

# File I

## Implementation

### 1 l3backend-basics Implementation

```
1 <*package>
```

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

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

```
2 \ProvidesExplFile
3 <*dvipdfmx>
4   {l3backend-dvipdfmx.def}{2022-08-05}{ }
5   {L3 backend support: dvipdfmx}
6 </dvipdfmx>
7 <*dvips>
8   {l3backend-dvips.def}{2022-08-05}{ }
9   {L3 backend support: dvips}
10 </dvips>
11 <*dvisvgm>
12   {l3backend-dvisvgm.def}{2022-08-05}{ }
13   {L3 backend support: dvisvgm}
14 </dvisvgm>
15 <*luatex>
16   {l3backend-luatex.def}{2022-08-05}{ }
17   {L3 backend support: PDF output (LuaTeX)}
18 </luatex>
19 <*pdftex>
20   {l3backend-pdftex.def}{2022-08-05}{ }
21   {L3 backend support: PDF output (pdfTeX)}
22 </pdftex>
23 <*xetex>
24   {l3backend-xetex.def}{2022-08-05}{ }
25   {L3 backend support: XeTeX}
26 </xetex>
```

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

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

```

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

```

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

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

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

```

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

```

(End definition for `__kernel_backend_literal:e`.)

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

```

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

```

(End definition for `__kernel_backend_first_shipout:n`.)

## 1.1 dvips backend

```

60 <*dvips>

```

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

```

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

```

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

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

```

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

```

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

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

```

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

```

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

```

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

```

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

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

```

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

```

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

```

87 </dvips>

```

## 1.2 LuaTeX and pdfTeX backends

88 `<*luatex | pdftex>`

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

\\_kernel\_backend\_literal\_pdf:n  
\\_kernel\_backend\_literal\_pdf:x

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

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

(End definition for \\_kernel\_backend\_literal\_pdf:n.)

\\_kernel\_backend\_literal\_page:n

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

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

(End definition for \\_kernel\_backend\_literal\_page:n.)

\\_kernel\_backend\_scope\_begin:

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

\\_kernel\_backend\_scope\_end:

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

```

126 </pdfTex>
127 }

```

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

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

```

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

```

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

```

139 </luatex | pdfTex>

```

### 1.3 dvipdfmx backend

```

140 <*dvipdfmx | xetex>

```

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

```

\_kernel_backend_literal_pdf:n
\_kernel_backend_literal_pdf:x

```

```

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

```

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

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

```

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

```

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

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

```

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

```

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

```

150 </dvipdfmx | xetex>

```

## 1.4 dvisvgm backend

151  $\langle *dvisvgm \rangle$

\\_kernel\_backend\_literal\_svg:n  
\\_kernel\_backend\_literal\_svg:x

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

152 \cs\_new\_protected:Npn \\_kernel\_backend\_literal\_svg:n #1  
153 { \\_kernel\_backend\_literal:n { dvisvgm:raw~ #1 { ?nl } } }  
154 \cs\_generate\_variant:Nn \\_kernel\_backend\_literal\_svg:n { x }

(End definition for \\_kernel\_backend\_literal\_svg:n.)

\g\_kernel\_backend\_scope\_int  
\l\_kernel\_backend\_scope\_int

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

155 \int\_new:N \g\_kernel\_backend\_scope\_int  
156 \int\_new:N \l\_kernel\_backend\_scope\_int

(End definition for \g\_kernel\_backend\_scope\_int and \l\_kernel\_backend\_scope\_int.)

\\_kernel\_backend\_scope\_begin:  
\\_kernel\_backend\_scope\_end:  
\\_kernel\_backend\_scope\_begin:n  
\\_kernel\_backend\_scope\_begin:x  
\\_kernel\_backend\_scope:n  
\\_kernel\_backend\_scope:x

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

157 \cs\_new\_protected:Npn \\_kernel\_backend\_scope\_begin:  
158 {  
159 \\_kernel\_backend\_literal\_svg:n { <g> }  
160 \int\_set\_eq:NN  
161 \l\_kernel\_backend\_scope\_int  
162 \g\_kernel\_backend\_scope\_int  
163 \group\_begin:  
164 \int\_gset:Nn \g\_kernel\_backend\_scope\_int { 1 }  
165 }  
166 \cs\_new\_protected:Npn \\_kernel\_backend\_scope\_end:  
167 {  
168 \prg\_replicate:nn  
169 { \g\_kernel\_backend\_scope\_int }  
170 { \\_kernel\_backend\_literal\_svg:n { </g> } }  
171 \group\_end:  
172 \int\_gset\_eq:NN  
173 \g\_kernel\_backend\_scope\_int  
174 \l\_kernel\_backend\_scope\_int  
175 }  
176 \cs\_new\_protected:Npn \\_kernel\_backend\_scope\_begin:n #1  
177 {  
178 \\_kernel\_backend\_literal\_svg:n { <g ~ #1 > }  
179 \int\_set\_eq:NN  
180 \l\_kernel\_backend\_scope\_int  
181 \g\_kernel\_backend\_scope\_int  
182 \group\_begin:  
183 \int\_gset:Nn \g\_kernel\_backend\_scope\_int { 1 }  
184 }  
185 \cs\_generate\_variant:Nn \\_kernel\_backend\_scope\_begin:n { x }

```

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

(End definition for \__kernel_backend_scope_begin: and others.)

192 </dvisvgm>
193 </package>

```

## 2 l3backend-box Implementation

```

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

```

### 2.1 dvips backend

```

196 <*dvips>

```

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

```

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

```

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

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

```

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

```

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

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

```

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

```

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

250 </dvips>

## 2.2 LuaTeX and pdfTeX backends

251 <\*luatex | pdftex>

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

```

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

```



```

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

```

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

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

```

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

```

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

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

```

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

```

```

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

```

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

```

307 </luatex | pdftex>

```

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

```

308 <*dvipdfmx | xetex>

```

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

```

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

```

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

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

```

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

```

```

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

```

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

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

```

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

```

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

```

351 </dviPDFmx | xetex>

```

## 2.4 dvisvgm backend

```

352 <*dvisvgm>

```

`\__box_backend_clip:N`  
`\g__kernel_clip_path_int`

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

```

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

```

```

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

```

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

```

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

```

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

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

```

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

```

```

414     \__kernel_backend_scope_end:
415 }

```

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

\\_\_box\_backend\_scale:Nnn

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

```

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

```

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

```

435 </dvisvgm>

```

```

436 </package>

```

## 3 l3backend-color Implementation

```

437 <*package>

```

```

438 <@@=color>

```

Color support is split into parts: collecting data from L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>, the color stack, general color, separations, and color for drawings. We have different approaches in each backend, and have some choices to make about dvipdfmx/X<sub>Y</sub>L<sub>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

```

439 <*dvisvgm | dvipdfmx | dvips | xetex>

```

\\_\_color\_backend\_pickup:N

\\_\_color\_backend\_pickup:w

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). The x-type expansion is there to cover the case where xcolor is in use.

```

440 \cs_new_protected:Npn \__color_backend_pickup:N #1 { }
441 \cs_if_exist:cT { ver@color.sty }

```

```

442 {
443   \cs_set_protected:Npn \__color_backend_pickup:N #1
444   {
445     \exp_args:NV \tl_if_head_is_space:nTF \current@color
446     {
447       \tl_set:Nx #1
448       {
449         { named }
450         { \exp_after:wN \use:n \current@color }
451       }
452     }
453     {
454       \exp_last_unbraced:Nx \__color_backend_pickup:w
455       { \current@color } \s__color_stop #1
456     }
457   }
458   \cs_new_protected:Npn \__color_backend_pickup:w #1 ~ #2 \s__color_stop #3
459   { \tl_set:Nn #3 { {#1} {#2} } }
460 }

```

(End definition for \\_\_color\_backend\_pickup:N and \\_\_color\_backend\_pickup:w.)

```

461 </dvipsvm | dvipdfmx | dvips | xetex>

```

### 3.1.2 LuaTeX and pdfTeX

```

462 < *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

```

463 \cs_new_protected:Npn \__color_backend_pickup:N #1 { }
464 \cs_if_exist:cT { ver@color.sty }
465 {
466   \cs_set_protected:Npn \__color_backend_pickup:N #1
467   {
468     \exp_last_unbraced:Nx \__color_backend_pickup:w
469     { \current@color } ~ 0 ~ 0 ~ 0 \s__color_stop #1
470   }
471   \cs_new_protected:Npn \__color_backend_pickup:w
472   #1 ~ #2 ~ #3 ~ #4 ~ #5 ~ #6 \s__color_stop #7
473   {
474     \str_if_eq:nnTF {#2} { g }
475     { \tl_set:Nn #7 { { gray } {#1} } }
476     {
477       \str_if_eq:nnTF {#4} { rg }
478       { \tl_set:Nn #7 { { rgb } { #1 ~ #2 ~ #3 } } }
479       {
480         \str_if_eq:nnTF {#5} { k }
481         { \tl_set:Nn #7 { { cmyk } { #1 ~ #2 ~ #3 ~ #4 } } }
482         {
483           \str_if_eq:nnTF {#2} { cs }
484           {

```

```

485         \tl_set:Nx #7 { { \use:n #1 } { #5 } }
486     }
487     {
488         \tl_set:Nn #7 { { gray } { 0 } }
489     }
490 }
491 }
492 }
493 }
494 }

```

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

```

495 </luatex | pdftex>

```

## 3.2 The color stack

For PDF-based engines, we have a color stack available inside the specials. This is used for concepts beyond color itself: it is needed to manage the graphics state generally. Although `dvipdfmx/XqTeX` have multiple color stacks in recent releases, the way these interact with the original single stack and with other graphic state operations means that currently it is not feasible to use the multiple stacks.

### 3.2.1 Common code

```

496 <*luatex | pdftex>

```

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

```

497 \int_new:N \l__color_backend_stack_int

```

(End definition for `\l__color_backend_stack_int`.)

```

498 </luatex | pdftex>

```

### 3.2.2 LuaTeX and pdfTeX

```

499 <*luatex | pdftex>

```

`\__kernel_color_backend_stack_init:Nnn`

```

500 \cs_new_protected:Npn \__kernel_color_backend_stack_init:Nnn #1#2#3
501 {
502     \int_const:Nn #1
503     {
504         <*luatex>
505             \tex_pdffeedback:D colorstackinit ~
506         </luatex>
507         <*pdftex>
508             \tex_pdfcolorstackinit:D
509         </pdftex>
510         \tl_if_blank:nF {#2} { #2 ~ }
511         {#3}
512     }
513 }

```

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

```

\__kernel_color_backend_stack_push:nn
\__kernel_color_backend_stack_pop:n

```

```

514 \cs_new_protected:Npn \__kernel_color_backend_stack_push:nn #1#2
515 {
516   \*luatex
517   \tex_pdfextension:D colorstack ~
518   \*pdfTeX
519   \tex_pdfcolorstack:D
520   \int_eval:n {#1} ~ push ~ {#2}
521 }
522 \cs_new_protected:Npn \__kernel_color_backend_stack_pop:n #1
523 {
524   \*luatex
525   \tex_pdfextension:D colorstack ~
526   \*pdfTeX
527   \tex_pdfcolorstack:D
528   \int_eval:n {#1} ~ pop \scan_stop:
529 }

```

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

```

534 \*luatex | pdfTeX

```

### 3.3 General color

#### 3.3.1 dvips-style

```

535 \*dvips | dvisvgm

```

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

```

536 \cs_new_protected:Npn \__color_backend_select_cmyk:n #1
537 { \__color_backend_select:n { cmyk ~ #1 } }
538 \cs_new_protected:Npn \__color_backend_select_gray:n #1
539 { \__color_backend_select:n { gray ~ #1 } }
540 \cs_new_protected:Npn \__color_backend_select_named:n #1
541 { \__color_backend_select:n { ~ #1 } }
542 \cs_new_protected:Npn \__color_backend_select_rgb:n #1
543 { \__color_backend_select:n { rgb ~ #1 } }
544 \cs_new_protected:Npn \__color_backend_select:n #1
545 {
546   \__kernel_backend_literal:n { color~push~ #1 }
547   \*dvips
548   \__kernel_backend_postscript:n { /color.sc ~ { } ~ def }
549   \*dvips
550 }
551 \cs_new_protected:Npn \__color_backend_reset:
552 { \__kernel_backend_literal:n { color~pop } }

```

(End definition for \\_\_color\_backend\_select\_cmyk:n and others. This function is documented on page ??.)

```

553 \*dvips | dvisvgm

```



### 3.3.2 LuaTeX and pdfTeX

554  $\langle *luatex | pdftex \rangle$

$\backslash l\_color\_backend\_fill\_tl$   
 $\backslash l\_color\_backend\_stroke\_tl$

555  $\backslash tl\_new:N \backslash l\_color\_backend\_fill\_tl$   
556  $\backslash tl\_new:N \backslash l\_color\_backend\_stroke\_tl$

(End definition for  $\backslash l\_color\_backend\_fill\_tl$  and  $\backslash l\_color\_backend\_stroke\_tl$ .)

$\backslash\_color\_backend\_select\_cmyk:n$   
 $\backslash\_color\_backend\_select\_gray:n$   
 $\backslash\_color\_backend\_select\_rgb:n$   
 $\backslash\_color\_backend\_select:nn$   
 $\backslash\_color\_backend\_reset:$

Store the values then pass to the stack.

557  $\backslash cs\_new\_protected:Npn \backslash\_color\_backend\_select\_cmyk:n \#1$   
558  $\{ \backslash\_color\_backend\_select:nn \{ \#1 \sim k \} \{ \#1 \sim K \} \}$   
559  $\backslash cs\_new\_protected:Npn \backslash\_color\_backend\_select\_gray:n \#1$   
560  $\{ \backslash\_color\_backend\_select:nn \{ \#1 \sim g \} \{ \#1 \sim G \} \}$   
561  $\backslash cs\_new\_protected:Npn \backslash\_color\_backend\_select\_rgb:n \#1$   
562  $\{ \backslash\_color\_backend\_select:nn \{ \#1 \sim rg \} \{ \#1 \sim RG \} \}$   
563  $\backslash cs\_new\_protected:Npn \backslash\_color\_backend\_select:nn \#1\#2$   
564  $\{$   
565  $\backslash tl\_set:Nn \backslash l\_color\_backend\_fill\_tl \{ \#1 \}$   
566  $\backslash tl\_set:Nn \backslash l\_color\_backend\_stroke\_tl \{ \#2 \}$   
567  $\backslash\_kernel\_color\_backend\_stack\_push:nn \backslash l\_color\_backend\_stack\_int \{ \#1 \sim \#2 \}$   
568  $\}$   
569  $\backslash cs\_new\_protected:Npn \backslash\_color\_backend\_reset:$   
570  $\{ \backslash\_kernel\_color\_backend\_stack\_pop:n \backslash l\_color\_backend\_stack\_int \}$

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

571  $\langle /luatex | pdftex \rangle$

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

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

572  $\langle *dvipdfmx | xetex \rangle$

$\backslash\_color\_backend\_select:n$   
 $\backslash\_color\_backend\_select\_cmyk:n$   
 $\backslash\_color\_backend\_select\_gray:n$   
 $\backslash\_color\_backend\_select\_rgb:n$   
 $\backslash\_color\_backend\_reset:$

Using the single stack is relatively easy as there is only one route.

573  $\backslash cs\_new\_protected:Npn \backslash\_color\_backend\_select:n \#1$   
574  $\{ \backslash\_kernel\_backend\_literal:n \{ pdf : bc \sim [ \#1 ] \} \}$   
575  $\backslash cs\_new\_eq:NN \backslash\_color\_backend\_select\_cmyk:n \backslash\_color\_backend\_select:n$   
576  $\backslash cs\_new\_eq:NN \backslash\_color\_backend\_select\_gray:n \backslash\_color\_backend\_select:n$   
577  $\backslash cs\_new\_eq:NN \backslash\_color\_backend\_select\_rgb:n \backslash\_color\_backend\_select:n$   
578  $\backslash cs\_new\_protected:Npn \backslash\_color\_backend\_reset:$   
579  $\{ \backslash\_kernel\_backend\_literal:n \{ pdf : ec \} \}$

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

`\_color_backend_select_named:n` For classical named colors, the only value we should get is Black.

```

580 \cs_new_protected:Npn \_color_backend_select_named:n #1
581 {
582   \str_if_eq:nnTF {#1} { Black }
583     { \_color_backend_select_gray:n { 0 } }
584     { \msg_error:nnn { color } { unknown-named-color } {#1} }
585 }
586 \msg_new:nnn { color } { unknown-named-color }
587 { Named~color~'~{#1}'~is~not~known. }

```

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

```

588 </dvipdfmx | xetex>

```

### 3.4 Separations

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

```

589 <*dvipdfmx | luatex | pdftex | xetex | dvips>

```

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

`\g_color_backend_colorant_prop`

```

590 \prop_new:N \g_color_backend_colorant_prop

```

(End definition for `\g_color_backend_colorant_prop`.)

`\_color_backend_devicen_colorants:n`

`\_color_backend_devicen_colorants:w`

```

591 \cs_new:Npx \_color_backend_devicen_colorants:n #1
592 {
593   \exp_not:N \tl_if_blank:nF {#1}
594   {
595     \c_space_tl
596     << ~
597     /Colorants ~
598     << ~
599     \exp_not:N \_color_backend_devicen_colorants:w #1 ~
600     \exp_not:N \q_recursion_tail \c_space_tl
601     \exp_not:N \q_recursion_stop
602     >> ~
603     >>
604   }
605 }
606 \cs_new:Npn \_color_backend_devicen_colorants:w #1 ~
607 {
608   \quark_if_recursion_tail_stop:n {#1}
609   \prop_if_in:NnT \g_color_backend_colorant_prop {#1}
610   {
611     #1 ~
612     \prop_item:Nn \g_color_backend_colorant_prop {#1} ~
613   }
614   \_color_backend_devicen_colorants:w
615 }

```

```

(End definition for \_color_backend_devicen_colorants:n and \_color_backend_devicen_colorants:w.)

616 </dvipdfmx | luatex | pdftex | xetex | dvips>
617 <dvips>

\_color_backend_select_separation:nn
\_color_backend_select_devicen:nn
618 \cs_new_protected:Npn \_color_backend_select_separation:nn #1#2
619 { \_color_backend_select:n { separation ~ #1 ~ #2 } }
620 \cs_new_eq:NN \_color_backend_select_devicen:nn \_color_backend_select_separation:nn

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

\_color_backend_select_iccbased:nn No support.
621 \cs_new_protected:Npn \_color_backend_select_iccbased:nn #1#2 { }

(End definition for \_color_backend_select_iccbased:nn.)

\_color_backend_separation_init:nnnnn
\_color_backend_separation_init:nxxnn
\_color_backend_separation_init_aux:nnnnnn
lor_backend_separation_init/DeviceCMYK:nnn
lor_backend_separation_init/DeviceGray:nnn
lor_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.

622 \cs_new_protected:Npx \_color_backend_separation_init:nnnnn #1#2#3#4#5
623 {
624   \bool_if:NT \g__kernel_backend_header_bool
625   {
626     \exp_args:Nx \_kernel_backend_first_shipout:n
627     {
628       \exp_not:N \_color_backend_separation_init_aux:nnnnnn
629       { \exp_not:N \int_use:N \g__color_model_int }
630       {#1} {#2} {#3} {#4} {#5}
631     }
632     \prop_gput:Nxx \exp_not:N \g__color_backend_colorant_prop
633     { / \exp_not:N \str_convert_pdfname:n {#1} }
634     {
635       << ~
636       /setcolorspace ~ {} ~
637       >> ~ begin ~
638       color \exp_not:N \int_use:N \g__color_model_int \c_space_tl
639       end
640     }
641   }
642 }
643 \cs_generate_variant:Nn \_color_backend_separation_init:nnnnn { nxx }
644 \cs_new_protected:Npn \_color_backend_separation_init_aux:nnnnnn #1#2#3#4#5#6
645 {
646   \_kernel_backend_literal:e
647   {
648     !
649     TeXDict ~ begin ~
650     /color #1
651     {
652       [ ~
653       /Separation ~ ( \str_convert_pdfname:n {#2} ) ~
654       [ ~ #3 ~ ] ~

```

```

655         {
656             \cs_if_exist_use:cF { __color_backend_separation_init_ #3 :nnn }
657             { \__color_backend_separation_init:nnn }
658             {#4} {#5} {#6}
659         }
660     ] ~ setcolorspace
661 } ~ def ~
662 end
663 }
664 }
665 \cs_new:cpn { __color_backend_separation_init_ /DeviceCMYK :nnn } #1#2#3
666 { \__color_backend_separation_init_Device:Nn 4 {#3} }
667 \cs_new:cpn { __color_backend_separation_init_ /DeviceGray :nnn } #1#2#3
668 { \__color_backend_separation_init_Device:Nn 1 {#3} }
669 \cs_new:cpn { __color_backend_separation_init_ /DeviceRGB :nnn } #1#2#3
670 { \__color_backend_separation_init_Device:Nn 2 {#3} }
671 \cs_new:Npn \__color_backend_separation_init_Device:Nn #1#2
672 {
673     #2 ~
674     \prg_replicate:nn {#1}
675     { #1 ~ index ~ mul ~ #1 ~ 1 ~ roll ~ }
676     \int_eval:n { #1 + 1 } ~ -1 ~ roll ~ pop
677 }

```

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.

```

678 \cs_new:Npn \__color_backend_separation_init:nnn #1#2#3
679 {
680     \exp_args:Ne \__color_backend_separation_init:nnnn
681     { \__color_backend_separation_init_count:n {#2} }
682     {#1} {#2} {#3}
683 }
684 \cs_new:Npn \__color_backend_separation_init_count:n #1
685 { \int_eval:n { 0 \__color_backend_separation_init_count:w #1 ~ \s__color_stop } }
686 \cs_new:Npn \__color_backend_separation_init_count:w #1 ~ #2 \s__color_stop
687 {
688     +1
689     \tl_if_blank:nF {#2}
690     { \__color_backend_separation_init_count:w #2 \s__color_stop }
691 }

```

Now we implement the algorithm. In the terms in the PostScript manual, we have **N** = 1 and **Domain** = [0 1], with **Range** as #2, **C0** as #3 and **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 **C0** and **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.

```

692 \cs_new:Npn \__color_backend_separation_init:nnnn #1#2#3#4
693 {
694   \__color_backend_separation_init:w #3 ~ \s__color_stop #4 ~ \s__color_stop
695   \prg_replicate:nn {#1}
696   {
697     pop ~ 1 ~ index ~ neg ~ 1 ~ index ~ add ~
698     \int_eval:n { 3 * #1 } ~ index ~ mul ~
699     2 ~ index ~ add ~
700     \int_eval:n { 3 * #1 } ~ #1 ~ roll ~
701   }
702   \int_step_function:nnnN {#1} { -1 } { 1 }
703   \__color_backend_separation_init:n
704   \int_eval:n { 4 * #1 + 1 } ~ #1 ~ roll ~
705   \prg_replicate:nn { 3 * #1 + 1 } { pop ~ }
706   \tl_if_blank:nF {#2}
707   { \__color_backend_separation_init:nw {#1} #2 ~ \s__color_stop }
708 }
709 \cs_new:Npn \__color_backend_separation_init:w
710 #1 ~ #2 \s__color_stop #3 ~ #4 \s__color_stop
711 {
712   #1 ~ #3 ~ 0 ~
713   \tl_if_blank:nF {#2}
714   { \__color_backend_separation_init:w #2 \s__color_stop #4 \s__color_stop }
715 }
716 \cs_new:Npn \__color_backend_separation_init:n #1
717 { \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.

```

718 \cs_new:Npn \__color_backend_separation_init:nw #1#2 ~ #3 ~ #4 \s__color_stop
719 {
720   #2 ~ #3 ~
721   2 ~ index ~ 2 ~ index ~ lt ~
722   { ~ pop ~ exch ~ pop ~ } ~
723   { ~
724     2 ~ index ~ 1 ~ index ~ gt ~
725     { ~ exch ~ pop ~ exch ~ pop ~ } ~
726     { ~ pop ~ pop ~ } ~
727     ifelse ~
728   }
729   ifelse ~
730   #1 ~ 1 ~ roll ~
731   \tl_if_blank:nF {#4}
732   { \__color_backend_separation_init:nw {#1} #4 \s__color_stop }
733 }

```

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.

```

734 \cs_new_protected:Npn \__color_backend_separation_init_CIELAB:nnn #1#2#3
735 {
736   \__color_backend_separation_init:nxxxnn
737   {#2}
738   {
739     /CIEBasedABC ~

```

```

740 << ~
741 /RangeABC ~ [ ~ \c__color_model_range_CIELAB_t1 \c_space_t1 ] ~
742 /DecodeABC ~
743 [ ~
744 { ~ 16 ~ add ~ 116 ~ div ~ } ~ bind ~
745 { ~ 500 ~ div ~ } ~ bind ~
746 { ~ 200 ~ div ~ } ~ bind ~
747 ] ~
748 /MatrixABC ~ [ ~ 1 ~ 1 ~ 1 ~ 1 ~ 0 ~ 0 ~ 0 ~ 0 ~ -1 ~ ] ~
749 /DecodeLMN ~
750 [ ~
751 { ~
752 dup ~ 6 ~ 29 ~ div ~ ge ~
753 { ~ dup ~ dup ~ mul ~ mul ~ } ~
754 { ~ 4 ~ 29 ~ div ~ sub ~ 108 ~ 841 ~ div ~ mul ~ } ~
755 ifelse ~
756 0.9505 ~ mul ~
757 } ~ bind ~
758 { ~
759 dup ~ 6 ~ 29 ~ div ~ ge ~
760 { ~ dup ~ dup ~ mul ~ mul ~ } ~
761 { ~ 4 ~ 29 ~ div ~ sub ~ 108 ~ 841 ~ div ~ mul ~ } ~
762 ifelse ~
763 } ~ bind ~
764 { ~
765 dup ~ 6 ~ 29 ~ div ~ ge ~
766 { ~ dup ~ dup ~ mul ~ mul ~ } ~
767 { ~ 4 ~ 29 ~ div ~ sub ~ 108 ~ 841 ~ div ~ mul ~ } ~
768 ifelse ~
769 1.0890 ~ mul ~
770 } ~ bind
771 ] ~
772 /WhitePoint ~
773 [ ~ \tl_use:c { c__color_model_whitepoint_CIELAB_ #1 _tl } ~ ] ~
774 >>
775 }
776 { \c__color_model_range_CIELAB_t1 }
777 { 100 ~ 0 ~ 0 }
778 {#3}
779 }

```

(End definition for \\_color\_backend\_separation\_init:nnnnn and others.)

\\_color\_backend\_devicen\_init:nnn Trivial as almost all of the work occurs in the shared code.

```

780 \cs_new_protected:Npn \_color_backend_devicen_init:nnn #1#2#3
781 {
782   \_kernel_backend_literal:e
783   {
784     !
785     TeXDict ~ begin ~
786     /color \int_use:N \g__color_model_int
787     {
788       [ ~
789       /DeviceN ~

```

```

790         [ ~ #1 ~ ] ~
791         #2 ~
792         { ~ #3 ~ } ~
793         \_color_backend_devicen_colorants:n {#1}
794     ] ~ setcolorspace
795 } ~ def ~
796 end
797 }
798 }

```

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

\\_color\_backend\_iccbased\_init:nnn No support at present.

```

799 \cs_new_protected:Npn \_color_backend_iccbased_init:nnn #1#2#3 { }

```

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

```

800 </dvips>
801 <*dvisvgm>

```

\\_color\_backend\_select\_separation:nn No support at present.

\\_color\_backend\_select\_devicen:nn

```

802 \cs_new_protected:Npn \_color_backend_select_separation:nn #1#2 { }
803 \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 No support at present.

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

```

804 \cs_new_protected:Npn \_color_backend_separation_init:nnnnn #1#2#3#4#5 { }
805 \cs_new_protected:Npn \_color_backend_separation_init_CIELAB:nnnnnn #1#2#3 { }

```

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

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

```

806 \cs_new_protected:Npn \_color_backend_select_iccbased:nn #1#2
807 {
808     \_kernel_backend_literal_svg:x
809     {
810         <style>
811             @color-profile ~
812             \str_if_eq:nnTF {#2} { cmyk }
813             { device-cmyk }
814             { --color \int_use:N \g__color_model_int }
815             \c_space_tl
816             {
817                 src:("#1")
818             }
819         </style>
820     }
821 }

```

(End definition for \\_color\_backend\_select\_iccbased:nn.)

```

822 </dvisvgm>
823 <*dviPDFmx | luatex | pdftex | xetex>

```

`\_color_backend_select_separation:nn`  
`\_color_backend_select_devicen:nn`  
`\_color_backend_select_iccbased:nn`

```

824 <*dvipdfmx | xetex>
825 \cs_new_protected:Npn \_color_backend_select_separation:nn #1#2
826 { \_kernel_backend_literal:x { pdf : bc ~ \pdf_object_ref:n {#1} ~ [ #2 ] } }
827 </dvipdfmx | xetex>
828 <*luatex | pdftex>
829 \cs_new_protected:Npn \_color_backend_select_separation:nn #1#2
830 { \_color_backend_select:nn { /#1 ~ cs ~ #2 ~ scn } { /#1 ~ CS ~ #2 ~ SCN } }
831 </luatex | pdftex>
832 \cs_new_eq:NN \_color_backend_select_devicen:nn \_color_backend_select_separation:nn
833 \cs_new_eq:NN \_color_backend_select_iccbased:nn \_color_backend_select_separation:nn

```

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

`\_color_backend_init_resource:n`

Resource initiation comes up a few times. For dvipdfmx/X<sub>Y</sub>TeX, we skip this as at present it's handled by the backend.

