

# 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 0.95a, released 2021-02-22

## 1 l3backend-testphase Implementation

```
1 <drivers>\ProvidesExplFile
2 <*dvipdfmx>
3   {l3backend-testphase-dvipdfmx.def}{2021-02-22}{ }
4   {LaTeX~PDF~management~testphase~bundle~backend~support: dvipdfmx}
5 </dvipdfmx>
6 <*dvips>
7   {l3backend-testphase-dvips.def}{2021-02-22}{ }
8   {LaTeX~PDF~management~testphase~bundle~backend~support: dvips}
9 </dvips>
10 <*dvisvgm>
11   {l3backend-testphase-dvisvgm.def}{2021-02-22}{ }
12   {LaTeX~PDF~management~testphase~bundle~backend~support: dvisvgm}
13 </dvisvgm>
14 <*luatex>
15   {l3backend-testphase-luatex.def}{2021-02-22}{ }
16   {LaTeX~PDF~management~testphase~bundle~backend~support: PDF output (LuaTeX)}
17 </luatex>
18 <*pdftex>
19   {l3backend-testphase-pdftex.def}{2021-02-22}{ }
20   {LaTeX~PDF~management~testphase~bundle~backend~support: PDF output (pdfTeX)}
21 </pdftex>
22 <*xdvipdfmx>
23   {l3backend-testphase-xetex.def}{2021-02-22}{ }
24   {LaTeX~PDF~management~testphase~bundle~backend~support: XeTeX}
25 </xdvipdfmx>
```

### 1.1 Crossreferences

This uses the temporary l3ref-tmp.sty. It will be replaced by kernel code later. It is only need to get a reference for the absolute page counter. This uses the counter from the new lthooks/lthshipout package.

```
26 <@@=pdf>
27 <*drivers>
```

---

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

```

28 \RequirePackage{l3ref-tmp}
29 \cs_generate_variant:Nn \ref_label:nn {en}
30 \cs_generate_variant:Nn \ref_value:nn {en}
31 \cs_new_protected:Npn \__pdf_backend_ref_label:nn #1 #2
32 {
33   \@bsphack
34   \ref_label:nn{#1}{abspage}
35   \@esphack
36 }
37 \cs_new:Npn \__pdf_backend_ref_value:nn #1 #2
38 {
39   \ref_value:nn{#1}{#2}
40 }
41 \cs_generate_variant:Nn \__pdf_backend_ref_label:nn {en}
42 \cs_generate_variant:Nn \__pdf_backend_ref_value:nn {en}
43 </drivers>
44 <*dvipdfmx | xdvipdfmx>
45 % avoid that destinations names are optimized.
46 % is this still needed??
47 % see https://tug.org/pipermail/dvipdfmx/2019-May/000002.html
48 \__kernel_backend_literal:x { dvipdfmx:config~C~ 0x0010 }
49 </dvipdfmx | xdvipdfmx>

```

```

\g__pdf_tmpa_prop Some scratch variables
\l__pdf_tmpa_tl
\l__pdf_backend_tmpa_box
50 <*drivers>
51 \prop_new:N \g__pdf_tmpa_prop
52 \tl_new:N \l__pdf_tmpa_tl
53 \box_new:N \l__pdf_backend_tmpa_box
54 </drivers>

```

(End definition for `\g__pdf_tmpa_prop`, `\l__pdf_tmpa_tl`, and `\l__pdf_backend_tmpa_box`.)

```

\g__pdf_backend_resourceid_int a counter to create labels for the resources, a counter to number properties in bdc marks,
\g__pdf_backend_name_int a counter for the \pdfpageref implementation.
\g__pdf_backend_page_int
55 <*drivers>
56 \int_new:N \g__pdf_backend_resourceid_int
57 \int_new:N \g__pdf_backend_name_int
58 \int_new:N \g__pdf_backend_page_int
59 </drivers>

```

(End definition for `\g__pdf_backend_resourceid_int`, `\g__pdf_backend_name_int`, and `\g__pdf_backend_page_int`.)

## 1.2 luacode

Load the lua code.

```

60 <*luatex>
61 \directlua { require("l3backend-testphase.lua") }
62 </luatex>

```

## 1.3 Hooks

### 1.3.1 Add the “end run” hooks

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

```
63 <*pdftex | luatex>
64 % put in \@kernel@after@enddocument@afterlastpage
65 \tl_gput_right:Nn \@kernel@after@enddocument@afterlastpage
66 {
67   \g__kernel_pdfmanagement_end_run_code_tl
68 }
69 </pdftex | luatex>
70 <*dvipdfmx | xdvipdfmx>
71 % put in \@kernel@after@shipout@lastpage
72 \tl_gput_right:Nn \@kernel@after@shipout@lastpage
73 {
74   \g__kernel_pdfmanagement_end_run_code_tl
75 }
76 </dvipdfmx | xdvipdfmx>
77 <*dvips>
78 % put in \@kernel@after@shipout@lastpage
79 \tl_gput_right:Nn \@kernel@after@shipout@lastpage
80 {
81   \g__kernel_pdfmanagement_end_run_code_tl
82 }
83 </dvips>
```

### 1.3.2 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 ...

```
84 <*drivers>
85 \tl_if_exist:NTF \@kernel@after@shipout@background
86 {
87   \g@addto@macro \@kernel@before@shipout@background{\relax}
88   \g@addto@macro \@kernel@after@shipout@background
89   {
90     \g__kernel_pdfmanagement_thispage_shipout_code_tl
91   }
92   \tl_gput_left:Nn \@kernel@after@shipout@lastpage
93   {
94     \g__kernel_pdfmanagement_lastpage_shipout_code_tl
95   }
96 }
97 {
98   \hook_gput_code:nnn{shipout/background}{pdf}
99   {
100     \g__kernel_pdfmanagement_thispage_shipout_code_tl
101   }
102   \hook_gput_code:nnn {shipout/lastpage} {pdf}
103   {
104     \g__kernel_pdfmanagement_lastpage_shipout_code_tl
105   }
}
```

```

106 }
107
108 </drivers>

```

## 1.4 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 dvipdfmx are additive. luatex sets it in lua. The higher level code has to take this into account.

```

109 <*pdftex>
110 \cs_new_protected:Npn \_pdf_backend_Pages_primitive:n #1
111 {
112   \tex_global:D \tex_pdfpagesattr:D { #1 }
113 }
114 </pdftex>
115 <*luatex>
116 %luatex: does it in lua
117 \sys_if_engine_luatex:T
118 {
119   \cs_new_protected:Npn \_pdf_backend_Pages_primitive:n #1
120   {
121     \tex_directlua:D
122     {
123       pdf.setpagesattributes( \_pdf_backend_luastring:n { #1 } )
124     }
125   }
126 }
127 </luatex>
128 <*dvips>
129 \cs_new_protected:Npx \_pdf_backend_Pages_primitive:n #1
130 {
131   \tex_special:D{ps:~[#1~/PAGES-pdfmark} %]
132 }
133 </dvips>
134 <*dvipdfmx | xdvipdfmx>
135 \cs_new_protected:Npn \_pdf_backend_Pages_primitive:n #1
136 {
137   \_pdf_backend:n{put~@pages~<<#1>>}
138 }
139 </dvipdfmx | xdvipdfmx>
140 <*dvisvgm>
141 \cs_new_protected:Npn \_pdf_backend_Pages_primitive:n #1
142 {}
143 </dvisvgm>

