

The `l3backend-testphase` package  
Additional backend PDF features  
L<sup>A</sup>T<sub>E</sub>X PDF management testphase bundle

The L<sup>A</sup>T<sub>E</sub>X Project\*

Version 0.96r, released 2025-05-15

## 1 `l3backend-testphase` Implementation

```
1 <drivers>\ProvidesExplFile
2 <*dvipdfmx>
3   {l3backend-testphase-dvipdfmx.def}{2025-05-15}{ }
4   {LaTeX-PDF-management-testphase-bundle-backend-support: dvipdfmx}
5 </dvipdfmx>
6 <*dvips>
7   {l3backend-testphase-dvips.def}{2025-05-15}{ }
8   {LaTeX-PDF-management-testphase-bundle-backend-support: dvips}
9 </dvips>
10 <*dvisvgm>
11   {l3backend-testphase-dvisvgm.def}{2025-05-15}{ }
12   {LaTeX-PDF-management-testphase-bundle-backend-support: dvisvgm}
13 </dvisvgm>
14 <*luatex>
15   {l3backend-testphase-luatex.def}{2025-05-15}{ }
16   {LaTeX-PDF-management-testphase-bundle-backend-support: PDF output (LuaTeX)}
17 </luatex>
18 <*pdftex>
19   {l3backend-testphase-pdftex.def}{2025-05-15}{ }
20   {LaTeX-PDF-management-testphase-bundle-backend-support: PDF output (pdfTeX)}
21 </pdftex>
22 <*xdvipdfmx>
23   {l3backend-testphase-xetex.def}{2025-05-15}{ }
24   {LaTeX-PDF-management-testphase-bundle-backend-support: XeTeX}
25 </xdvipdfmx>
```

### 1.1 Variants

We need to generate temporarily a few e-types variants of kernel backend commands. These can be removed once the kernel provides them.

```
26 <@@=pdf>
27 <*luatex | pdftex>
28 \cs_generate_variant:Nn \_kernel_backend_literal_page:n { e }
```

---

\*E-mail: [latex-team@latex-project.org](mailto:latex-team@latex-project.org)

```

29 </luatex | pdftex>
30 <*dvipdfmx | xdvipdfmx>
31 \cs_generate_variant:Nn \_kernel_backend_literal:n { e }
32 \cs_generate_variant:Nn \_pdf_backend:n { e }
33 </dvipdfmx | xdvipdfmx>
34 <*dvips>
35 \cs_generate_variant:Nn \_kernel_backend_postscript:n { e }
36 \cs_generate_variant:Nn \_pdf_backend_pdfmark:n { e }
37 </dvips>

```

## 1.2 Support for delayed literal and special

Starting with TeXlive 2023 the engines support a `shipout` keyword for `\pdfliteral` and `\special`. When used the argument is not expanded when the command is used but only when the page is shipped out. This allows for example the tagging code to delay the page-wise numbering of MC-chunks until the page is actually built. For now we test the engine support. The boolean is setup in `pdfmanagement-testphase.dtx`.

```
38 <*drivers>
```

The following commands provide the needed kernel backend support. This are basically copies of similar commands of `l3backend-basics`.

`\_kernel_backend_shipout_literal:e` The one shared function for all backends is access to the basic `\special` primitive.

```

39 \bool_if:NT \l_pdfmanagement_delayed_shipout_bool
40 {
41   \cs_new_protected:Npn \_kernel_backend_shipout_literal:e #1
42     { \tex_special:D~shipout { #1} }
43 </drivers>

```

*(End of definition for \\_kernel\_backend\_shipout\_literal:e.)*

```
44 <*luatex | pdftex>
```

`\_kernel_backend_shipout_literal_pdf:e` 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).

```

45 \cs_new_protected:Npn \_kernel_backend_shipout_literal_pdf:e #1
46   {
47 <*luatex>
48   \tex_pdfextension:D ~ literal ~ shipout ~
49 </luatex>
50 <*pdftex>
51   \tex_pdfliteral:D ~ shipout ~
52 </pdftex>
53   { #1 }
54   }

```

*(End of definition for \\_kernel\_backend\_shipout\_literal\_pdf:e.)*

`\_kernel_backend_shipout_literal_page:e` Page literals are pretty simple.

```

55 \cs_new_protected:Npn \_kernel_backend_shipout_literal_page:e #1
56   {
57 <*luatex>
58   \tex_pdfextension:D ~ literal ~ shipout ~
59 </luatex>

```

```

60 <*pdftex>
61     \tex_pdfliteral:D ~ shipout ~
62 </pdftex>
63     page { #1 }
64 }
65 </luatex | pdftex>
66 <drivers> }

```

(End of definition for \\_kernel\_backend\_shipout\_literal\_page:e.)

### 1.3 Crossreferences

Commands to get a reference for the absolute page counter.

```

67 <*drivers>
68 \cs_new_protected:Npn \_pdf_backend_record_abspage:n #1
69 {
70     \@bsphack
71     \property_record:nn{#1}{abspage}
72     \@esphack
73 }
74 \cs_new:Npn \_pdf_backend_ref_abspage:n #1
75 {
76     \property_ref:nn{#1}{abspage}
77 }
78
79 \cs_generate_variant:Nn \_pdf_backend_record_abspage:n {e}
80 \cs_generate_variant:Nn \_pdf_backend_ref_abspage:n {e}
81 </drivers>

```

avoid that destinations names are optimized with xelatex/dvipdfmx see <https://tug.org/pipermail/dvipdfmx/200002.html>

```

82 <*dvipdfmx | xdvipdfmx>
83     \_kernel_backend_literal:n { dvipdfmx:config~C~ 0x0010 }
84 </dvipdfmx | xdvipdfmx>

```

```

\_pdf_backend_resourceid_int
\_pdf_backend_name_int
\_pdf_backend_page_int

```

Some scratch variables

```

85 <*drivers>
86 \prop_new:N \_pdf_backend_resourceid_int
87 \tl_new:N \_pdf_backend_name_int
88 \box_new:N \_pdf_backend_page_int
89 \box_new:N \_pdf_backend_tpb_box
90 </drivers>

```

(End of definition for \\_pdf\_backend\_resourceid\_int, \\_pdf\_backend\_name\_int, and \\_pdf\_backend\_page\_int.)

```

\_pdf_backend_resourceid_int
\_pdf_backend_name_int
\_pdf_backend_page_int

```

a counter to create labels for the resources, a counter to number properties in bdc marks, a counter for the \pdfpageref implementation.

```

91 <*drivers>
92 \int_new:N \_pdf_backend_resourceid_int
93 \int_new:N \_pdf_backend_name_int
94 \int_new:N \_pdf_backend_page_int
95 </drivers>

```

(End of definition for \\_pdf\_backend\_resourceid\_int, \\_pdf\_backend\_name\_int, and \\_pdf\_backend\_page\_int.)

## 1.4 luacode

Load the lua code.

```
96 <*luatex>
97   \directlua { require("l3backend-testphase.lua") }
98 </luatex>
```

## 1.5 Converting unicode strings to a pdfname

dvips needs a special function here, so we add this as backend function.

```
99 <*pdftex | luatex | dvipdfmx | xdvipdfmx | dvisvgm>
100 \cs_new:Npn \__kernel_pdf_name_from_unicode_e:n #1
101   {
102     / \str_convert_pdfname:e { \text_expand:n { #1 } }
103   }
104 </pdftex | luatex | dvipdfmx | xdvipdfmx | dvisvgm>
105 <*dvips>
106 \cs_new:Npn \__kernel_pdf_name_from_unicode_e:n #1
107   {
108     ~ ( \text_expand:n { #1 } ) ~ cvn
109   }
110 </dvips>
```

## 1.6 Hooks

### 1.6.1 Add the “end run” hooks

Here we add the end run hook to suitable end hooks.

```
111 <*pdftex | luatex>
112 % put in \@kernel@after@enddocument@afterlastpage
113 \tl_gput_right:Nn \@kernel@after@enddocument@afterlastpage
114   {
115     \g__kernel_pdfmanagement_end_run_code_tl
116   }
117 </pdftex | luatex>
118 <*dvipdfmx | xdvipdfmx>
119 % put in \@kernel@after@shipout@lastpage
120 \tl_gput_right:Nn \@kernel@after@shipout@lastpage
121   {
122     \g__kernel_pdfmanagement_end_run_code_tl
123   }
124 </dvipdfmx | xdvipdfmx>
125 <*dvips>
126 % put in \@kernel@after@shipout@lastpage
127 \tl_gput_right:Nn \@kernel@after@shipout@lastpage
128   {
129     \g__kernel_pdfmanagement_end_run_code_tl
130   }
131 </dvips>
```

## 1.6.2 Add the “shipout” hooks

Now we add to the shipout hooks the relevant token lists. We also push the page resources in shipout/firstpage (AtBeginDvi) as the backend code sets color stack there. The xetex driver needs a rule here. If it clashes on the first page, we will need a test ...

```
132 <*drivers>
133 \tl_if_exist:NTF \@kernel@after@shipout@background
134 {
135   \g@addto@macro \@kernel@before@shipout@background{\relax}
136   \g@addto@macro \@kernel@after@shipout@background
137   {
138     \g__kernel_pdfmanagement_thispage_shipout_code_tl
139   }
140 }
141 {
142   \hook_gput_code:nnn{shipout/background}{pdf}
143   {
144     \g__kernel_pdfmanagement_thispage_shipout_code_tl
145   }
146 }
147
148 </drivers>
```

## 1.7 The /Pages dictionary (pdfpagesattr)

\\_pdf\_backend\_Pages\_primitive:n

This is the primitive command to add something to the /Pages dictionary. It works differently for the backends: pdftex and luatex overwrite existing content, dvips and dviPDFM are additive. luatex sets it in lua. The higher level code has to take this into account.

```
149 <*pdftex>
150 \cs_new_protected:Npn \_pdf_backend_Pages_primitive:n #1
151 {
152   \tex_global:D \tex_pdfpagesattr:D { #1 }
153 }
154 </pdftex>
155 <*luatex>
156 %luatex: does it in lua
157 \sys_if_engine_luatex:T
158 {
159   \cs_new_protected:Npn \_pdf_backend_Pages_primitive:n #1
160   {
161     \tex_directlua:D
162     {
163       pdf.setpagesattributes( \_pdf_backend_luastring:n { #1 } )
164     }
165   }
166 }
167 </luatex>
168 <*dvips>
169 \cs_new_protected:Npx \_pdf_backend_Pages_primitive:n #1
170 {
171   \tex_special:D{ps:~[#1~/PAGES-pdfmark] %}
172 }
```

```

173 </dvips>
174 <*dvipdfmx | xdvipdfmx>
175 \cs_new_protected:Npn \__pdf_backend_Pages_primitive:n #1
176 {
177   \__pdf_backend:n{put~@pages~<<#1>>}
178 }
179 </dvipdfmx | xdvipdfmx>
180 <*dvisvgm>
181 \cs_new_protected:Npn \__pdf_backend_Pages_primitive:n #1
182 {}
183 </dvisvgm>

```

(End of definition for \\_\_pdf\_backend\_Pages\_primitive:n.)

## 1.8 “Page” and “ThisPage” attributes (pdfpageattr)

<pre> \__pdf_backend_Page_primitive:n \__pdf_backend_Page_gput:nn \__pdf_backend_Page_gremove:n \__pdf_backend_ThisPage_gput:nn \__pdf_backend_ThisPage_gpush:n </pre>	<p>\__pdf_backend_Page_primitive:n is the primitive command to add something to the /Page dictionary. It works differently for the backends: pdftex and luatex overwrite existing content, dvips and dvipdfmx are additive. luatex sets it in lua. The higher level code has to take this into account. \__pdf_backend_Page_gput:nn stores default values. \__pdf_backend_Page_gremove:n allows to remove a value. \__pdf_backend_ThisPage_gput:nn adds a value to the current page. \__pdf_backend_ThisPage_gpush:n merges the default and the current page values and add them to the dictionary of the current page in \g__pdf_backend_thispage_shipout_tl.</p>
--	--

```

184 % backend commands
185 <*pdftex>
186 %the primitive
187 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
188 {
189   \tex_global:D \tex_pdfpageattr:D { #1 }
190 }
191 % the command to store default values.
192 % Uses a prop with pdflatex + dvi,
193 % sets a lua table with luatex
194 \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2 %key,value
195 {
196   \pdfdict_gput:nnn {g__pdf_Core/Page}{ #1 }{ #2 }
197 }
198 % the command to remove a default value.
199 % Uses a prop with pdflatex + dvi,
200 % changes a lua table with luatex
201 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
202 {
203   \pdfdict_gremove:nn {g__pdf_Core/Page}{ #1 }
204 }
205 % the command used in the document.
206 % direct call of the primitive special with dvips/dvipdfmx
207 % \latelua: fill a page related table with luatex, merge it with the page
208 % table and push it directly
209 % write to aux and store in prop with pdflatex
210 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
211 {
212   %we need to know the page the resource should be added too.

