

The l3backend-testphase package

Additional backend PDF features

L^AT_EX PDF management testphase bundle

The L^AT_EX Project*

Version 0.96n, released 2024-10-27

1 l3backend-testphase Implementation

```
1 <drivers>\ProvidesExplFile
2 <*dvipdfmx>
3   {l3backend-testphase-dvipdfmx.def}{2024-10-27}{ }
4   {LaTeX-PDF-management-testphase-bundle-backend-support: dvipdfmx}
5 </dvipdfmx>
6 <*dvips>
7   {l3backend-testphase-dvips.def}{2024-10-27}{ }
8   {LaTeX-PDF-management-testphase-bundle-backend-support: dvips}
9 </dvips>
10 <*dvisvgm>
11   {l3backend-testphase-dvisvgm.def}{2024-10-27}{ }
12   {LaTeX-PDF-management-testphase-bundle-backend-support: dvisvgm}
13 </dvisvgm>
14 <*luatex>
15   {l3backend-testphase-luatex.def}{2024-10-27}{ }
16   {LaTeX-PDF-management-testphase-bundle-backend-support: PDF output (LuaTeX)}
17 </luatex>
18 <*pdftex>
19   {l3backend-testphase-pdftex.def}{2024-10-27}{ }
20   {LaTeX-PDF-management-testphase-bundle-backend-support: PDF output (pdfTeX)}
21 </pdftex>
22 <*xdvipdfmx>
23   {l3backend-testphase-xetex.def}{2024-10-27}{ }
24   {LaTeX-PDF-management-testphase-bundle-backend-support: XeTeX}
25 </xdvipdfmx>
```

1.1 Variants

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

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

*E-mail: latex-team@latex-project.org

```

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

```

1.2 Support for delayed literal and special

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

```
38 <*drivers>
```

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

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

```

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

```

(End of definition for _kernel_backend_shipout_literal:e.)

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

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

```

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

```

(End of definition for _kernel_backend_shipout_literal_pdf:e.)

`_kernel_backend_shipout_literal_page:e` Page literals are pretty simple.

```

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

```

```

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

```

(End of definition for _kernel_backend_shipout_literal_page:e.)

1.3 Crossreferences

Commands to get a reference for the absolute page counter.

```

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

```

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

```

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

```

```

\_pdf_backend_resourceid_int
\_pdf_backend_name_int
\_pdf_backend_page_int

```

Some scratch variables

```

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

```

(End of definition for _pdf_backend_resourceid_int, _pdf_backend_name_int, and _pdf_backend_tmpa_box.)

```

\_pdf_backend_resourceid_int
\_pdf_backend_name_int
\_pdf_backend_page_int

```

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

```

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

```

(End of definition for _pdf_backend_resourceid_int, _pdf_backend_name_int, and _pdf_backend_page_int.)

1.4 luacode

Load the lua code.

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

1.5 Converting unicode strings to a pdfname

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

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

1.6 Hooks

1.6.1 Add the “end run” hooks

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

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

1.6.2 Add the “shipout” hooks

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

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

1.7 The /Pages dictionary (pdfpagesattr)

_pdf_backend_Pages_primitive:n

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

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

```

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

```

(End of definition for __pdf_backend_Pages_primitive:n.)

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

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

```

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

```

```

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

```

```

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

```



```

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

```

```

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

```

(End of definition for __pdf_backend_Page_primitive:n and others.)

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

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

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

```

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

```

(End of definition for `\c__pdf_backend_PageResources_clist`.)

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

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

#1 : name of the resource (ExtGState, ColorSpace, Shading, Pattern)

#2 : a pdf name without slash

#3 : value

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

`_pdf_backend_PageResources_obj_gpush:`

```

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

```

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

```

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

```

```

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

```

code for end of document code

```

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

```

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

```

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

```

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

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

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

dvipsvgm unneeded, or no-op

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

(End of definition for __pdf_backend_PageResources_gput:nnn and __pdf_backend_PageResources_obj_gpush:.)

1.9.1 Page resources /Properties + BDC operators

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

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

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

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

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

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

```

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

```

```

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

```

```

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

```



```

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

```

```

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

```

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

```

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

```

```

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

```

We use by default the direct BDC.

```

842  \cs_set_eq:NN \_pdf_backend_bdc:nn \_pdf_backend_bdc_contstream:nn
843
844  \bool_if:NT\l__pdfmanagement_delayed_shipout_bool
845  {
846    \cs_set_protected:Npn \_pdf_backend_bdc_shipout_contstream:ee #1 #2
847    {
848      \_kernel_backend_shipout_literal_page:e { /#1~<<#2>>~BDC }
849    }
850    \cs_set_eq:NN \_pdf_backend_bdc_shipout:ee \_pdf_backend_bdc_shipout_contstream:ee
851  }
852
853  \cs_set_protected:Npn \_pdf_backend_emc:

```

```

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

```

(End of definition for __pdf_backend_bdc:nn and others.)

1.10 “Catalog” & subdirectories (pdfcatalog)

The backend command is already in the driver: __pdf_backend_catalog_gput:nn

1.10.1 Special case: the /Names/EmbeddedFiles dictionary

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

```

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

```

```

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

```

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

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

```

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

```

(End of definition for __pdf_backend_NamesEmbeddedFiles_add:nn.)

1.10.2 Additional annotation commands

Starting with texlive 2021 pdftex and luatex offer commands to interrupt a link. That can for example be used to exclude the header and footer from the link. We add here backend support for this.

```
940 <*drivers>
941 \cs_new_protected:Npn \__pdf_backend_link_off: {}
942 \cs_new_protected:Npn \__pdf_backend_link_on: {}
943 </drivers>
944 <*pdftex>
945 \cs_if_exist:NT \pdfrunninglinkoff
946 {
947   \cs_set_protected:Npn \__pdf_backend_link_off:
948     {
949       \pdfrunninglinkoff
950     }
951   \cs_set_protected:Npn \__pdf_backend_link_on:
952     {
953       \pdfrunninglinkon
954     }
955 }
956 </pdftex>
957 <*luatex>
958 \int_compare:nNnT {\tex_luatexversion:D } > {112}
959 {
960   \cs_set_protected:Npn \__pdf_backend_link_off:
961     {
962       \pdfextension linkstate 1
963     }
964   \cs_set_protected:Npn \__pdf_backend_link_on:
965     {
966       \pdfextension linkstate 0
967     }
968 }
969 </luatex>
970 <*dviPDFmx | xdvipdfmx>
971 \cs_set_protected:Npn \__pdf_backend_link_off:
972   {
973     \__pdf_backend:n { noline }
974   }
975 \cs_set_protected:Npn \__pdf_backend_link_on:
976   {
977     \__pdf_backend:n { link }
978   }
979 </dviPDFmx | xdvipdfmx>
```

1.10.3 Form XObject / backend

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

```
\__pdf_backend_xform_use:n
\__pdf_backend_xform_ref:n
```

```

980 <*pdfTeX>
981 \cs_new_protected:Npn \l__pdf_backend_xform_new:nnnn #1 #2 #3 #4
982 % #1 name
983 % #2 attributes
984 % #3 resources
985 % #4 content, not necessarily a box!
986 {
987   \hbox_set:Nn \l__pdf_backend_tmpa_box
988     {
989       \bool_set_true:N \l__pdf_backend_xform_bool
990       \prop_gc_clear:c { \__kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Properties } }
991       #4
992     }
993   %store the dimensions
994   \tl_const:ce
995     { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
996     { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
997   \tl_const:ce
998     { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
999     { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
1000   \tl_const:ce
1001     { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1002     { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
1003   %% do we need to test if #2 and #3 are empty??
1004   \tex_immediate:D \tex_pdfxform:D
1005     ~ attr ~ { #2 }
1006   %% which other resources should be default? Is an argument actually needed?
1007   ~ resources ~
1008   {
1009     #3
1010     \int_compare:nNnT
1011       { \prop_count:c { \__kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Properties } }
1012         >
1013         { 0 }
1014         {
1015           /Properties~
1016           <<
1017             \pdfdict_use:n { g__pdf_Core/Xform/Resources/Properties }
1018           >>
1019         }
1020
1021     \prop_if_empty:cF
1022       { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/ExtGState } }
1023       {
1024         /ExtGState~ \pdf_object_ref:n { __pdf/Page/Resources/ExtGState }
1025       }
1026     \prop_if_empty:cF
1027       { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/Pattern } }
1028       {
1029         /Pattern~ \pdf_object_ref:n { __pdf/Page/Resources/Pattern }
1030       }
1031     \prop_if_empty:cF
1032       { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/Shading } }
1033     {

```

```

1034         /Shading~ \pdf_object_ref:n { __pdf/Page/Resources/Shading }
1035     }
1036     \prop_if_empty:cF
1037     { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/ColorSpace } }
1038     {
1039         /ColorSpace~ \pdf_object_ref:n { __pdf/Page/Resources/ColorSpace }
1040     }
1041 }
1042 \l__pdf_backend_tmpa_box
1043 \int_const:cn
1044 { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1045 { \tex_pdflastxform:D }
1046 }
1047
1048 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
1049 {
1050     \tex_pdfrefxform:D
1051     \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1052     \scan_stop:
1053 }
1054
1055 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1056 {
1057     \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int } ~ 0 ~ R
1058 }
1059 </pdftex>
1060 <*luatex>
1061 %luatex
1062 %nearly identical but not completely ...
1063 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4
1064 % #1 name
1065 % #2 attributes
1066 % #3 resources
1067 % #4 content, not necessarily a box!
1068 {
1069     \hbox_set:Nn \l__pdf_backend_tmpa_box
1070     {
1071         \bool_set_true:N \l__pdf_backend_xform_bool
1072         \prop_gclear:c { \__kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Properties } }
1073         #4
1074     }
1075     \tl_const:ce
1076     { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1077     { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
1078     \tl_const:ce
1079     { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1080     { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
1081     \tl_const:ce
1082     { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1083     { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
1084     %% do we need to test if #2 and #3 are empty??
1085     \tex_immediate:D \tex_pdfxform:D
1086     ~ attr ~ { #2 }
1087     %% which resources should be default? Is an argument actually needed?

