

# 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}{2022-04-10}{ }
5   {L3 backend support: dvipdfmx}
6 </dvipdfmx>
7 <*dvips>
8   {l3backend-dvips.def}{2022-04-10}{ }
9   {L3 backend support: dvips}
10 </dvips>
11 <*dvisvgm>
12   {l3backend-dvisvgm.def}{2022-04-10}{ }
13   {L3 backend support: dvisvgm}
14 </dvisvgm>
15 <*luatex>
16   {l3backend-luatex.def}{2022-04-10}{ }
17   {L3 backend support: PDF output (LuaTeX)}
18 </luatex>
19 <*pdftex>
20   {l3backend-pdftex.def}{2022-04-10}{ }
21   {L3 backend support: PDF output (pdfTeX)}
22 </pdftex>
23 <*xetex>
24   {l3backend-xetex.def}{2022-04-10}{ }
25   {L3 backend support: XeTeX}
26 </xetex>
```

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

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

```

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

```

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

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

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

```

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

```

(End definition for `__kernel_backend_literal:e`.)

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

```

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

```

(End definition for `__kernel_backend_first_shipout:n`.)

## 1.1 dvips backend

```

60 <*dvips>

```

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

```

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

```

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

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

`\_kernel_backend_postscript:x`

```

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

```

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

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

```

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

```

`\_kernel_backend_align_begin:`

`\_kernel_backend_align_end:`

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

```

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

```

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

`\_kernel_backend_scope_begin:`

`\_kernel_backend_scope_end:`

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

```

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

```

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

```

87 </dvips>

```

## 1.2 LuaTeX and pdfTeX backends

88 `<*luatex | pdftex>`

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

\\_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 X<sub>Y</sub>TeX. 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 X<sub>Y</sub>TeX 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 `xdvidfpmx` (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         { \exp_after:wN \use:n \current@color }
450         { 1 }
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. The exact form depends on the engine, and for `dvipdfmx/XYTeX` the backend version.

### 3.2.1 Common code

```

496 <*dvipdfmx | luatex | pdftex | xetex>

```

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

```

497 \int_new:N \l__color_backend_stack_int

```

(End definition for `\l__color_backend_stack_int`.)

```

498 </dvipdfmx | luatex | pdftex | xetex>

```

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

```

499 <*dvipdfmx | xetex>

```

`\__kernel_color_backend_stack_init:Nnn`  
`\g__color_backend_stack_int`  
`\c__color_backend_main_stack_int`

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

```

500 \int_new:N \g__color_backend_stack_int
501 \cs_new_protected:Npx \__kernel_color_backend_stack_init:Nnn #1#2#3
502 {
503     \int_gincr:N \exp_not:N \g__color_backend_stack_int
504     \int_const:Nn #1 { \exp_not:N \g__color_backend_stack_int }
505     \use:x
506     {
507         \__kernel_backend_first_shipout:n
508         {
509             \__kernel_backend_literal:n
510             {
511                 pdfcolorstackinit ~
512                 \exp_not:N \int_use:N \exp_not:N \g__color_backend_stack_int
513                 \c_space_tl
514                 \exp_not:N \tl_if_blank:nF {#2} { #2 ~ }
515                 (#3)
516             }

```

```

517     }
518   }
519 }
520 \cs_if_exist:cTF { main@pdfcolorstack }
521 {
522   \int_set:Nn \l__color_backend_stack_int
523     { \int_use:c { main@pdfcolorstack } }
524 }
525 {
526   \__kernel_color_backend_stack_init:Nnn \c__color_backend_main_stack_int
527     { page ~ direct } { 0 ~ g ~ 0 ~ G }
528   \int_set_eq:NN \l__color_backend_stack_int
529     \c__color_backend_main_stack_int
530   \int_const:cn { main@pdfcolorstack } { \c__color_backend_main_stack_int }
531 }

```

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

```

532 \cs_gset_protected:Npn \__kernel_backend_scope_end:
533 {
534   \__kernel_backend_literal:n { x:grestore }
535   \__kernel_backend_literal:x
536   {
537     pdfcolorstack ~
538     \int_use:N \g__color_backend_stack_int \c_space_tl current
539   }
540 }

```

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

```

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

```

Simple enough but needs a version check.

```

541 \cs_new_protected:Npn \__kernel_color_backend_stack_push:nn #1#2
542 {
543   \__kernel_backend_literal:x
544   {
545     pdfcolorstack ~
546     \int_eval:n {#1} ~
547     push ~ (#2)
548   }
549 }
550 \cs_generate_variant:Nn \__kernel_color_backend_stack_push:nn { nx }
551 \cs_new_protected:Npn \__kernel_color_backend_stack_pop:n #1
552 {
553   \__kernel_backend_literal:x
554   {
555     pdfcolorstack ~
556     \int_eval:n {#1} ~
557     pop
558   }
559 }

```

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

```

560 </dviPDFmx | xetex>

```

### 3.2.3 LuaTeX and pdfTeX

561  $\langle *luatex \mid pdftex \rangle$

$\backslash\_kernel\_color\_backend\_stack\_init:Nnn$

```

562 \cs_new_protected:Npn \__kernel_color_backend_stack_init:Nnn #1#2#3
563 {
564   \int_const:Nn #1
565   {
566      $\langle *luatex \rangle$ 
567     \tex_pdffeedback:D colorstackinit ~
568    $\langle /luatex \rangle$ 
569      $\langle *pdftex \rangle$ 
570     \tex_pdfcolorstackinit:D
571    $\langle /pdftex \rangle$ 
572     \tl_if_blank:nF {#2} { #2 ~ }
573     {#3}
574   }
575 }
```

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

$\backslash\_kernel\_color\_backend\_stack\_push:nn$

$\backslash\_kernel\_color\_backend\_stack\_push:nx$

$\backslash\_kernel\_color\_backend\_stack\_pop:n$

```

576 \cs_new_protected:Npn \__kernel_color_backend_stack_push:nn #1#2
577 {
578    $\langle *luatex \rangle$ 
579   \tex_pdfextension:D colorstack ~
580  $\langle /luatex \rangle$ 
581    $\langle *pdftex \rangle$ 
582   \tex_pdfcolorstack:D
583  $\langle /pdftex \rangle$ 
584   \int_eval:n {#1} ~ push ~ {#2}
585 }
586 \cs_generate_variant:Nn \__kernel_color_backend_stack_push:nn { nx }
587 \cs_new_protected:Npn \__kernel_color_backend_stack_pop:n #1
588 {
589    $\langle *luatex \rangle$ 
590   \tex_pdfextension:D colorstack ~
591  $\langle /luatex \rangle$ 
592    $\langle *pdftex \rangle$ 
593   \tex_pdfcolorstack:D
594  $\langle /pdftex \rangle$ 
595   \int_eval:n {#1} ~ pop \scan_stop:
596 }
```

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

597  $\langle /luatex \mid pdftex \rangle$

## 3.3 General color

### 3.3.1 dvips-style

598  $\langle *dvips \mid dvisvgm \rangle$

```

\__color_backend_select_cmyk:n
\__color_backend_select_gray:n
\__color_backend_select_rgb:n
\__color_backend_select:n
\__color_backend_reset:
    color.sc
599 \cs_new_protected:Npn \__color_backend_select_cmyk:n #1
600 { \__color_backend_select:n { cmyk ~ #1 } }
601 \cs_new_protected:Npn \__color_backend_select_gray:n #1
602 { \__color_backend_select:n { gray ~ #1 } }
603 \cs_new_protected:Npn \__color_backend_select_rgb:n #1
604 { \__color_backend_select:n { rgb ~ #1 } }
605 \cs_new_protected:Npn \__color_backend_select:n #1
606 {
607     \__kernel_backend_literal:n { color~push~ #1 }
608     <dvips>
609     \__kernel_backend_postscript:n { /color.sc ~ { } ~ def }
610     </dvips>
611 }
612 \cs_new_protected:Npn \__color_backend_reset:
613 { \__kernel_backend_literal:n { color~pop } }

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

614 </dvips | dvisvgm>

```

### 3.3.2 LuaTeX and pdfTeX

```

615 <dvipdfmx | luatex | pdftex | xetex>

\l__color_backend_fill_tl
\l__color_backend_stroke_tl
616 \tl_new:N \l__color_backend_fill_tl
617 \tl_new:N \l__color_backend_stroke_tl

(End definition for \l__color_backend_fill_tl and \l__color_backend_stroke_tl.)

\__color_backend_select_cmyk:n
\__color_backend_select_gray:n
\__color_backend_select_rgb:n
\__color_backend_select:nn
\__color_backend_reset:
618 \cs_new_protected:Npn \__color_backend_select_cmyk:n #1
619 { \__color_backend_select:nn { #1 ~ k } { #1 ~ K } }
620 \cs_new_protected:Npn \__color_backend_select_gray:n #1
621 { \__color_backend_select:nn { #1 ~ g } { #1 ~ G } }
622 \cs_new_protected:Npn \__color_backend_select_rgb:n #1
623 { \__color_backend_select:nn { #1 ~ rg } { #1 ~ RG } }
624 \cs_new_protected:Npn \__color_backend_select:nn #1#2
625 {
626     \tl_set:Nn \l__color_backend_fill_tl {#1}
627     \tl_set:Nn \l__color_backend_stroke_tl {#2}
628     \__kernel_color_backend_stack_push:nn \l__color_backend_stack_int { #1 ~ #2 }
629 }
630 \cs_new_protected:Npn \__color_backend_reset:
631 { \__kernel_color_backend_stack_pop:n \l__color_backend_stack_int }

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

632 </dvipdfmx | luatex | pdftex | xetex>

```

### 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. Thus it is used in preference to the `dvips`-style interface or the “native” color specials (which have only one stack).

## 3.4 Separations

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

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

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

```
\g_color_backend_colorant_prop

634 \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

635 \cs_new:Npx \__color_backend_devicen_colorants:n #1
636 {
637   \exp_not:N \tl_if_blank:nF {#1}
638   {
639     \c_space_tl
640     << ~
641     /Colorants ~
642     << ~
643     \exp_not:N \__color_backend_devicen_colorants:w #1 ~
644     \exp_not:N \q_recursion_tail \c_space_tl
645     \exp_not:N \q_recursion_stop
646     >> ~
647   }
648 }
649 }
650 \cs_new:Npn \__color_backend_devicen_colorants:w #1 ~
651 {
652   \quark_if_recursion_tail_stop:n {#1}
653   \prop_if_in:NnT \g__color_backend_colorant_prop {#1}
654   {
655     #1 ~
656     \prop_item:Nn \g__color_backend_colorant_prop {#1} ~
657   }
658   \__color_backend_devicen_colorants:w
659 }

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

660 < /dvipdfmx | luatex | pdftex | xetex | dvips >
661 < *dvips >
```

```

\__color_backend_select_separation:nn
\__color_backend_select_device:nn
662 \cs_new_protected:Npn \__color_backend_select_separation:nn #1#2
663 { \__color_backend_select:n { separation ~ #1 ~ #2 } }
664 \cs_new_eq:NN \__color_backend_select_device:nn \__color_backend_select_separation:nn

(End definition for \__color_backend_select_separation:nn and \__color_backend_select_device:nn.)

\__color_backend_select_iccbased:nn No support.
665 \cs_new_protected:Npn \__color_backend_select_iccbased:nn #1#2 { }

(End definition for \__color_backend_select_iccbased:nn.)

\__color_backend_separation_init:nnnnn
\__color_backend_separation_init:nxxnn
\__color_backend_separation_init_aux:nnnnnn
\__color_backend_separation_init_/DeviceCMYK:nnn
\__color_backend_separation_init_/DeviceGray:nnn
\__color_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
666 \cs_new_protected:Npx \__color_backend_separation_init:nnnnn #1#2#3#4#5
667 {
668   \bool_if:NT \g__kernel_backend_header_bool
669   {
670     \exp_args:Nx \__kernel_backend_first_shipout:n
671     {
672       \exp_not:N \__color_backend_separation_init_aux:nnnnnn
673       { \exp_not:N \int_use:N \g__color_model_int }
674       {#1} {#2} {#3} {#4} {#5}
675     }
676     \prop_gput:Nxx \exp_not:N \g__color_backend_colorant_prop
677     { / \exp_not:N \str_convert_pdfname:n {#1} }
678     {
679       << ~
680       /setcolorspace ~ {} ~
681       >> ~ begin ~
682       color \exp_not:N \int_use:N \g__color_model_int \c_space_tl
683       end
684     }
685   }
686 }
687 \cs_generate_variant:Nn \__color_backend_separation_init:nnnnn { nxx }
688 \cs_new_protected:Npn \__color_backend_separation_init_aux:nnnnnn #1#2#3#4#5#6
689 {
690   \__kernel_backend_literal:e
691   {
692     !
693     TeXDict ~ begin ~
694     /color #1
695     {
696       [ ~
697       /Separation ~ ( \str_convert_pdfname:n {#2} ) ~
698       [ ~ #3 ~ ] ~
699       {
700         \cs_if_exist_use:cF { __color_backend_separation_init_ #3 :nnn }
701         { \__color_backend_separation_init:nnn }
702         {#4} {#5} {#6}
703       }

```

```

704         ] ~ setcolorspace
705     } ~ def ~
706     end
707 }
708 }
709 \cs_new:cpn { __color_backend_separation_init_ /DeviceCMYK :nnn } #1#2#3
710 { \__color_backend_separation_init_Device:Nn 4 {#3} }
711 \cs_new:cpn { __color_backend_separation_init_ /DeviceGray :nnn } #1#2#3
712 { \__color_backend_separation_init_Device:Nn 1 {#3} }
713 \cs_new:cpn { __color_backend_separation_init_ /DeviceRGB :nnn } #1#2#3
714 { \__color_backend_separation_init_Device:Nn 2 {#3} }
715 \cs_new:Npn \__color_backend_separation_init_Device:Nn #1#2
716 {
717     #2 ~
718     \prg_replicate:nn {#1}
719     { #1 ~ index ~ mul ~ #1 ~ 1 ~ roll ~ }
720     \int_eval:n { #1 + 1 } ~ -1 ~ roll ~ pop
721 }

```

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.

```

722 \cs_new:Npn \__color_backend_separation_init:nnn #1#2#3
723 {
724     \exp_args:Ne \__color_backend_separation_init:nnnn
725     { \__color_backend_separation_init_count:n {#2} }
726     {#1} {#2} {#3}
727 }
728 \cs_new:Npn \__color_backend_separation_init_count:n #1
729 { \int_eval:n { 0 \__color_backend_separation_init_count:w #1 ~ \s_color_stop } }
730 \cs_new:Npn \__color_backend_separation_init_count:w #1 ~ #2 \s_color_stop
731 {
732     +1
733     \tl_if_blank:nF {#2}
734     { \__color_backend_separation_init_count:w #2 \s_color_stop }
735 }

```

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

```

736 \cs_new:Npn \__color_backend_separation_init:nnnn #1#2#3#4
737 {
738     \__color_backend_separation_init:w #3 ~ \s_color_stop #4 ~ \s_color_stop
739     \prg_replicate:nn {#1}
740     {

```

```

741     pop ~ 1 ~ index ~ neg ~ 1 ~ index ~ add ~
742     \int_eval:n { 3 * #1 } ~ index ~ mul ~
743     2 ~ index ~ add ~
744     \int_eval:n { 3 * #1 } ~ #1 ~ roll ~
745   }
746   \int_step_function:nnnN {#1} { -1 } { 1 }
747   \__color_backend_separation_init:n
748   \int_eval:n { 4 * #1 + 1 } ~ #1 ~ roll ~
749   \prg_replicate:nn { 3 * #1 + 1 } { pop ~ }
750   \tl_if_blank:nF {#2}
751   { \__color_backend_separation_init:nw {#1} #2 ~ \s__color_stop }
752 }
753 \cs_new:Npn \__color_backend_separation_init:w
754 #1 ~ #2 \s__color_stop #3 ~ #4 \s__color_stop
755 {
756   #1 ~ #3 ~ 0 ~
757   \tl_if_blank:nF {#2}
758   { \__color_backend_separation_init:w #2 \s__color_stop #4 \s__color_stop }
759 }
760 \cs_new:Npn \__color_backend_separation_init:n #1
761 { \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.

```

762 \cs_new:Npn \__color_backend_separation_init:nw #1#2 ~ #3 ~ #4 \s__color_stop
763 {
764   #2 ~ #3 ~
765   2 ~ index ~ 2 ~ index ~ lt ~
766   { ~ pop ~ exch ~ pop ~ } ~
767   { ~
768     2 ~ index ~ 1 ~ index ~ gt ~
769     { ~ exch ~ pop ~ exch ~ pop ~ } ~
770     { ~ pop ~ pop ~ } ~
771     ifelse ~
772   }
773   ifelse ~
774   #1 ~ 1 ~ roll ~
775   \tl_if_blank:nF {#4}
776   { \__color_backend_separation_init:nw {#1} #4 \s__color_stop }
777 }

```

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.

```

778 \cs_new_protected:Npn \__color_backend_separation_init_CIELAB:nnn #1#2#3
779 {
780   \__color_backend_separation_init:nxxxnn
781   {#2}
782   {
783     /CIEBasedABC ~
784     << ~
785     /RangeABC ~ [ ~ \c__color_model_range_CIELAB_tl \c_space_tl ] ~
786     /DecodeABC ~
787     [ ~
788     { ~ 16 ~ add ~ 116 ~ div ~ } ~ bind ~

```

```

789         { ~ 500 ~ div ~ } ~ bind ~
790         { ~ 200 ~ div ~ } ~ bind ~
791     ] ~
792     /MatrixABC ~ [ ~ 1 ~ 1 ~ 1 ~ 1 ~ 0 ~ 0 ~ 0 ~ 0 ~ -1 ~ ] ~
793     /DecodeLMN ~
794     [ ~
795     { ~
796         dup ~ 6 ~ 29 ~ div ~ ge ~
797         { ~ dup ~ dup ~ mul ~ mul ~ ~ } ~
798         { ~ 4 ~ 29 ~ div ~ sub ~ 108 ~ 841 ~ div ~ mul ~ } ~
799         ifelse ~
800         0.9505 ~ mul ~
801     } ~ bind ~
802     { ~
803         dup ~ 6 ~ 29 ~ div ~ ge ~
804         { ~ dup ~ dup ~ mul ~ mul ~ ~ } ~
805         { ~ 4 ~ 29 ~ div ~ sub ~ 108 ~ 841 ~ div ~ mul ~ } ~
806         ifelse ~
807     } ~ bind ~
808     { ~
809         dup ~ 6 ~ 29 ~ div ~ ge ~
810         { ~ dup ~ dup ~ mul ~ mul ~ ~ } ~
811         { ~ 4 ~ 29 ~ div ~ sub ~ 108 ~ 841 ~ div ~ mul ~ } ~
812         ifelse ~
813         1.0890 ~ mul ~
814     } ~ bind
815     ] ~
816     /WhitePoint ~
817     [ ~ \tl_use:c { c__color_model_whitepoint_CIELAB_ #1 _tl } ~ ] ~
818     >>
819 }
820 { \c__color_model_range_CIELAB_tl }
821 { 100 ~ 0 ~ 0 }
822 {#3}
823 }

```

(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.

```

824 \cs_new_protected:Npn \_color_backend_devicen_init:nnn #1#2#3
825 {
826     \_kernel_backend_literal:e
827     {
828         !
829         TeXDict ~ begin ~
830         /color \int_use:N \g__color_model_int
831         {
832             [ ~
833             /DeviceN ~
834             [ ~ #1 ~ ] ~
835             #2 ~
836             { ~ #3 ~ } ~
837             \_color_backend_devicen_colorants:n {#1}
838             ] ~ setcolorspace

```

```

839         } ~ def ~
840     end
841 }
842 }

```

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

`\_color_backend_iccbased_init:nnn` No support at present.

```

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

```

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

```

844 </dvips>
845 <*dvisvgm>

```

`\_color_backend_select_separation:nn` No support at present.

`\_color_backend_select_devicen:nn`

```

846 \cs_new_protected:Npn \_color_backend_select_separation:nn #1#2 { }

```

```

847 \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`

```

848 \cs_new_protected:Npn \_color_backend_separation_init:nnnnn #1#2#3#4#5 { }

```

```

849 \cs_new_protected:Npn \_color_backend_separation_init_CIELAB:nnnnn #1#2#3 { }

```

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

`\_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.

```

850 \cs_new_protected:Npn \_color_backend_select_iccbased:nn #1#2
851 {
852     \_kernel_backend_literal_svg:x
853     {
854         <style>
855             @color-profile ~
856             \str_if_eq:nnTF {#2} { cmyk }
857             { device-cmyk }
858             { --color \int_use:N \g__color_model_int }
859             \c_space_tl
860             {
861                 src:("#1")
862             }
863         </style>
864     }
865 }

```

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

```

866 </dvisvgm>
867 <*dvipdfmx | luatex | pdftex | xetex>

```

```

\__color_backend_select_separation:nn
\__color_backend_select_devicen:nn
\__color_backend_select_iccbased:nn

```

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

```

868 \cs_new_protected:Npn \__color_backend_select_separation:nn #1#2
869 { \__color_backend_select:nn { /#1 ~ cs ~ #2 ~ scn } { /#1 ~ CS ~ #2 ~ SCN } }
870 \cs_new_eq:NN \__color_backend_select_devicen:nn \__color_backend_select_separation:nn
871 \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_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.

```

872 \cs_new_protected:Npn \__color_backend_separation_init:nnnnn #1#2#3#4#5
873 {
874   \pdf_object_unnamed_write:nx { dict }
875   {
876     /FunctionType ~ 2
877     /Domain ~ [0 ~ 1]
878     \tl_if_blank:nF {#3} { /Range ~ [#3] }
879     /C0 ~ [#4] ~
880     /C1 ~ [#5] /N ~ 1
881   }
882   \exp_args:Nx \__color_backend_separation_init:nn
883   { \str_convert_pdfname:n {#1} } {#2}
884   \bool_lazy_and:nnT
885   { \cs_if_exist_p:N \pdfmanagement_if_active_p: }
886   { \pdfmanagement_if_active_p: }
887   {
888     \use:x
889     {
890       \pdfmanagement_add:nnn
891       { Page / Resources / ColorSpace }
892       { color \int_use:N \g__color_model_int }
893       { \pdf_object_ref_last: }
894     }
895   }
896 }
897 \cs_new_protected:Npn \__color_backend_separation_init:nn #1#2
898 {
899   \pdf_object_unnamed_write:nx { array }
900   { /Separation /#1 ~ #2 ~ \pdf_object_ref_last: }
901   \prop_gput:Nnx \g__color_backend_colorant_prop { /#1 }
902   { \pdf_object_ref_last: }
903 }

```

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

```

904 \cs_new_protected:Npn \__color_backend_separation_init_CIELAB:nnn #1#2#3
905 {
906   \pdf_object_if_exist:nF { __color_illuminant_CIELAB_ #1 }
907   {
908     \pdf_object_new:nn { __color_illuminant_CIELAB_ #1 } { array }

```

```

909     \pdf_object_write:nx { __color_illuminant_CIELAB_ #1 }
910     {
911         /Lab ~
912         <<
913         /WhitePoint ~
914         [ \tl_use:c { c__color_model_whitepoint_CIELAB_ #1 _tl } ]
915         /Range ~ [ \c__color_model_range_CIELAB_tl ]
916         >>
917     }
918 }
919 \__color_backend_separation_init:nnnnn
920 {#2}
921 { \pdf_object_ref:n { __color_illuminant_CIELAB_ #1 } }
922 { \c__color_model_range_CIELAB_tl }
923 { 100 ~ 0 ~ 0 }
924 {#3}
925 }

```

(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.

```

926 \cs_new_protected:Npn \__color_backend_devicen_init:nnn #1#2#3
927 {
928     \pdf_object_unnamed_write:nx { stream }
929     {
930         {
931             /FunctionType ~ 4 ~
932             /Domain ~
933             [ ~
934                 \prg_replicate:nn
935                 { 0 \__color_backend_devicen_init:w #1 ~ \s__color_stop }
936                 { 0 ~ 1 ~ }
937             ] ~
938             /Range ~
939             [ ~
940                 \str_case:nn {#2}
941                 {
942                     { /DeviceCMYK } { 0 ~ 1 ~ 0 ~ 1 ~ 0 ~ 1 ~ 0 ~ 1 }
943                     { /DeviceGray } { 0 ~ 1 }
944                     { /DeviceRGB } { 0 ~ 1 ~ 0 ~ 1 ~ 0 ~ 1 }
945                 } ~
946             ]
947         }
948         { {#3} }
949     }
950     \pdf_object_unnamed_write:nx { array }
951     {
952         /DeviceN ~
953         [ ~ #1 ~ ] ~
954         #2 ~
955         \pdf_object_ref_last:
956         \__color_backend_devicen_colorants:n {#1}

```

```

957     }
958     \bool_lazy_and:nnT
959     { \cs_if_exist_p:N \pdfmanagement_if_active_p: }
960     { \pdfmanagement_if_active_p: }
961     {
962         \use:x
963         {
964             \pdfmanagement_add:nnn
965             { Page / Resources / ColorSpace }
966             { color \int_use:N \g__color_model_int }
967             { \pdf_object_ref_last: }
968         }
969     }
970 }
971 \cs_new:Npn \__color_backend_devicen_init:w #1 ~ #2 \s__color_stop
972 {
973     + 1
974     \tl_if_blank:nF {#2}
975     { \__color_backend_devicen_init:w #2 \s__color_stop }
976 }

```

(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.

```

977 \cs_new_protected:Npn \__color_backend_iccbased_init:nnn #1#2#3
978 {
979     \pdf_object_if_exist:nF { __color_icc_ #1 }
980     {
981         \pdf_object_new:nn { __color_icc_ #1 } { fstream }
982         \pdf_object_write:nx { __color_icc_ #1 }
983         {
984             {
985                 /N ~ \exp_not:n { #2 } ~
986                 \tl_if_empty:nF { #3 } { /Range~[ #3 ] }
987             }
988             {#1}
989         }
990     }
991     \pdf_object_unnamed_write:nx { array }
992     { /ICCBased ~ \pdf_object_ref:n { __color_icc_ #1 } }
993     \cs_if_exist:NT \pdfmanagement_add:nnn
994     {
995         \use:x
996         {
997             \pdfmanagement_add:nnn { Page / Resources / ColorSpace }
998             { color \int_use:N \g__color_model_int }
999             { ~ \pdf_object_ref_last: }
1000         }
1001     }
1002 }

```

(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 skip is adding it to the page resources.

```

1003 \cs_new_protected:Npn \__color_backend_iccbased_device:nnn #1#2#3
1004 {
1005   \pdf_object_if_exist:nF { __color_icc_ #1 }
1006   {
1007     \pdf_object_new:nn { __color_icc_ #1 } { fstream }
1008     \pdf_object_write:nn { __color_icc_ #1 }
1009     {
1010       { /N ~ #3 }
1011       {#1}
1012     }
1013   }
1014   \pdf_object_unnamed_write:nx { array }
1015   { /ICCBased ~ \pdf_object_ref:n { __color_icc_ #1 } }
1016   \cs_if_exist:NT \pdfmanagement_add:nnn
1017   {
1018     \use:x
1019     {
1020       \pdfmanagement_add:nnn
1021       { Page / Resources / ColorSpace }
1022       { Default #2 }
1023       { \pdf_object_ref_last: }
1024     }
1025   }
1026 }

