

# File I

## Implementation

### 1 l3backend-basics Implementation

```
1 <*initex | 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 <*package>
3 \ProvidesExplFile
4 <*dvipdfmx>
5 {l3backend-dvipdfmx.def}{2020-02-21}{ }
6 {L3 backend support: dvipdfmx}
7 </dvipdfmx>
8 <*dvips>
9 {l3backend-dvips.def}{2020-02-21}{ }
10 {L3 backend support: dvips}
11 </dvips>
12 <*dvisvgm>
13 {l3backend-dvisvgm.def}{2020-02-21}{ }
14 {L3 backend support: dvisvgm}
15 </dvisvgm>
16 <*pdfmode>
17 {l3backend-pdfmode.def}{2020-02-21}{ }
18 {L3 backend support: PDF mode}
19 </pdfmode>
20 <*xdvipdfmx>
21 {l3backend-xdvipdfmx.def}{2020-02-21}{ }
22 {L3 backend support: xdvipdfmx}
23 </xdvipdfmx>
24 </package>
```

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 `pdfmode`-like.
- `pdfmode` and `(x)dvipdfmx` share drawing routines.
- `xdvipdfmx` is largely the same as `dvipdfmx` so takes most of the same code.

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.

```
25 \cs_new_eq:NN \__kernel_backend_literal:e \tex_special:D
26 \cs_new_protected:Npn \__kernel_backend_literal:n #1
27 { \__kernel_backend_literal:e { \exp_not:n {#1} } }
28 \cs_generate_variant:Nn \__kernel_backend_literal:n { x }
```

(End definition for `\__kernel_backend_literal:e`.)

## 1.1 dvips backend

29 `<*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.

```
30 \cs_new_protected:Npn \__kernel_backend_literal_postscript:n #1
31 { \__kernel_backend_literal:n { ps:: #1 } }
32 \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 `"`).

```
33 \cs_new_protected:Npn \__kernel_backend_postscript:n #1
34 { \__kernel_backend_literal:n { ps: SDict ~ begin ~ #1 ~ end } }
35 \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.

```
36 \cs_if_exist:NTF \AtBeginDvi
37 { \exp_not:N \AtBeginDvi }
38 { \use:n }
39 {
40     \bool_if:NT \g__kernel_backend_header_bool
41     { \__kernel_backend_literal:n { header = l3backend-dvips.pro } }
42 }
```

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

```
43 \cs_new_protected:Npn \__kernel_backend_align_begin:
44 {
45     \__kernel_backend_literal:n { ps::[begin] }
46     \__kernel_backend_literal_postscript:n { currentpoint }
47     \__kernel_backend_literal_postscript:n { currentpoint~translate }
48 }
49 \cs_new_protected:Npn \__kernel_backend_align_end:
50 {
51     \__kernel_backend_literal_postscript:n { neg-exch-neg-exch~translate }
52     \__kernel_backend_literal:n { ps::[end] }
53 }
```

(End definition for `\__kernel_backend_align_begin:` and `\__kernel_backend_align_end:.`)

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

```

54 \cs_new_protected:Npn \_kernel_backend_scope_begin:
55   { \_kernel_backend_literal:n { ps:gsave } }
56 \cs_new_protected:Npn \_kernel_backend_scope_end:
57   { \_kernel_backend_literal:n { ps:grestore } }

```

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

```

58 </dvips>

```

## 1.2 pdfmode backend

```

59 <*pdfmode>

```

The direct PDF backend covers both pdfTeX and LuaTeX. The latter renames and restructures the backend primitives but this can be handled at one level of abstraction. As such, we avoid using two separate backends for this material at the cost of some `x`-type definitions to get everything expanded up-front.

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

`\_kernel_backend_literal_pdf:x`

```

60 \cs_new_protected:Npx \_kernel_backend_literal_pdf:n #1
61   {
62     \cs_if_exist:NTF \tex_pdfextension:D
63       { \tex_pdfextension:D literal }
64       { \tex_pdfliteral:D }
65       { \exp_not:N \exp_not:n {#1} }
66   }
67 \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.

```

68 \cs_new_protected:Npx \_kernel_backend_literal_page:n #1
69   {
70     \cs_if_exist:NTF \tex_pdfextension:D
71       { \tex_pdfextension:D literal ~ }
72       { \tex_pdfliteral:D }
73     page
74     { \exp_not:N \exp_not:n {#1} }
75   }

```

(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:`

```

76 \cs_new_protected:Npx \_kernel_backend_scope_begin:
77   {
78     \cs_if_exist:NTF \tex_pdfextension:D
79       { \tex_pdfextension:D save \scan_stop: }
80       { \tex_pdfsave:D }
81   }
82 \cs_new_protected:Npx \_kernel_backend_scope_end:
83   {

```

```

84 \cs_if_exist:NTF \tex_pdfextension:D
85 { \tex_pdfextension:D restore \scan_stop: }
86 { \tex_pdfrestore:D }
87 }

```

(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.

```

88 \cs_new_protected:Npx \_kernel_backend_matrix:n #1
89 {
90   \cs_if_exist:NTF \tex_pdfextension:D
91   { \tex_pdfextension:D setmatrix }
92   { \tex_pdfsetmatrix:D }
93   { \exp_not:N \exp_not:n {#1} }
94 }
95 \cs_generate_variant:Nn \_kernel_backend_matrix:n { x }

```

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

```

96 </pdfmode>

```

### 1.3 dvipdfmx backend

```

97 <dvipdfmx | xdvipdfmx>

```

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

`\_kernel_backend_literal_pdf:n` Equivalent to `pdf:content` but favored as the link to the pdfTeX primitive approach is clearer.

`\_kernel_backend_literal_pdf:x`

```

98 \cs_new_protected:Npn \_kernel_backend_literal_pdf:n #1
99 { \_kernel_backend_literal:n { pdf:literal~ #1 } }
100 \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!

```

101 \cs_new_protected:Npn \_kernel_backend_literal_page:n #1
102 { \_kernel_backend_literal:n { pdf:literal~direct~ #1 } }

```

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

`\_kernel_backend_scope_begin:` Scoping is done using the backend-specific specials.

`\_kernel_backend_scope_end:`

```

103 \cs_new_protected:Npn \_kernel_backend_scope_begin:
104 { \_kernel_backend_literal:n { x:gsave } }
105 \cs_new_protected:Npn \_kernel_backend_scope_end:
106 { \_kernel_backend_literal:n { x:grestore } }

```

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

```

107 </dvipdfmx | xdvipdfmx>

```

## 1.4 dvisvgm backend

108  $\langle *dvisvgm \rangle$

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

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.

109  $\backslash cs\_new\_protected:Npn \backslash\_kernel\_backend\_literal\_svg:n \#1$   
 110  $\{ \backslash\_kernel\_backend\_literal:n \{ dvisvgm:raw~ \#1 \{ ?nl \} \} \}$   
 111  $\backslash cs\_generate\_variant:Nn \backslash\_kernel\_backend\_literal\_svg:n \{ x \}$

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

$\backslash\_kernel\_backend\_scope\_begin:$   
 $\backslash\_kernel\_backend\_scope\_end:$

A scope in SVG terms is slightly different to the other backends as operations have to be “tied” to these not simply inside them.

112  $\backslash cs\_new\_protected:Npn \backslash\_kernel\_backend\_scope\_begin:$   
 113  $\{ \backslash\_kernel\_backend\_literal\_svg:n \{ <g> \} \}$   
 114  $\backslash cs\_new\_protected:Npn \backslash\_kernel\_backend\_scope\_end:$   
 115  $\{ \backslash\_kernel\_backend\_literal\_svg:n \{ </g> \} \}$

(End definition for  $\backslash\_kernel\_backend\_scope\_begin:$  and  $\backslash\_kernel\_backend\_scope\_end:.$ )

$\backslash\_kernel\_backend\_scope\_begin:n$   
 $\backslash\_kernel\_backend\_scope\_begin:x$

In SVG transformations, clips and so on are attached directly to scopes so we need a way or allowing for that. This is rather more useful than  $\backslash\_kernel\_backend\_scope\_begin:$  as a result. No assumptions are made about the nature of the scoped operation(s).

116  $\backslash cs\_new\_protected:Npn \backslash\_kernel\_backend\_scope\_begin:n \#1$   
 117  $\{ \backslash\_kernel\_backend\_literal\_svg:n \{ <g~ \#1 > \} \}$   
 118  $\backslash cs\_generate\_variant:Nn \backslash\_kernel\_backend\_scope\_begin:n \{ x \}$

(End definition for  $\backslash\_kernel\_backend\_scope\_begin:n$ .)

119  $\langle /dvisvgm \rangle$

120  $\langle /initex | package \rangle$

## 2 l3backend-box Implementation

121  $\langle *initex | package \rangle$

122  $\langle @@=box \rangle$

### 2.1 dvips backend

123  $\langle *dvips \rangle$

$\backslash\_box\_backend\_clip:N$

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

124  $\backslash cs\_new\_protected:Npn \backslash\_box\_backend\_clip:N \#1$   
 125  $\{$   
 126  $\backslash\_kernel\_backend\_scope\_begin:$   
 127  $\backslash\_kernel\_backend\_align\_begin:$   
 128  $\backslash\_kernel\_backend\_literal\_postscript:n \{ matrix-currentmatrix \}$   
 129  $\backslash\_kernel\_backend\_literal\_postscript:n$

```

130     { Resolution~72~div~VResolution~72~div~scale }
131   \__kernel_backend_literal_postscript:n { DVImag-dup~scale }
132   \__kernel_backend_literal_postscript:x
133   {
134     0 ~
135     \dim_to_decimal_in_bp:n { \box_dp:N #1 } ~
136     \dim_to_decimal_in_bp:n { \box_wd:N #1 } ~
137     \dim_to_decimal_in_bp:n { -\box_ht:N #1 - \box_dp:N #1 } ~
138     rectclip
139   }
140   \__kernel_backend_literal_postscript:n { setmatrix }
141   \__kernel_backend_align_end:
142   \hbox_overlap_right:n { \box_use:N #1 }
143   \__kernel_backend_scope_end:
144   \skip_horizontal:n { \box_wd:N #1 }
145 }

```

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

\\_\_box\_backend\_rotate:Nn  
\\_\_box\_backend\_rotate\_aux: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.

```

146 \cs_new_protected:Npn \__box_backend_rotate:Nn #1#2
147 { \exp_args:Nnf \__box_backend_rotate_aux:Nn #1 { \fp_eval:n {#2} } }
148 \cs_new_protected:Npn \__box_backend_rotate_aux:Nn #1#2
149 {
150   \__kernel_backend_scope_begin:
151   \__kernel_backend_align_begin:
152   \__kernel_backend_literal_postscript:x
153   {
154     \fp_compare:nNnTF {#2} = \c_zero_fp
155     { 0 }
156     { \fp_eval:n { round ( -(#2) , 5 ) } } ~
157     rotate
158   }
159   \__kernel_backend_align_end:
160   \box_use:N #1
161   \__kernel_backend_scope_end:
162 }

```

(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.

```

163 \cs_new_protected:Npn \__box_backend_scale:Nnn #1#2#3
164 {
165   \__kernel_backend_scope_begin:
166   \__kernel_backend_align_begin:
167   \__kernel_backend_literal_postscript:x
168   {
169     \fp_eval:n { round ( #2 , 5 ) } ~
170     \fp_eval:n { round ( #3 , 5 ) } ~
171     scale
172   }
173   \__kernel_backend_align_end:

```

```

174 \hbox_overlap_right:n { \box_use:N #1 }
175 \__kernel_backend_scope_end:
176 }

```

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

```

177 </dvips>

```

## 2.2 pdfmode backend

```

178 <*pdfmode>

```

\\_\_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.

```

179 \cs_new_protected:Npn \__box_backend_clip:N #1
180 {
181   \__kernel_backend_scope_begin:
182   \__kernel_backend_literal_pdf:x
183   {
184     0~
185     \dim_to_decimal_in_bp:n { -\box_dp:N #1 } ~
186     \dim_to_decimal_in_bp:n { \box_wd:N #1 } ~
187     \dim_to_decimal_in_bp:n { \box_ht:N #1 + \box_dp:N #1 } ~
188     re~W~n
189   }
190   \hbox_overlap_right:n { \box_use:N #1 }
191   \__kernel_backend_scope_end:
192   \skip_horizontal:n { \box_wd:N #1 }
193 }

```

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

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

```

194 \cs_new_protected:Npn \__box_backend_rotate:Nn #1#2
195 { \exp_args:NNf \__box_backend_rotate_aux:Nn #1 { \fp_eval:n {#2} } }
196 \cs_new_protected:Npn \__box_backend_rotate_aux:Nn #1#2
197 {
198   \__kernel_backend_scope_begin:
199   \box_set_wd:Nn #1 { Opt }
200   \fp_set:Nn \l__box_backend_cos_fp { round ( cosd ( #2 ) , 5 ) }
201   \fp_compare:nNnT \l__box_backend_cos_fp = \c_zero_fp
202     { \fp_zero:N \l__box_backend_cos_fp }
203   \fp_set:Nn \l__box_backend_sin_fp { round ( sind ( #2 ) , 5 ) }
204   \__kernel_backend_matrix:x
205   {
206     \fp_use:N \l__box_backend_cos_fp \c_space_tl

```

```

207 \fp_compare:nNnTF \l__box_backend_sin_fp = \c_zero_fp
208 { 0~0 }
209 {
210   \fp_use:N \l__box_backend_sin_fp
211   \c_space_tl
212   \fp_eval:n { -\l__box_backend_sin_fp }
213 }
214 \c_space_tl
215 \fp_use:N \l__box_backend_cos_fp
216 }
217 \box_use:N #1
218 \__kernel_backend_scope_end:
219 }
220 \fp_new:N \l__box_backend_cos_fp
221 \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.

```

222 \cs_new_protected:Npn \__box_backend_scale:Nnn #1#2#3
223 {
224   \__kernel_backend_scope_begin:
225   \__kernel_backend_matrix:x
226   {
227     \fp_eval:n { round ( #2 , 5 ) } ~
228     0~0~
229     \fp_eval:n { round ( #3 , 5 ) }
230   }
231   \hbox_overlap_right:n { \box_use:N #1 }
232   \__kernel_backend_scope_end:
233 }

```

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

234 </pdfmode>

## 2.3 dvipdfmx backend

235 <\*dvipdfmx | xdvipdfmx>

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

```

236 \cs_new_protected:Npn \__box_backend_clip:N #1
237 {
238   \__kernel_backend_scope_begin:
239   \__kernel_backend_literal_pdf:x
240   {
241     0~
242     \dim_to_decimal_in_bp:n { -\box_dp:N #1 } ~
243     \dim_to_decimal_in_bp:n { \box_wd:N #1 } ~
244     \dim_to_decimal_in_bp:n { \box_ht:N #1 + \box_dp:N #1 } ~
245     re~W~n
246   }
247   \hbox_overlap_right:n { \box_use:N #1 }
248   \__kernel_backend_scope_end:

```



```

249     \skip_horizontal:n { \box_wd:N #1 }
250 }

```

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

`\_box_backend_rotate:Nn`  
`\_box_backend_rotate_aux:Nn`

Rotating in (x)dvipdfmx 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.

```

251 \cs_new_protected:Npn \_box_backend_rotate:Nn #1#2
252 { \exp_args:Nnf \_box_backend_rotate_aux:Nn #1 { \fp_eval:n {#2} } }
253 \cs_new_protected:Npn \_box_backend_rotate_aux:Nn #1#2
254 {
255   \__kernel_backend_scope_begin:
256   \__kernel_backend_literal:x
257   {
258     x:rotate~
259     \fp_compare:nNnTF {#2} = \c_zero_fp
260     { 0 }
261     { \fp_eval:n { round ( #2 , 5 ) } }
262   }
263   \box_use:N #1
264   \__kernel_backend_scope_end:
265 }

```

(End definition for `\_box_backend_rotate:Nn` and `\_box_backend_rotate_aux:Nn`.)

`\_box_backend_scale:Nnn`

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

```

266 \cs_new_protected:Npn \_box_backend_scale:Nnn #1#2#3
267 {
268   \__kernel_backend_scope_begin:
269   \__kernel_backend_literal:x
270   {
271     x:scale~
272     \fp_eval:n { round ( #2 , 5 ) } ~
273     \fp_eval:n { round ( #3 , 5 ) }
274   }
275   \hbox_overlap_right:n { \box_use:N #1 }
276   \__kernel_backend_scope_end:
277 }

```

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

278 `</dvipdfmx | xdvipdfmx>`

## 2.4 dvisvgm backend

279 `<*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!

```

280 \cs_new_protected:Npn \__box_backend_clip:N #1
281 {
282   \int_gincr:N \g__box_clip_path_int
283   \__kernel_backend_literal_svg:x
284   { < clipPath~id = " l3cp \int_use:N \g__box_clip_path_int " > }
285   \__kernel_backend_literal_svg:x
286   {
287     <
288     path ~ d =
289     "
290       M ~ 0 ~
291       \dim_to_decimal:n { -\box_dp:N #1 } ~
292       L ~ \dim_to_decimal:n { \box_wd:N #1 } ~
293       \dim_to_decimal:n { -\box_dp:N #1 } ~
294       L ~ \dim_to_decimal:n { \box_wd:N #1 } ~
295       \dim_to_decimal:n { \box_ht:N #1 + \box_dp:N #1 } ~
296       L ~ 0 ~
297       \dim_to_decimal:n { \box_ht:N #1 + \box_dp:N #1 } ~
298       Z
299     "
300   />
301   }
302   \__kernel_backend_literal_svg:n
303   { < /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 it lines up with the current point, as does the T<sub>E</sub>X box.

```

304   \__kernel_backend_scope_begin:n
305   {
306     transform =
307     "
308       translate ( { ?x } , { ?y } ) ~
309       scale ( 1 , -1 )
310     "
311   }
312   \__kernel_backend_scope_begin:x
313   {
314     clip-path =
315     "url ( \c_hash_str l3cp \int_use:N \g__box_clip_path_int ) "
316   }
317   \__kernel_backend_scope_begin:n
318   {
319     transform =
320     "
321       scale ( -1 , 1 ) ~
322       translate ( { ?x } , { ?y } ) ~
323       scale ( -1 , -1 )

```

```

324         "
325     }
326     \box_use:N #1
327     \__kernel_backend_scope_end:
328     \__kernel_backend_scope_end:
329     \__kernel_backend_scope_end:
330 %     \skip_horizontal:n { \box_wd:N #1 }
331 }
332 \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.

```

333 \cs_new_protected:Npn \__box_backend_rotate:Nn #1#2
334 {
335     \__kernel_backend_scope_begin:x
336     {
337         transform =
338         "
339             rotate
340             ( \fp_eval:n { round ( -(#2) , 5 ) } , ~ { ?x } , ~ { ?y } )
341         "
342     }
343     \box_use:N #1
344     \__kernel_backend_scope_end:
345 }

```

(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).

```

346 \cs_new_protected:Npn \__box_backend_scale:Nnn #1#2#3
347 {
348     \__kernel_backend_scope_begin:x
349     {
350         transform =
351         "
352             translate ( { ?x } , { ?y } ) ~
353             scale
354             (
355                 \fp_eval:n { round ( -#2 , 5 ) } ,
356                 \fp_eval:n { round ( -#3 , 5 ) }
357             ) ~
358             translate ( { ?x } , { ?y } ) ~
359             scale ( -1 )
360         "
361     }
362     \hbox_overlap_right:n { \box_use:N #1 }
363     \__kernel_backend_scope_end:
364 }

```

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

```
365 </dvisvgm>
366 </initex | package>
```

### 3 I3backend-color Implementation

```
367 <*initex | package>
368 <@@=color>
```

Color support is split into two parts: a “general” concept and one directly linked to drawings (or rather the split between filling and stroking). General color is relatively easy to handle: we have a color stack available with all modern drivers, and can use that. Whilst (x)dvipdfmx does have its own approach to color specials, it is easier to use dvips-like ones for all cases except direct PDF output.

#### 3.1 dvips-style

```
369 <*dvisvgm | dvipdfmx | dvips | xdvipdfmx>
```

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).

```
370 <*package>
371 \cs_new_protected:Npn \_color_backend_pickup:N #1 { }
372 \AtBeginDocument
373 {
374   \cs_if_exist:cT { ver@color.sty }
375   {
376     \cs_set_protected:Npn \_color_backend_pickup:N #1
377     {
378       \exp_args:NV \tl_if_head_is_space:nTF \current@color
379       {
380         \tl_set:Nx #1
381         {
382           spot ~
383           \exp_after:wN \use:n \current@color \c_space_tl 1
384         }
385       }
386       {
387         \exp_last_unbraced:Nx \_color_backend_pickup:w
388         { \current@color } \q_stop #1
389       }
390     }
391     \cs_new_protected:Npn \_color_backend_pickup:w #1 ~ #2 \q_stop #3
392     { \tl_set:Nn #3 { #1 ~ #2 } }
393   }
394 }
395 </package>
```

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

`\_color_backend_cmyk:nnnn` Push the data to the stack. In the case of dvips also reset the drawing fill color in raw PostScript.

```
396 \cs_new_protected:Npn \_color_backend_cmyk:nnnn #1#2#3#4
397 {
```

```
\_color_backend_gray:n
\_color_backend_rgb:nnn
\_color_backend_spot:nn
\_color_backend_select:n
\_color_backend_select:x
\_color_backend_reset:
color.fc
```

```

398   \_color_backend_select:x
399   {
400     cmyk~
401     \fp_eval:n {#1} ~ \fp_eval:n {#2} ~
402     \fp_eval:n {#3} ~ \fp_eval:n {#4}
403   }
404 }
405 \cs_new_protected:Npn \_color_backend_gray:n #1
406 { \_color_backend_select:x { gray~ \fp_eval:n {#1} } }
407 \cs_new_protected:Npn \_color_backend_rgb:nnn #1#2#3
408 {
409   \_color_backend_select:x
410   { rgb~ \fp_eval:n {#1} ~ \fp_eval:n {#2} ~ \fp_eval:n {#3} }
411 }
412 \cs_new_protected:Npn \_color_backend_spot:nn #1#2
413 { \_color_backend_select:n { #1 } }
414 \cs_new_protected:Npn \_color_backend_select:n #1
415 {
416   \_kernel_backend_literal:n { color-push~ #1 }
417   <*dvips>
418   \_kernel_backend_postscript:n { /color.fc~{ }~def }
419   </dvips>
420   \group_insert_after:N \_color_backend_reset:
421 }
422 \cs_generate_variant:Nn \_color_backend_select:n { x }
423 \cs_new_protected:Npn \_color_backend_reset:
424 { \_kernel_backend_literal:n { color~pop } }

```

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

```

425 </divisgm | dvipdfmx | dvips | xdvipdfmx>

```

## 3.2 pdfmode

```

426 <*pdfmode>

```

\\_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

```

427 <*package>
428 \cs_new_protected:Npn \_color_backend_pickup:N #1 { }
429 \AtBeginDocument
430 {
431   \cs_if_exist:cT { ver@color.sty }
432   {
433     \cs_set_protected:Npn \_color_backend_pickup:N #1
434     {
435       \exp_last_unbraced:Nx \_color_backend_pickup:w
436       { \current@color } ~ 0 ~ 0 ~ 0 \q_stop #1
437     }
438     \cs_new_protected:Npn \_color_backend_pickup:w
439     #1 ~ #2 ~ #3 ~ #4 ~ #5 ~ #6 \q_stop #7
440     {

```

```

441 \str_if_eq:nnTF {#2} { g }
442 { \tl_set:Nn #7 { gray ~ #1 } }
443 {
444 \str_if_eq:nnTF {#4} { rg }
445 { \tl_set:Nn #7 { rgb ~ #1 ~ #2 ~ #3 } }
446 {
447 \str_if_eq:nnTF {#5} { k }
448 { \tl_set:Nn #7 { cmyk ~ #1 ~ #2 ~ #3 ~ #4 } }
449 {
450 \str_if_eq:nnTF {#2} { cs }
451 {
452 \tl_set:Nx #7 { spot ~ \use_none:n #1 ~ #5 }
453 }
454 {
455 \tl_set:Nn #7 { gray ~ 0 }
456 }
457 }
458 }
459 }
460 }
461 }
462 }
463 </package>

```

(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.

```

464 \int_new:N \l__kernel_color_stack_int

```

(End definition for `\l__kernel_color_stack_int`.)

`\__color_backend_cmyk:nnnn` Simply dump the data, but allowing for LuaTeX.