```

```

213 \int_gincr:N\g__pdf_backend_resourceid_int
214 \__pdf_backend_record_abbrev:e { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
215 \tl_set:Ne \l__pdf_tmpa_tl
216 {
217   \__pdf_backend_ref_abbrev:e {l3pdf\int_use:N\g__pdf_backend_resourceid_int}
218 }
219 \pdfdict_if_exist:nF { g__pdf_Core/backend_Page\l__pdf_tmpa_tl}
220 {
221   \pdfdict_new:n { g__pdf_Core/backend_Page\l__pdf_tmpa_tl}
222 }
223 %backend_Page has no handler.
224 \pdfdict_gput:nnn {g__pdf_Core/backend_Page\l__pdf_tmpa_tl}{ #1 }{ #2 }
225 }
226 %the code to push the values, used in shipout
227 %merges the two props and then fills the register in pdflatex
228 %merges the two tables and then fills (in lua) in luatex
229 %issues the values stored in the global prop with dvi
230 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
231 {
232   \prop_gset_eq:Nc \g__pdf_tmpa_prop { \__kernel_pdfdict_name:n { g__pdf_Core/Page } }
233   \prop_if_exist:cT { \__kernel_pdfdict_name:n { g__pdf_Core/backend_Page#1 } }
234   {
235     \prop_map_inline:cn { \__kernel_pdfdict_name:n { g__pdf_Core/backend_Page#1 } }
236     {
237       \prop_gput:Nnn \g__pdf_tmpa_prop { ##1 }{ ##2 }
238     }
239   }
240   \__pdf_backend_Page_primitive:e
241   {
242     \prop_map_function:NN \g__pdf_tmpa_prop \pdfdict_item:ne
243   }
244 }
245 </pdfTeX>
246 <*luatex>
247 % do we need to use some escaping for the values?????
248 \cs_new:Npn \__pdf_backend_luastring:n #1
249 {
250   "\tex_luaescapestring:D { \tex_unexpanded:D { #1 } }"
251 }
252 %not used, only there for consistency
253 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
254 {
255   \tex_latelua:D
256   {
257     pdf.setpageattributes(\__pdf_backend_luastring:n { #1 })
258   }
259 }
260 % the command to store default values.
261 % Uses a prop with pdflatex + dvi,
262 % sets a lua table with luatex
263 \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2
264 {
265   \tex_directlua:D
266   {

```

```

267         ltx.__pdf.backend_Page_gput
268         (
269             \__pdf_backend_luastring:n { #1 },
270             \__pdf_backend_luastring:n { #2 }
271         )
272     }
273 }
274 % the command to remove a default value.
275 % Uses a prop with pdflatex + dvi,
276 % changes a lua table with luatex
277 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
278 {
279     \tex_directlua:D
280     {
281         ltx.__pdf.backend_Page_gremove (\__pdf_backend_luastring:n { #1 })
282     }
283 }
284 % the command used in the document.
285 % direct call of the primitive special with dvips/dvipdfmx
286 % \latelua: fill a page related table with luatex, merge it with the page
287 % table and push it directly
288 % write to aux and store in prop with pdflatex
289 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
290 {
291     \tex_latelua:D
292     {
293         ltx.__pdf.backend_ThisPage_gput
294         (
295             tex.count["g_shipout_readonly_int"],
296             \__pdf_backend_luastring:n { #1 },
297             \__pdf_backend_luastring:n { #2 }
298         )
299         ltx.__pdf.backend_ThisPage_gpush (tex.count["g_shipout_readonly_int"])
300     }
301 }
302 %the code to push the values, used in shipout
303 %merges the two props and then fills the register in pdflatex
304 %merges the two tables (the one is probably still empty) and then fills (in lua) in luatex
305 %issues the values stored in the global prop with dvi
306 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
307 {
308     \tex_latelua:D
309     {
310         ltx.__pdf.backend_ThisPage_gpush (tex.count["g_shipout_readonly_int"])
311     }
312 }
313
314 </luatex>
315 < *dvipdfmx | xdvipdfmx >
316 %the primitive
317 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
318 {
319     \tex_special:D{pdf:-put~@thispage-<<#1>>}
320 }

```



```

321 % the command to store default values.
322 % Uses a prop with pdflatex + dvi,
323 % sets a lua table with luatex
324 \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2
325 {
326   \pdfdict_gput:nnn {g__pdf_Core/Page}{ #1 }{ #2 }
327 }
328 % the command to remove a default value.
329 % Uses a prop with pdflatex + dvi,
330 % changes a lua table with luatex
331 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
332 {
333   \pdfdict_gremove:nn {g__pdf_Core/Page}{ #1 }
334 }
335 % the command used in the document.
336 % direct call of the primitive special with dvips/dvipdfmx
337 % \lattelua: fill a page related table with luatex, merge it with the page
338 % table and push it directly
339 % write to aux and store in prop with pdflatex
340 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
341 {
342   \__pdf_backend_Page_primitive:n { /#1~#2 }
343 }
344 %the code to push the values, used in shipout
345 %merges the two props and then fills the register in pdflatex
346 %merges the two tables (the one is probably still empty)
347 % and then fills (in lua) in luatex
348 %issues the values stored in the global prop with dvi
349 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
350 {
351   \__pdf_backend_Page_primitive:e
352   { \pdfdict_use:n { g__pdf_Core/Page} }
353 }
354 </dvipdfmx | xdvipdfmx>
355 <*dvips>
356 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
357 {
358   \tex_special:D{ps:-[ThisPage]<<#1>>~/PUT~pdfmark} %]
359 }
360 % the command to store default values.
361 % Uses a prop with pdflatex + dvi,
362 % sets a lua table with luatex
363 \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2
364 {
365   \pdfdict_gput:nnn {g__pdf_Core/Page}{ #1 }{ #2 }
366 }
367 % the command to remove a default value.
368 % Uses a prop with pdflatex + dvi,
369 % changes a lua table with luatex
370 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
371 {
372   \pdfdict_gremove:nn {g__pdf_Core/Page}{ #1 }
373 }
374 % the command used in the document.

```

```

375 % direct call of the primitive special with dvips/dvipdfmx
376 % \lattelua: fill a page related table with luatex, merge it with the page
377 % table and push it directly
378 % write to aux and store in prop with pdflatex
379 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
380 {
381   \__pdf_backend_Page_primitive:n { /#1~#2 }
382 }
383 %the code to push the values, used in shipout
384 %merges the two props and then fills the register in pdflatex
385 %merges the two tables (the one is probably still empty)
386 %and then fills (in lua) in luatex
387 %issues the values stored in the global prop with dvi
388 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
389 {
390   \__pdf_backend_Page_primitive:e
391     { \pdfdict_use:n { g__pdf_Core/Page} }
392 }
393 </dvips>
394 <*dvisvgm>
395 % mostly only dummies ...
396 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
397 {}
398 % Uses a prop with pdflatex + dvi,
399 \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2
400 {
401   \pdfdict_gput:nnn {g__pdf_Core/Page}{ #1 }{ #2 }
402 }
403 % the command to remove a default value.
404 % Uses a prop with pdflatex + dvi,
405 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
406 {
407   \pdfdict_gremove:nn {g__pdf_Core/Page}{ #1 }
408 }
409 % the command used in the document.
410 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
411 {}
412 %the code to push the values, used in shipout
413 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
414 {}
415 </dvisvgm>
416 <*drivers>
417 \cs_generate_variant:Nn \__pdf_backend_Page_primitive:n { e }
418 </drivers>

```

*(End of definition for \\_\_pdf\_backend\_Page\_primitive:n and others.)*

## 1.9 “Page/Resources”: ExtGState, ColorSpace, Shading, Pattern

Path: Page/Resources/ExtGState etc. The actual output of the resources is handled together with the bdc/Properties. Here is only special code.

`\c__pdf_backend_PageResources_clist` The names are quite often needed a similar list is now in l3pdfmanagement. Perhaps it should be merged.

```

419 <*drivers>
420 \clist_const:Nn \c__pdf_backend_PageResources_clist
421 {
422   ExtGState,
423   ColorSpace,
424   Pattern,
425   Shading,
426 }
427 </drivers>

```

(End of definition for `\c__pdf_backend_PageResources_clist`.)

Now the backend commands the command to fill the register and to push the values.

`\_pdf_backend_PageResources_gput:nnn` stores values for the page resources.

**#1** : name of the resource (ExtGState, ColorSpace, Shading, Pattern)  
**#2** : a pdf name without slash  
**#3** : value

This pushes out the objects. It should be a no-op with xdvipdfmx and dvips as it currently issued in the end-of-run hook! create the backend objects:

`\_pdf_backend_PageResources_obj_gpush:`

```

428 <*pdfTeX | luatex>
429 \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
430 {
431   \pdf_object_new:n {\_pdf/Page/Resources/#1}
432   \cs_if_exist:NT \tex_directlua:D
433     {
434       \tex_directlua:D
435         {
436           ltx.__pdf.object["\_pdf/Page/Resources/#1"]
437           =
438           "\pdf_object_ref:n{\_pdf/Page/Resources/#1}"
439         }
440     }
441 }
442 </pdfTeX | luatex>

```

values are only stored in a prop and will be output at end document. luatex must also trigger the lua side

```

443 <*luatex>
444 \cs_new_protected:Npn \_pdf_backend_PageResources_gput:nnn #1 #2 #3
445 {
446   \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
447   \tex_directlua:D{ltx.__pdf.Page.Resources.#1=true}
448   \tex_directlua:D
449     {
450       ltx.pdf.Page_Resources_gpush(tex.count["g_shipout_readonly_int"])
451     }
452 }
453 </luatex>
454 <*pdfTeX>
455 \cs_new_protected:Npn \_pdf_backend_PageResources_gput:nnn #1 #2 #3
456 {

```

```

457     \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
458   }
459 </pdfTeX>

```

code for end of document code

```

460 <*pdfTeX | luatex>
461 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush:
462 {
463   \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
464     {
465     \prop_if_empty:cF
466     { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/##1} }
467     {
468     \pdf_object_write:mne
469     { __pdf/Page/Resources/##1 } { dict }
470     { \pdfdict_use:n { g__pdf_Core/Page/Resources/##1} }
471     }
472   }
473 }
474 </pdfTeX | luatex>

```

xdvipdfmx doesn't work correctly with object names ... <https://tug.org/pipermail/dvipdfmx/2019-August/000021.html>, so we use this must be issued on every page! objects should not only be created but also initialized initialization should be done before anyone tries to write so we add rules for the backend. The push command should not be used as it is in the wrong end document hook. If needed a new command must be added.

```

475 <*dvipdfmx | xdvipdfmx>
476 <xdvipdfmx> \hook_gset_rule:nnnn{shipout/firstpage}{l3backend-xetex}{after}{pdf}
477 <dvipdfmx> \hook_gset_rule:nnnn{shipout/firstpage}{l3backend-dvipdfmx}{after}{pdf}
478 %
479 \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
480 {
481   \pdf_object_new:n { __pdf/Page/Resources/#1 }
482   \hook_gput_code:nnn
483     {shipout/firstpage}
484     {pdf}
485     {\pdf_object_write:nnn { __pdf/Page/Resources/#1 } { dict } {}}
486 }
487 \cs_new_protected:Npn \__pdf_backend_PageResources:n #1
488 {
489   \__pdf_backend:n {put~@resources~<<#1>>}
490 }
491 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
492 {
493   % this is not used for output, but there is a test if the resource is empty
494   \prop_gput:cne { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/#1} }
495     { \str_convert_pdfname:n {#2} }{ #3 }
496   %objects are not filled with \pdf_object_write as this is not additive!
497   \__pdf_backend:e
498   {
499     put~\pdf_object_ref:n {__pdf/Page/Resources/#1}<</#2~#3>>
500   }
501 }
502
503 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush: {}

```

```
504 </dvipdfmx | xdvipdfmx>
```

dvips unneeded, or no-op. The push command should not be used as it is in the wrong end document hook. If needed a new command must be added.

```
505 <*dvips>
506 \cs_new_protected:Npn \__pdf_backend_PageResources:n #1 {}
507 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
508 { %only for the show command TEST!!
509   \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
510 }
511 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush: {}
512 </dvips>
```

dvipsvgm unneeded, or no-op

```
513 <*dvisvgm>
514 \cs_new_protected:Npn \__pdf_backend_PageResources:n #1 {}
515 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
516 { %only for the show command TEST!!
517   \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
518 }
519 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush: {}
520 </dvisvgm>
```

*(End of definition for \\_\_pdf\_backend\_PageResources\_gput:nnn and \\_\_pdf\_backend\_PageResources\_obj\_gpush:.)*

### 1.9.1 Page resources /Properties + BDC operators

```
\__pdf_backend_bdc:nn \__pdf_backend_bdc:nn, \__pdf_backend_shipout_bdc:ee, \__pdf_backend_bdcobject:nn,
  \__pdf_backend_shipout_bdc:ee \__pdf_backend_bdcobject:n, \__pdf_backend_bmc:n and \__pdf_backend_emc: are
\__pdf_backend_bdcobject:nn the backend command that create the bdc/emc marker and store the properties.
\__pdf_backend_bdcobject:n \__pdf_backend_PageResources_gpush:n outputs the /Properties and/or the other re-
  \__pdf_backend_bmc:n sources for the current page.
  \__pdf_backend_emc: pdftex and luatex (and perhaps dvips ...) need to know if there are in a xform stream
\__pdf_backend_PageResources_gpush:n ...
```

```
521 <*drivers>
522 \bool_new:N \l__pdf_backend_xform_bool
523 </drivers>
```

dvips is easy: create an object, and reference it in the bdc ghostscript will then automatically replace it by a name and add the name to the /Properties dict, special variant von accsupp <https://chat.stackexchange.com/transcript/message/50831812#50831812>

```
524 <*dvips>
525 %
526 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2 % #1 eg. Span, #2: dict_content
527 {
528   \__pdf_backend_pdfmark:n{/#1~<<#2>>~/BDC}
529 }
```

There is not difference here between inline and property BDC, it is always a property:

```
530 \cs_set_eq:NN \__pdf_backend_bdc_contobj:nn \__pdf_backend_bdc:nn
531 \cs_set_eq:NN \__pdf_backend_bdc_contstream:nn \__pdf_backend_bdc:nn
532
533 \bool_if:NT\l__pdfmanagement_delayed_shipout_bool
```

```

534 {
535   \cs_new_protected:Npn \__pdf_backend_bdc_shipout:ee #1 #2 % #1 eg. Span, #2: dict_content
536     {
537       \__kernel_backend_shipout_literal:e
538       {ps: SDict ~ begin ~ mark /#1~<<#2>>~/BDC ~ pdfmark ~ end }
539     }
540 }
541
542 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
543 {
544   \__pdf_backend_pdfmark:e{/#1~\pdf_object_ref:n{#2}~/BDC}
545 }
546 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1 % #1 eg. Span,
547 {
548   \__pdf_backend_pdfmark:e{/#1~\__pdf_backend_object_last:~/BDC}
549 }
550 \cs_set_protected:Npn \__pdf_backend_emc:
551 {
552   \__pdf_backend_pdfmark:n{/EMC} %
553 }
554 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
555 {
556   \__pdf_backend_pdfmark:n{/#1~/BMC} %
557 }
558 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1 {}
559
560 </dvips>
561 <*dvisvgm>
562 % dvisvgm should do nothing
563 %
564 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2 % #1 eg. Span, #2: dict_content
565 {}
566 \cs_set_eq:NN \__pdf_backend_bdc_contobj:nn \__pdf_backend_bdc:nn
567 \cs_set_eq:NN \__pdf_backend_bdc_contstream:nn \__pdf_backend_bdc:nn
568
569 \bool_if:NT\l__pdfmanagement_delayed_shipout_bool
570 {
571   \cs_set_protected:Npn \__pdf_backend_shipout_bdc:ee #1 #2 % #1 eg. Span, #2: dict_content
572     {}
573 }
574 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
575 {}
576 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1 % #1 eg. Span,
577 {}
578 \cs_set_protected:Npn \__pdf_backend_emc:
579 {}
580 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
581 {}
582 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1 {}
583
584 </dvisvgm>
585 %
586 % xetex has to create the entries in the /Properties manually
587 % (like the other backends)

```

```

588 % use pdfbase special
589 % https://chat.stackexchange.com/transcript/message/50832016#50832016
590 % the property is added to xform resources automatically,
591 % no need to worry about it.
592 (*dvipdfmx|xvipdfmx)
593 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
594 {
595   \int_gincr:N \g__pdf_backend_name_int
596   \__kernel_backend_literal:e
597   {
598     pdf:code~/#1/l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC
599   }
600   \__kernel_backend_literal:e
601   {
602     pdf:put~@resources~
603     <<
604       /Properties~
605       <<
606         /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl
607         \pdf_object_ref:n { #2 }
608       >>
609     >>
610   }
611 }
612 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1 % #1 eg. Span
613 {
614   \int_gincr:N \g__pdf_backend_name_int
615   \__kernel_backend_literal:e
616   {
617     pdf:code~/\exp_not:n{#1}/l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC
618   }
619   \__kernel_backend_literal:e
620   {
621     pdf:put~@resources~
622     <<
623       /Properties~
624       <<
625         /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl
626         \__pdf_backend_object_last:
627       >>
628     >>
629   }
630 }
631 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
632 {
633   \__kernel_backend_literal:n {pdf:code~/#1~BMC} %pdfbase
634 }
635
636 %this require management
637 \cs_set_protected:Npn \__pdf_backend_bdc_contobj:nn #1 #2
638 {
639   \pdf_object_unnamed_write:nn { dict }{ #2 }
640   \__pdf_backend_bdcobject:n { #1 }
641 }

