

The l3backend-pdfmanagement package

Additional backend PDF features

L^AT_EX PDF management bundle

The L^AT_EX Project*

Version 0.97d, released 2026-10-06

1 l3backend-pdfmanagement Implementation

```
1 <drivers>\ProvidesExplFile
2 <*dvipdfmx>
3   {l3backend-pdf-dvipdfmx.def}{2026-10-06}{0.97d}
4   {LaTeX~PDF~management~bundle~backend~support:~dvipdfmx}
5 </dvipdfmx>
6 <*dvips>
7   {l3backend-pdf-dvips.def}{2026-10-06}{0.97d}
8   {LaTeX~PDF~management~bundle~backend~support:~dvips}
9 </dvips>
10 <*dvisvgm>
11   {l3backend-pdf-dvisvgm.def}{2026-10-06}{0.97d}
12   {LaTeX~PDF~management~bundle~backend~support:~dvisvgm}
13 </dvisvgm>
14 <*luatex>
15   {l3backend-pdf-luatex.def}{2026-10-06}{0.97d}
16   {LaTeX~PDF~management~bundle~backend~support:~PDF~output~(LuaTeX)}
17 </luatex>
18 <*pdftex>
19   {l3backend-pdf-pdftex.def}{2026-10-06}{0.97d}
20   {LaTeX~PDF~management~bundle~backend~support:~PDF~output~(pdfTeX)}
21 </pdftex>
22 <*xdvipdfmx>
23   {l3backend-pdf-xetex.def}{2026-10-06}{0.97d}
24   {LaTeX~PDF~management~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

```

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 \cs_new_protected:Npn \__kernel_backend_shipout_literal:e #1
40 { \tex_special:D~shipout { #1} }
41 </drivers>

```

(End of definition for __kernel_backend_shipout_literal:e.)

```

42 <*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).

```

43 \cs_new_protected:Npn \__kernel_backend_shipout_literal_pdf:e #1
44 {
45 <*luatex>
46   \tex_pdfextension:D ~ literal ~ shipout ~
47 </luatex>
48 <*pdftex>
49   \tex_pdfliteral:D ~ shipout ~
50 </pdftex>
51   { #1 }
52 }

```

(End of definition for __kernel_backend_shipout_literal_pdf:e.)

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

```

53 \cs_new_protected:Npn \__kernel_backend_shipout_literal_page:e #1
54 {
55 <*luatex>
56   \tex_pdfextension:D ~ literal ~ shipout ~
57 </luatex>
58 <*pdftex>
59   \tex_pdfliteral:D ~ shipout ~

```

```

60 </pdfTeX>
61     page { #1 }
62 }
63 </luatex | pdfTeX>

```

(End of definition for `\__kernel_backend_shipout_literal_page:e`.)

1.3 Crossreferences

Commands to get a reference for the absolute page counter.

```

64 <*drivers>
65 \cs_new_protected:Npn \__pdf_backend_record_abspage:n #1
66 {
67     \@bsphack
68     \property_record:nn{#1}{abspage}
69     \@esphack
70 }
71 \cs_new:Npn \__pdf_backend_ref_abspage:n #1
72 {
73     \property_ref:nn{#1}{abspage}
74 }
75
76 \cs_generate_variant:Nn \__pdf_backend_record_abspage:n {e}
77 \cs_generate_variant:Nn \__pdf_backend_ref_abspage:n {e}
78 </drivers>

```

avoid that destinations names are optimized with xelatex/dvipdfmx see <https://tug.org/piper-mail/dvipdfmx/2019-May/000002.html>

```

79 <dvipdfmx | xdvipdfmx>
80     \__kernel_backend_literal:n { dvipdfmx:config~C~ 0x0010 }
81 </dvipdfmx | xdvipdfmx>

```

```

\g__pdf_tmpa_prop
\l__pdf_tmpa_tl
\l__pdf_backend_tmpa_box

```

Some scratch variables

```

82 <*drivers>
83 \prop_new:N \g__pdf_tmpa_prop
84 \tl_new:N \l__pdf_tmpa_tl
85 \box_new:N \l__pdf_backend_tmpa_box
86 \box_new:N \l__pdf_backend_tmpb_box
87 </drivers>

```

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

```

\g__pdf_backend_resourceid_int
\g__pdf_backend_name_int
\g__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.

```

88 <*drivers>
89 \int_new:N \g__pdf_backend_resourceid_int
90 \int_new:N \g__pdf_backend_name_int
91 \int_new:N \g__pdf_backend_page_int
92 </drivers>

```

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

1.4 luacode

Load the lua code.

```
93 <*luatex>
94     \directlua { require("l3backend-pdf.lua") }
95 </luatex>
```

1.5 Converting unicode strings to a pdfname

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

```
96 <*pdftex | luatex | dvipdfmx | xdvipdfmx | dvisvgm>
97 \cs_new:Npn \__kernel_pdf_name_from_unicode_e:n #1
98 {
99     / \str_convert_pdfname:e { \text_expand:n { #1 } }
100 }
101 </pdftex | luatex | dvipdfmx | xdvipdfmx | dvisvgm>
102 <*dvips>
103 \cs_new:Npn \__kernel_pdf_name_from_unicode_e:n #1
104 {
105     ~ ( \text_expand:n { #1 } ) ~ cvn
106 }
107 </dvips>
```

1.6 Hooks

1.6.1 Add the “end run” hooks

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

```
108 <*pdftex | luatex>
109 % put in \@kernel@after@enddocument@afterlastpage
110 \tl_gput_right:Nn \@kernel@after@enddocument@afterlastpage
111 {
112     \g__kernel_pdfmanagement_end_run_code_tl
113 }
114 </pdftex | luatex>
115 <*dvipdfmx | xdvipdfmx>
116 % put in \@kernel@after@shipout@lastpage
117 \tl_gput_right:Nn \@kernel@after@shipout@lastpage
118 {
119     \g__kernel_pdfmanagement_end_run_code_tl
120 }
121 </dvipdfmx | xdvipdfmx>
122 <*dvips>
123 % put in \@kernel@after@shipout@lastpage
124 \tl_gput_right:Nn \@kernel@after@shipout@lastpage
125 {
126     \g__kernel_pdfmanagement_end_run_code_tl
127 }
128 </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 ...

```

129 <*drivers>
130 \tl_if_exist:NTF \@kernel@after@shipout@background
131 {
132   \g@addto@macro \@kernel@before@shipout@background{\relax}
133   \g@addto@macro \@kernel@after@shipout@background
134   {
135     \g__kernel_pdfmanagement_thispage_shipout_code_tl
136   }
137 }
138 {
139   \hook_gput_code:nnn{shipout/background}{pdf}
140   {
141     \g__kernel_pdfmanagement_thispage_shipout_code_tl
142   }
143 }
144
145 </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 dvipdfmx are additive. luatex sets it in lua. The higher level code has to take this into account.

```

146 <*pdftex>
147 \cs_new_protected:Npn \_pdf_backend_Pages_primitive:n #1
148 {
149   \tex_global:D \tex_pdfpagesattr:D { #1 }
150 }
151 </pdftex>
152 <*luatex>
153 %luatex: does it in lua
154 \cs_new_protected:Npn \_pdf_backend_Pages_primitive:n #1
155 {
156   \tex_directlua:D
157   {
158     pdf.setpagesattributes( \_pdf_backend_luastring:n { #1 } )
159   }
160 }
161 </luatex>
162 <*dvips>
163 \cs_new_protected:Npx \_pdf_backend_Pages_primitive:n #1
164 {
165   \tex_special:D{ps:~[#1~/PAGES~pdfmark} %]
166 }
167 </dvips>
168 <*dvipdfmx | xdvipdfmx>
169 \cs_new_protected:Npn \_pdf_backend_Pages_primitive:n #1

```

```

170 {
171   \__pdf_backend:n{put~@pages~<<#1>>}
172 }
173 </dvipdfmx | xdvipdfmx>
174 <*dvisvgm>
175 \cs_new_protected:Npn \__pdf_backend_Pages_primitive:n #1
176 {}
177 </dvisvgm>

```

(End of definition for __pdf_backend_Pages_primitive:n.)

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

__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.

```

178 % backend commands
179 <*pdftex>
180 %the primitive
181 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
182 {
183   \tex_global:D \tex_pdfpageattr:D { #1 }
184 }
185 % the command to store default values.
186 % Uses a prop with pdflatex + dvi,
187 % sets a lua table with luatex
188 \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2 %key,value
189 {
190   \pdfdict_gput:nnn {g__pdf_Core/Page}{ #1 }{ #2 }
191 }
192 % the command to remove a default value.
193 % Uses a prop with pdflatex + dvi,
194 % changes a lua table with luatex
195 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
196 {
197   \pdfdict_gremove:nn {g__pdf_Core/Page}{ #1 }
198 }
199 % the command used in the document.
200 % direct call of the primitive special with dvips/dvipdfmx
201 % \latelua: fill a page related table with luatex, merge it with the page
202 % table and push it directly
203 % write to aux and store in prop with pdflatex
204 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
205 {
206   %we need to know the page the resource should be added too.
207   \int_gincr:N\g__pdf_backend_resourceid_int
208   \__pdf_backend_record_abspace:e { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
209   \tl_set:Ne \l__pdf_tmpa_tl

```

```

210     {
211       \__pdf_backend_ref_abspage:e {l3pdf\int_use:N\g__pdf_backend_resourceid_int}
212     }
213     \pdfdict_if_exist:nF { g__pdf_Core/backend_Page\l__pdf_tmpa_tl}
214     {
215       \pdfdict_new:n { g__pdf_Core/backend_Page\l__pdf_tmpa_tl}
216     }
217     %backend_Page has no handler.
218     \pdfdict_gput:nnn {g__pdf_Core/backend_Page\l__pdf_tmpa_tl}{ #1 }{ #2 }
219   }
220   %the code to push the values, used in shipout
221   %merges the two props and then fills the register in pdflatex
222   %merges the two tables and then fills (in lua) in luatex
223   %issues the values stored in the global prop with dvi
224   \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
225   {
226     \prop_gset_eq:Nc \g__pdf_tmpa_prop { \__kernel_pdfdict_name:n { g__pdf_Core/Page } }
227     \prop_if_exist:cT { \__kernel_pdfdict_name:n { g__pdf_Core/backend_Page#1 } }
228     {
229       \prop_map_inline:cn { \__kernel_pdfdict_name:n { g__pdf_Core/backend_Page#1 } }
230       {
231         \prop_gput:Nnn \g__pdf_tmpa_prop { ##1 }{ ##2 }
232       }
233     }
234     \__pdf_backend_Page_primitive:e
235     {
236       \prop_map_function:NN \g__pdf_tmpa_prop \pdfdict_item:ne
237     }
238   }
239   </pdftex>
240   <*luatex>
241   % do we need to use some escaping for the values????
242   \cs_new:Npn \__pdf_backend_luastring:n #1
243   {
244     "\tex_luaescapestring:D { \tex_unexpanded:D { #1 } }"
245   }
246   %not used, only there for consistency
247   \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
248   {
249     \tex_latelua:D
250     {
251       pdf.setpageattributes(\__pdf_backend_luastring:n { #1 })
252     }
253   }
254   % the command to store default values.
255   % Uses a prop with pdflatex + dvi,
256   % sets a lua table with luatex
257   \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2
258   {
259     \tex_directlua:D
260     {
261       ltx.__pdf.backend_Page_gput
262       (
263         \__pdf_backend_luastring:n { #1 },

```

```

264         \__pdf_backend_luastring:n { #2 }
265     )
266 }
267 }
268 % the command to remove a default value.
269 % Uses a prop with pdflatex + dvi,
270 % changes a lua table with luatex
271 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
272 {
273     \tex_directlua:D
274     {
275         ltx.__pdf.backend_Page_gremove (\__pdf_backend_luastring:n { #1 })
276     }
277 }
278 % the command used in the document.
279 % direct call of the primitive special with dvips/dvipdfmx
280 % \latelua: fill a page related table with luatex, merge it with the page
281 % table and push it directly
282 % write to aux and store in prop with pdflatex
283 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
284 {
285     \tex_latelua:D
286     {
287         ltx.__pdf.backend_ThisPage_gput
288         (
289             tex.count["g_shipout_readonly_int"],
290             \__pdf_backend_luastring:n { #1 },
291             \__pdf_backend_luastring:n { #2 }
292         )
293         ltx.__pdf.backend_ThisPage_gpush (tex.count["g_shipout_readonly_int"])
294     }
295 }
296 %the code to push the values, used in shipout
297 %merges the two props and then fills the register in pdflatex
298 %merges the two tables (the one is probably still empty) and then fills (in lua) in luatex
299 %issues the values stored in the global prop with dvi
300 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
301 {
302     \tex_latelua:D
303     {
304         ltx.__pdf.backend_ThisPage_gpush (tex.count["g_shipout_readonly_int"])
305     }
306 }
307
308 </luatex>
309 <*dvipdfmx | xdvipdfmx>
310 %the primitive
311 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
312 {
313     \tex_special:D{pdf:-put~@thispage-<<#1>>}
314 }
315 % the command to store default values.
316 % Uses a prop with pdflatex + dvi,
317 % sets a lua table with luatex

```

```

318 \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2
319 {
320   \pdfdict_gput:nnn {g__pdf_Core/Page}{ #1 }{ #2 }
321 }
322 % the command to remove a default value.
323 % Uses a prop with pdflatex + dvi,
324 % changes a lua table with luatex
325 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
326 {
327   \pdfdict_gremove:nn {g__pdf_Core/Page}{ #1 }
328 }
329 % the command used in the document.
330 % direct call of the primitive special with dvips/dvipdfmx
331 % \latelua: fill a page related table with luatex, merge it with the page
332 % table and push it directly
333 % write to aux and store in prop with pdflatex
334 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
335 {
336   \__pdf_backend_Page_primitive:n { /#1~#2 }
337 }
338 %the code to push the values, used in shipout
339 %merges the two props and then fills the register in pdflatex
340 %merges the two tables (the one is probably still empty)
341 % and then fills (in lua) in luatex
342 %issues the values stored in the global prop with dvi
343 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
344 {
345   \__pdf_backend_Page_primitive:e
346   { \pdfdict_use:n { g__pdf_Core/Page} }
347 }
348 </dvipdfmx | xdvipdfmx>
349 <*dvips>
350 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
351 {
352   \tex_special:D{ps:-[ThisPage]<<#1>>~/PUT~pdfmark} %]
353 }
354 % the command to store default values.
355 % Uses a prop with pdflatex + dvi,
356 % sets a lua table with luatex
357 \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2
358 {
359   \pdfdict_gput:nnn {g__pdf_Core/Page}{ #1 }{ #2 }
360 }
361 % the command to remove a default value.
362 % Uses a prop with pdflatex + dvi,
363 % changes a lua table with luatex
364 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
365 {
366   \pdfdict_gremove:nn {g__pdf_Core/Page}{ #1 }
367 }
368 % the command used in the document.
369 % direct call of the primitive special with dvips/dvipdfmx
370 % \latelua: fill a page related table with luatex, merge it with the page
371 % table and push it directly

```