```

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

```

1027 </dvipdfmx | luatex | pdftex | xetex>

```

### 3.5 Fill and stroke color

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

```

1028 <*dvipdfmx | luatex | pdftex | xetex>

```

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

```

1029 \cs_new_protected:Npn \__color_backend_fill_cmyk:n #1
1030 { \__color_backend_fill:n { #1 ~ k } }
1031 \cs_new_protected:Npn \__color_backend_fill_gray:n #1
1032 { \__color_backend_fill:n { #1 ~ g } }
1033 \cs_new_protected:Npn \__color_backend_fill_rgb:n #1
1034 { \__color_backend_fill:n { #1 ~ rg } }
1035 \cs_new_protected:Npn \__color_backend_fill:n #1
1036 {
1037   \tl_set:Nn \l__color_backend_fill_tl {#1}
1038   \__kernel_color_backend_stack_push:nn \l__color_backend_stack_int
1039   { #1 ~ \l__color_backend_stroke_tl }
1040   \group_insert_after:N \__color_backend_reset:
1041 }
1042 \cs_new_protected:Npn \__color_backend_stroke_cmyk:n #1

```

```

1043 { \_color_backend_stroke:n { #1 ~ K } }
1044 \cs_new_protected:Npn \_color_backend_stroke_gray:n #1
1045 { \_color_backend_stroke:n { #1 ~ G } }
1046 \cs_new_protected:Npn \_color_backend_stroke_rgb:n #1
1047 { \_color_backend_stroke:n { #1 ~ RG } }
1048 \cs_new_protected:Npn \_color_backend_stroke:n #1
1049 {
1050   \tl_set:Nn \l__color_backend_stroke_tl {#1}
1051   \__kernel_color_backend_stack_push:nn \l__color_backend_stack_int
1052   { \l__color_backend_fill_tl \c_space_tl #1 }
1053   \group_insert_after:N \_color_backend_reset:
1054 }

```

(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
1055 \cs_new_protected:Npn \_color_backend_fill_separation:nn #1#2
1056 { \_color_backend_fill:n { /#1 ~ cs ~ #2 ~ scn } }
1057 \cs_new_protected:Npn \_color_backend_stroke_separation:nn #1#2
1058 { \_color_backend_stroke:n { /#1 ~ CS ~ #2 ~ SCN } }
1059 \cs_new_eq:NN \_color_backend_fill_devicen:nn \_color_backend_fill_separation:nn
1060 \cs_new_eq:NN \_color_backend_stroke_devicen:nn \_color_backend_stroke_separation:nn

```

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

```

1061 </dviptfm | luatex | pdftex | xetex>
1062 <*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
1063 \cs_new_protected:Npn \_color_backend_fill_cmyk:n #1
1064 { \_color_backend_fill:n { cmyk ~ #1 } }
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   \group_insert_after:N \_color_backend_reset:
1073 }
1074 \cs_new_protected:Npn \_color_backend_stroke_cmyk:n #1
1075 { \__kernel_backend_postscript:n { /color.sc { #1 ~ setcmykcolor } def } }
1076 \cs_new_protected:Npn \_color_backend_stroke_gray:n #1
1077 { \__kernel_backend_postscript:n { /color.sc { #1 ~ setgray } def } }
1078 \cs_new_protected:Npn \_color_backend_stroke_rgb:n #1
1079 { \__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
1080 \cs_new_protected:Npn \_color_backend_fill_separation:nn #1#2
1081 { \_color_backend_fill:n { separation ~ #1 ~ #2 } }
1082 \cs_new_protected:Npn \_color_backend_stroke_separation:nn #1#2
1083 { \__kernel_backend_postscript:n { /color.sc { separation ~ #1 ~ #2 } def } }
1084 \cs_new_eq:NN \_color_backend_fill_devicen:nn \_color_backend_fill_separation:nn
1085 \cs_new_eq:NN \_color_backend_stroke_devicen:nn \_color_backend_stroke_separation:nn

```

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

1086 `</dvips>`

1087 `<*dvisvgm>`

`\_color_backend_fill_cmyk:n` Fill color here is the same as general color *except* we skip the stroke part.

```

1088 \cs_new_protected:Npn \_color_backend_fill_cmyk:n #1
1089 { \_color_backend_fill:n { cmyk ~ #1 } }
1090 \cs_new_protected:Npn \_color_backend_fill_gray:n #1
1091 { \_color_backend_fill:n { gray ~ #1 } }
1092 \cs_new_protected:Npn \_color_backend_fill_rgb:n #1
1093 { \_color_backend_fill:n { rgb ~ #1 } }
1094 \cs_new_protected:Npn \_color_backend_fill:n #1
1095 {
1096   \_kernel_backend_literal:n { color~push~ #1 }
1097   \group_insert_after:N \_color_backend_reset:
1098 }

```

(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).

```

1099 \cs_new_protected:Npn \_color_backend_stroke_cmyk:n #1
1100 { \_color_backend_cmyk:w #1 \s_color_stop }
1101 \cs_new_protected:Npn \_color_backend_stroke_cmyk:w
1102 #1 ~ #2 ~ #3 ~ #4 \s_color_stop
1103 {
1104   \use:x
1105   {
1106     \_color_backend:nnn
1107     { \fp_eval:n { -100 * ( 1 - min ( 1 , #1 + #4 ) ) } }
1108     { \fp_eval:n { -100 * ( 1 - min ( 1 , #2 + #4 ) ) } }
1109     { \fp_eval:n { -100 * ( 1 - min ( 1 , #3 + #4 ) ) } }
1110   }
1111 }
1112 \cs_new_protected:Npn \_color_backend_stroke_gray:n #1
1113 {
1114   \use:x
1115   {
1116     \_color_backend_stroke_gray_aux:n
1117     { \fp_eval:n { 100 * (#1) } }
1118   }
1119 }
1120 \cs_new_protected:Npn \_color_backend_stroke_gray_aux:n #1
1121 { \_color_backend:nnn {#1} {#1} {#1} }
1122 \cs_new_protected:Npn \_color_backend_stroke_rgb:n #1
1123 { \_color_backend_rgb:w #1 \s_color_stop }
1124 \cs_new_protected:Npn \_color_backend_stroke_rgb:w
1125 #1 ~ #2 ~ #3 \s_color_stop
1126 {
1127   \use:x
1128   {
1129     \_color_backend:nnn
1130     { \fp_eval:n { 100 * (#1) } }

```

```

1131         { \fp_eval:n { 100 * (#2) } }
1132         { \fp_eval:n { 100 * (#3) } }
1133     }
1134 }
1135 \cs_new_protected:Npx \__color_backend:nnn #1#2#3
1136 {
1137     \__kernel_backend_scope:n
1138     {
1139         stroke =
1140         "
1141         rgb
1142         (
1143             #1 \c_percent_str ,
1144             #2 \c_percent_str ,
1145             #3 \c_percent_str
1146         )
1147         "
1148     }
1149 }

```

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

At present, these are no-ops.

```

\__color_backend_fill_separation:nn 1150 \cs_new_protected:Npn \__color_backend_fill_separation:nn #1#2 { }
\__color_backend_stroke_separation:nn 1151 \cs_new_protected:Npn \__color_backend_stroke_separation:nn #1#2 { }
\__color_backend_fill_devicen:nn 1152 \cs_new_eq:NN \__color_backend_fill_devicen:nn \__color_backend_fill_separation:nn
\__color_backend_stroke_devicen:nn 1153 \cs_new_eq:NN \__color_backend_stroke_devicen:nn \__color_backend_stroke_separation:nn

```

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

No support at present.

```

\__color_backend_devicen_init:nnn 1154 \cs_new_protected:Npn \__color_backend_devicen_init:nnn #1#2#3 { }
\__color_backend_iccbased_init:nnn 1155 \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.)

```

1156 </divisvgm>
1157 </package>

```

## 4 l3backend-draw Implementation

```

1158 <*package>
1159 <@@=draw>

```

### 4.1 dvips backend

```

1160 <*dvips>

```

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

```

\__draw_backend_literal:n 1161 \cs_new_eq:NN \__draw_backend_literal:n \__kernel_backend_literal_postscript:n
\__draw_backend_literal:x 1162 \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.

```

1163 \cs_new_protected:Npn \__draw_backend_begin:
1164 {
1165     \__kernel_backend_literal:n { ps::[begin] }
1166     \__draw_backend_literal:n { @beginspecial }
1167 }
1168 \cs_new_protected:Npn \__draw_backend_end:
1169 {
1170     \__draw_backend_literal:n { @endspecial }
1171     \__kernel_backend_literal:n { ps::[end] }
1172 }

```

(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:`

```

1173 \cs_new_protected:Npn \__draw_backend_scope_begin:
1174 { \__draw_backend_literal:n { save } }
1175 \cs_new_protected:Npn \__draw_backend_scope_end:
1176 { \__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.

```

1177 \cs_new_protected:Npn \__draw_backend_moveto:nn #1#2
1178 {
1179     \__draw_backend_literal:x
1180     {
1181         \dim_to_decimal_in_bp:n {#1} ~
1182         \dim_to_decimal_in_bp:n {#2} ~ moveto
1183     }
1184 }
1185 \cs_new_protected:Npn \__draw_backend_lineto:nn #1#2
1186 {
1187     \__draw_backend_literal:x
1188     {
1189         \dim_to_decimal_in_bp:n {#1} ~
1190         \dim_to_decimal_in_bp:n {#2} ~ lineto
1191     }
1192 }
1193 \cs_new_protected:Npn \__draw_backend_rectangle:nnnn #1#2#3#4

```

```

1194 {
1195   \__draw_backend_literal:x
1196   {
1197     \dim_to_decimal_in_bp:n {#4} ~ \dim_to_decimal_in_bp:n {#3} ~
1198     \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~
1199     moveto~dup~0~rlineto~exch~0~exch~rlineto~neg~0~rlineto~closepath
1200   }
1201 }
1202 \cs_new_protected:Npn \__draw_backend_curveto:nnnnnn #1#2#3#4#5#6
1203 {
1204   \__draw_backend_literal:x
1205   {
1206     \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~
1207     \dim_to_decimal_in_bp:n {#3} ~ \dim_to_decimal_in_bp:n {#4} ~
1208     \dim_to_decimal_in_bp:n {#5} ~ \dim_to_decimal_in_bp:n {#6} ~
1209     curveto
1210   }
1211 }

```

(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
1212 \cs_new_protected:Npn \__draw_backend_evenodd_rule:
1213 { \bool_gset_true:N \g__draw_draw_eor_bool }
1214 \cs_new_protected:Npn \__draw_backend_nonzero_rule:
1215 { \bool_gset_false:N \g__draw_draw_eor_bool }
1216 \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
1217 \cs_new_protected:Npn \__draw_backend_closepath:
1218 { \__draw_backend_literal:n { closepath } }
1219 \cs_new_protected:Npn \__draw_backend_stroke:
1220 {
1221   \__draw_backend_literal:n { gsave }
1222   \__draw_backend_literal:n { color.sc }
1223   \__draw_backend_literal:n { stroke }
1224   \__draw_backend_literal:n { grestore }
1225   \bool_if:NT \g__draw_draw_clip_bool
1226   {
1227     \__draw_backend_literal:x
1228     {
1229       \bool_if:NT \g__draw_draw_eor_bool { eo }
1230       clip
1231     }
1232   }
1233   \__draw_backend_literal:n { newpath }

```

```

1234     \bool_gset_false:N \g__draw_draw_clip_bool
1235   }
1236   \cs_new_protected:Npn \__draw_backend_closestroke:
1237   {
1238     \__draw_backend_closepath:
1239     \__draw_backend_stroke:
1240   }
1241   \cs_new_protected:Npn \__draw_backend_fill:
1242   {
1243     \__draw_backend_literal:x
1244     {
1245       \bool_if:NT \g__draw_draw_eor_bool { eo }
1246       fill
1247     }
1248     \bool_if:NT \g__draw_draw_clip_bool
1249     {
1250       \__draw_backend_literal:x
1251       {
1252         \bool_if:NT \g__draw_draw_eor_bool { eo }
1253         clip
1254       }
1255     }
1256     \__draw_backend_literal:n { newpath }
1257     \bool_gset_false:N \g__draw_draw_clip_bool
1258   }
1259   \cs_new_protected:Npn \__draw_backend_fillstroke:
1260   {
1261     \__draw_backend_literal:x
1262     {
1263       \bool_if:NT \g__draw_draw_eor_bool { eo }
1264       fill
1265     }
1266     \__draw_backend_literal:n { gsave }
1267     \__draw_backend_literal:n { color.sc }
1268     \__draw_backend_literal:n { stroke }
1269     \__draw_backend_literal:n { grestore }
1270     \bool_if:NT \g__draw_draw_clip_bool
1271     {
1272       \__draw_backend_literal:x
1273       {
1274         \bool_if:NT \g__draw_draw_eor_bool { eo }
1275         clip
1276       }
1277     }
1278     \__draw_backend_literal:n { newpath }
1279     \bool_gset_false:N \g__draw_draw_clip_bool
1280   }
1281   \cs_new_protected:Npn \__draw_backend_clip:
1282   { \bool_gset_true:N \g__draw_draw_clip_bool }
1283   \bool_new:N \g__draw_draw_clip_bool
1284   \cs_new_protected:Npn \__draw_backend_discardpath:
1285   {
1286     \bool_if:NT \g__draw_draw_clip_bool
1287     {

```

```

1288     \__draw_backend_literal:x
1289     {
1290         \bool_if:NT \g__draw_draw_eor_bool { eo }
1291         clip
1292     }
1293 }
1294 \__draw_backend_literal:n { newpath }
1295 \bool_gset_false:N \g__draw_draw_clip_bool
1296 }

```

(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:
1297 \cs_new_protected:Npn \__draw_backend_dash_pattern:nn #1#2
1298 {
1299     \__draw_backend_literal:x
1300     {
1301         [
1302             \exp_args:Nf \use:n
1303             { \clist_map_function:nN {#1} \__draw_backend_dash:n }
1304         ] ~
1305         \dim_to_decimal_in_bp:n {#2} ~ setdash
1306     }
1307 }
1308 \cs_new:Npn \__draw_backend_dash:n #1
1309 { ~ \dim_to_decimal_in_bp:n {#1} }
1310 \cs_new_protected:Npn \__draw_backend_linewidth:n #1
1311 {
1312     \__draw_backend_literal:x
1313     { \dim_to_decimal_in_bp:n {#1} ~ setlinewidth }
1314 }
1315 \cs_new_protected:Npn \__draw_backend_miterlimit:n #1
1316 { \__draw_backend_literal:n { #1 ~ setmiterlimit } }
1317 \cs_new_protected:Npn \__draw_backend_cap_but:
1318 { \__draw_backend_literal:n { 0 ~ setlinecap } }
1319 \cs_new_protected:Npn \__draw_backend_cap_round:
1320 { \__draw_backend_literal:n { 1 ~ setlinecap } }
1321 \cs_new_protected:Npn \__draw_backend_cap_rectangle:
1322 { \__draw_backend_literal:n { 2 ~ setlinecap } }
1323 \cs_new_protected:Npn \__draw_backend_join_miter:
1324 { \__draw_backend_literal:n { 0 ~ setlinejoin } }
1325 \cs_new_protected:Npn \__draw_backend_join_round:
1326 { \__draw_backend_literal:n { 1 ~ setlinejoin } }
1327 \cs_new_protected:Npn \__draw_backend_join_bevel:
1328 { \__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.

```

1329 \cs_new_protected:Npn \__draw_backend_cm:nnnn #1#2#3#4
1330 {

```

```

1331   \_draw_backend_literal:n
1332   { [ #1 ~ #2 ~ #3 ~ #4 ~ 0 ~ 0 ] ~ concat }
1333 }

```

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

`\_draw_backend_box_use:Nnnnn`

Inside a picture `@beginspecial/@endspecial` are active, which is normally a good thing but means that the position and scaling would be off if the box was inserted directly. To deal with that, there are a number of possible approaches. The implementation here was suggested by Tom Rokici (author of `dvips`). We end the current special placement, then set the current point with a literal `[begin]`. As for general literals, we then use the stack to store the current point and move to it. To insert the required transformation, we have to flip the  $y$ -axis, once before and once after it. Then we get back to the  $\text{\TeX}$  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).

```

1334 \cs_new_protected:Npn \_draw_backend_box_use:Nnnnn #1#2#3#4#5
1335 {
1336   \_draw_backend_literal:n { @endspecial }
1337   \_draw_backend_literal:n { [end] }
1338   \_draw_backend_literal:n { [begin] }
1339   \_draw_backend_literal:n { save }
1340   \_draw_backend_literal:n { currentpoint }
1341   \_draw_backend_literal:n { currentpoint~translate }
1342   \_draw_backend_cm:nmm { 1 } { 0 } { 0 } { -1 }
1343   \_draw_backend_cm:nmm { #2 } { #3 } { #4 } { #5 }
1344   \_draw_backend_cm:nmm { 1 } { 0 } { 0 } { -1 }
1345   \_draw_backend_literal:n { neg~exch~neg~exch~translate }
1346   \_draw_backend_literal:n { [end] }
1347   \hbox_overlap_right:n { \box_use:N #1 }
1348   \_draw_backend_literal:n { [begin] }
1349   \_draw_backend_literal:n { restore }
1350   \_draw_backend_literal:n { [end] }
1351   \_draw_backend_literal:n { [begin] }
1352   \_draw_backend_literal:n { @beginspecial }
1353 }

```

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

1354 `</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.

1355 `<*dvipdfmx | luatex | pdftex | xetex>`

### 4.2.1 Drawing

`\_draw_backend_literal:n`  
`\_draw_backend_literal:x`

Pass data through using a dedicated interface.

```

1356 \cs_new_eq:NN \_draw_backend_literal:n \_kernel_backend_literal_pdf:n
1357 \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: 1358 \cs_new_protected:Npn \_draw_backend_begin:
                      1359 { \_draw_backend_scope_begin: }
                      1360 \cs_new_protected:Npn \_draw_backend_end:
                      1361 { \_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: 1362 \cs_new_eq:NN \_draw_backend_scope_begin: \_kernel_backend_scope_begin:
                          1363 \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 1364 \cs_new_protected:Npn \_draw_backend_moveto:nn #1#2
\__draw_backend_curveto:nnnnnn 1365 {
\__draw_backend_rectangle:nnnn 1366 \_draw_backend_literal:x
                                1367 { \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~ m }
                                1368 }
                                1369 \cs_new_protected:Npn \_draw_backend_lineto:nn #1#2
                                1370 {
                                1371 \_draw_backend_literal:x
                                1372 { \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~ l }
                                1373 }
                                1374 \cs_new_protected:Npn \_draw_backend_curveto:nnnnnn #1#2#3#4#5#6
                                1375 {
                                1376 \_draw_backend_literal:x
                                1377 {
                                1378 \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~
                                1379 \dim_to_decimal_in_bp:n {#3} ~ \dim_to_decimal_in_bp:n {#4} ~
                                1380 \dim_to_decimal_in_bp:n {#5} ~ \dim_to_decimal_in_bp:n {#6} ~
                                1381 c
                                1382 }
                                1383 }
                                1384 \cs_new_protected:Npn \_draw_backend_rectangle:nnnn #1#2#3#4
                                1385 {
                                1386 \_draw_backend_literal:x
                                1387 {
                                1388 \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~
                                1389 \dim_to_decimal_in_bp:n {#3} ~ \dim_to_decimal_in_bp:n {#4} ~
                                1390 re
                                1391 }
                                1392 }

```

(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: 1393 \cs_new_protected:Npn \_draw_backend_evenodd_rule:
\g__draw_draw_eor_bool 1394 { \bool_gset_true:N \g__draw_draw_eor_bool }
                          1395 \cs_new_protected:Npn \_draw_backend_nonzero_rule:
                          1396 { \bool_gset_false:N \g__draw_draw_eor_bool }
                          1397 \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:
1398 \cs_new_protected:Npn \__draw_backend_closepath:
1399 { \__draw_backend_literal:n { h } }
1400 \cs_new_protected:Npn \__draw_backend_stroke:
1401 { \__draw_backend_literal:n { S } }
1402 \cs_new_protected:Npn \__draw_backend_closestroke:
1403 { \__draw_backend_literal:n { s } }
1404 \cs_new_protected:Npn \__draw_backend_fill:
1405 {
1406   \__draw_backend_literal:x
1407   { f \bool_if:NT \g__draw_draw_eor_bool * }
1408 }
1409 \cs_new_protected:Npn \__draw_backend_fillstroke:
1410 {
1411   \__draw_backend_literal:x
1412   { B \bool_if:NT \g__draw_draw_eor_bool * }
1413 }
1414 \cs_new_protected:Npn \__draw_backend_clip:
1415 {
1416   \__draw_backend_literal:x
1417   { W \bool_if:NT \g__draw_draw_eor_bool * }
1418 }
1419 \cs_new_protected:Npn \__draw_backend_discardpath:
1420 { \__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:
1421 \cs_new_protected:Npn \__draw_backend_dash_pattern:nn #1#2
1422 {
1423   \__draw_backend_literal:x
1424   {
1425     [
1426       \exp_args:Nf \use:n
1427       { \clist_map_function:nN {#1} \__draw_backend_dash:n }
1428     ] ~
1429     \dim_to_decimal_in_bp:n {#2} ~ d
1430   }
1431 }
1432 \cs_new:Npn \__draw_backend_dash:n #1
1433 { ~ \dim_to_decimal_in_bp:n {#1} }
1434 \cs_new_protected:Npn \__draw_backend_linewidth:n #1
1435 {
1436   \__draw_backend_literal:x
1437   { \dim_to_decimal_in_bp:n {#1} ~ w }
1438 }
1439 \cs_new_protected:Npn \__draw_backend_miterlimit:n #1
1440 { \__draw_backend_literal:x { #1 ~ M } }
1441 \cs_new_protected:Npn \__draw_backend_cap_butt:
1442 { \__draw_backend_literal:n { 0 ~ J } }
1443 \cs_new_protected:Npn \__draw_backend_cap_round:

```

```

1444 { \__draw_backend_literal:n { 1 ~ J } }
1445 \cs_new_protected:Npn \__draw_backend_cap_rectangle:
1446 { \__draw_backend_literal:n { 2 ~ J } }
1447 \cs_new_protected:Npn \__draw_backend_join_miter:
1448 { \__draw_backend_literal:n { 0 ~ j } }
1449 \cs_new_protected:Npn \__draw_backend_join_round:
1450 { \__draw_backend_literal:n { 1 ~ j } }
1451 \cs_new_protected:Npn \__draw_backend_join_bevel:
1452 { \__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!

```

1453 \cs_new_protected:Npn \__draw_backend_cm:nnnn #1#2#3#4
1454 {
1455 <| luatex | pdftex >
1456 \__kernel_backend_matrix:n { #1 ~ #2 ~ #3 ~ #4 }
1457 </| luatex | pdftex >
1458 <| dvipdfmx | xetex >
1459 \__draw_backend_cm_decompose:nnnnN {#1} {#2} {#3} {#4}
1460 \__draw_backend_cm_aux:nnnn
1461 </| dvipdfmx | xetex >
1462 }
1463 <| dvipdfmx | xetex >
1464 \cs_new_protected:Npn \__draw_backend_cm_aux:nnnn #1#2#3#4
1465 {
1466 \__kernel_backend_literal:x
1467 {
1468 x:rotate~
1469 \fp_compare:nNnTF {#1} = \c_zero_fp
1470 { 0 }
1471 { \fp_eval:n { round ( -#1 , 5 ) } }
1472 }
1473 \__kernel_backend_literal:x
1474 {
1475 x:scale~
1476 \fp_eval:n { round ( #2 , 5 ) } ~
1477 \fp_eval:n { round ( #3 , 5 ) }
1478 }
1479 \__kernel_backend_literal:x
1480 {
1481 x:rotate~
1482 \fp_compare:nNnTF {#4} = \c_zero_fp
1483 { 0 }
1484 { \fp_eval:n { round ( -#4 , 5 ) } }
1485 }
1486 }
1487 </| 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.

```

1488 <*dvipdfmx | xetex>
1489 \cs_new_protected:Npn \_draw_backend_cm_decompose:nnnnN #1#2#3#4#5
1490 {
1491   \use:x
1492   {
1493     \_draw_backend_cm_decompose_auxi:nnnnN
1494     { \fp_eval:n { (#1 + #4) / 2 } }
1495     { \fp_eval:n { (#1 - #4) / 2 } }
1496     { \fp_eval:n { (#3 + #2) / 2 } }
1497     { \fp_eval:n { (#3 - #2) / 2 } }
1498   }
1499   #5
1500 }
1501 \cs_new_protected:Npn \_draw_backend_cm_decompose_auxi:nnnnN #1#2#3#4#5
1502 {
1503   \use:x
1504   {
1505     \_draw_backend_cm_decompose_auxii:nnnnN
1506     { \fp_eval:n { 2 * sqrt ( #1 * #1 + #4 * #4 ) } }
1507     { \fp_eval:n { 2 * sqrt ( #2 * #2 + #3 * #3 ) } }
1508     { \fp_eval:n { atand ( #3 , #2 ) } }
1509     { \fp_eval:n { atand ( #4 , #1 ) } }
1510   }
1511   #5

```

```

1512 }
1513 \cs_new_protected:Npn \__draw_backend_cm_decompose_auxii:nnnnN #1#2#3#4#5
1514 {
1515   \use:x
1516   {
1517     \__draw_backend_cm_decompose_auxiii:nnnnN
1518     { \fp_eval:n { ( #4 - #3 ) / 2 } }
1519     { \fp_eval:n { ( #1 + #2 ) / 2 } }
1520     { \fp_eval:n { ( #1 - #2 ) / 2 } }
1521     { \fp_eval:n { ( #4 + #3 ) / 2 } }
1522   }
1523   #5
1524 }
1525 \cs_new_protected:Npn \__draw_backend_cm_decompose_auxiii:nnnnN #1#2#3#4#5
1526 {
1527   \fp_compare:nNnTF { abs( #2 ) } > { abs ( #3 ) }
1528   { #5 {#1} {#2} {#3} {#4} }
1529   { #5 {#1} {#3} {#2} {#4} }
1530 }
1531 \</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.