```

```

642
643 \cs_set_protected:Npn \__pdf_backend_bdc_contstream:nn #1 #2
644 {
645   \__kernel_backend_literal:n {pdf:code~ /#1~<<#2>>~BDC }
646 }
647
648 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2
649 {
650   \bool_if:NTF \g__pdfmanagement_active_bool
651     {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contobj:nn}
652     {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contstream:nn}
653     \__pdf_backend_bdc:nn {#1}{#2}
654 }
655
656 \bool_if:NT\l__pdfmanagement_delayed_shipout_bool
657 {
658   \cs_set_protected:Npn \__pdf_backend_bdc_shipout_contstream:ee #1 #2
659   {
660     \__kernel_backend_shipout_literal:e {pdf:code~ /#1~<<#2>>~BDC }
661   }
662   \cs_set_eq:NN \__pdf_backend_bdc_shipout:ee \__pdf_backend_bdc_shipout_contstream:ee
663 }
664 \cs_set_protected:Npn \__pdf_backend_emc:
665 {
666   \__kernel_backend_literal:n {pdf:code~EMC} %pdfbase
667 }
668 % properties are handled automatically, but the other resources should be added
669 % at shipout
670 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1
671 {
672   \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
673   {
674     \prop_if_empty:cF { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/##1} }
675     {
676       \__kernel_backend_literal:e
677       {
678         pdf:put~@resources~
679         <</#1~\pdf_object_ref:n {__pdf/Page/Resources/##1}>>
680       }
681     }
682   }
683 }
684 </dviPDFmx | xdvipdfmx>
685 % luatex + pdftex
686 <*luatex>
687 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
688 {
689   \int_gincr:N \g__pdf_backend_name_int
690   \__kernel_backend_literal_page:e
691   { /#1 ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
692   \bool_if:NTF \l__pdf_backend_xform_bool
693   {
694     \pdfdict_gput:nee
695     { g__pdf_Core/Xform/Resources/Properties }

```



```

696         { l3pdf\int_use:N\g__pdf_backend_name_int }
697         { \pdf_object_ref:n { #2 } }
698     }
699     {
700     \exp_args:Ne \tex_latelua:D
701     {
702         ltx.pdf.Page_Resources_Properties_gput
703         (
704             tex.count["g_shipout_readonly_int"],
705             "l3pdf\int_use:N\g__pdf_backend_name_int",
706             "\pdf_object_ref:n { #2 }"
707         )
708     }
709 }
710 }
711 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1% #1 eg. Span
712 {
713     \int_gincr:N \g__pdf_backend_name_int
714     \__kernel_backend_literal_page:e
715     { /\exp_not:n{#1} ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
716     \bool_if:NTF \l__pdf_backend_xform_bool
717     {
718         \pdfdict_gput:nee %no handler needed
719         { g__pdf_Core/Xform/Resources/Properties }
720         { l3pdf\int_use:N\g__pdf_backend_name_int }
721         { \__pdf_backend_object_last: }
722     }
723     {
724     \exp_args:Ne \tex_latelua:D
725     {
726         ltx.pdf.Page_Resources_Properties_gput
727         (
728             tex.count["g_shipout_readonly_int"],
729             "l3pdf\int_use:N\g__pdf_backend_name_int",
730             "\__pdf_backend_object_last:"
731         )
732     }
733 }
734 }
735 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
736 {
737     \__kernel_backend_literal_page:n { /#1~BMC }
738 }
739 \cs_set_protected:Npn \__pdf_backend_bdc_contobj:nn #1 #2
740 {
741     \pdf_object_unnamed_write:nn { dict } { #2 }
742     \__pdf_backend_bdcobject:n { #1 }
743 }
744 \cs_set_protected:Npn \__pdf_backend_bdc_contstream:nn #1 #2
745 {
746     \__kernel_backend_literal_page:n { /#1-<<#2>>~BDC }
747 }
748
749 \cs_set_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contstream:nn

```

```

750
751 \bool_if:NT\l__pdfmanagement_delayed_shipout_bool
752 {
753   \cs_set_protected:Npn \__pdf_backend_bdc_shipout_contstream:ee #1 #2
754   {
755     \__kernel_backend_shipout_literal_page:e { /#1~<<#2>>~BDC }
756   }
757   \cs_set_eq:NN \__pdf_backend_bdc_shipout:ee \__pdf_backend_bdc_shipout_contstream:ee
758 }
759
760 \cs_set_protected:Npn \__pdf_backend_emc:
761 {
762   \__kernel_backend_literal_page:n { EMC }
763 }
764
765 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1 {}
766 </luatex>

```

pdflatex is the most complicated if we want to use properties as it has to go through the aux ... the push command is extended to take other resources too

```

767 <*pdftex>
768 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
769 {
770   \int_gincr:N \g__pdf_backend_name_int
771   \__kernel_backend_literal_page:e
772   { /#1 ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
773   % code to set the property ...
774   \int_gincr:N\g__pdf_backend_resourceid_int
775   \bool_if:NTF \l__pdf_backend_xform_bool
776   {
777     \pdfdict_gput:nee %no handler needed
778     { g__pdf_Core/Xform/Resources/Properties }
779     { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
780     { \pdf_object_ref:n { #2 } }
781   }
782   {
783     \__pdf_backend_record_abspage:e {l3pdf\int_use:N\g__pdf_backend_resourceid_int}
784     \tl_set:Ne \l__pdf_tmpa_tl
785     {
786       \__pdf_backend_ref_abspage:e{l3pdf\int_use:N\g__pdf_backend_resourceid_int}
787     }
788     \pdfdict_if_exist:nF { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
789     {
790       \pdfdict_new:n { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
791     }
792     \pdfdict_gput:nee
793     { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
794     { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
795     { \pdf_object_ref:n{#2} }
796   }
797 }
798 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1% #1 eg. Span
799 {
800   \int_gincr:N \g__pdf_backend_name_int

```

```

801  \__kernel_backend_literal_page:e
802  { /\exp_not:n{#1} ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
803  % code to set the property ...
804  \int_gincr:N\g__pdf_backend_resourceid_int
805  \bool_if:NTF \l__pdf_backend_xform_bool
806  {
807    \pdfdict_gput:nee
808    { g__pdf_Core/Xform/Resources/Properties }
809    { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
810    { \__pdf_backend_object_last: }
811  }
812  {
813    \__pdf_backend_record_abspage:e{l3pdf\int_use:N\g__pdf_backend_resourceid_int}
814    \tl_set:Ne \l__pdf_tmpa_tl
815    {
816      \__pdf_backend_ref_abspage:e{l3pdf\int_use:N\g__pdf_backend_resourceid_int}
817    }
818    \pdfdict_if_exist:nF { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties
819    {
820      \pdfdict_new:n { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
821    }
822    \pdfdict_gput:nee
823    { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
824    { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
825    { \__pdf_backend_object_last: }
826    %\pdfdict_show:n { g_backend_Page\l__pdf_tmpa_tl/Resources/Properties }
827  }
828  }
829  \cs_set_protected:Npn \__pdf_backend_bmc:n #1
830  {
831    \__kernel_backend_literal_page:n { /#1~BMC }
832  }
833  \cs_set_protected:Npn \__pdf_backend_bdc_contobj:nn #1 #2
834  {
835    \pdf_object_unnamed_write:nn { dict } { #2 }
836    \__pdf_backend_bdcobject:n { #1 }
837  }
838  \cs_set_protected:Npn \__pdf_backend_bdc_contstream:nn #1 #2
839  {
840    \__kernel_backend_literal_page:n { /#1~<<#2>>~BDC }
841  }

```

We use by default the direct BDC.

```

842  \cs_set_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contstream:nn
843
844  \bool_if:NT\l__pdfmanagement_delayed_shipout_bool
845  {
846    \cs_set_protected:Npn \__pdf_backend_bdc_shipout_contstream:ee #1 #2
847    {
848      \__kernel_backend_shipout_literal_page:e { /#1~<<#2>>~BDC }
849    }
850    \cs_set_eq:NN \__pdf_backend_bdc_shipout:ee \__pdf_backend_bdc_shipout_contstream:ee
851  }
852
853  \cs_set_protected:Npn \__pdf_backend_emc:

```

```

854 {
855   \__kernel_backend_literal_page:n { EMC }
856 }
857
858 \cs_new:Npn \__pdf_backend_PageResources_gpush_aux:n #1 %#1 ExtGState etc
859 {
860   \prop_if_empty:cF
861     { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/#1} }
862     {
863       \pdfdict_item:ne { #1 }{ \pdf_object_ref:n {__pdf/Page/Resources/#1}}
864     }
865 }
866
867 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1
868 {
869   \exp_args:NNe \tex_global:D \tex_pdfpageresources:D
870   {
871     \prop_if_exist:cT
872       { \__kernel_pdfdict_name:n { g__pdf_Core/backend_Page#1/Resources/Properties } }
873       {
874         /Properties~
875         <<
876         \prop_map_function:cN
877           { \__kernel_pdfdict_name:n { g__pdf_Core/backend_Page#1/Resources/Property
878             \pdfdict_item:ne
879           }
880         }
881         %% add ExtGState etc
882         \clist_map_function:NN
883           \c__pdf_backend_PageResources_clist
884           \__pdf_backend_PageResources_gpush_aux:n
885       }
886   }
887
888 </pdftex>

```

*(End of definition for \\_\_pdf\_backend\_bdc:nn and others.)*

## 1.10 “Catalog” & subdirectories (pdfcatalog)

The backend command is already in the driver: \\_\_pdf\_backend\_catalog\_gput:nn

### 1.10.1 Special case: the /Names/EmbeddedFiles dictionary

Entries to /Names are handled differently, in part (/Desc) it is automatic, for other special commands like \pdfnames must be used. For EmbeddedFiles dvips wants code for every file and then creates the Name tree automatically. Other name trees are ignored. TODO: Currently the code for EmbeddedFiles is still a bit different but this should be merged, all name trees should be handled with the same code.

```

889 % pdflatex
890 <*pdftex>
891 \cs_new_protected:Npn \__pdf_backend_Names_gpush:nn #1 #2 %#1 name of name tree, #2 array co
892 {
893   \pdf_object_unnamed_write:nn {dict} {/Names [#2] }

```

```

894     \tex_pdfnames:D {/#1~\pdf_object_ref_last:}
895   }
896 </pdftex>
897 <*luatex>
898 \cs_new_protected:Npn \__pdf_backend_Names_gpush:nn #1 #2 %#1 name of name tree, #2 array co
899   {
900     \pdf_object_unnamed_write:nn {dict} {/Names [#2] }
901     \tex_pdfextension:D~names~ {/#1~\pdf_object_ref_last:}
902   }
903 </luatex>
904 <*dvipdfmx | xdvipdfmx>
905 \cs_new_protected:Npn \__pdf_backend_Names_gpush:nn #1 #2 %#1 name of name tree, #2 array co
906   {
907     \pdf_object_unnamed_write:nn {dict} {/Names [#2] }
908     \__pdf_backend:e {put~@names~<</#1~\pdf_object_ref_last: >>}
909   }
910 </dvipdfmx | xdvipdfmx>
911
912 %dvips: noop
913 <*dvips>
914 \cs_new_protected:Npn \__pdf_backend_Names_gpush:nn #1 #2 {}
915 </dvips>
916 %dvisvgm: noop
917 <*dvisvgm>
918 \cs_new_protected:Npn \__pdf_backend_Names_gpush:nn #1 #2 {}
919 </dvisvgm>

```

EmbeddedFiles is a bit special. For once we need backend commands for dvips. But we want also an option to create the name on the fly.

`\__pdf_backend_NamesEmbeddedFiles_add:nn` dvips need special backend code to create the name tree. With the other engines it does nothing.

```

920 <*pdftex | luatex | dvipdfmx | xdvipdfmx>
921 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_add:nn #1 #2 {}
922 </pdftex | luatex | dvipdfmx | xdvipdfmx>
923 <*dvips>
924 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_add:nn #1 #2
925   {
926     \__pdf_backend_pdfmark:e
927     {
928       /Name~#1~
929       /FS~#2~
930       /EMBED
931     }
932   }
933 </dvips>
934 <*dvisvgm>
935 %no op. Or is there any sensible use for it?
936 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_add:nn #1 #2
937   {}
938
939 </dvisvgm>

```

*(End of definition for \\_\_pdf\_backend\_NamesEmbeddedFiles\_add:nn.)*

### 1.10.2 Additional annotation commands

Starting with texlive 2021 pdftex and luatex offer commands to interrupt a link. That can for example be used to exclude the header and footer from the link. The backend support is now in l3kernel. We only provide the user command.

```
940 <*pdftex>
941 \cs_if_exist:NT \pdfrunninglinkoff
942 {
943   \cs_set_protected:Npn \__pdfannot_backend_link_off:
944     {
945       \pdfrunninglinkoff
946     }
947   \cs_set_protected:Npn \__pdfannot_backend_link_on:
948     {
949       \pdfrunninglinkon
950     }
951 }
952 </pdftex>
```

### 1.10.3 FormXObject / backend

```
\__pdf_backend_xform_new:nnnn #1 : name
                               #2 : attributes
                               #3 : resources needed?? or are all resources autogenerated?
                               #4 : content, this doesn't need to be a box!
```

```
\__pdf_backend_xform_use:n      953 <*pdftex>
\__pdf_backend_xform_ref:n      954 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4
                               955 % #1 name
                               956 % #2 attributes
                               957 % #3 resources
                               958 % #4 content, not necessarily a box!
                               959 {
                               960   \hbox_set:Nn \l__pdf_backend_tmpa_box
                               961     {
                               962       \bool_set_true:N \l__pdf_backend_xform_bool
                               963       \prop_gc_clear:c {\__kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Properties }}
                               964       #4
                               965     }
                               966   %store the dimensions
                               967   \tl_const:ce
                               968     { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
                               969     { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
                               970   \tl_const:ce
                               971     { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
                               972     { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
                               973   \tl_const:ce
                               974     { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
                               975     { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
                               976   %% do we need to test if #2 and #3 are empty??
                               977   \tex_immediate:D \tex_pdfxform:D
                               978     ~ attr      ~ { #2 }
```

```

979     %% which other resources should be default? Is an argument actually needed?
980     ~ resources ~
981     {
982         #3
983         \int_compare:nNnT
984             { \prop_count:c { \__kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Propertie
985             >
986             { 0 }
987             {
988                 /Properties~
989                 <<
990                 \pdfdict_use:n { g__pdf_Core/Xform/Resources/Properties }
991                 >>
992             }
993
994         \prop_if_empty:cF
995             { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/ExtGState } }
996             {
997                 /ExtGState~ \pdf_object_ref:n { __pdf/Page/Resources/ExtGState }
998             }
999         \prop_if_empty:cF
1000             { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/Pattern } }
1001             {
1002                 /Pattern~ \pdf_object_ref:n { __pdf/Page/Resources/Pattern }
1003             }
1004         \prop_if_empty:cF
1005             { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/Shading } }
1006             {
1007                 /Shading~ \pdf_object_ref:n { __pdf/Page/Resources/Shading }
1008             }
1009         \prop_if_empty:cF
1010             { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/ColorSpace } }
1011             {
1012                 /ColorSpace~ \pdf_object_ref:n { __pdf/Page/Resources/ColorSpace }
1013             }
1014     }
1015     \l__pdf_backend_tmpa_box
1016     \int_const:cn
1017     { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1018     { \tex_pdflastxform:D }
1019 }
1020
1021 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
1022 {
1023     \tex_pdfrefxform:D
1024     \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1025     \scan_stop:
1026 }
1027
1028 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1029 {
1030     \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int } ~ 0 ~ R
1031 }
1032 </pdfTeX>