```

372 % write to aux and store in prop with pdflatex
373 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
374 {
375   \__pdf_backend_Page_primitive:n { /#1~#2 }
376 }
377 %the code to push the values, used in shipout
378 %merges the two props and then fills the register in pdflatex
379 %merges the two tables (the one is probably still empty)
380 %and then fills (in lua) in luatex
381 %issues the values stored in the global prop with dvi
382 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
383 {
384   \__pdf_backend_Page_primitive:e
385   { \pdfdict_use:n { g__pdf_Core/Page} }
386 }
387 </dvips>
388 <*dvisvgm>
389 % mostly only dummies ...
390 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
391 {}
392 % Uses a prop with pdflatex + dvi,
393 \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2
394 {
395   \pdfdict_gput:nnn {g__pdf_Core/Page}{ #1 }{ #2 }
396 }
397 % the command to remove a default value.
398 % Uses a prop with pdflatex + dvi,
399 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
400 {
401   \pdfdict_gremove:nn {g__pdf_Core/Page}{ #1 }
402 }
403 % the command used in the document.
404 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
405 {}
406 %the code to push the values, used in shipout
407 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
408 {}
409 </dvisvgm>
410 <*drivers>
411 \cs_generate_variant:Nn \__pdf_backend_Page_primitive:n { e }
412 </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.

```

413 <*drivers>
414 \clist_const:Nn \c__pdf_backend_PageResources_clist

```

```

415 {
416   ExtGState,
417   ColorSpace,
418   Pattern,
419   Shading,
420 }
421 </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 is currently issued in the end-of-run hook! create the backend objects:

`\__pdf_backend_PageResources_obj_gpush:`

```

422 <*pdfTeX | luatex>
423 \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
424 {
425   \pdf_object_new:n {\__pdf/Page/Resources/#1}
426   \cs_if_exist:NT \tex_directlua:D
427   {
428     \tex_directlua:D
429     {
430       ltx.__pdf.object["__pdf/Page/Resources/#1"]
431       =
432       "\pdf_object_ref:n{\__pdf/Page/Resources/#1}"
433     }
434   }
435 }
436 </pdfTeX | luatex>

```

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

```

437 <*luatex>
438 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
439 {
440   \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
441   \tex_directlua:D{ltx.__pdf.Page.Resources.#1=true}
442   \tex_directlua:D
443   {
444     ltx.pdf.Page.Resources_gpush(tex.count["g_shipout_readonly_int"])
445   }
446 }
447 </luatex>
448 <*pdfTeX>
449 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
450 {
451   \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
452 }
453 </pdfTeX>

```

code for end of document code. Change 2026-03-11: we push out the objects even if they are empty as tikz references them in X-objects (fadings)

```

454 <*pdftex | luatex>
455 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush:
456 {
457   \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
458   {
459     \pdf_object_write:nne
460     { __pdf/Page/Resources/##1 } { dict }
461     { \pdfdict_use:n { g__pdf_Core/Page/Resources/##1 } }
462   }
463 }
464 </pdftex | luatex>

```

xdvipdfmx doesn't work correctly with object names ... <https://tug.org/piper-mail/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.

```

465 <*dvipdfmx | xdvipdfmx>
466 <xdvipdfmx> \hook_gset_rule:nnnn{shipout/firstpage}{l3backend-xetex}{after}{pdf}
467 <dvipdfmx> \hook_gset_rule:nnnn{shipout/firstpage}{l3backend-dvipdfmx}{after}{pdf}
468 %
469 \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
470 {
471   \pdf_object_new:n { __pdf/Page/Resources/#1 }
472   \hook_gput_code:nnn
473   {shipout/firstpage}
474   {pdf}
475   {\pdf_object_write:nnn { __pdf/Page/Resources/#1 } { dict } {}}
476 }
477 \cs_new_protected:Npn \__pdf_backend_PageResources:n #1
478 {
479   \__pdf_backend:n {put~@resources~<<#1>>}
480 }
481 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
482 {
483   % this is not used for output, but there is a test if the resource is empty
484   \prop_gput:cne { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/#1 } }
485   { \str_convert_pdfname:n {#2} }{ #3 }
486   %objects are not filled with \pdf_object_write as this is not additive!
487   \__pdf_backend:e
488   {
489     put~\pdf_object_ref:n {__pdf/Page/Resources/#1}<<#2~#3>>
490   }
491 }
492
493 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush: {}
494 </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.

```

495 <*dvips>

```

```

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   \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
500 }
501 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush: {}
502 </dvips>

```

dvipsvgm unneeded, or no-op

```

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

```

(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_bdcobject:n, __pdf_backend_bmc:n and __pdf_backend_emc: are 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 resources for the current page. __pdf_backend_bmc:n __pdf_backend_emc: pdftex and luatex (and perhaps dvips ...) need to know if there are in a xform stream __pdf_backend_PageResources_gpush:n ...

```

511 <*drivers>
512 \bool_new:N \l__pdf_backend_xform_bool
513 </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>

```

514 <*dvips>
515 %
516 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2 % #1 eg. Span, #2: dict_content
517 {
518   \__pdf_backend_pdfmark:n{/#1~<<#2>>~/BDC}
519 }

```

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

```

520 \cs_set_eq:NN \__pdf_backend_bdc_contobj:nn \__pdf_backend_bdc:nn
521 \cs_set_eq:NN \__pdf_backend_bdc_contstream:nn \__pdf_backend_bdc:nn
522
523 \cs_new_protected:Npn \__pdf_backend_bdc_shipout:ee #1 #2 % #1 eg. Span, #2: dict_content
524 {
525   \__kernel_backend_shipout_literal:e
526   {ps: SDict ~ begin ~ mark /#1~<<#2>>~/BDC ~ pdfmark ~ end }
527 }
528

```

```

529 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
530 {
531   \__pdf_backend_pdfmark:e{/#1~\pdf_object_ref:n{#2}~/BDC}
532 }
533 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1 % #1 eg. Span,
534 {
535   \__pdf_backend_pdfmark:e{/#1~\__pdf_backend_object_last:~/BDC}
536 }
537 \cs_set_protected:Npn \__pdf_backend_emc:
538 {
539   \__pdf_backend_pdfmark:n{/EMC} %
540 }
541 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
542 {
543   \__pdf_backend_pdfmark:n{/#1~/BMC} %
544 }
545 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1 {}
546
547 </dvips>
548 <*dvisvgm>
549 % dvisvgm should do nothing
550 %
551 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2 % #1 eg. Span, #2: dict_content
552 {}
553 \cs_set_eq:NN \__pdf_backend_bdc_contobj:nn \__pdf_backend_bdc:nn
554 \cs_set_eq:NN \__pdf_backend_bdc_contstream:nn \__pdf_backend_bdc:nn
555
556 \cs_set_protected:Npn \__pdf_backend_shipout_bdc:ee #1 #2 % #1 eg. Span, #2: dict_content
557 {}
558 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
559 {}
560 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1 % #1 eg. Span,
561 {}
562 \cs_set_protected:Npn \__pdf_backend_emc:
563 {}
564 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
565 {}
566 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1 {}
567
568 </dvisvgm>
569 %
570 % xetex has to create the entries in the /Properties manually
571 % (like the other backends)
572 % use pdfbase special
573 % https://chat.stackexchange.com/transcript/message/50832016#50832016
574 % the property is added to xform resources automatically,
575 % no need to worry about it.
576 <*dvipdfmx|xvipdfmx>
577 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
578 {
579   \int_gincr:N \g__pdf_backend_name_int
580   \__kernel_backend_literal:e
581   {
582     pdf:code~/#1/l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC

```

```

583     }
584     \__kernel_backend_literal:e
585     {
586         pdf:put~@resources~
587         <<
588         /Properties~
589         <<
590         /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_t1
591         \pdf_object_ref:n { #2 }
592         >>
593     >>
594     }
595 }
596 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1 % #1 eg. Span
597 {
598     \int_gincr:N \g__pdf_backend_name_int
599     \__kernel_backend_literal:e
600     {
601         pdf:code~/\exp_not:n{#1}/l3pdf\int_use:N\g__pdf_backend_name_int\c_space_t1 BDC
602     }
603     \__kernel_backend_literal:e
604     {
605         pdf:put~@resources~
606         <<
607         /Properties~
608         <<
609         /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_t1
610         \__pdf_backend_object_last:
611         >>
612     >>
613     }
614 }
615 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
616 {
617     \__kernel_backend_literal:n {pdf:code~/#1~BMC} %pdfbase
618 }
619
620 %this require management
621 \cs_set_protected:Npn \__pdf_backend_bdc_contobj:nn #1 #2
622 {
623     \pdf_object_unnamed_write:nn { dict }{ #2 }
624     \__pdf_backend_bdcobject:n { #1 }
625 }
626
627 \cs_set_protected:Npn \__pdf_backend_bdc_contstream:nn #1 #2
628 {
629     \__kernel_backend_literal:n {pdf:code~ /#1~<<#2>>~BDC }
630 }
631
632 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2
633 {
634     \cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contobj:nn
635     \__pdf_backend_bdc:nn {#1}{#2}
636 }

```

```

637
638 \cs_set_protected:Npn \__pdf_backend_bdc_shipout_contstream:ee #1 #2
639 {
640   \__kernel_backend_shipout_literal:e {pdf:code~ /#1~<<#2>>~BDC }
641 }
642 \cs_set_eq:NN \__pdf_backend_bdc_shipout:ee \__pdf_backend_bdc_shipout_contstream:ee
643
644 \cs_set_protected:Npn \__pdf_backend_emc:
645 {
646   \__kernel_backend_literal:n {pdf:code~EMC} %pdfbase
647 }
648 % properties are handled automatically, but the other resources should be added
649 % at shipout
650 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1
651 {
652   \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
653   {
654     \prop_if_empty:cF { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/##1} }
655     {
656       \__kernel_backend_literal:e
657       {
658         pdf:put~@resources~
659         <</##1~\pdf_object_ref:n {__pdf/Page/Resources/##1}>>
660       }
661     }
662   }
663 }
664 </dvipdfmx|xdvipdfmx>
665 % luatex + pdftex
666 <*luatex>
667 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
668 {
669   \int_gincr:N \g__pdf_backend_name_int
670   \__kernel_backend_literal_page:e
671   { /#1 ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
672   \bool_if:NTF \l__pdf_backend_xform_bool
673   {
674     \pdfdict_gput:nee
675     { g__pdf_Core/Xform/Resources/Properties }
676     { l3pdf\int_use:N\g__pdf_backend_name_int }
677     { \pdf_object_ref:n { #2 } }
678   }
679   {
680     \exp_args:Ne \tex_latelua:D
681     {
682       ltx.pdf.Page_Resources_Properties_gput
683       (
684         tex.count["g_shipout_readonly_int"],
685         "l3pdf\int_use:N\g__pdf_backend_name_int",
686         "\pdf_object_ref:n { #2 }"
687       )
688     }
689   }
690 }

```

```

691 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1% #1 eg. Span
692 {
693   \int_gincr:N \g__pdf_backend_name_int
694   \__kernel_backend_literal_page:e
695   { /\exp_not:n{#1} ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
696   \bool_if:NTF \l__pdf_backend_xform_bool
697   {
698     \pdfdict_gput:nee %no handler needed
699     { g__pdf_Core/Xform/Resources/Properties }
700     { l3pdf\int_use:N\g__pdf_backend_name_int }
701     { \__pdf_backend_object_last: }
702   }
703   {
704     \exp_args:Ne \tex_latelua:D
705     {
706       ltx.pdf.Page_Resources_Properties_gput
707       (
708         tex.count["g_shipout_readonly_int"],
709         "l3pdf\int_use:N\g__pdf_backend_name_int",
710         "\__pdf_backend_object_last:"
711       )
712     }
713   }
714 }
715 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
716 {
717   \__kernel_backend_literal_page:n { /#1~BMC }
718 }
719 \cs_set_protected:Npn \__pdf_backend_bdc_contobj:nn #1 #2
720 {
721   \pdf_object_unnamed_write:nn { dict } { #2 }
722   \__pdf_backend_bdcobject:n { #1 }
723 }
724 \cs_set_protected:Npn \__pdf_backend_bdc_contstream:nn #1 #2
725 {
726   \__kernel_backend_literal_page:n { /#1~<<#2>>~BDC }
727 }
728
729 \cs_set_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contstream:nn
730
731 \cs_set_protected:Npn \__pdf_backend_bdc_shipout_contstream:ee #1 #2
732 {
733   \__kernel_backend_shipout_literal_page:e { /#1~<<#2>>~BDC }
734 }
735 \cs_set_eq:NN \__pdf_backend_bdc_shipout:ee \__pdf_backend_bdc_shipout_contstream:ee
736
737 \cs_set_protected:Npn \__pdf_backend_emc:
738 {
739   \__kernel_backend_literal_page:n { EMC }
740 }
741
742 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1 {}
743 </luatex>

```

pdf_latex 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

```

744 <*pdftex>
745 \csset_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
746 {
747   \int_gincr:N \g__pdf_backend_name_int
748   \__kernel_backend_literal_page:e
749   { /#1 ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
750   % code to set the property ....
751   \int_gincr:N\g__pdf_backend_resourceid_int
752   \bool_if:NTF \l__pdf_backend_xform_bool
753   {
754     \pdfdict_gput:nee %no handler needed
755     { g__pdf_Core/Xform/Resources/Properties }
756     { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
757     { \pdf_object_ref:n { #2 } }
758   }
759   {
760     \__pdf_backend_record_abspage:e {l3pdf\int_use:N\g__pdf_backend_resourceid_int}
761     \tl_set:Ne \l__pdf_tmpa_tl
762     {
763       \__pdf_backend_ref_abspage:e{l3pdf\int_use:N\g__pdf_backend_resourceid_int}
764     }
765     \pdfdict_if_exist:nF { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
766     {
767       \pdfdict_new:n { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
768     }
769     \pdfdict_gput:nee
770     { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
771     { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
772     { \pdf_object_ref:n{#2} }
773   }
774 }
775 \csset_protected:Npn \__pdf_backend_bdcobject:n #1% #1 eg. Span
776 {
777   \int_gincr:N \g__pdf_backend_name_int
778   \__kernel_backend_literal_page:e
779   { /\exp_not:n{#1} ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
780   % code to set the property ....
781   \int_gincr:N\g__pdf_backend_resourceid_int
782   \bool_if:NTF \l__pdf_backend_xform_bool
783   {
784     \pdfdict_gput:nee
785     { g__pdf_Core/Xform/Resources/Properties }
786     { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
787     { \__pdf_backend_object_last: }
788   }
789   {
790     \__pdf_backend_record_abspage:e{l3pdf\int_use:N\g__pdf_backend_resourceid_int}
791     \tl_set:Ne \l__pdf_tmpa_tl
792     {
793       \__pdf_backend_ref_abspage:e{l3pdf\int_use:N\g__pdf_backend_resourceid_int}
794     }
795     \pdfdict_if_exist:nF { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }

```

```

796         {
797             \pdfdict_new:n { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
798         }
799         \pdfdict_gput:nee
800         { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
801         { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
802         { \__pdf_backend_object_last: }
803         %\pdfdict_show:n { g_backend_Page\l__pdf_tmpa_tl/Resources/Properties }
804     }
805 }
806 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
807 {
808     \__kernel_backend_literal_page:n { /#1~BMC }
809 }
810 \cs_set_protected:Npn \__pdf_backend_bdc_contobj:nn #1 #2
811 {
812     \pdf_object_unnamed_write:nn { dict } { #2 }
813     \__pdf_backend_bdcobject:n { #1 }
814 }
815 \cs_set_protected:Npn \__pdf_backend_bdc_contstream:nn #1 #2
816 {
817     \__kernel_backend_literal_page:n { /#1~<<#2>>~BDC }
818 }

```

We use by default the direct BDC.