```

1532 \cs_new_protected:Npn \__draw_backend_box_use:Nnnnn #1#2#3#4#5
1533 {
1534   \__kernel_backend_scope_begin:
1535   \< *luatex | pdftex >
1536   \__draw_backend_cm:nnnn {#2} {#3} {#4} {#5}
1537   \< /luatex | pdftex >
1538   \< *dvipdfmx | xetex >
1539   \__kernel_backend_literal:n
1540   { pdf:btrans-matrix~ #2 ~ #3 ~ #4 ~ #5 ~ 0 ~ 0 }
1541   \< /dvipdfmx | xetex >
1542   \hbox_overlap_right:n { \box_use:N #1 }
1543   \< *dvipdfmx | xetex >
1544   \__kernel_backend_literal:n { pdf:etrans }
1545   \< /dvipdfmx | xetex >
1546   \__kernel_backend_scope_end:
1547 }

```

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

```

1548 \< /dvipdfmx | luatex | pdftex | xetex >

```

### 4.3 dvisvgm backend

```

1549 \< *dvisvgm >

```

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

The same as the more general literal call.

```

1550 \cs_new_eq:NN \__draw_backend_literal:n \__kernel_backend_literal_svg:n
1551 \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:` 1552 `\cs_new_eq:NN \_draw_backend_scope_begin: \_kernel_backend_scope_begin:`  
 1553 `\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

1554 `\cs_new_protected:Npn \_draw_backend_begin:`  
 1555 `{`  
 1556 `\_kernel_backend_scope_begin:`  
 1557 `\_kernel_backend_scope:n { transform="translate({?x},{?y})~scale(1,-1)" }`  
 1558 `}`  
 1559 `\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`

1560 `\cs_new_protected:Npn \_draw_backend_moveto:nn #1#2`  
 1561 `{`  
 1562 `\_draw_backend_add_to_path:n`  
 1563 `{ M ~ \dim_to_decimal:n {#1} ~ \dim_to_decimal:n {#2} }`  
 1564 `}`  
 1565 `\cs_new_protected:Npn \_draw_backend_lineto:nn #1#2`  
 1566 `{`  
 1567 `\_draw_backend_add_to_path:n`  
 1568 `{ L ~ \dim_to_decimal:n {#1} ~ \dim_to_decimal:n {#2} }`  
 1569 `}`  
 1570 `\cs_new_protected:Npn \_draw_backend_rectangle:nnnn #1#2#3#4`  
 1571 `{`  
 1572 `\_draw_backend_add_to_path:n`  
 1573 `{`  
 1574 `M ~ \dim_to_decimal:n {#1} ~ \dim_to_decimal:n {#2}`  
 1575 `h ~ \dim_to_decimal:n {#3} ~`  
 1576 `v ~ \dim_to_decimal:n {#4} ~`  
 1577 `h ~ \dim_to_decimal:n { -#3 } ~`  
 1578 `Z`  
 1579 `}`  
 1580 `}`  
 1581 `\cs_new_protected:Npn \_draw_backend_curveto:nnnnnn #1#2#3#4#5#6`  
 1582 `{`  
 1583 `\_draw_backend_add_to_path:n`  
 1584 `{`  
 1585 `C ~`  
 1586 `\dim_to_decimal:n {#1} ~ \dim_to_decimal:n {#2} ~`  
 1587 `\dim_to_decimal:n {#3} ~ \dim_to_decimal:n {#4} ~`  
 1588 `\dim_to_decimal:n {#5} ~ \dim_to_decimal:n {#6}`  
 1589 `}`  
 1590 `}`

```

1590 }
1591 \cs_new_protected:Npn \__draw_backend_add_to_path:n #1
1592 {
1593   \tl_gset:Nx \g__draw_backend_path_tl
1594   {
1595     \g__draw_backend_path_tl
1596     \tl_if_empty:NF \g__draw_backend_path_tl { \c_space_tl }
1597     #1
1598   }
1599 }
1600 \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:
1601 \cs_new_protected:Npn \__draw_backend_evenodd_rule:
1602 { \__kernel_backend_scope:n { fill-rule="evenodd" } }
1603 \cs_new_protected:Npn \__draw_backend_nonzero_rule:
1604 { \__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:
1605 \cs_new_protected:Npn \__draw_backend_closepath:
1606 { \__draw_backend_add_to_path:n { Z } }
\__draw_backend_discardpath:
1607 \cs_new_protected:Npn \__draw_backend_path:n #1
1608 {
1609   \bool_if:NTF \g__draw_draw_clip_bool
1610   {
1611     \int_gincr:N \g__kernel_clip_path_int
1612     \__draw_backend_literal:x
1613     {
1614       < clipPath~id = " l3cp \int_use:N \g__kernel_clip_path_int " >
1615       { ?nl }
1616       <path~d=" \g__draw_backend_path_tl "> { ?nl }
1617       < /clipPath > { ? nl }
1618       <
1619       use~xlink:href =
1620       "\c_hash_str l3path \int_use:N \g__draw_backend_path_int " ~
1621       #1
1622     } />
1623   }
1624   \__kernel_backend_scope:x
1625   {
1626     clip-path =
1627     "url( \c_hash_str l3cp \int_use:N \g__kernel_clip_path_int )"
1628   }
1629 }
1630 {
1631   \__draw_backend_literal:x

```

```

1632         { <path ~ d=" \g__draw_backend_path_tl " ~ #1 /> }
1633     }
1634     \tl_gclear:N \g__draw_backend_path_tl
1635     \bool_gset_false:N \g__draw_draw_clip_bool
1636 }
1637 \int_new:N \g__draw_backend_path_int
1638 \cs_new_protected:Npn \__draw_backend_stroke:
1639 { \__draw_backend_path:n { style="fill:none" } }
1640 \cs_new_protected:Npn \__draw_backend_closestroke:
1641 {
1642     \__draw_backend_closepath:
1643     \__draw_backend_stroke:
1644 }
1645 \cs_new_protected:Npn \__draw_backend_fill:
1646 { \__draw_backend_path:n { style="stroke:none" } }
1647 \cs_new_protected:Npn \__draw_backend_fillstroke:
1648 { \__draw_backend_path:n { } }
1649 \cs_new_protected:Npn \__draw_backend_clip:
1650 { \bool_gset_true:N \g__draw_draw_clip_bool }
1651 \bool_new:N \g__draw_draw_clip_bool
1652 \cs_new_protected:Npn \__draw_backend_discardpath:
1653 {
1654     \bool_if:NT \g__draw_draw_clip_bool
1655     {
1656         \int_gincr:N \g__kernel_clip_path_int
1657         \__draw_backend_literal:x
1658         {
1659             < clipPath~id = " l3cp \int_use:N \g__kernel_clip_path_int " >
1660             { ?nl }
1661             <path~d=" \g__draw_backend_path_tl "/> { ?nl }
1662             < /clipPath >
1663         }
1664         \__kernel_backend_scope:x
1665         {
1666             clip-path =
1667             "url( \c_hash_str l3cp \int_use:N \g__kernel_clip_path_int)"
1668         }
1669     }
1670     \tl_gclear:N \g__draw_path_tl
1671     \bool_gset_false:N \g__draw_draw_clip_bool
1672 }

```

(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_butt: \__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> 1673 \cs_new_protected:Npn \__draw_backend_dash_pattern:nn #1#2 1674 { 1675     \use:x 1676     { 1677         \__draw_backend_dash_aux:nn 1678         { \clist_map_function:nn {#1} \__draw_backend_dash:n } 1679         { \dim_to_decimal:n {#2} } 1680     } </pre>
--	--

```

1681 }
1682 \cs_new:Npn \__draw_backend_dash:n #1
1683 { , \dim_to_decimal_in_bp:n {#1} }
1684 \cs_new_protected:Npn \__draw_backend_dash_aux:nn #1#2
1685 {
1686   \__kernel_backend_scope:x
1687   {
1688     stroke-dasharray =
1689     "
1690     \tl_if_empty:nTF {#1}
1691     { none }
1692     { \use_none:n #1 }
1693     " ~
1694     stroke-offset=" #2 "
1695   }
1696 }
1697 \cs_new_protected:Npn \__draw_backend_linewidth:n #1
1698 { \__kernel_backend_scope:x { stroke-width=" \dim_to_decimal:n {#1} " } }
1699 \cs_new_protected:Npn \__draw_backend_miterlimit:n #1
1700 { \__kernel_backend_scope:x { stroke-miterlimit=" #1 " } }
1701 \cs_new_protected:Npn \__draw_backend_cap_but:
1702 { \__kernel_backend_scope:n { stroke-linecap="butt" } }
1703 \cs_new_protected:Npn \__draw_backend_cap_round:
1704 { \__kernel_backend_scope:n { stroke-linecap="round" } }
1705 \cs_new_protected:Npn \__draw_backend_cap_rectangle:
1706 { \__kernel_backend_scope:n { stroke-linecap="square" } }
1707 \cs_new_protected:Npn \__draw_backend_join_miter:
1708 { \__kernel_backend_scope:n { stroke-linejoin="miter" } }
1709 \cs_new_protected:Npn \__draw_backend_join_round:
1710 { \__kernel_backend_scope:n { stroke-linejoin="round" } }
1711 \cs_new_protected:Npn \__draw_backend_join_bevel:
1712 { \__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.

```

1713 \cs_new_protected:Npn \__draw_backend_cm:nnnn #1#2#3#4
1714 {
1715   \__kernel_backend_scope:n
1716   {
1717     transform =
1718     " matrix ( #1 , #2 , #3 , #4 , Opt , Opt ) "
1719   }
1720 }

```

(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.

```

1721 \cs_new_protected:Npn \__draw_backend_box_use:Nnnnn #1#2#3#4#5
1722 {
1723   \__kernel_backend_scope_begin:
1724   \__draw_backend_cm:nnnn {#2} {#3} {#4} {#5}

```

```

1725 \__kernel_backend_literal_svg:n
1726 {
1727   < g~
1728     stroke="none"~
1729     transform="scale(-1,1)~translate({?x},{?y})~scale(-1,-1)"
1730   >
1731 }
1732 \box_set_wd:Nn #1 { Opt }
1733 \box_set_ht:Nn #1 { Opt }
1734 \box_set_dp:Nn #1 { Opt }
1735 \box_use:N #1
1736 \__kernel_backend_literal_svg:n { </g> }
1737 \__kernel_backend_scope_end:
1738 }

```

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

```
1739 </dvisvgm>
```

```
1740 </package>
```

## 5 l3backend-graphics Implementation

```

1741 <*package>
1742 <@@=graphics>

```

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

```

1743 \cs_new_protected:Npn \__graphics_backend_loaded:n #1
1744 {
1745   \cs_if_exist:NTF \hook_gput_code:nnn
1746   {
1747     \hook_gput_code:nnn
1748     { file / l3graphics.sty / after }
1749     { backend }
1750     {#1}
1751   }
1752   {#1}
1753 }

```

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

### 5.1 dvips backend

```
1754 <*dvips>
```

\l\_graphics\_search\_ext\_seq

```

1755 \__graphics_backend_loaded:n
1756 { \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

```

1757 \__graphics_backend_loaded:n
1758 {
1759   \cs_new_eq:NN \__graphics_backend_getbb_eps:n \__graphics_read_bb:n
1760   \cs_new_eq:NN \__graphics_backend_getbb_ps:n \__graphics_read_bb:n
1761 }

```

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

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

```

1762 \cs_new_protected:Npn \_graphics_backend_include_eps:n #1
1763 {
1764   \_kernel_backend_literal:x
1765   {
1766     PSfile = #1 \c_space_tl
1767     llx = \dim_to_decimal_in_bp:n \l__graphics_llx_dim \c_space_tl
1768     lly = \dim_to_decimal_in_bp:n \l__graphics_lly_dim \c_space_tl
1769     urx = \dim_to_decimal_in_bp:n \l__graphics_urx_dim \c_space_tl
1770     ury = \dim_to_decimal_in_bp:n \l__graphics_ury_dim
1771   }
1772 }
1773 \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`

```

1774 \_graphics_backend_loaded:n
1775 { \cs_new_eq:NN \_graphics_backend_get_pagecount:n \_graphics_get_pagecount:n }

```

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

```

1776 </dvips>

```

## 5.2 LuaTeX and pdfTeX backends

```

1777 < *luatex | pdftex >

```

`\l_graphics_search_ext_seq`

```

1778 \_graphics_backend_loaded:n
1779 {
1780   \seq_set_from_clist:Nn
1781   \l_graphics_search_ext_seq
1782   { .pdf , .eps , .png , .jpg , .jpeg }
1783 }

```

(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.

```

1784 \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.

```

1785 \cs_new_protected:Npn \_graphics_backend_getbb_jpg:n #1

```

```

1786 {
1787   \int_zero:N \l__graphics_page_int
1788   \tl_clear:N \l__graphics_pagebox_tl
1789   \tl_set:Nx \l__graphics_graphics_attr_tl
1790   {
1791     \tl_if_empty:NF \l__graphics_decodearray_str
1792     { :D \l__graphics_decodearray_str }
1793     \bool_if:NT \l__graphics_interpolate_bool
1794     { :I }
1795   }
1796   \tl_clear:N \l__graphics_graphics_attr_tl
1797   \__graphics_backend_getbb_auxi:n {#1}
1798 }
1799 \cs_new_eq:NN \__graphics_backend_getbb_jpeg:n \__graphics_backend_getbb_jpg:n
1800 \cs_new_eq:NN \__graphics_backend_getbb_png:n \__graphics_backend_getbb_jpg:n
1801 \cs_new_protected:Npn \__graphics_backend_getbb_pdf:n #1
1802 {
1803   \tl_clear:N \l__graphics_decodearray_str
1804   \bool_set_false:N \l__graphics_interpolate_bool
1805   \tl_set:Nx \l__graphics_graphics_attr_tl
1806   {
1807     : \l__graphics_pagebox_tl
1808     \int_compare:nNnT \l__graphics_page_int > 1
1809     { :P \int_use:N \l__graphics_page_int }
1810   }
1811   \__graphics_backend_getbb_auxi:n {#1}
1812 }
1813 \cs_new_protected:Npn \__graphics_backend_getbb_auxi:n #1
1814 {
1815   \__graphics_bb_restore:xF { #1 \l__graphics_graphics_attr_tl }
1816   { \__graphics_backend_getbb_auxii:n {#1} }
1817 }

```

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.

```

1818 \cs_new_protected:Npn \__graphics_backend_getbb_auxii:n #1
1819 {
1820   \exp_args:Ne \__graphics_backend_getbb_auxiii:n
1821   { \__graphics_backend_dequote:w #1 " #1 " \s__graphics_stop }
1822   \int_const:cn { c__graphics_ #1 \l__graphics_graphics_attr_tl _int }
1823   { \tex_the:D \tex_pdflastximage:D }
1824   \__graphics_bb_save:x { #1 \l__graphics_graphics_attr_tl }
1825 }
1826 \cs_new_protected:Npn \__graphics_backend_getbb_auxiii:n #1
1827 {
1828   \tex_immediate:D \tex_pdfximage:D
1829   \bool_lazy_or:nnT
1830   { \l__graphics_interpolate_bool }
1831   { ! \tl_if_empty_p:N \l__graphics_decodearray_str }
1832   {
1833     attr ~
1834     {

```

```

1835         \tl_if_empty:NF \l__graphics_decodearray_str
1836         { /Decode~[ \l__graphics_decodearray_str ] }
1837         \bool_if:NT \l__graphics_interpolate_bool
1838         { /Interpolate~true }
1839     }
1840 }
1841 \int_compare:nNnT \l__graphics_page_int > 0
1842 { page ~ \int_use:N \l__graphics_page_int }
1843 \tl_if_empty:NF \l__graphics_pagebox_tl
1844 { \l__graphics_pagebox_tl }
1845 {#1}
1846 \hbox_set:Nn \l__graphics_internal_box
1847 { \tex_pdfrefximage:D \tex_pdflastximage:D }
1848 \dim_set:Nn \l__graphics_urx_dim { \box_wd:N \l__graphics_internal_box }
1849 \dim_set:Nn \l__graphics_ury_dim { \box_ht:N \l__graphics_internal_box }
1850 }
1851 \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.

```

1852 \cs_new_protected:Npn \__graphics_backend_include_jpg:n #1
1853 {
1854     \tex_pdfrefximage:D
1855     \int_use:c { c__graphics_ #1 \l__graphics_graphics_attr_tl _int }
1856 }
1857 \cs_new_eq:NN \__graphics_backend_include_jpeg:n \__graphics_backend_include_jpg:n
1858 \cs_new_eq:NN \__graphics_backend_include_pdf:n \__graphics_backend_include_jpg:n
1859 \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\_eps:nm  
\\_\_graphics\_backend\_include\_eps: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<sub>2 $\epsilon$</sub>  package, but simplified, conversion takes place here if we have shell access.

```

1860 \sys_if_shell:T
1861 {
1862     \str_new:N \l__graphics_backend_dir_str
1863     \str_new:N \l__graphics_backend_name_str
1864     \str_new:N \l__graphics_backend_ext_str
1865     \cs_new_protected:Npn \__graphics_backend_getbb_eps:n #1
1866     {
1867         \file_parse_full_name:nNNN {#1}
1868         \l__graphics_backend_dir_str
1869         \l__graphics_backend_name_str
1870         \l__graphics_backend_ext_str
1871         \exp_args:Nx \__graphics_backend_getbb_eps:nm
1872         {
1873             \exp_args:Ne \__kernel_file_name_quote:n
1874             {
1875                 \l__graphics_backend_name_str
1876                 - \str_tail:N \l__graphics_backend_ext_str

```

```

1877         -converted-to.pdf
1878     }
1879 }
1880 {#1}
1881 }
1882 \cs_new_protected:Npn \__graphics_backend_getbb_eps:nn #1#2
1883 {
1884     \file_compare_timestamp:nNnT {#2} > {#1}
1885     {
1886         \sys_shell_now:n
1887         { repstopdf ~ #2 ~ #1 }
1888     }
1889     \tl_set:Nn \l__graphics_final_name_str {#1}
1890     \__graphics_backend_getbb_pdf:n {#1}
1891 }
1892 \cs_new_protected:Npn \__graphics_backend_include_eps:n #1
1893 {
1894     \file_parse_full_name:nNNN {#1}
1895     \l__graphics_backend_dir_str \l__graphics_backend_name_str \l__graphics_backend_ext_str
1896     \exp_args:Nx \__graphics_backend_include_pdf:n
1897     {
1898         \exp_args:Ne \__kernel_file_name_quote:n
1899         {
1900             \l__graphics_backend_name_str
1901             - \str_tail:N \l__graphics_backend_ext_str
1902             -converted-to.pdf
1903         }
1904     }
1905 }
1906 }

```

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

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

```

1907 \cs_new_protected:Npn \__graphics_backend_get_pagecount:n #1
1908 {
1909     \tex_immediate:D \tex_pdfximage:D {#1}
1910     \int_const:cn { c__graphics_ #1 _pages_int }
1911     { \int_use:N \tex_pdflastximagepages:D }
1912 }

```

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

1913 </luatex | pdftex>

### 5.3 dvipdfmx backend

1914 <\*dvipdfmx | xetex>

\l\_graphics\_search\_ext\_seq

```

1915 \__graphics_backend_loaded:n
1916 {
1917     \seq_set_from_clist:Nn \l_graphics_search_ext_seq
1918     { .pdf , .eps , .ps , .png , .jpg ., jpeg , .bmp }
1919 }

```

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

Simply use the generic functions: only for dvipdfmx in the extraction cases.

```

\__graphics_backend_getbb_eps:n
\__graphics_backend_getbb_jpg:n
\__graphics_backend_getbb_jpeg:n
\__graphics_backend_getbb_pdf:n
\__graphics_backend_getbb_png:n
\__graphics_backend_getbb_bmp:n
1920 \__graphics_backend_loaded:n
1921 {
1922   \cs_new_eq:NN \__graphics_backend_getbb_eps:n \__graphics_read_bb:n
1923   \cs_new_eq:NN \__graphics_backend_getbb_ps:n \__graphics_read_bb:n
1924 }
1925 <*dvipdfmx>
1926 \cs_new_protected:Npn \__graphics_backend_getbb_jpg:n #1
1927 {
1928   \int_zero:N \l__graphics_page_int
1929   \tl_clear:N \l__graphics_pagebox_tl
1930   \__graphics_extract_bb:n {#1}
1931 }
1932 \cs_new_eq:NN \__graphics_backend_getbb_jpeg:n \__graphics_backend_getbb_jpg:n
1933 \cs_new_eq:NN \__graphics_backend_getbb_png:n \__graphics_backend_getbb_jpg:n
1934 \cs_new_eq:NN \__graphics_backend_getbb_bmp:n \__graphics_backend_getbb_jpg:n
1935 \cs_new_protected:Npn \__graphics_backend_getbb_pdf:n #1
1936 {
1937   \tl_clear:N \l__graphics_decodearray_str
1938   \bool_set_false:N \l__graphics_interpolate_bool
1939   \__graphics_extract_bb:n {#1}
1940 }
1941 </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.

```

1942 \int_new:N \g__graphics_track_int

```

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

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

```

\__graphics_backend_include_eps:n
\__graphics_backend_include_ps:n
\__graphics_backend_include_jpg:n
\__graphics_backend_include_jpseg:n
\__graphics_backend_include_pdf:n
\__graphics_backend_include_png:n
\__graphics_backend_include_bmp:n
\__graphics_backend_include_auxi:nn
\__graphics_backend_include_auxii:nnn
\__graphics_backend_include_auxiii:nnnn
1943 \cs_new_protected:Npn \__graphics_backend_include_eps:n #1
1944 {
1945   \__kernel_backend_literal:x
1946   {
1947     PSfile = #1 \c_space_tl
1948     llx = \dim_to_decimal_in_bp:n \l__graphics_llx_dim \c_space_tl
1949     lly = \dim_to_decimal_in_bp:n \l__graphics_lly_dim \c_space_tl
1950     urx = \dim_to_decimal_in_bp:n \l__graphics_urx_dim \c_space_tl
1951     ury = \dim_to_decimal_in_bp:n \l__graphics_ury_dim
1952   }
1953 }
1954 \cs_new_eq:NN \__graphics_backend_include_ps:n \__graphics_backend_include_eps:n
1955 \cs_new_protected:Npn \__graphics_backend_include_jpg:n #1
1956 { \__graphics_backend_include_auxi:nn {#1} { image } }
1957 \cs_new_eq:NN \__graphics_backend_include_jpeg:n \__graphics_backend_include_jpg:n
1958 \cs_new_eq:NN \__graphics_backend_include_png:n \__graphics_backend_include_jpg:n
1959 \cs_new_eq:NN \__graphics_backend_include_bmp:n \__graphics_backend_include_jpg:n
1960 <*dvipdfmx>

```

```

1961 \cs_new_protected:Npn \__graphics_backend_include_pdf:n #1
1962 { \__graphics_backend_include_auxi:nn {#1} { epdf } }
1963 </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.

```

1964 \cs_new_protected:Npn \__graphics_backend_include_auxi:nn #1#2
1965 {
1966   \__graphics_backend_include_auxii:xnn
1967   {
1968     \tl_if_empty:NF \l__graphics_pagebox_tl
1969     { : \l__graphics_pagebox_tl }
1970     \int_compare:nNnT \l__graphics_page_int > 1
1971     { :P \int_use:N \l__graphics_page_int }
1972     \tl_if_empty:NF \l__graphics_decodearray_str
1973     { :D \l__graphics_decodearray_str }
1974     \bool_if:NT \l__graphics_interpolate_bool
1975     { :I }
1976   }
1977   {#1} {#2}
1978 }
1979 \cs_new_protected:Npn \__graphics_backend_include_auxii:nnn #1#2#3
1980 {
1981   \int_if_exist:cTF { c__graphics_ #2#1 _int }
1982   {
1983     \__kernel_backend_literal:x
1984     { pdf:usexobj~@graphic \int_use:c { c__graphics_ #2#1 _int } }
1985   }
1986   { \__graphics_backend_include_auxiii:nnn {#2} {#1} {#3} }
1987 }
1988 \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!