```

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

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

`\_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`

`\_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`.

```

144 % backend commands
145 <*pdfTeX>
146 %the primitive
147 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
148 {
149   \tex_global:D \tex_pdfpageattr:D { #1 }
150 }
151 % the command to store default values.
152 % Uses a prop with pdfLaTeX + dvi,
153 % sets a lua table with LuaLaTeX
154 \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2 %key,value
155 {
156   \pdfdict_gput:nnn {g__pdf_Core/Page}{ #1 }{ #2 }
157 }
158 % the command to remove a default value.
159 % Uses a prop with pdfLaTeX + dvi,
160 % changes a lua table with LuaLaTeX
161 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
162 {
163   \pdfdict_gremove:nn {g__pdf_Core/Page}{ #1 }
164 }
165 % the command used in the document.
166 % direct call of the primitive special with dvips/dvipdfmx
167 % \latelua: fill a page related table with LuaLaTeX, merge it with the page
168 % table and push it directly
169 % write to aux and store in prop with pdfLaTeX
170 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
171 {
172   %we need to know the page the resource should be added too.
173   \int_gincr:N\g__pdf_backend_resourceid_int
174   %\zref@labelbylist {l3pdf\int_use:N\g__pdf_backend_resourceid_int} {l3pdf}
175   %\ref_label:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
176   \__pdf_backend_ref_label:en { l3pdf\int_use:N\g__pdf_backend_resourceid_int }{abspage}
177   \tl_set:Nx \l__pdf_tmpa_tl
178   {
179     %\zref@extractdefault
180     {l3pdf\int_use:N\g__pdf_backend_resourceid_int}
181     {pdf@abspage}
182     {0}
183     \ref_value:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
184     \__pdf_backend_ref_value:en {l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
185   }
186   \pdfdict_if_exist:nF { g__pdf_Core/backend_Page\l__pdf_tmpa_tl}
187   {
188     \pdfdict_new:n { g__pdf_Core/backend_Page\l__pdf_tmpa_tl}
189   }
190   %backend_Page has no handler.
191   \pdfdict_gput:nnn {g__pdf_Core/backend_Page\l__pdf_tmpa_tl}{ #1 }{ #2 }

```

```

192 }
193 %the code to push the values, used in shipout
194 %merges the two props and then fills the register in pdflatex
195 %merges the two tables and then fills (in lua) in luatex
196 %issues the values stored in the global prop with dvi
197 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
198 {
199   \prop_gset_eq:Nc \g__pdf_tmpa_prop { \__kernel_pdftdict_name:n { g__pdf_Core/Page } }
200   \prop_if_exist:cT { \__kernel_pdftdict_name:n { g__pdf_Core/backend_Page#1 } }
201   {
202     \prop_map_inline:cn { \__kernel_pdftdict_name:n { g__pdf_Core/backend_Page#1 } }
203     {
204       \prop_gput:Nnn \g__pdf_tmpa_prop { ##1 }{ ##2 }
205     }
206   }
207   \exp_args:Nx \__pdf_backend_Page_primitive:n
208   {
209     \prop_map_function:NN \g__pdf_tmpa_prop \pdftdict_item:ne
210   }
211 }
212 </pdftex>
213 <*luatex>
214 % do we need to use some escaping for the values????
215 \cs_new:Npn \__pdf_backend_luastring:n #1
216 {
217   "\tex_luaescapestring:D { \tex_unexpanded:D { #1 } }"
218 }
219 %not used, only there for consistency
220 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
221 {
222   \tex_latelua:D
223   {
224     pdf.setpageattributes(\__pdf_backend_luastring:n { #1 })
225   }
226 }
227 % the command to store default values.
228 % Uses a prop with pdflatex + dvi,
229 % sets a lua table with luatex
230 \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2
231 {
232   \tex_directlua:D
233   {
234     ltx.__pdf.backend_Page_gput
235     (
236       \__pdf_backend_luastring:n { #1 },
237       \__pdf_backend_luastring:n { #2 }
238     )
239   }
240 }
241 % the command to remove a default value.
242 % Uses a prop with pdflatex + dvi,
243 % changes a lua table with luatex
244 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
245 {

```

```

246     \tex_directlua:D
247     {
248         ltx.__pdf.backend_Page_gremove (\__pdf_backend_luastring:n { #1 })
249     }
250 }
251 % the command used in the document.
252 % direct call of the primitive special with dvips/dvipdfmx
253 % \latelua: fill a page related table with lualatex, merge it with the page
254 % table and push it directly
255 % write to aux and store in prop with pdflatex
256 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
257 {
258     \tex_latelua:D
259     {
260         ltx.__pdf.backend_ThisPage_gput
261         (
262             tex.count["g_shipout_readonly_int"],
263             \__pdf_backend_luastring:n { #1 },
264             \__pdf_backend_luastring:n { #2 }
265         )
266         ltx.__pdf.backend_ThisPage_gpush (tex.count["g_shipout_readonly_int"])
267     }
268 }
269 %the code to push the values, used in shipout
270 %merges the two props and then fills the register in pdflatex
271 %merges the two tables (the one is probably still empty) and then fills (in lua) in luatex
272 %issues the values stored in the global prop with dvi
273 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
274 {
275     \tex_latelua:D
276     {
277         ltx.__pdf.backend_ThisPage_gpush (tex.count["g_shipout_readonly_int"])
278     }
279 }
280
281 </luatex>
282 <*dvipdfmx | xdvipdfmx>
283 %the primitive
284 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
285 {
286     \tex_special:D{pdf:~put~@thispage-<<#1>>}
287 }
288 % the command to store default values.
289 % Uses a prop with pdflatex + dvi,
290 % sets a lua table with lualatex
291 \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2
292 {
293     \pdfdict_gput:nnn {g__pdf_Core/Page}{ #1 }{ #2 }
294 }
295 % the command to remove a default value.
296 % Uses a prop with pdflatex + dvi,
297 % changes a lua table with lualatex
298 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
299 {

```

```

300     \pdfdict_gremove:nn {g__pdf_Core/Page}{ #1 }
301   }
302   % the command used in the document.
303   % direct call of the primitive special with dvips/dvipdfmx
304   % \lualatex: fill a page related table with luatex, merge it with the page
305   % table and push it directly
306   % write to aux and store in prop with pdflatex
307   \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
308   {
309     \__pdf_backend_Page_primitive:n { /#1~#2 }
310   }
311   %the code to push the values, used in shipout
312   %merges the two props and then fills the register in pdflatex
313   %merges the two tables (the one is probably still empty)
314   % and then fills (in lua) in luatex
315   %issues the values stored in the global prop with dvi
316   \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
317   {
318     \exp_args:Nx \__pdf_backend_Page_primitive:n
319       { \pdfdict_use:n { g__pdf_Core/Page} }
320   }
321   </dvipdfmx | xdvipdfmx>
322   <*dvips>
323   \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
324   {
325     \tex_special:D{ps:~[{ThisPage}<<#1>>~/PUT~pdfmark} %]
326   }
327   % the command to store default values.
328   % Uses a prop with pdflatex + dvi,
329   % sets a lua table with luatex
330   \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2
331   {
332     \pdfdict_gput:nnn {g__pdf_Core/Page}{ #1 }{ #2 }
333   }
334   % the command to remove a default value.
335   % Uses a prop with pdflatex + dvi,
336   % changes a lua table with luatex
337   \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
338   {
339     \pdfdict_gremove:nn {g__pdf_Core/Page}{ #1 }
340   }
341   % the command used in the document.
342   % direct call of the primitive special with dvips/dvipdfmx
343   % \lualatex: fill a page related table with luatex, merge it with the page
344   % table and push it directly
345   % write to aux and store in prop with pdflatex
346   \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
347   {
348     \__pdf_backend_Page_primitive:n { /#1~#2 }
349   }
350   %the code to push the values, used in shipout
351   %merges the two props and then fills the register in pdflatex
352   %merges the two tables (the one is probably still empty)
353   %and then fills (in lua) in luatex

```



```

354 %issues the values stored in the global prop with dvi
355 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
356 {
357   \exp_args:Nx \__pdf_backend_Page_primitive:n
358     { \pdfdict_use:n { g__pdf_Core/Page} }
359 }
360 </dvips>
361 <*dvisvgm>
362 % mostly only dummies ...
363 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
364 {}
365 % Uses a prop with pdflatex + dvi,
366 \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2
367 {
368   \pdfdict_gput:nnn {g__pdf_Core/Page}{ #1 }{ #2 }
369 }
370 % the command to remove a default value.
371 % Uses a prop with pdflatex + dvi,
372 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
373 {
374   \pdfdict_gremove:nn {g__pdf_Core/Page}{ #1 }
375 }
376 % the command used in the document.
377 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
378 {}
379 %the code to push the values, used in shipout
380 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
381 {}
382 </dvisvgm>

```

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

## 1.6 “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.

\\_\_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 is a no-op with xdvipdfmx and dvips.

