~ 841 ~ div ~ mul ~ } ~
796     ifelse ~
797     } ~ bind ~
798     { ~
799     dup ~ 6 ~ 29 ~ div ~ ge ~

```

```

800         { ~ dup ~ dup ~ mul ~ mul ~ } ~
801         { ~ 4 ~ 29 ~ div ~ sub ~ 108 ~ 841 ~ div ~ mul ~ } ~
802         elseif ~
803         1.0890 ~ mul ~
804     } ~ bind
805 ] ~
806 /WhitePoint ~
807 [ ~ \tl_use:c { c__color_model_whitepoint_CIELAB_ #1 _tl } ~ ] ~
808 >>
809 }
810 { \c__color_model_range_CIELAB_tl }
811 { 100 ~ 0 ~ 0 }
812 {#3}
813 }

```

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

`\_color_backend_devicen_init:nmn` Trivial as almost all of the work occurs in the shared code.

```

814 \cs_new_protected:Npn \_color_backend_devicen_init:nmn #1#2#3
815 {
816     \__kernel_backend_literal:e
817     {
818         !
819         TeXDict ~ begin ~
820         /color \int_use:N \g__color_model_int
821         {
822             [ ~
823                 /DeviceN ~
824                 [ ~ #1 ~ ] ~
825                 #2 ~
826                 { ~ #3 ~ } ~
827             ] ~ setcolorspace
828         } ~ def ~
829     end
830 }
831 }

```

(End definition for `\_color_backend_devicen_init:nmn`.)

832  $\langle$ /dvips $\rangle$

833  $\langle$ \*dvisvgn $\rangle$

`\_color_backend_select_separation:nn` No support at present.

```

\_color_backend_select_devicen:nn 834 \cs_new_protected:Npn \_color_backend_select_separation:nn #1#2 { }
835 \cs_new_protected:Npn \_color_backend_select_devicen:nn #1#2 { }

```

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

`\_color_backend_separation_init:nnnnn` No support at present.

```

\_color_backend_separation_init_CIELAB:nnn 836 \cs_new_protected:Npn \_color_backend_separation_init:nnnnn #1#2#3#4#5 { }
837 \cs_new_protected:Npn \_color_backend_separation_init_CIELAB:nnnnn #1#2#3 { }

```

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

838  $\langle$ /dvisvgn $\rangle$

839  $\langle *dvipdfmx | luatex | pdftex | xetex \rangle$

Although (x)dvipdfmx has a built-in approach to color spaces, that can't be used with the generic color stacks. So we take an approach in which we share the same code as for pdfTEX.

```
840 \cs_new_protected:Npn \__color_backend_select_separation:nn #1#2
841   { \__color_backend_select:nn { /#1 ~ cs ~ #2 ~ scn } { /#1 ~ CS ~ #2 ~ SCN } }
842 \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.)
```

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.

```
\__color_backend_separation_init:nmnnn
\__color_backend_separation_init:n
\__color_backend_separation_init_CIELAB:nnn

843 \cs_new_protected:Npn \__color_backend_separation_init:nmnnn #1#2#3#4#5
844   {
845     \pdf_object_unnamed_write:nx { dict }
846     {
847       /FunctionType ~ 2
848       /Domain ~ [0 ~ 1]
849       \tl_if_blank:nF {#3} { /Range ~ [#3] }
850       /C0 ~ [#4] ~
851       /C1 ~ [#5] /N ~ 1
852     }
853     \__color_backend_separation_init:n
854     {
855       /Separation ~
856       / \str_convert_pdfname:n {#1} ~ #2 ~
857       \pdf_object_ref_last:
858     }
859     \cs_if_exist:NT \pdfmanagement_add:nnn
860     {
861       \use:x
862       {
863         \pdfmanagement_add:nnn
864         { Page / Resources / ColorSpace }
865         { color \int_use:N \g__color_model_int }
866         { \pdf_object_ref_last: }
867       }
868     }
869   }
870 \cs_new_protected:Npn \__color_backend_separation_init:n #1
871   {
872     \pdf_object_unnamed_write:nx { array } {#1}
873   }
```

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

```
874 \cs_new_protected:Npn \__color_backend_separation_init_CIELAB:nnn #1#2#3
875   {
876     \pdf_object_if_exist:nF { __color_illuminant_CIELAB_ #1 }
877     {
878       \pdf_object_new:nn { __color_illuminant_CIELAB_ #1 } { array }
879       \pdf_object_write:nx { __color_illuminant_CIELAB_ #1 }
```



```

880     {
881         /Lab ~
882         <<
883         /WhitePoint ~
884         [ \tl_use:c { c__color_model_whitepoint_CIELAB_ #1 _t1 } ]
885         /Range ~ [ \c__color_model_range_CIELAB_t1 ]
886         >>
887     }
888 }
889 \__color_backend_separation_init:nnnnn
890 {#2}
891 { \pdf_object_ref:n { __color_illuminant_CIELAB_ #1 } }
892 { \c__color_model_range_CIELAB_t1 }
893 { 100 ~ 0 ~ 0 }
894 {#3}
895 }

```

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

\\_\_color\_backend\_devicen\_init:nnn Similar to the Separations case, but with an arbitrary function for the alternative space work.

```

\__color_backend_devicen_init:w
\__color_backend_devicen_init:n
896 \cs_new_protected:Npn \__color_backend_devicen_init:nnn #1#2#3
897 {
898     \pdf_object_unnamed_write:nx { stream }
899     {
900         {
901             /FunctionType ~ 4 ~
902             /Domain ~
903             [ ~
904                 \prg_replicate:nn
905                 { 0 \__color_backend_devicen_init:w #1 ~ \s__color_stop }
906                 { 0 ~ 1 ~ } ~
907             ] ~
908             /Range ~
909             [ ~
910                 \str_case:nn {#2}
911                 {
912                     { /DeviceCMYK } { 0 ~ 1 ~ 0 ~ 1 ~ 0 ~ 1 ~ 0 ~ 1 }
913                     { /DeviceGray } { 0 ~ 1 }
914                     { /DeviceRGB } { 0 ~ 1 ~ 0 ~ 1 ~ 0 ~ 1 }
915                 } ~
916             ]
917         }
918         {#3}
919     }
920     \__color_backend_separation_init:n
921     {
922         /DeviceN ~
923         [ ~ #1 ~ ] ~
924         #2 ~
925         \pdf_object_ref_last:
926     }
927     \cs_if_exist:NT \pdfmanagement_add:nnn

```

```

928     {
929         \use:x
930         {
931             \pdfmanagement_add:nnn
932             { Page / Resources / ColorSpace }
933             { color \int_use:N \g__color_model_int }
934             { \pdf_object_ref_last: }
935         }
936     }
937 }
938 \cs_new:Npn \__color_backend_devicen_init:w #1 ~ #2 \s__color_stop
939 {
940     + 1
941     \tl_if_blank:nF {#2}
942     { \__color_backend_devicen_init:w #2 \s__color_stop }
943 }
944 \cs_new_eq:NN \__color_backend_devicen_init:n \__color_backend_separation_init:n

```

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

```

945 </dvipdfmx | luatex | pdftex | xetex>
946 <*dvipdfmx | xetex>

```

\\_\_color\_backend\_select\_separation:nn For older (x)dvipdfmx, we *could* support separations using a dedicated mechanism, but  
 \\_\_color\_backend\_select\_devicen:nn it was not added that long before the color stacks. So instead of having two complex paths, just disable here.

```

947 \int_compare:nNnT \c__kernel_sys_dvipdfmx_version_int < { 20201111 }
948 {
949     \cs_gset_protected:Npn \__color_backend_select_separation:nn #1#2 { }
950     \cs_gset_eq:NN \__color_backend_select_devicen:nn
951     \__color_backend_select_separation:nn
952 }

```

(End definition for \\_\_color\_backend\_select\_separation:nn and \\_\_color\_backend\_select\_devicen:nn.)

```

953 </dvipdfmx | xetex>

```

### 3.5 Fill and stroke color

Here, dvipdfmx/X<sub>Y</sub>TeX follows LuaTeX and pdfTeX, while for dvips we have to manage fill and stroke color ourselves. We also handle dvisvgm independently, as there we can create SVG directly.

```

954 <*dvipdfmx | luatex | pdftex | xetex>

```

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.

```

955 \cs_new_protected:Npn \__color_backend_fill_cmyk:n #1
956     { \__color_backend_fill:n { #1 ~ k } }
957 \cs_new_protected:Npn \__color_backend_fill_gray:n #1
958     { \__color_backend_fill:n { #1 ~ g } }
959 \cs_new_protected:Npn \__color_backend_fill_rgb:n #1

```

```

960 { \_color_backend_fill:n { #1 ~ rg } }
961 \cs_new_protected:Npn \_color_backend_fill:n #1
962 {
963   \tl_set:Nn \l__color_backend_fill_tl {#1}
964   \__kernel_color_backend_stack_push:nn \l__color_backend_stack_int
965   { #1 ~ \l__color_backend_stroke_tl }
966   \group_insert_after:N \_color_backend_reset:
967 }
968 \cs_new_protected:Npn \_color_backend_stroke_cmyk:n #1
969 { \_color_backend_stroke:n { #1 ~ K } }
970 \cs_new_protected:Npn \_color_backend_stroke_gray:n #1
971 { \_color_backend_stroke:n { #1 ~ G } }
972 \cs_new_protected:Npn \_color_backend_stroke_rgb:n #1
973 { \_color_backend_stroke:n { #1 ~ RG } }
974 \cs_new_protected:Npn \_color_backend_stroke:n #1
975 {
976   \tl_set:Nn \l__color_backend_stroke_tl {#1}
977   \__kernel_color_backend_stack_push:nn \l__color_backend_stack_int
978   { \l__color_backend_fill_tl \c_space_tl #1 }
979   \group_insert_after:N \_color_backend_reset:
980 }

```

(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
981 \cs_new_protected:Npn \_color_backend_fill_separation:nn #1#2
982 { \_color_backend_fill:n { /#1 ~ cs ~ #2 ~ scn } }
983 \cs_new_protected:Npn \_color_backend_stroke_separation:nn #1#2
984 { \_color_backend_stroke:n { /#1 ~ CS ~ #2 ~ SCN } }
985 \cs_new_eq:NN \_color_backend_fill_devicen:nn \_color_backend_fill_separation:nn
986 \cs_new_eq:NN \_color_backend_stroke_devicen:nn \_color_backend_stroke_separation:nn

```

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

```

987 </dvi/pdf/mx | luatex | pdftex | xetex>
988 <*dvi/pdf/mx | xetex>

```

```

\_color_backend_fill_cmyk:n Deal with older (x)dvi/pdf/mx.
\_color_backend_fill_gray:n
\_color_backend_fill_rgb:n
\_color_backend_reset:
\_color_backend_stroke:n
\_color_backend_fill_separation:nn
\_color_backend_stroke_separation:nn
989 \int_compare:nNnT \c__kernel_sys_dvipdfmx_version_int < { 20201111 }
990 {
991   \cs_gset_protected:Npn \_color_backend_fill_cmyk:n #1
992   {
993     \__kernel_backend_literal:n { pdf: bc ~ [#1] }
994     \group_insert_after:N \_color_backend_reset:
995   }
996   \cs_gset_eq:NN \_color_backend_fill_gray:n \_color_backend_fill_cmyk:n
997   \cs_gset_eq:NN \_color_backend_fill_rgb:n \_color_backend_fill_cmyk:n
998   \cs_gset_protected:Npn \_color_backend_reset:
999   { \__kernel_backend_literal:n { pdf: ec } }
1000   \cs_gset_protected:Npn \_color_backend_stroke:n #1
1001   { \__kernel_backend_literal:n {#1} }
1002   \cs_gset_protected:Npn \_color_backend_fill_separation:nn #1#2 { }
1003   \cs_gset_eq:NN \_color_backend_fill_devicen:nn
1004   \_color_backend_fill_separation:nn
1005   \cs_gset_eq:NN \_color_backend_stroke_devicen:nn

```

```

1006     \_color_backend_fill_separation:nn
1007     \cs_gset_eq:NN \_color_backend_stroke_devicen:nn
1008     \_color_backend_stroke_separation:nn
1009   }

```

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

```
1010 </dvipdfmx | xetex>
```

```
1011 <*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 1012 \cs_new_protected:Npn \_color_backend_fill_cmyk:n #1
\_color_backend_fill_rgb:n 1013 { \_color_backend_fill:n { cmyk ~ #1 } }
\_color_backend_fill:n 1014 \cs_new_protected:Npn \_color_backend_fill_gray:n #1
\_color_backend_stroke_cmyk:n 1015 { \_color_backend_fill:n { gray ~ #1 } }
\_color_backend_stroke_gray:n 1016 \cs_new_protected:Npn \_color_backend_fill_rgb:n #1
\_color_backend_stroke_rgb:n 1017 { \_color_backend_fill:n { rgb ~ #1 } }
1018 \cs_new_protected:Npn \_color_backend_fill:n #1
1019 {
1020   \_kernel_backend_literal:n { color~push~ #1 }
1021   \group_insert_after:N \_color_backend_reset:
1022 }
1023 \cs_new_protected:Npn \_color_backend_stroke_cmyk:n #1
1024 { \_kernel_backend_postscript:n { /color.sc { #1 ~ setcmykcolor } def } }
1025 \cs_new_protected:Npn \_color_backend_stroke_gray:n #1
1026 { \_kernel_backend_postscript:n { /color.sc { #1 ~ setgray } def } }
1027 \cs_new_protected:Npn \_color_backend_stroke_rgb:n #1
1028 { \_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 1029 \cs_new_protected:Npn \_color_backend_fill_separation:nn #1#2
\_color_backend_fill_devicen:nn 1030 { \_color_backend_fill:n { separation ~ #1 ~ #2 } }
\_color_backend_stroke_devicen:nn 1031 \cs_new_protected:Npn \_color_backend_stroke_separation:nn #1#2
1032 { \_kernel_backend_postscript:n { /color.sc { separation ~ #1 ~ #2 } def } }
1033 \cs_new_eq:NN \_color_backend_fill_devicen:nn \_color_backend_fill_separation:nn
1034 \cs_new_eq:NN \_color_backend_stroke_devicen:nn \_color_backend_stroke_separation:nn

```

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

```
1035 </dvips>
```

```
1036 <*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 1037 \cs_new_protected:Npn \_color_backend_fill_cmyk:n #1
\_color_backend_fill_rgb:n 1038 { \_color_backend_fill:n { cmyk ~ #1 } }
\_color_backend_fill:n 1039 \cs_new_protected:Npn \_color_backend_fill_gray:n #1
1040 { \_color_backend_fill:n { gray ~ #1 } }
1041 \cs_new_protected:Npn \_color_backend_fill_rgb:n #1
1042 { \_color_backend_fill:n { rgb ~ #1 } }
1043 \cs_new_protected:Npn \_color_backend_fill:n #1
1044 {
1045   \_kernel_backend_literal:n { color~push~ #1 }
1046   \group_insert_after:N \_color_backend_reset:
1047 }

```

(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
1048 \cs_new_protected:Npn \_color_backend_stroke_cmyk:n #1
1049 { \_color_backend_cmyk:w #1 \s_color_stop }
1050 \cs_new_protected:Npn \_color_backend_stroke_new_cmyk:w
1051 #1 ~ #2 ~ #3 ~ #4 \s_color_stop
1052 {
1053   \use:x
1054   {
1055     \_color_backend:nnn
1056     { \fp_eval:n { -100 * ( 1 - min ( 1 , #1 + #4 ) ) } }
1057     { \fp_eval:n { -100 * ( 1 - min ( 1 , #2 + #4 ) ) } }
1058     { \fp_eval:n { -100 * ( 1 - min ( 1 , #3 + #4 ) ) } }
1059   }
1060 }
1061 \cs_new_protected:Npn \_color_backend_stroke_gray:n #1
1062 {
1063   \use:x
1064   {
1065     \_color_backend_stroke_gray_aux:n
1066     { \fp_eval:n { 100 * (#1) } }
1067   }
1068 }
1069 \cs_new_protected:Npn \_color_backend_stroke_gray_aux:n #1
1070 { \_color_backend:nnn {#1} {#1} {#1} }
1071 \cs_new_protected:Npn \_color_backend_stroke_rgb:n #1
1072 { \_color_backend_rgb:w #1 \s_color_stop }
1073 \cs_new_protected:Npn \_color_backend_stroke_new_rgb:w
1074 #1 ~ #2 ~ #3 \s_color_stop
1075 {
1076   \use:x
1077   {
1078     \_color_backend:nnn
1079     { \fp_eval:n { 100 * (#1) } }
1080     { \fp_eval:n { 100 * (#2) } }
1081     { \fp_eval:n { 100 * (#3) } }
1082   }
1083 }
1084 \cs_new_protected:Npx \_color_backend:nnn #1#2#3
1085 {
1086   \_kernel_backend_scope:n
1087   {
1088     stroke =
1089     "
1090     rgb
1091     (
1092       #1 \c_percent_str ,
1093       #2 \c_percent_str ,
1094       #3 \c_percent_str
1095     )
1096     "
1097   }

```

```

1098 }
(End definition for \_color_backend_stroke_cmyk:n and others.)

```

At present, these are no-ops.

```

\_color_backend_fill_separation:nn 1099 \cs_new_protected:Npn \_color_backend_fill_separation:nn #1#2 { }
\_color_backend_stroke_separation:nn 1100 \cs_new_protected:Npn \_color_backend_stroke_separation:nn #1#2 { }
\_color_backend_fill_devicen:nn 1101 \cs_new_eq:NN \_color_backend_fill_devicen:nn \_color_backend_fill_separation:nn
\_color_backend_stroke_devicen:nn 1102 \cs_new_eq:NN \_color_backend_stroke_devicen:nn \_color_backend_stroke_separation:nn
(End definition for \_color_backend_fill_separation:nn and others.)
1103 </dvisvgm>
1104 </package>

```

## 4 I3backend-draw Implementation

```

1105 <*package>
1106 <@@=draw>

```

### 4.1 dvips backend

```

1107 <*dvips>

```

The same as literal PostScript: same arguments about positioning apply her.

```

\_draw_backend_literal:n 1108 \cs_new_eq:NN \_draw_backend_literal:n \_kernel_backend_literal_postscript:n
\_draw_backend_literal:x 1109 \cs_generate_variant:Nn \_draw_backend_literal:n { x }
(End definition for \_draw_backend_literal:n.)

```

The `ps::[begin]` special here deals with positioning but allows us to continue on to a 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.

```

1110 \cs_new_protected:Npn \_draw_backend_begin:
1111 {
1112   \_kernel_backend_literal:n { ps::[begin] }
1113   \_draw_backend_literal:n { @beginspecial }
1114 }
1115 \cs_new_protected:Npn \_draw_backend_end:
1116 {
1117   \_draw_backend_literal:n { @endspecial }
1118   \_kernel_backend_literal:n { ps::[end] }
1119 }
(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.

```
1120 \cs_new_protected:Npn \_draw_backend_scope_begin:
1121 { \_draw_backend_literal:n { save } }
1122 \cs_new_protected:Npn \_draw_backend_scope_end:
1123 { \_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  
`\_draw_backend_lineto:nn` the need to convert to bp. Notice that x-type expansion is included here to ensure that  
`\_draw_backend_rectangle:nmmn` any variable values are forced to literals before any possible caching. There is no native  
`\_draw_backend_curveto:nmmmmn` rectangular path command (without also clipping, filling or stroking), so that task is  
done using a small amount of PostScript.

```
1124 \cs_new_protected:Npn \_draw_backend_moveto:nn #1#2
1125 {
1126   \_draw_backend_literal:x
1127   {
1128     \dim_to_decimal_in_bp:n {#1} ~
1129     \dim_to_decimal_in_bp:n {#2} ~ moveto
1130   }
1131 }
1132 \cs_new_protected:Npn \_draw_backend_lineto:nn #1#2
1133 {
1134   \_draw_backend_literal:x
1135   {
1136     \dim_to_decimal_in_bp:n {#1} ~
1137     \dim_to_decimal_in_bp:n {#2} ~ lineto
1138   }
1139 }
1140 \cs_new_protected:Npn \_draw_backend_rectangle:nmmn #1#2#3#4
1141 {
1142   \_draw_backend_literal:x
1143   {
1144     \dim_to_decimal_in_bp:n {#4} ~ \dim_to_decimal_in_bp:n {#3} ~
1145     \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~
1146     moveto-dup-0~rlineto-exch-0~exch~rlineto-neg-0~rlineto-closepath
1147   }
1148 }
1149 \cs_new_protected:Npn \_draw_backend_curveto:nmmmmn #1#2#3#4#5#6
1150 {
1151   \_draw_backend_literal:x
1152   {
1153     \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~
1154     \dim_to_decimal_in_bp:n {#3} ~ \dim_to_decimal_in_bp:n {#4} ~
1155     \dim_to_decimal_in_bp:n {#5} ~ \dim_to_decimal_in_bp:n {#6} ~
1156     curveto
1157   }
1158 }
```

(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.  
`\_draw_backend_nonzero_rule:`  
`\g__draw_draw_eor_bool`

```
1159 \cs_new_protected:Npn \_draw_backend_evenodd_rule:
```

```

1160 { \bool_gset_true:N \g__draw_draw_eor_bool }
1161 \cs_new_protected:Npn \__draw_backend_nonzero_rule:
1162 { \bool_gset_false:N \g__draw_draw_eor_bool }
1163 \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 also desirable to have the `clip` keyword after a stroke or fill. To achieve those outcomes, there is some work to do. For color, the stroke color is simple but the fill one has to be inserted by hand. For clipping, the required ordering is achieved using a T<sub>E</sub>X switch. `\__draw_backend_fillstroke:` All of the operations end with a new path instruction as they do not terminate (again in contrast to PDF).

```

\__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
1164 \cs_new_protected:Npn \__draw_backend_closepath:
1165 { \__draw_backend_literal:n { closepath } }
1166 \cs_new_protected:Npn \__draw_backend_stroke:
1167 {
1168   \__draw_backend_literal:n { gsave }
1169   \__draw_backend_literal:n { color.sc }
1170   \__draw_backend_literal:n { stroke }
1171   \__draw_backend_literal:n { grestore }
1172   \bool_if:NT \g__draw_draw_clip_bool
1173   {
1174     \__draw_backend_literal:x
1175     {
1176       \bool_if:NT \g__draw_draw_eor_bool { eo }
1177       clip
1178     }
1179   }
1180   \__draw_backend_literal:n { newpath }
1181   \bool_gset_false:N \g__draw_draw_clip_bool
1182 }
1183 \cs_new_protected:Npn \__draw_backend_closestroke:
1184 {
1185   \__draw_backend_closepath:
1186   \__draw_backend_stroke:
1187 }
1188 \cs_new_protected:Npn \__draw_backend_fill:
1189 {
1190   \__draw_backend_literal:x
1191   {
1192     \bool_if:NT \g__draw_draw_eor_bool { eo }
1193     fill
1194   }
1195   \bool_if:NT \g__draw_draw_clip_bool
1196   {
1197     \__draw_backend_literal:x
1198     {
1199       \bool_if:NT \g__draw_draw_eor_bool { eo }
1200       clip
1201     }
1202   }
1203   \__draw_backend_literal:n { newpath }

```



```

1204     \bool_gset_false:N \g__draw_draw_clip_bool
1205   }
1206 \cs_new_protected:Npn \__draw_backend_fillstroke:
1207 {
1208   \__draw_backend_literal:x
1209   {
1210     \bool_if:NT \g__draw_draw_eor_bool { eo }
1211     fill
1212   }
1213   \__draw_backend_literal:n { gsave }
1214   \__draw_backend_literal:n { color.sc }
1215   \__draw_backend_literal:n { stroke }
1216   \__draw_backend_literal:n { grestore }
1217   \bool_if:NT \g__draw_draw_clip_bool
1218   {
1219     \__draw_backend_literal:x
1220     {
1221       \bool_if:NT \g__draw_draw_eor_bool { eo }
1222       clip
1223     }
1224   }
1225   \__draw_backend_literal:n { newpath }
1226   \bool_gset_false:N \g__draw_draw_clip_bool
1227 }
1228 \cs_new_protected:Npn \__draw_backend_clip:
1229 { \bool_gset_true:N \g__draw_draw_clip_bool }
1230 \bool_new:N \g__draw_draw_clip_bool
1231 \cs_new_protected:Npn \__draw_backend_discardpath:
1232 {
1233   \bool_if:NT \g__draw_draw_clip_bool
1234   {
1235     \__draw_backend_literal:x
1236     {
1237       \bool_if:NT \g__draw_draw_eor_bool { eo }
1238       clip
1239     }
1240   }
1241   \__draw_backend_literal:n { newpath }
1242   \bool_gset_false:N \g__draw_draw_clip_bool
1243 }

```

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

```

\__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:

```

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

```

1244 \cs_new_protected:Npn \__draw_backend_dash_pattern:nn #1#2
1245 {
1246   \__draw_backend_literal:x
1247   {
1248     [
1249       \exp_args:Nf \use:n
1250       { \clist_map_function:nN {#1} \__draw_backend_dash:n }
1251     ] ~
1252     \dim_to_decimal_in_bp:n {#2} ~ setdash
1253   }

```

```

1254 }
1255 \cs_new:Npn \__draw_backend_dash:n #1
1256 { ~ \dim_to_decimal_in_bp:n {#1} }
1257 \cs_new_protected:Npn \__draw_backend_linewidth:n #1
1258 {
1259   \__draw_backend_literal:x
1260   { \dim_to_decimal_in_bp:n {#1} ~ setlinewidth }
1261 }
1262 \cs_new_protected:Npn \__draw_backend_miterlimit:n #1
1263 { \__draw_backend_literal:n { #1 ~ setmiterlimit } }
1264 \cs_new_protected:Npn \__draw_backend_cap_but:
1265 { \__draw_backend_literal:n { 0 ~ setlinecap } }
1266 \cs_new_protected:Npn \__draw_backend_cap_round:
1267 { \__draw_backend_literal:n { 1 ~ setlinecap } }
1268 \cs_new_protected:Npn \__draw_backend_cap_rectangle:
1269 { \__draw_backend_literal:n { 2 ~ setlinecap } }
1270 \cs_new_protected:Npn \__draw_backend_join_miter:
1271 { \__draw_backend_literal:n { 0 ~ setlinejoin } }
1272 \cs_new_protected:Npn \__draw_backend_join_round:
1273 { \__draw_backend_literal:n { 1 ~ setlinejoin } }
1274 \cs_new_protected:Npn \__draw_backend_join_bevel:
1275 { \__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.