```

819 \cs_set_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contstream:nn
820
821 \cs_set_protected:Npn \__pdf_backend_bdc_shipout_contstream:ee #1 #2
822 {
823     \__kernel_backend_shipout_literal_page:e { /#1~<<#2>>~BDC }
824 }
825 \cs_set_eq:NN \__pdf_backend_bdc_shipout:ee \__pdf_backend_bdc_shipout_contstream:ee
826
827
828 \cs_set_protected:Npn \__pdf_backend_emc:
829 {
830     \__kernel_backend_literal_page:n { EMC }
831 }
832
833 \cs_new:Npn \__pdf_backend_PageResources_gpush_aux:n #1 %#1 ExtGState etc
834 {
835     \prop_if_empty:cF
836     { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/#1 } }
837     {
838         \pdfdict_item:ne { #1 } { \pdf_object_ref:n { __pdf/Page/Resources/#1 } }
839     }
840 }
841
842 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1
843 {
844     \exp_args:NNe \tex_global:D \tex_pdfpageresources:D
845     {
846         \prop_if_exist:cT
847         { \__kernel_pdfdict_name:n { g__pdf_Core/backend_Page#1/Resources/Properties } }
848         {

```

```

849         /Properties~
850         <<
851         \prop_map_function:cN
852         { \__kernel_pdffdict_name:n { g__pdf_Core/backend_Page#1/Resources/Properties
853         \pdffdict_item:ne
854         >>
855     }
856     %% add ExtGState etc
857     \clist_map_function:NN
858     \c__pdf_backend_PageResources_clist
859     \__pdf_backend_PageResources_gpunch_aux:n
860 }
861 }
862
863 </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.

```

864 % pdfLaTeX
865 <*pdfTeX>
866 \cs_new_protected:Npn \__pdf_backend_Names_gpunch:nn #1 #2 % #1 name of name tree, #2 array of
867 {
868     \pdf_object_unnamed_write:nn {dict} {/Names [#2] }
869     \tex_pdfnames:D {/#1~\pdf_object_ref_last:}
870 }
871 </pdfTeX>
872 <*LaTeX>
873 \cs_new_protected:Npn \__pdf_backend_Names_gpunch:nn #1 #2 % #1 name of name tree, #2 array of
874 {
875     \pdf_object_unnamed_write:nn {dict} {/Names [#2] }
876     \tex_pdfextension:D~names~ {/#1~\pdf_object_ref_last:}
877 }
878 </LaTeX>
879 <*dvipdfmx | xdvipdfmx>
880 \cs_new_protected:Npn \__pdf_backend_Names_gpunch:nn #1 #2 % #1 name of name tree, #2 array of
881 {
882     \pdf_object_unnamed_write:nn {dict} {/Names [#2] }
883     \__pdf_backend:e {put~@names~</#1~\pdf_object_ref_last: >>}
884 }
885 </dvipdfmx | xdvipdfmx>
886
887 %dvips: noop
888 <*dvips>

```

```

889 \cs_new_protected:Npn \__pdf_backend_Names_gpush:nn #1 #2 {}
890 </dvips>
891 %dvisvgm: noop
892 <*dvisvgm>
893 \cs_new_protected:Npn \__pdf_backend_Names_gpush:nn #1 #2 {}
894 </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.

```

895 <*pdftex | luatex | dvipdfmx | xdvipdfmx>
896 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_add:nn #1 #2 {}
897 </pdftex | luatex | dvipdfmx | xdvipdfmx>
898 <*dvips>
899 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_add:nn #1 #2
900 {
901     \__pdf_backend_pdfmark:e
902     {
903         /Name~#1~
904         /FS~#2~
905         /EMBED
906     }
907 }
908 </dvips>
909 <*dvisvgm>
910 %no op. Or is there any sensible use for it?
911 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_add:nn #1 #2
912 {}
913
914 </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.

```

915 <*pdftex>
916 \cs_if_exist:NT \pdfrunninglinkoff
917 {
918     \cs_set_protected:Npn \__pdfannot_backend_link_off:
919     {
920         \pdfrunninglinkoff
921     }
922     \cs_set_protected:Npn \__pdfannot_backend_link_on:
923     {
924         \pdfrunninglinkon
925     }
926 }
927 </pdftex>

```

1.10.3 Split links

With luatex we use luacode to handle link annotations. This allows us to retrieve the annotations as needed by the tagging code. It also allow to prevent that link areas spill over into unwanted regions like footnotes. See the documentation in lualinksplit.lua for more details. We therefore add a plug for the build/column/footnotes. We use a negative number as link state as this ensures that the linkstate is automatically reverted at the end of the box, see the documentation in lualinksplit.lua.

```

928 <*luatex>
929 \lua_load_module:n{lualinksplit}
930 \NewSocketPlug{build/column/footnotes}{lualinksplit}{%
931   \setbox\footins=\vbox{\pdfextension linkstate-2\unvbox\footins}%
932 }
933 \AssignSocketPlug{build/column/footnotes}{lualinksplit}
934 </luatex>

```

1.10.4 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!

```

We want to be able to suppress the global resources, for this we used booleans

```

\__pdf_backend_xform_use:n
\__pdf_backend_xform_ref:n

935 <*drivers>
936 \bool_new:N\l__pdf_backend_xform_res_pattern_bool
937 \bool_new:N\l__pdf_backend_xform_res_extgstate_bool
938 \bool_new:N\l__pdf_backend_xform_res_properties_bool
939 \bool_new:N\l__pdf_backend_xform_res_colorspace_bool
940 \bool_new:N\l__pdf_backend_xform_res_shading_bool
941 \cs_new_protected:Npn \__pdf_backend_xform_res_reset:
942 {
943   \bool_set_true:N\l__pdf_backend_xform_res_pattern_bool
944   \bool_set_true:N\l__pdf_backend_xform_res_extgstate_bool
945   \bool_set_true:N\l__pdf_backend_xform_res_properties_bool
946   \bool_set_true:N\l__pdf_backend_xform_res_colorspace_bool
947   \bool_set_true:N\l__pdf_backend_xform_res_shading_bool
948 }
949 \__pdf_backend_xform_res_reset:
950 </drivers>
951 <*pdftex>
952 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnnn #1 #2 #3 #4 #5
953 % #1 name
954 % #2 attributes
955 % #3 resources
956 % #4 content, not necessarily a box!
957 % #5 a clist to exclude resources
958 {
959   \hbox_set:Nn \l__pdf_backend_tmpa_box
960   {
961     \bool_set_true:N \l__pdf_backend_xform_bool
962     \prop_gc_clear:c {\__kernel_pdftdict_name:n { g__pdf_Core/Xform/Resources/Properties }}
963     #4

```

```

964     }
965     %store the dimensions
966     \tl_const:ce
967     { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
968     { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
969     \tl_const:ce
970     { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
971     { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
972     \tl_const:ce
973     { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
974     { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
975     \clist_map_inline:nn {#5}
976     { \bool_set_false:c { l__pdf_backend_xform_res_##1_bool } }
977     %% do we need to test if #2 and #3 are empty??
978     \tex_immediate:D \tex_pdfxform:D
979     ~ attr ~ { #2 }
980     %% which other resources should be default? Is an argument actually needed?
981     ~ resources ~
982     {
983     #3
984     \bool_if:NT \l__pdf_backend_xform_res_properties_bool
985     {
986         \int_compare:nNnT
987         { \prop_count:c { \__kernel_pdffdict_name:n { g__pdf_Core/Xform/Resources/Properties } }
988         >
989         { 0 }
990         {
991             /Properties~
992             <<
993             \pdffdict_use:n { g__pdf_Core/Xform/Resources/Properties }
994             >>
995         }
996     }
997     \bool_if:NT \l__pdf_backend_xform_res_extgstate_bool
998     {
999         \prop_if_empty:cF
1000         { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/ExtGState } }
1001         {
1002             /ExtGState~ \pdf_object_ref:n { __pdf/Page/Resources/ExtGState }
1003         }
1004     }
1005     \bool_if:NT \l__pdf_backend_xform_res_pattern_bool
1006     {
1007         \prop_if_empty:cF
1008         { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/Pattern } }
1009         {
1010             /Pattern~ \pdf_object_ref:n { __pdf/Page/Resources/Pattern }
1011         }
1012     }
1013     \bool_if:NT \l__pdf_backend_xform_res_shading_bool
1014     {
1015         \prop_if_empty:cF
1016         { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/Shading } }
1017         {

```

```

1018         /Shading~ \pdf_object_ref:n { __pdf/Page/Resources/Shading }
1019     }
1020 }
1021 \bool_if:NT \l__pdf_backend_xform_res_colorspace_bool
1022 {
1023     \prop_if_empty:cF
1024     { \__kernel_pdftdict_name:n { g__pdf_Core/Page/Resources/ColorSpace } }
1025     {
1026         /ColorSpace~ \pdf_object_ref:n { __pdf/Page/Resources/ColorSpace }
1027     }
1028 }
1029 }
1030 \l__pdf_backend_tmpa_box
1031 \__pdf_backend_xform_res_reset:
1032 \int_const:cn
1033 { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1034 { \tex_pdflastxform:D }
1035 }
1036
1037 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
1038 {
1039     \tex_pdfrefxform:D
1040     \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1041     \scan_stop:
1042 }
1043
1044 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1045 {
1046     \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int } ~ 0 ~ R
1047 }
1048 </pdftex>
1049 <*luatex>
1050 %luatex
1051 %nearly identical but not completely ...
1052 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnnn #1 #2 #3 #4 #5
1053 % #1 name
1054 % #2 attributes
1055 % #3 resources
1056 % #4 content, not necessarily a box!
1057 % #5 a clist to exclude resources
1058 {
1059     \hbox_set:Nn \l__pdf_backend_tmpa_box
1060     {
1061         \bool_set_true:N \l__pdf_backend_xform_bool
1062         \prop_gclear:c { \__kernel_pdftdict_name:n { g__pdf_Core/Xform/Resources/Properties }
1063         #4
1064     }
1065     \tl_const:ce
1066     { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1067     { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
1068     \tl_const:ce
1069     { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1070     { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
1071     \tl_const:ce

```

```

1072     { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1073     { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
1074 \clist_map_inline:nn {#5}
1075   { \bool_set_false:c { l__pdf_backend_xform_res_##1_bool } }
1076   %% do we need to test if #2 and #3 are empty??
1077 \tex_immediate:D \tex_pdfxform:D
1078   ~ attr ~ { #2 }
1079   %% which resources should be default? Is an argument actually needed?
1080   ~ resources ~
1081   {
1082     #3
1083     \bool_if:NT \l__pdf_backend_xform_res_properties_bool
1084     {
1085       \int_compare:nNnT
1086         { \prop_count:c { \__kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Properties } }
1087         >
1088         { 0 }
1089         {
1090           /Properties~
1091           <<
1092             \pdfdict_use:n { g__pdf_Core/Xform/Resources/Properties }
1093           >>
1094         }
1095     }
1096     \bool_if:NT \l__pdf_backend_xform_res_extgstate_bool
1097     {
1098       \prop_if_empty:cF
1099       { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/ExtGState } }
1100       {
1101         /ExtGState~ \pdf_object_ref:n { __pdf/Page/Resources/ExtGState }
1102       }
1103     }
1104     \bool_if:NT \l__pdf_backend_xform_res_pattern_bool
1105     {
1106       \prop_if_empty:cF
1107       { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/Pattern } }
1108       {
1109         /Pattern~ \pdf_object_ref:n { __pdf/Page/Resources/Pattern }
1110       }
1111     }
1112     \bool_if:NT \l__pdf_backend_xform_res_shading_bool
1113     {
1114       \prop_if_empty:cF
1115       { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/Shading } }
1116       {
1117         /Shading~ \pdf_object_ref:n { __pdf/Page/Resources/Shading }
1118       }
1119     }
1120     \bool_if:NT \l__pdf_backend_xform_res_colorspace_bool
1121     {
1122       \prop_if_empty:cF
1123       { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/ColorSpace } }
1124       {
1125         /ColorSpace~ \pdf_object_ref:n { __pdf/Page/Resources/ColorSpace }

```