```



```

1088 ~ resources ~
1089 {
1090   #3
1091   \int_compare:nNnT
1092     {\prop_count:c { \__kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Properties
1093     >
1094     { 0 }
1095     {
1096       /Properties~
1097       <<
1098       \pdfdict_use:n { g__pdf_Core/Xform/Resources/Properties }
1099       >>
1100     }
1101   \prop_if_empty:cF
1102     { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/ExtGState } }
1103     {
1104       /ExtGState~ \pdf_object_ref:n { __pdf/Page/Resources/ExtGState }
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   \prop_if_empty:cF
1112     { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/Shading } }
1113     {
1114       /Shading~ \pdf_object_ref:n { __pdf/Page/Resources/Shading }
1115     }
1116   \prop_if_empty:cF
1117     { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/ColorSpace } }
1118     {
1119       /ColorSpace~ \pdf_object_ref:n { __pdf/Page/Resources/ColorSpace }
1120     }
1121   }
1122   \l__pdf_backend_tmpa_box
1123   \int_const:cn
1124     { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1125     { \tex_pdflastxform:D }
1126 }
1127
1128 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1 %protected as with xelatex
1129 {
1130   \tex_pdfrefxform:D \int_use:c
1131   {
1132     c__pdf_backend_xform_ \tl_to_str:n {#1} _int
1133   }
1134   \scan_stop:
1135 }
1136
1137 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1138 { \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int } ~ 0 ~ R }
1139
1140 </luatex>
1141 <*dviPDFmx | xdvipdfmx>

```

```

1142 % xetex
1143 % it needs a bit testing if it really works to set the box to 0 before the special ...
1144 % does it disturb viewing the xobject?
1145 % what happens with the resources (bdc)? (should work as they are specials too)
1146 % xetex requires that the special is in horizontal mode. This means it affects
1147 % typesetting. But we can no delay the whole form code to shipout
1148 % as the object reference and the size is often wanted on the current page.
1149 % so we need to allocate a box - but probably they won't be thousands xform
1150 % in a document so it shouldn't matter.
1151 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4
1152 % #1 name
1153 % #2 attributes
1154 % #3 resources
1155 % #4 content, not necessarily a box!
1156 {
1157   \int_gincr:N \g__pdf_backend_object_int
1158   \int_const:cn
1159     { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1160     { \g__pdf_backend_object_int }
1161   \box_new:c { g__pdf_backend_xform_#1_box }
1162   \hbox_gset:cn { g__pdf_backend_xform_#1_box }
1163     {
1164       \bool_set_true:N \l__pdf_backend_xform_bool
1165       #4
1166     }
1167   \tl_const:ce
1168     { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1169     { \tex_the:D \box_wd:c { g__pdf_backend_xform_#1_box } }
1170   \tl_const:ce
1171     { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1172     { \tex_the:D \box_ht:c { g__pdf_backend_xform_#1_box } }
1173   \tl_const:ce
1174     { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1175     { \tex_the:D \box_dp:c { g__pdf_backend_xform_#1_box } }
1176   \box_set_dp:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1177   \box_set_ht:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1178   \box_set_wd:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1179   \hook_gput_next_code:nn {shipout/background}
1180   {
1181     \mode_leave_vertical: %needed, the xform disappears without it.
1182     \__pdf_backend:e
1183     {
1184       bxobj ~ \__pdf_backend_xform_ref:n { #1 }
1185       \c_space_tl width ~ \pdfxform_wd:n { #1 }
1186       \c_space_tl height ~ \pdfxform_ht:n { #1 }
1187       \c_space_tl depth ~ \pdfxform_dp:n { #1 }
1188     }
1189     \box_use_drop:c { g__pdf_backend_xform_#1_box }
1190     \__pdf_backend:e {put ~ @resources ~<<#3>> }
1191     \__pdf_backend:e
1192     {
1193       put~ @resources ~
1194       <<
1195       /ExtGState~ \pdf_object_ref:n { __pdf/Page/Resources/ExtGState }

```

```

1196         >>
1197     }
1198     \__pdf_backend:e
1199     {
1200         put~ @resources ~
1201         <<
1202             /Pattern~ \pdf_object_ref:n { __pdf/Page/Resources/Pattern }
1203         >>
1204     }
1205     \__pdf_backend:e
1206     {
1207         put~ @resources ~
1208         <<
1209             /Shading~ \pdf_object_ref:n { __pdf/Page/Resources/Shading }
1210         >>
1211     }
1212     \__pdf_backend:e
1213     {
1214         put~ @resources ~
1215         <<
1216             /ColorSpace~
1217             \pdf_object_ref:n { __pdf/Page/Resources/ColorSpace }
1218         >>
1219     }
1220     \__pdf_backend:e {exobj ~<<#2>>}
1221 }
1222 }
1223
1224
1225
1226 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1227 {
1228     @pdf.xform \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1229 }
1230
1231 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
1232 {
1233     \hbox_set:Nn \l__pdf_backend_tmpa_box
1234     {
1235         \__pdf_backend:e
1236         {
1237             uxobj~ \__pdf_backend_xform_ref:n { #1 }
1238         }
1239     }
1240     \box_set_wd:Nn \l__pdf_backend_tmpa_box { \pdfxform_wd:n { #1 } }
1241     \box_set_ht:Nn \l__pdf_backend_tmpa_box { \pdfxform_ht:n { #1 } }
1242     \box_set_dp:Nn \l__pdf_backend_tmpa_box { \pdfxform_dp:n { #1 } }
1243     \box_use_drop:N \l__pdf_backend_tmpa_box
1244 }
1245 </dviptfm | xdvipdfm>
1246 <*dvisvgm>
1247 % unclear what it should do!!
1248 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4 {}
1249 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1 {}

```

```

1250 \cs_new:Npn \__pdf_backend_xform_ref:n {}
1251 </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

```

1252 <*dvips>
1253 \tl_new:N \l__pdf_backend_xform_tmpwd_tl
1254 \tl_new:N \l__pdf_backend_xform_tmpdp_tl
1255 \tl_new:N \l__pdf_backend_xform_tmph_tl

1256 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4 % #1 name, #2 attribute, #4
1257 {
1258   \int_gincr:N \g__pdf_backend_object_int
1259   \int_const:cn
1260   { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1261   { \g__pdf_backend_object_int }
1262
1263   \hbox_set:Nn \l__pdf_backend_tmpa_box
1264   {
1265     \bool_set_true:N \l__pdf_backend_xform_bool
1266     \prop_gclear:c {\__kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Properties }}
1267     #4
1268   }
1269   %store the dimensions
1270   \tl_const:ce
1271   { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1272   { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
1273   \tl_const:ce
1274   { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1275   { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
1276   \tl_const:ce
1277   { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1278   { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
1279   %store content dimensions in DPI units (Dots) (code from issue 25)
1280   \tl_set:Ne \l__pdf_backend_xform_tmpwd_tl
1281   {
1282     \dim_to_decimal_in_sp:n{ \box_wd:N \l__pdf_backend_tmpa_box }~
1283     65536~div~72.27~div~DVImag~mul~Resolution~mul~
1284   }
1285   \tl_set:Ne \l__pdf_backend_xform_tmph_tl
1286   {
1287     \dim_to_decimal_in_sp:n{ \box_ht:N \l__pdf_backend_tmpa_box }~
1288     65536~div~72.27~div~DVImag~mul~VResolution~mul~
1289   }
1290   \tl_set:Ne \l__pdf_backend_xform_tmpdp_tl
1291   {
1292     \dim_to_decimal_in_sp:n{ \box_dp:N \l__pdf_backend_tmpa_box }~
1293     65536~div~72.27~div~DVImag~mul~VResolution~mul~
1294   }
1295   % mirror the box
1296   %\box_scale:Nnn \l__pdf_backend_tmpa_box {1} {-1}
1297   \hbox_set:Nn \l__pdf_backend_tmpb_box
1298   {

```

```

1299     \__kernel_backend_postscript:e
1300     {
1301         gsave~currentpoint~
1302         initclip~ % restore default clipping path (page device/whole page)
1303         clippath~pathbbox~newpath~pop~pop~
1304         \tl_use:N\l__pdf_backend_xform_tmpdp_tl~add~translate~
1305         mark~
1306         /_objdef~{ pdf.obj \int_use:N\g__pdf_backend_object_int } \c_space_tl~
1307         /BBox[
1308             0~
1309             \tl_use:N\l__pdf_backend_xform_tmpt_tl~
1310             \tl_use:N\l__pdf_backend_xform_tmpwd_tl~
1311             \tl_use:N\l__pdf_backend_xform_tmpdp_tl~
1312             neg
1313         ]
1314         \str_if_eq:eeF{#1}{}
1315         {
1316             product~(Distiller)~search~{pop~pop~pop~#2}{pop}ifelse~
1317         }
1318         /BP~pdfmark-1~-1~scale~neg~exch~neg~exch~translate
1319     }
1320     \box_use_drop:N\l__pdf_backend_tmpa_box
1321     \__kernel_backend_postscript:n
1322     {
1323         mark ~ /EP~pdfmark ~ grestore
1324     }
1325     \str_if_eq:eeF{#1}{}
1326     {
1327         \__kernel_backend_postscript:e
1328         {
1329             product~(Ghostscript)~search~
1330             {
1331                 pop~pop~pop~
1332                 mark~
1333                 { pdf.obj \int_use:c{c__pdf_backend_xform_ \tl_to_str:n {#1} _int} }
1334                 ~<<#2>>~/PUT~pdfmark
1335             }{pop}ifelse
1336         }
1337     }
1338 }
1339 \box_set_dp:Nn \l__pdf_backend_tmpb_box { \c_zero_dim }
1340 \box_set_ht:Nn \l__pdf_backend_tmpb_box { \c_zero_dim }
1341 \box_set_wd:Nn \l__pdf_backend_tmpb_box { \c_zero_dim }
1342 \hook_gput_code:nnn {begindocument/end}{pdfxform}
1343 {
1344     \mode_leave_vertical:
1345     \box_use:N\l__pdf_backend_tmpb_box
1346 }
1347 }
1348
1349
1350 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
1351 {
1352     \hbox_set:Nn \l__pdf_backend_tmpa_box

```