```

1276 \cs_new_protected:Npn \__draw_backend_cm:nnnn #1#2#3#4
1277 {
1278   \__draw_backend_literal:n
1279   { [ #1 ~ #2 ~ #3 ~ #4 ~ 0 ~ 0 ] ~ concat }
1280 }

```

(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).

```

1281 \cs_new_protected:Npn \__draw_backend_box_use:Nnnnn #1#2#3#4#5
1282 {
1283   \__draw_backend_literal:n { @endspecial }
1284   \__draw_backend_literal:n { [end] }
1285   \__draw_backend_literal:n { [begin] }

```

```

1286     \_draw_backend_literal:n { save }
1287     \_draw_backend_literal:n { currentpoint }
1288     \_draw_backend_literal:n { currentpoint~translate }
1289     \_draw_backend_cm:nmmm { 1 } { 0 } { 0 } { -1 }
1290     \_draw_backend_cm:nmmm {#2} {#3} {#4} {#5}
1291     \_draw_backend_cm:nmmm { 1 } { 0 } { 0 } { -1 }
1292     \_draw_backend_literal:n { neg~exch~neg~exch~translate }
1293     \_draw_backend_literal:n { [end] }
1294     \hbox_overlap_right:n { \box_use:N #1 }
1295     \_draw_backend_literal:n { [begin] }
1296     \_draw_backend_literal:n { restore }
1297     \_draw_backend_literal:n { [end] }
1298     \_draw_backend_literal:n { [begin] }
1299     \_draw_backend_literal:n { @beginspecial }
1300 }

```

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

```
1301 </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.

```
1302 <*dvipdfmx | luatex | pdftex | xetex>
```

### 4.2.1 Drawing

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

```

\_draw_backend_literal:x 1303 \cs_new_eq:NN \_draw_backend_literal:n \_kernel_backend_literal_pdf:n
1304 \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: 1305 \cs_new_protected:Npn \_draw_backend_begin:
1306 { \_draw_backend_scope_begin: }
1307 \cs_new_protected:Npn \_draw_backend_end:
1308 { \_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: 1309 \cs_new_eq:NN \_draw_backend_scope_begin: \_kernel_backend_scope_begin:
1310 \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 1311 \cs_new_protected:Npn \_draw_backend_moveto:nn #1#2
\_draw_backend_curveto:nnmmmm 1312 {
1313     \_draw_backend_literal:x
1314     { \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~ m }
1315 }

```

```

1316 \cs_new_protected:Npn \__draw_backend_lineto:nn #1#2
1317 {
1318   \__draw_backend_literal:x
1319   { \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~ 1 }
1320 }
1321 \cs_new_protected:Npn \__draw_backend_curveto:nnnnn #1#2#3#4#5#6
1322 {
1323   \__draw_backend_literal:x
1324   {
1325     \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~
1326     \dim_to_decimal_in_bp:n {#3} ~ \dim_to_decimal_in_bp:n {#4} ~
1327     \dim_to_decimal_in_bp:n {#5} ~ \dim_to_decimal_in_bp:n {#6} ~
1328     c
1329   }
1330 }
1331 \cs_new_protected:Npn \__draw_backend_rectangle:nnnn #1#2#3#4
1332 {
1333   \__draw_backend_literal:x
1334   {
1335     \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~
1336     \dim_to_decimal_in_bp:n {#3} ~ \dim_to_decimal_in_bp:n {#4} ~
1337     re
1338   }
1339 }

```

(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.
\__draw_backend_nonzero_rule:
\g__draw_draw_eor_bool
1340 \cs_new_protected:Npn \__draw_backend_evenodd_rule:
1341 { \bool_gset_true:N \g__draw_draw_eor_bool }
1342 \cs_new_protected:Npn \__draw_backend_nonzero_rule:
1343 { \bool_gset_false:N \g__draw_draw_eor_bool }
1344 \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.
\__draw_backend_stroke:
\__draw_backend_closestroke:
\__draw_backend_fill:
\__draw_backend_fillstroke:
\__draw_backend_clip:
\__draw_backend_discardpath:
1345 \cs_new_protected:Npn \__draw_backend_closepath:
1346 { \__draw_backend_literal:n { h } }
1347 \cs_new_protected:Npn \__draw_backend_stroke:
1348 { \__draw_backend_literal:n { S } }
1349 \cs_new_protected:Npn \__draw_backend_closestroke:
1350 { \__draw_backend_literal:n { s } }
1351 \cs_new_protected:Npn \__draw_backend_fill:
1352 {
1353   \__draw_backend_literal:x
1354   { f \bool_if:NT \g__draw_draw_eor_bool * }
1355 }
1356 \cs_new_protected:Npn \__draw_backend_fillstroke:
1357 {
1358   \__draw_backend_literal:x
1359   { B \bool_if:NT \g__draw_draw_eor_bool * }
1360 }
1361 \cs_new_protected:Npn \__draw_backend_clip:

```

```

1362 {
1363   \__draw_backend_literal:x
1364   { W \bool_if:NT \g__draw_draw_eor_bool * }
1365 }
1366 \cs_new_protected:Npn \__draw_backend_discardpath:
1367 { \__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:
1368 \cs_new_protected:Npn \__draw_backend_dash_pattern:nn #1#2
1369 {
1370   \__draw_backend_literal:x
1371   {
1372     [
1373       \exp_args:Nf \use:n
1374       { \clist_map_function:nN {#1} \__draw_backend_dash:n }
1375     ] ~
1376     \dim_to_decimal_in_bp:n {#2} ~ d
1377   }
1378 }
1379 \cs_new:Npn \__draw_backend_dash:n #1
1380 { ~ \dim_to_decimal_in_bp:n {#1} }
1381 \cs_new_protected:Npn \__draw_backend_linewidth:n #1
1382 {
1383   \__draw_backend_literal:x
1384   { \dim_to_decimal_in_bp:n {#1} ~ w }
1385 }
1386 \cs_new_protected:Npn \__draw_backend_miterlimit:n #1
1387 { \__draw_backend_literal:x { #1 ~ M } }
1388 \cs_new_protected:Npn \__draw_backend_cap_but:
1389 { \__draw_backend_literal:n { 0 ~ J } }
1390 \cs_new_protected:Npn \__draw_backend_cap_round:
1391 { \__draw_backend_literal:n { 1 ~ J } }
1392 \cs_new_protected:Npn \__draw_backend_cap_rectangle:
1393 { \__draw_backend_literal:n { 2 ~ J } }
1394 \cs_new_protected:Npn \__draw_backend_join_miter:
1395 { \__draw_backend_literal:n { 0 ~ j } }
1396 \cs_new_protected:Npn \__draw_backend_join_round:
1397 { \__draw_backend_literal:n { 1 ~ j } }
1398 \cs_new_protected:Npn \__draw_backend_join_bevel:
1399 { \__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: versions!

```

1400 \cs_new_protected:Npn \__draw_backend_cm:nnnn #1#2#3#4
1401 {

```

```

1402 <*luatex | pdftex>
1403   \__kernel_backend_matrix:n { #1 ~ #2 ~ #3 ~ #4 }
1404 </luatex | pdftex>
1405 <*dvipdfmx | xetex>
1406   \__draw_backend_cm_decompose:nnnnN {#1} {#2} {#3} {#4}
1407   \__draw_backend_cm_aux:nnnn
1408 </dvipdfmx | xetex>
1409 }
1410 <*dvipdfmx | xetex>
1411 \cs_new_protected:Npn \__draw_backend_cm_aux:nnnn #1#2#3#4
1412 {
1413   \__kernel_backend_literal:x
1414   {
1415     x:rotate~
1416     \fp_compare:nNnTF {#1} = \c_zero_fp
1417     { 0 }
1418     { \fp_eval:n { round ( -#1 , 5 ) } }
1419   }
1420   \__kernel_backend_literal:x
1421   {
1422     x:scale~
1423     \fp_eval:n { round ( #2 , 5 ) } ~
1424     \fp_eval:n { round ( #3 , 5 ) }
1425   }
1426   \__kernel_backend_literal:x
1427   {
1428     x:rotate~
1429     \fp_compare:nNnTF {#4} = \c_zero_fp
1430     { 0 }
1431     { \fp_eval:n { round ( -#4 , 5 ) } }
1432   }
1433 }
1434 </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.

```

1435 <*dviptdftm|xetex>
1436 \cs_new_protected:Npn \__draw_backend_cm_decompose:nnnnN #1#2#3#4#5
1437 {
1438   \use:x
1439   {
1440     \__draw_backend_cm_decompose_auxi:nnnnN
1441     { \fp_eval:n { (#1 + #4) / 2 } }
1442     { \fp_eval:n { (#1 - #4) / 2 } }
1443     { \fp_eval:n { (#3 + #2) / 2 } }
1444     { \fp_eval:n { (#3 - #2) / 2 } }
1445   }
1446   #5
1447 }
1448 \cs_new_protected:Npn \__draw_backend_cm_decompose_auxi:nnnnN #1#2#3#4#5
1449 {
1450   \use:x
1451   {
1452     \__draw_backend_cm_decompose_auxii:nnnnN
1453     { \fp_eval:n { 2 * sqrt ( #1 * #1 + #4 * #4 ) } }
1454     { \fp_eval:n { 2 * sqrt ( #2 * #2 + #3 * #3 ) } }
1455     { \fp_eval:n { atand ( #3 , #2 ) } }
1456     { \fp_eval:n { atand ( #4 , #1 ) } }
1457   }
1458   #5
1459 }
1460 \cs_new_protected:Npn \__draw_backend_cm_decompose_auxii:nnnnN #1#2#3#4#5
1461 {
1462   \use:x
1463   {
1464     \__draw_backend_cm_decompose_auxiii:nnnnN
1465     { \fp_eval:n { ( #4 - #3 ) / 2 } }
1466     { \fp_eval:n { ( #1 + #2 ) / 2 } }
1467     { \fp_eval:n { ( #1 - #2 ) / 2 } }
1468     { \fp_eval:n { ( #4 + #3 ) / 2 } }
1469   }
1470   #5
1471 }
1472 \cs_new_protected:Npn \__draw_backend_cm_decompose_auxiii:nnnnN #1#2#3#4#5
1473 {
1474   \fp_compare:nNnTF { abs ( #2 ) } > { abs ( #3 ) }

```

```

1475     { #5 {#1} {#2} {#3} {#4} }
1476     { #5 {#1} {#3} {#2} {#4} }
1477   }
1478 </dvipdfmx | xetex>

```

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

`\_draw_backend_box_use:Nnnnn`

Inserting a  $\TeX$  box transformed to the requested position and using the current matrix is done using a mixture of  $\TeX$  and low-level manipulation. The offset can be handled by  $\TeX$ , 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.

```

1479 \cs_new_protected:Npn \_draw_backend_box_use:Nnnnn #1#2#3#4#5
1480 {
1481   \_kernel_backend_scope_begin:
1482   <*luatex | pdftex>
1483     \_draw_backend_cm:nnnn {#2} {#3} {#4} {#5}
1484   </luatex | pdftex>
1485   <*dvipdfmx | xetex>
1486     \_kernel_backend_literal:n
1487     { pdf:btrans-matrix~ #2 ~ #3 ~ #4 ~ #5 ~ 0 ~ 0 }
1488   </dvipdfmx | xetex>
1489   \hbox_overlap_right:n { \box_use:N #1 }
1490   <*dvipdfmx | xetex>
1491   \_kernel_backend_literal:n { pdf:etrans }
1492   </dvipdfmx | xetex>
1493   \_kernel_backend_scope_end:
1494 }

```

(End definition for `\_draw_backend_box_use:Nnnnn`.)

```

1495 </dvipdfmx | luatex | pdftex | xetex>

```

### 4.3 dvisvgm backend

```

1496 <*dvisvgm>

```

`\_draw_backend_literal:n`

The same as the more general literal call.

`\_draw_backend_literal:x`

```

1497 \cs_new_eq:NN \_draw_backend_literal:n \_kernel_backend_literal_svg:n
1498 \cs_generate_variant:Nn \_draw_backend_literal:n { x }

```

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

`\_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

`\_draw_backend_end:`

```

1499 \cs_new_protected:Npn \_draw_backend_begin:
1500 {
1501   \_kernel_backend_scope_begin:
1502   \_kernel_backend_scope:n { transform="translate({?x},{?y})~scale(1,-1)" }
1503 }
1504 \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.

`\_draw_backend_lineto:nn` Since paths should be fully expanded there is no need to worry about the internal x-type expansion.

`\_draw_backend_rectangle:nnnn`  
`\_draw_backend_curveto:nnnnnn`  
`\_draw_backend_add_to_path:n`  
`\g__draw_draw_path_tl`

```

1505 \cs_new_protected:Npn \_draw_backend_moveto:nn #1#2
1506 {
1507   \_draw_backend_add_to_path:n
1508     { M ~ \dim_to_decimal:n {#1} ~ \dim_to_decimal:n {#2} }
1509 }
1510 \cs_new_protected:Npn \_draw_backend_lineto:nn #1#2
1511 {
1512   \_draw_backend_add_to_path:n
1513     { L ~ \dim_to_decimal:n {#1} ~ \dim_to_decimal:n {#2} }
1514 }
1515 \cs_new_protected:Npn \_draw_backend_rectangle:nnnn #1#2#3#4
1516 {
1517   \_draw_backend_add_to_path:n
1518     {
1519     M ~ \dim_to_decimal:n {#1} ~ \dim_to_decimal:n {#2}
1520     h ~ \dim_to_decimal:n {#3} ~
1521     v ~ \dim_to_decimal:n {#4} ~
1522     h ~ \dim_to_decimal:n { -#3 } ~
1523     Z
1524   }
1525 }
1526 \cs_new_protected:Npn \_draw_backend_curveto:nnnnnn #1#2#3#4#5#6
1527 {
1528   \_draw_backend_add_to_path:n
1529     {
1530     C ~
1531     \dim_to_decimal:n {#1} ~ \dim_to_decimal:n {#2} ~
1532     \dim_to_decimal:n {#3} ~ \dim_to_decimal:n {#4} ~
1533     \dim_to_decimal:n {#5} ~ \dim_to_decimal:n {#6}
1534   }
1535 }
1536 \cs_new_protected:Npn \_draw_backend_add_to_path:n #1
1537 {
1538   \tl_gset:Nx \g__draw_draw_path_tl
1539     {
1540     \g__draw_draw_path_tl
1541     \tl_if_empty:NF \g__draw_draw_path_tl { \c_space_tl }
1542     #1
1543   }
1544 }
1545 \tl_new:N \g__draw_draw_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.

```

1546 \cs_new_protected:Npn \_draw_backend_evenodd_rule:
1547   { \_draw_backend_scope:n { fill-rule="evenodd" } }
1548 \cs_new_protected:Npn \_draw_backend_nonzero_rule:
1549   { \_draw_backend_scope:n { fill-rule="nonzero" } }

```

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

```

\__draw_backend_path:n Setting fill and stroke effects and doing clipping all has to be done using scopes. This
\__draw_backend_closepath: means setting up the various requirements in a shared auxiliary which deals with the
\__draw_backend_stroke: bits and pieces. Clipping paths are reused for path drawing: not essential but avoids
\__draw_backend_closestroke: constructing them twice. Discarding a path needs a separate function as it's not quite
\__draw_backend_fill: the same.
\__draw_backend_fillstroke:
1550 \cs_new_protected:Npn \__draw_backend_closepath:
\__draw_backend_clip: 1551 { \__draw_backend_add_to_path:n { Z } }
\__draw_backend_discardpath: 1552 \cs_new_protected:Npn \__draw_backend_path:n #1
\g__draw_draw_clip_bool 1553 {
\g__draw_draw_path_int 1554   \bool_if:NTF \g__draw_draw_clip_bool
1555   {
1556     \int_gincr:N \g__draw_clip_path_int
1557     \__draw_backend_literal:x
1558     {
1559       < clipPath~id = " l3cp \int_use:N \g__draw_clip_path_int " >
1560       { ?nl }
1561       <path~d=" \g__draw_draw_path_tl "/> { ?nl }
1562       < /clipPath > { ? nl }
1563       <
1564         use~xlink:href =
1565         "\c_hash_str l3path \int_use:N \g__draw_path_int " ~
1566         #1
1567       />
1568     }
1569     \__draw_backend_scope:x
1570     {
1571       clip-path =
1572       "url( \c_hash_str l3cp \int_use:N \g__draw_clip_path_int)"
1573     }
1574   }
1575   {
1576     \__draw_backend_literal:x
1577     { <path ~ d=" \g__draw_draw_path_tl " ~ #1 /> }
1578   }
1579   \tl_gclear:N \g__draw_draw_path_tl
1580   \bool_gset_false:N \g__draw_draw_clip_bool
1581 }
1582 \int_new:N \g__draw_path_int
1583 \cs_new_protected:Npn \__draw_backend_stroke:
1584 { \__draw_backend_path:n { style="fill:none" } }
1585 \cs_new_protected:Npn \__draw_backend_closestroke:
1586 {
1587   \__draw_backend_closepath:
1588   \__draw_backend_stroke:
1589 }
1590 \cs_new_protected:Npn \__draw_backend_fill:
1591 { \__draw_backend_path:n { style="stroke:none" } }
1592 \cs_new_protected:Npn \__draw_backend_fillstroke:
1593 { \__draw_backend_path:n { } }
1594 \cs_new_protected:Npn \__draw_backend_clip:
1595 { \bool_gset_true:N \g__draw_draw_clip_bool }
1596 \bool_new:N \g__draw_draw_clip_bool

```

```

1597 \cs_new_protected:Npn \__draw_backend_discardpath:
1598 {
1599   \bool_if:NT \g__draw_draw_clip_bool
1600   {
1601     \int_gincr:N \g__draw_clip_path_int
1602     \__draw_backend_literal:x
1603     {
1604       < clipPath~id = " l3cp \int_use:N \g__draw_clip_path_int " >
1605       { ?nl }
1606       <path~d=" \g__draw_draw_path_tl "/> { ?nl }
1607       < /clipPath >
1608     }
1609     \__draw_backend_scope:x
1610     {
1611       clip-path =
1612       "url( \c_hash_str l3cp \int_use:N \g__draw_clip_path_int)"
1613     }
1614   }
1615   \tl_gclear:N \g__draw_draw_path_tl
1616   \bool_gset_false:N \g__draw_draw_clip_bool
1617 }

```

(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:
1618 \cs_new_protected:Npn \__draw_backend_dash_pattern:nn #1#2
1619 {
1620   \use:x
1621   {
1622     \__draw_backend_dash_aux:nn
1623     { \clist_map_function:nn {#1} \__draw_backend_dash:n }
1624     { \dim_to_decimal:n {#2} }
1625   }
1626 }
1627 \cs_new:Npn \__draw_backend_dash:n #1
1628 { , \dim_to_decimal_in_bp:n {#1} }
1629 \cs_new_protected:Npn \__draw_backend_dash_aux:nn #1#2
1630 {
1631   \__draw_backend_scope:x
1632   {
1633     stroke-dasharray =
1634     "
1635     \tl_if_empty:oTF { \use_none:n #1 }
1636     { none }
1637     { \use_none:n #1 }
1638     " ~
1639     stroke-offset=" #2 "
1640   }
1641 }
1642 \cs_new_protected:Npn \__draw_backend_linewidth:n #1
1643 { \__draw_backend_scope:x { stroke-width=" \dim_to_decimal:n {#1} " } }
1644 \cs_new_protected:Npn \__draw_backend_miterlimit:n #1
1645 { \__draw_backend_scope:x { stroke-miterlimit=" #1 " } }

```

```

1646 \cs_new_protected:Npn \__draw_backend_cap_but:
1647   { \__draw_backend_scope:n { stroke-linecap="butt" } }
1648 \cs_new_protected:Npn \__draw_backend_cap_round:
1649   { \__draw_backend_scope:n { stroke-linecap="round" } }
1650 \cs_new_protected:Npn \__draw_backend_cap_rectangle:
1651   { \__draw_backend_scope:n { stroke-linecap="square" } }
1652 \cs_new_protected:Npn \__draw_backend_join_miter:
1653   { \__draw_backend_scope:n { stroke-linejoin="miter" } }
1654 \cs_new_protected:Npn \__draw_backend_join_round:
1655   { \__draw_backend_scope:n { stroke-linejoin="round" } }
1656 \cs_new_protected:Npn \__draw_backend_join_bevel:
1657   { \__draw_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.

```

1658 \cs_new_protected:Npn \__draw_backend_cm:nnnn #1#2#3#4
1659   {
1660     \__draw_backend_scope:n
1661     {
1662       transform =
1663       " matrix ( #1 , #2 , #3 , #4 , Opt , Opt ) "
1664     }
1665   }

```

(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.

```

1666 \cs_new_protected:Npn \__draw_backend_box_use:Nnnnn #1#2#3#4#5#6#7
1667   {
1668     \__kernel_backend_scope_begin:
1669     \__draw_backend_cm:nnnn {#2} {#3} {#4} {#5}
1670     \__kernel_backend_literal_svg:n
1671     {
1672       < g~
1673       stroke="none"~
1674       transform="scale(-1,1)~translate({?x},{?y})~scale(-1,-1)"
1675     >
1676     }
1677     \box_set_wd:Nn #1 { Opt }
1678     \box_set_ht:Nn #1 { Opt }
1679     \box_set_dp:Nn #1 { Opt }
1680     \box_use:N #1
1681     \__kernel_backend_literal_svg:n { </g> }
1682     \__kernel_backend_scope_end:
1683   }

```

(End definition for `\__draw_backend_box_use:Nnnnn`.)

```
1684 </divisvgn>
```

```
1685 </package>
```

## 5 I3backend-graphics Implementation

```
1686 <*package>
1687 <@@=graphics>
```

### 5.1 dvips backend

```
1688 <*dvips>
```

`\_graphics_backend_getbb_eps:n` Simply use the generic function.

```
1689 \cs_new_eq:NN \_graphics_backend_getbb_eps:n \graphics_read_bb:n
```

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

`\_graphics_backend_include_eps:n` The special syntax is relatively clear here: remember we need PostScript sizes here.

```
1690 \cs_new_protected:Npn \_graphics_backend_include_eps:n #1
1691 {
1692   \_kernel_backend_literal:x
1693   {
1694     PSfile = #1 \c_space_tl
1695     llx = \dim_to_decimal_in_bp:n \l_graphics_llx_dim \c_space_tl
1696     lly = \dim_to_decimal_in_bp:n \l_graphics_lly_dim \c_space_tl
1697     urx = \dim_to_decimal_in_bp:n \l_graphics_urx_dim \c_space_tl
1698     ury = \dim_to_decimal_in_bp:n \l_graphics_ury_dim
1699   }
1700 }
```

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

```
1701 </dvips>
```

### 5.2 LuaTeX and pdfTeX backends

```
1702 <*luatex | pdftex>
```

`\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.

```
1703 \tl_new:N \l__graphics_graphics_attr_tl
```

(End definition for `\l__graphics_graphics_attr_tl`.)

`\_graphics_backend_getbb_jpg:n`  
`\_graphics_backend_getbb_pdf:n`  
`\_graphics_backend_getbb_png:n`  
`\_graphics_backend_getbb_auxi:n`  
`\_graphics_backend_getbb_auxii:n` 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.

```
1704 \cs_new_protected:Npn \_graphics_backend_getbb_jpg:n #1
1705 {
1706   \int_zero:N \l_graphics_page_int
1707   \tl_clear:N \l_graphics_pagebox_tl
1708   \tl_set:Nx \l__graphics_graphics_attr_tl
1709   {
1710     \tl_if_empty:NF \l_graphics_decodearray_tl
```

```

1711     { :D \l_graphics_decodearray_tl }
1712     \bool_if:NT \l_graphics_interpolate_bool
1713     { :I }
1714   }
1715   \tl_clear:N \l__graphics_graphics_attr_tl
1716   \__graphics_backend_getbb_auxi:n {#1}
1717 }
1718 \cs_new_eq:NN \__graphics_backend_getbb_png:n \__graphics_backend_getbb_jpg:n
1719 \cs_new_protected:Npn \__graphics_backend_getbb_pdf:n #1
1720 {
1721   \tl_clear:N \l_graphics_decodearray_tl
1722   \bool_set_false:N \l_graphics_interpolate_bool
1723   \tl_set:Nx \l__graphics_graphics_attr_tl
1724     {
1725     : \l_graphics_pagebox_tl
1726     \int_compare:nNnT \l_graphics_page_int > 1
1727     { :P \int_use:N \l_graphics_page_int }
1728   }
1729   \__graphics_backend_getbb_auxi:n {#1}
1730 }
1731 \cs_new_protected:Npn \__graphics_backend_getbb_auxi:n #1
1732 {
1733   \graphics_bb_restore:xF { #1 \l__graphics_graphics_attr_tl }
1734   { \__graphics_backend_getbb_auxii:n {#1} }
1735 }

```

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.