```

1126     }
1127   }
1128 }
1129 \l__pdf_backend_tmpa_box
1130 \__pdf_backend_xform_res_reset:
1131 \int_const:cn
1132   { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1133   { \tex_pdflastxform:D }
1134 }
1135
1136 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1 %protected as with xelatex
1137 {
1138   \tex_pdfrefxform:D \int_use:c
1139   {
1140     c__pdf_backend_xform_ \tl_to_str:n {#1} _int
1141   }
1142   \scan_stop:
1143 }
1144
1145 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1146 { \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int } ~ 0 ~ R }
1147
1148 </luatex>
1149 <*dvipdfmx|xdvipdfmx>
1150 % xetex
1151   % it needs a bit testing if it really works to set the box to 0 before the special ...
1152   % does it disturb viewing the xobject?
1153   % what happens with the resources (bdc)? (should work as they are specials too)
1154   % xetex requires that the special is in horizontal mode. This means it affects
1155   % typesetting. But we can no delay the whole form code to shipout
1156   % as the object reference and the size is often wanted on the current page.
1157   % so we need to allocate a box - but probably they won't be thousands xform
1158   % in a document so it shouldn't matter.
1159   \cs_new_protected:Npn \__pdf_backend_xform_new:nnnnn #1 #2 #3 #4 #5
1160   % #1 name
1161   % #2 attributes
1162   % #3 resources
1163   % #4 content, not necessarily a box!
1164   % #5 a clist to exclude resources
1165   {
1166     \int_gincr:N \g__pdf_backend_object_int
1167     \int_const:cn
1168       { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1169     { \g__pdf_backend_object_int }
1170     \box_new:c { g__pdf_backend_xform_#1_box }
1171     \hbox_gset:cn { g__pdf_backend_xform_#1_box }
1172     {
1173       \bool_set_true:N \l__pdf_backend_xform_bool
1174       #4
1175     }
1176     \tl_const:ce
1177       { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1178       { \tex_the:D \box_wd:c { g__pdf_backend_xform_#1_box } }
1179     \tl_const:ce

```

```

1180 { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1181 { \tex_the:D \box_ht:c { g__pdf_backend_xform_#1_box } }
1182 \tl_const:ce
1183 { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1184 { \tex_the:D \box_dp:c { g__pdf_backend_xform_#1_box } }
1185 \box_set_dp:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1186 \box_set_ht:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1187 \box_set_wd:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1188 \hook_gput_next_code:nn {shipout/background}
1189 {
1190   \mode_leave_vertical: %needed, the xform disappears without it.
1191   \__pdf_backend:e
1192   {
1193     bxobj ~ \__pdf_backend_xform_ref:n { #1 }
1194     \c_space_tl width ~ \pdfxform_wd:n { #1 }
1195     \c_space_tl height ~ \pdfxform_ht:n { #1 }
1196     \c_space_tl depth ~ \pdfxform_dp:n { #1 }
1197   }
1198   \box_use_drop:c { g__pdf_backend_xform_#1_box }
1199   \__pdf_backend:e {put ~ @resources ~<<#3>> }
1200   \clist_map_inline:nn {#5}
1201   { \bool_set_false:c { l__pdf_backend_xform_res_##1_bool } }
1202   \bool_if:NT \l__pdf_backend_xform_res_extgstate_bool
1203   {
1204     \__pdf_backend:e
1205     {
1206       put~ @resources ~
1207       <<
1208         /ExtGState~ \pdf_object_ref:n { __pdf/Page/Resources/ExtGState }
1209       >>
1210     }
1211   }
1212   \bool_if:NT \l__pdf_backend_xform_res_pattern_bool
1213   {
1214     \__pdf_backend:e
1215     {
1216       put~ @resources ~
1217       <<
1218         /Pattern~ \pdf_object_ref:n { __pdf/Page/Resources/Pattern }
1219       >>
1220     }
1221   }
1222   \bool_if:NT \l__pdf_backend_xform_res_shading_bool
1223   {
1224     \__pdf_backend:e
1225     {
1226       put~ @resources ~
1227       <<
1228         /Shading~ \pdf_object_ref:n { __pdf/Page/Resources/Shading }
1229       >>
1230     }
1231   }
1232   \bool_if:NT \l__pdf_backend_xform_res_colorspace_bool
1233   {

```

```

1234         \__pdf_backend:e
1235         {
1236             put~ @resources ~
1237             <<
1238             /ColorSpace~
1239             \pdf_object_ref:n { __pdf/Page/Resources/ColorSpace }
1240             >>
1241         }
1242     }
1243     \__pdf_backend:e {exobj ~<<#2>>}
1244     \__pdf_backend_xform_res_reset:
1245 }
1246 }
1247
1248
1249
1250 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1251 {
1252     @pdf.xform \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1253 }
1254
1255 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
1256 {
1257     \hbox_set:Nn \l__pdf_backend_tmpa_box
1258     {
1259         \__pdf_backend:e
1260         {
1261             uxobj~ \__pdf_backend_xform_ref:n { #1 }
1262         }
1263     }
1264     \box_set_wd:Nn \l__pdf_backend_tmpa_box { \pdfxform_wd:n { #1 } }
1265     \box_set_ht:Nn \l__pdf_backend_tmpa_box { \pdfxform_ht:n { #1 } }
1266     \box_set_dp:Nn \l__pdf_backend_tmpa_box { \pdfxform_dp:n { #1 } }
1267     \box_use_drop:N \l__pdf_backend_tmpa_box
1268 }
1269 </dvipdfmx|xdvipdfmx>
1270 <*dvisvgm>
1271 % unclear what it should do!!
1272 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnnn #1 #2 #3 #4 #5 {}
1273 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1 {}
1274 \cs_new:Npn \__pdf_backend_xform_ref:n {}
1275 </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

```

1276 <*dvips>
1277 \tl_new:N \l__pdf_backend_xform_tmpwd_tl
1278 \tl_new:N \l__pdf_backend_xform_tmphd_tl
1279 \tl_new:N \l__pdf_backend_xform_tmphd_tl
1280 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnnn #1 #2 #3 #4 #5 % #1 name, #2 attribute,
1281 {
1282     \int_gincr:N \g__pdf_backend_object_int

```

```

1283 \int_const:cn
1284 { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1285 { \g__pdf_backend_object_int }
1286
1287 \hbox_set:Nn \l__pdf_backend_tmpa_box
1288 {
1289   \bool_set_true:N \l__pdf_backend_xform_bool
1290   \prop_gclear:c {\__kernel_pdffdict_name:n { g__pdf_Core/Xform/Resources/Properties }}
1291   #4
1292 }
1293 %store the dimensions
1294 \tl_const:ce
1295 { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1296 { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
1297 \tl_const:ce
1298 { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1299 { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
1300 \tl_const:ce
1301 { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1302 { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
1303 %store content dimensions in DPI units (Dots) (code from issue 25)
1304 \tl_set:N\l__pdf_backend_xform_tmpwd_tl
1305 {
1306   \dim_to_decimal_in_sp:n{ \box_wd:N \l__pdf_backend_tmpa_box }~
1307   65536~div~72.27~div~DVImag~mul~Resolution~mul~
1308 }
1309 \tl_set:N\l__pdf_backend_xform_tmph_tl
1310 {
1311   \dim_to_decimal_in_sp:n{ \box_ht:N \l__pdf_backend_tmpa_box }~
1312   65536~div~72.27~div~DVImag~mul~VResolution~mul~
1313 }
1314 \tl_set:N\l__pdf_backend_xform_tmpdp_tl
1315 {
1316   \dim_to_decimal_in_sp:n{ \box_dp:N \l__pdf_backend_tmpa_box }~
1317   65536~div~72.27~div~DVImag~mul~VResolution~mul~
1318 }
1319 % mirror the box
1320 %\box_scale:Nnn \l__pdf_backend_tmpa_box {1} {-1}
1321 \hbox_set:Nn\l__pdf_backend_tmpb_box
1322 {
1323   \__kernel_backend_postscript:e
1324   {
1325     gsave~currentpoint~
1326     initclip~ % restore default clipping path (page device/whole page)
1327     clippath~pathbbox~newpath~pop~pop~
1328     \tl_use:N\l__pdf_backend_xform_tmpdp_tl~add~translate~
1329     mark~
1330     /_objdef~{ pdf.obj \int_use:N\g__pdf_backend_object_int }\c_space_tl~
1331     /BBox[
1332       0~
1333       \tl_use:N\l__pdf_backend_xform_tmph_tl~
1334       \tl_use:N\l__pdf_backend_xform_tmpwd_tl~
1335       \tl_use:N\l__pdf_backend_xform_tmpdp_tl~
1336       neg

```

```

1337     ]
1338     \str_if_eq:eeF{#1}{}
1339     {
1340         product~(Distiller)~search~{pop~pop~pop~#2}{pop}ifelse~
1341     }
1342     /BP~pdfmark~1~-1~scale~neg~exch~neg~exch~translate
1343 }
1344 \box_use_drop:N\l__pdf_backend_tmpa_box
1345 \__kernel_backend_postscript:n
1346 {
1347     mark ~ /EP~pdfmark ~ grestore
1348 }
1349 \str_if_eq:eeF{#1}{}
1350 {
1351     \__kernel_backend_postscript:e
1352     {
1353         product~(Ghostscript)~search~
1354         {
1355             pop~pop~pop~
1356             mark~
1357             { pdf.obj \int_use:c{c__pdf_backend_xform_ \tl_to_str:n {#1} _int} }
1358             ~<<#2>>~/PUT~pdfmark
1359         }{pop}ifelse
1360     }
1361 }
1362 }
1363 \box_set_dp:Nn \l__pdf_backend_tmpb_box { \c_zero_dim }
1364 \box_set_ht:Nn \l__pdf_backend_tmpb_box { \c_zero_dim }
1365 \box_set_wd:Nn \l__pdf_backend_tmpb_box { \c_zero_dim }
1366 \hook_gput_code:nnn {begindocument/end}{pdfxform}
1367 {
1368     \mode_leave_vertical:
1369     \box_use:N\l__pdf_backend_tmpb_box
1370 }
1371 }
1372
1373
1374 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
1375 {
1376     \hbox_set:Nn \l__pdf_backend_tmpa_box
1377     {
1378         \__kernel_backend_postscript:e
1379         {
1380             gsave~currentpoint~translate~1~-1~scale~
1381             mark~{ pdf.obj \int_use:c{c__pdf_backend_xform_ \tl_to_str:n {#1} _int} }~
1382             /SP~pdfmark ~ grestore
1383         }
1384     }
1385     \box_set_wd:Nn \l__pdf_backend_tmpa_box { \pdfxform_wd:n { #1 } }
1386     \box_set_ht:Nn \l__pdf_backend_tmpa_box { \pdfxform_ht:n { #1 } }
1387     \box_set_dp:Nn \l__pdf_backend_tmpa_box { \pdfxform_dp:n { #1 } }
1388     \box_use_drop:N \l__pdf_backend_tmpa_box
1389 }
1390 \cs_new:Npn \__pdf_backend_xform_ref:n #1

```

```

1391 {
1392   { pdf.obj \int_use:c{c__pdf_backend_xform_ \tl_to_str:n {#1} _int} }
1393 }
1394
1395 </dvips>
1396 <*drivers>
1397 %% all
1398 \prg_new_conditional:Npnn \__pdf_backend_xform_if_exist:n #1 { p , T , F , TF }
1399 {
1400   \int_if_exist:cTF { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1401   { \prg_return_true: }
1402   { \prg_return_false:}
1403 }
1404 \prg_new_eq_conditional:NNn \pdfxform_if_exist:n\__pdf_backend_xform_if_exist:n
1405 { TF , T , F , p }
1406 </drivers>

```

(End of definition for __pdf_backend_xform_new:nnnn, __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_EXlive 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

1407 <*drivers>
1408 \tl_new:N \l_pdf_current_structure_destination_tl
1409 </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:nnnn -> \__pdf_backend_structure_destination:nnnn
\__pdfannot_backend_link_begin_goto:nnw -> \__pdf_backend_link_begin_structure_goto:nnw
\__pdf_backend_destination:nn -> \__pdf_backend_indexed_structure_destination:nn
\__pdf_backend_destination:nnnn -> \__pdf_backend_indexed_structure_destination:nnnn
\__pdfannot_backend_link_begin_goto:nnw -> \__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:
1410 <*drivers>
1411 \cs_new_protected:Npn \pdf_activate_structure_destination:
1412 {
1413   \cs_gset_eq:NN \__pdf_backend_destination:nn \__pdf_backend_structure_destination:nn
1414   \cs_gset_eq:NN \__pdf_backend_destination:nnnn \__pdf_backend_structure_destination:nnnn
1415   \cs_gset_eq:NN \__pdfannot_backend_link_begin_goto:nnw \__pdfannot_backend_link_begin_structur
1416 }
1417 \cs_new_protected:Npn \pdf_activate_indexed_structure_destination:
1418 {
1419   \cs_gset_eq:NN \__pdf_backend_destination:nn \__pdf_backend_indexed_structure_desti
1420   \cs_gset_eq:NN \__pdf_backend_destination:nnnn \__pdf_backend_indexed_structure_desti
1421   \cs_gset_eq:NN \__pdfannot_backend_link_begin_goto:nnw \__pdfannot_backend_link_begin_st
1422 }
1423 </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.

```

1424 <*drivers>
1425 \cs_set_eq:NN \__pdf_backend_structure_destination:nn \__pdf_backend_destination:nn
1426 \cs_set_eq:NN \__pdf_backend_structure_destination:nnnn \__pdf_backend_destination:nnnn
1427 \cs_set_eq:NN \__pdfannot_backend_link_begin_structure_goto:nnw \__pdfannot_backend_link_be
1428 \cs_set_eq:NN \__pdf_backend_indexed_structure_destination:nn \__pdf_backend_destinati
1429 \cs_set_eq:NN \__pdf_backend_indexed_structure_destination:nnnn \__pdf_backend_destinati
1430 </drivers>

```

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

```

1431 <*xdvipdfmx|dvipdfmx>
1432 \cs_set_protected:Npn \__pdf_backend_structure_destination:nn #1#2

```

```

1433 {
1434   \__pdf_backend:e
1435   {
1436     dest ~ ( \exp_not:n {#1} )
1437     [
1438       @thispage
1439       \str_case:nnF {#2}
1440       {
1441         { xyz } { /XYZ ~ @xpos ~ @ypos ~ null }
1442         { fit } { /Fit }
1443         { fitb } { /FitB }
1444         { fitbh } { /FitBH }
1445         { fitbv } { /FitBV ~ @xpos }
1446         { fith } { /FitH ~ @ypos }
1447         { fitv } { /FitV ~ @xpos }
1448         { fitr } { /Fit }
1449       }
1450       { /XYZ ~ @xpos ~ @ypos ~ \fp_eval:n { (#2) / 100 } }
1451     ]
1452   }