```

1353     {
1354     \__kernel_backend_postscript:e
1355     {
1356         gsave~currentpoint~translate~1~-1~scale~
1357         mark~{ pdf.obj \int_use:c{c__pdf_backend_xform_ \tl_to_str:n {#1} _int }}~
1358         /SP~pdfmark ~ grestore
1359     }
1360 }
1361 \box_set_wd:Nn \l__pdf_backend_tmpa_box { \pdfxform_wd:n { #1 } }
1362 \box_set_ht:Nn \l__pdf_backend_tmpa_box { \pdfxform_ht:n { #1 } }
1363 \box_set_dp:Nn \l__pdf_backend_tmpa_box { \pdfxform_dp:n { #1 } }
1364 \box_use_drop:N \l__pdf_backend_tmpa_box
1365 }
1366 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1367 {
1368     { pdf.obj \int_use:c{c__pdf_backend_xform_ \tl_to_str:n {#1} _int} }
1369 }
1370
1371 </dvips>
1372 <*drivers>
1373 %% all
1374 \prg_new_conditional:Npnn \__pdf_backend_xform_if_exist:n #1 { p , T , F , TF }
1375 {
1376     \int_if_exist:cTF { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1377     { \prg_return_true: }
1378     { \prg_return_false:}
1379 }
1380 \prg_new_eq_conditional:NNn \pdfxform_if_exist:n\__pdf_backend_xform_if_exist:n
1381 { TF , T , F , p }
1382 </drivers>

```

(End of definition for `__pdf_backend_xform_new:n`, `__pdf_backend_xform_use:n`, and `__pdf_backend_xform_ref:n`.)

1.11 Structure Destinations

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

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

Currently structure destinations and `GoTo` links making use of it could natively only be created with the `dvipdfmx` backend. With `pdftex` and `lualatex` it was only possible to create a restricted type which used only the “Fit” mode. Starting with `TEXlive 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
1383 <*drivers>
1384 \tl_new:N \l_pdf_current_structure_destination_tl
1385 </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:nmnn -> \__pdf_backend_structure_destination:nmnn
\__pdf_backend_link_begin_goto:nmw -> \__pdf_backend_link_begin_structure_goto:nmw
\__pdf_backend_destination:nn -> \__pdf_backend_indexed_structure_destination:nn
\__pdf_backend_destination:nmnn -> \__pdf_backend_indexed_structure_destination:nmnn
\__pdf_backend_link_begin_goto:nmw -> \__pdf_backend_indexed_link_begin_structure_goto:nmw
```

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:
1386 <*drivers>
1387 \cs_new_protected:Npn \pdf_activate_structure_destination:
1388 {
1389   \cs_gset_eq:NN \__pdf_backend_destination:nn \__pdf_backend_structure_destination:nn
1390   \cs_gset_eq:NN \__pdf_backend_destination:nmnn \__pdf_backend_structure_destination:nmnn
1391   \cs_gset_eq:NN \__pdf_backend_link_begin_goto:nmw \__pdf_backend_link_begin_structure_goto:nmw
1392 }
1393 \cs_new_protected:Npn \pdf_activate_indexed_structure_destination:
1394 {
1395   \cs_gset_eq:NN \__pdf_backend_destination:nn \__pdf_backend_indexed_structure_destination:nn
1396   \cs_gset_eq:NN \__pdf_backend_destination:nmnn \__pdf_backend_indexed_structure_destination:nmnn
1397   \cs_gset_eq:NN \__pdf_backend_link_begin_goto:nmw \__pdf_backend_indexed_link_begin_structure_goto:nmw
1398 }
1399 </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.

```

1400 <*drivers>
1401 \cs_set_eq:NN \_pdf_backend_structure_destination:nn \_pdf_backend_destination:nn
1402 \cs_set_eq:NN \_pdf_backend_structure_destination:nmnn \_pdf_backend_destination:nmnn
1403 \cs_set_eq:NN \_pdf_backend_link_begin_structure_goto:nw \_pdf_backend_link_begin_goto:nw
1404 \cs_set_eq:NN \_pdf_backend_indexed_structure_destination:nn \_pdf_backend_destination:nn
1405 \cs_set_eq:NN \_pdf_backend_indexed_structure_destination:nmnn \_pdf_backend_destination:nmnn
1406 </drivers>

```

```

\_pdf_backend_structure_destination:nn
\_pdf_backend_structure_destination:nmnn
\_pdf_backend_link_begin_structure_goto:nw

```

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

```

1407 <*xdvipdfmx | dvipdfmx>
1408 \cs_set_protected:Npn \_pdf_backend_structure_destination:nn #1#2
1409 {
1410   \_pdf_backend:e
1411   {
1412     dest ~ ( \exp_not:n {#1} )
1413     [
1414       @thispage
1415       \str_case:nnF {#2}
1416       {
1417         { xyz } { /XYZ ~ @xpos ~ @ypos ~ null }
1418         { fit } { /Fit }
1419         { fitb } { /FitB }
1420         { fitbh } { /FitBH }
1421         { fitbv } { /FitBV ~ @xpos }
1422         { fith } { /FitH ~ @ypos }
1423         { fitv } { /FitV ~ @xpos }
1424         { fitr } { /Fit }
1425       }
1426       { /XYZ ~ @xpos ~ @ypos ~ \fp_eval:n { (#2) / 100 } }
1427     ]
1428   }

```

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.

```

1429 \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1430 {
1431   \_pdf_backend:e
1432   {
1433     obj ~ @pdf.SDest.\exp_not:n{#1}
1434     [
1435       \exp_args:Ne \pdf_object_ref:n { \l_pdf_current_structure_destination_tl }
1436       \str_case:nnF {#2}
1437       {
1438         { xyz } { /XYZ ~ @xpos ~ @ypos ~ null }
1439         { fit } { /Fit }
1440         { fitb } { /FitB }
1441         { fitbh } { /FitBH }
1442         { fitbv } { /FitBV ~ @xpos }
1443         { fith } { /FitH ~ @ypos }
1444         { fitv } { /FitV ~ @xpos }
1445         { fitr } { /Fit }

```



```

1446     }
1447     { /XYZ ~ @xpos ~ @ypos ~ \fp_eval:n { (#2) / 100 } }
1448   ]
1449 }
1450 }
1451 }

```

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

```

1452 \cs_new_protected:Npn \__pdf_backend_structure_destination_aux:nmmn #1#2#3#4
1453 {
1454   \vbox_to_zero:n
1455   {
1456     \__kernel_kern:n {#4}
1457     \hbox:n
1458     {
1459       \__pdf_backend:n { obj ~ @pdf_ #2 _llx ~ @xpos }
1460       \__pdf_backend:n { obj ~ @pdf_ #2 _lly ~ @ypos }
1461     }
1462     \tex_vss:D
1463   }
1464   \__kernel_kern:n {#1}
1465   \vbox_to_zero:n
1466   {
1467     \__kernel_kern:n { -#3 }
1468     \hbox:n
1469     {
1470       \__pdf_backend:n
1471       {
1472         dest ~ (#2)
1473         [
1474           @thispage
1475           /FitR ~
1476           @pdf_ #2 _llx ~ @pdf_ #2 _lly ~
1477           @xpos ~ @ypos
1478         ]
1479       }
1480     }
1481   }
1482   \__pdf_backend:e
1483   {
1484     obj ~ @pdf.SDest.\exp_not:n{#2}
1485     [
1486       \exp_args:Ne \pdf_object_ref:n { \l_pdf_current_structure_destination_
1487       /FitR ~
1488       @pdf_ #2 _llx ~ @pdf_ #2 _lly ~
1489       @xpos ~ @ypos
1490     ]
1491   }
1492 }
1493 }
1494 \tex_vss:D
1495 }

```

Here we add the structure destination to the same box

```

1480     \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1481     {
1482       \__pdf_backend:e
1483       {
1484         obj ~ @pdf.SDest.\exp_not:n{#2}
1485         [
1486           \exp_args:Ne \pdf_object_ref:n { \l_pdf_current_structure_destination_
1487           /FitR ~
1488           @pdf_ #2 _llx ~ @pdf_ #2 _lly ~
1489           @xpos ~ @ypos
1490         ]
1491       }
1492     }
1493 }
1494 \tex_vss:D
1495 }

```

```

1496   \_kernel_kern:n { -#1 }
1497 }

```

And now we redefine the destination command:

```

1498 \cs_set_protected:Npn \_pdf_backend_structure_destination:nmmn #1#2#3#4
1499 {
1500   \exp_args:Ne \_pdf_backend_structure_destination_aux:nmmn
1501     { \dim_eval:n {#2} } {#1} {#3} {#4}
1502 }

```

At last the goto link.

```

1503 \cs_set_protected:Npn \_pdf_backend_link_begin_structure_goto:nmw #1#2
1504 {
1505   \_pdf_backend_link_begin:n { #1 /Subtype /Link /A << /S /GoTo /D ( #2 ) /SD~@pdf.SDest.
1506 }
1507 </xdvipdfmx | dvipdfmx>