```

1736 \cs_new_protected:Npn \__graphics_backend_getbb_auxii:n #1
1737 {
1738   \tex_immediate:D \tex_pdfximage:D
1739   \bool_lazy_or:nnT
1740     { \l_graphics_interpolate_bool }
1741     { ! \tl_if_empty_p:N \l_graphics_decodearray_tl }
1742   {
1743     attr ~
1744     {
1745       \tl_if_empty:NF \l_graphics_decodearray_tl
1746       { /Decode~[ \l_graphics_decodearray_tl ] }
1747       \bool_if:NT \l_graphics_interpolate_bool
1748       { /Interpolate~true }
1749     }
1750   }
1751   \int_compare:nNnT \l_graphics_page_int > 0
1752   { page ~ \int_use:N \l_graphics_page_int }
1753   \tl_if_empty:NF \l_graphics_pagebox_tl
1754   { \l_graphics_pagebox_tl }
1755   {#1}
1756   \hbox_set:Nn \l__graphics_internal_box
1757     { \tex_pdfrefximage:D \tex_pdflastximage:D }
1758   \dim_set:Nn \l_graphics_urx_dim { \box_wd:N \l__graphics_internal_box }
1759   \dim_set:Nn \l_graphics_ury_dim { \box_ht:N \l__graphics_internal_box }
1760   \int_const:cn { c__graphics_graphics_ #1 \l__graphics_graphics_attr_tl _int }

```

```

1761     { \tex_the:D \tex_pdflastximage:D }
1762     \graphics_bb_save:x { #1 \l__graphics_graphics_attr_tl }
1763   }

```

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

`\__graphics_backend_include_jpg: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.

```

1764 \cs_new_protected:Npn \__graphics_backend_include_jpg:n #1
1765   {
1766     \tex_pdfrefximage:D
1767     \int_use:c { c__graphics_graphics_ #1 \l__graphics_graphics_attr_tl _int }
1768   }
1769 \cs_new_eq:NN \__graphics_backend_include_pdf:n \__graphics_backend_include_jpg:n
1770 \cs_new_eq:NN \__graphics_backend_include_png:n \__graphics_backend_include_jpg:n

```

(End definition for `\__graphics_backend_include_jpg:n`, `\__graphics_backend_include_pdf:n`, and `\__graphics_backend_include_png:n`.)

`\__graphics_backend_getbb_eps:n` EPS graphics may be included in Lua<sub>T</sub><sub>E</sub>X/pdfT<sub>E</sub>X by conversion to PDF: this requires restricted shell escape. Modelled on the `epstopdf LATEX 2ε` package, but simplified, conversion takes place here if we have shell access.

```

\l__graphics_backend_dir_str 1771 \sys_if_shell:T
\l__graphics_backend_name_str 1772   {
\l__graphics_backend_ext_str 1773     \str_new:N \l__graphics_backend_dir_str
1774     \str_new:N \l__graphics_backend_name_str
1775     \str_new:N \l__graphics_backend_ext_str
1776     \cs_new_protected:Npn \__graphics_backend_getbb_eps:n #1
1777       {
1778         \file_parse_full_name:nNNN {#1}
1779         \l__graphics_backend_dir_str
1780         \l__graphics_backend_name_str
1781         \l__graphics_backend_ext_str
1782         \exp_args:Nx \__graphics_backend_getbb_eps:nn
1783         {
1784           \l__graphics_backend_name_str - \str_tail:N \l__graphics_backend_ext_str
1785           -converted-to.pdf
1786         }
1787         {#1}
1788       }
1789     \cs_new_protected:Npn \__graphics_backend_getbb_eps:nn #1#2
1790       {
1791         \file_compare_timestamp:nNnT {#2} > {#1}
1792         {
1793           \sys_shell_now:n
1794           { repstopdf ~ #2 ~ #1 }
1795         }
1796         \tl_set:Nn \l__graphics_name_tl {#1}
1797         \__graphics_backend_getbb_pdf:n {#1}
1798       }
1799     \cs_new_protected:Npn \__graphics_backend_include_eps:n #1
1800       {
1801         \file_parse_full_name:nNNN {#1}

```

```

1802         \l__graphics_backend_dir_str \l__graphics_backend_name_str \l__graphics_backend_ext_str
1803         \exp_args:Nx \__graphics_backend_include_pdf:n
1804         {
1805             \l__graphics_backend_name_str - \str_tail:N \l__graphics_backend_ext_str
1806             -converted-to.pdf
1807         }
1808     }
1809 }

```

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

```
1810 </luatex | pdftex>
```

### 5.3 dvipdfmx backend

```
1811 <*dvipdfmx | xetex>
```

`\__graphics_backend_getbb_eps:n` Simply use the generic functions: only for `dvipdfmx` in the extraction cases.

```

\__graphics_backend_getbb_jpg:n
\__graphics_backend_getbb_pdf:n
\__graphics_backend_getbb_png:n
1812 \cs_new_eq:NN \__graphics_backend_getbb_eps:n \graphics_read_bb:n
1813 <*dvipdfmx>
1814 \cs_new_protected:Npn \__graphics_backend_getbb_jpg:n #1
1815 {
1816     \int_zero:N \l_graphics_page_int
1817     \tl_clear:N \l_graphics_pagebox_tl
1818     \graphics_extract_bb:n {#1}
1819 }
1820 \cs_new_eq:NN \__graphics_backend_getbb_png:n \__graphics_backend_getbb_jpg:n
1821 \cs_new_protected:Npn \__graphics_backend_getbb_pdf:n #1
1822 {
1823     \tl_clear:N \l_graphics_decodearray_tl
1824     \bool_set_false:N \l_graphics_interpolate_bool
1825     \graphics_extract_bb:n {#1}
1826 }
1827 </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.

```
1828 \int_new:N \g__graphics_track_int
```

(End definition for `\g__graphics_track_int`.)

`\__graphics_backend_include_eps:n` The special syntax depends on the file type. There is a difference in how PDF graphics are best handled between `dvipdfmx` and `XYTEX`: 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_jpg:n
\__graphics_backend_include_pdf:n
\__graphics_backend_include_png:n
\__graphics_backend_include_auxi:n
\__graphics_backend_include_auxii:nn
\__graphics_backend_include_auxiii:nn
1829 \cs_new_protected:Npn \__graphics_backend_include_eps:n #1
1830 {
1831     \__kernel_backend_literal:x
1832     {
1833         PSfile = #1 \c_space_tl
1834         llx = \dim_to_decimal_in_bp:n \l_graphics_llx_dim \c_space_tl
1835         lly = \dim_to_decimal_in_bp:n \l_graphics_lly_dim \c_space_tl
1836         urx = \dim_to_decimal_in_bp:n \l_graphics_urx_dim \c_space_tl
1837         ury = \dim_to_decimal_in_bp:n \l_graphics_ury_dim
1838     }

```



```

1839 }
1840 \cs_new_protected:Npn \__graphics_backend_include_jpg:n #1
1841 { \__graphics_backend_include_auxi:nn {#1} { image } }
1842 \cs_new_eq:NN \__graphics_backend_include_png:n \__graphics_backend_include_jpg:n
1843 {*dvipdfmx}
1844 \cs_new_protected:Npn \__graphics_backend_include_pdf:n #1
1845 { \__graphics_backend_include_auxi:nn {#1} { epdf } }
1846 </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.

```

1847 \cs_new_protected:Npn \__graphics_backend_include_auxi:nn #1#2
1848 {
1849   \__graphics_backend_include_auxii:xnn
1850   {
1851     \tl_if_empty:NF \l_graphics_pagebox_tl
1852     { : \l_graphics_pagebox_tl }
1853     \int_compare:nNnT \l_graphics_page_int > 1
1854     { :P \int_use:N \l_graphics_page_int }
1855     \tl_if_empty:NF \l_graphics_decodearray_tl
1856     { :D \l_graphics_decodearray_tl }
1857     \bool_if:NT \l_graphics_interpolate_bool
1858     { :I }
1859   }
1860   {#1} {#2}
1861 }
1862 \cs_new_protected:Npn \__graphics_backend_include_auxii:nnn #1#2#3
1863 {
1864   \int_if_exist:cTF { c__graphics_graphics_ #2#1 _int }
1865   {
1866     \__kernel_backend_literal:x
1867     { pdf:useobj~@graphic \int_use:c { c__graphics_graphics_ #2#1 _int } }
1868   }
1869   { \__graphics_backend_include_auxiii:nnn {#2} {#1} {#3} }
1870 }
1871 \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!

```

1872 \cs_new_protected:Npn \__graphics_backend_include_auxiii:nnn #1#2#3
1873 {
1874   \int_gincr:N \g__graphics_track_int
1875   \int_const:cn { c__graphics_graphics_ #1#2 _int } { \g__graphics_track_int }
1876   \__kernel_backend_literal:x
1877   {
1878     pdf:#3~
1879     @graphic \int_use:c { c__graphics_graphics_ #1#2 _int } ~
1880     \int_compare:nNnT \l_graphics_page_int > 1
1881     { page ~ \int_use:N \l_graphics_page_int \c_space_tl }
1882     \tl_if_empty:NF \l_graphics_pagebox_tl
1883     {

```

```

1884         pagebox ~ \l_graphics_pagebox_tl \c_space_tl
1885         bbox ~
1886             \dim_to_decimal_in_bp:n \l_graphics_llx_dim \c_space_tl
1887             \dim_to_decimal_in_bp:n \l_graphics_lly_dim \c_space_tl
1888             \dim_to_decimal_in_bp:n \l_graphics_urx_dim \c_space_tl
1889             \dim_to_decimal_in_bp:n \l_graphics_ury_dim \c_space_tl
1890     }
1891     (#1)
1892     \bool_lazy_or:nnT
1893     { \l_graphics_interpolate_bool }
1894     { ! \tl_if_empty_p:N \l_graphics_decodearray_tl }
1895     {
1896         <<
1897         \tl_if_empty:NF \l_graphics_decodearray_tl
1898         { /Decode~[ \l_graphics_decodearray_tl ] }
1899         \bool_if:NT \l_graphics_interpolate_bool
1900         { /Interpolate~true> }
1901         >>
1902     }
1903 }
1904 }

```

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

```
1905 </dviptfm | xetex>
```

## 5.4 X<sub>Y</sub>TeX backend

```
1906 < *xetex >
```

### 5.4.1 Images

For X<sub>Y</sub>TeX, 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>TeX primitive omits the text box from the page box specification, so there is also some “trimming” to do here.

```

\__graphics_backend_getbb_jpg:n
\__graphics_backend_getbb_pdf:n
\__graphics_backend_getbb_png:n
\__graphics_backend_getbb_auxi:nN
\__graphics_backend_getbb_auxii:nNn
\__graphics_backend_getbb_auxiii:nNnn
\__graphics_backend_getbb_auxiv:nNnn
\__graphics_backend_getbb_auxv:nNnn
\__graphics_backend_getbb_auxv:nNnn
\__graphics_backend_getbb_pagebox:w

```

```

1907 \cs_new_protected:Npn \__graphics_backend_getbb_jpg:n #1
1908 {
1909     \int_zero:N \l_graphics_page_int
1910     \tl_clear:N \l_graphics_pagebox_tl
1911     \__graphics_backend_getbb_auxi:nN {#1} \tex_XeTeXpicfile:D
1912 }
1913 \cs_new_eq:NN \__graphics_backend_getbb_png:n \__graphics_backend_getbb_jpg:n
1914 \cs_new_protected:Npn \__graphics_backend_getbb_pdf:n #1
1915 {
1916     \tl_clear:N \l_graphics_decodearray_tl
1917     \bool_set_false:N \l_graphics_interpolate_bool
1918     \__graphics_backend_getbb_auxi:nN {#1} \tex_XeTeXpdffile:D
1919 }
1920 \cs_new_protected:Npn \__graphics_backend_getbb_auxi:nN #1#2
1921 {
1922     \int_compare:nNnTF \l_graphics_page_int > 1
1923     { \__graphics_backend_getbb_auxii:VnN \l_graphics_page_int {#1} #2 }
1924     { \__graphics_backend_getbb_auxiii:nNnn {#1} #2 { :P 1 } { page 1 } }
1925 }

```

```

1926 \cs_new_protected:Npn \__graphics_backend_getbb_auxii:nnN #1#2#3
1927   { \__graphics_backend_getbb_auxiii:nNnn {#2} #3 { :P #1 } { page #1 } }
1928 \cs_generate_variant:Nn \__graphics_backend_getbb_auxii:nnN { V }
1929 \cs_new_protected:Npn \__graphics_backend_getbb_auxiii:nNnn #1#2#3#4
1930   {
1931     \tl_if_empty:NTF \l_graphics_pagebox_tl
1932       { \__graphics_backend_getbb_auxiv:VnNnn \l_graphics_pagebox_tl }
1933       { \__graphics_backend_getbb_auxv:nNnn }
1934       {#1} #2 {#3} {#4}
1935   }
1936 \cs_new_protected:Npn \__graphics_backend_getbb_auxiv:nnNnn #1#2#3#4#5
1937   {
1938     \use:x
1939     {
1940       \__graphics_backend_getbb_auxv:nNnn {#2} #3 { : #1 #4 }
1941       { #5 ~ \__graphics_backend_getbb_pagebox:w #1 }
1942     }
1943   }
1944 \cs_generate_variant:Nn \__graphics_backend_getbb_auxiv:nnNnn { V }
1945 \cs_new_protected:Npn \__graphics_backend_getbb_auxv:nNnn #1#2#3#4
1946   {
1947     \graphics_bb_restore:nF {#1#3}
1948     { \__graphics_backend_getbb_auxvi:nNnn {#1} #2 {#3} {#4} }
1949   }
1950 \cs_new_protected:Npn \__graphics_backend_getbb_auxvi:nNnn #1#2#3#4
1951   {
1952     \hbox_set:Nn \l_graphics_internal_box { #2 #1 ~ #4 }
1953     \dim_set:Nn \l_graphics_urx_dim { \box_wd:N \l_graphics_internal_box }
1954     \dim_set:Nn \l_graphics_ury_dim { \box_ht:N \l_graphics_internal_box }
1955     \graphics_bb_save:n {#1#3}
1956   }
1957 \cs_new:Npn \__graphics_backend_getbb_pagebox:w #1 box {#1}

```

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

For PDF graphics, properly supporting the pagebox concept in X<sub>Y</sub>T<sub>E</sub>X 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`.

```

1958 \cs_new_protected:Npn \__graphics_backend_include_pdf:n #1
1959   {
1960     \tex_XeTeXpdffile:D
1961     \__graphics_backend_include_pdf_quote:w #1 "#1" \s__graphics_stop \c_space_tl
1962     \int_compare:nNnT \l_graphics_page_int > 0
1963       { page ~ \int_use:N \l_graphics_page_int \c_space_tl }
1964     \exp_after:wN \__graphics_backend_getbb_pagebox:w \l_graphics_pagebox_tl
1965   }
1966 \cs_new:Npn \__graphics_backend_include_pdf_quote:w #1 " #2 " #3 \s__graphics_stop
1967   { " #2 " }

```

(End definition for \\_\_graphics\_backend\_include\_pdf:n and \\_\_graphics\_backend\_include\_bitmap\_quote:w.)

```

1968 </xetex>

```

## 5.5 dvisvgm backend

1969 `<*dvisvgm>`

`\_graphics_backend_getbb_eps:n` Simply use the generic function.

```
1970 \cs_new_eq:NN \_graphics_backend_getbb_eps:n \graphics_read_bb:n
```

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

`\_graphics_backend_getbb_png:n` These can be included by extracting the bounding box data.

`\_graphics_backend_getbb_jpg:n`

```
1971 \cs_new_protected:Npn \_graphics_backend_getbb_jpg:n #1
```

```
1972 {
```

```
1973   \int_zero:N \l_graphics_page_int
```

```
1974   \tl_clear:N \l_graphics_pagebox_tl
```

```
1975   \graphics_extract_bb:n {#1}
```

```
1976 }
```

```
1977 \cs_new_eq:NN \_graphics_backend_getbb_png:n \_graphics_backend_getbb_jpg:n
```

(End definition for `\_graphics_backend_getbb_png:n` and `\_graphics_backend_getbb_jpg:n`.)

`\_graphics_backend_getbb_pdf:n` Same as for `dvipdfmx`: use the generic function

```
1978 \cs_new_protected:Npn \_graphics_backend_getbb_pdf:n #1
```

```
1979 {
```

```
1980   \tl_clear:N \l_graphics_decodearray_tl
```

```
1981   \bool_set_false:N \l_graphics_interpolate_bool
```

```
1982   \graphics_extract_bb:n {#1}
```

```
1983 }
```

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

`\_graphics_backend_include_eps:n` The special syntax is relatively clear here: remember we need PostScript sizes here. (This is the same as the `dvips` code.)

`\_graphics_backend_include_pdf:n`

`\_graphics_backend_include:nn`

```
1984 \cs_new_protected:Npn \_graphics_backend_include_eps:n #1
```

```
1985 { \_graphics_backend_include:nn { PSfile } {#1} }
```

```
1986 \cs_new_protected:Npn \_graphics_backend_include_pdf:n #1
```

```
1987 { \_graphics_backend_include:nn { pdffile } {#1} }
```

```
1988 \cs_new_protected:Npn \_graphics_backend_include:nn #1#2
```

```
1989 {
```

```
1990   \_kernel_backend_literal:x
```

```
1991   {
```

```
1992     #1 = #2 \c_space_tl
```

```
1993     llx = \dim_to_decimal_in_bp:n \l_graphics_llx_dim \c_space_tl
```

```
1994     lly = \dim_to_decimal_in_bp:n \l_graphics_lly_dim \c_space_tl
```

```
1995     urx = \dim_to_decimal_in_bp:n \l_graphics_urx_dim \c_space_tl
```

```
1996     ury = \dim_to_decimal_in_bp:n \l_graphics_ury_dim
```

```
1997   }
```

```
1998 }
```

(End definition for `\_graphics_backend_include_eps:n`, `\_graphics_backend_include_pdf:n`, and `\_graphics_backend_include:nn`.)

`\_graphics_backend_include_png:n` 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).

`\_graphics_backend_include_jpg:n`

`\_graphics_backend_include_bitmap_quote:n`

The only 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.

```

1999 \cs_new_protected:Npn \__graphics_backend_include_png:n #1
2000 {
2001   \__kernel_backend_literal:x
2002   {
2003     dvisvgm:img~
2004     \dim_to_decimal:n { \l_graphics_ury_dim } ~
2005     \dim_to_decimal:n { \l_graphics_ury_dim } ~
2006     \__graphics_backend_include_bitmap_quote:w #1 " #1 " \s__graphics_stop
2007   }
2008 }
2009 \cs_new_eq:NN \__graphics_backend_include_jpg:n \__graphics_backend_include_png:n
2010 \cs_new:Npn \__graphics_backend_include_bitmap_quote:w #1 " #2 " #3 \s__graphics_stop
2011 { " #2 " }

(End definition for \__graphics_backend_include_png:n, \__graphics_backend_include_jpg:n, and
\__graphics_backend_include_bitmap_quote:w.)

2012 </dvisvgm>
2013 </package>

```

## 6 I3backend-pdf Implementation

```

2014 <*package>
2015 <@=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
2016 \box_new:N \l__pdf_internal_box

```

(End definition for `\l__pdf_internal_box`.)

### 6.2 dvips backend

```

2017 <*dvips>

```

`\__pdf_backend_pdfmark:n` Used often enough it should be a separate function.

```

\__pdf_backend_pdfmark:x
2018 \cs_new_protected:Npn \__pdf_backend_pdfmark:n #1
2019 { \__kernel_backend_postscript:n { mark #1 ~ pdfmark } }
2020 \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
2021 \cs_new_protected:Npn \__pdf_backend_catalog_gput:nn #1#2
2022 { \__pdf_backend_pdfmark:n { { Catalog } << /#1 ~ #2 >> /PUT } }
2023 \cs_new_protected:Npn \__pdf_backend_info_gput:nn #1#2
2024 { \__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

\g\_\_pdf\_backend\_object\_int For tracking objects to allow finalisation.

```

\g__pdf_backend_object_prop
2025 \int_new:N \g__pdf_backend_object_int
2026 \prop_new:N \g__pdf_backend_object_prop

(End definition for \g__pdf_backend_object_int and \g__pdf_backend_object_prop.)

```

\\_\_pdf\_backend\_object\_new:nn Tracking objects is similar to dvipdfmx.

```

\__pdf_backend_object_ref:nn
2027 \cs_new_protected:Npn \__pdf_backend_object_new:nn #1#2
2028 {
2029   \int_gincr:N \g__pdf_backend_object_int
2030   \int_const:cn
2031   { c__pdf_backend_object_ \t1_to_str:n {#1} _int }
2032   { \g__pdf_backend_object_int }
2033   \prop_gput:Nnn \g__pdf_backend_object_prop {#1} {#2}
2034 }
2035 \cs_new:Npn \__pdf_backend_object_ref:nn #1
2036 { { pdf.obj \int_use:c { c__pdf_backend_object_ \t1_to_str:n {#1} _int } } }

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

```

\\_\_pdf\_backend\_object\_write:nn This is where we choose the actual type: some work to get things right.

```

\__pdf_backend_object_write:nx
2037 \cs_new_protected:Npn \__pdf_backend_object_write:nn #1#2
\__pdf_backend_object_write_array:nn
2038 {
\__pdf_backend_object_write_dict:nn
2039   \__pdf_backend_pdfmark:x
\__pdf_backend_object_write_fstream:nn
2040   {
\__pdf_backend_object_write_stream:nn
2041     /objdef ~ \__pdf_backend_object_ref:nn {#1}
2042     /type
\__pdf_backend_object_write_stream:nnn
2043     \str_case_e:nn
2044     { \prop_item:Nn \g__pdf_backend_object_prop {#1} }
2045     {
2046       { array } { /array }
2047       { dict } { /dict }
2048       { fstream } { /stream }
2049       { stream } { /stream }
2050     }
2051     /OBJ
2052   }
2053   \use:c
2054   { __pdf_backend_object_write_ \prop_item:Nn \g__pdf_backend_object_prop {#1} :nn }
2055   { \__pdf_backend_object_ref:nn {#1} } {#2}
2056 }
2057 \cs_generate_variant:Nn \__pdf_backend_object_write:nn { nx }
2058 \cs_new_protected:Npn \__pdf_backend_object_write_array:nn #1#2

```

```

2059 {
2060   \_pdf_backend_pdfmark:x
2061   { #1 ~0~ [ ~ \exp_not:n {#2} ~ ] ~ /PUTINTERVAL }
2062 }
2063 \cs_new_protected:Npn \_pdf_backend_object_write_dict:nn #1#2
2064 {
2065   \_pdf_backend_pdfmark:x
2066   { #1 << \exp_not:n {#2} >> /PUT }
2067 }
2068 \cs_new_protected:Npn \_pdf_backend_object_write_fstream:nn #1#2
2069 {
2070   \exp_args:Nx
2071   \_pdf_backend_object_write_fstream:nnn {#1} #2
2072 }
2073 \cs_new_protected:Npn \_pdf_backend_object_write_fstream:nnn #1#2#3
2074 {
2075   \_kernel_backend_postscript:n
2076   {
2077     SDict ~ begin ~
2078     mark ~ #1 ~ << #2 >> /PUT ~ pdfmark ~
2079     mark ~ #1 ~ ( #3 )~ ( r )~ file ~ /PUT ~ pdfmark ~
2080     end
2081   }
2082 }
2083 \cs_new_protected:Npn \_pdf_backend_object_write_stream:nn #1#2
2084 {
2085   \exp_args:Nx
2086   \_pdf_backend_object_write_stream:nnn {#1} #2
2087 }
2088 \cs_new_protected:Npn \_pdf_backend_object_write_stream:nnn #1#2#3
2089 {
2090   \_kernel_backend_postscript:n
2091   {
2092     mark ~ #1 ~ ( #3 ) /PUT ~ pdfmark ~
2093     mark ~ #1 ~ << #2 >> /PUT ~ pdfmark
2094   }
2095 }

```

(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
2096 \cs_new_protected:Npn \_pdf_backend_object_now:nn #1#2
2097 {
2098   \int_gincr:N \g__pdf_backend_object_int
2099   \_pdf_backend_pdfmark:x
2100   {
2101     /objdef ~ { pdf.obj \int_use:N \g__pdf_backend_object_int }
2102     /type
2103     \str_case:nn
2104     {#1}
2105     {
2106       { array } { /array }
2107       { dict } { /dict }
2108       { fstream } { /stream }

```

```

2109         { stream } { /stream }
2110     }
2111     /OBJ
2112 }
2113 \exp_args:Nnx \use:c { __pdf_backend_object_write_ #1 :nn }
2114 { { pdf.obj \int_use:N \g__pdf_backend_object_int } } {#2}
2115 }
2116 \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.

```

2117 \cs_new:Npn \__pdf_backend_object_last:
2118 { { 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.

```

2119 \cs_new:Npn \__pdf_backend_pageobject_ref:n #1
2120 { { 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.

```

2121 \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.

```

2122 \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.

```

2123 \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  $\LaTeX$  2<sub>ε</sub> `picture` of zero size). Once the data is collected, use it to set up the annotation border.

```

2124 \cs_new_protected:Npn \__pdf_backend_annotation:nnnn #1#2#3#4
2125 {
2126     \exp_args:Nf \__pdf_backend_annotation_aux:nnnn
2127     { \dim_eval:n {#1} } {#2} {#3} {#4}
2128 }
2129 \cs_new_protected:Npn \__pdf_backend_annotation_aux:nnnn #1#2#3#4
2130 {

```



```

2131 \box_move_down:nn {#3}
2132   { \hbox:n { \__kernel_backend_postscript:n { pdf.save.ll } } }
2133 \box_move_up:nn {#2}
2134   {
2135     \hbox:n
2136     {
2137       \__kernel_kern:n {#1}
2138       \__kernel_backend_postscript:n { pdf.save.ur }
2139       \__kernel_kern:n { -#1 }
2140     }
2141   }
2142 \int_gincr:N \g__pdf_backend_object_int
2143 \int_gset_eq:NN \g__pdf_backend_annotation_int \g__pdf_backend_object_int
2144 \__pdf_backend_pdfmark:x
2145   {
2146     /objdef { pdf.obj \int_use:N \g__pdf_backend_object_int }
2147     pdf.rect
2148     #4 ~
2149     /ANN
2150   }
2151 }

```

(End definition for `\__pdf_backend_annotation:mnmn`.)

`\__pdf_backend_annotation_last`: Provide the last annotation we created: could get tricky of course if other packages are loaded.

```

2152 \cs_new:Npn \__pdf_backend_annotation_last:
2153   { { 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.

```

2154 \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.

```

2155 \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.

```

2156 \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.

```

2157 \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.

```

2158 \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.

```

2159 \tl_new:N \l__pdf_breaklink_pdfmark_tl
2160 \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.

```

2161 \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.