\\_\_pdf\_backend\_PageResources\_obj\_gpush:

```

383 % backend commands the command to fill the register
384 % and to push the values.
385 %
386 % The names are quite often needed
387 % a similar list is now in l3pdfmanagement. Perhaps it should be merged.
388 <*drivers>
389 \clist_const:Nn \c__pdf_backend_PageResources_clist
390 {
391   ExtGState,
392   ColorSpace,

```

```

393     Pattern,
394     Shading,
395   }
396 </drivers>
397 % pdftex and luatex
398 <*pdftex | luatex>
399 %create the backend objects:
400 \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
401 {
402   \__pdf_backend_object_new:nn {Page/Resources/#1} {dict}
403   \cs_if_exist:NT \tex_directlua:D
404     {
405       \tex_directlua:D
406       {
407         ltx.__pdf.object["Page/Resources/#1"]
408         =
409         "\__pdf_backend_object_ref:n{Page/Resources/#1}"
410       }
411     }
412 }
413 </pdftex | luatex>
414 <*luatex>
415 %values are only stored in a prop and will be output at end document.
416 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
417 {
418   \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
419   % luatex must also trigger the lua side
420   \tex_latelua:D{ltx.__pdf.Page.Resources.#1=true}
421   \tex_latelua:D
422   {
423     ltx.pdf.Page_Resources_gpush(tex.count["g_shipout_readonly_int"])
424   }
425 }
426 </luatex>
427 <*pdftex>
428 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
429 {
430   \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
431 }
432 </pdftex>
433 <*pdftex | luatex>
434 %code for end of document code
435 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush:
436 {
437   \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
438   {
439     \prop_if_empty:cF
440     { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/##1} }
441     {
442       \__pdf_backend_object_write:nx
443       { Page/Resources/##1 }
444       { \pdfdict_use:n { g__pdf_Core/Page/Resources/##1} }
445     }
446   }

```

```

447 }
448 </pdftex | luatex>
449 % xdvipdfmx
450 % \special{pdf:pageresources<<#1>>} doesn't work correctly with object names ...
451 % https://tug.org/pipermail/dvipdfmx/2019-August/000021.html,
452 % so we use \special{pdf:put @resources}
453 % this must be issued on every page!
454 <*dvipdfmx | xdvipdfmx>
455 %objects should not only be created but also "initialized"
456 % initialization should be done before anyone tries to write
457 % so we add rules for the backend.
458 <xdvipdfmx> \hook_gset_rule:nnn{shipout/firstpage}{l3backend-xetex}{after}{pdf}
459 <dvipdfmx> \hook_gset_rule:nnn{shipout/firstpage}{l3backend-dvipdfmx}{after}{pdf}
460 %
461 \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
462 {
463   \__pdf_backend_object_new:nn { Page/Resources/#1 } { dict }
464   \hook_gput_code:nnn{shipout/firstpage}{pdf}{\__pdf_backend_object_write:nn { Page/Resour
465 }
466 \cs_new_protected:Npn \__pdf_backend_PageResources:n #1
467 {
468   \__pdf_backend:n {put~@resources~<<#1>>}
469 }
470 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
471 {
472   % this is not used for output, but there is a test if the resource is empty
473   \exp_args:Nnx
474   \prop_gput:cnn { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/#1 } }
475   { \str_convert_pdfname:n {#2} }{ #3 }
476   %objects are not filled with \pdf_object_write as this is not additive!
477   \__pdf_backend:x
478   {
479     put~\__pdf_backend_object_ref:n {Page/Resources/#1}<<#2~#3>>
480   }
481 }
482
483 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush: {}
484 </dvipdfmx | xdvipdfmx>
485 <*dvips>
486 % dvips unneeded, or no-op
487 \cs_new_protected:Npn \__pdf_backend_PageResources:n #1 {}
488 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
489 { %only for the show command TEST!!
490   \pdffdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
491 }
492 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush: {}
493 </dvips>
494 <*dvisvgm>
495 % dvips unneeded, or no-op
496 \cs_new_protected:Npn \__pdf_backend_PageResources:n #1 {}
497 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
498 { %only for the show command TEST!!
499   \pdffdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
500 }

```

```

501 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush: {}
502 </dvisvgm>

```

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

### 1.6.1 Page resources /Properties + BDC operators

```

\__pdf_backend_bdc:nn \__pdf_backend_bdcobject:nn, \__pdf_backend_bdcobject:n,
\__pdf_backend_bdcobject:nn \__pdf_backend_bmc:n and \__pdf_backend_emc: are the backend command that cre-
\__pdf_backend_bdcobject:n ate the bdc/emc marker and store the properties. \__pdf_backend_PageResources_-
\__pdf_backend_bmc:n gpush:n outputs the /Properties and/or the other resources for the current page.
\__pdf_backend_emc:
\__pdf_backend_PageResources_gpush:n
503 % pdfTeX and luatex (and perhaps dvips ...) need to know if there are in a
504 % xform stream ...
505 <*drivers>
506 \bool_new:N \l__pdf_backend_xform_bool
507 </drivers>
508 <*dvips>
509 % dvips is easy: create an object, and reference it in the bdc
510 % ghostscript will then automatically replace it by a name
511 % and add the name to the /Properties dict
512 % special variant von accsupp
513 % https://chat.stackexchange.com/transcript/message/50831812#50831812
514 %
515 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2 % #1 eg. Span, #2: dict_content
516 {
517   \__pdf_backend_pdfmark:x{/#1-<<#2>>~/BDC}
518 }
519 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
520 {
521   \__pdf_backend_pdfmark:x{/#1~\__pdf_backend_object_ref:n{#2}~/BDC}
522 }
523 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1 % #1 eg. Span,
524 {
525   \__pdf_backend_pdfmark:x{/#1~\__pdf_backend_object_last:~/BDC}
526 }
527 \cs_set_protected:Npn \__pdf_backend_emc:
528 {
529   \__pdf_backend_pdfmark:n{/EMC} %
530 }
531 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
532 {
533   \__pdf_backend_pdfmark:n{/#1~/BMC} %
534 }
535 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1 {}
536
537 </dvips>
538 <*dvisvgm>
539 % dvisvgm should do nothing
540 %
541 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2 % #1 eg. Span, #2: dict_content
542 {}
543 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
544 {}

```

```

545 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1 % #1 eg. Span,
546 {}
547 \cs_set_protected:Npn \__pdf_backend_emc:
548 {}
549 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
550 {}
551 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1 {}
552
553 </divsvgm>
554
555 % xetex has to create the entries in the /Properties manually
556 % (like the other backends)
557 % use pdfbase special
558 % https://chat.stackexchange.com/transcript/message/50832016#50832016
559 % the property is added to xform resources automatically,
560 % no need to worry about it.
561 <*dvipdfmx | xdvipdfmx>
562 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
563 {
564   \int_gincr:N \g__pdf_backend_name_int
565   \__kernel_backend_literal:x
566   {
567     pdf:code~/#1/l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC
568   }
569   \__kernel_backend_literal:x
570   {
571     pdf:put~@resources~
572     <<
573       /Properties~
574       <<
575         /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl
576         \__pdf_backend_object_ref:n { #2 }
577       >>
578     >>
579   }
580 }
581 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1 % #1 eg. Span
582 {
583   \int_gincr:N \g__pdf_backend_name_int
584   \__kernel_backend_literal:x
585   {
586     pdf:code~/#1/l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC
587   }
588   \__kernel_backend_literal:x
589   {
590     pdf:put~@resources~
591     <<
592       /Properties~
593       <<
594         /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl
595         \__pdf_backend_object_last:
596       >>
597     >>
598   }