```

1989 \cs_new_protected:Npn \__graphics_backend_include_auxiii:nnn #1#2#3
1990 {
1991   \int_gincr:N \g__graphics_track_int
1992   \int_const:cn { c__graphics_ #1#2 _int } { \g__graphics_track_int }
1993   \__kernel_backend_literal:x
1994   {
1995     pdf:#3~
1996     @graphic \int_use:c { c__graphics_ #1#2 _int } ~
1997     \int_compare:nNnT \l__graphics_page_int > 1
1998     { page ~ \int_use:N \l__graphics_page_int \c_space_tl }
1999     \tl_if_empty:NF \l__graphics_pagebox_tl
2000     {
2001       pagebox ~ \l__graphics_pagebox_tl \c_space_tl
2002       bbox ~
2003       \dim_to_decimal_in_bp:n \l__graphics_llx_dim \c_space_tl
2004       \dim_to_decimal_in_bp:n \l__graphics_lly_dim \c_space_tl
2005       \dim_to_decimal_in_bp:n \l__graphics_urx_dim \c_space_tl

```

```

2006         \dim_to_decimal_in_bp:n \l__graphics_ury_dim \c_space_tl
2007     }
2008     (#1)
2009     \bool_lazy_or:nnT
2010     { \l__graphics_interpolate_bool }
2011     { ! \tl_if_empty_p:N \l__graphics_decodearray_str }
2012     {
2013         <<
2014         \tl_if_empty:NF \l__graphics_decodearray_str
2015         { /Decode~[ \l__graphics_decodearray_str ] }
2016         \bool_if:NT \l__graphics_interpolate_bool
2017         { /Interpolate~true> }
2018     }
2019     >>
2020 }
2021 }

```

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

`\__graphics_backend_get_pagecount:n`

```

2022 <*\dvipdfmx>
2023 \__graphics_backend_loaded:n
2024 { \cs_new_eq:NN \__graphics_backend_get_pagecount:n \__graphics_get_pagecount:n }
2025 </dvipdfmx>

```

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

```
2026 </dvipdfmx | xetex>
```

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

```
2027 <*\xetex>
```

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

```

2028 \cs_new_protected:Npn \__graphics_backend_getbb_jpg:n #1
2029 {
2030     \int_zero:N \l__graphics_page_int
2031     \tl_clear:N \l__graphics_pagebox_tl
2032     \__graphics_backend_getbb_auxi:nN {#1} \tex_XeTeXpicfile:D
2033 }
2034 \cs_new_eq:NN \__graphics_backend_getbb_jpeg:n \__graphics_backend_getbb_jpg:n
2035 \cs_new_eq:NN \__graphics_backend_getbb_png:n \__graphics_backend_getbb_jpg:n
2036 \cs_new_eq:NN \__graphics_backend_getbb_bmp:n \__graphics_backend_getbb_jpg:n
2037 \cs_new_protected:Npn \__graphics_backend_getbb_pdf:n #1
2038 {
2039     \tl_clear:N \l__graphics_decodearray_str
2040     \bool_set_false:N \l__graphics_interpolate_bool
2041     \__graphics_backend_getbb_auxi:nN {#1} \tex_XeTeXpdffile:D
2042 }
2043 \cs_new_protected:Npn \__graphics_backend_getbb_auxi:nN #1#2
2044 {
2045     \int_compare:nNnTF \l__graphics_page_int > 1

```

```

2046     { \_graphics_backend_getbb_auxii:VnN \l\_graphics_page_int {#1} #2 }
2047     { \_graphics_backend_getbb_auxiii:nNnn {#1} #2 { :P 1 } { page 1 } }
2048   }
2049   \cs_new_protected:Npn \_graphics_backend_getbb_auxii:nnN #1#2#3
2050   { \_graphics_backend_getbb_auxiii:nNnn {#2} #3 { :P #1 } { page #1 } }
2051   \cs_generate_variant:Nn \_graphics_backend_getbb_auxii:nnN { V }
2052   \cs_new_protected:Npn \_graphics_backend_getbb_auxiii:nNnn #1#2#3#4
2053   {
2054     \tl_if_empty:NTF \l\_graphics_pagebox_tl
2055     { \_graphics_backend_getbb_auxiv:VnNnn \l\_graphics_pagebox_tl }
2056     { \_graphics_backend_getbb_auxv:nNnn }
2057     {#1} #2 {#3} {#4}
2058   }
2059   \cs_new_protected:Npn \_graphics_backend_getbb_auxiv:nnNnn #1#2#3#4#5
2060   {
2061     \use:x
2062     {
2063       \_graphics_backend_getbb_auxv:nNnn {#2} #3 { : #1 #4 }
2064       {
2065         #5
2066         \tl_if_blank:nF {#1}
2067         { \c_space_tl \_graphics_backend_getbb_pagebox:w #1 }
2068       }
2069     }
2070   }
2071   \cs_generate_variant:Nn \_graphics_backend_getbb_auxiv:nnNnn { V }
2072   \cs_new_protected:Npn \_graphics_backend_getbb_auxv:nNnn #1#2#3#4
2073   {
2074     \_graphics_bb_restore:nF {#1#3}
2075     { \_graphics_backend_getbb_auxvi:nNnn {#1} #2 {#3} {#4} }
2076   }
2077   \cs_new_protected:Npn \_graphics_backend_getbb_auxvi:nNnn #1#2#3#4
2078   {
2079     \hbox_set:Nn \l\_graphics_internal_box { #2 #1 ~ #4 }
2080     \dim_set:Nn \l\_graphics_urx_dim { \box_wd:N \l\_graphics_internal_box }
2081     \dim_set:Nn \l\_graphics_ury_dim { \box_ht:N \l\_graphics_internal_box }
2082     \_graphics_bb_save:n {#1#3}
2083   }
2084   \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\_XeTeXpdfnfile: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.

```

2085   \cs_new_protected:Npn \_graphics_backend_include_pdf:n #1
2086   {
2087     \tex_XeTeXpdfnfile:D #1 ~
2088     \int_compare:nNnT \l\_graphics_page_int > 0
2089     { page ~ \int_use:N \l\_graphics_page_int \c_space_tl }
2090     \exp_after:wN \_graphics_backend_getbb_pagebox:w \l\_graphics_pagebox_tl
2091   }

```

(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.

```
2092 \cs_new_protected:Npn \_graphics_backend_get_pagecount:n #1
2093 {
2094   \int_const:cn { c__graphics_ #1 _pages_int }
2095   {
2096     \int_max:nn
2097     { \int_use:N \tex_XeTeXpdfpagecount:D #1 ~ }
2098     { 1 }
2099   }
2100 }
```

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

```
2101 \</xetex>
```

## 5.5 dvisvgm backend

```
2102 \*dvisvgm>
```

`\l_graphics_search_ext_seq`

```
2103 \_graphics_backend_loaded:n
2104 {
2105   \seq_set_from_clist:Nn
2106   \l_graphics_search_ext_seq
2107   { .pdf , .eps , .png , .jpg , .jpeg }
2108 }
```

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

`\_graphics_backend_getbb_eps:n`

Simply use the generic function.

```
2109 \_graphics_backend_loaded:n
2110 { \cs_new_eq:NN \_graphics_backend_getbb_eps:n \_graphics_read_bb:n }
```

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

`\_graphics_backend_getbb_png:n`

These can be included by extracting the bounding box data.

`\_graphics_backend_getbb_jpg:n`

`\_graphics_backend_getbb_jpeg:n`

```
2111 \cs_new_protected:Npn \_graphics_backend_getbb_jpg:n #1
2112 {
2113   \int_zero:N \l__graphics_page_int
2114   \tl_clear:N \l__graphics_pagebox_tl
2115   \_graphics_extract_bb:n {#1}
2116 }
2117 \cs_new_eq:NN \_graphics_backend_getbb_jpeg:n \_graphics_backend_getbb_jpg:n
2118 \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

```
2119 \cs_new_protected:Npn \_graphics_backend_getbb_pdf:n #1
2120 {
2121   \tl_clear:N \l__graphics_decodearray_str
2122   \bool_set_false:N \l__graphics_interpolate_bool
2123   \_graphics_extract_bb:n {#1}
2124 }
```

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

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

```

2125 \cs_new_protected:Npn \_graphics_backend_include_eps:n #1
2126 { \_graphics_backend_include:nn { PSfile } {#1} }
2127 \cs_new_protected:Npn \_graphics_backend_include_pdf:n #1
2128 { \_graphics_backend_include:nn { pdffile } {#1} }
2129 \cs_new_protected:Npn \_graphics_backend_include:nn #1#2
2130 {
2131   \_kernel_backend_literal:x
2132   {
2133     #1 = #2 \c_space_tl
2134     llx = \dim_to_decimal_in_bp:n \l__graphics_llx_dim \c_space_tl
2135     lly = \dim_to_decimal_in_bp:n \l__graphics_lly_dim \c_space_tl
2136     urx = \dim_to_decimal_in_bp:n \l__graphics_urx_dim \c_space_tl
2137     ury = \dim_to_decimal_in_bp:n \l__graphics_ury_dim
2138   }
2139 }

```

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

`\_graphics_backend_include_png:n`  
`\_graphics_backend_include_jpg:n`  
`\_graphics_backend_include_jpeg:n`  
`\_graphics_backend_include_bitmap_quote: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). The only issue is that `#1` must be quote-corrected. The `dvisvgm:img` operation quotes the file name, but if it is already quoted (contains spaces) then we have an issue: we simply strip off any quotes as a result.

```

2140 \cs_new_protected:Npn \_graphics_backend_include_png:n #1
2141 {
2142   \_kernel_backend_literal:x
2143   {
2144     dvisvgm:img~
2145     \dim_to_decimal:n { \l__graphics_ury_dim } ~
2146     \dim_to_decimal:n { \l__graphics_ury_dim } ~
2147     \_graphics_backend_include_bitmap_quote:w #1 " #1 " \s__graphics_stop
2148   }
2149 }
2150 \cs_new_eq:NN \_graphics_backend_include_jpeg:n \_graphics_backend_include_png:n
2151 \cs_new_eq:NN \_graphics_backend_include_jpg:n \_graphics_backend_include_png:n
2152 \cs_new:Npn \_graphics_backend_include_bitmap_quote:w #1 " #2 " #3 \s__graphics_stop
2153 { " #2 " }

```

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

`\_graphics_backend_get_pagecount:n`

```

2154 \_graphics_backend_loaded:n
2155 { \cs_new_eq:NN \_graphics_backend_get_pagecount:n \_graphics_get_pagecount:n }

```

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

```

2156 </dvisvgm>
2157 </package>

```

## 6 l3backend-pdf Implementation

```
2158 ⟨*package⟩
2159 ⟨@@=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
2160 \box_new:N \l__pdf_internal_box
(End definition for \l__pdf_internal_box.)
```

### 6.2 dvips backend

```
2161 ⟨*dvips⟩
Used often enough it should be a separate function.
\__pdf_backend_pdfmark:n
\__pdf_backend_pdfmark:x
2162 \cs_new_protected:Npn \__pdf_backend_pdfmark:n #1
2163 { \__kernel_backend_postscript:n { mark #1 ~ pdfmark } }
2164 \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
2165 \cs_new_protected:Npn \__pdf_backend_catalog_gput:nn #1#2
2166 { \__pdf_backend_pdfmark:n { { Catalog } << /#1 ~ #2 >> /PUT } }
2167 \cs_new_protected:Npn \__pdf_backend_info_gput:nn #1#2
2168 { \__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
\g__pdf_backend_object_prop
2169 \int_new:N \g__pdf_backend_object_int
2170 \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
\__pdf_backend_object_ref:n
Tracking objects is similar to dvipdfmx.
2171 \cs_new_protected:Npn \__pdf_backend_object_new:nn #1#2
2172 {
2173   \int_gincr:N \g__pdf_backend_object_int
2174   \int_const:cn
2175   { c__pdf_backend_object_ \tl_to_str:n {#1} _int }
2176   { \g__pdf_backend_object_int }
```

```

2177     \prop_gput:Nnn \g__pdf_backend_object_prop {#1} {#2}
2178   }
2179   \cs_new:Npn \__pdf_backend_object_ref:n #1
2180     { { pdf.obj \int_use:c { c__pdf_backend_object_ \tl_to_str:n {#1} _int } } }

```

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

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

```

\__pdf_backend_object_write:nn
\__pdf_backend_object_write:nx
\__pdf_backend_object_write_array:nn
\__pdf_backend_object_write_dict:nn
\__pdf_backend_object_write_fstream:nn
\__pdf_backend_object_write_stream:nn
\__pdf_backend_object_write_stream:nnn

2181 \cs_new_protected:Npn \__pdf_backend_object_write:nn #1#2
2182 {
2183   \__pdf_backend_pdfmark:x
2184   {
2185     /objdef ~ \__pdf_backend_object_ref:n {#1}
2186     /type
2187     \str_case_e:nn
2188       { \prop_item:Nn \g__pdf_backend_object_prop {#1} }
2189       {
2190         { array } { /array }
2191         { dict } { /dict }
2192         { fstream } { /stream }
2193         { stream } { /stream }
2194       }
2195     /OBJ
2196   }
2197   \use:c
2198     { \__pdf_backend_object_write_ \prop_item:Nn \g__pdf_backend_object_prop {#1} :nn }
2199     { \__pdf_backend_object_ref:n {#1} } {#2}
2200 }
2201 \cs_generate_variant:Nn \__pdf_backend_object_write:nn { nx }
2202 \cs_new_protected:Npn \__pdf_backend_object_write_array:nn #1#2
2203 {
2204   \__pdf_backend_pdfmark:x
2205     { #1 ~0~ [ ~ \exp_not:n {#2} ~ ] ~ /PUTINTERVAL }
2206 }
2207 \cs_new_protected:Npn \__pdf_backend_object_write_dict:nn #1#2
2208 {
2209   \__pdf_backend_pdfmark:x
2210     { #1 << \exp_not:n {#2} >> /PUT }
2211 }
2212 \cs_new_protected:Npn \__pdf_backend_object_write_fstream:nn #1#2
2213 {
2214   \exp_args:Nx
2215     \__pdf_backend_object_write_fstream:nnn {#1} #2
2216 }
2217 \cs_new_protected:Npn \__pdf_backend_object_write_fstream:nnn #1#2#3
2218 {
2219   \__kernel_backend_postscript:n
2220   {
2221     SDict ~ begin ~
2222     mark ~ #1 ~ << #2 >> /PUT ~ pdfmark ~
2223     mark ~ #1 ~ ( #3 )~ ( r )~ file ~ /PUT ~ pdfmark ~
2224     end
2225   }
2226 }

```

```

2227 \cs_new_protected:Npn \__pdf_backend_object_write_stream:nn #1#2
2228 {
2229   \exp_args:Nx
2230   \__pdf_backend_object_write_stream:nnn {#1} #2
2231 }
2232 \cs_new_protected:Npn \__pdf_backend_object_write_stream:nnn #1#2#3
2233 {
2234   \__kernel_backend_postscript:n
2235   {
2236     mark ~ #1 ~ ( #3 ) /PUT ~ pdfmark ~
2237     mark ~ #1 ~ << #2 >> /PUT ~ pdfmark
2238   }
2239 }

```

(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 2240 \cs_new_protected:Npn \__pdf_backend_object_now:nn #1#2
2241 {
2242   \int_gincr:N \g__pdf_backend_object_int
2243   \__pdf_backend_pdfmark:x
2244   {
2245     /objdef ~ { pdf.obj \int_use:N \g__pdf_backend_object_int }
2246     /type
2247     \str_case:nn
2248       {#1}
2249       {
2250         { array } { /array }
2251         { dict } { /dict }
2252         { fstream } { /stream }
2253         { stream } { /stream }
2254       }
2255     /OBJ
2256   }
2257   \exp_args:Nnx \use:c { __pdf_backend_object_write_ #1 :nn }
2258   { { pdf.obj \int_use:N \g__pdf_backend_object_int } } {#2}
2259 }
2260 \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.

```

2261 \cs_new:Npn \__pdf_backend_object_last:
2262 { { 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.

```

2263 \cs_new:Npn \__pdf_backend_pageobject_ref:n #1
2264 { { 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.
2265 \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.
2266 \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.
2267 \int_new:N \g__pdf_backend_annotation_int
(End definition for \g__pdf_backend_annotation_int.)

\__pdf_backend_annotation:nnnn Annotations are objects, but we track them separately. Notably, they are not in the
object data lists. Here, to get the co-ordinates of the annotation, we need to have the
data collected at the PostScript level. That requires a bit of box trickery (effectively a
LATEX 2ε picture of zero size). Once the data is collected, use it to set up the annotation
border.
2268 \cs_new_protected:Npn \__pdf_backend_annotation:nnnn #1#2#3#4
2269 {
2270   \exp_args:Nf \__pdf_backend_annotation_aux:nnnn
2271   { \dim_eval:n {#1} } {#2} {#3} {#4}
2272 }
2273 \cs_new_protected:Npn \__pdf_backend_annotation_aux:nnnn #1#2#3#4
2274 {
2275   \box_move_down:nn {#3}
2276   { \hbox:n { \__kernel_backend_postscript:n { pdf.save.ll } } }
2277   \box_move_up:nn {#2}
2278   {
2279     \hbox:n
2280     {
2281       \__kernel_kern:n {#1}
2282       \__kernel_backend_postscript:n { pdf.save.ur }
2283       \__kernel_kern:n { -#1 }
2284     }
2285   }
2286   \int_gincr:N \g__pdf_backend_object_int
2287   \int_gset_eq:NN \g__pdf_backend_annotation_int \g__pdf_backend_object_int
2288   \__pdf_backend_pdfmark:x
2289   {
2290     /objdef { pdf.obj \int_use:N \g__pdf_backend_object_int }
2291     pdf.rect
2292     #4 ~
2293     /ANN
2294   }
2295 }
(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.

```

2296 \cs_new:Npn \__pdf_backend_annotation_last:
2297   { { 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.

```

2298 \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.

```

2299 \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.

```

2300 \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.

```

2301 \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.

```

2302 \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.

```

2303 \tl_new:N \l__pdf_breaklink_pdfmark_tl
2304 \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.

```

2305 \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.

```

2306 \cs_new_eq:NN \__pdf_breaklink_usebox:N \box_use:N

```

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

```

\__pdf_backend_link_begin_goto:nnw
\__pdf_backend_link_begin_user:nnw
\__pdf_backend_link:nw
\__pdf_backend_link_aux:nw
\__pdf_backend_link_end:
\__pdf_backend_link_end_aux:
\__pdf_backend_link_minima:
\__pdf_backend_link_outerbox:n
\__pdf_backend_link_sf_save:
\__pdf_backend_link_sf_restore:
pdf.linkdp.pad
pdf.linkht.pad
pdf.llx
pdf.lly
pdf.ury
pdf.link.dict
pdf.outerbox
pdf.baselineskip

```

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

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

Taking the idea of `evenboxes` from `hypdvips`, we implement a minimum box height and depth for link placement. This means that “underlining” with a hyperlink will generally give an even appearance. However, to ensure that the full content is always above the link border, we do not allow this to be negative (contrast `hypdvips` approach). The result should be similar to `pdfTeX` in the vast majority of foreseeable cases.

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

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.

```

2307 \cs_new_protected:Npn \__pdf_backend_link_begin_goto:nnw #1#2
2308 {
2309   \__pdf_backend_link_begin:nw
2310   { #1 /Subtype /Link /Action << /S /GoTo /D ( #2 ) >> }
2311 }
2312 \cs_new_protected:Npn \__pdf_backend_link_begin_user:nnw #1#2
2313 { \__pdf_backend_link_begin:nw {#1#2} }
2314 \cs_new_protected:Npn \__pdf_backend_link_begin:nw #1
2315 {
2316   \bool_if:NF \g__pdf_backend_link_bool
2317   { \__pdf_backend_link_begin_aux:nw {#1} }
2318 }

```

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.

```

2319 \cs_new_protected:Npn \__pdf_backend_link_begin_aux:nw #1
2320 {
2321   \bool_gset_true:N \g__pdf_backend_link_bool
2322   \__kernel_backend_postscript:n
2323   { /pdf.link.dict ( #1 ) def }
2324   \tl_gset:Nn \g__pdf_backend_link_dict_tl {#1}
2325   \__pdf_backend_link_sf_save:
2326   \mode_if_math:TF
2327   { \bool_gset_true:N \g__pdf_backend_link_math_bool }
2328   { \bool_gset_false:N \g__pdf_backend_link_math_bool }
2329   \hbox_set:Nw \l__pdf_backend_content_box
2330   \__pdf_backend_link_sf_restore:
2331   \bool_if:NT \g__pdf_backend_link_math_bool
2332   { \c_math_toggle_token }
2333 }
2334 \cs_new_protected:Npn \__pdf_backend_link_end:
2335 {
2336   \bool_if:NT \g__pdf_backend_link_bool
2337   { \__pdf_backend_link_end_aux: }
2338 }
2339 \cs_new_protected:Npn \__pdf_backend_link_end_aux:

```

```

2340 {
2341     \bool_if:NT \g__pdf_backend_link_math_bool
2342     { \c_math_toggle_token }
2343     \__pdf_backend_link_sf_save:
2344     \hbox_set_end:
2345     \__pdf_backend_link_minima:
2346     \hbox_set:Nn \l__pdf_backend_model_box { Gg }
2347     \exp_args:Nx \__pdf_backend_link_outerbox:n
2348     {
2349         \int_if_odd:nTF { \value { page } }
2350         { \oddsidemargin }
2351         { \evensidemargin }
2352     }
2353     \box_move_down:nn { \box_dp:N \l__pdf_backend_content_box }
2354     { \hbox:n { \__kernel_backend_postscript:n { pdf.save.linkll } } }
2355     \__pdf_breaklink_postscript:n { pdf.bordertracking.begin }
2356     \__pdf_breaklink_usebox:N \l__pdf_backend_content_box
2357     \__pdf_breaklink_postscript:n { pdf.bordertracking.end }
2358     \box_move_up:nn { \box_ht:N \l__pdf_backend_content_box }
2359     {
2360         \hbox:n
2361         { \__kernel_backend_postscript:n { pdf.save.linkur } }
2362     }
2363     \int_gincr:N \g__pdf_backend_object_int
2364     \int_gset_eq:NN \g__pdf_backend_link_int \g__pdf_backend_object_int
2365     \__kernel_backend_postscript:x
2366     {
2367         mark
2368         /_objdef { pdf.obj \int_use:N \g__pdf_backend_link_int }
2369         \g__pdf_backend_link_dict_tl \c_space_tl
2370         pdf.rect
2371         /ANN ~ \l__pdf_breaklink_pdfmark_tl
2372     }
2373     \__pdf_backend_link_sf_restore:
2374     \bool_gset_false:N \g__pdf_backend_link_bool
2375 }
2376 \cs_new_protected:Npn \__pdf_backend_link_minima:
2377 {
2378     \hbox_set:Nn \l__pdf_backend_model_box { Gg }
2379     \__kernel_backend_postscript:x
2380     {
2381         /pdf.linkdp.pad ~
2382         \dim_to_decimal:n
2383         {
2384             \dim_max:nn
2385             {
2386                 \box_dp:N \l__pdf_backend_model_box
2387                 - \box_dp:N \l__pdf_backend_content_box
2388             }
2389             { Opt }
2390         } ~
2391         pdf.pt.dvi ~ def
2392         /pdf.linkht.pad ~
2393         \dim_to_decimal:n

```

```

2394         {
2395             \dim_max:nn
2396             {
2397                 \box_ht:N \l__pdf_backend_model_box
2398                 - \box_ht:N \l__pdf_backend_content_box
2399             }
2400             { 0pt }
2401         } ~
2402         pdf.pt.dvi ~ def
2403     }
2404 }
2405 \cs_new_protected:Npn \__pdf_backend_link_outerbox:n #1
2406 {
2407     \__kernel_backend_postscript:x
2408     {
2409         /pdf.outerbox
2410         [
2411             \dim_to_decimal:n {#1} ~
2412             \dim_to_decimal:n { -\box_dp:N \l__pdf_backend_model_box } ~
2413             \dim_to_decimal:n { #1 + \textwidth } ~
2414             \dim_to_decimal:n { \box_ht:N \l__pdf_backend_model_box }
2415         ]
2416         [ exch { pdf.pt.dvi } forall ] def
2417         /pdf.baselineskip ~
2418         \dim_to_decimal:n { \tex_baselineskip:D } ~ dup ~ 0 ~ gt
2419         { pdf.pt.dvi ~ def }
2420         { pop ~ pop }
2421         ifelse
2422     }
2423 }
2424 \cs_new_protected:Npn \__pdf_backend_link_sf_save:
2425 {
2426     \int_gset:Nn \g__pdf_backend_link_sf_int
2427     {
2428         \mode_if_horizontal:TF
2429         { \tex_spacefactor:D }
2430         { 0 }
2431     }
2432 }
2433 \cs_new_protected:Npn \__pdf_backend_link_sf_restore:
2434 {
2435     \mode_if_horizontal:T
2436     {
2437         \int_compare:nNnT \g__pdf_backend_link_sf_int > { 0 }
2438         { \int_set_eq:NN \tex_spacefactor:D \g__pdf_backend_link_sf_int }
2439     }
2440 }

```

(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.

```

2441 \use_none:n
2442 {
2443   \cs_if_exist:NT \@makecol@hook
2444   {
2445     \tl_put_right:Nn \@makecol@hook
2446     {
2447       \box_if_empty:NF \@cclv
2448       {
2449         \vbox_set:Nn \@cclv
2450         {
2451           \__kernel_backend_postscript:n
2452           {
2453             pdf.globaldict /pdf.brokenlink.rect ~ known
2454             { pdf.bordertracking.continue }
2455             if
2456           }
2457           \vbox_unpack_drop:N \@cclv
2458           \__kernel_backend_postscript:n
2459           { pdf.bordertracking.endpage }
2460         }
2461       }
2462     }
2463     \tl_set:Nn \l__pdf_breaklink_pdfmark_tl { pdf.pdfmark }
2464     \cs_set_eq:NN \__pdf_breaklink_postscript:n \__kernel_backend_postscript:n
2465     \cs_set_eq:NN \__pdf_breaklink_usebox:N \hbox_unpack:N
2466   }
2467 }

```

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

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

```

2468 \cs_new:Npn \__pdf_backend_link_last:
2469 { { 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.

```

2470 \cs_new_protected:Npn \__pdf_backend_link_margin:n #1
2471 {
2472   \__kernel_backend_postscript:x
2473   {
2474     /pdf.linkmargin { \dim_to_decimal:n {#1} ~ pdf.pt.dvi } def
2475   }
2476 }

```

(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.

```

2477 \cs_new_protected:Npn \__pdf_backend_destination:nn #1#2
2478 {
2479   \__kernel_backend_postscript:n { pdf.dest.anchor }

```

```

2480 \__pdf_backend_pdfmark:x
2481 {
2482   /View
2483   [
2484     \str_case:nnF {#2}
2485     {
2486       { xyz } { /XYZ ~ pdf.dest.point ~ null }
2487       { fit } { /Fit }
2488       { fitb } { /FitB }
2489       { fitbh } { /FitBH ~ pdf.dest.y }
2490       { fitbv } { /FitBV ~ pdf.dest.x }
2491       { fith } { /FitH ~ pdf.dest.y }
2492       { fitv } { /FitV ~ pdf.dest.x }
2493       { fitr } { /Fit }
2494     }
2495     {
2496       /XYZ ~ pdf.dest.point ~ \fp_eval:n { (#2) / 100 }
2497     }
2498   ]
2499   /Dest ( \exp_not:n {#1} ) cvn
2500   /DEST
2501 }
2502 }
2503 \cs_new_protected:Npn \__pdf_backend_destination:nnnn #1#2#3#4
2504 {
2505   \exp_args:Ne \__pdf_backend_destination_aux:nnnn
2506   { \dim_eval:n {#2} } {#1} {#3} {#4}
2507 }
2508 \cs_new_protected:Npn \__pdf_backend_destination_aux:nnnn #1#2#3#4
2509 {
2510   \vbox_to_zero:n
2511   {
2512     \__kernel_kern:n {#4}
2513     \hbox:n { \__kernel_backend_postscript:n { pdf.save.ll } }
2514     \tex_vss:D
2515   }
2516   \__kernel_kern:n {#1}
2517   \vbox_to_zero:n
2518   {
2519     \__kernel_kern:n { -#3 }
2520     \hbox:n { \__kernel_backend_postscript:n { pdf.save.ur } }
2521     \tex_vss:D
2522   }
2523   \__kernel_kern:n { -#1 }
2524   \__pdf_backend_pdfmark:n
2525   {
2526     /View
2527     [
2528       /FitR ~
2529       pdf.llx ~ pdf.lly ~ pdf.dest2device ~
2530       pdf.urx ~ pdf.ury ~ pdf.dest2device
2531     ]
2532     /Dest ( #2 ) cvn
2533     /DEST

```

```

2534     }
2535 }

```

(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
2536 \cs_new_protected:Npn \_pdf_backend_compresslevel:n #1
2537 {
2538   \int_compare:nNnT {#1} = 0
2539   {
2540     \_kernel_backend_literal_postscript:n
2541     {
2542       /setdistillerparams ~ where
2543       { pop << /CompressPages ~ false >> setdistillerparams }
2544       if
2545     }
2546   }
2547 }
2548 \cs_new_protected:Npn \_pdf_backend_compress_objects:n #1
2549 {
2550   \bool_if:nF {#1}
2551   {
2552     \_kernel_backend_literal_postscript:n
2553     {
2554       /setdistillerparams ~ where
2555       { pop << /CompressStreams ~ false >> setdistillerparams }
2556       if
2557     }
2558   }
2559 }

```

(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
2560 \cs_new_protected:Npn \_pdf_backend_version_major_gset:n #1
2561 {
2562   \cs_gset:Npx \_pdf_backend_version_major: { \int_eval:n {#1} }
2563 }
2564 \cs_new_protected:Npn \_pdf_backend_version_minor_gset:n #1
2565 {
2566   \cs_gset:Npx \_pdf_backend_version_minor: { \int_eval:n {#1} }
2567 }

```

(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:
2568 \cs_new:Npn \_pdf_backend_version_major: { -1 }
2569 \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