```

2162 \cs_new_eq:NN \__pdf_breaklink_usebox:N \box_use:N
(End definition for \__pdf_breaklink_usebox:N.)

```

`\__pdf_backend_link_begin_goto:nmw` 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:nmw`

`\__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:`

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

`\__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`

```

2163 \cs_new_protected:Npn \__pdf_backend_link_begin_goto:nmw #1#2
2164 { \__pdf_backend_link_begin:nw { #1 /Subtype /Link /A << /S /GoTo /D ( #2 ) >> } }
2165 \cs_new_protected:Npn \__pdf_backend_link_begin_user:nmw #1#2
2166 { \__pdf_backend_link_begin:nw {#1#2} }
2167 \cs_new_protected:Npn \__pdf_backend_link_begin:nw #1
2168 {
2169   \bool_if:NF \g__pdf_backend_link_bool
2170     { \__pdf_backend_link_begin_aux:nw {#1} }
2171 }
2172 \cs_new_protected:Npn \__pdf_backend_link_begin_aux:nw #1
2173 {
2174   \bool_gset_true:N \g__pdf_backend_link_bool
2175   \__kernel_backend_postscript:n
2176     { /pdf.link.dict ( #1 ) def }
2177   \tl_gset:Nn \g__pdf_backend_link_dict_tl {#1}
2178   \__pdf_backend_link_sf_save:
2179   \mode_if_math:TF
2180     { \bool_gset_true:N \g__pdf_backend_link_math_bool }
2181     { \bool_gset_false:N \g__pdf_backend_link_math_bool }
2182   \hbox_set:Nw \l__pdf_backend_content_box
2183     \__pdf_backend_link_sf_restore:
2184     \bool_if:NT \g__pdf_backend_link_math_bool
2185       { \c_math_toggle_token }

```

```

2186 }
2187 \cs_new_protected:Npn \__pdf_backend_link_end:
2188 {
2189   \bool_if:NT \g__pdf_backend_link_bool
2190     { \__pdf_backend_link_end_aux: }
2191 }
2192 \cs_new_protected:Npn \__pdf_backend_link_end_aux:
2193 {
2194   \bool_if:NT \g__pdf_backend_link_math_bool
2195     { \c_math_toggle_token }
2196   \__pdf_backend_link_sf_save:
2197   \hbox_set_end:
2198   \__pdf_backend_link_minima:
2199   \hbox_set:Nn \l__pdf_backend_model_box { Gg }
2200   \exp_args:Nx \__pdf_backend_link_outerbox:n
2201     {
2202       \int_if_odd:nTF { \value { page } }
2203         { \oddsidemargin }
2204         { \evensidemargin }
2205     }
2206   \box_move_down:nn { \box_dp:N \l__pdf_backend_content_box }
2207     { \hbox:n { \__kernel_backend_postscript:n { pdf.save.linkll } } }
2208   \__pdf_breaklink_postscript:n { pdf.bordertracking.begin }
2209   \__pdf_breaklink_usebox:N \l__pdf_backend_content_box
2210   \__pdf_breaklink_postscript:n { pdf.bordertracking.end }
2211   \box_move_up:nn { \box_ht:N \l__pdf_backend_content_box }
2212     {
2213       \hbox:n
2214         { \__kernel_backend_postscript:n { pdf.save.linkur } }
2215     }
2216   \int_gincr:N \g__pdf_backend_object_int
2217   \int_gset_eq:NN \g__pdf_backend_link_int \g__pdf_backend_object_int
2218   \__kernel_backend_postscript:x
2219     {
2220       mark
2221       /_objdef { pdf.obj \int_use:N \g__pdf_backend_link_int }
2222       \g__pdf_backend_link_dict_tl \c_space_tl
2223       pdf.rect
2224       /ANN ~ \l__pdf_breaklink_pdfmark_tl
2225     }
2226   \__pdf_backend_link_sf_restore:
2227   \bool_gset_false:N \g__pdf_backend_link_bool
2228 }
2229 \cs_new_protected:Npn \__pdf_backend_link_minima:
2230 {
2231   \hbox_set:Nn \l__pdf_backend_model_box { Gg }
2232   \__kernel_backend_postscript:x
2233     {
2234       /pdf.linkdp.pad ~
2235       \dim_to_decimal:n
2236         {
2237           \dim_max:nn
2238             {
2239               \box_dp:N \l__pdf_backend_model_box

```

```

2240         - \box_dp:N \l__pdf_backend_content_box
2241     }
2242     { Opt }
2243 } ~
2244 pdf.pt.dvi ~ def
2245 /pdf.linkht.pad ~
2246 \dim_to_decimal:n
2247 {
2248     \dim_max:nn
2249     {
2250         \box_ht:N \l__pdf_backend_model_box
2251         - \box_ht:N \l__pdf_backend_content_box
2252     }
2253     { Opt }
2254 } ~
2255 pdf.pt.dvi ~ def
2256 }
2257 }
2258 \cs_new_protected:Npn \__pdf_backend_link_outerbox:n #1
2259 {
2260     \__kernel_backend_postscript:x
2261     {
2262         /pdf.outerbox
2263         [
2264             \dim_to_decimal:n {#1} ~
2265             \dim_to_decimal:n { -\box_dp:N \l__pdf_backend_model_box } ~
2266             \dim_to_decimal:n { #1 + \textwidth } ~
2267             \dim_to_decimal:n { \box_ht:N \l__pdf_backend_model_box }
2268         ]
2269         [ exch { pdf.pt.dvi } forall ] def
2270     /pdf.baselineskip ~
2271     \dim_to_decimal:n { \tex_baselineskip:D } ~ dup ~ 0 ~ gt
2272     { pdf.pt.dvi ~ def }
2273     { pop ~ pop }
2274     ifelse
2275     }
2276 }
2277 \cs_new_protected:Npn \__pdf_backend_link_sf_save:
2278 {
2279     \int_gset:Nn \g__pdf_backend_link_sf_int
2280     {
2281         \mode_if_horizontal:TF
2282         { \tex_spacefactor:D }
2283         { 0 }
2284     }
2285 }
2286 \cs_new_protected:Npn \__pdf_backend_link_sf_restore:
2287 {
2288     \mode_if_horizontal:T
2289     {
2290         \int_compare:nNnT \g__pdf_backend_link_sf_int > { 0 }
2291         { \int_set_eq:NN \tex_spacefactor:D \g__pdf_backend_link_sf_int }
2292     }
2293 }

```

(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.

```

2294 \use_none:n
2295 {
2296   \cs_if_exist:NT \@makecol@hook
2297   {
2298     \tl_put_right:Nn \@makecol@hook
2299     {
2300       \box_if_empty:NF \@cclv
2301       {
2302         \vbox_set:Nn \@cclv
2303         {
2304           \__kernel_backend_postscript:n
2305           {
2306             pdf.globaldict /pdf.brokenlink.rect ~ known
2307             { pdf.bordertracking.continue }
2308             if
2309             }
2310             \vbox_unpack_drop:N \@cclv
2311             \__kernel_backend_postscript:n
2312             { pdf.bordertracking.endpage }
2313           }
2314         }
2315       }
2316       \tl_set:Nn \l__pdf_breaklink_pdfmark_tl { pdf.pdfmark }
2317       \cs_set_eq:NN \__pdf_breaklink_postscript:n \__kernel_backend_postscript:n
2318       \cs_set_eq:NN \__pdf_breaklink_usebox:N \hbox_unpack:N
2319     }
2320   }

```

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

`\_pdf_backend_link_last:` The same as annotations, but with a custom integer.

```

2321 \cs_new:Npn \_pdf_backend_link_last:
2322   { { 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.

```

2323 \cs_new_protected:Npn \_pdf_backend_link_margin:n #1
2324   {
2325     \__kernel_backend_postscript:x
2326     {
2327       /pdf.linkmargin { \dim_to_decimal:n {#1} ~ pdf.pt.dvi } def
2328     }
2329   }

```

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

`\_pdf_backend_destination:nn
 \_pdf_backend_destination:nmmn
 \_pdf_backend_destination_aux:nmmn`

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.

```

2330 \cs_new_protected:Npn \_pdf_backend_destination:nn #1#2
2331 {
2332   \_kernel_backend_postscript:n { pdf.dest.anchor }
2333   \_pdf_backend_pdfmark:x
2334   {
2335     /View
2336     [
2337       \str_case:nnF {#2}
2338       {
2339         { xyz } { /XYZ ~ pdf.dest.point ~ null }
2340         { fit } { /Fit }
2341         { fitb } { /FitB }
2342         { fitbh } { /FitBH ~ pdf.dest.y }
2343         { fitbv } { /FitBV ~ pdf.dest.x }
2344         { fith } { /FitH ~ pdf.dest.y }
2345         { fitv } { /FitV ~ pdf.dest.x }
2346         { fitr } { /Fit }
2347       }
2348       {
2349         /XYZ ~ pdf.dest.point ~ \fp_eval:n { (#2) / 100 }
2350       }
2351     ]
2352     /Dest ( \exp_not:n {#1} ) cvn
2353     /DEST
2354   }
2355 }
2356 \cs_new_protected:Npn \_pdf_backend_destination:nmmn #1#2#3#4
2357 {
2358   \exp_args:Ne \_pdf_backend_destination_aux:nmmn
2359   { \dim_eval:n {#2} } {#1} {#3} {#4}
2360 }
2361 \cs_new_protected:Npn \_pdf_backend_destination_aux:nmmn #1#2#3#4
2362 {
2363   \vbox_to_zero:n
2364   {
2365     \_kernel_kern:n {#4}
2366     \hbox:n { \_kernel_backend_postscript:n { pdf.save.ll } }
2367     \tex_vss:D
2368   }
2369   \_kernel_kern:n {#1}
2370   \vbox_to_zero:n
2371   {
2372     \_kernel_kern:n { -#3 }
2373     \hbox:n { \_kernel_backend_postscript:n { pdf.save.ur } }
2374     \tex_vss:D
2375   }
2376   \_kernel_kern:n { -#1 }
2377   \_pdf_backend_pdfmark:n
2378   {
2379     /View
  
```

```

2380     [
2381     /FitR ~
2382     pdf.llx ~ pdf.lly ~ pdf.dest2device ~
2383     pdf.urx ~ pdf.ury ~ pdf.dest2device
2384     ]
2385     /Dest ( #2 ) cvn
2386     /DEST
2387   }
2388 }

```

(End definition for `\_pdf_backend_destination:n`, `\_pdf_backend_destination:nmnn`, and `\_pdf_backend_destination_aux:nmnn`.)

## 6.2.4 Structure

`\_pdf_backend_compresslevel:n` Doable for the usual `ps2pdf` method.

```

\_pdf_backend_compress_objects:n 2389 \cs_new_protected:Npn \_pdf_backend_compresslevel:n #1
2390 {
2391   \int_compare:nNnT {#1} = 0
2392   {
2393     \__kernel_backend_literal_postscript:n
2394     {
2395       /setdistillerparams ~ where
2396       { pop << /CompressPages ~ false >> setdistillerparams }
2397       if
2398     }
2399   }
2400 }
2401 \cs_new_protected:Npn \_pdf_backend_compress_objects:n #1
2402 {
2403   \bool_if:nF {#1}
2404   {
2405     \__kernel_backend_literal_postscript:n
2406     {
2407       /setdistillerparams ~ where
2408       { pop << /CompressStreams ~ false >> setdistillerparams }
2409       if
2410     }
2411   }
2412 }

```

(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`

```

2413 \cs_new_protected:Npn \_pdf_backend_version_major_gset:n #1
2414 {
2415   \cs_gset:Npx \_pdf_backend_version_major: { \int_eval:n {#1} }
2416 }
2417 \cs_new_protected:Npn \_pdf_backend_version_minor_gset:n #1
2418 {
2419   \cs_gset:Npx \_pdf_backend_version_minor: { \int_eval:n {#1} }
2420 }

```

(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: 2421 \cs_new:Npn \__pdf_backend_version_major: { -1 }
                               2422 \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: 2423 \cs_new_protected:Npn \__pdf_backend_bdc:nn #1#2
                    2424 { \__pdf_backend_pdfmark:n { /#1 ~ #2 /BDC } }
                    2425 \cs_new_protected:Npn \__pdf_backend_emc:
                    2426 { \__pdf_backend_pdfmark:n { /EMC } }

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

2427 </dvips>

```

## 6.3 LuaTeX and pdfTeX backend

```
2428 <*luatex | pdftex>
```

### 6.3.1 Annotations

```

\__pdf_backend_annotation:nmmn Simply pass the raw data through, just dealing with evaluation of dimensions.
2429 \cs_new_protected:Npn \__pdf_backend_annotation:nmmn #1#2#3#4
2430 {
2431 <*luatex>
2432   \tex_pdfextension:D annot ~
2433 </luatex>
2434 <*pdftex>
2435   \tex_pdfannot:D
2436 </pdftex>
2437   width ~ \dim_eval:n {#1} ~
2438   height ~ \dim_eval:n {#2} ~
2439   depth ~ \dim_eval:n {#3} ~
2440   {#4}
2441 }

(End definition for \__pdf_backend_annotation:nmmn.)

```

`\__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.

```

2442 \cs_new:Npx \__pdf_backend_annotation_last:
2443 {
2444   \exp_not:N \int_value:w
2445 <*luatex>
2446   \exp_not:N \tex_pdffeedback:D lastannot ~
2447 </luatex>
2448 <*pdftex>
2449   \exp_not:N \tex_pdflastannot:D
2450 </pdftex>
2451   \c_space_tl 0 ~ R
2452 }

```



(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 2453 \cs_new_protected:Npn \_pdf_backend_link_begin_goto:nnw #1#2
  \_pdf_backend_link_begin:nnnw { \_pdf_backend_link_begin:nnnw {#1} { goto~name } {#2} }
\_pdf_backend_link_end:          2454 { \_pdf_backend_link_begin:nnnw {#1} { user } {#2} }
                                2455 \cs_new_protected:Npn \_pdf_backend_link_begin_user:nnw #1#2
                                2456 { \_pdf_backend_link_begin:nnnw {#1} { user } {#2} }
                                2457 \cs_new_protected:Npn \_pdf_backend_link_begin:nnw #1#2#3
                                2458 {
                                2459 <*luatex>
                                2460   \tex_pdfextension:D startlink ~
                                2461 </luatex>
                                2462 <*pdftex>
                                2463   \tex_pdfstartlink:D
                                2464 </pdftex>
                                2465     attr {#1}
                                2466     #2 {#3}
                                2467   }
                                2468 \cs_new_protected:Npn \_pdf_backend_link_end:
                                2469 {
                                2470 <*luatex>
                                2471   \tex_pdfextension:D endlink \scan_stop:
                                2472 </luatex>
                                2473 <*pdftex>
                                2474   \tex_pdfendlink:D
                                2475 </pdftex>
                                2476   }
```

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

```
\_pdf_backend_link_last: Formatted for direct use.
                                2477 \cs_new:Npx \_pdf_backend_link_last:
                                2478 {
                                2479   \exp_not:N \int_value:w
                                2480 <*luatex>
                                2481   \exp_not:N \tex_pdffeedback:D lastlink ~
                                2482 </luatex>
                                2483 <*pdftex>
                                2484   \exp_not:N \tex_pdflastlink:D
                                2485 </pdftex>
                                2486   \c_space_tl 0 ~ R
                                2487 }
```

(End definition for `\_pdf_backend_link_last:`)

```
\_pdf_backend_link_margin:n A simple task: pass the data to the primitive.
                                2488 \cs_new_protected:Npn \_pdf_backend_link_margin:n #1
                                2489 {
                                2490 <*luatex>
                                2491   \tex_pdfvariable:D linkmargin
                                2492 </luatex>
                                2493 <*pdftex>
                                2494   \tex_pdflinkmargin:D
                                2495 </pdftex>
```

```

2496     \dim_eval:n {#1} \scan_stop:
2497   }

```

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

`\_pdf_backend_destination:nn` `\_pdf_backend_destination:nmmn` 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.

```

2498 \cs_new_protected:Npn \_pdf_backend_destination:nn #1#2
2499 {
2500 <*luatex>
2501   \tex_pdfextension:D dest ~
2502 </luatex>
2503 <*pdftex>
2504   \tex_pdfdest:D
2505 </pdftex>
2506   name {#1}
2507   \str_case:nnF {#2}
2508   {
2509     { xyz } { xyz }
2510     { fit } { fit }
2511     { fitb } { fitb }
2512     { fitbh } { fitbh }
2513     { fitbv } { fitbv }
2514     { fith } { fith }
2515     { fitv } { fitv }
2516     { fitr } { fitr }
2517   }
2518   { xyz ~ zoom \fp_eval:n { #2 * 10 } }
2519   \scan_stop:
2520 }
2521 \cs_new_protected:Npn \_pdf_backend_destination:nmmn #1#2#3#4
2522 {
2523 <*luatex>
2524   \tex_pdfextension:D dest ~
2525 </luatex>
2526 <*pdftex>
2527   \tex_pdfdest:D
2528 </pdftex>
2529   name {#1}
2530   fitr ~
2531   width \dim_eval:n {#2} ~
2532   height \dim_eval:n {#3} ~
2533   depth \dim_eval:n {#4} \scan_stop:
2534 }

```

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

### 6.3.2 Catalogue entries

```

\_pdf_backend_catalog_gput:nn
\_pdf_backend_info_gput:nn
2535 \cs_new_protected:Npn \_pdf_backend_catalog_gput:nn #1#2
2536 {
2537 <*luatex>

```

```

2538     \tex_pdfextension:D catalog
2539 </luatex>
2540 <*pdftex>
2541     \tex_pdfcatalog:D
2542 </pdftex>
2543     { / #1 ~ #2 }
2544 }
2545 \cs_new_protected:Npn \__pdf_backend_info_gput:nn #1#2
2546 {
2547 <*luatex>
2548     \tex_pdfextension:D info
2549 </luatex>
2550 <*pdftex>
2551     \tex_pdfinfo:D
2552 </pdftex>
2553     { / #1 ~ #2 }
2554 }

```

(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.

```
2555 \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

```

2556 \cs_new_protected:Npn \__pdf_backend_object_new:nn #1#2
2557 {
2558 <*luatex>
2559     \tex_pdfextension:D obj ~
2560 </luatex>
2561 <*pdftex>
2562     \tex_pdfobj:D
2563 </pdftex>
2564     reserveobjnum ~
2565     \int_const:cn
2566     { c__pdf_backend_object_ \tl_to_str:n {#1} _int }
2567 <*luatex>
2568     { \tex_pdffeedback:D lastobj }
2569 </luatex>
2570 <*pdftex>
2571     { \tex_pdflastobj:D }
2572 </pdftex>
2573     \prop_gput:Nnn \g__pdf_backend_object_prop {#1} {#2}
2574 }
2575 \cs_new:Npn \__pdf_backend_object_ref:n #1
2576 { \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

```

2577 \cs_new_protected:Npn \__pdf_backend_object_write:nn #1#2
2578 {

```

```

2579 <*luatex>
2580   \tex_immediate:D \tex_pdfextension:D obj ~
2581 </luatex>
2582 <*pdftex>
2583   \tex_immediate:D \tex_pdfobj:D
2584 </pdftex>
2585   useobjnum ~
2586   \int_use:c
2587   { c__pdf_backend_object_ \tl_to_str:n {#1} _int }
2588   \str_case_e:nn
2589   { \prop_item:Nn \g__pdf_backend_object_prop {#1} }
2590   {
2591     { array } { { [ ~ \exp_not:n {#2} ~ ] } }
2592     { dict } { { << ~ \exp_not:n {#2} ~ >> } }
2593     { fstream }
2594     {
2595       stream ~ attr ~ { \__pdf_exp_not_i:nn #2 } ~
2596       file ~ { \__pdf_exp_not_ii:nn #2 }
2597     }
2598     { stream }
2599     {
2600       stream ~ attr ~ { \__pdf_exp_not_i:nn #2 } ~
2601       { \__pdf_exp_not_ii:nn #2 }
2602     }
2603   }
2604 }
2605 \cs_generate_variant:Nn \__pdf_backend_object_write:nn { nx }
2606 \cs_new:Npn \__pdf_exp_not_i:nn #1#2 { \exp_not:n {#1} }
2607 \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
2608 \cs_new_protected:Npn \__pdf_backend_object_now:nn #1#2
2609 {
2610 <*luatex>
2611   \tex_immediate:D \tex_pdfextension:D obj ~
2612 </luatex>
2613 <*pdftex>
2614   \tex_immediate:D \tex_pdfobj:D
2615 </pdftex>
2616   \str_case:nn
2617   {#1}
2618   {
2619     { array } { { [ ~ \exp_not:n {#2} ~ ] } }
2620     { dict } { { << ~ \exp_not:n {#2} ~ >> } }
2621     { fstream }
2622     {
2623       stream ~ attr ~ { \__pdf_exp_not_i:nn #2 } ~
2624       file ~ { \__pdf_exp_not_ii:nn #2 }
2625     }
2626     { stream }
2627     {

```

```

2628         stream ~ attr ~ { \_pdf_exp_not_i:nn #2 } ~
2629         { \_pdf_exp_not_ii:nn #2 }
2630     }
2631 }
2632 }
2633 \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.

```

2634 \cs_new:Npx \_pdf_backend_object_last:
2635 {
2636     \exp_not:N \int_value:w
2637 <*luatex>
2638     \exp_not:N \tex_pdffeedback:D lastobj ~
2639 </luatex>
2640 <*pdftex>
2641     \exp_not:N \tex_pdflastobj:D
2642 </pdftex>
2643     \c_space_tl 0 ~ R
2644 }
(End definition for \_pdf_backend_object_last:.)

```

\\_pdf\_backend\_pageobject\_ref:n The usual wrapper situation; the three spaces here are essential.

```

2645 \cs_new:Npx \_pdf_backend_pageobject_ref:n #1
2646 {
2647     \exp_not:N \int_value:w
2648 <*luatex>
2649     \exp_not:N \tex_pdffeedback:D pageref
2650 </luatex>
2651 <*pdftex>
2652     \exp_not:N \tex_pdfpageref:D
2653 </pdftex>
2654     \c_space_tl #1 \c_space_tl \c_space_tl \c_space_tl 0 ~ R
2655 }
(End definition for \_pdf_backend_pageobject_ref:n.)

```

### 6.3.4 Structure

\\_pdf\_backend\_compresslevel:n Simply pass data to the engine.

```

\_pdf_backend_compresslevel:n
\_pdf_backend_compress_objects:n
\_pdf_backend_objcompresslevel:n
2656 \cs_new_protected:Npn \_pdf_backend_compresslevel:n #1
2657 {
2658     \tex_global:D
2659 <*luatex>
2660     \tex_pdfvariable:D compresslevel
2661 </luatex>
2662 <*pdftex>
2663     \tex_pdfcompresslevel:D
2664 </pdftex>
2665     \int_value:w \int_eval:n {#1} \scan_stop:
2666 }
2667 \cs_new_protected:Npn \_pdf_backend_compress_objects:n #1
2668 {

```

```

2669     \bool_if:nTF {#1}
2670     { \__pdf_backend_objcompresslevel:n { 2 } }
2671     { \__pdf_backend_objcompresslevel:n { 0 } }
2672   }
2673 \cs_new_protected:Npn \__pdf_backend_objcompresslevel:n #1
2674 {
2675   \tex_global:D
2676   \*luatex
2677   \tex_pdfvariable:D objcompresslevel
2678   \*pdfTeX
2679   \*pdfTeX
2680   \tex_pdfobjcompresslevel:D
2681   \*pdfTeX
2682   #1 \scan_stop:
2683 }

```

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

`\__pdf_backend_version_major_gset:n` The availability of the primitive is not universal, so we have to test at load time.  
`\__pdf_backend_version_minor_gset:n`

```

2684 \cs_new_protected:Npx \__pdf_backend_version_major_gset:n #1
2685 {
2686   \*luatex
2687   \int_compare:nNnT \tex_luatexversion:D > { 106 }
2688   {
2689     \exp_not:N \tex_global:D \tex_pdfvariable:D majorversion
2690     \exp_not:N \int_eval:n {#1} \scan_stop:
2691   }
2692   \*pdfTeX
2693   \*pdfTeX
2694   \cs_if_exist:NT \tex_pdfmajorversion:D
2695   {
2696     \exp_not:N \tex_global:D \tex_pdfmajorversion:D
2697     \exp_not:N \int_eval:n {#1} \scan_stop:
2698   }
2699   \*pdfTeX
2700 }
2701 \cs_new_protected:Npn \__pdf_backend_version_minor_gset:n #1
2702 {
2703   \tex_global:D
2704   \*luatex
2705   \tex_pdfvariable:D minorversion
2706   \*pdfTeX
2707   \*pdfTeX
2708   \tex_pdfminorversion:D
2709   \*pdfTeX
2710   \int_eval:n {#1} \scan_stop:
2711 }

```

(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: 2712 \cs_new:Npx \__pdf_backend_version_major:
2713 {
2714 \*luatex

```

```

2715     \int_compare:nNnTF \tex luatexversion:D > { 106 }
2716     { \exp_not:N \tex_the:D \tex_pdfvariable:D majorversion }
2717     { 1 }
2718 </luatex>
2719 <*pdftex>
2720     \cs_if_exist:NTF \tex_pdfmajorversion:D
2721     { \exp_not:N \tex_the:D \tex_pdfmajorversion:D }
2722     { 1 }
2723 </pdftex>
2724 }
2725 \cs_new:Npn \__pdf_backend_version_minor:
2726 {
2727     \tex_the:D
2728 <*luatex>
2729     \tex_pdfvariable:D minorversion
2730 </luatex>
2731 <*pdftex>
2732     \tex_pdfminorversion:D
2733 </pdftex>
2734 }

```

(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:

```

2735 \cs_new_protected:Npn \__pdf_backend_bdc:nn #1#2
2736 { \__kernel_backend_literal_page:n { /#1 ~ #2 ~ BDC } }
2737 \cs_new_protected:Npn \__pdf_backend_emc:
2738 { \__kernel_backend_literal_page:n { EMC } }

```

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

```
2739 </luatex | pdftex>
```

## 6.4 dvipdfmx backend

```
2740 <*dvipdfmx | xetex>
```

\\_\_pdf\_backend:n A generic function for the backend PDF specials: used where we can.

```

\__pdf_backend:x
2741 \cs_new_protected:Npx \__pdf_backend:n #1
2742 { \__kernel_backend_literal:n { pdf: #1 } }
2743 \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

```

2744 \cs_new_protected:Npn \__pdf_backend_catalog_gput:nn #1#2
2745 { \__pdf_backend:n { put ~ @catalog << /#1 ~ #2 >> } }
2746 \cs_new_protected:Npn \__pdf_backend_info_gput:nn #1#2
2747 { \__pdf_backend:n { docinfo << /#1 ~ #2 >> } }

```

(End definition for \\_\_pdf\_backend\_catalog\_gput:nn and \\_\_pdf\_backend\_info\_gput:nn.)

## 6.4.2 Objects

```

\g__pdf_backend_object_int  For tracking objects to allow finalisation.
\g__pdf_backend_object_prop 2748 \int_new:N \g__pdf_backend_object_int
                             2749 \prop_new:N \g__pdf_backend_object_prop

(End definition for \g__pdf_backend_object_int and \g__pdf_backend_object_prop.)

