

# 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}{2020-09-24}{ }
5   {L3 backend support: dvipdfmx}
6 </dvipdfmx>
7 <*dvips>
8   {l3backend-dvips.def}{2020-09-24}{ }
9   {L3 backend support: dvips}
10 </dvips>
11 <*dvisvgm>
12   {l3backend-dvisvgm.def}{2020-09-24}{ }
13   {L3 backend support: dvisvgm}
14 </dvisvgm>
15 <*luatex>
16   {l3backend-luatex.def}{2020-09-24}{ }
17   {L3 backend support: PDF output (LuaTeX)}
18 </luatex>
19 <*pdftex>
20   {l3backend-pdftex.def}{2020-09-24}{ }
21   {L3 backend support: PDF output (pdfTeX)}
22 </pdftex>
23 <*xetex>
24   {l3backend-xetex.def}{2020-09-24}{ }
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 {2020-09-01}
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`.)

## 1.1 dvips backend

```

50 <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.

```

51 \cs_new_protected:Npn \__kernel_backend_literal_postscript:n #1
52 { \__kernel_backend_literal:n { ps:: #1 } }
53 \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 `"`).

```

54 \cs_new_protected:Npn \__kernel_backend_postscript:n #1
55 { \__kernel_backend_literal:n { ps: SDict ~ begin ~ #1 ~ end } }
56 \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.

```

57 \bool_if:NT \g__kernel_backend_header_bool
58 {

```

```

59 \cs_if_exist:NTF \AtBeginDvi
60 { \AtBeginDvi }
61 { \use:n }
62 { \__kernel_backend_literal:n { header = l3backend-dvips.pro } }
63 }

```

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.

```

64 \cs_new_protected:Npn \__kernel_backend_align_begin:
65 {
66   \__kernel_backend_literal:n { ps::[begin] }
67   \__kernel_backend_literal_postscript:n { currentpoint }
68   \__kernel_backend_literal_postscript:n { currentpoint~translate }
69 }
70 \cs_new_protected:Npn \__kernel_backend_align_end:
71 {
72   \__kernel_backend_literal_postscript:n { neg-exch-neg-exch~translate }
73   \__kernel_backend_literal:n { ps::[end] }
74 }

```

(End definition for `\__kernel_backend_align_begin:` and `\__kernel_backend_align_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.

```

75 \cs_new_protected:Npn \__kernel_backend_scope_begin:
76 { \__kernel_backend_literal:n { ps:gsave } }
77 \cs_new_protected:Npn \__kernel_backend_scope_end:
78 { \__kernel_backend_literal:n { ps:grestore } }

```

(End definition for `\__kernel_backend_scope_begin:` and `\__kernel_backend_scope_end:.`)

```

79 </dvips>

```

## 1.2 LuaTeX and pdfTeX backends

```

80 <*luatex | pdftex>

```

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

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

```

81 \cs_new_protected:Npn \__kernel_backend_literal_pdf:n #1
82 {
83   <*luatex>
84   \tex_pdfextension:D literal
85   </luatex>
86   <*pdftex>

```

```

87     \tex_pdfliteral:D
88   </pdftex>
89     { \exp_not:n {#1} }
90   }
91   \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.

```

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

```

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

```

(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.

```

120 \cs_new_protected:Npn \__kernel_backend_matrix:n #1
121 {
122 <*luatex>
123   \tex_pdfextension:D setmatrix
124 </luatex>
125 <*pdftex>
126   \tex_pdfsetmatrix:D
127 </pdftex>

```

```

128         { \exp_not:n {#1} }
129     }
130 \cs_generate_variant:Nn \__kernel_backend_matrix:n { x }

(End definition for \__kernel_backend_matrix:n.)

131 </luatex | pdfTeX>

```

### 1.3 dvipdfmx backend

```

132 <*dvipdfmx | xetex>

```

The `dvipdfmx` shares code with the PDF mode one (using the common section to this file) but also with `XYTeX`. 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 `XYTeX` as required. Equivalent to `pdf:content` but favored as the link to the `pdfTeX` primitive approach is clearer.

```

\__kernel_backend_literal_pdf:n
\__kernel_backend_literal_pdf:x

```

```

133 \cs_new_protected:Npn \__kernel_backend_literal_pdf:n #1
134 { \__kernel_backend_literal:n { pdf:literal~ #1 } }
135 \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!

```

136 \cs_new_protected:Npn \__kernel_backend_literal_page:n #1
137 { \__kernel_backend_literal:n { pdf:literal~direct~ #1 } }

(End definition for \__kernel_backend_literal_page:n.)

```

```

\__kernel_backend_scope_begin:
\__kernel_backend_scope_end:

```

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

```

138 \cs_new_protected:Npn \__kernel_backend_scope_begin:
139 { \__kernel_backend_literal:n { x:gsave } }
140 \cs_new_protected:Npn \__kernel_backend_scope_end:
141 { \__kernel_backend_literal:n { x:grestore } }

(End definition for \__kernel_backend_scope_begin: and \__kernel_backend_scope_end:.)

```

```

142 </dvipdfmx | xetex>

```

### 1.4 dvisvgm backend

```

143 <*dvisvgm>

```

```

\__kernel_backend_literal_svg:n
\__kernel_backend_literal_svg:x

```

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

```

144 \cs_new_protected:Npn \__kernel_backend_literal_svg:n #1
145 { \__kernel_backend_literal:n { dvisvgm:raw~ #1 { ?nl } } }
146 \cs_generate_variant:Nn \__kernel_backend_literal_svg:n { x }

(End definition for \__kernel_backend_literal_svg:n.)

```

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

```
147 \int_new:N \g__kernel_backend_scope_int
148 \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:` 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.

```
\__kernel_backend_scope_begin:n
\__kernel_backend_scope_end:
\__kernel_backend_scope_begin:n
\__kernel_backend_scope_begin:x
\__kernel_backend_scope:n
\__kernel_backend_scope:x
149 \cs_new_protected:Npn \__kernel_backend_scope_begin:
150 {
151   \__kernel_backend_literal_svg:n { <g> }
152   \int_set_eq:NN
153     \l__kernel_backend_scope_int
154     \g__kernel_backend_scope_int
155   \group_begin:
156     \int_gset:Nn \g__kernel_backend_scope_int { 1 }
157 }
158 \cs_new_protected:Npn \__kernel_backend_scope_end:
159 {
160   \prg_replicate:nn
161     { \g__kernel_backend_scope_int }
162     { \__kernel_backend_literal_svg:n { </g> } }
163   \group_end:
164   \int_gset_eq:NN
165     \g__kernel_backend_scope_int
166     \l__kernel_backend_scope_int
167 }
168 \cs_new_protected:Npn \__kernel_backend_scope_begin:n #1
169 {
170   \__kernel_backend_literal_svg:n { <g ~ #1 > }
171   \int_set_eq:NN
172     \l__kernel_backend_scope_int
173     \g__kernel_backend_scope_int
174   \group_begin:
175     \int_gset:Nn \g__kernel_backend_scope_int { 1 }
176 }
177 \cs_generate_variant:Nn \__kernel_backend_scope_begin:n { x }
178 \cs_new_protected:Npn \__kernel_backend_scope:n #1
179 {
180   \__kernel_backend_literal_svg:n { <g ~ #1 > }
181   \int_gincr:N \g__kernel_backend_scope_int
182 }
183 \cs_generate_variant:Nn \__kernel_backend_scope:n { x }
```

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

```
184 </dvisvgm>
```

```
185 </package>
```

## 2 l3backend-box Implementation

```
186 <*package>
187 <@@=box>
```

### 2.1 dvips backend

```
188 <*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.

```
189 \cs_new_protected:Npn \_box_backend_clip:N #1
190 {
191   \__kernel_backend_scope_begin:
192   \__kernel_backend_align_begin:
193   \__kernel_backend_literal_postscript:n { matrix~currentmatrix }
194   \__kernel_backend_literal_postscript:n
195     { Resolution~72~div~VResolution~72~div~scale }
196   \__kernel_backend_literal_postscript:n { DVImag~dup~scale }
197   \__kernel_backend_literal_postscript:x
198     {
199       0 ~
200       \dim_to_decimal_in_bp:n { \box_dp:N #1 } ~
201       \dim_to_decimal_in_bp:n { \box_wd:N #1 } ~
202       \dim_to_decimal_in_bp:n { -\box_ht:N #1 - \box_dp:N #1 } ~
203       rectclip
204     }
205   \__kernel_backend_literal_postscript:n { setmatrix }
206   \__kernel_backend_align_end:
207   \hbox_overlap_right:n { \box_use:N #1 }
208   \__kernel_backend_scope_end:
209   \skip_horizontal:n { \box_wd:N #1 }
210 }
```

(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.

```
211 \cs_new_protected:Npn \_box_backend_rotate:Nn #1#2
212 { \exp_args:Nnf \_box_backend_rotate_aux:Nn #1 { \fp_eval:n {#2} } }
213 \cs_new_protected:Npn \_box_backend_rotate_aux:Nn #1#2
214 {
215   \__kernel_backend_scope_begin:
216   \__kernel_backend_align_begin:
217   \__kernel_backend_literal_postscript:x
218     {
219       \fp_compare:nNnTF {#2} = \c_zero_fp
220       { 0 }
221       { \fp_eval:n { round ( -(#2) , 5 ) } } } ~
222     rotate
223 }
```

```

224   \__kernel_backend_align_end:
225   \box_use:N #1
226   \__kernel_backend_scope_end:
227 }

```

(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.

```

228 \cs_new_protected:Npn \__box_backend_scale:Nnn #1#2#3
229 {
230   \__kernel_backend_scope_begin:
231   \__kernel_backend_align_begin:
232   \__kernel_backend_literal_postscript:x
233   {
234     \fp_eval:n { round ( #2 , 5 ) } ~
235     \fp_eval:n { round ( #3 , 5 ) } ~
236     scale
237   }
238   \__kernel_backend_align_end:
239   \hbox_overlap_right:n { \box_use:N #1 }
240   \__kernel_backend_scope_end:
241 }

```

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

```

242 </dvips>

```

## 2.2 LuaTeX and pdfTeX backends

```

243 < *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.

```

244 \cs_new_protected:Npn \__box_backend_clip:N #1
245 {
246   \__kernel_backend_scope_begin:
247   \__kernel_backend_literal_pdf:x
248   {
249     0~
250     \dim_to_decimal_in_bp:n { -\box_dp:N #1 } ~
251     \dim_to_decimal_in_bp:n { \box_wd:N #1 } ~
252     \dim_to_decimal_in_bp:n { \box_ht:N #1 + \box_dp:N #1 } ~
253     re~W~n
254   }
255   \hbox_overlap_right:n { \box_use:N #1 }
256   \__kernel_backend_scope_end:
257   \skip_horizontal:n { \box_wd:N #1 }
258 }

```

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



`\__box_backend_rotate:Nn`  
`\__box_backend_rotate_aux:Nn`  
`\l__box_backend_cos_fp`  
`\l__box_backend_sin_fp`

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.

```

259 \cs_new_protected:Npn \__box_backend_rotate:Nn #1#2
260 { \exp_args:Nnf \__box_backend_rotate_aux:Nn #1 { \fp_eval:n {#2} } }
261 \cs_new_protected:Npn \__box_backend_rotate_aux:Nn #1#2
262 {
263   \__kernel_backend_scope_begin:
264   \box_set_wd:Nn #1 { Opt }
265   \fp_set:Nn \l__box_backend_cos_fp { round ( cosd ( #2 ) , 5 ) }
266   \fp_compare:nNnTF \l__box_backend_cos_fp = \c_zero_fp
267     { \fp_zero:N \l__box_backend_cos_fp }
268   \fp_set:Nn \l__box_backend_sin_fp { round ( sind ( #2 ) , 5 ) }
269   \__kernel_backend_matrix:x
270   {
271     \fp_use:N \l__box_backend_cos_fp \c_space_tl
272     \fp_compare:nNnTF \l__box_backend_sin_fp = \c_zero_fp
273       { 0~0 }
274       {
275         \fp_use:N \l__box_backend_sin_fp
276         \c_space_tl
277         \fp_eval:n { -\l__box_backend_sin_fp }
278       }
279     \c_space_tl
280     \fp_use:N \l__box_backend_cos_fp
281   }
282   \box_use:N #1
283   \__kernel_backend_scope_end:
284 }
285 \fp_new:N \l__box_backend_cos_fp
286 \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.

```

287 \cs_new_protected:Npn \__box_backend_scale:Nnn #1#2#3
288 {
289   \__kernel_backend_scope_begin:
290   \__kernel_backend_matrix:x
291   {
292     \fp_eval:n { round ( #2 , 5 ) } ~
293     0~0~
294     \fp_eval:n { round ( #3 , 5 ) }
295   }
296   \hbox_overlap_right:n { \box_use:N #1 }
297   \__kernel_backend_scope_end:
298 }

```

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

299 `</luatex | pdftex>`

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

300  $\langle$ \*dvipdfmx | xetex $\rangle$

$\_\_\text{box\_backend\_clip:N}$  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.

```

301 \cs_new_protected:Npn \_\_\text{box\_backend\_clip:N} #1
302 {
303   \_\_\text{kernel\_backend\_scope\_begin}:
304   \_\_\text{kernel\_backend\_literal\_pdf:x}
305   {
306     0~
307     \dim_to_decimal_in_bp:n { -\box_dp:N #1 } ~
308     \dim_to_decimal_in_bp:n { \box_wd:N #1 } ~
309     \dim_to_decimal_in_bp:n { \box_ht:N #1 + \box_dp:N #1 } ~
310     re~W~n
311   }
312   \hbox_overlap_right:n { \box_use:N #1 }
313   \_\_\text{kernel\_backend\_scope\_end}:
314   \skip_horizontal:n { \box_wd:N #1 }
315 }

```

(End definition for  $\_\_\text{box\_backend\_clip:N}$ .)

$\_\_\text{box\_backend\_rotate:Nn}$   
 $\_\_\text{box\_backend\_rotate\_aux:Nn}$

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

```

316 \cs_new_protected:Npn \_\_\text{box\_backend\_rotate:Nn} #1#2
317 { \exp_args:Nnf \_\_\text{box\_backend\_rotate\_aux:Nn} #1 { \fp_eval:n {#2} } }
318 \cs_new_protected:Npn \_\_\text{box\_backend\_rotate\_aux:Nn} #1#2
319 {
320   \_\_\text{kernel\_backend\_scope\_begin}:
321   \_\_\text{kernel\_backend\_literal:x}
322   {
323     x:rotate~
324     \fp_compare:nNnTF {#2} = \c_zero_fp
325     { 0 }
326     { \fp_eval:n { round ( #2 , 5 ) } }
327   }
328   \box_use:N #1
329   \_\_\text{kernel\_backend\_scope\_end}:
330 }

```

(End definition for  $\_\_\text{box\_backend\_rotate:Nn}$  and  $\_\_\text{box\_backend\_rotate\_aux:Nn}$ .)

$\_\_\text{box\_backend\_scale:Nnn}$

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

```

331 \cs_new_protected:Npn \_\_\text{box\_backend\_scale:Nnn} #1#2#3
332 {
333   \_\_\text{kernel\_backend\_scope\_begin}:
334   \_\_\text{kernel\_backend\_literal:x}

```

```

335     {
336       x:scale~
337       \fp_eval:n { round ( #2 , 5 ) } ~
338       \fp_eval:n { round ( #3 , 5 ) }
339     }
340     \hbox_overlap_right:n { \box_use:N #1 }
341     \__kernel_backend_scope_end:
342   }

```

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

```

343 \</dvipdfmx | xetex>

```

## 2.4 dvisvgm backend

```

344 \*dvisvgm>

```

```

\__box_backend_clip:N
\g__box_clip_path_int

```

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

```

345 \cs_new_protected:Npn \__box_backend_clip:N #1
346 {
347   \int_gincr:N \g__box_clip_path_int
348   \__kernel_backend_literal_svg:x
349   { < clipPath-id = " l3cp \int_use:N \g__box_clip_path_int " > }
350   \__kernel_backend_literal_svg:x
351   {
352     <
353     path ~ d =
354     "
355       M ~ 0 ~
356       \dim_to_decimal:n { -\box_dp:N #1 } ~
357       L ~ \dim_to_decimal:n { \box_wd:N #1 } ~
358       \dim_to_decimal:n { -\box_dp:N #1 } ~
359       L ~ \dim_to_decimal:n { \box_wd:N #1 } ~
360       \dim_to_decimal:n { \box_ht:N #1 + \box_dp:N #1 } ~
361       L ~ 0 ~
362       \dim_to_decimal:n { \box_ht:N #1 + \box_dp:N #1 } ~
363       Z
364     "
365     />
366   }
367   \__kernel_backend_literal_svg:n
368   { < /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 T<sub>E</sub>X box is inserted to get things back on track. The clip path needs to come between those two such that if lines up with the current point, as does the T<sub>E</sub>X box.

```

369 \__kernel_backend_scope_begin:n

```

```

370     {
371         transform =
372         "
373             translate ( { ?x } , { ?y } ) ~
374             scale ( 1 , -1 )
375         "
376     }
377     \__kernel_backend_scope:x
378     {
379         clip-path =
380         "url ( \c_hash_str l3cp \int_use:N \g__box_clip_path_int ) "
381     }
382     \__kernel_backend_scope:n
383     {
384         transform =
385         "
386             scale ( -1 , 1 ) ~
387             translate ( { ?x } , { ?y } ) ~
388             scale ( -1 , -1 )
389         "
390     }
391     \box_use:N #1
392     \__kernel_backend_scope_end:
393 }
394 \int_new:N \g__box_clip_path_int

```

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

\\_\_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.

```

395 \cs_new_protected:Npn \__box_backend_rotate:Nn #1#2
396 {
397     \__kernel_backend_scope_begin:x
398     {
399         transform =
400         "
401             rotate
402             ( \fp_eval:n { round ( -(#2) , 5 ) } , ~ { ?x } , ~ { ?y } )
403         "
404     }
405     \box_use:N #1
406     \__kernel_backend_scope_end:
407 }

```

(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).

```

408 \cs_new_protected:Npn \__box_backend_scale:Nnn #1#2#3
409 {
410     \__kernel_backend_scope_begin:x
411     {

```

```

412     transform =
413     "
414         translate ( { ?x } , { ?y } ) ~
415         scale
416         (
417             \fp_eval:n { round ( -#2 , 5 ) } ,
418             \fp_eval:n { round ( -#3 , 5 ) }
419         ) ~
420         translate ( { ?x } , { ?y } ) ~
421         scale ( -1 )
422     "
423 }
424 \hbox_overlap_right:n { \box_use:N #1 }
425 \__kernel_backend_scope_end:
426 }

```

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

```

427 </dvisvgm>
428 </package>

```

### 3 l3backend-color Implementation

```

429 <*package>
430 <@@=color>

```

Color support is split into parts: 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>TeX 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>TeX is PDF-based means it (largely) sticks closer to direct PDF output.

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

```

431 <*dvipdfmx | xetex>

```

These backends have the most possible approaches: it recognises both dvips-based color specials and it's own format, plus one can include PDF statements directly. The latter are not subject to the stack, so are not suitable for general use. Of the two stack methods, the dedicated one has been extended to cover color spaces, so it is used in preference to the dvips one.

The L<sup>A</sup>T<sub>E</sub>X<sub>2<sub>ε</sub></sub> backend code uses dvips-based code with dvipdfmx/X<sub>Y</sub>TeX, and so we leave getting color from L<sup>A</sup>T<sub>E</sub>X<sub>2<sub>ε</sub></sub> to a shared code path below.

Push the data to the stack.

```

\__color_backend_select_cmyk:n
\__color_backend_select_gray:n
\__color_backend_select_rgb:n
\__color_backend_reset:
color.sc
color.fc
432 \cs_new_protected:Npn \__color_backend_select_cmyk:n #1
433 {
434     \__kernel_backend_literal:n { pdf: bc ~ [#1] }
435     \group_insert_after:N \__color_backend_reset:
436 }
437 \cs_new_eq:NN \__color_backend_select_gray:n \__color_backend_select_cmyk:n
438 \cs_new_eq:NN \__color_backend_select_rgb:n \__color_backend_select_cmyk:n
439 \cs_new_protected:Npn \__color_backend_reset:
440 { \__kernel_backend_literal:n { pdf: ec } }

```

(End definition for `\_color_backend_select_cmyk:n` and others. These functions are documented on page ??.)

```
441 </dvipdfmx | xetex>
```

## 3.2 dvips-style

```
442 <*dvisvgm | dvipdfmx | dvips | xetex>
```

`\_color_backend_pickup:N` Allow for L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> color. Here, the possible input values are limited: dvips-style colors can 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.

```
443 \cs_new_protected:Npn \_color_backend_pickup:N #1 { }
444 \cs_if_exist:cT { ver@color.sty }
445 {
446   \cs_set_protected:Npn \_color_backend_pickup:N #1
447   {
448     \exp_args:NV \tl_if_head_is_space:nTF \current@color
449     {
450       \tl_set:Nx #1
451       {
452         { \exp_after:wN \use:n \current@color }
453         { 1 }
454       }
455     }
456     {
457       \exp_last_unbraced:Nx \_color_backend_pickup:w
458       { \current@color } \s_color_stop #1
459     }
460   }
461   \cs_new_protected:Npn \_color_backend_pickup:w #1 ~ #2 \s_color_stop #3
462   { \tl_set:Nn #3 { {#1} {#2} } }
463 }
```

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

```
464 </dvisvgm | dvipdfmx | dvips | xetex>
```

```
465 <*dvisvgm | dvips>
```

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

```
\_color_backend_select_gray:n
\_color_backend_select_rgb:n
\_color_backend_select:n
\_color_backend_reset:
color.sc
color.fc
466 \cs_new_protected:Npn \_color_backend_select_cmyk:n #1
467 { \_color_backend_select:n { cmyk ~ #1 } }
468 \cs_new_protected:Npn \_color_backend_select_gray:n #1
469 { \_color_backend_select:n { gray ~ #1 } }
470 \cs_new_protected:Npn \_color_backend_select_rgb:n #1
471 { \_color_backend_select:n { rgb ~ #1 } }
472 \cs_new_protected:Npn \_color_backend_select:n #1
473 {
474   \_kernel_backend_literal:n { color~push~ #1 }
475 <*dvips>
476   \_kernel_backend_postscript:n { /color.sc~ { ~ } ~def }
477   \_kernel_backend_postscript:n { /color.fc~ { ~ } ~def }
478 </dvips>
479   \group_insert_after:N \_color_backend_reset:
```

```

480 }
481 \cs_new_protected:Npn \__color_backend_reset:
482 { \__kernel_backend_literal:n { color-pop } }

```

(End definition for `\__color_backend_select_cmyk:n` and others. These functions are documented on page ??.)

```

483 </dvisvgm | dvips>

```

### 3.3 LuaTeX and pdfTeX

```

484 <*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

```

485 \cs_new_protected:Npn \__color_backend_pickup:N #1 { }
486 \cs_if_exist:cT { ver@color.sty }
487 {
488   \cs_set_protected:Npn \__color_backend_pickup:N #1
489   {
490     \exp_last_unbraced:Nx \__color_backend_pickup:w
491     { \current@color } ~ 0 ~ 0 ~ 0 \s__color_stop #1
492   }
493   \cs_new_protected:Npn \__color_backend_pickup:w
494   #1 ~ #2 ~ #3 ~ #4 ~ #5 ~ #6 \s__color_stop #7
495   {
496     \str_if_eq:nnTF {#2} { g }
497     { \tl_set:Nn #7 { { gray } {#1} } }
498     {
499       \str_if_eq:nnTF {#4} { rg }
500       { \tl_set:Nn #7 { { rgb } { #1 ~ #2 ~ #3 } } }
501       {
502         \str_if_eq:nnTF {#5} { k }
503         { \tl_set:Nn #7 { { cmyk } { #1 ~ #2 ~ #3 ~ #4 } } }
504         {
505           \str_if_eq:nnTF {#2} { cs }
506           {
507             \tl_set:Nx #7 { { \use:n #1 } { #5 } }
508           }
509           {
510             \tl_set:Nn #7 { { gray } { 0 } }
511           }
512         }
513       }
514     }
515   }
516 }

```

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

`\l__kernel_color_stack_int` pdfTeX and LuaTeX have multiple stacks available, and to track which one is in use a variable is required.

```

517 \int_new:N \l__kernel_color_stack_int

```

(End definition for \l\_\_kernel\_color\_stack\_int.)

```
\_color_backend_select_cmyk:n
\_color_backend_select_gray:n
\_color_backend_select_rgb:n
\_color_backend_reset:
```

Simply dump the data, but allowing for LuaTeX.

```
518 \cs_new_protected:Npn \_color_backend_select_cmyk:n #1
519 { \_color_backend_select:n { #1 ~ k ~ #1 ~ K } }
520 \cs_new_protected:Npn \_color_backend_select_gray:n #1
521 { \_color_backend_select:n { #1 ~ g ~ #1 ~ G } }
522 \cs_new_protected:Npn \_color_backend_select_rgb:n #1
523 { \_color_backend_select:n { #1 ~ rg ~ #1 ~ RG } }
524 \cs_new_protected:Npn \_color_backend_select:n #1
525 {
526 \*luatex
527 \tex_pdfextension:D colorstack
528 \</luatex>
529 \*pdfTeX
530 \tex_pdfcolorstack:D
531 \</pdfTeX>
532 \l__kernel_color_stack_int push {#1}
533 \group_insert_after:N \_color_backend_reset:
534 }
535 \cs_new_protected:Npn \_color_backend_reset:
536 {
537 \*luatex
538 \tex_pdfextension:D colorstack
539 \</luatex>
540 \*pdfTeX
541 \tex_pdfcolorstack:D
542 \</pdfTeX>
543 \l__kernel_color_stack_int pop \scan_stop:
544 }
```

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

```
545 \</luatex | pdfTeX>
```

### 3.4 Separations

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

```
546 \*dvips
```

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

```
547 \cs_new_protected:Npn \_color_backend_select_separation:nn #1#2
548 { \_color_backend_select:n { separation ~ #1 ~ #2 } }
549 \cs_new_eq:NN \_color_backend_select_devicen:nn \_color_backend_select_separation:nn
```

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

```
\_color_backend_separation_init:nnnnn
\_color_backend_separation_init:nxxnn
\_color_backend_separation_init_aux:nnnnn
lor_backend_separation_init/DeviceCMYK:nnn
lor_backend_separation_init/DeviceGray:nnn
lor_backend_separation_init/DeviceRGB:nnn
\_color_backend_separation_init_Device:Nn
\_color_backend_separation_init:nnn
\_color_backend_separation_init_count:n
\_color_backend_separation_init_count:w
\_color_backend_separation_init:nnnn
\_color_backend_separation_init:w
\_color_backend_separation_init:n
\_color_backend_separation_init:nw
\_color_backend_separation_init_CIELAB:nnn
```

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

```
550 \cs_new_protected:Npx \_color_backend_separation_init:nnnnn #1#2#3#4#5
551 {
552 \bool_if:NT \g__kernel_backend_header_bool
```



```

553 {
554   \cs_if_exist:NTF \AtBeginDvi
555   { \exp_not:N \AtBeginDvi }
556   { \use:n }
557   {
558     \exp_not:N \__color_backend_separation_init_aux:nnnnn
559     {#1} {#2} {#3} {#4} {#5}
560   }
561 }
562 }
563 \cs_generate_variant:Nn \__color_backend_separation_init:nnnnn { nxx }
564 \cs_new_protected:Npn \__color_backend_separation_init_aux:nnnnn #1#2#3#4#5
565 {
566   \__kernel_backend_literal:e
567   {
568     !
569     TeXDict ~ begin ~
570     /color \int_use:N \g__color_model_int
571     {
572       [ ~
573       /Separation ~ ( \str_convert_pdfname:n {#1} ) ~
574       [ ~ #2 ~ ] ~
575       {
576         \cs_if_exist_use:cF { __color_backend_separation_init_ #2 :nnn }
577         { \__color_backend_separation_init:nnn }
578         {#3} {#4} {#5}
579       }
580       ] ~ setcolorspace
581     } ~ def ~
582   end
583 }
584 }
585 \cs_new:cpn { __color_backend_separation_init_ /DeviceCMYK :nnn } #1#2#3
586 { \__color_backend_separation_init_Device:Nn 4 {#3} }
587 \cs_new:cpn { __color_backend_separation_init_ /DeviceGray :nnn } #1#2#3
588 { \__color_backend_separation_init_Device:Nn 1 {#3} }
589 \cs_new:cpn { __color_backend_separation_init_ /DeviceRGB :nnn } #1#2#3
590 { \__color_backend_separation_init_Device:Nn 2 {#3} }
591 \cs_new:Npn \__color_backend_separation_init_Device:Nn #1#2
592 {
593   #2 ~
594   \prg_replicate:nn {#1}
595   { #1 ~ index ~ mul ~ #1 ~ 1 ~ roll ~ }
596   \int_eval:n { #1 + 1 } ~ -1 ~ roll ~ pop
597 }

```

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.

```

598 \cs_new:Npn \__color_backend_separation_init:nnn #1#2#3
599 {
600   \exp_args:Ne \__color_backend_separation_init:nnnn
601   { \__color_backend_separation_init_count:n {#2} }

```

```

602     {#1} {#2} {#3}
603   }
604   \cs_new:Npn \__color_backend_separation_init_count:n #1
605     { \int_eval:n { 0 \__color_backend_separation_init_count:w #1 ~ \s__color_stop } }
606   \cs_new:Npn \__color_backend_separation_init_count:w #1 ~ #2 \s__color_stop
607     {
608       +1
609       \tl_if_blank:nF {#2}
610       { \__color_backend_separation_init_count:w #2 \s__color_stop }
611     }

```

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

```

612   \cs_new:Npn \__color_backend_separation_init:nnnn #1#2#3#4
613     {
614       \__color_backend_separation_init:w #3 ~ \s__color_stop #4 ~ \s__color_stop
615       \prg_replicate:nn {#1}
616       {
617         pop ~ 1 ~ index ~ neg ~ 1 ~ index ~ add ~
618         \int_eval:n { 3 * #1 } ~ index ~ mul ~
619         2 ~ index ~ add ~
620         \int_eval:n { 3 * #1 } ~ #1 ~ roll ~
621       }
622       \int_step_function:nnnN {#1} { -1 } { 1 }
623       \__color_backend_separation_init:n
624       \int_eval:n { 4 * #1 + 1 } ~ #1 ~ roll ~
625       \prg_replicate:nn { 3 * #1 + 1 } { pop ~ }
626       \tl_if_blank:nF {#2}
627       { \__color_backend_separation_init:nw {#1} #2 ~ \s__color_stop }
628     }
629   \cs_new:Npn \__color_backend_separation_init:w
630     #1 ~ #2 \s__color_stop #3 ~ #4 \s__color_stop
631     {
632       #1 ~ #3 ~ 0 ~
633       \tl_if_blank:nF {#2}
634       { \__color_backend_separation_init:w #2 \s__color_stop #4 \s__color_stop }
635     }
636   \cs_new:Npn \__color_backend_separation_init:n #1
637     { \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.