```

\__color_backend_cmyk_aux:nnnn
\__color_backend_gray:n
\__color_backend_gray_aux:n
\__color_backend_rgb:nnn
\__color_backend_rgb_aux:nnn
\__color_backend_spot:nn
\__color_backend_select:n
\__color_backend_select:x
\__color_backend_reset:
465 \cs_new_protected:Npn \__color_backend_cmyk:nnnn #1#2#3#4
466 {
467 \use:x
468 {
469 \__color_backend_cmyk_aux:nnnn
470 { \fp_eval:n {#1} }
471 { \fp_eval:n {#2} }
472 { \fp_eval:n {#3} }
473 { \fp_eval:n {#4} }
474 }
475 }
476 \cs_new_protected:Npn \__color_backend_cmyk_aux:nnnn #1#2#3#4
477 {
478 \__color_backend_select:n
479 { #1 ~ #2 ~ #3 ~ #4 ~ k ~ #1 ~ #2 ~ #3 ~ #4 ~ K }
480 }
481 \cs_new_protected:Npn \__color_backend_gray:n #1
482 { \exp_args:Nx \__color_backend_gray_aux:n { \fp_eval:n {#1} } }
483 \cs_new_protected:Npn \__color_backend_gray_aux:n #1
484 { \__color_backend_select:n { #1 ~ g ~ #1 ~ G } }
485 \cs_new_protected:Npn \__color_backend_rgb:nnn #1#2#3

```

```

486 {
487   \use:x
488   {
489     \_color_backend_rgb_aux:nnn
490     { \fp_eval:n {#1} }
491     { \fp_eval:n {#2} }
492     { \fp_eval:n {#3} }
493   }
494 }
495 \cs_new_protected:Npn \_color_backend_rgb_aux:nnn #1#2#3
496 { \_color_backend_select:n { #1 ~ #2 ~ #3 ~ rg ~ #1 ~ #2 ~ #3 ~ RG } }
497 \cs_new_protected:Npn \_color_backend_spot:nn #1#2
498 { \_color_backend_select:n { /#1 ~ cs ~ /#1 ~ CS ~ #2 ~ sc ~ #2 ~ SC } }
499 \cs_new_protected:Npx \_color_backend_select:n #1
500 {
501   \cs_if_exist:NTF \tex_pdfextension:D
502   { \tex_pdfextension:D colorstack }
503   { \tex_pdfcolorstack:D }
504   \exp_not:N \l__kernel_color_stack_int push {#1}
505   \group_insert_after:N \exp_not:N \_color_backend_reset:
506 }
507 \cs_generate_variant:Nn \_color_backend_select:n { x }
508 \cs_new_protected:Npx \_color_backend_reset:
509 {
510   \cs_if_exist:NTF \tex_pdfextension:D
511   { \tex_pdfextension:D colorstack }
512   { \tex_pdfcolorstack:D }
513   \exp_not:N \l__kernel_color_stack_int pop \scan_stop:
514 }

```

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

```

515 </pdfmode>
516 </initex | package>

```

## 4 I3backend-draw Implementation

```

517 <*initex | package>
518 <@@=draw>

```

### 4.1 dvips backend

```

519 <*dvips>

```

\\_draw\_backend\_literal:n The same as literal PostScript: same arguments about positioning apply her.

```

520 \cs_new_eq:NN \_draw_backend_literal:n \_kernel_backend_literal_postscript:n
521 \cs_generate_variant:Nn \_draw_backend_literal:n { x }

```

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

\\_draw\_backend\_begin: The ps::[begin] special here deals with positioning but allows us to continue on to a matching ps::[end]: contrast with ps:, which positions but where we can't split material between separate calls. The @beginspecial/@endspecial pair are from special.pro and correct the scale and y-axis direction. The definition of /color.fc deals with fill color in paths. In contrast to pgf, we don't save the current point: discussion with

```

\_draw_backend_end:
color.fc

```

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.

```

522 \cs_new_protected:Npn \__draw_backend_begin:
523 {
524   \__kernel_backend_literal:n { ps::[begin] }
525   \__draw_backend_literal:n { @beginspecial }
526   \__draw_backend_literal:n { SDict ~ begin ~ /color.fc ~ { } ~ def ~ end }
527 }
528 \cs_new_protected:Npn \__draw_backend_end:
529 {
530   \__draw_backend_literal:n { @endspecial }
531   \__kernel_backend_literal:n { ps::[end] }
532 }

```

(End definition for `\__draw_backend_begin:`, `\__draw_backend_end:`, and `color.fc`. This function is documented on page ??.)

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

```

533 \cs_new_protected:Npn \__draw_backend_scope_begin:
534 { \__draw_backend_literal:n { save } }
535 \cs_new_protected:Npn \__draw_backend_scope_end:
536 { \__draw_backend_literal:n { restore } }

```

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

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

```

537 \cs_new_protected:Npn \__draw_backend_moveto:nn #1#2
538 {
539   \__draw_backend_literal:x
540   {
541     \dim_to_decimal_in_bp:n {#1} ~
542     \dim_to_decimal_in_bp:n {#2} ~ moveto
543   }
544 }
545 \cs_new_protected:Npn \__draw_backend_lineto:nn #1#2
546 {
547   \__draw_backend_literal:x
548   {
549     \dim_to_decimal_in_bp:n {#1} ~
550     \dim_to_decimal_in_bp:n {#2} ~ lineto
551   }
552 }
553 \cs_new_protected:Npn \__draw_backend_rectangle:nnnn #1#2#3#4
554 {
555   \__draw_backend_literal:x
556   {
557     \dim_to_decimal_in_bp:n {#4} ~ \dim_to_decimal_in_bp:n {#3} ~

```



```

558         \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~
559         moveto~dup~0~rlineto~exch~0~exch~rlineto~neg~0~rlineto~closepath
560     }
561 }
562 \cs_new_protected:Npn \__draw_backend_curveto:nnnnnn #1#2#3#4#5#6
563 {
564     \__draw_backend_literal:x
565     {
566         \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~
567         \dim_to_decimal_in_bp:n {#3} ~ \dim_to_decimal_in_bp:n {#4} ~
568         \dim_to_decimal_in_bp:n {#5} ~ \dim_to_decimal_in_bp:n {#6} ~
569         curveto
570     }
571 }

```

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

```

\__draw_backend_evenodd_rule: The even-odd rule here can be implemented as a simply switch.
\__draw_backend_nonzero_rule:
\g__draw_draw_eor_bool
572 \cs_new_protected:Npn \__draw_backend_evenodd_rule:
573 { \bool_gset_true:N \g__draw_draw_eor_bool }
574 \cs_new_protected:Npn \__draw_backend_nonzero_rule:
575 { \bool_gset_false:N \g__draw_draw_eor_bool }
576 \bool_new:N \g__draw_draw_eor_bool

```

(End definition for \\_\_draw\_backend\_evenodd\_rule:, \\_\_draw\_backend\_nonzero\_rule:, and \g\_\_draw\_draw\_eor\_bool.)

\\_\_draw\_backend\_closepath: Unlike PDF, PostScript doesn't track separate colors for strokes and other elements. It is also desirable to have the clip keyword after a stroke or fill. To achieve those outcomes, there is some work to do. For color, the stroke color is simple but the fill one has to be inserted by hand. For clipping, the required ordering is achieved using a T<sub>E</sub>X switch. All of the operations end with a new path instruction as they do not terminate (again in contrast to PDF).

```

\__draw_backend_closepath:
\__draw_backend_stroke:
\__draw_backend_closestroke:
\__draw_backend_fill:
\__draw_backend_fillstroke:
\__draw_backend_clip:
\__draw_backend_discardpath:
\g__draw_draw_clip_bool
577 \cs_new_protected:Npn \__draw_backend_closepath:
578 { \__draw_backend_literal:n { closepath } }
579 \cs_new_protected:Npn \__draw_backend_stroke:
580 {
581     \__draw_backend_literal:n { stroke }
582     \bool_if:NT \g__draw_draw_clip_bool
583     {
584         \__draw_backend_literal:x
585         {
586             \bool_if:NT \g__draw_draw_eor_bool { eo }
587             clip
588         }
589     }
590     \__draw_backend_literal:n { newpath }
591     \bool_gset_false:N \g__draw_draw_clip_bool
592 }
593 \cs_new_protected:Npn \__draw_backend_closestroke:
594 {
595     \__draw_backend_closepath:
596     \__draw_backend_stroke:
597 }

```

```

598 \cs_new_protected:Npn \__draw_backend_fill:
599 {
600   \__draw_backend_literal:n { gsave }
601   \__draw_backend_literal:n { color.fc }
602   \__draw_backend_literal:x
603   {
604     \bool_if:NT \g__draw_draw_eor_bool { eo }
605     fill
606   }
607   \__draw_backend_literal:n { grestore }
608   \bool_if:NT \g__draw_draw_clip_bool
609   {
610     \__draw_backend_literal:x
611     {
612       \bool_if:NT \g__draw_draw_eor_bool { eo }
613       clip
614     }
615   }
616   \__draw_backend_literal:n { newpath }
617   \bool_gset_false:N \g__draw_draw_clip_bool
618 }
619 \cs_new_protected:Npn \__draw_backend_fillstroke:
620 {
621   \__draw_backend_literal:n { gsave }
622   \__draw_backend_literal:n { color.fc }
623   \__draw_backend_literal:x
624   {
625     \bool_if:NT \g__draw_draw_eor_bool { eo }
626     fill
627   }
628   \__draw_backend_literal:n { grestore }
629   \__draw_backend_literal:n { stroke }
630   \bool_if:NT \g__draw_draw_clip_bool
631   {
632     \__draw_backend_literal:x
633     {
634       \bool_if:NT \g__draw_draw_eor_bool { eo }
635       clip
636     }
637   }
638   \__draw_backend_literal:n { newpath }
639   \bool_gset_false:N \g__draw_draw_clip_bool
640 }
641 \cs_new_protected:Npn \__draw_backend_clip:
642 { \bool_gset_true:N \g__draw_draw_clip_bool }
643 \bool_new:N \g__draw_draw_clip_bool
644 \cs_new_protected:Npn \__draw_backend_discardpath:
645 {
646   \bool_if:NT \g__draw_draw_clip_bool
647   {
648     \__draw_backend_literal:x
649     {
650       \bool_if:NT \g__draw_draw_eor_bool { eo }
651       clip

```

```

652     }
653   }
654   \__draw_backend_literal:n { newpath }
655   \bool_gset_false:N \g__draw_draw_clip_bool
656 }

```

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

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

```

657 \cs_new_protected:Npn \__draw_backend_dash_pattern:nn #1#2
658 {
659   \__draw_backend_literal:x
660   {
661     [
662       \exp_args:Nf \use:n
663       { \clist_map_function:nN {#1} \__draw_backend_dash:n }
664     ] ~
665     \dim_to_decimal_in_bp:n {#2} ~ setdash
666   }
667 }
668 \cs_new:Npn \__draw_backend_dash:n #1
669 { ~ \dim_to_decimal_in_bp:n {#1} }
670 \cs_new_protected:Npn \__draw_backend_linewidth:n #1
671 {
672   \__draw_backend_literal:x
673   { \dim_to_decimal_in_bp:n {#1} ~ setlinewidth }
674 }
675 \cs_new_protected:Npn \__draw_backend_miterlimit:n #1
676 { \__draw_backend_literal:x { \fp_eval:n {#1} ~ setmiterlimit } }
677 \cs_new_protected:Npn \__draw_backend_cap_but:
678 { \__draw_backend_literal:n { 0 ~ setlinecap } }
679 \cs_new_protected:Npn \__draw_backend_cap_round:
680 { \__draw_backend_literal:n { 1 ~ setlinecap } }
681 \cs_new_protected:Npn \__draw_backend_cap_rectangle:
682 { \__draw_backend_literal:n { 2 ~ setlinecap } }
683 \cs_new_protected:Npn \__draw_backend_join_miter:
684 { \__draw_backend_literal:n { 0 ~ setlinejoin } }
685 \cs_new_protected:Npn \__draw_backend_join_round:
686 { \__draw_backend_literal:n { 1 ~ setlinejoin } }
687 \cs_new_protected:Npn \__draw_backend_join_bevel:
688 { \__draw_backend_literal:n { 2 ~ setlinejoin } }

```

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

For dvips, we can use the standard color stack to deal with stroke color, but for fills have to switch to raw PostScript. This is thus not handled by the stack, but the context is very restricted. See also how fills are implemented.

```

689 \cs_new_protected:Npn \__draw_backend_color_fill_cmyk:nnnn #1#2#3#4
690 {
691   \__draw_backend_color_fill:x
692   {
693     \fp_eval:n {#1} ~ \fp_eval:n {#2} ~
694     \fp_eval:n {#3} ~ \fp_eval:n {#4} ~
695     setcmykcolor

```

```

696     }
697   }
698   \cs_new_protected:Npn \__draw_backend_color_stroke_cmyk:nnnn #1#2#3#4
699   {
700     \__draw_backend_color_stroke:x
701     {
702       cmyk ~
703       \fp_eval:n {#1} ~ \fp_eval:n {#2} ~
704       \fp_eval:n {#3} ~ \fp_eval:n {#4}
705     }
706   }
707   \cs_new_protected:Npn \__draw_backend_color_fill_gray:n #1
708   { \__draw_backend_color_fill:x { \fp_eval:n {#1} ~ setgray } }
709   \cs_new_protected:Npn \__draw_backend_color_stroke_gray:n #1
710   { \__draw_backend_color_stroke:x { gray ~ \fp_eval:n {#1} } }
711   \cs_new_protected:Npn \__draw_backend_color_fill_rgb:nnn #1#2#3
712   {
713     \__draw_backend_color_fill:x
714     { \fp_eval:n {#1} ~ \fp_eval:n {#2} ~ \fp_eval:n {#3} ~ setrgbcolor }
715   }
716   \cs_new_protected:Npn \__draw_backend_color_stroke_rgb:nnn #1#2#3
717   {
718     \__draw_backend_color_stroke:x
719     { rgb ~ \fp_eval:n {#1} ~ \fp_eval:n {#2} ~ \fp_eval:n {#3} }
720   }
721   \cs_new_protected:Npn \__draw_backend_color_fill:n #1
722   {
723     \__kernel_backend_postscript:n
724     { /color.fc ~ { #1 } ~ def }
725   }
726   \cs_generate_variant:Nn \__draw_backend_color_fill:n { x }
727   \cs_new_protected:Npn \__draw_backend_color_stroke:n #1
728   {
729     \__kernel_backend_literal:n { color~push~#1 }
730     \group_insert_after:N \__draw_color_reset:
731   }
732   \cs_generate_variant:Nn \__draw_backend_color_stroke:n { x }

```

(End definition for \\_\_draw\_backend\_color\_fill\_cmyk:nnnn 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. (x)dvipdfmx). Thus we take the shortest path available and simply dump the matrix as given.

```

733   \cs_new_protected:Npn \__draw_backend_cm:nnnn #1#2#3#4
734   {
735     \__draw_backend_literal:n
736     {
737       [
738         \fp_eval:n {#1} ~ \fp_eval:n {#2} ~
739         \fp_eval:n {#3} ~ \fp_eval:n {#4} ~
740         0 ~ 0
741       ] ~
742     concat

```

```

743     }
744 }

```

(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).

```

745 \cs_new_protected:Npn \_draw_backend_box_use:Nnnnn #1#2#3#4#5
746 {
747   \_draw_backend_literal:n { @endspecial }
748   \_draw_backend_literal:n { [end] }
749   \_draw_backend_literal:n { [begin] }
750   \_draw_backend_literal:n { save }
751   \_draw_backend_literal:n { currentpoint }
752   \_draw_backend_literal:n { currentpoint~translate }
753   \_draw_backend_cm:nnnn { 1 } { 0 } { 0 } { -1 }
754   \_draw_backend_cm:nnnn {#2} {#3} {#4} {#5}
755   \_draw_backend_cm:nnnn { 1 } { 0 } { 0 } { -1 }
756   \_draw_backend_literal:n { neg~exch~neg~exch~translate }
757   \_draw_backend_literal:n { [end] }
758   \hbox_overlap_right:n { \box_use:N #1 }
759   \_draw_backend_literal:n { [begin] }
760   \_draw_backend_literal:n { restore }
761   \_draw_backend_literal:n { [end] }
762   \_draw_backend_literal:n { [begin] }
763   \_draw_backend_literal:n { @beginspecial }
764 }

```

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

765 `</dvips>`

## 4.2 pdfmode and (x)dvipdfmx

Both `pdfmode` and `(x)dvipdfmx` directly produce PDF output and understand a shared set of specials for drawing commands.

766 `<*dvipdfmx | pdfmode | xdvipdfmx>`

### 4.2.1 Drawing

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

Pass data through using a dedicated interface.

```

767 \cs_new_eq:NN \_draw_backend_literal:n \_kernel_backend_literal_pdf:n
768 \cs_generate_variant:Nn \_draw_backend_literal:n { x }

```

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

`\_draw_backend_begin:` No special requirements here, so simply set up a drawing scope.

```
\_draw_backend_end:
769 \cs_new_protected:Npn \_draw_backend_begin:
770 { \_draw_backend_scope_begin: }
771 \cs_new_protected:Npn \_draw_backend_end:
772 { \_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:
773 \cs_new_eq:NN \_draw_backend_scope_begin: \_kernel_backend_scope_begin:
774 \cs_new_eq:NN \_draw_backend_scope_end: \_kernel_backend_scope_end:
```

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

`\_draw_backend_moveto:nn` Path creation operations all resolve directly to PDF primitive steps, with only the need to convert to bp.

```
\_draw_backend_lineto:nn
\_draw_backend_curveto:nnnnnn
\_draw_backend_rectangle:nnnn
775 \cs_new_protected:Npn \_draw_backend_moveto:nn #1#2
776 {
777   \_draw_backend_literal:x
778   { \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~ m }
779 }
780 \cs_new_protected:Npn \_draw_backend_lineto:nn #1#2
781 {
782   \_draw_backend_literal:x
783   { \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~ l }
784 }
785 \cs_new_protected:Npn \_draw_backend_curveto:nnnnnn #1#2#3#4#5#6
786 {
787   \_draw_backend_literal:x
788   {
789     \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~
790     \dim_to_decimal_in_bp:n {#3} ~ \dim_to_decimal_in_bp:n {#4} ~
791     \dim_to_decimal_in_bp:n {#5} ~ \dim_to_decimal_in_bp:n {#6} ~
792     c
793   }
794 }
795 \cs_new_protected:Npn \_draw_backend_rectangle:nnnn #1#2#3#4
796 {
797   \_draw_backend_literal:x
798   {
799     \dim_to_decimal_in_bp:n {#1} ~ \dim_to_decimal_in_bp:n {#2} ~
800     \dim_to_decimal_in_bp:n {#3} ~ \dim_to_decimal_in_bp:n {#4} ~
801     re
802   }
803 }
```

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

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

```
\_draw_backend_nonzero_rule:
\g__draw_draw_eor_bool
804 \cs_new_protected:Npn \_draw_backend_evenodd_rule:
805 { \bool_gset_true:N \g__draw_draw_eor_bool }
806 \cs_new_protected:Npn \_draw_backend_nonzero_rule:
807 { \bool_gset_false:N \g__draw_draw_eor_bool }
808 \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:`.)

<code>\__draw_backend_closepath:</code>	Converting paths to output is again a case of mapping directly to PDF operations.
<code>\__draw_backend_stroke:</code>	809 <code>\cs_new_protected:Npn \__draw_backend_closepath:</code>
<code>\__draw_backend_closestroke:</code>	810 <code>{ \__draw_backend_literal:n { h } }</code>
<code>\__draw_backend_fill:</code>	811 <code>\cs_new_protected:Npn \__draw_backend_stroke:</code>
<code>\__draw_backend_fillstroke:</code>	812 <code>{ \__draw_backend_literal:n { S } }</code>
<code>\__draw_backend_clip:</code>	813 <code>\cs_new_protected:Npn \__draw_backend_closestroke:</code>
<code>\__draw_backend_discardpath:</code>	814 <code>{ \__draw_backend_literal:n { s } }</code>
	815 <code>\cs_new_protected:Npn \__draw_backend_fill:</code>
	816 <code>{</code>
	817 <code>\__draw_backend_literal:x</code>
	818 <code>{ f \bool_if:NT \g__draw_draw_eor_bool * }</code>
	819 <code>}</code>
	820 <code>\cs_new_protected:Npn \__draw_backend_fillstroke:</code>
	821 <code>{</code>
	822 <code>\__draw_backend_literal:x</code>
	823 <code>{ B \bool_if:NT \g__draw_draw_eor_bool * }</code>
	824 <code>}</code>
	825 <code>\cs_new_protected:Npn \__draw_backend_clip:</code>
	826 <code>{</code>
	827 <code>\__draw_backend_literal:x</code>
	828 <code>{ W \bool_if:NT \g__draw_draw_eor_bool * }</code>
	829 <code>}</code>
	830 <code>\cs_new_protected:Npn \__draw_backend_discardpath:</code>
	831 <code>{ \__draw_backend_literal:n { n } }</code>

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

<code>\__draw_backend_dash_pattern:nn</code>	Converting paths to output is again a case of mapping directly to PDF operations.
<code>\__draw_backend_dash:n</code>	832 <code>\cs_new_protected:Npn \__draw_backend_dash_pattern:nn #1#2</code>
<code>\__draw_backend_linewidth:n</code>	833 <code>{</code>
<code>\__draw_backend_miterlimit:n</code>	834 <code>\__draw_backend_literal:x</code>
<code>\__draw_backend_cap_butt:</code>	835 <code>{</code>
<code>\__draw_backend_cap_round:</code>	836 <code>[</code>
<code>\__draw_backend_cap_rectangle:</code>	837 <code>\exp_args:Nf \use:n</code>
<code>\__draw_backend_join_miter:</code>	838 <code>{ \clist_map_function:nN {#1} \__draw_backend_dash:n }</code>
<code>\__draw_backend_join_round:</code>	839 <code>] ~</code>
<code>\__draw_backend_join_bevel:</code>	840 <code>\dim_to_decimal_in_bp:n {#2} ~ d</code>
	841 <code>}</code>
	842 <code>}</code>
	843 <code>\cs_new:Npn \__draw_backend_dash:n #1</code>
	844 <code>{ ~ \dim_to_decimal_in_bp:n {#1} }</code>
	845 <code>\cs_new_protected:Npn \__draw_backend_linewidth:n #1</code>
	846 <code>{</code>
	847 <code>\__draw_backend_literal:x</code>
	848 <code>{ \dim_to_decimal_in_bp:n {#1} ~ w }</code>
	849 <code>}</code>
	850 <code>\cs_new_protected:Npn \__draw_backend_miterlimit:n #1</code>
	851 <code>{ \__draw_backend_literal:x { \fp_eval:n {#1} ~ M } }</code>
	852 <code>\cs_new_protected:Npn \__draw_backend_cap_butt:</code>
	853 <code>{ \__draw_backend_literal:n { 0 ~ J } }</code>
	854 <code>\cs_new_protected:Npn \__draw_backend_cap_round:</code>

```

855 { \_draw_backend_literal:n { 1 ~ J } }
856 \cs_new_protected:Npn \_draw_backend_cap_rectangle:
857 { \_draw_backend_literal:n { 2 ~ J } }
858 \cs_new_protected:Npn \_draw_backend_join_miter:
859 { \_draw_backend_literal:n { 0 ~ j } }
860 \cs_new_protected:Npn \_draw_backend_join_round:
861 { \_draw_backend_literal:n { 1 ~ j } }
862 \cs_new_protected:Npn \_draw_backend_join_bevel:
863 { \_draw_backend_literal:n { 2 ~ j } }

```

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

Color has to be split between (x)dvipdfmx and the PDF engines as there is no color stack for fill/stroke separation in the former.

```

\_draw_backend_color_fill_cmyk:nnnn
\_draw_backend_color_stroke_cmyk:nnnn
  \_draw_backend_color_fill_gray:n
  \_draw_backend_color_stroke_gray:n
  \_draw_backend_color_fill_rgb:nnn
  \_draw_backend_color_stroke_rgb:nnn
    \_draw_backend_color_select:n
    \_draw_backend_color_select:x
\_draw_backend_color_reset:
864 \cs_new_protected:Npn \_draw_backend_color_fill_cmyk:nnnn #1#2#3#4
865 {
866   \_draw_backend_color_select:x
867   {
868     \fp_eval:n {#1} ~ \fp_eval:n {#2} ~
869     \fp_eval:n {#3} ~ \fp_eval:n {#4} ~
870     k
871   }
872 }
873 \cs_new_protected:Npn \_draw_backend_color_stroke_cmyk:nnnn #1#2#3#4
874 {
875   \_draw_backend_color_select:x
876   {
877     \fp_eval:n {#1} ~ \fp_eval:n {#2} ~
878     \fp_eval:n {#3} ~ \fp_eval:n {#4} ~
879     k
880   }
881 }
882 \cs_new_protected:Npn \_draw_backend_color_fill_gray:n #1
883 { \_draw_backend_color_select:x { \fp_eval:n {#1} ~ g } }
884 \cs_new_protected:Npn \_draw_backend_color_stroke_gray:n #1
885 { \_draw_backend_color_select:x { \fp_eval:n {#1} ~ G } }
886 \cs_new_protected:Npn \_draw_backend_color_fill_rgb:nnn #1#2#3
887 {
888   \_draw_backend_color_select:x
889   { \fp_eval:n {#1} ~ \fp_eval:n {#2} ~ \fp_eval:n {#3} ~ rg }
890 }
891 \cs_new_protected:Npn \_draw_backend_color_stroke_rgb:nnn #1#2#3
892 {
893   \_draw_backend_color_select:x
894   { \fp_eval:n {#1} ~ \fp_eval:n {#2} ~ \fp_eval:n {#3} ~ RG }
895 }
896 <*pdfmode>
897 \cs_new_protected:Npx \_draw_backend_color_select:n #1
898 {
899   \cs_if_exist:NTF \tex_pdfextension:D
900   { \tex_pdfextension:D colorstack }
901   { \tex_pdfcolorstack:D }
902   \exp_not:N \l__kernel_color_stack_int push {#1}
903   \group_insert_after:N \exp_not:N \_draw_backend_color_reset:

```



```

904 }
905 \cs_new_protected:Npx \__draw_backend_color_reset:
906 {
907   \cs_if_exist:NTF \tex_pdfextension:D
908   { \tex_pdfextension:D colorstack }
909   { \tex_pdfcolorstack:D }
910   \exp_not:N \l__kernel_color_stack_int pop \scan_stop:
911 }
912 </pdfmode>
913 <*dvipdfmx | xdvipdfmx>
914 \cs_new_eq:NN \__draw_backend_color_select:n \__kernel_backend_literal_pdf:n
915 </dvipdfmx | xdvipdfmx>
916 \cs_generate_variant:Nn \__draw_backend_color_select:n { x }

```

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

\\_\_draw\_backend\_cm:nnnn  
\\_\_draw\_backend\_cm\_aux:nnnn

Another split here between pdfmode and (x)dvipdfmx. In the former, we have a direct method to maintain alignment: the backend can use a matrix itself. For (x)dvipdfmx, 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 (x)dvipdfmx, but as a matched pair so not suitable for the “stand alone” transformation set up here.)

```

917 \cs_new_protected:Npn \__draw_backend_cm:nnnn #1#2#3#4
918 {
919 <*pdfmode>
920   \__kernel_backend_matrix:x
921   {
922     \fp_eval:n {#1} ~ \fp_eval:n {#2} ~
923     \fp_eval:n {#3} ~ \fp_eval:n {#4}
924   }
925 </pdfmode>
926 <*dvipdfmx | xdvipdfmx>
927   \__draw_backend_cm_decompose:nnnnN {#1} {#2} {#3} {#4}
928   \__draw_backend_cm_aux:nnnn
929 </dvipdfmx | xdvipdfmx>
930 }
931 <*dvipdfmx | xdvipdfmx>
932 \cs_new_protected:Npn \__draw_backend_cm_aux:nnnn #1#2#3#4
933 {
934   \__kernel_backend_literal:x
935   {
936     x:rotate~
937     \fp_compare:nNnTF {#1} = \c_zero_fp
938     { 0 }
939     { \fp_eval:n { round ( -#1 , 5 ) } }
940   }
941   \__kernel_backend_literal:x
942   {
943     x:scale~
944     \fp_eval:n { round ( #2 , 5 ) } ~
945     \fp_eval:n { round ( #3 , 5 ) }
946   }
947   \__kernel_backend_literal:x
948   {

```

```

949      x:rotate~
950      \fp_compare:nNnTF {#4} = \c_zero_fp
951      { 0 }
952      { \fp_eval:n { round ( -#4 , 5 ) } }
953    }
954  }
955 </dviptdmtx | xdvipdmtx>

```

(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.

```

956 <*dviptdmtx | xdvipdmtx>
957 \cs_new_protected:Npn \_draw_backend_cm_decompose:nnnnN #1#2#3#4#5
958 {
959   \use:x
960   {
961     \_draw_backend_cm_decompose_auxi:nnnnN
962     { \fp_eval:n { (#1 + #4) / 2 } }
963     { \fp_eval:n { (#1 - #4) / 2 } }
964     { \fp_eval:n { (#3 + #2) / 2 } }
965     { \fp_eval:n { (#3 - #2) / 2 } }
966   }
967   #5
968 }
969 \cs_new_protected:Npn \_draw_backend_cm_decompose_auxi:nnnnN #1#2#3#4#5
970 {
971   \use:x

```

```

972     {
973       \__draw_backend_cm_decompose_auxii:nnnnN
974       { \fp_eval:n { 2 * sqrt ( #1 * #1 + #4 * #4 ) } }
975       { \fp_eval:n { 2 * sqrt ( #2 * #2 + #3 * #3 ) } }
976       { \fp_eval:n { atand ( #3 , #2 ) } }
977       { \fp_eval:n { atand ( #4 , #1 ) } }
978     }
979     #5
980   }
981   \cs_new_protected:Npn \__draw_backend_cm_decompose_auxii:nnnnN #1#2#3#4#5
982   {
983     \use:x
984     {
985       \__draw_backend_cm_decompose_auxiii:nnnnN
986       { \fp_eval:n { ( #4 - #3 ) / 2 } }
987       { \fp_eval:n { ( #1 + #2 ) / 2 } }
988       { \fp_eval:n { ( #1 - #2 ) / 2 } }
989       { \fp_eval:n { ( #4 + #3 ) / 2 } }
990     }
991     #5
992   }
993   \cs_new_protected:Npn \__draw_backend_cm_decompose_auxiii:nnnnN #1#2#3#4#5
994   {
995     \fp_compare:nNnTF { abs ( #2 ) } > { abs ( #3 ) }
996     { #5 {#1} {#2} {#3} {#4} }
997     { #5 {#1} {#3} {#2} {#4} }
998   }
999   </dviPDFmx | xdvipdfmx>

```

(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.

```

1000 \cs_new_protected:Npn \__draw_backend_box_use:Nnnnn #1#2#3#4#5
1001 {
1002   \__kernel_backend_scope_begin:
1003   <*pdfmode>
1004   \__draw_backend_cm:nnnn {#2} {#3} {#4} {#5}
1005   </pdfmode>
1006   <*dviPDFmx | xdvipdfmx>
1007   \__kernel_backend_literal:x
1008   {
1009     pdf:btrans-matrix~
1010     \fp_eval:n {#2} ~ \fp_eval:n {#3} ~
1011     \fp_eval:n {#4} ~ \fp_eval:n {#5} ~
1012     0 ~ 0
1013   }
1014   </dviPDFmx | xdvipdfmx>
1015   \hbox_overlap_right:n { \box_use:N #1 }
1016   <*dviPDFmx | xdvipdfmx>
1017   \__kernel_backend_literal:n { pdf:etrans }

```

```

1018 </dvipdfmx | xdvipdfmx>
1019     \__kernel_backend_scope_end:
1020 }

(End definition for \__draw_backend_box_use:Nnnnn.)

1021 </dvipdfmx | pdfmode | xdvipdfmx>

```

### 4.3 dvisvgm backend

```

1022 <*dvisvgm>

```

The same as the more general literal call.

```

\__draw_backend_literal:n
\__draw_backend_literal:x
1023 \cs_new_eq:NN \__draw_backend_literal:n \__kernel_backend_literal_svg:n
1024 \cs_generate_variant:Nn \__draw_backend_literal:n { x }

```

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

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

```

1025 \cs_new_protected:Npn \__draw_backend_begin:
1026 {
1027     \__draw_backend_scope_begin:
1028     \__draw_backend_scope:n { transform="translate({?x},{?y})~scale(1,-1)" }
1029 }
1030 \cs_new_protected:Npn \__draw_backend_end:
1031 { \__draw_backend_scope_end: }

```

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

\\_\_draw\_backend\_scope\_begin: Several settings that with other backends are “stand alone” have to be given as part of a scope in SVG. As a result, there is a need to provide a mechanism to automatically close these extra scopes. That is done using a dedicated function and a pair of tracking variables. Within each graphics scope we use a global variable to do the work, with a group used to save the value between scopes. The result is that no direct action is needed when creating a scope.

```

1032 \cs_new_protected:Npn \__draw_backend_scope_begin:
1033 {
1034     \int_set_eq:NN
1035     \l__draw_draw_scope_int
1036     \g__draw_draw_scope_int
1037     \group_begin:
1038     \int_gzero:N \g__draw_draw_scope_int
1039 }
1040 \cs_new_protected:Npn \__draw_backend_scope_end:
1041 {
1042     \prg_replicate:nn
1043     { \g__draw_draw_scope_int }
1044     { \__draw_backend_literal:n { </g> } }
1045     \group_end:
1046     \int_gset_eq:NN
1047     \g__draw_draw_scope_int
1048     \l__draw_draw_scope_int
1049 }
1050 \cs_new_protected:Npn \__draw_backend_scope:n #1

```

```

1051 {
1052   \_draw_backend_literal:n { <g~ #1 > }
1053   \int_gincr:N \g__draw_draw_scope_int
1054 }
1055 \cs_generate_variant:Nn \_draw_backend_scope:n { x }
1056 \int_new:N \g__draw_draw_scope_int
1057 \int_new:N \l__draw_draw_scope_int

```

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

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

```

\_draw_backend_lineto:nn
  \_draw_backend_rectangle:nnnn
  \_draw_backend_curveto:nnnnnn
  \_draw_backend_add_to_path:n
\g__draw_draw_path_tl
1058 \cs_new_protected:Npn \_draw_backend_moveto:nn #1#2
1059 {
1060   \_draw_backend_add_to_path:n
1061   { M ~ \dim_to_decimal:n {#1} ~ \dim_to_decimal:n {#2} }
1062 }
1063 \cs_new_protected:Npn \_draw_backend_lineto:nn #1#2
1064 {
1065   \_draw_backend_add_to_path:n
1066   { L ~ \dim_to_decimal:n {#1} ~ \dim_to_decimal:n {#2} }
1067 }
1068 \cs_new_protected:Npn \_draw_backend_rectangle:nnnn #1#2#3#4
1069 {
1070   \_draw_backend_add_to_path:n
1071   {
1072     M ~ \dim_to_decimal:n {#1} ~ \dim_to_decimal:n {#2}
1073     h ~ \dim_to_decimal:n {#3} ~
1074     v ~ \dim_to_decimal:n {#4} ~
1075     h ~ \dim_to_decimal:n { -#3 } ~
1076     Z
1077   }
1078 }
1079 \cs_new_protected:Npn \_draw_backend_curveto:nnnnnn #1#2#3#4#5#6
1080 {
1081   \_draw_backend_add_to_path:n
1082   {
1083     C ~
1084     \dim_to_decimal:n {#1} ~ \dim_to_decimal:n {#2} ~
1085     \dim_to_decimal:n {#3} ~ \dim_to_decimal:n {#4} ~
1086     \dim_to_decimal:n {#5} ~ \dim_to_decimal:n {#6}
1087   }
1088 }
1089 \cs_new_protected:Npn \_draw_backend_add_to_path:n #1
1090 {
1091   \tl_gset:Nx \g__draw_draw_path_tl
1092   {
1093     \g__draw_draw_path_tl
1094     \tl_if_empty:NF \g__draw_draw_path_tl { \c_space_tl }
1095     #1
1096   }

```

```

1097 }
1098 \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:
1099 \cs_new_protected:Npn \__draw_backend_evenodd_rule:
1100 { \__draw_backend_scope:n { fill-rule="evenodd" } }
1101 \cs_new_protected:Npn \__draw_backend_nonzero_rule:
1102 { \__draw_backend_scope:n { fill-rule="nonzero" } }

(End definition for \__draw_backend_evenodd_rule: and \__draw_backend_nonzero_rule:.)

```

```

\__draw_backend_path:n Setting fill and stroke effects and doing clipping all has to be done using scopes. This
\__draw_backend_closepath: means setting up the various requirements in a shared auxiliary which deals with the
\__draw_backend_stroke: bits and pieces. Clipping paths are reused for path drawing: not essential but avoids
\__draw_backend_closestroke: constructing them twice. Discarding a path needs a separate function as it's not quite
\__draw_backend_fill: the same.
\__draw_backend_fillstroke:
1103 \cs_new_protected:Npn \__draw_backend_closepath:
1104 { \__draw_backend_add_to_path:n { Z } }
\__draw_backend_clip:
1105 \cs_new_protected:Npn \__draw_backend_path:n #1
1106 {
1107     \bool_if:NTF \g__draw_draw_clip_bool
1108     {
1109         \int_gincr:N \g__draw_clip_path_int
1110         \__draw_backend_literal:x
1111         {
1112             < clipPath~id = " l3cp \int_use:N \g__draw_clip_path_int " >
1113             { ?nl }
1114             <path~d=" \g__draw_draw_path_tl "/> { ?nl }
1115             < /clipPath > { ? nl }
1116             <
1117             use~xlink:href =
1118             "\c_hash_str l3path \int_use:N \g__draw_path_int " ~
1119             #1
1120             />
1121         }
1122         \__draw_backend_scope:x
1123         {
1124             clip-path =
1125             "url( \c_hash_str l3cp \int_use:N \g__draw_clip_path_int )"
1126         }
1127     }
1128     {
1129         \__draw_backend_literal:x
1130         { <path ~ d=" \g__draw_draw_path_tl " ~ #1 /> }
1131     }
1132     \tl_gclear:N \g__draw_draw_path_tl
1133     \bool_gset_false:N \g__draw_draw_clip_bool
1134 }
1135 \int_new:N \g__draw_path_int
1136 \cs_new_protected:Npn \__draw_backend_stroke:
1137 { \__draw_backend_path:n { style="fill:none" } }
1138 \cs_new_protected:Npn \__draw_backend_closestroke:

```

```

1139 {
1140   \__draw_backend_closepath:
1141   \__draw_backend_stroke:
1142 }
1143 \cs_new_protected:Npn \__draw_backend_fill:
1144 { \__draw_backend_path:n { style="stroke:none" } }
1145 \cs_new_protected:Npn \__draw_backend_fillstroke:
1146 { \__draw_backend_path:n { } }
1147 \cs_new_protected:Npn \__draw_backend_clip:
1148 { \bool_gset_true:N \g__draw_draw_clip_bool }
1149 \bool_new:N \g__draw_draw_clip_bool
1150 \cs_new_protected:Npn \__draw_backend_discardpath:
1151 {
1152   \bool_if:NT \g__draw_draw_clip_bool
1153   {
1154     \int_gincr:N \g__draw_clip_path_int
1155     \__draw_backend_literal:x
1156     {
1157       < clipPath~id = " l3cp \int_use:N \g__draw_clip_path_int " >
1158       { ?nl }
1159       <path~d=" \g__draw_draw_path_tl "/> { ?nl }
1160       < /clipPath >
1161     }
1162     \__draw_backend_scope:x
1163     {
1164       clip-path =
1165       "url( \c_hash_str l3cp \int_use:N \g__draw_clip_path_int)"
1166     }
1167   }
1168   \tl_gclear:N \g__draw_draw_path_tl
1169   \bool_gset_false:N \g__draw_draw_clip_bool
1170 }

```

(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:
1171 \cs_new_protected:Npn \__draw_backend_dash_pattern:nn #1#2
1172 {
1173   \use:x
1174   {
1175     \__draw_backend_dash_aux:nn
1176     { \clist_map_function:nn {#1} \__draw_backend_dash:n }
1177     { \dim_to_decimal:n {#2} }
1178   }
1179 }
1180 \cs_new:Npn \__draw_backend_dash:n #1
1181 { , \dim_to_decimal_in_bp:n {#1} }
1182 \cs_new_protected:Npn \__draw_backend_dash_aux:nn #1#2
1183 {
1184   \__draw_backend_scope:x
1185   {
1186     stroke-dasharray =
1187     "

```

```

1188         \tl_if_empty:oTF { \use_none:n #1 }
1189         { none }
1190         { \use_none:n #1 }
1191     " ~
1192     stroke-offset=" #2 "
1193 }
1194 }
1195 \cs_new_protected:Npn \__draw_backend_linewidth:n #1
1196 { \__draw_backend_scope:x { stroke-width=" \dim_to_decimal:n {#1} " } }
1197 \cs_new_protected:Npn \__draw_backend_miterlimit:n #1
1198 { \__draw_backend_scope:x { stroke-miterlimit=" \fp_eval:n {#1} " } }
1199 \cs_new_protected:Npn \__draw_backend_cap_but:
1200 { \__draw_backend_scope:n { stroke-linecap="butt" } }
1201 \cs_new_protected:Npn \__draw_backend_cap_round:
1202 { \__draw_backend_scope:n { stroke-linecap="round" } }
1203 \cs_new_protected:Npn \__draw_backend_cap_rectangle:
1204 { \__draw_backend_scope:n { stroke-linecap="square" } }
1205 \cs_new_protected:Npn \__draw_backend_join_miter:
1206 { \__draw_backend_scope:n { stroke-linejoin="miter" } }
1207 \cs_new_protected:Npn \__draw_backend_join_round:
1208 { \__draw_backend_scope:n { stroke-linejoin="round" } }
1209 \cs_new_protected:Npn \__draw_backend_join_bevel:
1210 { \__draw_backend_scope:n { stroke-linejoin="bevel" } }

```

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

\\_draw\_backend\_color\_fill\_cmyk:nnnn SVG fill color has to be covered outside of the stack, as for dvips. Here, we are only allowed RGB colors so there is some conversion to do.

```

\_draw_backend_color_fill_cmyk:nnnn 1211 \cs_new_protected:Npn \__draw_backend_color_fill_cmyk:nnnn #1#2#3#4
\_draw_backend_color_stroke_cmyk:nnnn 1212 {
\_draw_backend_color_fill_gray:n 1213     \use:x
\_draw_backend_color_stroke_gray:n 1214     {
\_draw_backend_color_fill_rgb:nnn 1215         \__draw_backend_color_fill:nnn
\_draw_backend_color_stroke_rgb:nnn 1216         { \fp_eval:n { -100 * ( (#1) * ( 1 - (#4) ) - 1 ) } }
\_draw_backend_color_fill:nnn 1217         { \fp_eval:n { -100 * ( (#2) * ( 1 - (#4) ) + #4 - 1 ) } }
1218         { \fp_eval:n { -100 * ( (#3) * ( 1 - (#4) ) + #4 - 1 ) } }
1219     }
1220 }
1221 \cs_new_protected:Npn \__draw_backend_color_stroke_cmyk:nnnn #1#2#3#4
1222 {
1223     \__draw_backend_select:x
1224     {
1225         cmyk~
1226         \fp_eval:n {#1} ~ \fp_eval:n {#2} ~
1227         \fp_eval:n {#3} ~ \fp_eval:n {#4}
1228     }
1229 }
1230 \cs_new_protected:Npn \__draw_backend_color_fill_gray:n #1
1231 {
1232     \use:x
1233     {
1234         \__draw_backend_color_gray_aux:n
1235         { \fp_eval:n { 100 * (#1) } }
1236     }

```



```

1237 }
1238 \cs_new_protected:Npn \__draw_backend_color_gray_aux:n #1
1239 { \__draw_backend_color_fill:nnn {#1} {#1} {#1} }
1240 \cs_new_protected:Npn \__draw_backend_color_stroke_gray:n #1
1241 { \__draw_backend_select:x { gray~ \fp_eval:n {#1} } }
1242 \cs_new_protected:Npn \__draw_backend_color_fill_rgb:nnn #1#2#3
1243 {
1244   \use:x
1245   {
1246     \__draw_backend_color_fill:nnn
1247     { \fp_eval:n { 100 * (#1) } }
1248     { \fp_eval:n { 100 * (#2) } }
1249     { \fp_eval:n { 100 * (#3) } }
1250   }
1251 }
1252 \cs_new_protected:Npn \__draw_backend_color_fill:nnn #1#2#3
1253 {
1254   \__draw_backend_scope:x
1255   {
1256     fill =
1257     "
1258     rgb
1259     (
1260       #1 \c_percent_str ,
1261       #2 \c_percent_str ,
1262       #3 \c_percent_str
1263     )
1264     "
1265   }
1266 }
1267 \cs_new_protected:Npn \__draw_backend_color_stroke_rgb:nnn #1#2#3
1268 {
1269   \__draw_backend_select:x
1270   { rgb~ \fp_eval:n {#1} ~ \fp_eval:n {#2} ~ \fp_eval:n {#3} }
1271 }

```

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

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

```

1272 \cs_new_protected:Npn \__draw_backend_cm:nnnn #1#2#3#4
1273 {
1274   \__draw_backend_scope:n
1275   {
1276     transform =
1277     "
1278     matrix
1279     (
1280       \fp_eval:n {#1} , \fp_eval:n {#2} ,
1281       \fp_eval:n {#3} , \fp_eval:n {#4} ,
1282       Opt , Opt
1283     )
1284     "
1285   }
1286 }

```

(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.

```

1287 \cs_new_protected:Npn \_draw_backend_box_use:Nnnnn #1#2#3#4#5#6#7
1288 {
1289   \_kernel_backend_scope_begin:
1290   \_draw_backend_cm:nnnn {#2} {#3} {#4} {#5}
1291   \_kernel_backend_literal_svg:n
1292   {
1293     < g~
1294     stroke="none"~
1295     transform="scale(-1,1)~translate({?x},{?y})~scale(-1,-1)"
1296     >
1297   }
1298   \box_set_wd:Nn #1 { Opt }
1299   \box_set_ht:Nn #1 { Opt }
1300   \box_set_dp:Nn #1 { Opt }
1301   \box_use:N #1
1302   \_kernel_backend_literal_svg:n { </g> }
1303   \_kernel_backend_scope_end:
1304 }

```

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

```

1305 </dvisvgm>
1306 </initex | package>

```

## 5 l3backend-graphics Implementation

```

1307 <*initex | package>
1308 <@@=graphics>

```

### 5.1 dvips backend

```

1309 <*dvips>

```

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

```

1310 <*initex>
1311 \use:n
1312 </initex>
1313 <*package>
1314 \AtBeginDocument
1315 </package>
1316 { \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.

```

1317 \cs_new_protected:Npn \_graphics_backend_include_eps:n #1
1318 {
1319   \_kernel_backend_literal:x
1320   {
1321     PSfile = #1 \c_space_tl
1322     llx = \dim_to_decimal_in_bp:n \l_graphics_llx_dim \c_space_tl

```

```

1323         lly = \dim_to_decimal_in_bp:n \l_graphics_lly_dim \c_space_tl
1324         urx = \dim_to_decimal_in_bp:n \l_graphics_urx_dim \c_space_tl
1325         ury = \dim_to_decimal_in_bp:n \l_graphics_ury_dim
1326     }
1327 }

(End definition for \__graphics_backend_include_eps:n.)

1328 </dvips>

```

## 5.2 pdfmode backend

```

1329 <(*pdfmode)

```

`\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.

```

1330 \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.

```

1331 \cs_new_protected:Npn \__graphics_backend_getbb_jpg:n #1
1332 {
1333     \int_zero:N \l_graphics_page_int
1334     \tl_clear:N \l_graphics_pagebox_tl
1335     \tl_set:Nx \l__graphics_graphics_attr_tl
1336     {
1337         \tl_if_empty:NF \l_graphics_decodearray_tl
1338         { :D \l_graphics_decodearray_tl }
1339         \bool_if:NT \l_graphics_interpolate_bool
1340         { :I }
1341     }
1342     \tl_clear:N \l__graphics_graphics_attr_tl
1343     \__graphics_backend_getbb_auxi:n {#1}
1344 }
1345 \cs_new_eq:NN \__graphics_backend_getbb_png:n \__graphics_backend_getbb_jpg:n
1346 \cs_new_protected:Npn \__graphics_backend_getbb_pdf:n #1
1347 {
1348     \tl_clear:N \l_graphics_decodearray_tl
1349     \bool_set_false:N \l_graphics_interpolate_bool
1350     \tl_set:Nx \l__graphics_graphics_attr_tl
1351     {
1352         : \l_graphics_pagebox_tl
1353         \int_compare:nNnT \l_graphics_page_int > 1
1354         { :P \int_use:N \l_graphics_page_int }
1355     }
1356     \__graphics_backend_getbb_auxi:n {#1}
1357 }

```

```

1358 \cs_new_protected:Npn \__graphics_backend_getbb_auxi:n #1
1359 {
1360   \graphics_bb_restore:xF { #1 \l__graphics_graphics_attr_tl }
1361   { \__graphics_backend_getbb_auxii:n {#1} }
1362 }
1363 % \begin{macrocode}
1364 % Measuring the graphic is done by boxing up: for PDF graphics we could
1365 % use |\tex_pdfximagebbox:D|, but if doesn't work for other types.
1366 % As the box always starts at $(0,0)$ there is no need to worry about
1367 % the lower-left position.
1368 % \begin{macrocode}
1369 \cs_new_protected:Npn \__graphics_backend_getbb_auxii:n #1
1370 {
1371   \tex_immediate:D \tex_pdfximage:D
1372   \bool_lazy_or:nnT
1373     { \l_graphics_interpolate_bool }
1374     { ! \tl_if_empty_p:N \l_graphics_decodearray_tl }
1375     {
1376       attr ~
1377       {
1378         \tl_if_empty:NF \l_graphics_decodearray_tl
1379         { /Decode~[ \l_graphics_decodearray_tl ] }
1380         \bool_if:NT \l_graphics_interpolate_bool
1381         { /Interpolate~true }
1382       }
1383     }
1384     \int_compare:nNnT \l_graphics_page_int > 0
1385     { page ~ \int_use:N \l_graphics_page_int }
1386     \tl_if_empty:NF \l_graphics_pagebox_tl
1387     { \l_graphics_pagebox_tl }
1388     {#1}
1389     \hbox_set:Nn \l__graphics_internal_box
1390     { \tex_pdfrefximage:D \tex_pdflastximage:D }
1391     \dim_set:Nn \l_graphics_urx_dim { \box_wd:N \l__graphics_internal_box }
1392     \dim_set:Nn \l_graphics_ury_dim { \box_ht:N \l__graphics_internal_box }
1393     \int_const:cn { c__graphics_graphics_#1 \l__graphics_graphics_attr_tl _int }
1394     { \tex_the:D \tex_pdflastximage:D }
1395     \graphics_bb_save:x { #1 \l__graphics_graphics_attr_tl }
1396 }

```

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

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

```

1397 \cs_new_protected:Npn \__graphics_backend_include_jpg:n #1
1398 {
1399   \tex_pdfrefximage:D
1400   \int_use:c { c__graphics_graphics_#1 \l__graphics_graphics_attr_tl _int }
1401 }
1402 \cs_new_eq:NN \__graphics_backend_include_pdf:n \__graphics_backend_include_jpg:n
1403 \cs_new_eq:NN \__graphics_backend_include_png:n \__graphics_backend_include_jpg:n

```

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

```

\__graphics_backend_getbb_eps:n EPS graphics may be included in pdfmode by conversion to PDF: this requires restricted
\__graphics_backend_getbb_eps:nm shell escape. Modelled on the epstopdf LATEX 2ε package, but simplified, conversion takes
\__graphics_backend_include_eps:n place here if we have shell access.