\_pdf_backend_object_new:nn Objects are tracked at the macro level, but we don't have to do anything at this stage.
\_pdf_backend_object_ref:n 2750 \cs_new_protected:Npn \_pdf_backend_object_new:nn #1#2
                             2751 {
                             2752   \int_gincr:N \g__pdf_backend_object_int
                             2753   \int_const:cn
                             2754   { c__pdf_backend_object_ \tl_to_str:n {#1} _int }
                             2755   { \g__pdf_backend_object_int }
                             2756   \prop_gput:Nnn \g__pdf_backend_object_prop {#1} {#2}
                             2757 }
                             2758 \cs_new:Npn \_pdf_backend_object_ref:n #1
                             2759 { @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.)

\_pdf_backend_object_write:nn This is where we choose the actual type.
\_pdf_backend_object_write:nx 2760 \cs_new_protected:Npn \_pdf_backend_object_write:nn #1#2
\_pdf_backend_object_write:nnn 2761 {
\_pdf_backend_object_write_array:nn 2762   \exp_args:Nx \_pdf_backend_object_write:nnn
\_pdf_backend_object_write_dict:nn 2763   { \prop_item:Nn \g__pdf_backend_object_prop {#1} } {#1} {#2}
\_pdf_backend_object_write_fstream:nn 2764 }
\_pdf_backend_object_write_stream:nn 2765 \cs_generate_variant:Nn \_pdf_backend_object_write:nn { nx }
\_pdf_backend_object_write_stream:nnnn 2766 \cs_new_protected:Npn \_pdf_backend_object_write:nnn #1#2#3
2767 {
2768   \use:c { __pdf_backend_object_write_ #1 :nn }
2769   { \_pdf_backend_object_ref:n {#2} } {#3}
2770 }
2771 \cs_new_protected:Npn \_pdf_backend_object_write_array:nn #1#2
2772 {
2773   \_pdf_backend:x
2774   { obj ~ #1 ~ [ ~ \exp_not:n {#2} ~ ] }
2775 }
2776 \cs_new_protected:Npn \_pdf_backend_object_write_dict:nn #1#2
2777 {
2778   \_pdf_backend:x
2779   { obj ~ #1 ~ << ~ \exp_not:n {#2} ~ >> }
2780 }
2781 \cs_new_protected:Npn \_pdf_backend_object_write_fstream:nn #1#2
2782 { \_pdf_backend_object_write_stream:nnnn { f } {#1} #2 }
2783 \cs_new_protected:Npn \_pdf_backend_object_write_stream:nn #1#2
2784 { \_pdf_backend_object_write_stream:nnnn { } {#1} #2 }
2785 \cs_new_protected:Npn \_pdf_backend_object_write_stream:nnnn #1#2#3#4
2786 {
2787   \_pdf_backend:x
2788   {
2789     #1 stream ~ #2 ~
2790     ( \exp_not:n {#4} ) ~ << \exp_not:n {#3} >>
2791   }

```



```
2792 }
(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 2793 \cs_new_protected:Npn \_pdf_backend_object_now:nn #1#2
2794 {
2795   \int_gincr:N \g__pdf_backend_object_int
2796   \exp_args:Nnx \use:c { \_pdf_backend_object_write_ #1 :nn }
2797   { @pdf.obj \int_use:N \g__pdf_backend_object_int }
2798   {#2}
2799 }
2800 \cs_generate_variant:Nn \_pdf_backend_object_now:nn { nx }
(End definition for \_pdf_backend_object_now:nn.)
```

\\_pdf\_backend\_object\_last:

```
2801 \cs_new:Npn \_pdf_backend_object_last:
2802 { @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.

```
2803 \cs_new:Npn \_pdf_backend_pageobject_ref:n #1
2804 { @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.

```
2805 \int_new:N \g__pdf_backend_annotation_int
(End definition for \g__pdf_backend_annotation_int.)
```

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

```
2806 \cs_new_protected:Npn \_pdf_backend_annotation:nmmn #1#2#3#4
2807 {
2808   \int_gincr:N \g__pdf_backend_object_int
2809   \int_gset_eq:NN \g__pdf_backend_annotation_int \g__pdf_backend_object_int
2810   \_pdf_backend:x
2811   {
2812     ann ~ @pdf.obj \int_use:N \g__pdf_backend_object_int \c_space_tl
2813     width ~ \dim_eval:n {#1} ~
2814     height ~ \dim_eval:n {#2} ~
2815     depth ~ \dim_eval:n {#3} ~
2816     <</Type/Annot #4 >>
2817   }
2818 }
(End definition for \_pdf_backend_annotation:nmmn.)
```

\\_pdf\_backend\_annotation\_last:

```
2819 \cs_new:Npn \_pdf_backend_annotation_last:
2820 { @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.

```
2821 \int_new:N \g_pdf_backend_link_int
```

(End definition for `\g_pdf_backend_link_int:`)

`\_pdf_backend_link_begin_goto:nw` All created using the same internals.

`\_pdf_backend_link_begin_user:nw`

```
2822 \cs_new_protected:Npn \_pdf_backend_link_begin_goto:nw #1#2
```

`\_pdf_backend_link_begin:n`

```
2823 { \_pdf_backend_link_begin:n { #1 /Subtype /Link /A << /S /GoTo /D ( #2 ) >> } }
```

`\_pdf_backend_link_end:`

```
2824 \cs_new_protected:Npn \_pdf_backend_link_begin_user:nw #1#2
```

```
2825 { \_pdf_backend_link_begin:n {#1#2} }
```

```
2826 \cs_new_protected:Npx \_pdf_backend_link_begin:n #1
```

```
2827 {
```

```
2828   \int_compare:nNnF \c_kernel_sys_dvipdfmx_version_int < { 20201111 }
```

```
2829   {
```

```
2830     \exp_not:N \int_gincr:N \exp_not:N \g_pdf_backend_link_int
```

```
2831   }
```

```
2832 \_pdf_backend:x
```

```
2833 {
```

```
2834   bann ~
```

```
2835   \int_compare:nNnF \c_kernel_sys_dvipdfmx_version_int < { 20201111 }
```

```
2836   {
```

```
2837     @pdf.lnk
```

```
2838     \exp_not:N \int_use:N \exp_not:N \g_pdf_backend_link_int
```

```
2839     \c_space_tl
```

```
2840   }
```

```
2841   <<
```

```
2842     /Type /Annot
```

```
2843     #1
```

```
2844     >>
```

```
2845   }
```

```
2846 }
```

```
2847 \cs_new_protected:Npn \_pdf_backend_link_end:
```

```
2848 { \_pdf_backend:n { eann } }
```

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

`\_pdf_backend_link_last:` Available using the backend mechanism with a suitably-recent version.

```
2849 \cs_new:Npx \_pdf_backend_link_last:
```

```
2850 {
```

```
2851   \int_compare:nNnF \c_kernel_sys_dvipdfmx_version_int < { 20201111 }
```

```
2852   {
```

```
2853     @pdf.lnk
```

```
2854     \exp_not:N \int_use:N \exp_not:N \g_pdf_backend_link_int
```

```
2855   }
```

```
2856 }
```

(End definition for `\_pdf_backend_link_last:`)

`\_pdf_backend_link_margin:n` Pass to `dvipdfmx`.

```
2857 \cs_new_protected:Npn \_pdf_backend_link_margin:n #1
```

```
2858 { \_kernel_backend_literal:x { dvipdfmx:config~g~ \dim_eval:n {#1} } }
```

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

`\_pdf_backend_destination:nn`  
`\_pdf_backend_destination:nmmn`  
`\_pdf_backend_destination_aux:nmmn`

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 `TeX` by using the backend data for `@xpos` and `@ypos`. `/FitR` without rule spec doesn't work, so it falls back to `/Fit` here.

```

2859 \cs_new_protected:Npn \_pdf_backend_destination:nn #1#2
2860 {
2861   \_pdf_backend:x
2862   {
2863     dest ~ ( \exp_not:n {#1} )
2864     [
2865       @thispage
2866       \str_case:nmF {#2}
2867       {
2868         { xyz } { /XYZ ~ @xpos ~ @ypos ~ null }
2869         { fit } { /Fit }
2870         { fitb } { /FitB }
2871         { fitbh } { /FitBH }
2872         { fitbv } { /FitBV ~ @xpos }
2873         { fith } { /FitH ~ @ypos }
2874         { fitv } { /FitV ~ @xpos }
2875         { fitr } { /Fit }
2876       }
2877       { /XYZ ~ @xpos ~ @ypos ~ \fp_eval:n { (#2) / 100 } }
2878     ]
2879   }
2880 }
2881 \cs_new_protected:Npn \_pdf_backend_destination:nmmn #1#2#3#4
2882 {
2883   \exp_args:Ne \_pdf_backend_destination_aux:nmmn
2884   { \dim_eval:n {#2} } {#1} {#3} {#4}
2885 }
2886 \cs_new_protected:Npn \_pdf_backend_destination_aux:nmmn #1#2#3#4
2887 {
2888   \vbox_to_zero:n
2889   {
2890     \_kernel_kern:n {#4}
2891     \hbox:n
2892     {
2893       \_pdf_backend:n { obj ~ @pdf_ #2 _llx ~ @xpos }
2894       \_pdf_backend:n { obj ~ @pdf_ #2 _lly ~ @ypos }
2895     }
2896     \tex_vss:D
2897   }
2898   \_kernel_kern:n {#1}
2899   \vbox_to_zero:n
2900   {
2901     \_kernel_kern:n { -#3 }
2902     \hbox:n
2903     {
2904       \_pdf_backend:n
2905       {
2906         dest ~ (#2)
2907         [
2908           @thispage

```

```

2909             /FitR ~
2910             @pdf_ #2 _llx ~ @pdf_ #2 _lly ~
2911             @xpos ~ @ypos
2912         ]
2913     }
2914 }
2915 \tex_vss:D
2916 }
2917 \__kernel_kern:n { -#1 }
2918 }

```

(End definition for `\__pdf_backend_destination:nn`, `\__pdf_backend_destination:nmmn`, and `\__pdf_backend_destination_aux:nmmn`.)

#### 6.4.4 Structure

`\__pdf_backend_compresslevel:n`  
`\__pdf_backend_compress_objects:n`

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

```

2919 \cs_new_protected:Npn \__pdf_backend_compresslevel:n #1
2920 { \__kernel_backend_literal:x { dvipdfmx:config-z~ \int_eval:n {#1} } }
2921 \cs_new_protected:Npn \__pdf_backend_compress_objects:n #1
2922 {
2923   \bool_if:nF {#1}
2924   { \__kernel_backend_literal:n { dvipdfmx:config-C-0x40 } }
2925 }

```

(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.

```

2926 \cs_new_protected:Npn \__pdf_backend_version_major_gset:n #1
2927 {
2928   \cs_gset:Npx \__pdf_backend_version_major: { \int_eval:n {#1} }
2929   \__kernel_backend_literal:x { pdf:majorversion~ \__pdf_backend_version_major: }
2930 }
2931 \cs_new_protected:Npn \__pdf_backend_version_minor_gset:n #1
2932 {
2933   \cs_gset:Npx \__pdf_backend_version_minor: { \int_eval:n {#1} }
2934   \__kernel_backend_literal:x { pdf:minorversion~ \__pdf_backend_version_minor: }
2935 }

```

(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.

```

2936 \cs_new:Npn \__pdf_backend_version_major: { 1 }
2937 \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>.

```

2938 \cs_new_protected:Npn \__pdf_backend_bdc:nn #1#2
2939 { \__kernel_backend_literal_page:n { /#1 ~ #2 ~ BDC } }
2940 \cs_new_protected:Npn \__pdf_backend_emc:
2941 { \__kernel_backend_literal_page:n { EMC } }

```

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

2942 `</dvipdfmx | xetex>`

## 6.5 dvisvgm backend

2943 `<*dvisvgm>`

### 6.5.1 Catalogue entries

No-op.

2944 `\cs_new_protected:Npn \_pdf_backend_catalog_gput:nn #1#2 { }`

2945 `\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`.)

### 6.5.2 Objects

All no-ops here.

2946 `\cs_new_protected:Npn \_pdf_backend_object_new:nn #1#2 { }`

2947 `\cs_new:Npn \_pdf_backend_object_ref:n #1 { }`

2948 `\cs_new_protected:Npn \_pdf_backend_object_write:nn #1#2 { }`

2949 `\cs_new_protected:Npn \_pdf_backend_object_write:nx #1#2 { }`

2950 `\cs_new_protected:Npn \_pdf_backend_object_now:nn #1#2 { }`

2951 `\cs_new_protected:Npn \_pdf_backend_object_now:nx #1#2 { }`

2952 `\cs_new:Npn \_pdf_backend_object_last: { }`

2953 `\cs_new:Npn \_pdf_backend_pageobject_ref:n #1 { }`

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

### 6.5.3 Structure

These are all no-ops.

2954 `\cs_new_protected:Npn \_pdf_backend_compresslevel:n #1 { }`

2955 `\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!

2956 `\cs_new_protected:Npn \_pdf_backend_version_major_gset:n #1 { }`

2957 `\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!

2958 `\cs_new:Npn \_pdf_backend_version_major: { -1 }`

2959 `\cs_new:Npn \_pdf_backend_version_minor: { -1 }`

(End definition for `\_pdf_backend_version_major:` and `\_pdf_backend_version_minor:.`)

More no-ops.

2960 `\cs_new_protected:Npn \_pdf_backend_bdc:nn #1#2 { }`

2961 `\cs_new_protected:Npn \_pdf_backend_emc: { }`

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

2962 `</dvisvgm>`

2963 `</package>`

## 7 I3backend-opacity Implementation

```
2964 (*package)
2965 (@@=opacity)
```

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.

```
2966 (*dvips)
```

`\_opacity_backend_select:n` No stack so set values directly.

```
\_opacity_backend_select_aux:n
2967 \cs_new_protected:Npn \_opacity_backend_select:n #1
2968 {
2969   \exp_args:Nx \_opacity_backend_select_aux:n
2970   { \fp_eval:n { min(max(0,#1),1) } }
2971 }
2972 \cs_new_protected:Npn \_opacity_backend_select_aux:n #1
2973 {
2974   \_kernel_backend_postscript:n
2975   { #1 ~ .setfillconstantalpha ~ #1 ~ .setstrokeconstantalpha }
2976 }
```

(End definition for `\_opacity_backend_select:n` and `\_opacity_backend_select_aux:n`.)

`\_opacity_backend_fill:n` Similar to the above but with no stack and only adding to one or other of the entries.

```
\_opacity_backend_stroke:n
\_opacity_backend:nn
\_opacity_backend:xn
2977 \cs_new_protected:Npn \_opacity_backend_fill:n #1
2978 { \_opacity_backend:xn { \fp_eval:n { min(max(0,#1),1) } } { fill } }
2979 \cs_new_protected:Npn \_opacity_backend_stroke:n #1
2980 { \_opacity_backend:xn { \fp_eval:n { min(max(0,#1),1) } } { stroke } }
2981 \cs_new_protected:Npn \_opacity_backend:nn #1#2
2982 {
2983   \_kernel_backend_postscript:n { #1 ~ .set #2 constantalpha }
2984 }
2985 \cs_generate_variant:Nn \_opacity_backend:nn { x }
```

(End definition for `\_opacity_backend_fill:n`, `\_opacity_backend_stroke:n`, and `\_opacity_backend:nn`.)

```
2986 (/dvips)
```

```
2987 (*dvipdfmx | luatex | pdftex | xetex)
```

`\c_opacity_backend_stack_int` Set up a stack.

```
2988 \cs_if_exist:NT \pdfmanagement_add:nnn
2989 {
2990   \_kernel_color_backend_stack_init:Nnn \c_opacity_backend_stack_int
2991   { page ~ direct } { /opacity 1 ~ gs }
2992   \pdfmanagement_add:nnn { Page / Resources / ExtGState }
2993   { opacity 1 } { << /ca ~ 1 /CA ~ 1 >> }
2994 }
```

(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
2995 \tl_new:N \l__opacity_backend_fill_tl
2996 \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:
2997 \cs_new_protected:Npn \__opacity_backend_select:n #1
2998 {
2999   \exp_args:Nx \__opacity_backend_select_aux:n
3000   { \fp_eval:n { min(max(0,#1),1) } }
3001 }
3002 \cs_new_protected:Npn \__opacity_backend_select_aux:n #1
3003 {
3004   \tl_set:Nn \l__opacity_backend_fill_tl {#1}
3005   \tl_set:Nn \l__opacity_backend_stroke_tl {#1}
3006   \pdfmanagement_add:nnn { Page / Resources / ExtGState }
3007   { opacity #1 }
3008   { << /ca ~ #1 /CA ~ #1 >> }
3009   \__kernel_color_backend_stack_push:nm \c__opacity_backend_stack_int
3010   { /opacity #1 ~ gs }
3011   \group_insert_after:N \__opacity_backend_reset:
3012 }
3013 \cs_if_exist:NF \pdfmanagement_add:nnn
3014 {
3015   \cs_gset_protected:Npn \__opacity_backend_select_aux:n #1 { }
3016 }
3017 \cs_new_protected:Npn \__opacity_backend_reset:
3018 { \__kernel_color_backend_stack_pop:n \c__opacity_backend_stack_int }

(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:mn
  \__opacity_backend_fillstroke:xx
3019 \cs_new_protected:Npn \__opacity_backend_fill:n #1
3020 {
3021   \__opacity_backend_fill_stroke:xx
3022   { \fp_eval:n { min(max(0,#1),1) } }
3023   \l__opacity_backend_stroke_tl
3024 }
3025 \cs_new_protected:Npn \__opacity_backend_stroke:n #1
3026 {
3027   \__opacity_backend_fill_stroke:xx
3028   \l__opacity_backend_fill_tl
3029   { \fp_eval:n { min(max(0,#1),1) } }
3030 }
3031 \cs_new_protected:Npn \__opacity_backend_fill_stroke:nn #1#2
3032 {
3033   \str_if_eq:nnTF {#1} {#2}
3034   { \__opacity_backend_select_aux:n {#1} }
3035   {
3036     \tl_set:Nn \l__opacity_backend_fill_tl {#1}
3037     \tl_set:Nn \l__opacity_backend_stroke_tl {#2}

```

```

3038     \pdfmanagement_add:nnn { Page / Resources / ExtGState }
3039     { opacity.fill #1 }
3040     { << /ca ~ #1 >> }
3041     \pdfmanagement_add:nnn { Page / Resources / ExtGState }
3042     { opacity.stroke #1 }
3043     { << /CA ~ #2 >> }
3044     \__kernel_color_backend_stack_push:nn \c__opacity_backend_stack_int
3045     { /opacity.fill #1 ~ gs /opacity.stroke #2 ~ gs }
3046     \group_insert_after:N \__opacity_backend_reset:
3047   }
3048 }
3049 \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.)

```

3050 </dvipdfmx | luatex | pdftex | xetex>
3051 <*dvipdfmx | xdvipdfmx>

```

\\_\_opacity\_backend\_select:n Older backends have no stack support, so everything is done directly.

```

3052 \int_compare:nNnT \c__kernel_sys_dvipdfmx_version_int < { 20201111 }
3053 {
3054   \cs_gset_protected:Npn \__opacity_backend_select_aux:n #1
3055   {
3056     \tl_set:Nn \l__opacity_backend_fill_tl {#1}
3057     \tl_set:Nn \l__opacity_backend_stroke_tl {#1}
3058     \pdfmanagement_add:nnn { Page / Resources / ExtGState }
3059     { opacity #1 }
3060     { << /ca ~ #1 /CA ~ #1 >> }
3061     \__kernel_backend_literal_pdf:n { /opacity #1 ~ gs }
3062   }
3063   \cs_gset_protected:Npn \__opacity_backend_fill_stroke:nn #1#2
3064   {
3065     \str_if_eq:nnTF {#1} {#2}
3066     { \__opacity_backend_select_aux:n {#1} }
3067     {
3068       \tl_set:Nn \l__opacity_backend_fill_tl {#1}
3069       \tl_set:Nn \l__opacity_backend_stroke_tl {#2}
3070       \pdfmanagement_add:nnn { Page / Resources / ExtGState }
3071       { opacity.fill #1 }
3072       { << /ca ~ #1 >> }
3073       \pdfmanagement_add:nnn { Page / Resources / ExtGState }
3074       { opacity.stroke #1 }
3075       { << /CA ~ #2 >> }
3076       \__kernel_backend_literal_pdf:n
3077       { /opacity.fill #1 ~ gs /opacity.stroke #2 ~ gs }
3078     }
3079   }
3080 }

```

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

```

3081 </dvipdfmx | xdvipdfmx>
3082 <*dvisvgm>

```



```

\__opacity_backend_select:n Once again, we use a scope here. There is a general opacity function for SVG, but that
\__opacity_backend_fill:n is of course not set up using the stack.
\__opacity_backend_stroke:n
  \__opacity_backend:nn
3083 \cs_new_protected:Npn \__opacity_backend_select:n #1
3084   { \__opacity_backend:nn {#1} { } }
3085 \cs_new_protected:Npn \__opacity_backend_fill:n #1
3086   { \__opacity_backend:nn {#1} { fill- } }
3087 \cs_new_protected:Npn \__opacity_backend_stroke:n #1
3088   { \__opacity_backend:nn {#1} { stroke- } }
3089 \cs_new_protected:Npn \__opacity_backend:nn #1#2
3090   { \__kernel_backend_scope:x { #2 opacity = " \fp_eval:n { min(max(0,#1),1) } " } }

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

3091 </dvisvgm>
3092 </package>

```

## 8 l3backend-header Implementation

```

3093 <*dvips & header>

color.sc Empty definition for color at the top level.
3094 /color.sc { } def

(End definition for color.sc. This function is documented on page ??.)