```

638   \cs_new:Npn \__color_backend_separation_init:nw #1#2 ~ #3 ~ #4 \s__color_stop
639     {

```

```

640     #2 ~ #3 ~
641     2 ~ index ~ 2 ~ index ~ lt ~
642     { ~ pop ~ exch ~ pop ~ } ~
643     { ~
644         2 ~ index ~ 1 ~ index ~ gt ~
645         { ~ exch ~ pop ~ exch ~ pop ~ } ~
646         { ~ pop ~ pop ~ } ~
647     } ~
648     }
649     ifelse ~
650     #1 ~ 1 ~ roll ~
651     \tl_if_blank:nF {#4}
652     { \__color_backend_separation_init:nw {#1} #4 \s__color_stop }
653 }

```

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.

```

654 \cs_new_protected:Npn \__color_backend_separation_init_CIELAB:nnn #1#2#3
655 {
656     \__color_backend_separation_init:nxxxnn
657     {#2}
658     {
659         /CIEBasedABC ~
660         << ~
661         /RangeABC ~ [ ~ \c__color_model_range_CIELAB_tl \c_space_tl ] ~
662         /DecodeABC ~
663         [ ~
664             { ~ 16 ~ add ~ 116 ~ div ~ } ~ bind ~
665             { ~ 500 ~ div ~ } ~ bind ~
666             { ~ 200 ~ div ~ } ~ bind ~
667         ] ~
668         /MatrixABC ~ [ ~ 1 ~ 1 ~ 1 ~ 1 ~ 0 ~ 0 ~ 0 ~ 0 ~ -1 ~ ] ~
669         /DecodeLMN ~
670         [ ~
671             { ~
672                 dup ~ 6 ~ 29 ~ div ~ ge ~
673                 { ~ dup ~ dup ~ mul ~ mul ~ ~ } ~
674                 { ~ 4 ~ 29 ~ div ~ sub ~ 108 ~ 841 ~ div ~ mul ~ } ~
675             } ~
676             ifelse ~
677             0.9505 ~ mul ~
678             } ~ bind ~
679             { ~
680                 dup ~ 6 ~ 29 ~ div ~ ge ~
681                 { ~ dup ~ dup ~ mul ~ mul ~ ~ } ~
682                 { ~ 4 ~ 29 ~ div ~ sub ~ 108 ~ 841 ~ div ~ mul ~ } ~
683             } ~
684             ifelse ~
685             1.0890 ~ mul ~
686             } ~ bind ~
687         ] ~
688         ifelse ~
689         1.0890 ~ mul ~
690     } ~ bind

```

```

691         ] ~
692         /WhitePoint ~
693         [ ~ \tl_use:c { c__color_model_whitepoint_CIELAB_ #1 _tl } ~ ] ~
694     >>
695 }
696 { \c__color_model_range_CIELAB_tl }
697 { 100 ~ 0 ~ 0 }
698 {#3}
699 }

```

(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.

```

700 \cs_new_protected:Npn \_color_backend_devicen_init:nnn #1#2#3
701 {
702     \_kernel_backend_literal:e
703     {
704         !
705         TeXDict ~ begin ~
706         /color \int_use:N \g__color_model_int
707         {
708             [ ~
709             /DeviceN ~
710             [ ~ #1 ~ ] ~
711             #2 ~
712             { ~ #3 ~ } ~
713             ] ~ setcolorspace
714             } ~ def ~
715         end
716     }
717 }

```

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

```

718 </dvips>
719 <*dvisvgm>

```

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

```

\color_backend_select_devicen:nn
720 \cs_new_protected:Npn \_color_backend_select_separation:nn #1#2 { }
721 \cs_new_protected:Npn \_color_backend_select_devicen:nn #1#2 { }

```

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

\\_color\_backend\_separation\_init:nnnnn No support at present.

```

\_color_backend_separation_init_CIELAB:nnn
722 \cs_new_protected:Npn \_color_backend_separation_init:nnnnn #1#2#3#4#5 { }
723 \cs_new_protected:Npn \_color_backend_separation_init_CIELAB:nnnnnn #1#2#3 { }

```

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

```

724 </dvisvgm>
725 <*dviPDFmx | luatex | pdftex | xetex>

```

`\_color_backend_select_separation:nn`  
`\_color_backend_select_devicen:nn`  
`\_color_backend_select:n`

Different syntaxes here as the stacks are accessed very differently.

```

726 \cs_new_protected:Npn \_color_backend_select_separation:nn #1#2
727 < *dvipdfmx | xetex >
728 { \_color_backend_select:n { @#1 ~ [#2] } }
729 < /dvipdfmx | xetex >
730 < *luatex | pdftex >
731 { \_color_backend_select:n { /#1 ~ cs ~ /#1 ~ CS ~ #2 ~ scn ~ #2 ~ SCN } }
732 < /luatex | pdftex >
733 < *dvipdfmx | xetex >
734 \cs_new_protected:Npn \_color_backend_select:n #1
735 {
736   \_kernel_backend_literal:n { pdf: bc ~ #1 }
737   \group_insert_after:N \_color_backend_reset:
738 }
739 < /dvipdfmx | xetex >
740 \cs_new_eq:NN \_color_backend_select_devicen:nn \_color_backend_select_separation:nn

```

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

`\_color_backend_separation_init:nnnnn`  
`\_color_backend_separation_init:n`  
`\_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. The latter uses the internal name of the `cs`. For `dvipdfmx/XYTEX`, the backend does most of the work so we need a simplified version. We use a separate object for the tint transformation following the model in the PDF reference.

```

741 \cs_new_protected:Npn \_color_backend_separation_init:nnnnn #1#2#3#4#5
742 {
743   \pdf_object_now:nx { dict }
744   {
745     /FunctionType ~ 2
746     /Domain ~ [0 ~ 1]
747     \tl_if_blank:nF {#3} { /Range ~ [#3] }
748     /C0 ~ [#4] ~
749     /C1 ~ [#5] /N ~ 1
750   }
751   \_color_backend_separation_init:n
752   {
753     /Separation ~
754     / \str_convert_pdfname:n {#1} ~ #2 ~
755     \pdf_object_last:
756   }
757 < *luatex | pdftex >
758   \use:x
759   {
760     \pdfcoredict_gput:nnn
761     { Page / Resources / ColorSpace }
762     { color \int_use:N \g__color_model_int }
763     { \pdf_object_last: }
764   }
765 < /luatex | pdftex >
766 }
767 \cs_if_exist:NF \pdf_object_now:nn
768 { \cs_gset_protected:Npn \_color_backend_separation_init:nnnnn #1#2#3#4#5 { } }
769 \cs_new_protected:Npn \_color_backend_separation_init:n #1

```

```

770 {
771 <*dviptfm | xetex>
772   \_kernel_backend_literal:x
773   {
774     pdf:obj ~ @color \int_use:N \g\_color_model_int \c_space_tl
775     [#1]
776   }
777 </dviptfm | xetex>
778 <*luatex | pdftex>
779   \pdf_object_now:nx { array } {#1}
780 </luatex | pdftex>
781 }

```

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

```

782 \cs_new_protected:Npn \_color_backend_separation_init_CIELAB:nnn #1#2#3
783 {
784   \pdf_object_if_exist:nF { \_color_illuminant_CIELAB_ #1 }
785   {
786     \pdf_object_new:nn { \_color_illuminant_CIELAB_ #1 } { array }
787     \pdf_object_write:nx { \_color_illuminant_CIELAB_ #1 }
788     {
789       /Lab ~
790       <<
791       /WhitePoint ~
792       [ \tl_use:c { c\_color_model_whitepoint_CIELAB_ #1 _tl } ]
793       /Range ~ [ \c\_color_model_range_CIELAB_tl ]
794       >>
795     }
796   }
797   \_color_backend_separation_init:nnnnn
798   {#2}
799   { \pdf_object_ref:n { \_color_illuminant_CIELAB_ #1 } }
800   { \c\_color_model_range_CIELAB_tl }
801   { 100 ~ 0 ~ 0 }
802   {#3}
803 }
804 \cs_if_exist:NF \pdf_object_now:nn
805 {
806   \cs_gset_protected:Npn \_color_backend_separation_init_CIELAB:nnn #1#2#3
807   { }
808 }

```

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

```

\_color_backend_devicen_init:nnm
\_color_backend_devicen_init:w
\_color_backend_devicen_init:n

```

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

```

809 \cs_new_protected:Npn \_color_backend_devicen_init:nnn #1#2#3
810 {
811   \pdf_object_now:nx { stream }
812   {
813     {
814       /FunctionType ~ 4 ~
815       /Domain ~

```

```

816         [ ~
817         \prg_replicate:nn
818         { 0 \__color_backend_devicen_init:w #1 ~ \s__color_stop }
819         { 0 ~ 1 ~ } ~
820     ] ~
821 /Range ~
822 [ ~
823     \str_case:nn {#2}
824     {
825         { /DeviceCMYK } { 0 ~ 1 ~ 0 ~ 1 ~ 0 ~ 1 ~ 0 ~ 1 }
826         { /DeviceGray } { 0 ~ 1 }
827         { /DeviceRGB } { 0 ~ 1 ~ 0 ~ 1 ~ 0 ~ 1 }
828     } ~
829 ]
830 }
831 {#3}
832 }
833 \__color_backend_separation_init:n
834 {
835     /DeviceN ~
836     [ ~ #1 ~ ] ~
837     #2 ~
838     \pdf_object_last:
839 }
840 < *luatex | pdftex >
841 \use:x
842 {
843     \pdfcoredict_gput:nnn
844     { Page / Resources / ColorSpace }
845     { color \int_use:N \g__color_model_int }
846     { \pdf_object_last: }
847 }
848 < /luatex | pdftex >
849 }
850 \cs_if_exist:NF \pdf_object_now:nn
851 { \cs_gset_protected:Npn \__color_backend_devicen_init:nnn #1#2#3 { } }
852 \cs_new:Npn \__color_backend_devicen_init:w #1 ~ #2 \s__color_stop
853 {
854     + 1
855     \tl_if_blank:nF {#2}
856     { \__color_backend_devicen_init:w #2 \s__color_stop }
857 }
858 \cs_new_eq:NN \__color_backend_devicen_init:n \__color_backend_separation_init:n

```

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

```

859 < /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.

```

860 < *dvipdfmx | luatex | pdftex | xetex >

```

`\_color_backend_fill_cmyk:n` Drawing (fill/stroke) color is handled in `dvipdfmx/XgTeX` in the same way as `LuaTeX/pdfTeX`.  
`\_color_backend_fill_gray:n` We use the same approach as earlier, except the color stack is not involved so the generic  
`\_color_backend_fill_rgb:n` direct PDF operation is used. There is no worry about the nature of strokes: everything  
`\_color_backend_stroke_cmyk:n` is handled automatically.  
`\_color_backend_stroke_gray:n`  
`\_color_backend_stroke_rgb:n`

```

861 \cs_new_protected:Npn \_color_backend_fill_cmyk:n #1
862 { \_kernel_backend_literal_pdf:n { #1 ~ k } }
863 \cs_new_protected:Npn \_color_backend_fill_gray:n #1
864 { \_kernel_backend_literal_pdf:n { #1 ~ g } }
865 \cs_new_protected:Npn \_color_backend_fill_rgb:n #1
866 { \_kernel_backend_literal_pdf:n { #1 ~ rg } }
867 \cs_new_protected:Npn \_color_backend_stroke_cmyk:n #1
868 { \_kernel_backend_literal_pdf:n { #1 ~ K } }
869 \cs_new_protected:Npn \_color_backend_stroke_gray:n #1
870 { \_kernel_backend_literal_pdf:n { #1 ~ G } }
871 \cs_new_protected:Npn \_color_backend_stroke_rgb:n #1
872 { \_kernel_backend_literal_pdf:n { #1 ~ RG } }

```

(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
873 \cs_new_protected:Npn \_color_backend_fill_separation:nn #1#2
874 { \_kernel_backend_literal_pdf:n { /#1 ~ cs ~ #2 ~ scn } }
875 \cs_new_protected:Npn \_color_backend_stroke_separation:nn #1#2
876 { \_kernel_backend_literal_pdf:n { /#1 ~ CS ~ #2 ~ SCN } }
877 \cs_new_eq:NN \_color_backend_fill_devicen:nn \_color_backend_fill_separation:nn
878 \cs_new_eq:NN \_color_backend_stroke_devicen:nn \_color_backend_stroke_separation:nn

```

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

```

879 </dvipdfmx | luatex | pdftex | xetex>
880 <*dvips>

```

`\_color_backend_fill_cmyk:n` All questions of saving the non-stacked data.

```

\_color_backend_fill_gray:n
\_color_backend_fill_rgb:n
\_color_backend_stroke_cmyk:n
\_color_backend_stroke_gray:n
\_color_backend_stroke_rgb:n
881 \cs_new_protected:Npn \_color_backend_fill_cmyk:n #1
882 { \_kernel_backend_postscript:n { /color.fc { #1 ~ setcmykcolor } def } }
883 \cs_new_protected:Npn \_color_backend_fill_gray:n #1
884 { \_kernel_backend_postscript:n { /color.fc { #1 ~ setgray } def } }
885 \cs_new_protected:Npn \_color_backend_fill_rgb:n #1
886 { \_kernel_backend_postscript:n { /color.fc { #1 ~ setrgbcolor } def } }
887 \cs_new_protected:Npn \_color_backend_stroke_cmyk:n #1
888 { \_kernel_backend_postscript:n { /color.sc { #1 ~ setcmykcolor } def } }
889 \cs_new_protected:Npn \_color_backend_stroke_gray:n #1
890 { \_kernel_backend_postscript:n { /color.sc { #1 ~ setgray } def } }
891 \cs_new_protected:Npn \_color_backend_stroke_rgb:n #1
892 { \_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
893 \cs_new_protected:Npn \_color_backend_fill_separation:nn #1#2
894 { \_kernel_backend_postscript:n { /color.fc { #1 } def } }
895 \cs_new_protected:Npn \_color_backend_stroke_separation:nn #1#2
896 { \_kernel_backend_postscript:n { /color.sc { #1 } def } }
897 \cs_new_eq:NN \_color_backend_fill_devicen:nn \_color_backend_fill_separation:nn
898 \cs_new_eq:NN \_color_backend_stroke_devicen:nn \_color_backend_stroke_separation:nn

```



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

899 `</dvips>`

900 `<*dvisvgm>`

`\_color_backend_fill_cmyk:n` For drawings in SVG, we use scopes for all colors. That requires using RGB values, which  
`\_color_backend_stroke_cmyk:n` luckily are easy to convert here (cmyk to RGB is a fixed function).  
`\_color_backend_cmyk:nw`

```

901 \cs_new_protected:Npn \_color_backend_fill_cmyk:n #1
902 { \_color_backend_cmyk:nw { fill } #1 \s_color_stop }
903 \cs_new_protected:Npn \_color_backend_stroke_cmyk:n #1
904 { \_color_backend_cmyk:nw { stroke } #1 \s_color_stop }
905 \cs_new_protected:Npn \_color_backend_cmyk:nw
906 #1#2 ~ #3 ~ #4 ~ #5 \s_color_stop
907 {
908   \use:x
909   {
910     \_color_backend:nnnn
911     {#1}
912     { \fp_eval:n { -100 * ( 1 - min ( 1 , #2 + #5 ) ) } }
913     { \fp_eval:n { -100 * ( 1 - min ( 1 , #3 + #5 ) ) } }
914     { \fp_eval:n { -100 * ( 1 - min ( 1 , #4 + #5 ) ) } }
915   }
916 }
917 \cs_new_protected:Npn \_color_backend_fill_gray:n #1
918 { \_color_backend_grab:nn { fill } {#1} }
919 \cs_new_protected:Npn \_color_backend_stroke_gray:n #1
920 { \_color_backend_grab:nn { stroke } {#1} }
921 \cs_new_protected:Npn \_color_backend_gray:nn #1#2
922 {
923   \use:x
924   {
925     \_color_backend_gray_aux:nn
926     {#1}
927     { \fp_eval:n { 100 * (#2) } }
928   }
929 }
930 \cs_new_protected:Npn \_color_backend_gray_aux:nn #1#2
931 { \_color_backend:nnn {#1} {#2} {#2} {#2} }
932 \cs_new_protected:Npn \_color_backend_fill_rgb:n #1
933 { \_color_backend_rgb:nw { fill } #1 \s_color_stop }
934 \cs_new_protected:Npn \_color_backend_stroke_rgb:n #1
935 { \_color_backend_rgb:nw { stroke } #1 \s_color_stop }
936 \cs_new_protected:Npn \_color_backend_rgb:nw
937 #1#2 ~ #3 ~ #4 \s_color_stop
938 {
939   \use:x
940   {
941     \_color_backend:nnnn
942     { fill }
943     { \fp_eval:n { 100 * (#2) } }
944     { \fp_eval:n { 100 * (#3) } }
945     { \fp_eval:n { 100 * (#4) } }
946   }
947 }

```

```

948 \cs_new_protected:Npx \__color_backend:nnnn #1#2#3#4
949 {
950   \__kernel_backend_scope:n
951   {
952     #1 =
953     "
954     rgb
955     (
956       #2 \c_percent_str ,
957       #3 \c_percent_str ,
958       #4 \c_percent_str
959     )
960     "
961   }
962 }

```

(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

```

At present, these are no-ops.

```

963 \cs_new_protected:Npn \__color_backend_fill_separation:nn #1#2 { }
964 \cs_new_protected:Npn \__color_backend_stroke_separation:nn #1#2 { }
965 \cs_new_eq:NN \__color_backend_fill_devicen:nn \__color_backend_fill_separation:nn
966 \cs_new_eq:NN \__color_backend_stroke_devicen:nn \__color_backend_stroke_separation:nn

```

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

```

967 </divisvgn>
968 </package>

```

## 4 l3backend-draw Implementation

```

969 <*package>
970 <@@=draw>

```

### 4.1 dvips backend

```

971 <*dvips>

```

```

\__draw_backend_literal:n
\__draw_backend_literal:x

```

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

```

972 \cs_new_eq:NN \__draw_backend_literal:n \__kernel_backend_literal_postscript:n
973 \cs_generate_variant:Nn \__draw_backend_literal:n { x }

```

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

```

\__draw_backend_begin:
\__draw_backend_end:

```

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.

```

974 \cs_new_protected:Npn \__draw_backend_begin:

```

```

975 {
976   \__kernel_backend_literal:n { ps::[begin] }
977   \__draw_backend_literal:n { @beginspecial }
978 }
979 \cs_new_protected:Npn \__draw_backend_end:
980 {
981   \__draw_backend_literal:n { @endspecial }
982   \__kernel_backend_literal:n { ps::[end] }
983 }

```

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

\\_\_draw\_backend\_scope\_begin: Scope here may need to contain saved definitions, so the entire memory rather than just  
 \\_\_draw\_backend\_scope\_end: the graphic state has to be sent to the stack.

```

984 \cs_new_protected:Npn \__draw_backend_scope_begin:
985 { \__draw_backend_literal:n { save } }
986 \cs_new_protected:Npn \__draw_backend_scope_end:
987 { \__draw_backend_literal:n { restore } }

```

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

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

```

988 \cs_new_protected:Npn \__draw_backend_moveto:nn #1#2
989 {
990   \__draw_backend_literal:x
991   {
992     \dim_to_decimal_in_bp:n {#1} ~
993     \dim_to_decimal_in_bp:n {#2} ~ moveto
994   }
995 }
996 \cs_new_protected:Npn \__draw_backend_lineto:nn #1#2
997 {
998   \__draw_backend_literal:x
999   {
1000     \dim_to_decimal_in_bp:n {#1} ~
1001     \dim_to_decimal_in_bp:n {#2} ~ lineto
1002   }
1003 }
1004 \cs_new_protected:Npn \__draw_backend_rectangle:nnnn #1#2#3#4
1005 {
1006   \__draw_backend_literal:x
1007   {
1008     \dim_to_decimal_in_bp:n {#4} ~ \dim_to_decimal_in_bp:n {#3} ~
1009     \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~
1010     moveto~dup~0~rlineto~exch~0~exch~rlineto~neg~0~rlineto~closepath
1011   }
1012 }
1013 \cs_new_protected:Npn \__draw_backend_curveto:nnnnnn #1#2#3#4#5#6
1014 {
1015   \__draw_backend_literal:x

```

```

1016     {
1017         \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~
1018         \dim_to_decimal_in_bp:n {#3} ~ \dim_to_decimal_in_bp:n {#4} ~
1019         \dim_to_decimal_in_bp:n {#5} ~ \dim_to_decimal_in_bp:n {#6} ~
1020         curveto
1021     }
1022 }

```

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

The even-odd rule here can be implemented as a simply switch.

```

\draw_backend_evenodd_rule:
\draw_backend_nonzero_rule:
\gdraw_draw_eor_bool
1023 \cs_new_protected:Npn \draw_backend_evenodd_rule:
1024 { \bool_gset_true:N \gdraw_draw_eor_bool }
1025 \cs_new_protected:Npn \draw_backend_nonzero_rule:
1026 { \bool_gset_false:N \gdraw_draw_eor_bool }
1027 \bool_new:N \gdraw_draw_eor_bool

```

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

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_closepath:
\_draw_backend_stroke:
\_draw_backend_closestroke:
\_draw_backend_fill:
\_draw_backend_fillstroke:
\_draw_backend_clip:
\_draw_backend_discardpath:
\gdraw_draw_clip_bool
1028 \cs_new_protected:Npn \draw_backend_closepath:
1029 { \draw_backend_literal:n { closepath } }
1030 \cs_new_protected:Npn \draw_backend_stroke:
1031 {
1032     \draw_backend_literal:n { gsave }
1033     \draw_backend_literal:n { color.sc }
1034     \draw_backend_literal:n { stroke }
1035     \draw_backend_literal:n { grestore }
1036     \bool_if:NT \gdraw_draw_clip_bool
1037     {
1038         \draw_backend_literal:x
1039         {
1040             \bool_if:NT \gdraw_draw_eor_bool { eo }
1041             clip
1042         }
1043     }
1044     \draw_backend_literal:n { newpath }
1045     \bool_gset_false:N \gdraw_draw_clip_bool
1046 }
1047 \cs_new_protected:Npn \draw_backend_closestroke:
1048 {
1049     \draw_backend_closepath:
1050     \draw_backend_stroke:
1051 }
1052 \cs_new_protected:Npn \draw_backend_fill:
1053 {
1054     \draw_backend_literal:n { gsave }
1055     \draw_backend_literal:n { color.fc }

```

```

1056 \__draw_backend_literal:x
1057 {
1058   \bool_if:NT \g__draw_draw_eor_bool { eo }
1059   fill
1060 }
1061 \__draw_backend_literal:n { grestore }
1062 \bool_if:NT \g__draw_draw_clip_bool
1063 {
1064   \__draw_backend_literal:x
1065   {
1066     \bool_if:NT \g__draw_draw_eor_bool { eo }
1067     clip
1068   }
1069 }
1070 \__draw_backend_literal:n { newpath }
1071 \bool_gset_false:N \g__draw_draw_clip_bool
1072 }
1073 \cs_new_protected:Npn \__draw_backend_fillstroke:
1074 {
1075   \__draw_backend_literal:n { gsave }
1076   \__draw_backend_literal:n { color.sc }
1077   \__draw_backend_literal:n { color.fc }
1078   \__draw_backend_literal:x
1079   {
1080     \bool_if:NT \g__draw_draw_eor_bool { eo }
1081     fill
1082   }
1083   \__draw_backend_literal:n { grestore }
1084   \__draw_backend_literal:n { stroke }
1085   \bool_if:NT \g__draw_draw_clip_bool
1086   {
1087     \__draw_backend_literal:x
1088     {
1089       \bool_if:NT \g__draw_draw_eor_bool { eo }
1090       clip
1091     }
1092   }
1093   \__draw_backend_literal:n { newpath }
1094   \bool_gset_false:N \g__draw_draw_clip_bool
1095 }
1096 \cs_new_protected:Npn \__draw_backend_clip:
1097 { \bool_gset_true:N \g__draw_draw_clip_bool }
1098 \bool_new:N \g__draw_draw_clip_bool
1099 \cs_new_protected:Npn \__draw_backend_discardpath:
1100 {
1101   \bool_if:NT \g__draw_draw_clip_bool
1102   {
1103     \__draw_backend_literal:x
1104     {
1105       \bool_if:NT \g__draw_draw_eor_bool { eo }
1106       clip
1107     }
1108   }
1109   \__draw_backend_literal:n { newpath }

```

```

1110 \bool_gset_false:N \g__draw_draw_clip_bool
1111 }

```

(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:
1112 \cs_new_protected:Npn \__draw_backend_dash_pattern:nn #1#2
1113 {
1114   \__draw_backend_literal:x
1115   {
1116     [
1117       \exp_args:Nf \use:n
1118       { \clist_map_function:nN {#1} \__draw_backend_dash:n }
1119     ] ~
1120     \dim_to_decimal_in_bp:n {#2} ~ setdash
1121   }
1122 }
1123 \cs_new:Npn \__draw_backend_dash:n #1
1124 { ~ \dim_to_decimal_in_bp:n {#1} }
1125 \cs_new_protected:Npn \__draw_backend_linewidth:n #1
1126 {
1127   \__draw_backend_literal:x
1128   { \dim_to_decimal_in_bp:n {#1} ~ setlinewidth }
1129 }
1130 \cs_new_protected:Npn \__draw_backend_miterlimit:n #1
1131 { \__draw_backend_literal:n { #1 ~ setmiterlimit } }
1132 \cs_new_protected:Npn \__draw_backend_cap_but:
1133 { \__draw_backend_literal:n { 0 ~ setlinecap } }
1134 \cs_new_protected:Npn \__draw_backend_cap_round:
1135 { \__draw_backend_literal:n { 1 ~ setlinecap } }
1136 \cs_new_protected:Npn \__draw_backend_cap_rectangle:
1137 { \__draw_backend_literal:n { 2 ~ setlinecap } }
1138 \cs_new_protected:Npn \__draw_backend_join_miter:
1139 { \__draw_backend_literal:n { 0 ~ setlinejoin } }
1140 \cs_new_protected:Npn \__draw_backend_join_round:
1141 { \__draw_backend_literal:n { 1 ~ setlinejoin } }
1142 \cs_new_protected:Npn \__draw_backend_join_bevel:
1143 { \__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.