```

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

```

1508 <*pdftex>
1509 \bool_lazy_and:nnT
1510 { \int_compare_p:nNn {\tex_pdftexversion:D } > {139} }
1511 { \int_compare_p:nNn {\tex_pdftexrevision:D } > {23} }
1512 {
1513   \cs_set_protected:Npn \_pdf_backend_structure_destination:nn #1#2
1514   {
1515     \tex_pdfdest:D
1516       name {#1}
1517       \str_case:nnF {#2}
1518       {
1519         { xyz } { xyz }
1520         { fit } { fit }
1521         { fitb } { fitb }
1522         { fitbh } { fitbh }
1523         { fitbv } { fitbv }
1524         { fith } { fith }
1525         { fitv } { fitv }
1526         { fitr } { fitr }
1527       }
1528     { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1529     \scan_stop:
1530     \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1531     {
1532       \tex_pdfdest:D
1533       struct~
1534       \int_use:c
1535         { c_pdf_object_ \exp_args:Ne \tl_to_str:n {\l_pdf_current_structure_destin
1536       name {#1}
1537       \str_case:nnF {#2}
1538       {
1539         { xyz } { xyz }
1540         { fit } { fit }
1541         { fitb } { fitb }
1542         { fitbh } { fitbh }
1543         { fitbv } { fitbv }
1544         { fith } { fith }

```

```

1545         { fitv } { fitv }
1546         { fitr } { fitr }
1547     }
1548     { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1549 \scan_stop:
1550 }
1551 }
1552 \cs_set_protected:Npn \__pdf_backend_structure_destination:nnnn #1#2#3#4
1553 {
1554     \tex_pdfdest:D
1555     name {#1}
1556     fitr ~
1557     width \dim_eval:n {#2} ~
1558     height \dim_eval:n {#3} ~
1559     depth \dim_eval:n {#4} \scan_stop:
1560 \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1561 {
1562     \tex_pdfdest:D
1563     struct~
1564     \int_use:c
1565     { c__pdf_object_ \exp_args:Ne \tl_to_str:n { \l_pdf_current_structure_destination_tl }
1566     name {#1}
1567     fitr ~
1568     width \dim_eval:n {#2} ~
1569     height \dim_eval:n {#3} ~
1570     depth \dim_eval:n {#4} \scan_stop:
1571 }
1572 }
1573 \cs_set_protected:Npn \__pdf_backend_link_begin_structure_goto:nnw #1#2
1574 {
1575     \__pdf_backend_link_begin:nnnw {#1} { goto~struct~name~{#2}~name } {#2}
1576 }
1577 }
1578 </pdfTeX>

```

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

```

1579 <*luatex>
1580 \int_compare:nNnT {\directlua{tex.print(status.list()["development_id"])} } > {7468}
1581 {
1582     \cs_set_protected:Npn \__pdf_backend_structure_destination:nn #1#2
1583     {
1584         \tex_pdfextension:D dest
1585         name {#1}
1586         \str_case:nnF {#2}
1587         {
1588             { xyz } { xyz }
1589             { fit } { fit }
1590             { fitb } { fitb }
1591             { fitbh } { fitbh }
1592             { fitbv } { fitbv }
1593             { fith } { fith }
1594             { fitv } { fitv }
1595             { fitr } { fitr }
1596         }
1597         { xyz ~ zoom \fp_eval:n { #2 * 10 } }

```

```

1598     \scan_stop:
1599     \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1600     {
1601         \tex_pdfextension:D dest
1602         struct~
1603         \int_use:c
1604         { c_pdf_object_ \exp_args:Ne \tl_to_str:n {\l_pdf_current_structure_destin
1605         name {#1}
1606         \str_case:nnF {#2}
1607         {
1608             { xyz } { xyz }
1609             { fit } { fit }
1610             { fitb } { fitb }
1611             { fitbh } { fitbh }
1612             { fitbv } { fitbv }
1613             { fith } { fith }
1614             { fitv } { fitv }
1615             { fitr } { fitr }
1616         }
1617         { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1618     \scan_stop:
1619     }
1620 }
1621 \cs_set_protected:Npn \__pdf_backend_structure_destination:nnnn #1#2#3#4
1622 {
1623     \tex_pdfextension:D dest
1624     name {#1}
1625     fitr ~
1626     width \dim_eval:n {#2} ~
1627     height \dim_eval:n {#3} ~
1628     depth \dim_eval:n {#4} \scan_stop:
1629     \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1630     {
1631         \tex_pdfextension:D dest
1632         struct~
1633         \int_use:c
1634         { c_pdf_object_ \exp_args:Ne \tl_to_str:n {\l_pdf_current_structure_destinat
1635         name {#1}
1636         fitr ~
1637         width \dim_eval:n {#2} ~
1638         height \dim_eval:n {#3} ~
1639         depth \dim_eval:n {#4} \scan_stop:
1640     }
1641 }
1642 \cs_set_protected:Npn \__pdf_backend_link_begin_structure_goto:nnw #1#2
1643 {
1644     \__pdf_backend_link_begin:nnnw {#1} { goto~struct~name~{#2}~name } {#2}
1645 }
1646 }
1647 \end{luatex}

```

(End of definition for __pdf_backend_structure_destination:nn, __pdf_backend_structure_destination:nnnn, and __pdf_backend_link_begin_structure_goto:nnw.)

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:

```

1648 <*xdvipdfmx | dvipdfmx>
1649 \cs_set_protected:Npn \__pdf_backend_indexed_structure_destination:nn #1#2
1650 {
1651   \__pdf_backend:e
1652   {
1653     dest ~ ( \exp_not:n {#1} )
1654     [
1655       @thispage
1656       \str_case:nnF {#2}
1657       {
1658         { xyz } { /XYZ ~ @xpos ~ @ypos ~ null }
1659         { fit } { /Fit }
1660         { fitb } { /FitB }
1661         { fitbh } { /FitBH }
1662         { fitbv } { /FitBV ~ @xpos }
1663         { fith } { /FitH ~ @ypos }
1664         { fitv } { /FitV ~ @xpos }
1665         { fitr } { /Fit }
1666       }
1667       { /XYZ ~ @xpos ~ @ypos ~ \fp_eval:n { (#2) / 100 } }
1668     ]
1669   }

```

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.

```

1670   \__pdf_backend:e
1671   {
1672     obj ~ @pdf.SDest.\exp_not:n{#1}
1673     [
1674       \exp_after:wN \pdf_object_ref_indexed:nn \l_pdf_current_structure_destination_t
1675       \str_case:nnF {#2}
1676       {
1677         { xyz } { /XYZ ~ @xpos ~ @ypos ~ null }
1678         { fit } { /Fit }
1679         { fitb } { /FitB }
1680         { fitbh } { /FitBH }
1681         { fitbv } { /FitBV ~ @xpos }
1682         { fith } { /FitH ~ @ypos }
1683         { fitv } { /FitV ~ @xpos }
1684         { fitr } { /Fit }
1685       }
1686       { /XYZ ~ @xpos ~ @ypos ~ \fp_eval:n { (#2) / 100 } }
1687     ]
1688   }
1689 }

```

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

```

1690 \cs_new_protected:Npn \__pdf_backend_indexed_structure_destination_aux:nnnn #1#2#3#4
1691 {
1692   \vbox_to_zero:n

```

```

1693     {
1694       \__kernel_kern:n {#4}
1695       \hbox:n
1696         {
1697           \__pdf_backend:n { obj ~ @pdf_ #2 _llx ~ @xpos }
1698           \__pdf_backend:n { obj ~ @pdf_ #2 _lly ~ @ypos }
1699         }
1700       \tex_vss:D
1701     }
1702 \__kernel_kern:n {#1}
1703 \vbox_to_zero:n
1704 {
1705   \__kernel_kern:n { -#3 }
1706   \hbox:n
1707     {
1708       \__pdf_backend:n
1709         {
1710           dest ~ (#2)
1711           [
1712             @thispage
1713             /FitR ~
1714             @pdf_ #2 _llx ~ @pdf_ #2 _lly ~
1715             @xpos ~ @ypos
1716           ]
1717         }

```

Here we add the structure destination to the same box

```

1718       \__pdf_backend:e
1719       {
1720         obj ~ @pdf.SDest.\exp_not:n{#2}
1721         [
1722           \exp_after:wN \pdf_object_ref_indexed:nn \l_pdf_current_structure_destin
1723           /FitR ~
1724           @pdf_ #2 _llx ~ @pdf_ #2 _lly ~
1725           @xpos ~ @ypos
1726         ]
1727       }
1728     }
1729     \tex_vss:D
1730   }
1731 \__kernel_kern:n { -#1 }
1732 }

```

And now we redefine the destination command:

```

1733 \cs_set_protected:Npn \__pdf_backend_indexed_structure_destination:nnnn #1#2#3#4
1734 {
1735   \exp_args:Ne \__pdf_backend_indexed_structure_destination_aux:nnnn
1736   { \dim_eval:n {#2} } {#1} {#3} {#4}
1737 }
1738 </xdvipdfmx | dvipdfmx>

```

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

```

1739 <*pdftex>
1740 \bool_lazy_and:nnT
1741 { \int_compare_p:nNn {\tex_pdftexversion:D } > {139} }