```

```

599     }
600 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
601     {
602     \__kernel_backend_literal:n {pdf:code~/#1~BMC} %pdfbase
603     }
604
605 %this require management
606 \cs_set_protected:Npn \__pdf_backend_bdc_contobj:nn #1 #2
607     {
608     \pdf_object_unnamed_write:nn { dict }{ #2 }
609     \__pdf_backend_bdcobject:n { #1 }
610     }
611
612 \cs_set_protected:Npn \__pdf_backend_bdc_contstream:nn #1 #2
613     {
614     \__kernel_backend_literal:n {pdf:code~ /#1~<<#2>>~BDC }
615     }
616
617 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2
618     {
619     \bool_if:NTF \g__pdfmanagement_active_bool
620     {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contobj:nn}
621     {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contstream:nn}
622     \__pdf_backend_bdc:nn {#1}{#2}
623     }
624 \cs_set_protected:Npn \__pdf_backend_emc:
625     {
626     \__kernel_backend_literal:n {pdf:code~EMC} %pdfbase
627     }
628 % properties are handled automatically, but the other resources should be added
629 % at shipout
630 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1
631     {
632     \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
633     {
634     \prop_if_empty:cF { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/##1} }
635     {
636     \__kernel_backend_literal:x
637     {
638     pdf:put~@resources~
639     <</##1~\__pdf_backend_object_ref:n {Page/Resources/##1}>>
640     }
641     }
642     }
643     }
644 </dvipdfmx | xdvipdfmx>
645 % luatex + pdftex
646 <*luatex>
647 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
648     {
649     \int_gincr:N \g__pdf_backend_name_int
650     \exp_args:Nx\__kernel_backend_literal_page:n
651     { /#1 ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
652     \bool_if:NTF \l__pdf_backend_xform_bool

```

```

653 {
654   \exp_args:Nnx\pdfdict_gput:nnn
655   { g__pdf_Core/Xform/Resources/Properties }
656   { l3pdf\int_use:N\g__pdf_backend_name_int }
657   { \__pdf_backend_object_ref:n { #2 } }
658 }
659 {
660   \exp_args:Nx \tex_latelua:D
661   {
662     ltx.pdf.Page_Resources_Properties_gput
663     (
664       tex.count["g_shipout_readonly_int"],
665       "l3pdf\int_use:N\g__pdf_backend_name_int",
666       "\__pdf_backend_object_ref:n { #2 }"
667     )
668   }
669 }
670 }
671 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1% #1 eg. Span
672 {
673   \int_gincr:N \g__pdf_backend_name_int
674   \exp_args:Nx\__kernel_backend_literal_page:n
675   { /#1 ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
676   \bool_if:NTF \l__pdf_backend_xform_bool
677   {
678     \exp_args:Nnx\pdfdict_gput:nnn %no handler needed
679     { g__pdf_Core/Xform/Resources/Properties }
680     { l3pdf\int_use:N\g__pdf_backend_name_int }
681     { \__pdf_backend_object_last: }
682   }
683   {
684     \exp_args:Nx \tex_latelua:D
685     {
686       ltx.pdf.Page_Resources_Properties_gput
687       (
688         tex.count["g_shipout_readonly_int"],
689         "l3pdf\int_use:N\g__pdf_backend_name_int",
690         "\__pdf_backend_object_last:"
691       )
692     }
693   }
694 }
695 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
696 {
697   \__kernel_backend_literal_page:n { /#1~BMC }
698 }
699 \cs_set_protected:Npn \__pdf_backend_bdc_contobj:nn #1 #2
700 {
701   \pdf_object_unnamed_write:nn { dict } { #2 }
702   \__pdf_backend_bdcobject:n { #1 }
703 }
704 \cs_set_protected:Npn \__pdf_backend_bdc_contstream:nn #1 #2
705 {
706   \__kernel_backend_literal_page:n { /#1~<<#2>>~BDC }

```

```

707 }
708 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2
709 {
710   \bool_if:NTF \g__pdfmanagement_active_bool
711     {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contobj:nn}
712     {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contstream:nn}
713     \__pdf_backend_bdc:nn {#1}{#2}
714 }
715 \cs_set_protected:Npn \__pdf_backend_emc:
716 {
717   \__kernel_backend_literal_page:n { EMC }
718 }
719
720 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1 {}
721 </luatex>
722 <*pdfTeX>
723 % pdfLaTeX is the most complicated as it has to go through the aux ...
724 % the push command is extended to take other resources too
725 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
726 {
727   \int_gincr:N \g__pdf_backend_name_int
728   \exp_args:Nx \__kernel_backend_literal_page:n
729     { /#1 ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
730   % code to set the property ....
731   \int_gincr:N \g__pdf_backend_resourceid_int
732   \bool_if:NTF \l__pdf_backend_xform_bool
733   {
734     \exp_args:Nnxx \pdfdict_gput:nnn %no handler needed
735       { g__pdf_Core/Xform/Resources/Properties }
736       { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
737       { \__pdf_backend_object_ref:n { #2 } }
738   }
739   {
740     %\zref@labelbylist
741     { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
742     { l3pdf }
743     % \ref_label:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
744     \__pdf_backend_ref_label:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
745     \tl_set:Nx \l__pdf_tmpa_tl
746     {
747       %\zref@extractdefault
748       { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
749       {pdf@abspage}
750       {0}
751       %\ref_value:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
752       \__pdf_backend_ref_value:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
753     }
754     \pdfdict_if_exist:nF { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
755     {
756       \pdfdict_new:n { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
757     }
758     \exp_args:Nnxx \pdfdict_gput:nnn
759       { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
760       { l3pdf\int_use:N\g__pdf_backend_resourceid_int }

```



```

761         { \_pdf_backend_object_ref:n{#2} }
762     }
763 }
764 \cs_set_protected:Npn \_pdf_backend_bdcobject:n #1% #1 eg. Span
765 {
766     \int_gincr:N \g__pdf_backend_name_int
767     \exp_args:Nx\__kernel_backend_literal_page:n
768     { /#1 ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
769     % code to set the property ....
770     \int_gincr:N\g__pdf_backend_resourceid_int
771     \bool_if:NTF \l__pdf_backend_xform_bool
772     {
773         \exp_args:Nxxx\pdfdict_gput:nnn
774         { g__pdf_Core/Xform/Resources/Properties }
775         { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
776         { \_pdf_backend_object_last: }
777     }
778     {
779         %\zref@labelbylist
780         % { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
781         % { l3pdf }
782         %\ref_label:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
783         \_pdf_backend_ref_label:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
784         \tl_set:Nx \l__pdf_tmpa_tl
785         {
786             %\zref@extractdefault
787             % { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
788             % {pdf@abspage}
789             % {0}
790             % \ref_value:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
791             \_pdf_backend_ref_value:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
792         }
793         \pdfdict_if_exist:nF { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties
794         {
795             \pdfdict_new:n { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
796         }
797         \exp_args:Nxxx\pdfdict_gput:nnn
798         { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
799         { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
800         { \_pdf_backend_object_last: }
801         %\pdfdict_show:n { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
802     }
803 }
804 \cs_set_protected:Npn \_pdf_backend_bmc:n #1
805 {
806     \__kernel_backend_literal_page:n { /#1-BMC }
807 }
808 \cs_set_protected:Npn \_pdf_backend_bdc_contobj:nn #1 #2
809 {
810     \pdf_object_unnamed_write:nn { dict } { #2 }
811     \_pdf_backend_bdcobject:n { #1 }
812 }
813 \cs_set_protected:Npn \_pdf_backend_bdc_contstream:nn #1 #2
814 {