```

```

1033 <*luatex>
1034 %luatex
1035 %nearly identical but not completely ...
1036 \cs_new_protected:Npn \__pdf_backend_xform_new:nmmm #1 #2 #3 #4
1037 % #1 name
1038 % #2 attributes
1039 % #3 resources
1040 % #4 content, not necessarily a box!
1041 {
1042   \hbox_set:Nn \l__pdf_backend_tmpa_box
1043   {
1044     \bool_set_true:N \l__pdf_backend_xform_bool
1045     \prop_gclear:c { \__kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Properties }
1046     #4
1047   }
1048   \tl_const:ce
1049   { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1050   { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
1051   \tl_const:ce
1052   { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1053   { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
1054   \tl_const:ce
1055   { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1056   { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
1057   %% do we need to test if #2 and #3 are empty??
1058   \tex_immediate:D \tex_pdfxform:D
1059   ~ attr ~ { #2 }
1060   %% which resources should be default? Is an argument actually needed?
1061   ~ resources ~
1062   {
1063     #3
1064     \int_compare:nNnT
1065     { \prop_count:c { \__kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Properties
1066     >
1067     { 0 }
1068     {
1069       /Properties~
1070       <<
1071       \pdfdict_use:n { g__pdf_Core/Xform/Resources/Properties }
1072       >>
1073     }
1074     \prop_if_empty:cF
1075     { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/ExtGState } }
1076     {
1077       /ExtGState~ \pdf_object_ref:n { __pdf/Page/Resources/ExtGState }
1078     }
1079     \prop_if_empty:cF
1080     { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/Pattern } }
1081     {
1082       /Pattern~ \pdf_object_ref:n { __pdf/Page/Resources/Pattern }
1083     }
1084     \prop_if_empty:cF
1085     { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/Shading } }
1086     {

```



```

1087         /Shading~ \pdf_object_ref:n { __pdf/Page/Resources/Shading }
1088     }
1089     \prop_if_empty:cF
1090     { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/ColorSpace } }
1091     {
1092         /ColorSpace~ \pdf_object_ref:n { __pdf/Page/Resources/ColorSpace }
1093     }
1094 }
1095 \l__pdf_backend_tmpa_box
1096 \int_const:cn
1097 { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1098 { \tex_pdflastxform:D }
1099 }
1100
1101 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1 %protected as with xelatex
1102 {
1103     \tex_pdfrefxform:D \int_use:c
1104     {
1105         c__pdf_backend_xform_ \tl_to_str:n {#1} _int
1106     }
1107     \scan_stop:
1108 }
1109
1110 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1111 { \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int } ~ 0 ~ R }
1112
1113 </luatex>
1114 <*dvipdfmx | xdvipdfmx>
1115 % xetex
1116 % it needs a bit testing if it really works to set the box to 0 before the special ...
1117 % does it disturb viewing the xobject?
1118 % what happens with the resources (bdc)? (should work as they are specials too)
1119 % xetex requires that the special is in horizontal mode. This means it affects
1120 % typesetting. But we can no delay the whole form code to shipout
1121 % as the object reference and the size is often wanted on the current page.
1122 % so we need to allocate a box - but probably they won't be thousands xform
1123 % in a document so it shouldn't matter.
1124 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4
1125 % #1 name
1126 % #2 attributes
1127 % #3 resources
1128 % #4 content, not necessarily a box!
1129 {
1130     \int_gincr:N \g__pdf_backend_object_int
1131     \int_const:cn
1132     { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1133     { \g__pdf_backend_object_int }
1134     \box_new:c { g__pdf_backend_xform_#1_box }
1135     \hbox_gset:cn { g__pdf_backend_xform_#1_box }
1136     {
1137         \bool_set_true:N \l__pdf_backend_xform_bool
1138         #4
1139     }
1140     \tl_const:ce

```

```

1141     { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1142     { \tex_the:D \box_wd:c { g__pdf_backend_xform_#1_box } }
1143 \tl_const:ce
1144     { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1145     { \tex_the:D \box_ht:c { g__pdf_backend_xform_#1_box } }
1146 \tl_const:ce
1147     { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1148     { \tex_the:D \box_dp:c { g__pdf_backend_xform_#1_box } }
1149 \box_set_dp:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1150 \box_set_ht:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1151 \box_set_wd:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1152 \hook_gput_next_code:nn {shipout/background}
1153 {
1154     \mode_leave_vertical: %needed, the xform disappears without it.
1155     \__pdf_backend:e
1156     {
1157         bxobj ~ \__pdf_backend_xform_ref:n { #1 }
1158         \c_space_tl width ~ \pdfxform_wd:n { #1 }
1159         \c_space_tl height ~ \pdfxform_ht:n { #1 }
1160         \c_space_tl depth ~ \pdfxform_dp:n { #1 }
1161     }
1162     \box_use_drop:c { g__pdf_backend_xform_#1_box }
1163     \__pdf_backend:e {put ~ @resources ~<<#3>> }
1164     \__pdf_backend:e
1165     {
1166         put~ @resources ~
1167         <<
1168             /ExtGState~ \pdf_object_ref:n { __pdf/Page/Resources/ExtGState }
1169         >>
1170     }
1171     \__pdf_backend:e
1172     {
1173         put~ @resources ~
1174         <<
1175             /Pattern~ \pdf_object_ref:n { __pdf/Page/Resources/Pattern }
1176         >>
1177     }
1178     \__pdf_backend:e
1179     {
1180         put~ @resources ~
1181         <<
1182             /Shading~ \pdf_object_ref:n { __pdf/Page/Resources/Shading }
1183         >>
1184     }
1185     \__pdf_backend:e
1186     {
1187         put~ @resources ~
1188         <<
1189             /ColorSpace~
1190             \pdf_object_ref:n { __pdf/Page/Resources/ColorSpace }
1191         >>
1192     }
1193     \__pdf_backend:e {exobj ~<<#2>>}
1194 }

```

```

1195     }
1196
1197
1198
1199 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1200 {
1201     @pdf.xform \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1202 }
1203
1204 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
1205 {
1206     \hbox_set:Nn \l__pdf_backend_tmpa_box
1207     {
1208         \__pdf_backend:e
1209         {
1210             uxobj~ \__pdf_backend_xform_ref:n { #1 }
1211         }
1212     }
1213     \box_set_wd:Nn \l__pdf_backend_tmpa_box { \pdfxform_wd:n { #1 } }
1214     \box_set_ht:Nn \l__pdf_backend_tmpa_box { \pdfxform_ht:n { #1 } }
1215     \box_set_dp:Nn \l__pdf_backend_tmpa_box { \pdfxform_dp:n { #1 } }
1216     \box_use_drop:N \l__pdf_backend_tmpa_box
1217 }
1218 </dviptfm>|xdviptfm>
1219 <*dvisvgm>
1220 % unclear what it should do!!
1221 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4 {}
1222 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1 {}
1223 \cs_new:Npn \__pdf_backend_xform_ref:n {}
1224 </dvisvgm>

```

The xform code for dvips is based on code from the attachfile2 package (in atfi-dvips), along with some ideas from pdfbase and has been corrected with the help of Alexander Grahn. Details like clipping and landscape will probably be corrected in the future. We need some temporary variables to store dimensions

```

1225 <*dvips>
1226 \tl_new:N \l__pdf_backend_xform_tmpwd_tl
1227 \tl_new:N \l__pdf_backend_xform_tmpdp_tl
1228 \tl_new:N \l__pdf_backend_xform_tmpht_tl
1229 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4 % #1 name, #2 attribute, #4
1230 {
1231     \int_gincr:N \g__pdf_backend_object_int
1232     \int_const:cn
1233     { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1234     { \g__pdf_backend_object_int }
1235
1236     \hbox_set:Nn \l__pdf_backend_tmpa_box
1237     {
1238         \bool_set_true:N \l__pdf_backend_xform_bool
1239         \prop_gclear:c {\__kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Properties }}
1240         #4
1241     }
1242     %store the dimensions
1243     \tl_const:ce

```

```

1244     { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1245     { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
1246 \tl_const:ce
1247     { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1248     { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
1249 \tl_const:ce
1250     { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1251     { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
1252 %store content dimensions in DPI units (Dots) (code from issue 25)
1253 \tl_set:N\l__pdf_backend_xform_tmpwd_tl
1254 {
1255     \dim_to_decimal_in_sp:n{ \box_wd:N \l__pdf_backend_tmpa_box }~
1256     65536~div~72.27~div~DVImag~mul~Resolution~mul~
1257 }
1258 \tl_set:N\l__pdf_backend_xform_tmph_tl
1259 {
1260     \dim_to_decimal_in_sp:n{ \box_ht:N \l__pdf_backend_tmpa_box }~
1261     65536~div~72.27~div~DVImag~mul~VResolution~mul~
1262 }
1263 \tl_set:N\l__pdf_backend_xform_tmppd_tl
1264 {
1265     \dim_to_decimal_in_sp:n{ \box_dp:N \l__pdf_backend_tmpa_box }~
1266     65536~div~72.27~div~DVImag~mul~VResolution~mul~
1267 }
1268 % mirror the box
1269 %\box_scale:Nnn \l__pdf_backend_tmpa_box {1} {-1}
1270 \hbox_set:Nn\l__pdf_backend_tmpb_box
1271 {
1272     \__kernel_backend_postscript:e
1273     {
1274         gsave~currentpoint~
1275         initclip~ % restore default clipping path (page device/whole page)
1276         clippath~pathbbox~newpath~pop~pop~
1277         \tl_use:N\l__pdf_backend_xform_tmppd_tl~add~translate~
1278         mark~
1279         /objdef~{ pdf.obj \int_use:N\g__pdf_backend_object_int }~\c_space_tl~
1280         /BBox[
1281             0~
1282             \tl_use:N\l__pdf_backend_xform_tmph_tl~
1283             \tl_use:N\l__pdf_backend_xform_tmpwd_tl~
1284             \tl_use:N\l__pdf_backend_xform_tmppd_tl~
1285             neg
1286         ]
1287         \str_if_eq:eeF{#1}{}
1288         {
1289             product~(Distiller)~search~{pop~pop~pop~#2}{pop}ifelse~
1290         }
1291         /BP~pdfmark~1~-1~scale~neg~exch~neg~exch~translate
1292     }
1293 \box_use_drop:N\l__pdf_backend_tmpa_box
1294 \__kernel_backend_postscript:n
1295 {
1296     mark ~ /EP~pdfmark ~ grestore
1297 }

```

```

1298     \str_if_eq:eeF{#1}{}
1299     {
1300         \__kernel_backend_postscript:e
1301         {
1302             product~(Ghostscript)~search~
1303             {
1304                 pop~pop~pop~
1305                 mark~
1306                 { pdf.obj \int_use:c{c__pdf_backend_xform_ \tl_to_str:n {#1} _int} }
1307                 ~<<#2>>~/PUT~pdfmark
1308             }{pop}ifelse
1309         }
1310     }
1311 }
1312 \box_set_dp:Nn \l__pdf_backend_tmpb_box { \c_zero_dim }
1313 \box_set_ht:Nn \l__pdf_backend_tmpb_box { \c_zero_dim }
1314 \box_set_wd:Nn \l__pdf_backend_tmpb_box { \c_zero_dim }
1315 \hook_gput_code:nnn {begindocument/end}{pdfxform}
1316 {
1317     \mode_leave_vertical:
1318     \box_use:N\l__pdf_backend_tmpb_box
1319 }
1320 }
1321
1322
1323 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
1324 {
1325     \hbox_set:Nn \l__pdf_backend_tmpa_box
1326     {
1327         \__kernel_backend_postscript:e
1328         {
1329             gsave~currentpoint~translate~1~-1~scale~
1330             mark~{ pdf.obj \int_use:c{c__pdf_backend_xform_ \tl_to_str:n {#1} _int} }~
1331             /SP~pdfmark ~ grestore
1332         }
1333     }
1334     \box_set_wd:Nn \l__pdf_backend_tmpa_box { \pdfxform_wd:n { #1 } }
1335     \box_set_ht:Nn \l__pdf_backend_tmpa_box { \pdfxform_ht:n { #1 } }
1336     \box_set_dp:Nn \l__pdf_backend_tmpa_box { \pdfxform_dp:n { #1 } }
1337     \box_use_drop:N \l__pdf_backend_tmpa_box
1338 }
1339 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1340 {
1341     { pdf.obj \int_use:c{c__pdf_backend_xform_ \tl_to_str:n {#1} _int} }
1342 }
1343
1344 </dvips>
1345 <*drivers>
1346 %% all
1347 \prg_new_conditional:Npnn \__pdf_backend_xform_if_exist:n #1 { p , T , F , TF }
1348 {
1349     \int_if_exist:cTF { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1350     { \prg_return_true: }
1351     { \prg_return_false: }

```

```

1352 }
1353 \prg_new_eq_conditional:Nn \pdfxform_if_exist:n\__pdf_backend_xform_if_exist:n
1354 { TF , T , F , p }
1355 </drivers>

```

*(End of definition for \\_\_pdf\_backend\_xform\_new:n, \\_\_pdf\_backend\_xform\_use:n, and \\_\_pdf\_backend\_xform\_ref:n.)*

## 1.11 Structure Destinations

Standard destinations consist of a reference to a page in the pdf and instructions how to display it—typically they will put a specific location in the left top corner of the viewer and so give the impression that a link jumped to the word in this place. But in reality they are not connected to the content.

Starting with pdf 2.0 destinations can in a tagged PDF also point to a structure, to a /StructElem object. GoTo links can then additionally to the /D key pointing to a page destination also point to such a structure destination with an /SD key. Programs that e.g. convert such a PDF to html can then create better links. (According to the reference, PDF-viewer should prefer the structure destination over the page destination, but as far as it is known this isn't done yet.)