```

```

1742 { \int_compare_p:nNn {\tex_pdfptexrevision:D } > {23} }
1743 {
1744   \cs_set_protected:Npn \__pdf_backend_indexed_structure_destination:nn #1#2
1745   {
1746     \tex_pdfdest:D
1747     name {#1}
1748     \str_case:nnF {#2}
1749     {
1750       { xyz } { xyz }
1751       { fit } { fit }
1752       { fitb } { fitb }
1753       { fitbh } { fitbh }
1754       { fitbv } { fitbv }
1755       { fith } { fith }
1756       { fitv } { fitv }
1757       { fitr } { fitr }
1758     }
1759     { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1760     \scan_stop:
1761     \tex_pdfdest:D
1762     struct~
1763     \exp_after:wN \__kernel_pdf_object_id_indexed:nn \l_pdf_current_structure_dest
1764     name {#1}
1765     \str_case:nnF {#2}
1766     {
1767       { xyz } { xyz }
1768       { fit } { fit }
1769       { fitb } { fitb }
1770       { fitbh } { fitbh }
1771       { fitbv } { fitbv }
1772       { fith } { fith }
1773       { fitv } { fitv }
1774       { fitr } { fitr }
1775     }
1776     { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1777     \scan_stop:
1778   }
1779   \cs_set_protected:Npn \__pdf_backend_indexed_structure_destination:nnnn #1#2#3#4
1780   {
1781     \tex_pdfdest:D
1782     name {#1}
1783     fitr ~
1784     width \dim_eval:n {#2} ~
1785     height \dim_eval:n {#3} ~
1786     depth \dim_eval:n {#4} \scan_stop:
1787     \tex_pdfdest:D
1788     struct~
1789     \exp_after:wN \__kernel_pdf_object_id_indexed:nn \l_pdf_current_structure_destinati
1790     name {#1}
1791     fitr ~
1792     width \dim_eval:n {#2} ~
1793     height \dim_eval:n {#3} ~
1794     depth \dim_eval:n {#4} \scan_stop:
1795   }

```

```

1796 }
1797 </pdfTeX>
luatex is quite similar to pdfTeX. Mostly the test for the version is different
1798 <*luatex>
1799 \int_compare:nNtT {\directlua{tex.print(status.list()["development_id"])} } > {7468}
1800 {
1801   \cs_set_protected:Npn \__pdf_backend_indexed_structure_destination:nn #1#2
1802   {
1803     \tex_pdfextension:D dest
1804     name {#1}
1805     \str_case:nnF {#2}
1806     {
1807       { xyz } { xyz }
1808       { fit } { fit }
1809       { fitb } { fitb }
1810       { fitbh } { fitbh }
1811       { fitbv } { fitbv }
1812       { fith } { fith }
1813       { fitv } { fitv }
1814       { fitr } { fitr }
1815     }
1816     { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1817     \scan_stop:
1818     \tex_pdfextension:D dest
1819     struct~
1820     \exp_after:wN \__kernel_pdf_object_id_indexed:nn \l_pdf_current_structure_desti
1821     name {#1}
1822     \str_case:nnF {#2}
1823     {
1824       { xyz } { xyz }
1825       { fit } { fit }
1826       { fitb } { fitb }
1827       { fitbh } { fitbh }
1828       { fitbv } { fitbv }
1829       { fith } { fith }
1830       { fitv } { fitv }
1831       { fitr } { fitr }
1832     }
1833     { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1834     \scan_stop:
1835   }
1836   \cs_set_protected:Npn \__pdf_backend_indexed_structure_destination:nnnn #1#2#3#4
1837   {
1838     \tex_pdfextension:D dest
1839     name {#1}
1840     fitr ~
1841     width \dim_eval:n {#2} ~
1842     height \dim_eval:n {#3} ~
1843     depth \dim_eval:n {#4} \scan_stop:
1844     \tex_pdfextension:D dest
1845     struct~
1846     \exp_after:wN \__kernel_pdf_object_id_indexed:nn \l_pdf_current_structure_destinati
1847     name {#1}
1848     fitr ~

```



```

1849         width \dim_eval:n {#2} ~
1850         height \dim_eval:n {#3} ~
1851         depth \dim_eval:n {#4} \scan_stop:
1852     }
1853     \cs_set_protected:Npn \__pdf_backend_link_begin_structure_goto:nnw #1#2
1854     {
1855         \__pdf_backend_link_begin:nnnw {#1} { goto~struct~name~{#2}~name } {#2}
1856     }
1857 }
1858 </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

```

1859 <*drivers>
1860 \cs_new_protected:Npn \__pdf_backend_set_regression_data:
1861 {
1862     \sys_gset_rand_seed:n{1000}
1863     \pdfmanagement_add:nnn{Info}{Creator}{(TeX)}
1864 </drivers>
1865 <*dvips>
1866     \AddToHook{begindocument}{\pdfmanagement_add:nnn{Info}{Producer}{(pdfTeX+dvips)}}
1867     \__kernel_backend_literal:e{!~<</DocumentUUID~(DocumentUUID)>>~setpagedevice}
1868     \__kernel_backend_literal:e{!~<</InstanceUUID~(InstanceUUID)>>~setpagedevice}
1869     \str_if_exist:NTF\c_sys_timestamp_str
1870     {
1871         \pdfmanagement_add:nne{Info}{CreationDate}{(\c_sys_timestamp_str)}
1872         \pdfmanagement_add:nne{Info}{ModDate}{(\c_sys_timestamp_str)}
1873     }
1874     {
1875         \pdfmanagement_add:nnn{Info}{CreationDate}{(D:20010101205959-00'00')}
1876         \pdfmanagement_add:nnn{Info}{ModDate}{(D:20010101205959-00'00')}
1877     }
1878 </dvips>
1879 <*dviPDFmx>
1880     \pdfmanagement_add:nnn{Info}{Producer}{(dviPDFmx)}
1881     \__kernel_backend_literal:e
1882     {pdf:trailerid [~
1883     <00112233445566778899aabbccddeeff>~
1884     <00112233445566778899aabbccddeeff>~
1885     ]}
1886 </dviPDFmx>
1887 <*xdviPDFmx>
1888     \pdfmanagement_add:nnn{Info}{Producer}{(xetex)}
1889     \__kernel_backend_literal:e
1890     {pdf:trailerid [~
1891     <00112233445566778899aabbccddeeff>~
1892     <00112233445566778899aabbccddeeff>~
1893     ]}

```

```

1894 </xdvipdfmx>
1895 <*pdftex>
1896   \pdfmanagement_add:nnn{Info}{Producer}{(pdfTeX)}
1897   \tex_pdfsuppressptexinfo:D 7 \scan_stop:
1898   \pdftrailerid{2350CAD05F8A7AF0AA4058486855344F}
1899 </pdftex>
1900 <*luatex>
1901   \pdfmanagement_add:nnn{Info}{Producer}{(LuaTeX)}
1902   \tex_pdfvariable:D suppressoptionalinfo 7\relax
1903   \tex_pdfvariable:D trailerid
1904   {[~
1905     <2350CAD05F8A7AF0AA4058486855344F>~
1906     <2350CAD05F8A7AF0AA4058486855344F>~
1907   ]}
1908 </luatex>
1909 <*drivers>
1910   \str_if_exist:NF\c_sys_timestamp_str
1911   {
1912     \pdfmanagement_add:nnn{Info}{CreationDate}{(D:20010101205959-00'00')}
1913     \pdfmanagement_add:nnn{Info}{ModDate}{(D:20010101205959-00'00')}
1914     \AddToDocumentProperties[document]{creationdate}{D:20010101205959-00'00'}
1915     \AddToDocumentProperties[document]{moddate}{D:20010101205959-00'00'}
1916     \AddToDocumentProperties[hyperref]{pdfmetadate}{D:20010101205959-00'00'}
1917     \AddToDocumentProperties[hyperref]{pdfdate}{D:20010101205959-00'00'}
1918   }
1919   \AddToDocumentProperties[hyperref]{pdfinstanceid}{uuid:0a57c455-157a-4141-8c19-6237d832f}
1920   \AddToDocumentProperties[hyperref]{pdfproducer}{\c_sys_engine_exec_str-NN.NN.NN}
1921 }
1922 </drivers>

```

1.13 Uncompressed metadata object stream

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

```

1923 <*luatex>
1924 \cs_new_protected:Npn \__pdf_backend_metadata_stream:n #1
1925 {
1926   \tex_immediate:D \tex_pdfextension:D obj ~uncompressed~
1927   \__pdf_backend_object_write:nn {stream} {{/Type~/Metadata~/Subtype~/XML}{#1}}
1928 }
1929 </luatex>
1930 <*pdftex>
1931 \cs_new_protected:Npn \__pdf_backend_metadata_stream:n #1
1932 {
1933   \group_begin:
1934   \tex_pdfcompresslevel:D 0 \scan_stop:
1935   \tex_immediate:D \tex_pdfobj:D

```

```

1936     \__pdf_backend_object_write:n {stream} {{/Type~/Metadata~/Subtype~/XML}{#1}}
1937     \group_end:
1938   }
1939 </pdftex>
1940 <*\xdvipdfmx | dvipdfmx | dvips | dvisvgm>
1941 \cs_new_protected:Npn \__pdf_backend_metadata_stream:n #1
1942 {
1943   \pdf_object_unnamed_write:n {stream}{{/Type~/Metadata~/Subtype~/XML}{#1}}
1944 }
1945 </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.

```

1946 <*\xdvipdfmx | dvipdfmx | dvips | dvisvgm>
1947 \cs_new_protected:Npn \__pdf_backend_omit_charset:n #1 {} % #1 number
1948 </xdvipdfmx | dvipdfmx | dvips | dvisvgm>
1949 <*\pdftex>
1950 \cs_new_protected:Npn \__pdf_backend_omit_charset:n #1 % #1 number
1951 {
1952   \tex_pdfomitcharset:D = #1 \scan_stop:
1953 }
1954 </pdftex>
1955 <*\luatex>
1956 \cs_new_protected:Npn \__pdf_backend_omit_charset:n #1 % #1 number
1957 {
1958   \tex_pdfvariable:D omitcharset = #1 \scan_stop:
1959 }
1960 </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.