TeXcolorseparation Support for separation/spot colors: this strange naming is so things work with the color
separation stack.
3095 TeXDict begin
3096 /TeXcolorseparation { setcolor } def
3097 end

(End definition for TeXcolorseparation and separation. These functions are documented on page ??.)

pdf.globaldict A small global dictionary for backend use.
3098 true setglobal
3099 /pdf.globaldict 4 dict def
3100 false setglobal

(End definition for pdf.globaldict. This function is documented on page ??.)

pdf.cvs Small utilities for PostScript manipulations. Conversion to DVI dimensions is done here
pdf.dvi.pt to allow for Resolution. The total height of a rectangle (an array) needs a little maths,
pdf.pt.dvi in contrast to simply extracting a value.
pdf.rect.ht
3101 /pdf.cvs { 65534 string cvs } def
3102 /pdf.dvi.pt { 72.27 mul Resolution div } def
3103 /pdf.pt.dvi { 72.27 div Resolution mul } def
3104 /pdf.rect.ht { dup 1 get neg exch 3 get add } def

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

pdf.linkmargin Settings which are defined up-front in SDict.
pdf.linkdp.pad
pdf.linkht.pad
3105 /pdf.linkmargin { 1 pdf.pt.dvi } def
3106 /pdf.linkdp.pad { 0 } def
3107 /pdf.linkht.pad { 0 } def

```

(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 3108 /pdf.rect
pdf.save.linkur 3109 { /Rect [ pdf.llx pdf.lly pdf.urx pdf.ury ] } def
pdf.llx         3110 /pdf.save.ll
pdf.lly         3111 {
pdf.urx         3112   currentpoint
pdf.ury         3113   /pdf.lly exch def
                3114   /pdf.llx exch def
                3115 }
                3116 def
pdf.save.ur     3117 /pdf.save.ur
                3118 {
                3119   currentpoint
                3120   /pdf.ury exch def
                3121   /pdf.urx exch def
                3122 }
                3123 def
pdf.save.linkll 3124 /pdf.save.linkll
                3125 {
                3126   currentpoint
                3127   pdf.linkmargin add
                3128   pdf.linkdp.pad add
                3129   /pdf.lly exch def
                3130   pdf.linkmargin sub
                3131   /pdf.llx exch def
                3132 }
                3133 def
pdf.save.linkur 3134 /pdf.save.linkur
                3135 {
                3136   currentpoint
                3137   pdf.linkmargin sub
                3138   pdf.linkht.pad sub
                3139   /pdf.ury exch def
                3140   pdf.linkmargin add
                3141   /pdf.urx exch def
                3142 }
                3143 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      3144 /pdf.dest.anchor
pdf.dev.y      3145 {
pdf.tmpa       3146   currentpoint exch
pdf.tmpb       3147   pdf.dvi.pt 72 add
pdf.tmpc
pdf.tmpd
```

```

3148 /pdf.dest.x exch def
3149 pdf.dvi.pt
3150 vsize 72 sub exch sub
3151 /pdf.dest.y exch def
3152 }
3153 def
3154 /pdf.dest.point
3155 { pdf.dest.x pdf.dest.y } def
3156 /pdf.dest2device
3157 {
3158 /pdf.dest.y exch def
3159 /pdf.dest.x exch def
3160 matrix currentmatrix
3161 matrix defaultmatrix
3162 matrix invertmatrix
3163 matrix concatmatrix
3164 cvx exec
3165 /pdf.dev.y exch def
3166 /pdf.dev.x exch def
3167 /pdf.tmpd exch def
3168 /pdf.tmpc exch def
3169 /pdf.tmpb exch def
3170 /pdf.tmpa exch def
3171 pdf.dest.x pdf.tmpa mul
3172 pdf.dest.y pdf.tmpc mul add
3173 pdf.dev.x add
3174 pdf.dest.x pdf.tmpb mul
3175 pdf.dest.y pdf.tmpd mul add
3176 pdf.dev.y add
3177 }
3178 def

```

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

<pre> pdf.bordertracking pdf.bordertracking.begin pdf.bordertracking.end pdf.leftboundary pdf.rightboundary pdf.brokenlink.rect pdf.brokenlink.skip pdf.brokenlink.dict pdf.bordertracking.endpage pdf.bordertracking.continue pdf.originx pdf.originy </pre>	<pre> 3179 /pdf.bordertracking false def 3180 /pdf.bordertracking.begin 3181 { 3182   SDict /pdf.bordertracking true put 3183   SDict /pdf.leftboundary undef 3184   SDict /pdf.rightboundary undef 3185   /a where 3186   { 3187     /a 3188     { 3189       currentpoint pop 3190       SDict /pdf.rightboundary known dup 3191       { 3192         SDict /pdf.rightboundary get 2 index lt 3193         { not } 3194         if </pre>	<p>To know where a breakable link can go, we need to track the boundary rectangle. That can be done by hooking into a and x operations: those names have to be retained. The boundary is stored at the end of the operation. Special effort is needed at the start and end of pages (or rather galleys), such that everything works properly.</p>
---	---	---

```

3195     }
3196     if
3197     { pop }
3198     { SDict exch /pdf.rightboundary exch put }
3199     ifelse
3200     moveto
3201     currentpoint pop
3202     SDict /pdf.leftboundary known dup
3203     {
3204         SDict /pdf.leftboundary get 2 index gt
3205         { not }
3206         if
3207     }
3208     if
3209     { pop }
3210     { SDict exch /pdf.leftboundary exch put }
3211     ifelse
3212 }
3213 put
3214 }
3215 if
3216 }
3217 def
3218 /pdf.bordertracking.end
3219 {
3220 /a where { /a { moveto } put } if
3221 /x where { /x { 0 exch rmoveto } put } if
3222 SDict /pdf.leftboundary known
3223 { pdf.outerbox 0 pdf.leftboundary put }
3224 if
3225 SDict /pdf.rightboundary known
3226 { pdf.outerbox 2 pdf.rightboundary put }
3227 if
3228 SDict /pdf.bordertracking false put
3229 }
3230 def
3231 /pdf.bordertracking.endpage
3232 {
3233 pdf.bordertracking
3234 {
3235 pdf.bordertracking.end
3236 true setglobal
3237 pdf.globaldict
3238 /pdf.brokenlink.rect [ pdf.outerbox aload pop ] put
3239 pdf.globaldict
3240 /pdf.brokenlink.skip pdf.baselineskip put
3241 pdf.globaldict
3242 /pdf.brokenlink.dict
3243 pdf.link.dict pdf.cvs put
3244 false setglobal
3245 mark pdf.link.dict cvx exec /Rect
3246 [
3247 pdf.llx
3248 pdf.lly

```

```

3249         pdf.outerbox 2 get pdf.linkmargin add
3250         currentpoint exch pop
3251         pdf.outerbox pdf.rect.ht sub pdf.linkmargin sub
3252     ]
3253     /ANN pdf.pdfmark
3254 }
3255 if
3256 }
3257 def
3258 /pdf.bordertracking.continue
3259 {
3260     /pdf.link.dict pdf.globaldict
3261     /pdf.brokenlink.dict get def
3262     /pdf.outerbox pdf.globaldict
3263     /pdf.brokenlink.rect get def
3264     /pdf.baselineskip pdf.globaldict
3265     /pdf.brokenlink.skip get def
3266     pdf.globaldict dup dup
3267     /pdf.brokenlink.dict undef
3268     /pdf.brokenlink.skip undef
3269     /pdf.brokenlink.rect undef
3270     currentpoint
3271     /pdf.originy exch def
3272     /pdf.originx exch def
3273     /a where
3274     {
3275         /a
3276         {
3277             moveto
3278             SDict
3279             begin
3280                 currentpoint pdf.originy ne exch
3281                 pdf.originx ne or
3282                 {
3283                     pdf.save.linkll
3284                     /pdf.lly
3285                     pdf.lly pdf.outerbox 1 get sub def
3286                     pdf.bordertracking.begin
3287                 }
3288                 if
3289                 end
3290             }
3291             put
3292         }
3293     if
3294     /x where
3295     {
3296         /x
3297         {
3298             0 exch rmoveto
3299             SDict
3300             begin
3301                 currentpoint
3302                 pdf.originy ne exch pdf.originx ne or

```

```

3303         {
3304             pdf.save.linkll
3305             /pdf.lly
3306             pdf.lly pdf.outerbox 1 get sub def
3307             pdf.bordertracking.begin
3308         }
3309         if
3310         end
3311     }
3312     put
3313 }
3314 if
3315 }
3316 def

```

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

**pdf.breaklink** 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.

```

3317 /pdf.breaklink
3318 {
3319     pop
3320     counttomark 2 mod 0 eq
3321     {
3322         counttomark /pdf.count exch def
3323         {
3324             pdf.count 0 eq { exit } if
3325             counttomark 2 roll
3326             1 index /Rect eq
3327             {
3328                 dup 4 array copy
3329                 dup dup
3330                 1 get
3331                 pdf.outerbox pdf.rect.ht
3332                 pdf.linkmargin 2 mul add sub
3333                 3 exch put
3334             dup
3335                 pdf.outerbox 2 get
3336                 pdf.linkmargin add
3337                 2 exch put
3338             dup dup
3339                 3 get
3340                 pdf.outerbox pdf.rect.ht
3341                 pdf.linkmargin 2 mul add add
3342                 1 exch put
3343             /pdf.currentrect exch def
3344             pdf.breaklink.write
3345             {
3346                 pdf.currentrect
3347                 dup
3348                 pdf.outerbox 0 get

```

```

3349         pdf.linkmargin sub
3350         0 exch put
3351     dup
3352         pdf.outerbox 2 get
3353         pdf.linkmargin add
3354         2 exch put
3355     dup dup
3356         1 get
3357         pdf.baselineskip add
3358         1 exch put
3359     dup dup
3360         3 get
3361         pdf.baselineskip add
3362         3 exch put
3363     /pdf.currentrect exch def
3364     pdf.breaklink.write
3365 }
3366     1 index 3 get
3367     pdf.linkmargin 2 mul add
3368     pdf.outerbox pdf.rect.ht add
3369     2 index 1 get sub
3370     pdf.baselineskip div round cvi 1 sub
3371     exch
3372 repeat
3373 pdf.currentrect
3374 dup
3375     pdf.outerbox 0 get
3376     pdf.linkmargin sub
3377     0 exch put
3378 dup dup
3379     1 get
3380     pdf.baselineskip add
3381     1 exch put
3382 dup dup
3383     3 get
3384     pdf.baselineskip add
3385     3 exch put
3386 dup 2 index 2 get 2 exch put
3387 /pdf.currentrect exch def
3388 pdf.breaklink.write
3389 SDict /pdf.pdfmark.good false put
3390 exit
3391 }
3392 { pdf.count 2 sub /pdf.count exch def }
3393 ifelse
3394 }
3395 loop
3396 }
3397 if
3398 /ANN
3399 }
3400 def
3401 /pdf.breaklink.write
3402 {

```

```

3403     counttomark 1 sub
3404     index /_objdef eq
3405     {
3406         counttomark -2 roll
3407         dup wcheck
3408         {
3409             readonly
3410             counttomark 2 roll
3411         }
3412         { pop pop }
3413         ifelse
3414     }
3415     if
3416     counttomark 1 add copy
3417     pop pdf.currentrect
3418     /ANN pdfmark
3419 }
3420 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 function. Only mark types which are known are altered. At present, this is purely ANN marks, which are measured relative to the size of the baseline skip. If they are more than pdf.outerbox one apparent line high, breaking is applied. pdf.baselineskip pdf.pdfmark.dict

```

3421 /pdf.pdfmark
3422 {
3423     SDict /pdf.pdfmark.good true put
3424     dup /ANN eq
3425     {
3426         pdf.pdfmark.store
3427         pdf.pdfmark.dict
3428         begin
3429             Subtype /Link eq
3430             currentdict /Rect known and
3431             SDict /pdf.outerbox known and
3432             SDict /pdf.baselineskip known and
3433             {
3434                 Rect 3 get
3435                 pdf.linkmargin 2 mul add
3436                 pdf.outerbox pdf.rect.ht add
3437                 Rect 1 get sub
3438                 pdf.baselineskip div round cvi 0 gt
3439                 { pdf.breaklink }
3440                 if
3441             }
3442             if
3443             end
3444             SDict /pdf.outerbox undef
3445             SDict /pdf.baselineskip undef
3446             currentdict /pdf.pdfmark.dict undef
3447         }
3448         if

```



```

3449 pdf.pdfmark.good
3450 { pdfmark }
3451 { cleartomark }
3452 ifelse
3453 }
3454 def
3455 /pdf.pdfmark.store
3456 {
3457 /pdf.pdfmark.dict 65534 dict def
3458 counttomark 1 add copy
3459 pop
3460 {
3461 dup mark eq
3462 {
3463 pop
3464 exit
3465 }
3466 {
3467 pdf.pdfmark.dict
3468 begin def end
3469 }
3470 ifelse
3471 }
3472 loop
3473 }
3474 def

```

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

```

3475 </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.

<code>\_</code> .....	147	<code>\_box_backend_rotate_aux:Nn</code> ...	228, 276, 333
<b>A</b>		<code>\_box_backend_scale:Nnn</code> .....	245, 304, 348, 425
<code>\AtBeginDvi</code> .....	51	<code>\l_box_backend_sin_fp</code> .....	276
<b>B</b>		<code>\g_box_clip_path_int</code> .....	362
bool commands:		<b>C</b>	
<code>\bool_gset_false:N</code> .....	1162, 1181, 1204, 1226, 1242, 1343, 1580, 1616, 2181, 2227	char commands:	<code>\char_set_catcode_space:n</code> .....
<code>\bool_gset_true:N</code> .....	1160, 1229, 1341, 1595, 2174, 2180	clist commands:	<code>\clist_map_function:nN</code> ... 1250, 1374
<code>\bool_if:NTF</code> .....	60, 668, 1172, 1176, 1192, 1195, 1199, 1210, 1217, 1221, 1233, 1237, 1354, 1359, 1364, 1554, 1599, 1712, 1747, 1857, 1899, 2169, 2184, 2189, 2194	<code>\clist_map_function:nn</code> .....	1623
<code>\bool_if:nTF</code> .....	2403, 2669, 2923	color internal commands:	<code>\_color_backend:nnn</code> .....
<code>\bool_lazy_or:nnTF</code> .....	1739, 1892	<code>\_color_backend_cmyk:w</code> .....	1048
<code>\bool_new:N</code> .....	1163, 1230, 1344, 1596, 2157, 2158	<code>\_color_backend_devicen_init:n</code>	896
<code>\bool_set_false:N</code> .....	1722, 1824, 1917, 1981	<code>\_color_backend_devicen_</code>	
box commands:		<code>init:nnm</code> .....	814, 896
<code>\box_dp:N</code> .....	217, 219, 267, 269, 324, 326, 373, 375, 377, 379, 2206, 2239, 2240, 2265	<code>\_color_backend_devicen_init:w</code>	896
<code>\box_ht:N</code> .....	219, 269, 326, 377, 379, 1759, 1954, 2211, 2250, 2251, 2267	<code>\_color_backend_fill:n</code> .....	955, 982, 1012, 1030, 1037
<code>\box_if_empty:NTF</code> .....	2300	<code>\_color_backend_fill_cmyk:n</code> ...	955, 989, 1012, 1037
<code>\box_move_down:nn</code> .....	2131, 2206	<code>\_color_backend_fill_devicen:nn</code>	981, 1003, 1029, 1099
<code>\box_move_up:nn</code> .....	2133, 2211	<code>\_color_backend_fill_gray:n</code> ...	955, 989, 1012, 1037
<code>\box_new:N</code> .....	2016, 2121, 2122	<code>\_color_backend_fill_rgb:n</code> ...	955, 989, 1012, 1037
<code>\box_set_dp:Nn</code> .....	1679	<code>\_color_backend_fill_separation:nn</code>	981, 989, 1029, 1099
<code>\box_set_ht:Nn</code> .....	1678	<code>\l_color_backend_fill_tl</code> .....	630, 640, 963, 978
<code>\box_set_wd:Nn</code> .....	281, 1677	<code>\c_color_backend_main_stack_int</code>	509
<code>\box_use:N</code> .....	224, 242, 256, 272, 299, 313, 329, 345, 357, 408, 422, 441, 1294, 1489, 1680, 2162	<code>\_color_backend_pickup:N</code> ..	449, 472
<code>\box_wd:N</code> .....	218, 226, 268, 274, 325, 331, 374, 376, 1758, 1953	<code>\_color_backend_pickup:w</code>	14, 449, 472
box internal commands:		<code>\_color_backend_reset:</code> .....	612, 632, 649, 966, 979, 989, 1021, 1046
<code>\_box_backend_clip:N</code> .....	206, 261, 318, 362	<code>\_color_backend_rgb:w</code> .....	1072
<code>\l_box_backend_cos_fp</code> .....	276	<code>\_color_backend_select:n</code> ..	612, 664
<code>\_box_backend_rotate:Nn</code> .....	228, 276, 333, 412	<code>\_color_backend_select:nn</code> ..	632, 841
		<code>\_color_backend_select_cmyk:n</code> ..	612, 632, 649
		<code>\_color_backend_select_devicen:nn</code>	663, 834, 840, 947
		<code>\_color_backend_select_gray:n</code> ..	612, 632, 649

<code>\__color_backend_select_rgb:n</code> . . . . .	<code>\l__color_backend_stroke_tl</code> . . . . .
. . . . . <a href="#">612</a> , <a href="#">632</a> , <a href="#">649</a>	. . . . . <a href="#">630</a> , <a href="#">641</a> , <a href="#">965</a> , <a href="#">976</a>
<code>\__color_backend_select_separation:nn</code>	<code>\g_color_model_int</code> <a href="#">684</a> , <a href="#">820</a> , <a href="#">865</a> , <a href="#">933</a>
. . . . . <a href="#">663</a> , <a href="#">834</a> , <a href="#">840</a> , <a href="#">947</a>	<code>\c_color_model_range_CIELAB_tl</code> . . . . .
<code>\__color_backend_separation_-</code>	. . . . . <a href="#">775</a> , <a href="#">810</a> , <a href="#">885</a> , <a href="#">892</a>
init:n . . . . . <a href="#">666</a> , <a href="#">843</a> , <a href="#">920</a> , <a href="#">944</a>	<code>color.sc</code> . . . . . <a href="#">612</a> , <a href="#">3094</a>
<code>\__color_backend_separation_-</code>	cs commands:
init:nnn . . . . . <a href="#">666</a>	<code>\cs_generate_variant:Nn</code> . . . . . <a href="#">49</a> , <a href="#">56</a> ,
<code>\__color_backend_separation_-</code>	<a href="#">59</a> , <a href="#">92</a> , <a href="#">131</a> , <a href="#">136</a> , <a href="#">163</a> , <a href="#">194</a> , <a href="#">200</a> , <a href="#">562</a> ,
init:nnnn . . . . . <a href="#">666</a>	<a href="#">599</a> , <a href="#">677</a> , <a href="#">1109</a> , <a href="#">1304</a> , <a href="#">1498</a> , <a href="#">1871</a> ,
<code>\__color_backend_separation_-</code>	<a href="#">1928</a> , <a href="#">1944</a> , <a href="#">2020</a> , <a href="#">2057</a> , <a href="#">2116</a> , <a href="#">2605</a> ,
init:nnnnn . . . . . <a href="#">666</a> , <a href="#">836</a> , <a href="#">843</a>	<a href="#">2633</a> , <a href="#">2743</a> , <a href="#">2765</a> , <a href="#">2800</a> , <a href="#">2985</a> , <a href="#">3049</a>
<code>\__color_backend_separation_-</code>	<code>\cs_gset:Npx</code> . . . . . <a href="#">2415</a> , <a href="#">2419</a> , <a href="#">2928</a> , <a href="#">2933</a>
init:nw . . . . . <a href="#">666</a>	<code>\cs_gset_eq:NN</code> . . . . . <a href="#">656</a> ,
<code>\__color_backend_separation_-</code>	<a href="#">657</a> , <a href="#">950</a> , <a href="#">996</a> , <a href="#">997</a> , <a href="#">1003</a> , <a href="#">1005</a> , <a href="#">1007</a>
init:w . . . . . <a href="#">666</a>	<code>\cs_gset_protected:Npn</code> . . . . .
<code>\__color_backend_separation_-</code>	. . . . . <a href="#">544</a> , <a href="#">651</a> , <a href="#">658</a> , <a href="#">949</a> ,
init_/DeviceCMYK:nnn . . . . . <a href="#">666</a>	<a href="#">991</a> , <a href="#">998</a> , <a href="#">1000</a> , <a href="#">1002</a> , <a href="#">3015</a> , <a href="#">3054</a> , <a href="#">3063</a>
<code>\__color_backend_separation_-</code>	<code>\cs_if_exist:NTF</code> . . . . .
init_/DeviceGray:nnn . . . . . <a href="#">666</a>	. . . . . <a href="#">27</a> , <a href="#">50</a> , <a href="#">450</a> , <a href="#">473</a> , <a href="#">532</a> ,
<code>\__color_backend_separation_-</code>	<a href="#">859</a> , <a href="#">927</a> , <a href="#">2296</a> , <a href="#">2694</a> , <a href="#">2720</a> , <a href="#">2988</a> , <a href="#">3013</a>
init_/DeviceRGB:nnn . . . . . <a href="#">666</a>	<code>\cs_if_exist_use:NTF</code> . . . . . <a href="#">38</a> , <a href="#">690</a>
<code>\__color_backend_separation_-</code>	<code>\cs_new:Npn</code> . . . . . <a href="#">699</a> , <a href="#">701</a> , <a href="#">703</a> ,
init_aux:nnnnn . . . . . <a href="#">666</a>	<a href="#">705</a> , <a href="#">712</a> , <a href="#">718</a> , <a href="#">720</a> , <a href="#">726</a> , <a href="#">743</a> , <a href="#">750</a> ,
<code>\__color_backend_separation_-</code>	<a href="#">752</a> , <a href="#">938</a> , <a href="#">1255</a> , <a href="#">1379</a> , <a href="#">1627</a> , <a href="#">1957</a> ,
init_CIELAB:nnn . . . . . <a href="#">666</a> , <a href="#">836</a> , <a href="#">843</a>	<a href="#">1966</a> , <a href="#">2010</a> , <a href="#">2035</a> , <a href="#">2117</a> , <a href="#">2119</a> , <a href="#">2152</a> ,
<code>\__color_backend_separation_-</code>	<a href="#">2321</a> , <a href="#">2421</a> , <a href="#">2422</a> , <a href="#">2575</a> , <a href="#">2606</a> , <a href="#">2607</a> ,
init_CIELAB:nnnnn . . . . . <a href="#">837</a>	<a href="#">2725</a> , <a href="#">2758</a> , <a href="#">2801</a> , <a href="#">2803</a> , <a href="#">2819</a> , <a href="#">2936</a> ,
<code>\__color_backend_separation_-</code>	<a href="#">2937</a> , <a href="#">2947</a> , <a href="#">2952</a> , <a href="#">2953</a> , <a href="#">2958</a> , <a href="#">2959</a>
init_count:n . . . . . <a href="#">666</a>	<code>\cs_new:Npx</code> . . . . .
<code>\__color_backend_separation_-</code>	. . . . . <a href="#">2442</a> , <a href="#">2477</a> , <a href="#">2634</a> , <a href="#">2645</a> , <a href="#">2712</a> , <a href="#">2849</a>
init_count:w . . . . . <a href="#">666</a>	<code>\cs_new_eq:NN</code> . . . . .
<code>\__color_backend_separation_-</code>	. . . . . <a href="#">46</a> , <a href="#">51</a> , <a href="#">52</a> , <a href="#">665</a> , <a href="#">842</a> , <a href="#">944</a> ,
init_Device:Nn . . . . . <a href="#">666</a>	<a href="#">985</a> , <a href="#">986</a> , <a href="#">1033</a> , <a href="#">1034</a> , <a href="#">1101</a> , <a href="#">1102</a> ,
<code>\g_color_backend_stack_int</code> . . . . . <a href="#">509</a>	<a href="#">1108</a> , <a href="#">1303</a> , <a href="#">1309</a> , <a href="#">1310</a> , <a href="#">1497</a> , <a href="#">1504</a> ,
<code>\l_color_backend_stack_int</code> . . . . .	<a href="#">1689</a> , <a href="#">1718</a> , <a href="#">1769</a> , <a href="#">1770</a> , <a href="#">1812</a> , <a href="#">1820</a> ,
. . . . . <a href="#">506</a> , <a href="#">534</a> , <a href="#">540</a> , <a href="#">642</a> , <a href="#">646</a> , <a href="#">964</a> , <a href="#">977</a>	<a href="#">1842</a> , <a href="#">1913</a> , <a href="#">1970</a> , <a href="#">1977</a> , <a href="#">2009</a> , <a href="#">2162</a>
<code>\__color_backend_stroke:n</code> . . . . .	<code>\cs_new_protected:Npn</code> . . . . . <a href="#">47</a> ,
. . . . . <a href="#">955</a> , <a href="#">984</a> , <a href="#">989</a>	<a href="#">54</a> , <a href="#">57</a> , <a href="#">65</a> , <a href="#">71</a> , <a href="#">76</a> , <a href="#">78</a> , <a href="#">82</a> , <a href="#">93</a> , <a href="#">103</a> ,
<code>\__color_backend_stroke_cmyk:n</code> . . . . .	<a href="#">112</a> , <a href="#">121</a> , <a href="#">134</a> , <a href="#">137</a> , <a href="#">139</a> , <a href="#">141</a> , <a href="#">161</a> ,
. . . . . <a href="#">955</a> , <a href="#">1012</a> , <a href="#">1048</a>	<a href="#">166</a> , <a href="#">175</a> , <a href="#">185</a> , <a href="#">195</a> , <a href="#">206</a> , <a href="#">228</a> , <a href="#">230</a> ,
<code>\__color_backend_stroke_cmyk:w</code> <a href="#">1048</a>	<a href="#">245</a> , <a href="#">261</a> , <a href="#">276</a> , <a href="#">278</a> , <a href="#">304</a> , <a href="#">318</a> , <a href="#">333</a> ,
<code>\__color_backend_stroke_devicen:nn</code>	<a href="#">335</a> , <a href="#">348</a> , <a href="#">362</a> , <a href="#">412</a> , <a href="#">425</a> , <a href="#">449</a> , <a href="#">467</a> ,
. . . . . <a href="#">981</a> , <a href="#">1007</a> , <a href="#">1029</a> , <a href="#">1099</a>	<a href="#">472</a> , <a href="#">480</a> , <a href="#">510</a> , <a href="#">553</a> , <a href="#">563</a> , <a href="#">575</a> , <a href="#">589</a> ,
<code>\__color_backend_stroke_gray:n</code> . . . . .	<a href="#">600</a> , <a href="#">612</a> , <a href="#">614</a> , <a href="#">616</a> , <a href="#">618</a> , <a href="#">626</a> , <a href="#">632</a> ,
. . . . . <a href="#">955</a> , <a href="#">1012</a> , <a href="#">1048</a>	<a href="#">634</a> , <a href="#">636</a> , <a href="#">638</a> , <a href="#">645</a> , <a href="#">663</a> , <a href="#">678</a> , <a href="#">768</a> ,
<code>\__color_backend_stroke_gray_-</code>	<a href="#">814</a> , <a href="#">834</a> , <a href="#">835</a> , <a href="#">836</a> , <a href="#">837</a> , <a href="#">840</a> , <a href="#">843</a> ,
aux:n . . . . . <a href="#">1048</a>	<a href="#">870</a> , <a href="#">874</a> , <a href="#">896</a> , <a href="#">955</a> , <a href="#">957</a> , <a href="#">959</a> , <a href="#">961</a> ,
<code>\__color_backend_stroke_rgb:n</code> . . . . .	<a href="#">968</a> , <a href="#">970</a> , <a href="#">972</a> , <a href="#">974</a> , <a href="#">981</a> , <a href="#">983</a> , <a href="#">1012</a> ,
. . . . . <a href="#">955</a> , <a href="#">1012</a> , <a href="#">1048</a>	<a href="#">1014</a> , <a href="#">1016</a> , <a href="#">1018</a> , <a href="#">1023</a> , <a href="#">1025</a> , <a href="#">1027</a> ,
<code>\__color_backend_stroke_rgb:w</code> . <a href="#">1048</a>	<a href="#">1029</a> , <a href="#">1031</a> , <a href="#">1037</a> , <a href="#">1039</a> , <a href="#">1041</a> , <a href="#">1043</a> ,
<code>\__color_backend_stroke_separation:nn</code>	<a href="#">1048</a> , <a href="#">1050</a> , <a href="#">1061</a> , <a href="#">1069</a> , <a href="#">1071</a> , <a href="#">1073</a> ,
. . . . . <a href="#">981</a> , <a href="#">989</a> , <a href="#">1029</a> , <a href="#">1099</a>	<a href="#">1099</a> , <a href="#">1100</a> , <a href="#">1110</a> , <a href="#">1115</a> , <a href="#">1120</a> , <a href="#">1122</a> ,
	<a href="#">1124</a> , <a href="#">1132</a> , <a href="#">1140</a> , <a href="#">1149</a> , <a href="#">1159</a> , <a href="#">1161</a> ,