```

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.

```

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{#1}
1458       [
1459         \exp_args:Ne \pdf_object_ref:n { \l_pdf_current_structure_destination_tl }
1460         \str_case:nnF {#2}
1461         {
1462           { xyz } { /XYZ ~ @xpos ~ @ypos ~ null }
1463           { fit } { /Fit }
1464           { fitb } { /FitB }
1465           { fitbh } { /FitBH }
1466           { fitbv } { /FitBV ~ @xpos }
1467           { fith } { /FitH ~ @ypos }
1468           { fitv } { /FitV ~ @xpos }
1469           { fitr } { /Fit }
1470         }
1471         { /XYZ ~ @xpos ~ @ypos ~ \fp_eval:n { (#2) / 100 } }
1472       ]
1473     }
1474   }
1475 }

```

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

```

1476 \cs_new_protected:Npn \__pdf_backend_structure_destination_aux:nnnn #1#2#3#4
1477 {
1478   \vbox_to_zero:n
1479   {
1480     \__kernel_kern:n {#4}

```

```

1481     \hbox:n
1482     {
1483         \_pdf_backend:n { obj ~ @pdf_ #2 _llx ~ @xpos }
1484         \_pdf_backend:n { obj ~ @pdf_ #2 _lly ~ @ypos }
1485     }
1486     \tex_vss:D
1487 }
1488 \__kernel_kern:n {#1}
1489 \vbox_to_zero:n
1490 {
1491     \__kernel_kern:n { -#3 }
1492     \hbox:n
1493     {
1494         \_pdf_backend:n
1495         {
1496             dest ~ (#2)
1497             [
1498                 @thispage
1499                 /FitR ~
1500                 @pdf_ #2 _llx ~ @pdf_ #2 _lly ~
1501                 @xpos ~ @ypos
1502             ]
1503         }
1504     }

```

Here we add the structure destination to the same box

```

1504     \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1505     {
1506         \_pdf_backend:e
1507         {
1508             obj ~ @pdf.SDest.\exp_not:n{#2}
1509             [
1510                 \exp_args:Ne \pdf_object_ref:n { \l_pdf_current_structure_destination_
1511                 /FitR ~
1512                 @pdf_ #2 _llx ~ @pdf_ #2 _lly ~
1513                 @xpos ~ @ypos
1514             ]
1515         }
1516     }
1517 }
1518 \tex_vss:D
1519 }
1520 \__kernel_kern:n { -#1 }
1521 }

```

And now we redefine the destination command:

```

1522 \cs_set_protected:Npn \_pdf_backend_structure_destination:nnnn #1#2#3#4
1523 {
1524     \exp_args:Ne \_pdf_backend_structure_destination_aux:nnnn
1525     { \dim_eval:n {#2} } {#1} {#3} {#4}
1526 }

```

At last the goto link.

```

1527 \cs_set_protected:Npn \_pdfannot_backend_link_begin_structure_goto:nnw #1#2
1528 {
1529     \_pdfannot_backend_link_begin:n { #1 /Subtype /Link /A << /S /GoTo /D ( #2 ) /SD-@pdf.S

```

```

1530 }
1531 </xdvipdfmx|dvipdfmx>

```

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

```

1532 <*pdftex>
1533 \bool_lazy_and:nnT
1534 { \int_compare_p:nNn {\tex_pdftexversion:D } > {139} }
1535 { \int_compare_p:nNn {\tex_pdftexrevision:D } > {23} }
1536 {
1537   \cs_set_protected:Npn \__pdf_backend_structure_destination:nn #1#2
1538   {
1539     \tex_pdfdest:D
1540     name {#1}
1541     \str_case:nnF {#2}
1542     {
1543       { xyz } { xyz }
1544       { fit } { fit }
1545       { fitb } { fitb }
1546       { fitbh } { fitbh }
1547       { fitbv } { fitbv }
1548       { fith } { fith }
1549       { fitv } { fitv }
1550       { fitr } { fitr }
1551     }
1552     { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1553     \scan_stop:
1554     \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1555     {
1556       \tex_pdfdest:D
1557       struct~
1558       \int_use:c
1559       { c__pdf_object_ \exp_args:Ne \tl_to_str:n {\l_pdf_current_structure_destin
1560       name {#1}
1561       \str_case:nnF {#2}
1562       {
1563         { xyz } { xyz }
1564         { fit } { fit }
1565         { fitb } { fitb }
1566         { fitbh } { fitbh }
1567         { fitbv } { fitbv }
1568         { fith } { fith }
1569         { fitv } { fitv }
1570         { fitr } { fitr }
1571       }
1572       { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1573       \scan_stop:
1574     }
1575   }
1576   \cs_set_protected:Npn \__pdf_backend_structure_destination:nnnn #1#2#3#4
1577   {
1578     \tex_pdfdest:D
1579     name {#1}
1580     fitr ~
1581     width \dim_eval:n {#2} ~
1582     height \dim_eval:n {#3} ~

```

```

1583     depth \dim_eval:n {#4} \scan_stop:
1584 \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1585 {
1586     \tex_pdfdest:D
1587     struct~
1588     \int_use:c
1589     { c__pdf_object_ \exp_args:Ne \tl_to_str:n {\l_pdf_current_structure_destinat
1590     name {#1}
1591     fitr ~
1592     width \dim_eval:n {#2} ~
1593     height \dim_eval:n {#3} ~
1594     depth \dim_eval:n {#4} \scan_stop:
1595 }
1596 }
1597 \cs_set_protected:Npn \__pdfannot_backend_link_begin_structure_goto:nnw #1#2
1598 {
1599     \__pdfannot_backend_link_begin:nnnw {#1} { goto~struct~name~{#2}~name } {#2}
1600 }
1601 }
1602 </pdftex>

```

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

```

1603 <*luatex>
1604 \int_compare:nNnT {\directlua{tex.print(status.list()["development_id"])} } > {7468}
1605 {
1606     \cs_set_protected:Npn \__pdf_backend_structure_destination:nn #1#2
1607     {
1608         \tex_pdfextension:D dest
1609         name {#1}
1610         \str_case:nnF {#2}
1611         {
1612             { xyz } { xyz }
1613             { fit } { fit }
1614             { fitb } { fitb }
1615             { fitbh } { fitbh }
1616             { fitbv } { fitbv }
1617             { fith } { fith }
1618             { fitv } { fitv }
1619             { fitr } { fitr }
1620         }
1621         { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1622         \scan_stop:
1623     \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1624     {
1625         \tex_pdfextension:D dest
1626         struct~
1627         \int_use:c
1628         { c__pdf_object_ \exp_args:Ne \tl_to_str:n {\l_pdf_current_structure_destin
1629         name {#1}
1630         \str_case:nnF {#2}
1631         {
1632             { xyz } { xyz }
1633             { fit } { fit }
1634             { fitb } { fitb }
1635             { fitbh } { fitbh }

```

```

1636         { fitbv } { fitbv }
1637         { fith } { fith }
1638         { fitv } { fitv }
1639         { fitr } { fitr }
1640     }
1641     { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1642     \scan_stop:
1643 }
1644 }
1645 \cs_set_protected:Npn \__pdf_backend_structure_destination:nnnn #1#2#3#4
1646 {
1647     \tex_pdfextension:D dest
1648     name {#1}
1649     fitr ~
1650     width \dim_eval:n {#2} ~
1651     height \dim_eval:n {#3} ~
1652     depth \dim_eval:n {#4} \scan_stop:
1653     \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1654     {
1655         \tex_pdfextension:D dest
1656         struct~
1657         \int_use:c
1658         { c__pdf_object_ \exp_args:Ne \tl_to_str:n { \l_pdf_current_structure_destination_tl }
1659         name {#1}
1660         fitr ~
1661         width \dim_eval:n {#2} ~
1662         height \dim_eval:n {#3} ~
1663         depth \dim_eval:n {#4} \scan_stop:
1664     }
1665 }
1666 \cs_set_protected:Npn \__pdfannot_backend_link_begin_structure_goto:nnw #1#2
1667 {
1668     \__pdfannot_backend_link_begin:nnnw {#1} { goto~struct~name~{#2}~name } {#2}
1669 }
1670 }
1671 </luatex>

```

(End of definition for `\__pdf_backend_structure_destination:nn`, `\__pdf_backend_structure_destination:nnnn`, and `\__pdfannot_backend_link_begin_structure_goto:nnw`.)

`\__pdf_backend_indexed_structure_destination:nn`
`\__pdf_backend_indexed_structure_destination:nnnn`

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:

```

1672 < *xdvipdfmx | dvipdfmx >
1673 \cs_set_protected:Npn \__pdf_backend_indexed_structure_destination:nn #1#2
1674 {
1675     \__pdf_backend:e
1676     {
1677         dest ~ ( \exp_not:n {#1} )
1678         [
1679             @thispage
1680             \str_case:nnF {#2}
1681             {
1682                 { xyz } { /XYZ ~ @xpos ~ @ypos ~ null }

```

```

1683         { fit }    { /Fit }
1684         { fitb }   { /FitB }
1685         { fitbh }  { /FitBH }
1686         { fitbv }  { /FitBV ~ @xpos }
1687         { fith }   { /FitH ~ @ypos }
1688         { fitv }   { /FitV ~ @xpos }
1689         { fitr }   { /Fit }
1690     }
1691     { /XYZ ~ @xpos ~ @ypos ~ \fp_eval:n { (#2) / 100 } }
1692 ]
1693 }

```

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.

```

1694 \__pdf_backend:e
1695 {
1696     obj ~ @pdf.SDest.\exp_not:n{#1}
1697     [
1698         \exp_after:wN \pdf_object_ref_indexed:nn \l_pdf_current_structure_destination_t
1699         \str_case:nnF {#2}
1700         {
1701             { xyz }    { /XYZ ~ @xpos ~ @ypos ~ null }
1702             { fit }    { /Fit }
1703             { fitb }   { /FitB }
1704             { fitbh }  { /FitBH }
1705             { fitbv }  { /FitBV ~ @xpos }
1706             { fith }   { /FitH ~ @ypos }
1707             { fitv }   { /FitV ~ @xpos }
1708             { fitr }   { /Fit }
1709         }
1710         { /XYZ ~ @xpos ~ @ypos ~ \fp_eval:n { (#2) / 100 } }
1711     ]
1712 }
1713 }

```

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

```

1714 \cs_new_protected:Npn \__pdf_backend_indexed_structure_destination_aux:nnnn #1#2#3#4
1715 {
1716     \vbox_to_zero:n
1717     {
1718         \__kernel_kern:n {#4}
1719         \hbox:n
1720         {
1721             \__pdf_backend:n { obj ~ @pdf_ #2 _llx ~ @xpos }
1722             \__pdf_backend:n { obj ~ @pdf_ #2 _lly ~ @ypos }
1723         }
1724         \tex_vss:D
1725     }
1726     \__kernel_kern:n {#1}
1727     \vbox_to_zero:n
1728     {
1729         \__kernel_kern:n { -#3 }
1730         \hbox:n

```

```

1731     {
1732       \__pdf_backend:n
1733       {
1734         dest ~ (#2)
1735         [
1736           @thispage
1737           /FitR ~
1738           @pdf_ #2 _llx ~ @pdf_ #2 _lly ~
1739           @xpos ~ @ypos
1740         ]
1741       }

```

Here we add the structure destination to the same box

```

1742       \__pdf_backend:e
1743       {
1744         obj ~ @pdf.SDest.\exp_not:n{#2}
1745         [
1746           \exp_after:wN \pdf_object_ref_indexed:nn \l_pdf_current_structure_destin
1747           /FitR ~
1748           @pdf_ #2 _llx ~ @pdf_ #2 _lly ~
1749           @xpos ~ @ypos
1750         ]
1751       }
1752     }
1753     \tex_vss:D
1754   }
1755   \__kernel_kern:n { -#1 }
1756 }

```

And now we redefine the destination command:

```

1757 \cs_set_protected:Npn \__pdf_backend_indexed_structure_destination:nnnn #1#2#3#4
1758 {
1759   \exp_args:Ne \__pdf_backend_indexed_structure_destination_aux:nnnn
1760   { \dim_eval:n {#2} } {#1} {#3} {#4}
1761 }
1762 </xdvipdfmx|dvipdfmx>

```

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

```

1763 <*pdftex>
1764 \bool_lazy_and:nnT
1765 { \int_compare_p:nNn {\tex_pdftexversion:D } > {139} }
1766 { \int_compare_p:nNn {\tex_pdftexrevision:D } > {23} }
1767 {
1768   \cs_set_protected:Npn \__pdf_backend_indexed_structure_destination:nn #1#2
1769   {
1770     \tex_pdfdest:D
1771     name {#1}
1772     \str_case:nnF {#2}
1773     {
1774       { xyz } { xyz }
1775       { fit } { fit }
1776       { fitb } { fitb }
1777       { fitbh } { fitbh }
1778       { fitbv } { fitbv }
1779       { fith } { fith }

```

```

1780         { fitv } { fitv }
1781         { fitr } { fitr }
1782     }
1783     { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1784 \scan_stop:
1785 \tex_pdfdest:D
1786     struct~
1787     \exp_after:wN \_kernel_pdf_object_id_indexed:nn \l_pdf_current_structure_des
1788     name {#1}
1789     \str_case:nnF {#2}
1790     {
1791         { xyz } { xyz }
1792         { fit } { fit }
1793         { fitb } { fitb }
1794         { fitbh } { fitbh }
1795         { fitbv } { fitbv }
1796         { fith } { fith }
1797         { fitv } { fitv }
1798         { fitr } { fitr }
1799     }
1800     { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1801 \scan_stop:
1802 }
1803 \cs_set_protected:Npn \_pdf_backend_indexed_structure_destination:nnnn #1#2#3#4
1804 {
1805     \tex_pdfdest:D
1806     name {#1}
1807     fitr ~
1808     width \dim_eval:n {#2} ~
1809     height \dim_eval:n {#3} ~
1810     depth \dim_eval:n {#4} \scan_stop:
1811 \tex_pdfdest:D
1812     struct~
1813     \exp_after:wN \_kernel_pdf_object_id_indexed:nn \l_pdf_current_structure_destinati
1814     name {#1}
1815     fitr ~
1816     width \dim_eval:n {#2} ~
1817     height \dim_eval:n {#3} ~
1818     depth \dim_eval:n {#4} \scan_stop:
1819 }
1820 }
1821 </pdfTeX>