```

1961 <*\xdvipdfmx | dvipdfmx | dvips | dvisvgm>
1962 \cs_new_protected:Npn \__pdf_backend_omit_info:n #1 {} % #1 number
1963 </xdvipdfmx | dvipdfmx | dvips | dvisvgm>
1964 <*\pdftex>
1965 \bool_lazy_and:nnTF
1966 { \int_compare_p:nNn {\tex_pdfversion:D } > {139} }
1967 { \int_compare_p:nNn {\tex_pdfrevision:D } > {24} }
1968 {
1969   \cs_new_protected:Npn \__pdf_backend_omit_info:n #1 % #1 number
1970   {
1971     \pdfomitinfodict = #1 \scan_stop:
1972   }
1973 }
1974 {
1975   \cs_new_protected:Npn \__pdf_backend_omit_info:n #1 {} % #1 number
1976 }

```

```

1977 }
1978 </pdfTeX>
1979 <*luatex>
1980 \int_compare:nNnTF {\directlua{tex.print(status.list()["development_id"])} } > {7560}
1981 {
1982   \cs_new_protected:Npn \__pdf_backend_omit_info:n #1 %#1 number
1983   {
1984     \tex_pdfvariable:D omitinfodict = #1 \scan_stop:
1985   }
1986 }
1987 {
1988   \cs_new_protected:Npn \__pdf_backend_omit_info:n #1 {} %#1 number
1989 }
1990 </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.

```

1991 <*xdvipdfmx | dvipdfmx | dvips | dvisvgm | pdfTeX>
1992 \cs_new_protected:Npn \__pdf_backend_omit_cidset:n #1 {} %#1 number
1993 </xdvipdfmx | dvipdfmx | dvips | dvisvgm | pdfTeX>
1994 <*luatex>
1995 \cs_new_protected:Npn \__pdf_backend_omit_cidset:n #1 %#1 number
1996 {
1997   \tex_pdfvariable:D omitcidset = #1 \scan_stop:
1998 }
1999 </luatex>

```

(End of definition for __pdf_backend_omit_cidset:n.)

1.15 lua code for lualatex

```

2000 <*lua>
2001 ltx= ltx or {}
2002 ltx.__pdf      = ltx.__pdf or {}
2003 ltx.__pdf.Page = ltx.__pdf.Page or {}
2004 ltx.__pdf.Page.dflt = ltx.__pdf.Page.dflt or {}
2005 ltx.__pdf.Page.Resources = ltx.__pdf.Page.Resources or {}
2006 ltx.__pdf.Page.Resources.Properties = ltx.__pdf.Page.Resources.Properties or {}
2007 ltx.__pdf.Page.Resources.List={"ExtGState","ColorSpace","Pattern","Shading"}
2008 ltx.__pdf.object = ltx.__pdf.object or {}
2009
2010 ltx.pdf= ltx.pdf or {} -- for "public" functions
2011
2012 local __pdf = ltx.__pdf
2013 local pdf = pdf
2014
2015 local function __pdf_backend_Page_gput (name,value)
2016   __pdf.Page.dflt[name]=value
2017 end
2018
2019 local function __pdf_backend_Page_gremove (name)
2020   __pdf.Page.dflt[name]=nil

```

```

2021 end
2022
2023 local function __pdf_backend_Page_gclear ()
2024 __pdf.Page.dflt={}
2025 end
2026
2027 local function __pdf_backend_ThisPage_gput (page,name,value)
2028 __pdf.Page[page] = __pdf.Page[page] or {}
2029 __pdf.Page[page][name]=value
2030 end
2031
2032 local function __pdf_backend_ThisPage_gpush (page)
2033 local token=""
2034 local t = {}
2035 local tkeys= {}
2036 for name,value in pairs(__pdf.Page.dflt) do
2037 t[name]=value
2038 end
2039 if __pdf.Page[page] then
2040 for name,value in pairs(__pdf.Page[page]) do
2041 t[name] = value
2042 end
2043 end
2044 -- sort the table to get reliable test files.
2045 for name,value in pairs(t) do
2046 table.insert(tkeys,name)
2047 end
2048 table.sort(tkeys)
2049 for _,name in ipairs(tkeys) do
2050 token = token .. "/"..name.." "..t[name]
2051 end
2052 return token
2053 end
2054
2055 function ltx.__pdf.backend_ThisPage_gput (page,name,value) -- tex.count["g_shipout_readonly"]
2056 __pdf_backend_ThisPage_gput (page,name,value)
2057 end
2058
2059 function ltx.__pdf.backend_ThisPage_gpush (page)
2060 pdf.setpageattributes(__pdf_backend_ThisPage_gpush (page))
2061 end
2062
2063 function ltx.__pdf.backend_Page_gput (name,value)
2064 __pdf_backend_Page_gput (name,value)
2065 end
2066
2067 function ltx.__pdf.backend_Page_gremove (name)
2068 __pdf_backend_Page_gremove (name)
2069 end
2070
2071 function ltx.__pdf.backend_Page_gclear ()
2072 __pdf_backend_Page_gclear ()
2073 end
2074