\l__graphics_backend_dir_str 1404 \sys_if_shell:T
\l__graphics_backend_name_str 1405 {
\l__graphics_backend_ext_str 1406 \str_new:N \l__graphics_backend_dir_str
1407 \str_new:N \l__graphics_backend_name_str
1408 \str_new:N \l__graphics_backend_ext_str
1409 \cs_new_protected:Npn \__graphics_backend_getbb_eps:n #1
1410 {
1411 \file_parse_full_name:nNNN {#1}
1412 \l__graphics_backend_dir_str
1413 \l__graphics_backend_name_str
1414 \l__graphics_backend_ext_str
1415 \exp_args:Nx \__graphics_backend_getbb_eps:nn
1416 {
1417 \l__graphics_backend_name_str - \str_tail:N \l__graphics_backend_ext_str
1418 -converted-to.pdf
1419 }
1420 {#1}
1421 }
1422 \cs_new_protected:Npn \__graphics_backend_getbb_eps:nn #1#2
1423 {
1424 \file_compare_timestamp:nNnT {#2} > {#1}
1425 {
1426 \sys_shell_now:n
1427 { repstopdf ~ #2 ~ #1 }
1428 }
1429 \tl_set:Nn \l_graphics_name_tl {#1}
1430 \__graphics_backend_getbb_pdf:n {#1}
1431 }
1432 \cs_new_protected:Npn \__graphics_backend_include_eps:n #1
1433 {
1434 \file_parse_full_name:nNNN {#1}
1435 \l__graphics_backend_dir_str \l__graphics_backend_name_str \l__graphics_backend_ext_str
1436 \exp_args:Nx \__graphics_backend_include_pdf:n
1437 {
1438 \l__graphics_backend_name_str - \str_tail:N \l__graphics_backend_ext_str
1439 -converted-to.pdf
1440 }
1441 }
1442 }

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

1443 </pdfmode>

```

### 5.3 dvipdfmx backend

```

1444 <*dvipdfmx | xdvipdfmx>

\__graphics_backend_getbb_eps:n Simply use the generic functions: only for dvipdfmx in the extraction cases.
\__graphics_backend_getbb_jpg:n 1445 <*initex>
\__graphics_backend_getbb_pdf:n 1446 \use:n
\__graphics_backend_getbb_png:n 1447 </initex>

```

```

1448 <*package>
1449 \AtBeginDocument
1450 </package>
1451 { \cs_new_eq:NN \__graphics_backend_getbb_eps:n \graphics_read_bb:n }
1452 <*dvipdfmx>
1453 \cs_new_protected:Npn \__graphics_backend_getbb_jpg:n #1
1454 {
1455   \int_zero:N \l_graphics_page_int
1456   \tl_clear:N \l_graphics_pagebox_tl
1457   \graphics_extract_bb:n {#1}
1458 }
1459 \cs_new_eq:NN \__graphics_backend_getbb_png:n \__graphics_backend_getbb_jpg:n
1460 \cs_new_protected:Npn \__graphics_backend_getbb_pdf:n #1
1461 {
1462   \tl_clear:N \l_graphics_decodearray_tl
1463   \bool_set_false:N \l_graphics_interpolate_bool
1464   \graphics_extract_bb:n {#1}
1465 }
1466 </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.

```

1467 \int_new:N \g__graphics_track_int

```

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

<pre> \__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 </pre>	<p>The special syntax depends on the file type. There is a difference in how PDF graphics are best handled between dvipdfmx and xdvipdfmx: for the latter it is better to use the primitive route. The relevant code for that is included later in this file.</p> <pre> 1468 \cs_new_protected:Npn \__graphics_backend_include_eps:n #1 1469 { 1470   \__kernel_backend_literal:x 1471   { 1472     PSfile = #1 \c_space_tl 1473     llx = \dim_to_decimal_in_bp:n \l_graphics_llx_dim \c_space_tl 1474     lly = \dim_to_decimal_in_bp:n \l_graphics_lly_dim \c_space_tl 1475     urx = \dim_to_decimal_in_bp:n \l_graphics_urx_dim \c_space_tl 1476     ury = \dim_to_decimal_in_bp:n \l_graphics_ury_dim 1477   } 1478 } 1479 \cs_new_protected:Npn \__graphics_backend_include_jpg:n #1 1480 { \__graphics_backend_include_auxi:nn {#1} { image } } 1481 \cs_new_eq:NN \__graphics_backend_include_png:n \__graphics_backend_include_jpg:n 1482 &lt;*dvipdfmx&gt; 1483 \cs_new_protected:Npn \__graphics_backend_include_pdf:n #1 1484 { \__graphics_backend_include_auxi:nn {#1} { epdf } } 1485 &lt;/dvipdfmx&gt; </pre>
---	--

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.

```

1486 \cs_new_protected:Npn \__graphics_backend_include_auxi:nn #1#2

```

```

1487 {
1488   \_graphics_backend_include_auxii:xnn
1489   {
1490     \tl_if_empty:NF \l_graphics_pagebox_tl
1491     { : \l_graphics_pagebox_tl }
1492     \int_compare:nNnT \l_graphics_page_int > 1
1493     { :P \int_use:N \l_graphics_page_int }
1494     \tl_if_empty:NF \l_graphics_decodearray_tl
1495     { :D \l_graphics_decodearray_tl }
1496     \bool_if:NT \l_graphics_interpolate_bool
1497     { :I }
1498   }
1499   {#1} {#2}
1500 }
1501 \cs_new_protected:Npn \_graphics_backend_include_auxii:nnn #1#2#3
1502 {
1503   \int_if_exist:cTF { c__graphics_graphics_ #2#1 _int }
1504   {
1505     \__kernel_backend_literal:x
1506     { pdf:usexobj~@graphic \int_use:c { c__graphics_graphics_ #2#1 _int } }
1507   }
1508   { \_graphics_backend_include_auxiii:nnn {#2} {#1} {#3} }
1509 }
1510 \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!

```

1511 \cs_new_protected:Npn \_graphics_backend_include_auxiii:nnn #1#2#3
1512 {
1513   \int_gincr:N \g__graphics_track_int
1514   \int_const:cn { c__graphics_graphics_ #1#2 _int } { \g__graphics_track_int }
1515   \__kernel_backend_literal:x
1516   {
1517     pdf:#3~
1518     @graphic \int_use:c { c__graphics_graphics_ #1#2 _int } ~
1519     \int_compare:nNnT \l_graphics_page_int > 1
1520     { page ~ \int_use:N \l_graphics_page_int \c_space_tl }
1521     \tl_if_empty:NF \l_graphics_pagebox_tl
1522     {
1523       pagebox ~ \l_graphics_pagebox_tl \c_space_tl
1524       bbox ~
1525       \dim_to_decimal_in_bp:n \l_graphics_llx_dim \c_space_tl
1526       \dim_to_decimal_in_bp:n \l_graphics_lly_dim \c_space_tl
1527       \dim_to_decimal_in_bp:n \l_graphics_urx_dim \c_space_tl
1528       \dim_to_decimal_in_bp:n \l_graphics_ury_dim \c_space_tl
1529     }
1530     (#1)
1531     \bool_lazy_or:nnT
1532     { \l_graphics_interpolate_bool }
1533     { ! \tl_if_empty_p:N \l_graphics_decodearray_tl }
1534     {
1535       <<
1536       \tl_if_empty:NF \l_graphics_decodearray_tl

```

```

1537             { /Decode~[ \l_graphics_decodearray_tl ] }
1538             \bool_if:NT \l_graphics_interpolate_bool
1539             { /Interpolate~true> }
1540         >>
1541     }
1542 }
1543 }

```

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

```

1544 </dvipdfmx | xdvipdfmx>

```

## 5.4 xdvipdfmx backend

```

1545 < *xdvipdfmx>

```

### 5.4.1 Images

For `xdvipdfmx`, 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  $\text{Xe}\text{L}\text{A}\text{T}\text{E}\text{X}$  primitive omits the text box from the page box specification, so there is also some “trimming” to do here.

```

\graphics_backend_getbb_jpg:n
\graphics_backend_getbb_pdf:n
\graphics_backend_getbb_png:n
\graphics_backend_getbb_auxi:nN
\graphics_backend_getbb_auxii:nNn
\graphics_backend_getbb_auxiii:nNnn
\graphics_backend_getbb_auxiv:nNnn
\graphics_backend_getbb_auxv:nNnn
\graphics_backend_getbb_auxvi:nNnn
\graphics_backend_getbb_pagebox:w

```

```

1546 \cs_new_protected:Npn \_graphics_backend_getbb_jpg:n #1
1547 {
1548     \int_zero:N \l_graphics_page_int
1549     \tl_clear:N \l_graphics_pagebox_tl
1550     \_graphics_backend_getbb_auxi:nN {#1} \tex_XeTeXpicfile:D
1551 }
1552 \cs_new_eq:NN \_graphics_backend_getbb_png:n \_graphics_backend_getbb_jpg:n
1553 \cs_new_protected:Npn \_graphics_backend_getbb_pdf:n #1
1554 {
1555     \tl_clear:N \l_graphics_decodearray_tl
1556     \bool_set_false:N \l_graphics_interpolate_bool
1557     \_graphics_backend_getbb_auxi:nN {#1} \tex_XeTeXpdffile:D
1558 }
1559 \cs_new_protected:Npn \_graphics_backend_getbb_auxi:nN #1#2
1560 {
1561     \int_compare:nNnTF \l_graphics_page_int > 1
1562     { \_graphics_backend_getbb_auxii:nN \l_graphics_page_int {#1} #2 }
1563     { \_graphics_backend_getbb_auxiii:nNnn {#1} #2 { :P 1 } { page 1 } }
1564 }
1565 \cs_new_protected:Npn \_graphics_backend_getbb_auxii:nNn #1#2#3
1566 { \_graphics_backend_getbb_auxiii:nNnn {#2} #3 { :P #1 } { page #1 } }
1567 \cs_generate_variant:Nn \_graphics_backend_getbb_auxii:nNn { V }
1568 \cs_new_protected:Npn \_graphics_backend_getbb_auxiii:nNnn #1#2#3#4
1569 {
1570     \tl_if_empty:NTF \l_graphics_pagebox_tl
1571     { \_graphics_backend_getbb_auxiv:nNnn \l_graphics_pagebox_tl }
1572     { \_graphics_backend_getbb_auxv:nNnn }
1573     {#1} #2 {#3} {#4}
1574 }
1575 \cs_new_protected:Npn \_graphics_backend_getbb_auxiv:nNnn #1#2#3#4#5
1576 {
1577     \use:x
1578     {

```



```

1579         \__graphics_backend_getbb_auxv:nNnn {#2} #3 { : #1 #4 }
1580         { #5 ~ \__graphics_backend_getbb_pagebox:w #1 }
1581     }
1582 }
1583 \cs_generate_variant:Nn \__graphics_backend_getbb_auxiv:nnNnn { V }
1584 \cs_new_protected:Npn \__graphics_backend_getbb_auxv:nNnn #1#2#3#4
1585 {
1586     \graphics_bb_restore:nF {#1#3}
1587     { \__graphics_backend_getbb_auxvi:nNnn {#1} #2 {#3} {#4} }
1588 }
1589 \cs_new_protected:Npn \__graphics_backend_getbb_auxvi:nNnn #1#2#3#4
1590 {
1591     \hbox_set:Nn \l__graphics_internal_box { #2 #1 ~ #4 }
1592     \dim_set:Nn \l_graphics_urx_dim { \box_wd:N \l__graphics_internal_box }
1593     \dim_set:Nn \l_graphics_ury_dim { \box_ht:N \l__graphics_internal_box }
1594     \graphics_bb_save:n {#1#3}
1595 }
1596 \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 X<sub>Y</sub>TeX is best done using the \tex\_XeTeXpdf file:D primitive. The syntax here is the same as for the graphic measurement part, although we know at this stage that there must be some valid setting for \l\_graphics\_pagebox\_tl.

```

1597 \cs_new_protected:Npn \__graphics_backend_include_pdf:n #1
1598 {
1599     \tex_XeTeXpdf file:D
1600     \__graphics_backend_include_pdf_quote:w #1 "#1" \q_stop \c_space_tl
1601     \int_compare:nNnT \l_graphics_page_int > 0
1602         { page ~ \int_use:N \l_graphics_page_int \c_space_tl }
1603     \exp_after:wN \__graphics_backend_getbb_pagebox:w \l_graphics_pagebox_tl
1604 }
1605 \cs_new:Npn \__graphics_backend_include_pdf_quote:w #1 " #2 " #3 \q_stop
1606 { " #2 " }

```

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

```

1607 </xdvipdfmx>

```

## 5.5 dvisvgm backend

```

1608 <*dvisvgm>

```

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

```

1609 <*initex>
1610 \use:n
1611 </initex>
1612 <*package>
1613 \AtBeginDocument
1614 </package>
1615 { \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`

```

1616 \cs_new_protected:Npn \_graphics_backend_getbb_jpg:n #1
1617 {
1618   \int_zero:N \l_graphics_page_int
1619   \tl_clear:N \l_graphics_pagebox_tl
1620   \graphics_extract_bb:n {#1}
1621 }
1622 \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

```

1623 \cs_new_protected:Npn \_graphics_backend_getbb_pdf:n #1
1624 {
1625   \tl_clear:N \l_graphics_decodearray_tl
1626   \bool_set_false:N \l_graphics_interpolate_bool
1627   \graphics_extract_bb:n {#1}
1628 }

```

(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`

```

1629 \cs_new_protected:Npn \_graphics_backend_include_eps:n #1
1630 { \_graphics_backend_include:nn { PSfile } {#1} }
1631 \cs_new_protected:Npn \_graphics_backend_include_pdf:n #1
1632 { \_graphics_backend_include:nn { pdffile } {#1} }
1633 \cs_new_protected:Npn \_graphics_backend_include:nn #1#2
1634 {
1635   \__kernel_backend_literal:x
1636   {
1637     #1 = #2 \c_space_tl
1638     llx = \dim_to_decimal_in_bp:n \l_graphics_llx_dim \c_space_tl
1639     lly = \dim_to_decimal_in_bp:n \l_graphics_lly_dim \c_space_tl
1640     urx = \dim_to_decimal_in_bp:n \l_graphics_urx_dim \c_space_tl
1641     ury = \dim_to_decimal_in_bp:n \l_graphics_ury_dim
1642   }
1643 }

```

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

`\_graphics_backend_include_png:n` The backend here has built-in support for basic graphic inclusion (see `dvisvgm.def` for a more complex approach, needed if clipping, *etc.*, is covered at the graphic backend level).

`\_graphics_backend_include_jpg:n`

`\_graphics_backend_include_bitmap_quote:w` 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.

```

1644 \cs_new_protected:Npn \_graphics_backend_include_png:n #1
1645 {
1646   \__kernel_backend_literal:x
1647   {
1648     dvisvgm:img~
1649     \dim_to_decimal:n { \l_graphics_ury_dim } ~
1650     \dim_to_decimal:n { \l_graphics_ury_dim } ~

```

```

1651         \_graphics_backend_include_bitmap_quote:w #1 " #1 " \q_stop
1652     }
1653 }
1654 \cs_new_eq:NN \_graphics_backend_include_jpg:n \_graphics_backend_include_png:n
1655 \cs_new:Npn \_graphics_backend_include_bitmap_quote:w #1 " #2 " #3 \q_stop
1656 { " #2 " }

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

1657 \</divsvgm>
1658 \</initex | package>

```

## 6 l3backend-pdf Implementation

```

1659 <*initex | package>
1660 <@@=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
1661 \box_new:N \l\_pdf\_internal\_box

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

```

### 6.2 dvips backend

```

1662 <*dvips>

\_pdf_backend_pdfmark:n Used often enough it should be a separate function.
\_pdf_backend_pdfmark:x
1663 \cs_new_protected:Npn \_pdf_backend_pdfmark:n #1
1664 { \_kernel_backend_postscript:n { mark #1 ~ pdfmark } }
1665 \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
1666 \cs_new_protected:Npn \_pdf_backend_catalog_gput:nn #1#2
1667 { \_pdf_backend_pdfmark:n { { Catalog } << /#1 ~ #2 >> /PUT } }
1668 \cs_new_protected:Npn \_pdf_backend_info_gput:nn #1#2
1669 { \_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.

```
1670 \int_new:N \g__pdf_backend_object_int
1671 \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.

```
1672 \cs_new_protected:Npn \__pdf_backend_object_new:nn #1#2
1673 {
1674   \int_gincr:N \g__pdf_backend_object_int
1675   \int_const:cn
1676     { c__pdf_backend_object_ \tl_to_str:n {#1} _int }
1677     { \g__pdf_backend_object_int }
1678   \prop_gput:Nnn \g__pdf_backend_object_prop {#1} {#2}
1679 }
1680 \cs_new:Npn \__pdf_backend_object_ref:n #1
1681 { { 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_stream:nn`  
`\__pdf_backend_object_write_stream:nnn`

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

```
1682 \cs_new_protected:Npn \__pdf_backend_object_write:nn #1#2
1683 {
1684   \__pdf_backend_pdfmark:x
1685   {
1686     /_objdef ~ \__pdf_backend_object_ref:n {#1}
1687     /type
1688     \str_case_e:nn
1689       { \prop_item:Nn \g__pdf_backend_object_prop {#1} }
1690       {
1691         { array } { /array }
1692         { dict } { /dict }
1693         { fstream } { /stream }
1694         { stream } { /stream }
1695       }
1696     /OBJ
1697   }
1698   \use:c
1699     { __pdf_backend_object_write_ \prop_item:Nn \g__pdf_backend_object_prop {#1} :nn }
1700     { \__pdf_backend_object_ref:n {#1} } {#2}
1701 }
1702 \cs_generate_variant:Nn \__pdf_backend_object_write:nn { nx }
1703 \cs_new_protected:Npn \__pdf_backend_object_write_array:nn #1#2
1704 {
1705   \__pdf_backend_pdfmark:x
1706     { #1 [ ~ \exp_not:n {#2} ~ ] ~ /PUTINTERVAL }
1707 }
1708 \cs_new_protected:Npn \__pdf_backend_object_write_dict:nn #1#2
1709 {
1710   \__pdf_backend_pdfmark:x
1711     { #1 << \exp_not:n {#2} >> /PUT }
1712 }
1713 \cs_new_protected:Npn \__pdf_backend_object_write_stream:nn #1#2
```

```

1714 {
1715   \exp_args:Nx
1716   \__pdf_backend_object_write_stream:nnn {#1} #2
1717 }
1718 \cs_new_protected:Npn \__pdf_backend_object_write_stream:nnn #1#2#3
1719 {
1720   \__kernel_backend_postscript:n
1721   {
1722     [nobreak]
1723     mark ~ #1 ~ ( #3 ) /PUT ~ pdfmark ~
1724     mark ~ #1 ~ << #2 >> /PUT ~ pdfmark
1725   }
1726 }

```

(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 1727 \cs_new_protected:Npn \__pdf_backend_object_now:nn #1#2
1728 {
1729   \int_gincr:N \g__pdf_backend_object_int
1730   \__pdf_backend_pdfmark:x
1731   {
1732     /objdef ~ { pdf.obj \int_use:N \g__pdf_backend_object_int }
1733     /type
1734     \str_case:nn
1735       {#1}
1736       {
1737         { array } { /array }
1738         { dict } { /dict }
1739         { fstream } { /stream }
1740         { stream } { /stream }
1741       }
1742     /OBJ
1743   }
1744   \exp_args:Nnx \use:c { __pdf_backend_object_write_ #1 :nn }
1745   { { pdf.obj \int_use:N \g__pdf_backend_object_int } } {#2}
1746 }
1747 \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.

```

1748 \cs_new:Npn \__pdf_backend_object_last:
1749 { { pdf.obj \int_use:N \g__pdf_backend_object_int } }

```

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

### 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.

```

1750 \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.

1751 \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.

1752 \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  
\\_pdf\_backend\_annotation\_aux:nnnn object data lists. Here, to get the co-ordinates of the annotation, we need to have the  
pdf.llx data collected at the PostScript level. That requires a bit of box trickery (effectively a  
pdf.lly 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  
pdf.urx border. There is a split into two parts here to allow an easy way of applying the Adobe  
pdf.ury Reader fix.

1753 \cs\_new\_protected:Npn \\_pdf\_backend\_annotation:nnnn #1#2#3#4

1754 {

1755 \\_pdf\_backend\_annotation\_aux:nnnn {#1} {#2} {#3} {#4}

1756 \int\_gincr:N \g\_\_pdf\_backend\_object\_int

1757 \int\_gset\_eq:NN \g\_\_pdf\_backend\_annotation\_int \g\_\_pdf\_backend\_object\_int

1758 \\_pdf\_backend\_pdfmark:x

1759 {

1760

1761 /\_objdef { pdf.obj \int\_use:N \g\_\_pdf\_backend\_object\_int }

1762 pdf.rect ~

1763 #4 ~

1764 /ANN

1765 }

1766 }

1767 \cs\_new\_protected:Npn \\_pdf\_backend\_annotation\_aux:nnnn #1#2#3#4

1768 {

1769 \box\_move\_down:nn {#3}

1770 { \hbox:n { \\_kernel\_backend\_postscript:n { pdf.save.ll } } }

1771 \hbox:n {#4}

1772 \box\_move\_up:nn {#2}

1773 {

1774 \hbox:n

1775 {

1776 \tex\_kern:D \dim\_eval:n {#1} \scan\_stop:

1777 \\_kernel\_backend\_postscript:n { pdf.save.ur }

1778 }

1779 }

1780 \int\_gincr:N \g\_\_pdf\_backend\_object\_int

1781 \int\_gset\_eq:NN \g\_\_pdf\_backend\_annotation\_int \g\_\_pdf\_backend\_object\_int

1782 \\_pdf\_backend\_pdfmark:x

1783 {

1784 /\_objdef { pdf.obj \int\_use:N \g\_\_pdf\_backend\_object\_int }

1785 pdf.rect

1786 /ANN

1787 }

1788 }

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

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

```
1789 \cs_new:Npn \_pdf_backend_annotation_last:
1790 { { 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.

```
1791 \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.

```
1792 \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.

```
1793 \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.

```
1794 \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.

```
1795 \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.

```
1796 \tl_new:N \l__pdf_breaklink_pdfmark_tl
1797 \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.

```
1798 \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.

```
1799 \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.