```

834 \cs_new_protected:Npn \_color_backend_init_resource:n #1
835 {
836 <*luatex | pdftex>
837   \bool_lazy_and:nnT
838     { \cs_if_exist_p:N \pdfmanagement_if_active_p: }
839     { \pdfmanagement_if_active_p: }
840   {
841     \use:x
842     {
843       \pdfmanagement_add:nnn
844         { Page / Resources / ColorSpace }
845         { #1 }
846         { \pdf_object_ref_last: }
847     }
848   }
849 </luatex | pdftex>
850 }

```

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

`\_color_backend_separation_init:nnnnn`  
`\_color_backend_separation_init:nn`  
`\_color_backend_separation_init_CIELAB:nnn`

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

```

851 \cs_new_protected:Npn \_color_backend_separation_init:nnnnn #1#2#3#4#5
852 {
853   \pdf_object_unnamed_write:nx { dict }
854   {
855     /FunctionType ~ 2
856     /Domain ~ [ 0 ~ 1 ]
857     \tl_if_blank:nF {#3} { /Range ~ [ #3 ] }
858     /C0 ~ [ #4 ] ~
859     /C1 ~ [ #5 ] /N ~ 1
860   }
861   \exp_args:Nx \_color_backend_separation_init:nn
862     { \str_convert_pdfname:n {#1} } {#2}
863   \_color_backend_init_resource:n { color \int_use:N \g_color_model_int }

```



```

864 }
865 \cs_new_protected:Npn \__color_backend_separation_init:nn #1#2
866 {
867   \use:x
868   {
869     \pdf_object_new:nn { color \int_use:N \g__color_model_int } { array }
870     \pdf_object_write:nn { color \int_use:N \g__color_model_int }
871     { /Separation /#1 ~ #2 ~ \pdf_object_ref_last: }
872   }
873   \prop_gput:Nnx \g__color_backend_colorant_prop { /#1 }
874   { \pdf_object_ref_last: }
875 }

```

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

```

876 \cs_new_protected:Npn \__color_backend_separation_init_CIELAB:nnn #1#2#3
877 {
878   \pdf_object_if_exist:nF { __color_illuminant_CIELAB_ #1 }
879   {
880     \pdf_object_new:nn { __color_illuminant_CIELAB_ #1 } { array }
881     \pdf_object_write:nx { __color_illuminant_CIELAB_ #1 }
882     {
883       /Lab ~
884       <<
885       /WhitePoint ~
886       [ \tl_use:c { c__color_model_whitepoint_CIELAB_ #1 _tl } ]
887       /Range ~ [ \c__color_model_range_CIELAB_tl ]
888       >>
889     }
890   }
891   \__color_backend_separation_init:nnnnn
892   {#2}
893   { \pdf_object_ref:n { __color_illuminant_CIELAB_ #1 } }
894   { \c__color_model_range_CIELAB_tl }
895   { 100 ~ 0 ~ 0 }
896   {#3}
897 }

```

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

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

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

```

898 \cs_new_protected:Npn \__color_backend_devicen_init:nnn #1#2#3
899 {
900   \pdf_object_unnamed_write:nx { stream }
901   {
902     {
903       /FunctionType ~ 4 ~
904       /Domain ~
905       [ ~
906         \prg_replicate:nn
907         { 0 \__color_backend_devicen_init:w #1 ~ \s__color_stop }
908         { 0 ~ 1 ~ }
909       ] ~

```

```

910         /Range ~
911         [ ~
912         \str_case:nn {#2}
913         {
914             { /DeviceCMYK } { 0 ~ 1 ~ 0 ~ 1 ~ 0 ~ 1 ~ 0 ~ 1 }
915             { /DeviceGray } { 0 ~ 1 }
916             { /DeviceRGB } { 0 ~ 1 ~ 0 ~ 1 ~ 0 ~ 1 }
917         } ~
918         ]
919     }
920     { {#3} }
921 }
922 \use:x
923 {
924     \pdf_object_new:nn { color \int_use:N \g__color_model_int } { array }
925     \pdf_object_write:nn { color \int_use:N \g__color_model_int }
926     {
927         /DeviceN ~
928         [ ~ #1 ~ ] ~
929         #2 ~
930         \pdf_object_ref_last:
931         \__color_backend_devicen_colorants:n {#1}
932     }
933 }
934 \__color_backend_init_resource:n { color \int_use:N \g__color_model_int }
935 }
936 \cs_new:Npn \__color_backend_devicen_init:w #1 ~ #2 \s__color_stop
937 {
938     + 1
939     \tl_if_blank:nF {#2}
940     { \__color_backend_devicen_init:w #2 \s__color_stop }
941 }

```

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

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

```

942 \cs_new_protected:Npn \__color_backend_iccbased_init:nnn #1#2#3
943 {
944     \pdf_object_if_exist:nF { \__color_icc_ #1 }
945     {
946         \pdf_object_new:nn { \__color_icc_ #1 } { fstream }
947         \pdf_object_write:nx { \__color_icc_ #1 }
948         {
949             {
950                 /N ~ \exp_not:n { #2 } ~
951                 \tl_if_empty:nF { #3 } { /Range~[ #3 ] }
952             }
953             {#1}
954         }
955     }
956     \pdf_object_unnamed_write:nx { array }
957     { /ICCBased ~ \pdf_object_ref:n { \__color_icc_ #1 } }
958     \__color_backend_init_resource:n { color \int_use:N \g__color_model_int }
959 }

```

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

`\_color_backend_iccbased_device:nnn` This is very similar to setting up a color space: the only part we add to the page resources differently.

```

960 \cs_new_protected:Npn \_color_backend_iccbased_device:nnn #1#2#3
961 {
962   \pdf_object_if_exist:nF { \_color_icc_ #1 }
963   {
964     \pdf_object_new:nn { \_color_icc_ #1 } { fstream }
965     \pdf_object_write:nn { \_color_icc_ #1 }
966     {
967       { /N ~ #3 }
968       {#1}
969     }
970   }
971   \pdf_object_unnamed_write:nx { array }
972   { /ICCBased ~ \pdf_object_ref:n { \_color_icc_ #1 } }
973   \_color_backend_init_resource:n { Default #2 }
974 }

```

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

```

975 </dvipdfmx | luatex | pdftex | xetex>

```

### 3.5 Fill and stroke color

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

```

976 <*dvipdfmx | xetex>

```

```

\_color_backend_fill:n
\_color_backend_fill_cmyk:n
\_color_backend_fill_gray:n
\_color_backend_fill_rgb:n
\_color_backend_stroke:n
  \_color_backend_stroke_cmyk:n
  \_color_backend_stroke_gray:n
  \_color_backend_stroke_rgb:n
977 \cs_new_protected:Npn \_color_backend_fill:n #1
978 { \_kernel_backend_literal:n { pdf : bc ~ fill ~ [ #1 ] } }
979 \cs_new_eq:NN \_color_backend_fill_cmyk:n \_color_backend_fill:n
980 \cs_new_eq:NN \_color_backend_fill_gray:n \_color_backend_fill:n
981 \cs_new_eq:NN \_color_backend_fill_rgb:n \_color_backend_fill:n
982 \cs_new_protected:Npn \_color_backend_stroke:n #1
983 { \_kernel_backend_literal:n { pdf : bc ~ stroke ~ [ #1 ] } }
984 \cs_new_eq:NN \_color_backend_stroke_cmyk:n \_color_backend_stroke:n
985 \cs_new_eq:NN \_color_backend_stroke_gray:n \_color_backend_stroke:n
986 \cs_new_eq:NN \_color_backend_stroke_rgb:n \_color_backend_stroke:n

```

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

```

\_color_backend_fill_separation:nn
\_color_backend_stroke_separation:nn
\_color_backend_fill_device:nn
\_color_backend_stroke_device:nn
987 \cs_new_protected:Npn \_color_backend_fill_separation:nn #1#2
988 {
989   \_kernel_backend_literal:x
990   { pdf : bc ~ fill ~ \pdf_object_ref:n {#1} ~ [ #2 ] }
991 }
992 \cs_new_protected:Npn \_color_backend_stroke_separation:nn #1#2

```

```

993 {
994   \__kernel_backend_literal:x
995   { pdf : bc ~ stroke ~ \pdf_object_ref:n {#1} ~ [ #2 ] }
996 }
997 \cs_new_eq:NN \__color_backend_fill_devicen:nn \__color_backend_fill_separation:nn
998 \cs_new_eq:NN \__color_backend_stroke_devicen:nn \__color_backend_stroke_separation:nn

```

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

```

\__color_backend_fill_reset:
  \__color_backend_stroke_reset:
999 \cs_new_eq:NN \__color_backend_fill_reset: \__color_backend_reset:
1000 \cs_new_eq:NN \__color_backend_stroke_reset: \__color_backend_reset:

(End definition for \__color_backend_fill_reset: and \__color_backend_stroke_reset:.)

1001 </dviptdpmx | xetex>
1002 <*luatex | pdftex>

```

\\_\_color\_backend\_fill\_cmyk:n Drawing (fill/stroke) color is handled in dvipdfmx/X<sub>q</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.

```

1003 \cs_new_protected:Npn \__color_backend_fill_cmyk:n #1
1004 { \__color_backend_fill:n { #1 ~ k } }
1005 \cs_new_protected:Npn \__color_backend_fill_gray:n #1
1006 { \__color_backend_fill:n { #1 ~ g } }
1007 \cs_new_protected:Npn \__color_backend_fill_rgb:n #1
1008 { \__color_backend_fill:n { #1 ~ rg } }
1009 \cs_new_protected:Npn \__color_backend_fill:n #1
1010 {
1011   \tl_set:Nn \l__color_backend_fill_tl {#1}
1012   \__kernel_color_backend_stack_push:nn \l__color_backend_stack_int
1013   { #1 ~ \l__color_backend_stroke_tl }
1014 }
1015 \cs_new_protected:Npn \__color_backend_stroke_cmyk:n #1
1016 { \__color_backend_stroke:n { #1 ~ K } }
1017 \cs_new_protected:Npn \__color_backend_stroke_gray:n #1
1018 { \__color_backend_stroke:n { #1 ~ G } }
1019 \cs_new_protected:Npn \__color_backend_stroke_rgb:n #1
1020 { \__color_backend_stroke:n { #1 ~ RG } }
1021 \cs_new_protected:Npn \__color_backend_stroke:n #1
1022 {
1023   \tl_set:Nn \l__color_backend_stroke_tl {#1}
1024   \__kernel_color_backend_stack_push:nn \l__color_backend_stack_int
1025   { \l__color_backend_fill_tl \c_space_tl #1 }
1026 }

```

(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
1027 \cs_new_protected:Npn \__color_backend_fill_separation:nn #1#2
1028 { \__color_backend_fill:n { /#1 ~ cs ~ #2 ~ scn } }
1029 \cs_new_protected:Npn \__color_backend_stroke_separation:nn #1#2
1030 { \__color_backend_stroke:n { /#1 ~ CS ~ #2 ~ SCN } }
1031 \cs_new_eq:NN \__color_backend_fill_devicen:nn \__color_backend_fill_separation:nn
1032 \cs_new_eq:NN \__color_backend_stroke_devicen:nn \__color_backend_stroke_separation:nn

```

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

```
\_color_backend_fill_reset:
  \_color_backend_stroke_reset:
1033 \cs_new_eq:NN \_color_backend_fill_reset: \_color_backend_reset:
1034 \cs_new_eq:NN \_color_backend_stroke_reset: \_color_backend_reset:

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

1035 </luatex | pdftex>
1036 <*dvips>
```

```
\_color_backend_fill_cmyk:n Fill color here is the same as general color except we skip the stroke part.
\_color_backend_fill_gray:n
\_color_backend_fill_rgb:n
  \_color_backend_fill:n
    \_color_backend_stroke_cmyk:n
    \_color_backend_stroke_gray:n
    \_color_backend_stroke_rgb:n
1037 \cs_new_protected:Npn \_color_backend_fill_cmyk:n #1
1038 { \_color_backend_fill:n { cmyk ~ #1 } }
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 }
1047 \cs_new_protected:Npn \_color_backend_stroke_cmyk:n #1
1048 { \_kernel_backend_postscript:n { /color.sc { #1 ~ setcmykcolor } def } }
1049 \cs_new_protected:Npn \_color_backend_stroke_gray:n #1
1050 { \_kernel_backend_postscript:n { /color.sc { #1 ~ setgray } def } }
1051 \cs_new_protected:Npn \_color_backend_stroke_rgb:n #1
1052 { \_kernel_backend_postscript:n { /color.sc { #1 ~ setrgbcolor } def } }

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

```
\_color_backend_fill_separation:nn
\_color_backend_stroke_separation:nn
  \_color_backend_fill_devicen:nn
  \_color_backend_stroke_devicen:nn
1053 \cs_new_protected:Npn \_color_backend_fill_separation:nn #1#2
1054 { \_color_backend_fill:n { separation ~ #1 ~ #2 } }
1055 \cs_new_protected:Npn \_color_backend_stroke_separation:nn #1#2
1056 { \_kernel_backend_postscript:n { /color.sc { separation ~ #1 ~ #2 } def } }
1057 \cs_new_eq:NN \_color_backend_fill_devicen:nn \_color_backend_fill_separation:nn
1058 \cs_new_eq:NN \_color_backend_stroke_devicen:nn \_color_backend_stroke_separation:nn

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

```
\_color_backend_fill_reset:
  \_color_backend_stroke_reset:
1059 \cs_new_eq:NN \_color_backend_fill_reset: \_color_backend_reset:
1060 \cs_new_protected:Npn \_color_backend_stroke_reset: { }

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

1061 </dvips>
1062 <*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 1063 \cs_new_protected:Npn \_color_backend_fill_cmyk:n #1
\_color_backend_fill_rgb:n 1064 { \_color_backend_fill:n { cmyk ~ #1 } }
\_color_backend_fill:n 1065 \cs_new_protected:Npn \_color_backend_fill_gray:n #1
1066 { \_color_backend_fill:n { gray ~ #1 } }
1067 \cs_new_protected:Npn \_color_backend_fill_rgb:n #1
1068 { \_color_backend_fill:n { rgb ~ #1 } }
1069 \cs_new_protected:Npn \_color_backend_fill:n #1
1070 {
1071 \_kernel_backend_literal:n { color~push~ #1 }
1072 }

```

(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 1073 \cs_new_protected:Npn \_color_backend_stroke_cmyk:n #1
\_color_backend_stroke_gray:n 1074 { \_color_backend_cmyk:w #1 \s__color_stop }
\_color_backend_stroke_gray:w 1075 \cs_new_protected:Npn \_color_backend_stroke_cmyk:w
\_color_backend_stroke_rgb:n 1076 #1 ~ #2 ~ #3 ~ #4 \s__color_stop
\_color_backend_stroke_rgb:w 1077 {
\_color_backend:nnn 1078 \use:x
1079 {
1080 \_color_backend:nnn
1081 { \fp_eval:n { -100 * ( 1 - min ( 1 , #1 + #4 ) ) } }
1082 { \fp_eval:n { -100 * ( 1 - min ( 1 , #2 + #4 ) ) } }
1083 { \fp_eval:n { -100 * ( 1 - min ( 1 , #3 + #4 ) ) } }
1084 }
1085 }
1086 \cs_new_protected:Npn \_color_backend_stroke_gray:n #1
1087 {
1088 \use:x
1089 {
1090 \_color_backend_stroke_gray_aux:n
1091 { \fp_eval:n { 100 * (#1) } }
1092 }
1093 }
1094 \cs_new_protected:Npn \_color_backend_stroke_gray_aux:n #1
1095 { \_color_backend:nnn {#1} {#1} {#1} }
1096 \cs_new_protected:Npn \_color_backend_stroke_rgb:n #1
1097 { \_color_backend_rgb:w #1 \s__color_stop }
1098 \cs_new_protected:Npn \_color_backend_stroke_rgb:w
1099 #1 ~ #2 ~ #3 \s__color_stop
1100 {
1101 \use:x
1102 {
1103 \_color_backend:nnn
1104 { \fp_eval:n { 100 * (#1) } }
1105 { \fp_eval:n { 100 * (#2) } }
1106 { \fp_eval:n { 100 * (#3) } }
1107 }
1108 }
1109 \cs_new_protected:Npx \_color_backend:nnn #1#2#3
1110 {

```

```

1111     \__kernel_backend_scope:n
1112     {
1113         stroke =
1114         "
1115         rgb
1116         (
1117             #1 \c_percent_str ,
1118             #2 \c_percent_str ,
1119             #3 \c_percent_str
1120         )
1121         "
1122     }
1123 }

```

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

At present, these are no-ops.

```

1124 \cs_new_protected:Npn \__color_backend_fill_separation:nn #1#2 { }
1125 \cs_new_protected:Npn \__color_backend_stroke_separation:nn #1#2 { }
1126 \cs_new_eq:NN \__color_backend_fill_devicen:nn \__color_backend_fill_separation:nn
1127 \cs_new_eq:NN \__color_backend_stroke_devicen:nn \__color_backend_stroke_separation:nn

```

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

```

\__color_backend_fill_reset:
\__color_backend_stroke_reset:
1128 \cs_new_eq:NN \__color_backend_fill_reset: \__color_backend_reset:
1129 \cs_new_protected:Npn \__color_backend_stroke_reset: { }

```

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

No support at present.

```

1130 \cs_new_protected:Npn \__color_backend_devicen_init:nnn #1#2#3 { }
1131 \cs_new_protected:Npn \__color_backend_iccbased_init:nnn #1#2#3 { }

```

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

```

1132 </divisgms>
1133 </package>

```

## 4 l3backend-draw Implementation

```

1134 <*package>
1135 <@@=draw>

```

### 4.1 dvips backend

```

1136 <*dvips>

```

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

```

1137 \cs_new_eq:NN \__draw_backend_literal:n \__kernel_backend_literal_postscript:n
1138 \cs_generate_variant:Nn \__draw_backend_literal:n { x }

```

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

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

```

1139 \cs_new_protected:Npn \__draw_backend_begin:
1140 {
1141   \__kernel_backend_literal:n { ps::[begin] }
1142   \__draw_backend_literal:n { @beginspecial }
1143 }
1144 \cs_new_protected:Npn \__draw_backend_end:
1145 {
1146   \__draw_backend_literal:n { @endspecial }
1147   \__kernel_backend_literal:n { ps::[end] }
1148 }

```

(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 the graphic state has to be sent to the stack.

`\__draw_backend_scope_end:`

```

1149 \cs_new_protected:Npn \__draw_backend_scope_begin:
1150 { \__draw_backend_literal:n { save } }
1151 \cs_new_protected:Npn \__draw_backend_scope_end:
1152 { \__draw_backend_literal:n { restore } }

```

(End definition for `\__draw_backend_scope_begin:` and `\__draw_backend_scope_end:.`)

`\__draw_backend_moveto:nn`

`\__draw_backend_lineto:nn`

`\__draw_backend_rectangle:nnnn`

`\__draw_backend_curveto:nnnnnn`

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

```

1153 \cs_new_protected:Npn \__draw_backend_moveto:nn #1#2
1154 {
1155   \__draw_backend_literal:x
1156   {
1157     \dim_to_decimal_in_bp:n {#1} ~
1158     \dim_to_decimal_in_bp:n {#2} ~ moveto
1159   }
1160 }
1161 \cs_new_protected:Npn \__draw_backend_lineto:nn #1#2
1162 {
1163   \__draw_backend_literal:x
1164   {
1165     \dim_to_decimal_in_bp:n {#1} ~
1166     \dim_to_decimal_in_bp:n {#2} ~ lineto
1167   }
1168 }
1169 \cs_new_protected:Npn \__draw_backend_rectangle:nnnn #1#2#3#4

```



```

1170 {
1171   \_draw_backend_literal:x
1172   {
1173     \dim_to_decimal_in_bp:n {#4} ~ \dim_to_decimal_in_bp:n {#3} ~
1174     \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~
1175     moveto~dup~0~rlineto~exch~0~exch~rlineto~neg~0~rlineto~closepath
1176   }
1177 }
1178 \cs_new_protected:Npn \_draw_backend_curveto:nnnnnn #1#2#3#4#5#6
1179 {
1180   \_draw_backend_literal:x
1181   {
1182     \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~
1183     \dim_to_decimal_in_bp:n {#3} ~ \dim_to_decimal_in_bp:n {#4} ~
1184     \dim_to_decimal_in_bp:n {#5} ~ \dim_to_decimal_in_bp:n {#6} ~
1185     curveto
1186   }
1187 }

```

(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
1188 \cs_new_protected:Npn \_draw_backend_evenodd_rule:
1189 { \bool_gset_true:N \g__draw_draw_eor_bool }
1190 \cs_new_protected:Npn \_draw_backend_nonzero_rule:
1191 { \bool_gset_false:N \g__draw_draw_eor_bool }
1192 \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. All of the operations end with a new path instruction as they do not terminate (again in contrast to PDF).

```

\_draw_backend_stroke:
\_draw_backend_closestroke:
\_draw_backend_fill:
\_draw_backend_fillstroke:
\_draw_backend_clip:
\_draw_backend_discardpath:
\g__draw_draw_clip_bool
1193 \cs_new_protected:Npn \_draw_backend_closepath:
1194 { \_draw_backend_literal:n { closepath } }
1195 \cs_new_protected:Npn \_draw_backend_stroke:
1196 {
1197   \_draw_backend_literal:n { gsave }
1198   \_draw_backend_literal:n { color.sc }
1199   \_draw_backend_literal:n { stroke }
1200   \_draw_backend_literal:n { grestore }
1201   \bool_if:NT \g__draw_draw_clip_bool
1202   {
1203     \_draw_backend_literal:x
1204     {
1205       \bool_if:NT \g__draw_draw_eor_bool { eo }
1206       clip
1207     }
1208   }
1209   \_draw_backend_literal:n { newpath }

```

```

1210     \bool_gset_false:N \g__draw_draw_clip_bool
1211   }
1212   \cs_new_protected:Npn \__draw_backend_closestroke:
1213   {
1214     \__draw_backend_closepath:
1215     \__draw_backend_stroke:
1216   }
1217   \cs_new_protected:Npn \__draw_backend_fill:
1218   {
1219     \__draw_backend_literal:x
1220     {
1221       \bool_if:NT \g__draw_draw_eor_bool { eo }
1222       fill
1223     }
1224     \bool_if:NT \g__draw_draw_clip_bool
1225     {
1226       \__draw_backend_literal:x
1227       {
1228         \bool_if:NT \g__draw_draw_eor_bool { eo }
1229         clip
1230       }
1231     }
1232     \__draw_backend_literal:n { newpath }
1233     \bool_gset_false:N \g__draw_draw_clip_bool
1234   }
1235   \cs_new_protected:Npn \__draw_backend_fillstroke:
1236   {
1237     \__draw_backend_literal:x
1238     {
1239       \bool_if:NT \g__draw_draw_eor_bool { eo }
1240       fill
1241     }
1242     \__draw_backend_literal:n { gsave }
1243     \__draw_backend_literal:n { color.sc }
1244     \__draw_backend_literal:n { stroke }
1245     \__draw_backend_literal:n { grestore }
1246     \bool_if:NT \g__draw_draw_clip_bool
1247     {
1248       \__draw_backend_literal:x
1249       {
1250         \bool_if:NT \g__draw_draw_eor_bool { eo }
1251         clip
1252       }
1253     }
1254     \__draw_backend_literal:n { newpath }
1255     \bool_gset_false:N \g__draw_draw_clip_bool
1256   }
1257   \cs_new_protected:Npn \__draw_backend_clip:
1258   { \bool_gset_true:N \g__draw_draw_clip_bool }
1259   \bool_new:N \g__draw_draw_clip_bool
1260   \cs_new_protected:Npn \__draw_backend_discardpath:
1261   {
1262     \bool_if:NT \g__draw_draw_clip_bool
1263     {

```

```

1264         \__draw_backend_literal:x
1265         {
1266             \bool_if:NT \g__draw_draw_eor_bool { eo }
1267             clip
1268         }
1269     }
1270     \__draw_backend_literal:n { newpath }
1271     \bool_gset_false:N \g__draw_draw_clip_bool
1272 }

```

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

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

```

\__draw_backend_dash_pattern:nn
\__draw_backend_dash:n
\__draw_backend_linewidth:n
\__draw_backend_miterlimit:n
\__draw_backend_cap_but:
\__draw_backend_cap_round:
\__draw_backend_cap_rectangle:
\__draw_backend_join_miter:
\__draw_backend_join_round:
\__draw_backend_join_bevel:
1273 \cs_new_protected:Npn \__draw_backend_dash_pattern:nn #1#2
1274 {
1275     \__draw_backend_literal:x
1276     {
1277         [
1278             \exp_args:Nf \use:n
1279             { \clist_map_function:nN {#1} \__draw_backend_dash:n }
1280         ] ~
1281         \dim_to_decimal_in_bp:n {#2} ~ setdash
1282     }
1283 }
1284 \cs_new:Npn \__draw_backend_dash:n #1
1285 { ~ \dim_to_decimal_in_bp:n {#1} }
1286 \cs_new_protected:Npn \__draw_backend_linewidth:n #1
1287 {
1288     \__draw_backend_literal:x
1289     { \dim_to_decimal_in_bp:n {#1} ~ setlinewidth }
1290 }
1291 \cs_new_protected:Npn \__draw_backend_miterlimit:n #1
1292 { \__draw_backend_literal:n { #1 ~ setmiterlimit } }
1293 \cs_new_protected:Npn \__draw_backend_cap_but:
1294 { \__draw_backend_literal:n { 0 ~ setlinecap } }
1295 \cs_new_protected:Npn \__draw_backend_cap_round:
1296 { \__draw_backend_literal:n { 1 ~ setlinecap } }
1297 \cs_new_protected:Npn \__draw_backend_cap_rectangle:
1298 { \__draw_backend_literal:n { 2 ~ setlinecap } }
1299 \cs_new_protected:Npn \__draw_backend_join_miter:
1300 { \__draw_backend_literal:n { 0 ~ setlinejoin } }
1301 \cs_new_protected:Npn \__draw_backend_join_round:
1302 { \__draw_backend_literal:n { 1 ~ setlinejoin } }
1303 \cs_new_protected:Npn \__draw_backend_join_bevel:
1304 { \__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/X<sub>Y</sub>TeX). Thus we take the shortest path available and simply dump the matrix as given.

```

1305 \cs_new_protected:Npn \__draw_backend_cm:nnnn #1#2#3#4
1306 {

```

```

1307   \_draw_backend_literal:n
1308   { [ #1 ~ #2 ~ #3 ~ #4 ~ 0 ~ 0 ] ~ concat }
1309 }

```

(End definition for `\_draw_backend_cm:nmmn`.)

`\_draw_backend_box_use:Nmmmm`

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  $\text{\TeX}$  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).

```

1310 \cs_new_protected:Npn \_draw_backend_box_use:Nmmmm #1#2#3#4#5
1311 {
1312   \_draw_backend_literal:n { @endspecial }
1313   \_draw_backend_literal:n { [end] }
1314   \_draw_backend_literal:n { [begin] }
1315   \_draw_backend_literal:n { save }
1316   \_draw_backend_literal:n { currentpoint }
1317   \_draw_backend_literal:n { currentpoint~translate }
1318   \_draw_backend_cm:nmmn { 1 } { 0 } { 0 } { -1 }
1319   \_draw_backend_cm:nmmn {#2} {#3} {#4} {#5}
1320   \_draw_backend_cm:nmmn { 1 } { 0 } { 0 } { -1 }
1321   \_draw_backend_literal:n { neg~exch~neg~exch~translate }
1322   \_draw_backend_literal:n { [end] }
1323   \hbox_overlap_right:n { \box_use:N #1 }
1324   \_draw_backend_literal:n { [begin] }
1325   \_draw_backend_literal:n { restore }
1326   \_draw_backend_literal:n { [end] }
1327   \_draw_backend_literal:n { [begin] }
1328   \_draw_backend_literal:n { @beginspecial }
1329 }

```

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

1330 `</dvips>`

## 4.2 Lua $\text{\TeX}$ , pdf $\text{\TeX}$ , dvipdfmx and X $\text{\TeX}$

Lua $\text{\TeX}$ , pdf $\text{\TeX}$ , dvipdfmx and X $\text{\TeX}$  directly produce PDF output and understand a shared set of specials for drawing commands.

1331 `<*dvipdfmx | luatex | pdftex | xetex>`

### 4.2.1 Drawing

`\_draw_backend_literal:n`

Pass data through using a dedicated interface.

`\_draw_backend_literal:x`

```

1332 \cs_new_eq:NN \_draw_backend_literal:n \_kernel_backend_literal_pdf:n
1333 \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: 1334 \cs_new_protected:Npn \_draw_backend_begin:
1335   { \_draw_backend_scope_begin: }
1336 \cs_new_protected:Npn \_draw_backend_end:
1337   { \_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: 1338 \cs_new_eq:NN \_draw_backend_scope_begin: \_kernel_backend_scope_begin:
1339 \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 1340 \cs_new_protected:Npn \_draw_backend_moveto:nn #1#2
\__draw_backend_curveto:nnnnnn 1341   {
1342     \_draw_backend_literal:x
1343     { \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~ m }
1344   }
1345 \cs_new_protected:Npn \_draw_backend_lineto:nn #1#2
1346   {
1347     \_draw_backend_literal:x
1348     { \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~ l }
1349   }
1350 \cs_new_protected:Npn \_draw_backend_curveto:nnnnnn #1#2#3#4#5#6
1351   {
1352     \_draw_backend_literal:x
1353     {
1354       \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~
1355       \dim_to_decimal_in_bp:n {#3} ~ \dim_to_decimal_in_bp:n {#4} ~
1356       \dim_to_decimal_in_bp:n {#5} ~ \dim_to_decimal_in_bp:n {#6} ~
1357       c
1358     }
1359   }
1360 \cs_new_protected:Npn \_draw_backend_rectangle:nnnn #1#2#3#4
1361   {
1362     \_draw_backend_literal:x
1363     {
1364       \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~
1365       \dim_to_decimal_in_bp:n {#3} ~ \dim_to_decimal_in_bp:n {#4} ~
1366       re
1367     }
1368   }

```

(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: 1369 \cs_new_protected:Npn \_draw_backend_evenodd_rule:
\g__draw_draw_eor_bool 1370   { \bool_gset_true:N \g__draw_draw_eor_bool }
1371 \cs_new_protected:Npn \_draw_backend_nonzero_rule:
1372   { \bool_gset_false:N \g__draw_draw_eor_bool }
1373 \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:`.)

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

```

\__draw_backend_closepath:
  \__draw_backend_stroke:
\__draw_backend_closestroke:
  \__draw_backend_fill:
\__draw_backend_fillstroke:
  \__draw_backend_clip:
\__draw_backend_discardpath:
1374 \cs_new_protected:Npn \__draw_backend_closepath:
1375   { \__draw_backend_literal:n { h } }
1376 \cs_new_protected:Npn \__draw_backend_stroke:
1377   { \__draw_backend_literal:n { S } }
1378 \cs_new_protected:Npn \__draw_backend_closestroke:
1379   { \__draw_backend_literal:n { s } }
1380 \cs_new_protected:Npn \__draw_backend_fill:
1381   {
1382     \__draw_backend_literal:x
1383     { f \bool_if:NT \g__draw_draw_eor_bool * }
1384   }
1385 \cs_new_protected:Npn \__draw_backend_fillstroke:
1386   {
1387     \__draw_backend_literal:x
1388     { B \bool_if:NT \g__draw_draw_eor_bool * }
1389   }
1390 \cs_new_protected:Npn \__draw_backend_clip:
1391   {
1392     \__draw_backend_literal:x
1393     { W \bool_if:NT \g__draw_draw_eor_bool * }
1394   }
1395 \cs_new_protected:Npn \__draw_backend_discardpath:
1396   { \__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_butt:
\__draw_backend_cap_round:
  \__draw_backend_cap_rectangle:
\__draw_backend_join_miter:
\__draw_backend_join_round:
\__draw_backend_join_bevel:
1397 \cs_new_protected:Npn \__draw_backend_dash_pattern:nn #1#2
1398   {
1399     \__draw_backend_literal:x
1400     {
1401       [
1402         \exp_args:Nf \use:n
1403         { \clist_map_function:nN {#1} \__draw_backend_dash:n }
1404       ] ~
1405       \dim_to_decimal_in_bp:n {#2} ~ d
1406     }
1407   }
1408 \cs_new:Npn \__draw_backend_dash:n #1
1409   { ~ \dim_to_decimal_in_bp:n {#1} }
1410 \cs_new_protected:Npn \__draw_backend_linewidth:n #1
1411   {
1412     \__draw_backend_literal:x
1413     { \dim_to_decimal_in_bp:n {#1} ~ w }
1414   }
1415 \cs_new_protected:Npn \__draw_backend_miterlimit:n #1
1416   { \__draw_backend_literal:x { #1 ~ M } }
1417 \cs_new_protected:Npn \__draw_backend_cap_butt:
1418   { \__draw_backend_literal:n { 0 ~ J } }
1419 \cs_new_protected:Npn \__draw_backend_cap_round:

```

```

1420 { \_draw_backend_literal:n { 1 ~ J } }
1421 \cs_new_protected:Npn \_draw_backend_cap_rectangle:
1422 { \_draw_backend_literal:n { 2 ~ J } }
1423 \cs_new_protected:Npn \_draw_backend_join_miter:
1424 { \_draw_backend_literal:n { 0 ~ j } }
1425 \cs_new_protected:Npn \_draw_backend_join_round:
1426 { \_draw_backend_literal:n { 1 ~ j } }
1427 \cs_new_protected:Npn \_draw_backend_join_bevel:
1428 { \_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!