```

1144 \cs_new_protected:Npn \__draw_backend_cm:nnnn #1#2#3#4
1145 {
1146   \__draw_backend_literal:n
1147   { [ #1 ~ #2 ~ #3 ~ #4 ~ 0 ~ 0 ] ~ concat }
1148 }

```

(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).

```

1149 \cs_new_protected:Npn \_draw_backend_box_use:Nnnnn #1#2#3#4#5
1150 {
1151   \_draw_backend_literal:n { @endspecial }
1152   \_draw_backend_literal:n { [end] }
1153   \_draw_backend_literal:n { [begin] }
1154   \_draw_backend_literal:n { save }
1155   \_draw_backend_literal:n { currentpoint }
1156   \_draw_backend_literal:n { currentpoint~translate }
1157   \_draw_backend_cm:nnnn { 1 } { 0 } { 0 } { -1 }
1158   \_draw_backend_cm:nnnn { #2 } { #3 } { #4 } { #5 }
1159   \_draw_backend_cm:nnnn { 1 } { 0 } { 0 } { -1 }
1160   \_draw_backend_literal:n { neg~exch~neg~exch~translate }
1161   \_draw_backend_literal:n { [end] }
1162   \hbox_overlap_right:n { \box_use:N #1 }
1163   \_draw_backend_literal:n { [begin] }
1164   \_draw_backend_literal:n { restore }
1165   \_draw_backend_literal:n { [end] }
1166   \_draw_backend_literal:n { [begin] }
1167   \_draw_backend_literal:n { @beginspecial }
1168 }

```

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

1169 `</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.

1170 `<*dvipdfmx | luatex | pdftex | xetex>`

### 4.2.1 Drawing

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

Pass data through using a dedicated interface.

```

1171 \cs_new_eq:NN \_draw_backend_literal:n \_kernel_backend_literal_pdf:n
1172 \cs_generate_variant:Nn \_draw_backend_literal:n { x }

```

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

`\_draw_backend_begin:`  
`\_draw_backend_end:`

No special requirements here, so simply set up a drawing scope.

```

1173 \cs_new_protected:Npn \_draw_backend_begin:
1174 { \_draw_backend_scope_begin: }

```

```

1175 \cs_new_protected:Npn \__draw_backend_end:
1176 { \__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: 1177 \cs_new_eq:NN \__draw_backend_scope_begin: \__kernel_backend_scope_begin:
1178 \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 1179 \cs_new_protected:Npn \__draw_backend_moveto:nn #1#2
\__draw_backend_curveto:nnnnnn 1180 {
\__draw_backend_rectangle:nnnn 1181 \__draw_backend_literal:x
1182 { \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~ m }
1183 }
1184 \cs_new_protected:Npn \__draw_backend_lineto:nn #1#2
1185 {
1186 \__draw_backend_literal:x
1187 { \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~ l }
1188 }
1189 \cs_new_protected:Npn \__draw_backend_curveto:nnnnnn #1#2#3#4#5#6
1190 {
1191 \__draw_backend_literal:x
1192 {
1193 \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~
1194 \dim_to_decimal_in_bp:n {#3} ~ \dim_to_decimal_in_bp:n {#4} ~
1195 \dim_to_decimal_in_bp:n {#5} ~ \dim_to_decimal_in_bp:n {#6} ~
1196 c
1197 }
1198 }
1199 \cs_new_protected:Npn \__draw_backend_rectangle:nnnn #1#2#3#4
1200 {
1201 \__draw_backend_literal:x
1202 {
1203 \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~
1204 \dim_to_decimal_in_bp:n {#3} ~ \dim_to_decimal_in_bp:n {#4} ~
1205 re
1206 }
1207 }

```

(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: 1208 \cs_new_protected:Npn \__draw_backend_evenodd_rule:
\g__draw_draw_eor_bool 1209 { \bool_gset_true:N \g__draw_draw_eor_bool }
1210 \cs_new_protected:Npn \__draw_backend_nonzero_rule:
1211 { \bool_gset_false:N \g__draw_draw_eor_bool }
1212 \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:
1213 \cs_new_protected:Npn \__draw_backend_closepath:
1214 { \__draw_backend_literal:n { h } }
1215 \cs_new_protected:Npn \__draw_backend_stroke:
1216 { \__draw_backend_literal:n { S } }
1217 \cs_new_protected:Npn \__draw_backend_closestroke:
1218 { \__draw_backend_literal:n { s } }
1219 \cs_new_protected:Npn \__draw_backend_fill:
1220 {
1221   \__draw_backend_literal:x
1222   { f \bool_if:NT \g__draw_draw_eor_bool * }
1223 }
1224 \cs_new_protected:Npn \__draw_backend_fillstroke:
1225 {
1226   \__draw_backend_literal:x
1227   { B \bool_if:NT \g__draw_draw_eor_bool * }
1228 }
1229 \cs_new_protected:Npn \__draw_backend_clip:
1230 {
1231   \__draw_backend_literal:x
1232   { W \bool_if:NT \g__draw_draw_eor_bool * }
1233 }
1234 \cs_new_protected:Npn \__draw_backend_discardpath:
1235 { \__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:
1236 \cs_new_protected:Npn \__draw_backend_dash_pattern:nn #1#2
1237 {
1238   \__draw_backend_literal:x
1239   {
1240     [
1241       \exp_args:Nf \use:n
1242       { \clist_map_function:nN {#1} \__draw_backend_dash:n }
1243     ] ~
1244     \dim_to_decimal_in_bp:n {#2} ~ d
1245   }
1246 }
1247 \cs_new:Npn \__draw_backend_dash:n #1
1248 { ~ \dim_to_decimal_in_bp:n {#1} }
1249 \cs_new_protected:Npn \__draw_backend_linewidth:n #1
1250 {
1251   \__draw_backend_literal:x
1252   { \dim_to_decimal_in_bp:n {#1} ~ w }
1253 }
1254 \cs_new_protected:Npn \__draw_backend_miterlimit:n #1
1255 { \__draw_backend_literal:x { #1 ~ M } }
1256 \cs_new_protected:Npn \__draw_backend_cap_butt:
1257 { \__draw_backend_literal:n { 0 ~ J } }
1258 \cs_new_protected:Npn \__draw_backend_cap_round:
1259 { \__draw_backend_literal:n { 1 ~ J } }
1260 \cs_new_protected:Npn \__draw_backend_cap_rectangle:
1261 { \__draw_backend_literal:n { 2 ~ J } }

```

```

1262 \cs_new_protected:Npn \__draw_backend_join_miter:
1263 { \__draw_backend_literal:n { 0 ~ j } }
1264 \cs_new_protected:Npn \__draw_backend_join_round:
1265 { \__draw_backend_literal:n { 1 ~ j } }
1266 \cs_new_protected:Npn \__draw_backend_join_bevel:
1267 { \__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!

```

1268 \cs_new_protected:Npn \__draw_backend_cm:nnnn #1#2#3#4
1269 {
1270 <*luatex | pdftex>
1271   \__kernel_backend_matrix:n { #1 ~ #2 ~ #3 ~ #4 }
1272 </luatex | pdftex>
1273 <*dvipdfmx | xetex>
1274   \__draw_backend_cm_decompose:nnnnN {#1} {#2} {#3} {#4}
1275   \__draw_backend_cm_aux:nnnn
1276 </dvipdfmx | xetex>
1277 }
1278 <*dvipdfmx | xetex>
1279 \cs_new_protected:Npn \__draw_backend_cm_aux:nnnn #1#2#3#4
1280 {
1281   \__kernel_backend_literal:x
1282   {
1283     x:rotate~
1284     \fp_compare:nNnTF {#1} = \c_zero_fp
1285       { 0 }
1286       { \fp_eval:n { round ( -#1 , 5 ) } }
1287   }
1288   \__kernel_backend_literal:x
1289   {
1290     x:scale~
1291     \fp_eval:n { round ( #2 , 5 ) } ~
1292     \fp_eval:n { round ( #3 , 5 ) }
1293   }
1294   \__kernel_backend_literal:x
1295   {
1296     x:rotate~
1297     \fp_compare:nNnTF {#4} = \c_zero_fp
1298       { 0 }
1299       { \fp_eval:n { round ( -#4 , 5 ) } }
1300   }
1301 }
1302 </dvipdfmx | xetex>

```

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

`\_draw_backend_cm_decompose:nnnnN`  
`\_draw_backend_cm_decompose_auxi:nnnnN`  
`\_draw_backend_cm_decompose_auxii:nnnnN`  
`\_draw_backend_cm_decompose_auxiii:nnnnN`

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

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

The parent matrix can be converted to

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

From these, we can find that

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

at which point we just have to do various pieces of re-arrangement to get all of the values. (See J. Blinn, *IEEE Comput. Graph. Appl.*, 1996, **16**, 82–88.) There is one wrinkle: the PostScript (and PDF) way of specifying a transformation matrix exchanges where one would normally expect  $B$  and  $C$  to be.

```

1303 <*dvipdfmx | xetex>
1304 \cs_new_protected:Npn \_draw_backend_cm_decompose:nnnnN #1#2#3#4#5
1305 {
1306   \use:x
1307   {
1308     \_draw_backend_cm_decompose_auxi:nnnnN
1309     { \fp_eval:n { (#1 + #4) / 2 } }
1310     { \fp_eval:n { (#1 - #4) / 2 } }
1311     { \fp_eval:n { (#3 + #2) / 2 } }
1312     { \fp_eval:n { (#3 - #2) / 2 } }
1313   }
1314   #5
1315 }
1316 \cs_new_protected:Npn \_draw_backend_cm_decompose_auxi:nnnnN #1#2#3#4#5
1317 {
1318   \use:x
1319   {
1320     \_draw_backend_cm_decompose_auxii:nnnnN
1321     { \fp_eval:n { 2 * sqrt ( #1 * #1 + #4 * #4 ) } }
1322     { \fp_eval:n { 2 * sqrt ( #2 * #2 + #3 * #3 ) } }
1323     { \fp_eval:n { atand ( #3 , #2 ) } }
1324     { \fp_eval:n { atand ( #4 , #1 ) } }
1325   }
1326   #5
1327 }
1328 \cs_new_protected:Npn \_draw_backend_cm_decompose_auxii:nnnnN #1#2#3#4#5

```

```

1329 {
1330   \use:x
1331   {
1332     \__draw_backend_cm_decompose_auxiii:nnnnN
1333     { \fp_eval:n { ( #4 - #3 ) / 2 } }
1334     { \fp_eval:n { ( #1 + #2 ) / 2 } }
1335     { \fp_eval:n { ( #1 - #2 ) / 2 } }
1336     { \fp_eval:n { ( #4 + #3 ) / 2 } }
1337   }
1338   #5
1339 }
1340 \cs_new_protected:Npn \__draw_backend_cm_decompose_auxiii:nnnnN #1#2#3#4#5
1341 {
1342   \fp_compare:nNnTF { abs( #2 ) } > { abs ( #3 ) }
1343   { #5 {#1} {#2} {#3} {#4} }
1344   { #5 {#1} {#3} {#2} {#4} }
1345 }
1346 </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.

```

1347 \cs_new_protected:Npn \__draw_backend_box_use:Nnnnn #1#2#3#4#5
1348 {
1349   \__kernel_backend_scope_begin:
1350   < *luatex | pdftex >
1351   \__draw_backend_cm:nnnn {#2} {#3} {#4} {#5}
1352   < /luatex | pdftex >
1353   < *dvipdfmx | xetex >
1354   \__kernel_backend_literal:n
1355   { pdf:btrans-matrix~ #2 ~ #3 ~ #4 ~ #5 ~ 0 ~ 0 }
1356   < /dvipdfmx | xetex >
1357   \hbox_overlap_right:n { \box_use:N #1 }
1358   < *dvipdfmx | xetex >
1359   \__kernel_backend_literal:n { pdf:etrans }
1360   < /dvipdfmx | xetex >
1361   \__kernel_backend_scope_end:
1362 }

```

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

```

1363 < /dvipdfmx | luatex | pdftex | xetex >

```

### 4.3 dvisvgm backend

```

1364 < *dvisvgm >

```

`\__draw_backend_literal:n`

The same as the more general literal call.

`\__draw_backend_literal:x`

```

1365 \cs_new_eq:NN \__draw_backend_literal:n \__kernel_backend_literal_svg:n
1366 \cs_generate_variant:Nn \__draw_backend_literal:n { x }

```

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

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

```

1367 \cs_new_protected:Npn \_draw_backend_begin:
1368 {
1369   \_kernel_backend_scope_begin:
1370   \_kernel_backend_scope:n { transform="translate({?x},{?y})~scale(1,-1)" }
1371 }
1372 \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_draw_path_tl`

```

1373 \cs_new_protected:Npn \_draw_backend_moveto:nn #1#2
1374 {
1375   \_draw_backend_add_to_path:n
1376   { M ~ \dim_to_decimal:n {#1} ~ \dim_to_decimal:n {#2} }
1377 }
1378 \cs_new_protected:Npn \_draw_backend_lineto:nn #1#2
1379 {
1380   \_draw_backend_add_to_path:n
1381   { L ~ \dim_to_decimal:n {#1} ~ \dim_to_decimal:n {#2} }
1382 }
1383 \cs_new_protected:Npn \_draw_backend_rectangle:nnnn #1#2#3#4
1384 {
1385   \_draw_backend_add_to_path:n
1386   {
1387     M ~ \dim_to_decimal:n {#1} ~ \dim_to_decimal:n {#2}
1388     h ~ \dim_to_decimal:n {#3} ~
1389     v ~ \dim_to_decimal:n {#4} ~
1390     h ~ \dim_to_decimal:n { -#3 } ~
1391     Z
1392   }
1393 }
1394 \cs_new_protected:Npn \_draw_backend_curveto:nnnnnn #1#2#3#4#5#6
1395 {
1396   \_draw_backend_add_to_path:n
1397   {
1398     C ~
1399     \dim_to_decimal:n {#1} ~ \dim_to_decimal:n {#2} ~
1400     \dim_to_decimal:n {#3} ~ \dim_to_decimal:n {#4} ~
1401     \dim_to_decimal:n {#5} ~ \dim_to_decimal:n {#6}
1402   }
1403 }
1404 \cs_new_protected:Npn \_draw_backend_add_to_path:n #1
1405 {
1406   \tl_gset:Nx \g__draw_draw_path_tl
1407   {
1408     \g__draw_draw_path_tl

```

```

1409         \tl_if_empty:NF \g__draw_draw_path_tl { \c_space_tl }
1410         #1
1411     }
1412 }
1413 \tl_new:N \g__draw_draw_path_tl

```

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

\\_\_draw\_backend\_evenodd\_rule: The fill rules here have to be handled as scopes.

```

\__draw_backend_nonzero_rule:
1414 \cs_new_protected:Npn \__draw_backend_evenodd_rule:
1415 { \__draw_backend_scope:n { fill-rule="evenodd" } }
1416 \cs_new_protected:Npn \__draw_backend_nonzero_rule:
1417 { \__draw_backend_scope:n { fill-rule="nonzero" } }

```

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

\\_\_draw\_backend\_path:n Setting fill and stroke effects and doing clipping all has to be done using scopes. This 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_fillstroke:
\__draw_backend_clip:
\__draw_backend_discardpath:
\g__draw_draw_clip_bool
\g__draw_draw_path_int
1418 \cs_new_protected:Npn \__draw_backend_closepath:
1419 { \__draw_backend_add_to_path:n { Z } }
1420 \cs_new_protected:Npn \__draw_backend_path:n #1
1421 {
1422     \bool_if:NTF \g__draw_draw_clip_bool
1423     {
1424         \int_gincr:N \g__draw_clip_path_int
1425         \__draw_backend_literal:x
1426         {
1427             < clipPath~id = " l3cp \int_use:N \g__draw_clip_path_int " >
1428             { ?nl }
1429             <path~d=" \g__draw_draw_path_tl "/> { ?nl }
1430             < /clipPath > { ? nl }
1431             <
1432                 use~xlink:href =
1433                 "\c_hash_str l3path \int_use:N \g__draw_path_int " ~
1434                 #1
1435             />
1436         }
1437         \__draw_backend_scope:x
1438         {
1439             clip-path =
1440             "url( \c_hash_str l3cp \int_use:N \g__draw_clip_path_int )"
1441         }
1442     }
1443     {
1444         \__draw_backend_literal:x
1445         { <path ~ d=" \g__draw_draw_path_tl " ~ #1 /> }
1446     }
1447     \tl_gclear:N \g__draw_draw_path_tl
1448     \bool_gset_false:N \g__draw_draw_clip_bool
1449 }
1450 \int_new:N \g__draw_path_int

```

```

1451 \cs_new_protected:Npn \__draw_backend_stroke:
1452 { \__draw_backend_path:n { style="fill:none" } }
1453 \cs_new_protected:Npn \__draw_backend_closestroke:
1454 {
1455   \__draw_backend_closepath:
1456   \__draw_backend_stroke:
1457 }
1458 \cs_new_protected:Npn \__draw_backend_fill:
1459 { \__draw_backend_path:n { style="stroke:none" } }
1460 \cs_new_protected:Npn \__draw_backend_fillstroke:
1461 { \__draw_backend_path:n { } }
1462 \cs_new_protected:Npn \__draw_backend_clip:
1463 { \bool_gset_true:N \g__draw_draw_clip_bool }
1464 \bool_new:N \g__draw_draw_clip_bool
1465 \cs_new_protected:Npn \__draw_backend_discardpath:
1466 {
1467   \bool_if:NT \g__draw_draw_clip_bool
1468   {
1469     \int_gincr:N \g__draw_clip_path_int
1470     \__draw_backend_literal:x
1471     {
1472       < clipPath~id = " l3cp \int_use:N \g__draw_clip_path_int " >
1473       { ?nl }
1474       <path~d=" \g__draw_draw_path_tl "/> { ?nl }
1475       < /clipPath >
1476     }
1477     \__draw_backend_scope:x
1478     {
1479       clip-path =
1480       "url( \c_hash_str l3cp \int_use:N \g__draw_clip_path_int)"
1481     }
1482   }
1483   \tl_gclear:N \g__draw_draw_path_tl
1484   \bool_gset_false:N \g__draw_draw_clip_bool
1485 }

```

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

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

```

\__draw_backend_dash_pattern:nn
\__draw_backend_dash:n
\__draw_backend_dash_aux:nn
\__draw_backend_linewidth:n
\__draw_backend_miterlimit:n
\__draw_backend_cap_butt:
\__draw_backend_cap_round:
\__draw_backend_cap_rectangle:
\__draw_backend_join_miter:
\__draw_backend_join_round:
\__draw_backend_join_bevel:
1486 \cs_new_protected:Npn \__draw_backend_dash_pattern:nn #1#2
1487 {
1488   \use:x
1489   {
1490     \__draw_backend_dash_aux:nn
1491     { \clist_map_function:nn {#1} \__draw_backend_dash:n }
1492     { \dim_to_decimal:n {#2} }
1493   }
1494 }
1495 \cs_new:Npn \__draw_backend_dash:n #1
1496 { , \dim_to_decimal_in_bp:n {#1} }
1497 \cs_new_protected:Npn \__draw_backend_dash_aux:nn #1#2
1498 {
1499   \__draw_backend_scope:x

```

```

1500     {
1501         stroke-dasharray =
1502         "
1503             \tl_if_empty:oTF { \use_none:n #1 }
1504             { none }
1505             { \use_none:n #1 }
1506         " ~
1507         stroke-offset=" #2 "
1508     }
1509 }
1510 \cs_new_protected:Npn \__draw_backend_linewidth:n #1
1511 { \__draw_backend_scope:x { stroke-width=" \dim_to_decimal:n {#1} " } }
1512 \cs_new_protected:Npn \__draw_backend_miterlimit:n #1
1513 { \__draw_backend_scope:x { stroke-miterlimit=" #1 " } }
1514 \cs_new_protected:Npn \__draw_backend_cap_but:
1515 { \__draw_backend_scope:n { stroke-linecap="butt" } }
1516 \cs_new_protected:Npn \__draw_backend_cap_round:
1517 { \__draw_backend_scope:n { stroke-linecap="round" } }
1518 \cs_new_protected:Npn \__draw_backend_cap_rectangle:
1519 { \__draw_backend_scope:n { stroke-linecap="square" } }
1520 \cs_new_protected:Npn \__draw_backend_join_miter:
1521 { \__draw_backend_scope:n { stroke-linejoin="miter" } }
1522 \cs_new_protected:Npn \__draw_backend_join_round:
1523 { \__draw_backend_scope:n { stroke-linejoin="round" } }
1524 \cs_new_protected:Npn \__draw_backend_join_bevel:
1525 { \__draw_backend_scope:n { stroke-linejoin="bevel" } }

```

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

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

```

1526 \cs_new_protected:Npn \__draw_backend_cm:nnnn #1#2#3#4
1527 {
1528     \__draw_backend_scope:n
1529     {
1530         transform =
1531         " matrix ( #1 , #2 , #3 , #4 , Opt , Opt ) "
1532     }
1533 }

```

(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.

```

1534 \cs_new_protected:Npn \__draw_backend_box_use:Nnnnn #1#2#3#4#5#6#7
1535 {
1536     \__kernel_backend_scope_begin:
1537     \__draw_backend_cm:nnnn {#2} {#3} {#4} {#5}
1538     \__kernel_backend_literal_svg:n
1539     {
1540         < g~
1541         stroke="none"~
1542         transform="scale(-1,1)~translate({?x},{?y})~scale(-1,-1)"
1543         >

```



```

1544     }
1545     \box_set_wd:Nn #1 { Opt }
1546     \box_set_ht:Nn #1 { Opt }
1547     \box_set_dp:Nn #1 { Opt }
1548     \box_use:N #1
1549     \__kernel_backend_literal_svg:n { </g> }
1550     \__kernel_backend_scope_end:
1551 }

```

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

```

1552 </dvismgm>
1553 </package>

```

## 5 l3backend-graphics Implementation

```

1554 <*package>
1555 <@@=graphics>

```

### 5.1 dvips backend

```

1556 <*dvips>

```

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

```

1557 \cs_new_eq:NN \__graphics_backend_getbb_eps:n \graphics_read_bb:n

```

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

\\_\_graphics\_backend\_include\_eps:n The special syntax is relatively clear here: remember we need PostScript sizes here.

```

1558 \cs_new_protected:Npn \__graphics_backend_include_eps:n #1
1559 {
1560     \__kernel_backend_literal:x
1561     {
1562         PSfile = #1 \c_space_tl
1563         llx = \dim_to_decimal_in_bp:n \l_graphics_llx_dim \c_space_tl
1564         lly = \dim_to_decimal_in_bp:n \l_graphics_lly_dim \c_space_tl
1565         urx = \dim_to_decimal_in_bp:n \l_graphics_urx_dim \c_space_tl
1566         ury = \dim_to_decimal_in_bp:n \l_graphics_ury_dim
1567     }
1568 }

```

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

```

1569 </dvips>

```

### 5.2 LuaTeX and pdfTeX backends

```

1570 <*luatex | pdftex>

```

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

```

1571 \tl_new:N \l_graphics_graphics_attr_tl

```

(End definition for \l\_\_graphics\_graphics\_attr\_tl.)