```

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

```

1822 <*luatex>
1823 \int_compare:nNtT {\directlua{tex.print(status.list()["development_id"])} } > {7468}
1824 {
1825     \cs_set_protected:Npn \_pdf_backend_indexed_structure_destination:nn #1#2
1826     {
1827         \tex_pdfextension:D dest
1828         name {#1}
1829         \str_case:nnF {#2}
1830         {
1831             { xyz } { xyz }
1832             { fit } { fit }

```

```

1833         { fitb } { fitb }
1834         { fitbh } { fitbh }
1835         { fitbv } { fitbv }
1836         { fith } { fith }
1837         { fitv } { fitv }
1838         { fitr } { fitr }
1839     }
1840     { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1841     \scan_stop:
1842     \tex_pdfextension:D dest
1843     struct~
1844     \exp_after:wN \__kernel_pdf_object_id_indexed:nn \l_pdf_current_structure_desti
1845     name {#1}
1846     \str_case:nnF {#2}
1847     {
1848         { xyz } { xyz }
1849         { fit } { fit }
1850         { fitb } { fitb }
1851         { fitbh } { fitbh }
1852         { fitbv } { fitbv }
1853         { fith } { fith }
1854         { fitv } { fitv }
1855         { fitr } { fitr }
1856     }
1857     { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1858     \scan_stop:
1859 }
1860 \cs_set_protected:Npn \__pdf_backend_indexed_structure_destination:nnnn #1#2#3#4
1861 {
1862     \tex_pdfextension:D dest
1863     name {#1}
1864     fitr ~
1865     width \dim_eval:n {#2} ~
1866     height \dim_eval:n {#3} ~
1867     depth \dim_eval:n {#4} \scan_stop:
1868     \tex_pdfextension:D dest
1869     struct~
1870     \exp_after:wN \__kernel_pdf_object_id_indexed:nn \l_pdf_current_structure_destinati
1871     name {#1}
1872     fitr ~
1873     width \dim_eval:n {#2} ~
1874     height \dim_eval:n {#3} ~
1875     depth \dim_eval:n {#4} \scan_stop:
1876 }
1877 \cs_set_protected:Npn \__pdfannot_backend_link_begin_structure_goto:nnw #1#2
1878 {
1879     \__pdfannot_backend_link_begin:nnnw {#1} { goto~struct~name~{#2}~name } {#2}
1880 }
1881 }
1882 \end{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

```

1883 <*drivers>
1884 \cs_new_protected:Npn \__pdf_backend_set_regression_data:
1885 {
1886   \sys_gset_rand_seed:n{1000}
1887   \pdfmanagement_add:nnn{Info}{Creator}{(TeX)}
1888 </drivers>
1889 <*dvips>
1890   \AddToHook{begindocument}{\pdfmanagement_add:nnn{Info}{Producer}{(pdfTeX+dvips)}}
1891   \__kernel_backend_literal:e{!~<</DocumentUUID~(DocumentUUID)>>~setpagedevice}
1892   \__kernel_backend_literal:e{!~<</InstanceUUID~(InstanceUUID)>>~setpagedevice}
1893   \pdfmanagement_add:nne{Info}{CreationDate}{(\c_sys_timestamp_str)}
1894   \pdfmanagement_add:nne{Info}{ModDate}{(\c_sys_timestamp_str)}
1895 </dvips>
1896 <*dvipdfmx>
1897   \pdfmanagement_add:nnn{Info}{Producer}{(dvipdfmx)}
1898   \__kernel_backend_literal:e
1899   {pdf:trailerid [~
1900     <00112233445566778899aabbccddeeff>~
1901     <00112233445566778899aabbccddeeff>~
1902   ]}
1903 </dvipdfmx>
1904 <*xdvipdfmx>
1905   \pdfmanagement_add:nnn{Info}{Producer}{(xetex)}
1906   \__kernel_backend_literal:e
1907   {pdf:trailerid [~
1908     <00112233445566778899aabbccddeeff>~
1909     <00112233445566778899aabbccddeeff>~
1910   ]}
1911 </xdvipdfmx>
1912 <*pdftex>
1913   \pdfmanagement_add:nnn{Info}{Producer}{(pdfTeX)}
1914   \tex_pdfsuppressptexinfo:D 7 \scan_stop:
1915   \pdftrailerid{2350CAD05F8A7AF0AA4058486855344F}
1916 </pdftex>
1917 <*luatex>
1918   \pdfmanagement_add:nnn{Info}{Producer}{(LuaTeX)}
1919   \tex_pdfvariable:D suppressoptionalinfo 7\relax
1920   \tex_pdfvariable:D trailerid
1921   {[~
1922     <2350CAD05F8A7AF0AA4058486855344F>~
1923     <2350CAD05F8A7AF0AA4058486855344F>~
1924   ]}
1925 </luatex>

```

Embedded files should also have a fix date.

```

1926 <*drivers>
1927   \pdfdict_put:nne {l_pdffile/Params} {ModDate}{(\c_sys_timestamp_str)}
1928   \AddToDocumentProperties[hyperref]{pdfinstanceid}{uuid:0a57c455-157a-4141-
      8c19-6237d832fc80}

```

```

1929     \AddToDocumentProperties[hyperref]{pdfproducer}{\c_sys_engine_exec_str-NN.NN.NN}
1930   }
1931 </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 the `__pdfmeta/xmp` object which should be declared before.

`luatex` has of now (2025-11-12) a bug: using the `uncompressed` key disables object compression for all following objects. We therefor delay the writing into the `enddocument/end` hook after the `tagpdf` code.

```

1932 <*luatex>
1933 \cs_new_protected:Npn \__pdf_backend_metadata_stream:n #1
1934 {
1935   \AddToHook{enddocument/end}
1936   {
1937     \tex_immediate:D \tex_pdfextension:D obj ~
1938       useobjnum ~ \int_eval:n{\__pdf_object_retrieve:n {\__pdfmeta/xmp}}~uncompressed~
1939       \__pdf_backend_object_write:nn {stream}
1940       {{/Type~/Metadata~/Subtype~/XML}{#1}}
1941   }
1942 }
1943 </luatex>
1944 <*pdftex>
1945 \cs_new_protected:Npn \__pdf_backend_metadata_stream:n #1
1946 {
1947   \group_begin:
1948     \tex_pdfcompresslevel:D 0 \scan_stop:
1949     \tex_immediate:D \tex_pdfobj:D useobjnum ~ \int_eval:n{\__pdf_object_retrieve:n
1950       {\__pdfmeta/xmp}}~
1951     \__pdf_backend_object_write:nn {stream} {{/Type~/Metadata~/Subtype~/XML}{#1}}
1952   \group_end:
1953 }
1954 </pdftex>
1955 <*xdvipdfmx | dvipdfmx | dvips | dvisvgm>
1956 \cs_new_protected:Npn \__pdf_backend_metadata_stream:n #1
1957 {
1958   \pdf_object_write:nnn{\__pdfmeta/xmp} {stream}{{/Type~/Metadata~/Subtype~/XML}{#1}}
1959 }
1960 </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.

```

1961 <*xdvipdfmx | dvipdfmx | dvips | dvisvgm>
1962 \cs_new_protected:Npn \_pdf_backend_omit_charset:n #1 {} % #1 number
1963 </xdvipdfmx | dvipdfmx | dvips | dvisvgm>
1964 <*pdftex>
1965 \cs_new_protected:Npn \_pdf_backend_omit_charset:n #1 % #1 number
1966 {
1967   \tex_pdfomitcharset:D = #1 \scan_stop:
1968 }
1969 </pdftex>
1970 <*luatex>
1971 \cs_new_protected:Npn \_pdf_backend_omit_charset:n #1 % #1 number
1972 {
1973   \tex_pdfvariable:D omitcharset = #1 \scan_stop:
1974 }
1975 </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.

```

1976 <*xdvipdfmx | dvipdfmx | dvips | dvisvgm>
1977 \cs_new_protected:Npn \_pdf_backend_omit_info:n #1 {} % #1 number
1978 </xdvipdfmx | dvipdfmx | dvips | dvisvgm>
1979 <*pdftex>
1980 \bool_lazy_and:nnTF
1981 { \int_compare_p:nNn {\tex_pdftexversion:D } > {139} }
1982 { \int_compare_p:nNn {\tex_pdftexrevision:D } > {24} }
1983 {
1984   \cs_new_protected:Npn \_pdf_backend_omit_info:n #1 % #1 number
1985   {
1986     \pdfomitinfodict = #1 \scan_stop:
1987   }
1988 }
1989 {
1990   \cs_new_protected:Npn \_pdf_backend_omit_info:n #1 {} % #1 number
1991 }
1992 </pdftex>
1993 <*luatex>
1994 \int_compare:nNnTF {\directlua{tex.print(status.list()["development_id"])} } > {7560}
1995 {
1996   \cs_new_protected:Npn \_pdf_backend_omit_info:n #1 % #1 number
1997   {
1998     \tex_pdfvariable:D omitinfodict = #1 \scan_stop:
1999   }
2000 }
2001 }
2002 {
2003   \cs_new_protected:Npn \_pdf_backend_omit_info:n #1 {} % #1 number
2004 }
2005 </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.

```

2006 <*xdvipdfmx|dvipdfmx|dvips|dvisvgm|pdftex>
2007 \cs_new_protected:Npn \__pdf_backend_omit_cidset:n #1 {} % #1 number
2008 </xdvipdfmx|dvipdfmx|dvips|dvisvgm|pdftex>
2009 <*luatex>
2010 \cs_new_protected:Npn \__pdf_backend_omit_cidset:n #1 % #1 number
2011 {
2012     \tex_pdfvariable:D omitcidset = #1 \scan_stop:
2013 }
2014 </luatex>

```

(End of definition for `\__pdf_backend_omit_cidset:n`.)

1.15 Outline backend

```

2015 <*dvips>
2016 \cs_new_protected:Npn \__pdf_backend_action:nnnnnnn #1#2#3#4#5#6#7
2017 % #1 level (ignored), #2 kid count, #3 open? #4 color (space separated values), #5 flag, #6
2018 {
2019     \__pdf_backend_pdfmark:e
2020     {
2021         /Title~(\exp_not:n{#7})~
2022         /Count~\bool_if:nF{#3}{-}#2~
2023         \tl_if_blank:nF { #4 }{/C-[#4]}
2024         /F~#5
2025         /Action <<\exp_not:n{#6}>>
2026         /OUT
2027     }
2028 }
2029 \cs_new_protected:Npn \__pdf_backend_goto:nnnnnnn #1#2#3#4#5#6#7
2030 % #1 level (ignored), #2 kid count, #3 open? #4 color (space separated values), #5 flag, #6
2031 {
2032     \__pdf_backend_pdfmark:e
2033     {
2034         /Title~(\exp_not:n{#7})~
2035         /Count~\bool_if:nF{#3}{-}#2~
2036         \tl_if_blank:nF { #4 }{/C-[#4]}
2037         /F~#5
2038         /Action/GoTo/Dest(\exp_not:n{#6})~cvn
2039         /OUT
2040     }
2041 }
2042 </dvips>
2043 <*pdftex>
2044 \cs_new_protected:Npn \__pdf_backend_action:nnnnnnn #1#2#3#4#5#6#7
2045 % #1 level (ignored), #2 kid count, #3 open? #4 color (space separated values), #5 flag, #6
2046 {
2047     \tex_pdfoutline:D~attr
2048     {
2049         /F~#5~

```

```

2050     \tl_if_blank:nF { #4 }{/C~[#4]}
2051   }~
2052   user~{<<\exp_not:n{#6}>>}~
2053   count~\bool_if:nF{#3}{-}#2~
2054   {\exp_not:n{#7}}
2055 }
2056 \cs_new_protected:Npn \__pdf_backend_goto:nnnnnnn #1#2#3#4#5#6#7
2057 % #1 level (ignored), #2 kid count, #3 open? #4 color (space separated values), #5 flag, #6
2058 {
2059   \tex_pdfoutline:D~attr
2060   {
2061     /F~#5~
2062     \tl_if_blank:nF { #4 }{/C~[#4]}
2063   }~
2064   \tag_if_active:TF
2065     {goto~struct~name~{#6}~name~{#6}}
2066     {goto~name~{#6}}~
2067   count~\bool_if:nF{#3}{-}#2~
2068   {\exp_not:n{#7}}
2069 }
2070 </pdftex>
2071 <*luatex>
2072 \cs_new_protected:Npn \__pdf_backend_action:nnnnnnn #1#2#3#4#5#6#7
2073 % #1 level (ignored), #2 kid count, #3 open? #4 color (space separated values), #5 flag, #6
2074 {
2075   \tex_pdfextension:D~outline~attr
2076   {
2077     /F~#5~
2078     \tl_if_blank:nF { #4 }{/C~[#4]}
2079   }~
2080   user~{<<\exp_not:n{#6}>>}~
2081   count~\bool_if:nF{#3}{-}#2~
2082   {\exp_not:n{#7}}
2083 }
2084 \cs_new_protected:Npn \__pdf_backend_goto:nnnnnnn #1#2#3#4#5#6#7
2085 % #1 level (ignored), #2 kid count, #3 open? #4 color (space separated values), #5 flag, #6
2086 {
2087   \tex_pdfextension:D~outline~attr
2088   {
2089     /F~#5~
2090     \tl_if_blank:nF { #4 }{/Color~[#4]}
2091   }~
2092   \tag_if_active:TF
2093     {goto~struct~name~{#6}~name~{#6}}
2094     {goto~name~{#6}}~
2095   count~\bool_if:nF{#3}{-}#2~
2096   {\exp_not:n{#7}}
2097 }
2098 </luatex>
2099 <*xdvipdfmx | dvipdfmx>
2100 \cs_new_protected:Npn \__pdf_backend_action:nnnnnnn #1#2#3#4#5#6#7
2101 % #1 level, #2 kid count (ignored), #3 open? #4 color (space separated values), #5 flag, #6
2102 {
2103   \tex_special:D

```