```

1800 \cs_new_protected:Npn \__pdf_backend_link_begin_goto:nnw #1#2
1801 { \__pdf_backend_link_begin:nw { #1 /Subtype /Link /A << /S /GoTo /D ( #2 ) >> } }
1802 \cs_new_protected:Npn \__pdf_backend_link_begin_user:nnw #1#2
1803 { \__pdf_backend_link_begin:nw {#1#2} }
1804 \cs_new_protected:Npn \__pdf_backend_link_begin:nw #1
1805 {
1806   \bool_if:NF \g__pdf_backend_link_bool
1807   { \__pdf_backend_link_begin_aux:nw {#1} }
1808 }
1809 \cs_new_protected:Npn \__pdf_backend_link_begin_aux:nw #1
1810 {
1811   \bool_gset_true:N \g__pdf_backend_link_bool
1812   \__kernel_backend_postscript:n
1813   { /pdf.link.dict ( #1 ) def }
1814   \tl_gset:Nn \g__pdf_backend_link_dict_tl {#1}
1815   \__pdf_backend_link_sf_save:
1816   \mode_if_math:TF
1817   { \bool_gset_true:N \g__pdf_backend_link_math_bool }
1818   { \bool_gset_false:N \g__pdf_backend_link_math_bool }
1819   \hbox_set:Nw \l__pdf_backend_content_box
1820   \__pdf_backend_link_sf_restore:
1821   \bool_if:NT \g__pdf_backend_link_math_bool
1822   { \c_math_toggle_token }
1823 }
1824 \cs_new_protected:Npn \__pdf_backend_link_end:
1825 {
1826   \bool_if:NT \g__pdf_backend_link_bool
1827   { \__pdf_backend_link_end_aux: }
1828 }
1829 \cs_new_protected:Npn \__pdf_backend_link_end_aux:
1830 {
1831   \bool_if:NT \g__pdf_backend_link_math_bool
1832   { \c_math_toggle_token }
1833   \__pdf_backend_link_sf_save:
1834   \hbox_set_end:
1835   \__pdf_backend_link_minima:
1836   \hbox_set:Nn \l__pdf_backend_model_box { Gg }
1837   \exp_args:Nx \__pdf_backend_link_outerbox:n
1838   {

```



```

1839 <*initex>
1840 \l_galley_total_left_margin_dim
1841 </initex>
1842 <*package>
1843 \int_if_odd:nTF { \value { page } }
1844 { \oddsidemargin }
1845 { \evensidemargin }
1846 </package>
1847 }
1848 \box_move_down:nn { \box_dp:N \l__pdf_backend_content_box }
1849 { \hbox:n { \__kernel_backend_postscript:n { pdf.save.linkll } } }
1850 \__pdf_breaklink_postscript:n { pdf.bordertracking.begin }
1851 \__pdf_breaklink_usebox:N \l__pdf_backend_content_box
1852 \__pdf_breaklink_postscript:n { pdf.bordertracking.end }
1853 \box_move_up:nn { \box_ht:N \l__pdf_backend_content_box }
1854 {
1855 \hbox:n
1856 { \__kernel_backend_postscript:n { pdf.save.linkur } }
1857 }
1858 \int_gincr:N \g__pdf_backend_object_int
1859 \int_gset_eq:NN \g__pdf_backend_link_int \g__pdf_backend_object_int
1860 \__kernel_backend_postscript:x
1861 {
1862 mark
1863 /objdef { pdf.obj \int_use:N \g__pdf_backend_link_int }
1864 \g__pdf_backend_link_dict_tl \c_space_tl
1865 pdf.rect
1866 /ANN ~ \l__pdf_breaklink_pdfmark_tl
1867 }
1868 \__pdf_backend_link_sf_restore:
1869 \bool_gset_false:N \g__pdf_backend_link_bool
1870 }
1871 \cs_new_protected:Npn \__pdf_backend_link_minima:
1872 {
1873 \hbox_set:Nn \l__pdf_backend_model_box { Gg }
1874 \__kernel_backend_postscript:x
1875 {
1876 /pdf.linkdp.pad ~
1877 \dim_to_decimal:n
1878 {
1879 \dim_max:nn
1880 {
1881 \box_dp:N \l__pdf_backend_model_box
1882 - \box_dp:N \l__pdf_backend_content_box
1883 }
1884 { Opt }
1885 } ~
1886 pdf.pt.dvi ~ def
1887 /pdf.linkht.pad ~
1888 \dim_to_decimal:n
1889 {
1890 \dim_max:nn
1891 {
1892 \box_ht:N \l__pdf_backend_model_box

```

```

1893         - \box_ht:N \l__pdf_backend_content_box
1894     }
1895     { Opt }
1896 } ~
1897 pdf.pt.dvi ~ def
1898 }
1899 }
1900 \cs_new_protected:Npn \__pdf_backend_link_outerbox:n #1
1901 {
1902     \__kernel_backend_postscript:x
1903     {
1904         /pdf.outerbox
1905         [
1906             \dim_to_decimal:n {#1} ~
1907             \dim_to_decimal:n { -\box_dp:N \l__pdf_backend_model_box } ~
1908         (*initex)
1909             \dim_to_decimal:n { #1 + \l_galley_text_width_dim } ~
1910         (/initex)
1911         (*package)
1912             \dim_to_decimal:n { #1 + \textwidth } ~
1913         (/package)
1914             \dim_to_decimal:n { \box_ht:N \l__pdf_backend_model_box }
1915         ]
1916         [ exch { pdf.pt.dvi } forall ] def
1917         /pdf.baselineskip ~
1918         \dim_to_decimal:n { \tex_baselineskip:D } ~ dup ~ 0 ~ gt
1919         { pdf.pt.dvi ~ def }
1920         { pop ~ pop }
1921         ifelse
1922     }
1923 }
1924 \cs_new_protected:Npn \__pdf_backend_link_sf_save:
1925 {
1926     \int_gset:Nn \g__pdf_backend_link_sf_int
1927     {
1928         \mode_if_horizontal:TF
1929         { \tex_spacefactor:D }
1930         { 0 }
1931     }
1932 }
1933 \cs_new_protected:Npn \__pdf_backend_link_sf_restore:
1934 {
1935     \mode_if_horizontal:T
1936     {
1937         \int_compare:nNnT \g__pdf_backend_link_sf_int > { 0 }
1938         { \int_set_eq:NN \tex_spacefactor:D \g__pdf_backend_link_sf_int }
1939     }
1940 }

```

(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.

```

1941 <*package>
1942 \use_none:n
1943 {
1944   \cs_if_exist:NT \@makecol@hook
1945   {
1946     \tl_put_right:Nn \@makecol@hook
1947     {
1948       \box_if_empty:NF \@cclv
1949       {
1950         \vbox_set:Nn \@cclv
1951         {
1952           \__kernel_backend_postscript:n
1953           {
1954             pdf.globaldict /pdf.brokenlink.rect ~ known
1955             { pdf.bordertracking.continue }
1956             if
1957           }
1958           \vbox_unpack_drop:N \@cclv
1959           \__kernel_backend_postscript:n
1960           { pdf.bordertracking.endpage }
1961         }
1962       }
1963     }
1964     \tl_set:Nn \l__pdf_breaklink_pdfmark_tl { pdf.pdfmark }
1965     \cs_set_eq:NN \__pdf_breaklink_postscript:n \__kernel_backend_postscript:n
1966     \cs_set_eq:NN \__pdf_breaklink_usebox:N \hbox_unpack:N
1967   }
1968 }
1969 </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.

```

1970 \cs_new:Npn \__pdf_backend_link_last:
1971 { { 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.

```

1972 \cs_new_protected:Npn \__pdf_backend_link_margin:n #1
1973 {
1974   \__kernel_backend_postscript:x
1975   {
1976     /pdf.linkmargin { \dim_to_decimal:n {#1} ~ pdf.pt.dvi } def
1977   }
1978 }

```

(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.

```

1979 \cs_new_protected:Npn \__pdf_backend_destination:nn #1#2

```

```

1980 {
1981   \_kernel_backend_postscript:n { pdf.dest.anchor }
1982   \_pdf_backend_pdfmark:x
1983   {
1984     /View
1985     [
1986       \str_case:nnF {#2}
1987       {
1988         { xyz } { /XYZ ~ pdf.dest.point ~ null }
1989         { fit } { /Fit }
1990         { fitb } { /FitB }
1991         { fitbh } { /FitBH ~ pdf.dest.y }
1992         { fitbv } { /FitBV ~ pdf.dest.x }
1993         { fith } { /FitH ~ pdf.dest.y }
1994         { fitv } { /FitV ~ pdf.dest.x }
1995       }
1996       {
1997         /XYZ ~ pdf.dest.point ~ \fp_eval:n { (#2) / 100 }
1998       }
1999     ]
2000     /Dest ( \exp_not:n {#1} ) cvn
2001     /DEST
2002   }
2003 }
2004 \cs_new_protected:Npn \_pdf_backend_destination_rectangle:nn #1#2
2005 {
2006   \group_begin:
2007   \hbox_set:Nn \l__pdf_internal_box {#2}
2008   \box_move_down:nn
2009   { \box_dp:N \l__pdf_internal_box }
2010   { \hbox:n { \_kernel_backend_postscript:n { pdf.save.ll } } }
2011   \box_use:N \l__pdf_internal_box
2012   \box_move_up:nn
2013   { \box_ht:N \l__pdf_internal_box }
2014   { \hbox:n { \_kernel_backend_postscript:n { pdf.save.ur } } }
2015   \_pdf_backend_pdfmark:n
2016   {
2017     /View
2018     [
2019       /FitR ~
2020       pdf.llx ~ pdf.lly ~ pdf.dest2device ~
2021       pdf.urx ~ pdf.ury ~ pdf.dest2device
2022     ]
2023     /Dest ( #1 ) cvn
2024     /DEST
2025   }
2026   \group_end:
2027 }

```

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

#### 6.2.4 Structure

\\_pdf\_backend\_compresslevel:n These are all no-ops.  
\\_pdf\_backend\_compress\_objects:n

```

2028 \cs_new_protected:Npn \__pdf_backend_compresslevel:n #1 { }
2029 \cs_new_protected:Npn \__pdf_backend_compress_objects:n #1 { }

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

```

```

\__pdf_backend_version_major_gset:n Data not available!
\__pdf_backend_version_minor_gset:n
2030 \cs_new_protected:Npn \__pdf_backend_version_major_gset:n #1 { }
2031 \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:
2032 \cs_new:Npn \__pdf_backend_version_major: { -1 }
2033 \cs_new:Npn \__pdf_backend_version_minor: { -1 }

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

```

### 6.2.5 Marked content

```

\__pdf_backend_bdc:nn Simple wrappers.
\__pdf_backend_emc:
2034 \cs_new_protected:Npn \__pdf_backend_bdc:nn #1#2
2035 { \__pdf_backend_pdfmark:n { /#1 ~ #2 /BDC } }
2036 \cs_new_protected:Npn \__pdf_backend_emc:
2037 { \__pdf_backend_pdfmark:n { /EMC } }

(End definition for \__pdf_backend_bdc:nn and \__pdf_backend_emc:.)
2038 </dvips>

```

## 6.3 pdfmode backend

```

2039 <*pdfmode>

```

### 6.3.1 Annotations

```

\__pdf_backend_annotation:nnnn Simply pass the raw data through, just dealing with evaluation of dimensions.
2040 \cs_new_protected:Npx \__pdf_backend_annotation:nnnn #1#2#3#4
2041 {
2042   \cs_if_exist:NTF \tex_pdfextension:D
2043   { \tex_pdfextension:D annot ~ }
2044   { \tex_pdfannot:D }
2045   width ~ \exp_not:N \dim_eval:n {#1} ~
2046   height ~ \exp_not:N \dim_eval:n {#2} ~
2047   depth ~ \exp_not:N \dim_eval:n {#3} ~
2048   {#4}
2049 }

(End definition for \__pdf_backend_annotation:nnnn.)

```

```

\__pdf_backend_annotation_last: A tiny amount of extra data gets added here.
2050 \cs_new:Npx \__pdf_backend_annotation_last:
2051 {
2052   \exp_not:N \int_value:w
2053   \cs_if_exist:NTF \tex_pdffeedback:D
2054   { \exp_not:N \tex_pdffeedback:D lastannot ~ }
2055   { \exp_not:N \tex_pdflastannot:D }
2056   \c_space_tl 0 ~ R
2057 }

```

(End definition for `\_pdf_backend_annotation_last:`)

```

\_pdf_backend_link_begin_goto:nnw Links are all created using the same internals.
\_pdf_backend_link_begin_user:nnw
\_pdf_backend_link_begin:nnnw
\_pdf_backend_link_end:
2058 \cs_new_protected:Npn \_pdf_backend_link_begin_goto:nnw #1#2
2059 { \_pdf_backend_link_begin:nnnw {#1} { goto~name } {#2} }
2060 \cs_new_protected:Npn \_pdf_backend_link_begin_user:nnw #1#2
2061 { \_pdf_backend_link_begin:nnnw {#1} { user } {#2} }
2062 \cs_new_protected:Npx \_pdf_backend_link_begin:nnnw #1#2#3
2063 {
2064   \cs_if_exist:NTF \tex_pdfextension:D
2065   { \tex_pdfextension:D startlink ~ }
2066   { \tex_pdfstartlink:D }
2067   attr {#1}
2068   #2 {#3}
2069 }
2070 \cs_new_protected:Npx \_pdf_backend_link_end:
2071 {
2072   \cs_if_exist:NTF \tex_pdfextension:D
2073   { \tex_pdfextension:D endlink \scan_stop: }
2074   { \tex_pdfendlink:D }
2075 }

```

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

`\_pdf_backend_link_last:` Formatted for direct use.

```

2076 \cs_new:Npx \_pdf_backend_link_last:
2077 {
2078   \exp_not:N \int_value:w
2079   \cs_if_exist:NTF \tex_pdffeedback:D
2080   { \exp_not:N \tex_pdffeedback:D lastlink ~ }
2081   { \exp_not:N \tex_pdflastlink:D }
2082   \c_space_tl 0 ~ R
2083 }

```

(End definition for `\_pdf_backend_link_last:`)

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

```

2084 \cs_new_protected:Npx \_pdf_backend_link_margin:n #1
2085 {
2086   \cs_if_exist:NTF \tex_pdfvariable:D
2087   { \exp_not:N \tex_pdfvariable:D linkmargin }
2088   { \exp_not:N \tex_pdflinkmargin:D }
2089   \exp_not:N \dim_eval:n {#1} \scan_stop:
2090 }

```

(End definition for `\_pdf_backend_link_margin:n`)

`\_pdf_backend_destination:nn` A simple task: pass the data to the primitive. The `\scan_stop:` deals with the danger  
`\_pdf_backend_destination_rectangle:nn` 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.

```

2091 \cs_new_protected:Npx \_pdf_backend_destination:nn #1#2
2092 {
2093   \cs_if_exist:NTF \tex_pdfextension:D
2094   { \exp_not:N \tex_pdfextension:D dest ~ }

```

```

2095 { \exp_not:N \tex_pdfdest:D }
2096   name {#1}
2097   \exp_not:N \str_case:nnF {#2}
2098   {
2099     { xyz } { xyz }
2100     { fit } { fit }
2101     { fitb } { fitb }
2102     { fitbh } { fitbh }
2103     { fitbv } { fitbv }
2104     { fith } { fith }
2105     { fitv } { fitv }
2106   }
2107   { xyz ~ zoom \exp_not:N \fp_eval:n { #2 * 10 } }
2108   \scan_stop:
2109 }
2110 \cs_new_protected:Npx \__pdf_backend_destination_rectangle:nn #1#2
2111 {
2112   \group_begin:
2113   \hbox_set:Nn \l__pdf_internal_box {#2}
2114   \cs_if_exist:NTF \tex_pdfextension:D
2115   { \exp_not:N \tex_pdfextension:D dest ~ }
2116   { \exp_not:N \tex_pdfdest:D }
2117   name {#1}
2118   fitr ~
2119   width \exp_not:N \box_wd:N \l__pdf_internal_box
2120   height \exp_not:N \box_ht:N \l__pdf_internal_box
2121   depth \exp_not:N \box_dp:N \l__pdf_internal_box
2122   \box_use:N \l__pdf_internal_box
2123   \group_end:
2124 }

```

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

### 6.3.2 Catalogue entries

```

\__pdf_backend_catalog_gput:nn
\__pdf_backend_info_gput:nn
2125 \cs_new_protected:Npx \__pdf_backend_catalog_gput:nn #1#2
2126 {
2127   \cs_if_exist:NTF \tex_pdfextension:D
2128   { \tex_pdfextension:D catalog }
2129   { \tex_pdfcatalog:D }
2130   { / #1 ~ #2 }
2131 }
2132 \cs_new_protected:Npx \__pdf_backend_info_gput:nn #1#2
2133 {
2134   \cs_if_exist:NTF \tex_pdfextension:D
2135   { \tex_pdfextension:D info }
2136   { \tex_pdfinfo:D }
2137   { / #1 ~ #2 }
2138 }

```

(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.

2139 `\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`

```
2140 \group_begin:
2141 \cs_set_protected:Npn \__pdf_tmp:w #1#2
2142 {
2143   \cs_new_protected:Npx \__pdf_backend_object_new:nn ##1##2
2144   {
2145     #1 reserveobjnum ~
2146     \int_const:cn
2147     { c__pdf_backend_object_ \exp_not:N \tl_to_str:n {##1} _int }
2148     {#2}
2149     \prop_gput:Nnn \exp_not:N \g__pdf_backend_object_prop {##1} {##2}
2150   }
2151 }
2152 \cs_if_exist:NTF \tex_pdfextension:D
2153 {
2154   \__pdf_tmp:w
2155   { \tex_pdfextension:D obj ~ }
2156   { \exp_not:N \tex_pdffeedback:D lastobj }
2157 }
2158 { \__pdf_tmp:w { \tex_pdfobj:D } { \tex_pdflastobj:D } }
2159 \group_end:
2160 \cs_new:Npn \__pdf_backend_object_ref:n #1
2161 { \int_use:c { c__pdf_backend_object_ \tl_to_str:n {#1} _int } ~ 0 ~ R }
```

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

`\__pdf_backend_object_write:nn` Writing the data needs a little information about the structure of the object.

`\__pdf_backend_object_write:nx`

`\__pdf_exp_not_i:nn`

`\__pdf_exp_not_ii:nn`

```
2162 \group_begin:
2163 \cs_set_protected:Npn \__pdf_tmp:w #1
2164 {
2165   \cs_new_protected:Npn \__pdf_backend_object_write:nn ##1##2
2166   {
2167     \tex_immediate:D #1 useobjnum ~
2168     \int_use:c
2169     { c__pdf_backend_object_ \tl_to_str:n {##1} _int }
2170     \str_case_e:nn
2171     { \prop_item:Nn \g__pdf_backend_object_prop {##1} }
2172     {
2173       { array } { { [ ~ \exp_not:n {##2} ~ ] } }
2174       { dict } { { < ~ \exp_not:n {##2} ~ > } }
2175       { fstream }
2176       {
2177         stream ~ attr ~ { \__pdf_exp_not_i:nn ##2 } ~
2178         file ~ { \__pdf_exp_not_ii:nn ##2 }
2179       }
2180     } stream }
2181     {
2182       stream ~ attr ~ { \__pdf_exp_not_i:nn ##2 } ~
```



```

2183         { \_pdf_exp_not_ii:nn ##2 }
2184     }
2185 }
2186 }
2187 }
2188 \cs_if_exist:NTF \tex_pdfextension:D
2189 { \_pdf_tmp:w { \tex_pdfextension:D obj ~ } }
2190 { \_pdf_tmp:w { \tex_pdfobj:D } }
2191 \group_end:
2192 \cs_generate_variant:Nn \_pdf_backend_object_write:nn { nx }
2193 \cs_new:Npn \_pdf_exp_not_i:nn #1#2 { \exp_not:n {#1} }
2194 \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
2195 \group_begin:
2196 \cs_set_protected:Npn \_pdf_tmp:w #1
2197 {
2198     \cs_new_protected:Npn \_pdf_backend_object_now:nn ##1##2
2199     {
2200         \tex_immediate:D #1
2201         \str_case:nn
2202         {##1}
2203         {
2204             { array } { { [ ~ \exp_not:n {##2} ~ ] } }
2205             { dict } { { << ~ \exp_not:n {##2} ~ >> } }
2206             { fstream }
2207             {
2208                 stream ~ attr ~ { \_pdf_exp_not_i:nn ##2 } ~
2209                 file ~ { \_pdf_exp_not_ii:nn ##2 }
2210             }
2211             { stream }
2212             {
2213                 stream ~ attr ~ { \_pdf_exp_not_i:nn ##2 } ~
2214                 { \_pdf_exp_not_ii:nn ##2 }
2215             }
2216         }
2217     }
2218 }
2219 \cs_if_exist:NTF \tex_pdfextension:D
2220 { \_pdf_tmp:w { \tex_pdfextension:D obj ~ } }
2221 { \_pdf_tmp:w { \tex_pdfobj:D } }
2222 \group_end:
2223 \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.

```

2224 \cs_new:Npx \_pdf_backend_object_last:
2225 {
2226     \exp_not:N \int_value:w
2227     \cs_if_exist:NTF \tex_pdffeedback:D
2228     { \exp_not:N \tex_pdffeedback:D lastobj ~ }

```

```

2229     { \exp_not:N \tex_pdflastobj:D }
2230     \c_space_tl 0 ~ R
2231   }

```

(End definition for `\__pdf_backend_object_last:.`)

### 6.3.4 Structure

Simply pass data to the engine.

```

\__pdf_backend_compresslevel:n
\__pdf_backend_compress_objects:n
\__pdf_backend_objcompresslevel:n
2232 \cs_new_protected:Npx \__pdf_backend_compresslevel:n #1
2233 {
2234   \exp_not:N \tex_global:D
2235   \cs_if_exist:NTF \tex_pdfcompresslevel:D
2236   { \tex_pdfcompresslevel:D }
2237   { \tex_pdfvariable:D compresslevel }
2238   \exp_not:N \int_value:w \exp_not:N \int_eval:n {#1} \scan_stop:
2239 }
2240 \cs_new_protected:Npn \__pdf_backend_compress_objects:n #1
2241 {
2242   \bool_if:nTF {#1}
2243   { \__pdf_backend_objcompresslevel:n { 2 } }
2244   { \__pdf_backend_objcompresslevel:n { 0 } }
2245 }
2246 \cs_new_protected:Npx \__pdf_backend_objcompresslevel:n #1
2247 {
2248   \exp_not:N \tex_global:D
2249   \cs_if_exist:NTF \tex_pdfobjcompresslevel:D
2250   { \tex_pdfobjcompresslevel:D }
2251   { \tex_pdfvariable:D objcompresslevel }
2252   #1 \scan_stop:
2253 }

```

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

`\__pdf_backend_version_major_gset:n` At present, we don't have a primitive for the major version in pdfTeX, but we anticipate one ...

```

\__pdf_backend_version_minor_gset:n
2254 \cs_new_protected:Npx \__pdf_backend_version_major_gset:n #1
2255 {
2256   \cs_if_exist:NTF \tex_pdfvariable:D
2257   {
2258     \int_compare:nNnT \tex luatexversion:D > { 106 }
2259     {
2260       \exp_not:N \tex_global:D \tex_pdfvariable:D majorversion
2261       \exp_not:N \int_eval:n {#1} \scan_stop:
2262     }
2263   }
2264   {
2265     \cs_if_exist:NT \tex_pdfmajorversion:D
2266     {
2267       \exp_not:N \tex_global:D \tex_pdfmajorversion:D
2268       \exp_not:N \int_eval:n {#1} \scan_stop:
2269     }
2270   }

```

```

2271 }
2272 \cs_new_protected:Npx \__pdf_backend_version_minor_gset:n #1
2273 {
2274   \exp_not:N \tex_global:D
2275   \cs_if_exist:NTF \tex_pdfminorversion:D
2276     { \exp_not:N \tex_pdfminorversion:D }
2277     { \tex_pdfvariable:D minorversion }
2278     \exp_not:N \int_eval:n {#1} \scan_stop:
2279 }

```

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

\\_\_pdf\_backend\_version\_major: At present, we don't have a primitive for the major version!

```

\__pdf_backend_version_minor: 2280 \cs_new:Npx \__pdf_backend_version_major:
2281 {
2282   \cs_if_exist:NTF \tex_pdfvariable:D
2283     {
2284       \int_compare:nNnTF \tex_luatexversion:D > { 106 }
2285         { \exp_not:N \tex_the:D \tex_pdfvariable:D majorversion }
2286         { 1 }
2287     }
2288     {
2289       \cs_if_exist:NTF \tex_pdfmajorversion:D
2290         { \exp_not:N \tex_the:D \tex_pdfmajorversion:D }
2291         { 1 }
2292     }
2293 }
2294 \cs_new:Npx \__pdf_backend_version_minor:
2295 {
2296   \exp_not:N \tex_the:D
2297   \cs_if_exist:NTF \tex_pdfminorversion:D
2298     { \exp_not:N \tex_pdfminorversion:D }
2299     { \tex_pdfvariable:D minorversion }
2300 }

```

(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>.

```

2301 \cs_new_protected:Npn \__pdf_backend_bdc:nn #1#2
2302 { \__kernel_backend_literal_page:n { /#1 ~ #2 ~ BDC } }
2303 \cs_new_protected:Npn \__pdf_backend_emc:
2304 { \__kernel_backend_literal_page:n { EMC } }