1164, 1166, 1183, 1188, 1206, 1228,  
 1231, 1244, 1257, 1262, 1264, 1266,  
 1268, 1270, 1272, 1274, 1276, 1281,  
 1305, 1307, 1311, 1316, 1321, 1331,  
 1340, 1342, 1345, 1347, 1349, 1351,  
 1356, 1361, 1366, 1368, 1381, 1386,  
 1388, 1390, 1392, 1394, 1396, 1398,  
 1400, 1411, 1436, 1448, 1460, 1472,  
 1479, 1499, 1505, 1510, 1515, 1526,  
 1536, 1546, 1548, 1550, 1552, 1583,  
 1585, 1590, 1592, 1594, 1597, 1618,  
 1629, 1642, 1644, 1646, 1648, 1650,  
 1652, 1654, 1656, 1658, 1666, 1690,  
 1704, 1719, 1731, 1736, 1764, 1776,  
 1789, 1799, 1814, 1821, 1829, 1840,  
 1844, 1847, 1862, 1872, 1907, 1914,  
 1920, 1926, 1929, 1936, 1945, 1950,  
 1958, 1971, 1978, 1984, 1986, 1988,  
 1999, 2018, 2021, 2023, 2027, 2037,  
 2058, 2063, 2068, 2073, 2083, 2088,  
 2096, 2124, 2129, 2161, 2163, 2165,  
 2167, 2172, 2187, 2192, 2229, 2258,  
 2277, 2286, 2323, 2330, 2356, 2361,  
 2389, 2401, 2413, 2417, 2423, 2425,  
 2429, 2453, 2455, 2457, 2468, 2488,  
 2498, 2521, 2535, 2545, 2556, 2577,  
 2608, 2656, 2667, 2673, 2701, 2735,  
 2737, 2744, 2746, 2750, 2760, 2766,  
 2771, 2776, 2781, 2783, 2785, 2793,  
 2806, 2822, 2824, 2847, 2857, 2859,  
 2881, 2886, 2919, 2921, 2926, 2931,  
 2938, 2940, 2944, 2945, 2946, 2948,  
 2949, 2950, 2951, 2954, 2955, 2956,  
 2957, 2960, 2961, 2967, 2972, 2977,  
 2979, 2981, 2997, 3002, 3017, 3019,  
 3025, 3031, 3083, 3085, 3087, 3089

`\cs_new_protected:Npx` . . . . .  
 . . . . . 513, 666, 1084, 2684, 2741, 2826

`\cs_set:Npn` . . . . . 145

`\cs_set_eq:NN` . . . . . 2317, 2318

`\cs_set_protected:Npn` . . . . . 452, 475

## D

dim commands:

`\dim_eval:n` . . . . . 2127, 2359,  
 2437, 2438, 2439, 2496, 2531, 2532,  
 2533, 2813, 2814, 2815, 2858, 2884

`\dim_max:nn` . . . . . 2237, 2248

`\dim_set:Nn` . . . . . 1758, 1759, 1953, 1954

`\dim_to_decimal:n` . . . 373, 374, 375,  
 376, 377, 379, 1508, 1513, 1519,  
 1520, 1521, 1522, 1531, 1532, 1533,  
 1624, 1643, 2004, 2005, 2235, 2246,  
 2264, 2265, 2266, 2267, 2271, 2327

`\dim_to_decimal_in_bp:n` . . . . .  
 . . . . . 217, 218, 219, 267, 268, 269,  
 324, 325, 326, 1128, 1129, 1136,  
 1137, 1144, 1145, 1153, 1154, 1155,  
 1252, 1256, 1260, 1314, 1319, 1325,  
 1326, 1327, 1335, 1336, 1376, 1380,  
 1384, 1628, 1695, 1696, 1697, 1698,  
 1834, 1835, 1836, 1837, 1886, 1887,  
 1888, 1889, 1993, 1994, 1995, 1996

draw internal commands:

`\__draw_align_currentpoint_` . . . . . 34

`\__draw_backend_add_to_path:n` . . .  
 . . . . . 1505, 1551

`\__draw_backend_begin:` . . . . .

. . . . . 1110, 1305, 1499

`\__draw_backend_box_use:Nnnnn` . . .  
 . . . . . 30, 1281, 1479, 1666

`\__draw_backend_cap_but:` . . . . .  
 . . . . . 1244, 1368, 1618

`\__draw_backend_cap_rectangle:` . .  
 . . . . . 1244, 1368, 1618

`\__draw_backend_cap_round:` . . . . .  
 . . . . . 1244, 1368, 1618

`\__draw_backend_clip:` 1164, 1345, 1550

`\__draw_backend_closepath:` . . . . .  
 . . . . . 1164, 1345, 1550

`\__draw_backend_closestroke:` . . .  
 . . . . . 1164, 1345, 1550

`\__draw_backend_cm:n` 1276, 1289,  
 1290, 1291, 1400, 1483, 1658, 1669

`\__draw_backend_cm_aux:n` . . . . . 1400

`\__draw_backend_cm_decompose:n` . . . . .  
 . . . . . 1406, 1435

`\__draw_backend_cm_decompose_`  
`auxi:n` . . . . . 1435

`\__draw_backend_cm_decompose_`  
`auxii:n` . . . . . 1435

`\__draw_backend_cm_decompose_`  
`auxiii:n` . . . . . 1435

`\__draw_backend_curveto:n` . . . . .  
 . . . . . 1124, 1311, 1505

`\__draw_backend_dash:n` . . . . .  
 . . . . . 1244, 1368, 1618

`\__draw_backend_dash_aux:nn` . . . 1618

`\__draw_backend_dash_pattern:nn` . . .  
 . . . . . 1244, 1368, 1618

`\__draw_backend_discardpath:` . . .  
 . . . . . 1164, 1345, 1550

`\__draw_backend_end:` 1110, 1305, 1499

`\__draw_backend_evenodd_rule:` . . .  
 . . . . . 1159, 1340, 1546

`\__draw_backend_fill:` 1164, 1345, 1550

`\__draw_backend_fillstroke:` . . . . .  
 . . . . . 1164, 1345, 1550

<code>\__draw_backend_join_bevel:</code> . . . .	
. . . . .	<a href="#">1244</a> , <a href="#">1368</a> , <a href="#">1618</a>
<code>\__draw_backend_join_miter:</code> . . . .	
. . . . .	<a href="#">1244</a> , <a href="#">1368</a> , <a href="#">1618</a>
<code>\__draw_backend_join_round:</code> . . . .	
. . . . .	<a href="#">1244</a> , <a href="#">1368</a> , <a href="#">1618</a>
<code>\__draw_backend_lineto:nn</code> . . . . .	
. . . . .	<a href="#">1124</a> , <a href="#">1311</a> , <a href="#">1505</a>
<code>\__draw_backend_linewidth:n</code> . . . .	
. . . . .	<a href="#">1244</a> , <a href="#">1368</a> , <a href="#">1618</a>
<code>\__draw_backend_literal:n</code> . . . . .	
. . . . .	<a href="#">1108</a> , <a href="#">1113</a> , <a href="#">1117</a> , <a href="#">1121</a> , <a href="#">1123</a> , <a href="#">1126</a> , <a href="#">1134</a> , <a href="#">1142</a> , <a href="#">1151</a> , <a href="#">1165</a> , <a href="#">1168</a> , <a href="#">1169</a> , <a href="#">1170</a> , <a href="#">1171</a> , <a href="#">1174</a> , <a href="#">1180</a> , <a href="#">1190</a> , <a href="#">1197</a> , <a href="#">1203</a> , <a href="#">1208</a> , <a href="#">1213</a> , <a href="#">1214</a> , <a href="#">1215</a> , <a href="#">1216</a> , <a href="#">1219</a> , <a href="#">1225</a> , <a href="#">1235</a> , <a href="#">1241</a> , <a href="#">1246</a> , <a href="#">1259</a> , <a href="#">1263</a> , <a href="#">1265</a> , <a href="#">1267</a> , <a href="#">1269</a> , <a href="#">1271</a> , <a href="#">1273</a> , <a href="#">1275</a> , <a href="#">1278</a> , <a href="#">1283</a> , <a href="#">1284</a> , <a href="#">1285</a> , <a href="#">1286</a> , <a href="#">1287</a> , <a href="#">1288</a> , <a href="#">1292</a> , <a href="#">1293</a> , <a href="#">1295</a> , <a href="#">1296</a> , <a href="#">1297</a> , <a href="#">1298</a> , <a href="#">1299</a> , <a href="#">1303</a> , <a href="#">1313</a> , <a href="#">1318</a> , <a href="#">1323</a> , <a href="#">1333</a> , <a href="#">1346</a> , <a href="#">1348</a> , <a href="#">1350</a> , <a href="#">1353</a> , <a href="#">1358</a> , <a href="#">1363</a> , <a href="#">1367</a> , <a href="#">1370</a> , <a href="#">1383</a> , <a href="#">1387</a> , <a href="#">1389</a> , <a href="#">1391</a> , <a href="#">1393</a> , <a href="#">1395</a> , <a href="#">1397</a> , <a href="#">1399</a> , <a href="#">1497</a> , <a href="#">1557</a> , <a href="#">1576</a> , <a href="#">1602</a>
<code>\__draw_backend_miterlimit:n</code> . . . .	
. . . . .	<a href="#">1244</a> , <a href="#">1368</a> , <a href="#">1618</a>
<code>\__draw_backend_moveto:nn</code> . . . . .	
. . . . .	<a href="#">1124</a> , <a href="#">1311</a> , <a href="#">1505</a>
<code>\__draw_backend_nonzero_rule:</code> . . . .	
. . . . .	<a href="#">1159</a> , <a href="#">1340</a> , <a href="#">1546</a>
<code>\__draw_backend_path:n</code> . . . . .	<a href="#">1550</a>
<code>\__draw_backend_rectangle:nnnn</code> . . . .	
. . . . .	<a href="#">1124</a> , <a href="#">1311</a> , <a href="#">1505</a>
<code>\__draw_backend_scope:n</code> <a href="#">1547</a> , <a href="#">1549</a> ,	
<a href="#">1569</a> , <a href="#">1609</a> , <a href="#">1631</a> , <a href="#">1643</a> , <a href="#">1645</a> , <a href="#">1647</a> ,	
<a href="#">1649</a> , <a href="#">1651</a> , <a href="#">1653</a> , <a href="#">1655</a> , <a href="#">1657</a> , <a href="#">1660</a>	
<code>\__draw_backend_scope_begin:</code> . . . .	
. . . . .	<a href="#">1120</a> , <a href="#">1306</a> , <a href="#">1309</a>
<code>\__draw_backend_scope_end:</code> . . . . .	
. . . . .	<a href="#">1120</a> , <a href="#">1308</a> , <a href="#">1309</a>
<code>\__draw_backend_stroke:</code> . . . . .	
. . . . .	<a href="#">1164</a> , <a href="#">1345</a> , <a href="#">1550</a>
<code>\g__draw_clip_path_int</code> . . . . .	
. . . . .	<a href="#">1556</a> , <a href="#">1559</a> , <a href="#">1572</a> , <a href="#">1601</a> , <a href="#">1604</a> , <a href="#">1612</a>
<code>\g__draw_draw_clip_bool</code> . . . . .	<a href="#">1164</a> , <a href="#">1550</a>
<code>\g__draw_draw_eor_bool</code> . . . . .	
. . . . .	<a href="#">1159</a> , <a href="#">1176</a> , <a href="#">1192</a> , <a href="#">1199</a> , <a href="#">1210</a> , <a href="#">1221</a> , <a href="#">1237</a> , <a href="#">1340</a> , <a href="#">1354</a> , <a href="#">1359</a> , <a href="#">1364</a>
<code>\g__draw_draw_path_int</code> . . . . .	<a href="#">1550</a>
<code>\g__draw_draw_path_tl</code> . . . . .	
. . . . .	<a href="#">1505</a> , <a href="#">1561</a> , <a href="#">1577</a> , <a href="#">1579</a> , <a href="#">1606</a> , <a href="#">1615</a>
<code>\g__draw_path_int</code> . . . . .	<a href="#">1565</a> , <a href="#">1582</a>
	<b>E</b>
<code>\errmessage</code> . . . . .	<a href="#">38</a>
<code>\evensidemargin</code> . . . . .	<a href="#">2204</a>
exp commands:	
<code>\exp_after:wN</code> . . . . .	<a href="#">152</a> , <a href="#">458</a> , <a href="#">1964</a>
<code>\exp_args:Ne</code> . . . . .	<a href="#">714</a> , <a href="#">2358</a> , <a href="#">2883</a>
<code>\exp_args:Nf</code> . . . . .	<a href="#">1249</a> , <a href="#">1373</a> , <a href="#">2126</a>
<code>\exp_args:NNf</code> . . . . .	<a href="#">229</a> , <a href="#">277</a> , <a href="#">334</a>
<code>\exp_args:Nnx</code> . . . . .	<a href="#">2113</a> , <a href="#">2796</a>
<code>\exp_args:NV</code> . . . . .	<a href="#">454</a>
<code>\exp_args:Nx</code> . . . . .	<a href="#">1782</a> , <a href="#">1803</a> , <a href="#">2070</a> , <a href="#">2085</a> , <a href="#">2200</a> , <a href="#">2762</a> , <a href="#">2969</a> , <a href="#">2999</a>
<code>\exp_last_unbraced:Nx</code> . . . . .	<a href="#">463</a> , <a href="#">477</a>
<code>\exp_not:N</code> . . . . .	<a href="#">515</a> , <a href="#">516</a> , <a href="#">524</a> , <a href="#">526</a> , <a href="#">672</a> , <a href="#">2444</a> , <a href="#">2446</a> , <a href="#">2449</a> , <a href="#">2479</a> , <a href="#">2481</a> , <a href="#">2484</a> , <a href="#">2636</a> , <a href="#">2638</a> , <a href="#">2641</a> , <a href="#">2647</a> , <a href="#">2649</a> , <a href="#">2652</a> , <a href="#">2689</a> , <a href="#">2690</a> , <a href="#">2696</a> , <a href="#">2697</a> , <a href="#">2716</a> , <a href="#">2721</a> , <a href="#">2830</a> , <a href="#">2838</a> , <a href="#">2854</a>
<code>\exp_not:n</code> . . . . .	<a href="#">48</a> , <a href="#">90</a> , <a href="#">101</a> , <a href="#">129</a> , <a href="#">2061</a> , <a href="#">2066</a> , <a href="#">2352</a> , <a href="#">2591</a> , <a href="#">2592</a> , <a href="#">2606</a> , <a href="#">2607</a> , <a href="#">2619</a> , <a href="#">2620</a> , <a href="#">2774</a> , <a href="#">2779</a> , <a href="#">2790</a> , <a href="#">2863</a>
<code>\ExplBackendFileDate</code> . . . . .	<a href="#">1</a>
	<b>F</b>
file commands:	
<code>\file_compare_timestamp:nNnTF</code> . . . . .	<a href="#">1791</a>
<code>\file_parse_full_name:nNNN</code> . . . . .	<a href="#">1778</a> , <a href="#">1801</a>
fp commands:	
<code>\fp_compare:nNnTF</code> . . . . .	<a href="#">236</a> , <a href="#">283</a> , <a href="#">289</a> , <a href="#">341</a> , <a href="#">1416</a> , <a href="#">1429</a> , <a href="#">1474</a>
<code>\fp_eval:n</code> . . . . .	<a href="#">229</a> , <a href="#">238</a> , <a href="#">251</a> , <a href="#">252</a> , <a href="#">277</a> , <a href="#">294</a> , <a href="#">309</a> , <a href="#">311</a> , <a href="#">334</a> , <a href="#">343</a> , <a href="#">354</a> , <a href="#">355</a> , <a href="#">419</a> , <a href="#">434</a> , <a href="#">435</a> , <a href="#">1056</a> , <a href="#">1057</a> , <a href="#">1058</a> , <a href="#">1066</a> , <a href="#">1079</a> , <a href="#">1080</a> , <a href="#">1081</a> , <a href="#">1418</a> , <a href="#">1423</a> , <a href="#">1424</a> , <a href="#">1431</a> , <a href="#">1441</a> , <a href="#">1442</a> , <a href="#">1443</a> , <a href="#">1444</a> , <a href="#">1453</a> , <a href="#">1454</a> , <a href="#">1455</a> , <a href="#">1456</a> , <a href="#">1465</a> , <a href="#">1466</a> , <a href="#">1467</a> , <a href="#">1468</a> , <a href="#">2349</a> , <a href="#">2518</a> , <a href="#">2877</a> , <a href="#">2970</a> , <a href="#">2978</a> , <a href="#">2980</a> , <a href="#">3000</a> , <a href="#">3022</a> , <a href="#">3029</a> , <a href="#">3090</a>
<code>\fp_new:N</code> . . . . .	<a href="#">302</a> , <a href="#">303</a>
<code>\fp_set:Nn</code> . . . . .	<a href="#">282</a> , <a href="#">285</a>
<code>\fp_use:N</code> . . . . .	<a href="#">288</a> , <a href="#">292</a> , <a href="#">297</a>
<code>\fp_zero:N</code> . . . . .	<a href="#">284</a>
<code>\c_zero_fp</code> <a href="#">236</a> , <a href="#">283</a> , <a href="#">289</a> , <a href="#">341</a> , <a href="#">1416</a> , <a href="#">1429</a>	
	<b>G</b>
graphics commands:	
<code>\graphics_bb_restore:nTF</code> . . . . .	<a href="#">1733</a> , <a href="#">1947</a>
<code>\graphics_bb_save:n</code> . . . . .	<a href="#">1762</a> , <a href="#">1955</a>
<code>\l_graphics_decodearray_tl</code> . . . . .	<a href="#">1710</a> , <a href="#">1711</a> , <a href="#">1721</a> , <a href="#">1741</a> , <a href="#">1745</a> , <a href="#">1746</a> , <a href="#">1823</a> , <a href="#">1855</a> , <a href="#">1856</a> , <a href="#">1894</a> , <a href="#">1897</a> , <a href="#">1898</a> , <a href="#">1916</a> , <a href="#">1980</a>
<code>\graphics_extract_bb:n</code> . . . . .	<a href="#">1818</a> , <a href="#">1825</a> , <a href="#">1975</a> , <a href="#">1982</a>

<code>\l_graphics_interpolate_bool</code> . . .	<code>\__graphics_backend_include:nn</code> <a href="#">1984</a>
. . . . . <a href="#">1712, 1722, 1740, 1747,</a>	<code>\__graphics_backend_include_-</code>
<a href="#">1824, 1857, 1893, 1899, 1917, 1981</a>	auxi:nn . . . . . <a href="#">1829</a>
<code>\l_graphics_llx_dim</code> . . . . .	<code>\__graphics_backend_include_-</code>
. . . . . <a href="#">1695, 1834, 1886, 1993</a>	auxii:nnn . . . . . <a href="#">1829</a>
<code>\l_graphics_lly_dim</code> . . . . .	<code>\__graphics_backend_include_-</code>
. . . . . <a href="#">1696, 1835, 1887, 1994</a>	auxiii:nnn . . . . . <a href="#">1829</a>
<code>\l_graphics_name_tl</code> . . . . . <a href="#">1796</a>	<code>\__graphics_backend_include_-</code>
<code>\l_graphics_page_int</code> . . . . .	bitmap_quote:w . . . . . <a href="#">1958, 1999</a>
. . . . . <a href="#">1706, 1726, 1727, 1751,</a>	<code>\__graphics_backend_include_-</code>
<a href="#">1752, 1816, 1853, 1854, 1880, 1881,</a>	eps:n . . . . . <a href="#">1690, 1771, 1829, 1984</a>
<a href="#">1909, 1922, 1923, 1962, 1963, 1973</a>	<code>\__graphics_backend_include_-</code>
<code>\l_graphics_pagebox_tl</code> . . . . .	jpg:n . . . . . <a href="#">1764, 1829, 1999</a>
. . . . . <a href="#">51, 1707, 1725,</a>	<code>\__graphics_backend_include_-</code>
<a href="#">1753, 1754, 1817, 1851, 1852, 1882,</a>	pdf:n . . . . . <a href="#">1764, 1803, 1829, 1958, 1984</a>
<a href="#">1884, 1910, 1931, 1932, 1964, 1974</a>	<code>\__graphics_backend_include_pdf_-</code>
<code>\graphics_read_bb:n</code> . . . . . <a href="#">1689, 1812, 1970</a>	quote:w . . . . . <a href="#">1961, 1966</a>
<code>\l_graphics_urx_dim</code> . . . . .	<code>\__graphics_backend_include_-</code>
. . . . . <a href="#">1697, 1758, 1836, 1888, 1953, 1995</a>	png:n . . . . . <a href="#">1764, 1829, 1999</a>
<code>\l_graphics_ury_dim</code> . . . . . <a href="#">1698, 1759,</a>	<code>\l__graphics_backend_name_str</code> . . . . . <a href="#">1771</a>
<a href="#">1837, 1889, 1954, 1996, 2004, 2005</a>	<code>\l__graphics_graphics_attr_tl</code> . . . . .
graphics internal commands:	. . . . . <a href="#">1703, 1708,</a>
<code>\l__graphics_backend_dir_str</code> . . . . . <a href="#">1771</a>	<a href="#">1715, 1723, 1733, 1760, 1762, 1767</a>
<code>\l__graphics_backend_ext_str</code> . . . . . <a href="#">1771</a>	<code>\l__graphics_internal_box</code> . . . . .
<code>\__graphics_backend_getbb_auxi:n</code>	. . . . . <a href="#">1756, 1758, 1759, 1952, 1953, 1954</a>
. . . . . <a href="#">1704</a>	<code>\g__graphics_track_int</code> . . . . .
<code>\__graphics_backend_getbb_-</code>	. . . . . <a href="#">1828, 1874, 1875</a>
auxi:nN . . . . . <a href="#">1907</a>	group commands:
<code>\__graphics_backend_getbb_-</code>	<code>\group_begin:</code> . . . . . <a href="#">144, 172, 191</a>
auxii:n . . . . . <a href="#">1704</a>	<code>\group_end:</code> . . . . . <a href="#">157, 180</a>
<code>\__graphics_backend_getbb_-</code>	<code>\group_insert_after:N</code> <a href="#">624, 643, 654,</a>
auxii:nnN . . . . . <a href="#">1907</a>	<a href="#">966, 979, 994, 1021, 1046, 3011, 3046</a>
<code>\__graphics_backend_getbb_-</code>	
auxiii:nNnn . . . . . <a href="#">1907</a>	<b>H</b>
<code>\__graphics_backend_getbb_-</code>	hbox commands:
auxiv:nnNnn . . . . . <a href="#">1907</a>	<code>\hbox:n</code> . . . . . <a href="#">2132, 2135,</a>
<code>\__graphics_backend_getbb_-</code>	<a href="#">2207, 2213, 2366, 2373, 2891, 2902</a>
auxv:nNnn . . . . . <a href="#">1907</a>	<code>\hbox_overlap_right:n</code> . . . . . <a href="#">224,</a>
<code>\__graphics_backend_getbb_-</code>	<a href="#">256, 272, 313, 329, 357, 441, 1294, 1489</a>
auxvi:nNnn . . . . . <a href="#">1948, 1950</a>	<code>\hbox_set:Nn</code> . . . . . <a href="#">1756, 1952, 2199, 2231</a>
<code>\__graphics_backend_getbb_eps:n</code> . . . . .	<code>\hbox_set:Nw</code> . . . . . <a href="#">2182</a>
. . . . . <a href="#">1689, 1771, 1812, 1970</a>	<code>\hbox_set_end:</code> . . . . . <a href="#">2197</a>
<code>\__graphics_backend_getbb_eps:nm</code>	<code>\hbox_unpack:N</code> . . . . . <a href="#">2318</a>
. . . . . <a href="#">1771</a>	
<code>\__graphics_backend_getbb_eps:nn</code>	<b>I</b>
. . . . . <a href="#">1782, 1789</a>	int commands:
<code>\__graphics_backend_getbb_jpg:n</code> . . . . .	<code>\int_compare:nNnTF</code> . . . . .
. . . . . <a href="#">1704, 1812, 1907, 1971</a>	. . . . . <a href="#">509, 551, 649, 947, 989, 1726, 1751,</a>
<code>\__graphics_backend_getbb_-</code>	<a href="#">1853, 1880, 1922, 1962, 2290, 2391,</a>
pagebox:w . . . . . <a href="#">1907, 1964</a>	<a href="#">2687, 2715, 2828, 2835, 2851, 3052</a>
<code>\__graphics_backend_getbb_pdf:n</code> . . . . .	<code>\int_const:Nn</code> . . . . . <a href="#">150, 156, 516,</a>
. . . . . <a href="#">1704, 1797, 1812, 1907, 1978</a>	<a href="#">542, 577, 1760, 1875, 2030, 2565, 2753</a>
<code>\__graphics_backend_getbb_png:n</code> . . . . .	<code>\int_eval:n</code> . . . . .
. . . . . <a href="#">1704, 1812, 1907, 1971</a>	. . . . . <a href="#">558, 568, 597, 608, 710, 719, 732,</a>

734, 738, 751, 2415, 2419, 2665,  
2690, 2697, 2710, 2920, 2928, 2933  
`\int_gincr:N` ..... 198, 364,  
515, 1556, 1601, 1874, 2029, 2098,  
2142, 2216, 2752, 2795, 2808, 2830  
`\int_gset:Nn` ..... 173, 192, 2279  
`\int_gset_eq:NN` 181, 2143, 2217, 2809  
`\int_if_exist:NTF` ..... 1864  
`\int_if_odd:nTF` ..... 2202  
`\int_new:N` ..... 164, 165,  
411, 506, 512, 1582, 1828, 2025,  
2123, 2154, 2156, 2748, 2805, 2821  
`\int_set:Nn` ..... 534  
`\int_set_eq:NN` ... 169, 188, 540, 2291  
`\int_step_function:nnnN` ..... 736  
`\int_use:N` ..... 366,  
397, 524, 535, 684, 820, 865, 933,  
1559, 1565, 1572, 1604, 1612, 1727,  
1752, 1767, 1854, 1867, 1879, 1881,  
1963, 2036, 2101, 2114, 2118, 2146,  
2153, 2221, 2322, 2576, 2586, 2759,  
2797, 2802, 2812, 2820, 2838, 2854  
`\int_value:w` .....  
..... 2444, 2479, 2636, 2647, 2665  
`\int_zero:N` ... 1706, 1816, 1909, 1973