Simple wrappers.

```

\__pdf_backend_bdc:nn
\__pdf_backend_emc:
2570 \cs_new_protected:Npn \__pdf_backend_bdc:nn #1#2
2571 { \__pdf_backend_pdfmark:n { /#1 ~ #2 /BDC } }
2572 \cs_new_protected:Npn \__pdf_backend_emc:
2573 { \__pdf_backend_pdfmark:n { /EMC } }

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

2574 </dvips>

```

## 6.3 LuaTeX and pdfTeX backend

2575 <\*luatex | pdftex>

### 6.3.1 Annotations

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

```

2576 \cs_new_protected:Npn \__pdf_backend_annotation:nnnn #1#2#3#4
2577 {
2578 <*luatex>
2579 \tex_pdfextension:D annot ~
2580 </luatex>
2581 <*pdftex>
2582 \tex_pdfannot:D
2583 </pdftex>
2584 width ~ \dim_eval:n {#1} ~
2585 height ~ \dim_eval:n {#2} ~
2586 depth ~ \dim_eval:n {#3} ~
2587 {#4}
2588 }

```

(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.

```

2589 \cs_new:Npx \__pdf_backend_annotation_last:
2590 {
2591 \exp_not:N \int_value:w
2592 <*luatex>
2593 \exp_not:N \tex_pdffeedback:D lastannot ~
2594 </luatex>
2595 <*pdftex>
2596 \exp_not:N \tex_pdflastannot:D
2597 </pdftex>
2598 \c_space_tl 0 ~ R
2599 }

```

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

\\_\_pdf\_backend\_link\_begin\_goto:nnw Links are all created using the same internals.

```

\__pdf_backend_link_begin_user:nnw
\__pdf_backend_link_begin:nnnw
2600 \cs_new_protected:Npn \__pdf_backend_link_begin_goto:nnw #1#2
2601 { \__pdf_backend_link_begin:nnnw {#1} { goto~name } {#2} }
\__pdf_backend_link_end:
2602 \cs_new_protected:Npn \__pdf_backend_link_begin_user:nnw #1#2

```

```

2603 { \_pdf_backend_link_begin:nnnw {#1} { user } {#2} }
2604 \cs_new_protected:Npn \_pdf_backend_link_begin:nnnw #1#2#3
2605 {
2606   \*luatex
2607   \tex_pdfextension:D startlink ~
2608   \*pdftex
2609   \tex_pdfstartlink:D
2610   \*pdftex
2611   attr {#1}
2612   #2 {#3}
2613 }
2614 \cs_new_protected:Npn \_pdf_backend_link_end:
2615 {
2616   \*luatex
2617   \tex_pdfextension:D endlink \scan_stop:
2618   \*pdftex
2619   \tex_pdfendlink:D
2620   \*pdftex
2621 }
2622
2623

```

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

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

```

2624 \cs_new:Npx \_pdf_backend_link_last:
2625 {
2626   \exp_not:N \int_value:w
2627   \*luatex
2628   \exp_not:N \tex_pdffeedback:D lastlink ~
2629   \*pdftex
2630   \exp_not:N \tex_pdflastlink:D
2631   \*pdftex
2632   \c_space_tl 0 ~ R
2633 }
2634

```

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

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

```

2635 \cs_new_protected:Npn \_pdf_backend_link_margin:n #1
2636 {
2637   \*luatex
2638   \tex_pdfvariable:D linkmargin
2639   \*pdftex
2640   \tex_pdflinkmargin:D
2641   \*pdftex
2642   \dim_eval:n {#1} \scan_stop:
2643 }
2644

```

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

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

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.

```

2645 \cs_new_protected:Npn \_pdf_backend_destination:nn #1#2
2646 {
2647   \*luatex
2648     \tex_pdfextension:D dest ~
2649   \*pdfTeX
2650   \tex_pdfdest:D
2651   \*pdfTeX
2652   name {#1}
2653   \str_case:nnF {#2}
2654   {
2655     { xyz } { xyz }
2656     { fit } { fit }
2657     { fitb } { fitb }
2658     { fitbh } { fitbh }
2659     { fitbv } { fitbv }
2660     { fith } { fith }
2661     { fitv } { fitv }
2662     { fitr } { fitr }
2663   }
2664   { xyz ~ zoom \fp_eval:n { #2 * 10 } }
2665   \scan_stop:
2666 }
2667 \cs_new_protected:Npn \_pdf_backend_destination:nnnn #1#2#3#4
2668 {
2669   \*luatex
2670     \tex_pdfextension:D dest ~
2671   \*pdfTeX
2672   \tex_pdfdest:D
2673   \*pdfTeX
2674   name {#1}
2675   fitr ~
2676   width \dim_eval:n {#2} ~
2677   height \dim_eval:n {#3} ~
2678   depth \dim_eval:n {#4} \scan_stop:
2679 }

```

(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`

```

2682 \cs_new_protected:Npn \_pdf_backend_catalog_gput:nn #1#2
2683 {
2684   \*luatex
2685     \tex_pdfextension:D catalog
2686   \*pdfTeX
2687   \tex_pdfcatalog:D
2688   \*pdfTeX

```

```

2690     { / #1 ~ #2 }
2691   }
2692   \cs_new_protected:Npn \__pdf_backend_info_gput:nn #1#2
2693   {
2694     <*luatex>
2695       \tex_pdfextension:D info
2696     </luatex>
2697     <*pdftex>
2698       \tex_pdfinfo:D
2699     </pdftex>
2700     { / #1 ~ #2 }
2701   }

```

(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.

```

2702 \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

```

2703 \cs_new_protected:Npn \__pdf_backend_object_new:nn #1#2
2704 {
2705   <*luatex>
2706     \tex_pdfextension:D obj ~
2707   </luatex>
2708   <*pdftex>
2709     \tex_pdfobj:D
2710   </pdftex>
2711     reserveobjnum ~
2712     \int_const:cn
2713     { c__pdf_backend_object_ \tl_to_str:n {#1} _int }
2714   <*luatex>
2715     { \tex_pdffeedback:D lastobj }
2716   </luatex>
2717   <*pdftex>
2718     { \tex_pdflastobj:D }
2719   </pdftex>
2720   \prop_gput:Nnn \g__pdf_backend_object_prop {#1} {#2}
2721 }
2722 \cs_new:Npn \__pdf_backend_object_ref:n #1
2723 { \int_use:c { c__pdf_backend_object_ \tl_to_str:n {#1} _int } ~ 0 ~ R }

```

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

\\_\_pdf\_backend\_object\_write:nn Writing the data needs a little information about the structure of the object.

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

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

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

```

2724 \cs_new_protected:Npn \__pdf_backend_object_write:nn #1#2
2725 {
2726   <*luatex>
2727     \tex_immediate:D \tex_pdfextension:D obj ~
2728   </luatex>
2729   <*pdftex>
2730     \tex_immediate:D \tex_pdfobj:D

```

```

2731 </pdfTeX>
2732     useobjnum ~
2733     \int_use:c
2734     { c__pdf_backend_object_ \tl_to_str:n {#1} _int }
2735     \str_case_e:nn
2736     { \prop_item:Nn \g__pdf_backend_object_prop {#1} }
2737     {
2738       { array } { { [ ~ \exp_not:n {#2} ~ ] } }
2739       { dict } { { << ~ \exp_not:n {#2} ~ >> } }
2740       { fstream }
2741       {
2742         stream ~ attr ~ { \__pdf_exp_not_i:nn #2 } ~
2743         file ~ { \__pdf_exp_not_ii:nn #2 }
2744       }
2745       { stream }
2746       {
2747         stream ~ attr ~ { \__pdf_exp_not_i:nn #2 } ~
2748         { \__pdf_exp_not_ii:nn #2 }
2749       }
2750     }
2751   }
2752   \cs_generate_variant:Nn \__pdf_backend_object_write:nn { nx }
2753   \cs_new:Npn \__pdf_exp_not_i:nn #1#2 { \exp_not:n {#1} }
2754   \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
2755   \cs_new_protected:Npn \__pdf_backend_object_now:nn #1#2
2756   {
2757     <*luatex>
2758     \tex_immediate:D \tex_pdfextension:D obj ~
2759     </luatex>
2760     <*pdfTeX>
2761     \tex_immediate:D \tex_pdfobj:D
2762     </pdfTeX>
2763     \str_case:nn
2764     {#1}
2765     {
2766       { array } { { [ ~ \exp_not:n {#2} ~ ] } }
2767       { dict } { { << ~ \exp_not:n {#2} ~ >> } }
2768       { fstream }
2769       {
2770         stream ~ attr ~ { \__pdf_exp_not_i:nn #2 } ~
2771         file ~ { \__pdf_exp_not_ii:nn #2 }
2772       }
2773       { stream }
2774       {
2775         stream ~ attr ~ { \__pdf_exp_not_i:nn #2 } ~
2776         { \__pdf_exp_not_ii:nn #2 }
2777       }
2778     }
2779   }
2780   \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.

```

2781 \cs_new:Npx \_pdf_backend_object_last:
2782 {
2783   \exp_not:N \int_value:w
2784   <*luatex>
2785   \exp_not:N \tex_pdffeedback:D lastobj ~
2786   </luatex>
2787   <*pdftex>
2788   \exp_not:N \tex_pdflastobj:D
2789   </pdftex>
2790   \c_space_tl 0 ~ R
2791 }

```

(End definition for `\_pdf_backend_object_last:.`)

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

```

2792 \cs_new:Npx \_pdf_backend_pageobject_ref:n #1
2793 {
2794   \exp_not:N \int_value:w
2795   <*luatex>
2796   \exp_not:N \tex_pdffeedback:D pageref
2797   </luatex>
2798   <*pdftex>
2799   \exp_not:N \tex_pdfpageref:D
2800   </pdftex>
2801   \c_space_tl #1 \c_space_tl \c_space_tl \c_space_tl 0 ~ R
2802 }

```

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

### 6.3.4 Structure

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

```

\_pdf_backend_compress_objects:n
\_pdf_backend_objcompresslevel:n
2803 \cs_new_protected:Npn \_pdf_backend_compresslevel:n #1
2804 {
2805   \tex_global:D
2806   <*luatex>
2807   \tex_pdfvariable:D compresslevel
2808   </luatex>
2809   <*pdftex>
2810   \tex_pdfcompresslevel:D
2811   </pdftex>
2812   \int_value:w \int_eval:n {#1} \scan_stop:
2813 }
2814 \cs_new_protected:Npn \_pdf_backend_compress_objects:n #1
2815 {
2816   \bool_if:nTF {#1}
2817   { \_pdf_backend_objcompresslevel:n { 2 } }
2818   { \_pdf_backend_objcompresslevel:n { 0 } }
2819 }
2820 \cs_new_protected:Npn \_pdf_backend_objcompresslevel:n #1
2821 {

```

```

2822     \tex_global:D
2823   \*luatex
2824     \tex_pdfvariable:D objcompresslevel
2825   \*pdftex
2826   \tex_pdfobjcompresslevel:D
2827   \*pdftex
2828   #1 \scan_stop:
2829 }

```

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

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

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

```

2831 \cs_new_protected:Npx \_pdf_backend_version_major_gset:n #1
2832 {
2833   \*luatex
2834     \int_compare:nNnT \tex_luatexversion:D > { 106 }
2835     {
2836       \exp_not:N \tex_global:D \tex_pdfvariable:D majorversion
2837       \exp_not:N \int_eval:n {#1} \scan_stop:
2838     }
2839   \*pdftex
2840   \cs_if_exist:NT \tex_pdfmajorversion:D
2841   {
2842     \exp_not:N \tex_global:D \tex_pdfmajorversion:D
2843     \exp_not:N \int_eval:n {#1} \scan_stop:
2844   }
2845   \*pdftex
2846 }
2847 \cs_new_protected:Npn \_pdf_backend_version_minor_gset:n #1
2848 {
2849   \tex_global:D
2850   \*luatex
2851     \tex_pdfvariable:D minorversion
2852   \*pdftex
2853     \tex_pdfminorversion:D
2854   \int_eval:n {#1} \scan_stop:
2855 }

```

(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.

```

2859 \cs_new:Npx \_pdf_backend_version_major:
2860 {
2861   \*luatex
2862     \int_compare:nNnTF \tex_luatexversion:D > { 106 }
2863     { \exp_not:N \tex_the:D \tex_pdfvariable:D majorversion }
2864     { 1 }
2865   \*pdftex
2866   \cs_if_exist:NTF \tex_pdfmajorversion:D

```

```

2868     { \exp_not:N \tex_the:D \tex_pdfmajorversion:D }
2869     { 1 }
2870 </pdftex>
2871 }
2872 \cs_new:Npn \__pdf_backend_version_minor:
2873 {
2874     \tex_the:D
2875 <*luatex>
2876     \tex_pdfvariable:D minorversion
2877 </luatex>
2878 <*pdftex>
2879     \tex_pdfminorversion:D
2880 </pdftex>
2881 }

```

(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>.

```

2882 \cs_new_protected:Npn \__pdf_backend_bdc:nn #1#2
2883 { \__kernel_backend_literal_page:n { /#1 ~ #2 ~ BDC } }
2884 \cs_new_protected:Npn \__pdf_backend_emc:
2885 { \__kernel_backend_literal_page:n { EMC } }

```

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

```

2886 </luatex | pdftex>

```

## 6.4 dvipdfmx backend

```

2887 <*dvipdfmx | xetex>

```

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

```

\__pdf_backend:x
2888 \cs_new_protected:Npx \__pdf_backend:n #1
2889 { \__kernel_backend_literal:n { pdf: #1 } }
2890 \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
2891 \cs_new_protected:Npn \__pdf_backend_catalog_gput:nn #1#2
2892 { \__pdf_backend:n { put ~ @catalog << /#1 ~ #2 >> } }
2893 \cs_new_protected:Npn \__pdf_backend_info_gput:nn #1#2
2894 { \__pdf_backend:n { docinfo << /#1 ~ #2 >> } }

```

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

## 6.4.2 Objects

For tracking objects to allow finalisation.

```
2895 \int_new:N \g__pdf_backend_object_int
2896 \prop_new:N \g__pdf_backend_object_prop
```

(End definition for \g\_\_pdf\_backend\_object\_int and \g\_\_pdf\_backend\_object\_prop.)

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

```
2897 \cs_new_protected:Npn \__pdf_backend_object_new:nn #1#2
2898 {
2899   \int_gincr:N \g__pdf_backend_object_int
2900   \int_const:cn
2901     { c__pdf_backend_object_ \tl_to_str:n {#1} _int }
2902     { \g__pdf_backend_object_int }
2903   \prop_gput:Nnn \g__pdf_backend_object_prop {#1} {#2}
2904 }
2905 \cs_new:Npn \__pdf_backend_object_ref:n #1
2906 { @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.

```
2907 \cs_new_protected:Npn \__pdf_backend_object_write:nn #1#2
2908 {
2909   \exp_args:Nx \__pdf_backend_object_write:nnn
2910     { \prop_item:Nn \g__pdf_backend_object_prop {#1} } {#1} {#2}
2911 }
2912 \cs_generate_variant:Nn \__pdf_backend_object_write:nn { nx }
2913 \cs_new_protected:Npn \__pdf_backend_object_write:nnn #1#2#3
2914 {
2915   \use:c { __pdf_backend_object_write_ #1 :nn }
2916   { \__pdf_backend_object_ref:n {#2} } {#3}
2917 }
2918 \cs_new_protected:Npn \__pdf_backend_object_write_array:nn #1#2
2919 {
2920   \__pdf_backend:x
2921     { obj ~ #1 ~ [ ~ \exp_not:n {#2} ~ ] }
2922 }
2923 \cs_new_protected:Npn \__pdf_backend_object_write_dict:nn #1#2
2924 {
2925   \__pdf_backend:x
2926     { obj ~ #1 ~ << ~ \exp_not:n {#2} ~ >> }
2927 }
2928 \cs_new_protected:Npn \__pdf_backend_object_write_fstream:nn #1#2
2929 { \__pdf_backend_object_write_stream:nnnn { f } {#1} #2 }
2930 \cs_new_protected:Npn \__pdf_backend_object_write_stream:nn #1#2
2931 { \__pdf_backend_object_write_stream:nnnn { } {#1} #2 }
2932 \cs_new_protected:Npn \__pdf_backend_object_write_stream:nnnn #1#2#3#4
2933 {
2934   \__pdf_backend:x
2935   {
2936     #1 stream ~ #2 ~
2937     ( \exp_not:n {#4} ) ~ << \exp_not:n {#3} >>
2938   }
```

```
2939 }
```

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

`\_pdf_backend_object_now:nn` No anonymous objects with dvipdfmx so we have to give an object name.

```
\_pdf_backend_object_now:nx
2940 \cs_new_protected:Npn \_pdf_backend_object_now:nn #1#2
2941 {
2942   \int_gincr:N \g__pdf_backend_object_int
2943   \exp_args:Nnx \use:c { \_pdf_backend_object_write_ #1 :nn }
2944   { @pdf.obj \int_use:N \g__pdf_backend_object_int }
2945   {#2}
2946 }
2947 \cs_generate_variant:Nn \_pdf_backend_object_now:nn { nx }
```

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

`\_pdf_backend_object_last:`

```
2948 \cs_new:Npn \_pdf_backend_object_last:
2949 { @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.

```
2950 \cs_new:Npn \_pdf_backend_pageobject_ref:n #1
2951 { @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.

```
2952 \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.

```
2953 \cs_new_protected:Npn \_pdf_backend_annotation:nnnn #1#2#3#4
2954 {
2955   \int_gincr:N \g__pdf_backend_object_int
2956   \int_gset_eq:NN \g__pdf_backend_annotation_int \g__pdf_backend_object_int
2957   \_pdf_backend:x
2958   {
2959     ann ~ @pdf.obj \int_use:N \g__pdf_backend_object_int \c_space_tl
2960     width ~ \dim_eval:n {#1} ~
2961     height ~ \dim_eval:n {#2} ~
2962     depth ~ \dim_eval:n {#3} ~
2963     << /Type /Annot #4 >>
2964   }
2965 }
```

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

`\_pdf_backend_annotation_last:`

```
2966 \cs_new:Npn \_pdf_backend_annotation_last:
2967 { @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.

2968 \int\_new:N \g\_pdf\_backend\_link\_int

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

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

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

\\_pdf\_backend\_link\_begin:n

\\_pdf\_backend\_link\_end:

2969 \cs\_new\_protected:Npn \\_pdf\_backend\_link\_begin\_goto:nnw #1#2

2970 { \\_pdf\_backend\_link\_begin:n { #1 /Subtype /Link /A << /S /GoTo /D ( #2 ) >> } }

2971 \cs\_new\_protected:Npn \\_pdf\_backend\_link\_begin\_user:nnw #1#2

2972 { \\_pdf\_backend\_link\_begin:n {#1#2} }

2973 \cs\_new\_protected:Npx \\_pdf\_backend\_link\_begin:n #1

2974 {

2975 \exp\_not:N \int\_gincr:N \exp\_not:N \g\_pdf\_backend\_link\_int

2976 \\_pdf\_backend:x

2977 {

2978 bann ~

2979 @pdf.lnk

2980 \exp\_not:N \int\_use:N \exp\_not:N \g\_pdf\_backend\_link\_int

2981 \c\_space\_tl

2982 <<

2983 /Type /Annot

2984 #1

2985 >>

2986 }

2987 }

2988 \cs\_new\_protected:Npn \\_pdf\_backend\_link\_end:

2989 { \\_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.

2990 \cs\_new:Npn \\_pdf\_backend\_link\_last:

2991 { @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.

2992 \cs\_new\_protected:Npn \\_pdf\_backend\_link\_margin:n #1

2993 { \\_kernel\_backend\_literal:x { dvipdfmx:config-g~ \dim\_eval:n {#1} } }

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

\\_pdf\_backend\_destination:nn

\\_pdf\_backend\_destination:nnnn

\\_pdf\_backend\_destination\_aux:nnnn

Here, we need to turn the zoom into a scale. The method for FitR is from Alexander Grahn: the idea is to avoid needing to do any calculations in 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.

2994 \cs\_new\_protected:Npn \\_pdf\_backend\_destination:nn #1#2

2995 {

2996 \\_pdf\_backend:x

2997 {

2998 dest ~ ( \exp\_not:n {#1} )

2999 [

3000 @thispage

```

3001 \str_case:nnF {#2}
3002 {
3003   { xyz } { /XYZ ~ @xpos ~ @ypos ~ null }
3004   { fit } { /Fit }
3005   { fitb } { /FitB }
3006   { fitbh } { /FitBH }
3007   { fitbv } { /FitBV ~ @xpos }
3008   { fith } { /FitH ~ @ypos }
3009   { fitv } { /FitV ~ @xpos }
3010   { fitr } { /Fit }
3011 }
3012 { /XYZ ~ @xpos ~ @ypos ~ \fp_eval:n { (#2) / 100 } }
3013 ]
3014 }
3015 }
3016 \cs_new_protected:Npn \__pdf_backend_destination:nnnn #1#2#3#4
3017 {
3018   \exp_args:Ne \__pdf_backend_destination_aux:nnnn
3019   { \dim_eval:n {#2} } {#1} {#3} {#4}
3020 }
3021 \cs_new_protected:Npn \__pdf_backend_destination_aux:nnnn #1#2#3#4
3022 {
3023   \vbox_to_zero:n
3024   {
3025     \__kernel_kern:n {#4}
3026     \hbox:n
3027     {
3028       \__pdf_backend:n { obj ~ @pdf_ #2 _llx ~ @xpos }
3029       \__pdf_backend:n { obj ~ @pdf_ #2 _lly ~ @ypos }
3030     }
3031     \tex_vss:D
3032   }
3033   \__kernel_kern:n {#1}
3034   \vbox_to_zero:n
3035   {
3036     \__kernel_kern:n { -#3 }
3037     \hbox:n
3038     {
3039       \__pdf_backend:n
3040       {
3041         dest ~ (#2)
3042         [
3043           @thispage
3044           /FitR ~
3045           @pdf_ #2 _llx ~ @pdf_ #2 _lly ~
3046           @xpos ~ @ypos
3047         ]
3048       }
3049     }
3050     \tex_vss:D
3051   }
3052   \__kernel_kern:n { -#1 }
3053 }

```

(End definition for \\_\_pdf\_backend\_destination:nn, \\_\_pdf\_backend\_destination:nnnn, and \\_\_-

`pdf_backend_destination_aux:nnnn.)`

#### 6.4.4 Structure

`\_pdf_backend_compresslevel:n`  
`\_pdf_backend_compress_objects:n`

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

```

3054 \cs_new_protected:Npn \_pdf_backend_compresslevel:n #1
3055   { \__kernel_backend_literal:x { dvipdfmx:config~z~ \int_eval:n {#1} } }
3056 \cs_new_protected:Npn \_pdf_backend_compress_objects:n #1
3057   {
3058     \bool_if:nF {#1}
3059     { \__kernel_backend_literal:n { dvipdfmx:config~C~0x40 } }
3060   }

```

(End definition for `\_pdf_backend_compresslevel:n` and `\_pdf_backend_compress_objects:n`.)

`\_pdf_backend_version_major_gset:n`  
`\_pdf_backend_version_minor_gset:n`

We start with the assumption that the default is active.

```

3061 \cs_new_protected:Npn \_pdf_backend_version_major_gset:n #1
3062   {
3063     \cs_gset:Npx \_pdf_backend_version_major: { \int_eval:n {#1} }
3064     \__kernel_backend_literal:x { pdf:majorversion~ \_pdf_backend_version_major: }
3065   }
3066 \cs_new_protected:Npn \_pdf_backend_version_minor_gset:n #1
3067   {
3068     \cs_gset:Npx \_pdf_backend_version_minor: { \int_eval:n {#1} }
3069     \__kernel_backend_literal:x { pdf:minorversion~ \_pdf_backend_version_minor: }
3070   }

```

(End definition for `\_pdf_backend_version_major_gset:n` and `\_pdf_backend_version_minor_gset:n`.)

`\_pdf_backend_version_major:`  
`\_pdf_backend_version_minor:`

We start with the assumption that the default is active.

```

3071 \cs_new:Npn \_pdf_backend_version_major: { 1 }
3072 \cs_new:Npn \_pdf_backend_version_minor: { 5 }

```

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

#### 6.4.5 Marked content

`\_pdf_backend_bdc:nn`  
`\_pdf_backend_emc:`

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

```

3073 \cs_new_protected:Npn \_pdf_backend_bdc:nn #1#2
3074   { \__kernel_backend_literal_page:n { /#1 ~ #2 ~ BDC } }
3075 \cs_new_protected:Npn \_pdf_backend_emc:
3076   { \__kernel_backend_literal_page:n { EMC } }

```

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

```

3077 </dvipdfmx | xetex>

```

## 6.5 dvisvgm backend

3078  $\langle *dvisvgm \rangle$

### 6.5.1 Catalogue entries

No-op.

$\backslash\_pdf\_backend\_catalog\_gput:nn$   
 $\backslash\_pdf\_backend\_info\_gput:nn$

3079  $\backslash cs\_new\_protected:Npn \backslash\_pdf\_backend\_catalog\_gput:nn \#1\#2 \{ \}$   
3080  $\backslash cs\_new\_protected:Npn \backslash\_pdf\_backend\_info\_gput:nn \#1\#2 \{ \}$

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

### 6.5.2 Objects

All no-ops here.

$\backslash\_pdf\_backend\_object\_new:nn$   
 $\backslash\_pdf\_backend\_object\_ref:n$   
 $\backslash\_pdf\_backend\_object\_write:nn$   
 $\backslash\_pdf\_backend\_object\_write:nx$   
 $\backslash\_pdf\_backend\_object\_now:nn$   
 $\backslash\_pdf\_backend\_object\_now:nx$   
 $\backslash\_pdf\_backend\_object\_last:$   
 $\backslash\_pdf\_backend\_pageobject\_ref:n$

3081  $\backslash cs\_new\_protected:Npn \backslash\_pdf\_backend\_object\_new:nn \#1\#2 \{ \}$   
3082  $\backslash cs\_new:Npn \backslash\_pdf\_backend\_object\_ref:n \#1 \{ \}$   
3083  $\backslash cs\_new\_protected:Npn \backslash\_pdf\_backend\_object\_write:nn \#1\#2 \{ \}$   
3084  $\backslash cs\_new\_protected:Npn \backslash\_pdf\_backend\_object\_write:nx \#1\#2 \{ \}$   
3085  $\backslash cs\_new\_protected:Npn \backslash\_pdf\_backend\_object\_now:nn \#1\#2 \{ \}$   
3086  $\backslash cs\_new\_protected:Npn \backslash\_pdf\_backend\_object\_now:nx \#1\#2 \{ \}$   
3087  $\backslash cs\_new:Npn \backslash\_pdf\_backend\_object\_last: \{ \}$   
3088  $\backslash cs\_new:Npn \backslash\_pdf\_backend\_pageobject\_ref:n \#1 \{ \}$

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

### 6.5.3 Structure

These are all no-ops.

$\backslash\_pdf\_backend\_compresslevel:n$   
 $\backslash\_pdf\_backend\_compress\_objects:n$

3089  $\backslash cs\_new\_protected:Npn \backslash\_pdf\_backend\_compresslevel:n \#1 \{ \}$   
3090  $\backslash cs\_new\_protected:Npn \backslash\_pdf\_backend\_compress\_objects:n \#1 \{ \}$

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

$\backslash\_pdf\_backend\_version\_major\_gset:n$   
 $\backslash\_pdf\_backend\_version\_minor\_gset:n$

Data not available!

3091  $\backslash cs\_new\_protected:Npn \backslash\_pdf\_backend\_version\_major\_gset:n \#1 \{ \}$   
3092  $\backslash cs\_new\_protected:Npn \backslash\_pdf\_backend\_version\_minor\_gset:n \#1 \{ \}$

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

$\backslash\_pdf\_backend\_version\_major:$   
 $\backslash\_pdf\_backend\_version\_minor:$

Data not available!

3093  $\backslash cs\_new:Npn \backslash\_pdf\_backend\_version\_major: \{ -1 \}$   
3094  $\backslash cs\_new:Npn \backslash\_pdf\_backend\_version\_minor: \{ -1 \}$

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

$\backslash\_pdf\_backend\_bdc:nn$   
 $\backslash\_pdf\_backend\_emc:$

More no-ops.

3095  $\backslash cs\_new\_protected:Npn \backslash\_pdf\_backend\_bdc:nn \#1\#2 \{ \}$   
3096  $\backslash cs\_new\_protected:Npn \backslash\_pdf\_backend\_emc: \{ \}$

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

3097  $\langle /dvisvgm \rangle$

3098  $\langle /package \rangle$

## 7 l3backend-opacity Implementation

```

3099 <*package>
3100 <@@=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.

```

3101 <*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.