\\_graphics\_backend\_getbb\_jpg:n  
\\_graphics\_backend\_getbb\_pdf:n  
\\_graphics\_backend\_getbb\_png:n  
\\_graphics\_backend\_getbb\_auxi:n  
\\_graphics\_backend\_getbb\_auxii:n

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

```

1572 \cs_new_protected:Npn \_graphics_backend_getbb_jpg:n #1
1573 {
1574   \int_zero:N \l_graphics_page_int
1575   \tl_clear:N \l_graphics_pagebox_tl
1576   \tl_set:Nx \l__graphics_graphics_attr_tl
1577   {
1578     \tl_if_empty:NF \l_graphics_decodearray_tl
1579     { :D \l_graphics_decodearray_tl }
1580     \bool_if:NT \l_graphics_interpolate_bool
1581     { :I }
1582   }
1583   \tl_clear:N \l__graphics_graphics_attr_tl
1584   \_graphics_backend_getbb_auxi:n {#1}
1585 }
1586 \cs_new_eq:NN \_graphics_backend_getbb_png:n \_graphics_backend_getbb_jpg:n
1587 \cs_new_protected:Npn \_graphics_backend_getbb_pdf:n #1
1588 {
1589   \tl_clear:N \l_graphics_decodearray_tl
1590   \bool_set_false:N \l_graphics_interpolate_bool
1591   \tl_set:Nx \l__graphics_graphics_attr_tl
1592   {
1593     : \l_graphics_pagebox_tl
1594     \int_compare:nNnT \l_graphics_page_int > 1
1595     { :P \int_use:N \l_graphics_page_int }
1596   }
1597   \_graphics_backend_getbb_auxi:n {#1}
1598 }
1599 \cs_new_protected:Npn \_graphics_backend_getbb_auxi:n #1
1600 {
1601   \graphics_bb_restore:xF { #1 \l__graphics_graphics_attr_tl }
1602   { \_graphics_backend_getbb_auxii:n {#1} }
1603 }

```

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.

```

1604 \cs_new_protected:Npn \_graphics_backend_getbb_auxii:n #1
1605 {
1606   \tex_immediate:D \tex_pdfximage:D
1607   \bool_lazy_or:nnT
1608   { \l_graphics_interpolate_bool }
1609   { ! \tl_if_empty_p:N \l_graphics_decodearray_tl }
1610   {
1611     attr ~
1612     {
1613       \tl_if_empty:NF \l_graphics_decodearray_tl
1614       { /Decode~[ \l_graphics_decodearray_tl ] }

```

```

1615         \bool_if:NT \l_graphics_interpolate_bool
1616         { /Interpolate~true }
1617     }
1618 }
1619 \int_compare:nNnT \l_graphics_page_int > 0
1620 { page ~ \int_use:N \l_graphics_page_int }
1621 \tl_if_empty:NF \l_graphics_pagebox_tl
1622 { \l_graphics_pagebox_tl }
1623 {#1}
1624 \hbox_set:Nn \l__graphics_internal_box
1625 { \tex_pdfrefximage:D \tex_pdflastximage:D }
1626 \dim_set:Nn \l_graphics_urx_dim { \box_wd:N \l__graphics_internal_box }
1627 \dim_set:Nn \l_graphics_ury_dim { \box_ht:N \l__graphics_internal_box }
1628 \int_const:cn { c__graphics_graphics_ #1 \l__graphics_graphics_attr_tl _int }
1629 { \tex_the:D \tex_pdflastximage:D }
1630 \graphics_bb_save:x { #1 \l__graphics_graphics_attr_tl }
1631 }

```

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

\\_graphics\_backend\_include\_jpg: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.

```

1632 \cs_new_protected:Npn \__graphics_backend_include_jpg:n #1
1633 {
1634     \tex_pdfrefximage:D
1635     \int_use:c { c__graphics_graphics_ #1 \l__graphics_graphics_attr_tl _int }
1636 }
1637 \cs_new_eq:NN \__graphics_backend_include_pdf:n \__graphics_backend_include_jpg:n
1638 \cs_new_eq:NN \__graphics_backend_include_png:n \__graphics_backend_include_jpg:n

```

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

\\_graphics\_backend\_getbb\_eps:n  
\\_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 2<sub>ε</sub> package, but simplified, conversion takes place here if we have shell access.

```

1639 \sys_if_shell:T
1640 {
1641     \str_new:N \l__graphics_backend_dir_str
1642     \str_new:N \l__graphics_backend_name_str
1643     \str_new:N \l__graphics_backend_ext_str
1644     \cs_new_protected:Npn \__graphics_backend_getbb_eps:n #1
1645     {
1646         \file_parse_full_name:nNNN {#1}
1647         \l__graphics_backend_dir_str
1648         \l__graphics_backend_name_str
1649         \l__graphics_backend_ext_str
1650         \exp_args:Nx \__graphics_backend_getbb_eps:nm
1651         {
1652             \l__graphics_backend_name_str - \str_tail:N \l__graphics_backend_ext_str
1653             -converted-to.pdf
1654         }
1655         {#1}

```

```

1656     }
1657     \cs_new_protected:Npn \__graphics_backend_getbb_eps:n #1#2
1658     {
1659         \file_compare_timestamp:nNnT {#2} > {#1}
1660         {
1661             \sys_shell_now:n
1662             { repstopdf ~ #2 ~ #1 }
1663         }
1664         \tl_set:Nn \l_graphics_name_tl {#1}
1665         \__graphics_backend_getbb_pdf:n {#1}
1666     }
1667     \cs_new_protected:Npn \__graphics_backend_include_eps:n #1
1668     {
1669         \file_parse_full_name:nNNN {#1}
1670         \l__graphics_backend_dir_str \l__graphics_backend_name_str \l__graphics_backend_ext_str
1671         \exp_args:Nx \__graphics_backend_include_pdf:n
1672         {
1673             \l__graphics_backend_name_str - \str_tail:N \l__graphics_backend_ext_str
1674             -converted-to.pdf
1675         }
1676     }
1677 }

```

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

```
1678 </luatex | pdftex>
```

### 5.3 dvipdfmx backend

```
1679 < *dvipdfmx | xetex>
```

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

```

\__graphics_backend_getbb_jpg:n 1680 \cs_new_eq:NN \__graphics_backend_getbb_eps:n \graphics_read_bb:n
\__graphics_backend_getbb_pdf:n 1681 < *dvipdfmx>
\__graphics_backend_getbb_png:n 1682 \cs_new_protected:Npn \__graphics_backend_getbb_jpg:n #1
1683 {
1684     \int_zero:N \l_graphics_page_int
1685     \tl_clear:N \l_graphics_pagebox_tl
1686     \graphics_extract_bb:n {#1}
1687 }
1688 \cs_new_eq:NN \__graphics_backend_getbb_png:n \__graphics_backend_getbb_jpg:n
1689 \cs_new_protected:Npn \__graphics_backend_getbb_pdf:n #1
1690 {
1691     \tl_clear:N \l_graphics_decodearray_tl
1692     \bool_set_false:N \l_graphics_interpolate_bool
1693     \graphics_extract_bb:n {#1}
1694 }
1695 < /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.

```
1696 \int_new:N \g__graphics_track_int
```

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

`\_graphics_backend_include_eps:n
 \_graphics_backend_include_jpg:n
 \_graphics_backend_include_pdf:n
 \_graphics_backend_include_png:n
 \_graphics_backend_include_auxi:nn
 \_graphics_backend_include_auxii:nnn
 \_graphics_backend_include_auxiii:nnn`

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.

```

1697 \cs_new_protected:Npn \_graphics_backend_include_eps:n #1
1698 {
1699   \__kernel_backend_literal:x
1700   {
1701     PSfile = #1 \c_space_tl
1702     llx = \dim_to_decimal_in_bp:n \l_graphics_llx_dim \c_space_tl
1703     lly = \dim_to_decimal_in_bp:n \l_graphics_lly_dim \c_space_tl
1704     urx = \dim_to_decimal_in_bp:n \l_graphics_urx_dim \c_space_tl
1705     ury = \dim_to_decimal_in_bp:n \l_graphics_ury_dim
1706   }
1707 }
1708 \cs_new_protected:Npn \_graphics_backend_include_jpg:n #1
1709 { \_graphics_backend_include_auxi:nn {#1} { image } }
1710 \cs_new_eq:NN \_graphics_backend_include_png:n \_graphics_backend_include_jpg:n
1711 <*dvipdfmx>
1712 \cs_new_protected:Npn \_graphics_backend_include_pdf:n #1
1713 { \_graphics_backend_include_auxi:nn {#1} { epdf } }
1714 </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.

```

1715 \cs_new_protected:Npn \_graphics_backend_include_auxi:nn #1#2
1716 {
1717   \_graphics_backend_include_auxii:xnn
1718   {
1719     \tl_if_empty:NF \l_graphics_pagebox_tl
1720     { : \l_graphics_pagebox_tl }
1721     \int_compare:nNnT \l_graphics_page_int > 1
1722     { :P \int_use:N \l_graphics_page_int }
1723     \tl_if_empty:NF \l_graphics_decodearray_tl
1724     { :D \l_graphics_decodearray_tl }
1725     \bool_if:NT \l_graphics_interpolate_bool
1726     { :I }
1727   }
1728   {#1} {#2}
1729 }
1730 \cs_new_protected:Npn \_graphics_backend_include_auxii:nnn #1#2#3
1731 {
1732   \int_if_exist:cTF { c__graphics_graphics_ #2#1 _int }
1733   {
1734     \__kernel_backend_literal:x
1735     { pdf:usexobj~@graphic \int_use:c { c__graphics_graphics_ #2#1 _int } }
1736   }
1737   { \_graphics_backend_include_auxiii:nnn {#2} {#1} {#3} }
1738 }
1739 \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!

```

1740 \cs_new_protected:Npn \__graphics_backend_include_auxiii:nnn #1#2#3
1741 {
1742   \int_gincr:N \g__graphics_track_int
1743   \int_const:cn { c__graphics_graphics_ #1#2 _int } { \g__graphics_track_int }
1744   \__kernel_backend_literal:x
1745   {
1746     pdf:#3~
1747     @graphic \int_use:c { c__graphics_graphics_ #1#2 _int } ~
1748     \int_compare:nNnT \l_graphics_page_int > 1
1749     { page ~ \int_use:N \l_graphics_page_int \c_space_tl }
1750     \tl_if_empty:NF \l_graphics_pagebox_tl
1751     {
1752       pagebox ~ \l_graphics_pagebox_tl \c_space_tl
1753       bbox ~
1754         \dim_to_decimal_in_bp:n \l_graphics_llx_dim \c_space_tl
1755         \dim_to_decimal_in_bp:n \l_graphics_lly_dim \c_space_tl
1756         \dim_to_decimal_in_bp:n \l_graphics_urx_dim \c_space_tl
1757         \dim_to_decimal_in_bp:n \l_graphics_ury_dim \c_space_tl
1758     }
1759     (#1)
1760     \bool_lazy_or:nnT
1761     { \l_graphics_interpolate_bool }
1762     { ! \tl_if_empty_p:N \l_graphics_decodearray_tl }
1763     {
1764       <<
1765       \tl_if_empty:NF \l_graphics_decodearray_tl
1766       { /Decode~[ \l_graphics_decodearray_tl ] }
1767       \bool_if:NT \l_graphics_interpolate_bool
1768       { /Interpolate~true> }
1769     }
1770     >>
1771   }
1772 }

```

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

```

1773 </dvipdfmx | xetex>

```

## 5.4 Xe<sub>La</sub>TeX backend

```

1774 <*xetex>

```

### 5.4.1 Images

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

```

1775 \cs_new_protected:Npn \__graphics_backend_getbb_jpg:n #1
1776 {
1777   \int_zero:N \l_graphics_page_int
1778   \tl_clear:N \l_graphics_pagebox_tl
1779   \__graphics_backend_getbb_auxi:nN {#1} \tex_XeTeXpicfile:D
1780 }

```

```

\__graphics_backend_getbb_jpg:n
\__graphics_backend_getbb_pdf:n
\__graphics_backend_getbb_png:n
\__graphics_backend_getbb_auxi:nN
\__graphics_backend_getbb_auxii:nnN
\__graphics_backend_getbb_auxiii:nNnN
\__graphics_backend_getbb_auxiv:nnNnN
\__graphics_backend_getbb_auxiv:NnNnN
\__graphics_backend_getbb_auxv:nNnN
\__graphics_backend_getbb_auxv:NnNnN
\__graphics_backend_getbb_pagebox:w

```

```

1781 \cs_new_eq:NN \__graphics_backend_getbb_png:n \__graphics_backend_getbb_jpg:n
1782 \cs_new_protected:Npn \__graphics_backend_getbb_pdf:n #1
1783 {
1784   \tl_clear:N \l_graphics_decodearray_tl
1785   \bool_set_false:N \l_graphics_interpolate_bool
1786   \__graphics_backend_getbb_auxi:nN {#1} \tex_XeTeXpdffile:D
1787 }
1788 \cs_new_protected:Npn \__graphics_backend_getbb_auxi:nN #1#2
1789 {
1790   \int_compare:nNnTF \l_graphics_page_int > 1
1791     { \__graphics_backend_getbb_auxii:VnN \l_graphics_page_int {#1} #2 }
1792     { \__graphics_backend_getbb_auxiii:nNnn {#1} #2 { :P 1 } { page 1 } }
1793 }
1794 \cs_new_protected:Npn \__graphics_backend_getbb_auxii:nnN #1#2#3
1795 { \__graphics_backend_getbb_auxiii:nNnn {#2} #3 { :P #1 } { page #1 } }
1796 \cs_generate_variant:Nn \__graphics_backend_getbb_auxii:nnN { V }
1797 \cs_new_protected:Npn \__graphics_backend_getbb_auxiii:nNnn #1#2#3#4
1798 {
1799   \tl_if_empty:NTF \l_graphics_pagebox_tl
1800     { \__graphics_backend_getbb_auxiv:VnNnn \l_graphics_pagebox_tl }
1801     { \__graphics_backend_getbb_auxv:nNnn }
1802     {#1} #2 {#3} {#4}
1803 }
1804 \cs_new_protected:Npn \__graphics_backend_getbb_auxiv:nnNnn #1#2#3#4#5
1805 {
1806   \use:x
1807   {
1808     \__graphics_backend_getbb_auxv:nNnn {#2} #3 { : #1 #4 }
1809     { #5 ~ \__graphics_backend_getbb_pagebox:w #1 }
1810   }
1811 }
1812 \cs_generate_variant:Nn \__graphics_backend_getbb_auxiv:nnNnn { V }
1813 \cs_new_protected:Npn \__graphics_backend_getbb_auxv:nNnn #1#2#3#4
1814 {
1815   \graphics_bb_restore:nF {#1#3}
1816   { \__graphics_backend_getbb_auxvi:nNnn {#1} #2 {#3} {#4} }
1817 }
1818 \cs_new_protected:Npn \__graphics_backend_getbb_auxvi:nNnn #1#2#3#4
1819 {
1820   \hbox_set:Nn \l__graphics_internal_box { #2 #1 ~ #4 }
1821   \dim_set:Nn \l_graphics_urx_dim { \box_wd:N \l__graphics_internal_box }
1822   \dim_set:Nn \l_graphics_ury_dim { \box_ht:N \l__graphics_internal_box }
1823   \graphics_bb_save:n {#1#3}
1824 }
1825 \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  
 \\_\_graphics\_backend\_include\_bitmap\_quote:w

For PDF graphics, properly supporting the `pagebox` concept in  $\text{\TeX}$  is best done using the `\tex_XeTeXpdffile:D` primitive. The syntax here is the same as for the graphic measurement part, although we know at this stage that there must be some valid setting for `\l_graphics_pagebox_tl`.

```

1826 \cs_new_protected:Npn \__graphics_backend_include_pdf:n #1
1827 {

```

```

1828 \tex_XeTeXpdffile:D
1829 \__graphics_backend_include_pdf_quote:w #1 "#1" \s__graphics_stop \c_space_tl
1830 \int_compare:nNnT \l_graphics_page_int > 0
1831 { page ~ \int_use:N \l_graphics_page_int \c_space_tl }
1832 \exp_after:wN \__graphics_backend_getbb_pagebox:w \l_graphics_pagebox_tl
1833 }
1834 \cs_new:Npn \__graphics_backend_include_pdf_quote:w #1 " #2 " #3 \s__graphics_stop
1835 { " #2 " }

```

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

```
1836 </xetex>
```

## 5.5 dvisvgm backend

```
1837 <*dvisvgm>
```

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

```
1838 \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

```

1839 \cs_new_protected:Npn \__graphics_backend_getbb_jpg:n #1
1840 {
1841   \int_zero:N \l_graphics_page_int
1842   \tl_clear:N \l_graphics_pagebox_tl
1843   \graphics_extract_bb:n {#1}
1844 }
1845 \cs_new_eq:NN \__graphics_backend_getbb_png:n \__graphics_backend_getbb_jpg:n

```

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

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

```

1846 \cs_new_protected:Npn \__graphics_backend_getbb_pdf:n #1
1847 {
1848   \tl_clear:N \l_graphics_decodearray_tl
1849   \bool_set_false:N \l_graphics_interpolate_bool
1850   \graphics_extract_bb:n {#1}
1851 }

```

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

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

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

\\_\_graphics\_backend\_include:nn

```

1852 \cs_new_protected:Npn \__graphics_backend_include_eps:n #1
1853 { __graphics_backend_include:nn { PSfile } {#1} }
1854 \cs_new_protected:Npn \__graphics_backend_include_pdf:n #1
1855 { __graphics_backend_include:nn { pdffile } {#1} }
1856 \cs_new_protected:Npn \__graphics_backend_include:nn #1#2
1857 {
1858   \__kernel_backend_literal:x
1859   {
1860     #1 = #2 \c_space_tl
1861     llx = \dim_to_decimal_in_bp:n \l_graphics_llx_dim \c_space_tl

```



```

1862         lly = \dim_to_decimal_in_bp:n \l_graphics_lly_dim \c_space_tl
1863         urx = \dim_to_decimal_in_bp:n \l_graphics_urx_dim \c_space_tl
1864         ury = \dim_to_decimal_in_bp:n \l_graphics_ury_dim
1865     }
1866 }

```

(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_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.

```

1867 \cs_new_protected:Npn \_graphics_backend_include_png:n #1
1868 {
1869     \_kernel_backend_literal:x
1870     {
1871         dvisvgm:img~
1872         \dim_to_decimal:n { \l_graphics_ury_dim } ~
1873         \dim_to_decimal:n { \l_graphics_ury_dim } ~
1874         \_graphics_backend_include_bitmap_quote:w #1 " #1 " \s__graphics_stop
1875     }
1876 }
1877 \cs_new_eq:NN \_graphics_backend_include_jpg:n \_graphics_backend_include_png:n
1878 \cs_new:Npn \_graphics_backend_include_bitmap_quote:w #1 " #2 " #3 \s__graphics_stop
1879 { " #2 " }

```

(End definition for `\_graphics_backend_include_png:n`, `\_graphics_backend_include_jpg:n`, and `\_graphics_backend_include_bitmap_quote:w`.)

```

1880 </dvisvgm>
1881 </package>

```

## 6 l3backend-pdf Implementation

```

1882 <*package>
1883 <@@=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

1884 \box_new:N \l__pdf_internal_box

(End definition for \l__pdf_internal_box.)

```

## 6.2 dvips backend

1885 `<*dvips>`

`\__pdf_backend_pdfmark:n`  
`\__pdf_backend_pdfmark:x`

Used often enough it should be a separate function.

```
1886 \cs_new_protected:Npn \__pdf_backend_pdfmark:n #1
1887 { \__kernel_backend_postscript:n { mark #1 ~ pdfmark } }
1888 \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`

```
1889 \cs_new_protected:Npn \__pdf_backend_catalog_gput:nn #1#2
1890 { \__pdf_backend_pdfmark:n { { Catalog } << /#1 ~ #2 >> /PUT } }
1891 \cs_new_protected:Npn \__pdf_backend_info_gput:nn #1#2
1892 { \__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`

For tracking objects to allow finalisation.

```
1893 \int_new:N \g__pdf_backend_object_int
1894 \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`.

```
1895 \cs_new_protected:Npn \__pdf_backend_object_new:nn #1#2
1896 {
1897   \int_gincr:N \g__pdf_backend_object_int
1898   \int_const:cn
1899   { c__pdf_backend_object_ \tl_to_str:n {#1} _int }
1900   { \g__pdf_backend_object_int }
1901   \prop_gput:Nnn \g__pdf_backend_object_prop {#1} {#2}
1902 }
1903 \cs_new:Npn \__pdf_backend_object_ref:n #1
1904 { { pdf.obj \int_use:c { c__pdf_backend_object_ \tl_to_str:n {#1} _int } } }
```

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

`\_pdf_backend_object_write:nn`  
`\_pdf_backend_object_write:nx`  
`\__pdf_backend_object_write_array:nn`  
`\_pdf_backend_object_write_dict:nn`  
`\__pdf_backend_object_write_fstream:nn`  
`\_pdf_backend_object_write_stream:nn`  
`\_pdf_backend_object_write_stream:nmm`

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

```
1905 \cs_new_protected:Npn \__pdf_backend_object_write:nn #1#2
1906 {
1907   \__pdf_backend_pdfmark:x
1908   {
1909     /_objdef ~ \__pdf_backend_object_ref:n {#1}
1910     /type
1911     \str_case_e:nn
1912     { \prop_item:Nn \g__pdf_backend_object_prop {#1} }
1913     {
1914       { array } { /array }
1915       { dict } { /dict }
```

```

1916         { fstream } { /stream }
1917         { stream } { /stream }
1918     }
1919     /OBJ
1920 }
1921 \use:c
1922 { __pdf_backend_object_write_ \prop_item:Nn \g__pdf_backend_object_prop {#1} :nn }
1923 { \__pdf_backend_object_ref:n {#1} } {#2}
1924 }
1925 \cs_generate_variant:Nn \__pdf_backend_object_write:nn { nx }
1926 \cs_new_protected:Npn \__pdf_backend_object_write_array:nn #1#2
1927 {
1928     \__pdf_backend_pdfmark:x
1929     { #1 ~0~ [ ~ \exp_not:n {#2} ~ ] ~ /PUTINTERVAL }
1930 }
1931 \cs_new_protected:Npn \__pdf_backend_object_write_dict:nn #1#2
1932 {
1933     \__pdf_backend_pdfmark:x
1934     { #1 << \exp_not:n {#2} >> /PUT }
1935 }
1936 \cs_new_protected:Npn \__pdf_backend_object_write_fstream:nn #1#2
1937 {
1938     \exp_args:Nx
1939     \__pdf_backend_object_write_fstream:nnn {#1} #2
1940 }
1941 \cs_new_protected:Npn \__pdf_backend_object_write_fstream:nnn #1#2#3
1942 {
1943     \__kernel_backend_postscript:n
1944     {
1945         SDict ~ begin ~
1946         mark ~ #1 ~ << #2 >> /PUT ~ pdfmark ~
1947         mark ~ #1 ~ ( #3 )~ ( r )~ file ~ /PUT ~ pdfmark ~
1948         end
1949     }
1950 }
1951 \cs_new_protected:Npn \__pdf_backend_object_write_stream:nn #1#2
1952 {
1953     \exp_args:Nx
1954     \__pdf_backend_object_write_stream:nnn {#1} #2
1955 }
1956 \cs_new_protected:Npn \__pdf_backend_object_write_stream:nnn #1#2#3
1957 {
1958     \__kernel_backend_postscript:n
1959     {
1960         mark ~ #1 ~ ( #3 ) /PUT ~ pdfmark ~
1961         mark ~ #1 ~ << #2 >> /PUT ~ pdfmark
1962     }
1963 }

```

(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 1964 \cs_new_protected:Npn \__pdf_backend_object_now:nn #1#2
1965 {

```

```

1966 \int_gincr:N \g__pdf_backend_object_int
1967 \__pdf_backend_pdfmark:x
1968 {
1969   /objdef ~ { pdf.obj \int_use:N \g__pdf_backend_object_int }
1970   /type
1971   \str_case:nn
1972     {#1}
1973     {
1974       { array } { /array }
1975       { dict } { /dict }
1976       { fstream } { /stream }
1977       { stream } { /stream }
1978     }
1979   /OBJ
1980 }
1981 \exp_args:Nnx \use:c { __pdf_backend_object_write_ #1 :nn }
1982 { { pdf.obj \int_use:N \g__pdf_backend_object_int } } {#2}
1983 }
1984 \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.

```

1985 \cs_new:Npn \__pdf_backend_object_last:
1986   { { 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.

```

1987 \cs_new:Npn \_pdf_backend_pageobject_ref:n #1
1988   { { 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.

```

1989 \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.

```

1990 \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.

```

1991 \int_new:N \g__pdf_backend_annotation_int

```

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

`\_pdf_backend_annotation:nnnn`

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

```
1992 \cs_new_protected:Npn \_pdf_backend_annotation:nnnn #1#2#3#4
1993 {
1994   \exp_args:Nf \_pdf_backend_annotation_aux:nnnn
1995   { \dim_eval:n {#1} } {#2} {#3} {#4}
1996 }
1997 \cs_new_protected:Npn \_pdf_backend_annotation_aux:nnnn #1#2#3#4
1998 {
1999   \box_move_down:nn {#3}
2000   { \hbox:n { \_kernel_backend_postscript:n { pdf.save.ll } } }
2001   \box_move_up:nn {#2}
2002   {
2003     \hbox:n
2004     {
2005       \tex_kern:D #1 \scan_stop:
2006       \_kernel_backend_postscript:n { pdf.save.ur }
2007       \tex_kern:D -#1 \scan_stop:
2008     }
2009   }
2010   \int_gincr:N \g__pdf_backend_object_int
2011   \int_gset_eq:NN \g__pdf_backend_annotation_int \g__pdf_backend_object_int
2012   \_pdf_backend_pdfmark:x
2013   {
2014     /objdef { pdf.obj \int_use:N \g__pdf_backend_object_int }
2015     pdf.rect
2016     #4 ~
2017     /ANN
2018   }
2019 }
```

(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.

```
2020 \cs_new:Npn \_pdf_backend_annotation_last:
2021 { { 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.

```
2022 \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.

```
2023 \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.

```
2024 \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.

2025 `\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.

2026 `\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.

2027 `\tl_new:N \l__pdf_breaklink_pdfmark_tl`

2028 `\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.

2029 `\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.

2030 `\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 crated like annotations but with dedicated code to allow for adjusting the size of the rectangle. In contrast to `hyperref`, we grab the link content as a box which can then unbox: this allows the same interface as for `pdfTeX`.

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 format mode are still to re-examine.