```

2104 {
2105     pdf:out~
2106     [\bool_if:nF{#3}{-}]~#1~
2107     <<
2108         /A<<\exp_not:n{#6}>>~
2109         /Title~(\exp_not:n{#7})~
2110         /F~#5~
2111         \tl_if_blank:nF { #4 }{/C-[#4]}
2112     >>
2113 }
2114 }
2115 \cs_new_protected:Npn \__pdf_backend_goto:nnnnnnn #1#2#3#4#5#6#7
2116 % #1 level, #2 kid count (ignored), #3 open? #4 color (space separated values), #5 flag, #6
2117 {
2118     \tex_special:D
2119     {
2120         pdf:out~
2121         [\bool_if:nF{#3}{-}]~#1~
2122         <<
2123             /A
2124             <<
2125                 /S/GoTo/D(\exp_not:n{#6})
2126                 \tag_if_active:T
2127                 {
2128                     /SD~@pdf.SDest.#6
2129                 }
2130             >>~
2131             /Title~(\exp_not:n{#7})~
2132             /F~#5~
2133             \tl_if_blank:nF { #4 }{/C-[#4]}
2134         >>
2135     }
2136 }
2137 </xdvipdfmx | dvipdfmx>
2138 <*dvisvgm>
2139 \cs_new_protected:Npn \__pdf_backend_action:nnnnnnn #1#2#3#4#5#6#7{}
2140 \cs_new_protected:Npn \__pdf_backend_goto:nnnnnnn #1#2#3#4#5#6#7 {}
2141 </dvisvgm>

```

1.16 lua code for lualatex

```

2142 <lua>
2143 ltx= ltx or {}
2144 ltx.__pdf      = ltx.__pdf or {}
2145 ltx.__pdf.Page = ltx.__pdf.Page or {}
2146 ltx.__pdf.Page.dflt = ltx.__pdf.Page.dflt or {}
2147 ltx.__pdf.Page.Resources = ltx.__pdf.Resources or {}
2148 ltx.__pdf.Page.Resources.Properties = ltx.__pdf.Page.Resources.Properties or {}
2149 ltx.__pdf.Page.Resources.List={"ExtGState","ColorSpace","Pattern","Shading"}
2150 ltx.__pdf.object = ltx.__pdf.object or {}
2151
2152 ltx.pdf= ltx.pdf or {} -- for "public" functions
2153
2154 local __pdf = ltx.__pdf
2155 local pdf = pdf

```

```

2156
2157 local function __pdf_backend_Page_gput (name,value)
2158   __pdf.Page.dflt[name]=value
2159 end
2160
2161 local function __pdf_backend_Page_gremove (name)
2162   __pdf.Page.dflt[name]=nil
2163 end
2164
2165 local function __pdf_backend_Page_gclear ()
2166   __pdf.Page.dflt={}
2167 end
2168
2169 local function __pdf_backend_ThisPage_gput (page,name,value)
2170   __pdf.Page[page] = __pdf.Page[page] or {}
2171   __pdf.Page[page][name]=value
2172 end
2173
2174 local function __pdf_backend_ThisPage_gpush (page)
2175   local token=""
2176   local t = {}
2177   local tkeys= {}
2178   for name,value in pairs(__pdf.Page.dflt) do
2179     t[name]=value
2180   end
2181   if __pdf.Page[page] then
2182     for name,value in pairs(__pdf.Page[page]) do
2183       t[name] = value
2184     end
2185   end
2186   -- sort the table to get reliable test files.
2187   for name,value in pairs(t) do
2188     table.insert(tkeys,name)
2189   end
2190   table.sort(tkeys)
2191   for _,name in ipairs(tkeys) do
2192     token = token .. "/"..name.." " ..t[name]
2193   end
2194   return token
2195 end
2196
2197 function ltx.__pdf.backend_ThisPage_gput (page,name,value) -- tex.count["g_shipout_readonly_
2198   __pdf_backend_ThisPage_gput (page,name,value)
2199 end
2200
2201 function ltx.__pdf.backend_ThisPage_gpush (page)
2202   pdf.setpageattributes(__pdf_backend_ThisPage_gpush (page))
2203 end
2204
2205 function ltx.__pdf.backend_Page_gput (name,value)
2206   __pdf_backend_Page_gput (name,value)
2207 end
2208
2209 function ltx.__pdf.backend_Page_gremove (name)

```

```

2210 __pdf_backend_Page_gremove (name)
2211 end
2212
2213 function ltx.__pdf.backend_Page_gclear ()
2214 __pdf_backend_Page_gclear ()
2215 end
2216
2217
2218 local Properties = ltx.__pdf.Page.Resources.Properties
2219 local ResourceList= ltx.__pdf.Page.Resources.List
2220 local function __pdf_backend_PageResources_gpush (page)
2221 local token=""
2222 if Properties[page] then
2223 -- we sort the table, so that the pdf test works
2224 local t = {}
2225 for name,value in pairs (Properties[page]) do
2226 table.insert (t,name)
2227 end
2228 table.sort (t)
2229 for _,name in ipairs(t) do
2230 token = token .. "/"..name.." ".. Properties[page][name]
2231 end
2232 token = "/Properties <<"..token..">>"
2233 end
2234 for i,name in ipairs(ResourceList) do
2235 if ltx.__pdf.Page.Resources[name] then
2236 token = token .. "/"..name.." "..ltx.pdf.object_ref("__pdf/Page/Resources/"..name)
2237 end
2238 end
2239 return token
2240 end
2241
2242 -- the function is public, as I probably need it in tagpdf too ...
2243 function ltx.pdf.Page_Resources_Properties_gput (page,name,value) -- tex.count["g_shipout_re
2244 Properties[page] = Properties[page] or {}
2245 Properties[page][name]=value
2246 pdf.setpageresources(__pdf_backend_PageResources_gpush (page))
2247 end
2248
2249 function ltx.pdf.Page_Resources_gpush(page)
2250 pdf.setpageresources(__pdf_backend_PageResources_gpush (page))
2251 end
2252
2253 function ltx.pdf.object_ref (objname)
2254 if ltx.__pdf.object[objname] then
2255 local ref= ltx.__pdf.object[objname]
2256 return ref
2257 else
2258 return "false"
2259 end
2260 end
2261 </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.

A

\AddToDocumentProperties 1928, 1929
 \AddToHook 1890, 1935
 \AssignSocketPlug 933

B

bool commands:
 \bool_if:NTF
 672, 696, 752, 782, 984, 997,
 1005, 1013, 1021, 1083, 1096, 1104,
 1112, 1120, 1202, 1212, 1222, 1232
 \bool_if:nTF 2022, 2035,
 2053, 2067, 2081, 2095, 2106, 2121
 \bool_lazy_and:nnTF . 1533, 1764, 1980
 \bool_new:N 512, 936, 937, 938, 939, 940
 \bool_set_false:N . . . 976, 1075, 1201
 \bool_set_true:N 943, 944,
 945, 946, 947, 961, 1061, 1173, 1289
 box commands:
 \box_dp:N . 974, 1073, 1184, 1302, 1316
 \box_ht:N . 971, 1070, 1181, 1299, 1311
 \box_new:N 85, 86, 1170
 \box_scale:Nnn 1320
 \box_set_dp:Nn . 1185, 1266, 1363, 1387
 \box_set_ht:Nn . 1186, 1265, 1364, 1386
 \box_set_wd:Nn . 1187, 1264, 1365, 1385
 \box_use:N 1369
 \box_use_drop:N 1198, 1267, 1344, 1388
 \box_wd:N . 968, 1067, 1178, 1296, 1306

C

clist commands:
 \clist_const:Nn 414
 \clist_map_function:NN 857
 \clist_map_inline:Nn 423, 457, 469, 652
 \clist_map_inline:nn . 975, 1074, 1200
 cs commands:
 \cs_generate_variant:Nn
 28, 31, 32, 35, 36, 76, 77, 411
 \cs_gset_eq:NN
 634, 1413, 1414, 1415, 1419, 1420, 1421
 \cs_if_exist:NTF 426, 916
 \cs_new:Npn 71, 97, 103,
 242, 833, 1044, 1145, 1250, 1274, 1390
 \cs_new_protected:Npn
 39, 43, 53, 65, 147, 154,
 169, 175, 181, 188, 195, 204, 224,
 247, 257, 271, 283, 300, 311, 318,

325, 334, 343, 350, 357, 364, 373,
 382, 390, 393, 399, 404, 407, 438,
 449, 455, 477, 481, 493, 496, 497,
 501, 504, 505, 509, 523, 545, 566,
 650, 742, 842, 866, 873, 880, 889,
 893, 896, 899, 911, 941, 952, 1037,
 1052, 1136, 1159, 1255, 1272, 1273,
 1280, 1374, 1411, 1417, 1476, 1714,
 1884, 1933, 1945, 1956, 1962, 1965,
 1971, 1977, 1984, 1990, 1997, 2003,
 2007, 2010, 2016, 2029, 2044, 2056,
 2072, 2084, 2100, 2115, 2139, 2140
 \cs_new_protected:Npx 163
 \cs_set_eq:NN
 . 520, 521, 553, 554, 642, 729, 735,
 819, 825, 1425, 1426, 1427, 1428, 1429
 \cs_set_protected:Npn
 516, 529, 533, 537, 541,
 551, 556, 558, 560, 562, 564, 577,
 596, 615, 621, 627, 632, 638, 644,
 667, 691, 715, 719, 724, 731, 737,
 745, 775, 806, 810, 815, 821, 828,
 918, 922, 1432, 1522, 1527, 1537,
 1576, 1597, 1606, 1645, 1666, 1673,
 1757, 1768, 1803, 1825, 1860, 1877

D

dim commands:
 \dim_eval:n 1525, 1581,
 1582, 1583, 1592, 1593, 1594, 1650,
 1651, 1652, 1661, 1662, 1663, 1760,
 1808, 1809, 1810, 1816, 1817, 1818,
 1865, 1866, 1867, 1873, 1874, 1875
 \dim_to_decimal_in_sp:n
 1306, 1311, 1316
 \c_zero_dim
 . . 1185, 1186, 1187, 1363, 1364, 1365
 \directlua 94, 1604, 1823, 1995

E

exp commands:
 \exp_after:wN
 . . 1698, 1746, 1787, 1813, 1844, 1870
 \exp_args:Ne . 680, 704, 1453, 1459,
 1504, 1510, 1524, 1554, 1559, 1584,
 1589, 1623, 1628, 1653, 1658, 1759
 \exp_args:NNe 844
 \exp_not:n 601, 695, 779, 1436, 1457,
 1508, 1677, 1696, 1744, 2021, 2025,