```

1429 \cs_new_protected:Npn \_draw_backend_cm:nnnn #1#2#3#4
1430 {
1431 <| luatex | pdftex >
1432 \_kernel_backend_matrix:n { #1 ~ #2 ~ #3 ~ #4 }
1433 </| luatex | pdftex >
1434 <| dvipdfmx | xetex >
1435 \_draw_backend_cm_decompose:nnnnN {#1} {#2} {#3} {#4}
1436 \_draw_backend_cm_aux:nnnn
1437 </| dvipdfmx | xetex >
1438 }
1439 <| dvipdfmx | xetex >
1440 \cs_new_protected:Npn \_draw_backend_cm_aux:nnnn #1#2#3#4
1441 {
1442 \_kernel_backend_literal:x
1443 {
1444 x:rotate~
1445 \fp_compare:nNnTF {#1} = \c_zero_fp
1446 { 0 }
1447 { \fp_eval:n { round ( -#1 , 5 ) } }
1448 }
1449 \_kernel_backend_literal:x
1450 {
1451 x:scale~
1452 \fp_eval:n { round ( #2 , 5 ) } ~
1453 \fp_eval:n { round ( #3 , 5 ) }
1454 }
1455 \_kernel_backend_literal:x
1456 {
1457 x:rotate~
1458 \fp_compare:nNnTF {#4} = \c_zero_fp
1459 { 0 }
1460 { \fp_eval:n { round ( -#4 , 5 ) } }
1461 }
1462 }
1463 </| dvipdfmx | xetex >

```

(End definition for `\_draw_backend_cm:nnnn` and `\_draw_backend_cm_aux:nnnn`.)

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.

```

1464 <*dvipdfmx | xetex>
1465 \cs_new_protected:Npn \_draw_backend_cm_decompose:nnnnN #1#2#3#4#5
1466 {
1467   \use:x
1468   {
1469     \_draw_backend_cm_decompose_auxi:nnnnN
1470     { \fp_eval:n { (#1 + #4) / 2 } }
1471     { \fp_eval:n { (#1 - #4) / 2 } }
1472     { \fp_eval:n { (#3 + #2) / 2 } }
1473     { \fp_eval:n { (#3 - #2) / 2 } }
1474   }
1475   #5
1476 }
1477 \cs_new_protected:Npn \_draw_backend_cm_decompose_auxi:nnnnN #1#2#3#4#5
1478 {
1479   \use:x
1480   {
1481     \_draw_backend_cm_decompose_auxii:nnnnN
1482     { \fp_eval:n { 2 * sqrt ( #1 * #1 + #4 * #4 ) } }
1483     { \fp_eval:n { 2 * sqrt ( #2 * #2 + #3 * #3 ) } }
1484     { \fp_eval:n { atand ( #3 , #2 ) } }
1485     { \fp_eval:n { atand ( #4 , #1 ) } }
1486   }
1487   #5

```



```

1488 }
1489 \cs_new_protected:Npn \__draw_backend_cm_decompose_auxii:nnnnN #1#2#3#4#5
1490 {
1491   \use:x
1492   {
1493     \__draw_backend_cm_decompose_auxiii:nnnnN
1494     { \fp_eval:n { ( #4 - #3 ) / 2 } }
1495     { \fp_eval:n { ( #1 + #2 ) / 2 } }
1496     { \fp_eval:n { ( #1 - #2 ) / 2 } }
1497     { \fp_eval:n { ( #4 + #3 ) / 2 } }
1498   }
1499   #5
1500 }
1501 \cs_new_protected:Npn \__draw_backend_cm_decompose_auxiii:nnnnN #1#2#3#4#5
1502 {
1503   \fp_compare:nNnTF { abs( #2 ) } > { abs ( #3 ) }
1504   { #5 {#1} {#2} {#3} {#4} }
1505   { #5 {#1} {#3} {#2} {#4} }
1506 }
1507 </dviPDFmx | xetex>

```

(End definition for \\_\_draw\_backend\_cm\_decompose:nnnnN and others.)

\\_\_draw\_backend\_box\_use:Nnnnn

Inserting a T<sub>E</sub>X box transformed to the requested position and using the current matrix is done using a mixture of T<sub>E</sub>X and low-level manipulation. The offset can be handled by T<sub>E</sub>X, 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.

```

1508 \cs_new_protected:Npn \__draw_backend_box_use:Nnnnn #1#2#3#4#5
1509 {
1510   \__kernel_backend_scope_begin:
1511   <*luatex | pdftex>
1512   \__draw_backend_cm:nnnn {#2} {#3} {#4} {#5}
1513   </luatex | pdftex>
1514   <*dviPDFmx | xetex>
1515   \__kernel_backend_literal:n
1516   { pdf:btrans-matrix~ #2 ~ #3 ~ #4 ~ #5 ~ 0 ~ 0 }
1517   </dviPDFmx | xetex>
1518   \hbox_overlap_right:n { \box_use:N #1 }
1519   <*dviPDFmx | xetex>
1520   \__kernel_backend_literal:n { pdf:etrans }
1521   </dviPDFmx | xetex>
1522   \__kernel_backend_scope_end:
1523 }

```

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

```

1524 </dviPDFmx | luatex | pdftex | xetex>

```

### 4.3 dvisvgm backend

```

1525 <*dvisvgm>

```

\\_\_draw\_backend\_literal:n  
\\_\_draw\_backend\_literal:x

The same as the more general literal call.

```

1526 \cs_new_eq:NN \__draw_backend_literal:n \__kernel_backend_literal_svg:n
1527 \cs_generate_variant:Nn \__draw_backend_literal:n { x }

```

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

`\_draw_backend_scope_begin:` Use the backend-level scope mechanisms.

`\_draw_backend_scope_end:` 1528 `\cs_new_eq:NN \_draw_backend_scope_begin: \_kernel_backend_scope_begin:`  
1529 `\cs_new_eq:NN \_draw_backend_scope_end: \_kernel_backend_scope_end:`

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

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

1530 `\cs_new_protected:Npn \_draw_backend_begin:`  
1531 `{`  
1532 `\_kernel_backend_scope_begin:`  
1533 `\_kernel_backend_scope:n { transform="translate({?x},{?y})~scale(1,-1)" }`  
1534 `}`  
1535 `\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  
`\_draw_backend_lineto:nn` values as they are given, the entire path needs to be collected up before being output  
`\_draw_backend_rectangle:nnnn` in one go. For that we use a dedicated storage routine, which adds spaces as required.  
`\_draw_backend_curveto:nnnnnn` Since paths should be fully expanded there is no need to worry about the internal x-type  
`\_draw_backend_add_to_path:n` expansion.  
`\g__draw_backend_path_tl`

1536 `\cs_new_protected:Npn \_draw_backend_moveto:nn #1#2`  
1537 `{`  
1538 `\_draw_backend_add_to_path:n`  
1539 `{ M ~ \dim_to_decimal:n {#1} ~ \dim_to_decimal:n {#2} }`  
1540 `}`  
1541 `\cs_new_protected:Npn \_draw_backend_lineto:nn #1#2`  
1542 `{`  
1543 `\_draw_backend_add_to_path:n`  
1544 `{ L ~ \dim_to_decimal:n {#1} ~ \dim_to_decimal:n {#2} }`  
1545 `}`  
1546 `\cs_new_protected:Npn \_draw_backend_rectangle:nnnn #1#2#3#4`  
1547 `{`  
1548 `\_draw_backend_add_to_path:n`  
1549 `{`  
1550 `M ~ \dim_to_decimal:n {#1} ~ \dim_to_decimal:n {#2}`  
1551 `h ~ \dim_to_decimal:n {#3} ~`  
1552 `v ~ \dim_to_decimal:n {#4} ~`  
1553 `h ~ \dim_to_decimal:n { -#3 } ~`  
1554 `Z`  
1555 `}`  
1556 `}`  
1557 `\cs_new_protected:Npn \_draw_backend_curveto:nnnnnn #1#2#3#4#5#6`  
1558 `{`  
1559 `\_draw_backend_add_to_path:n`  
1560 `{`  
1561 `C ~`  
1562 `\dim_to_decimal:n {#1} ~ \dim_to_decimal:n {#2} ~`  
1563 `\dim_to_decimal:n {#3} ~ \dim_to_decimal:n {#4} ~`  
1564 `\dim_to_decimal:n {#5} ~ \dim_to_decimal:n {#6}`  
1565 `}`  
`}`

```

1566 }
1567 \cs_new_protected:Npn \__draw_backend_add_to_path:n #1
1568 {
1569   \tl_gset:Nx \g__draw_backend_path_tl
1570   {
1571     \g__draw_backend_path_tl
1572     \tl_if_empty:NF \g__draw_backend_path_tl { \c_space_tl }
1573     #1
1574   }
1575 }
1576 \tl_new:N \g__draw_backend_path_tl

```

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

```

\__draw_backend_evenodd_rule: The fill rules here have to be handled as scopes.
\__draw_backend_nonzero_rule:
1577 \cs_new_protected:Npn \__draw_backend_evenodd_rule:
1578 { \__kernel_backend_scope:n { fill-rule="evenodd" } }
1579 \cs_new_protected:Npn \__draw_backend_nonzero_rule:
1580 { \__kernel_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 means setting up the various requirements in a shared auxiliary which deals with the bits and pieces. Clipping paths are reused for path drawing: not essential but avoids constructing them twice. Discarding a path needs a separate function as it's not quite the same.

```

\__draw_backend_closepath:
\__draw_backend_stroke:
\__draw_backend_closestroke:
\__draw_backend_fill:
\__draw_backend_fillstroke:
\__draw_backend_clip:
\__draw_backend_discardpath:
\g__draw_draw_clip_bool
\g__draw_draw_path_int
1581 \cs_new_protected:Npn \__draw_backend_closepath:
1582 { \__draw_backend_add_to_path:n { Z } }
1583 \cs_new_protected:Npn \__draw_backend_path:n #1
1584 {
1585   \bool_if:NTF \g__draw_draw_clip_bool
1586   {
1587     \int_gincr:N \g__kernel_clip_path_int
1588     \__draw_backend_literal:x
1589     {
1590       < clipPath~id = " l3cp \int_use:N \g__kernel_clip_path_int " >
1591       { ?nl }
1592       <path~d=" \g__draw_backend_path_tl "/> { ?nl }
1593       < /clipPath > { ? nl }
1594       <
1595       use~xlink:href =
1596       "\c_hash_str l3path \int_use:N \g__draw_backend_path_int " ~
1597       #1
1598       />
1599     }
1600     \__kernel_backend_scope:x
1601     {
1602       clip-path =
1603       "url( \c_hash_str l3cp \int_use:N \g__kernel_clip_path_int )"
1604     }
1605   }
1606   {
1607     \__draw_backend_literal:x

```

```

1608         { <path ~ d=" \g__draw_backend_path_tl " ~ #1 /> }
1609     }
1610     \tl_gclear:N \g__draw_backend_path_tl
1611     \bool_gset_false:N \g__draw_draw_clip_bool
1612 }
1613 \int_new:N \g__draw_backend_path_int
1614 \cs_new_protected:Npn \__draw_backend_stroke:
1615 { \__draw_backend_path:n { style="fill:none" } }
1616 \cs_new_protected:Npn \__draw_backend_closestroke:
1617 {
1618     \__draw_backend_closepath:
1619     \__draw_backend_stroke:
1620 }
1621 \cs_new_protected:Npn \__draw_backend_fill:
1622 { \__draw_backend_path:n { style="stroke:none" } }
1623 \cs_new_protected:Npn \__draw_backend_fillstroke:
1624 { \__draw_backend_path:n { } }
1625 \cs_new_protected:Npn \__draw_backend_clip:
1626 { \bool_gset_true:N \g__draw_draw_clip_bool }
1627 \bool_new:N \g__draw_draw_clip_bool
1628 \cs_new_protected:Npn \__draw_backend_discardpath:
1629 {
1630     \bool_if:NT \g__draw_draw_clip_bool
1631     {
1632         \int_gincr:N \g__kernel_clip_path_int
1633         \__draw_backend_literal:x
1634         {
1635             < clipPath~id = " l3cp \int_use:N \g__kernel_clip_path_int " >
1636             { ?nl }
1637             <path~d=" \g__draw_backend_path_tl "/> { ?nl }
1638             < /clipPath >
1639         }
1640         \__kernel_backend_scope:x
1641         {
1642             clip-path =
1643             "url( \c_hash_str l3cp \int_use:N \g__kernel_clip_path_int)"
1644         }
1645     }
1646     \tl_gclear:N \g__draw_path_tl
1647     \bool_gset_false:N \g__draw_draw_clip_bool
1648 }

```

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

<pre> \__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_but: \__draw_backend_cap_round: \__draw_backend_cap_rectangle: \__draw_backend_join_miter: \__draw_backend_join_round: \__draw_backend_join_bevel: </pre>	<p>All of these ideas are properties of scopes in SVG. The only slight complexity is converting the dash array properly (doing any required maths).</p> <pre> 1649 \cs_new_protected:Npn \__draw_backend_dash_pattern:nn #1#2 1650 { 1651     \use:x 1652     { 1653         \__draw_backend_dash_aux:nn 1654         { \clist_map_function:nn {#1} \__draw_backend_dash:n } 1655         { \dim_to_decimal:n {#2} } 1656     } </pre>
---	--

```

1657 }
1658 \cs_new:Npn \__draw_backend_dash:n #1
1659 { , \dim_to_decimal_in_bp:n {#1} }
1660 \cs_new_protected:Npn \__draw_backend_dash_aux:nn #1#2
1661 {
1662   \__kernel_backend_scope:x
1663   {
1664     stroke-dasharray =
1665     "
1666     \tl_if_empty:nTF {#1}
1667     { none }
1668     { \use_none:n #1 }
1669     " ~
1670     stroke-offset=" #2 "
1671   }
1672 }
1673 \cs_new_protected:Npn \__draw_backend_linewidth:n #1
1674 { \__kernel_backend_scope:x { stroke-width=" \dim_to_decimal:n {#1} " } }
1675 \cs_new_protected:Npn \__draw_backend_miterlimit:n #1
1676 { \__kernel_backend_scope:x { stroke-miterlimit=" #1 " } }
1677 \cs_new_protected:Npn \__draw_backend_cap_but:
1678 { \__kernel_backend_scope:n { stroke-linecap="butt" } }
1679 \cs_new_protected:Npn \__draw_backend_cap_round:
1680 { \__kernel_backend_scope:n { stroke-linecap="round" } }
1681 \cs_new_protected:Npn \__draw_backend_cap_rectangle:
1682 { \__kernel_backend_scope:n { stroke-linecap="square" } }
1683 \cs_new_protected:Npn \__draw_backend_join_miter:
1684 { \__kernel_backend_scope:n { stroke-linejoin="miter" } }
1685 \cs_new_protected:Npn \__draw_backend_join_round:
1686 { \__kernel_backend_scope:n { stroke-linejoin="round" } }
1687 \cs_new_protected:Npn \__draw_backend_join_bevel:
1688 { \__kernel_backend_scope:n { stroke-linejoin="bevel" } }

```

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

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

```

1689 \cs_new_protected:Npn \__draw_backend_cm:nnnn #1#2#3#4
1690 {
1691   \__kernel_backend_scope:n
1692   {
1693     transform =
1694     " matrix ( #1 , #2 , #3 , #4 , Opt , Opt ) "
1695   }
1696 }

```

(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.

```

1697 \cs_new_protected:Npn \__draw_backend_box_use:Nnnnn #1#2#3#4#5
1698 {
1699   \__kernel_backend_scope_begin:
1700   \__draw_backend_cm:nnnn {#2} {#3} {#4} {#5}

```

```

1701 \__kernel_backend_literal_svg:n
1702 {
1703   < g~
1704     stroke="none"~
1705     transform="scale(-1,1)~translate({?x},{?y})~scale(-1,-1)"
1706   >
1707 }
1708 \box_set_wd:Nn #1 { Opt }
1709 \box_set_ht:Nn #1 { Opt }
1710 \box_set_dp:Nn #1 { Opt }
1711 \box_use:N #1
1712 \__kernel_backend_literal_svg:n { </g> }
1713 \__kernel_backend_scope_end:
1714 }

```

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

```
1715 </dvisvgm>
```

```
1716 </package>
```

## 5 l3backend-graphics Implementation

```

1717 <*package>
1718 <@@=graphics>

```

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

```

1719 \cs_new_protected:Npn \__graphics_backend_loaded:n #1
1720 {
1721   \cs_if_exist:NTF \hook_gput_code:nnn
1722   {
1723     \hook_gput_code:nnn
1724     { file / l3graphics.sty / after }
1725     { backend }
1726     {#1}
1727   }
1728   {#1}
1729 }

```

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

### 5.1 dvips backend

```
1730 <*dvips>
```

\l\_graphics\_search\_ext\_seq

```

1731 \__graphics_backend_loaded:n
1732 { \seq_set_from_clist:Nn \l_graphics_search_ext_seq { .eps , .ps } }

```

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

\\_graphics\_backend\_getbb\_eps:n

Simply use the generic function.

\\_graphics\_backend\_getbb\_ps:n

```

1733 \__graphics_backend_loaded:n
1734 {
1735   \cs_new_eq:NN \__graphics_backend_getbb_eps:n \__graphics_read_bb:n
1736   \cs_new_eq:NN \__graphics_backend_getbb_ps:n \__graphics_read_bb:n
1737 }

```

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

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

```

\__graphics_backend_include_eps:n
\__graphics_backend_include_ps:n
1738 \cs_new_protected:Npn \__graphics_backend_include_eps:n #1
1739 {
1740   \__kernel_backend_literal:x
1741   {
1742     PSfile = #1 \c_space_tl
1743     llx = \dim_to_decimal_in_bp:n \l__graphics_llx_dim \c_space_tl
1744     lly = \dim_to_decimal_in_bp:n \l__graphics_lly_dim \c_space_tl
1745     urx = \dim_to_decimal_in_bp:n \l__graphics_urx_dim \c_space_tl
1746     ury = \dim_to_decimal_in_bp:n \l__graphics_ury_dim
1747   }
1748 }
1749 \cs_new_eq:NN \__graphics_backend_include_ps:n \__graphics_backend_include_eps:n

```

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

```

\_graphics_backend_get_pagecount:n
1750 \__graphics_backend_loaded:n
1751 { \cs_new_eq:NN \__graphics_backend_get_pagecount:n \__graphics_get_pagecount:n }

```

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

```

1752 </dvips>

```

## 5.2 LuaTeX and pdfTeX backends

```

1753 < *luatex | pdftex >

```

```

\l_graphics_search_ext_seq
1754 \__graphics_backend_loaded:n
1755 {
1756   \seq_set_from_clist:Nn
1757   \l_graphics_search_ext_seq
1758   { .pdf , .eps , .ps , .png , .jpg , .jpeg }
1759 }

```

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

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

```

1760 \tl_new:N \l__graphics_graphics_attr_tl

```

(End definition for `\l__graphics_graphics_attr_tl`.)

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

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

```

1761 \cs_new_protected:Npn \__graphics_backend_getbb_jpg:n #1

```

```

1762 {
1763   \int_zero:N \l__graphics_page_int
1764   \tl_clear:N \l__graphics_pagebox_tl
1765   \tl_set:Nx \l__graphics_graphics_attr_tl
1766   {
1767     \tl_if_empty:NF \l__graphics_decodearray_str
1768     { :D \l__graphics_decodearray_str }
1769     \bool_if:NT \l__graphics_interpolate_bool
1770     { :I }
1771   }
1772   \tl_clear:N \l__graphics_graphics_attr_tl
1773   \__graphics_backend_getbb_auxi:n {#1}
1774 }
1775 \cs_new_eq:NN \__graphics_backend_getbb_jpeg:n \__graphics_backend_getbb_jpg:n
1776 \cs_new_eq:NN \__graphics_backend_getbb_png:n \__graphics_backend_getbb_jpg:n
1777 \cs_new_protected:Npn \__graphics_backend_getbb_pdf:n #1
1778 {
1779   \tl_clear:N \l__graphics_decodearray_str
1780   \bool_set_false:N \l__graphics_interpolate_bool
1781   \tl_set:Nx \l__graphics_graphics_attr_tl
1782   {
1783     : \l__graphics_pagebox_tl
1784     \int_compare:nNnT \l__graphics_page_int > 1
1785     { :P \int_use:N \l__graphics_page_int }
1786   }
1787   \__graphics_backend_getbb_auxi:n {#1}
1788 }
1789 \cs_new_protected:Npn \__graphics_backend_getbb_auxii:n #1
1790 {
1791   \__graphics_bb_restore:xF { #1 \l__graphics_graphics_attr_tl }
1792   { \__graphics_backend_getbb_auxiii:n {#1} }
1793 }

```

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

```

1794 \cs_new_protected:Npn \__graphics_backend_getbb_auxiii:n #1
1795 {
1796   \exp_args:Ne \__graphics_backend_getbb_auxiiii:n
1797   { \__graphics_backend_dequote:w #1 " #1 " \s__graphics_stop }
1798   \int_const:cn { c__graphics_ #1 \l__graphics_graphics_attr_tl _int }
1799   { \tex_the:D \tex_pdflastximage:D }
1800   \__graphics_bb_save:x { #1 \l__graphics_graphics_attr_tl }
1801 }
1802 \cs_new_protected:Npn \__graphics_backend_getbb_auxiiii:n #1
1803 {
1804   \tex_immediate:D \tex_pdfximage:D
1805   \bool_lazy_or:nnT
1806   { \l__graphics_interpolate_bool }
1807   { ! \tl_if_empty_p:N \l__graphics_decodearray_str }
1808   {
1809     attr ~
1810     {

```



```

1811         \tl_if_empty:NF \l__graphics_decodearray_str
1812         { /Decode~[ \l__graphics_decodearray_str ] }
1813         \bool_if:NT \l__graphics_interpolate_bool
1814         { /Interpolate~true }
1815     }
1816 }
1817 \int_compare:nNnT \l__graphics_page_int > 0
1818 { page ~ \int_use:N \l__graphics_page_int }
1819 \tl_if_empty:NF \l__graphics_pagebox_tl
1820 { \l__graphics_pagebox_tl }
1821 {#1}
1822 \hbox_set:Nn \l__graphics_internal_box
1823 { \tex_pdfrefximage:D \tex_pdflastximage:D }
1824 \dim_set:Nn \l__graphics_urx_dim { \box_wd:N \l__graphics_internal_box }
1825 \dim_set:Nn \l__graphics_ury_dim { \box_ht:N \l__graphics_internal_box }
1826 }
1827 \cs_new:Npn \__graphics_backend_dequote:w #1 " #2 " #3 \s__graphics_stop {#2}

```

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

\\_\_graphics\_backend\_include\_jpg:n  
\\_\_graphics\_backend\_include\_jpeg:n  
\\_\_graphics\_backend\_include\_pdf:n  
\\_\_graphics\_backend\_include\_png:n

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

```

1828 \cs_new_protected:Npn \__graphics_backend_include_jpg:n #1
1829 {
1830     \tex_pdfrefximage:D
1831     \int_use:c { c__graphics_ #1 \l__graphics_graphics_attr_tl _int }
1832 }
1833 \cs_new_eq:NN \__graphics_backend_include_jpeg:n \__graphics_backend_include_jpg:n
1834 \cs_new_eq:NN \__graphics_backend_include_pdf:n \__graphics_backend_include_jpg:n
1835 \cs_new_eq:NN \__graphics_backend_include_png:n \__graphics_backend_include_jpg:n

```

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

\\_\_graphics\_backend\_getbb\_eps:n  
\\_\_graphics\_backend\_getbb\_ps:n  
\\_\_graphics\_backend\_getbb\_eps:nm  
\\_\_graphics\_backend\_include\_eps:n  
\\_\_graphics\_backend\_include\_ps:n  
\l\_\_graphics\_backend\_dir\_str  
\l\_\_graphics\_backend\_name\_str  
\l\_\_graphics\_backend\_ext\_str

EPS graphics may be included in LuaTeX/pdfTeX by conversion to PDF: this requires restricted shell escape. Modelled on the `epstopdf` L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> package, but simplified, conversion takes place here if we have shell access.

```

1836 \sys_if_shell:T
1837 {
1838     \str_new:N \l__graphics_backend_dir_str
1839     \str_new:N \l__graphics_backend_name_str
1840     \str_new:N \l__graphics_backend_ext_str
1841     \cs_new_protected:Npn \__graphics_backend_getbb_eps:n #1
1842     {
1843         \file_parse_full_name:nNNN {#1}
1844         \l__graphics_backend_dir_str
1845         \l__graphics_backend_name_str
1846         \l__graphics_backend_ext_str
1847         \exp_args:Nx \__graphics_backend_getbb_eps:nm
1848         {
1849             \exp_args:Ne \__kernel_file_name_quote:n
1850             {
1851                 \l__graphics_backend_name_str
1852                 - \str_tail:N \l__graphics_backend_ext_str

```

```

1853         -converted-to.pdf
1854     }
1855 }
1856 {#1}
1857 }
1858 \cs_new_eq:NN \__graphics_backend_getbb_ps:n \__graphics_backend_getbb_eps:n
1859 \cs_new_protected:Npn \__graphics_backend_getbb_eps:nn #1#2
1860 {
1861     \file_compare_timestamp:nNnT {#2} > {#1}
1862     {
1863         \sys_shell_now:n
1864         { repstopdf ~ #2 ~ #1 }
1865     }
1866     \tl_set:Nn \l__graphics_final_name_str {#1}
1867     \__graphics_backend_getbb_pdf:n {#1}
1868 }
1869 \cs_new_protected:Npn \__graphics_backend_include_eps:n #1
1870 {
1871     \file_parse_full_name:nNNN {#1}
1872     \l__graphics_backend_dir_str \l__graphics_backend_name_str \l__graphics_backend_ext_str
1873     \exp_args:Nx \__graphics_backend_include_pdf:n
1874     {
1875         \exp_args:Ne \__kernel_file_name_quote:n
1876         {
1877             \l__graphics_backend_name_str
1878             - \str_tail:N \l__graphics_backend_ext_str
1879             -converted-to.pdf
1880         }
1881     }
1882 }
1883 \cs_new_eq:NN \__graphics_backend_include_ps:n \__graphics_backend_include_eps:n
1884 }

```

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

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

```

1885 \cs_new_protected:Npn \__graphics_backend_get_pagecount:n #1
1886 {
1887     \tex_pdfximage:D {#1}
1888     \int_const:cn { c__graphics_ #1 _pages_int }
1889     { \int_use:N \tex_pdflastximagepages:D }
1890 }

```

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

1891 </luatex | pdftex>

### 5.3 dvipdfmx backend

1892 <\*dvipdfmx | xetex>

\l\_graphics\_search\_ext\_seq

```

1893 \__graphics_backend_loaded:n
1894 {
1895     \seq_set_from_clist:Nn \l_graphics_search_ext_seq

```

```

1896     { .pdf , .eps , .ps , .png , .jpg ., jpeg , .bmp }
1897 }

```

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

```

\graphics_backend_getbb_eps:n Simply use the generic functions: only for dvipdfmx in the extraction cases.
\graphics_backend_getbb_ps:n 1898 \__graphics_backend_loaded:n
\graphics_backend_getbb_jpg:n 1899 {
\graphics_backend_getbb_jpeg:n 1900     \cs_new_eq:NN \__graphics_backend_getbb_eps:n \__graphics_read_bb:n
\graphics_backend_getbb_pdf:n 1901     \cs_new_eq:NN \__graphics_backend_getbb_ps:n \__graphics_read_bb:n
\graphics_backend_getbb_png:n 1902 }
\graphics_backend_getbb_bmp:n 1903 <*dvipdfmx>
1904 \cs_new_protected:Npn \__graphics_backend_getbb_jpg:n #1
1905 {
1906     \int_zero:N \l__graphics_page_int
1907     \tl_clear:N \l__graphics_pagebox_tl
1908     \__graphics_extract_bb:n {#1}
1909 }
1910 \cs_new_eq:NN \__graphics_backend_getbb_jpeg:n \__graphics_backend_getbb_jpg:n
1911 \cs_new_eq:NN \__graphics_backend_getbb_png:n \__graphics_backend_getbb_jpg:n
1912 \cs_new_eq:NN \__graphics_backend_getbb_bmp:n \__graphics_backend_getbb_jpg:n
1913 \cs_new_protected:Npn \__graphics_backend_getbb_pdf:n #1
1914 {
1915     \tl_clear:N \l__graphics_decodearray_str
1916     \bool_set_false:N \l__graphics_interpolate_bool
1917     \__graphics_extract_bb:n {#1}
1918 }
1919 </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.