```

```

815     \_kernel_backend_literal_page:n { /#1~<<#2>>~BDC }
816   }
817   \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2
818   {
819     \bool_if:NTF \g__pdfmanagement_active_bool
820     {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contobj:nn}
821     {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contstream:nn}
822     \__pdf_backend_bdc:nn {#1}{#2}
823   }
824   \cs_set_protected:Npn \__pdf_backend_emc:
825   {
826     \_kernel_backend_literal_page:n { EMC }
827   }
828
829   \cs_new:Npn \__pdf_backend_PageResources_gpush_aux:n #1 %#1 ExtGState etc
830   {
831     \prop_if_empty:cF
832     { \_kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/#1} }
833     {
834       \pdffdict_item:ne { #1 }{ \pdf_object_ref:n {Page/Resources/#1}}
835     }
836   }
837
838   \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1
839   {
840     \exp_args:NNx \tex_global:D \tex_pdfpageresources:D
841     {
842       \prop_if_exist:cT
843       { \_kernel_pdffdict_name:n { g__pdf_Core/backend_Page#1/Resources/Properties } }
844       {
845         /Properties~
846         <<
847         \prop_map_function:cN
848         { \_kernel_pdffdict_name:n { g__pdf_Core/backend_Page#1/Resources/Property
849         \pdffdict_item:ne
850         >>
851       }
852       %% add ExtGState etc
853       \clist_map_function:NN
854       \c__pdf_backend_PageResources_clist
855       \__pdf_backend_PageResources_gpush_aux:n
856     }
857   }
858
859   </pdftex>

```

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

## 1.7 “Catalog” & subdirectories (pdfcatalog)

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

### 1.7.1 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 we need some code to push the tree if files have been added. dvips wants code for every file and then creates the Name tree automatically.

```

860 % pdflatex
861 <*pdftex>
862 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_gpush:n #1 %array content
863 {
864     \pdf_object_unnamed_write:nn {dict} {/Names [#1] }
865     \tex_pdfnames:D {/EmbeddedFiles~\pdf_object_ref_last:}
866 }
867 </pdftex>
868 <*luatex>
869 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_gpush:n #1 %array content
870 {
871     \pdf_object_unnamed_write:nn {dict} {/Names [#1] }
872     \tex_pdfextension:D~names~{/EmbeddedFiles~\pdf_object_ref_last: }
873 }
874 </luatex>
875 <*dviPDFmx | xdvipdfmx>
876 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_gpush:n #1 %array content
877 {
878     \pdf_object_unnamed_write:nn {dict} { /Names [#1] }
879     %n or x?
880     \__pdf_backend:x {put~@names~<</EmbeddedFiles~\pdf_object_ref_last: >>}
881 }
882 </dviPDFmx | xdvipdfmx>
883
884 %dvips: noop
885 <*dvips>
886 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_gpush:n #1 {}
887 </dvips>
888 %dvisvgm: noop
889 <*dvisvgm>
890 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_gpush:n #1 {}
891 </dvisvgm>
892

```

Names in the EmbeddedFiles name tree must sorted alphabetically, so we need commands to create this names. And we need a sequence to store the names and the objects. We use the prefix l3ef, and we assume that at most 9999 files will be used.

\g\_\_pdf\_backend\_EmbeddedFiles\_int

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

\\_\_pdf\_backend\_EmbeddedFiles\_name:

```

893 <*drivers>
894 \int_new:N \g__pdf_backend_EmbeddedFiles_int
895 \cs_new:Npn \__pdf_backend_EmbeddedFiles_name:
896 {
897     (
898     l3ef

```

```

899 \int_compare:nNnT {\g__pdf_backend_EmbeddedFiles_int} < {10}
900 {0}
901 \int_compare:nNnT {\g__pdf_backend_EmbeddedFiles_int} < {100}
902 {0}
903 \int_compare:nNnT {\g__pdf_backend_EmbeddedFiles_int} < {1000}
904 {0}
905 \int_use:N \g__pdf_backend_EmbeddedFiles_int
906 )
907 }
908 </drivers>

```

(End definition for \\_\_pdf\_backend\_EmbeddedFiles\_name:.)

\g\_\_pdf\_backend\_EmbeddedFiles\_seq  
\g\_\_pdf\_backend\_EmbeddedFiles\_named\_prop

The sequence will hold the content of the array that is pushed out at then end (not with dvips), the prop holds the obj names-names relation.

(End definition for \g\_\_pdf\_backend\_EmbeddedFiles\_seq and \g\_\_pdf\_backend\_EmbeddedFiles\_named\_prop.)

```

909 <*drivers>
910 \seq_new:N \g__pdf_backend_EmbeddedFiles_seq
911 \prop_new:N \g__pdf_backend_EmbeddedFiles_named_prop
912 </drivers>

```

\\_\_pdf\_backend\_NamesEmbeddedFiles\_add:n

This command saves an object reference of a filespec dictionary in the EmbeddedFiles name tree. We define a prop to store the relation between object name and name in the name tree.

```

913 <*pdftex | luatex | dvipdfmx | xdvipdfmx>
914 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_add:n #1
915   {%#1 object ref
916   {
917     \int_gincr:N \g__pdf_backend_EmbeddedFiles_int
918     \prop_gput:Nnx \g__pdf_backend_EmbeddedFiles_named_prop
919       { #1 }
920     { \__pdf_backend_EmbeddedFiles_name: }
921     \seq_gput_right:Nx \g__pdf_backend_EmbeddedFiles_seq
922       { \__pdf_backend_EmbeddedFiles_name: \c_space_tl #1 }
923   }
924
925 </pdftex | luatex | dvipdfmx | xdvipdfmx>
926 <*dvips>
927 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_add:n #1
928   {
929     \int_gincr:N \g__pdf_backend_EmbeddedFiles_int
930     \prop_gput:Nnx \g__pdf_backend_EmbeddedFiles_named_prop
931       { #1 }
932     { \__pdf_backend_EmbeddedFiles_name: }
933     \__pdf_backend_pdfmark:x
934     {
935       /Name~\__pdf_backend_EmbeddedFiles_name:~
936       /FS~#1~
937       /EMBED
938     }
939   }
940 </dvips>

```

```

941 <*dvisvgm>
942 %no op. Or is there any sensible use for it?
943 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_add:n #1
944 {}
945 </dvisvgm>

```

(End definition for \\_\_pdf\_backend\_NamesEmbeddedFiles\_add:n.)

## 1.7.2 Form XObject / 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 946 <*pdfTeX>
\__pdf_backend_xform_ref:n 947 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4
                                948 % #1 name
                                949 % #2 attributes
                                950 % #3 resources
                                951 % #4 content, not necessarily a box!
                                952 {
                                953   \hbox_set:Nn \l__pdf_backend_tmpa_box
                                954   {
                                955     \bool_set_true:N \l__pdf_backend_xform_bool
                                956     \prop_gc_clear:c { \__kernel_pdffdict_name:n { g__pdf_Core/Xform/Resources/Properties } }
                                957     #4
                                958   }
                                959   %store the dimensions
                                960   \tl_const:cx
                                961   { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
                                962   { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
                                963   \tl_const:cx
                                964   { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
                                965   { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
                                966   \tl_const:cx
                                967   { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
                                968   { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
                                969   %% do we need to test if #2 and #3 are empty??
                                970   \tex_immediate:D \tex_pdfxform:D
                                971   ~ attr ~ { #2 }
                                972   %% which other resources should be default? Is an argument actually needed?
                                973   ~ resources ~
                                974   {
                                975     #3
                                976     \int_compare:nNnT
                                977     { \prop_count:c { \__kernel_pdffdict_name:n { g__pdf_Core/Xform/Resources/Properties } }
                                978     >
                                979     { 0 }
                                980     {
                                981       /Properties~
                                982       <<
                                983       \pdffdict_use:n { g__pdf_Core/Xform/Resources/Properties }

```