Currently structure destinations and GoTo links making use of it could natively only be created with the dvipdfmx backend. With pdftex and luatex it was only possible to create a restricted type which used only the “Fit” mode. Starting with T<sub>E</sub>Xlive 2022 (earlier in miktex) both engine will know new keywords which allow to create structure destination easily.

The following backend code prepares the use of structure destinations. The general idea is that if structure destinations are used, they should be used always. So we define alternative commands which can be activated by mapping them to the standard backend commands.

The needed code differ depending on if structure objects use standard or indexed object names. At the end we will probably always use indexed objects, but for now we offer both options.

`\l_pdf_current_structure_destination_tl`

This command holds the name of the structure object to use in the following commands which creates a destination. The code which activates structure destinations must also ensure that it has a sensible, expandable content. `tagpdf` for example will define it as

```

\tl_set:Nn \l_pdf_current_structure_destination_tl { __tag/struct/\g__tag_struct_stack
or if indexed structure object names are used

```

```

\tl_set:Nn \l_pdf_current_structure_destination_tl { {__tag/struct}{\g__tag_struct_sta
1356 <*drivers>
1357 \tl_new:N \l_pdf_current_structure_destination_tl
1358 </drivers>

```

*(End of definition for \l\_pdf\_current\_structure\_destination\_tl.)*

We will define alternatives for three backend commands:

```

\__pdf_backend_destination:nn -> \__pdf_backend_structure_destination:nn
\__pdf_backend_destination:nm -> \__pdf_backend_structure_destination:nm
\__pdfannot_backend_link_begin_goto:nm -> \__pdf_backend_link_begin_structure_goto:nm
\__pdf_backend_destination:nn -> \__pdf_backend_indexed_structure_destination:nn

```

```

\__pdf_backend_destination:nmmm -> \__pdf_backend_indexed_structure_destination:nmmm
\__pdfannot_backend_link_begin_goto:nmw -> \__pdf_backend_indexed_link_begin_structur

```

Activating means mapping them onto the original commands. Be aware that not all engines and compilation routes support structure destinations, for them the command will be a no-op.

```

\pdf_activate_structure_destination:
pdf_activate_indexed_structure_destination:
1359 <*drivers>
1360 \cs_new_protected:Npn \pdf_activate_structure_destination:
1361 {
1362   \cs_gset_eq:NN \__pdf_backend_destination:nn \__pdf_backend_structure_destination:nn
1363   \cs_gset_eq:NN \__pdf_backend_destination:nmmm \__pdf_backend_structure_destination:nmmm
1364   \cs_gset_eq:NN \__pdfannot_backend_link_begin_goto:nmw \__pdfannot_backend_link_begin_goto:nmw
1365 }
1366 \cs_new_protected:Npn \pdf_activate_indexed_structure_destination:
1367 {
1368   \cs_gset_eq:NN \__pdf_backend_destination:nn \__pdf_backend_indexed_structure_destination:nn
1369   \cs_gset_eq:NN \__pdf_backend_destination:nmmm \__pdf_backend_indexed_structure_destination:nmmm
1370   \cs_gset_eq:NN \__pdfannot_backend_link_begin_goto:nmw \__pdfannot_backend_link_begin_goto:nmw
1371 }
1372 </drivers>

```

(End of definition for \pdf\_activate\_structure\_destination: and \pdf\_activate\_indexed\_structure\_destination:.)

Now the driver dependent parts. By default the new commands are simply copies of the original commands. We adapt them then for the engines and engine version which provide support for structure destinations.

```

1373 <*drivers>
1374 \cs_set_eq:NN \__pdf_backend_structure_destination:nn \__pdf_backend_destination:nn
1375 \cs_set_eq:NN \__pdf_backend_structure_destination:nmmm \__pdf_backend_destination:nmmm
1376 \cs_set_eq:NN \__pdfannot_backend_link_begin_structure_goto:nmw \__pdfannot_backend_link_begin_goto:nmw
1377 \cs_set_eq:NN \__pdf_backend_indexed_structure_destination:nn \__pdf_backend_destination:nn
1378 \cs_set_eq:NN \__pdf_backend_indexed_structure_destination:nmmm \__pdf_backend_destination:nmmm
1379 </drivers>

```

```

\__pdf_backend_structure_destination:nn
\__pdf_backend_structure_destination:nmmm
\__pdfannot_backend_link_begin_structure_goto:nmw

```

These commands are the backend commands to create a destination. which create also a structure destination. At first xetex/dvipdfmx. The structure destination is an array, so we use obj for it so that we can reference it:

```

1380 <*xdvipdfmx | dvipdfmx>
1381 \cs_set_protected:Npn \__pdf_backend_structure_destination:nn #1#2
1382 {
1383   \__pdf_backend:e
1384   {
1385     dest ~ ( \exp_not:n {#1} )
1386     [
1387       @thispage
1388       \str_case:nnF {#2}
1389       {
1390         { xyz } { /XYZ ~ @xpos ~ @ypos ~ null }
1391         { fit } { /Fit }
1392         { fitb } { /FitB }
1393         { fitbh } { /FitBH }

```

```

1394         { fitbv } { /FitBV ~ @xpos }
1395         { fith } { /FitH ~ @ypos }
1396         { fitv } { /FitV ~ @xpos }
1397         { fitr } { /Fit }
1398     }
1399     { /XYZ ~ @xpos ~ @ypos ~ \fp_eval:n { (#2) / 100 } }
1400 ]
1401 }

```

We test if the structure object exist. The object of the structure destination gets the name `@pdf.Sdest.<destname>`, where `<destname>` is the name of the standard destination so that we can reference it in the GoTo links.

```

1402 \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1403 {
1404   \__pdf_backend:e
1405   {
1406     obj ~ @pdf.SDest.\exp_not:n{#1}
1407     [
1408       \exp_args:Ne \pdf_object_ref:n { \l_pdf_current_structure_destination_tl }
1409       \str_case:nnF {#2}
1410       {
1411         { xyz } { /XYZ ~ @xpos ~ @ypos ~ null }
1412         { fit } { /Fit }
1413         { fitb } { /FitB }
1414         { fitbh } { /FitBH }
1415         { fitbv } { /FitBV ~ @xpos }
1416         { fith } { /FitH ~ @ypos }
1417         { fitv } { /FitV ~ @xpos }
1418         { fitr } { /Fit }
1419       }
1420       { /XYZ ~ @xpos ~ @ypos ~ \fp_eval:n { (#2) / 100 } }
1421     ]
1422   }
1423 }
1424 }

```

The second destination command is for the boxed destination. Here we need to define an new auxiliary command:

```

1425 \cs_new_protected:Npn \__pdf_backend_structure_destination_aux:nnnn #1#2#3#4
1426 {
1427   \vbox_to_zero:n
1428   {
1429     \__kernel_kern:n {#4}
1430     \hbox:n
1431     {
1432       \__pdf_backend:n { obj ~ @pdf_ #2 _llx ~ @xpos }
1433       \__pdf_backend:n { obj ~ @pdf_ #2 _lly ~ @ypos }
1434     }
1435     \tex_vss:D
1436   }
1437   \__kernel_kern:n {#1}
1438   \vbox_to_zero:n
1439   {
1440     \__kernel_kern:n { -#3 }
1441     \hbox:n

```



```

1442     {
1443       \_pdf_backend:n
1444       {
1445         dest ~ (#2)
1446         [
1447           @thispage
1448           /FitR ~
1449           @pdf_ #2 _llx ~ @pdf_ #2 _lly ~
1450           @xpos ~ @ypos
1451         ]
1452       }

```

Here we add the structure destination to the same box

```

1453       \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1454       {
1455         \_pdf_backend:e
1456         {
1457           obj ~ @pdf.SDest.\exp_not:n{#2}
1458           [
1459             \exp_args:Ne \pdf_object_ref:n { \l_pdf_current_structure_destination_
1460             /FitR ~
1461             @pdf_ #2 _llx ~ @pdf_ #2 _lly ~
1462             @xpos ~ @ypos
1463           ]
1464         }
1465       }
1466     }
1467     \tex_vss:D
1468   }
1469   \_kernel_kern:n { -#1 }
1470 }

```

And now we redefine the destination command:

```

1471 \cs_set_protected:Npn \_pdf_backend_structure_destination:nnnn #1#2#3#4
1472 {
1473   \exp_args:Ne \_pdf_backend_structure_destination_aux:nnnn
1474   { \dim_eval:n {#2} } {#1} {#3} {#4}
1475 }

```

At last the goto link.

```

1476 \cs_set_protected:Npn \_pdfannot_backend_link_begin_structure_goto:nnw #1#2
1477 {
1478   \_pdfannot_backend_link_begin:n { #1 /Subtype /Link /A << /S /GoTo /D ( #2 ) /SD~@pdf.S
1479 }
1480 </xdvipdfmx | dvipdfmx>

```

Now pdftex. We only redefine for version 1.40 revision 24 or later.

```

1481 <*pdftex>
1482 \bool_lazy_and:nnT
1483 { \int_compare_p:nNn {\tex_pdftexversion:D } > {139} }
1484 { \int_compare_p:nNn {\tex_pdftexrevision:D } > {23} }
1485 {
1486   \cs_set_protected:Npn \_pdf_backend_structure_destination:nn #1#2
1487   {
1488     \tex_pdfdest:D
1489     name {#1}

```

```

1490     \str_case:nnF {#2}
1491     {
1492         { xyz }   { xyz }
1493         { fit }   { fit }
1494         { fitb }  { fitb }
1495         { fitbh } { fitbh }
1496         { fitbv } { fitbv }
1497         { fith }  { fith }
1498         { fitv }  { fitv }
1499         { fitr }  { fitr }
1500     }
1501     { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1502     \scan_stop:
1503 \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1504 {
1505     \tex_pdfdest:D
1506     struct~
1507     \int_use:c
1508     { c__pdf_object_ \exp_args:Ne \tl_to_str:n {\l_pdf_current_structure_destin
1509     name {#1}
1510     \str_case:nnF {#2}
1511     {
1512         { xyz }   { xyz }
1513         { fit }   { fit }
1514         { fitb }  { fitb }
1515         { fitbh } { fitbh }
1516         { fitbv } { fitbv }
1517         { fith }  { fith }
1518         { fitv }  { fitv }
1519         { fitr }  { fitr }
1520     }
1521     { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1522     \scan_stop:
1523 }
1524 }
1525 \cs_set_protected:Npn \__pdf_backend_structure_destination:nnnn #1#2#3#4
1526 {
1527     \tex_pdfdest:D
1528     name {#1}
1529     fitr ~
1530     width \dim_eval:n {#2} ~
1531     height \dim_eval:n {#3} ~
1532     depth \dim_eval:n {#4} \scan_stop:
1533 \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1534 {
1535     \tex_pdfdest:D
1536     struct~
1537     \int_use:c
1538     { c__pdf_object_ \exp_args:Ne \tl_to_str:n {\l_pdf_current_structure_destinat
1539     name {#1}
1540     fitr ~
1541     width \dim_eval:n {#2} ~
1542     height \dim_eval:n {#3} ~
1543     depth \dim_eval:n {#4} \scan_stop:

```

```

1544     }
1545   }
1546   \cs_set_protected:Npn \__pdfannot_backend_link_begin_structure_goto:nmw #1#2
1547   {
1548     \__pdfannot_backend_link_begin:nmw {#1} { goto~struct~name~{#2}~name } {#2}
1549   }
1550 }
1551 </pdfTeX>

```

luatex is quite similar to pdfTeX. Mostly the test for the version is different

```

1552 <*luatex>
1553 \int_compare:nNnT {\directlua{tex.print(status.list()["development_id"])} } > {7468}
1554 {
1555   \cs_set_protected:Npn \__pdf_backend_structure_destination:nn #1#2
1556   {
1557     \tex_pdfextension:D dest
1558     name {#1}
1559     \str_case:nnF {#2}
1560     {
1561       { xyz } { xyz }
1562       { fit } { fit }
1563       { fitb } { fitb }
1564       { fitbh } { fitbh }
1565       { fitbv } { fitbv }
1566       { fith } { fith }
1567       { fitv } { fitv }
1568       { fitr } { fitr }
1569     }
1570     { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1571     \scan_stop:
1572     \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1573     {
1574       \tex_pdfextension:D dest
1575       struct~
1576       \int_use:c
1577       { c__pdf_object_ \exp_args:Ne \tl_to_str:n {\l_pdf_current_structure_destin
1578       name {#1}
1579       \str_case:nnF {#2}
1580       {
1581         { xyz } { xyz }
1582         { fit } { fit }
1583         { fitb } { fitb }
1584         { fitbh } { fitbh }
1585         { fitbv } { fitbv }
1586         { fith } { fith }
1587         { fitv } { fitv }
1588         { fitr } { fitr }
1589       }
1590       { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1591       \scan_stop:
1592     }
1593   }
1594   \cs_set_protected:Npn \__pdf_backend_structure_destination:nnnn #1#2#3#4
1595   {
1596     \tex_pdfextension:D dest

```

```

1597     name {#1}
1598     fitr ~
1599     width \dim_eval:n {#2} ~
1600     height \dim_eval:n {#3} ~
1601     depth \dim_eval:n {#4} \scan_stop:
1602 \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1603 {
1604     \tex_pdfextension:D dest
1605     struct~
1606     \int_use:c
1607     { c_pdf_object_ \exp_args:Ne \tl_to_str:n {\l_pdf_current_structure_destination_tl}
1608     name {#1}
1609     fitr ~
1610     width \dim_eval:n {#2} ~
1611     height \dim_eval:n {#3} ~
1612     depth \dim_eval:n {#4} \scan_stop:
1613 }
1614 }
1615 \cs_set_protected:Npn \__pdfannot_backend_link_begin_structure_goto:nmw #1#2
1616 {
1617     \__pdfannot_backend_link_begin:nmw {#1} { goto~struct~name~{#2}~name } {#2}
1618 }
1619 }
1620 </luatex>

```

(End of definition for `\__pdf_backend_structure_destination:nn`, `\__pdf_backend_structure_destination:nmnn`, and `\__pdfannot_backend_link_begin_structure_goto:nmw`.)