```

1920 \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
\graphics_backend_include_ps:n are best handled between dvipdfmx and XeTeX: for the latter it is better to use the
\graphics_backend_include_jpg:n primitive route. The relevant code for that is included later in this file.
\graphics_backend_include_jpeg:n 1921 \cs_new_protected:Npn \graphics_backend_include_eps:n #1
\graphics_backend_include_pdf:n 1922 {
\graphics_backend_include_png:n 1923     \__kernel_backend_literal:x
\graphics_backend_include_bmp:n 1924     {
\graphics_backend_include_auxi:nn 1925         PSfile = #1 \c_space_tl
\graphics_backend_include_auxii:nnn 1926         llx = \dim_to_decimal_in_bp:n \l__graphics_llx_dim \c_space_tl
\graphics_backend_include_auxiii:nnn 1927         lly = \dim_to_decimal_in_bp:n \l__graphics_lly_dim \c_space_tl
1928         urx = \dim_to_decimal_in_bp:n \l__graphics_urx_dim \c_space_tl
1929         ury = \dim_to_decimal_in_bp:n \l__graphics_ury_dim
1930     }
1931 }
1932 \cs_new_eq:NN \graphics_backend_include_ps:n \graphics_backend_include_eps:n
1933 \cs_new_protected:Npn \graphics_backend_include_jpg:n #1
1934 { \__graphics_backend_include_auxi:nn {#1} { image } }
1935 \cs_new_eq:NN \graphics_backend_include_jpeg:n \graphics_backend_include_jpg:n
1936 \cs_new_eq:NN \graphics_backend_include_png:n \graphics_backend_include_jpg:n

```

```

1937 \cs_new_eq:NN \__graphics_backend_include_bmp:n \__graphics_backend_include_jpg:n
1938 <*dvipdfmx>
1939 \cs_new_protected:Npn \__graphics_backend_include_pdf:n #1
1940 { \__graphics_backend_include_auxi:nn {#1} { epdf } }
1941 </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.

```

1942 \cs_new_protected:Npn \__graphics_backend_include_auxi:nn #1#2
1943 {
1944   \__graphics_backend_include_auxii:nnn
1945   {
1946     \tl_if_empty:NF \l__graphics_pagebox_tl
1947     { : \l__graphics_pagebox_tl }
1948     \int_compare:nNnT \l__graphics_page_int > 1
1949     { :P \int_use:N \l__graphics_page_int }
1950     \tl_if_empty:NF \l__graphics_decodearray_str
1951     { :D \l__graphics_decodearray_str }
1952     \bool_if:NT \l__graphics_interpolate_bool
1953     { :I }
1954   }
1955   {#1} {#2}
1956 }
1957 \cs_new_protected:Npn \__graphics_backend_include_auxii:nnn #1#2#3
1958 {
1959   \int_if_exist:cTF { c__graphics_ #2#1 _int }
1960   {
1961     \__kernel_backend_literal:x
1962     { pdf:usexobj~@graphic \int_use:c { c__graphics_ #2#1 _int } }
1963   }
1964   { \__graphics_backend_include_auxiii:nnn {#2} {#1} {#3} }
1965 }
1966 \cs_generate_variant:Nn \__graphics_backend_include_auxii:nnn { x }

```

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

```

1967 \cs_new_protected:Npn \__graphics_backend_include_auxiii:nnn #1#2#3
1968 {
1969   \int_gincr:N \g__graphics_track_int
1970   \int_const:cn { c__graphics_ #1#2 _int } { \g__graphics_track_int }
1971   \__kernel_backend_literal:x
1972   {
1973     pdf:#3~
1974     @graphic \int_use:c { c__graphics_ #1#2 _int } ~
1975     \int_compare:nNnT \l__graphics_page_int > 1
1976     { page ~ \int_use:N \l__graphics_page_int \c_space_tl }
1977     \tl_if_empty:NF \l__graphics_pagebox_tl
1978     {
1979       pagebox ~ \l__graphics_pagebox_tl \c_space_tl
1980       bbox ~
1981       \dim_to_decimal_in_bp:n \l__graphics_llx_dim \c_space_tl

```

```

1982         \dim_to_decimal_in_bp:n \l__graphics_lly_dim \c_space_tl
1983         \dim_to_decimal_in_bp:n \l__graphics_urx_dim \c_space_tl
1984         \dim_to_decimal_in_bp:n \l__graphics_ury_dim \c_space_tl
1985     }
1986     (#1)
1987     \bool_lazy_or:nnT
1988     { \l__graphics_interpolate_bool }
1989     { ! \tl_if_empty_p:N \l__graphics_decodearray_str }
1990     {
1991         <<
1992         \tl_if_empty:NF \l__graphics_decodearray_str
1993         { /Decode~[ \l__graphics_decodearray_str ] }
1994         \bool_if:NT \l__graphics_interpolate_bool
1995         { /Interpolate~true> }
1996         >>
1997     }
1998 }
1999 }

```

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

`\__graphics_backend_get_pagecount:n`

```

2000 < *dvipdfmx >
2001 \__graphics_backend_loaded:n
2002 { \cs_new_eq:NN \__graphics_backend_get_pagecount:n \__graphics_get_pagecount:n }
2003 < /dvipdfmx >

```

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

```

2004 < /dvipdfmx | xetex >

```

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

```

2005 < *xetex >

```

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_jpeg:n
\__graphics_backend_getbb_pdf:n
\__graphics_backend_getbb_png:n
\__graphics_backend_getbb_bmp:n
\__graphics_backend_getbb_auxi:nN
\__graphics_backend_getbb_auxii:nnN
\__graphics_backend_getbb_auxiii:VnN
\__graphics_backend_getbb_auxiiii:nnNn
\__graphics_backend_getbb_auxiv:nnNnn
\__graphics_backend_getbb_auxiv:VnNnn
\__graphics_backend_getbb_auxv:nnNnn
\__graphics_backend_getbb_auxv:nnNnn
\__graphics_backend_getbb_pagebox:w
2006 \cs_new_protected:Npn \__graphics_backend_getbb_jpg:n #1
2007 {
2008     \int_zero:N \l__graphics_page_int
2009     \tl_clear:N \l__graphics_pagebox_tl
2010     \__graphics_backend_getbb_auxi:nN {#1} \tex_XeTeXpicfile:D
2011 }
2012 \cs_new_eq:NN \__graphics_backend_getbb_jpeg:n \__graphics_backend_getbb_jpg:n
2013 \cs_new_eq:NN \__graphics_backend_getbb_png:n \__graphics_backend_getbb_jpg:n
2014 \cs_new_eq:NN \__graphics_backend_getbb_bmp:n \__graphics_backend_getbb_jpg:n
2015 \cs_new_protected:Npn \__graphics_backend_getbb_pdf:n #1
2016 {
2017     \tl_clear:N \l__graphics_decodearray_str
2018     \bool_set_false:N \l__graphics_interpolate_bool
2019     \__graphics_backend_getbb_auxi:nN {#1} \tex_XeTeXpdfvfile:D
2020 }
2021 \cs_new_protected:Npn \__graphics_backend_getbb_auxi:nN #1#2

```

```

2022 {
2023   \int_compare:nNnTF \l__graphics_page_int > 1
2024     { \__graphics_backend_getbb_auxii:VnN \l__graphics_page_int {#1} #2 }
2025     { \__graphics_backend_getbb_auxiii:nNnn {#1} #2 { :P 1 } { page 1 } }
2026   }
2027   \cs_new_protected:Npn \__graphics_backend_getbb_auxii:nnN #1#2#3
2028     { \__graphics_backend_getbb_auxiii:nNnn {#2} #3 { :P #1 } { page #1 } }
2029   \cs_generate_variant:Nn \__graphics_backend_getbb_auxii:nnN { V }
2030   \cs_new_protected:Npn \__graphics_backend_getbb_auxiii:nNnn #1#2#3#4
2031     {
2032       \tl_if_empty:NTF \l__graphics_pagebox_tl
2033         { \__graphics_backend_getbb_auxiv:VnNnn \l__graphics_pagebox_tl }
2034         { \__graphics_backend_getbb_auxv:nNnn }
2035       {#1} #2 {#3} {#4}
2036     }
2037   \cs_new_protected:Npn \__graphics_backend_getbb_auxiv:nnNnn #1#2#3#4#5
2038     {
2039       \use:x
2040       {
2041         \__graphics_backend_getbb_auxv:nNnn {#2} #3 { : #1 #4 }
2042         {
2043           #5
2044           \tl_if_blank:nF {#1}
2045             { \c_space_tl \__graphics_backend_getbb_pagebox:w #1 }
2046         }
2047       }
2048     }
2049   \cs_generate_variant:Nn \__graphics_backend_getbb_auxiv:nnNnn { V }
2050   \cs_new_protected:Npn \__graphics_backend_getbb_auxv:nNnn #1#2#3#4
2051     {
2052       \__graphics_bb_restore:nF {#1#3}
2053       { \__graphics_backend_getbb_auxvi:nNnn {#1} #2 {#3} {#4} }
2054     }
2055   \cs_new_protected:Npn \__graphics_backend_getbb_auxvi:nNnn #1#2#3#4
2056     {
2057       \hbox_set:Nn \l__graphics_internal_box { #2 #1 ~ #4 }
2058       \dim_set:Nn \l__graphics_urx_dim { \box_wd:N \l__graphics_internal_box }
2059       \dim_set:Nn \l__graphics_ury_dim { \box_ht:N \l__graphics_internal_box }
2060       \__graphics_bb_save:n {#1#3}
2061     }
2062   \cs_new:Npn \__graphics_backend_getbb_pagebox:w #1 box {#1}

```

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

\\_\_graphics\_backend\_include\_pdf:n For PDF graphics, properly supporting the pagebox concept in X<sub>Y</sub>TeX is best done using the \tex\_XeTeXpdf\_file: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.

```

2063   \cs_new_protected:Npn \__graphics_backend_include_pdf:n #1
2064     {
2065       \tex_XeTeXpdf_file:D #1 ~
2066       \int_compare:nNnTF \l__graphics_page_int > 0
2067         { page ~ \int_use:N \l__graphics_page_int \c_space_tl }
2068         \exp_after:wN \__graphics_backend_getbb_pagebox:w \l__graphics_pagebox_tl

```

```
2069 }
(End definition for \_graphics_backend_include_pdf:n.)
```

```
\_graphics_backend_get_pagecount:n Very little to do here other than cover the case of a non-PDF file.
2070 \cs_new_protected:Npn \_graphics_backend_get_pagecount:n #1
2071 {
2072   \int_const:cn { c__graphics_ #1 _pages_int }
2073   {
2074     \int_max:nn
2075       { \int_use:N \tex_XeTeXpdfpagecount:D #1 ~ }
2076       { 1 }
2077   }
2078 }
(End definition for \_graphics_backend_get_pagecount:n.)
2079 </xetex>
```

## 5.5 dvisvgm backend

```
2080 <*dvisvgm>
\l_graphics_search_ext_seq
2081 \_graphics_backend_loaded:n
2082 {
2083   \seq_set_from_clist:Nn
2084     \l_graphics_search_ext_seq
2085     { .svg , .pdf , .eps , .ps , .png , .jpg , .jpeg }
2086 }
(End definition for \l_graphics_search_ext_seq. This variable is documented on page ??.)
```

```
\_graphics_backend_getbb_svg:n This is relatively similar to reading bounding boxes for .eps files. Life is though made
\_graphics_backend_getbb_svg_auxi:nNn more tricky as we cannot pick a single line for the data. So we have to loop until we
\_graphics_backend_getbb_svg_auxii:w collect up both height and width. To do that, we can use a marker value. We also have
\_graphics_backend_getbb_svg_auxiii:Nw to allow for the default units of the lengths: they are big points and may be omitted.
\_graphics_backend_getbb_svg_auxiv:Nw
\_graphics_backend_getbb_svg_auxv:Nw
\_graphics_backend_getbb_svg_auxvi:Nn
\_graphics_backend_getbb_svg_auxvii:w
2087 \cs_new_protected:Npn \_graphics_backend_getbb_svg:n #1
2088 {
2089   \_graphics_bb_restore:nF {#1}
2090   {
2091     \ior_open:Nn \l__graphics_internal_ior {#1}
2092     \ior_if_eof:NTF \l__graphics_internal_ior
2093       { \msg_error:nnn { graphics } { graphic-not-found } {#1} }
2094     {
2095       \dim_zero:N \l__graphics_llx_dim
2096       \dim_zero:N \l__graphics_lly_dim
2097       \dim_set:Nn \l__graphics_urx_dim { -\c_max_dim }
2098       \dim_set:Nn \l__graphics_ury_dim { -\c_max_dim }
2099       \ior_str_map_inline:Nn \l__graphics_internal_ior
2100         {
2101           \dim_compare:nNnT \l__graphics_urx_dim = { -\c_max_dim }
2102             {
2103               \_graphics_backend_getbb_svg_auxi:nNn
2104                 { width } \l__graphics_urx_dim {##1}
```

```

2105     }
2106     \dim_compare:nNnT \l__graphics_ury_dim = { -\c_max_dim }
2107     {
2108         \__graphics_backend_getbb_svg_auxi:nNn
2109         { height } \l__graphics_ury_dim {##1}
2110     }
2111     \bool_lazy_and:nnF
2112     { \dim_compare_p:nNn \l__graphics_urx_dim = { -\c_max_dim } }
2113     { \dim_compare_p:nNn \l__graphics_ury_dim = { -\c_max_dim } }
2114     { \ior_map_break: }
2115 }
2116 \__graphics_bb_save:n {#1}
2117 }
2118 \ior_close:N \l__graphics_internal_ior
2119 }
2120 }
2121 \cs_new_protected:Npn \__graphics_backend_getbb_svg_auxi:nNn #1#2#3
2122 {
2123     \use:x
2124     {
2125         \cs_set_protected:Npn \__graphics_backend_getbb_svg_auxii:w
2126         ###1 \tl_to_str:n {#1} = ###2 \tl_to_str:n {#1} = ###3
2127         \s__graphics_stop
2128     }
2129     {
2130         \tl_if_blank:nF {##2}
2131         {
2132             \peek_remove_spaces:n
2133             {
2134                 \peek_meaning:NTF ' % '
2135                 { \__graphics_backend_getbb_svg_auxiii:Nw #2 }
2136                 {
2137                     \peek_meaning:NTF " % "
2138                     { \__graphics_backend_getbb_svg_auxiv:Nw #2 }
2139                     { \__graphics_backend_getbb_svg_auxv:Nw #2 }
2140                 }
2141             }
2142             ##2 \s__graphics_stop
2143         }
2144     }
2145     \use:x
2146     {
2147         \__graphics_backend_getbb_svg_auxii:w #3
2148         \tl_to_str:n {#1} = \tl_to_str:n {#1} =
2149         \s__graphics_stop
2150     }
2151 }
2152 \cs_new_protected:Npn \__graphics_backend_getbb_svg_auxii:w { }
2153 \cs_new_protected:Npn \__graphics_backend_getbb_svg_auxiii:Nw #1 ' #2 ' #3 \s__graphics_stop
2154 { \__graphics_backend_getbb_svg_auxvi:Nn #1 {#2} }
2155 \cs_new_protected:Npn \__graphics_backend_getbb_svg_auxiv:Nw #1 " #2 " #3 \s__graphics_stop
2156 { \__graphics_backend_getbb_svg_auxvi:Nn #1 {#2} }
2157 \cs_new_protected:Npn \__graphics_backend_getbb_svg_auxv:Nw #1 #2 ~ #3 \s__graphics_stop
2158 { \__graphics_backend_getbb_svg_auxvi:Nn #1 {#2} }

```



```

2159 \cs_new_protected:Npn \__graphics_backend_getbb_svg_auxvi:Nn #1#2
2160 {
2161   \tex_afterassignment:D \__graphics_backend_getbb_svg_auxvii:w
2162   \l__graphics_internal_dim #2 bp \scan_stop:
2163   \dim_set_eq:NN #1 \l__graphics_internal_dim
2164 }
2165 \cs_new_protected:Npn \__graphics_backend_getbb_svg_auxvii:w #1 \scan_stop: { }

```

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

\\_\_graphics\_backend\_getbb\_eps:n Simply use the generic function.

```

\__graphics_backend_getbb_ps:n
2166 \__graphics_backend_loaded:n
2167 {
2168   \cs_new_eq:NN \__graphics_backend_getbb_eps:n \__graphics_read_bb:n
2169   \cs_new_eq:NN \__graphics_backend_getbb_ps:n \__graphics_read_bb:n
2170 }

```

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

\\_\_graphics\_backend\_getbb\_png:n These can be included by extracting the bounding box data.

```

\__graphics_backend_getbb_jpg:n
\__graphics_backend_getbb_jpeg:n
2171 \cs_new_protected:Npn \__graphics_backend_getbb_jpg:n #1
2172 {
2173   \int_zero:N \l__graphics_page_int
2174   \tl_clear:N \l__graphics_pagebox_tl
2175   \__graphics_extract_bb:n {#1}
2176 }
2177 \cs_new_eq:NN \__graphics_backend_getbb_jpeg:n \__graphics_backend_getbb_jpg:n
2178 \cs_new_eq:NN \__graphics_backend_getbb_png:n \__graphics_backend_getbb_jpg:n

```

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

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

```

2179 \cs_new_protected:Npn \__graphics_backend_getbb_pdf:n #1
2180 {
2181   \tl_clear:N \l__graphics_decodearray_str
2182   \bool_set_false:N \l__graphics_interpolate_bool
2183   \__graphics_extract_bb:n {#1}
2184 }

```

(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_ps:n
\__graphics_backend_include_pdf:n
\__graphics_backend_include_nn
2185 \cs_new_protected:Npn \__graphics_backend_include_eps:n #1
2186 { \__graphics_backend_include:nn { PSfile } {#1} }
2187 \cs_new_eq:NN \__graphics_backend_include_ps:n \__graphics_backend_include_eps:n
2188 \cs_new_protected:Npn \__graphics_backend_include_pdf:n #1
2189 { \__graphics_backend_include:nn { pdffile } {#1} }
2190 \cs_new_protected:Npn \__graphics_backend_include:nn #1#2
2191 {
2192   \__kernel_backend_literal:x
2193   {
2194     #1 = #2 \c_space_tl
2195     llx = \dim_to_decimal_in_bp:n \l__graphics_llx_dim \c_space_tl

```

```

2196         lly = \dim_to_decimal_in_bp:n \l__graphics_lly_dim \c_space_tl
2197         urx = \dim_to_decimal_in_bp:n \l__graphics_urx_dim \c_space_tl
2198         ury = \dim_to_decimal_in_bp:n \l__graphics_ury_dim
2199     }
2200 }

```

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

```

\__graphics_backend_include_svg:n
\__graphics_backend_include_png:n
\__graphics_backend_include_jpg:n
\__graphics_backend_include_jpeg:n
\__graphics_backend_include_dequote:w

```

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

```

2201 \cs_new_protected:Npn \__graphics_backend_include_svg:n #1
2202 {
2203     \box_move_up:nn { \l__graphics_ury_dim }
2204     {
2205         \hbox:n
2206         {
2207             \__kernel_backend_literal:x
2208             {
2209                 dvisvgm:img~
2210                 \dim_to_decimal:n { \l__graphics_urx_dim } ~
2211                 \dim_to_decimal:n { \l__graphics_ury_dim } ~
2212                 \__graphics_backend_include_dequote:w #1 " #1 " \s__graphics_stop
2213             }
2214         }
2215     }
2216 }
2217 \cs_new_eq:NN \__graphics_backend_include_png:n \__graphics_backend_include_svg:n
2218 \cs_new_eq:NN \__graphics_backend_include_jpeg:n \__graphics_backend_include_svg:n
2219 \cs_new_eq:NN \__graphics_backend_include_jpg:n \__graphics_backend_include_svg:n
2220 \cs_new:Npn \__graphics_backend_include_dequote:w #1 " #2 " #3 \s__graphics_stop
2221 {#2}

```

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

```

\__graphics_backend_get_pagecount:n

```

```

2222 \__graphics_backend_loaded:n
2223 { \cs_new_eq:NN \__graphics_backend_get_pagecount:n \__graphics_get_pagecount:n }

```

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

```

2224 </dvisvgm>
2225 </package>

```

## 6 l3backend-pdf Implementation

```

2226 <*package>
2227 <@@=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
2228 \box_new:N \l__pdf_internal_box
(End definition for \l__pdf_internal_box.)
```

## 6.2 dvips backend

2229 *⟨\*dvips⟩*

```
\__pdf_backend_pdfmark:n Used often enough it should be a separate function.
\__pdf_backend_pdfmark:x 2230 \cs_new_protected:Npn \__pdf_backend_pdfmark:n #1
2231 { \__kernel_backend_postscript:n { mark #1 ~ pdfmark } }
2232 \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 2233 \cs_new_protected:Npn \_pdf_backend_catalog_gput:nn #1#2
2234 { \__pdf_backend_pdfmark:n { { Catalog } << /#1 ~ #2 >> /PUT } }
2235 \cs_new_protected:Npn \__pdf_backend_info_gput:nn #1#2
2236 { \__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 2237 \int_new:N \g__pdf_backend_object_int
2238 \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:n 2239 \cs_new_protected:Npn \_pdf_backend_object_new:nn #1#2
2240 {
2241   \int_gincr:N \g__pdf_backend_object_int
2242   \int_const:cn
2243     { c__pdf_backend_object_ \tl_to_str:n {#1} _int }
2244     { \g__pdf_backend_object_int }
2245   \prop_gput:Nnn \g__pdf_backend_object_prop {#1} {#2}
2246 }
2247 \cs_new:Npn \__pdf_backend_object_ref:n #1
2248 { { 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
\__pdf_backend_object_write:nx
\__pdf_backend_object_write_array:nn
\__pdf_backend_object_write_dict:nn
\__pdf_backend_object_write_fstream:nn
\__pdf_backend_object_write_stream:nn
\__pdf_backend_object_write_stream:nnn

```

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

```

2249 \cs_new_protected:Npn \__pdf_backend_object_write:nn #1#2
2250 {
2251   \__pdf_backend_pdfmark:x
2252   {
2253     /objdef ~ \__pdf_backend_object_ref:n {#1}
2254     /type
2255     \str_case_e:nn
2256       { \prop_item:Nn \g__pdf_backend_object_prop {#1} }
2257       {
2258         { array } { /array }
2259         { dict } { /dict }
2260         { fstream } { /stream }
2261         { stream } { /stream }
2262       }
2263     /OBJ
2264   }
2265   \use:c
2266     { \__pdf_backend_object_write_ \prop_item:Nn \g__pdf_backend_object_prop {#1} :nn }
2267     { \__pdf_backend_object_ref:n {#1} } {#2}
2268 }
2269 \cs_generate_variant:Nn \__pdf_backend_object_write:nn { nx }
2270 \cs_new_protected:Npn \__pdf_backend_object_write_array:nn #1#2
2271 {
2272   \__pdf_backend_pdfmark:x
2273   { #1 ~0~ [ ~ \exp_not:n {#2} ~ ] ~ /PUTINTERVAL }
2274 }
2275 \cs_new_protected:Npn \__pdf_backend_object_write_dict:nn #1#2
2276 {
2277   \__pdf_backend_pdfmark:x
2278   { #1 << \exp_not:n {#2} >> /PUT }
2279 }
2280 \cs_new_protected:Npn \__pdf_backend_object_write_fstream:nn #1#2
2281 {
2282   \exp_args:Nx
2283     \__pdf_backend_object_write_fstream:nnn {#1} #2
2284 }
2285 \cs_new_protected:Npn \__pdf_backend_object_write_fstream:nnn #1#2#3
2286 {
2287   \__kernel_backend_postscript:n
2288   {
2289     SDict ~ begin ~
2290     mark ~ #1 ~ << #2 >> /PUT ~ pdfmark ~
2291     mark ~ #1 ~ ( #3 )~ ( r )~ file ~ /PUT ~ pdfmark ~
2292     end
2293   }
2294 }
2295 \cs_new_protected:Npn \__pdf_backend_object_write_stream:nn #1#2
2296 {
2297   \exp_args:Nx
2298     \__pdf_backend_object_write_stream:nnn {#1} #2
2299 }
2300 \cs_new_protected:Npn \__pdf_backend_object_write_stream:nnn #1#2#3
2301 {

```

```

2302 \__kernel_backend_postscript:n
2303 {
2304     mark ~ #1 ~ ( #3 ) /PUT ~ pdfmark ~
2305     mark ~ #1 ~ << #2 >> /PUT ~ pdfmark
2306 }
2307 }

```

(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
2308 \cs_new_protected:Npn \__pdf_backend_object_now:nn #1#2
2309 {
2310     \int_gincr:N \g__pdf_backend_object_int
2311     \__pdf_backend_pdfmark:x
2312     {
2313         /objdef ~ { pdf.obj \int_use:N \g__pdf_backend_object_int }
2314         /type
2315         \str_case:nn
2316             {#1}
2317             {
2318                 { array } { /array }
2319                 { dict } { /dict }
2320                 { fstream } { /stream }
2321                 { stream } { /stream }
2322             }
2323         /OBJ
2324     }
2325     \exp_args:Nnx \use:c { __pdf_backend_object_write_ #1 :nn }
2326     { { pdf.obj \int_use:N \g__pdf_backend_object_int } } {#2}
2327 }
2328 \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.

```

2329 \cs_new:Npn \__pdf_backend_object_last:
2330 { { 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.

```

2331 \cs_new:Npn \__pdf_backend_pageobject_ref:n #1
2332 { { 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.

```

2333 \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.

```

2334 \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.

```

2335 \int_new:N \g_pdf_backend_annotation_int
(End definition for \g_pdf_backend_annotation_int.)

```

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

```

2336 \cs_new_protected:Npn \__pdf_backend_annotation:nnnn #1#2#3#4
2337 {
2338   \exp_args:Nf \__pdf_backend_annotation_aux:nnnn
2339   { \dim_eval:n {#1} } {#2} {#3} {#4}
2340 }
2341 \cs_new_protected:Npn \__pdf_backend_annotation_aux:nnnn #1#2#3#4
2342 {
2343   \box_move_down:nn {#3}
2344   { \hbox:n { \__kernel_backend_postscript:n { pdf.save.ll } } }
2345   \box_move_up:nn {#2}
2346   {
2347     \hbox:n
2348     {
2349       \__kernel_kern:n {#1}
2350       \__kernel_backend_postscript:n { pdf.save.ur }
2351       \__kernel_kern:n { -#1 }
2352     }
2353   }
2354   \int_gincr:N \g_pdf_backend_object_int
2355   \int_gset_eq:NN \g_pdf_backend_annotation_int \g_pdf_backend_object_int
2356   \__pdf_backend_pdfmark:x
2357   {
2358     /objdef { pdf.obj \int_use:N \g_pdf_backend_object_int }
2359     pdf.rect
2360     #4 ~
2361     /ANN
2362   }
2363 }
(End definition for \__pdf_backend_annotation:nnnn.)

```

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

```

2364 \cs_new:Npn \__pdf_backend_annotation_last:
2365 { { 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.

```

2366 \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.  
<sup>2367</sup> `\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.  
<sup>2368</sup> `\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.  
<sup>2369</sup> `\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.  
<sup>2370</sup> `\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.  
<sup>2371</sup> `\tl_new:N \l__pdf_breaklink_pdfmark_tl`  
<sup>2372</sup> `\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.  
<sup>2373</sup> `\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.  
<sup>2374</sup> `\cs_new_eq:NN \__pdf_breaklink_usebox:N \box_use:N`  
(End definition for `\__pdf_breaklink_usebox:N`.)