```

```

2075
2076 local Properties = ltx.__pdf.Page.Resources.Properties
2077 local ResourceList= ltx.__pdf.Page.Resources.List
2078 local function __pdf_backend_PageResources_gpush (page)
2079   local token=""
2080   if Properties[page] then
2081     -- we sort the table, so that the pdf test works
2082     local t = {}
2083     for name,value in pairs (Properties[page]) do
2084       table.insert (t,name)
2085     end
2086     table.sort (t)
2087     for _,name in ipairs(t) do
2088       token = token .. "/"..name.." ".. Properties[page][name]
2089     end
2090     token = "/Properties <<"..token..">>"
2091   end
2092   for i,name in ipairs(ResourceList) do
2093     if ltx.__pdf.Page.Resources[name] then
2094       token = token .. "/"..name.." "..ltx.pdf.object_ref("__pdf/Page/Resources/"..name)
2095     end
2096   end
2097   return token
2098 end
2099
2100 -- the function is public, as I probably need it in tagpdf too ...
2101 function ltx.pdf.Page_Resources_Properties_gput (page,name,value) -- tex.count["g_shipout_re
2102   Properties[page] = Properties[page] or {}
2103   Properties[page][name]=value
2104   pdf.setpageresources(__pdf_backend_PageResources_gpush (page))
2105 end
2106
2107 function ltx.pdf.Page_Resources_gpush(page)
2108   pdf.setpageresources(__pdf_backend_PageResources_gpush (page))
2109 end
2110
2111 function ltx.pdf.object_ref (objname)
2112   if ltx.__pdf.object[objname] then
2113     local ref= ltx.__pdf.object[objname]
2114     return ref
2115   else
2116     return "false"
2117   end
2118 end
2119 </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`
 . . 1914, 1915, 1916, 1917, 1919, 1920₄₆

<code>\AddToHook</code>	1866	578, 580, 593, 612, 631, 637, 643, 648, 658, 664, 687, 711, 735, 739, 744, 753, 760, 768, 798, 829, 833, 838, 846, 853, 947, 951, 960, 964, 971, 975, 1408, 1498, 1503, 1513, 1552, 1573, 1582, 1621, 1642, 1649, 1733, 1744, 1779, 1801, 1836, 1853
B		
bool commands:		
<code>\bool_if:NTF</code>	39, 533, 569, 650, 656, 692, 716, 751, 775, 805, 844	
<code>\bool_lazy_and:nnTF</code> ..	1509, 1740, 1965	
<code>\bool_new:N</code>	522	
<code>\bool_set_true:N</code> ..	989, 1071, 1164, 1265	
box commands:		
<code>\box_dp:N</code> ..	1002, 1083, 1175, 1278, 1292	
<code>\box_ht:N</code> ..	999, 1080, 1172, 1275, 1287	
<code>\box_new:N</code>	88, 89, 1161	
<code>\box_scale:Nnn</code>	1296	
<code>\box_set_dp:Nn</code> ..	1176, 1242, 1339, 1363	
<code>\box_set_ht:Nn</code> ..	1177, 1241, 1340, 1362	
<code>\box_set_wd:Nn</code> ..	1178, 1240, 1341, 1361	
<code>\box_use:N</code>	1345	
<code>\box_use_drop:N</code> ..	1189, 1243, 1320, 1364	
<code>\box_wd:N</code> ..	996, 1077, 1169, 1272, 1282	
C		
clist commands:		
<code>\clist_const:Nn</code>	420	
<code>\clist_map_function:NN</code>	882	
<code>\clist_map_inline:Nn</code> ..	429, 463, 479, 672	
cs commands:		
<code>\cs_generate_variant:Nn</code>	28, 31, 32, 35, 36, 79, 80, 417	
<code>\cs_gset_eq:NN</code>	651, 652, 1389, 1390, 1391, 1395, 1396, 1397	
<code>\cs_if_exist:NTF</code>	432, 945	
<code>\cs_new:Npn</code>	74, 100, 106, 248, 858, 1055, 1137, 1226, 1250, 1366	
<code>\cs_new_protected:Npn</code> ...	41, 45, 55, 68, 150, 159, 175, 181, 187, 194, 201, 210, 230, 253, 263, 277, 289, 306, 317, 324, 331, 340, 349, 356, 363, 370, 379, 388, 396, 399, 405, 410, 413, 444, 455, 461, 487, 491, 503, 506, 507, 511, 514, 515, 519, 535, 558, 582, 670, 765, 867, 891, 898, 905, 914, 918, 921, 924, 936, 941, 942, 981, 1048, 1063, 1128, 1151, 1231, 1248, 1249, 1256, 1350, 1387, 1393, 1452, 1690, 1860, 1924, 1931, 1941, 1947, 1950, 1956, 1962, 1969, 1975, 1982, 1988, 1992, 1995	
<code>\cs_new_protected:Npx</code>	169	
<code>\cs_set_eq:NN</code>	530, 531, 566, 567, 662, 749, 757, 842, 850, 1401, 1402, 1403, 1404, 1405	
<code>\cs_set_protected:Npn</code> ..	526, 542, 546, 550, 554, 564, 571, 574, 576,	
D		
dim commands:		
<code>\dim_eval:n</code>	1501, 1557, 1558, 1559, 1568, 1569, 1570, 1626, 1627, 1628, 1637, 1638, 1639, 1736, 1784, 1785, 1786, 1792, 1793, 1794, 1841, 1842, 1843, 1849, 1850, 1851	
<code>\dim_to_decimal_in_sp:n</code>	1282, 1287, 1292	
<code>\c_zero_dim</code>	1176, 1177, 1178, 1339, 1340, 1341	
<code>\directlua</code>	97, 1580, 1799, 1980	
E		
exp commands:		
<code>\exp_after:wN</code>	1674, 1722, 1763, 1789, 1820, 1846	
<code>\exp_args:Ne</code> ..	700, 724, 1429, 1435, 1480, 1486, 1500, 1530, 1535, 1560, 1565, 1599, 1604, 1629, 1634, 1735	
<code>\exp_args:NNe</code>	869	
<code>\exp_not:n</code>	617, 715, 802, 1412, 1433, 1484, 1653, 1672, 1720	
F		
fp commands:		
<code>\fp_eval:n</code>	1426, 1447, 1528, 1548, 1597, 1617, 1667, 1686, 1759, 1776, 1816, 1833	
G		
group commands:		
<code>\group_begin:</code>	1933	
<code>\group_end:</code>	1937	
H		
hbox commands:		
<code>\hbox:n</code>	1457, 1468, 1695, 1706	
<code>\hbox_gset:Nn</code>	1162	
<code>\hbox_set:Nn</code>	987, 1069, 1233, 1263, 1297, 1352	
hook commands:		
<code>\hook_gput_code:nnn</code> ..	142, 482, 1342	
<code>\hook_gput_next_code:nn</code>	1179	
<code>\hook_gset_rule:nmnn</code>	476, 477	

I

int commands:
`\int_compare:nNnTF` 958, 1010, 1091, 1580, 1799, 1980
`\int_compare_p:nNn` 1510, 1511, 1741, 1742, 1966, 1967
`\int_const:Nn` 1043, 1123, 1158, 1259
`\int_gincr:N` 213, 595, 614, 689, 713, 770, 774, 800, 804, 1157, 1258
`\int_if_exist:NTF` 1376
`\int_new:N` 92, 93, 94
`\int_use:N` 214, 217, 598, 606, 617, 625, 691, 696, 705, 715, 720, 729, 772, 779, 783, 786, 794, 802, 809, 813, 816, 824, 1051, 1057, 1130, 1138, 1228, 1306, 1333, 1357, 1368, 1534, 1564, 1603, 1633

K

kernel internal commands:
`__kernel_backend_literal:n` 31, 83, 596, 600, 615, 619, 633, 645, 666, 676, 1867, 1868, 1881, 1889
`__kernel_backend_literal_page:n` 28, 690, 714, 737, 746, 762, 771, 801, 831, 840, 855
`__kernel_backend_postscript:n` 35, 1299, 1321, 1327, 1354
`__kernel_backend_shipout_literal:n` 39, 41, 537, 660
`__kernel_backend_shipout_literal_page:n` 55, 55, 755, 848
`__kernel_backend_shipout_literal_pdf:n` 45, 45
`__kernel_kern:n` 1456, 1464, 1467, 1496, 1694, 1702, 1705, 1731
`__kernel_pdf_name_from_unicode_e:n` 100, 106
`__kernel_pdf_object_id_indexed:nn` 1763, 1789, 1820, 1846
`__kernel_pdfdict_name:n` 232, 233, 235, 466, 494, 674, 861, 872, 877, 990, 1011, 1022, 1027, 1032, 1037, 1072, 1092, 1102, 1107, 1112, 1117, 1266
`\g__kernel_pdfmanagement_end_run_code_tl` 115, 122, 129
`\g__kernel_pdfmanagement_thispage_shipout_code_tl` 138, 144

L

l^{at}lua commands:
`\latelua:` 207, 286, 337, 376

M

mode commands:
`\mode_leave_vertical:` 1181, 1344

P

pdf commands:
`\pdf_activate_indexed_structure_destination:` 1386, 1393
`\pdf_activate_structure_destination:` 1386, 1387
`\l_pdf_current_structure_destination_tl` 1383, 1429, 1435, 1480, 1486, 1530, 1535, 1560, 1565, 1599, 1604, 1629, 1634, 1674, 1722, 1763, 1789, 1820, 1846
`\pdf_object_if_exist:nTF` 1429, 1480, 1530, 1560, 1599, 1629
`\pdf_object_new:n` 431, 481
`\pdf_object_ref:n` 438, 499, 544, 607, 679, 697, 706, 780, 795, 863, 1024, 1029, 1034, 1039, 1104, 1109, 1114, 1119, 1195, 1202, 1209, 1217, 1435, 1486
`\pdf_object_ref_indexed:nn` 1674, 1722
`\pdf_object_ref_last:` 894, 901, 908
`\pdf_object_unnamed_write:nn` 639, 741, 835, 893, 900, 907, 1943
`\pdf_object_write` 496
`\pdf_object_write:nnn` 468, 485

pdf internal commands:
`__pdf_backend:n` 32, 177, 489, 497, 908, 973, 977, 1182, 1190, 1191, 1198, 1205, 1212, 1220, 1235, 1410, 1431, 1459, 1460, 1470, 1482, 1651, 1670, 1697, 1698, 1708, 1718
`__pdf_backend_bdc:nn` 13, 521, 526, 530, 531, 564, 566, 567, 648, 651, 652, 653, 749, 842
`__pdf_backend_bdc_contobj:nn` 530, 566, 637, 651, 739, 833
`__pdf_backend_bdc_contstream:nn` 531, 567, 643, 652, 744, 749, 838, 842
`__pdf_backend_bdc_shipout:nn` 535, 662, 757, 850
`__pdf_backend_bdc_shipout_contstream:nn` 658, 662, 753, 757, 846, 850
`__pdf_backend_bdcobject:n` 13, 521, 546, 576, 612, 640, 711, 742, 798, 836
`__pdf_backend_bdcobject:nn` 13, 521, 542, 574, 593, 687, 768
`__pdf_backend_bmc:n` 13, 521, 554, 580, 631, 735, 829