```

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

2305 </pdfmode>

## 6.4 dvipdfmx backend

```

2306 \*dvipdfmx | xdvipdfmx\
\__pdf_backend:n A generic function for the backend PDF specials: used where we can.
\__pdf_backend:x
2307 \cs_new_protected:Npx \__pdf_backend:n #1
2308 { \__kernel_backend_literal:n { pdf: #1 } }
2309 \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
2310 \cs_new_protected:Npn \__pdf_backend_catalog_gput:nn #1#2
2311 { \__pdf_backend:n { put ~ @catalog << /#1 ~ #2 >> } }
2312 \cs_new_protected:Npn \__pdf_backend_info_gput:nn #1#2
2313 { \__pdf_backend:n { docinfo << /#1 ~ #2 >> } }

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

```

### 6.4.2 Objects

```

\g__pdf_backend_object_int For tracking objects to allow finalisation.
\g__pdf_backend_object_prop
2314 \int_new:N \g__pdf_backend_object_int
2315 \prop_new:N \g__pdf_backend_object_prop

(End definition for \g__pdf_backend_object_int and \g__pdf_backend_object_prop.)

\__pdf_backend_object_new:nn Objects are tracked at the macro level, but we don't have to do anything at this stage.
\__pdf_backend_object_ref:n
2316 \cs_new_protected:Npn \__pdf_backend_object_new:nn #1#2
2317 {
2318   \int_gincr:N \g__pdf_backend_object_int
2319   \int_const:cn
2320     { c__pdf_backend_object_ \tl_to_str:n {#1} _int }
2321     { \g__pdf_backend_object_int }
2322   \prop_gput:Nnn \g__pdf_backend_object_prop {#1} {#2}
2323 }
2324 \cs_new:Npn \__pdf_backend_object_ref:n #1
2325 { @pdf.obj \int_use:c { c__pdf_backend_object_ \tl_to_str:n {#1} _int } }

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

```

```

\__pdf_backend_object_write:nn This is where we choose the actual type.
\__pdf_backend_object_write:nx
\__pdf_backend_object_write:nnn
\__pdf_backend_object_write_array:nn
\__pdf_backend_object_write_dict:nn
\__pdf_backend_object_write_fstream:nn
\__pdf_backend_object_write_stream:nn
\__pdf_backend_object_write_stream:nnn
\__pdf_backend_object_write:nn
2326 \cs_new_protected:Npn \__pdf_backend_object_write:nn #1#2
2327 {
2328   \exp_args:Nx \__pdf_backend_object_write:nnn
2329   { \prop_item:Nn \g__pdf_backend_object_prop {#1} } {#1} {#2}
2330 }
2331 \cs_generate_variant:Nn \__pdf_backend_object_write:nn { nx }
2332 \cs_new_protected:Npn \__pdf_backend_object_write:nnn #1#2#3
2333 {
2334   \use:c { \__pdf_backend_object_write_ #1 :nn }
2335   { \__pdf_backend_object_ref:n {#2} } {#3}
2336 }

```

```

2337 \cs_new_protected:Npn \__pdf_backend_object_write_array:nn #1#2
2338 {
2339   \__pdf_backend:x
2340   { obj ~ #1 ~ [ ~ \exp_not:n {#2} ~ ] }
2341 }
2342 \cs_new_protected:Npn \__pdf_backend_object_write_dict:nn #1#2
2343 {
2344   \__pdf_backend:x
2345   { obj ~ #1 ~ << ~ \exp_not:n {#2} ~ >> }
2346 }
2347 \cs_new_protected:Npn \__pdf_backend_object_write_fstream:nn #1#2
2348 { \__pdf_backend_object_write_stream:nnnn { f } {#1} #2 }
2349 \cs_new_protected:Npn \__pdf_backend_object_write_stream:nn #1#2
2350 { \__pdf_backend_object_write_stream:nnnn { } {#1} #2 }
2351 \cs_new_protected:Npn \__pdf_backend_object_write_stream:nnnn #1#2#3#4
2352 {
2353   \__pdf_backend:x
2354   {
2355     #1 stream ~ #2 ~
2356     ( \exp_not:n {#4} ) ~ << \exp_not:n {#3} >>
2357   }
2358 }

```

(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
2359 \cs_new_protected:Npn \__pdf_backend_object_now:nn #1#2
2360 {
2361   \int_gincr:N \g__pdf_backend_object_int
2362   \exp_args:Nnx \use:c { __pdf_backend_object_write_ #1 :nn }
2363   { @pdf.obj \int_use:N \g__pdf_backend_object_int }
2364   {#2}
2365 }
2366 \cs_generate_variant:Nn \__pdf_backend_object_now:nn { nx }

```

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

\\_\_pdf\_backend\_object\_last:

```

2367 \cs_new:Npn \__pdf_backend_object_last:
2368 { @pdf.obj \int_use:N \g__pdf_backend_object_int }

```

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

### 6.4.3 Annotations

\g\_\_pdf\_landscape\_bool There is a bug in (x)dvipdfmx which means annotations do not rotate. As such, we need to know if landscape is active.

```

2369 \bool_new:N \g__pdf_landscape_bool
2370 <*package>
2371 \AtBeginDocument
2372 {
2373   \cs_if_exist:NT \landscape
2374   {
2375     \tl_put_right:Nn \landscape

```

```

2376         { \bool_gset_true:N \g__pdf_landscape_bool }
2377         \tl_put_left:Nn \endlandscape
2378         { \bool_gset_false:N \g__pdf_landscape_bool }
2379     }
2380 }
2381 \</package>

```

(End definition for \g\_\_pdf\_landscape\_bool.)

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

```

2382 \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.

```

2383 \cs_new_protected:Npn \__pdf_backend_annotation:nnnn #1#2#3#4
2384 {
2385     \bool_if:NTF \g__pdf_landscape_bool
2386     {
2387         \box_move_up:nn {#2}
2388         {
2389             \vbox:n
2390             {
2391                 \__pdf_backend_annotation_aux:nnnn
2392                 { #2 + #3 } {#1} { Opt } {#4}
2393             }
2394         }
2395     }
2396     { \__pdf_backend_annotation_aux:nnnn {#1} {#2} {#3} {#4} }
2397 }
2398 \cs_new_protected:Npn \__pdf_backend_annotation_aux:nnnn #1#2#3#4
2399 {
2400     \int_gincr:N \g__pdf_backend_object_int
2401     \int_gset_eq:NN \g__pdf_backend_annotation_int \g__pdf_backend_object_int
2402     \__pdf_backend:x
2403     {
2404         ann ~ @pdf.obj \int_use:N \g__pdf_backend_object_int \c_space_tl
2405         width ~ \dim_eval:n {#1} ~
2406         height ~ \dim_eval:n {#2} ~
2407         depth ~ \dim_eval:n {#3} ~
2408         <</Type/Annot #4 >>
2409     }
2410 }

```

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

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

```

2411 \cs_new:Npn \__pdf_backend_annotation_last:
2412 { @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:
2413 \cs_new_protected:Npn \__pdf_backend_link_begin_goto:nnw #1#2
2414 { \__pdf_backend_link_begin:n { #1 /Subtype /Link /A << /S /GoTo /D ( #2 ) >> } }
2415 \cs_new_protected:Npn \__pdf_backend_link_begin_user:nnw #1#2
2416 { \__pdf_backend_link_begin:n {#1#2} }
2417 \cs_new_protected:Npn \__pdf_backend_link_begin:n #1
2418 {
2419   \__pdf_backend:n
2420   {
2421     bann
2422     <<
2423     /Type /Annot
2424     #1
2425     >>
2426   }
2427 }
2428 \cs_new_protected:Npn \__pdf_backend_link_end:
2429 { \__pdf_backend:n { eann } }

```

All created using the same internals.

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

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

```
2430 \cs_new:Npn \__pdf_backend_link_last: { }
```

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

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

```
2431 \cs_new_protected:Npn \__pdf_backend_link_margin:n #1
2432 { \__kernel_backend_literal:x { dvipdfmx:config~g~ \dim_eval:n {#1} } }
```

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

\\_\_pdf\_backend\_destination:nn Here, we need to turn the zoom into a scale. The method for FitR is from Alexander Grahn: the idea is to avoid needing to do any calculations in T<sub>E</sub>X by using the backend data for @xpos and @ypos.

```

2433 \cs_new_protected:Npn \__pdf_backend_destination:nn #1#2
2434 {
2435   \__pdf_backend:x
2436   {
2437     dest ~ ( \exp_not:n {#1} )
2438     [
2439       @thispage
2440       \str_case:nnF {#2}
2441       {
2442         { xyz } { /XYZ ~ @xpos ~ @ypos ~ null }
2443         { fit } { /Fit }
2444         { fitb } { /FitB }
2445         { fitbh } { /FitBH }
2446         { fitbv } { /FitBV ~ @xpos }
2447         { fith } { /FitH ~ @ypos }
2448         { fitv } { /FitV ~ @xpos }
2449       }
2450       { /XYZ ~ @xpos ~ @ypos ~ \fp_eval:n { (#2) / 100 } }
2451     ]

```

```

2452     }
2453   }
2454   \cs_new_protected:Npn \__pdf_backend_destination_rectangle:nn #1#2
2455   {
2456     \group_begin:
2457     \hbox_set:Nn \l__pdf_internal_box {#2}
2458     \box_move_down:nn { \box_dp:N \l__pdf_internal_box }
2459     {
2460       \hbox:n
2461       {
2462         \__pdf_backend:n { obj ~ @pdf_ #1 _llx ~ @xpos }
2463         \__pdf_backend:n { obj ~ @pdf_ #1 _lly ~ @ypos }
2464       }
2465     }
2466     \box_use:N \l__pdf_internal_box
2467     \box_move_up:nn { \box_ht:N \l__pdf_internal_box }
2468     {
2469       \hbox:n
2470       {
2471         \__pdf_backend:n
2472         {
2473           dest ~ (#1)
2474           [
2475             @thispage
2476             /FitR ~
2477             @pdf_ #1 _llx ~ @pdf_ #1 _lly ~
2478             @xpos ~ @ypos
2479           ]
2480         }
2481       }
2482     }
2483   \group_end:
2484 }

```

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

#### 6.4.4 Structure

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

```

\__pdf_backend_compress_objects:n
2485 \cs_new_protected:Npn \__pdf_backend_compresslevel:n #1
2486 { \__kernel_backend_literal:x { dvipdfmx:config-z~ \int_eval:n {#1} } }
2487 \cs_new_protected:Npn \__pdf_backend_compress_objects:n #1
2488 {
2489   \bool_if:nF {#1}
2490   { \__kernel_backend_literal:n { dvipdfmx:config-C~0x40 } }
2491 }

```

(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
2492 \cs_new_protected:Npn \__pdf_backend_version_major_gset:n #1
2493 {
2494   \cs_gset:Npx \__pdf_backend_version_major: { \int_eval:n {#1} }
2495   \__kernel_backend_literal:x { pdf:majorversion~ \__pdf_backend_version_major: }

```



```

2496 }
2497 \cs_new_protected:Npn \__pdf_backend_version_minor_gset:n #1
2498 {
2499   \cs_gset:Npx \__pdf_backend_version_minor: { \int_eval:n {#1} }
2500   \__kernel_backend_literal:x { pdf:minorversion~ \__pdf_backend_version_minor: }
2501 }

```

(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:
2502 \cs_new:Npn \__pdf_backend_version_major: { 1 }
2503 \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>.

\\_\_pdf\_backend\_emc:

```

2504 \cs_new_protected:Npn \__pdf_backend_bdc:nn #1#2
2505 { \__kernel_backend_literal_page:n { /#1 ~ #2 ~ BDC } }
2506 \cs_new_protected:Npn \__pdf_backend_emc:
2507 { \__kernel_backend_literal_page:n { EMC } }

```

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

```

2508 </dvipdfmx | xdvipdfmx>

```

### 6.5 dvisvgm backend

```

2509 <*dvisvgm>

```

#### 6.5.1 Catalogue entries

\\_\_pdf\_backend\_catalog\_gput:nn No-op.

\\_\_pdf\_backend\_info\_gput:nn

```

2510 \cs_new_protected:Npn \__pdf_backend_catalog_gput:nn #1#2 { }
2511 \cs_new_protected:Npn \__pdf_backend_info_gput:nn #1#2 { }

```

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

#### 6.5.2 Objects

\\_\_pdf\_backend\_object\_new:nn All no-ops here.

\\_\_pdf\_backend\_object\_ref:n

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

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

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

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

\\_\_pdf\_backend\_object\_last:

```

2512 \cs_new_protected:Npn \__pdf_backend_object_new:nn #1#2 { }
2513 \cs_new:Npn \__pdf_backend_object_ref:n #1 { }
2514 \cs_new_protected:Npn \__pdf_backend_object_write:nn #1#2 { }
2515 \cs_new_protected:Npn \__pdf_backend_object_write:nx #1#2 { }
2516 \cs_new_protected:Npn \__pdf_backend_object_now:nn #1#2 { }
2517 \cs_new_protected:Npn \__pdf_backend_object_now:nx #1#2 { }
2518 \cs_new:Npn \__pdf_backend_object_last: { }

```

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

### 6.5.3 Structure

```

\_pdf_backend_compresslevel:n These are all no-ops.
\_pdf_backend_compress_objects:n
2519 \cs_new_protected:Npn \__pdf_backend_compresslevel:n #1 { }
2520 \cs_new_protected:Npn \__pdf_backend_compress_objects:n #1 { }

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

\_pdf_backend_version_major_gset:n Data not available!
\_pdf_backend_version_minor_gset:n
2521 \cs_new_protected:Npn \__pdf_backend_version_major_gset:n #1 { }
2522 \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:
2523 \cs_new:Npn \__pdf_backend_version_major: { -1 }
2524 \cs_new:Npn \__pdf_backend_version_minor: { -1 }

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

\_pdf_backend_bdc:nn More no-ops.
\_pdf_backend_emc:
2525 \cs_new_protected:Npn \__pdf_backend_bdc:nn #1#2 { }
2526 \cs_new_protected:Npn \__pdf_backend_emc: { }

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

2527 </dvisvgm>
2528 </initex | package>

```

## 7 l3backend-header Implementation

```

2529 <*dvips & header>

pdf.globaldict A small global dictionary for backend use.
2530 true setglobal
2531 /pdf.globaldict 4 dict def
2532 false setglobal

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

pdf.cvs Small utilities for PostScript manipulations. Conversion to DVI dimensions is done here
pdf.dvi.pt to allow for Resolution. The total height of a rectangle (an array) needs a little maths,
pdf.pt.dvi in contrast to simply extracting a value.
pdf.rect.ht
2533
2534 /pdf.cvs { 65534 string cvs } def
2535 /pdf.dvi.pt { 72.27 mul Resolution div } def
2536 /pdf.pt.dvi { 72.27 div Resolution mul } def
2537 /pdf.rect.ht { dup 1 get neg exch 3 get add } def

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

pdf.linkmargin Settings which are defined up-front in SDict.
pdf.linkdp.pad
2538 /pdf.linkmargin { 1 pdf.pt.dvi } def
pdf.linkht.pad
2539 /pdf.linkdp.pad { 0 } def
2540 /pdf.linkht.pad { 0 } def

```

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

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

```
pdf.save.linkll 2541 /pdf.rect
pdf.save.linkur 2542 { /Rect [ pdf.llx pdf.lly pdf.urx pdf.ury ] } def
pdf.llx         2543 /pdf.save.ll
pdf.lly         2544 {
pdf.urx         2545     currentpoint
pdf.ury         2546     /pdf.lly exch def
                2547     /pdf.llx exch def
                2548 }
                2549 def
pdf.save.ur     2550 /pdf.save.ur
                2551 {
                2552     currentpoint
                2553     /pdf.ury exch def
                2554     /pdf.urx exch def
                2555 }
                2556 def
pdf.save.linkll 2557 /pdf.save.linkll
                2558 {
                2559     currentpoint
                2560     pdf.linkmargin add
                2561     pdf.linkdp.pad add
                2562     /pdf.lly exch def
                2563     pdf.linkmargin sub
                2564     /pdf.llx exch def
                2565 }
                2566 def
pdf.save.linkur 2567 /pdf.save.linkur
                2568 {
                2569     currentpoint
                2570     pdf.linkmargin sub
                2571     pdf.linkht.pad sub
                2572     /pdf.ury exch def
                2573     pdf.linkmargin add
                2574     /pdf.urx exch def
                2575 }
                2576 def
```

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

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

```
pdf.dev.x 2577 /pdf.dest.anchor
pdf.dev.y 2578 {
pdf.tmpa 2579     currentpoint exch
pdf.tmpb 2580     pdf.dvi.pt 72 add
pdf.tmpc
pdf.tmpd
```

```

2581     /pdf.dest.x exch def
2582     pdf.dvi.pt
2583     vsize 72 sub exch sub
2584     /pdf.dest.y exch def
2585   }
2586   def
2587 /pdf.dest.point
2588   { pdf.dest.x pdf.dest.y } def
2589 /pdf.dest2device
2590   {
2591     /pdf.dest.y exch def
2592     /pdf.dest.x exch def
2593     matrix currentmatrix
2594     matrix defaultmatrix
2595     matrix invertmatrix
2596     matrix concatmatrix
2597     cvx exec
2598     /pdf.dev.y exch def
2599     /pdf.dev.x exch def
2600     /pdf.tmpd exch def
2601     /pdf.tmpc exch def
2602     /pdf.tmpb exch def
2603     /pdf.tmpa exch def
2604     pdf.dest.x pdf.tmpa mul
2605       pdf.dest.y pdf.tmpc mul add
2606       pdf.dev.x add
2607     pdf.dest.x pdf.tmpb mul
2608       pdf.dest.y pdf.tmpd mul add
2609       pdf.dev.y add
2610   }
2611   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	2612 /pdf.bordertracking false def
pdf.brokenlink.skip	2613 /pdf.bordertracking.begin
pdf.brokenlink.dict	2614 {
pdf.bordertracking.endpage	2615   SDict /pdf.bordertracking true put
pdf.bordertracking.continue	2616   SDict /pdf.leftboundary undef
pdf.originx	2617   SDict /pdf.rightboundary undef
pdf.originy	2618   /a where
	2619     {
	2620       /a
	2621       {
	2622         currentpoint pop
	2623         SDict /pdf.rightboundary known dup
	2624         {
	2625           SDict /pdf.rightboundary get 2 index lt
	2626           { not }
	2627         if

```

2628     }
2629     if
2630     { pop }
2631     { SDict exch /pdf.rightboundary exch put }
2632     ifelse
2633     moveto
2634     currentpoint pop
2635     SDict /pdf.leftboundary known dup
2636     {
2637         SDict /pdf.leftboundary get 2 index gt
2638         { not }
2639         if
2640     }
2641     if
2642     { pop }
2643     { SDict exch /pdf.leftboundary exch put }
2644     ifelse
2645 }
2646 put
2647 }
2648 if
2649 }
2650 def
2651 /pdf.bordertracking.end
2652 {
2653     /a where { /a { moveto } put } if
2654     /x where { /x { 0 exch rmoveto } put } if
2655     SDict /pdf.leftboundary known
2656     { pdf.outerbox 0 pdf.leftboundary put }
2657     if
2658     SDict /pdf.rightboundary known
2659     { pdf.outerbox 2 pdf.rightboundary put }
2660     if
2661     SDict /pdf.bordertracking false put
2662 }
2663 def
2664 /pdf.bordertracking.endpage
2665 {
2666     pdf.bordertracking
2667     {
2668         pdf.bordertracking.end
2669         true setglobal
2670         pdf.globaldict
2671         /pdf.brokenlink.rect [ pdf.outerbox aload pop ] put
2672         pdf.globaldict
2673         /pdf.brokenlink.skip pdf.baselineskip put
2674         pdf.globaldict
2675         /pdf.brokenlink.dict
2676         pdf.link.dict pdf.cvs put
2677         false setglobal
2678         mark pdf.link.dict cvx exec /Rect
2679         [
2680             pdf.llx
2681             pdf.lly

```

```

2682         pdf.outerbox 2 get pdf.linkmargin add
2683         currentpoint exch pop
2684         pdf.outerbox pdf.rect.ht sub pdf.linkmargin sub
2685     ]
2686     /ANN pdf.pdfmark
2687 }
2688 if
2689 }
2690 def
2691 /pdf.bordertracking.continue
2692 {
2693     /pdf.link.dict pdf.globaldict
2694     /pdf.brokenlink.dict get def
2695     /pdf.outerbox pdf.globaldict
2696     /pdf.brokenlink.rect get def
2697     /pdf.baselineskip pdf.globaldict
2698     /pdf.brokenlink.skip get def
2699     pdf.globaldict dup dup
2700     /pdf.brokenlink.dict undef
2701     /pdf.brokenlink.skip undef
2702     /pdf.brokenlink.rect undef
2703     currentpoint
2704     /pdf.originy exch def
2705     /pdf.originx exch def
2706     /a where
2707     {
2708         /a
2709         {
2710             moveto
2711             SDict
2712             begin
2713                 currentpoint pdf.originy ne exch
2714                 pdf.originx ne or
2715                 {
2716                     pdf.save.linkll
2717                     /pdf.lly
2718                     pdf.lly pdf.outerbox 1 get sub def
2719                     pdf.bordertracking.begin
2720                 }
2721                 if
2722                 end
2723             }
2724             put
2725         }
2726     if
2727     /x where
2728     {
2729         /x
2730         {
2731             0 exch rmoveto
2732             SDict~
2733             begin
2734                 currentpoint
2735                 pdf.originy ne exch pdf.originx ne or

```

```

2736         {
2737             pdf.save.linkll
2738             /pdf.lly
2739             pdf.lly pdf.outerbox 1 get sub def
2740             pdf.bordertracking.begin
2741         }
2742         if
2743         end
2744     }
2745     put
2746 }
2747 if
2748 }
2749 def

```

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

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

```

2750 /pdf.breaklink
2751 {
2752     pop
2753     counttomark 2 mod 0 eq
2754     {
2755         counttomark /pdf.count exch def
2756         {
2757             pdf.count 0 eq { exit } if
2758             counttomark 2 roll
2759             1 index /Rect eq
2760             {
2761                 dup 4 array copy
2762                 dup dup
2763                 1 get
2764                 pdf.outerbox pdf.rect.ht
2765                 pdf.linkmargin 2 mul add sub
2766                 3 exch put
2767                 dup
2768                 pdf.outerbox 2 get
2769                 pdf.linkmargin add
2770                 2 exch put
2771                 dup dup
2772                 3 get
2773                 pdf.outerbox pdf.rect.ht
2774                 pdf.linkmargin 2 mul add add
2775                 1 exch put
2776             /pdf.currentrect exch def
2777             pdf.breaklink.write
2778             {
2779                 pdf.currentrect
2780                 dup
2781                 pdf.outerbox 0 get

```

```

2782         pdf.linkmargin sub
2783         0 exch put
2784     dup
2785         pdf.outerbox 2 get
2786         pdf.linkmargin add
2787         2 exch put
2788     dup dup
2789         1 get
2790         pdf.baselineskip add
2791         1 exch put
2792     dup dup
2793         3 get
2794         pdf.baselineskip add
2795         3 exch put
2796     /pdf.currentrect exch def
2797     pdf.breaklink.write
2798 }
2799     1 index 3 get
2800     pdf.linkmargin 2 mul add
2801     pdf.outerbox pdf.rect.ht add
2802     2 index 1 get sub
2803     pdf.baselineskip div round cvi 1 sub
2804     exch
2805     repeat
2806     pdf.currentrect
2807     dup
2808         pdf.outerbox 0 get
2809         pdf.linkmargin sub
2810         0 exch put
2811     dup dup
2812         1 get
2813         pdf.baselineskip add
2814         1 exch put
2815     dup dup
2816         3 get
2817         pdf.baselineskip add
2818         3 exch put
2819     dup 2 index 2 get 2 exch put
2820     /pdf.currentrect exch def
2821     pdf.breaklink.write
2822     SDict /pdf.pdfmark.good false put
2823     exit
2824 }
2825 { pdf.count 2 sub /pdf.count exch def }
2826 ifelse
2827 }
2828 loop
2829 }
2830 if
2831 /ANN
2832 }
2833 def
2834 /pdf.breaklink.write
2835 {

```



```

2836     counttomark 1 sub
2837     index /_objdef eq
2838     {
2839         counttomark -2 roll
2840         dup wcheck
2841         {
2842             readonly
2843             counttomark 2 roll
2844         }
2845         { pop pop }
2846         ifelse
2847     }
2848     if
2849     counttomark 1 add copy
2850     pop pdf.currentrect
2851     /ANN pdfmark
2852 }
2853 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.