2034, 2038, 2052, 2054, 2068, 2080, 2082, 2096, 2108, 2109, 2125, 2131	
F	
\footins 931	
fp commands:	
\fp_eval:n 1450, 1471, 1552, 1572, 1621, 1641, 1691, 1710, 1783, 1800, 1840, 1857	
G	
group commands:	
\group_begin: 1947	
\group_end: 1952	
H	
hbox commands:	
\hbox:n 1481, 1492, 1719, 1730	
\hbox_gset:Nn 1171	
\hbox_set:Nn 959, 1059, 1257, 1287, 1321, 1376	
hook commands:	
\hook_gput_code:nnn ... 139, 472, 1366	
\hook_gput_next_code:nn 1188	
\hook_gset_rule:nnnn 466, 467	
I	
int commands:	
\int_compare:nNnTF 986, 1085, 1604, 1823, 1995	
\int_compare_p:nNn 1534, 1535, 1765, 1766, 1981, 1982	
\int_const:Nn .. 1032, 1131, 1167, 1283	
\int_eval:n 1938, 1949	
\int_gincr:N 207, 579, 598, 669, 693, 747, 751, 777, 781, 1166, 1282	
\int_if_exist:NnTF 1400	
\int_new:N 89, 90, 91	
\int_use:N 208, 211, 582, 590, 601, 609, 671, 676, 685, 695, 700, 709, 749, 756, 760, 763, 771, 779, 786, 790, 793, 801, 1040, 1046, 1138, 1146, 1252, 1330, 1357, 1381, 1392, 1558, 1588, 1627, 1657	
K	
kernel internal commands:	
_kernel_backend_literal:n 31, 80, 580, 584, 599, 603, 617, 629, 646, 656, 1891, 1892, 1898, 1906	
_kernel_backend_literal_page:n 28, 670, 694, 717, 726, 739, 748, 778, 808, 817, 830	
_kernel_backend_postscript:n 35, 1323, 1345, 1351, 1378	
_kernel_backend_shipout_- literal:n 39, 39, 525, 640	
_kernel_backend_shipout_- literal_page:n ... 53, 53, 733, 823	
_kernel_backend_shipout_- literal_pdf:n 43, 43	
_kernel_kern:n 1480, 1488, 1491, 1520, 1718, 1726, 1729, 1755	
_kernel_pdf_name_from_unicode_- e:n 97, 103	
_kernel_pdf_object_id_indexed:nn 1787, 1813, 1844, 1870	
_kernel_pdftdict_name:n 226, 227, 229, 484, 654, 836, 847, 852, 962, 987, 1000, 1008, 1016, 1024, 1062, 1086, 1099, 1107, 1115, 1123, 1290	
_kernel_pdfmanagement_end_- run_code_tl 112, 119, 126	
_kernel_pdfmanagement_- thispage_shipout_code_tl 135, 141	
L	
lualua commands:	
\lualua: 201, 280, 331, 370	
lua commands:	
\lua_load_module:n 929	
M	
mode commands:	
\mode_leave_vertical: 1190, 1368	
N	
\NewSocketPlug 930	
P	
pdf commands:	
\pdf_activate_indexed_structure_- destination: 1410, 1417	
\pdf_activate_structure_destination: 1410, 1411	
\l_pdf_current_structure_- destination_tl 1407, 1453, 1459, 1504, 1510, 1554, 1559, 1584, 1589, 1623, 1628, 1653, 1658, 1698, 1746, 1787, 1813, 1844, 1870	
\pdf_object_if_exist:nTF 1453, 1504, 1554, 1584, 1623, 1653	
\pdf_object_new:n 425, 471	
\pdf_object_ref:n 432, 489, 531, 591, 659, 677, 686, 757, 772, 838, 1002, 1010, 1018, 1026, 1101, 1109, 1117, 1125, 1208, 1218, 1228, 1239, 1459, 1510	
\pdf_object_ref_indexed:nn 1698, 1746	
\pdf_object_ref_last: .. 869, 876, 883	

```

\pdf_object_unnamed_write:nn ...
..... 623, 721, 812, 868, 875, 882
\pdf_object_write ..... 486
\pdf_object_write:nnn . 459, 475, 1958
pdf internal commands:
  \__pdf_backend:n .....
    32, 171, 479, 487, 883, 1191, 1199,
    1204, 1214, 1224, 1234, 1243, 1259,
    1434, 1455, 1483, 1484, 1494, 1506,
    1675, 1694, 1721, 1722, 1732, 1742
  \__pdf_backend_action:nnnnnn ..
    ..... 2016, 2044, 2072, 2100, 2139
  \__pdf_backend_bdc:nn .....
    ..... 13, 511, 516, 520, 521,
    551, 553, 554, 632, 634, 635, 729, 819
  \__pdf_backend_bdc_contobj:nn ..
    ..... 520, 553, 621, 634, 719, 810
  \__pdf_backend_bdc_contstream:nn
    .... 521, 554, 627, 724, 729, 815, 819
  \__pdf_backend_bdc_shipout:nn ..
    ..... 523, 642, 735, 825
  \__pdf_backend_bdc_shipout_-
    contstream:nn .....
    ..... 638, 642, 731, 735, 821, 825
  \__pdf_backend_bdcobject:n ....
    ..... 13, 511,
    533, 560, 596, 624, 691, 722, 775, 813
  \__pdf_backend_bdcobject:nn ....
    ..... 13, 511, 529, 558, 577, 667, 745
  \__pdf_backend_bmc:n .....
    ..... 13, 511, 541, 564, 615, 715, 806
  \__pdf_backend_catalog_gput:nn .. 20
  \__pdf_backend_destination:nn ..
    ..... 1413, 1419, 1425, 1428
  \__pdf_backend_destination:nnnn
    ..... 1414, 1420, 1426, 1429
  \__pdf_backend_emc: .....
    ..... 13, 511, 537, 562, 644, 737, 828
  \__pdf_backend_goto:nnnnnn .....
    ..... 2029, 2056, 2084, 2115, 2140
  \__pdf_backend_indexed_structure_-
    destination:nn .....
    .. 1419, 1428, 1672, 1673, 1768, 1825
  \__pdf_backend_indexed_structure_-
    destination:nnnn .....
    .. 1420, 1429, 1672, 1757, 1803, 1860
  \__pdf_backend_indexed_structure_-
    destination_aux:nnnn . 1714, 1759
  \__pdf_backend_luastring:n ....
    158, 242, 251, 263, 264, 275, 290, 291
  \__pdf_backend_metadata_stream:n
    ..... 1933, 1945, 1956
  \g_pdf_backend_name_int .....
    ..... 88, 579, 582, 590,
    598, 601, 609, 669, 671, 676, 685,
    693, 695, 700, 709, 747, 749, 777, 779
  \__pdf_backend_Names_gpush:nn ..
    ..... 866, 873, 880, 889, 893
  \__pdf_backend_NamesEmbeddedFiles_-
    add:nn ..... 895, 896, 899, 911
  \g_pdf_backend_object_int ....
    ..... 1166, 1169, 1282, 1285, 1330
  \__pdf_backend_object_last: ....
    ..... 535, 610, 701, 710, 787, 802
  \__pdf_backend_object_write:nn .
    ..... 1939, 1951
  \__pdf_backend_omit_charset:n ..
    ..... 1961, 1962, 1965, 1971
  \__pdf_backend_omit_cidset:n ...
    ..... 2006, 2007, 2010
  \__pdf_backend_omit_info:n ....
    .. 1976, 1977, 1984, 1990, 1997, 2003
  \__pdf_backend_Page_gput:nn ....
    ..... 6, 178, 188, 257, 318, 357, 393
  \__pdf_backend_Page_gremove:n ..
    ..... 6, 178, 195, 271, 325, 364, 399
  \g_pdf_backend_page_int ..... 88
  \__pdf_backend_Page_primitive:n
    ..... 6, 178, 181, 234, 247,
    311, 336, 345, 350, 375, 384, 390, 411
  \__pdf_backend_PageResources:n .
    ..... 477, 496, 504
  \c_pdf_backend_PageResources_-
    clist .. 413, 423, 457, 469, 652, 858
  \__pdf_backend_PageResources_-
    gpush:n .....
    ..... 13, 511, 545, 566, 650, 742, 842
  \__pdf_backend_PageResources_-
    gpush_aux:n ..... 833, 859
  \__pdf_backend_PageResources_-
    gput:nnn 422, 438, 449, 481, 497, 505
  \__pdf_backend_PageResources_-
    obj_gpush: . 422, 455, 493, 501, 509
  \__pdf_backend_Pages_primitive:n
    ..... 146, 147, 154, 163, 169, 175
  \__pdf_backend_pdfmark:n .... 36,
    518, 531, 535, 539, 543, 901, 2019, 2032
  \__pdf_backend_record_abspage:n
    ..... 65, 76, 208, 760, 790
  \__pdf_backend_ref_abspage:n ...
    ..... 71, 77, 211, 763, 793
  \g_pdf_backend_resourceid_int .
    ..... 88, 207, 208, 211, 751, 756,
    760, 763, 771, 781, 786, 790, 793, 801
  \__pdf_backend_set_regression_-
    data: ..... 1884
  \__pdf_backend_shipout_bdc:nn ..
    ..... 13, 511, 556

```

```

\__pdf_backend_structure_-
  destination:nn .....
  .. 1413, 1425, 1431, 1432, 1537, 1606
\__pdf_backend_structure_-
  destination:nnnn .....
  .. 1414, 1426, 1431, 1522, 1576, 1645
\__pdf_backend_structure_-
  destination_aux:nnnn . 1476, 1524
\__pdf_backend_ThisPage_gpush:n
  ..... 6, 178, 224, 300, 343, 382, 407
\__pdf_backend_ThisPage_gput:nn
  ..... 6, 178, 204, 283, 334, 373, 404
\g__pdf_backend_thispage_-
  shipout_tl ..... 6
\l__pdf_backend_tmpa_box .....
  .... 82, 959, 968, 971, 974, 1030,
  1059, 1067, 1070, 1073, 1129, 1257,
  1264, 1265, 1266, 1267, 1287, 1296,
  1299, 1302, 1306, 1311, 1316, 1320,
  1344, 1376, 1385, 1386, 1387, 1388
\l__pdf_backend_tmpb_box .....
  .... 86, 1321, 1363, 1364, 1365, 1369
\l__pdf_backend_xform_bool ....
  ..... 512, 672,
  696, 752, 782, 961, 1061, 1173, 1289
\__pdf_backend_xform_if_exist:n
  ..... 1398, 1404
\__pdf_backend_xform_new:nnnn .. 935
\__pdf_backend_xform_new:nnnnn .
  ..... 952, 1052, 1159, 1272, 1280
\__pdf_backend_xform_ref:n ....
  ..... 935, 1044,
  1145, 1193, 1250, 1261, 1274, 1390
\l__pdf_backend_xform_res_-
  colorspace_bool .....
  ..... 939, 946, 1021, 1120, 1232
\l__pdf_backend_xform_res_-
  extgstate_bool .....
  ..... 937, 944, 997, 1096, 1202
\l__pdf_backend_xform_res_-
  pattern_bool .....
  ..... 936, 943, 1005, 1104, 1212
\l__pdf_backend_xform_res_-
  properties_bool 938, 945, 984, 1083
\__pdf_backend_xform_res_reset:
  ..... 941, 949, 1031, 1130, 1244
\l__pdf_backend_xform_res_-
  shading_bool .....
  ..... 940, 947, 1013, 1112, 1222
\l__pdf_backend_xform_tmpdp_tl .
  ..... 1278, 1314, 1328, 1335
\l__pdf_backend_xform_tmpht_tl .
  ..... 1279, 1309, 1333

\l__pdf_backend_xform_tmpwd_tl .
  ..... 1277, 1304, 1334
\__pdf_backend_xform_use:n ....
  .... 935, 1037, 1136, 1255, 1273, 1374
\__pdf_object_retrieve:n . 1938, 1949
\g__pdf_tmpa_prop .. 82, 226, 231, 236
\l__pdf_tmpa_tl .....
  ..... 82, 209, 213, 215, 218, 761,
  765, 767, 770, 791, 795, 797, 800, 803
pdfannot internal commands:
  \__pdfannot_backend_link_begin:n
    ..... 1529
  \__pdfannot_backend_link_-
    begin:nnnw ..... 1599, 1668, 1879
  \__pdfannot_backend_link_begin_-
    goto:nnw ..... 1415, 1421, 1427
  \__pdfannot_backend_link_begin_-
    structure_goto:nnw 1415, 1421,
    1427, 1431, 1527, 1597, 1666, 1877
  \__pdfannot_backend_link_off: .. 918
  \__pdfannot_backend_link_on: ... 922
pdfdict commands:
  \pdfdict_gput:nnn .....
    . 190, 218, 320, 359, 395, 440, 451,
    499, 507, 674, 698, 754, 769, 784, 799
  \pdfdict_gremove:nn 197, 327, 366, 401
  \pdfdict_if_exist:nTF .. 213, 765, 795
  \pdfdict_item:nn ..... 236, 838, 853
  \pdfdict_new:n ..... 215, 767, 797
  \pdfdict_put:nnn ..... 1927
  \pdfdict_show:n ..... 803
  \pdfdict_use:n 346, 385, 461, 993, 1092
\pdfextension ..... 931
\pdfliteral ..... 2
pdfmanagement commands:
  \pdfmanagement_add:nnn 1887, 1890,
  1893, 1894, 1897, 1905, 1913, 1918
\pdfnames ..... 20
\pdfomitinfodict ..... 1986
\pdfpageref ..... 3
\pdfrunninglinkoff ..... 916, 920
\pdfrunninglinkon ..... 924
\pdftrailerid ..... 1915
pdfxform commands:
  \pdfxform_dp:n ..... 1196, 1266, 1387
  \pdfxform_ht:n ..... 1195, 1265, 1386
  \pdfxform_if_exist:n ..... 1404
  \pdfxform_wd:n ..... 1194, 1264, 1385
prg commands:
  \prg_new_conditional:Npnn .... 1398
  \prg_new_eq_conditional:NNn ... 1404
  \prg_return_false: ..... 1402
  \prg_return_true: ..... 1401

```

prop commands:		\g@addto@macro 132, 133
\prop_count:N 987, 1086		\special 2
\prop_gclear:N 962, 1062, 1290	tex commands:	
\prop_gput:Nnn 231, 484	\tex_directlua:D	
\prop_gset_eq:NN 226	 156, 259, 273, 426, 428, 441, 442	
\prop_if_empty:NTF	\tex_global:D 149, 183, 844	
. 654, 835, 999, 1007,	\tex_immediate:D 978, 1077, 1937, 1949	
1015, 1023, 1098, 1106, 1114, 1122	\tex_latelua:D 249, 285, 302, 680, 704	
\prop_if_exist:NTF 227, 846	\tex_luaescapestring:D 244	
\prop_map_function:NN 236, 851	\tex_pdfcompresslevel:D 1948	
\prop_map_inline:Nn 229	\tex_pdfdest:D 1539, 1556,	
\prop_new:N 83	1578, 1586, 1770, 1785, 1805, 1811	
property commands:	\tex_pdfextension:D 46, 56,	
\property_record:nn 68	876, 1608, 1625, 1647, 1655, 1827,	
\property_ref:nn 73	1842, 1862, 1868, 1937, 2075, 2087	
\ProvidesExplFile 1	\tex_pdflastxform:D 1034, 1133	
	\tex_pdfliteral:D 49, 59	
	\tex_pdfnames:D 869	
	\tex_pdfobj:D 1949	
	\tex_pdfomitcharset:D 1967	
	\tex_pdfoutline:D 2047, 2059	
	\tex_pdfpageattr:D 183	
	\tex_pdfpageresources:D 844	
	\tex_pdfpagesattr:D 149	
	\tex_pdfrefxform:D 1039, 1138	
	\tex_pdfsuppressptexinfo:D 1914	
	\tex_pdftexrevision:D 1535, 1766, 1982	
	\tex_pdftexversion:D 1534, 1765, 1981	
	\tex_pdfvariable:D	
	 1919, 1920, 1973, 1999, 2012	
	\tex_pdfxform:D 978, 1077	
	\tex_special:D	
	 40, 165, 313, 352, 2103, 2118	
	\tex_the:D	
	 968, 971, 974, 1067, 1070, 1073,	
	1178, 1181, 1184, 1296, 1299, 1302	
	\tex_unexpanded:D 244	
	\tex_vss:D 1486, 1518, 1724, 1753	
	text commands:	
	\text_expand:n 99, 105	
	tl commands:	
	\c_space_tl 582, 590, 601, 609, 671,	
	695, 749, 779, 1194, 1195, 1196, 1330	
	\tl_const:Nn	
	 966, 969, 972, 1065, 1068, 1071,	
	1176, 1179, 1182, 1294, 1297, 1300	
	\tl_gput_right:Nn 110, 117, 124	
	\tl_if_blank:nTF 2023, 2036,	
	2050, 2062, 2078, 2090, 2111, 2133	
	\tl_if_exist:NTF 130	
	\tl_new:N 84, 1277, 1278, 1279, 1408	
	\tl_set:Nn	
	 209, 761, 791, 1304, 1309, 1314	
R		
\relax 132, 1919		
S		
scan commands:		
\scan_stop: 1041,		
1142, 1553, 1573, 1583, 1594, 1622,		
1642, 1652, 1663, 1784, 1801, 1810,		
1818, 1841, 1858, 1867, 1875, 1914,		
1948, 1967, 1973, 1986, 1999, 2012		
\setbox 931		
\special 2		
str commands:		
\str_case:nnTF		
1439, 1460, 1541, 1561, 1610, 1630,		
1680, 1699, 1772, 1789, 1829, 1846		
\str_convert_pdfname:n 99, 485		
\str_if_eq:nnTF 1338, 1349		
sys commands:		
\c_sys_engine_exec_str 1929		
\sys_gset_rand_seed:n 1886		
\c_sys_timestamp_str 1893, 1894, 1927		
T		
tag commands:		
\tag_if_active:TF 2064, 2092, 2126		
TeX and L ^A T _E X 2 _ε commands:		
\@bsphack 67		
\@esphack 69		
\@kernel@after@enddocument@afterlastpage		
. 109, 110		
\@kernel@after@shipout@background		
. 130, 133		
\@kernel@after@shipout@lastpage		
. 116, 117, 123, 124		
\@kernel@before@shipout@background		
. 132		

<code>\tl_to_str:n</code>	U	
..... 967, 970, 973, 1033, 1040,	<code>\unvbox</code>	931
1046, 1066, 1069, 1072, 1132, 1140,		
1146, 1168, 1177, 1180, 1183, 1252,	V	
1284, 1295, 1298, 1301, 1357, 1381,	<code>\vbox</code>	931
1392, 1400, 1559, 1589, 1628, 1658	<code>vbox commands:</code>	
<code>\tl_use:N</code> 1328, 1333, 1334, 1335	<code>\vbox_to_zero:n</code>	1478, 1489, 1716, 1727