```

984         >>
985     }
986
987     \prop_if_empty:cF
988     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/ExtGState } }
989     {
990         /ExtGState~ \pdf_object_ref:n { Page/Resources/ExtGState }
991     }
992     \prop_if_empty:cF
993     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/Pattern } }
994     {
995         /Pattern~ \pdf_object_ref:n { Page/Resources/Pattern }
996     }
997     \prop_if_empty:cF
998     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/Shading } }
999     {
1000         /Shading~ \pdf_object_ref:n { Page/Resources/Shading }
1001     }
1002     \prop_if_empty:cF
1003     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/ColorSpace } }
1004     {
1005         /ColorSpace~ \pdf_object_ref:n { Page/Resources/ColorSpace }
1006     }
1007 }
1008 \l__pdf_backend_tmpa_box
1009 \int_const:cn
1010 { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1011 { \tex_pdflastxform:D }
1012 }
1013
1014 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
1015 {
1016     \tex_pdfrefxform:D
1017     \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1018     \scan_stop:
1019 }
1020
1021 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1022 {
1023     \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int } ~ 0 ~ R
1024 }
1025 </pdftex>
1026 <*luatex>
1027 %luatex
1028 %nearly identical but not completely ...
1029 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4
1030 % #1 name
1031 % #2 attributes
1032 % #3 resources
1033 % #4 content, not necessarily a box!
1034 {
1035     \hbox_set:Nn \l__pdf_backend_tmpa_box
1036     {
1037         \bool_set_true:N \l__pdf_backend_xform_bool

```

```

1038     \prop_gclear:c { \__kernel_pdffdict_name:n { g__pdf_Core/Xform/Resources/Properties }
1039     #4
1040   }
1041   \tl_const:cx
1042     { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1043     { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
1044   \tl_const:cx
1045     { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1046     { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
1047   \tl_const:cx
1048     { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1049     { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
1050   %% do we need to test if #2 and #3 are empty??
1051   \tex_immediate:D \tex_pdfxform:D
1052     ~ attr ~ { #2 }
1053   %% which resources should be default? Is an argument actually needed?
1054   ~ resources ~
1055   {
1056     #3
1057     \int_compare:nNnT
1058       { \prop_count:c { \__kernel_pdffdict_name:n { g__pdf_Core/Xform/Resources/Properties }
1059       >
1060       { 0 }
1061       {
1062         /Properties~
1063         <<
1064         \pdffdict_use:n { g__pdf_Core/Xform/Resources/Properties }
1065         >>
1066       }
1067     \prop_if_empty:cF
1068       { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/ExtGState } }
1069       {
1070         /ExtGState~ \pdf_object_ref:n { Page/Resources/ExtGState }
1071       }
1072     \prop_if_empty:cF
1073       { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/Pattern } }
1074       {
1075         /Pattern~ \pdf_object_ref:n { Page/Resources/Pattern }
1076       }
1077     \prop_if_empty:cF
1078       { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/Shading } }
1079       {
1080         /Shading~ \pdf_object_ref:n { Page/Resources/Shading }
1081       }
1082     \prop_if_empty:cF
1083       { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/ColorSpace } }
1084       {
1085         /ColorSpace~ \pdf_object_ref:n { Page/Resources/ColorSpace }
1086       }
1087   }
1088   \l__pdf_backend_tmpa_box
1089   \int_const:cn
1090     { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1091     { \tex_pdflastxform:D }

```

```

1092 }
1093
1094 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1 %protected as with xelatex
1095 {
1096   \tex_pdfrefxform:D \int_use:c
1097   {
1098     c__pdf_backend_xform_ \tl_to_str:n {#1} _int
1099   }
1100   \scan_stop:
1101 }
1102
1103 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1104 { \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int } ~ 0 ~ R }
1105
1106 </luatex>
1107 <*dvipdfmx|xdvipdfmx>
1108 % xetex
1109 % it needs a bit testing if it really works to set the box to 0 before the special ...
1110 % does it disturb viewing the xobject?
1111 % what happens with the resources (bdc)? (should work as they are specials too)
1112 % xetex requires that the special is in horizontal mode. This means it affects
1113 % typesetting. But we can no delay the whole form code to shipout
1114 % as the object reference and the size is often wanted on the current page.
1115 % so we need to allocate a box - but probably they won't be thousands xform
1116 % in a document so it shouldn't matter.
1117 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4
1118 % #1 name
1119 % #2 attributes
1120 % #3 resources
1121 % #4 content, not necessarily a box!
1122 {
1123   \int_gincr:N \g__pdf_backend_object_int
1124   \int_const:cn
1125   { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1126   { \g__pdf_backend_object_int }
1127   \box_new:c { g__pdf_backend_xform_#1_box }
1128   \hbox_gset:cn { g__pdf_backend_xform_#1_box }
1129   {
1130     \bool_set_true:N \l__pdf_backend_xform_bool
1131     #4
1132   }
1133   \tl_const:cx
1134   { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1135   { \tex_the:D \box_wd:c { g__pdf_backend_xform_#1_box } }
1136   \tl_const:cx
1137   { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1138   { \tex_the:D \box_ht:c { g__pdf_backend_xform_#1_box } }
1139   \tl_const:cx
1140   { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1141   { \tex_the:D \box_dp:c { g__pdf_backend_xform_#1_box } }
1142   \box_set_dp:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1143   \box_set_ht:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1144   \box_set_wd:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1145   \hook_gput_next_code:nn {shipout/background}

```



```

1146 {
1147     \mode_leave_vertical: %needed, the xform disappears without it.
1148     \__pdf_backend:x
1149     {
1150         bxobj ~ \__pdf_backend_xform_ref:n { #1 }
1151         \c_space_tl width ~ \pdfxform_wd:n { #1 }
1152         \c_space_tl height ~ \pdfxform_ht:n { #1 }
1153         \c_space_tl depth ~ \pdfxform_dp:n { #1 }
1154     }
1155     \box_use_drop:c { g__pdf_backend_xform_#1_box }
1156     \__pdf_backend:x {put ~ @resources ~<<#3>> }
1157     \__pdf_backend:x
1158     {
1159         put~ @resources ~
1160         <<
1161             /ExtGState~ \pdf_object_ref:n { Page/Resources/ExtGState }
1162         >>
1163     }
1164     \__pdf_backend:x
1165     {
1166         put~ @resources ~
1167         <<
1168             /Pattern~ \pdf_object_ref:n { Page/Resources/Pattern }
1169         >>
1170     }
1171     \__pdf_backend:x
1172     {
1173         put~ @resources ~
1174         <<
1175             /Shading~ \pdf_object_ref:n { Page/Resources/Shading }
1176         >>
1177     }
1178     \__pdf_backend:x
1179     {
1180         put~ @resources ~
1181         <<
1182             /ColorSpace~
1183             \pdf_object_ref:n { Page/Resources/ColorSpace }
1184         >>
1185     }
1186     \exp_args:Nx
1187     \__pdf_backend:x {exobj ~<<#2>>}
1188 }
1189 }
1190
1191
1192
1193 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1194 {
1195     @pdf.xform \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1196 }
1197
1198 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
1199 {

```

```

1200     \hbox_set:Nn \l__pdf_backend_tmpa_box
1201     {
1202         \__pdf_backend:x
1203         {
1204             uxobj~ \__pdf_backend_xform_ref:n { #1 }
1205         }
1206     }
1207     \box_set_wd:Nn \l__pdf_backend_tmpa_box { \pdfxform_wd:n { #1 } }
1208     \box_set_ht:Nn \l__pdf_backend_tmpa_box { \pdfxform_ht:n { #1 } }
1209     \box_set_dp:Nn \l__pdf_backend_tmpa_box { \pdfxform_dp:n { #1 } }
1210     \box_use_drop:N \l__pdf_backend_tmpa_box
1211 }
1212 </dvipdfmx | xdvipdfmx>
1213 <*dvisvgm>
1214 % unclear what it should do!!
1215 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4 {}
1216 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1 {}
1217 \cs_new:Npn \__pdf_backend_xform_ref:n {}
1218 </dvisvgm>
1219 <*drivers>
1220 %% all
1221 \prg_new_conditional:Npnn \__pdf_backend_xform_if_exist:n #1 { p , T , F , TF }
1222 {
1223     \int_if_exist:cTF { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1224     { \prg_return_true: }
1225     { \prg_return_false: }
1226 }
1227 \prg_new_eq_conditional:NNn \pdfxform_if_exist:n \__pdf_backend_xform_if_exist:n
1228 { TF , T , F , p }
1229 </drivers>

```

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

## 1.8 lua code for lualatex

