The l3pdfannot module
Commands for PDF annotations
LaTeX PDF management testphase bundle

The LaTeX Project*

Version 0.95e, released 2021-06-14

1 l3pdfannot documentation

This module contains a number of commands to create PDF annotations. The commands are *not* always simple wrappers around primitive commands. To allow external packages to configure links and other annotations, some of the commands have hooks and use shared attribute dictionaries. For these commands the hooks and dictionaries are selected depending on the ⟨*type*⟩ of the annotation. Currently the module only supports some general commands and link annotations. Commands for other annotations like widgets will be added later.

1.1 General annotation commands

\pdfannot_box:nnnn
\pdfannot_box:nnnx

New: 2019-09-05
Updated: 2020-04-14

\pdfannot_box:nnnn {⟨*width*⟩} {⟨*height*⟩} {⟨*depth*⟩} {⟨*annot spec*⟩}

This creates an /Type/Annot object with the given dimensions. It doesn't use hooks or dictionaries. The annotation doesn't occupy space but as it is a whatsit it can affect spacing.

\pdfannot_box_ref_last:

New: 2019-09-05

\pdfannot_box_ref_last:

This retrieves the object reference of the last box annotation created.

1.2 Dictionary for the annotation spec

⟨*annot spec*⟩ in the above command can be given in two ways. One way is to enter the needed dictionary keys and values directly:

\pdfannot_box:nnnn{1cm}{1cm}{0cm}{/Subtype/Link /Border[0~0~1]}

A second method is to make use of the dictionary commands provided by l3pdfdict:

```
\pdfdict_new:n  {l_my_annot}
\pdfdict_put:nnn{l_my_annot}{Subtype}{/Link}
\pdfdict_put:nnn{l_my_annot}{Border}{[0~0~1]}
\pdfannot_box:nnnx{1cm}{1cm}{0cm}{\pdfdict_use:n{l_my_annot}}
```

The second method is clearly slower and more to type. But it has the advantage that using such a dictionary makes it easy to add, remove and change entries. It also avoids the potential problem that a key is added twice with different values. This allows to create user interfaces to change settings and also makes it easy to extend the interfaces in case some new setting should be included. For these reasons both the PDF management itself, but also the specific annotation commands in the following sections all make use of such dictionaries.

1.3   Link annotations

Link annotations are special cases of annotations. In the PDF they are identified by an `/Subtype/Link` entry in the dictionary. Link annotations are quite important as many documents contain links, both internal and external. They need a set of special commands for two reasons:

At first the content of links are not only boxes. Links can contain line and page breaks (this is normally implemented by the primitive command by creating a set of annotations).

At second link annotations are objects that need some "management" as more than one package wants to configure their look and behaviour. For example hyperref, ocgx2 and the code for tagged PDF (currently in `tagpdf`) all want to add keys and values to the dictionaries of link annotation and code around links. So commands to create link annotations should offer suitable hooks. There are three standard places in a link where such hooks are needed: At the begin (for example for a structure command or color), in the *attr spec* dictionary of the link (for example for the border), and at the end of the link (to close a structure or the color group). For the begin and end hooks of the LaTeX hook management are predefined and used. To add and remove values from the *attr spec* dictionary special commands described below are provided. The link commands switch to horizontal mode as the commands of pdftex and luatex can't be used in vertical mode.

`\c_pdfannot_link_types_seq` There are currently five link types, `URI`, `GoToR`, `Launch`, `GoTo` or `Named`, and there are store in this constant.

`pdfannot/link/TYPE/before` These are the hooks used by the following commands. TYPE can be one of `URI`, `GoToR`,
`pdfannot/link/TYPE/begin` `Launch`, `GoTo` or `Named`
`pdfannot/link/TYPE/end`
`pdfannot/link/TYPE/after`

`link/TYPE` These is the name of the dictionary used by the following commands. TYPE can be one of `URI`, `GoToR`, `Launch`, `GoTo` or `Named`. The dictionary can be changed by the commands `\pdfannot_dict_put:nnn` and friends described below.

`\pdfannot_link:nnn {⟨type⟩} {⟨user action spec⟩} {⟨link text⟩}`

This creates a link around the ⟨*link text*⟩. `/Subtype/Link` is added automatically through the dictionary. ⟨*user action spec*⟩[1]. is provided as a fast method to add dictionary contents, but it should be noted that no provision is taken to avoid clashes with values added through the dictionary. If needed clashing entries should be removed from the dictionary first. Normally the argument is not needed, all entries can be added through the dictionary too. ⟨*type*⟩ should be one of `URI`, `GoToR`, `Launch`, `GoTo` or `Named`. The `GoTo` variant does *not* complain if the destination name is not known like `\pdfannot_-link_goto_begin:nw`. The attributes stored in the local dictionary `link/⟨type⟩` are inserted as *attr spec* before ⟨*user action spec*⟩. The code in the begin and end hook `pdfannot/link/⟨type⟩/before` and `pdfannot/link/⟨type⟩/after` is executed before and after the link (outside the link command) while `pdfannot/link/⟨type⟩/begin` and `pdfannot/link/⟨type⟩/end` are directly around the link text. None of the hooks introduce a group. ⟨*type*⟩ should normally be identical to the value of the `/S` key in the action dictionary. As example either with a direct action

```
\pdfannot_link:nnn { URI }
  {
    /A<<
      /Type/Action
      /S/URI
      /URI(https://www.latex-project.org)
    >>
  }
  { link text}
```