```

3102 \cs_new_protected:Npn \__opacity_backend_select:n #1
3103 {
3104   \exp_args:Nx \__opacity_backend_select_aux:n
3105   { \fp_eval:n { min(max(0,#1),1) } }
3106 }
3107 \cs_new_protected:Npn \__opacity_backend_select_aux:n #1
3108 {
3109   \__opacity_backend:nnn {#1} { fill } { ca }
3110   \__opacity_backend:nnn {#1} { stroke } { CA }
3111 }
3112 \cs_new_protected:Npn \__opacity_backend_fill:n #1
3113 {
3114   \__opacity_backend:xnn
3115   { \fp_eval:n { min(max(0,#1),1) } }
3116   { fill }
3117   { ca }
3118 }
3119 \cs_new_protected:Npn \__opacity_backend_stroke:n #1
3120 {
3121   \__opacity_backend:xnn
3122   { \fp_eval:n { min(max(0,#1),1) } }
3123   { stroke }
3124   { CA }
3125 }
3126 \cs_new_protected:Npn \__opacity_backend:nnn #1#2#3
3127 {
3128   \__kernel_backend_postscript:n
3129   {
3130     product ~ (Ghostscript) ~ search
3131     {
3132       pop ~ pop ~ pop ~
3133       #1 ~ .set #2 constantalpha
3134     }
3135     {
3136       pop ~
3137       mark ~
3138       /#3 ~ #1

```

```

3139         /SetTransparency ~
3140         pdfmark
3141     }
3142     ifelse
3143 }
3144 }
3145 \cs_generate_variant:Nn \_opacity_backend:nnn { x }

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

3146 </dvips>
3147 <*dvipdfmx | luatex | pdftex | xetex>

```

\c\_\_opacity\_backend\_stack\_int Set up a stack.

```

3148 \bool_lazy_and:nnT
3149 { \cs_if_exist_p:N \pdfmanagement_if_active_p: }
3150 { \pdfmanagement_if_active_p:}
3151 {
3152     \__kernel_color_backend_stack_init:Nnn \c__opacity_backend_stack_int
3153     { page ~ direct } { /opacity 1 ~ gs }
3154     \pdfmanagement_add:nnn { Page / Resources / ExtGState }
3155     { opacity 1 } { << /ca ~ 1 /CA ~ 1 >> }
3156 }

(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
3157 \tl_new:N \l__opacity_backend_fill_tl
3158 \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:
3159 \cs_new_protected:Npn \_opacity_backend_select:n #1
3160 {
3161     \exp_args:Nx \_opacity_backend_select_aux:n
3162     { \fp_eval:n { min(max(0,#1),1) } }
3163 }
3164 \cs_new_protected:Npn \_opacity_backend_select_aux:n #1
3165 {
3166     \tl_set:Nn \l__opacity_backend_fill_tl {#1}
3167     \tl_set:Nn \l__opacity_backend_stroke_tl {#1}
3168     \pdfmanagement_add:nnn { Page / Resources / ExtGState }
3169     { opacity #1 }
3170     { << /ca ~ #1 /CA ~ #1 >> }
3171     \__kernel_color_backend_stack_push:nn \c__opacity_backend_stack_int
3172     { /opacity #1 ~ gs }
3173     \group_insert_after:N \_opacity_backend_reset:
3174 }
3175 \bool_lazy_and:nnF
3176 { \cs_if_exist_p:N \pdfmanagement_if_active_p: }
3177 { \pdfmanagement_if_active_p:}
3178 {
3179     \cs_gset_protected:Npn \_opacity_backend_select_aux:n #1 { }
3180 }

```

```

3181 \cs_new_protected:Npn \__opacity_backend_reset:
3182 { \__kernel_color_backend_stack_pop:n \c__opacity_backend_stack_int }

(End definition for \__opacity_backend_select:n, \__opacity_backend_select_aux:n, and \__opacity_backend_reset:.)

```

For separate fill and stroke, we need to work out if we need to do more work or if we can stick to a single setting.

```

3183 \cs_new_protected:Npn \__opacity_backend_fill:n #1
3184 {
3185   \__opacity_backend_fill_stroke:xx
3186   { \fp_eval:n { min(max(0,#1),1) } }
3187   \l__opacity_backend_stroke_tl
3188 }
3189 \cs_new_protected:Npn \__opacity_backend_stroke:n #1
3190 {
3191   \__opacity_backend_fill_stroke:xx
3192   \l__opacity_backend_fill_tl
3193   { \fp_eval:n { min(max(0,#1),1) } }
3194 }
3195 \cs_new_protected:Npn \__opacity_backend_fill_stroke:nn #1#2
3196 {
3197   \str_if_eq:nnTF {#1} {#2}
3198   { \__opacity_backend_select_aux:n {#1} }
3199   {
3200     \tl_set:Nn \l__opacity_backend_fill_tl {#1}
3201     \tl_set:Nn \l__opacity_backend_stroke_tl {#2}
3202     \pdfmanagement_add:nnn { Page / Resources / ExtGState }
3203     { opacity.fill #1 }
3204     { << /ca ~ #1 >> }
3205     \pdfmanagement_add:nnn { Page / Resources / ExtGState }
3206     { opacity.stroke #1 }
3207     { << /CA ~ #2 >> }
3208     \__kernel_color_backend_stack_push:nn \c__opacity_backend_stack_int
3209     { /opacity.fill #1 ~ gs /opacity.stroke #2 ~ gs }
3210     \group_insert_after:N \__opacity_backend_reset:
3211   }
3212 }
3213 \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.)

```

3214 </dviptfm | luatex | pdftex | xetex>
3215 <*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.

```

3216 \cs_new_protected:Npn \__opacity_backend_select:n #1
3217 { \__opacity_backend:nn {#1} { } }
3218 \cs_new_protected:Npn \__opacity_backend_fill:n #1
3219 { \__opacity_backend:nn {#1} { fill- } }
3220 \cs_new_protected:Npn \__opacity_backend_stroke:n #1
3221 { \__opacity_backend:nn {#1} { stroke- } }
3222 \cs_new_protected:Npn \__opacity_backend:nn #1#2
3223 { \__kernel_backend_scope:x { #2 opacity = " \fp_eval:n { min(max(0,#1),1) } " } }

```

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

3224 `</dvisvgm>`

3225 `</package>`

## 8 l3backend-header Implementation

3226 `<*dvips & header>`

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

3227 `/color.sc { } def`

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

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

3228 `TeXDict begin`

3229 `/TeXcolorseparation { setcolor } def`

3230 `end`

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

`pdf.globaldict` A small global dictionary for backend use.

3231 `true setglobal`

3232 `/pdf.globaldict 4 dict def`

3233 `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` 3234 `/pdf.cvs { 65534 string cvs } def`

3235 `/pdf.dvi.pt { 72.27 mul Resolution div } def`

3236 `/pdf.pt.dvi { 72.27 div Resolution mul } def`

3237 `/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` 3238 `/pdf.linkmargin { 1 pdf.pt.dvi } def`

`pdf.linkht.pad` 3239 `/pdf.linkdp.pad { 0 } def`

3240 `/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` 3241 `/pdf.rect`

`pdf.save.linkur` 3242 `{ /Rect [ pdf.llx pdf.lly pdf.urx pdf.ury ] } def`

`pdf.llx` 3243 `/pdf.save.ll`

`pdf.lly` 3244 `{`

`pdf.urx`

`pdf.ury`

```

3245     currentpoint
3246     /pdf.lly exch def
3247     /pdf.llx exch def
3248   }
3249   def
3250 /pdf.save.ur
3251 {
3252     currentpoint
3253     /pdf.ury exch def
3254     /pdf.urx exch def
3255 }
3256 def
3257 /pdf.save.linkll
3258 {
3259     currentpoint
3260     pdf.linkmargin add
3261     pdf.linkdp.pad add
3262     /pdf.lly exch def
3263     pdf.linkmargin sub
3264     /pdf.llx exch def
3265 }
3266 def
3267 /pdf.save.linkur
3268 {
3269     currentpoint
3270     pdf.linkmargin sub
3271     pdf.linkht.pad sub
3272     /pdf.ury exch def
3273     pdf.linkmargin add
3274     /pdf.urx exch def
3275 }
3276 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 3277 /pdf.dest.anchor
pdf.dev.y 3278 {
pdf.tmpa 3279     currentpoint exch
pdf.tmpb 3280     pdf.dvi.pt 72 add
pdf.tmpc 3281     /pdf.dest.x exch def
pdf.tmpd 3282     pdf.dvi.pt
3283     vsize 72 sub exch sub
3284     /pdf.dest.y exch def
3285 }
3286 def
3287 /pdf.dest.point
3288 { pdf.dest.x pdf.dest.y } def
3289 /pdf.dest2device
3290 {

```

```

3291     /pdf.dest.y exch def
3292     /pdf.dest.x exch def
3293     matrix currentmatrix
3294     matrix defaultmatrix
3295     matrix invertmatrix
3296     matrix concatmatrix
3297     cvx exec
3298     /pdf.dev.y exch def
3299     /pdf.dev.x exch def
3300     /pdf.tmpd exch def
3301     /pdf.tmpc exch def
3302     /pdf.tmpb exch def
3303     /pdf.tmpa exch def
3304     pdf.dest.x pdf.tmpa mul
3305         pdf.dest.y pdf.tmpc mul add
3306         pdf.dev.x add
3307     pdf.dest.x pdf.tmpb mul
3308         pdf.dest.y pdf.tmpd mul add
3309         pdf.dev.y add
3310 }
3311 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	3312 /pdf.bordertracking false def
pdf.brokenlink.skip	3313 /pdf.bordertracking.begin
pdf.brokenlink.dict	3314 {
pdf.bordertracking.endpage	3315 SDict /pdf.bordertracking true put
pdf.bordertracking.continue	3316 SDict /pdf.leftboundary undef
pdf.originx	3317 SDict /pdf.rightboundary undef
pdf.originy	3318 /a where
	3319 {
	3320 /a
	3321 {
	3322 currentpoint pop
	3323 SDict /pdf.rightboundary known dup
	3324 {
	3325 SDict /pdf.rightboundary get 2 index lt
	3326 { not }
	3327 if
	3328 }
	3329 if
	3330 { pop }
	3331 { SDict exch /pdf.rightboundary exch put }
	3332 ifelse
	3333 moveto
	3334 currentpoint pop
	3335 SDict /pdf.leftboundary known dup
	3336 {
	3337 SDict /pdf.leftboundary get 2 index gt

```

3338         { not }
3339         if
3340         }
3341     if
3342     { pop }
3343     { SDict exch /pdf.leftboundary exch put }
3344     ifelse
3345     }
3346     put
3347 }
3348 if
3349 }
3350 def
3351 /pdf.bordertracking.end
3352 {
3353     /a where { /a { moveto } put } if
3354     /x where { /x { 0 exch rmoveto } put } if
3355     SDict /pdf.leftboundary known
3356     { pdf.outerbox 0 pdf.leftboundary put }
3357     if
3358     SDict /pdf.rightboundary known
3359     { pdf.outerbox 2 pdf.rightboundary put }
3360     if
3361     SDict /pdf.bordertracking false put
3362 }
3363 def
3364 /pdf.bordertracking.endpage
3365 {
3366     pdf.bordertracking
3367     {
3368         pdf.bordertracking.end
3369         true setglobal
3370         pdf.globaldict
3371         /pdf.brokenlink.rect [ pdf.outerbox aload pop ] put
3372         pdf.globaldict
3373         /pdf.brokenlink.skip pdf.baselineskip put
3374         pdf.globaldict
3375         /pdf.brokenlink.dict
3376         pdf.link.dict pdf.cvs put
3377         false setglobal
3378         mark pdf.link.dict cvx exec /Rect
3379         [
3380             pdf.llx
3381             pdf.lly
3382             pdf.outerbox 2 get pdf.linkmargin add
3383             currentpoint exch pop
3384             pdf.outerbox pdf.rect.ht sub pdf.linkmargin sub
3385         ]
3386         /ANN pdf.pdfmark
3387     }
3388     if
3389 }
3390 def
3391 /pdf.bordertracking.continue

```

```

3392 {
3393   /pdf.link.dict pdf.globaldict
3394   /pdf.brokenlink.dict get def
3395   /pdf.outerbox pdf.globaldict
3396   /pdf.brokenlink.rect get def
3397   /pdf.baselineskip pdf.globaldict
3398   /pdf.brokenlink.skip get def
3399   pdf.globaldict dup dup
3400   /pdf.brokenlink.dict undef
3401   /pdf.brokenlink.skip undef
3402   /pdf.brokenlink.rect undef
3403   currentpoint
3404   /pdf.originy exch def
3405   /pdf.originx exch def
3406   /a where
3407   {
3408     /a
3409     {
3410       moveto
3411       SDict
3412       begin
3413         currentpoint pdf.originy ne exch
3414         pdf.originx ne or
3415         {
3416           pdf.save.linkll
3417           /pdf.lly
3418             pdf.lly pdf.outerbox 1 get sub def
3419           pdf.bordertracking.begin
3420         }
3421         if
3422         end
3423       }
3424       put
3425     }
3426   if
3427   /x where
3428   {
3429     /x
3430     {
3431       0 exch rmoveto
3432       SDict
3433       begin
3434         currentpoint
3435         pdf.originy ne exch pdf.originx ne or
3436         {
3437           pdf.save.linkll
3438           /pdf.lly
3439             pdf.lly pdf.outerbox 1 get sub def
3440           pdf.bordertracking.begin
3441         }
3442         if
3443         end
3444       }
3445       put

```

```

3446     }
3447     if
3448 }
3449 def

```

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

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

```

3450 /pdf.breaklink
3451 {
3452     pop
3453     counttomark 2 mod 0 eq
3454     {
3455         counttomark /pdf.count exch def
3456         {
3457             pdf.count 0 eq { exit } if
3458             counttomark 2 roll
3459             1 index /Rect eq
3460             {
3461                 dup 4 array copy
3462                 dup dup
3463                 1 get
3464                 pdf.outerbox pdf.rect.ht
3465                 pdf.linkmargin 2 mul add sub
3466                 3 exch put
3467                 dup
3468                 pdf.outerbox 2 get
3469                 pdf.linkmargin add
3470                 2 exch put
3471                 dup dup
3472                 3 get
3473                 pdf.outerbox pdf.rect.ht
3474                 pdf.linkmargin 2 mul add add
3475                 1 exch put
3476             /pdf.currentrect exch def
3477             pdf.breaklink.write
3478             {
3479                 pdf.currentrect
3480                 dup
3481                 pdf.outerbox 0 get
3482                 pdf.linkmargin sub
3483                 0 exch put
3484                 dup
3485                 pdf.outerbox 2 get
3486                 pdf.linkmargin add
3487                 2 exch put
3488                 dup dup
3489                 1 get
3490                 pdf.baselineskip add
3491                 1 exch put

```

```

3492         dup dup
3493         3 get
3494         pdf.baselineskip add
3495         3 exch put
3496         /pdf.currentrect exch def
3497         pdf.breaklink.write
3498     }
3499     1 index 3 get
3500     pdf.linkmargin 2 mul add
3501     pdf.outerbox pdf.rect.ht add
3502     2 index 1 get sub
3503     pdf.baselineskip div round cvi 1 sub
3504     exch
3505     repeat
3506     pdf.currentrect
3507     dup
3508         pdf.outerbox 0 get
3509         pdf.linkmargin sub
3510         0 exch put
3511     dup dup
3512         1 get
3513         pdf.baselineskip add
3514         1 exch put
3515     dup dup
3516         3 get
3517         pdf.baselineskip add
3518         3 exch put
3519     dup 2 index 2 get 2 exch put
3520     /pdf.currentrect exch def
3521     pdf.breaklink.write
3522     SDict /pdf.pdfmark.good false put
3523     exit
3524 }
3525 { pdf.count 2 sub /pdf.count exch def }
3526 ifelse
3527 }
3528 loop
3529 }
3530 if
3531 /ANN
3532 }
3533 def
3534 /pdf.breaklink.write
3535 {
3536     counttomark 1 sub
3537     index /_objdef eq
3538     {
3539         counttomark -2 roll
3540         dup wcheck
3541         {
3542             readonly
3543             counttomark 2 roll
3544         }
3545         { pop pop }

```

```

3546         ifelse
3547     }
3548     if
3549     counttomark 1 add copy
3550     pop pdf.currentrect
3551     /ANN pdfmark
3552 }
3553 def

```

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

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

```

3554 /pdf.pdfmark
3555 {
3556     SDict /pdf.pdfmark.good true put
3557     dup /ANN eq
3558     {
3559         pdf.pdfmark.store
3560         pdf.pdfmark.dict
3561         begin
3562             Subtype /Link eq
3563             currentdict /Rect known and
3564             SDict /pdf.outerbox known and
3565             SDict /pdf.baselineskip known and
3566             {
3567                 Rect 3 get
3568                 pdf.linkmargin 2 mul add
3569                 pdf.outerbox pdf.rect.ht add
3570                 Rect 1 get sub
3571                 pdf.baselineskip div round cvi 0 gt
3572                 { pdf.breaklink }
3573                 if
3574             }
3575             if
3576         end
3577         SDict /pdf.outerbox undef
3578         SDict /pdf.baselineskip undef
3579         currentdict /pdf.pdfmark.dict undef
3580     }
3581     if
3582     pdf.pdfmark.good
3583     { pdfmark }
3584     { cleartomark }
3585     ifelse
3586 }
3587 def
3588 /pdf.pdfmark.store
3589 {
3590     /pdf.pdfmark.dict 65534 dict def
3591     counttomark 1 add copy

```

```

3592     pop
3593     {
3594         dup mark eq
3595         {
3596             pop
3597             exit
3598         }
3599         {
3600             pdf.pdfmark.dict
3601             begin def end
3602         }
3603     } ifelse
3604 }
3605 loop
3606 }
3607 def