`\__pdf_backend_link_begin_goto:nnw` Links are crated like annotations but with dedicated code to allow for adjusting the size  
`\__pdf_backend_link_begin_user:nnw` of the rectangle. In contrast to `hyperref`, we grab the link content as a box which can  
`\__pdf_backend_link:nw` then unbox: this allows the same interface as for `pdfTeX`.  
`\__pdf_backend_link_aux:nw` Notice that the link setup here uses `/Action` not `/A`. That is because Distiller *requires*  
`\__pdf_backend_link_end:` this trigger word, rather than a “raw” PDF dictionary key (Ghostscript can handle either  
`\__pdf_backend_link_end_aux:` form).  
`\__pdf_backend_link_minima:` Taking the idea of `evenboxes` from `hypdvips`, we implement a minimum box height  
`\__pdf_backend_link_outerbox:n` and depth for link placement. This means that “underlining” with a hyperlink will  
`\__pdf_backend_link_sf_save:` generally give an even appearance. However, to ensure that the full content is always  
`\__pdf_backend_link_sf_restore:` above the link border, we do not allow this to be negative (contrast `hypdvips` approach).  
`pdf.linkdp.pad` The result should be similar to `pdfTeX` in the vast majority of foreseeable cases.  
`pdf.linkht.pad` The object number for a link is saved separately from the rest of the dictionary as  
`pdf.llx` this allows us to insert it just once, at either an unbroken link or only in the first line of  
`pdf.lly` a broken one. That makes the code clearer but also avoids a low-level PostScript error  
`pdf.ury` with the code as taken from `hypdvips`.  
`pdf.link.dict`  
`pdf.outerbox`  
`pdf.baselineskip`

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

```

2375 \cs_new_protected:Npn \__pdf_backend_link_begin_goto:nnw #1#2
2376 {
2377   \__pdf_backend_link_begin:nw
2378   { #1 /Subtype /Link /Action << /S /GoTo /D ( #2 ) >> }
2379 }
2380 \cs_new_protected:Npn \__pdf_backend_link_begin_user:nnw #1#2
2381 { \__pdf_backend_link_begin:nw {#1#2} }
2382 \cs_new_protected:Npn \__pdf_backend_link_begin:nw #1
2383 {
2384   \bool_if:NF \g__pdf_backend_link_bool
2385   { \__pdf_backend_link_begin_aux:nw {#1} }
2386 }

```

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

```

2387 \cs_new_protected:Npn \__pdf_backend_link_begin_aux:nw #1
2388 {
2389   \bool_gset_true:N \g__pdf_backend_link_bool
2390   \__kernel_backend_postscript:n
2391   { /pdf.link.dict ( #1 ) def }
2392   \tl_gset:Nn \g__pdf_backend_link_dict_tl {#1}
2393   \__pdf_backend_link_sf_save:
2394   \mode_if_math:TF
2395   { \bool_gset_true:N \g__pdf_backend_link_math_bool }
2396   { \bool_gset_false:N \g__pdf_backend_link_math_bool }
2397   \hbox_set:Nw \l__pdf_backend_content_box
2398   \__pdf_backend_link_sf_restore:
2399   \bool_if:NT \g__pdf_backend_link_math_bool
2400   { \c_math_toggle_token }
2401 }
2402 \cs_new_protected:Npn \__pdf_backend_link_end:
2403 {
2404   \bool_if:NT \g__pdf_backend_link_bool
2405   { \__pdf_backend_link_end_aux: }
2406 }
2407 \cs_new_protected:Npn \__pdf_backend_link_end_aux:
2408 {
2409   \bool_if:NT \g__pdf_backend_link_math_bool
2410   { \c_math_toggle_token }
2411   \__pdf_backend_link_sf_save:
2412   \hbox_set_end:
2413   \__pdf_backend_link_minima:
2414   \hbox_set:Nn \l__pdf_backend_model_box { Gg }
2415   \exp_args:Nx \__pdf_backend_link_outerbox:n
2416   {
2417     \int_if_odd:nTF { \value { page } }
2418     { \oddsidemargin }
2419     { \evensidemargin }
2420   }
2421   \box_move_down:nn { \box_dp:N \l__pdf_backend_content_box }
2422   { \hbox:n { \__kernel_backend_postscript:n { pdf.save.linkll } } }
2423   \__pdf_breaklink_postscript:n { pdf.bordertracking.begin }

```



```

2424 \_pdf_breaklink_usebox:N \l__pdf_backend_content_box
2425 \_pdf_breaklink_postscript:n { pdf.bordertracking.end }
2426 \box_move_up:nn { \box_ht:N \l__pdf_backend_content_box }
2427 {
2428   \hbox:n
2429   { \_kernel_backend_postscript:n { pdf.save.linkur } }
2430 }
2431 \int_gincr:N \g__pdf_backend_object_int
2432 \int_gset_eq:NN \g__pdf_backend_link_int \g__pdf_backend_object_int
2433 \_kernel_backend_postscript:x
2434 {
2435   mark
2436   /_objdef { pdf.obj \int_use:N \g__pdf_backend_link_int }
2437   \g__pdf_backend_link_dict_tl \c_space_tl
2438   pdf.rect
2439   /ANN ~ \l__pdf_breaklink_pdfmark_tl
2440 }
2441 \_pdf_backend_link_sf_restore:
2442 \bool_gset_false:N \g__pdf_backend_link_bool
2443 }
2444 \cs_new_protected:Npn \_pdf_backend_link_minima:
2445 {
2446   \hbox_set:Nn \l__pdf_backend_model_box { Gg }
2447   \_kernel_backend_postscript:x
2448   {
2449     /pdf.linkdp.pad ~
2450     \dim_to_decimal:n
2451     {
2452       \dim_max:nn
2453       {
2454         \box_dp:N \l__pdf_backend_model_box
2455         - \box_dp:N \l__pdf_backend_content_box
2456       }
2457       { Opt }
2458     } ~
2459     pdf.pt.dvi ~ def
2460     /pdf.linkht.pad ~
2461     \dim_to_decimal:n
2462     {
2463       \dim_max:nn
2464       {
2465         \box_ht:N \l__pdf_backend_model_box
2466         - \box_ht:N \l__pdf_backend_content_box
2467       }
2468       { Opt }
2469     } ~
2470     pdf.pt.dvi ~ def
2471   }
2472 }
2473 \cs_new_protected:Npn \_pdf_backend_link_outerbox:n #1
2474 {
2475   \_kernel_backend_postscript:x
2476   {
2477     /pdf.outerbox

```

```

2478     [
2479       \dim_to_decimal:n {#1} ~
2480       \dim_to_decimal:n { -\box_dp:N \l__pdf_backend_model_box } ~
2481       \dim_to_decimal:n { #1 + \textwidth } ~
2482       \dim_to_decimal:n { \box_ht:N \l__pdf_backend_model_box }
2483     ]
2484     [ exch { pdf.pt.dvi } forall ] def
2485   /pdf.baselineskip ~
2486   \dim_to_decimal:n { \tex_baselineskip:D } ~ dup ~ 0 ~ gt
2487     { pdf.pt.dvi ~ def }
2488     { pop ~ pop }
2489   ifelse
2490 }
2491 }
2492 \cs_new_protected:Npn \__pdf_backend_link_sf_save:
2493 {
2494   \int_gset:Nn \g__pdf_backend_link_sf_int
2495   {
2496     \mode_if_horizontal:TF
2497     { \tex_spacefactor:D }
2498     { 0 }
2499   }
2500 }
2501 \cs_new_protected:Npn \__pdf_backend_link_sf_restore:
2502 {
2503   \mode_if_horizontal:T
2504   {
2505     \int_compare:nNnT \g__pdf_backend_link_sf_int > { 0 }
2506     { \int_set_eq:NN \tex_spacefactor:D \g__pdf_backend_link_sf_int }
2507   }
2508 }

```

(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.

```

2509 \use_none:n
2510 {
2511   \cs_if_exist:NT \@makecol@hook
2512   {
2513     \tl_put_right:Nn \@makecol@hook
2514     {
2515       \box_if_empty:NF \@cclv
2516       {
2517         \vbox_set:Nn \@cclv
2518         {
2519           \__kernel_backend_postscript:n
2520           {
2521             pdf.globaldict /pdf.brokenlink.rect ~ known
2522             { pdf.bordertracking.continue }
2523             if
2524           }

```

```

2525         \vbox_unpack_drop:N \@cclv
2526         \__kernel_backend_postscript:n
2527         { pdf.bordertracking.endpage }
2528     }
2529 }
2530 }
2531 \tl_set:Nn \l__pdf_breaklink_pdfmark_tl { pdf.pdfmark }
2532 \cs_set_eq:NN \__pdf_breaklink_postscript:n \__kernel_backend_postscript:n
2533 \cs_set_eq:NN \__pdf_breaklink_usebox:N \hbox_unpack:N
2534 }
2535 }

```

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

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

```

2536 \cs_new:Npn \__pdf_backend_link_last:
2537 { { 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.

```

2538 \cs_new_protected:Npn \__pdf_backend_link_margin:n #1
2539 {
2540     \__kernel_backend_postscript:x
2541     {
2542         /pdf.linkmargin { \dim_to_decimal:n {#1} ~ pdf.pt.dvi } def
2543     }
2544 }

```

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

\\_pdf\_backend\_destination:nn 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.

\\_pdf\_backend\_destination:nnnn  
\\_pdf\_backend\_destination\_aux:nnnn

```

2545 \cs_new_protected:Npn \__pdf_backend_destination:nn #1#2
2546 {
2547     \__kernel_backend_postscript:n { pdf.dest.anchor }
2548     \__pdf_backend_pdfmark:x
2549     {
2550         /View
2551         [
2552             \str_case:nnF {#2}
2553             {
2554                 { xyz } { /XYZ ~ pdf.dest.point ~ null }
2555                 { fit } { /Fit }
2556                 { fitb } { /FitB }
2557                 { fitbh } { /FitBH ~ pdf.dest.y }
2558                 { fitbv } { /FitBV ~ pdf.dest.x }
2559                 { fith } { /FitH ~ pdf.dest.y }
2560                 { fitv } { /FitV ~ pdf.dest.x }
2561                 { fitr } { /Fit }
2562             }
2563         ]

```

```

2564         /XYZ ~ pdf.dest.point ~ \fp_eval:n { (#2) / 100 }
2565     }
2566 ]
2567 /Dest ( \exp_not:n {#1} ) cvn
2568 /DEST
2569 }
2570 }
2571 \cs_new_protected:Npn \__pdf_backend_destination:nnnn #1#2#3#4
2572 {
2573     \exp_args:Ne \__pdf_backend_destination_aux:nnnn
2574     { \dim_eval:n {#2} } {#1} {#3} {#4}
2575 }
2576 \cs_new_protected:Npn \__pdf_backend_destination_aux:nnnn #1#2#3#4
2577 {
2578     \vbox_to_zero:n
2579     {
2580         \__kernel_kern:n {#4}
2581         \hbox:n { \__kernel_backend_postscript:n { pdf.save.ll } }
2582         \tex_vss:D
2583     }
2584     \__kernel_kern:n {#1}
2585     \vbox_to_zero:n
2586     {
2587         \__kernel_kern:n { -#3 }
2588         \hbox:n { \__kernel_backend_postscript:n { pdf.save.ur } }
2589         \tex_vss:D
2590     }
2591     \__kernel_kern:n { -#1 }
2592     \__pdf_backend_pdfmark:n
2593     {
2594         /View
2595         [
2596             /FitR ~
2597             pdf.llx ~ pdf.lly ~ pdf.dest2device ~
2598             pdf.urx ~ pdf.ury ~ pdf.dest2device
2599         ]
2600         /Dest ( #2 ) cvn
2601         /DEST
2602     }
2603 }

```

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

#### 6.2.4 Structure

Doable for the usual ps2pdf method.

```

\__pdf_backend_compresslevel:n
\__pdf_backend_compress_objects:n
2604 \cs_new_protected:Npn \__pdf_backend_compresslevel:n #1
2605 {
2606     \int_compare:nNnT {#1} = 0
2607     {
2608         \__kernel_backend_literal_postscript:n
2609         {
2610             /setdistillerparams ~ where

```

```

2611         { pop << /CompressPages ~ false >> setdistillerparams }
2612     if
2613     }
2614 }
2615 }
2616 \cs_new_protected:Npn \__pdf_backend_compress_objects:n #1
2617 {
2618     \bool_if:nF {#1}
2619     {
2620         \__kernel_backend_literal_postscript:n
2621         {
2622             /setdistillerparams ~ where
2623             { pop << /CompressStreams ~ false >> setdistillerparams }
2624             if
2625             }
2626         }
2627     }

```

(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

```

2628 \cs_new_protected:Npn \__pdf_backend_version_major_gset:n #1
2629 {
2630     \cs_gset:Npx \__pdf_backend_version_major: { \int_eval:n {#1} }
2631 }
2632 \cs_new_protected:Npn \__pdf_backend_version_minor_gset:n #1
2633 {
2634     \cs_gset:Npx \__pdf_backend_version_minor: { \int_eval:n {#1} }
2635 }

```

(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: 2636 \cs_new:Npn \__pdf_backend_version_major: { -1 }
2637 \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:

```

2638 \cs_new_protected:Npn \__pdf_backend_bdc:nn #1#2
2639 { \__pdf_backend_pdfmark:n { /#1 ~ #2 /BDC } }
2640 \cs_new_protected:Npn \__pdf_backend_emc:
2641 { \__pdf_backend_pdfmark:n { /EMC } }

```

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

```

2642 </dvips>

```

## 6.3 LuaTeX and pdfTeX backend

2643 `<*luatex | pdftex>`

### 6.3.1 Annotations

`\_pdf_backend_annotation:nnnn` Simply pass the raw data through, just dealing with evaluation of dimensions.

```
2644 \cs_new_protected:Npn \_pdf_backend_annotation:nnnn #1#2#3#4
2645 {
2646   <*luatex>
2647     \tex_pdfextension:D annot ~
2648   </luatex>
2649   <*pdftex>
2650     \tex_pdfannot:D
2651   </pdftex>
2652     width ~ \dim_eval:n {#1} ~
2653     height ~ \dim_eval:n {#2} ~
2654     depth ~ \dim_eval:n {#3} ~
2655     {#4}
2656 }
```

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

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

```
2657 \cs_new:Npx \_pdf_backend_annotation_last:
2658 {
2659   \exp_not:N \int_value:w
2660   <*luatex>
2661     \exp_not:N \tex_pdffeedback:D lastannot ~
2662   </luatex>
2663   <*pdftex>
2664     \exp_not:N \tex_pdflastannot:D
2665   </pdftex>
2666   \c_space_tl 0 ~ R
2667 }
```

(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 2668 \cs_new_protected:Npn \_pdf_backend_link_begin_goto:nnw #1#2
\_pdf_backend_link_begin:nnnw 2669 { \_pdf_backend_link_begin:nnnw {#1} { goto~name } {#2} }
\_pdf_backend_link_end: 2670 \cs_new_protected:Npn \_pdf_backend_link_begin_user:nnw #1#2
2671 { \_pdf_backend_link_begin:nnnw {#1} { user } {#2} }
2672 \cs_new_protected:Npn \_pdf_backend_link_begin:nnnw #1#2#3
2673 {
2674   <*luatex>
2675     \tex_pdfextension:D startlink ~
2676   </luatex>
2677   <*pdftex>
2678     \tex_pdfstartlink:D
2679   </pdftex>
2680     attr {#1}
2681     #2 {#3}
```

```

2682 }
2683 \cs_new_protected:Npn \__pdf_backend_link_end:
2684 {
2685   <*luatex>
2686     \tex_pdfextension:D endlink \scan_stop:
2687 </luatex>
2688 <*pdftex>
2689   \tex_pdfendlink:D
2690 </pdftex>
2691 }

```

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

\\_\_pdf\_backend\_link\_last: Formatted for direct use.

```

2692 \cs_new:Npx \__pdf_backend_link_last:
2693 {
2694   \exp_not:N \int_value:w
2695 <*luatex>
2696   \exp_not:N \tex_pdffeedback:D lastlink ~
2697 </luatex>
2698 <*pdftex>
2699   \exp_not:N \tex_pdflastlink:D
2700 </pdftex>
2701   \c_space_tl 0 ~ R
2702 }

```

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

\\_\_pdf\_backend\_link\_margin:n A simple task: pass the data to the primitive.

```

2703 \cs_new_protected:Npn \__pdf_backend_link_margin:n #1
2704 {
2705   <*luatex>
2706     \tex_pdfvariable:D linkmargin
2707 </luatex>
2708 <*pdftex>
2709     \tex_pdflinkmargin:D
2710 </pdftex>
2711     \dim_eval:n {#1} \scan_stop:
2712 }

```

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

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

\\_\_pdf\_backend\_destination:nnnn

```

2713 \cs_new_protected:Npn \__pdf_backend_destination:nn #1#2
2714 {
2715   <*luatex>
2716     \tex_pdfextension:D dest ~
2717 </luatex>
2718 <*pdftex>
2719     \tex_pdfdest:D
2720 </pdftex>
2721     name {#1}
2722     \str_case:nnF {#2}

```

```

2723         {
2724             { xyz } { xyz }
2725             { fit } { fit }
2726             { fitb } { fitb }
2727             { fitbh } { fitbh }
2728             { fitbv } { fitbv }
2729             { fith } { fith }
2730             { fitv } { fitv }
2731             { fitr } { fitr }
2732         }
2733         { xyz ~ zoom \fp_eval:n { #2 * 10 } }
2734         \scan_stop:
2735     }
2736     \cs_new_protected:Npn \__pdf_backend_destination:nnnn #1#2#3#4
2737     {
2738         \*luatex
2739         \tex_pdfextension:D dest ~
2740         \*pdftex
2741         \tex_pdfdest:D
2742         \*pdftex
2743         name {#1}
2744         fitr ~
2745         width \dim_eval:n {#2} ~
2746         height \dim_eval:n {#3} ~
2747         depth \dim_eval:n {#4} \scan_stop:
2748     }
2749 }

```

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

### 6.3.2 Catalogue entries

```

\__pdf_backend_catalog_gput:nn
\__pdf_backend_info_gput:nn
2750 \cs_new_protected:Npn \__pdf_backend_catalog_gput:nn #1#2
2751 {
2752     \*luatex
2753     \tex_pdfextension:D catalog
2754     \*pdftex
2755     \tex_pdfcatalog:D
2756     \*pdftex
2757     { / #1 ~ #2 }
2758 }
2759 \cs_new_protected:Npn \__pdf_backend_info_gput:nn #1#2
2760 {
2761     \*luatex
2762     \tex_pdfextension:D info
2763     \*pdftex
2764     \tex_pdfinfo:D
2765     \*pdftex
2766     { / #1 ~ #2 }
2767 }
2768 }
2769 }

```

(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.

```
2770 \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`

```
2771 \cs_new_protected:Npn \__pdf_backend_object_new:nn #1#2
2772 {
2773   \*luatex
2774     \tex_pdfextension:D obj ~
2775   \*pdfTeX
2776     \tex_pdfobj:D
2777   \*pdfTeX
2778     \tex_pdfobj:D
2779     reserveobjnum ~
2780     \int_const:cn
2781     { c__pdf_backend_object_ \tl_to_str:n {#1} _int }
2782   \*luatex
2783     { \tex_pdffeedback:D lastobj }
2784   \*pdfTeX
2785     { \tex_pdflastobj:D }
2786   \*pdfTeX
2787     \prop_gput:Nnn \g__pdf_backend_object_prop {#1} {#2}
2788   }
2789   \cs_new:Npn \__pdf_backend_object_ref:n #1
2790     { \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`

`\__pdf_backend_object_write:nx`

`\__pdf_exp_not_i:nn`

`\__pdf_exp_not_ii:nn`

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

```
2792 \cs_new_protected:Npn \__pdf_backend_object_write:nn #1#2
2793 {
2794   \*luatex
2795     \tex_immediate:D \tex_pdfextension:D obj ~
2796   \*pdfTeX
2797     \tex_immediate:D \tex_pdfobj:D
2798   \*pdfTeX
2799     useobjnum ~
2800     \int_use:c
2801     { c__pdf_backend_object_ \tl_to_str:n {#1} _int }
2802   \str_case_e:nn
2803     { \prop_item:Nn \g__pdf_backend_object_prop {#1} }
2804     {
2805       { array } { { [ ~ \exp_not:n {#2} ~ ] } }
2806       { dict } { { << ~ \exp_not:n {#2} ~ >> } }
2807       { fstream }
2808       {
2809         stream ~ attr ~ { \__pdf_exp_not_i:nn #2 } ~
2810         file ~ { \__pdf_exp_not_ii:nn #2 }
2811       }
2812     }
2813   { stream }
```

```

2814         {
2815             stream ~ attr ~ { \_pdf_exp_not_i:nn #2 } ~
2816             { \_pdf_exp_not_ii:nn #2 }
2817         }
2818     }
2819 }
2820 \cs_generate_variant:Nn \_pdf_backend_object_write:nn { nx }
2821 \cs_new:Npn \_pdf_exp_not_i:nn #1#2 { \exp_not:n {#1} }
2822 \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
2823 \cs_new_protected:Npn \_pdf_backend_object_now:nn #1#2
2824 {
2825     \*luatex
2826     \tex_immediate:D \tex_pdfextension:D obj ~
2827     \*pdftex
2828     \tex_immediate:D \tex_pdfobj:D
2829     \*pdftex
2830     \str_case:nn
2831     {#1}
2832     {
2833         { array } { { [ ~ \exp_not:n {#2} ~ ] } }
2834         { dict } { { << ~ \exp_not:n {#2} ~ >> } }
2835         { fstream }
2836         {
2837             stream ~ attr ~ { \_pdf_exp_not_i:nn #2 } ~
2838             file ~ { \_pdf_exp_not_ii:nn #2 }
2839         }
2840     { stream }
2841     {
2842         stream ~ attr ~ { \_pdf_exp_not_i:nn #2 } ~
2843         { \_pdf_exp_not_ii:nn #2 }
2844     }
2845 }
2846 }
2847 }
2848 \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.

```

2849 \cs_new:Npx \_pdf_backend_object_last:
2850 {
2851     \exp_not:N \int_value:w
2852     \*luatex
2853     \exp_not:N \tex_pdffeedback:D lastobj ~
2854     \*pdftex
2855     \exp_not:N \tex_pdflastobj:D
2856     \*pdftex
2857     \c_space_tl 0 ~ R
2858 }
2859 }

```

(End definition for `\_pdf_backend_object_last:.`)

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

```

2860 \cs_new:Npx \_pdf_backend_pageobject_ref:n #1
2861 {
2862   \exp_not:N \int_value:w
2863   <*luatex>
2864   \exp_not:N \tex_pdffeedback:D pageref
2865   </luatex>
2866   <*pdftex>
2867   \exp_not:N \tex_pdfpageref:D
2868   </pdftex>
2869   \c_space_tl #1 \c_space_tl \c_space_tl \c_space_tl 0 ~ R
2870 }

```

(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
2871 \cs_new_protected:Npn \_pdf_backend_compresslevel:n #1
2872 {
2873   \tex_global:D
2874   <*luatex>
2875   \tex_pdfvariable:D compresslevel
2876   </luatex>
2877   <*pdftex>
2878   \tex_pdfcompresslevel:D
2879   </pdftex>
2880   \int_value:w \int_eval:n {#1} \scan_stop:
2881 }
2882 \cs_new_protected:Npn \_pdf_backend_compress_objects:n #1
2883 {
2884   \bool_if:nTF {#1}
2885   { \_pdf_backend_objcompresslevel:n { 2 } }
2886   { \_pdf_backend_objcompresslevel:n { 0 } }
2887 }
2888 \cs_new_protected:Npn \_pdf_backend_objcompresslevel:n #1
2889 {
2890   \tex_global:D
2891   <*luatex>
2892   \tex_pdfvariable:D objcompresslevel
2893   </luatex>
2894   <*pdftex>
2895   \tex_pdfobjcompresslevel:D
2896   </pdftex>
2897   #1 \scan_stop:
2898 }

```

(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_major_gset:n
\_pdf_backend_version_minor_gset:n
2899 \cs_new_protected:Npx \_pdf_backend_version_major_gset:n #1

```

```

2900 {
2901 <*luatex>
2902   \int_compare:nNnT \tex luatexversion:D > { 106 }
2903   {
2904     \exp_not:N \tex_global:D \tex_pdfvariable:D majorversion
2905     \exp_not:N \int_eval:n {#1} \scan_stop:
2906   }
2907 </luatex>
2908 <*pdftex>
2909   \cs_if_exist:NT \tex_pdfmajorversion:D
2910   {
2911     \exp_not:N \tex_global:D \tex_pdfmajorversion:D
2912     \exp_not:N \int_eval:n {#1} \scan_stop:
2913   }
2914 </pdftex>
2915 }
2916 \cs_new_protected:Npn \__pdf_backend_version_minor_gset:n #1
2917 {
2918   \tex_global:D
2919   <*luatex>
2920     \tex_pdfvariable:D minorversion
2921   </luatex>
2922   <*pdftex>
2923     \tex_pdfminorversion:D
2924   </pdftex>
2925   \int_eval:n {#1} \scan_stop:
2926 }

```

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

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

As above.

```

2927 \cs_new:Npx \__pdf_backend_version_major:
2928 {
2929 <*luatex>
2930   \int_compare:nNnTF \tex luatexversion:D > { 106 }
2931   { \exp_not:N \tex_the:D \tex_pdfvariable:D majorversion }
2932   { 1 }
2933 </luatex>
2934 <*pdftex>
2935   \cs_if_exist:NTF \tex_pdfmajorversion:D
2936   { \exp_not:N \tex_the:D \tex_pdfmajorversion:D }
2937   { 1 }
2938 </pdftex>
2939 }
2940 \cs_new:Npn \__pdf_backend_version_minor:
2941 {
2942   \tex_the:D
2943   <*luatex>
2944     \tex_pdfvariable:D minorversion
2945   </luatex>
2946   <*pdftex>
2947     \tex_pdfminorversion:D
2948   </pdftex>
2949 }

```

(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:`

```
2950 \cs_new_protected:Npn \_pdf_backend_bdc:nn #1#2
2951   { \_kernel_backend_literal_page:n { /#1 ~ #2 ~ BDC } }
2952 \cs_new_protected:Npn \_pdf_backend_emc:
2953   { \_kernel_backend_literal_page:n { EMC } }
```

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

```
2954 </luatex | pdftex>
```

## 6.4 dviPDFmx backend

```
2955 <*dviPDFmx | xetex>
```

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

```
\_pdf_backend:x
2956 \cs_new_protected:Npx \_pdf_backend:n #1
2957   { \_kernel_backend_literal:n { pdf: #1 } }
2958 \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
2959 \cs_new_protected:Npn \_pdf_backend_catalog_gput:nn #1#2
2960   { \_pdf_backend:n { put ~ @catalog << /#1 ~ #2 >> } }
2961 \cs_new_protected:Npn \_pdf_backend_info_gput:nn #1#2
2962   { \_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
2963 \int_new:N \g__pdf_backend_object_int
2964 \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
2965 \cs_new_protected:Npn \_pdf_backend_object_new:nn #1#2
2966   {
2967     \int_gincr:N \g__pdf_backend_object_int
2968     \int_const:cn
2969     { c__pdf_backend_object_ \tl_to_str:n {#1} _int }
2970     { \g__pdf_backend_object_int }
2971     \prop_gput:Nnn \g__pdf_backend_object_prop {#1} {#2}
2972   }
2973 \cs_new:Npn \_pdf_backend_object_ref:n #1
2974   { @pdf.obj \int_use:c { c__pdf_backend_object_ \tl_to_str:n {#1} _int } }
```

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

This is where we choose the actual type.

```

\_pdf_backend_object_write:nn
\_pdf_backend_object_write:nx
\_pdf_backend_object_write:nnn
\_pdf_backend_object_write_array:nn
\_pdf_backend_object_write_dict:nn
\_pdf_backend_object_write_fstream:nn
\_pdf_backend_object_write_stream:nn
\_pdf_backend_object_write_stream:nnnn
2975 \cs_new_protected:Npn \_pdf_backend_object_write:nn #1#2
2976 {
2977   \exp_args:Nx \_pdf_backend_object_write:nnn
2978   { \prop_item:Nn \g__pdf_backend_object_prop {#1} } {#1} {#2}
2979 }
2980 \cs_generate_variant:Nn \_pdf_backend_object_write:nn { nx }
2981 \cs_new_protected:Npn \_pdf_backend_object_write:nnn #1#2#3
2982 {
2983   \use:c { \_pdf_backend_object_write_ #1 :nn }
2984   { \_pdf_backend_object_ref:n {#2} } {#3}
2985 }
2986 \cs_new_protected:Npn \_pdf_backend_object_write_array:nn #1#2
2987 {
2988   \_pdf_backend:x
2989   { obj ~ #1 ~ [ ~ \exp_not:n {#2} ~ ] }
2990 }
2991 \cs_new_protected:Npn \_pdf_backend_object_write_dict:nn #1#2
2992 {
2993   \_pdf_backend:x
2994   { obj ~ #1 ~ << ~ \exp_not:n {#2} ~ >> }
2995 }
2996 \cs_new_protected:Npn \_pdf_backend_object_write_fstream:nn #1#2
2997 { \_pdf_backend_object_write_stream:nnnn { f } {#1} #2 }
2998 \cs_new_protected:Npn \_pdf_backend_object_write_stream:nn #1#2
2999 { \_pdf_backend_object_write_stream:nnnn { } {#1} #2 }
3000 \cs_new_protected:Npn \_pdf_backend_object_write_stream:nnnn #1#2#3#4
3001 {
3002   \_pdf_backend:x
3003   {
3004     #1 stream ~ #2 ~
3005     ( \exp_not:n {#4} ) ~ << \exp_not:n {#3} >>
3006   }
3007 }

```

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

No anonymous objects with dvipdfmx so we have to give an object name.

```

\_pdf_backend_object_now:nn
\_pdf_backend_object_now:nx
3008 \cs_new_protected:Npn \_pdf_backend_object_now:nn #1#2
3009 {
3010   \int_gincr:N \g__pdf_backend_object_int
3011   \exp_args:Nnx \use:c { \_pdf_backend_object_write_ #1 :nn }
3012   { @pdf.obj \int_use:N \g__pdf_backend_object_int }
3013   {#2}
3014 }
3015 \cs_generate_variant:Nn \_pdf_backend_object_now:nn { nx }

```

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

`\_pdf_backend_object_last:`

```

3016 \cs_new:Npn \_pdf_backend_object_last:
3017 { @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.

```

3018 \cs_new:Npn \_pdf_backend_pageobject_ref:n #1
3019 { @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.

```

3020 \int_new:N \g_pdf_backend_annotation_int

```

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

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

```

3021 \cs_new_protected:Npn \_pdf_backend_annotation:nnnn #1#2#3#4
3022 {
3023   \int_gincr:N \g_pdf_backend_object_int
3024   \int_gset_eq:NN \g_pdf_backend_annotation_int \g_pdf_backend_object_int
3025   \_pdf_backend:x
3026   {
3027     ann ~ @pdf.obj \int_use:N \g_pdf_backend_object_int \c_space_tl
3028     width ~ \dim_eval:n {#1} ~
3029     height ~ \dim_eval:n {#2} ~
3030     depth ~ \dim_eval:n {#3} ~
3031     << /Type /Annot #4 >>
3032   }
3033 }

```

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

\\_pdf\_backend\_annotation\_last:

```

3034 \cs_new:Npn \_pdf_backend_annotation_last:
3035 { @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.

```

3036 \int_new:N \g_pdf_backend_link_int

```

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

\\_pdf\_backend\_link\_begin\_goto:nnw  
\\_pdf\_backend\_link\_begin\_user:nnw  
\\_pdf\_backend\_link\_begin:n  
\\_pdf\_backend\_link\_end:

All created using the same internals.

```

3037 \cs_new_protected:Npn \_pdf_backend_link_begin_goto:nnw #1#2
3038 { \_pdf_backend_link_begin:n { #1 /Subtype /Link /A << /S /GoTo /D ( #2 ) >> } }
3039 \cs_new_protected:Npn \_pdf_backend_link_begin_user:nnw #1#2
3040 { \_pdf_backend_link_begin:n {#1#2} }
3041 \cs_new_protected:Npn \_pdf_backend_link_begin:n #1
3042 {
3043   \exp_not:N \int_gincr:N \exp_not:N \g_pdf_backend_link_int
3044   \_pdf_backend:x
3045   {
3046     bann ~
3047     @pdf.lnk

```

```

3048         \exp_not:N \int_use:N \exp_not:N \g__pdf_backend_link_int
3049         \c_space_tl
3050         <<
3051         /Type /Annot
3052         #1
3053         >>
3054     }
3055 }
3056 \cs_new_protected:Npn \__pdf_backend_link_end:
3057 { \__pdf_backend:n { eann } }

```

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

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

```

3058 \cs_new:Npn \__pdf_backend_link_last:
3059 { @pdf.lnk \int_use:N \g__pdf_backend_link_int }

```

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

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

```

3060 \cs_new_protected:Npn \__pdf_backend_link_margin:n #1
3061 { \__kernel_backend_literal:x { dvipdfmx:config-g~ \dim_eval:n {#1} } }

```

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

\\_\_pdf\_backend\_destination:nn 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 T<sub>E</sub>X by using the backend data for @xpos and @ypos. /FitR without rule spec doesn't work, so it falls back to /Fit here.

```

3062 \cs_new_protected:Npn \__pdf_backend_destination:nn #1#2
3063 {
3064     \__pdf_backend:x
3065     {
3066         dest ~ ( \exp_not:n {#1} )
3067         [
3068             @thispage
3069             \str_case:nnF {#2}
3070             {
3071                 { xyz } { /XYZ ~ @xpos ~ @ypos ~ null }
3072                 { fit } { /Fit }
3073                 { fitb } { /FitB }
3074                 { fitbh } { /FitBH }
3075                 { fitbv } { /FitBV ~ @xpos }
3076                 { fith } { /FitH ~ @ypos }
3077                 { fitv } { /FitV ~ @xpos }
3078                 { fitr } { /Fit }
3079             }
3080             { /XYZ ~ @xpos ~ @ypos ~ \fp_eval:n { (#2) / 100 } }
3081         ]
3082     }
3083 }
3084 \cs_new_protected:Npn \__pdf_backend_destination:nnnn #1#2#3#4
3085 {
3086     \exp_args:Ne \__pdf_backend_destination_aux:nnnn

```



```

3087     { \dim_eval:n {#2} } {#1} {#3} {#4}
3088   }
3089   \cs_new_protected:Npn \__pdf_backend_destination_aux:nnnn #1#2#3#4
3090   {
3091     \vbox_to_zero:n
3092     {
3093       \__kernel_kern:n {#4}
3094       \hbox:n
3095       {
3096         \__pdf_backend:n { obj ~ @pdf_ #2 _llx ~ @xpos }
3097         \__pdf_backend:n { obj ~ @pdf_ #2 _lly ~ @ypos }
3098       }
3099       \tex_vss:D
3100     }
3101     \__kernel_kern:n {#1}
3102     \vbox_to_zero:n
3103     {
3104       \__kernel_kern:n { -#3 }
3105       \hbox:n
3106       {
3107         \__pdf_backend:n
3108         {
3109           dest ~ (#2)
3110           [
3111             @thispage
3112             /FitR ~
3113             @pdf_ #2 _llx ~ @pdf_ #2 _lly ~
3114             @xpos ~ @ypos
3115           ]
3116         }
3117       }
3118       \tex_vss:D
3119     }
3120     \__kernel_kern:n { -#1 }
3121   }

```

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

#### 6.4.4 Structure

\\_pdf\_backend\_compresslevel:n Pass data to the backend: these are a one-shot.

```

\_pdf_backend_compress_objects:n
3122 \cs_new_protected:Npn \__pdf_backend_compresslevel:n #1
3123 { \__kernel_backend_literal:x { dvipdfmx:config~z~ \int_eval:n {#1} } }
3124 \cs_new_protected:Npn \__pdf_backend_compress_objects:n #1
3125 {
3126   \bool_if:nF {#1}
3127   { \__kernel_backend_literal:n { dvipdfmx:config~C~0x40 } }
3128 }

```

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

\\_pdf\_backend\_version\_major\_gset:n We start with the assumption that the default is active.

```

\_pdf_backend_version_minor_gset:n
3129 \cs_new_protected:Npn \__pdf_backend_version_major_gset:n #1
3130 {

```

```

3131 \cs_gset:Npx \__pdf_backend_version_major: { \int_eval:n {#1} }
3132 \__kernel_backend_literal:x { pdf:majorversion~ \__pdf_backend_version_major: }
3133 }
3134 \cs_new_protected:Npn \__pdf_backend_version_minor_gset:n #1
3135 {
3136 \cs_gset:Npx \__pdf_backend_version_minor: { \int_eval:n {#1} }
3137 \__kernel_backend_literal:x { pdf:minorversion~ \__pdf_backend_version_minor: }
3138 }

```

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

\\_pdf\_backend\_version\_major: We start with the assumption that the default is active.

```

\__pdf_backend_version_minor: 3139 \cs_new:Npn \__pdf_backend_version_major: { 1 }
3140 \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 Simple wrappers. May need refinement: see <https://chat.stackexchange.com/transcript/message/49970158#49970158>.