## K

kernel internal commands:

`\__kernel_backend_align_begin:` ..  
..... 65, 209, 233, 248  
`\__kernel_backend_align_end:` ...  
..... 65, 223, 241, 255  
`\__kernel_backend_first_shipout:n`  
..... 50, 62, 519, 670  
`\g__kernel_backend_header_bool` ..  
..... 60, 668  
`\__kernel_backend_literal:n` ....  
..... 46, 55, 58, 63,  
67, 74, 77, 79, 135, 138, 140, 142,  
162, 338, 351, 521, 546, 547, 555,  
565, 620, 627, 653, 659, 680, 816,  
993, 999, 1001, 1020, 1045, 1112,  
1118, 1413, 1420, 1426, 1486, 1491,  
1692, 1831, 1866, 1876, 1990, 2001,  
2742, 2858, 2920, 2924, 2929, 2934  
`\__kernel_backend_literal_page:n`  
..... 93, 137, 2736, 2738, 2939, 2941  
`\__kernel_backend_literal_pdf:n` .  
.. 82, 134, 264, 321, 1303, 3061, 3076  
`\__kernel_backend_literal_-  
postscript:n` .....  
..... 54, 68, 69, 73, 210, 211, 213,  
214, 222, 234, 249, 1108, 2393, 2405

`\__kernel_backend_literal_svg:n` .  
..... 161, 168, 179, 187,  
197, 365, 367, 384, 1497, 1670, 1681  
`\__kernel_backend_matrix:n` .....  
..... 121, 286, 307, 1403  
`\__kernel_backend_postscript:n` ..  
..... 57, 622,  
1024, 1026, 1028, 1032, 2019, 2075,  
2090, 2132, 2138, 2175, 2207, 2214,  
2218, 2232, 2260, 2304, 2311, 2317,  
2325, 2332, 2366, 2373, 2974, 2983  
`\__kernel_backend_scope:n` .....  
..... 166, 394, 399, 1086, 1502, 3090  
`\__kernel_backend_scope_begin:` ..  
..... 76, 103, 139,  
166, 208, 232, 247, 263, 280, 306,  
320, 337, 350, 1309, 1481, 1501, 1668  
`\__kernel_backend_scope_begin:n` .  
..... 166, 386, 414, 427  
`\__kernel_backend_scope_end:` ...  
.. 76, 103, 139, 166, 225, 243, 257,  
273, 300, 314, 330, 346, 358, 409,  
423, 442, 544, 1310, 1493, 1504, 1682  
`\g__kernel_backend_scope_int` ...  
..... 164, 171, 173, 178, 182, 190, 192, 198  
`\l__kernel_backend_scope_int` ...  
..... 164, 170, 183, 189  
`\__kernel_color_backend_stack_-  
init:Nnn` ..... 509, 575, 2990  
`\__kernel_color_backend_stack_-  
pop:n` ..... 551, 589, 646, 3018  
`\__kernel_color_backend_stack_-  
push:nn` .....  
.. 551, 589, 642, 964, 977, 3009, 3044  
`\__kernel_dependency_version_-  
check:Nn` ..... 1  
`\__kernel_dependency_version_-  
check:nn` ..... 27, 29  
`\__kernel_kern:n` .....  
..... 2137, 2139, 2365, 2369,  
2372, 2376, 2890, 2898, 2901, 2917  
`\c__kernel_sys_dvipdfmx_version_-  
int` ..... 144, 509, 551,  
649, 947, 989, 2828, 2835, 2851, 3052

## M

`\MessageBreak` ..... 40  
mode commands:  
`\mode_if_horizontal:TF` ... 2281, 2288  
`\mode_if_math:TF` ..... 2179

## O

`\oddsidemargin` ..... 2203  
opacity internal commands:  
`\__opacity_backend:nn` ... 2977, 3083

\\_opacity\_backend\_fill:n .....  
     ..... [2977](#), [3019](#), [3083](#)  
 \\_opacity\_backend\_fill\_stroke:nn  
     ..... [3021](#), [3027](#), [3031](#), [3049](#), [3063](#)  
 \l\_opacity\_backend\_fill\_tl ....  
     .. [2995](#), [3004](#), [3028](#), [3036](#), [3056](#), [3068](#)  
 \\_opacity\_backend\_fillstroke:nn  
     ..... [3019](#)  
 \\_opacity\_backend\_reset: [2997](#), [3046](#)  
 \\_opacity\_backend\_select:n ....  
     ..... [2967](#), [2997](#), [3052](#), [3083](#)  
 \\_opacity\_backend\_select\_aux:n .  
     ..... [2967](#), [2997](#), [3034](#), [3054](#), [3066](#)  
 \c\_opacity\_backend\_stack\_int ...  
     ..... [2988](#), [3009](#), [3018](#), [3044](#)  
 \\_opacity\_backend\_stroke:n ....  
     ..... [2977](#), [3019](#), [3083](#)  
 \l\_opacity\_backend\_stroke\_tl ...  
     .. [2995](#), [3005](#), [3023](#), [3037](#), [3057](#), [3069](#)

## P

pdf commands:

\pdf\_object\_if\_exist:nTF ..... [876](#)  
 \pdf\_object\_new:nn ..... [878](#)  
 \pdf\_object\_ref:n ..... [891](#)  
 \pdf\_object\_ref\_last: .....  
     ..... [857](#), [866](#), [925](#), [934](#)  
 \pdf\_object\_unnamed\_write:nn ...  
     ..... [845](#), [872](#), [898](#)  
 \pdf\_object\_write:nn ..... [879](#)

pdf internal commands:

\\_pdf\_backend:n ..... [2741](#),  
     [2745](#), [2747](#), [2773](#), [2778](#), [2787](#), [2810](#),  
     [2832](#), [2848](#), [2861](#), [2893](#), [2894](#), [2904](#)  
 \\_pdf\_backend\_annotation:nmmn ..  
     ..... [2124](#), [2429](#), [2806](#)  
 \\_pdf\_backend\_annotation\_  
     aux:nmmn ..... [2126](#), [2129](#)  
 \g\_pdf\_backend\_annotation\_int ..  
     .. [2123](#), [2143](#), [2153](#), [2805](#), [2809](#), [2820](#)  
 \\_pdf\_backend\_annotation\_last: .  
     ..... [2152](#), [2442](#), [2819](#)  
 \\_pdf\_backend\_bdc:nn .....  
     ..... [2423](#), [2735](#), [2938](#), [2960](#)  
 \\_pdf\_backend\_catalog\_gput:nn ..  
     ..... [2021](#), [2535](#), [2744](#), [2944](#)  
 \\_pdf\_backend\_compress\_objects:n  
     ..... [2389](#), [2656](#), [2919](#), [2954](#)  
 \\_pdf\_backend\_compresslevel:n ..  
     ..... [2389](#), [2656](#), [2919](#), [2954](#)  
 \l\_pdf\_backend\_content\_box [2121](#),  
     [2182](#), [2206](#), [2209](#), [2211](#), [2240](#), [2251](#)  
 \\_pdf\_backend\_destination:nn ...  
     ..... [2330](#), [2498](#), [2859](#)

\\_pdf\_backend\_destination:nmmn .  
     ..... [2330](#), [2498](#), [2859](#)  
 \\_pdf\_backend\_destination\_  
     aux:nmmn ..... [2330](#), [2859](#)  
 \\_pdf\_backend\_emc: .....  
     ..... [2423](#), [2735](#), [2938](#), [2960](#)  
 \\_pdf\_backend\_info\_gput:nn ....  
     ..... [2021](#), [2535](#), [2744](#), [2944](#)  
 \\_pdf\_backend\_link:nw ..... [2163](#)  
 \\_pdf\_backend\_link\_aux:nw ... [2163](#)  
 \\_pdf\_backend\_link\_begin:n .. [2822](#)  
 \\_pdf\_backend\_link\_begin:nmmw [2453](#)  
 \\_pdf\_backend\_link\_begin:nw ...  
     ..... [2164](#), [2166](#), [2167](#)  
 \\_pdf\_backend\_link\_begin\_aux:nw  
     ..... [2170](#), [2172](#)  
 \\_pdf\_backend\_link\_begin\_  
     goto:nmw ..... [2163](#), [2453](#), [2822](#)  
 \\_pdf\_backend\_link\_begin\_  
     user:nmw ..... [2163](#), [2453](#), [2822](#)  
 \g\_pdf\_backend\_link\_bool .....  
     ..... [2158](#), [2169](#), [2174](#), [2189](#), [2227](#)  
 \g\_pdf\_backend\_link\_dict\_tl ...  
     ..... [2155](#), [2177](#), [2222](#)  
 \\_pdf\_backend\_link\_end: .....  
     ..... [2163](#), [2453](#), [2822](#)  
 \\_pdf\_backend\_link\_end\_aux: . [2163](#)  
 \g\_pdf\_backend\_link\_int .....  
     ..... [2154](#), [2217](#),  
     [2221](#), [2322](#), [2821](#), [2830](#), [2838](#), [2854](#)  
 \\_pdf\_backend\_link\_last: .....  
     ..... [2321](#), [2477](#), [2849](#)  
 \\_pdf\_backend\_link\_margin:n ...  
     ..... [2323](#), [2488](#), [2857](#)  
 \g\_pdf\_backend\_link\_math\_bool ..  
     ..... [2157](#), [2180](#), [2181](#), [2184](#), [2194](#)  
 \\_pdf\_backend\_link\_minima: .. [2163](#)  
 \\_pdf\_backend\_link\_outerbox:n [2163](#)  
 \g\_pdf\_backend\_link\_sf\_int ....  
     ..... [2156](#), [2279](#), [2290](#), [2291](#)  
 \\_pdf\_backend\_link\_sf\_restore: [2163](#)  
 \\_pdf\_backend\_link\_sf\_save: . [2163](#)  
 \l\_pdf\_backend\_model\_box . [2122](#),  
     [2199](#), [2231](#), [2239](#), [2250](#), [2265](#), [2267](#)  
 \\_pdf\_backend\_objcompresslevel:n  
     ..... [2656](#)  
 \g\_pdf\_backend\_object\_int .....  
     ..... [2025](#), [2029](#), [2032](#),  
     [2098](#), [2101](#), [2114](#), [2118](#), [2142](#), [2143](#),  
     [2146](#), [2216](#), [2217](#), [2748](#), [2752](#), [2755](#),  
     [2795](#), [2797](#), [2802](#), [2808](#), [2809](#), [2812](#)  
 \\_pdf\_backend\_object\_last: .....  
     ..... [2117](#), [2634](#), [2801](#), [2946](#)



\_pdf_backend_object_new:nn . . .	pdf.bordertracking.end . . . . .	<a href="#">3179</a>
. . . . . <a href="#">2027</a> , <a href="#">2556</a> , <a href="#">2750</a> , <a href="#">2946</a>	pdf.bordertracking.endpage . . . . .	<a href="#">3179</a>
\_pdf_backend_object_now:nn . . .	pdf.breaklink . . . . .	<a href="#">3317</a>
. . . . . <a href="#">2096</a> , <a href="#">2608</a> , <a href="#">2793</a> , <a href="#">2946</a>	pdf.breaklink.write . . . . .	<a href="#">3317</a>
\g_pdf_backend_object_prop . . . .	pdf.brokenlink.dict . . . . .	<a href="#">3179</a>
. . . . . <a href="#">2025</a> , <a href="#">2033</a> , <a href="#">2044</a> , <a href="#">2054</a> ,	pdf.brokenlink.rect . . . . .	<a href="#">3179</a>
<a href="#">2555</a> , <a href="#">2573</a> , <a href="#">2589</a> , <a href="#">2748</a> , <a href="#">2756</a> , <a href="#">2763</a>	pdf.brokenlink.skip . . . . .	<a href="#">3179</a>
\_pdf_backend_object_ref:n <a href="#">2027</a> ,	pdf.count . . . . .	<a href="#">3317</a>
<a href="#">2041</a> , <a href="#">2055</a> , <a href="#">2556</a> , <a href="#">2750</a> , <a href="#">2769</a> , <a href="#">2946</a>	pdf.currentrect . . . . .	<a href="#">3317</a>
\_pdf_backend_object_write:nn . .	pdf.cvs . . . . .	<a href="#">3101</a>
. . . . . <a href="#">2037</a> , <a href="#">2577</a> , <a href="#">2760</a> , <a href="#">2946</a>	pdf.dest.anchor . . . . .	<a href="#">3144</a>
\_pdf_backend_object_write:nnn <a href="#">2760</a>	pdf.dest.point . . . . .	<a href="#">3144</a>
\_pdf_backend_object_write_-	pdf.dest.x . . . . .	<a href="#">3144</a>
array:nn . . . . . <a href="#">2037</a> , <a href="#">2760</a>	pdf.dest.y . . . . .	<a href="#">3144</a>
\_pdf_backend_object_write_-	pdf.dest2device . . . . .	<a href="#">3144</a>
dict:nn . . . . . <a href="#">2037</a> , <a href="#">2760</a>	pdf.dev.x . . . . .	<a href="#">3144</a>
\_pdf_backend_object_write_-	pdf.dev.y . . . . .	<a href="#">3144</a>
fstream:nn . . . . . <a href="#">2037</a> , <a href="#">2760</a>	pdf.dvi.pt . . . . .	<a href="#">3101</a>
\_pdf_backend_object_write_-	pdf.globaldict . . . . .	<a href="#">3098</a>
fstream:nnn . . . . . <a href="#">2071</a> , <a href="#">2073</a>	pdf.leftboundary . . . . .	<a href="#">3179</a>
\_pdf_backend_object_write_-	pdf.link.dict . . . . .	<a href="#">2163</a>
stream:nn . . . . . <a href="#">2037</a> , <a href="#">2760</a>	pdf.linkdp.pad . . . . .	<a href="#">2163</a> , <a href="#">3105</a>
\_pdf_backend_object_write_-	pdf.linkht.pad . . . . .	<a href="#">2163</a> , <a href="#">3105</a>
stream:nnn . . . . . <a href="#">2037</a>	pdf.linkmargin . . . . .	<a href="#">3105</a>
\_pdf_backend_object_write_-	pdf.llx . . . . .	<a href="#">2163</a> , <a href="#">3108</a>
stream:nnnn . . . . . <a href="#">2760</a>	pdf.lly . . . . .	<a href="#">2163</a> , <a href="#">3108</a>
\_pdf_backend_pageobject_ref:n . .	pdf.originx . . . . .	<a href="#">3179</a>
. . . . . <a href="#">2119</a> , <a href="#">2645</a> , <a href="#">2803</a> , <a href="#">2946</a>	pdf.originy . . . . .	<a href="#">3179</a>
\_pdf_backend_pdfmark:n . . . . .	pdf.outerbox . . . . .	<a href="#">2163</a> , <a href="#">3421</a>
<a href="#">2018</a> , <a href="#">2022</a> , <a href="#">2024</a> , <a href="#">2039</a> , <a href="#">2060</a> , <a href="#">2065</a> ,	pdf.pdfmark . . . . .	<a href="#">3421</a>
<a href="#">2099</a> , <a href="#">2144</a> , <a href="#">2333</a> , <a href="#">2377</a> , <a href="#">2424</a> , <a href="#">2426</a>	pdf.pdfmark.dict . . . . .	<a href="#">3421</a>
\_pdf_backend_version_major: . . .	pdf.pdfmark.good . . . . .	<a href="#">3421</a>
. . . . . <a href="#">2415</a> ,	pdf.pt.dvi . . . . .	<a href="#">3101</a>
<a href="#">2421</a> , <a href="#">2712</a> , <a href="#">2928</a> , <a href="#">2929</a> , <a href="#">2936</a> , <a href="#">2958</a>	pdf.rect . . . . .	<a href="#">3108</a>
\_pdf_backend_version_major_-	pdf.rect.ht . . . . .	<a href="#">3101</a>
gset:n . . . . . <a href="#">2413</a> , <a href="#">2684</a> , <a href="#">2926</a> , <a href="#">2956</a>	pdf.rightboundary . . . . .	<a href="#">3179</a>
\_pdf_backend_version_minor: . . .	pdf.save.linkll . . . . .	<a href="#">3108</a>
. . . . . <a href="#">2419</a> ,	pdf.save.linkur . . . . .	<a href="#">3108</a>
<a href="#">2421</a> , <a href="#">2712</a> , <a href="#">2933</a> , <a href="#">2934</a> , <a href="#">2936</a> , <a href="#">2958</a>	pdf.save.ll . . . . .	<a href="#">3108</a>
\_pdf_backend_version_minor_-	pdf.save.ur . . . . .	<a href="#">3108</a>
gset:n . . . . . <a href="#">2413</a> , <a href="#">2684</a> , <a href="#">2926</a> , <a href="#">2956</a>	pdf.tmpa . . . . .	<a href="#">3144</a>
\l_pdf_breaklink_pdfmark_tl . . .	pdf.tmpb . . . . .	<a href="#">3144</a>
. . . . . <a href="#">2159</a> , <a href="#">2224</a> , <a href="#">2316</a>	pdf.tmpc . . . . .	<a href="#">3144</a>
\_pdf_breaklink_postscript:n . . .	pdf.tmpd . . . . .	<a href="#">3144</a>
. . . . . <a href="#">2161</a> , <a href="#">2208</a> , <a href="#">2210</a> , <a href="#">2317</a>	pdf.urx . . . . .	<a href="#">3108</a>
\_pdf_breaklink_usebox:N . . . . .	pdf.ury . . . . .	<a href="#">2163</a> , <a href="#">3108</a>
. . . . . <a href="#">2162</a> , <a href="#">2209</a> , <a href="#">2318</a>	pdfmanagement commands:	
\_pdf_exp_not_i:nn . <a href="#">2577</a> , <a href="#">2623</a> , <a href="#">2628</a>	\pdfmanagement_add:nnn . . . . .	<a href="#">859</a> ,
\_pdf_exp_not_ii:nn <a href="#">2577</a> , <a href="#">2624</a> , <a href="#">2629</a>	. . . . . <a href="#">863</a> , <a href="#">927</a> , <a href="#">931</a> , <a href="#">2988</a> , <a href="#">2992</a> , <a href="#">3006</a> ,	
\l_pdf_internal_box . . . . . <a href="#">2016</a>	. . . . . <a href="#">3013</a> , <a href="#">3038</a> , <a href="#">3041</a> , <a href="#">3058</a> , <a href="#">3070</a> , <a href="#">3073</a>	
pdf.baselineskip . . . . . <a href="#">2163</a> , <a href="#">3421</a>	prg commands:	
pdf.bordertracking . . . . . <a href="#">3179</a>	\prg_replicate:nn . . . . .	
pdf.bordertracking.begin . . . . . <a href="#">3179</a>	. . . . . <a href="#">177</a> , <a href="#">708</a> , <a href="#">729</a> , <a href="#">739</a> , <a href="#">904</a>	
pdf.bordertracking.continue . . . . . <a href="#">3179</a>		

prop commands:		\tex_global:D .....	.....
\prop_gput:Nnn	..... 2033, 2573, 2756	.....	2658, 2675, 2689, 2696, 2703
\prop_item:Nn	.. 2044, 2054, 2589, 2763	\tex_immediate:D .....	.....
\prop_new:N	..... 2026, 2555, 2749	.....	1738, 2580, 2583, 2611, 2614
\ProvidesExplFile	..... 2	\tex_luatexversion:D	.... 2687, 2715
<b>Q</b>		\tex_pdfannot:D	..... 2435
quark commands:		\tex_pdfcatalog:D	..... 2541
\q_stop	..... 145, 153	\tex_pdfcolorstack:D	..... 595, 606
<b>S</b>		\tex_pdfcolorstackinit:D	..... 583
scan commands:		\tex_pdfcompresslevel:D	..... 2663
\scan_stop:	.....	\tex_pdfdest:D	..... 2504, 2527
..	106, 115, 608, 2471, 2496, 2519,	\tex_pdfendlink:D	..... 2474
	2533, 2665, 2682, 2690, 2697, 2710	\tex_pdfextension:D	.....
scan internal commands:		.....	85, 96, 106, 115, 124,
\s_color_stop	.....	.....	592, 603, 2432, 2460, 2471, 2501,
....	464, 467, 478, 481, 719, 720,	.....	2524, 2538, 2548, 2559, 2580, 2611
	724, 728, 741, 744, 748, 752, 766,	\tex_pdffeedback:D	.....
	905, 938, 942, 1049, 1051, 1072, 1074	....	580, 2446, 2481, 2568, 2638, 2649
\s_graphics_stop	.....	\tex_pdfinfo:D	..... 2551
.....	1961, 1966, 2006, 2010	\tex_pdflastannot:D	..... 2449
separation	..... 3095	\tex_pdflastlink:D	..... 2484
skip commands:		\tex_pdflastobj:D	..... 2571, 2641
\skip_horizontal:n	.... 226, 274, 331	\tex_pdflastximage:D	.... 1757, 1761
str commands:		\tex_pdflinkmargin:D	..... 2494
\c_hash_str	.... 397, 1565, 1572, 1612	\tex_pdfliteral:D	..... 88, 99
\c_percent_str	..... 1092, 1093, 1094	\tex_pdfmajorversion:D	.....
\str_case:nn	..... 910, 2103, 2616	.....	2694, 2696, 2720, 2721
\str_case:nnTF	..... 2337, 2507, 2866	\tex_pdfminorversion:D	... 2708, 2732
\str_case:e:nn	..... 2043, 2588	\tex_pdfobj:D	..... 2562, 2583, 2614
\str_convert_pdfname:n	..... 687, 856	\tex_pdfobjcompresslevel:D	... 2680
\str_if_eq:nnTF	.....	\tex_pdfpageref:D	..... 2652
.....	483, 486, 489, 492, 3033, 3065	\tex_pdfrefximage:D	.... 1757, 1766
\str_new:N	..... 1773, 1774, 1775	\tex_pdfrestore:D	..... 118
\str_tail:N	..... 1784, 1805	\tex_pdfsave:D	..... 109
sys commands:		\tex_pdfsetmatrix:D	..... 127
\sys_get_shell:nnNTF	..... 146	\tex_pdfstartlink:D	..... 2463
\sys_if_shell:TF	..... 1771	\tex_pdfvariable:D	..... 2491,
\sys_shell_now:n	..... 1793	.....	2660, 2677, 2689, 2705, 2716, 2729
sys internal commands:		\tex_pdfximage:D	..... 1738
\l_sys_internal_tl	..... 148, 152	\tex_spacefactor:D	..... 2282, 2291
\_sys_tmp:w	..... 145, 152	\tex_special:D	..... 46
<b>T</b>		\tex_the:D	.... 1761, 2716, 2721, 2727
TeX and L <sup>A</sup> T <sub>E</sub> X 2 <sub>ε</sub> commands:		\tex_vss:D	.... 2367, 2374, 2896, 2915
\@cclv	..... 2300, 2302, 2310	\tex_XeTeXpdffile:D	.... 1918, 1960
\@ifl@t@r	..... 50	\tex_XeTeXpicfile:D	..... 1911
\@makecol@hook	..... 2294	TeXcolorseparation	..... 3095
\current@color	.. 14, 454, 458, 464, 478	\textwidth	..... 2266
\special	..... 2	tl commands:	
tex commands:		\c_space_tl	.... 288, 293, 296, 525,
\tex_baselineskip:D	..... 2271	.....	775, 978, 1541, 1694, 1695, 1696,
\tex_endinput:D	..... 44	.....	1697, 1833, 1834, 1835, 1836, 1881,
		.....	1884, 1886, 1887, 1888, 1889, 1961,
		.....	1963, 1992, 1993, 1994, 1995, 2222,
		.....	2451, 2486, 2643, 2654, 2812, 2839

<code>\tl_clear:N</code> . . . . .	1707, 1715, 1721,	2036, 2566, 2576, 2587, 2754, 2759
	1817, 1823, 1910, 1916, 1974, 1980	
<code>\tl_gclear:N</code> . . . . .	1579, 1615	<code>\tl_use:N</code> . . . . . 807, 884
<code>\tl_gset:Nn</code> . . . . .	1538, 2177	token commands:
<code>\tl_if_blank:nTF</code> . . . . .		<code>\c_math_toggle_token</code> . . . . 2185, 2195
	526, 585, 723, 740, 747, 765, 849, 941	
<code>\tl_if_empty:NTF</code> . . . . .	1541, 1710, 1745,	<b>U</b>
	1753, 1851, 1855, 1882, 1897, 1931	use commands:
<code>\tl_if_empty:nTF</code> . . . . .	1635	<code>\use:N</code> . . . . . 43, 2053, 2113, 2768, 2796
<code>\tl_if_empty_p:N</code> . . . . .	1741, 1894	<code>\use:n</code> . . . . . 52, 458, 494, 517,
<code>\tl_if_head_is_space:nTF</code> . . . . .	454	861, 929, 1053, 1063, 1076, 1249,
<code>\tl_new:N</code> . . . . .	630,	1373, 1438, 1450, 1462, 1620, 1938
	631, 1545, 1703, 2155, 2159, 2995, 2996	<code>\use_none:n</code> . . . . . 1635, 1637, 2294
<code>\tl_put_right:Nn</code> . . . . .	2298	<b>V</b>
<code>\tl_set:Nn</code> . . . . .	456, 468, 484, 487, 490,	<code>\value</code> . . . . . 2202
	494, 497, 640, 641, 963, 976, 1708,	vbox commands:
	1723, 1796, 2160, 2316, 3004, 3005,	<code>\vbox_set:Nn</code> . . . . . 2302
	3036, 3037, 3056, 3057, 3068, 3069	<code>\vbox_to_zero:n</code> 2363, 2370, 2888, 2899
<code>\tl_to_str:n</code> . . . . .	2031,	<code>\vbox_unpack_drop:N</code> . . . . . 2310