```

1230 <*lua>
1231 ltx= ltx or {}
1232 ltx.__pdf = ltx.__pdf or {}
1233 ltx.__pdf.Page = ltx.__pdf.Page or {}
1234 ltx.__pdf.Page.dflt = ltx.__pdf.Page.dflt or {}
1235 ltx.__pdf.Page.Resources = ltx.__pdf.Page.Resources or {}
1236 ltx.__pdf.Page.Resources.Properties = ltx.__pdf.Page.Resources.Properties or {}
1237 ltx.__pdf.Page.Resources.List={"ExtGState","ColorSpace","Pattern","Shading"}
1238 ltx.__pdf.object = ltx.__pdf.object or {}
1239
1240 ltx.pdf= ltx.pdf or {} -- for "public" functions
1241
1242 local __pdf = ltx.__pdf
1243 local pdf = pdf
1244
1245 local function __pdf_backend_Page_gput (name,value)
1246     __pdf.Page.dflt[name]=value
1247 end

```

```

1248
1249 local function __pdf_backend_Page_gremove (name)
1250   __pdf.Page.dflt[name]=nil
1251 end
1252
1253 local function __pdf_backend_Page_gclear ()
1254   __pdf.Page.dflt={}
1255 end
1256
1257 local function __pdf_backend_ThisPage_gput (page,name,value)
1258   __pdf.Page[page] = __pdf.Page[page] or {}
1259   __pdf.Page[page][name]=value
1260 end
1261
1262 local function __pdf_backend_ThisPage_gpush (page)
1263   local token=""
1264   local t = {}
1265   local tkeys= {}
1266   for name,value in pairs(__pdf.Page.dflt) do
1267     t[name]=value
1268   end
1269   if __pdf.Page[page] then
1270     for name,value in pairs(__pdf.Page[page]) do
1271       t[name] = value
1272     end
1273   end
1274   -- sort the table to get reliable test files.
1275   for name,value in pairs(t) do
1276     table.insert(tkeys,name)
1277   end
1278   table.sort(tkeys)
1279   for _,name in ipairs(tkeys) do
1280     token = token .. "/"..name.." " ..t[name]
1281   end
1282   return token
1283 end
1284
1285 function ltx.__pdf.backend_ThisPage_gput (page,name,value) -- tex.count["g_shipout_readonly"]
1286   __pdf_backend_ThisPage_gput (page,name,value)
1287 end
1288
1289 function ltx.__pdf.backend_ThisPage_gpush (page)
1290   pdf.setpageattributes(__pdf_backend_ThisPage_gpush (page))
1291 end
1292
1293 function ltx.__pdf.backend_Page_gput (name,value)
1294   __pdf_backend_Page_gput (name,value)
1295 end
1296
1297 function ltx.__pdf.backend_Page_gremove (name)
1298   __pdf_backend_Page_gremove (name)
1299 end
1300
1301 function ltx.__pdf.backend_Page_gclear ()

```