```

3141 \cs_new_protected:Npn \__pdf_backend_bdc:nn #1#2
3142 { \__kernel_backend_literal_page:n { /#1 ~ #2 ~ BDC } }
3143 \cs_new_protected:Npn \__pdf_backend_emc:
3144 { \__kernel_backend_literal_page:n { EMC } }

```

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

```
3145 </dviPDFmx | xetex>
```

### 6.5 dvisvgm backend

```
3146 <*dvisvgm>
```

#### 6.5.1 Annotations

\\_pdf\_backend\_annotation:nnnn

```
3147 \cs_new_protected:Npn \__pdf_backend_annotation:nnnn #1#2#3#4 { }
```

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

\\_pdf\_backend\_annotation\_last:

```
3148 \cs_new:Npn \__pdf_backend_annotation_last: { }
```

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

\\_pdf\_backend\_link\_begin\_goto:nnw

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

\\_pdf\_backend\_link\_begin:nnnw

\\_\_pdf\_backend\_link\_end:

```

3149 \cs_new_protected:Npn \__pdf_backend_link_begin_goto:nnw #1#2 { }
3150 \cs_new_protected:Npn \__pdf_backend_link_begin_user:nnw #1#2 { }
3151 \cs_new_protected:Npn \__pdf_backend_link_begin:nnnw #1#2#3 { }
3152 \cs_new_protected:Npn \__pdf_backend_link_end: { }

```

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

\\_\_pdf\_backend\_link\_last:

```
3153 \cs_new:Npx \__pdf_backend_link_last: { }
```

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

\\_pdf\_backend\_link\_margin:n A simple task: pass the data to the primitive.

3154 \cs\_new\_protected:Npn \\_pdf\_backend\_link\_margin:n #1 { }

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

\\_pdf\_backend\_destination:nn

\\_pdf\_backend\_destination:nnnn

3155 \cs\_new\_protected:Npn \\_pdf\_backend\_destination:nn #1#2 { }

3156 \cs\_new\_protected:Npn \\_pdf\_backend\_destination:nnnn #1#2#3#4 { }

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

### 6.5.2 Catalogue entries

\\_pdf\_backend\_catalog\_gput:nn

\\_pdf\_backend\_info\_gput:nn

No-op.

3157 \cs\_new\_protected:Npn \\_pdf\_backend\_catalog\_gput:nn #1#2 { }

3158 \cs\_new\_protected:Npn \\_pdf\_backend\_info\_gput:nn #1#2 { }

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

### 6.5.3 Objects

\\_pdf\_backend\_object\_new:nn

\\_pdf\_backend\_object\_ref:n

\\_pdf\_backend\_object\_write:nn

\\_pdf\_backend\_object\_write:nx

\\_pdf\_backend\_object\_now:nn

\\_pdf\_backend\_object\_now:nx

\\_pdf\_backend\_object\_last:

\\_pdf\_backend\_pageobject\_ref:n

All no-ops here.

3159 \cs\_new\_protected:Npn \\_pdf\_backend\_object\_new:nn #1#2 { }

3160 \cs\_new:Npn \\_pdf\_backend\_object\_ref:n #1 { }

3161 \cs\_new\_protected:Npn \\_pdf\_backend\_object\_write:nn #1#2 { }

3162 \cs\_new\_protected:Npn \\_pdf\_backend\_object\_write:nx #1#2 { }

3163 \cs\_new\_protected:Npn \\_pdf\_backend\_object\_now:nn #1#2 { }

3164 \cs\_new\_protected:Npn \\_pdf\_backend\_object\_now:nx #1#2 { }

3165 \cs\_new:Npn \\_pdf\_backend\_object\_last: { }

3166 \cs\_new:Npn \\_pdf\_backend\_pageobject\_ref:n #1 { }

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

### 6.5.4 Structure

\\_pdf\_backend\_compresslevel:n

\\_pdf\_backend\_compress\_objects:n

These are all no-ops.

3167 \cs\_new\_protected:Npn \\_pdf\_backend\_compresslevel:n #1 { }

3168 \cs\_new\_protected:Npn \\_pdf\_backend\_compress\_objects:n #1 { }

(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

Data not available!

3169 \cs\_new\_protected:Npn \\_pdf\_backend\_version\_major\_gset:n #1 { }

3170 \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.)

\\_pdf\_backend\_version\_major:

\\_pdf\_backend\_version\_minor:

Data not available!

3171 \cs\_new:Npn \\_pdf\_backend\_version\_major: { -1 }

3172 \cs\_new:Npn \\_pdf\_backend\_version\_minor: { -1 }

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

```

\__pdf_backend_bdc:nn More no-ops.
\__pdf_backend_emc:
3173 \cs_new_protected:Npn \__pdf_backend_bdc:nn #1#2 { }
3174 \cs_new_protected:Npn \__pdf_backend_emc: { }

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

3175 </dvisvgm>
3176 </package>

```

## 7 l3backend-opacity Implementation

```

3177 <*package>
3178 <@=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.

```

3179 <*dvips>

```

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

```

3180 \cs_new_protected:Npn \__opacity_backend_select:n #1
3181 {
3182   \exp_args:Nx \__opacity_backend_select_aux:n
3183   { \fp_eval:n { min(max(0,#1),1) } }
3184 }
3185 \cs_new_protected:Npn \__opacity_backend_select_aux:n #1
3186 {
3187   \__opacity_backend:nnn {#1} { fill } { ca }
3188   \__opacity_backend:nnn {#1} { stroke } { CA }
3189 }
3190 \cs_new_protected:Npn \__opacity_backend_fill:n #1
3191 {
3192   \__opacity_backend:xnn
3193   { \fp_eval:n { min(max(0,#1),1) } }
3194   { fill }
3195   { ca }
3196 }
3197 \cs_new_protected:Npn \__opacity_backend_stroke:n #1
3198 {
3199   \__opacity_backend:xnn
3200   { \fp_eval:n { min(max(0,#1),1) } }
3201   { stroke }
3202   { CA }
3203 }
3204 \cs_new_protected:Npn \__opacity_backend:nnn #1#2#3
3205 {
3206   \__kernel_backend_postscript:n

```

```

3207     {
3208         product ~ (Ghostscript) ~ search
3209         {
3210             pop ~ pop ~ pop ~
3211             #1 ~ .set #2 constantalpha
3212         }
3213         {
3214             pop ~
3215             mark ~
3216             /#3 ~ #1
3217             /SetTransparency ~
3218             pdfmark
3219         }
3220         ifelse
3221     }
3222 }
3223 \cs_generate_variant:Nn \_opacity_backend:nnn { x }

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

3224 </dvips>
3225 <*dvipdfmx | luatex | pdftex | xetex>

```

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

```

3226 \bool_lazy_and:nnT
3227 { \cs_if_exist_p:N \pdfmanagement_if_active_p: }
3228 { \pdfmanagement_if_active_p:}
3229 {
3230 <*luatex | pdftex>
3231     \_kernel_color_backend_stack_init:Nnn \c_opacity_backend_stack_int
3232     { page ~ direct } { /opacity 1 ~ gs }
3233 </luatex | pdftex>
3234     \pdfmanagement_add:nnn { Page / Resources / ExtGState }
3235     { opacity 1 } { << /ca ~ 1 /CA ~ 1 >> }
3236 }

(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
3237 \tl_new:N \l__opacity_backend_fill_tl
3238 \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:
3239 \cs_new_protected:Npn \__opacity_backend_select:n #1
3240 {
3241     \exp_args:Nx \__opacity_backend_select_aux:n
3242     { \fp_eval:n { min(max(0,#1),1) } }
3243 }
3244 \cs_new_protected:Npn \__opacity_backend_select_aux:n #1
3245 {
3246     \tl_set:Nn \l__opacity_backend_fill_tl {#1}
3247     \tl_set:Nn \l__opacity_backend_stroke_tl {#1}
3248     \pdfmanagement_add:nnn { Page / Resources / ExtGState }

```

```

3249     { opacity #1 }
3250     { << /ca ~ #1 /CA ~ #1 >> }
3251 <*dvipdfmx | xetex>
3252   \__kernel_backend_literal_pdf:n
3253 </dvipdfmx | xetex>
3254 <*luatex | pdftex>
3255   \__kernel_color_backend_stack_push:nn \c__opacity_backend_stack_int
3256 </luatex | pdftex>
3257   { /opacity #1 ~ gs }
3258   \group_insert_after:N \__opacity_backend_reset:
3259 }
3260 \bool_lazy_and:nnF
3261 { \cs_if_exist_p:N \pdfmanagement_if_active_p: }
3262 { \pdfmanagement_if_active_p:}
3263 {
3264   \cs_gset_protected:Npn \__opacity_backend_select_aux:n #1 { }
3265 }
3266 \cs_new_protected:Npn \__opacity_backend_reset:
3267 {
3268 <*dvipdfmx | xetex>
3269   \__kernel_backend_literal_pdf:n
3270   { /opacity1 ~ gs }
3271 </dvipdfmx | xetex>
3272 <*luatex | pdftex>
3273   \__kernel_color_backend_stack_pop:n \c__opacity_backend_stack_int
3274 </luatex | pdftex>
3275 }

```

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

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

```

\__opacity_backend_fillstroke:nn
\__opacity_backend_fillstroke:xx
3276 \cs_new_protected:Npn \__opacity_backend_fill:n #1
3277 {
3278   \__opacity_backend_fill_stroke:xx
3279   { \fp_eval:n { min(max(0,#1),1) } }
3280   \l__opacity_backend_stroke_tl
3281 }
3282 \cs_new_protected:Npn \__opacity_backend_stroke:n #1
3283 {
3284   \__opacity_backend_fill_stroke:xx
3285   \l__opacity_backend_fill_tl
3286   { \fp_eval:n { min(max(0,#1),1) } }
3287 }
3288 \cs_new_protected:Npn \__opacity_backend_fill_stroke:nn #1#2
3289 {
3290   \str_if_eq:nnTF {#1} {#2}
3291   { \__opacity_backend_select_aux:n {#1} }
3292   {
3293     \tl_set:Nn \l__opacity_backend_fill_tl {#1}
3294     \tl_set:Nn \l__opacity_backend_stroke_tl {#2}
3295     \pdfmanagement_add:nnn { Page / Resources / ExtGState }
3296     { opacity.fill #1 }

```

```

3297         { << /ca ~ #1 >> }
3298         \pdfmanagement_add:nnn { Page / Resources / ExtGState }
3299         { opacity.stroke #1 }
3300         { << /CA ~ #2 >> }
3301     <*dvipdfmx | xetex>
3302     \__kernel_backend_literal_pdf:n
3303 </dvipdfmx | xetex>
3304 <*luatex | pdftex>
3305     \__kernel_color_backend_stack_push:nn \c__opacity_backend_stack_int
3306 </luatex | pdftex>
3307     { /opacity.fill #1 ~ gs /opacity.stroke #2 ~ gs }
3308     \group_insert_after:N \__opacity_backend_reset:
3309 }
3310 }
3311 \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.)

3312 </dvipdfmx | luatex | pdftex | xetex>
3313 <*dvisvgm>

```

Once again, we use a scope here. There is a general opacity function for SVG, but that is of course not set up using the stack.

```

\__opacity_backend_select:n
  \__opacity_backend_fill:n
\__opacity_backend_stroke:n
  \__opacity_backend:nn
3314 \cs_new_protected:Npn \__opacity_backend_select:n #1
3315   { \__opacity_backend:nn {#1} { } }
3316 \cs_new_protected:Npn \__opacity_backend_fill:n #1
3317   { \__opacity_backend:nn {#1} { fill- } }
3318 \cs_new_protected:Npn \__opacity_backend_stroke:n #1
3319   { \__opacity_backend:nn { {#1} } { stroke- } }
3320 \cs_new_protected:Npn \__opacity_backend:nn #1#2
3321   { \__kernel_backend_scope:x { #2 opacity = " \fp_eval:n { min(max(0,#1),1) } " } }

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

3322 </dvisvgm>
3323 </package>

```

## 8 l3backend-header Implementation

```

3324 <*dvips & header>

```

**color.sc** Empty definition for color at the top level.

```

3325 /color.sc { } def

```

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

**TeXcolorseparation separation** Support for separation/spot colors: this strange naming is so things work with the color stack.

```

3326 TeXDict begin
3327 /TeXcolorseparation { setcolor } def
3328 end

```

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

pdf.globaldict A small global dictionary for backend use.

```
3329 true setglobal
3330 /pdf.globaldict 4 dict def
3331 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 3332 /pdf.cvs { 65534 string cvs } def
3333 /pdf.dvi.pt { 72.27 mul Resolution div } def
3334 /pdf.pt.dvi { 72.27 div Resolution mul } def
3335 /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 3336 /pdf.linkmargin { 1 pdf.pt.dvi } def
pdf.linkht.pad 3337 /pdf.linkdp.pad { 0 } def
3338 /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 3339 /pdf.rect
pdf.save.linkur 3340 { /Rect [ pdf.llx pdf.lly pdf.urx pdf.ury ] } def
pdf.llx 3341 /pdf.save.ll
pdf.lly 3342 {
pdf.urx 3343 currentpoint
pdf.ury 3344 /pdf.lly exch def
3345 /pdf.llx exch def
3346 }
3347 def
3348 /pdf.save.ur
3349 {
3350 currentpoint
3351 /pdf.ury exch def
3352 /pdf.urx exch def
3353 }
3354 def
3355 /pdf.save.linkll
3356 {
3357 currentpoint
3358 pdf.linkmargin add
3359 pdf.linkdp.pad add
3360 /pdf.lly exch def
3361 pdf.linkmargin sub
3362 /pdf.llx exch def
3363 }
3364 def
```



```

3365 /pdf.save.linkur
3366 {
3367     currentpoint
3368     pdf.linkmargin sub
3369     pdf.linkht.pad sub
3370     /pdf.ury exch def
3371     pdf.linkmargin add
3372     /pdf.urx exch def
3373 }
3374 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 function as it comes up a lot, and as this makes it easier to adjust if we need additional effects. We also need a more complex approach to convert a co-ordinate pair correctly when defining a rectangle: this can otherwise be out when using a landscape page. (Thanks to Alexander Grahn for the approach here.)

```

pdf.dev.x 3375 /pdf.dest.anchor
pdf.dev.y 3376 {
pdf.tmpa 3377     currentpoint exch
pdf.tmpb 3378     pdf.dvi.pt 72 add
pdf.tmpc 3379     /pdf.dest.x exch def
pdf.tmpd 3380     pdf.dvi.pt
3381     vsize 72 sub exch sub
3382     /pdf.dest.y exch def
3383 }
3384 def
3385 /pdf.dest.point
3386 { pdf.dest.x pdf.dest.y } def
3387 /pdf.dest2device
3388 {
3389     /pdf.dest.y exch def
3390     /pdf.dest.x exch def
3391     matrix currentmatrix
3392     matrix defaultmatrix
3393     matrix invertmatrix
3394     matrix concatmatrix
3395     cvx exec
3396     /pdf.dev.y exch def
3397     /pdf.dev.x exch def
3398     /pdf.tmpd exch def
3399     /pdf.tmpc exch def
3400     /pdf.tmpb exch def
3401     /pdf.tmpa exch def
3402     pdf.dest.x pdf.tmpa mul
3403     pdf.dest.y pdf.tmpc mul add
3404     pdf.dev.x add
3405     pdf.dest.x pdf.tmpb mul
3406     pdf.dest.y pdf.tmpd mul add
3407     pdf.dev.y add
3408 }
3409 def

```

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

pdf.bordertracking	To know where a breakable link can go, we need to track the boundary rectangle. That
pdf.bordertracking.begin	can be done by hooking into a and x operations: those names have to be retained. The
pdf.bordertracking.end	boundary is stored at the end of the operation. Special effort is needed at the start and
pdf.leftboundary	end of pages (or rather galleys), such that everything works properly.
pdf.rightboundary	
pdf.brokenlink.rect	3410 /pdf.bordertracking false def
pdf.brokenlink.skip	3411 /pdf.bordertracking.begin
pdf.brokenlink.dict	3412 {
pdf.bordertracking.endpage	3413 SDict /pdf.bordertracking true put
pdf.bordertracking.continue	3414 SDict /pdf.leftboundary undef
pdf.originx	3415 SDict /pdf.rightboundary undef
pdf.originy	3416 /a where
	3417 {
	3418 /a
	3419 {
	3420 currentpoint pop
	3421 SDict /pdf.rightboundary known dup
	3422 {
	3423 SDict /pdf.rightboundary get 2 index lt
	3424 { not }
	3425 if
	3426 }
	3427 if
	3428 { pop }
	3429 { SDict exch /pdf.rightboundary exch put }
	3430 ifelse
	3431 moveto
	3432 currentpoint pop
	3433 SDict /pdf.leftboundary known dup
	3434 {
	3435 SDict /pdf.leftboundary get 2 index gt
	3436 { not }
	3437 if
	3438 }
	3439 if
	3440 { pop }
	3441 { SDict exch /pdf.leftboundary exch put }
	3442 ifelse
	3443 }
	3444 put
	3445 }
	3446 if
	3447 }
	3448 def
	3449 /pdf.bordertracking.end
	3450 {
	3451 /a where { /a { moveto } put } if
	3452 /x where { /x { 0 exch rmoveto } put } if
	3453 SDict /pdf.leftboundary known
	3454 { pdf.outerbox 0 pdf.leftboundary put }
	3455 if
	3456 SDict /pdf.rightboundary known
	3457 { pdf.outerbox 2 pdf.rightboundary put }
	3458 if
	3459 SDict /pdf.bordertracking false put

```

3460 }
3461   def
3462   /pdf.bordertracking.endpage
3463 {
3464   pdf.bordertracking
3465   {
3466     pdf.bordertracking.end
3467     true setglobal
3468     pdf.globaldict
3469     /pdf.brokenlink.rect [ pdf.outerbox aload pop ] put
3470     pdf.globaldict
3471     /pdf.brokenlink.skip pdf.baselineskip put
3472     pdf.globaldict
3473     /pdf.brokenlink.dict
3474     pdf.link.dict pdf.cvs put
3475     false setglobal
3476     mark pdf.link.dict cvx exec /Rect
3477     [
3478       pdf.llx
3479       pdf.lly
3480       pdf.outerbox 2 get pdf.linkmargin add
3481       currentpoint exch pop
3482       pdf.outerbox pdf.rect.ht sub pdf.linkmargin sub
3483     ]
3484     /ANN pdf.pdfmark
3485   }
3486   if
3487 }
3488   def
3489   /pdf.bordertracking.continue
3490   {
3491     /pdf.link.dict pdf.globaldict
3492     /pdf.brokenlink.dict get def
3493     /pdf.outerbox pdf.globaldict
3494     /pdf.brokenlink.rect get def
3495     /pdf.baselineskip pdf.globaldict
3496     /pdf.brokenlink.skip get def
3497     pdf.globaldict dup dup
3498     /pdf.brokenlink.dict undef
3499     /pdf.brokenlink.skip undef
3500     /pdf.brokenlink.rect undef
3501     currentpoint
3502     /pdf.originy exch def
3503     /pdf.originx exch def
3504     /a where
3505     {
3506       /a
3507       {
3508         moveto
3509         SDict
3510         begin
3511           currentpoint pdf.originy ne exch
3512           pdf.originx ne or
3513           {

```

```

3514         pdf.save.linkll
3515         /pdf.lly
3516         pdf.lly pdf.outerbox 1 get sub def
3517         pdf.bordertracking.begin
3518     }
3519     if
3520     end
3521 }
3522 put
3523 }
3524 if
3525 /x where
3526 {
3527     /x
3528     {
3529         0 exch rmoveto
3530         SDict
3531         begin
3532         currentpoint
3533         pdf.originy ne exch pdf.originx ne or
3534         {
3535             pdf.save.linkll
3536             /pdf.lly
3537             pdf.lly pdf.outerbox 1 get sub def
3538             pdf.bordertracking.begin
3539         }
3540         if
3541         end
3542     }
3543     put
3544 }
3545 if
3546 }
3547 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  
`pdf.breaklink.write` in the dictionary, looping over key–value pairs. The first line is handled first, adjusting  
`pdf.count`          the rectangle to stay inside the text area. The second phase is a loop over the height of  
`pdf.currentrect`    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.

```

3548 /pdf.breaklink
3549 {
3550     pop
3551     counttomark 2 mod 0 eq
3552     {
3553         counttomark /pdf.count exch def
3554         {
3555             pdf.count 0 eq { exit } if
3556             counttomark 2 roll
3557             1 index /Rect eq
3558             {
3559                 dup 4 array copy

```

```

3560     dup dup
3561         1 get
3562     pdf.outerbox pdf.rect.ht
3563     pdf.linkmargin 2 mul add sub
3564     3 exch put
3565     dup
3566     pdf.outerbox 2 get
3567     pdf.linkmargin add
3568     2 exch put
3569     dup dup
3570     3 get
3571     pdf.outerbox pdf.rect.ht
3572     pdf.linkmargin 2 mul add add
3573     1 exch put
3574     /pdf.currentrect exch def
3575     pdf.breaklink.write
3576     {
3577         pdf.currentrect
3578         dup
3579         pdf.outerbox 0 get
3580         pdf.linkmargin sub
3581         0 exch put
3582         dup
3583         pdf.outerbox 2 get
3584         pdf.linkmargin add
3585         2 exch put
3586         dup dup
3587         1 get
3588         pdf.baselineskip add
3589         1 exch put
3590         dup dup
3591         3 get
3592         pdf.baselineskip add
3593         3 exch put
3594         /pdf.currentrect exch def
3595         pdf.breaklink.write
3596     }
3597     1 index 3 get
3598     pdf.linkmargin 2 mul add
3599     pdf.outerbox pdf.rect.ht add
3600     2 index 1 get sub
3601     pdf.baselineskip div round cvi 1 sub
3602     exch
3603     repeat
3604     pdf.currentrect
3605     dup
3606     pdf.outerbox 0 get
3607     pdf.linkmargin sub
3608     0 exch put
3609     dup dup
3610     1 get
3611     pdf.baselineskip add
3612     1 exch put
3613     dup dup

```

```

3614         3 get
3615         pdf.baselineskip add
3616         3 exch put
3617         dup 2 index 2 get 2 exch put
3618         /pdf.currentrect exch def
3619         pdf.breaklink.write
3620         SDict /pdf.pdfmark.good false put
3621         exit
3622     }
3623     { pdf.count 2 sub /pdf.count exch def }
3624     ifelse
3625 }
3626 loop
3627 }
3628 if
3629 /ANN
3630 }
3631 def
3632 /pdf.breaklink.write
3633 {
3634     counttomark 1 sub
3635     index /_objdef eq
3636     {
3637         counttomark -2 roll
3638         dup wcheck
3639         {
3640             readonly
3641             counttomark 2 roll
3642         }
3643         { pop pop }
3644         ifelse
3645     }
3646     if
3647     counttomark 1 add copy
3648     pop pdf.currentrect
3649     /ANN pdfmark
3650 }
3651 def

```

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

<p>pdf.pdfmark</p> <p>pdf.pdfmark.good</p> <p>pdf.outerbox</p> <p>pdf.baselineskip</p> <p>pdf.pdfmark.dict</p>	<p>The business end of breaking links starts by hooking into pdfmarks. Unlike hypdvips, 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 one apparent line high, breaking is applied.</p>
--	--

```

3652 /pdf.pdfmark
3653 {
3654     SDict /pdf.pdfmark.good true put
3655     dup /ANN eq
3656     {
3657         pdf.pdfmark.store
3658         pdf.pdfmark.dict
3659         begin

```

```

3660         Subtype /Link eq
3661         currentdict /Rect known and
3662         SDict /pdf.outerbox known and
3663         SDict /pdf.baselineskip known and
3664         {
3665             Rect 3 get
3666             pdf.linkmargin 2 mul add
3667             pdf.outerbox pdf.rect.ht add
3668             Rect 1 get sub
3669             pdf.baselineskip div round cvi 0 gt
3670             { pdf.breaklink }
3671             if
3672         }
3673         if
3674         end
3675         SDict /pdf.outerbox undef
3676         SDict /pdf.baselineskip undef
3677         currentdict /pdf.pdfmark.dict undef
3678     }
3679     if
3680     pdf.pdfmark.good
3681     { pdfmark }
3682     { cleartomark }
3683     ifelse
3684 }
3685 def
3686 /pdf.pdfmark.store
3687 {
3688     /pdf.pdfmark.dict 65534 dict def
3689     counttomark 1 add copy
3690     pop
3691     {
3692         dup mark eq
3693         {
3694             pop
3695             exit
3696         }
3697         {
3698             pdf.pdfmark.dict
3699             begin def end
3700         }
3701         ifelse
3702     }
3703     loop
3704 }
3705 def

```

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

```

3706 </dvips & header>

```

# Index

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

<b>A</b>	
<code>\AtBeginDvi</code> . . . . .	57
<b>B</b>	
bool commands:	
<code>\bool_gset_false:N</code> . . . . .	1191, 1210, 1233, 1255, 1271, 1372, 1611, 1647, 2396, 2442
<code>\bool_gset_true:N</code> . . . . .	1189, 1258, 1370, 1626, 2389, 2395
<code>\bool_if:NTF</code> . . . . .	67, 624, 1201, 1205, 1221, 1224, 1228, 1239, 1246, 1250, 1262, 1266, 1383, 1388, 1393, 1585, 1630, 1769, 1813, 1952, 1994, 2384, 2399, 2404, 2409
<code>\bool_if:nTF</code> . . . . .	2618, 2884, 3126
<code>\bool_lazy_and:nnTF</code> . . . . .	837, 2111, 3226, 3260
<code>\bool_lazy_or:nnTF</code> . . . . .	1805, 1987
<code>\bool_new:N</code> . . . . .	1192, 1259, 1373, 1627, 2369, 2370
<code>\bool_set_false:N</code> . . . . .	1780, 1916, 2018, 2182
box commands:	
<code>\box_dp:N</code> . . . . .	208, 210, 258, 260, 315, 317, 364, 366, 368, 370, 2421, 2454, 2455, 2480
<code>\box_ht:N</code> . . . . .	210, 260, 317, 368, 370, 1825, 2059, 2426, 2465, 2466, 2482
<code>\box_if_empty:N</code> . . . . .	2515
<code>\box_move_down:nn</code> . . . . .	2343, 2421
<code>\box_move_up:nn</code> . . . . .	2203, 2345, 2426
<code>\box_new:N</code> . . . . .	2228, 2333, 2334
<code>\box_set_dp:Nn</code> . . . . .	1710
<code>\box_set_ht:Nn</code> . . . . .	1709
<code>\box_set_wd:Nn</code> . . . . .	272, 1708
<code>\box_use:N</code> . . . . .	215, 233, 247, 263, 290, 304, 320, 336, 348, 399, 413, 432, 1323, 1518, 1711, 2374
<code>\box_wd:N</code> . . . . .	209, 217, 259, 265, 316, 322, 365, 367, 1824, 2058
box internal commands:	
<code>\__box_backend_clip:N</code> . . . . .	197, 197, 252, 252, 309, 309, 353, 353
<code>\l_box_backend_cos_fp</code> . . . . .	267
<code>\__box_backend_rotate:Nn</code> . . . . .	219, 219, 267, 267, 324, 324, 403, 403
<code>\__box_backend_rotate_aux:Nn</code> . . . . .	219, 220, 221, 267, 268, 269, 324, 325, 326
<code>\__box_backend_scale:Nnn</code> . . . . .	236, 236, 295, 295, 339, 339, 416, 416
<code>\l__box_backend_sin_fp</code> . . . . .	267
<b>C</b>	
clist commands:	
<code>\clist_map_function:nN</code> . . . . .	1279, 1403, 1654
color internal commands:	
<code>\__color_backend:nnn</code> . . . . .	1073, 1080, 1095, 1103, 1109
<code>\__color_backend_cmyk:w</code> . . . . .	1074
<code>\g_color_backend_colorant_prop</code> . . . . .	590, 609, 612, 632, 873
<code>\__color_backend_devicen_colorants:n</code> . . . . .	591, 591, 793, 931
<code>\__color_backend_devicen_colorants:w</code> . . . . .	591, 599, 606, 614
<code>\__color_backend_devicen_init:nnn</code> . . . . .	780, 780, 898, 898, 1130, 1130
<code>\__color_backend_devicen_init:w</code> . . . . .	898, 907, 936, 940
<code>\__color_backend_fill:n</code> . . . . .	977, 977, 979, 980, 981, 1003, 1004, 1006, 1008, 1009, 1028, 1037, 1038, 1040, 1042, 1043, 1054, 1063, 1064, 1066, 1068, 1069
<code>\__color_backend_fill_cmyk:n</code> . . . . .	977, 979, 1003, 1003, 1037, 1037, 1063, 1063
<code>\__color_backend_fill_devicen:nn</code> . . . . .	987, 997, 1027, 1031, 1053, 1057, 1124, 1126
<code>\__color_backend_fill_gray:n</code> . . . . .	977, 980, 1003, 1005, 1037, 1039, 1063, 1065
<code>\__color_backend_fill_reset:</code> . . . . .	999, 999, 1033, 1033, 1059, 1059, 1128, 1128
<code>\__color_backend_fill_rgb:n</code> . . . . .	977, 981, 1003, 1007, 1037, 1041, 1063, 1067
<code>\__color_backend_fill_separation:nn</code> . . . . .	987, 987, 997, 1027, 1027, 1031, 1053, 1053, 1057, 1124, 1124, 1126
<code>\l_color_backend_fill_tl</code> . . . . .	555, 565, 1011, 1025
<code>\__color_backend_iccbased_device:nnn</code> . . . . .	960, 960