2031 `\cs_new_protected:Npn \__pdf_backend_link_begin_goto:nnw #1#2`

2032 `{ \__pdf_backend_link_begin:nw { #1 /Subtype /Link /A << /S /GoTo /D ( #2 ) >> } }`

2033 `\cs_new_protected:Npn \__pdf_backend_link_begin_user:nnw #1#2`

2034 `{ \__pdf_backend_link_begin:nw {#1#2} }`

2035 `\cs_new_protected:Npn \__pdf_backend_link_begin:nw #1`

2036 `{`

2037 `\bool_if:NF \g__pdf_backend_link_bool`

2038 `{ \__pdf_backend_link_begin_aux:nw {#1} }`

2039 `}`

2040 `\cs_new_protected:Npn \__pdf_backend_link_begin_aux:nw #1`

2041 `{`

2042 `\bool_gset_true:N \g__pdf_backend_link_bool`

```

2043 \__kernel_backend_postscript:n
2044 { /pdf.link.dict ( #1 ) def }
2045 \tl_gset:Nn \g__pdf_backend_link_dict_tl {#1}
2046 \__pdf_backend_link_sf_save:
2047 \mode_if_math:TF
2048 { \bool_gset_true:N \g__pdf_backend_link_math_bool }
2049 { \bool_gset_false:N \g__pdf_backend_link_math_bool }
2050 \hbox_set:Nw \l__pdf_backend_content_box
2051 \__pdf_backend_link_sf_restore:
2052 \bool_if:NT \g__pdf_backend_link_math_bool
2053 { \c_math_toggle_token }
2054 }
2055 \cs_new_protected:Npn \__pdf_backend_link_end:
2056 {
2057   \bool_if:NT \g__pdf_backend_link_bool
2058   { \__pdf_backend_link_end_aux: }
2059 }
2060 \cs_new_protected:Npn \__pdf_backend_link_end_aux:
2061 {
2062   \bool_if:NT \g__pdf_backend_link_math_bool
2063   { \c_math_toggle_token }
2064   \__pdf_backend_link_sf_save:
2065   \hbox_set_end:
2066   \__pdf_backend_link_minima:
2067   \hbox_set:Nn \l__pdf_backend_model_box { Gg }
2068   \exp_args:Nx \__pdf_backend_link_outerbox:n
2069   {
2070     \int_if_odd:nTF { \value { page } }
2071     { \oddsidemargin }
2072     { \evensidemargin }
2073   }
2074   \box_move_down:nn { \box_dp:N \l__pdf_backend_content_box }
2075   { \hbox:n { \__kernel_backend_postscript:n { pdf.save.linkll } } }
2076   \__pdf_breaklink_postscript:n { pdf.bordertracking.begin }
2077   \__pdf_breaklink_usebox:N \l__pdf_backend_content_box
2078   \__pdf_breaklink_postscript:n { pdf.bordertracking.end }
2079   \box_move_up:nn { \box_ht:N \l__pdf_backend_content_box }
2080   {
2081     \hbox:n
2082     { \__kernel_backend_postscript:n { pdf.save.linkur } }
2083   }
2084   \int_gincr:N \g__pdf_backend_object_int
2085   \int_gset_eq:NN \g__pdf_backend_link_int \g__pdf_backend_object_int
2086   \__kernel_backend_postscript:x
2087   {
2088     mark
2089     /objdef { pdf.obj \int_use:N \g__pdf_backend_link_int }
2090     \g__pdf_backend_link_dict_tl \c_space_tl
2091     pdf.rect
2092     /ANN ~ \l__pdf_breaklink_pdfmark_tl
2093   }
2094   \__pdf_backend_link_sf_restore:
2095   \bool_gset_false:N \g__pdf_backend_link_bool
2096 }

```

```

2097 \cs_new_protected:Npn \__pdf_backend_link_minima:
2098 {
2099   \hbox_set:Nn \l__pdf_backend_model_box { Gg }
2100   \__kernel_backend_postscript:x
2101   {
2102     /pdf.linkdp.pad ~
2103     \dim_to_decimal:n
2104     {
2105       \dim_max:nn
2106       {
2107         \box_dp:N \l__pdf_backend_model_box
2108         - \box_dp:N \l__pdf_backend_content_box
2109       }
2110       { Opt }
2111     } ~
2112     pdf.pt.dvi ~ def
2113   /pdf.linkht.pad ~
2114   \dim_to_decimal:n
2115   {
2116     \dim_max:nn
2117     {
2118       \box_ht:N \l__pdf_backend_model_box
2119       - \box_ht:N \l__pdf_backend_content_box
2120     }
2121     { Opt }
2122   } ~
2123   pdf.pt.dvi ~ def
2124 }
2125 }
2126 \cs_new_protected:Npn \__pdf_backend_link_outerbox:n #1
2127 {
2128   \__kernel_backend_postscript:x
2129   {
2130     /pdf.outerbox
2131     [
2132       \dim_to_decimal:n {#1} ~
2133       \dim_to_decimal:n { -\box_dp:N \l__pdf_backend_model_box } ~
2134       \dim_to_decimal:n { #1 + \textwidth } ~
2135       \dim_to_decimal:n { \box_ht:N \l__pdf_backend_model_box }
2136     ]
2137     [ exch { pdf.pt.dvi } forall ] def
2138   /pdf.baselineskip ~
2139   \dim_to_decimal:n { \tex_baselineskip:D } ~ dup ~ 0 ~ gt
2140   { pdf.pt.dvi ~ def }
2141   { pop ~ pop }
2142   ifelse
2143 }
2144 }
2145 \cs_new_protected:Npn \__pdf_backend_link_sf_save:
2146 {
2147   \int_gset:Nn \g__pdf_backend_link_sf_int
2148   {
2149     \mode_if_horizontal:TF
2150     { \tex_spacefactor:D }

```



```

2151         { 0 }
2152     }
2153 }
2154 \cs_new_protected:Npn \__pdf_backend_link_sf_restore:
2155 {
2156     \mode_if_horizontal:T
2157     {
2158         \int_compare:nNnT \g__pdf_backend_link_sf_int > { 0 }
2159         { \int_set_eq:NN \tex_spacefactor:D \g__pdf_backend_link_sf_int }
2160     }
2161 }

```

(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.

```

2162 \*package
2163 \use_none:n
2164 {
2165     \cs_if_exist:NT \@makecol@hook
2166     {
2167         \tl_put_right:Nn \@makecol@hook
2168         {
2169             \box_if_empty:NF \@cclv
2170             {
2171                 \vbox_set:Nn \@cclv
2172                 {
2173                     \__kernel_backend_postscript:n
2174                     {
2175                         pdf.globaldict /pdf.brokenlink.rect ~ known
2176                         { pdf.bordertracking.continue }
2177                         if
2178                     }
2179                     \vbox_unpack_drop:N \@cclv
2180                     \__kernel_backend_postscript:n
2181                     { pdf.bordertracking.endpage }
2182                 }
2183             }
2184         }
2185         \tl_set:Nn \l__pdf_breaklink_pdfmark_tl { pdf.pdfmark }
2186         \cs_set_eq:NN \__pdf_breaklink_postscript:n \__kernel_backend_postscript:n
2187         \cs_set_eq:NN \__pdf_breaklink_usebox:N \hbox_unpack:N
2188     }
2189 }
2190 \</package>

```

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

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

```

2191 \cs_new:Npn \__pdf_backend_link_last:
2192 { { 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.

```

2193 \cs_new_protected:Npn \_pdf_backend_link_margin:n #1
2194 {
2195   \_kernel_backend_postscript:x
2196   {
2197     /pdf.linkmargin { \dim_to_decimal:n {#1} ~ pdf.pt.dvi } def
2198   }
2199 }

```

(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.

`\_pdf_backend_destination_box:nn`

```

2200 \cs_new_protected:Npn \_pdf_backend_destination:nn #1#2
2201 {
2202   \_kernel_backend_postscript:n { pdf.dest.anchor }
2203   \_pdf_backend_pdfmark:x
2204   {
2205     /View
2206     [
2207       \str_case:nnF {#2}
2208       {
2209         { xyz } { /XYZ ~ pdf.dest.point ~ null }
2210         { fit } { /Fit }
2211         { fitb } { /FitB }
2212         { fitbh } { /FitBH ~ pdf.dest.y }
2213         { fitbv } { /FitBV ~ pdf.dest.x }
2214         { fith } { /FitH ~ pdf.dest.y }
2215         { fitv } { /FitV ~ pdf.dest.x }
2216       }
2217       {
2218         /XYZ ~ pdf.dest.point ~ \fp_eval:n { (#2) / 100 }
2219       }
2220     ]
2221     /Dest ( \exp_not:n {#1} ) cvn
2222     /DEST
2223   }
2224 }
2225 \cs_new_protected:Npn \_pdf_backend_destination_box:nn #1#2
2226 {
2227   \group_begin:
2228   \hbox_set:Nn \l__pdf_internal_box {#2}
2229   \box_move_down:nn
2230   { \box_dp:N \l__pdf_internal_box }
2231   { \hbox:n { \_kernel_backend_postscript:n { pdf.save.ll } } }
2232   \box_use:N \l__pdf_internal_box
2233   \box_move_up:nn
2234   { \box_ht:N \l__pdf_internal_box }
2235   { \hbox:n { \_kernel_backend_postscript:n { pdf.save.ur } } }
2236   \_pdf_backend_pdfmark:n
2237   {
2238     /View
2239     [

```

```

2240         /FitR ~
2241         pdf.llx ~ pdf.lly ~ pdf.dest2device ~
2242         pdf.urx ~ pdf.ury ~ pdf.dest2device
2243     ]
2244     /Dest ( #1 ) cvn
2245     /DEST
2246 }
2247 \group_end:
2248 }

```

(End definition for `\_pdf_backend_destination:n` and `\_pdf_backend_destination_box:n`.)

## 6.2.4 Structure

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

```

\_pdf_backend_compress_objects:n
2249 \cs_new_protected:Npn \_pdf_backend_compresslevel:n #1
2250 {
2251     \int_compare:nNtT {#1} = 0
2252     {
2253         \__kernel_backend_literal_postscript:n
2254         {
2255             /setdistillerparams ~ where
2256             { pop << /CompressPages ~ false >> setdistillerparams }
2257             if
2258         }
2259     }
2260 }
2261 \cs_new_protected:Npn \_pdf_backend_compress_objects:n #1
2262 {
2263     \bool_if:nF {#1}
2264     {
2265         \__kernel_backend_literal_postscript:n
2266         {
2267             /setdistillerparams ~ where
2268             { pop << /CompressStreams ~ false >> setdistillerparams }
2269             if
2270         }
2271     }
2272 }

```

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

`\_pdf_backend_version_major_gset:n` Data not available!

```

\_pdf_backend_version_minor_gset:n
2273 \cs_new_protected:Npn \_pdf_backend_version_major_gset:n #1 { }
2274 \cs_new_protected:Npn \_pdf_backend_version_minor_gset:n #1 { }

```

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

`\_pdf_backend_version_major:` Data not available!

```

\_pdf_backend_version_minor:
2275 \cs_new:Npn \_pdf_backend_version_major: { -1 }
2276 \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.

```

2277 \cs_new_protected:Npn \__pdf_backend_bdc:nn #1#2
2278 { \__pdf_backend_pdfmark:n { /#1 ~ #2 /BDC } }
2279 \cs_new_protected:Npn \__pdf_backend_emc:
2280 { \__pdf_backend_pdfmark:n { /EMC } }

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

2281 </dvips>

```

## 6.3 LuaTeX and pdfTeX backend

```
2282 <*luatex | pdftex>
```

### 6.3.1 Annotations

Simply pass the raw data through, just dealing with evaluation of dimensions.

```

2283 \cs_new_protected:Npn \__pdf_backend_annotation:nnnn #1#2#3#4
2284 {
2285   <*luatex>
2286     \tex_pdfextension:D annot ~
2287   </luatex>
2288   <*pdftex>
2289     \tex_pdfannot:D
2290   </pdftex>
2291     width ~ \dim_eval:n {#1} ~
2292     height ~ \dim_eval:n {#2} ~
2293     depth ~ \dim_eval:n {#3} ~
2294     {#4}
2295 }

```

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

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.

```

2296 \cs_new:Npx \__pdf_backend_annotation_last:
2297 {
2298   \exp_not:N \int_value:w
2299   <*luatex>
2300     \exp_not:N \tex_pdffeedback:D lastannot ~
2301   </luatex>
2302   <*pdftex>
2303     \exp_not:N \tex_pdflastannot:D
2304   </pdftex>
2305     \c_space_tl 0 ~ R
2306 }

```

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

Links are all created using the same internals.

```

2307 \cs_new_protected:Npn \__pdf_backend_link_begin_goto:nnw #1#2
2308 { \__pdf_backend_link_begin:nnnw {#1} { goto~name } {#2} }
2309 \cs_new_protected:Npn \__pdf_backend_link_begin_user:nnw #1#2

```

```

2310 { \_pdf_backend_link_begin:nnnw {#1} { user } {#2} }
2311 \cs_new_protected:Npn \_pdf_backend_link_begin:nnnw #1#2#3
2312 {
2313   \*luatex
2314     \tex_pdfextension:D startlink ~
2315   \*pdftex
2316     \tex_pdfstartlink:D
2317   \*pdftex
2318     attr {#1}
2319     #2 {#3}
2320 }
2321 \cs_new_protected:Npn \_pdf_backend_link_end:
2322 {
2323   \*luatex
2324     \tex_pdfextension:D endlink \scan_stop:
2325   \*pdftex
2326     \tex_pdfendlink:D
2327   \*pdftex
2328 }
2329
2330

```

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

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

```

2331 \cs_new:Npx \_pdf_backend_link_last:
2332 {
2333   \exp_not:N \int_value:w
2334   \*luatex
2335     \exp_not:N \tex_pdffeedback:D lastlink ~
2336   \*pdftex
2337     \exp_not:N \tex_pdflastlink:D
2338   \c_space_tl 0 ~ R
2339 }
2340
2341

```

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

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

```

2342 \cs_new_protected:Npn \_pdf_backend_link_margin:n #1
2343 {
2344   \*luatex
2345     \tex_pdfvariable:D linkmargin
2346   \*pdftex
2347     \tex_pdflinkmargin:D
2348   \dim_eval:n {#1} \scan_stop:
2349 }
2350
2351

```

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

`\_pdf_backend_destination:nn`  
`\_pdf_backend_destination_box:nn`

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

```

2352 \cs_new_protected:Npn \_pdf_backend_destination:nn #1#2
2353 {
2354   \*luatex
2355   \tex_pdfextension:D dest ~
2356   \*luatex
2357   \*pdftex
2358   \tex_pdfdest:D
2359   \*pdftex
2360   name {#1}
2361   \str_case:nnF {#2}
2362   {
2363     { xyz } { xyz }
2364     { fit } { fit }
2365     { fitb } { fitb }
2366     { fitbh } { fitbh }
2367     { fitbv } { fitbv }
2368     { fith } { fith }
2369     { fitv } { fitv }
2370   }
2371   { xyz ~ zoom \fp_eval:n { #2 * 10 } }
2372   \scan_stop:
2373 }
2374 \cs_new_protected:Npn \_pdf_backend_destination_box:nn #1#2
2375 {
2376   \group_begin:
2377   \hbox_set:Nn \l__pdf_internal_box {#2}
2378   \*luatex
2379   \tex_pdfextension:D dest ~
2380   \*luatex
2381   \*pdftex
2382   \tex_pdfdest:D
2383   \*pdftex
2384   name {#1}
2385   \ftr ~
2386   width \box_wd:N \l__pdf_internal_box
2387   height \box_ht:N \l__pdf_internal_box
2388   depth \box_dp:N \l__pdf_internal_box
2389   \box_use:N \l__pdf_internal_box
2390   \group_end:
2391 }

```

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

### 6.3.2 Catalogue entries

`\_pdf_backend_catalog_gput:nn`  
`\_pdf_backend_info_gput:nn`

```

2392 \cs_new_protected:Npn \_pdf_backend_catalog_gput:nn #1#2
2393 {
2394   \*luatex
2395   \tex_pdfextension:D catalog
2396   \*luatex

```

```

2397 <*pdftex>
2398     \tex_pdfcatalog:D
2399 </pdftex>
2400     { / #1 ~ #2 }
2401 }
2402 \cs_new_protected:Npn \__pdf_backend_info_gput:nn #1#2
2403 {
2404 <*luatex>
2405     \tex_pdfextension:D info
2406 </luatex>
2407 <*pdftex>
2408     \tex_pdfinfo:D
2409 </pdftex>
2410     { / #1 ~ #2 }
2411 }

(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.

```

2412 \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
2413 \cs_new_protected:Npn \__pdf_backend_object_new:nn #1#2
2414 {
2415 <*luatex>
2416     \tex_pdfextension:D obj ~
2417 </luatex>
2418 <*pdftex>
2419     \tex_pdfobj:D
2420 </pdftex>
2421     reserveobjnum ~
2422     \int_const:cn
2423     { c__pdf_backend_object_ \tl_to_str:n {#1} _int }
2424 <*luatex>
2425     { \tex_pdffeedback:D lastobj }
2426 </luatex>
2427 <*pdftex>
2428     { \tex_pdflastobj:D }
2429 </pdftex>
2430     \prop_gput:Nnn \g__pdf_backend_object_prop {#1} {#2}
2431 }
2432 \cs_new:Npn \__pdf_backend_object_ref:n #1
2433 { \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
2434 \cs_new_protected:Npn \__pdf_backend_object_write:nn #1#2
2435 {
2436 <*luatex>
2437     \tex_immediate:D \tex_pdfextension:D obj ~

```

```

2438 </luatex>
2439 <*pdftex>
2440   \tex_immediate:D \tex_pdfobj:D
2441 </pdftex>
2442   useobjnum ~
2443   \int_use:c
2444   { c__pdf_backend_object_ \tl_to_str:n {#1} _int }
2445   \str_case:e:nn
2446   { \prop_item:Nn \g__pdf_backend_object_prop {#1} }
2447   {
2448     { array } { { [ ~ \exp_not:n {#2} ~ ] } }
2449     { dict } { { << ~ \exp_not:n {#2} ~ >> } }
2450     { fstream }
2451     {
2452       stream ~ attr ~ { \__pdf_exp_not_i:nn #2 } ~
2453       file ~ { \__pdf_exp_not_ii:nn #2 }
2454     }
2455     { stream }
2456     {
2457       stream ~ attr ~ { \__pdf_exp_not_i:nn #2 } ~
2458       { \__pdf_exp_not_ii:nn #2 }
2459     }
2460   }
2461 }
2462 \cs_generate_variant:Nn \__pdf_backend_object_write:nn { nx }
2463 \cs_new:Npn \__pdf_exp_not_i:nn #1#2 { \exp_not:n {#1} }
2464 \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
2465 \cs_new_protected:Npn \__pdf_backend_object_now:nn #1#2
2466 {
2467   <*luatex>
2468   \tex_immediate:D \tex_pdfextension:D obj ~
2469   </luatex>
2470   <*pdftex>
2471   \tex_immediate:D \tex_pdfobj:D
2472   </pdftex>
2473   \str_case:nn
2474   {#1}
2475   {
2476     { array } { { [ ~ \exp_not:n {#2} ~ ] } }
2477     { dict } { { << ~ \exp_not:n {#2} ~ >> } }
2478     { fstream }
2479     {
2480       stream ~ attr ~ { \__pdf_exp_not_i:nn #2 } ~
2481       file ~ { \__pdf_exp_not_ii:nn #2 }
2482     }
2483     { stream }
2484     {
2485       stream ~ attr ~ { \__pdf_exp_not_i:nn #2 } ~
2486       { \__pdf_exp_not_ii:nn #2 }

```



```

2487         }
2488     }
2489 }
2490 \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.

```

2491 \cs_new:Npx \_pdf_backend_object_last:
2492 {
2493     \exp_not:N \int_value:w
2494     \*luatex
2495     \exp_not:N \tex_pdffeedback:D lastobj ~
2496     \*luatex
2497     \*pdftex
2498     \exp_not:N \tex_pdflastobj:D
2499     \*pdftex
2500     \c_space_tl 0 ~ R
2501 }
(End definition for \_pdf_backend_object_last:.)

```

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

```

2502 \cs_new:Npx \_pdf_backend_pageobject_ref:n #1
2503 {
2504     \exp_not:N \int_value:w
2505     \*luatex
2506     \exp_not:N \tex_pdffeedback:D pageref
2507     \*luatex
2508     \*pdftex
2509     \exp_not:N \tex_pdfpageref:D
2510     \*pdftex
2511     \c_space_tl #1 \c_space_tl \c_space_tl \c_space_tl 0 ~ R
2512 }
(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
2513 \cs_new_protected:Npn \_pdf_backend_compresslevel:n #1
2514 {
2515     \tex_global:D
2516     \*luatex
2517     \tex_pdfvariable:D compresslevel
2518     \*luatex
2519     \*pdftex
2520     \tex_pdfcompresslevel:D
2521     \*pdftex
2522     \int_value:w \int_eval:n {#1} \scan_stop:
2523 }
2524 \cs_new_protected:Npn \_pdf_backend_compress_objects:n #1
2525 {
2526     \bool_if:nTF {#1}
2527     { \_pdf_backend_objcompresslevel:n { 2 } }

```

```

2528     { \_pdf_backend_objcompresslevel:n { 0 } }
2529   }
2530   \cs_new_protected:Npn \_pdf_backend_objcompresslevel:n #1
2531   {
2532     \tex_global:D
2533     <*luatex>
2534     \tex_pdfvariable:D objcompresslevel
2535     </luatex>
2536     <*pdftex>
2537     \tex_pdfobjcompresslevel:D
2538     </pdftex>
2539     #1 \scan_stop:
2540   }

```

(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.

```

2541   \cs_new_protected:Npx \_pdf_backend_version_major_gset:n #1
2542   {
2543     <*luatex>
2544     \int_compare:nNnT \tex_luatexversion:D > { 106 }
2545     {
2546       \exp_not:N \tex_global:D \tex_pdfvariable:D majorversion
2547       \exp_not:N \int_eval:n {#1} \scan_stop:
2548     }
2549     </luatex>
2550     <*pdftex>
2551     \cs_if_exist:NT \tex_pdfmajorversion:D
2552     {
2553       \exp_not:N \tex_global:D \tex_pdfmajorversion:D
2554       \exp_not:N \int_eval:n {#1} \scan_stop:
2555     }
2556     </pdftex>
2557   }
2558   \cs_new_protected:Npn \_pdf_backend_version_minor_gset:n #1
2559   {
2560     \tex_global:D
2561     <*luatex>
2562     \tex_pdfvariable:D minorversion
2563     </luatex>
2564     <*pdftex>
2565     \tex_pdfminorversion:D
2566     </pdftex>
2567     \int_eval:n {#1} \scan_stop:
2568   }

```

(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.

```

2569   \cs_new:Npx \_pdf_backend_version_major:
2570   {
2571     <*luatex>
2572     \int_compare:nNnTF \tex_luatexversion:D > { 106 }
2573     { \exp_not:N \tex_the:D \tex_pdfvariable:D majorversion }

```

```

2574     { 1 }
2575 </luatex>
2576 <*pdftex>
2577     \cs_if_exist:NTF \tex_pdfmajorversion:D
2578     { \exp_not:N \tex_the:D \tex_pdfmajorversion:D }
2579     { 1 }
2580 </pdftex>
2581 }
2582 \cs_new:Npn \__pdf_backend_version_minor:
2583 {
2584     \tex_the:D
2585 <*luatex>
2586     \tex_pdfvariable:D minorversion
2587 </luatex>
2588 <*pdftex>
2589     \tex_pdfminorversion:D
2590 </pdftex>
2591 }

```

(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>.

```

2592 \cs_new_protected:Npn \__pdf_backend_bdc:nn #1#2
2593 { \__kernel_backend_literal_page:n { /#1 ~ #2 ~ BDC } }
2594 \cs_new_protected:Npn \__pdf_backend_emc:
2595 { \__kernel_backend_literal_page:n { EMC } }

```

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

```

2596 </luatex | pdftex>

```

## 6.4 dvipdfmx backend

```

2597 <*dvipdfmx | xetex>

```

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

```

2598 \cs_new_protected:Npx \__pdf_backend:n #1
2599 { \__kernel_backend_literal:n { pdf: #1 } }
2600 \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

```

```

2601 \cs_new_protected:Npn \__pdf_backend_catalog_gput:nn #1#2
2602 { \__pdf_backend:n { put ~ @catalog << /#1 ~ #2 >> } }
2603 \cs_new_protected:Npn \__pdf_backend_info_gput:nn #1#2
2604 { \__pdf_backend:n { docinfo << /#1 ~ #2 >> } }

```

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

## 6.4.2 Objects

<code>\g__pdf_backend_object_int</code>	For tracking objects to allow finalisation.
<code>\g__pdf_backend_object_prop</code>	<pre> 2605 \int_new:N \g__pdf_backend_object_int 2606 \prop_new:N \g__pdf_backend_object_prop (End definition for \g__pdf_backend_object_int and \g__pdf_backend_object_prop.) </pre>
<code>\__pdf_backend_object_new:nn</code>	Objects are tracked at the macro level, but we don't have to do anything at this stage.
<code>\__pdf_backend_object_ref:n</code>	<pre> 2607 \cs_new_protected:Npn \__pdf_backend_object_new:nn #1#2 2608 { 2609   \int_gincr:N \g__pdf_backend_object_int 2610   \int_const:cn 2611     { c__pdf_backend_object_ \tl_to_str:n {#1} _int } 2612     { \g__pdf_backend_object_int } 2613   \prop_gput:Nnn \g__pdf_backend_object_prop {#1} {#2} 2614 } 2615 \cs_new:Npn \__pdf_backend_object_ref:n #1 2616 { @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.) </pre>
<code>\__pdf_backend_object_write:nn</code>	This is where we choose the actual type.
<code>\__pdf_backend_object_write:nx</code>	
<code>\__pdf_backend_object_write:nnn</code>	
<code>\__pdf_backend_object_write_array:nn</code>	
<code>\__pdf_backend_object_write_dict:nn</code>	
<code>\__pdf_backend_object_write_fstream:nn</code>	
<code>\__pdf_backend_object_write_stream:nn</code>	
<code>\__pdf_backend_object_write_stream:nnnn</code>	
	<pre> 2617 \cs_new_protected:Npn \__pdf_backend_object_write:nn #1#2 2618 { 2619   \exp_args:Nx \__pdf_backend_object_write:nnn 2620     { \prop_item:Nn \g__pdf_backend_object_prop {#1} } {#1} {#2} 2621 } 2622 \cs_generate_variant:Nn \__pdf_backend_object_write:nn { nx } 2623 \cs_new_protected:Npn \__pdf_backend_object_write:nnn #1#2#3 2624 { 2625   \use:c { __pdf_backend_object_write_ #1 :nn } 2626   { \__pdf_backend_object_ref:n {#2} } {#3} 2627 } 2628 \cs_new_protected:Npn \__pdf_backend_object_write_array:nn #1#2 2629 { 2630   \__pdf_backend:x 2631     { obj ~ #1 ~ [ ~ \exp_not:n {#2} ~ ] } 2632 } 2633 \cs_new_protected:Npn \__pdf_backend_object_write_dict:nn #1#2 2634 { 2635   \__pdf_backend:x 2636     { obj ~ #1 ~ &lt;&lt; ~ \exp_not:n {#2} ~ &gt;&gt; } 2637 } 2638 \cs_new_protected:Npn \__pdf_backend_object_write_fstream:nn #1#2 2639 { \__pdf_backend_object_write_stream:nnnn { f } {#1} #2 } 2640 \cs_new_protected:Npn \__pdf_backend_object_write_stream:nn #1#2 2641 { \__pdf_backend_object_write_stream:nnnn { } {#1} #2 } 2642 \cs_new_protected:Npn \__pdf_backend_object_write_stream:nnnn #1#2#3#4 2643 { 2644   \__pdf_backend:x 2645     { 2646       #1 stream ~ #2 ~ 2647       ( \exp_not:n {#4} ) ~ &lt;&lt; \exp_not:n {#3} &gt;&gt; 2648     } </pre>

```
2649 }
(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 2650 \cs_new_protected:Npn \_pdf_backend_object_now:nn #1#2
2651 {
2652   \int_gincr:N \g__pdf_backend_object_int
2653   \exp_args:Nnx \use:c { \_pdf_backend_object_write_ #1 :nn }
2654   { @pdf.obj \int_use:N \g__pdf_backend_object_int }
2655   {#2}
2656 }
2657 \cs_generate_variant:Nn \_pdf_backend_object_now:nn { nx }
(End definition for \_pdf_backend_object_now:nn.)
```

\\_pdf\_backend\_object\_last:

```
2658 \cs_new:Npn \_pdf_backend_object_last:
2659 { @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.