```

1302  __pdf_backend_Page_gclear ()
1303  end
1304
1305
1306  local Properties = ltx.__pdf.Page.Resources.Properties
1307  local ResourceList= ltx.__pdf.Page.Resources.List
1308  local function __pdf_backend_PageResources_gpush (page)
1309    local token=""
1310    if Properties[page] then
1311      -- we sort the table, so that the pdf test works
1312      local t = {}
1313      for name,value in pairs (Properties[page]) do
1314        table.insert (t,name)
1315      end
1316      table.sort (t)
1317      for _,name in ipairs(t) do
1318        token = token .. "/"..name.." ".. Properties[page][name]
1319      end
1320      token = "/Properties <<\"..token..\">>"
1321    end
1322    for i,name in ipairs(ResourceList) do
1323      if ltx.__pdf.Page.Resources[name] then
1324        token = token .. "/"..name.." "..ltx.pdf.object_ref("Page/Resources/"..name)
1325      end
1326    end
1327    return token
1328  end
1329
1330  -- the function is public, as I probably need it in tagpdf too ...
1331  function ltx.pdf.Page_Resources_Properties_gput (page,name,value) -- tex.count["g_shipout_re
1332    Properties[page] = Properties[page] or {}
1333    Properties[page][name]=value
1334    pdf.setpageresources(__pdf_backend_PageResources_gpush (page))
1335  end
1336
1337  function ltx.pdf.Page_Resources_gpush(page)
1338    pdf.setpageresources(__pdf_backend_PageResources_gpush (page))
1339  end
1340
1341  function ltx.pdf.object_ref (objname)
1342    if ltx.__pdf.object[objname] then
1343      local ref= ltx.__pdf.object[objname]
1344      return ref
1345    else
1346      return "false"
1347    end
1348  end
1349  </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>B</b>		<code>\directlua</code> ..... 61
bool commands:		<b>E</b>
<code>\bool_if:NTF</code> ..... 619, 652, 676, 710, 732, 771, 819		exp commands:
<code>\bool_new:N</code> ..... 506		<code>\exp_args:NNx</code> ..... 840
<code>\bool_set_true:N</code> ..... 955, 1037, 1130		<code>\exp_args:Nnx</code> ..... 473, 654, 678
box commands:		<code>\exp_args:Nnxx</code> ..... 734, 758, 773, 797
<code>\box_dp:N</code> ..... 968, 1049, 1141		<code>\exp_args:Nx</code> ..... 207, 318,
<code>\box_ht:N</code> ..... 965, 1046, 1138		357, 650, 660, 674, 684, 728, 767, 1186
<code>\box_new:N</code> ..... 53, 1127		<b>H</b>
<code>\box_set_dp:Nn</code> ..... 1142, 1209		hbox commands:
<code>\box_set_ht:Nn</code> ..... 1143, 1208		<code>\hbox_gset:Nn</code> ..... 1128
<code>\box_set_wd:Nn</code> ..... 1144, 1207		<code>\hbox_set:Nn</code> ..... 953, 1035, 1200
<code>\box_use_drop:N</code> ..... 1155, 1210		hook commands:
<code>\box_wd:N</code> ..... 962, 1043, 1135		<code>\hook_gput_code:nnn</code> .... 98, 102, 464
<b>C</b>		<code>\hook_gput_next_code:nn</code> ..... 1145
clist commands:		<code>\hook_gset_rule:nnnn</code> ..... 458, 459
<code>\clist_const:Nn</code> ..... 389		<b>I</b>
<code>\clist_map_function:NN</code> ..... 853		int commands:
<code>\clist_map_inline:Nn</code> 400, 437, 461, 632		<code>\int_compare:nNnTF</code> ..... 899, 901, 903, 976, 1057
cs commands:		<code>\int_const:Nn</code> ..... 1009, 1089, 1124
<code>\cs_generate_variant:Nn</code> 29, 30, 41, 42		<code>\int_gincr:N</code> ... 173, 564, 583, 649,
<code>\cs_gset_eq:NN</code> ..... 620, 621, 711, 712, 820, 821		673, 727, 731, 766, 770, 917, 929, 1123
<code>\cs_if_exist:NTF</code> ..... 403		<code>\int_if_exist:NTF</code> ..... 1223
<code>\cs_new:Npn</code> ..... 37,		<code>\int_new:N</code> ..... 56, 57, 58, 894
215, 829, 895, 1021, 1103, 1193, 1217		<code>\int_use:N</code> . 174, 175, 176, 180, 183,
<code>\cs_new_protected:Npn</code> ..... 31,		184, 567, 575, 586, 594, 651, 656,
110, 119, 135, 141, 147, 154, 161,		665, 675, 680, 689, 729, 736, 741,
170, 197, 220, 230, 244, 256, 273,		743, 744, 748, 751, 752, 760, 768,
284, 291, 298, 307, 316, 323, 330,		775, 780, 782, 783, 787, 790, 791,
337, 346, 355, 363, 366, 372, 377,		799, 905, 1017, 1023, 1096, 1104, 1195
380, 416, 428, 435, 466, 470, 483,		<b>K</b>
487, 488, 492, 496, 497, 501, 535,		kernel internal commands:
551, 630, 720, 838, 862, 869, 876,		<code>\__kernel_backend_literal:n</code> . 48,
886, 890, 914, 927, 943, 947, 1014,		565, 569, 584, 588, 602, 614, 626, 636
1029, 1094, 1117, 1198, 1215, 1216		<code>\__kernel_backend_literal_page:n</code>
<code>\cs_new_protected:Npx</code> ..... 129		..... 650, 674,
<code>\cs_set_protected:Npn</code> .. 515, 519,		697, 706, 717, 728, 767, 806, 815, 826
523, 527, 531, 541, 543, 545, 547,		<code>\__kernel_pdftdict_name:n</code> ... 199,
549, 562, 581, 600, 606, 612, 617,		200, 202, 440, 474, 634, 832, 843,
624, 647, 671, 695, 699, 704, 708,		848, 956, 977, 988, 993, 998, 1003,
715, 725, 764, 804, 808, 813, 817, 824		1038, 1058, 1068, 1073, 1078, 1083
<b>D</b>		<code>\g__kernel_pdfmanagement_end_</code>
dim commands:		<code>run_code_tl</code> ..... 67, 74, 81
<code>\c_zero_dim</code> ..... 1142, 1143, 1144		

```

\g__kernel_pdfmanagement_-
  lastpage_shipout_code_tl . 94, 104
\g__kernel_pdfmanagement_-
  thispage_shipout_code_tl . 90, 100

L
lualua commands:
  \lualua: . . . . . 167, 253, 304, 343

M
mode commands:
  \mode_leave_vertical: . . . . . 1147

P
pdf commands:
  \pdf_object_ref:n . . . . . 834,
    990, 995, 1000, 1005, 1070, 1075,
    1080, 1085, 1161, 1168, 1175, 1183
  \pdf_object_ref_last: . 865, 872, 880
  \pdf_object_unnamed_write:nn . . .
    . . . . . 608, 701, 810, 864, 871, 878
  \pdf_object_write . . . . . 476
pdf internal commands:
  \__pdf_backend:n . . . . .
    . . 137, 468, 477, 880, 1148, 1156,
    1157, 1164, 1171, 1178, 1187, 1202
  \__pdf_backend_bdc:nn . . . . . 12,
    503, 515, 541, 617, 620, 621, 622,
    708, 711, 712, 713, 817, 820, 821, 822
  \__pdf_backend_bdc_contobj:nn . . .
    . . . . . 606, 620, 699, 711, 808, 820
  \__pdf_backend_bdc_contstream:nn
    . . . . . 612, 621, 704, 712, 813, 821
  \__pdf_backend_bdcobject:n . . . . .
    . . . . . 12, 503,
    523, 545, 581, 609, 671, 702, 764, 811
  \__pdf_backend_bdcobject:nn . . . .
    . . . . . 12, 503, 519, 543, 562, 647, 725
  \__pdf_backend_bmc:n . . . . .
    . . . . . 12, 503, 531, 549, 600, 695, 804
  \__pdf_backend_catalog_gput:nn . . 18
  \g__pdf_backend_EmbeddedFiles_-
    int . . . . . 893,
    894, 899, 901, 903, 905, 917, 929
  \__pdf_backend_EmbeddedFiles_-
    name: . . 893, 895, 920, 922, 932, 935
  \g__pdf_backend_EmbeddedFiles_-
    named_prop . . . . 909, 911, 918, 930
  \g__pdf_backend_EmbeddedFiles_-
    seq . . . . . 909, 910, 921
  \__pdf_backend_emc: . . . . .
    . . . . . 12, 503, 527, 547, 624, 715, 824
  \__pdf_backend_luastring:n . . . .
    123, 215, 224, 236, 237, 248, 263, 264

  \g__pdf_backend_name_int . . . . .
    . . . . . 55, 564, 567, 575,
    583, 586, 594, 649, 651, 656, 665,
    673, 675, 680, 689, 727, 729, 766, 768
  \__pdf_backend_NamesEmbeddedFiles_-
    add:n . . . . . 913, 914, 927, 943
  \__pdf_backend_NamesEmbeddedFiles_-
    gpush:n . . . . 862, 869, 876, 886, 890
  \g__pdf_backend_object_int 1123, 1126
  \__pdf_backend_object_last: . . . .
    . . . . . 525, 595, 681, 690, 776, 800
  \__pdf_backend_object_new:nn 402, 463
  \__pdf_backend_object_ref:n 409,
    479, 521, 576, 639, 657, 666, 737, 761
  \__pdf_backend_object_write:nn . .
    . . . . . 442, 464
  \__pdf_backend_Page_gput:nn . . . .
    . . . . . 5, 144, 154, 230, 291, 330, 366
  \__pdf_backend_Page_gremove:n . . .
    . . . . . 5, 144, 161, 244, 298, 337, 372
  \g__pdf_backend_page_int . . . . . 55
  \__pdf_backend_Page_primitive:n .
    . . . . . 4, 144, 147, 207,
    220, 284, 309, 318, 323, 348, 357, 363
  \__pdf_backend_PageResources:n . .
    . . . . . 466, 487, 496
  \c__pdf_backend_PageResources_-
    clist . . 389, 400, 437, 461, 632, 854
  \__pdf_backend_PageResources_-
    gpush:n . . . . .
    . . . . . 12, 503, 535, 551, 630, 720, 838
  \__pdf_backend_PageResources_-
    gpush_aux:n . . . . . 829, 855
  \__pdf_backend_PageResources_-
    gput:nnn 383, 416, 428, 470, 488, 497
  \__pdf_backend_PageResources_-
    obj_gpush: . 383, 435, 483, 492, 501
  \__pdf_backend_Pages_primitive:n
    . . . . . 109, 110, 119, 129, 135, 141
  \__pdf_backend_pdfmark:n . . . . .
    . . . . . 517, 521, 525, 529, 533, 933
  \__pdf_backend_ref_label:nn . . . .
    . . . . . 31, 41, 176, 744, 783
  \__pdf_backend_ref_value:nn . . . .
    . . . . . 37, 42, 184, 752, 791
  \g__pdf_backend_resourceid_int . .
    . . . . . 55, 173, 174, 175,
    176, 180, 183, 184, 731, 736, 741,
    743, 744, 748, 751, 752, 760, 770,
    775, 780, 782, 783, 787, 790, 791, 799
  \__pdf_backend_ThisPage_gpush:n .
    . . . . . 5, 144, 197, 273, 316, 355, 380
  \__pdf_backend_ThisPage_gput:nn .
    . . . . . 5, 144, 170, 256, 307, 346, 377

```



<code>\tex_special:D</code> . . . . .	131, 286, 325	<code>\tl_gput_left:Nn</code> . . . . .	92
<code>\tex_the:D</code> . . . . .	962, 965,	<code>\tl_gput_right:Nn</code> . . . . .	65, 72, 79
968, 1043, 1046, 1049, 1135, 1138, 1141		<code>\tl_if_exist:NTF</code> . . . . .	85
<code>\tex_unexpanded:D</code> . . . . .	217	<code>\tl_new:N</code> . . . . .	52
tl commands:		<code>\tl_set:Nn</code> . . . . .	177, 745, 784
<code>\c_space_tl</code> 567, 575, 586, 594, 651,		<code>\tl_to_str:n</code> . . . . .	
675, 729, 768, 922, 1151, 1152, 1153		.. 961, 964, 967, 1010, 1017, 1023,	
<code>\tl_const:Nn</code> . . . . .	960, 963,	1042, 1045, 1048, 1090, 1098, 1104,	
966, 1041, 1044, 1047, 1133, 1136, 1139		1125, 1134, 1137, 1140, 1195, 1223	