`\__pdf_backend_indexed_structure_destination:mn`  
`\__pdf_backend_indexed_structure_destination:nmnn`

This are the indexed variants of the commands to create a destination and a structure destination. At first xetex/dvipdfmx. The structure destination is an array, so we use obj for it so that we can reference it:

```

1621 <*\xdvipdfmx | dvipdfmx>
1622 \cs_set_protected:Npn \__pdf_backend_indexed_structure_destination:mn #1#2
1623 {
1624     \__pdf_backend:e
1625     {
1626     dest ~ ( \exp_not:n {#1} )
1627     [
1628     @thispage
1629     \str_case:nnF {#2}
1630     {
1631     { xyz } { /XYZ ~ @xpos ~ @ypos ~ null }
1632     { fit } { /Fit }
1633     { fitb } { /FitB }
1634     { fitbh } { /FitBH }
1635     { fitbv } { /FitBV ~ @xpos }
1636     { fith } { /FitH ~ @ypos }
1637     { fitv } { /FitV ~ @xpos }
1638     { fitr } { /Fit }
1639     }
1640     { /XYZ ~ @xpos ~ @ypos ~ \fp_eval:n { (#2) / 100 } }
1641     ]
1642     }

```

We do not test anymore if the structure object exist. The object of the structure destination gets the name `@pdf.Sdest.<destname>`, where `<destname>` is the name of the standard destination so that we can reference it in the GoTo links.

```

1643     \_pdf_backend:e
1644     {
1645     obj ~ @pdf.SDest.\exp_not:n{#1}
1646     [
1647     \exp_after:wN \pdf_object_ref_indexed:nn \l_pdf_current_structure_destination_t
1648     \str_case:nnF {#2}
1649     {
1650     { xyz } { /XYZ ~ @xpos ~ @ypos ~ null }
1651     { fit } { /Fit }
1652     { fitb } { /FitB }
1653     { fitbh } { /FitBH }
1654     { fitbv } { /FitBV ~ @xpos }
1655     { fith } { /FitH ~ @ypos }
1656     { fitv } { /FitV ~ @xpos }
1657     { fitr } { /Fit }
1658     }
1659     { /XYZ ~ @xpos ~ @ypos ~ \fp_eval:n { (#2) / 100 } }
1660     ]
1661     }
1662     }

```

The second destination command is for the boxed destination. Here we need to define an new auxiliary command:

```

1663 \cs_new_protected:Npn \_pdf_backend_indexed_structure_destination_aux:nnnn #1#2#3#4
1664 {
1665   \vbox_to_zero:n
1666   {
1667     \_kernel_kern:n {#4}
1668     \hbox:n
1669     {
1670       \_pdf_backend:n { obj ~ @pdf_ #2 _llx ~ @xpos }
1671       \_pdf_backend:n { obj ~ @pdf_ #2 _lly ~ @ypos }
1672     }
1673     \tex_vss:D
1674   }
1675   \_kernel_kern:n {#1}
1676   \vbox_to_zero:n
1677   {
1678     \_kernel_kern:n { -#3 }
1679     \hbox:n
1680     {
1681       \_pdf_backend:n
1682       {
1683         dest ~ (#2)
1684         [
1685           @thispage
1686           /FitR ~
1687           @pdf_ #2 _llx ~ @pdf_ #2 _lly ~
1688           @xpos ~ @ypos
1689         ]
1690       }

```

Here we add the structure destination to the same box

```

1691         \_pdf_backend:e
1692         {
1693             obj ~ @pdf.SDest.\exp_not:n{#2}
1694             [
1695                 \exp_after:wN \pdf_object_ref_indexed:nn \l_pdf_current_structure_destin
1696                 /FitR ~
1697                 @pdf_ #2 _llx ~ @pdf_ #2 _lly ~
1698                 @xpos ~ @ypos
1699             ]
1700         }
1701     }
1702     \tex_vss:D
1703 }
1704 \_kernel_kern:n { -#1 }
1705 }

```

And now we redefine the destination command:

```

1706 \cs_set_protected:Npn \_pdf_backend_indexed_structure_destination:nnnn #1#2#3#4
1707 {
1708     \exp_args:Ne \_pdf_backend_indexed_structure_destination_aux:nnnn
1709     { \dim_eval:n {#2} } {#1} {#3} {#4}
1710 }
1711 </xdvipdfmx | dvipdfmx>

```

Now pdftex. We only redefine for version 1.40 revision 24 or later.

```

1712 <*pdftex>
1713 \bool_lazy_and:nnT
1714 { \int_compare_p:nNn {\tex_pdftexversion:D } > {139} }
1715 { \int_compare_p:nNn {\tex_pdftexrevision:D } > {23} }
1716 {
1717     \cs_set_protected:Npn \_pdf_backend_indexed_structure_destination:nn #1#2
1718     {
1719         \tex_pdfdest:D
1720         name {#1}
1721         \str_case:nnF {#2}
1722         {
1723             { xyz } { xyz }
1724             { fit } { fit }
1725             { fitb } { fitb }
1726             { fitbh } { fitbh }
1727             { fitbv } { fitbv }
1728             { fith } { fith }
1729             { fitv } { fitv }
1730             { fitr } { fitr }
1731         }
1732         { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1733     }
1734     \scan_stop:
1735     \tex_pdfdest:D
1736     struct~
1737     \exp_after:wN \_kernel_pdf_object_id_indexed:nn \l_pdf_current_structure_des
1738     name {#1}
1739     \str_case:nnF {#2}
1740     {
1741         { xyz } { xyz }

```

```

1741         { fit } { fit }
1742         { fitb } { fitb }
1743         { fitbh } { fitbh }
1744         { fitbv } { fitbv }
1745         { fith } { fith }
1746         { fitv } { fitv }
1747         { fitr } { fitr }
1748     }
1749     { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1750     \scan_stop:
1751 }
1752 \cs_set_protected:Npn \__pdf_backend_indexed_structure_destination:nnnn #1#2#3#4
1753 {
1754     \tex_pdfdest:D
1755     name {#1}
1756     fitr ~
1757     width \dim_eval:n {#2} ~
1758     height \dim_eval:n {#3} ~
1759     depth \dim_eval:n {#4} \scan_stop:
1760     \tex_pdfdest:D
1761     struct~
1762     \exp_after:wN \__kernel_pdf_object_id_indexed:nn \l_pdf_current_structure_destination:nn
1763     name {#1}
1764     fitr ~
1765     width \dim_eval:n {#2} ~
1766     height \dim_eval:n {#3} ~
1767     depth \dim_eval:n {#4} \scan_stop:
1768 }
1769 }
1770 </pdfTeX>

```

luatex is quite similar to pdftex. Mostly the test for the version is different

```

1771 <*luatex>
1772 \int_compare:nNnT {\directlua{tex.print(status.list()["development_id"])} } > {7468}
1773 {
1774     \cs_set_protected:Npn \__pdf_backend_indexed_structure_destination:nn #1#2
1775     {
1776         \tex_pdfextension:D dest
1777         name {#1}
1778         \str_case:nnF {#2}
1779         {
1780             { xyz } { xyz }
1781             { fit } { fit }
1782             { fitb } { fitb }
1783             { fitbh } { fitbh }
1784             { fitbv } { fitbv }
1785             { fith } { fith }
1786             { fitv } { fitv }
1787             { fitr } { fitr }
1788         }
1789         { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1790         \scan_stop:
1791         \tex_pdfextension:D dest
1792         struct~
1793         \exp_after:wN \__kernel_pdf_object_id_indexed:nn \l_pdf_current_structure_destination:nn

```

```

1794     name {#1}
1795     \str_case:nnF {#2}
1796     {
1797         { xyz } { xyz }
1798         { fit } { fit }
1799         { fitb } { fitb }
1800         { fitbh } { fitbh }
1801         { fitbv } { fitbv }
1802         { fith } { fith }
1803         { fitv } { fitv }
1804         { fitr } { fitr }
1805     }
1806     { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1807     \scan_stop:
1808 }
1809 \cs_set_protected:Npn \__pdf_backend_indexed_structure_destination:nnnn #1#2#3#4
1810 {
1811     \tex_pdfextension:D dest
1812     name {#1}
1813     fitr ~
1814     width \dim_eval:n {#2} ~
1815     height \dim_eval:n {#3} ~
1816     depth \dim_eval:n {#4} \scan_stop:
1817     \tex_pdfextension:D dest
1818     struct~
1819     \exp_after:wN \__kernel_pdf_object_id_indexed:nn \l_pdf_current_structure_destinati
1820     name {#1}
1821     fitr ~
1822     width \dim_eval:n {#2} ~
1823     height \dim_eval:n {#3} ~
1824     depth \dim_eval:n {#4} \scan_stop:
1825 }
1826 \cs_set_protected:Npn \__pdfannot_backend_link_begin_structure_goto:nnw #1#2
1827 {
1828     \__pdfannot_backend_link_begin:nnnw {#1} { goto~struct~name~{#2}~name } {#2}
1829 }
1830 }
1831 </luatex>

```

(End of definition for `\__pdf_backend_indexed_structure_destination:nn` and `\__pdf_backend_indexed_structure_destination:nnnn`.)

## 1.12 Settings for regression tests

When doing pdf based regression tests some meta data in the pdf should have fixed values to get identical pdf's. We define here the backend dependent part. The main command is then in `l3pdfmeta`

```

1832 <*drivers>
1833 \cs_new_protected:Npn \__pdf_backend_set_regression_data:
1834 {
1835     \sys_gset_rand_seed:n{1000}
1836     \pdfmanagement_add:nnn{Info}{Creator}{(TeX)}
1837 </drivers>
1838 <*dvips>

```



```

1839 \AddToHook{begindocument}{\pdfmanagement_add:nnn{Info}{Producer}{(pdfTeX+dvips)}}
1840 \__kernel_backend_literal:e{!~<</DocumentUUID~(DocumentUUID)>>~setpagedevice}
1841 \__kernel_backend_literal:e{!~<</InstanceUUID~(InstanceUUID)>>~setpagedevice}
1842 \pdfmanagement_add:nne{Info}{CreationDate}{(\c_sys_timestamp_str)}
1843 \pdfmanagement_add:nne{Info}{ModDate}{(\c_sys_timestamp_str)}
1844 </dvips>
1845 <*dvipdfmx>
1846 \pdfmanagement_add:nnn{Info}{Producer}{(dvipdfmx)}
1847 \__kernel_backend_literal:e
1848 {pdf:trailerid [~
1849 <00112233445566778899aabbccddeeff>~
1850 <00112233445566778899aabbccddeeff>~
1851 ]}
1852 </dvipdfmx>
1853 <*xdvipdfmx>
1854 \pdfmanagement_add:nnn{Info}{Producer}{(xetex)}
1855 \__kernel_backend_literal:e
1856 {pdf:trailerid [~
1857 <00112233445566778899aabbccddeeff>~
1858 <00112233445566778899aabbccddeeff>~
1859 ]}
1860 </xdvipdfmx>
1861 <*pdftex>
1862 \pdfmanagement_add:nnn{Info}{Producer}{(pdfTeX)}
1863 \tex_pdfsuppressptexinfo:D 7 \scan_stop:
1864 \pdftrailerid{2350CAD05F8A7AF0AA4058486855344F}
1865 </pdftex>
1866 <*luatex>
1867 \pdfmanagement_add:nnn{Info}{Producer}{(LuaTeX)}
1868 \tex_pdfvariable:D suppressoptionalinfo 7\relax
1869 \tex_pdfvariable:D trailerid
1870 {[~
1871 <2350CAD05F8A7AF0AA4058486855344F>~
1872 <2350CAD05F8A7AF0AA4058486855344F>~
1873 ]}
1874 </luatex>

```

Embedded files should also have a fix date.

```

1875 <*drivers>
1876 \pdfdict_put:nne {l_pdffile/Params} {ModDate}{(\c_sys_timestamp_str)}
1877 \AddToDocumentProperties[hyperref]{pdfinstanceid}{uuid:0a57c455-157a-4141-8c19-6237d832f}
1878 \AddToDocumentProperties[hyperref]{pdfproducer}{\c_sys_engine_exec_str-NN.NN.NN}
1879 }
1880 </drivers>

```

### 1.13 Uncompressed metadata object stream

The xmp metadata should be written “uncompressed” to pdf. It is not quite clear what exactly that means. Probably it only means that there should be no `/Filter` key in the stream, but packages like `pdfx` and `hyperref` try to suppress object compression too, so we add support for it too. With `luatex` this is possible by using the `uncompressed` key word. With `pdftex` one can change locally the `compresslevel`. `(x)dvipdfmx` does it automatically and doesn’t need some special command. No solution is known for the `dvips` route. We

need it only once, so we make it special and probably no public interface is needed. It writes an unnamed object so should be referenced directly with `\pdf_object_ref_last`:

```

1881 <*luatex>
1882 \cs_new_protected:Npn \__pdf_backend_metadata_stream:n #1
1883 {
1884   \tex_immediate:D \tex_pdfextension:D obj ~uncompressed-
1885   \__pdf_backend_object_write:nn {stream} {{/Type~/Metadata~/Subtype~/XML}{#1}}
1886 }
1887 </luatex>
1888 <*pdftex>
1889 \cs_new_protected:Npn \__pdf_backend_metadata_stream:n #1
1890 {
1891   \group_begin:
1892   \tex_pdfcompresslevel:D 0 \scan_stop:
1893   \tex_immediate:D \tex_pdfobj:D
1894   \__pdf_backend_object_write:nn {stream} {{/Type~/Metadata~/Subtype~/XML}{#1}}
1895   \group_end:
1896 }
1897 </pdftex>
1898 <*xdvipdfmx | dvipdfmx | dvips | dvisvgm>
1899 \cs_new_protected:Npn \__pdf_backend_metadata_stream:n #1
1900 {
1901   \pdf_object_unnamed_write:nn {stream}{{/Type~/Metadata~/Subtype~/XML}{#1}}
1902 }
1903 </xdvipdfmx | dvipdfmx | dvips | dvisvgm>

```

## 1.14 Suppressing deprecated PDF features

`/ProcSet`, `/CharSet` and the `/Info` dictionary are deprecated in PDF 2.0. For the pdf/A-4 standard they must be suppressed. Not every engine is able to do this, but for pdfTeX and luatex we define suitable backend command. `/ProcSet` is suppressed automatically for pdf version 2.0 starting with in texlive 2023.

`\__pdf_backend_omit_charset:n` The option to omit `/CharSet` exists already for quite some time for the two engines.

```

1904 <*xdvipdfmx | dvipdfmx | dvips | dvisvgm>
1905 \cs_new_protected:Npn \__pdf_backend_omit_charset:n #1 {} % #1 number
1906 </xdvipdfmx | dvipdfmx | dvips | dvisvgm>
1907 <*pdftex>
1908 \cs_new_protected:Npn \__pdf_backend_omit_charset:n #1 % #1 number
1909 {
1910   \tex_pdfomitcharset:D = #1 \scan_stop:
1911 }
1912 </pdftex>
1913 <*luatex>
1914 \cs_new_protected:Npn \__pdf_backend_omit_charset:n #1 % #1 number
1915 {
1916   \tex_pdfvariable:D omitcharset = #1 \scan_stop:
1917 }
1918 </luatex>

```

*(End of definition for `\__pdf_backend_omit_charset:n`.)*

`\__pdf_backend_omit_info:n` The option to suppress the info dictionary will be available in texlive 2023.

```

1919 <*xdvipdfmx | dvipdfmx | dvips | dvisvgm>

```

```

1920 \cs_new_protected:Npn \__pdf_backend_omit_info:n #1 {} % #1 number
1921 </xdvipdfmx | dvipdfmx | dvips | dvisvgm>
1922 <*pdfTeX>
1923 \bool_lazy_and:nnTF
1924 { \int_compare_p:nNn {\tex_pdftexversion:D } > {139} }
1925 { \int_compare_p:nNn {\tex_pdftexrevision:D } > {24} }
1926 {
1927   \cs_new_protected:Npn \__pdf_backend_omit_info:n #1 % #1 number
1928   {
1929     \pdfomitinfodict = #1 \scan_stop:
1930   }
1931 }
1932 {
1933   \cs_new_protected:Npn \__pdf_backend_omit_info:n #1 {} % #1 number
1934 }
1935 }
1936 </pdfTeX>
1937 <*luatex>
1938 \int_compare:nNnTF {\directlua{tex.print(status.list()["development_id"])} } > {7560}
1939 {
1940   \cs_new_protected:Npn \__pdf_backend_omit_info:n #1 % #1 number
1941   {
1942     \tex_pdfvariable:D omitinfodict = #1 \scan_stop:
1943   }
1944 }
1945 {
1946   \cs_new_protected:Npn \__pdf_backend_omit_info:n #1 {} % #1 number
1947 }
1948 </luatex>

```

*(End of definition for \\_\_pdf\_backend\_omit\_info:n.)*

With luatex it is for some standards also necessary to suppress the CidSet entry in the fonts (with xetex there seem to be no problem).