\_color_backend_iccbased_-	\_color_backend_separation_-
init:nnn ..... 799, 799, 942, 942, 1130, 1131	init_aux:nnnnnn .... 622, 628, 644
\_color_backend_init_resource:n	\_color_backend_separation_-
..... 834, 834, 863, 934, 958, 973	init_CIELAB:nnn ..... 622, 734, 804, 851, 876
\_color_backend_pickup:N	\_color_backend_separation_-
..... 440, 440, 443, 463, 463, 466	init_CIELAB:nnnnnn ..... 805
\_color_backend_pickup:w	\_color_backend_separation_-
..... 14, 440, 454, 458, 463, 468, 471	init_count:n ..... 622, 681, 684
\_color_backend_reset: .....	\_color_backend_separation_-
..... 536, 551, 557, 569, 573, 578, 999, 1000, 1033, 1034, 1059, 1128	init_count:w ... 622, 685, 686, 690
\_color_backend_rgb:w	\_color_backend_separation_-
..... 1097	init_Device:Nn ..... 622, 666, 668, 670, 671
\_color_backend_select:n	\l_color_backend_stack_int
..... 536, 537, 539, 541, 543, 544, 573, 573, 575, 576, 577, 619	..... 497, 567, 570, 1012, 1024
\_color_backend_select:nn	\_color_backend_stroke:n
..... 557, 558, 560, 562, 563, 830	..... 977, 982, 984, 985, 986, 1003, 1016, 1018, 1020, 1021, 1030
\_color_backend_select_cmyk:n	\_color_backend_stroke_cmyk:n
..... 536, 536, 557, 557, 573, 575	..... 977, 984, 1003, 1015, 1037, 1047, 1073, 1073
\_color_backend_select_devicen:nn	\_color_backend_stroke_cmyk:w
..... 618, 620, 802, 803, 824, 832	..... 1073, 1075
\_color_backend_select_gray:n	\_color_backend_stroke_devicen:nn
..... 536, 538, 557, 559, 573, 576, 583	..... 987, 998, 1027, 1032, 1053, 1058, 1124, 1127
\_color_backend_select_iccbased:nn	\_color_backend_stroke_gray:n
..... 621, 621, 806, 806, 824, 833	..... 977, 985, 1003, 1017, 1037, 1049, 1073, 1086
\_color_backend_select_named:n	\_color_backend_stroke_gray_-
..... 536, 540, 580, 580	aux:n ..... 1073, 1090, 1094
\_color_backend_select_rgb:n	\_color_backend_stroke_reset: ..
..... 536, 542, 557, 561, 573, 577	..... 999, 1000, 1033, 1034, 1059, 1060, 1128, 1129
\_color_backend_select_separation:nn	\_color_backend_stroke_rgb:n
..... 618, 618, 620, 802, 802, 803, 824, 825, 829, 832, 833	..... 977, 986, 1003, 1019, 1037, 1051, 1073, 1096
\_color_backend_separation_-	\_color_backend_stroke_rgb:w
init:n ..... 622, 703, 716	..... 1073, 1098
\_color_backend_separation_-	\_color_backend_stroke_separation:nn
init:nn ..... 851, 861, 865	.. 987, 992, 998, 1027, 1029, 1032, 1053, 1055, 1058, 1124, 1125, 1127
\_color_backend_separation_-	\l_color_backend_stroke_tl
init:nnn ..... 622, 657, 678	..... 555, 566, 1013, 1023
\_color_backend_separation_-	\g_color_model_int
init:nnnn ..... 622, 680, 692	..... 629, 638, 786, 814, 863, 869, 870, 924, 925, 934, 958
\_color_backend_separation_-	\c_color_model_range_CIELAB_tl
init:nnnnn ..... 622, 622, 643, 736, 804, 804, 851, 851, 891	..... 741, 776, 887, 894
\_color_backend_separation_-	color.sc ..... 536, 3325
init:nw ..... 622, 707, 718, 732	cs commands:
\_color_backend_separation_-	\cs_generate_variant:Nn ..... 49, 63, 66, 99, 138, 143, 154, 185, 191, 643, 1138, 1333, 1527, 1966,
init:w ..... 622, 694, 709, 714	
\_color_backend_separation_-	
init_/DeviceCMYK:nnn ..... 622	
\_color_backend_separation_-	
init_/DeviceGray:nnn ..... 622	
\_color_backend_separation_-	
init_/DeviceRGB:nnn ..... 622	

2029, 2049, 2232, 2269, 2328, 2820,	1374, 1376, 1378, 1380, 1385, 1390,
2848, 2958, 2980, 3015, 3223, 3311	1395, 1397, 1410, 1415, 1417, 1419,
\cs_gset:Npx .. 2630, 2634, 3131, 3136	1421, 1423, 1425, 1427, 1429, 1440,
\cs_gset_protected:Npn ..... 3264	1465, 1477, 1489, 1501, 1508, 1530,
\cs_if_exist:NTF ..... 27,	1536, 1541, 1546, 1557, 1567, 1577,
50, 441, 464, 1721, 2511, 2909, 2935	1579, 1581, 1583, 1614, 1616, 1621,
\cs_if_exist_p:N ..... 838, 3227, 3261	1623, 1625, 1628, 1649, 1660, 1673,
\cs_if_exist_use:NTF ..... 38, 656	1675, 1677, 1679, 1681, 1683, 1685,
\cs_new:Npn ..... 606, 665, 667, 669, 671, 678,	1687, 1689, 1697, 1719, 1738, 1761,
684, 686, 692, 709, 716, 718, 936,	1777, 1789, 1794, 1802, 1828, 1841,
1284, 1408, 1658, 1827, 2062, 2220,	1859, 1869, 1885, 1904, 1913, 1921,
2247, 2329, 2331, 2364, 2536, 2636,	1933, 1939, 1942, 1957, 1967, 2006,
2637, 2790, 2821, 2822, 2940, 2973,	2015, 2021, 2027, 2030, 2037, 2050,
3016, 3018, 3034, 3058, 3139, 3140,	2055, 2063, 2070, 2087, 2121, 2152,
3148, 3160, 3165, 3166, 3171, 3172	2153, 2155, 2157, 2159, 2165, 2171,
\cs_new:Npx ..... 591, 2657, 2692, 2849, 2860, 2927, 3153	2179, 2185, 2188, 2190, 2201, 2230,
\cs_new_eq:NN .... 46, 57, 59, 575,	2233, 2235, 2239, 2249, 2270, 2275,
576, 577, 620, 803, 832, 833, 979,	2280, 2285, 2295, 2300, 2308, 2336,
980, 981, 984, 985, 986, 997, 998,	2341, 2373, 2375, 2380, 2382, 2387,
999, 1000, 1031, 1032, 1033, 1034,	2402, 2407, 2444, 2473, 2492, 2501,
1057, 1058, 1059, 1126, 1127, 1128,	2538, 2545, 2571, 2576, 2604, 2616,
1137, 1332, 1338, 1339, 1526, 1528,	2628, 2632, 2638, 2640, 2644, 2668,
1529, 1535, 1735, 1736, 1749, 1751,	2670, 2672, 2683, 2703, 2713, 2736,
1775, 1776, 1833, 1834, 1835, 1858,	2750, 2760, 2771, 2792, 2823, 2871,
1883, 1900, 1901, 1910, 1911, 1912,	2882, 2888, 2916, 2950, 2952, 2959,
1932, 1935, 1936, 1937, 2002, 2012,	2961, 2965, 2975, 2981, 2986, 2991,
2013, 2014, 2168, 2169, 2177, 2178,	2996, 2998, 3000, 3008, 3021, 3037,
2187, 2217, 2218, 2219, 2223, 2374	3039, 3056, 3060, 3062, 3084, 3089,
\cs_new_protected:Npn ..... 47,	3122, 3124, 3129, 3134, 3141, 3143,
54, 61, 64, 72, 78, 83, 85, 89, 100,	3147, 3149, 3150, 3151, 3152, 3154,
110, 119, 128, 141, 144, 146, 148,	3155, 3156, 3157, 3158, 3159, 3161,
152, 157, 166, 176, 186, 197, 219,	3162, 3163, 3164, 3167, 3168, 3169,
221, 236, 252, 267, 269, 295, 309,	3170, 3173, 3174, 3180, 3185, 3190,
324, 326, 339, 353, 403, 416, 440,	3197, 3204, 3239, 3244, 3266, 3276,
458, 463, 471, 500, 514, 524, 536,	3282, 3288, 3314, 3316, 3318, 3320
538, 540, 542, 544, 551, 557, 559,	\cs_new_protected:Npx ..... 622, 1109, 2899, 2956, 3041
561, 563, 569, 573, 578, 580, 618,	\cs_set_eq:NN ..... 2532, 2533
621, 644, 734, 780, 799, 802, 804,	\cs_set_protected:Npn 443, 466, 2125
805, 806, 825, 829, 834, 851, 865,	
876, 898, 942, 960, 977, 982, 987,	
992, 1003, 1005, 1007, 1009, 1015,	
1017, 1019, 1021, 1027, 1029, 1037,	
1039, 1041, 1043, 1047, 1049, 1051,	
1053, 1055, 1060, 1063, 1065, 1067,	
1069, 1073, 1075, 1086, 1094, 1096,	
1098, 1124, 1125, 1129, 1130, 1131,	
1139, 1144, 1149, 1151, 1153, 1161,	
1169, 1178, 1188, 1190, 1193, 1195,	
1212, 1217, 1235, 1257, 1260, 1273,	
1286, 1291, 1293, 1295, 1297, 1299,	
1301, 1303, 1305, 1310, 1334, 1336,	
1340, 1345, 1350, 1360, 1369, 1371,	

**D**

dim commands:

\dim_compare:nNnTF ..... 2101, 2106
\dim_compare_p:nNn ..... 2112, 2113
\dim_eval:n ..... 2339, 2574,
2652, 2653, 2654, 2711, 2746, 2747,
2748, 3028, 3029, 3030, 3061, 3087
\dim_max:nn ..... 2452, 2463
\dim_set:Nn ..... 1824, 1825, 2058, 2059, 2097, 2098
\dim_set_eq:NN ..... 2163
\dim_to_decimal:n .. 364, 365, 366,
367, 368, 370, 1539, 1544, 1550,
1551, 1552, 1553, 1562, 1563, 1564,

1655, 1674, 2210, 2211, 2450, 2461,  
 2479, 2480, 2481, 2482, 2486, 2542  
 \dim\_to\_decimal\_in\_bp:n .....  
 .... 208, 209, 210, 258, 259, 260,  
 315, 316, 317, 1157, 1158, 1165,  
 1166, 1173, 1174, 1182, 1183, 1184,  
 1281, 1285, 1289, 1343, 1348, 1354,  
 1355, 1356, 1364, 1365, 1405, 1409,  
 1413, 1659, 1743, 1744, 1745, 1746,  
 1926, 1927, 1928, 1929, 1981, 1982,  
 1983, 1984, 2195, 2196, 2197, 2198  
 \dim\_zero:N ..... 2095, 2096  
 \c\_max\_dim .....  
 .. 2097, 2098, 2101, 2106, 2112, 2113  
 draw internal commands:  
 \\_\_draw\_align\_currentpoint... .. 36  
 \\_\_draw\_backend\_add\_to\_path:n ...  
 ..... 1536,  
 1538, 1543, 1548, 1559, 1567, 1582  
 \\_\_draw\_backend\_begin: .....  
 .. 1139, 1139, 1334, 1334, 1530, 1530  
 \\_\_draw\_backend\_box\_use:Nnnnn ...  
 .. 32, 1310, 1310, 1508, 1508, 1697, 1697  
 \\_\_draw\_backend\_cap\_but: .....  
 .. 1273, 1293, 1397, 1417, 1649, 1677  
 \\_\_draw\_backend\_cap\_rectangle: ..  
 .. 1273, 1297, 1397, 1421, 1649, 1681  
 \\_\_draw\_backend\_cap\_round: .....  
 .. 1273, 1295, 1397, 1419, 1649, 1679  
 \\_\_draw\_backend\_clip: .....  
 .. 1193, 1257, 1374, 1390, 1581, 1625  
 \\_\_draw\_backend\_closepath: .....  
 ..... 1193, 1193,  
 1214, 1374, 1374, 1581, 1581, 1618  
 \\_\_draw\_backend\_closestroke: ...  
 .. 1193, 1212, 1374, 1378, 1581, 1616  
 \\_\_draw\_backend\_cm:nnnn .....  
 .... 1305, 1305, 1318, 1319, 1320,  
 1429, 1429, 1512, 1689, 1689, 1700  
 \\_\_draw\_backend\_cm\_aux:nnnn .....  
 ..... 1429, 1436, 1440  
 \\_\_draw\_backend\_cm\_decompose:nnnnN  
 ..... 1435, 1464, 1465  
 \\_\_draw\_backend\_cm\_decompose\_-  
 auxi:nnnnN ..... 1464, 1469, 1477  
 \\_\_draw\_backend\_cm\_decompose\_-  
 auxii:nnnnN ..... 1464, 1481, 1489  
 \\_\_draw\_backend\_cm\_decompose\_-  
 auxiii:nnnnN ..... 1464, 1493, 1501  
 \\_\_draw\_backend\_curveto:nnnnnn ..  
 .. 1153, 1178, 1340, 1350, 1536, 1557  
 \\_\_draw\_backend\_dash:n .....  
 ..... 1273, 1279, 1284,  
 1397, 1403, 1408, 1649, 1654, 1658  
 \\_\_draw\_backend\_dash\_aux:nn .....  
 ..... 1649, 1653, 1660  
 \\_\_draw\_backend\_dash\_pattern:nn ..  
 .. 1273, 1273, 1397, 1397, 1649, 1649  
 \\_\_draw\_backend\_discardpath: ...  
 .. 1193, 1260, 1374, 1395, 1581, 1628  
 \\_\_draw\_backend\_end: .....  
 .. 1139, 1144, 1334, 1336, 1530, 1535  
 \\_\_draw\_backend\_evenodd\_rule: ...  
 .. 1188, 1188, 1369, 1369, 1577, 1577  
 \\_\_draw\_backend\_fill: .....  
 .. 1193, 1217, 1374, 1380, 1581, 1621  
 \\_\_draw\_backend\_fillstroke: ...  
 .. 1193, 1235, 1374, 1385, 1581, 1623  
 \\_\_draw\_backend\_join\_bevel: ....  
 .. 1273, 1303, 1397, 1427, 1649, 1687  
 \\_\_draw\_backend\_join\_miter: ....  
 .. 1273, 1299, 1397, 1423, 1649, 1683  
 \\_\_draw\_backend\_join\_round: ....  
 .. 1273, 1301, 1397, 1425, 1649, 1685  
 \\_\_draw\_backend\_lineto:nn .....  
 .. 1153, 1161, 1340, 1345, 1536, 1541  
 \\_\_draw\_backend\_linewidth:n .....  
 .. 1273, 1286, 1397, 1410, 1649, 1673  
 \\_\_draw\_backend\_literal:n .....  
 ..... 1137, 1137, 1138, 1142,  
 1146, 1150, 1152, 1155, 1163, 1171,  
 1180, 1194, 1197, 1198, 1199, 1200,  
 1203, 1209, 1219, 1226, 1232, 1237,  
 1242, 1243, 1244, 1245, 1248, 1254,  
 1264, 1270, 1275, 1288, 1292, 1294,  
 1296, 1298, 1300, 1302, 1304, 1307,  
 1312, 1313, 1314, 1315, 1316, 1317,  
 1321, 1322, 1324, 1325, 1326, 1327,  
 1328, 1332, 1332, 1333, 1342, 1347,  
 1352, 1362, 1375, 1377, 1379, 1382,  
 1387, 1392, 1396, 1399, 1412, 1416,  
 1418, 1420, 1422, 1424, 1426, 1428,  
 1526, 1526, 1527, 1588, 1607, 1633  
 \\_\_draw\_backend\_miterlimit:n ...  
 .. 1273, 1291, 1397, 1415, 1649, 1675  
 \\_\_draw\_backend\_moveto:nn .....  
 .. 1153, 1153, 1340, 1340, 1536, 1536  
 \\_\_draw\_backend\_nonzero\_rule: ...  
 .. 1188, 1190, 1369, 1371, 1577, 1579  
 \\_\_draw\_backend\_path:n .....  
 ..... 1581, 1583, 1615, 1622, 1624  
 \g\_\_draw\_backend\_path\_int 1596, 1613  
 \g\_\_draw\_backend\_path\_tl .....  
 ..... 1536, 1592, 1608, 1610, 1637  
 \\_\_draw\_backend\_rectangle:nnnn ..  
 .. 1153, 1169, 1340, 1360, 1536, 1546  
 \\_\_draw\_backend\_scope\_begin: 1149,  
 1149, 1335, 1338, 1338, 1528, 1528

`\__draw_backend_scope_end:` [1149](#),  
[1151](#), [1337](#), [1338](#), [1339](#), [1528](#), [1529](#)  
`\__draw_backend_stroke:` [1193](#), [1195](#),  
[1215](#), [1374](#), [1376](#), [1581](#), [1614](#), [1619](#)  
`\g__draw_draw_clip_bool` . . . [1193](#), [1581](#)  
`\g__draw_draw_eor_bool` . . . . .  
. . . [1188](#), [1205](#), [1221](#), [1228](#), [1239](#),  
[1250](#), [1266](#), [1369](#), [1383](#), [1388](#), [1393](#)  
`\g__draw_draw_path_int` . . . . . [1581](#)  
`\g__draw_path_tl` . . . . . [1646](#)

## E

`\errmessage` . . . . . [38](#)  
`\evensidemargin` . . . . . [2419](#)  
exp commands:  
`\exp_after:wN` . . . . . [450](#), [2068](#)  
`\exp_args:Ne` . . . . .  
. . . [680](#), [1796](#), [1849](#), [1875](#), [2573](#), [3086](#)  
`\exp_args:Nf` . . . . . [1278](#), [1402](#), [2338](#)  
`\exp_args:NNf` . . . . . [220](#), [268](#), [325](#)  
`\exp_args:Nnx` . . . . . [2325](#), [3011](#)  
`\exp_args:Nv` . . . . . [445](#)  
`\exp_args:Nx` . [626](#), [861](#), [1847](#), [1873](#),  
[2282](#), [2297](#), [2415](#), [2977](#), [3182](#), [3241](#)  
`\exp_last_unbraced:Nx` . . . . . [454](#), [468](#)  
`\exp_not:N` . . . . . [593](#), [599](#), [600](#), [601](#),  
[628](#), [629](#), [632](#), [633](#), [638](#), [2659](#), [2661](#),  
[2664](#), [2694](#), [2696](#), [2699](#), [2851](#), [2853](#),  
[2856](#), [2862](#), [2864](#), [2867](#), [2904](#), [2905](#),  
[2911](#), [2912](#), [2931](#), [2936](#), [3043](#), [3048](#)  
`\exp_not:n` [48](#), [97](#), [108](#), [136](#), [950](#), [2273](#),  
[2278](#), [2567](#), [2806](#), [2807](#), [2821](#), [2822](#),  
[2834](#), [2835](#), [2989](#), [2994](#), [3005](#), [3066](#)  
`\ExplBackendFileDate` . . . . . [1](#)

## F

file commands:  
`\file_compare_timestamp:nNnTF` . [1861](#)  
`\file_parse_full_name:nNNN` [1843](#), [1871](#)  
`\fmtversion` . . . . . [52](#)  
fp commands:  
`\fp_compare:nNnTF` . . . . .  
. [227](#), [274](#), [280](#), [332](#), [1445](#), [1458](#), [1503](#)  
`\fp_eval:n` . [220](#), [229](#), [242](#), [243](#), [268](#),  
[285](#), [300](#), [302](#), [325](#), [334](#), [345](#), [346](#),  
[410](#), [425](#), [426](#), [1081](#), [1082](#), [1083](#),  
[1091](#), [1104](#), [1105](#), [1106](#), [1447](#), [1452](#),  
[1453](#), [1460](#), [1470](#), [1471](#), [1472](#), [1473](#),  
[1482](#), [1483](#), [1484](#), [1485](#), [1494](#), [1495](#),  
[1496](#), [1497](#), [2564](#), [2733](#), [3080](#), [3183](#),  
[3193](#), [3200](#), [3242](#), [3279](#), [3286](#), [3321](#)  
`\fp_new:N` . . . . . [293](#), [294](#)  
`\fp_set:Nn` . . . . . [273](#), [276](#)  
`\fp_use:N` . . . . . [279](#), [283](#), [288](#)

`\fp_zero:N` . . . . . [275](#)  
`\c_zero_fp` [227](#), [274](#), [280](#), [332](#), [1445](#), [1458](#)

## G

graphics commands:  
`\l_graphics_search_ext_seq` . . . . .  
. . . . . [1731](#), [1754](#), [1893](#), [2081](#)  
graphics internal commands:  
`\__graphics_backend_dequote:w` . . . . .  
. . . . . [1761](#), [1797](#), [1827](#)  
`\l_graphics_backend_dir_str` . [1836](#)  
`\l_graphics_backend_ext_str` . [1836](#)  
`\__graphics_backend_get_pagecount:n`  
. . . . . [1750](#), [1751](#), [1885](#), [1885](#),  
[2000](#), [2002](#), [2070](#), [2070](#), [2222](#), [2223](#)  
`\__graphics_backend_getbb_auxi:n`  
. . . . . [1761](#), [1773](#), [1787](#), [1789](#)  
`\__graphics_backend_getbb_-  
auxi:nN` . . . . . [2006](#), [2010](#), [2019](#), [2021](#)  
`\__graphics_backend_getbb_-  
auxii:n` . . . . . [1761](#), [1792](#), [1794](#)  
`\__graphics_backend_getbb_-  
auxii:nnN` . . . . . [2006](#), [2024](#), [2027](#), [2029](#)  
`\__graphics_backend_getbb_-  
auxiii:n` . . . . . [1761](#), [1796](#), [1802](#)  
`\__graphics_backend_getbb_-  
auxiii:nNnn` . [2006](#), [2025](#), [2028](#), [2030](#)  
`\__graphics_backend_getbb_-  
auxiv:nnNnn` . [2006](#), [2033](#), [2037](#), [2049](#)  
`\__graphics_backend_getbb_-  
auxv:nNnn` . . . . . [2006](#), [2034](#), [2041](#), [2050](#)  
`\__graphics_backend_getbb_-  
auxvi:nNnn` . . . . . [2053](#), [2055](#)  
`\__graphics_backend_getbb_bmp:n` .  
. . . . . [1898](#), [1912](#), [2006](#), [2014](#)  
`\__graphics_backend_getbb_eps:n` .  
. . . . . [1733](#), [1735](#), [1836](#),  
[1841](#), [1858](#), [1898](#), [1900](#), [2166](#), [2168](#)  
`\__graphics_backend_getbb_eps:nm`  
. . . . . [1836](#)  
`\__graphics_backend_getbb_eps:nn`  
. . . . . [1847](#), [1859](#)  
`\__graphics_backend_getbb_jpeg:n`  
. . . . . [1761](#), [1775](#),  
[1898](#), [1910](#), [2006](#), [2012](#), [2171](#), [2177](#)  
`\__graphics_backend_getbb_jpg:n` .  
. . . . . [1761](#), [1761](#), [1775](#), [1776](#), [1898](#), [1904](#),  
[1910](#), [1911](#), [1912](#), [2006](#), [2006](#), [2012](#),  
[2013](#), [2014](#), [2171](#), [2171](#), [2177](#), [2178](#)  
`\__graphics_backend_getbb_-  
pagebox:w` . . . . . [2006](#), [2045](#), [2062](#), [2068](#)  
`\__graphics_backend_getbb_pdf:n` .  
. . . . . [1761](#), [1777](#), [1867](#),  
[1898](#), [1913](#), [2006](#), [2015](#), [2179](#), [2179](#)

<code>\__graphics_backend_getbb_png:n .</code>	<a href="#">1761</a> , <a href="#">1776</a> ,
<code>.....</code>	<a href="#">1898</a> , <a href="#">1911</a> , <a href="#">2006</a> , <a href="#">2013</a> , <a href="#">2171</a> , <a href="#">2178</a>
<code>\__graphics_backend_getbb_ps:n . .</code>	<a href="#">1733</a> , <a href="#">1736</a> ,
<code>.....</code>	<a href="#">1836</a> , <a href="#">1858</a> , <a href="#">1898</a> , <a href="#">1901</a> , <a href="#">2166</a> , <a href="#">2169</a>
<code>\__graphics_backend_getbb_svg:n .</code>	<a href="#">2087</a> , <a href="#">2087</a>
<code>\__graphics_backend_getbb_svg_-</code>	
<code>auxi:nNn . . .</code>	<a href="#">2087</a> , <a href="#">2103</a> , <a href="#">2108</a> , <a href="#">2121</a>
<code>\__graphics_backend_getbb_svg_-</code>	
<code>auxii:w . . . .</code>	<a href="#">2087</a> , <a href="#">2125</a> , <a href="#">2147</a> , <a href="#">2152</a>
<code>\__graphics_backend_getbb_svg_-</code>	
<code>auxiii:Nw . . . . .</code>	<a href="#">2087</a> , <a href="#">2135</a> , <a href="#">2153</a>
<code>\__graphics_backend_getbb_svg_-</code>	
<code>auxiv:Nw . . . . .</code>	<a href="#">2087</a> , <a href="#">2138</a> , <a href="#">2155</a>
<code>\__graphics_backend_getbb_svg_-</code>	
<code>auxv:Nw . . . . .</code>	<a href="#">2087</a> , <a href="#">2139</a> , <a href="#">2157</a>
<code>\__graphics_backend_getbb_svg_-</code>	
<code>auxvi:Nn . . . . .</code>	<a href="#">2087</a> , <a href="#">2154</a> , <a href="#">2156</a> , <a href="#">2158</a> , <a href="#">2159</a>
<code>\__graphics_backend_getbb_svg_-</code>	
<code>auxvii:w . . . . .</code>	<a href="#">2087</a> , <a href="#">2161</a> , <a href="#">2165</a>
<code>\__graphics_backend_include:nn . .</code>	<a href="#">2185</a> , <a href="#">2186</a> , <a href="#">2189</a> , <a href="#">2190</a>
<code>\__graphics_backend_include_-</code>	
<code>auxi:nn . . . .</code>	<a href="#">1921</a> , <a href="#">1934</a> , <a href="#">1940</a> , <a href="#">1942</a>
<code>\__graphics_backend_include_-</code>	
<code>auxii:nnn . . . .</code>	<a href="#">1921</a> , <a href="#">1944</a> , <a href="#">1957</a> , <a href="#">1966</a>
<code>\__graphics_backend_include_-</code>	
<code>auxiii:nnn . . . . .</code>	<a href="#">1921</a> , <a href="#">1964</a> , <a href="#">1967</a>
<code>\__graphics_backend_include_-</code>	
<code>bmp:n . . . . .</code>	<a href="#">1921</a> , <a href="#">1937</a>
<code>\__graphics_backend_include_-</code>	
<code>dequote:w . . . . .</code>	<a href="#">2201</a> , <a href="#">2212</a> , <a href="#">2220</a>
<code>\__graphics_backend_include_-</code>	
<code>eps:n . . . . .</code>	<a href="#">1738</a> ,
<code>.....</code>	<a href="#">1738</a> , <a href="#">1749</a> , <a href="#">1836</a> , <a href="#">1869</a> , <a href="#">1883</a> ,
<code>.....</code>	<a href="#">1921</a> , <a href="#">1921</a> , <a href="#">1932</a> , <a href="#">2185</a> , <a href="#">2185</a> , <a href="#">2187</a>
<code>\__graphics_backend_include_-</code>	
<code>jpeg:n . . . . .</code>	<a href="#">1828</a> , <a href="#">1833</a> , <a href="#">1935</a> , <a href="#">2201</a> , <a href="#">2218</a>
<code>\__graphics_backend_include_-</code>	
<code>jpg:n . . . . .</code>	<a href="#">1828</a> ,
<code>.....</code>	<a href="#">1828</a> , <a href="#">1833</a> , <a href="#">1834</a> , <a href="#">1835</a> , <a href="#">1921</a> ,
<code>.....</code>	<a href="#">1933</a> , <a href="#">1935</a> , <a href="#">1936</a> , <a href="#">1937</a> , <a href="#">2201</a> , <a href="#">2219</a>
<code>\__graphics_backend_include_-</code>	
<code>jpseg:n . . . . .</code>	<a href="#">1921</a>
<code>\__graphics_backend_include_-</code>	
<code>pdf:n . . . . .</code>	<a href="#">1828</a> , <a href="#">1834</a> , <a href="#">1873</a> ,
<code>.....</code>	<a href="#">1921</a> , <a href="#">1939</a> , <a href="#">2063</a> , <a href="#">2063</a> , <a href="#">2185</a> , <a href="#">2188</a>
<code>\__graphics_backend_include_-</code>	
<code>png:n . . . . .</code>	<a href="#">1828</a> , <a href="#">1835</a> , <a href="#">1921</a> , <a href="#">1936</a> , <a href="#">2201</a> , <a href="#">2217</a>
<code>.....</code>	<a href="#">1738</a> , <a href="#">1749</a> ,
<code>\__graphics_backend_include_ps:n</code>	<a href="#">1836</a> , <a href="#">1883</a> , <a href="#">1921</a> , <a href="#">1932</a> , <a href="#">2185</a> , <a href="#">2187</a>
<code>.....</code>	<a href="#">2201</a> , <a href="#">2201</a> , <a href="#">2217</a> , <a href="#">2218</a> , <a href="#">2219</a>
<code>\__graphics_backend_loaded:n . . .</code>	<a href="#">1719</a> , <a href="#">1719</a> , <a href="#">1731</a> , <a href="#">1733</a> , <a href="#">1750</a> , <a href="#">1754</a> ,
<code>.....</code>	<a href="#">1893</a> , <a href="#">1898</a> , <a href="#">2001</a> , <a href="#">2081</a> , <a href="#">2166</a> , <a href="#">2222</a>
<code>\l__graphics_backend_name_str .</code>	<a href="#">1836</a>
<code>\__graphics_bb_restore:nTF . . . . .</code>	<a href="#">1791</a> , <a href="#">2052</a> , <a href="#">2089</a>
<code>\__graphics_bb_save:n . . . . .</code>	<a href="#">1800</a> , <a href="#">2060</a> , <a href="#">2116</a>
<code>\l__graphics_decodearray_str . . .</code>	<a href="#">1767</a> , <a href="#">1768</a> ,
<code>.....</code>	<a href="#">1779</a> , <a href="#">1807</a> , <a href="#">1811</a> , <a href="#">1812</a> , <a href="#">1915</a> , <a href="#">1950</a> ,
<code>.....</code>	<a href="#">1951</a> , <a href="#">1989</a> , <a href="#">1992</a> , <a href="#">1993</a> , <a href="#">2017</a> , <a href="#">2181</a>
<code>\__graphics_extract_bb:n . . . . .</code>	<a href="#">1908</a> , <a href="#">1917</a> , <a href="#">2175</a> , <a href="#">2183</a>
<code>\l__graphics_final_name_str . .</code>	<a href="#">1866</a>
<code>\__graphics_get_pagecount:n . . . .</code>	<a href="#">1751</a> , <a href="#">2002</a> , <a href="#">2223</a>
<code>\l__graphics_graphics_attr_tl . . .</code>	<a href="#">1760</a> , <a href="#">1765</a> ,
<code>.....</code>	<a href="#">1772</a> , <a href="#">1781</a> , <a href="#">1791</a> , <a href="#">1798</a> , <a href="#">1800</a> , <a href="#">1831</a>
<code>\l__graphics_internal_box . . . . .</code>	<a href="#">1822</a> , <a href="#">1824</a> , <a href="#">1825</a> , <a href="#">2057</a> , <a href="#">2058</a> , <a href="#">2059</a>
<code>\l__graphics_internal_dim . . . . .</code>	<a href="#">2162</a> , <a href="#">2163</a>
<code>\l__graphics_internal_ior . . . . .</code>	<a href="#">2091</a> , <a href="#">2092</a> , <a href="#">2099</a> , <a href="#">2118</a>
<code>\l__graphics_interpolate_bool . . .</code>	<a href="#">1769</a> , <a href="#">1780</a> , <a href="#">1806</a> , <a href="#">1813</a> ,
<code>.....</code>	<a href="#">1916</a> , <a href="#">1952</a> , <a href="#">1988</a> , <a href="#">1994</a> , <a href="#">2018</a> , <a href="#">2182</a>
<code>\l__graphics_llx_dim . . . . .</code>	<a href="#">1743</a> , <a href="#">1926</a> , <a href="#">1981</a> , <a href="#">2095</a> , <a href="#">2195</a>
<code>\l__graphics_lly_dim . . . . .</code>	<a href="#">1744</a> , <a href="#">1927</a> , <a href="#">1982</a> , <a href="#">2096</a> , <a href="#">2196</a>
<code>\l__graphics_page_int . . . . .</code>	<a href="#">1763</a> , <a href="#">1784</a> , <a href="#">1785</a> , <a href="#">1817</a> ,
<code>.....</code>	<a href="#">1818</a> , <a href="#">1906</a> , <a href="#">1948</a> , <a href="#">1949</a> , <a href="#">1975</a> , <a href="#">1976</a> ,
<code>.....</code>	<a href="#">2008</a> , <a href="#">2023</a> , <a href="#">2024</a> , <a href="#">2066</a> , <a href="#">2067</a> , <a href="#">2173</a>
<code>\l__graphics_pagebox_tl . . . . .</code>	<a href="#">54</a> , <a href="#">1764</a> , <a href="#">1783</a> ,
<code>.....</code>	<a href="#">1819</a> , <a href="#">1820</a> , <a href="#">1907</a> , <a href="#">1946</a> , <a href="#">1947</a> , <a href="#">1977</a> ,
<code>.....</code>	<a href="#">1979</a> , <a href="#">2009</a> , <a href="#">2032</a> , <a href="#">2033</a> , <a href="#">2068</a> , <a href="#">2174</a>
<code>\__graphics_read_bb:n . . . . .</code>	<a href="#">1735</a> , <a href="#">1736</a> , <a href="#">1900</a> , <a href="#">1901</a> , <a href="#">2168</a> , <a href="#">2169</a>
<code>\g__graphics_track_int . . . . .</code>	<a href="#">1920</a> , <a href="#">1969</a> , <a href="#">1970</a>
<code>\l__graphics_urx_dim . . . . .</code>	<a href="#">1745</a> , <a href="#">1824</a> , <a href="#">1928</a> , <a href="#">1983</a> , <a href="#">2058</a> ,
<code>.....</code>	<a href="#">2097</a> , <a href="#">2101</a> , <a href="#">2104</a> , <a href="#">2112</a> , <a href="#">2197</a> , <a href="#">2210</a>
<code>\l__graphics_ury_dim . . . . .</code>	<a href="#">1746</a> , <a href="#">1825</a> , <a href="#">1929</a> , <a href="#">1984</a> , <a href="#">2059</a> , <a href="#">2098</a> ,
<code>.....</code>	<a href="#">2106</a> , <a href="#">2109</a> , <a href="#">2113</a> , <a href="#">2198</a> , <a href="#">2203</a> , <a href="#">2211</a>
group commands:	
<code>\group_begin: . . . . .</code>	<a href="#">163</a> , <a href="#">182</a>