```
2660 \cs_new:Npn \_pdf_backend_pageobject_ref:n #1
2661 { @page #1 }
(End definition for \_pdf_backend_pageobject_ref:n.)
```

### 6.4.3 Annotations

\g\_\_pdf\_landscape\_bool There is a bug in dvipdfmx/X<sub>Y</sub>TeX which means annotations do not rotate. As such, we need to know if landscape is active.

```
2662 \bool_new:N \g__pdf_landscape_bool
2663 \cs_if_exist:NT \landscape
2664 {
2665   \tl_put_right:Nn \landscape
2666   { \bool_gset_true:N \g__pdf_landscape_bool }
2667   \tl_put_left:Nn \endlandscape
2668   { \bool_gset_false:N \g__pdf_landscape_bool }
2669 }
(End definition for \g__pdf_landscape_bool.)
```

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

```
2670 \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. The only wrinkle is landscape: we have to adjust by hand.

```
\_pdf_backend_annotation_aux:nnnn 2671 \cs_new_protected:Npn \_pdf_backend_annotation:nnnn #1#2#3#4
2672 {
2673   \bool_if:NTF \g__pdf_landscape_bool
2674   {
2675     \box_move_up:nn {#2}
```

```

2676         {
2677             \vbox:n
2678             {
2679                 \__pdf_backend_annotation_aux:nnnn
2680                 { #2 + #3 } {#1} { Opt } {#4}
2681             }
2682         }
2683     }
2684     { \__pdf_backend_annotation_aux:nnnn {#1} {#2} {#3} {#4} }
2685 }
2686 \cs_new_protected:Npn \__pdf_backend_annotation_aux:nnnn #1#2#3#4
2687 {
2688     \int_gincr:N \g__pdf_backend_object_int
2689     \int_gset_eq:NN \g__pdf_backend_annotation_int \g__pdf_backend_object_int
2690     \__pdf_backend:x
2691     {
2692         ann ~ @pdf.obj \int_use:N \g__pdf_backend_object_int \c_space_tl
2693         width ~ \dim_eval:n {#1} ~
2694         height ~ \dim_eval:n {#2} ~
2695         depth ~ \dim_eval:n {#3} ~
2696         <</Type/Annot #4 >>
2697     }
2698 }

```

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

\\_\_pdf\_backend\_annotation\_last:

```

2699 \cs_new:Npn \__pdf_backend_annotation_last:
2700 { @pdf.obj \int_use:N \g__pdf_backend_annotation_int }

```

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

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

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

\\_\_pdf\_backend\_link\_begin:n

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

All created using the same internals.

```

2701 \cs_new_protected:Npn \__pdf_backend_link_begin_goto:nnw #1#2
2702 { \__pdf_backend_link_begin:n { #1 /Subtype /Link /A << /S /GoTo /D ( #2 ) >> } }
2703 \cs_new_protected:Npn \__pdf_backend_link_begin_user:nnw #1#2
2704 { \__pdf_backend_link_begin:n {#1#2} }
2705 \cs_new_protected:Npn \__pdf_backend_link_begin:n #1
2706 {
2707     \__pdf_backend:n
2708     {
2709         bann
2710         <<
2711         /Type /Annot
2712         #1
2713         >>
2714     }
2715 }
2716 \cs_new_protected:Npn \__pdf_backend_link_end:
2717 { \__pdf_backend:n { eann } }

```

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

\\_\_pdf\_backend\_link\_last: Data not available.

```

2718 \cs_new:Npn \__pdf_backend_link_last: { }

```

(End definition for `\_pdf_backend_link_last:`.)

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

```
2719 \cs_new_protected:Npn \_pdf_backend_link_margin:n #1
2720 { \_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_box:nn`

Here, we need to turn the zoom into a scale. The method for `FitR` is from Alexander Grahn: the idea is to avoid needing to do any calculations in  $\text{T}_{\text{E}}\text{X}$  by using the backend data for `@xpos` and `@ypos`.

```
2721 \cs_new_protected:Npn \_pdf_backend_destination:nn #1#2
2722 {
2723   \_pdf_backend:x
2724   {
2725     dest ~ ( \exp_not:n {#1} )
2726     [
2727       @thispage
2728       \str_case:nnF {#2}
2729       {
2730         { xyz } { /XYZ ~ @xpos ~ @ypos ~ null }
2731         { fit } { /Fit }
2732         { fitb } { /FitB }
2733         { fitbh } { /FitBH }
2734         { fitbv } { /FitBV ~ @xpos }
2735         { fith } { /FitH ~ @ypos }
2736         { fitv } { /FitV ~ @xpos }
2737       }
2738       { /XYZ ~ @xpos ~ @ypos ~ \fp_eval:n { (#2) / 100 } }
2739     ]
2740   }
2741 }
2742 \cs_new_protected:Npn \_pdf_backend_destination_box:nn #1#2
2743 {
2744   \group_begin:
2745   \hbox_set:Nn \l__pdf_internal_box {#2}
2746   \box_move_down:nn { \box_dp:N \l__pdf_internal_box }
2747   {
2748     \hbox:n
2749     {
2750       \_pdf_backend:n { obj ~ @pdf_ #1 _llx ~ @xpos }
2751       \_pdf_backend:n { obj ~ @pdf_ #1 _lly ~ @ypos }
2752     }
2753   }
2754   \box_use:N \l__pdf_internal_box
2755   \box_move_up:nn { \box_ht:N \l__pdf_internal_box }
2756   {
2757     \hbox:n
2758     {
2759       \_pdf_backend:n
2760       {
2761         dest ~ (#1)
2762         [
2763           @thispage
```

```

2764             /FitR ~
2765             @pdf_ #1 _llx ~ @pdf_ #1 _lly ~
2766             @xpos ~ @ypos
2767         ]
2768     }
2769 }
2770 }
2771 \group_end:
2772 }

```

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

#### 6.4.4 Structure

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

```

\_pdf_backend_compress_objects:n
2773 \cs_new_protected:Npn \_pdf_backend_compresslevel:n #1
2774 { \__kernel_backend_literal:x { dvipdfmx:config~z~ \int_eval:n {#1} } }
2775 \cs_new_protected:Npn \_pdf_backend_compress_objects:n #1
2776 {
2777     \bool_if:nF {#1}
2778     { \__kernel_backend_literal:n { dvipdfmx:config~C~0x40 } }
2779 }

```

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

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

```

\_pdf_backend_version_minor_gset:n
2780 \cs_new_protected:Npn \_pdf_backend_version_major_gset:n #1
2781 {
2782     \cs_gset:Npx \_pdf_backend_version_major: { \int_eval:n {#1} }
2783     \__kernel_backend_literal:x { pdf:majorversion~ \_pdf_backend_version_major: }
2784 }
2785 \cs_new_protected:Npn \_pdf_backend_version_minor_gset:n #1
2786 {
2787     \cs_gset:Npx \_pdf_backend_version_minor: { \int_eval:n {#1} }
2788     \__kernel_backend_literal:x { pdf:minorversion~ \_pdf_backend_version_minor: }
2789 }

```

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

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

```

\_pdf_backend_version_minor:
2790 \cs_new:Npn \_pdf_backend_version_major: { 1 }
2791 \cs_new:Npn \_pdf_backend_version_minor: { 5 }

```

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

#### 6.4.5 Marked content

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

```

2792 \cs_new_protected:Npn \_pdf_backend_bdc:nn #1#2
2793 { \__kernel_backend_literal_page:n { /#1 ~ #2 ~ BDC } }
2794 \cs_new_protected:Npn \_pdf_backend_emc:
2795 { \__kernel_backend_literal_page:n { EMC } }

```

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

```

2796 </dvipdfmx | xetex>

```



## 6.5 dvisvgm backend

2797  $\langle *dvisvgm \rangle$

### 6.5.1 Catalogue entries

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

No-op.

2798  $\backslash cs\_new\_protected:Npn \backslash\_pdf\_backend\_catalog\_gput:nn \#1\#2 \{ \}$   
2799  $\backslash cs\_new\_protected:Npn \backslash\_pdf\_backend\_info\_gput:nn \#1\#2 \{ \}$

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

### 6.5.2 Objects

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

All no-ops here.

2800  $\backslash cs\_new\_protected:Npn \backslash\_pdf\_backend\_object\_new:nn \#1\#2 \{ \}$   
2801  $\backslash cs\_new:Npn \backslash\_pdf\_backend\_object\_ref:n \#1 \{ \}$   
2802  $\backslash cs\_new\_protected:Npn \backslash\_pdf\_backend\_object\_write:nn \#1\#2 \{ \}$   
2803  $\backslash cs\_new\_protected:Npn \backslash\_pdf\_backend\_object\_write:nx \#1\#2 \{ \}$   
2804  $\backslash cs\_new\_protected:Npn \backslash\_pdf\_backend\_object\_now:nn \#1\#2 \{ \}$   
2805  $\backslash cs\_new\_protected:Npn \backslash\_pdf\_backend\_object\_now:nx \#1\#2 \{ \}$   
2806  $\backslash cs\_new:Npn \backslash\_pdf\_backend\_object\_last: \{ \}$   
2807  $\backslash cs\_new:Npn \backslash\_pdf\_backend\_pageobject\_ref:n \#1 \{ \}$

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

### 6.5.3 Structure

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

These are all no-ops.

2808  $\backslash cs\_new\_protected:Npn \backslash\_pdf\_backend\_compresslevel:n \#1 \{ \}$   
2809  $\backslash cs\_new\_protected:Npn \backslash\_pdf\_backend\_compress\_objects:n \#1 \{ \}$

(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!

2810  $\backslash cs\_new\_protected:Npn \backslash\_pdf\_backend\_version\_major\_gset:n \#1 \{ \}$   
2811  $\backslash cs\_new\_protected:Npn \backslash\_pdf\_backend\_version\_minor\_gset:n \#1 \{ \}$

(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!

2812  $\backslash cs\_new:Npn \backslash\_pdf\_backend\_version\_major: \{ -1 \}$   
2813  $\backslash cs\_new:Npn \backslash\_pdf\_backend\_version\_minor: \{ -1 \}$

(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.

2814  $\backslash cs\_new\_protected:Npn \backslash\_pdf\_backend\_bdc:nn \#1\#2 \{ \}$   
2815  $\backslash cs\_new\_protected:Npn \backslash\_pdf\_backend\_emc: \{ \}$

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

2816  $\langle /dvisvgm \rangle$

2817  $\langle /package \rangle$

## 7 l3backend-header Implementation

	2818 <code>&lt;*dvips &amp; header&gt;</code>	
<code>color.sc</code>	Empty definitions for color at the top level.	
<code>color.fc</code>	2819 <code>/color.sc { } def</code> 2820 <code>/color.fc { } def</code>	
	<i>(End definition for color.sc and color.fc. These functions are documented on page ??.)</i>	
<code>TeXcolorseparation</code>	Support for separation/spot colors: this strange naming is so things work with the color	
<code>separation</code>	stack.	
	2821 <code>TeXDict begin</code> 2822 <code>/TeXcolorseparation { setcolor } def</code> 2823 <code>end</code>	
	<i>(End definition for TeXcolorseparation and separation. These functions are documented on page ??.)</i>	
<code>pdf.globaldict</code>	A small global dictionary for backend use.	
	2824 <code>true setglobal</code> 2825 <code>/pdf.globaldict 4 dict def</code> 2826 <code>false setglobal</code>	
	<i>(End definition for pdf.globaldict. This function is documented on page ??.)</i>	
<code>pdf.cvs</code>	Small utilities for PostScript manipulations. Conversion to DVI dimensions is done here	
<code>pdf.dvi.pt</code>	to allow for Resolution. The total height of a rectangle (an array) needs a little maths,	
<code>pdf.pt.dvi</code>	in contrast to simply extracting a value.	
<code>pdf.rect.ht</code>	2827 <code>/pdf.cvs { 65534 string cvs } def</code> 2828 <code>/pdf.dvi.pt { 72.27 mul Resolution div } def</code> 2829 <code>/pdf.pt.dvi { 72.27 div Resolution mul } def</code> 2830 <code>/pdf.rect.ht { dup 1 get neg exch 3 get add } def</code>	
	<i>(End definition for pdf.cvs and others. These functions are documented on page ??.)</i>	
<code>pdf.linkmargin</code>	Settings which are defined up-front in SDict.	
<code>pdf.linkdp.pad</code>	2831 <code>/pdf.linkmargin { 1 pdf.pt.dvi } def</code>	
<code>pdf.linkht.pad</code>	2832 <code>/pdf.linkdp.pad { 0 } def</code> 2833 <code>/pdf.linkht.pad { 0 } def</code>	
	<i>(End definition for pdf.linkmargin, pdf.linkdp.pad, and pdf.linkht.pad. These functions are documented on page ??.)</i>	
<code>pdf.rect</code>	Functions for marking the limits of an annotation/link, plus drawing the border. We	
<code>pdf.save.ll</code>	separate links for generic annotations to support adding a margin and setting a minimal	
<code>pdf.save.ur</code>	size.	
<code>pdf.save.linkll</code>	2834 <code>/pdf.rect</code>	
<code>pdf.save.linkur</code>	2835 <code>{ /Rect [ pdf.llx pdf.lly pdf.urx pdf.ury ] } def</code>	
<code>pdf.llx</code>	2836 <code>/pdf.save.ll</code>	
<code>pdf.lly</code>	2837 <code>{</code>	
<code>pdf.urx</code>	2838 <code>currentpoint</code>	
<code>pdf.ury</code>	2839 <code>/pdf.lly exch def</code>	
	2840 <code>/pdf.llx exch def</code>	
	2841 <code>}</code>	
	2842 <code>def</code>	

```

2843 /pdf.save.ur
2844 {
2845     currentpoint
2846     /pdf.ury exch def
2847     /pdf.urx exch def
2848 }
2849 def
2850 /pdf.save.linkll
2851 {
2852     currentpoint
2853     pdf.linkmargin add
2854     pdf.linkdp.pad add
2855     /pdf.lly exch def
2856     pdf.linkmargin sub
2857     /pdf.llx exch def
2858 }
2859 def
2860 /pdf.save.linkur
2861 {
2862     currentpoint
2863     pdf.linkmargin sub
2864     pdf.linkht.pad sub
2865     /pdf.ury exch def
2866     pdf.linkmargin add
2867     /pdf.urx exch def
2868 }
2869 def

```

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

<p>pdf.dest.anchor</p> <p>pdf.dest.x</p> <p>pdf.dest.y</p> <p>pdf.dest.point</p> <p>pdf.dest2device</p> <p>pdf.dev.x</p> <p>pdf.dev.y</p> <p>pdf.tmpa</p> <p>pdf.tmpb</p> <p>pdf.tmpc</p> <p>pdf.tmpd</p>	<p>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.)</p> <pre> 2870 /pdf.dest.anchor 2871 { 2872     currentpoint exch 2873     pdf.dvi.pt 72 add 2874     /pdf.dest.x exch def 2875     pdf.dvi.pt 2876     vsize 72 sub exch sub 2877     /pdf.dest.y exch def 2878 } 2879 def 2880 /pdf.dest.point 2881 { pdf.dest.x pdf.dest.y } def 2882 /pdf.dest2device 2883 { 2884     /pdf.dest.y exch def 2885     /pdf.dest.x exch def 2886     matrix currentmatrix 2887     matrix defaultmatrix 2888     matrix invertmatrix </pre>
---	---

```

2889     matrix concatmatrix
2890     cvx exec
2891     /pdf.dev.y exch def
2892     /pdf.dev.x exch def
2893     /pdf.tmpd exch def
2894     /pdf.tmpc exch def
2895     /pdf.tmpb exch def
2896     /pdf.tmpa exch def
2897     pdf.dest.x pdf.tmpa mul
2898         pdf.dest.y pdf.tmpc mul add
2899         pdf.dev.x add
2900     pdf.dest.x pdf.tmpb mul
2901         pdf.dest.y pdf.tmpd mul add
2902         pdf.dev.y add
2903 }
2904 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	2905 /pdf.bordertracking false def
pdf.brokenlink.skip	2906 /pdf.bordertracking.begin
pdf.brokenlink.dict	2907 {
pdf.bordertracking.endpage	2908 SDict /pdf.bordertracking true put
pdf.bordertracking.continue	2909 SDict /pdf.leftboundary undef
pdf.originx	2910 SDict /pdf.rightboundary undef
pdf.originy	2911 /a where
	2912 {
	2913 /a
	2914 {
	2915 currentpoint pop
	2916 SDict /pdf.rightboundary known dup
	2917 {
	2918 SDict /pdf.rightboundary get 2 index lt
	2919 { not }
	2920 if
	2921 }
	2922 if
	2923 { pop }
	2924 { SDict exch /pdf.rightboundary exch put }
	2925 ifelse
	2926 moveto
	2927 currentpoint pop
	2928 SDict /pdf.leftboundary known dup
	2929 {
	2930 SDict /pdf.leftboundary get 2 index gt
	2931 { not }
	2932 if
	2933 }
	2934 if
	2935 { pop }

```

2936         { SDict exch /pdf.leftboundary exch put }
2937     ifelse
2938 }
2939 put
2940 }
2941 if
2942 }
2943 def
2944 /pdf.bordertracking.end
2945 {
2946     /a where { /a { moveto } put } if
2947     /x where { /x { 0 exch rmoveto } put } if
2948     SDict /pdf.leftboundary known
2949     { pdf.outerbox 0 pdf.leftboundary put }
2950     if
2951     SDict /pdf.rightboundary known
2952     { pdf.outerbox 2 pdf.rightboundary put }
2953     if
2954     SDict /pdf.bordertracking false put
2955 }
2956 def
2957 /pdf.bordertracking.endpage
2958 {
2959     pdf.bordertracking
2960     {
2961         pdf.bordertracking.end
2962         true setglobal
2963         pdf.globaldict
2964         /pdf.brokenlink.rect [ pdf.outerbox aload pop ] put
2965         pdf.globaldict
2966         /pdf.brokenlink.skip pdf.baselineskip put
2967         pdf.globaldict
2968         /pdf.brokenlink.dict
2969         pdf.link.dict pdf.cvs put
2970         false setglobal
2971         mark pdf.link.dict cvx exec /Rect
2972         [
2973             pdf.llx
2974             pdf.lly
2975             pdf.outerbox 2 get pdf.linkmargin add
2976             currentpoint exch pop
2977             pdf.outerbox pdf.rect.ht sub pdf.linkmargin sub
2978         ]
2979         /ANN pdf.pdfmark
2980     }
2981     if
2982 }
2983 def
2984 /pdf.bordertracking.continue
2985 {
2986     /pdf.link.dict pdf.globaldict
2987     /pdf.brokenlink.dict get def
2988     /pdf.outerbox pdf.globaldict
2989     /pdf.brokenlink.rect get def

```

```

2990 /pdf.baselineskip pdf.globaldict
2991 /pdf.brokenlink.skip get def
2992 pdf.globaldict dup dup
2993 /pdf.brokenlink.dict undef
2994 /pdf.brokenlink.skip undef
2995 /pdf.brokenlink.rect undef
2996 currentpoint
2997 /pdf.originy exch def
2998 /pdf.originx exch def
2999 /a where
3000 {
3001 /a
3002 {
3003 moveto
3004 SDict
3005 begin
3006 currentpoint pdf.originy ne exch
3007 pdf.originx ne or
3008 {
3009 pdf.save.linkll
3010 /pdf.lly
3011 pdf.lly pdf.outerbox 1 get sub def
3012 pdf.bordertracking.begin
3013 }
3014 if
3015 end
3016 }
3017 put
3018 }
3019 if
3020 /x where
3021 {
3022 /x
3023 {
3024 0 exch rmoveto
3025 SDict
3026 begin
3027 currentpoint
3028 pdf.originy ne exch pdf.originx ne or
3029 {
3030 pdf.save.linkll
3031 /pdf.lly
3032 pdf.lly pdf.outerbox 1 get sub def
3033 pdf.bordertracking.begin
3034 }
3035 if
3036 end
3037 }
3038 put
3039 }
3040 if
3041 }
3042 def

```

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

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

```

3043 /pdf.breaklink
3044 {
3045     pop
3046     counttomark 2 mod 0 eq
3047     {
3048         counttomark /pdf.count exch def
3049         {
3050             pdf.count 0 eq { exit } if
3051             counttomark 2 roll
3052             1 index /Rect eq
3053             {
3054                 dup 4 array copy
3055                 dup dup
3056                 1 get
3057                 pdf.outerbox pdf.rect.ht
3058                 pdf.linkmargin 2 mul add sub
3059                 3 exch put
3060                 dup
3061                 pdf.outerbox 2 get
3062                 pdf.linkmargin add
3063                 2 exch put
3064                 dup dup
3065                 3 get
3066                 pdf.outerbox pdf.rect.ht
3067                 pdf.linkmargin 2 mul add add
3068                 1 exch put
3069                 /pdf.currentrect exch def
3070                 pdf.breaklink.write
3071                 {
3072                     pdf.currentrect
3073                     dup
3074                     pdf.outerbox 0 get
3075                     pdf.linkmargin sub
3076                     0 exch put
3077                     dup
3078                     pdf.outerbox 2 get
3079                     pdf.linkmargin add
3080                     2 exch put
3081                     dup dup
3082                     1 get
3083                     pdf.baselineskip add
3084                     1 exch put
3085                     dup dup
3086                     3 get
3087                     pdf.baselineskip add
3088                     3 exch put
3089                     /pdf.currentrect exch def
3090                     pdf.breaklink.write

```

```

3091         }
3092         1 index 3 get
3093         pdf.linkmargin 2 mul add
3094         pdf.outerbox pdf.rect.ht add
3095         2 index 1 get sub
3096         pdf.baselineskip div round cvi 1 sub
3097         exch
3098         repeat
3099         pdf.currentrect
3100         dup
3101         pdf.outerbox 0 get
3102         pdf.linkmargin sub
3103         0 exch put
3104         dup dup
3105         1 get
3106         pdf.baselineskip add
3107         1 exch put
3108         dup dup
3109         3 get
3110         pdf.baselineskip add
3111         3 exch put
3112         dup 2 index 2 get 2 exch put
3113         /pdf.currentrect exch def
3114         pdf.breaklink.write
3115         SDict /pdf.pdfmark.good false put
3116         exit
3117     }
3118     { pdf.count 2 sub /pdf.count exch def }
3119 ifelse
3120 }
3121 loop
3122 }
3123 if
3124 /ANN
3125 }
3126 def
3127 /pdf.breaklink.write
3128 {
3129     counttomark 1 sub
3130     index /_objdef eq
3131     {
3132         counttomark -2 roll
3133         dup wcheck
3134         {
3135             readonly
3136             counttomark 2 roll
3137         }
3138         { pop pop }
3139     ifelse
3140 }
3141 if
3142 counttomark 1 add copy
3143 pop pdf.currentrect
3144 /ANN pdfmark

```



```

3145 }
3146 def

```

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

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

```

3147 /pdf.pdfmark
3148 {
3149   SDict /pdf.pdfmark.good true put
3150   dup /ANN eq
3151   {
3152     pdf.pdfmark.store
3153     pdf.pdfmark.dict
3154     begin
3155       Subtype /Link eq
3156       currentdict /Rect known and
3157       SDict /pdf.outerbox known and
3158       SDict /pdf.baselineskip known and
3159       {
3160         Rect 3 get
3161         pdf.linkmargin 2 mul add
3162         pdf.outerbox pdf.rect.ht add
3163         Rect 1 get sub
3164         pdf.baselineskip div round cvi 0 gt
3165         { pdf.breaklink }
3166         if
3167       }
3168       if
3169       end
3170       SDict /pdf.outerbox undef
3171       SDict /pdf.baselineskip undef
3172       currentdict /pdf.pdfmark.dict undef
3173     }
3174     if
3175     pdf.pdfmark.good
3176     { pdfmark }
3177     { cleartomark }
3178     ifelse
3179   }
3180   def
3181 /pdf.pdfmark.store
3182 {
3183   /pdf.pdfmark.dict 65534 dict def
3184   counttomark 1 add copy
3185   pop
3186   {
3187     dup mark eq
3188     {
3189       pop
3190       exit

```

```

3191         }
3192         {
3193             pdf.pdfmark.dict
3194             begin def end
3195         }
3196     ifelse
3197 }
3198 loop
3199 }
3200 def