\\_\_pdf\_backend\_omit\_cidset:n The option to omit /CharSet exists already for quite some time for the two engines.

```

1949 <*xdvipdfmx | dvipdfmx | dvips | dvisvgm | pdfTeX>
1950 \cs_new_protected:Npn \__pdf_backend_omit_cidset:n #1 {} % #1 number
1951 </xdvipdfmx | dvipdfmx | dvips | dvisvgm | pdfTeX>
1952 <*luatex>
1953 \cs_new_protected:Npn \__pdf_backend_omit_cidset:n #1 % #1 number
1954 {
1955   \tex_pdfvariable:D omitcidset = #1 \scan_stop:
1956 }
1957 </luatex>

```

*(End of definition for \\_\_pdf\_backend\_omit\_cidset:n.)*

## 1.15 lua code for lualatex

```

1958 <*lua>
1959 ltx= ltx or {}
1960 ltx.__pdf = ltx.__pdf or {}
1961 ltx.__pdf.Page = ltx.__pdf.Page or {}
1962 ltx.__pdf.Page.dflt = ltx.__pdf.Page.dflt or {}
1963 ltx.__pdf.Page.Resources = ltx.__pdf.Resources or {}

```

```

1964 ltx.__pdf.Page.Resources.Properties = ltx.__pdf.Page.Resources.Properties or {}
1965 ltx.__pdf.Page.Resources.List={"ExtGState","ColorSpace","Pattern","Shading"}
1966 ltx.__pdf.object = ltx.__pdf.object or {}
1967
1968 ltx.pdf= ltx.pdf or {} -- for "public" functions
1969
1970 local __pdf = ltx.__pdf
1971 local pdf = pdf
1972
1973 local function __pdf_backend_Page_gput (name,value)
1974 __pdf.Page.dflt[name]=value
1975 end
1976
1977 local function __pdf_backend_Page_gremove (name)
1978 __pdf.Page.dflt[name]=nil
1979 end
1980
1981 local function __pdf_backend_Page_gclear ()
1982 __pdf.Page.dflt={}
1983 end
1984
1985 local function __pdf_backend_ThisPage_gput (page,name,value)
1986 __pdf.Page[page] = __pdf.Page[page] or {}
1987 __pdf.Page[page][name]=value
1988 end
1989
1990 local function __pdf_backend_ThisPage_gpush (page)
1991 local token=""
1992 local t = {}
1993 local tkeys= {}
1994 for name,value in pairs(__pdf.Page.dflt) do
1995 t[name]=value
1996 end
1997 if __pdf.Page[page] then
1998 for name,value in pairs(__pdf.Page[page]) do
1999 t[name] = value
2000 end
2001 end
2002 -- sort the table to get reliable test files.
2003 for name,value in pairs(t) do
2004 table.insert(tkeys,name)
2005 end
2006 table.sort(tkeys)
2007 for _,name in ipairs(tkeys) do
2008 token = token .. "/"..name.." "..t[name]
2009 end
2010 return token
2011 end
2012
2013 function ltx.__pdf.backend_ThisPage_gput (page,name,value) -- tex.count["g_shipout_readonly_
2014 __pdf_backend_ThisPage_gput (page,name,value)
2015 end
2016
2017 function ltx.__pdf.backend_ThisPage_gpush (page)

```

```

2018 pdf.setpageattributes(__pdf_backend_ThisPage_gpush (page))
2019 end
2020
2021 function ltx.__pdf.backend_Page_gput (name,value)
2022 __pdf_backend_Page_gput (name,value)
2023 end
2024
2025 function ltx.__pdf.backend_Page_gremove (name)
2026 __pdf_backend_Page_gremove (name)
2027 end
2028
2029 function ltx.__pdf.backend_Page_gclear ()
2030 __pdf_backend_Page_gclear ()
2031 end
2032
2033
2034 local Properties = ltx.__pdf.Page.Resources.Properties
2035 local ResourceList= ltx.__pdf.Page.Resources.List
2036 local function __pdf_backend_PageResources_gpush (page)
2037 local token=""
2038 if Properties[page] then
2039 -- we sort the table, so that the pdf test works
2040 local t = {}
2041 for name,value in pairs (Properties[page]) do
2042 table.insert (t,name)
2043 end
2044 table.sort (t)
2045 for _,name in ipairs(t) do
2046 token = token .. "/"..name.." ".. Properties[page][name]
2047 end
2048 token = "/Properties <<"..token..>>"
2049 end
2050 for i,name in ipairs(ResourceList) do
2051 if ltx.__pdf.Page.Resources[name] then
2052 token = token .. "/"..name.." "..ltx.pdf.object_ref("__pdf/Page/Resources/"..name)
2053 end
2054 end
2055 return token
2056 end
2057
2058 -- the function is public, as I probably need it in tagpdf too ...
2059 function ltx.pdf.Page_Resources_Properties_gput (page,name,value) -- tex.count["g_shipout_re
2060 Properties[page] = Properties[page] or {}
2061 Properties[page][name]=value
2062 pdf.setpageresources(__pdf_backend_PageResources_gpush (page))
2063 end
2064
2065 function ltx.pdf.Page_Resources_gpush(page)
2066 pdf.setpageresources(__pdf_backend_PageResources_gpush (page))
2067 end
2068
2069 function ltx.pdf.object_ref (objname)
2070 if ltx.__pdf.object[objname] then
2071 local ref= ltx.__pdf.object[objname]

```

```

2072   return ref
2073   else
2074   return "false"
2075   end
2076 end
2077 </lua>

```

## 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>\AddToDocumentProperties</code> . . . . .	1877, 1878
<code>\AddToHook</code> . . . . .	1839
<b>B</b>	
bool commands:	
<code>\bool_if:NTF</code> . . . . .	39, 533, 569, 650, 656, 692, 716, 751, 775, 805, 844
<code>\bool_lazy_and:nnTF</code> . . . . .	1482, 1713, 1923
<code>\bool_new:N</code> . . . . .	522
<code>\bool_set_true:N</code> 962, 1044, 1137, 1238	
box commands:	
<code>\box_dp:N</code> . . . . .	975, 1056, 1148, 1251, 1265
<code>\box_ht:N</code> . . . . .	972, 1053, 1145, 1248, 1260
<code>\box_new:N</code> . . . . .	88, 89, 1134
<code>\box_scale:Nnn</code> . . . . .	1269
<code>\box_set_dp:Nn</code> 1149, 1215, 1312, 1336	
<code>\box_set_ht:Nn</code> 1150, 1214, 1313, 1335	
<code>\box_set_wd:Nn</code> 1151, 1213, 1314, 1334	
<code>\box_use:N</code> . . . . .	1318
<code>\box_use_drop:N</code> 1162, 1216, 1293, 1337	
<code>\box_wd:N</code> . . . . .	969, 1050, 1142, 1245, 1255
<b>C</b>	
clist commands:	
<code>\clist_const:Nn</code> . . . . .	420
<code>\clist_map_function:NN</code> . . . . .	882
<code>\clist_map_inline:Nn</code> 429, 463, 479, 672	
cs commands:	
<code>\cs_generate_variant:Nn</code> . . . . .	28, 31, 32, 35, 36, 79, 80, 417
<code>\cs_gset_eq:NN</code> . . . . .	651, 652, 1362, 1363, 1364, 1368, 1369, 1370
<code>\cs_if_exist:NTF</code> . . . . .	432, 941
<code>\cs_new:Npn</code> . . . . .	74, 100, 106, 248, 858, 1028, 1110, 1199, 1223, 1339
<code>\cs_new_protected:Npn</code> . . . . .	41, 45, 55, 68, 150, 159, 175, 181, 187, 194, 201, 210, 230, 253, 263,
277, 289, 306, 317, 324, 331, 340, 349, 356, 363, 370, 379, 388, 396, 399, 405, 410, 413, 444, 455, 461, 487, 491, 503, 506, 507, 511, 514, 515, 519, 535, 558, 582, 670, 765, 867, 891, 898, 905, 914, 918, 921, 924, 936, 954, 1021, 1036, 1101, 1124, 1204, 1221, 1222, 1229, 1323, 1360, 1366, 1425, 1663, 1833, 1882, 1889, 1899, 1905, 1908, 1914, 1920, 1927, 1933, 1940, 1946, 1950, 1953	
<code>\cs_new_protected:Npx</code> . . . . .	169
<code>\cs_set_eq:NN</code> . . . . .	530, 531, 566, 567, 662, 749, 757, 842, 850, 1374, 1375, 1376, 1377, 1378
<code>\cs_set_protected:Npn</code> . . . . .	526, 542, 546, 550, 554, 564, 571, 574, 576, 578, 580, 593, 612, 631, 637, 643, 648, 658, 664, 687, 711, 735, 739, 744, 753, 760, 768, 798, 829, 833, 838, 846, 853, 943, 947, 1381, 1471, 1476, 1486, 1525, 1546, 1555, 1594, 1615, 1622, 1706, 1717, 1752, 1774, 1809, 1826
<b>D</b>	
dim commands:	
<code>\dim_eval:n</code> . . . . .	1474, 1530, 1531, 1532, 1541, 1542, 1543, 1599, 1600, 1601, 1610, 1611, 1612, 1709, 1757, 1758, 1759, 1765, 1766, 1767, 1814, 1815, 1816, 1822, 1823, 1824
<code>\dim_to_decimal_in_sp:n</code> . . . . .	1255, 1260, 1265
<code>\c_zero_dim</code> . . . . .	1149, 1150, 1151, 1312, 1313, 1314
<code>\directlua</code> . . . . .	97, 1553, 1772, 1938

**E**

exp commands:

- `\exp_after:wN` ..... 1647, 1695, 1736, 1762, 1793, 1819
- `\exp_args:Ne` . 700, 724, 1402, 1408, 1453, 1459, 1473, 1503, 1508, 1533, 1538, 1572, 1577, 1602, 1607, 1708
- `\exp_args:NNe` ..... 869
- `\exp_not:n` ..... 617, 715, 802, 1385, 1406, 1457, 1626, 1645, 1693

**F**

fp commands:

- `\fp_eval:n` ..... 1399, 1420, 1501, 1521, 1570, 1590, 1640, 1659, 1732, 1749, 1789, 1806

**G**

group commands:

- `\group_begin:` ..... 1891
- `\group_end:` ..... 1895

**H**

hbox commands:

- `\hbox:n` ..... 1430, 1441, 1668, 1679
- `\hbox_gset:Nn` ..... 1135
- `\hbox_set:Nn` ..... 960, 1042, 1206, 1236, 1270, 1325

hook commands:

- `\hook_gput_code:nnn` .. 142, 482, 1315
- `\hook_gput_next_code:nn` ..... 1152
- `\hook_gset_rule:nnnn` ..... 476, 477

**I**

int commands:

- `\int_compare:nNnTF` ..... 983, 1064, 1553, 1772, 1938
- `\int_compare_p:nNn` ..... 1483, 1484, 1714, 1715, 1924, 1925
- `\int_const:Nn` . 1016, 1096, 1131, 1232
- `\int_gincr:N` ..... 213, 595, 614, 689, 713, 770, 774, 800, 804, 1130, 1231
- `\int_if_exist:NTF` ..... 1349
- `\int_new:N` ..... 92, 93, 94
- `\int_use:N` ..... 214, 217, 598, 606, 617, 625, 691, 696, 705, 715, 720, 729, 772, 779, 783, 786, 794, 802, 809, 813, 816, 824, 1024, 1030, 1103, 1111, 1201, 1279, 1306, 1330, 1341, 1507, 1537, 1576, 1606

**K**

kernel internal commands:

- `\__kernel_backend_literal:n` .... 31, 83, 596, 600, 615, 619, 633, 645, 666, 676, 1840, 1841, 1847, 1855
- `\__kernel_backend_literal_page:n` ..... 28, 690, 714, 737, 746, 762, 771, 801, 831, 840, 855
- `\__kernel_backend_postscript:n` .. 35, 1272, 1294, 1300, 1327
- `\__kernel_backend_shipout_literal:n` ..... 39, 41, 537, 660
- `\__kernel_backend_shipout_literal_page:n` ... 55, 55, 755, 848
- `\__kernel_backend_shipout_literal_pdf:n` ..... 45, 45
- `\__kernel_kern:n` ..... 1429, 1437, 1440, 1469, 1667, 1675, 1678, 1704
- `\__kernel_pdf_name_from_unicode_e:n` ..... 100, 106
- `\__kernel_pdf_object_id_indexed:nn` ..... 1736, 1762, 1793, 1819
- `\__kernel_pdffdict_name:n` ..... 232, 233, 235, 466, 494, 674, 861, 872, 877, 963, 984, 995, 1000, 1005, 1010, 1045, 1065, 1075, 1080, 1085, 1090, 1239
- `\g__kernel_pdfmanagement_end_run_code_tl` ..... 115, 122, 129
- `\g__kernel_pdfmanagement_thispage_shipout_code_tl` 138, 144

**L**

lualua commands:

- `\lualua:` ..... 207, 286, 337, 376

**M**

mode commands:

- `\mode_leave_vertical:` ... 1154, 1317

**P**

pdf commands:

- `\pdf_activate_indexed_structure_destination:` ..... 1359, 1366
- `\pdf_activate_structure_destination:` ..... 1359, 1360
- `\l_pdf_current_structure_destination_tl` ..... 1356, 1402, 1408, 1453, 1459, 1503, 1508, 1533, 1538, 1572, 1577, 1602, 1607, 1647, 1695, 1736, 1762, 1793, 1819
- `\pdf_object_if_exist:nTF` ..... 1402, 1453, 1503, 1533, 1572, 1602
- `\pdf_object_new:n` ..... 431, 481
- `\pdf_object_ref:n` ..... 438, 499, 544, 607, 679, 697, 706, 780, 795, 863, 997, 1002, 1007, 1012, 1077, 1082, 1087, 1092, 1168, 1175, 1182, 1190, 1408, 1459

```

\pdf_object_ref_indexed:nn 1647, 1695
\pdf_object_ref_last: . 894, 901, 908
\pdf_object_unnamed_write:nn . . .
. . . 639, 741, 835, 893, 900, 907, 1901
\pdf_object_write . . . . . 496
\pdf_object_write:nnn . . . . . 468, 485
pdf internal commands:
\_pdf_backend:n . . . . .
. . . . . 32, 177, 489, 497, 908, 1155, 1163,
. . . . . 1164, 1171, 1178, 1185, 1193, 1208,
. . . . . 1383, 1404, 1432, 1433, 1443, 1455,
. . . . . 1624, 1643, 1670, 1671, 1681, 1691
\_pdf_backend_bdc:nn . . . . .
. . . . . 13, 521, 526, 530, 531, 564,
. . . . . 566, 567, 648, 651, 652, 653, 749, 842
\_pdf_backend_bdc_contobj:nn . . .
. . . . . 530, 566, 637, 651, 739, 833
\_pdf_backend_bdc_contstream:nn
. . . . . 531, 567, 643, 652, 744, 749, 838, 842
\_pdf_backend_bdc_shipout:nn . . .
. . . . . 535, 662, 757, 850
\_pdf_backend_bdc_shipout_-
contstream:nn . . . . .
. . . . . 658, 662, 753, 757, 846, 850
\_pdf_backend_bdcobject:n . . . . .
. . . . . 13, 521,
. . . . . 546, 576, 612, 640, 711, 742, 798, 836
\_pdf_backend_bdcobject:nn . . . . .
. . . . . 13, 521, 542, 574, 593, 687, 768
\_pdf_backend_bmc:n . . . . .
. . . . . 13, 521, 554, 580, 631, 735, 829
\_pdf_backend_catalog_gput:nn . . 20
\_pdf_backend_destination:nn . . .
. . . . . 1362, 1368, 1374, 1377
\_pdf_backend_destination:nnnn .
. . . . . 1363, 1369, 1375, 1378
\_pdf_backend_emc: . . . . .
. . . . . 13, 521, 550, 578, 664, 760, 853
\_pdf_backend_indexed_structure_-
destination:nn . . . . .
. . . . . 1368, 1377, 1621, 1622, 1717, 1774
\_pdf_backend_indexed_structure_-
destination:nnnn . . . . .
. . . . . 1369, 1378, 1621, 1706, 1752, 1809
\_pdf_backend_indexed_structure_-
destination_aux:nnnn . . 1663, 1708
\_pdf_backend_luastring:n . . . . .
. . . . . 163, 248, 257, 269, 270, 281, 296, 297
\_pdf_backend_metadata_stream:n
. . . . . 1882, 1889, 1899
\g_pdf_backend_name_int . . . . .
. . . . . 91, 595, 598, 606,
. . . . . 614, 617, 625, 689, 691, 696, 705,
. . . . . 713, 715, 720, 729, 770, 772, 800, 802
\_pdf_backend_Names_gpush:nn . . .
. . . . . 891, 898, 905, 914, 918
\_pdf_backend_NamesEmbeddedFiles_-
add:nn . . . . . 920, 921, 924, 936
\g_pdf_backend_object_int . . . . .
. . . . . 1130, 1133, 1231, 1234, 1279
\_pdf_backend_object_last: . . . . .
. . . . . 548, 626, 721, 730, 810, 825
\_pdf_backend_object_write:nn . .
. . . . . 1885, 1894
\_pdf_backend_omit_charset:n . . .
. . . . . 1904, 1905, 1908, 1914
\_pdf_backend_omit_cidset:n . . .
. . . . . 1949, 1950, 1953
\_pdf_backend_omit_info:n . . . . .
. . . . . 1919, 1920, 1927, 1933, 1940, 1946
\_pdf_backend_Page_gput:nn . . . . .
. . . . . 6, 184, 194, 263, 324, 363, 399
\_pdf_backend_Page_gremove:n . . .
. . . . . 6, 184, 201, 277, 331, 370, 405
\g_pdf_backend_page_int . . . . . 91
\_pdf_backend_Page_primitive:n .
. . . . . 6, 184, 187, 240, 253,
. . . . . 317, 342, 351, 356, 381, 390, 396, 417
\_pdf_backend_PageResources:n . .
. . . . . 487, 506, 514
\c_pdf_backend_PageResources_-
clist . . 419, 429, 463, 479, 672, 883
\_pdf_backend_PageResources_-
gpush:n . . . . .
. . . . . 13, 521, 558, 582, 670, 765, 867
\_pdf_backend_PageResources_-
gpush_aux:n . . . . . 858, 884
\_pdf_backend_PageResources_-
gput:nnn 428, 444, 455, 491, 507, 515
\_pdf_backend_PageResources_-
obj_gpush: . 428, 461, 503, 511, 519
\_pdf_backend_Pages_primitive:n
. . . . . 149, 150, 159, 169, 175, 181
\_pdf_backend_pdfmark:n . . . . .
. . . . . 36, 528, 544, 548, 552, 556, 926
\_pdf_backend_record_abspage:n .
. . . . . 68, 79, 214, 783, 813
\_pdf_backend_ref_abspage:n . . .
. . . . . 74, 80, 217, 786, 816
\g_pdf_backend_resourceid_int . .
. . . . . 91, 213, 214, 217, 774, 779,
. . . . . 783, 786, 794, 804, 809, 813, 816, 824
\_pdf_backend_set_regression_-
data: . . . . . 1833
\_pdf_backend_shipout_bdc:nn . . .
. . . . . 13, 521, 571

```



\_pdf_backend_structure_- destination:nn .. 1362, 1374, 1380, 1381, 1486, 1555	\_pdfannot_backend_link_off: .. 943
\_pdf_backend_structure_- destination:nnnn .. 1363, 1375, 1380, 1471, 1525, 1594	\_pdfannot_backend_link_on: .. 947
\_pdf_backend_structure_- destination_aux:nnnn .. 1425, 1473	pdfdict commands:
\_pdf_backend_ThisPage_gpush:n .. 6, 184, 230, 306, 349, 388, 413	\pdfdict_gput:nnn .. 196, 224, 326, 365, 401, 446, 457, 509, 517, 694, 718, 777, 792, 807, 822
\_pdf_backend_ThisPage_gput:nn .. 6, 184, 210, 289, 340, 379, 410	\pdfdict_gremove:nn 203, 333, 372, 407
\g_pdf_backend_thispage_- shipout_tl .. 6	\pdfdict_if_exist:nTF .. 219, 788, 818
\l_pdf_backend_tmpa_box .. 85, 960, 969, 972, 975, 1015, 1042, 1050, 1053, 1056, 1095, 1206, 1213, 1214, 1215, 1216, 1236, 1245, 1248, 1251, 1255, 1260, 1265, 1269, 1293, 1325, 1334, 1335, 1336, 1337	\pdfdict_item:nn .. 242, 863, 878
\l_pdf_backend_tmpb_box .. 89, 1270, 1312, 1313, 1314, 1318	\pdfdict_new:n .. 221, 790, 820
\l_pdf_backend_xform_bool .. 522, 692, 716, 775, 805, 962, 1044, 1137, 1238	\pdfdict_put:nnn .. 1876
\_pdf_backend_xform_if_exist:n .. 1347, 1353	\pdfdict_show:n .. 826
\_pdf_backend_xform_new:nnnn .. 953, 954, 1036, 1124, 1221, 1229	\pdfdict_use:n 352, 391, 470, 990, 1071
\_pdf_backend_xform_ref:n .. 953, 1028, 1110, 1157, 1199, 1210, 1223, 1339	\pdfliteral .. 2
\l_pdf_backend_xform_tmpdp_tl .. 1227, 1263, 1277, 1284	pdfmanagement commands:
\l_pdf_backend_xform_tmpht_tl .. 1228, 1258, 1282	\pdfmanagement_add:nnn 1836, 1839, 1842, 1843, 1846, 1854, 1862, 1867
\l_pdf_backend_xform_tmpwd_tl .. 1226, 1253, 1283	pdfmanagement internal commands:
\_pdf_backend_xform_use:n .. 953, 1021, 1101, 1204, 1222, 1323	\g_pdfmanagement_active_bool .. 650
\g_pdf_tmpa_prop .. 85, 232, 237, 242	\l_pdfmanagement_delayed_- shipout_bool .. 39, 533, 569, 656, 751, 844
\l_pdf_tmpa_tl .. 85, 215, 219, 221, 224, 784, 788, 790, 793, 814, 818, 820, 823, 826	\pdfnames .. 20
pdfannot internal commands:	\pdfomitinfodict .. 1929
\_pdfannot_backend_link_begin:n .. 1478	\pdfpageref .. 3
\_pdfannot_backend_link_- begin:nnnw .. 1548, 1617, 1828	\pdfrunninglinkoff .. 941, 945
\_pdfannot_backend_link_begin_- goto:nnw .. 1364, 1370, 1376	\pdfrunninglinkon .. 949
\_pdfannot_backend_link_begin_- structure_goto:nnw 1364, 1370, 1376, 1380, 1476, 1546, 1615, 1826	\pdftrailerid .. 1864
	pdfxform commands:
	\pdfxform_dp:n .. 1160, 1215, 1336
	\pdfxform_ht:n .. 1159, 1214, 1335
	\pdfxform_if_exist:n .. 1353
	\pdfxform_wd:n .. 1158, 1213, 1334
	prg commands:
	\prg_new_conditional:Npnn .. 1347
	\prg_new_eq_conditional:NNn .. 1353
	\prg_return_false: .. 1351
	\prg_return_true: .. 1350
	prop commands:
	\prop_count:N .. 984, 1065
	\prop_gclear:N .. 963, 1045, 1239
	\prop_gput:Nnn .. 237, 494
	\prop_gset_eq:NN .. 232
	\prop_if_empty:NTF .. 465, 674, 860, 994, 999, 1004, 1009, 1074, 1079, 1084, 1089
	\prop_if_exist:NTF .. 233, 871
	\prop_map_function:NN .. 242, 876
	\prop_map_inline:Nn .. 235
	\prop_new:N .. 86
	property commands:
	\property_record:nn .. 71
	\property_ref:nn .. 76
	\ProvidesExplFile .. 1

<b>R</b>	
<code>\relax</code> .....	135, 1868
<b>S</b>	
scan commands:	
<code>\scan_stop:</code> .....	1025,
1107, 1502, 1522, 1532, 1543, 1571,	
1591, 1601, 1612, 1733, 1750, 1759,	
1767, 1790, 1807, 1816, 1824, 1863,	
1892, 1910, 1916, 1929, 1942, 1955	
<code>\special</code> .....	2
str commands:	
<code>\str_case:nnTF</code> .....	
1388, 1409, 1490, 1510, 1559, 1579,	
1629, 1648, 1721, 1738, 1778, 1795	
<code>\str_convert_pdfname:n</code> .....	102, 495
<code>\str_if_eq:nnTF</code> .....	1287, 1298
sys commands:	
<code>\c_sys_engine_exec_str</code> .....	1878
<code>\sys_gset_rand_seed:n</code> .....	1835
<code>\sys_if_engine luatex:TF</code> .....	157
<code>\c_sys_timestamp_str</code> .....	1842, 1843, 1876
<b>T</b>	
TeX and L <sup>A</sup> T <sub>E</sub> X 2 <sub>ε</sub> commands:	
<code>\bsphack</code> .....	70
<code>\esphack</code> .....	72
<code>\@kernel@after@enddocument@afterlastpage</code> .....	112, 113
<code>\@kernel@after@shipout@background</code> .....	133, 136
<code>\@kernel@after@shipout@lastpage</code> . .....	119, 120, 126, 127
<code>\@kernel@before@shipout@background</code> .....	135
<code>\g@addto@macro</code> .....	135, 136
<code>\special</code> .....	2
tex commands:	
<code>\tex_directlua:D</code> .....	
.....	161, 265, 279, 432, 434, 447, 448
<code>\tex_global:D</code> .....	152, 189, 869
<code>\tex_immediate:D</code> .....	977, 1058, 1884, 1893
<code>\tex_latelua:D</code> .....	255, 291, 308, 700, 724
<code>\tex_luaescapestring:D</code> .....	250
<code>\tex_pdfcompresslevel:D</code> .....	1892
<code>\tex_pdfdest:D</code> .....	1488, 1505,
1527, 1535, 1719, 1734, 1754, 1760	
<code>\tex_pdfextension:D</code> .....	
.....	48, 58, 901, 1557, 1574, 1596,
1604, 1776, 1791, 1811, 1817, 1884	
<code>\tex_pdflastxform:D</code> .....	1018, 1098
<code>\tex_pdfliteral:D</code> .....	51, 61
<code>\tex_pdfnames:D</code> .....	894
<code>\tex_pdfobj:D</code> .....	1893
<code>\tex_pdfomitcharset:D</code> .....	1910
<code>\tex_pdfpageattr:D</code> .....	189
<code>\tex_pdfpageresources:D</code> .....	869
<code>\tex_pdfpagesattr:D</code> .....	152
<code>\tex_pdfrefxform:D</code> .....	1023, 1103
<code>\tex_pdfsuppressptexinfo:D</code> .....	1863
<code>\tex_pdftexrevision:D</code> .....	1484, 1715, 1925
<code>\tex_pdftexversion:D</code> .....	1483, 1714, 1924
<code>\tex_pdfvariable:D</code> .....	
.....	1868, 1869, 1916, 1942, 1955
<code>\tex_pdfxform:D</code> .....	977, 1058
<code>\tex_special:D</code> .....	42, 171, 319, 358
<code>\tex_the:D</code> .....	
.....	969, 972, 975, 1050, 1053, 1056,
1142, 1145, 1148, 1245, 1248, 1251	
<code>\tex_unexpanded:D</code> .....	250
<code>\tex_vss:D</code> .....	1435, 1467, 1673, 1702
text commands:	
<code>\text_expand:n</code> .....	102, 108
tl commands:	
<code>\c_space_tl</code> .....	598, 606, 617, 625, 691,
715, 772, 802, 1158, 1159, 1160, 1279	
<code>\tl_const:Nn</code> .....	
.....	967, 970, 973, 1048, 1051, 1054,
1140, 1143, 1146, 1243, 1246, 1249	
<code>\tl_gput_right:Nn</code> .....	113, 120, 127
<code>\tl_if_exist:NTF</code> .....	133
<code>\tl_new:N</code> .....	87, 1226, 1227, 1228, 1357
<code>\tl_set:Nn</code> .....	
.....	215, 784, 814, 1253, 1258, 1263
<code>\tl_to_str:n</code> .....	
.....	968, 971, 974, 1017, 1024,
1030, 1049, 1052, 1055, 1097, 1105,	
1111, 1132, 1141, 1144, 1147, 1201,	
1233, 1244, 1247, 1250, 1306, 1330,	
1341, 1349, 1508, 1538, 1577, 1607	
<code>\tl_use:N</code> .....	1277, 1282, 1283, 1284
<b>V</b>	
vbox commands:	
<code>\vbox_to_zero:n</code> .....	1427, 1438, 1665, 1676