```

\__pdf_backend_catalog_gput:nn .. 20
\__pdf_backend_destination:nn ...
..... 1389, 1395, 1401, 1404
\__pdf_backend_destination:nynn ..
..... 1390, 1396, 1402, 1405
\__pdf_backend_emc: ..
..... 13, 521, 550, 578, 664, 760, 853
\__pdf_backend_indexed_structure_-
destination:nn ..
.. 1395, 1404, 1648, 1649, 1744, 1801
\__pdf_backend_indexed_structure_-
destination:nynn ..
.. 1396, 1405, 1648, 1733, 1779, 1836
\__pdf_backend_indexed_structure_-
destination_aux:nynn .. 1690, 1735
\__pdf_backend_link_begin:n .. 1505
\__pdf_backend_link_begin:nnw ..
..... 1575, 1644, 1855
\__pdf_backend_link_begin_-
goto:nnw ..
..... 1391, 1397, 1403
\__pdf_backend_link_begin_-
structure_goto:nnw 1391, 1397,
1403, 1407, 1503, 1573, 1642, 1853
\__pdf_backend_link_off: ..
..... 941, 947, 960, 971
\__pdf_backend_link_on: ..
..... 942, 951, 964, 975
\__pdf_backend_luastring:n ..
..... 163, 248, 257, 269, 270, 281, 296, 297
\__pdf_backend_metadata_stream:n
..... 1924, 1931, 1941
\g__pdf_backend_name_int ..
..... 91, 595, 598, 606,
614, 617, 625, 689, 691, 696, 705,
713, 715, 720, 729, 770, 772, 800, 802
\__pdf_backend_Names_gpsh:n ..
..... 891, 898, 905, 914, 918
\__pdf_backend_NamesEmbeddedFiles_-
add:nn ..
..... 920, 921, 924, 936
\g__pdf_backend_object_int ..
..... 1157, 1160, 1258, 1261, 1306
\__pdf_backend_object_last: ..
..... 548, 626, 721, 730, 810, 825
\__pdf_backend_object_write:nn ..
..... 1927, 1936
\__pdf_backend_omit_charset:n ..
..... 1946, 1947, 1950, 1956
\__pdf_backend_omit_cidset:n ..
..... 1991, 1992, 1995
\__pdf_backend_omit_info:n ..
.. 1961, 1962, 1969, 1975, 1982, 1988
\__pdf_backend_Page_gput:nn ..
..... 6, 184, 194, 263, 324, 363, 399
\__pdf_backend_Page_gremove:n ..
..... 6, 184, 201, 277, 331, 370, 405
\g__pdf_backend_page_int ..
..... 91
\__pdf_backend_Page_primitive:n ..
..... 6, 184, 187, 240, 253,
317, 342, 351, 356, 381, 390, 396, 417
\__pdf_backend_PageResources:n ..
..... 487, 506, 514
\c__pdf_backend_PageResources_-
clist .. 419, 429, 463, 479, 672, 883
\__pdf_backend_PageResources_-
gpsh:n ..
..... 13, 521, 558, 582, 670, 765, 867
\__pdf_backend_PageResources_-
gpsh_aux:n ..
..... 858, 884
\__pdf_backend_PageResources_-
gput:nn 428, 444, 455, 491, 507, 515
\__pdf_backend_PageResources_-
obj_gpsh: .. 428, 461, 503, 511, 519
\__pdf_backend_Pages_primitive:n
..... 149, 150, 159, 169, 175, 181
\__pdf_backend_pdfmark:n ..
..... 36, 528, 544, 548, 552, 556, 926
\__pdf_backend_record_abspage:n ..
..... 68, 79, 214, 783, 813
\__pdf_backend_ref_abspage:n ..
..... 74, 80, 217, 786, 816
\g__pdf_backend_resourceid_int ..
..... 91, 213, 214, 217, 774, 779,
783, 786, 794, 804, 809, 813, 816, 824
\__pdf_backend_set_regression_-
data: ..
..... 1860
\__pdf_backend_shipout_bdc:nn ..
..... 13, 521, 571
\__pdf_backend_structure_-
destination:nn ..
.. 1389, 1401, 1407, 1408, 1513, 1582
\__pdf_backend_structure_-
destination:nynn ..
.. 1390, 1402, 1407, 1498, 1552, 1621
\__pdf_backend_structure_-
destination_aux:nynn .. 1452, 1500
\__pdf_backend_ThisPage_gpsh:n ..
..... 6, 184, 230, 306, 349, 388, 413
\__pdf_backend_ThisPage_gput:nn ..
..... 6, 184, 210, 289, 340, 379, 410
\g__pdf_backend_thispage_-
shipout_tl ..
..... 6
\l__pdf_backend_tmpa_box ..
..... 85, 987, 996, 999, 1002, 1042,
1069, 1077, 1080, 1083, 1122, 1233,
1240, 1241, 1242, 1243, 1263, 1272,
1275, 1278, 1282, 1287, 1292, 1296,
1320, 1352, 1361, 1362, 1363, 1364

```

<code>\l__pdf_backend_tmpb_box</code>	<code>\pdfxform_if_exist:n</code> 1380
. 89, 1297, 1339, 1340, 1341, 1345	<code>\pdfxform_wd:n</code> 1185, 1240, 1361
<code>\l__pdf_backend_xform_bool</code>	prg commands:
. 522, 692,	<code>\prg_new_conditional:Npnn</code> 1374
716, 775, 805, 989, 1071, 1164, 1265	<code>\prg_new_eq_conditional:NNn</code> 1380
<code>__pdf_backend_xform_if_exist:n</code>	<code>\prg_return_false:</code> 1378
. 1374, 1380	<code>\prg_return_true:</code> 1377
<code>__pdf_backend_xform_new:nnnn</code>	prop commands:
. 980, 981, 1063, 1151, 1248, 1256	<code>\prop_count:N</code> 1011, 1092
<code>__pdf_backend_xform_ref:n</code>	<code>\prop_gclear:N</code> 990, 1072, 1266
. 980, 1055,	<code>\prop_gput:Nnn</code> 237, 494
1137, 1184, 1226, 1237, 1250, 1366	<code>\prop_gset_eq:NN</code> 232
<code>\l__pdf_backend_xform_tmpdp_tl</code>	<code>\prop_if_empty:NTF</code>
. 1254, 1290, 1304, 1311 465, 674, 860, 1021, 1026,
<code>\l__pdf_backend_xform_tmplt_tl</code>	1031, 1036, 1101, 1106, 1111, 1116
. 1255, 1285, 1309	<code>\prop_if_exist:NTF</code> 233, 871
<code>\l__pdf_backend_xform_tmpwd_tl</code>	<code>\prop_map_function:NN</code> 242, 876
. 1253, 1280, 1310	<code>\prop_map_inline:Nn</code> 235
<code>__pdf_backend_xform_use:n</code>	<code>\prop_new:N</code> 86
. 980, 1048, 1128, 1231, 1249, 1350	property commands:
<code>\g__pdf_tmpa_prop</code> 85, 232, 237, 242	<code>\property_record:nn</code> 71
<code>\l__pdf_tmpa_tl</code>	<code>\property_ref:nn</code> 76
. 85, 215, 219, 221, 224, 784,	<code>\ProvidesExplFile</code> 1
788, 790, 793, 814, 818, 820, 823, 826	
pdfdict commands:	R
<code>\pdfdict_gput:nnn</code>	<code>\relax</code> 135, 1902
. 196, 224, 326, 365, 401, 446, 457,	
509, 517, 694, 718, 777, 792, 807, 822	S
<code>\pdfdict_gremove:nn</code> 203, 333, 372, 407	scan commands:
<code>\pdfdict_if_exist:nTF</code> 219, 788, 818	<code>\scan_stop:</code> 1052,
<code>\pdfdict_item:nn</code> 242, 863, 878	1134, 1529, 1549, 1559, 1570, 1598,
<code>\pdfdict_new:n</code> 221, 790, 820	1618, 1628, 1639, 1760, 1777, 1786,
<code>\pdfdict_show:n</code> 826	1794, 1817, 1834, 1843, 1851, 1897,
<code>\pdfdict_use:n</code> 352, 391, 470, 1017, 1098	1934, 1952, 1958, 1971, 1984, 1997
<code>\pdfextension</code> 962, 966	<code>\special</code> 2
<code>\pdfliteral</code> 2	str commands:
pdfmanagement commands:	<code>\str_case:nnTF</code>
<code>\pdfmanagement_add:nnn</code>	1415, 1436, 1517, 1537, 1586, 1606,
1863, 1866, 1871, 1872, 1875, 1876,	1656, 1675, 1748, 1765, 1805, 1822
1880, 1888, 1896, 1901, 1912, 1913	<code>\str_convert_pdfname:n</code> 102, 495
pdfmanagement internal commands:	<code>\str_if_eq:nnTF</code> 1314, 1325
<code>\g__pdfmanagement_active_bool</code> 650	<code>\str_if_exist:NTF</code> 1869, 1910
<code>\l__pdfmanagement_delayed_</code>	sys commands:
<code>shipout_bool</code>	<code>\c_sys_engine_exec_str</code> 1920
. 39, 533, 569, 656, 751, 844	<code>\sys_gset_rand_seed:n</code> 1862
<code>\pdfnames</code> 20	<code>\sys_if_engine luatex:TF</code> 157
<code>\pdfomitinfodict</code> 1971	<code>\c_sys_timestamp_str</code>
<code>\pdfpageref</code> 3 1869, 1871, 1872, 1910
<code>\pdfrunninglinkoff</code> 945, 949	
<code>\pdfrunninglinkon</code> 953	T
<code>\pdftrailerid</code> 1898	\TeX and $\LaTeX 2_{\epsilon}$ commands:
pdfxform commands:	<code>\@bsphack</code> 70
<code>\pdfxform_dp:n</code> 1187, 1242, 1363	<code>\@esphack</code> 72
<code>\pdfxform_ht:n</code> 1186, 1241, 1362	<code>\@kernel@after@enddocument@afterlastpage</code>
 112, 113

<code>\@kernel@after@shipout@background</code>	133, 136	<code>\tex_pdftexversion:D</code>	1510, 1741, 1966
<code>\@kernel@after@shipout@lastpage</code>	119, 120, 126, 127	<code>\tex_pdfvariable:D</code>	1902, 1903, 1958, 1984, 1997
<code>\@kernel@before@shipout@background</code>	135	<code>\tex_pdfxform:D</code>	1004, 1085
<code>\g@addto@macro</code>	135, 136	<code>\tex_special:D</code>	42, 171, 319, 358
<code>\special</code>	2	<code>\tex_the:D</code>	996, 999, 1002, 1077, 1080, 1083, 1169, 1172, 1175, 1272, 1275, 1278
tex commands:		<code>\tex_unexpanded:D</code>	250
<code>\tex_directlua:D</code>	161, 265, 279, 432, 434	<code>\tex_vss:D</code>	1462, 1494, 1700, 1729
<code>\tex_global:D</code>	152, 189, 869	text commands:	
<code>\tex_immediate:D</code>	1004, 1085, 1926, 1935	<code>\text_expand:n</code>	102, 108
<code>\tex_latelua:D</code>	255, 291, 308, 447, 448, 700, 724	tl commands:	
<code>\tex_luaescapestring:D</code>	250	<code>\c_space_tl</code>	598, 606, 617, 625, 691, 715, 772, 802, 1185, 1186, 1187, 1306
<code>\tex luatexversion:D</code>	958	<code>\tl_const:Nn</code>	994, 997, 1000, 1075, 1078, 1081, 1167, 1170, 1173, 1270, 1273, 1276
<code>\tex_pdfcompresslevel:D</code>	1934	<code>\tl_gput_right:Nn</code>	113, 120, 127
<code>\tex_pdfdest:D</code>	1515, 1532, 1554, 1562, 1746, 1761, 1781, 1787	<code>\tl_if_exist:NTF</code>	133
<code>\tex_pdfextension:D</code>	48, 58, 901, 1584, 1601, 1623, 1631, 1803, 1818, 1838, 1844, 1926	<code>\tl_new:N</code>	87, 1253, 1254, 1255, 1384
<code>\tex_pdflastxform:D</code>	1045, 1125	<code>\tl_set:Nn</code>	215, 784, 814, 1280, 1285, 1290
<code>\tex_pdfliteral:D</code>	51, 61	<code>\tl_to_str:n</code>	995, 998, 1001, 1044, 1051, 1057, 1076, 1079, 1082, 1124, 1132, 1138, 1159, 1168, 1171, 1174, 1228, 1260, 1271, 1274, 1277, 1333, 1357, 1368, 1376, 1535, 1565, 1604, 1634
<code>\tex_pdfnames:D</code>	894	<code>\tl_use:N</code>	1304, 1309, 1310, 1311
<code>\tex_pdfobj:D</code>	1935	V	
<code>\tex_pdfomitcharset:D</code>	1952	vbox commands:	
<code>\tex_pdfpageattr:D</code>	189	<code>\vbox_to_zero:n</code>	1454, 1465, 1692, 1703
<code>\tex_pdfpageresources:D</code>	869		
<code>\tex_pdfpagesattr:D</code>	152		
<code>\tex_pdfrefxform:D</code>	1050, 1130		
<code>\tex_pdfsuppressptexinfo:D</code>	1897		
<code>\tex_pdftexrevision:D</code>	1511, 1742, 1967		