```

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

```

3201 </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> . . . . .	59, 60, 554, 555
<b>B</b>	
bool commands:	
<code>\bool_gset_false:N</code> . . . . .	1026, 1045, 1071, 1094, 1110, 1211, 1448, 1484, 2049, 2095, 2668
<code>\bool_gset_true:N</code> . . . . .	1024, 1097, 1209, 1463, 2042, 2048, 2666
<code>\bool_if:NTF</code> . . . . .	57, 552, 1036, 1040, 1058, 1062, 1066, 1080, 1085, 1089, 1101, 1105, 1222, 1227, 1232, 1422, 1467, 1580, 1615, 1725, 1767, 2037, 2052, 2057, 2062, 2673
<code>\bool_if:nTF</code> . . . . .	2263, 2526, 2777
<code>\bool_lazy_or:nnTF</code> . . . . .	1607, 1760
<code>\bool_new:N</code> . . . . .	1027, 1098, 1212, 1464, 2025, 2026, 2662
<code>\bool_set_false:N</code> . . . . .	1590, 1692, 1785, 1849
box commands:	
<code>\box_dp:N</code> . . . . .	200, 202, 250, 252, 307, 309, 356, 358, 360, 362, 2074, 2107, 2108, 2133, 2230, 2388, 2746
<code>\box_ht:N</code> . . . . .	202, 252, 309, 360, 362, 1627, 1822, 2079, 2118, 2119, 2135, 2234, 2387, 2755
<code>\box_if_empty:N</code> . . . . .	2169
<code>\box_move_down:nn</code> . . . . .	1999, 2074, 2229, 2746
<code>\box_move_up:nn</code> . . . . .	2001, 2079, 2233, 2675, 2755
<code>\box_new:N</code> . . . . .	1884, 1989, 1990
<code>\box_set_dp:Nn</code> . . . . .	1547
<code>\box_set_ht:Nn</code> . . . . .	1546
<code>\box_set_wd:Nn</code> . . . . .	264, 1545
<code>\box_use:N</code> . . . . .	207, 225, 239, 255, 282, 296, 312, 328, 340, 391, 405, 424, 1162, 1357, 1548, 2030, 2232, 2389, 2754
<code>\box_wd:N</code> . . . . .	201, 209, 251, 257, 308, 314, 357, 359, 1626, 1821, 2386
box internal commands:	
<code>\__box_backend_clip:N</code> . . . . .	189, 244, 301, 345
<code>\l__box_backend_cos_fp</code> . . . . .	259
<code>\__box_backend_rotate:Nn</code> . . . . .	211, 259, 316, 395
<code>\__box_backend_rotate_aux:Nn</code> . . . . .	211, 259, 316
<code>\__box_backend_scale:Nnn</code> . . . . .	228, 287, 331, 408
<code>\l__box_backend_sin_fp</code> . . . . .	259
<code>\g__box_clip_path_int</code> . . . . .	345
<b>C</b>	
clist commands:	
<code>\clist_map_function:nN</code> . . . . .	1118, 1242
<code>\clist_map_function:nn</code> . . . . .	1491
color internal commands:	
<code>\__color_backend:nnn</code> . . . . .	931
<code>\__color_backend:nnnn</code> . . . . .	901
<code>\__color_backend_cmyk:nw</code> . . . . .	901
<code>\__color_backend_devicen_init:n</code> . . . . .	809
<code>\__color_backend_devicen_-init:nnn</code> . . . . .	700, 809
<code>\__color_backend_devicen_init:w</code> . . . . .	809
<code>\__color_backend_fill_cmyk:n</code> . . . . .	861, 881, 901
<code>\__color_backend_fill_devicen:nn</code> . . . . .	873, 893, 963
<code>\__color_backend_fill_gray:n</code> . . . . .	861, 881, 901
<code>\__color_backend_fill_rgb:n</code> . . . . .	861, 881, 901
<code>\__color_backend_fill_separation:nn</code> . . . . .	873, 893, 963
<code>\__color_backend_grab:nn</code> . . . . .	918, 920
<code>\__color_backend_gray:nn</code> . . . . .	901
<code>\__color_backend_gray_aux:n</code> . . . . .	901
<code>\__color_backend_gray_aux:nn</code> . . . . .	925, 930
<code>\__color_backend_pickup:N</code> . . . . .	443, 485
<code>\__color_backend_pickup:w</code> . . . . .	15, 443, 485
<code>\__color_backend_reset:</code> . . . . .	432, 466, 518, 737
<code>\__color_backend_rgb:nw</code> . . . . .	901
<code>\__color_backend_select:n</code> . . . . .	466, 519, 521, 523, 524, 548, 726
<code>\__color_backend_select_cmyk:n</code> . . . . .	432, 466, 518
<code>\__color_backend_select_devicen:nn</code> . . . . .	547, 720, 726
<code>\__color_backend_select_gray:n</code> . . . . .	432, 466, 518

\__color_backend_select_rgb:n ...	1366, 1739, 1796, 1812, 1888, 1925,
.....	1984, 2462, 2490, 2600, 2622, 2657
\__color_backend_select_separation:nn	\cs_gset:Npx ..... 2782, 2787
.....	\cs_gset_protected:Npn . 768, 806, 851
\__color_backend_separation-	\cs_if_exist:NTF .....
init:n ..... 550, 741, 833, 858	..... 27, 59, 444, 486, 554,
\__color_backend_separation-	767, 804, 850, 2165, 2551, 2577, 2663
init:nnn ..... 550	\cs_if_exist_use:NTF ..... 38, 576
\__color_backend_separation-	\cs_new:Npn .... 585, 587, 589, 591,
init:nnnn ..... 550	598, 604, 606, 612, 629, 636, 638,
\__color_backend_separation-	852, 1123, 1247, 1495, 1825, 1834,
init:nnnnn ..... 550, 722, 741	1878, 1903, 1985, 1987, 2020, 2191,
\__color_backend_separation-	2275, 2276, 2432, 2463, 2464, 2582,
init:nw ..... 550	2615, 2658, 2660, 2699, 2718, 2790,
\__color_backend_separation-	2791, 2801, 2806, 2807, 2812, 2813
init:w ..... 550	\cs_new:Npx 2296, 2331, 2491, 2502, 2569
\__color_backend_separation-	\cs_new_eq:NN 46, 437, 438, 549, 740,
init_/DeviceCMYK:nnn ..... 550	858, 877, 878, 897, 898, 965, 966,
\__color_backend_separation-	972, 1171, 1177, 1178, 1365, 1372,
init_/DeviceGray:nnn ..... 550	1557, 1586, 1637, 1638, 1680, 1688,
\__color_backend_separation-	1710, 1781, 1838, 1845, 1877, 2030
init_/DeviceRGB:nnn ..... 550	\cs_new_protected:Npn ..... 47,
\__color_backend_separation-	51, 54, 64, 70, 75, 77, 81, 92, 102,
init_aux:nnnnn ..... 550	111, 120, 133, 136, 138, 140, 144,
\__color_backend_separation-	149, 158, 168, 178, 189, 211, 213,
init_CIELAB:nnn .... 550, 722, 741	228, 244, 259, 261, 287, 301, 316,
\__color_backend_separation-	318, 331, 345, 395, 408, 432, 439,
init_CIELAB:nnnnnn ..... 723	443, 461, 466, 468, 470, 472, 481,
\__color_backend_separation-	485, 493, 518, 520, 522, 524, 535,
init_count:n ..... 550	547, 564, 654, 700, 720, 721, 722,
\__color_backend_separation-	723, 726, 734, 741, 769, 782, 809,
init_count:w ..... 550	861, 863, 865, 867, 869, 871, 873,
\__color_backend_separation-	875, 881, 883, 885, 887, 889, 891,
init_Device:Nn ..... 550	893, 895, 901, 903, 905, 917, 919,
\__color_backend_stroke_cmyk:n ..	921, 930, 932, 934, 936, 963, 964,
..... 861, 881, 901	974, 979, 984, 986, 988, 996, 1004,
\__color_backend_stroke_devicen:nn	1013, 1023, 1025, 1028, 1030, 1047,
..... 873, 893, 963	1052, 1073, 1096, 1099, 1112, 1125,
\__color_backend_stroke_gray:n ..	1130, 1132, 1134, 1136, 1138, 1140,
..... 861, 881, 901	1142, 1144, 1149, 1173, 1175, 1179,
\__color_backend_stroke_rgb:n ...	1184, 1189, 1199, 1208, 1210, 1213,
..... 861, 881, 901	1215, 1217, 1219, 1224, 1229, 1234,
\__color_backend_stroke_separation:nn	1236, 1249, 1254, 1256, 1258, 1260,
..... 873, 893, 963	1262, 1264, 1266, 1268, 1279, 1304,
\g__color_model_int .....	1316, 1328, 1340, 1347, 1367, 1373,
..... 570, 706, 762, 774, 845	1378, 1383, 1394, 1404, 1414, 1416,
\c__color_model_range_CIELAB_t1 .	1418, 1420, 1451, 1453, 1458, 1460,
..... 661, 696, 793, 800	1462, 1465, 1486, 1497, 1510, 1512,
color.fc ..... 432, 466, 2819	1514, 1516, 1518, 1520, 1522, 1524,
color.sc ..... 432, 466, 2819	1526, 1534, 1558, 1572, 1587, 1599,
cs commands:	1604, 1632, 1644, 1657, 1667, 1682,
\cs_generate_variant:Nn .....	1689, 1697, 1708, 1712, 1715, 1730,
..... 49, 53, 56, 91, 130,	1740, 1775, 1782, 1788, 1794, 1797,
135, 146, 177, 183, 563, 973, 1172,	1804, 1813, 1818, 1826, 1839, 1846,

1852, 1854, 1856, 1867, 1886, 1889,	\_draw_backend_clip: <a href="#">1028</a> , <a href="#">1213</a> , <a href="#">1418</a>
1891, 1895, 1905, 1926, 1931, 1936,	\_draw_backend_closepath: . . . . .
1941, 1951, 1956, 1964, 1992, 1997,	<a href="#">1028</a> , <a href="#">1213</a> , <a href="#">1418</a>
2029, 2031, 2033, 2035, 2040, 2055,	\_draw_backend_closestroke: . . .
2060, 2097, 2126, 2145, 2154, 2193,	<a href="#">1028</a> , <a href="#">1213</a> , <a href="#">1418</a>
2200, 2225, 2249, 2261, 2273, 2274,	\_draw_backend_cm:nnnn <a href="#">1144</a> , <a href="#">1157</a> ,
2277, 2279, 2283, 2307, 2309, 2311,	<a href="#">1158</a> , <a href="#">1159</a> , <a href="#">1268</a> , <a href="#">1351</a> , <a href="#">1526</a> , <a href="#">1537</a>
2322, 2342, 2352, 2374, 2392, 2402,	\_draw_backend_cm_aux:nnnn . . <a href="#">1268</a>
2413, 2434, 2465, 2513, 2524, 2530,	\_draw_backend_cm_decompose:nnnnN
2558, 2592, 2594, 2601, 2603, 2607,	<a href="#">1274</a> , <a href="#">1303</a>
2617, 2623, 2628, 2633, 2638, 2640,	\_draw_backend_cm_decompose_-
2642, 2650, 2671, 2686, 2701, 2703,	auxi:nnnnN . . . . . <a href="#">1303</a>
2705, 2716, 2719, 2721, 2742, 2773,	\_draw_backend_cm_decompose_-
2775, 2780, 2785, 2792, 2794, 2798,	auxii:nnnnN . . . . . <a href="#">1303</a>
2799, 2800, 2802, 2803, 2804, 2805,	\_draw_backend_cm_decompose_-
2808, 2809, 2810, 2811, 2814, 2815	auxiii:nnnnN . . . . . <a href="#">1303</a>
\cs_new_protected:Npx . . . . .	\_draw_backend_curveto:nnnnnn . .
<a href="#">550</a> , <a href="#">948</a> , <a href="#">2541</a> , <a href="#">2598</a>	<a href="#">988</a> , <a href="#">1179</a> , <a href="#">1373</a>
\cs_set_eq:NN . . . . . <a href="#">2186</a> , <a href="#">2187</a>	\_draw_backend_dash:n . . . . .
\cs_set_protected:Npn . . . . . <a href="#">446</a> , <a href="#">488</a>	<a href="#">1112</a> , <a href="#">1236</a> , <a href="#">1486</a>
	\_draw_backend_dash_aux:nn . . <a href="#">1486</a>
	\_draw_backend_dash_pattern:nn .
	<a href="#">1112</a> , <a href="#">1236</a> , <a href="#">1486</a>
	\_draw_backend_discardpath: . . .
	<a href="#">1028</a> , <a href="#">1213</a> , <a href="#">1418</a>
	\_draw_backend_end: . <a href="#">974</a> , <a href="#">1173</a> , <a href="#">1367</a>
	\_draw_backend_evenodd_rule: . . .
	<a href="#">1023</a> , <a href="#">1208</a> , <a href="#">1414</a>
	\_draw_backend_fill: <a href="#">1028</a> , <a href="#">1213</a> , <a href="#">1418</a>
	\_draw_backend_fillstroke: . . . .
	<a href="#">1028</a> , <a href="#">1213</a> , <a href="#">1418</a>
	\_draw_backend_join_bevel: . . . .
	<a href="#">1112</a> , <a href="#">1236</a> , <a href="#">1486</a>
	\_draw_backend_join_miter: . . . .
	<a href="#">1112</a> , <a href="#">1236</a> , <a href="#">1486</a>
	\_draw_backend_join_round: . . . .
	<a href="#">1112</a> , <a href="#">1236</a> , <a href="#">1486</a>
	\_draw_backend_lineto:nn . . . . .
	<a href="#">988</a> , <a href="#">1179</a> , <a href="#">1373</a>
	\_draw_backend_linewidth:n . . . .
	<a href="#">1112</a> , <a href="#">1236</a> , <a href="#">1486</a>
	\_draw_backend_literal:n . . <a href="#">972</a> ,
	<a href="#">977</a> , <a href="#">981</a> , <a href="#">985</a> , <a href="#">987</a> , <a href="#">990</a> , <a href="#">998</a> , <a href="#">1006</a> ,
	<a href="#">1015</a> , <a href="#">1029</a> , <a href="#">1032</a> , <a href="#">1033</a> , <a href="#">1034</a> , <a href="#">1035</a> ,
	<a href="#">1038</a> , <a href="#">1044</a> , <a href="#">1054</a> , <a href="#">1055</a> , <a href="#">1056</a> , <a href="#">1061</a> ,
	<a href="#">1064</a> , <a href="#">1070</a> , <a href="#">1075</a> , <a href="#">1076</a> , <a href="#">1077</a> , <a href="#">1078</a> ,
	<a href="#">1083</a> , <a href="#">1084</a> , <a href="#">1087</a> , <a href="#">1093</a> , <a href="#">1103</a> , <a href="#">1109</a> ,
	<a href="#">1114</a> , <a href="#">1127</a> , <a href="#">1131</a> , <a href="#">1133</a> , <a href="#">1135</a> , <a href="#">1137</a> ,
	<a href="#">1139</a> , <a href="#">1141</a> , <a href="#">1143</a> , <a href="#">1146</a> , <a href="#">1151</a> , <a href="#">1152</a> ,
	<a href="#">1153</a> , <a href="#">1154</a> , <a href="#">1155</a> , <a href="#">1156</a> , <a href="#">1160</a> , <a href="#">1161</a> ,
	<a href="#">1163</a> , <a href="#">1164</a> , <a href="#">1165</a> , <a href="#">1166</a> , <a href="#">1167</a> , <a href="#">1171</a> ,
	<a href="#">1181</a> , <a href="#">1186</a> , <a href="#">1191</a> , <a href="#">1201</a> , <a href="#">1214</a> , <a href="#">1216</a> ,
	<a href="#">1218</a> , <a href="#">1221</a> , <a href="#">1226</a> , <a href="#">1231</a> , <a href="#">1235</a> , <a href="#">1238</a> ,

**D**

dim commands:

\dim_eval:n . . . . . <a href="#">1995</a> , <a href="#">2291</a> , <a href="#">2292</a> ,
<a href="#">2293</a> , <a href="#">2350</a> , <a href="#">2693</a> , <a href="#">2694</a> , <a href="#">2695</a> , <a href="#">2720</a>
\dim_max:nn . . . . . <a href="#">2105</a> , <a href="#">2116</a>
\dim_set:Nn . . . . . <a href="#">1626</a> , <a href="#">1627</a> , <a href="#">1821</a> , <a href="#">1822</a>
\dim_to_decimal:n . . . . . <a href="#">356</a> , <a href="#">357</a> , <a href="#">358</a> ,
<a href="#">359</a> , <a href="#">360</a> , <a href="#">362</a> , <a href="#">1376</a> , <a href="#">1381</a> , <a href="#">1387</a> ,
<a href="#">1388</a> , <a href="#">1389</a> , <a href="#">1390</a> , <a href="#">1399</a> , <a href="#">1400</a> , <a href="#">1401</a> ,
<a href="#">1492</a> , <a href="#">1511</a> , <a href="#">1872</a> , <a href="#">1873</a> , <a href="#">2103</a> , <a href="#">2114</a> ,
<a href="#">2132</a> , <a href="#">2133</a> , <a href="#">2134</a> , <a href="#">2135</a> , <a href="#">2139</a> , <a href="#">2197</a>
\dim_to_decimal_in_bp:n . . . . .
. . . . . <a href="#">200</a> , <a href="#">201</a> , <a href="#">202</a> , <a href="#">250</a> , <a href="#">251</a> ,
<a href="#">252</a> , <a href="#">307</a> , <a href="#">308</a> , <a href="#">309</a> , <a href="#">992</a> , <a href="#">993</a> , <a href="#">1000</a> ,
<a href="#">1001</a> , <a href="#">1008</a> , <a href="#">1009</a> , <a href="#">1017</a> , <a href="#">1018</a> , <a href="#">1019</a> ,
<a href="#">1120</a> , <a href="#">1124</a> , <a href="#">1128</a> , <a href="#">1182</a> , <a href="#">1187</a> , <a href="#">1193</a> ,
<a href="#">1194</a> , <a href="#">1195</a> , <a href="#">1203</a> , <a href="#">1204</a> , <a href="#">1244</a> , <a href="#">1248</a> ,
<a href="#">1252</a> , <a href="#">1496</a> , <a href="#">1563</a> , <a href="#">1564</a> , <a href="#">1565</a> , <a href="#">1566</a> ,
<a href="#">1702</a> , <a href="#">1703</a> , <a href="#">1704</a> , <a href="#">1705</a> , <a href="#">1754</a> , <a href="#">1755</a> ,
<a href="#">1756</a> , <a href="#">1757</a> , <a href="#">1861</a> , <a href="#">1862</a> , <a href="#">1863</a> , <a href="#">1864</a>

draw internal commands:

\_draw_align_currentpoint_... . . <a href="#">31</a>
\_draw_backend_add_to_path:n . . .
<a href="#">1373</a> , <a href="#">1419</a>
\_draw_backend_begin: <a href="#">974</a> , <a href="#">1173</a> , <a href="#">1367</a>
\_draw_backend_box_use:Nnnnn . . .
<a href="#">26</a> , <a href="#">1149</a> , <a href="#">1347</a> , <a href="#">1534</a>
\_draw_backend_cap_but: . . . . .
<a href="#">1112</a> , <a href="#">1236</a> , <a href="#">1486</a>
\_draw_backend_cap_rectangle: . .
<a href="#">1112</a> , <a href="#">1236</a> , <a href="#">1486</a>
\_draw_backend_cap_round: . . . . .
<a href="#">1112</a> , <a href="#">1236</a> , <a href="#">1486</a>

1251, 1255, 1257, 1259, 1261, 1263,  
1265, 1267, 1365, 1425, 1444, 1470

\\_draw\_backend\_miterlimit:n . . .  
..... 1112, 1236, 1486

\\_draw\_backend\_moveto:nn . . . . .  
..... 988, 1179, 1373

\\_draw\_backend\_nonzero\_rule: . . .  
..... 1023, 1208, 1414

\\_draw\_backend\_path:n . . . . . 1418

\\_draw\_backend\_rectangle:nnnn . .  
..... 988, 1179, 1373

\\_draw\_backend\_scope:n 1415, 1417,  
1437, 1477, 1499, 1511, 1513, 1515,  
1517, 1519, 1521, 1523, 1525, 1528

\\_draw\_backend\_scope\_begin: . . .  
..... 984, 1174, 1177

\\_draw\_backend\_scope\_end: . . . . .  
..... 984, 1176, 1177

\\_draw\_backend\_stroke: . . . . .  
..... 1028, 1213, 1418

\g\_draw\_clip\_path\_int . . . . .  
.. 1424, 1427, 1440, 1469, 1472, 1480

\g\_draw\_draw\_clip\_bool .. 1028, 1418

\g\_draw\_draw\_eor\_bool . . . . .  
... 1023, 1040, 1058, 1066, 1080,  
1089, 1105, 1208, 1222, 1227, 1232

\g\_draw\_draw\_path\_int . . . . . 1418

\g\_draw\_draw\_path\_tl . . . . .  
.. 1373, 1429, 1445, 1447, 1474, 1483

\g\_draw\_path\_int . . . . . 1433, 1450

## E

\endlandscape . . . . . 2667

\errmessage . . . . . 38

\evensidemargin . . . . . 2072

exp commands:

\exp\_after:wN . . . . . 452, 1832

\exp\_args:Ne . . . . . 600

\exp\_args:Nf . . . . . 1117, 1241, 1994

\exp\_args:NNf . . . . . 212, 260, 317

\exp\_args:Nnx . . . . . 1981, 2653

\exp\_args:NV . . . . . 448

\exp\_args:Nx . . . . .  
.. 1650, 1671, 1938, 1953, 2068, 2619

\exp\_last\_unbraced:Nx . . . . . 457, 490

\exp\_not:N . . . . . 555, 558,  
2298, 2300, 2303, 2333, 2335, 2338,  
2493, 2495, 2498, 2504, 2506, 2509,  
2546, 2547, 2553, 2554, 2573, 2578

\exp\_not:n .. 48, 89, 100, 128, 1929,  
1934, 2221, 2448, 2449, 2463, 2464,  
2476, 2477, 2631, 2636, 2647, 2725

\ExplBackendFileDate . . . . . 1

## F

file commands:

\file\_compare\_timestamp:nNnTF . 1659

\file\_parse\_full\_name:nNNN 1646, 1669

fp commands:

\fp\_compare:nNnTF . . . . .  
.. 219, 266, 272, 324, 1284, 1297, 1342

\fp\_eval:n . . . . . 212, 221,  
234, 235, 260, 277, 292, 294, 317,  
326, 337, 338, 402, 417, 418, 912,  
913, 914, 927, 943, 944, 945, 1286,  
1291, 1292, 1299, 1309, 1310, 1311,  
1312, 1321, 1322, 1323, 1324, 1333,  
1334, 1335, 1336, 2218, 2371, 2738

\fp\_new:N . . . . . 285, 286

\fp\_set:Nn . . . . . 265, 268

\fp\_use:N . . . . . 271, 275, 280

\fp\_zero:N . . . . . 267

\c\_zero\_fp 219, 266, 272, 324, 1284, 1297

## G

graphics commands:

\graphics\_bb\_restore:nTF . 1601, 1815

\graphics\_bb\_save:n . . . . . 1630, 1823

\l\_graphics\_decodearray\_tl . . . . .  
..... 1578, 1579,  
1589, 1609, 1613, 1614, 1691, 1723,  
1724, 1762, 1765, 1766, 1784, 1848

\graphics\_extract\_bb:n . . . . .  
..... 1686, 1693, 1843, 1850

\l\_graphics\_interpolate\_bool . . .  
..... 1580, 1590, 1608, 1615,  
1692, 1725, 1761, 1767, 1785, 1849

\l\_graphics\_llx\_dim . . . . .  
..... 1563, 1702, 1754, 1861

\l\_graphics\_lly\_dim . . . . .  
..... 1564, 1703, 1755, 1862

\l\_graphics\_name\_tl . . . . . 1664

\l\_graphics\_page\_int . . . . .  
..... 1574, 1594, 1595, 1619,  
1620, 1684, 1721, 1722, 1748, 1749,  
1777, 1790, 1791, 1830, 1831, 1841

\l\_graphics\_pagebox\_tl . . . . .  
..... 47, 1575, 1593,  
1621, 1622, 1685, 1719, 1720, 1750,  
1752, 1778, 1799, 1800, 1832, 1842

\graphics\_read\_bb:n . 1557, 1680, 1838

\l\_graphics\_urx\_dim . . . . .  
.. 1565, 1626, 1704, 1756, 1821, 1863

\l\_graphics\_ury\_dim .. 1566, 1627,  
1705, 1757, 1822, 1864, 1872, 1873

graphics internal commands:

\l\_graphics\_backend\_dir\_str . 1639

\l\_graphics\_backend\_ext\_str . 1639

<code>\__graphics_backend_getbb_auxi:n</code>	<code>\l__graphics_internal_box</code> .....
..... <a href="#">1572</a>	.. <a href="#">1624</a> , <a href="#">1626</a> , <a href="#">1627</a> , <a href="#">1820</a> , <a href="#">1821</a> , <a href="#">1822</a>
<code>\__graphics_backend_getbb_-auxi:nN</code> .....	<code>\g__graphics_track_int</code> .....
<a href="#">1775</a>	..... <a href="#">1696</a> , <a href="#">1742</a> , <a href="#">1743</a>
<code>\__graphics_backend_getbb_-auxii:n</code> .....	group commands:
<a href="#">1572</a>	<code>\group_begin:</code> <a href="#">155</a> , <a href="#">174</a> , <a href="#">2227</a> , <a href="#">2376</a> , <a href="#">2744</a>
<code>\__graphics_backend_getbb_-auxii:nnN</code> .....	<code>\group_end:</code> .... <a href="#">163</a> , <a href="#">2247</a> , <a href="#">2390</a> , <a href="#">2771</a>
<a href="#">1775</a>	<code>\group_insert_after:N</code> .....
<code>\__graphics_backend_getbb_-auxiii:nNnn</code> .....	..... <a href="#">435</a> , <a href="#">479</a> , <a href="#">533</a> , <a href="#">737</a>
<a href="#">1775</a>	
<code>\__graphics_backend_getbb_-auxiv:nnNnn</code> .....	<b>H</b>
<a href="#">1775</a>	hbox commands:
<code>\__graphics_backend_getbb_-auxv:nNnn</code> .....	<code>\hbox:n</code> .....
<a href="#">1775</a>	<a href="#">2000</a> , <a href="#">2003</a> , <a href="#">2075</a> , <a href="#">2081</a> , <a href="#">2231</a> , <a href="#">2235</a> , <a href="#">2748</a> , <a href="#">2757</a>
<code>\__graphics_backend_getbb_-auxvi:nNnn</code> .....	<code>\hbox_overlap_right:n</code> .....
<a href="#">1816</a> , <a href="#">1818</a>	<a href="#">207</a> , <a href="#">239</a> , <a href="#">255</a> , <a href="#">296</a> , <a href="#">312</a> , <a href="#">340</a> , <a href="#">424</a> , <a href="#">1162</a> , <a href="#">1357</a>
<code>\__graphics_backend_getbb_eps:n</code> ..	<code>\hbox_set:Nn</code> .....
..... <a href="#">1557</a> , <a href="#">1639</a> , <a href="#">1680</a> , <a href="#">1838</a>	<a href="#">1624</a> , <a href="#">1820</a> , <a href="#">2067</a> , <a href="#">2099</a> , <a href="#">2228</a> , <a href="#">2377</a> , <a href="#">2745</a>
<code>\__graphics_backend_getbb_eps:nm</code>	<code>\hbox_set:Nw</code> .....
..... <a href="#">1639</a>	<a href="#">2050</a>
<code>\__graphics_backend_getbb_eps:nn</code>	<code>\hbox_set_end:</code> .....
..... <a href="#">1650</a> , <a href="#">1657</a>	<a href="#">2065</a>
<code>\__graphics_backend_getbb_jpg:n</code> ..	<code>\hbox_unpack:N</code> .....
..... <a href="#">1572</a> , <a href="#">1680</a> , <a href="#">1775</a> , <a href="#">1839</a>	<a href="#">2187</a>
<code>\__graphics_backend_getbb_-pagebox:w</code> .....	
<a href="#">1775</a> , <a href="#">1832</a>	<b>I</b>
<code>\__graphics_backend_getbb_pdf:n</code> ..	int commands:
..... <a href="#">1572</a> , <a href="#">1665</a> , <a href="#">1680</a> , <a href="#">1775</a> , <a href="#">1846</a>	<code>\int_compare:nNnTF</code> .....
<code>\__graphics_backend_getbb_png:n</code> ..	..... <a href="#">1594</a> , <a href="#">1619</a> , <a href="#">1721</a> , <a href="#">1748</a> , <a href="#">1790</a> , <a href="#">1830</a> , <a href="#">2158</a> , <a href="#">2251</a> , <a href="#">2544</a> , <a href="#">2572</a>