```

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

```

3608 </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> .....	<a href="#">57</a>
<b>B</b>	
bool commands:	
<code>\bool_gset_false:N</code> .....	<a href="#">1215</a> , <a href="#">1234</a> , <a href="#">1257</a> , <a href="#">1279</a> , <a href="#">1295</a> , <a href="#">1396</a> , <a href="#">1635</a> , <a href="#">1671</a> , <a href="#">2328</a> , <a href="#">2374</a>
<code>\bool_gset_true:N</code> .....	<a href="#">1213</a> , <a href="#">1282</a> , <a href="#">1394</a> , <a href="#">1650</a> , <a href="#">2321</a> , <a href="#">2327</a>
<code>\bool_if:NTF</code> .....	<a href="#">67</a> , <a href="#">668</a> , <a href="#">1225</a> , <a href="#">1229</a> , <a href="#">1245</a> , <a href="#">1248</a> , <a href="#">1252</a> , <a href="#">1263</a> , <a href="#">1270</a> , <a href="#">1274</a> , <a href="#">1286</a> , <a href="#">1290</a> , <a href="#">1407</a> , <a href="#">1412</a> , <a href="#">1417</a> , <a href="#">1609</a> , <a href="#">1654</a> , <a href="#">1793</a> , <a href="#">1837</a> , <a href="#">1974</a> , <a href="#">2016</a> , <a href="#">2316</a> , <a href="#">2331</a> , <a href="#">2336</a> , <a href="#">2341</a>
<code>\bool_if:nTF</code> .....	<a href="#">2550</a> , <a href="#">2816</a> , <a href="#">3058</a>
<code>\bool_lazy_and:nnTF</code> .....	<a href="#">884</a> , <a href="#">958</a> , <a href="#">3148</a> , <a href="#">3175</a>
<code>\bool_lazy_or:nnTF</code> .....	<a href="#">1829</a> , <a href="#">2009</a>
<code>\bool_new:N</code> .....	<a href="#">1216</a> , <a href="#">1283</a> , <a href="#">1397</a> , <a href="#">1651</a> , <a href="#">2301</a> , <a href="#">2302</a>
<code>\bool_set_false:N</code> .....	<a href="#">1804</a> , <a href="#">1938</a> , <a href="#">2040</a> , <a href="#">2122</a>
box commands:	
<code>\box_dp:N</code> .....	<a href="#">208</a> , <a href="#">210</a> , <a href="#">258</a> , <a href="#">260</a> , <a href="#">315</a> , <a href="#">317</a> , <a href="#">364</a> , <a href="#">366</a> , <a href="#">368</a> , <a href="#">370</a> , <a href="#">2353</a> , <a href="#">2386</a> , <a href="#">2387</a> , <a href="#">2412</a>
<code>\box_ht:N</code> .....	<a href="#">210</a> , <a href="#">260</a> , <a href="#">317</a> , <a href="#">368</a> , <a href="#">370</a> , <a href="#">1849</a> , <a href="#">2081</a> , <a href="#">2358</a> , <a href="#">2397</a> , <a href="#">2398</a> , <a href="#">2414</a>
<code>\box_if_empty:N</code> .....	<a href="#">2447</a>
<code>\box_move_down:nn</code> .....	<a href="#">2275</a> , <a href="#">2353</a>
<code>\box_move_up:nn</code> .....	<a href="#">2277</a> , <a href="#">2358</a>
<code>\box_new:N</code> .....	<a href="#">2160</a> , <a href="#">2265</a> , <a href="#">2266</a>
<code>\box_set_dp:Nn</code> .....	<a href="#">1734</a>
<code>\box_set_ht:Nn</code> .....	<a href="#">1733</a>
<code>\box_set_wd:Nn</code> .....	<a href="#">272</a> , <a href="#">1732</a>
<code>\box_use:N</code> .....	<a href="#">215</a> , <a href="#">233</a> , <a href="#">247</a> , <a href="#">263</a> , <a href="#">290</a> , <a href="#">304</a> , <a href="#">320</a> , <a href="#">336</a> , <a href="#">348</a> , <a href="#">399</a> , <a href="#">413</a> , <a href="#">432</a> , <a href="#">1347</a> , <a href="#">1542</a> , <a href="#">1735</a> , <a href="#">2306</a>
<code>\box_wd:N</code> .....	<a href="#">209</a> , <a href="#">217</a> , <a href="#">259</a> , <a href="#">265</a> , <a href="#">316</a> , <a href="#">322</a> , <a href="#">365</a> , <a href="#">367</a> , <a href="#">1848</a> , <a href="#">2080</a>
box internal commands:	
<code>\__box_backend_clip:N</code> .....	<a href="#">197</a> , <a href="#">252</a> , <a href="#">309</a> , <a href="#">353</a>
<code>\l__box_backend_cos_fp</code> .....	<a href="#">267</a>
<code>\__box_backend_rotate:Nn</code> .....	<a href="#">219</a> , <a href="#">267</a> , <a href="#">324</a> , <a href="#">403</a>
<code>\__box_backend_rotate_aux:Nn</code> ...	<a href="#">219</a> , <a href="#">267</a> , <a href="#">324</a>
<code>\__box_backend_scale:Nnn</code> .....	<a href="#">236</a> , <a href="#">295</a> , <a href="#">339</a> , <a href="#">416</a>
<code>\l__box_backend_sin_fp</code> .....	<a href="#">267</a>
<b>C</b>	
clist commands:	
<code>\clist_map_function:nN</code> .....	<a href="#">1303</a> , <a href="#">1427</a> , <a href="#">1678</a>
color internal commands:	
<code>\__color_backend:nnn</code> .....	<a href="#">1099</a>
<code>\__color_backend_cmyk:w</code> .....	<a href="#">1100</a>
<code>\g__color_backend_colorant_prop</code> ..	<a href="#">634</a> , <a href="#">653</a> , <a href="#">656</a> , <a href="#">676</a> , <a href="#">901</a>
<code>\__color_backend_devicen_-colorants:n</code> .....	<a href="#">635</a> , <a href="#">837</a> , <a href="#">956</a>
<code>\__color_backend_devicen_-colorants:w</code> .....	<a href="#">635</a>
<code>\__color_backend_devicen_-init:nnn</code> .....	<a href="#">824</a> , <a href="#">926</a> , <a href="#">1154</a>
<code>\__color_backend_devicen_init:w</code> ..	<a href="#">926</a>
<code>\__color_backend_fill:n</code> .....	<a href="#">1029</a> , <a href="#">1056</a> , <a href="#">1063</a> , <a href="#">1081</a> , <a href="#">1088</a>
<code>\__color_backend_fill_cmyk:n</code> ...	<a href="#">1029</a> , <a href="#">1063</a> , <a href="#">1088</a>
<code>\__color_backend_fill_devicen:nn</code> ..	<a href="#">1055</a> , <a href="#">1080</a> , <a href="#">1150</a>
<code>\__color_backend_fill_gray:n</code> ...	<a href="#">1029</a> , <a href="#">1063</a> , <a href="#">1088</a>
<code>\__color_backend_fill_rgb:n</code> ...	<a href="#">1029</a> , <a href="#">1063</a> , <a href="#">1088</a>
<code>\__color_backend_fill_separation:nn</code> ..	<a href="#">1055</a> , <a href="#">1080</a> , <a href="#">1150</a>
<code>\l__color_backend_fill_tl</code> .....	<a href="#">616</a> , <a href="#">626</a> , <a href="#">1037</a> , <a href="#">1052</a>
<code>\__color_backend_iccbased_-device:nnn</code> .....	<a href="#">1003</a>
<code>\__color_backend_iccbased_-init:nnn</code> .....	<a href="#">843</a> , <a href="#">977</a> , <a href="#">1154</a>
<code>\c__color_backend_main_stack_int</code> ..	<a href="#">500</a>
<code>\__color_backend_pickup:N</code> ..	<a href="#">440</a> , <a href="#">463</a>
<code>\__color_backend_pickup:w</code> ..	<a href="#">14</a> , <a href="#">440</a> , <a href="#">463</a>
<code>\__color_backend_reset:</code> .....	<a href="#">599</a> , <a href="#">618</a> , <a href="#">1040</a> , <a href="#">1053</a> , <a href="#">1072</a> , <a href="#">1097</a>
<code>\__color_backend_rgb:w</code> .....	<a href="#">1123</a>
<code>\__color_backend_select:n</code> ..	<a href="#">599</a> , <a href="#">663</a>
<code>\__color_backend_select:nn</code> ..	<a href="#">618</a> , <a href="#">869</a>

\_color_backend_select_cmyk:n ..	\_color_backend_stroke_gray:n ..
..... 599, 618	..... 1029, 1063, 1099
\_color_backend_select_devicen:nn	\_color_backend_stroke_gray-
..... 662, 846, 868	aux:n ..... 1099
\_color_backend_select_gray:n ..	\_color_backend_stroke_rgb:n ...
..... 599, 618	..... 1029, 1063, 1099
\_color_backend_select_iccbased:nn	\_color_backend_stroke_rgb:w . 1099
..... 665, 850, 868	\_color_backend_stroke_separation:nn
\_color_backend_select_rgb:n ...	..... 1055, 1080, 1150
..... 599, 618	\l\_color_backend_stroke_tl ....
\_color_backend_select_separation:nn	..... 616, 627, 1039, 1050
..... 662, 846, 868	\g\_color_model_int ..... 673, 682, 830, 858, 892, 966, 998
\_color_backend_separation-	\c\_color_model_range_CIELAB_tl .
init:n ..... 666	..... 785, 820, 915, 922
\_color_backend_separation-	color.sc ..... 599, 3227
init:nn ..... 872	cs commands:
\_color_backend_separation-	\cs_generate_variant:Nn .. 49, 63,
init:nnn ..... 666	66, 99, 138, 143, 154, 185, 191, 550,
\_color_backend_separation-	586, 687, 1162, 1357, 1551, 1988,
init:nnnn ..... 666	2051, 2071, 2164, 2201, 2260, 2752,
\_color_backend_separation-	2780, 2890, 2912, 2947, 3145, 3213
init:nnnnn ..... 666, 848, 872	\cs_gset:Npx .. 2562, 2566, 3063, 3068
\_color_backend_separation-	\cs_gset_protected:Npn .... 532, 3179
init:nw ..... 666	\cs_if_exist:NTF . 27, 50, 441, 464,
\_color_backend_separation-	520, 993, 1016, 1745, 2443, 2841, 2867
init:w ..... 666	\cs_if_exist_p:N . 885, 959, 3149, 3176
\_color_backend_separation-	\cs_if_exist_use:NTF ..... 38, 700
init_/DeviceCMYK:nnn ..... 666	\cs_new:Npn 650, 709, 711, 713, 715,
\_color_backend_separation-	722, 728, 730, 736, 753, 760, 762,
init_/DeviceGray:nnn ..... 666	971, 1308, 1432, 1682, 1851, 2084,
\_color_backend_separation-	2152, 2179, 2261, 2263, 2296, 2468,
init_/DeviceRGB:nnn ..... 666	2568, 2569, 2722, 2753, 2754, 2872,
\_color_backend_separation-	2905, 2948, 2950, 2966, 2990, 3071,
init_aux:nnnnnn ..... 666	3072, 3082, 3087, 3088, 3093, 3094
\_color_backend_separation-	\cs_new:Npx ..... 635, 2589, 2624, 2781, 2792, 2859
init_CIELAB:nnn .... 666, 848, 872	\cs_new_eq:NN ..... 46,
\_color_backend_separation-	57, 59, 664, 847, 870, 871, 1059,
init_CIELAB:nnnnnn ..... 849	1060, 1084, 1085, 1152, 1153, 1161,
\_color_backend_separation-	1356, 1362, 1363, 1550, 1552, 1553,
init_count:n ..... 666	1559, 1759, 1760, 1773, 1775, 1799,
\_color_backend_separation-	1800, 1857, 1858, 1859, 1922, 1923,
init_count:w ..... 666	1932, 1933, 1934, 1954, 1957, 1958,
\_color_backend_separation-	1959, 2024, 2034, 2035, 2036, 2110,
init_Device:Nn ..... 666	2117, 2118, 2150, 2151, 2155, 2306
\g\_color_backend_stack_int ... 500	\cs_new_protected:Npn ..... 47, 54, 61, 64, 72, 78, 83, 85, 89,
\l\_color_backend_stack_int ....	100, 110, 119, 128, 141, 144, 146,
.. 497, 522, 528, 628, 631, 1038, 1051	148, 152, 157, 166, 176, 186, 197,
\_color_backend_stroke:n 1029, 1058	219, 221, 236, 252, 267, 269, 295,
\_color_backend_stroke_cmyk:n ..	309, 324, 326, 339, 353, 403, 416,
..... 1029, 1063, 1099	440, 458, 463, 471, 541, 551, 562,
\_color_backend_stroke_cmyk:w 1099	576, 587, 599, 601, 603, 605, 612,
\_color_backend_stroke_devicen:nn	
..... 1055, 1080, 1150	

618, 620, 622, 624, 630, 662, 665,  
 688, 778, 824, 843, 846, 848, 849,  
 850, 868, 872, 897, 904, 926, 977,  
 1003, 1029, 1031, 1033, 1035, 1042,  
 1044, 1046, 1048, 1055, 1057, 1063,  
 1065, 1067, 1069, 1074, 1076, 1078,  
 1080, 1082, 1088, 1090, 1092, 1094,  
 1099, 1101, 1112, 1120, 1122, 1124,  
 1150, 1151, 1154, 1155, 1163, 1168,  
 1173, 1175, 1177, 1185, 1193, 1202,  
 1212, 1214, 1217, 1219, 1236, 1241,  
 1259, 1281, 1284, 1297, 1310, 1315,  
 1317, 1319, 1321, 1323, 1325, 1327,  
 1329, 1334, 1358, 1360, 1364, 1369,  
 1374, 1384, 1393, 1395, 1398, 1400,  
 1402, 1404, 1409, 1414, 1419, 1421,  
 1434, 1439, 1441, 1443, 1445, 1447,  
 1449, 1451, 1453, 1464, 1489, 1501,  
 1513, 1525, 1532, 1554, 1560, 1565,  
 1570, 1581, 1591, 1601, 1603, 1605,  
 1607, 1638, 1640, 1645, 1647, 1649,  
 1652, 1673, 1684, 1697, 1699, 1701,  
 1703, 1705, 1707, 1709, 1711, 1713,  
 1721, 1743, 1762, 1785, 1801, 1813,  
 1818, 1826, 1852, 1865, 1882, 1892,  
 1907, 1926, 1935, 1943, 1955, 1961,  
 1964, 1979, 1989, 2028, 2037, 2043,  
 2049, 2052, 2059, 2072, 2077, 2085,  
 2092, 2111, 2119, 2125, 2127, 2129,  
 2140, 2162, 2165, 2167, 2171, 2181,  
 2202, 2207, 2212, 2217, 2227, 2232,  
 2240, 2268, 2273, 2305, 2307, 2312,  
 2314, 2319, 2334, 2339, 2376, 2405,  
 2424, 2433, 2470, 2477, 2503, 2508,  
 2536, 2548, 2560, 2564, 2570, 2572,  
 2576, 2600, 2602, 2604, 2615, 2635,  
 2645, 2668, 2682, 2692, 2703, 2724,  
 2755, 2803, 2814, 2820, 2848, 2882,  
 2884, 2891, 2893, 2897, 2907, 2913,  
 2918, 2923, 2928, 2930, 2932, 2940,  
 2953, 2969, 2971, 2988, 2992, 2994,  
 3016, 3021, 3054, 3056, 3061, 3066,  
 3073, 3075, 3079, 3080, 3081, 3083,  
 3084, 3085, 3086, 3089, 3090, 3091,  
 3092, 3095, 3096, 3102, 3107, 3112,  
 3119, 3126, 3159, 3164, 3181, 3183,  
 3189, 3195, 3216, 3218, 3220, 3222  
  
 \cs\_new\_protected:Npx .....  
 .... 501, 666, 1135, 2831, 2888, 2973  
  
 \cs\_set\_eq:NN ..... 2464, 2465  
  
 \cs\_set\_protected:Npn ..... 443, 466

## D

dim commands:

\dim\_eval:n ..... 2271, 2506,  
 2584, 2585, 2586, 2643, 2678, 2679,  
 2680, 2960, 2961, 2962, 2993, 3019  
 \dim\_max:nn ..... 2384, 2395  
 \dim\_set:Nn ... 1848, 1849, 2080, 2081  
 \dim\_to\_decimal:n .. 364, 365, 366,  
 367, 368, 370, 1563, 1568, 1574,  
 1575, 1576, 1577, 1586, 1587, 1588,  
 1679, 1698, 2145, 2146, 2382, 2393,  
 2411, 2412, 2413, 2414, 2418, 2474  
 \dim\_to\_decimal\_in\_bp:n .....  
 .... 208, 209, 210, 258, 259, 260,  
 315, 316, 317, 1181, 1182, 1189,  
 1190, 1197, 1198, 1206, 1207, 1208,  
 1305, 1309, 1313, 1367, 1372, 1378,  
 1379, 1380, 1388, 1389, 1429, 1433,  
 1437, 1683, 1767, 1768, 1769, 1770,  
 1948, 1949, 1950, 1951, 2003, 2004,  
 2005, 2006, 2134, 2135, 2136, 2137

draw internal commands:

\\_\_draw\_align\_currentpoint:... .. 36  
 \\_\_draw\_backend\_add\_to\_path:n ...  
 ..... 1560, 1606  
 \\_\_draw\_backend\_begin: .....  
 ..... 1163, 1358, 1554  
 \\_\_draw\_backend\_box\_use:Nnnnn ...  
 ..... 32, 1334, 1532, 1721  
 \\_\_draw\_backend\_cap\_but: .....  
 ..... 1297, 1421, 1673  
 \\_\_draw\_backend\_cap\_rectangle: ..  
 ..... 1297, 1421, 1673  
 \\_\_draw\_backend\_cap\_round: .....  
 ..... 1297, 1421, 1673  
 \\_\_draw\_backend\_clip: 1217, 1398, 1605  
 \\_\_draw\_backend\_closepath: .....  
 ..... 1217, 1398, 1605  
 \\_\_draw\_backend\_closestroke: ...  
 ..... 1217, 1398, 1605  
 \\_\_draw\_backend\_cm:nnnn 1329, 1342,  
 1343, 1344, 1453, 1536, 1713, 1724  
 \\_\_draw\_backend\_cm\_aux:nnnn .. 1453  
 \\_\_draw\_backend\_cm\_decompose:nnnnN  
 ..... 1459, 1488  
 \\_\_draw\_backend\_cm\_decompose\_-  
 auxi:nnnnN ..... 1488  
 \\_\_draw\_backend\_cm\_decompose\_-  
 auxii:nnnnN ..... 1488  
 \\_\_draw\_backend\_cm\_decompose\_-  
 auxiii:nnnnN ..... 1488  
 \\_\_draw\_backend\_curveto:nnnnnn ..  
 ..... 1177, 1364, 1560

<code>\__draw_backend_dash:n</code> .....	<a href="#">1297</a> , <a href="#">1421</a> , <a href="#">1673</a>
<code>\__draw_backend_dash_aux:nn</code> ..	<a href="#">1673</a>
<code>\__draw_backend_dash_pattern:nn</code> .	..... <a href="#">1297</a> , <a href="#">1421</a> , <a href="#">1673</a>
<code>\__draw_backend_discardpath:</code> ...	..... <a href="#">1217</a> , <a href="#">1398</a> , <a href="#">1605</a>
<code>\__draw_backend_end:</code> <a href="#">1163</a> , <a href="#">1358</a> , <a href="#">1554</a>	
<code>\__draw_backend_evenodd_rule:</code> ...	..... <a href="#">1212</a> , <a href="#">1393</a> , <a href="#">1601</a>
<code>\__draw_backend_fill:</code> <a href="#">1217</a> , <a href="#">1398</a> , <a href="#">1605</a>	
<code>\__draw_backend_fillstroke:</code> ....	..... <a href="#">1217</a> , <a href="#">1398</a> , <a href="#">1605</a>
<code>\__draw_backend_join_bevel:</code> ....	..... <a href="#">1297</a> , <a href="#">1421</a> , <a href="#">1673</a>
<code>\__draw_backend_join_miter:</code> ....	..... <a href="#">1297</a> , <a href="#">1421</a> , <a href="#">1673</a>
<code>\__draw_backend_join_round:</code> ....	..... <a href="#">1297</a> , <a href="#">1421</a> , <a href="#">1673</a>
<code>\__draw_backend_lineto:nn</code> .....	..... <a href="#">1177</a> , <a href="#">1364</a> , <a href="#">1560</a>
<code>\__draw_backend_linewidth:n</code> ....	..... <a href="#">1297</a> , <a href="#">1421</a> , <a href="#">1673</a>
<code>\__draw_backend_literal:n</code> .....	..... <a href="#">1161</a> , <a href="#">1166</a> , <a href="#">1170</a> , <a href="#">1174</a> , <a href="#">1176</a> , <a href="#">1179</a> , <a href="#">1187</a> , <a href="#">1195</a> , <a href="#">1204</a> , <a href="#">1218</a> , <a href="#">1221</a> , <a href="#">1222</a> , <a href="#">1223</a> , <a href="#">1224</a> , <a href="#">1227</a> , <a href="#">1233</a> , <a href="#">1243</a> , <a href="#">1250</a> , <a href="#">1256</a> , <a href="#">1261</a> , <a href="#">1266</a> , <a href="#">1267</a> , <a href="#">1268</a> , <a href="#">1269</a> , <a href="#">1272</a> , <a href="#">1278</a> , <a href="#">1288</a> , <a href="#">1294</a> , <a href="#">1299</a> , <a href="#">1312</a> , <a href="#">1316</a> , <a href="#">1318</a> , <a href="#">1320</a> , <a href="#">1322</a> , <a href="#">1324</a> , <a href="#">1326</a> , <a href="#">1328</a> , <a href="#">1331</a> , <a href="#">1336</a> , <a href="#">1337</a> , <a href="#">1338</a> , <a href="#">1339</a> , <a href="#">1340</a> , <a href="#">1341</a> , <a href="#">1345</a> , <a href="#">1346</a> , <a href="#">1348</a> , <a href="#">1349</a> , <a href="#">1350</a> , <a href="#">1351</a> , <a href="#">1352</a> , <a href="#">1356</a> , <a href="#">1366</a> , <a href="#">1371</a> , <a href="#">1376</a> , <a href="#">1386</a> , <a href="#">1399</a> , <a href="#">1401</a> , <a href="#">1403</a> , <a href="#">1406</a> , <a href="#">1411</a> , <a href="#">1416</a> , <a href="#">1420</a> , <a href="#">1423</a> , <a href="#">1436</a> , <a href="#">1440</a> , <a href="#">1442</a> , <a href="#">1444</a> , <a href="#">1446</a> , <a href="#">1448</a> , <a href="#">1450</a> , <a href="#">1452</a> , <a href="#">1550</a> , <a href="#">1612</a> , <a href="#">1631</a> , <a href="#">1657</a>
<code>\__draw_backend_miterlimit:n</code> ...	..... <a href="#">1297</a> , <a href="#">1421</a> , <a href="#">1673</a>
<code>\__draw_backend_moveto:nn</code> .....	..... <a href="#">1177</a> , <a href="#">1364</a> , <a href="#">1560</a>
<code>\__draw_backend_nonzero_rule:</code> ...	..... <a href="#">1212</a> , <a href="#">1393</a> , <a href="#">1601</a>
<code>\__draw_backend_path:n</code> .....	<a href="#">1605</a>
<code>\g__draw_backend_path_int</code> <a href="#">1620</a> , <a href="#">1637</a>	
<code>\g__draw_backend_path_tl</code> .....	..... <a href="#">1560</a> , <a href="#">1616</a> , <a href="#">1632</a> , <a href="#">1634</a> , <a href="#">1661</a>
<code>\__draw_backend_rectangle:nnnn</code> ..	..... <a href="#">1177</a> , <a href="#">1364</a> , <a href="#">1560</a>
<code>\__draw_backend_scope_begin:</code> ...	..... <a href="#">1173</a> , <a href="#">1359</a> , <a href="#">1362</a> , <a href="#">1552</a>
<code>\__draw_backend_scope_end:</code> .....	..... <a href="#">1173</a> , <a href="#">1361</a> , <a href="#">1362</a> , <a href="#">1552</a>
<code>\__draw_backend_stroke:</code> .....	..... <a href="#">1217</a> , <a href="#">1398</a> , <a href="#">1605</a>
<code>\g__draw_draw_clip_bool</code> ..	<a href="#">1217</a> , <a href="#">1605</a>
<code>\g__draw_draw_eor_bool</code> .....	... <a href="#">1212</a> , <a href="#">1229</a> , <a href="#">1245</a> , <a href="#">1252</a> , <a href="#">1263</a> , <a href="#">1274</a> , <a href="#">1290</a> , <a href="#">1393</a> , <a href="#">1407</a> , <a href="#">1412</a> , <a href="#">1417</a>
<code>\g__draw_draw_path_int</code> .....	<a href="#">1605</a>
<code>\g__draw_path_tl</code> .....	<a href="#">1670</a>
<b>E</b>	
<code>\errmessage</code> .....	<a href="#">38</a>
<code>\evensidemargin</code> .....	<a href="#">2351</a>
exp commands:	
<code>\exp_after:wN</code> .....	<a href="#">449</a> , <a href="#">2090</a>
<code>\exp_args:Ne</code> .....	... <a href="#">724</a> , <a href="#">1820</a> , <a href="#">1873</a> , <a href="#">1898</a> , <a href="#">2505</a> , <a href="#">3018</a>
<code>\exp_args:Nf</code> .....	<a href="#">1302</a> , <a href="#">1426</a> , <a href="#">2270</a>
<code>\exp_args:NNf</code> .....	<a href="#">220</a> , <a href="#">268</a> , <a href="#">325</a>
<code>\exp_args:Nnx</code> .....	<a href="#">2257</a> , <a href="#">2943</a>
<code>\exp_args:NV</code> .....	<a href="#">445</a>
<code>\exp_args:Nx</code> .	<a href="#">670</a> , <a href="#">882</a> , <a href="#">1871</a> , <a href="#">1896</a> , <a href="#">2214</a> , <a href="#">2229</a> , <a href="#">2347</a> , <a href="#">2909</a> , <a href="#">3104</a> , <a href="#">3161</a>
<code>\exp_last_unbraced:Nx</code> .....	<a href="#">454</a> , <a href="#">468</a>
<code>\exp_not:N</code> .....	<a href="#">503</a> , <a href="#">504</a> , <a href="#">512</a> , <a href="#">514</a> , <a href="#">637</a> , <a href="#">643</a> , <a href="#">644</a> , <a href="#">645</a> , <a href="#">672</a> , <a href="#">673</a> , <a href="#">676</a> , <a href="#">677</a> , <a href="#">682</a> , <a href="#">2591</a> , <a href="#">2593</a> , <a href="#">2596</a> , <a href="#">2626</a> , <a href="#">2628</a> , <a href="#">2631</a> , <a href="#">2783</a> , <a href="#">2785</a> , <a href="#">2788</a> , <a href="#">2794</a> , <a href="#">2796</a> , <a href="#">2799</a> , <a href="#">2836</a> , <a href="#">2837</a> , <a href="#">2843</a> , <a href="#">2844</a> , <a href="#">2863</a> , <a href="#">2868</a> , <a href="#">2975</a> , <a href="#">2980</a>
<code>\exp_not:n</code> <a href="#">48</a> , <a href="#">97</a> , <a href="#">108</a> , <a href="#">136</a> , <a href="#">985</a> , <a href="#">2205</a> , <a href="#">2210</a> , <a href="#">2499</a> , <a href="#">2738</a> , <a href="#">2739</a> , <a href="#">2753</a> , <a href="#">2754</a> , <a href="#">2766</a> , <a href="#">2767</a> , <a href="#">2921</a> , <a href="#">2926</a> , <a href="#">2937</a> , <a href="#">2998</a>	
<code>\ExplBackendFileDate</code> .....	<a href="#">1</a>
<b>F</b>	
file commands:	
<code>\file_compare_timestamp:nNnTF</code> .	<a href="#">1884</a>
<code>\file_parse_full_name:nNNN</code> <a href="#">1867</a> , <a href="#">1894</a>	
<code>\fmtversion</code> .....	<a href="#">52</a>
fp commands:	
<code>\fp_compare:nNnTF</code> .....	... <a href="#">227</a> , <a href="#">274</a> , <a href="#">280</a> , <a href="#">332</a> , <a href="#">1469</a> , <a href="#">1482</a> , <a href="#">1527</a>
<code>\fp_eval:n</code> .	<a href="#">220</a> , <a href="#">229</a> , <a href="#">242</a> , <a href="#">243</a> , <a href="#">268</a> , <a href="#">285</a> , <a href="#">300</a> , <a href="#">302</a> , <a href="#">325</a> , <a href="#">334</a> , <a href="#">345</a> , <a href="#">346</a> , <a href="#">410</a> , <a href="#">425</a> , <a href="#">426</a> , <a href="#">1107</a> , <a href="#">1108</a> , <a href="#">1109</a> , <a href="#">1117</a> , <a href="#">1130</a> , <a href="#">1131</a> , <a href="#">1132</a> , <a href="#">1471</a> , <a href="#">1476</a> , <a href="#">1477</a> , <a href="#">1484</a> , <a href="#">1494</a> , <a href="#">1495</a> , <a href="#">1496</a> , <a href="#">1497</a> , <a href="#">1506</a> , <a href="#">1507</a> , <a href="#">1508</a> , <a href="#">1509</a> , <a href="#">1518</a> , <a href="#">1519</a> , <a href="#">1520</a> , <a href="#">1521</a> , <a href="#">2496</a> , <a href="#">2665</a> , <a href="#">3012</a> , <a href="#">3105</a> , <a href="#">3115</a> , <a href="#">3122</a> , <a href="#">3162</a> , <a href="#">3186</a> , <a href="#">3193</a> , <a href="#">3223</a>
<code>\fp_new:N</code> .....	<a href="#">293</a> , <a href="#">294</a>
<code>\fp_set:Nn</code> .....	<a href="#">273</a> , <a href="#">276</a>
<code>\fp_use:N</code> .....	<a href="#">279</a> , <a href="#">283</a> , <a href="#">288</a>
<code>\fp_zero:N</code> .....	<a href="#">275</a>
<code>\c_zero_fp</code> <a href="#">227</a> , <a href="#">274</a> , <a href="#">280</a> , <a href="#">332</a> , <a href="#">1469</a> , <a href="#">1482</a>	

## G

### graphics commands:

\l\_graphics\_pagebox\_tl ..... [54](#)  
\l\_graphics\_search\_ext\_seq .....  
..... [1755](#), [1778](#), [1915](#), [2103](#)

### graphics internal commands:

\\_\_graphics\_backend\_dequote:w . [1785](#)  
\l\_graphics\_backend\_dir\_str . [1860](#)  
\l\_graphics\_backend\_ext\_str . [1860](#)  
\\_\_graphics\_backend\_get\_pagecount:n  
..... [1774](#), [1907](#), [2022](#), [2092](#), [2154](#)  
\\_\_graphics\_backend\_getbb\_auxi:n  
..... [1785](#)  
\\_\_graphics\_backend\_getbb\_-  
auxi:nN ..... [2028](#)  
\\_\_graphics\_backend\_getbb\_-  
auxii:n ..... [1785](#)  
\\_\_graphics\_backend\_getbb\_-  
auxii:nnN ..... [2028](#)  
\\_\_graphics\_backend\_getbb\_-  
auxiii:n ..... [1785](#)  
\\_\_graphics\_backend\_getbb\_-  
auxiii:nNnn ..... [2028](#)  
\\_\_graphics\_backend\_getbb\_-  
auxiv:nnNnn ..... [2028](#)  
\\_\_graphics\_backend\_getbb\_-  
auxv:nNnn ..... [2028](#)  
\\_\_graphics\_backend\_getbb\_-  
auxvi:nNnn ..... [2075](#), [2077](#)  
\\_\_graphics\_backend\_getbb\_bmp:n .  
..... [1920](#), [2028](#)  
\\_\_graphics\_backend\_getbb\_eps:n .  
..... [1757](#), [1860](#), [1920](#), [2109](#)  
\\_\_graphics\_backend\_getbb\_eps:nm  
..... [1860](#)  
\\_\_graphics\_backend\_getbb\_eps:nn  
..... [1871](#), [1882](#)  
\\_\_graphics\_backend\_getbb\_jpeg:n  
..... [1785](#), [1920](#), [2028](#), [2111](#)  
\\_\_graphics\_backend\_getbb\_jpg:n .  
..... [1785](#), [1920](#), [2028](#), [2111](#)  
\\_\_graphics\_backend\_getbb\_-  
pagebox:w ..... [2028](#), [2090](#)  
\\_\_graphics\_backend\_getbb\_pdf:n .  
..... [1785](#), [1890](#), [1920](#), [2028](#), [2119](#)  
\\_\_graphics\_backend\_getbb\_png:n .  
..... [1785](#), [1920](#), [2028](#), [2111](#)  
\\_\_graphics\_backend\_getbb\_ps:n .  
..... [1757](#), [1923](#)  
\\_\_graphics\_backend\_include:nn [2125](#)  
\\_\_graphics\_backend\_include\_-  
auxi:nn ..... [1943](#)  
\\_\_graphics\_backend\_include\_-  
auxii:nnn ..... [1943](#)

\\_\_graphics\_backend\_include\_-  
auxiii:nnn ..... [1943](#)  
\\_\_graphics\_backend\_include\_-  
bitmap\_quote:w ..... [2140](#)  
\\_\_graphics\_backend\_include\_-  
bmp:n ..... [1943](#)  
\\_\_graphics\_backend\_include\_-  
eps:n ..... [1762](#), [1860](#), [1943](#), [2125](#)  
\\_\_graphics\_backend\_include\_-  
jpeg:n ..... [1852](#), [1957](#), [2140](#)  
\\_\_graphics\_backend\_include\_-  
jpg:n ..... [1852](#), [1943](#), [2140](#)  
\\_\_graphics\_backend\_include\_-  
jpseg:n ..... [1943](#)  
\\_\_graphics\_backend\_include\_-  
pdf:n .. [1852](#), [1896](#), [1943](#), [2085](#), [2125](#)  
\\_\_graphics\_backend\_include\_-  
png:n ..... [1852](#), [1943](#), [2140](#)  
\\_\_graphics\_backend\_include\_ps:n  
..... [1762](#), [1943](#)  
\\_\_graphics\_backend\_loaded:n ...  
... [1743](#), [1755](#), [1757](#), [1774](#), [1778](#),  
[1915](#), [1920](#), [2023](#), [2103](#), [2109](#), [2154](#)  
\l\_graphics\_backend\_name\_str . [1860](#)  
\\_\_graphics\_bb\_restore:nTF [1815](#), [2074](#)  
\\_\_graphics\_bb\_save:n ... [1824](#), [2082](#)  
\l\_graphics\_decodearray\_str ...  
..... [1791](#), [1792](#),  
[1803](#), [1831](#), [1835](#), [1836](#), [1937](#), [1972](#),  
[1973](#), [2011](#), [2014](#), [2015](#), [2039](#), [2121](#)  
\\_\_graphics\_extract\_bb:n .....  
..... [1930](#), [1939](#), [2115](#), [2123](#)  
\l\_graphics\_final\_name\_str .. [1889](#)  
\\_\_graphics\_get\_pagecount:n ....  
..... [1775](#), [2024](#), [2155](#)  
\l\_graphics\_graphics\_attr\_tl ...  
..... [1784](#), [1789](#),  
[1796](#), [1805](#), [1815](#), [1822](#), [1824](#), [1855](#)  
\l\_graphics\_internal\_box .....  
.. [1846](#), [1848](#), [1849](#), [2079](#), [2080](#), [2081](#)  
\l\_graphics\_interpolate\_bool ...  
..... [1793](#), [1804](#), [1830](#), [1837](#),  
[1938](#), [1974](#), [2010](#), [2016](#), [2040](#), [2122](#)  
\l\_graphics\_llx\_dim .....  
..... [1767](#), [1948](#), [2003](#), [2134](#)  
\l\_graphics\_lly\_dim .....  
..... [1768](#), [1949](#), [2004](#), [2135](#)  
\l\_graphics\_page\_int .....  
..... [1787](#), [1808](#), [1809](#), [1841](#),  
[1842](#), [1928](#), [1970](#), [1971](#), [1997](#), [1998](#),  
[2030](#), [2045](#), [2046](#), [2088](#), [2089](#), [2113](#)  
\l\_graphics\_pagebox\_tl [1788](#), [1807](#),  
[1843](#), [1844](#), [1929](#), [1968](#), [1969](#), [1999](#),  
[2001](#), [2031](#), [2054](#), [2055](#), [2090](#), [2114](#)

<pre> \__graphics_read_bb:n ..... ..... 1759, 1760, 1922, 1923, 2110 \g__graphics_track_int ..... ..... 1942, 1991, 1992 \l__graphics_urx_dim ..... .. 1769, 1848, 1950, 2005, 2080, 2136 \l__graphics_ury_dim . 1770, 1849, 1951, 2006, 2081, 2137, 2145, 2146 group commands: \group_begin: ..... 163, 182 \group_end: ..... 171 \group_insert_after:N ..... .. 1040, 1053, 1072, 1097, 3173, 3210 </pre>	<pre> 830, 858, 892, 966, 998, 1614, 1620, 1627, 1659, 1667, 1809, 1842, 1855, 1911, 1971, 1984, 1996, 1998, 2089, 2097, 2180, 2245, 2258, 2262, 2290, 2297, 2368, 2469, 2723, 2733, 2906, 2944, 2949, 2959, 2967, 2980, 2991 \int_value:w ..... ..... 2591, 2626, 2783, 2794, 2812 \int_zero:N ... 1787, 1928, 2030, 2113 </pre>
---	---

  

### H

hbox commands:

```

\hbox:n ..... 2276, 2279,
2354, 2360, 2513, 2520, 3026, 3037
\hbox_overlap_right:n ..... 215,
247, 263, 304, 320, 348, 432, 1347, 1542
\hbox_set:Nn .. 1846, 2079, 2346, 2378
\hbox_set:Nw ..... 2329
\hbox_set_end: ..... 2344
\hbox_unpack:N ..... 2465
hook commands:
\hook_gput_code:nnn .. 55, 1745, 1747

```

### K

kernel internal commands:

```

\__kernel_backend_align_begin: ..
..... 72, 200, 224, 239
\__kernel_backend_align_end: ...
..... 72, 214, 232, 246
\__kernel_backend_first_shipout:n
..... 50, 69, 507, 670
\g__kernel_backend_header_bool ..
..... 67, 668
\__kernel_backend_literal:n ....
..... 46, 62, 65, 70, 74, 81,
84, 86, 142, 145, 147, 149, 153, 329,
342, 509, 534, 535, 543, 553, 607,
613, 690, 826, 1071, 1096, 1165,
1171, 1466, 1473, 1479, 1539, 1544,
1764, 1945, 1983, 1993, 2131, 2142,
2889, 2993, 3055, 3059, 3064, 3069
\__kernel_backend_literal_page:n
.... 100, 144, 2883, 2885, 3074, 3076
\__kernel_backend_literal_pdf:n .
..... 89, 141, 255, 312, 1356
\__kernel_backend_literal_-
postscript:n .....
.... 61, 75, 76, 80, 201, 202, 204,
205, 213, 225, 240, 1161, 2540, 2552
\__kernel_backend_literal_svg:n .
..... 152, 159, 170, 178, 188,
356, 358, 375, 852, 1550, 1725, 1736
\__kernel_backend_matrix:n .....
..... 128, 277, 298, 1456
\__kernel_backend_postscript:n ..
..... 64,
609, 1075, 1077, 1079, 1083, 2163,
2219, 2234, 2276, 2282, 2322, 2354,
2361, 2365, 2379, 2407, 2451, 2458,
2464, 2472, 2479, 2513, 2520, 3128
\__kernel_backend_scope:n 157, 385,
390, 1137, 1557, 1602, 1604, 1624,
1664, 1686, 1698, 1700, 1702, 1704,
1706, 1708, 1710, 1712, 1715, 3223
\__kernel_backend_scope_begin: ..
..... 83, 110, 146, 157,

```

  

### I

int commands:

```

\int_compare:nNnTF .....
..... 1808, 1841, 1970, 1997,
2045, 2088, 2437, 2538, 2834, 2862
\int_const:Nn . 504, 530, 564, 1822,
1910, 1992, 2094, 2174, 2712, 2900
\int_eval:n .....
. 546, 556, 584, 595, 720, 729, 742,
744, 748, 761, 2562, 2566, 2812,
2837, 2844, 2857, 3055, 3063, 3068
\int_gincr:N ..... 189, 355,
503, 1611, 1656, 1991, 2173, 2242,
2286, 2363, 2899, 2942, 2955, 2975
\int_gset:Nn ..... 164, 183, 2426
\int_gset_eq:NN 172, 2287, 2364, 2956
\int_if_exist:NnTF ..... 1981
\int_if_odd:nTF ..... 2349
\int_max:nn ..... 2096
\int_new:N ..... 155, 156,
402, 497, 500, 1637, 1942, 2169,
2267, 2298, 2300, 2895, 2952, 2968
\int_set:Nn ..... 522
\int_set_eq:NN ... 160, 179, 528, 2438
\int_step_function:nnnN ..... 746
\int_use:N .....
. 357, 388, 512, 523, 538, 673, 682,

```

[199](#), [223](#), [238](#), [254](#), [271](#), [297](#), [311](#),  
[328](#), [341](#), [1362](#), [1534](#), [1552](#), [1556](#), [1723](#)  
`\__kernel_backend_scope_begin:n` .  
[157](#), [377](#), [405](#), [418](#)  
`\__kernel_backend_scope_end:` [83](#),  
[110](#), [146](#), [157](#), [216](#), [234](#), [248](#), [264](#),  
[291](#), [305](#), [321](#), [337](#), [349](#), [400](#), [414](#),  
[433](#), [532](#), [1363](#), [1546](#), [1553](#), [1559](#), [1737](#)  
`\g__kernel_backend_scope_int` . . . .  
[155](#), [162](#), [164](#), [169](#), [173](#), [181](#), [183](#), [189](#)  
`\l__kernel_backend_scope_int` . . . .  
[155](#), [161](#), [174](#), [180](#)  
`\g__kernel_clip_path_int` . . . . .  
[353](#), [1611](#), [1614](#), [1627](#), [1656](#), [1659](#), [1667](#)  
`\__kernel_color_backend_stack_-`  
`init:Nnn` . . . . . [500](#), [562](#), [3152](#)  
`\__kernel_color_backend_stack_-`  
`pop:n` . . . . . [541](#), [576](#), [631](#), [3182](#)  
`\__kernel_color_backend_stack_-`  
`push:nn` . . . . . [541](#),  
[576](#), [628](#), [1038](#), [1051](#), [3171](#), [3208](#)  
`\__kernel_dependency_version_-`  
`check:Nn` . . . . . [1](#)  
`\__kernel_dependency_version_-`  
`check:nn` . . . . . [27](#), [29](#)  
`\__kernel_file_name_quote:n` . . . .  
. . . . . [1873](#), [1898](#)  
`\__kernel_kern:n` . . . . .  
. . . . . [2281](#), [2283](#), [2512](#), [2516](#),  
[2519](#), [2523](#), [3025](#), [3033](#), [3036](#), [3052](#)

## M

`\MessageBreak` . . . . . [40](#)  
mode commands:  
`\mode_if_horizontal:TF` . . . [2428](#), [2435](#)  
`\mode_if_math:TF` . . . . . [2326](#)

## O

`\oddsidemargin` . . . . . [2350](#)  
opacity internal commands:  
`\__opacity_backend:nn` . . . . . [3216](#)  
`\__opacity_backend:nnn` . . . . . [3102](#)  
`\__opacity_backend_fill:n` . . . . .  
. . . . . [3102](#), [3183](#), [3216](#)  
`\__opacity_backend_fill_stroke:nn`  
. . . . . [3185](#), [3191](#), [3195](#), [3213](#)  
`\l__opacity_backend_fill_tl` . . . .  
. . . . . [3157](#), [3166](#), [3192](#), [3200](#)  
`\__opacity_backend_fillstroke:nn`  
. . . . . [3183](#)  
`\__opacity_backend_reset:` [3159](#), [3210](#)  
`\__opacity_backend_select:n` . . . .  
. . . . . [3102](#), [3159](#), [3216](#)

`\__opacity_backend_select_aux:n` .  
. . . . . [3102](#), [3159](#), [3198](#)  
`\c__opacity_backend_stack_int` . . . .  
. . . . . [3148](#), [3171](#), [3182](#), [3208](#)  
`\__opacity_backend_stroke:n` . . . .  
. . . . . [3102](#), [3183](#), [3216](#)  
`\l__opacity_backend_stroke_tl` . . . .  
. . . . . [3157](#), [3167](#), [3187](#), [3201](#)

## P

pdf commands:

`\pdf_object_if_exist:nTF` . . . . .  
. . . . . [906](#), [979](#), [1005](#)  
`\pdf_object_new:nn` . . . [908](#), [981](#), [1007](#)  
`\pdf_object_ref:n` . . . . [921](#), [992](#), [1015](#)  
`\pdf_object_ref_last:` . . . . .  
. . . [893](#), [900](#), [902](#), [955](#), [967](#), [999](#), [1023](#)  
`\pdf_object_unnamed_write:nn` . . . .  
. . . . . [874](#), [899](#), [928](#), [950](#), [991](#), [1014](#)  
`\pdf_object_write:nn` . [909](#), [982](#), [1008](#)

pdf internal commands:

`\__pdf_backend:n` . . . . . [2888](#),  
[2892](#), [2894](#), [2920](#), [2925](#), [2934](#), [2957](#),  
[2976](#), [2989](#), [2996](#), [3028](#), [3029](#), [3039](#)  
`\__pdf_backend_annotation:nnnn` . . . .  
. . . . . [2268](#), [2576](#), [2953](#)  
`\__pdf_backend_annotation_-`  
`aux:nnnn` . . . . . [2270](#), [2273](#)  
`\g__pdf_backend_annotation_int` . . . .  
. . . [2267](#), [2287](#), [2297](#), [2952](#), [2956](#), [2967](#)  
`\__pdf_backend_annotation_last:` . . . .  
. . . . . [2296](#), [2589](#), [2966](#)  
`\__pdf_backend_bdc:nn` . . . . .  
. . . . . [2570](#), [2882](#), [3073](#), [3095](#)  
`\__pdf_backend_catalog_gput:nn` . . . .  
. . . . . [2165](#), [2682](#), [2891](#), [3079](#)  
`\__pdf_backend_compress_objects:n`  
. . . . . [2536](#), [2803](#), [3054](#), [3089](#)  
`\__pdf_backend_compresslevel:n` . . . .  
. . . . . [2536](#), [2803](#), [3054](#), [3089](#)  
`\l__pdf_backend_content_box` [2265](#),  
[2329](#), [2353](#), [2356](#), [2358](#), [2387](#), [2398](#)  
`\__pdf_backend_destination:nn` . . . .  
. . . . . [2477](#), [2645](#), [2994](#)  
`\__pdf_backend_destination:nnnn` . . . .  
. . . . . [2477](#), [2645](#), [2994](#)  
`\__pdf_backend_destination_-`  
`aux:nnnn` . . . . . [2477](#), [2994](#)  
`\__pdf_backend_emc:` . . . . .  
. . . . . [2570](#), [2882](#), [3073](#), [3095](#)  
`\__pdf_backend_info_gput:nn` . . . . .  
. . . . . [2165](#), [2682](#), [2891](#), [3079](#)  
`\__pdf_backend_link:nw` . . . . . [2307](#)  
`\__pdf_backend_link_aux:nw` . . . [2307](#)

<code>\__pdf_backend_link_begin:n</code> ..	<a href="#">2969</a>	<code>\__pdf_backend_object_write:nnn</code>	<a href="#">2907</a>
<code>\__pdf_backend_link_begin:nnnw</code>	<a href="#">2600</a>	<code>\__pdf_backend_object_write_-</code>	
<code>\__pdf_backend_link_begin:nw</code> ...		<code>array:nn</code> .....	<a href="#">2181</a> , <a href="#">2907</a>
.....	<a href="#">2309</a> , <a href="#">2313</a> , <a href="#">2314</a>	<code>\__pdf_backend_object_write_-</code>	
<code>\__pdf_backend_link_begin_aux:nw</code>		<code>dict:nn</code> .....	<a href="#">2181</a> , <a href="#">2907</a>
.....	<a href="#">2317</a> , <a href="#">2319</a>	<code>\__pdf_backend_object_write_-</code>	
<code>\__pdf_backend_link_begin_-</code>		<code>fstream:nn</code> .....	<a href="#">2181</a> , <a href="#">2907</a>
<code>goto:nnw</code> .....	<a href="#">2307</a> , <a href="#">2600</a> , <a href="#">2969</a>	<code>\__pdf_backend_object_write_-</code>	
<code>\__pdf_backend_link_begin_-</code>		<code>fstream:nnn</code> .....	<a href="#">2215</a> , <a href="#">2217</a>
<code>user:nnw</code> .....	<a href="#">2307</a> , <a href="#">2600</a> , <a href="#">2969</a>	<code>\__pdf_backend_object_write_-</code>	
<code>\g__pdf_backend_link_bool</code> .....		<code>stream:nn</code> .....	<a href="#">2181</a> , <a href="#">2907</a>
.....	<a href="#">2302</a> , <a href="#">2316</a> , <a href="#">2321</a> , <a href="#">2336</a> , <a href="#">2374</a>	<code>\__pdf_backend_object_write_-</code>	
<code>\g__pdf_backend_link_dict_tl</code> ...		<code>stream:nnn</code> .....	<a href="#">2181</a>
.....	<a href="#">2299</a> , <a href="#">2324</a> , <a href="#">2369</a>	<code>\__pdf_backend_object_write_-</code>	
<code>\__pdf_backend_link_end:</code> .....		<code>stream:nnnn</code> .....	<a href="#">2907</a>
.....	<a href="#">2307</a> , <a href="#">2600</a> , <a href="#">2969</a>	<code>\__pdf_backend_pageobject_ref:n</code> .	
<code>\__pdf_backend_link_end_aux:</code> .	<a href="#">2307</a>	.....	<a href="#">2263</a> , <a href="#">2792</a> , <a href="#">2950</a> , <a href="#">3081</a>
<code>\g__pdf_backend_link_int</code> .....		<code>\__pdf_backend_pdfmark:n</code> .....	
.....	<a href="#">2298</a> , <a href="#">2364</a> , <a href="#">2368</a> , <a href="#">2469</a> , <a href="#">2968</a> , <a href="#">2975</a> , <a href="#">2980</a> , <a href="#">2991</a>	.....	<a href="#">2162</a> , <a href="#">2166</a> , <a href="#">2168</a> , <a href="#">2183</a> , <a href="#">2204</a> , <a href="#">2209</a> , <a href="#">2243</a> , <a href="#">2288</a> , <a href="#">2480</a> , <a href="#">2524</a> , <a href="#">2571</a> , <a href="#">2573</a>
<code>\__pdf_backend_link_last:</code> .....		<code>\__pdf_backend_version_major:</code> ...	
.....	<a href="#">2468</a> , <a href="#">2624</a> , <a href="#">2990</a>	.....	<a href="#">2562</a> , <a href="#">2568</a> , <a href="#">2859</a> , <a href="#">3063</a> , <a href="#">3064</a> , <a href="#">3071</a> , <a href="#">3093</a>
<code>\__pdf_backend_link_margin:n</code> ...		<code>\__pdf_backend_version_major_-</code>	
.....	<a href="#">2470</a> , <a href="#">2635</a> , <a href="#">2992</a>	<code>gset:n</code> .....	<a href="#">2560</a> , <a href="#">2831</a> , <a href="#">3061</a> , <a href="#">3091</a>
<code>\g__pdf_backend_link_math_bool</code> .		<code>\__pdf_backend_version_minor:</code> ...	
.....	<a href="#">2301</a> , <a href="#">2327</a> , <a href="#">2328</a> , <a href="#">2331</a> , <a href="#">2341</a>	.....	<a href="#">2566</a> , <a href="#">2568</a> , <a href="#">2859</a> , <a href="#">3068</a> , <a href="#">3069</a> , <a href="#">3071</a> , <a href="#">3093</a>
<code>\__pdf_backend_link_minima:</code> ..	<a href="#">2307</a>	<code>\__pdf_backend_version_minor_-</code>	
<code>\__pdf_backend_link_outerbox:n</code>	<a href="#">2307</a>	<code>gset:n</code> .....	<a href="#">2560</a> , <a href="#">2831</a> , <a href="#">3061</a> , <a href="#">3091</a>
<code>\g__pdf_backend_link_sf_int</code> ...		<code>\l__pdf_breaklink_pdfmark_tl</code> ...	
.....	<a href="#">2300</a> , <a href="#">2426</a> , <a href="#">2437</a> , <a href="#">2438</a>	.....	<a href="#">2303</a> , <a href="#">2371</a> , <a href="#">2463</a>
<code>\__pdf_backend_link_sf_restore:</code>	<a href="#">2307</a>	<code>\__pdf_breaklink_postscript:n</code> ...	
<code>\__pdf_backend_link_sf_save:</code> .	<a href="#">2307</a>	.....	<a href="#">2305</a> , <a href="#">2355</a> , <a href="#">2357</a> , <a href="#">2464</a>
<code>\l__pdf_backend_model_box</code> .	<a href="#">2266</a> , <a href="#">2346</a> , <a href="#">2378</a> , <a href="#">2386</a> , <a href="#">2397</a> , <a href="#">2412</a> , <a href="#">2414</a>	<code>\__pdf_breaklink_usebox:N</code> .....	
<code>\__pdf_backend_objcompresslevel:n</code>		.....	<a href="#">2306</a> , <a href="#">2356</a> , <a href="#">2465</a>
.....	<a href="#">2803</a>	<code>\__pdf_exp_not_i:nn</code> .	<a href="#">2724</a> , <a href="#">2770</a> , <a href="#">2775</a>
<code>\g__pdf_backend_object_int</code> .....		<code>\__pdf_exp_not_ii:nn</code>	<a href="#">2724</a> , <a href="#">2771</a> , <a href="#">2776</a>
.....	<a href="#">2169</a> , <a href="#">2173</a> , <a href="#">2176</a> , <a href="#">2242</a> , <a href="#">2245</a> , <a href="#">2258</a> , <a href="#">2262</a> , <a href="#">2286</a> , <a href="#">2287</a> , <a href="#">2290</a> , <a href="#">2363</a> , <a href="#">2364</a> , <a href="#">2895</a> , <a href="#">2899</a> , <a href="#">2902</a> , <a href="#">2942</a> , <a href="#">2944</a> , <a href="#">2949</a> , <a href="#">2955</a> , <a href="#">2956</a> , <a href="#">2959</a>	<code>\l__pdf_internal_box</code> .....	<a href="#">2160</a>
<code>\__pdf_backend_object_last:</code> ...		<code>pdf.baselineskip</code> .....	<a href="#">2307</a> , <a href="#">3554</a>
.....	<a href="#">2261</a> , <a href="#">2781</a> , <a href="#">2948</a> , <a href="#">3081</a>	<code>pdf.bordertracking</code> .....	<a href="#">3312</a>
<code>\__pdf_backend_object_new:nn</code> ...		<code>pdf.bordertracking.begin</code> .....	<a href="#">3312</a>
.....	<a href="#">2171</a> , <a href="#">2703</a> , <a href="#">2897</a> , <a href="#">3081</a>	<code>pdf.bordertracking.continue</code> .....	<a href="#">3312</a>
<code>\__pdf_backend_object_now:nn</code> ...		<code>pdf.bordertracking.end</code> .....	<a href="#">3312</a>
.....	<a href="#">2240</a> , <a href="#">2755</a> , <a href="#">2940</a> , <a href="#">3081</a>	<code>pdf.bordertracking.endpage</code> .....	<a href="#">3312</a>
<code>\g__pdf_backend_object_prop</code> ...		<code>pdf.breaklink</code> .....	<a href="#">3450</a>
.....	<a href="#">2169</a> , <a href="#">2177</a> , <a href="#">2188</a> , <a href="#">2198</a> , <a href="#">2702</a> , <a href="#">2720</a> , <a href="#">2736</a> , <a href="#">2895</a> , <a href="#">2903</a> , <a href="#">2910</a>	<code>pdf.breaklink.write</code> .....	<a href="#">3450</a>
<code>\__pdf_backend_object_ref:n</code>	<a href="#">2171</a> , <a href="#">2185</a> , <a href="#">2199</a> , <a href="#">2703</a> , <a href="#">2897</a> , <a href="#">2916</a> , <a href="#">3081</a>	<code>pdf.brokenlink.dict</code> .....	<a href="#">3312</a>
<code>\__pdf_backend_object_write:nn</code> ..		<code>pdf.brokenlink.rect</code> .....	<a href="#">3312</a>
.....	<a href="#">2181</a> , <a href="#">2724</a> , <a href="#">2907</a> , <a href="#">3081</a>	<code>pdf.brokenlink.skip</code> .....	<a href="#">3312</a>
		<code>pdf.count</code> .....	<a href="#">3450</a>
		<code>pdf.currentrect</code> .....	<a href="#">3450</a>
		<code>pdf.cvs</code> .....	<a href="#">3234</a>
		<code>pdf.dest.anchor</code> .....	<a href="#">3277</a>

pdf.dest.point	3277		
pdf.dest.x	3277		
pdf.dest.y	3277		
pdf.dest2device	3277		
pdf.dev.x	3277		
pdf.dev.y	3277		
pdf.dvi.pt	3234		
pdf.globaldict	3231		
pdf.leftboundary	3312		
pdf.link.dict	2307		
pdf.linkdp.pad	2307, 3238		
pdf.linkht.pad	2307, 3238		
pdf.linkmargin	3238		
pdf.llx	2307, 3241		
pdf.lly	2307, 3241		
pdf.originx	3312		
pdf.originy	3312		
pdf.outerbox	2307, 3554		
pdf.pdfmark	3554		
pdf.pdfmark.dict	3554		
pdf.pdfmark.good	3554		
pdf.pt.dvi	3234		
pdf.rect	3241		
pdf.rect.ht	3234		
pdf.rightboundary	3312		
pdf.save.linkll	3241		
pdf.save.linkur	3241		
pdf.save.ll	3241		
pdf.save.ur	3241		
pdf.tmpa	3277		
pdf.tmpb	3277		
pdf.tmpc	3277		
pdf.tmpd	3277		
pdf.urx	3241		
pdf.ury	2307, 3241		
pdfmanagement commands:			
\pdfmanagement_add:nnn	890, 964, 993,		
	997, 1016, 1020, 3154, 3168, 3202, 3205		
\pdfmanagement_if_active_p:	885,		
	886, 959, 960, 3149, 3150, 3176, 3177		
prg commands:			
\prg_replicate:nn	168, 718, 739, 749, 934		
prop commands:			
\prop_gput:Nnn	676, 901, 2177, 2720, 2903		
\prop_if_in:NnTF	653		
\prop_item:Nn	656, 2188, 2198, 2736, 2910		
\prop_new:N	634, 2170, 2702, 2896		
\ProvidesExplFile	2		
		<b>Q</b>	
		quark commands:	
		\quark_if_recursion_tail_stop:n	652
		\q_recursion_stop	645
		\q_recursion_tail	644
		<b>S</b>	
		scan commands:	
		\scan_stop:	113, 122, 595, 2618, 2643, 2666,
			2680, 2812, 2829, 2837, 2844, 2857
		scan internal commands:	
		\s__color_stop	455, 458, 469, 472, 729, 730,
			734, 738, 751, 754, 758, 762, 776,
			935, 971, 975, 1100, 1102, 1123, 1125
		\s__graphics_stop	1821, 1851, 2147, 2152
		separation	3228
		seq commands:	
		\seq_set_from_clist:Nn	1756, 1780, 1917, 2105
		skip commands:	
		\skip_horizontal:n	217, 265, 322
		str commands:	
		\c_hash_str	388, 1620, 1627, 1667
		\c_percent_str	1143, 1144, 1145
		\str_case:nn	940, 2247, 2763
		\str_case:nnTF	2484, 2654, 3001
		\str_case_e:nn	2187, 2735
		\str_convert_pdfname:n	677, 697, 883
		\str_if_eq:nnTF	474, 477, 480, 483, 856, 3197
		\str_new:N	1862, 1863, 1864
		\str_tail:N	1876, 1901
		sys commands:	
		\sys_if_shell:TF	1860
		\sys_shell_now:n	1886
		<b>T</b>	
		T <sub>E</sub> X and L <sup>A</sup> T <sub>E</sub> X 2 <sub>ε</sub> commands:	
		\ccclv	2447, 2449, 2457
		\ifl@t@r	50, 52
		\makecol@hook	2441
		\current@color	14, 445, 449, 455, 469
		\special	2
		tex commands:	
		\tex_baselineskip:D	2418
		\tex_endinput:D	44
		\tex_global:D	2805, 2822, 2836, 2843, 2850
		\tex_immediate:D	1828, 1909, 2727, 2730, 2758, 2761
		\tex_luatexversion:D	2834, 2862