```

2854 /pdf.pdfmark
2855 {
2856     SDict /pdf.pdfmark.good true put
2857     dup /ANN eq
2858     {
2859         pdf.pdfmark.store
2860         pdf.pdfmark.dict
2861         begin
2862             Subtype /Link eq
2863             currentdict /Rect known and
2864             SDict /pdf.outerbox known and
2865             SDict /pdf.baselineskip known and
2866             {
2867                 Rect 3 get
2868                 pdf.linkmargin 2 mul add
2869                 pdf.outerbox pdf.rect.ht add
2870                 Rect 1 get sub
2871                 pdf.baselineskip div round cvi 0 gt
2872                 { pdf.breaklink }
2873                 if
2874             }
2875             if
2876         end
2877         SDict /pdf.outerbox undef
2878         SDict /pdf.baselineskip undef
2879         currentdict /pdf.pdfmark.dict undef
2880     }
2881     if

```

```

2882     pdf.pdfmark.good
2883     { pdfmark }
2884     { cleartomark }
2885     ifelse
2886   }
2887   def
2888 /pdf.pdfmark.store
2889 {
2890   /pdf.pdfmark.dict 65534 dict def
2891   counttomark 1 add copy
2892   pop
2893   {
2894     dup mark eq
2895     {
2896       pop
2897       exit
2898     }
2899     {
2900       pdf.pdfmark.dict
2901       begin def end
2902     }
2903     ifelse
2904   }
2905   loop
2906 }
2907 def

```

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

```

2908 </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>\AtBeginDocument</code> . . . . .	
. . . . .	<a href="#">372</a> , <a href="#">429</a> , <a href="#">1314</a> , <a href="#">1449</a> , <a href="#">1613</a> , <a href="#">2371</a>
<code>\AtBeginDvi</code> . . . . .	<a href="#">36</a> , <a href="#">37</a>
<b>B</b>	
<code>\begin</code> . . . . .	<a href="#">1363</a> , <a href="#">1368</a>
bool commands:	
<code>\bool_gset_false:N</code> . . . . .	
. . . . .	<a href="#">575</a> , <a href="#">591</a> , <a href="#">617</a> , <a href="#">639</a> ,
. . . . .	<a href="#">655</a> , <a href="#">807</a> , <a href="#">1133</a> , <a href="#">1169</a> , <a href="#">1818</a> , <a href="#">1869</a> , <a href="#">2378</a>
<code>\bool_gset_true:N</code> . . . . .	
. . . . .	<a href="#">573</a> , <a href="#">642</a> , <a href="#">805</a> , <a href="#">1148</a> , <a href="#">1811</a> , <a href="#">1817</a> , <a href="#">2376</a>
<code>\bool_if:NTF</code> . . . . .	
. . . . .	<a href="#">40</a> , <a href="#">582</a> , <a href="#">586</a> , <a href="#">604</a> , <a href="#">608</a> , <a href="#">612</a> ,
. . . . .	<a href="#">625</a> , <a href="#">630</a> , <a href="#">634</a> , <a href="#">646</a> , <a href="#">650</a> , <a href="#">818</a> , <a href="#">823</a> ,
. . . . .	<a href="#">828</a> , <a href="#">1107</a> , <a href="#">1152</a> , <a href="#">1339</a> , <a href="#">1380</a> , <a href="#">1496</a> ,
. . . . .	<a href="#">1538</a> , <a href="#">1806</a> , <a href="#">1821</a> , <a href="#">1826</a> , <a href="#">1831</a> , <a href="#">2385</a>
<code>\bool_if:nTF</code> . . . . .	<a href="#">2242</a> , <a href="#">2489</a>
<code>\bool_lazy_or:nnTF</code> . . . . .	<a href="#">1372</a> , <a href="#">1531</a>
<code>\bool_new:N</code> . . . . .	
. . . . .	<a href="#">576</a> , <a href="#">643</a> , <a href="#">808</a> , <a href="#">1149</a> , <a href="#">1794</a> , <a href="#">1795</a> , <a href="#">2369</a>
<code>\bool_set_false:N</code> . . . . .	
. . . . .	<a href="#">1349</a> , <a href="#">1463</a> , <a href="#">1556</a> , <a href="#">1626</a>
box commands:	
<code>\box_dp:N</code> . . . . .	<a href="#">135</a> , <a href="#">137</a> , <a href="#">185</a> , <a href="#">187</a> ,
. . . . .	<a href="#">242</a> , <a href="#">244</a> , <a href="#">291</a> , <a href="#">293</a> , <a href="#">295</a> , <a href="#">297</a> , <a href="#">1848</a> ,
. . . . .	<a href="#">1881</a> , <a href="#">1882</a> , <a href="#">1907</a> , <a href="#">2009</a> , <a href="#">2121</a> , <a href="#">2458</a>
<code>\box_ht:N</code> . . . . .	<a href="#">137</a> , <a href="#">187</a> ,
. . . . .	<a href="#">244</a> , <a href="#">295</a> , <a href="#">297</a> , <a href="#">1392</a> , <a href="#">1593</a> , <a href="#">1853</a> ,
. . . . .	<a href="#">1892</a> , <a href="#">1893</a> , <a href="#">1914</a> , <a href="#">2013</a> , <a href="#">2120</a> , <a href="#">2467</a>
<code>\box_if_empty:N</code> . . . . .	<a href="#">1948</a>
<code>\box_move_down:nn</code> . . . . .	
. . . . .	<a href="#">1769</a> , <a href="#">1848</a> , <a href="#">2008</a> , <a href="#">2458</a>
<code>\box_move_up:nn</code> . . . . .	
. . . . .	<a href="#">1772</a> , <a href="#">1853</a> , <a href="#">2012</a> , <a href="#">2387</a> , <a href="#">2467</a>
<code>\box_new:N</code> . . . . .	<a href="#">1661</a> , <a href="#">1750</a> , <a href="#">1751</a>
<code>\box_set_dp:Nn</code> . . . . .	<a href="#">1300</a>
<code>\box_set_ht:Nn</code> . . . . .	<a href="#">1299</a>
<code>\box_set_wd:Nn</code> . . . . .	<a href="#">199</a> , <a href="#">1298</a>
<code>\box_use:N</code> . . . . .	<a href="#">142</a> , <a href="#">160</a> , <a href="#">174</a> , <a href="#">190</a> , <a href="#">217</a> ,
. . . . .	<a href="#">231</a> , <a href="#">247</a> , <a href="#">263</a> , <a href="#">275</a> , <a href="#">326</a> , <a href="#">343</a> , <a href="#">362</a> ,
. . . . .	<a href="#">758</a> , <a href="#">1015</a> , <a href="#">1301</a> , <a href="#">1799</a> , <a href="#">2011</a> , <a href="#">2122</a> , <a href="#">2466</a>
<code>\box_wd:N</code> . . . . .	<a href="#">136</a> , <a href="#">144</a> , <a href="#">186</a> , <a href="#">192</a> , <a href="#">243</a> ,
. . . . .	<a href="#">249</a> , <a href="#">292</a> , <a href="#">294</a> , <a href="#">330</a> , <a href="#">1391</a> , <a href="#">1592</a> , <a href="#">2119</a>
box internal commands:	
<code>\__box_backend_clip:N</code> . . . . .	
. . . . .	<a href="#">124</a> , <a href="#">179</a> , <a href="#">236</a> , <a href="#">280</a>
<code>\l__box_backend_cos_fp</code> . . . . .	<a href="#">194</a>
<code>\__box_backend_rotate:Nn</code> . . . . .	
. . . . .	<a href="#">146</a> , <a href="#">194</a> , <a href="#">251</a> , <a href="#">333</a>
<code>\__box_backend_rotate_aux:Nn</code> . . . . .	
. . . . .	<a href="#">146</a> , <a href="#">194</a> , <a href="#">251</a>
<code>\__box_backend_scale:Nnn</code> . . . . .	
. . . . .	<a href="#">163</a> , <a href="#">222</a> , <a href="#">266</a> , <a href="#">346</a>
<code>\l__box_backend_sin_fp</code> . . . . .	<a href="#">194</a>
<code>\g__box_clip_path_int</code> . . . . .	<a href="#">280</a>
<b>C</b>	
clist commands:	
<code>\clist_map_function:nN</code> . . . . .	<a href="#">663</a> , <a href="#">838</a>
<code>\clist_map_function:nn</code> . . . . .	<a href="#">1176</a>
color internal commands:	
<code>\__color_backend_cmyk:nnnn</code> . . . . .	<a href="#">396</a> , <a href="#">465</a>
<code>\__color_backend_cmyk_aux:nnnn</code> . . . . .	<a href="#">465</a>
<code>\__color_backend_gray:n</code> . . . . .	<a href="#">396</a> , <a href="#">465</a>
<code>\__color_backend_gray_aux:n</code> . . . . .	<a href="#">465</a>
<code>\__color_backend_pickup:N</code> . . . . .	<a href="#">370</a> , <a href="#">427</a>
<code>\__color_backend_pickup:w</code> . . . . .	<a href="#">13</a> , <a href="#">370</a> , <a href="#">427</a>
<code>\__color_backend_reset:</code> . . . . .	<a href="#">396</a> , <a href="#">465</a>
<code>\__color_backend_rgb:nnn</code> . . . . .	<a href="#">396</a> , <a href="#">465</a>
<code>\__color_backend_rgb_aux:nnn</code> . . . . .	<a href="#">465</a>
<code>\__color_backend_select:n</code> . . . . .	<a href="#">396</a> , <a href="#">465</a>
<code>\__color_backend_spot:nn</code> . . . . .	<a href="#">396</a> , <a href="#">465</a>
<code>color.fc</code> . . . . .	<a href="#">396</a> , <a href="#">522</a>
cs commands:	
<code>\cs_generate_variant:Nn</code> . . . . .	<a href="#">28</a> ,
. . . . .	<a href="#">32</a> , <a href="#">35</a> , <a href="#">67</a> , <a href="#">95</a> , <a href="#">100</a> , <a href="#">111</a> , <a href="#">118</a> , <a href="#">422</a> ,
. . . . .	<a href="#">507</a> , <a href="#">521</a> , <a href="#">726</a> , <a href="#">732</a> , <a href="#">768</a> , <a href="#">916</a> , <a href="#">1024</a> ,
. . . . .	<a href="#">1055</a> , <a href="#">1510</a> , <a href="#">1567</a> , <a href="#">1583</a> , <a href="#">1665</a> , <a href="#">1702</a> ,
. . . . .	<a href="#">1747</a> , <a href="#">2192</a> , <a href="#">2223</a> , <a href="#">2309</a> , <a href="#">2331</a> , <a href="#">2366</a>
<code>\cs_gset:Npx</code> . . . . .	<a href="#">2494</a> , <a href="#">2499</a>
<code>\cs_if_exist:N</code> . . . . .	<a href="#">36</a> , <a href="#">62</a> , <a href="#">70</a> , <a href="#">78</a> ,
. . . . .	<a href="#">84</a> , <a href="#">90</a> , <a href="#">374</a> , <a href="#">431</a> , <a href="#">501</a> , <a href="#">510</a> , <a href="#">899</a> , <a href="#">907</a> ,
. . . . .	<a href="#">1944</a> , <a href="#">2042</a> , <a href="#">2053</a> , <a href="#">2064</a> , <a href="#">2072</a> , <a href="#">2079</a> ,
. . . . .	<a href="#">2086</a> , <a href="#">2093</a> , <a href="#">2114</a> , <a href="#">2127</a> , <a href="#">2134</a> , <a href="#">2152</a> ,
. . . . .	<a href="#">2188</a> , <a href="#">2219</a> , <a href="#">2227</a> , <a href="#">2235</a> , <a href="#">2249</a> , <a href="#">2256</a> ,
. . . . .	<a href="#">2265</a> , <a href="#">2275</a> , <a href="#">2282</a> , <a href="#">2289</a> , <a href="#">2297</a> , <a href="#">2373</a>
<code>\cs_new:Npn</code> . . . . .	<a href="#">668</a> ,
. . . . .	<a href="#">843</a> , <a href="#">1180</a> , <a href="#">1596</a> , <a href="#">1605</a> , <a href="#">1655</a> , <a href="#">1680</a> ,
. . . . .	<a href="#">1748</a> , <a href="#">1789</a> , <a href="#">1970</a> , <a href="#">2032</a> , <a href="#">2033</a> , <a href="#">2160</a> ,
. . . . .	<a href="#">2193</a> , <a href="#">2194</a> , <a href="#">2324</a> , <a href="#">2367</a> , <a href="#">2411</a> , <a href="#">2430</a> ,
. . . . .	<a href="#">2502</a> , <a href="#">2503</a> , <a href="#">2513</a> , <a href="#">2518</a> , <a href="#">2523</a> , <a href="#">2524</a>
<code>\cs_new:Npx</code> . . . . .	<a href="#">2050</a> , <a href="#">2076</a> , <a href="#">2224</a> , <a href="#">2280</a> , <a href="#">2294</a>
<code>\cs_new_eq:NN</code> . . . . .	
. . . . .	<a href="#">25</a> , <a href="#">520</a> , <a href="#">767</a> , <a href="#">773</a> , <a href="#">774</a> , <a href="#">914</a> , <a href="#">1023</a> ,

1316, 1345, 1402, 1403, 1451, 1459,  
1481, 1552, 1615, 1622, 1654, 1799

**D**

dim commands:

\cs\_new\_protected:Npn .....  
 .... 26, 30, 33, 43, 49, 54, 56, 98,  
 101, 103, 105, 109, 112, 114, 116,  
 124, 146, 148, 163, 179, 194, 196,  
 222, 236, 251, 253, 266, 280, 333,  
 346, 371, 391, 396, 405, 407, 412,  
 414, 423, 428, 438, 465, 476, 481,  
 483, 485, 495, 497, 522, 528, 533,  
 535, 537, 545, 553, 562, 572, 574,  
 577, 579, 593, 598, 619, 641, 644,  
 657, 670, 675, 677, 679, 681, 683,  
 685, 687, 689, 698, 707, 709, 711,  
 716, 721, 727, 733, 745, 769, 771,  
 775, 780, 785, 795, 804, 806, 809,  
 811, 813, 815, 820, 825, 830, 832,  
 845, 850, 852, 854, 856, 858, 860,  
 862, 864, 873, 882, 884, 886, 891,  
 917, 932, 957, 969, 981, 993, 1000,  
 1025, 1030, 1032, 1040, 1050, 1058,  
 1063, 1068, 1079, 1089, 1099, 1101,  
 1103, 1105, 1136, 1138, 1143, 1145,  
 1147, 1150, 1171, 1182, 1195, 1197,  
 1199, 1201, 1203, 1205, 1207, 1209,  
 1211, 1221, 1230, 1238, 1240, 1242,  
 1252, 1267, 1272, 1287, 1317, 1331,  
 1346, 1358, 1369, 1397, 1409, 1422,  
 1432, 1453, 1460, 1468, 1479, 1483,  
 1486, 1501, 1511, 1546, 1553, 1559,  
 1565, 1568, 1575, 1584, 1589, 1597,  
 1616, 1623, 1629, 1631, 1633, 1644,  
 1663, 1666, 1668, 1672, 1682, 1703,  
 1708, 1713, 1718, 1727, 1753, 1767,  
 1798, 1800, 1802, 1804, 1809, 1824,  
 1829, 1871, 1900, 1924, 1933, 1972,  
 1979, 2004, 2028, 2029, 2030, 2031,  
 2034, 2036, 2058, 2060, 2165, 2198,  
 2240, 2301, 2303, 2310, 2312, 2316,  
 2326, 2332, 2337, 2342, 2347, 2349,  
 2351, 2359, 2383, 2398, 2413, 2415,  
 2417, 2428, 2431, 2433, 2454, 2485,  
 2487, 2492, 2497, 2504, 2506, 2510,  
 2511, 2512, 2514, 2515, 2516, 2517,  
 2519, 2520, 2521, 2522, 2525, 2526

\cs\_new\_protected:Npx 60, 68, 76, 82,  
 88, 499, 508, 897, 905, 2040, 2062,  
 2070, 2084, 2091, 2110, 2125, 2132,  
 2143, 2232, 2246, 2254, 2272, 2307

\cs\_set\_eq:NN ..... 1965, 1966

\cs\_set\_protected:Npn .....  
 ..... 376, 433, 2141, 2163, 2196

\dim\_eval:n ..... 1776, 2045, 2046,  
 2047, 2089, 2405, 2406, 2407, 2432  
 \dim\_max:nn ..... 1879, 1890  
 \dim\_set:Nn ... 1391, 1392, 1592, 1593  
 \dim\_to\_decimal:n 291, 292, 293, 294,  
 295, 297, 1061, 1066, 1072, 1073,  
 1074, 1075, 1084, 1085, 1086, 1177,  
 1196, 1649, 1650, 1877, 1888, 1906,  
 1907, 1909, 1912, 1914, 1918, 1976  
 \dim\_to\_decimal\_in\_bp:n . 135, 136,  
 137, 185, 186, 187, 242, 243, 244,  
 541, 542, 549, 550, 557, 558, 566,  
 567, 568, 665, 669, 673, 778, 783,  
 789, 790, 791, 799, 800, 840, 844,  
 848, 1181, 1322, 1323, 1324, 1325,  
 1473, 1474, 1475, 1476, 1525, 1526,  
 1527, 1528, 1638, 1639, 1640, 1641

draw internal commands:

\_\_draw\_align\_currentpoint... .. 21  
 \_\_draw\_backend\_add\_to\_path:n ...  
 ..... 1058, 1104  
 \_\_draw\_backend\_begin: 522, 769, 1025  
 \_\_draw\_backend\_box\_use:Nnnnn ...  
 ..... 16, 745, 1000, 1287  
 \_\_draw\_backend\_cap\_but: .....  
 ..... 657, 832, 1171  
 \_\_draw\_backend\_cap\_rectangle: ..  
 ..... 657, 832, 1171  
 \_\_draw\_backend\_cap\_round: .....  
 ..... 657, 832, 1171  
 \_\_draw\_backend\_clip: 577, 809, 1103  
 \_\_draw\_backend\_closepath: .....  
 ..... 577, 809, 1103  
 \_\_draw\_backend\_closestroke: ...  
 ..... 577, 809, 1103  
 \_\_draw\_backend\_cm:nnnn ..... 733,  
 753, 754, 755, 917, 1004, 1272, 1290  
 \_\_draw\_backend\_cm\_aux:nnnn ... 917  
 \_\_draw\_backend\_cm\_decompose:nnnnN  
 ..... 927, 956  
 \_\_draw\_backend\_cm\_decompose\_-  
 auxi:nnnnN ..... 956  
 \_\_draw\_backend\_cm\_decompose\_-  
 auxii:nnnnN ..... 956  
 \_\_draw\_backend\_cm\_decompose\_-  
 auxiii:nnnnN ..... 956  
 \_\_draw\_backend\_color\_fill:n .. 689  
 \_\_draw\_backend\_color\_fill:nnn 1211  
 \_\_draw\_backend\_color\_fill\_-  
 cmyk:nnnn ..... 689, 864, 1211  
 \_\_draw\_backend\_color\_fill\_-  
 gray:n ..... 689, 864, 1211

\_draw_backend_color_fill_-		\_draw_backend_nonzero_rule: ...	
rgb:nnn	689, 864, 1211	...	572, 804, 1099
\_draw_backend_color_gray_aux:n		\_draw_backend_path:n	1103
...	1234, 1238	\_draw_backend_rectangle:nnnn	...
\_draw_backend_color_reset: ..	864	...	537, 775, 1058
\_draw_backend_color_select:n	864	\_draw_backend_scope:n	...
\_draw_backend_color_stroke:n	689	...	1028, 1032, 1100, 1102, 1122,
\_draw_backend_color_stroke_-		...	1162, 1184, 1196, 1198, 1200, 1202,
cmyk:nnnn	689, 864, 1211	...	1204, 1206, 1208, 1210, 1254, 1274
\_draw_backend_color_stroke_-		\_draw_backend_scope_begin: ...	
gray:n	689, 864, 1211	...	533, 770, 773, 1027, 1032
\_draw_backend_color_stroke_-		\_draw_backend_scope_end: ...	
rgb:nnn	689, 864, 1211	...	533, 772, 773, 1031, 1032
\_draw_backend_curveto:nnnnnn	...	\_draw_backend_select:n	...
...	537, 775, 1058	...	1223, 1241, 1269
\_draw_backend_dash:n	657, 832, 1171	\_draw_backend_stroke: 577, 809, 1103	
\_draw_backend_dash_aux:nn	1171	\g_draw_clip_path_int	...
\_draw_backend_dash_pattern:nn	...	...	1109, 1112, 1125, 1154, 1157, 1165
...	657, 832, 1171	\_draw_color_reset: ...	730
\_draw_backend_discardpath: ...		\g_draw_draw_clip_bool ...	577, 1103
...	577, 809, 1103	\g_draw_draw_eor_bool	...
\_draw_backend_end: .	522, 769, 1025	...	572, 586, 604,
\_draw_backend_evenodd_rule: ...		...	612, 625, 634, 650, 804, 818, 823, 828
...	572, 804, 1099	\g_draw_draw_path_int	1103
\_draw_backend_fill: 577, 809, 1103		\g_draw_draw_path_tl	...
\_draw_backend_fillstroke: ...		...	1058, 1114, 1130, 1132, 1159, 1168
...	577, 809, 1103	\g_draw_draw_scope_int	1032
\_draw_backend_join_bevel: ...		\l_draw_draw_scope_int	1032
...	657, 832, 1171	\g_draw_path_int	1118, 1135
\_draw_backend_join_miter: ...			
...	657, 832, 1171		
\_draw_backend_join_round: ...			
...	657, 832, 1171		
\_draw_backend_lineto:nn	...		
...	537, 775, 1058		
\_draw_backend_linewidth:n	...		
...	657, 832, 1171		
\_draw_backend_literal:n	520, 525,		
526, 530, 534, 536, 539, 547, 555,			
564, 578, 581, 584, 590, 600, 601,			
602, 607, 610, 616, 621, 622, 623,			
628, 629, 632, 638, 648, 654, 659,			
672, 676, 678, 680, 682, 684, 686,			
688, 735, 747, 748, 749, 750, 751,			
752, 756, 757, 759, 760, 761, 762,			
763, 767, 777, 782, 787, 797, 810,			
812, 814, 817, 822, 827, 831, 834,			
847, 851, 853, 855, 857, 859, 861,			
863, 1023, 1044, 1052, 1110, 1129, 1155			
\_draw_backend_miterlimit:n	...		
...	657, 832, 1171		
\_draw_backend_moveto:nn	...		
...	537, 775, 1058		

  

E	
\endlandscape	2377
\evensidemargin	1845
exp commands:	
\exp_after:wN	383, 1603
\exp_args:Nf	662, 837
\exp_args:NNf	147, 195, 252
\exp_args:Nnx	1744, 2362
\exp_args:NV	378
\exp_args:Nx	...
...	482, 1415, 1436, 1715, 1837, 2328
\exp_last_unbraced:Nx	387, 435
\exp_not:N	37,
65, 74, 93, 504, 505, 513, 902, 903,	
910, 2045, 2046, 2047, 2052, 2054,	
2055, 2078, 2080, 2081, 2087, 2088,	
2089, 2094, 2095, 2097, 2107, 2115,	
2116, 2119, 2120, 2121, 2147, 2149,	
2156, 2226, 2228, 2229, 2234, 2238,	
2248, 2260, 2261, 2267, 2268, 2274,	
2276, 2278, 2285, 2290, 2296, 2298	
\exp_not:n	27, 65, 74, 93, 1706,
1711, 2000, 2173, 2174, 2193, 2194,	
2204, 2205, 2340, 2345, 2356, 2437	

## F

file commands:

\file\_compare\_timestamp:nNnTF . 1424  
 \file\_parse\_full\_name:nNNN 1411, 1434

fp commands:

\fp\_compare:nNnTF .....  
 .... 154, 201, 207, 259, 937, 950, 995  
 \fp\_eval:n ..... 147, 156, 169,  
 170, 195, 212, 227, 229, 252, 261,  
 272, 273, 340, 355, 356, 401, 402,  
 406, 410, 470, 471, 472, 473, 482,  
 490, 491, 492, 676, 693, 694, 703,  
 704, 708, 710, 714, 719, 738, 739,  
 851, 868, 869, 877, 878, 883, 885,  
 889, 894, 922, 923, 939, 944, 945,  
 952, 962, 963, 964, 965, 974, 975,  
 976, 977, 986, 987, 988, 989, 1010,  
 1011, 1198, 1216, 1217, 1218, 1226,  
 1227, 1235, 1241, 1247, 1248, 1249,  
 1270, 1280, 1281, 1997, 2107, 2450  
 \fp\_new:N ..... 220, 221  
 \fp\_set:Nn ..... 200, 203  
 \fp\_use:N ..... 206, 210, 215  
 \fp\_zero:N ..... 202  
 \c\_zero\_fp . 154, 201, 207, 259, 937, 950

## G

galley commands:

\l\_galley\_text\_width\_dim ..... 1909  
 \l\_galley\_total\_left\_margin\_dim 1840

graphics commands:

\graphics\_bb\_restore:nTF . 1360, 1586  
 \graphics\_bb\_save:n ..... 1395, 1594  
 \l\_graphics\_decodearray\_tl .....  
 ..... 1337, 1338,  
 1348, 1374, 1378, 1379, 1462, 1494,  
 1495, 1533, 1536, 1537, 1555, 1625  
 \graphics\_extract\_bb:n .....  
 ..... 1457, 1464, 1620, 1627  
 \l\_graphics\_interpolate\_bool ...  
 ..... 1339, 1349, 1373, 1380,  
 1463, 1496, 1532, 1538, 1556, 1626  
 \l\_graphics\_llx\_dim .....  
 ..... 1322, 1473, 1525, 1638  
 \l\_graphics\_lly\_dim .....  
 ..... 1323, 1474, 1526, 1639  
 \l\_graphics\_name\_tl ..... 1429  
 \l\_graphics\_page\_int .....  
 ..... 1333, 1353, 1354, 1384,  
 1385, 1455, 1492, 1493, 1519, 1520,  
 1548, 1561, 1562, 1601, 1602, 1618  
 \l\_graphics\_pagebox\_tl .....  
 ..... 41, 1334, 1352,

1386, 1387, 1456, 1490, 1491, 1521,  
 1523, 1549, 1570, 1571, 1603, 1619  
 \graphics\_read\_bb:n . 1316, 1451, 1615  
 \l\_graphics\_urx\_dim .....  
 .. 1324, 1391, 1475, 1527, 1592, 1640  
 \l\_graphics\_ury\_dim .. 1325, 1392,  
 1476, 1528, 1593, 1641, 1649, 1650

graphics internal commands:

\l\_\_graphics\_backend\_dir\_str . 1404  
 \l\_\_graphics\_backend\_ext\_str . 1404  
 \\_\_graphics\_backend\_getbb\_auxi:n  
 ..... 1331  
 \\_\_graphics\_backend\_getbb\_-  
 auxi:nN ..... 1546  
 \\_\_graphics\_backend\_getbb\_-  
 auxii:n ..... 1331  
 \\_\_graphics\_backend\_getbb\_-  
 auxii:nnN ..... 1546  
 \\_\_graphics\_backend\_getbb\_-  
 auxiii:nNnn ..... 1546  
 \\_\_graphics\_backend\_getbb\_-  
 auxiv:nnNnn ..... 1546  
 \\_\_graphics\_backend\_getbb\_-  
 auxv:nNnn ..... 1546  
 \\_\_graphics\_backend\_getbb\_-  
 auxvi:nNnn ..... 1587, 1589  
 \\_\_graphics\_backend\_getbb\_eps:n .  
 ..... 1310, 1404, 1445, 1609  
 \\_\_graphics\_backend\_getbb\_eps:nm  
 ..... 1404  
 \\_\_graphics\_backend\_getbb\_eps:nn  
 ..... 1415, 1422  
 \\_\_graphics\_backend\_getbb\_jpg:n .  
 ..... 1331, 1445, 1546, 1616  
 \\_\_graphics\_backend\_getbb\_-  
 pagebox:w ..... 1546, 1603  
 \\_\_graphics\_backend\_getbb\_pdf:n .  
 ..... 1331, 1430, 1445, 1546, 1623  
 \\_\_graphics\_backend\_getbb\_png:n .  
 ..... 1331, 1445, 1546, 1616  
 \\_\_graphics\_backend\_include:nn 1629  
 \\_\_graphics\_backend\_include\_-  
 auxi:nn ..... 1468  
 \\_\_graphics\_backend\_include\_-  
 auxii:nnn ..... 1468  
 \\_\_graphics\_backend\_include\_-  
 auxiii:nnn ..... 1468  
 \\_\_graphics\_backend\_include\_-  
 bitmap\_quote:w ..... 1597, 1644  
 \\_\_graphics\_backend\_include\_-  
 eps:n ..... 1317, 1404, 1468, 1629  
 \\_\_graphics\_backend\_include\_-  
 jpg:n ..... 1397, 1468, 1644

<code>\__graphics_backend_include_-pdf:n</code> .. <a href="#">1397</a> , <a href="#">1436</a> , <a href="#">1468</a> , <a href="#">1597</a> , <a href="#">1629</a>	<code>\int_set_eq:NN</code> ..... <a href="#">1034</a> , <a href="#">1938</a>
<code>\__graphics_backend_include_pdf_-quote:w</code> ..... <a href="#">1600</a> , <a href="#">1605</a>	<code>\int_use:N</code> ..... <a href="#">284</a> , <a href="#">315</a> , <a href="#">1112</a> , <a href="#">1118</a> , <a href="#">1125</a> , <a href="#">1157</a> , <a href="#">1165</a> , <a href="#">1354</a> , <a href="#">1385</a> , <a href="#">1400</a> , <a href="#">1493</a> , <a href="#">1506</a> , <a href="#">1518</a> , <a href="#">1520</a> , <a href="#">1602</a> , <a href="#">1681</a> , <a href="#">1732</a> , <a href="#">1745</a> , <a href="#">1749</a> , <a href="#">1761</a> , <a href="#">1784</a> , <a href="#">1790</a> , <a href="#">1863</a> , <a href="#">1971</a> , <a href="#">2161</a> , <a href="#">2168</a> , <a href="#">2325</a> , <a href="#">2363</a> , <a href="#">2368</a> , <a href="#">2404</a> , <a href="#">2412</a>
<code>\__graphics_backend_include_-png:n</code> ..... <a href="#">1397</a> , <a href="#">1468</a> , <a href="#">1644</a>	<code>\int_value:w</code> .. <a href="#">2052</a> , <a href="#">2078</a> , <a href="#">2226</a> , <a href="#">2238</a>
<code>\l__graphics_backend_name_str</code> . <a href="#">1404</a>	<code>\int_zero:N</code> ... <a href="#">1333</a> , <a href="#">1455</a> , <a href="#">1548</a> , <a href="#">1618</a>
<code>\l__graphics_graphics_attr_tl</code> ... <a href="#">1330</a> , <a href="#">1335</a> , <a href="#">1342</a> , <a href="#">1350</a> , <a href="#">1360</a> , <a href="#">1393</a> , <a href="#">1395</a> , <a href="#">1400</a>	<b>K</b>
<code>\l__graphics_internal_box</code> ..... <a href="#">1389</a> , <a href="#">1391</a> , <a href="#">1392</a> , <a href="#">1591</a> , <a href="#">1592</a> , <a href="#">1593</a>	kernel internal commands:
<code>\g__graphics_track_int</code> ..... <a href="#">1467</a> , <a href="#">1513</a> , <a href="#">1514</a>	<code>\__kernel_backend_align_begin:</code> .. <a href="#">43</a> , <a href="#">127</a> , <a href="#">151</a> , <a href="#">166</a>
group commands:	<code>\__kernel_backend_align_end:</code> ... <a href="#">43</a> , <a href="#">141</a> , <a href="#">159</a> , <a href="#">173</a>
<code>\group_begin:</code> ..... <a href="#">1037</a> , <a href="#">2006</a> , <a href="#">2112</a> , <a href="#">2140</a> , <a href="#">2162</a> , <a href="#">2195</a> , <a href="#">2456</a>	<code>\g__kernel_backend_header_bool</code> .. <a href="#">40</a>
<code>\group_end:</code> ..... <a href="#">1045</a> , <a href="#">2026</a> , <a href="#">2123</a> , <a href="#">2159</a> , <a href="#">2191</a> , <a href="#">2222</a> , <a href="#">2483</a>	<code>\__kernel_backend_literal:n</code> .... <a href="#">25</a> , <a href="#">31</a> , <a href="#">34</a> , <a href="#">41</a> , <a href="#">45</a> , <a href="#">52</a> , <a href="#">55</a> , <a href="#">57</a> , <a href="#">99</a> , <a href="#">102</a> , <a href="#">104</a> , <a href="#">106</a> , <a href="#">110</a> , <a href="#">256</a> , <a href="#">269</a> , <a href="#">416</a> , <a href="#">424</a> , <a href="#">524</a> , <a href="#">531</a> , <a href="#">729</a> , <a href="#">934</a> , <a href="#">941</a> , <a href="#">947</a> , <a href="#">1007</a> , <a href="#">1017</a> , <a href="#">1319</a> , <a href="#">1470</a> , <a href="#">1505</a> , <a href="#">1515</a> , <a href="#">1635</a> , <a href="#">1646</a> , <a href="#">2308</a> , <a href="#">2432</a> , <a href="#">2486</a> , <a href="#">2490</a> , <a href="#">2495</a> , <a href="#">2500</a>
<code>\group_insert_after:N</code> ..... <a href="#">420</a> , <a href="#">505</a> , <a href="#">730</a> , <a href="#">903</a>	<code>\__kernel_backend_literal_page:n</code> ..... <a href="#">68</a> , <a href="#">101</a> , <a href="#">2302</a> , <a href="#">2304</a> , <a href="#">2505</a> , <a href="#">2507</a>
<b>H</b>	<code>\__kernel_backend_literal_pdf:n</code> . <a href="#">60</a> , <a href="#">98</a> , <a href="#">182</a> , <a href="#">239</a> , <a href="#">767</a> , <a href="#">914</a>
hbox commands:	<code>\__kernel_backend_literal_-postscript:n</code> ... <a href="#">30</a> , <a href="#">46</a> , <a href="#">47</a> , <a href="#">51</a> , <a href="#">128</a> , <a href="#">129</a> , <a href="#">131</a> , <a href="#">132</a> , <a href="#">140</a> , <a href="#">152</a> , <a href="#">167</a> , <a href="#">520</a>
<code>\hbox:n</code> ..... <a href="#">1770</a> , <a href="#">1771</a> , <a href="#">1774</a> , <a href="#">1849</a> , <a href="#">1855</a> , <a href="#">2010</a> , <a href="#">2014</a> , <a href="#">2460</a> , <a href="#">2469</a>	<code>\__kernel_backend_literal_svg:n</code> . <a href="#">109</a> , <a href="#">113</a> , <a href="#">115</a> , <a href="#">117</a> , <a href="#">283</a> , <a href="#">285</a> , <a href="#">302</a> , <a href="#">1023</a> , <a href="#">1291</a> , <a href="#">1302</a>
<code>\hbox_overlap_right:n</code> ..... <a href="#">142</a> , <a href="#">174</a> , <a href="#">190</a> , <a href="#">231</a> , <a href="#">247</a> , <a href="#">275</a> , <a href="#">362</a> , <a href="#">758</a> , <a href="#">1015</a>	<code>\__kernel_backend_matrix:n</code> ..... <a href="#">88</a> , <a href="#">204</a> , <a href="#">225</a> , <a href="#">920</a>
<code>\hbox_set:Nn</code> ..... <a href="#">1389</a> , <a href="#">1591</a> , <a href="#">1836</a> , <a href="#">1873</a> , <a href="#">2007</a> , <a href="#">2113</a> , <a href="#">2457</a>	<code>\__kernel_backend_postscript:n</code> .. <a href="#">33</a> , <a href="#">418</a> , <a href="#">723</a> , <a href="#">1664</a> , <a href="#">1720</a> , <a href="#">1770</a> , <a href="#">1777</a> , <a href="#">1812</a> , <a href="#">1849</a> , <a href="#">1856</a> , <a href="#">1860</a> , <a href="#">1874</a> , <a href="#">1902</a> , <a href="#">1952</a> , <a href="#">1959</a> , <a href="#">1965</a> , <a href="#">1974</a> , <a href="#">1981</a> , <a href="#">2010</a> , <a href="#">2014</a>
<code>\hbox_set:Nw</code> ..... <a href="#">1819</a>	<code>\__kernel_backend_scope_begin:</code> <a href="#">5</a> , <a href="#">54</a> , <a href="#">76</a> , <a href="#">103</a> , <a href="#">112</a> , <a href="#">126</a> , <a href="#">150</a> , <a href="#">165</a> , <a href="#">181</a> , <a href="#">198</a> , <a href="#">224</a> , <a href="#">238</a> , <a href="#">255</a> , <a href="#">268</a> , <a href="#">773</a> , <a href="#">1002</a> , <a href="#">1289</a>
<code>\hbox_set_end:</code> ..... <a href="#">1834</a>	<code>\__kernel_backend_scope_begin:n</code> . <a href="#">116</a> , <a href="#">304</a> , <a href="#">312</a> , <a href="#">317</a> , <a href="#">335</a> , <a href="#">348</a>
<code>\hbox_unpack:N</code> ..... <a href="#">1966</a>	<code>\__kernel_backend_scope_end:</code> ... <a href="#">54</a> , <a href="#">76</a> , <a href="#">103</a> , <a href="#">112</a> , <a href="#">143</a> , <a href="#">161</a> , <a href="#">175</a> , <a href="#">191</a> , <a href="#">218</a> , <a href="#">232</a> , <a href="#">248</a> , <a href="#">264</a> , <a href="#">276</a> , <a href="#">327</a> , <a href="#">328</a> , <a href="#">329</a> , <a href="#">344</a> , <a href="#">363</a> , <a href="#">774</a> , <a href="#">1019</a> , <a href="#">1303</a>
<b>I</b>	<code>\l__kernel_color_stack_int</code> ..... <a href="#">464</a> , <a href="#">504</a> , <a href="#">513</a> , <a href="#">902</a> , <a href="#">910</a>
int commands:	<b>L</b>
<code>\int_compare:nNnTF</code> <a href="#">1353</a> , <a href="#">1384</a> , <a href="#">1492</a> , <a href="#">1519</a> , <a href="#">1561</a> , <a href="#">1601</a> , <a href="#">1937</a> , <a href="#">2258</a> , <a href="#">2284</a>	<code>\landscape</code> ..... <a href="#">2373</a> , <a href="#">2375</a>
<code>\int_const:Nn</code> ..... <a href="#">1393</a> , <a href="#">1514</a> , <a href="#">1675</a> , <a href="#">2146</a> , <a href="#">2319</a>	
<code>\int_eval:n</code> ..... <a href="#">2238</a> , <a href="#">2261</a> , <a href="#">2268</a> , <a href="#">2278</a> , <a href="#">2486</a> , <a href="#">2494</a> , <a href="#">2499</a>	
<code>\int_gincr:N</code> ..... <a href="#">282</a> , <a href="#">1053</a> , <a href="#">1109</a> , <a href="#">1154</a> , <a href="#">1513</a> , <a href="#">1674</a> , <a href="#">1729</a> , <a href="#">1756</a> , <a href="#">1780</a> , <a href="#">1858</a> , <a href="#">2318</a> , <a href="#">2361</a> , <a href="#">2400</a>	
<code>\int_gset:Nn</code> ..... <a href="#">1926</a>	
<code>\int_gset_eq:NN</code> ..... <a href="#">1046</a> , <a href="#">1757</a> , <a href="#">1781</a> , <a href="#">1859</a> , <a href="#">2401</a>	
<code>\int_gzero:N</code> ..... <a href="#">1038</a>	
<code>\int_if_exist:NnTF</code> ..... <a href="#">1503</a>	
<code>\int_if_odd:nTF</code> ..... <a href="#">1843</a>	
<code>\int_new:N</code> ..... <a href="#">332</a> , <a href="#">464</a> , <a href="#">1056</a> , <a href="#">1057</a> , <a href="#">1135</a> , <a href="#">1467</a> , <a href="#">1670</a> , <a href="#">1752</a> , <a href="#">1791</a> , <a href="#">1793</a> , <a href="#">2314</a> , <a href="#">2382</a>	

**M**

math commands:

`\c_math_toggle_token` . . . . 1822, 1832

mode commands:

`\mode_if_horizontal:TF` . . . 1928, 1935

`\mode_if_math:TF` . . . . . 1816

**O**

`\oddsidemargin` . . . . . 1844

**P**

pdf internal commands:

`\__pdf_backend:n` . . . . . 2307,  
2311, 2313, 2339, 2344, 2353, 2402,  
2419, 2429, 2435, 2462, 2463, 2471

`\__pdf_backend_annotation:nnnn` . .  
. . . . . 1753, 2040, 2383

`\__pdf_backend_annotation_-  
aux:nnnn` . . . . . 1753, 2383

`\g__pdf_backend_annotation_int` . .  
. . . . . 1752,  
1757, 1781, 1790, 2382, 2401, 2412

`\__pdf_backend_annotation_last:` .  
. . . . . 1789, 2050, 2411

`\__pdf_backend_bdc:nn` . . . . .  
. . . . . 2034, 2301, 2504, 2525

`\__pdf_backend_catalog_gput:nn` . .  
. . . . . 1666, 2125, 2310, 2510

`\__pdf_backend_compress_objects:n`  
. . . . . 2028, 2232, 2485, 2519

`\__pdf_backend_compresslevel:n` . .  
. . . . . 2028, 2232, 2485, 2519

`\l__pdf_backend_content_box` 1750,  
1819, 1848, 1851, 1853, 1882, 1893

`\__pdf_backend_destination:nn` . . .  
. . . . . 1979, 2091, 2433

`\__pdf_backend_destination_-  
rectangle:nn` . . . . 1979, 2091, 2433

`\__pdf_backend_emc:` . . . . .  
. . . . . 2034, 2301, 2504, 2525

`\__pdf_backend_info_gput:nn` . . . .  
. . . . . 1666, 2125, 2310, 2510

`\__pdf_backend_link:nw` . . . . . 1800

`\__pdf_backend_link_aux:nw` . . . 1800

`\__pdf_backend_link_begin:n` . . 2413

`\__pdf_backend_link_begin:nnnw` 2058

`\__pdf_backend_link_begin:nw` . . .  
. . . . . 1801, 1803, 1804

`\__pdf_backend_link_begin_aux:nw`  
. . . . . 1807, 1809

`\__pdf_backend_link_begin_-  
goto:nnw` . . . . . 1800, 2058, 2413

`\__pdf_backend_link_begin_-  
user:nnw` . . . . . 1800, 2058, 2413

`\g__pdf_backend_link_bool` . . . . .  
. . . . . 1795, 1806, 1811, 1826, 1869

`\g__pdf_backend_link_dict_tl` . . .  
. . . . . 1792, 1814, 1864

`\__pdf_backend_link_end:` . . . . .  
. . . . . 1800, 2058, 2413

`\__pdf_backend_link_end_aux:` . 1800

`\g__pdf_backend_link_int` . . . . .  
. . . . . 1791, 1859, 1863, 1971

`\__pdf_backend_link_last:` . . . . .  
. . . . . 1970, 2076, 2430

`\__pdf_backend_link_margin:n` . . .  
. . . . . 1972, 2084, 2431

`\g__pdf_backend_link_math_bool` . .  
. . . . . 1794, 1817, 1818, 1821, 1831

`\__pdf_backend_link_minima:` . . 1800

`\__pdf_backend_link_outerbox:n` 1800

`\g__pdf_backend_link_sf_int` . . . .  
. . . . . 1793, 1926, 1937, 1938

`\__pdf_backend_link_sf_restore:` 1800

`\__pdf_backend_link_sf_save:` . 1800

`\l__pdf_backend_model_box` . 1751,  
1836, 1873, 1881, 1892, 1907, 1914

`\__pdf_backend_objcompresslevel:n`  
. . . . . 2232

`\g__pdf_backend_object_int` . . . . .  
1670, 1674, 1677, 1729, 1732, 1745,  
1749, 1756, 1757, 1761, 1780, 1781,  
1784, 1858, 1859, 2314, 2318, 2321,  
2361, 2363, 2368, 2400, 2401, 2404

`\__pdf_backend_object_last:` . . . .  
. . . . . 1748, 2224, 2367, 2512

`\__pdf_backend_object_new:nn` . . .  
. . . . . 1672, 2140, 2316, 2512

`\__pdf_backend_object_now:nn` . . .  
. . . . . 1727, 2195, 2359, 2512

`\g__pdf_backend_object_prop` . . . .  
. . . . . 1670, 1678, 1689, 1699,  
2139, 2149, 2171, 2314, 2322, 2329

`\__pdf_backend_object_ref:n` 1672,  
1686, 1700, 2140, 2316, 2335, 2512

`\__pdf_backend_object_write:nn` . .  
. . . . . 1682, 2162, 2326, 2512

`\__pdf_backend_object_write:nnn` 2326

`\__pdf_backend_object_write_-  
array:nn` . . . . . 1682, 2326

`\__pdf_backend_object_write_-  
dict:nn` . . . . . 1682, 2326

`\__pdf_backend_object_write_-  
fstream:nn` . . . . . 2326

`\__pdf_backend_object_write_-  
stream:nn` . . . . . 1682, 2326

`\__pdf_backend_object_write_-  
stream:nnn` . . . . . 1682





## T

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

`\ccclv` ..... 1948, 1950, 1958  
`\makecol@hook` ..... 1941  
`\current@color` . 13, 378, 383, 388, 436  
`\special` ..... 1

tex commands:

`\tex_baselineskip:D` ..... 1918  
`\tex_global:D` .....  
 ..... 2234, 2248, 2260, 2267, 2274  
`\tex_immediate:D` .... 1371, 2167, 2200  
`\tex_kern:D` ..... 1776  
`\tex_luatexversion:D` .... 2258, 2284  
`\tex_pdfannot:D` ..... 2044  
`\tex_pdfcatalog:D` ..... 2129  
`\tex_pdfcolorstack:D` 503, 512, 901, 909  
`\tex_pdfcompresslevel:D` .. 2235, 2236  
`\tex_pdfdest:D` ..... 2095, 2116  
`\tex_pdfendlink:D` ..... 2074  
`\tex_pdfextension:D` 62, 63, 70, 71,  
 78, 79, 84, 85, 90, 91, 501, 502, 510,  
 511, 899, 900, 907, 908, 2042, 2043,  
 2064, 2065, 2072, 2073, 2093, 2094,  
 2114, 2115, 2127, 2128, 2134, 2135,  
 2152, 2155, 2188, 2189, 2219, 2220  
`\tex_pdffeedback:D` ..... 2053,  
 2054, 2079, 2080, 2156, 2227, 2228  
`\tex_pdfinfo:D` ..... 2136  
`\tex_pdflastannot:D` ..... 2055  
`\tex_pdflastlink:D` ..... 2081  
`\tex_pdflastobj:D` ..... 2158, 2229  
`\tex_pdflastximage:D` .... 1390, 1394  
`\tex_pdflinkmargin:D` ..... 2088  
`\tex_pdfliteral:D` ..... 64, 72  
`\tex_pdfmajorversion:D` .....  
 ..... 2265, 2267, 2289, 2290  
`\tex_pdfminorversion:D` .....  
 ..... 2275, 2276, 2297, 2298  
`\tex_pdfobj:D` ..... 2158, 2190, 2221  
`\tex_pdfobjcompresslevel:D` 2249, 2250  
`\tex_pdfrefximage:D` ..... 1390, 1399  
`\tex_pdfrestore:D` ..... 86  
`\tex_pdfsave:D` ..... 80  
`\tex_pdfsetmatrix:D` ..... 92  
`\tex_pdfstartlink:D` ..... 2066  
`\tex_pdfvariable:D` .....  
 ..... 2086, 2087, 2237, 2251,  
 2256, 2260, 2277, 2282, 2285, 2299

`\tex_pdfximage:D` ..... 1371  
`\tex_pdfximagebbox:D` ..... 1365  
`\tex_spacefactor:D` ..... 1929, 1938  
`\tex_special:D` ..... 25  
`\tex_the:D` .... 1394, 2285, 2290, 2296  
`\tex_XeTeXpdffile:D` ..... 1557, 1599  
`\tex_XeTeXpicfile:D` ..... 1550  
`\textwidth` ..... 1912

tl commands:

`\c_space_tl` .....  
 .. 206, 211, 214, 383, 1094, 1321,  
 1322, 1323, 1324, 1472, 1473, 1474,  
 1475, 1520, 1523, 1525, 1526, 1527,  
 1528, 1600, 1602, 1637, 1638, 1639,  
 1640, 1864, 2056, 2082, 2230, 2404  
`\tl_clear:N` ..... 1334, 1342, 1348,  
 1456, 1462, 1549, 1555, 1619, 1625  
`\tl_gclear:N` ..... 1132, 1168  
`\tl_gset:Nn` ..... 1091, 1814  
`\tl_if_empty:NTF` . 1094, 1337, 1378,  
 1386, 1490, 1494, 1521, 1536, 1570  
`\tl_if_empty:nTF` ..... 1188  
`\tl_if_empty_p:N` ..... 1374, 1533  
`\tl_if_head_is_space:nTF` ..... 378  
`\tl_new:N` ..... 1098, 1330, 1792, 1796  
`\tl_put_left:Nn` ..... 2377  
`\tl_put_right:Nn` ..... 1946, 2375  
`\tl_set:Nn` . 380, 392, 442, 445, 448,  
 452, 455, 1335, 1350, 1429, 1797, 1964  
`\tl_to_str:n` ..... 1676,  
 1681, 2147, 2161, 2169, 2320, 2325

## U

use commands:

`\use:N` ..... 1698, 1744, 2334, 2362  
`\use:n` ..... 38, 383, 467, 487,  
 662, 837, 959, 971, 983, 1173, 1213,  
 1232, 1244, 1311, 1446, 1577, 1610  
`\use_none:n` .... 452, 1188, 1190, 1942

## V

`\value` ..... 1843  
 vbox commands:  
`\vbox:n` ..... 2389  
`\vbox_set:Nn` ..... 1950  
`\vbox_unpack_drop:N` ..... 1958