\group\_end: ..... 171  
 \group\_insert\_after:N ... 3258, 3308

## H

hbox commands:

\hbox:n ..... 2205, 2344, 2347,  
 2422, 2428, 2581, 2588, 3094, 3105  
 \hbox\_overlap\_right:n ..... 215,  
 247, 263, 304, 320, 348, 432, 1323, 1518  
 \hbox\_set:Nn .. 1822, 2057, 2414, 2446  
 \hbox\_set:Nw ..... 2397  
 \hbox\_set\_end: ..... 2412  
 \hbox\_unpack:N ..... 2533

hook commands:

\hook\_gput\_code:nnn .. 55, 1721, 1723

## I

int commands:

\int\_compare:nNnTF .....  
 ..... 1784, 1817, 1948, 1975,  
 2023, 2066, 2505, 2606, 2902, 2930  
 \int\_const:Nn ..... 502, 1798,  
 1888, 1970, 2072, 2242, 2780, 2968  
 \int\_eval:n 522, 532, 676, 685, 698,  
 700, 704, 717, 2630, 2634, 2880,  
 2905, 2912, 2925, 3123, 3131, 3136  
 \int\_gincr:N ..... 189,  
 355, 1587, 1632, 1969, 2241, 2310,  
 2354, 2431, 2967, 3010, 3023, 3043  
 \int\_gset:Nn ..... 164, 183, 2494  
 \int\_gset\_eq:NN 172, 2355, 2432, 3024  
 \int\_if\_exist:NTF ..... 1959  
 \int\_if\_odd:nTF ..... 2417  
 \int\_max:nn ..... 2074  
 \int\_new:N ..... 155,  
 156, 402, 497, 1613, 1920, 2237,  
 2335, 2366, 2368, 2963, 3020, 3036  
 \int\_set\_eq:NN ..... 160, 179, 2506  
 \int\_step\_function:nnnN ..... 702  
 \int\_use:N ..... 357,  
 388, 629, 638, 786, 814, 863, 869,  
 870, 924, 925, 934, 958, 1590, 1596,  
 1603, 1635, 1643, 1785, 1818, 1831,  
 1889, 1949, 1962, 1974, 1976, 2067,  
 2075, 2248, 2313, 2326, 2330, 2358,  
 2365, 2436, 2537, 2791, 2801, 2974,  
 3012, 3017, 3027, 3035, 3048, 3059  
 \int\_value:w .....  
 ..... 2659, 2694, 2851, 2862, 2880  
 \int\_zero:N ... 1763, 1906, 2008, 2173

ior commands:

\ior\_close:N ..... 2118  
 \ior\_if\_eof:NTF ..... 2092  
 \ior\_map\_break: ..... 2114

\ior\_open:Nn ..... 2091  
 \ior\_str\_map\_inline:Nn ..... 2099

## K

kernel internal commands:

\\_\_kernel\_backend\_align\_begin: ..  
 ..... 72, 72, 200, 224, 239  
 \\_\_kernel\_backend\_align\_end: ...  
 ..... 72, 78, 214, 232, 246  
 \\_\_kernel\_backend\_first\_shipout:n  
 ..... 50, 54, 57, 59, 69, 626  
 \g\_kernel\_backend\_header\_bool ..  
 ..... 67, 624  
 \\_\_kernel\_backend\_literal:n ....  
 ..... 46, 46, 47, 48,  
 49, 62, 65, 70, 74, 81, 84, 86, 142,  
 145, 147, 149, 153, 329, 342, 546,  
 552, 574, 579, 646, 782, 826, 978,  
 983, 989, 994, 1045, 1071, 1141,  
 1147, 1442, 1449, 1455, 1515, 1520,  
 1740, 1923, 1961, 1971, 2192, 2207,  
 2957, 3061, 3123, 3127, 3132, 3137  
 \\_\_kernel\_backend\_literal\_page:n  
 ..... 100,  
 100, 144, 144, 2951, 2953, 3142, 3144  
 \\_\_kernel\_backend\_literal\_pdf:n .  
 ..... 89, 89, 99, 141, 141,  
 143, 255, 312, 1332, 3252, 3269, 3302  
 \\_\_kernel\_backend\_literal\_-  
 postscript:n ..... 61,  
 61, 63, 75, 76, 80, 201, 202, 204,  
 205, 213, 225, 240, 1137, 2608, 2620  
 \\_\_kernel\_backend\_literal\_svg:n .  
 . 152, 152, 154, 159, 170, 178, 188,  
 356, 358, 375, 808, 1526, 1701, 1712  
 \\_\_kernel\_backend\_matrix:n .....  
 ..... 128, 128, 138, 277, 298, 1432  
 \\_\_kernel\_backend\_postscript:n ..  
 ..... 64, 64, 66,  
 548, 1048, 1050, 1052, 1056, 2231,  
 2287, 2302, 2344, 2350, 2390, 2422,  
 2429, 2433, 2447, 2475, 2519, 2526,  
 2532, 2540, 2547, 2581, 2588, 3206  
 \\_\_kernel\_backend\_scope:n .....  
 ..... 157, 186, 191, 385,  
 390, 1111, 1533, 1578, 1580, 1600,  
 1640, 1662, 1674, 1676, 1678, 1680,  
 1682, 1684, 1686, 1688, 1691, 3321  
 \\_\_kernel\_backend\_scope\_begin: ..  
 83, 83, 110, 110, 146, 146, 157, 157,  
 199, 223, 238, 254, 271, 297, 311,  
 328, 341, 1338, 1510, 1528, 1532, 1699  
 \\_\_kernel\_backend\_scope\_begin:n .  
 ..... 157, 176, 185, 377, 405, 418

<code>\_kernel_backend_scope_end:</code> . . .	<code>\_opacity_backend_select:n</code> . . .
. . . . . <a href="#">83</a> , <a href="#">85</a> , <a href="#">110</a> , <a href="#">119</a> ,	. . . <a href="#">3180</a> , <a href="#">3180</a> , <a href="#">3239</a> , <a href="#">3239</a> , <a href="#">3314</a> , <a href="#">3314</a>
<a href="#">146</a> , <a href="#">148</a> , <a href="#">157</a> , <a href="#">166</a> , <a href="#">216</a> , <a href="#">234</a> , <a href="#">248</a> ,	
<a href="#">264</a> , <a href="#">291</a> , <a href="#">305</a> , <a href="#">321</a> , <a href="#">337</a> , <a href="#">349</a> , <a href="#">400</a> ,	
<a href="#">414</a> , <a href="#">433</a> , <a href="#">1339</a> , <a href="#">1522</a> , <a href="#">1529</a> , <a href="#">1535</a> , <a href="#">1713</a>	
<code>\g_kernel_backend_scope_int</code> . . .	<code>\_opacity_backend_select_aux:n</code> .
<a href="#">155</a> , <a href="#">162</a> , <a href="#">164</a> , <a href="#">169</a> , <a href="#">173</a> , <a href="#">181</a> , <a href="#">183</a> , <a href="#">189</a>	. . . . . <a href="#">3180</a> , <a href="#">3182</a> ,
<code>\l_kernel_backend_scope_int</code> . . .	<a href="#">3185</a> , <a href="#">3239</a> , <a href="#">3241</a> , <a href="#">3244</a> , <a href="#">3264</a> , <a href="#">3291</a>
. . . . . <a href="#">155</a> , <a href="#">161</a> , <a href="#">174</a> , <a href="#">180</a>	<code>\c_opacity_backend_stack_int</code> . . .
<code>\g_kernel_clip_path_int</code> . . . . .	. . . . . <a href="#">3226</a> , <a href="#">3255</a> , <a href="#">3273</a> , <a href="#">3305</a>
<a href="#">353</a> , <a href="#">1587</a> , <a href="#">1590</a> , <a href="#">1603</a> , <a href="#">1632</a> , <a href="#">1635</a> , <a href="#">1643</a>	<code>\_opacity_backend_stroke:n</code> . . . .
<code>\_kernel_color_backend_stack-</code>	. . . <a href="#">3180</a> , <a href="#">3197</a> , <a href="#">3276</a> , <a href="#">3282</a> , <a href="#">3314</a> , <a href="#">3318</a>
<code>init:Nnn</code> . . . . . <a href="#">500</a> , <a href="#">500</a> , <a href="#">3231</a>	<code>\l_opacity_backend_stroke_tl</code> . . .
<code>\_kernel_color_backend_stack-</code>	. . . . . <a href="#">3237</a> , <a href="#">3247</a> , <a href="#">3280</a> , <a href="#">3294</a>
<code>pop:n</code> . . . . . <a href="#">514</a> , <a href="#">524</a> , <a href="#">570</a> , <a href="#">3273</a>	
<code>\_kernel_color_backend_stack-</code>	
<code>push:nn</code> . . . . . <a href="#">514</a> ,	
<a href="#">514</a> , <a href="#">567</a> , <a href="#">1012</a> , <a href="#">1024</a> , <a href="#">3255</a> , <a href="#">3305</a>	
<code>\_kernel_dependency_version-</code>	
<code>check:Nn</code> . . . . . <a href="#">1</a>	
<code>\_kernel_dependency_version-</code>	
<code>check:nn</code> . . . . . <a href="#">27</a> , <a href="#">29</a>	
<code>\_kernel_file_name_quote:n</code> . . . .	
. . . . . <a href="#">1849</a> , <a href="#">1875</a>	
<code>\_kernel_kern:n</code> . . . . .	
. . . . . <a href="#">2349</a> , <a href="#">2351</a> , <a href="#">2580</a> , <a href="#">2584</a> ,	
<a href="#">2587</a> , <a href="#">2591</a> , <a href="#">3093</a> , <a href="#">3101</a> , <a href="#">3104</a> , <a href="#">3120</a>	
<b>M</b>	
<code>\MessageBreak</code> . . . . . <a href="#">40</a>	
mode commands:	
<code>\mode_if_horizontal:TF</code> . . . <a href="#">2496</a> , <a href="#">2503</a>	
<code>\mode_if_math:TF</code> . . . . . <a href="#">2394</a>	
msg commands:	
<code>\msg_error:nnn</code> . . . . . <a href="#">584</a> , <a href="#">2093</a>	
<code>\msg_new:nnn</code> . . . . . <a href="#">586</a>	
<b>O</b>	
<code>\oddsidemargin</code> . . . . . <a href="#">2418</a>	
opacity internal commands:	
<code>\_opacity_backend:nn</code> . . . . .	
. . . . . <a href="#">3314</a> , <a href="#">3315</a> , <a href="#">3317</a> , <a href="#">3319</a> , <a href="#">3320</a>	
<code>\_opacity_backend:nnn</code> . . . . <a href="#">3180</a> ,	
<a href="#">3187</a> , <a href="#">3188</a> , <a href="#">3192</a> , <a href="#">3199</a> , <a href="#">3204</a> , <a href="#">3223</a>	
<code>\_opacity_backend_fill:n</code> . . . . .	
. . . <a href="#">3180</a> , <a href="#">3190</a> , <a href="#">3276</a> , <a href="#">3276</a> , <a href="#">3314</a> , <a href="#">3316</a>	
<code>\_opacity_backend_fill_stroke:nn</code>	
. . . . . <a href="#">3278</a> , <a href="#">3284</a> , <a href="#">3288</a> , <a href="#">3311</a>	
<code>\l_opacity_backend_fill_tl</code> . . . .	
. . . . . <a href="#">3237</a> , <a href="#">3246</a> , <a href="#">3285</a> , <a href="#">3293</a>	
<code>\_opacity_backend_fillstroke:nn</code>	
. . . . . <a href="#">3276</a>	
<code>\_opacity_backend_reset:</code> . . . . .	
. . . . . <a href="#">3239</a> , <a href="#">3258</a> , <a href="#">3266</a> , <a href="#">3308</a>	



\__pdf_backend_destination:nn ...	\g__pdf_backend_link_sf_int ....
..... 2545, 2545,	..... 2368, 2494, 2505, 2506
2713, 2713, 3062, 3062, 3155, 3155	\__pdf_backend_link_sf_restore: .
\__pdf_backend_destination:nnnn .	..... 2375, 2398, 2441, 2501
..... 2545, 2571,	\__pdf_backend_link_sf_save: ...
2713, 2736, 3062, 3084, 3155, 3156	..... 2375, 2393, 2411, 2492
\__pdf_backend_destination_-	\l__pdf_backend_model_box . 2334,
aux:nnnn .. 2545, 2573, 2576, 3062, 3086, 3089	2414, 2446, 2454, 2465, 2480, 2482
\__pdf_backend_emc: .. 2638, 2640,	\__pdf_backend_objcompresslevel:n
2950, 2952, 3141, 3143, 3173, 3174	..... 2871, 2885, 2886, 2888
\__pdf_backend_info_gput:nn ....	\g__pdf_backend_object_int ....
..... 2233, 2235,	..... 2237, 2241, 2244,
2750, 2760, 2959, 2961, 3157, 3158	2310, 2313, 2326, 2330, 2354, 2355,
\__pdf_backend_link:nw ..... 2375	2358, 2431, 2432, 2963, 2967, 2970,
\__pdf_backend_link_aux:nw ... 2375	3010, 3012, 3017, 3023, 3024, 3027
\__pdf_backend_link_begin:n ....	\__pdf_backend_object_last: ....
..... 3037, 3038, 3040, 3041	..... 2329, 2329,
\__pdf_backend_link_begin:nnnw ..	2849, 2849, 3016, 3016, 3159, 3165
.. 2668, 2669, 2671, 2672, 3149, 3151	\__pdf_backend_object_new:nn ...
\__pdf_backend_link_begin:nw ...	..... 2239, 2239,
..... 2377, 2381, 2382	2771, 2771, 2965, 2965, 3159, 3159
\__pdf_backend_link_begin_aux:nw	\__pdf_backend_object_now:nn ...
..... 2385, 2387	2308, 2308, 2328, 2823, 2823, 2848,
\__pdf_backend_link_begin_-	3008, 3008, 3015, 3159, 3163, 3164
goto:nnw ..... 2375, 2375,	\g__pdf_backend_object_prop ....
2668, 2668, 3037, 3037, 3149, 3149	..... 2237, 2245, 2256, 2266,
\__pdf_backend_link_begin_-	2770, 2788, 2804, 2963, 2971, 2978
user:nnw ..... 2375, 2380,	\__pdf_backend_object_ref:n ....
2668, 2670, 3037, 3039, 3149, 3150	... 2239, 2247, 2253, 2267, 2771,
\g__pdf_backend_link_bool ..... 2370, 2384, 2389, 2404, 2442	2790, 2965, 2973, 2984, 3159, 3160
\g__pdf_backend_link_dict_tl ...	\__pdf_backend_object_write:nn ..
..... 2367, 2392, 2437	2249, 2249, 2269, 2792, 2792, 2820,
\__pdf_backend_link_end: .....	2975, 2975, 2980, 3159, 3161, 3162
..... 2375, 2402,	\__pdf_backend_object_write:nnn .
2668, 2683, 3037, 3056, 3149, 3152	..... 2975, 2977, 2981
\__pdf_backend_link_end_aux: ...	\__pdf_backend_object_write_-
..... 2375, 2405, 2407	array:nn ... 2249, 2270, 2975, 2986
\g__pdf_backend_link_int .....	\__pdf_backend_object_write_-
..... 2366, 2432,	dict:nn .... 2249, 2275, 2975, 2991
2436, 2537, 3036, 3043, 3048, 3059	\__pdf_backend_object_write_-
\__pdf_backend_link_last: .....	fstream:nn . 2249, 2280, 2975, 2996
..... 2536, 2536,	\__pdf_backend_object_write_-
2692, 2692, 3058, 3058, 3153, 3153	fstream:nnn ..... 2283, 2285
\__pdf_backend_link_margin:n ...	\__pdf_backend_object_write_-
..... 2538, 2538,	stream:nn .. 2249, 2295, 2975, 2998
2703, 2703, 3060, 3060, 3154, 3154	\__pdf_backend_object_write_-
\g__pdf_backend_link_math_bool ..	stream:nnn ..... 2249, 2298, 2300
..... 2369, 2395, 2396, 2399, 2409	\__pdf_backend_object_write_-
\__pdf_backend_link_minima: ....	stream:nnnn . 2975, 2997, 2999, 3000
..... 2375, 2413, 2444	\__pdf_backend_pageobject_ref:n .
\__pdf_backend_link_outerbox:n ..	..... 2331, 2331,
..... 2375, 2415, 2473	2860, 2860, 3018, 3018, 3159, 3166
	\__pdf_backend_pdfmark:n .....
	..... 2230, 2230,



2232, 2234, 2236, 2251, 2272, 2277, 2311, 2356, 2548, 2592, 2639, 2641	pdf.lly ..... 2375, 3339
\\_pdf_backend_version_major: ... ... 2630, 2636, 2636, 2927, 2927, 3131, 3132, 3139, 3139, 3171, 3171	pdf.originx ..... 3410
\\_pdf_backend_version_major_- gset:n ..... 2628, 2628, 2899, 2899, 3129, 3129, 3169, 3169	pdf.originy ..... 3410
\\_pdf_backend_version_minor: ... ... 2634, 2636, 2637, 2927, 2940, 3136, 3137, 3139, 3140, 3171, 3172	pdf.outerbox ..... 2375, 3652
\\_pdf_backend_version_minor_- gset:n ..... 2628, 2632, 2899, 2916, 3129, 3134, 3169, 3170	pdf.pdfmark ..... 3652
\\l_pdf_breaklink_pdfmark_tl ... ..... 2371, 2439, 2531	pdf.pdfmark.dict ..... 3652
\\_pdf_breaklink_postscript:n ... ..... 2373, 2373, 2423, 2425, 2532	pdf.pdfmark.good ..... 3652
\\_pdf_breaklink_usebox:N ..... ..... 2374, 2374, 2424, 2533	pdf.pt.dvi ..... 3332
\\_pdf_exp_not_i:nn ..... .. 2792, 2810, 2815, 2821, 2838, 2843	pdf.rect ..... 3339
\\_pdf_exp_not_ii:nn ..... .. 2792, 2811, 2816, 2822, 2839, 2844	pdf.rect.ht ..... 3332
\\l_pdf_internal_box ..... 2228	pdf.rightboundary ..... 3410
pdf.baselineskip ..... 2375, 3652	pdf.save.linkll ..... 3339
pdf.bordertracking ..... 3410	pdf.save.linkur ..... 3339
pdf.bordertracking.begin ..... 3410	pdf.save.ll ..... 3339
pdf.bordertracking.continue ..... 3410	pdf.save.ur ..... 3339
pdf.bordertracking.end ..... 3410	pdf.tmpa ..... 3375
pdf.bordertracking.endpage ..... 3410	pdf.tmpb ..... 3375
pdf.breaklink ..... 3548	pdf.tmpc ..... 3375
pdf.breaklink.write ..... 3548	pdf.tmpd ..... 3375
pdf.brokenlink.dict ..... 3410	pdf.urx ..... 3339
pdf.brokenlink.rect ..... 3410	pdf.ury ..... 2375, 3339
pdf.brokenlink.skip ..... 3410	pdfmanagement commands:
pdf.count ..... 3548	\\pdfmanagement_add:nnn ..... ..... 843, 3234, 3248, 3295, 3298
pdf.currentrect ..... 3548	\\pdfmanagement_if_active_p: ... ..... 838, 839, 3227, 3228, 3261, 3262
pdf.cvs ..... 3332	peek commands:
pdf.dest.anchor ..... 3375	\\peek_meaning:Ntf ..... 2134, 2137
pdf.dest.point ..... 3375	\\peek_remove_spaces:n ..... 2132
pdf.dest.x ..... 3375	prg commands:
pdf.dest.y ..... 3375	\\prg_replicate:nn ..... ..... 168, 674, 695, 705, 906
pdf.dest2device ..... 3375	prop commands:
pdf.dev.x ..... 3375	\\prop_gput:Nnn ..... ..... 632, 873, 2245, 2788, 2971
pdf.dev.y ..... 3375	\\prop_if_in:Nntf ..... 609
pdf.dvi.pt ..... 3332	\\prop_item:Nn ..... ..... 612, 2256, 2266, 2804, 2978
pdf.globaldict ..... 3329	\\prop_new:N .... 590, 2238, 2770, 2964
pdf.leftboundary ..... 3410	\\ProvidesExplFile ..... 2
pdf.link.dict ..... 2375	
pdf.linkdp.pad ..... 2375, 3336	
pdf.linkht.pad ..... 2375, 3336	
pdf.linkmargin ..... 3336	
pdf.llx ..... 2375, 3339	

## Q

quark commands:	
\\quark_if_recursion_tail_stop:n	608
\\q_recursion_stop	601
\\q_recursion_tail	600

## S

scan commands:	
\\scan_stop:	113, 122, 532, 2162, 2165, 2686, 2711, 2734, 2748, 2880, 2897, 2905, 2912, 2925

scan internal commands:	517, 527, 2647, 2675, 2686, 2716, 2739, 2753, 2763, 2774, 2795, 2826
\__color_stop .....	\tex_pdffeedback:D .....
.... 455, 458, 469, 472, 685, 686, 690, 694, 707, 710, 714, 718, 732, 907, 936, 940, 1074, 1076, 1097, 1099	... 505, 2661, 2696, 2783, 2853, 2864
\__graphics_stop .....	\tex_pdfinfo:D ..... 2766
..... 1797, 1827, 2127, 2142, 2149, 2153, 2155, 2157, 2212, 2220	\tex_pdflastannot:D ..... 2664
separation ..... 3326	\tex_pdflastlink:D ..... 2699
seq commands:	\tex_pdflastobj:D ..... 2786, 2856
\seq_set_from_clist:Nn .....	\tex_pdflastximage:D .... 1799, 1823
..... 1732, 1756, 1895, 2083	\tex_pdflastximagepages:D .... 1889
skip commands:	\tex_pdflinkmargin:D ..... 2709
\skip_horizontal:n .... 217, 265, 322	\tex_pdfliteral:D ..... 95, 106
str commands:	\tex_pdfmajorversion:D .....
\c_hash_str .... 388, 1596, 1603, 1643	..... 2909, 2911, 2935, 2936
\c_percent_str .... 1117, 1118, 1119	\tex_pdfminorversion:D ... 2923, 2947
\str_case:nn ..... 912, 2315, 2831	\tex_pdfobj:D ..... 2777, 2798, 2829
\str_case:nnTF ..... 2552, 2722, 3069	\tex_pdfobjcompresslevel:D ... 2895
\str_case_e:nn ..... 2255, 2803	\tex_pdfpageref:D ..... 2867
\str_convert_pdfname:n . 633, 653, 862	\tex_pdfrefximage:D ..... 1823, 1830
\str_if_eq:nnTF .....	\tex_pdfrestore:D ..... 125
... 474, 477, 480, 483, 582, 812, 3290	\tex_pdfsave:D ..... 116
\str_new:N ..... 1838, 1839, 1840	\tex_pdfsetmatrix:D ..... 134
\str_tail:N ..... 1852, 1878	\tex_pdfstartlink:D ..... 2678
sys commands:	\tex_pdfvariable:D ..... 2706, 2875, 2892, 2904, 2920, 2931, 2944
\sys_if_shell:TF ..... 1836	\tex_pdfximage:D ..... 1804, 1887
\sys_shell_now:n ..... 1863	\tex_spacefactor:D ..... 2497, 2506
	\tex_special:D ..... 46
	\tex_the:D .... 1799, 2931, 2936, 2942
	\tex_vss:D .... 2582, 2589, 3099, 3118
	\tex_XeTeXpdffile:D ..... 2019, 2065
	\tex_XeTeXpdfpagecount:D ..... 2075
	\tex_XeTeXpicfile:D ..... 2010
	TeXcolorseparation ..... 3326
	\textwidth ..... 2481
	tl commands:
	\c_space_tl .....
	. 279, 284, 287, 595, 600, 638, 741, 815, 1025, 1572, 1742, 1743, 1744, 1745, 1925, 1926, 1927, 1928, 1976, 1979, 1981, 1982, 1983, 1984, 2045, 2067, 2194, 2195, 2196, 2197, 2437, 2666, 2701, 2858, 2869, 3027, 3049
	\tl_clear:N ..... 1764, 1772, 1779, 1907, 1915, 2009, 2017, 2174, 2181
	\tl_gclear:N ..... 1610, 1646
	\tl_gset:Nn ..... 1569, 2392
	\tl_if_blank:nTF ..... 510, 593, 689, 706, 713, 731, 857, 939, 2044, 2130
	\tl_if_empty:NTF . 1572, 1767, 1811, 1819, 1946, 1950, 1977, 1992, 2032
	\tl_if_empty:nTF ..... 951, 1666
	\tl_if_empty_p:N ..... 1807, 1989
	\tl_if_head_is_space:nTF ..... 445
TeX and L <sup>A</sup> T <sub>E</sub> X 2 <sub>ε</sub> commands:	
\@cclv ..... 2515, 2517, 2525	
\@ifl@t@r ..... 50, 52	
\@makecol@hook ..... 2509	
\current@color . 14, 445, 450, 455, 469	
\special ..... 2	
tex commands:	
\tex_afterassignment:D ..... 2161	
\tex_baselineskip:D ..... 2486	
\tex_endinput:D ..... 44	
\tex_global:D .....	
..... 2873, 2890, 2904, 2911, 2918	
\tex_immediate:D .....	
..... 1804, 2795, 2798, 2826, 2829	
\tex luatexversion:D .... 2902, 2930	
\tex_pdfannot:D ..... 2650	
\tex_pdfcatalog:D ..... 2756	
\tex_pdfcolorstack:D ..... 520, 530	
\tex_pdfcolorstackinit:D ..... 508	
\tex_pdfcompresslevel:D ..... 2878	
\tex_pdfdest:D ..... 2719, 2742	
\tex_pdfendlink:D ..... 2689	
\tex_pdfextension:D .....	
..... 92, 103, 113, 122, 131,	