..... <a href="#">1572</a> , <a href="#">1680</a> , <a href="#">1775</a> , <a href="#">1839</a>	<code>\int_const:Nn</code> .....
<a href="#">1852</a>	..... <a href="#">1628</a> , <a href="#">1743</a> , <a href="#">1898</a> , <a href="#">2422</a> , <a href="#">2610</a>
<code>\__graphics_backend_include_-auxi:nn</code> .....	<code>\int_eval:n</code> .....
<a href="#">1697</a>	<a href="#">596</a> , <a href="#">605</a> , <a href="#">618</a> , <a href="#">620</a> , <a href="#">624</a> , <a href="#">637</a> , <a href="#">2522</a> , <a href="#">2547</a> , <a href="#">2554</a> , <a href="#">2567</a> , <a href="#">2774</a> , <a href="#">2782</a> , <a href="#">2787</a>
<code>\__graphics_backend_include_-auxii:nnn</code> .....	<code>\int_gincr:N</code> .....
<a href="#">1697</a>	. <a href="#">181</a> , <a href="#">347</a> , <a href="#">1424</a> , <a href="#">1469</a> , <a href="#">1742</a> , <a href="#">1897</a> , <a href="#">1966</a> , <a href="#">2010</a> , <a href="#">2084</a> , <a href="#">2609</a> , <a href="#">2652</a> , <a href="#">2688</a>
<code>\__graphics_backend_include_-auxiii:nnn</code> .....	<code>\int_gset:Nn</code> .....
<a href="#">1697</a>	<a href="#">156</a> , <a href="#">175</a> , <a href="#">2147</a>
<code>\__graphics_backend_include_-bitmap_quote:w</code> .....	<code>\int_gset_eq:NN</code> <a href="#">164</a> , <a href="#">2011</a> , <a href="#">2085</a> , <a href="#">2689</a>
<a href="#">1826</a> , <a href="#">1867</a>	<code>\int_if_exist:NnTF</code> .....
<code>\__graphics_backend_include_-eps:n</code> .....	<a href="#">1732</a>
<a href="#">1558</a> , <a href="#">1639</a> , <a href="#">1697</a> , <a href="#">1852</a>	<code>\int_if_odd:nTF</code> .....
<code>\__graphics_backend_include_-jpg:n</code> .....	<a href="#">2070</a>
<a href="#">1632</a> , <a href="#">1697</a> , <a href="#">1867</a>	<code>\int_new:N</code> .....
<code>\__graphics_backend_include_-pdf:n</code> ..	.. <a href="#">147</a> , <a href="#">148</a> , <a href="#">394</a> , <a href="#">517</a> , <a href="#">1450</a> , <a href="#">1696</a> , <a href="#">1893</a> , <a href="#">1991</a> , <a href="#">2022</a> , <a href="#">2024</a> , <a href="#">2605</a> , <a href="#">2670</a>
<a href="#">1632</a> , <a href="#">1671</a> , <a href="#">1697</a> , <a href="#">1826</a> , <a href="#">1852</a>	<code>\int_set_eq:NN</code> .....
<code>\__graphics_backend_include_pdf_-quote:w</code> .....	<a href="#">152</a> , <a href="#">171</a> , <a href="#">2159</a>
<a href="#">1829</a> , <a href="#">1834</a>	<code>\int_step_function:nnnN</code> .....
<code>\__graphics_backend_include_-png:n</code> .....	<a href="#">622</a>
<a href="#">1632</a> , <a href="#">1697</a> , <a href="#">1867</a>	<code>\int_use:N</code> . <a href="#">349</a> , <a href="#">380</a> , <a href="#">570</a> , <a href="#">706</a> , <a href="#">762</a> , <a href="#">774</a> , <a href="#">845</a> , <a href="#">1427</a> , <a href="#">1433</a> , <a href="#">1440</a> , <a href="#">1472</a> , <a href="#">1480</a> , <a href="#">1595</a> , <a href="#">1620</a> , <a href="#">1635</a> , <a href="#">1722</a> , <a href="#">1735</a> , <a href="#">1747</a> , <a href="#">1749</a> , <a href="#">1831</a> , <a href="#">1904</a> , <a href="#">1969</a> , <a href="#">1982</a> , <a href="#">1986</a> , <a href="#">2014</a> , <a href="#">2021</a> , <a href="#">2089</a> , <a href="#">2192</a> , <a href="#">2433</a> , <a href="#">2443</a> , <a href="#">2616</a> , <a href="#">2654</a> , <a href="#">2659</a> , <a href="#">2692</a> , <a href="#">2700</a>
<code>\l__graphics_backend_name_str</code> . <a href="#">1639</a>	<code>\int_value:w</code> .....
<code>\l__graphics_graphics_attr_tl</code> ...	..... <a href="#">2298</a> , <a href="#">2333</a> , <a href="#">2493</a> , <a href="#">2504</a> , <a href="#">2522</a>
..... <a href="#">1571</a> , <a href="#">1576</a> , <a href="#">1583</a> , <a href="#">1591</a> , <a href="#">1601</a> , <a href="#">1628</a> , <a href="#">1630</a> , <a href="#">1635</a>	<code>\int_zero:N</code> ... <a href="#">1574</a> , <a href="#">1684</a> , <a href="#">1777</a> , <a href="#">1841</a>

<b>K</b>	
kernel internal commands:	
\_kernel_backend_align_begin: ..	
.....	<a href="#">64</a> , <a href="#">192</a> , <a href="#">216</a> , <a href="#">231</a>
\_kernel_backend_align_end: ...	
.....	<a href="#">64</a> , <a href="#">206</a> , <a href="#">224</a> , <a href="#">238</a>
\g_kernel_backend_header_bool ..	
.....	<a href="#">57</a> , <a href="#">552</a>
\_kernel_backend_literal:n .	<a href="#">46</a> ,
	<a href="#">52</a> , <a href="#">55</a> , <a href="#">62</a> , <a href="#">66</a> , <a href="#">73</a> , <a href="#">76</a> , <a href="#">78</a> , <a href="#">134</a> , <a href="#">137</a> ,
	<a href="#">139</a> , <a href="#">141</a> , <a href="#">145</a> , <a href="#">321</a> , <a href="#">334</a> , <a href="#">434</a> , <a href="#">440</a> ,
	<a href="#">474</a> , <a href="#">482</a> , <a href="#">566</a> , <a href="#">702</a> , <a href="#">736</a> , <a href="#">772</a> , <a href="#">976</a> ,
	<a href="#">982</a> , <a href="#">1281</a> , <a href="#">1288</a> , <a href="#">1294</a> , <a href="#">1354</a> , <a href="#">1359</a> ,
	<a href="#">1560</a> , <a href="#">1699</a> , <a href="#">1734</a> , <a href="#">1744</a> , <a href="#">1858</a> , <a href="#">1869</a> ,
	<a href="#">2599</a> , <a href="#">2720</a> , <a href="#">2774</a> , <a href="#">2778</a> , <a href="#">2783</a> , <a href="#">2788</a>
\_kernel_backend_literal_page:n	
.....	<a href="#">92</a> , <a href="#">136</a> , <a href="#">2593</a> , <a href="#">2595</a> , <a href="#">2793</a> , <a href="#">2795</a>
\_kernel_backend_literal_pdf:n .	
.....	<a href="#">81</a> , <a href="#">133</a> , <a href="#">247</a> , <a href="#">304</a> , <a href="#">862</a> ,
	<a href="#">864</a> , <a href="#">866</a> , <a href="#">868</a> , <a href="#">870</a> , <a href="#">872</a> , <a href="#">874</a> , <a href="#">876</a> , <a href="#">1171</a>
\_kernel_backend_literal_-	
postscript:n .....	
.....	<a href="#">51</a> , <a href="#">67</a> , <a href="#">68</a> , <a href="#">72</a> , <a href="#">193</a> , <a href="#">194</a> ,
	<a href="#">196</a> , <a href="#">197</a> , <a href="#">205</a> , <a href="#">217</a> , <a href="#">232</a> , <a href="#">972</a> , <a href="#">2253</a> , <a href="#">2265</a>
\_kernel_backend_literal_svg:n .	
.....	<a href="#">144</a> , <a href="#">151</a> , <a href="#">162</a> , <a href="#">170</a> ,
	<a href="#">180</a> , <a href="#">348</a> , <a href="#">350</a> , <a href="#">367</a> , <a href="#">1365</a> , <a href="#">1538</a> , <a href="#">1549</a>
\_kernel_backend_matrix:n .....	
.....	<a href="#">120</a> , <a href="#">269</a> , <a href="#">290</a> , <a href="#">1271</a>
\_kernel_backend_postscript:n ..	
.....	<a href="#">54</a> , <a href="#">476</a> , <a href="#">477</a> , <a href="#">882</a> ,
	<a href="#">884</a> , <a href="#">886</a> , <a href="#">888</a> , <a href="#">890</a> , <a href="#">892</a> , <a href="#">894</a> , <a href="#">896</a> ,
	<a href="#">1887</a> , <a href="#">1943</a> , <a href="#">1958</a> , <a href="#">2000</a> , <a href="#">2006</a> , <a href="#">2043</a> ,
	<a href="#">2075</a> , <a href="#">2082</a> , <a href="#">2086</a> , <a href="#">2100</a> , <a href="#">2128</a> , <a href="#">2173</a> ,
	<a href="#">2180</a> , <a href="#">2186</a> , <a href="#">2195</a> , <a href="#">2202</a> , <a href="#">2231</a> , <a href="#">2235</a>
\_kernel_backend_scope:n .....	
.....	<a href="#">149</a> , <a href="#">377</a> , <a href="#">382</a> , <a href="#">950</a> , <a href="#">1370</a>
\_kernel_backend_scope_begin: ..	
.....	<a href="#">75</a> , <a href="#">102</a> , <a href="#">138</a> ,
	<a href="#">149</a> , <a href="#">191</a> , <a href="#">215</a> , <a href="#">230</a> , <a href="#">246</a> , <a href="#">263</a> , <a href="#">289</a> ,
	<a href="#">303</a> , <a href="#">320</a> , <a href="#">333</a> , <a href="#">1177</a> , <a href="#">1349</a> , <a href="#">1369</a> , <a href="#">1536</a>
\_kernel_backend_scope_begin:n .	
.....	<a href="#">149</a> , <a href="#">369</a> , <a href="#">397</a> , <a href="#">410</a>
\_kernel_backend_scope_end: ...	
.....	<a href="#">75</a> , <a href="#">102</a> , <a href="#">138</a> , <a href="#">149</a> , <a href="#">208</a> , <a href="#">226</a> ,
	<a href="#">240</a> , <a href="#">256</a> , <a href="#">283</a> , <a href="#">297</a> , <a href="#">313</a> , <a href="#">329</a> , <a href="#">341</a> ,
	<a href="#">392</a> , <a href="#">406</a> , <a href="#">425</a> , <a href="#">1178</a> , <a href="#">1361</a> , <a href="#">1372</a> , <a href="#">1550</a>
\g_kernel_backend_scope_int ...	
.....	<a href="#">147</a> , <a href="#">154</a> , <a href="#">156</a> , <a href="#">161</a> , <a href="#">165</a> , <a href="#">173</a> , <a href="#">175</a> , <a href="#">181</a>
\l_kernel_backend_scope_int ...	
.....	<a href="#">147</a> , <a href="#">153</a> , <a href="#">166</a> , <a href="#">172</a>
\l_kernel_color_stack_int .....	
.....	<a href="#">517</a> , <a href="#">532</a> , <a href="#">543</a>
\_kernel_dependency_version_-	
check:Nn .....	<a href="#">1</a>
\_kernel_dependency_version_-	
check:nn .....	<a href="#">27</a> , <a href="#">29</a>
<b>L</b>	
\landscape .....	<a href="#">2663</a> , <a href="#">2665</a>
<b>M</b>	
math commands:	
\c_math_toggle_token ....	<a href="#">2053</a> , <a href="#">2063</a>
\MessageBreak .....	<a href="#">40</a>
mode commands:	
\mode_if_horizontal:TF ...	<a href="#">2149</a> , <a href="#">2156</a>
\mode_if_math:TF .....	<a href="#">2047</a>
<b>O</b>	
\oddsidemargin .....	<a href="#">2071</a>
<b>P</b>	
pdf commands:	
\pdf_object_if_exist:nTF .....	<a href="#">784</a>
\pdf_object_last: ..	<a href="#">755</a> , <a href="#">763</a> , <a href="#">838</a> , <a href="#">846</a>
\pdf_object_new:nn .....	<a href="#">786</a>
\pdf_object_now:nn .....	
.....	<a href="#">743</a> , <a href="#">767</a> , <a href="#">779</a> , <a href="#">804</a> , <a href="#">811</a> , <a href="#">850</a>
\pdf_object_ref:n .....	<a href="#">799</a>
\pdf_object_write:nn .....	<a href="#">787</a>
pdf internal commands:	
\_pdf_backend:n .....	<a href="#">2598</a> ,
	<a href="#">2602</a> , <a href="#">2604</a> , <a href="#">2630</a> , <a href="#">2635</a> , <a href="#">2644</a> , <a href="#">2690</a> ,
	<a href="#">2707</a> , <a href="#">2717</a> , <a href="#">2723</a> , <a href="#">2750</a> , <a href="#">2751</a> , <a href="#">2759</a>
\_pdf_backend_annotation:nnnn ..	
.....	<a href="#">1992</a> , <a href="#">2283</a> , <a href="#">2671</a>
\_pdf_backend_annotation_-	
aux:nnnn .....	<a href="#">1994</a> , <a href="#">1997</a> , <a href="#">2671</a>
\g_pdf_backend_annotation_int ..	
..	<a href="#">1991</a> , <a href="#">2011</a> , <a href="#">2021</a> , <a href="#">2670</a> , <a href="#">2689</a> , <a href="#">2700</a>
\_pdf_backend_annotation_last: .	
.....	<a href="#">2020</a> , <a href="#">2296</a> , <a href="#">2699</a>
\_pdf_backend_bdc:nn .....	
.....	<a href="#">2277</a> , <a href="#">2592</a> , <a href="#">2792</a> , <a href="#">2814</a>
\_pdf_backend_catalog_gput:nn ..	
.....	<a href="#">1889</a> , <a href="#">2392</a> , <a href="#">2601</a> , <a href="#">2798</a>
\_pdf_backend_compress_objects:n	
.....	<a href="#">2249</a> , <a href="#">2513</a> , <a href="#">2773</a> , <a href="#">2808</a>
\_pdf_backend_compresslevel:n ..	
.....	<a href="#">2249</a> , <a href="#">2513</a> , <a href="#">2773</a> , <a href="#">2808</a>
\l_pdf_backend_content_box ..	<a href="#">1989</a> ,
	<a href="#">2050</a> , <a href="#">2074</a> , <a href="#">2077</a> , <a href="#">2079</a> , <a href="#">2108</a> , <a href="#">2119</a>
\_pdf_backend_destination:nn ...	
.....	<a href="#">2200</a> , <a href="#">2352</a> , <a href="#">2721</a>
\_pdf_backend_destination_-	
box:nn .....	<a href="#">2200</a> , <a href="#">2352</a> , <a href="#">2721</a>



\__pdf_backend_emc: .....	\g__pdf_backend_object_prop .....
..... <a href="#">2277</a> , <a href="#">2592</a> , <a href="#">2792</a> , <a href="#">2814</a>	..... <a href="#">1893</a> , <a href="#">1901</a> , <a href="#">1912</a> , <a href="#">1922</a> ,
\__pdf_backend_info_gput:nn .....	<a href="#">2412</a> , <a href="#">2430</a> , <a href="#">2446</a> , <a href="#">2605</a> , <a href="#">2613</a> , <a href="#">2620</a>
..... <a href="#">1889</a> , <a href="#">2392</a> , <a href="#">2601</a> , <a href="#">2798</a>	\__pdf_backend_object_ref:n <a href="#">1895</a> ,
\__pdf_backend_link:nw .....	<a href="#">1909</a> , <a href="#">1923</a> , <a href="#">2413</a> , <a href="#">2607</a> , <a href="#">2626</a> , <a href="#">2800</a>
<a href="#">2031</a>	\__pdf_backend_object_write:nn ..
\__pdf_backend_link_aux:nw ... <a href="#">2031</a>	..... <a href="#">1905</a> , <a href="#">2434</a> , <a href="#">2617</a> , <a href="#">2800</a>
\__pdf_backend_link_begin:n .. <a href="#">2701</a>	\__pdf_backend_object_write:nnn <a href="#">2617</a>
\__pdf_backend_link_begin:nnnw <a href="#">2307</a>	\__pdf_backend_object_write_-
\__pdf_backend_link_begin:nw ...	array:nn .....
..... <a href="#">2032</a> , <a href="#">2034</a> , <a href="#">2035</a>	<a href="#">1905</a> , <a href="#">2617</a>
\__pdf_backend_link_begin_aux:nw	\__pdf_backend_object_write_-
..... <a href="#">2038</a> , <a href="#">2040</a>	dict:nn .....
\__pdf_backend_link_begin_-	<a href="#">1905</a> , <a href="#">2617</a>
goto:nnw .....	\__pdf_backend_object_write_-
<a href="#">2031</a> , <a href="#">2307</a> , <a href="#">2701</a>	fstream:nn .....
\__pdf_backend_link_begin_-	<a href="#">1905</a> , <a href="#">2617</a>
user:nnw .....	\__pdf_backend_object_write_-
<a href="#">2031</a> , <a href="#">2307</a> , <a href="#">2701</a>	fstream:nnn .....
\g__pdf_backend_link_bool .....	<a href="#">1939</a> , <a href="#">1941</a>
..... <a href="#">2026</a> , <a href="#">2037</a> , <a href="#">2042</a> , <a href="#">2057</a> , <a href="#">2095</a>	\__pdf_backend_object_write_-
\g__pdf_backend_link_dict_tl ...	stream:nn .....
..... <a href="#">2023</a> , <a href="#">2045</a> , <a href="#">2090</a>	<a href="#">1905</a> , <a href="#">2617</a>
\__pdf_backend_link_end: .....	\__pdf_backend_object_write_-
..... <a href="#">2031</a> , <a href="#">2307</a> , <a href="#">2701</a>	stream:nnn .....
\__pdf_backend_link_end_aux: . <a href="#">2031</a>	<a href="#">1905</a>
\g__pdf_backend_link_int .....	\__pdf_backend_object_write_-
..... <a href="#">2022</a> , <a href="#">2085</a> , <a href="#">2089</a> , <a href="#">2192</a>	stream:nnnn .....
\__pdf_backend_link_last: .....	<a href="#">2617</a>
..... <a href="#">2191</a> , <a href="#">2331</a> , <a href="#">2718</a>	\__pdf_backend_pageobject_ref:n .
\__pdf_backend_link_margin:n ...	..... <a href="#">1987</a> , <a href="#">2502</a> , <a href="#">2660</a> , <a href="#">2800</a>
..... <a href="#">2193</a> , <a href="#">2342</a> , <a href="#">2719</a>	\__pdf_backend_pdfmark:n .....
\g__pdf_backend_link_math_bool ..	<a href="#">1886</a> , <a href="#">1890</a> , <a href="#">1892</a> , <a href="#">1907</a> , <a href="#">1928</a> , <a href="#">1933</a> ,
..... <a href="#">2025</a> , <a href="#">2048</a> , <a href="#">2049</a> , <a href="#">2052</a> , <a href="#">2062</a>	<a href="#">1967</a> , <a href="#">2012</a> , <a href="#">2203</a> , <a href="#">2236</a> , <a href="#">2278</a> , <a href="#">2280</a>
\__pdf_backend_link_minima: . <a href="#">2031</a>	\__pdf_backend_version_major: ...
\__pdf_backend_link_outerbox:n <a href="#">2031</a>	.. <a href="#">2275</a> , <a href="#">2569</a> , <a href="#">2782</a> , <a href="#">2783</a> , <a href="#">2790</a> , <a href="#">2812</a>
\g__pdf_backend_link_sf_int ....	\__pdf_backend_version_major_-
..... <a href="#">2024</a> , <a href="#">2147</a> , <a href="#">2158</a> , <a href="#">2159</a>	gset:n .....
\__pdf_backend_link_sf_restore: <a href="#">2031</a>	<a href="#">2273</a> , <a href="#">2541</a> , <a href="#">2780</a> , <a href="#">2810</a>
\__pdf_backend_link_sf_save: . <a href="#">2031</a>	\__pdf_backend_version_minor: ...
\l__pdf_backend_model_box . <a href="#">1990</a> ,	.. <a href="#">2275</a> , <a href="#">2569</a> , <a href="#">2787</a> , <a href="#">2788</a> , <a href="#">2790</a> , <a href="#">2812</a>
<a href="#">2067</a> , <a href="#">2099</a> , <a href="#">2107</a> , <a href="#">2118</a> , <a href="#">2133</a> , <a href="#">2135</a>	\__pdf_backend_version_minor_-
\__pdf_backend_objcompresslevel:n	gset:n .....
..... <a href="#">2513</a>	<a href="#">2273</a> , <a href="#">2541</a> , <a href="#">2780</a> , <a href="#">2810</a>
\g__pdf_backend_object_int .....	\l__pdf_breaklink_pdfmark_tl ...
..... <a href="#">1893</a> , <a href="#">1897</a> , <a href="#">1900</a> ,	..... <a href="#">2027</a> , <a href="#">2092</a> , <a href="#">2185</a>
<a href="#">1966</a> , <a href="#">1969</a> , <a href="#">1982</a> , <a href="#">1986</a> , <a href="#">2010</a> , <a href="#">2011</a> ,	\__pdf_breaklink_postscript:n ...
<a href="#">2014</a> , <a href="#">2084</a> , <a href="#">2085</a> , <a href="#">2605</a> , <a href="#">2609</a> , <a href="#">2612</a> ,	..... <a href="#">2029</a> , <a href="#">2076</a> , <a href="#">2078</a> , <a href="#">2186</a>
<a href="#">2652</a> , <a href="#">2654</a> , <a href="#">2659</a> , <a href="#">2688</a> , <a href="#">2689</a> , <a href="#">2692</a>	\__pdf_breaklink_usebox:N .....
\__pdf_backend_object_last: ....	..... <a href="#">2030</a> , <a href="#">2077</a> , <a href="#">2187</a>
..... <a href="#">1985</a> , <a href="#">2491</a> , <a href="#">2658</a> , <a href="#">2800</a>	\__pdf_exp_not_i:nn . <a href="#">2434</a> , <a href="#">2480</a> , <a href="#">2485</a>
\__pdf_backend_object_new:nn ...	\__pdf_exp_not_ii:nn <a href="#">2434</a> , <a href="#">2481</a> , <a href="#">2486</a>
..... <a href="#">1895</a> , <a href="#">2413</a> , <a href="#">2607</a> , <a href="#">2800</a>	\l__pdf_internal_box . <a href="#">1884</a> , <a href="#">2228</a> ,
\__pdf_backend_object_now:nn ...	<a href="#">2230</a> , <a href="#">2232</a> , <a href="#">2234</a> , <a href="#">2377</a> , <a href="#">2386</a> , <a href="#">2387</a> ,
..... <a href="#">1964</a> , <a href="#">2465</a> , <a href="#">2650</a> , <a href="#">2800</a>	<a href="#">2388</a> , <a href="#">2389</a> , <a href="#">2745</a> , <a href="#">2746</a> , <a href="#">2754</a> , <a href="#">2755</a>
	\g__pdf_landscape_bool ... <a href="#">2662</a> , <a href="#">2673</a>
	pdf.baselineskip .....
	<a href="#">2031</a> , <a href="#">3147</a>
	pdf.bordertracking .....
	<a href="#">2905</a>
	pdf.bordertracking.begin .....
	<a href="#">2905</a>
	pdf.bordertracking.continue .....
	<a href="#">2905</a>
	pdf.bordertracking.end .....
	<a href="#">2905</a>
	pdf.bordertracking.endpage .....
	<a href="#">2905</a>
	pdf.breaklink .....
	<a href="#">3043</a>

pdf.breaklink.write	3043	
pdf.brokenlink.dict	2905	
pdf.brokenlink.rect	2905	
pdf.brokenlink.skip	2905	
pdf.count	3043	
pdf.currentrect	3043	
pdf.cvs	2827	
pdf.dest.anchor	2870	
pdf.dest.point	2870	
pdf.dest.x	2870	
pdf.dest.y	2870	
pdf.dest2device	2870	
pdf.dev.x	2870	
pdf.dev.y	2870	
pdf.dvi.pt	2827	
pdf.globaldict	2824	
pdf.leftboundary	2905	
pdf.link.dict	2031	
pdf.linkdp.pad	2031, 2831	
pdf.linkht.pad	2031, 2831	
pdf.linkmargin	2831	
pdf.llx	2031, 2834	
pdf.lly	2031, 2834	
pdf.originx	2905	
pdf.originy	2905	
pdf.outerbox	2031, 3147	
pdf.pdfmark	3147	
pdf.pdfmark.dict	3147	
pdf.pdfmark.good	3147	
pdf.pt.dvi	2827	
pdf.rect	2834	
pdf.rect.ht	2827	
pdf.rightboundary	2905	
pdf.save.linkll	2834	
pdf.save.linkur	2834	
pdf.save.ll	2834	
pdf.save.ur	2834	
pdf.tmpa	2870	
pdf.tmpb	2870	
pdf.tmpc	2870	
pdf.tmpd	2870	
pdf.urx	2834	
pdf.ury	2031, 2834	
pdfcoredict commands:		
\pdfcoredict_gput:nnn	760, 843	
prg commands:		
\prg_replicate:nn	160, 594, 615, 625, 817	
prop commands:		
\prop_gput:Nnn	1901, 2430, 2613	
\prop_item:Nn	1912, 1922, 2446, 2620	
\prop_new:N	1894, 2412, 2606	
\ProvidesExplFile	2	

Q

quark internal commands:

\s\_\_color\_stop

458, 461, 491, 494, 605, 606, 610, 614, 627, 630, 634, 638, 652, 818, 852, 856, 902, 904, 906, 933, 935, 937

\s\_\_graphics\_stop

1829, 1834, 1874, 1878

S

scan commands:

\scan\_stop:

105, 114, 543, 2005, 2007, 2325, 2350, 2372, 2522, 2539, 2547, 2554, 2567

separation

2821

skip commands:

\skip\_horizontal:n

209, 257, 314

str commands:

\c\_hash\_str

380, 1433, 1440, 1480

\c\_percent\_str

956, 957, 958

\str\_case:nn

823, 1971, 2473

\str\_case:nnTF

2207, 2361, 2728

\str\_case\_e:nn

1911, 2445

\str\_convert\_pdfname:n

573, 754

\str\_if\_eq:nnTF

496, 499, 502, 505

\str\_new:N

1641, 1642, 1643

\str\_tail:N

1652, 1673

sys commands:

\sys\_if\_shell:TF

1639

\sys\_shell\_now:n

1661

T

T<sub>E</sub>X and L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> commands:

\@cclv

2169, 2171, 2179

\@makecol@hook

2162

\current@color

15, 448, 452, 458, 491

\special

2

tex commands:

\tex\_baselineskip:D

2139

\tex\_endinput:D

44

\tex\_global:D

2515, 2532, 2546, 2553, 2560

\tex\_immediate:D

1606, 2437, 2440, 2468, 2471

\tex\_kern:D

2005, 2007

\tex luatexversion:D

2544, 2572

\tex\_pdfannot:D

2289

\tex\_pdfcatalog:D

2398

\tex\_pdfcolorstack:D

530, 541

\tex\_pdfcompresslevel:D

2520

\tex\_pdfdest:D

2358, 2382

\tex\_pdfendlink:D

2328

\tex\_pdfextension:D

84, 95, 105, 114, 123,