Or through a dictionary:

```
\pdfdict_new:n    {l_my_action_dict}
\pdfdict_put:nnn {l_my_action_dict}{Type}{/Action}
\pdfdict_put:nnn {l_my_action_dict}{S}{/URI}
\pdfdict_put:nnn {l_my_action_dict}{URI}{(https://www.latex-project.org)}

\pdfannot_dict_put:nnn
  {link/URI} { C } {[1~0~0]} %red border

\pdfannot_link:nxn { URI }
 {
   /A <<\pdfdict_use:n{l_my_action_dict}>>
 }
 { link text }
```

Or if you want to exclude the possibility of a duplicated /A entry (if the action is already in the link/GoTo dictionary e.g. if you can expect other packages to add a dictionary). An alternative is to ensure that no /A is there by removing it explicitly.

```
\pdfdict_new:n    {l_my_action_dict}
\pdfdict_put:nnn {l_my_action_dict}{Type}{/Action}
\pdfdict_put:nnn {l_my_action_dict}{S}{/URI}
\pdfdict_put:nnn {l_my_action_dict}{URI}{(https://www.latex-project.org)}


\pdfannot_dict_put:nnn
  {link/URI} { C } {[1~0~0]} %red border

\group_begin:
\pdfannot_dict_put:nnx {link/GoTo}{A}{<<\pdfdict_use:n{l_my_action_dict}>>}
\pdfannot_link:nnn { URI }{}{ link text }
\group_end:
```

`\pdfannot_link_begin:nnw`	`\pdfannot_link_begin:nnw {⟨type⟩} {⟨user action spec⟩} ⟨content⟩`
`\pdfannot_link_end:n`	`\pdfannot_link_end:n {⟨type⟩}`

Updated: 2020-12-06

This creates a link like the previous command. `/Subtype/Link` is added automatically through the dictionary. ⟨*user action spec*⟩[2]. is provided as a fast method to add dictionary contents, but it should be noted that no provision is taken to avoid clashes with values added through the dictionary. If needed clashing entries should be removed from the dictionary first. Normally the argument is not needed, all entries can be added through the dictionary too. `/Subtype/Link` is added automatically. In contrast to `\pdfannot_-link:nnn` this function does not absorb the argument when finding the ⟨*content*⟩, and so can be used in circumstances where the ⟨*content*⟩ may not be a simple argument. But beside this, it works similar and use the same hooks. As example

```
\pdfannot_link_begin:nnw { URI }
 {
   /A<<
     /Type/Action
     /S/URI
     /URI(https://www.latex-project.org)
   >>
 }
 link text
\pdfannot_link_end:n { URI }
```

`\pdfannot_link_goto_begin:nw`	`\pdfannot_link_goto_begin:nw {⟨destination⟩} ⟨content⟩`
`\pdfannot_link_goto_end:`	`\pdfannot_link_goto_end:`

Updated: 2020-12-06

This is a special, shorter version for links to internal destinations. It always uses the hooks and dictionary of the `GoTo` link type. ⟨*destination*⟩ is a destination name. In difference to `\pdfannot_link:nnn { GoTo }` it will complain if ⟨*destination*⟩ is an unknown destination and give the message

    name{ZZZZ} has been referenced but does not exist, replaced by a fixed one

`\pdfannot_link_ref_last:`

New: 2021-02-14

This retrieves the object reference a link created previously with the commands above. This doesn't work currently with xelatex but a feature request has been made. see https://tug.org/pipermail/dvipdfmx/2020-December/000134.html

`\pdfannot_ref_last:`

New: 2021-02-14

This retrieves the object reference a previously annotation created either with a link or a general box command. When the last was a link it won't work with xelatex. see https://tug.org/pipermail/dvipdfmx/2020-December/000134.html

`\pdfannot_link_margin:n`	`\pdfannot_link_margin:n {⟨dimen⟩}`

New: 2020-03-12

This sets the dimension of the link margin.

\pdfannot_dict_put:nnn

New: 2020-12-04

`\pdfannot_dict_put:nnn {⟨dictionary name⟩} {⟨key⟩} {⟨value⟩}`

This adds (locally) a key-value to the internal annot dictionaries used by the link commands above. ⟨*dictionary name*⟩ should be currently one of `link/URI`, `link/URI,link/GoToR`, `link/Launch`, `link/GoTo`, `link/Named`.

\pdfannot_dict_remove:nn

New: 2020-12-04

`\pdfannot_dict_remove:nn {⟨dictionary name⟩} {⟨key⟩}`

This removes a key-value from the internal annot dictionary ⟨*dictionary name*⟩ should be currently one of `link/URI`, `link/GoToR`, `link/Launch`, `link/GoTo`, `link/Named`.

\pdfannot_dict_show:n

New: 2020-12-04

`\pdfannot_dict_show:n {⟨dictionary name⟩}`

This shows the content of the internal annot dictionary. ⟨*dictionary name*⟩ should be currently one of `link/URI`, `link/URI`, `link/GoToR`, `link/Launch`, `link/GoTo`, `link/Named`.

\pdfannot_dict_use:n ⋆

New: 2021-03-03

`\pdfannot_dict_use:n {⟨dictionary name⟩}`

This outputs the property list of the dictionary as a list of `/key value` pairs. This can be used e.g. when writing a dictionary object with `\pdf_object_write:nx`

\l_pdfannot_F_bitset

New: 2020-12-28

This is a bitset variable, with the named index names suitable for the /F flag in an annotation. It can be used for example like this:

```
\bitset_set_true:Nn \l_pdfannot_F_bitset {Print}
\pdfannot_dict_put:nnx {link/URI} {F}
   { \bitset_to_arabic:N \l_pdfannot_F_bitset }
```

The known keys for the bitset are `Invisible`, `Hidden`, `Print`, `NoZoom`, `NoRotate`, `NoView`, `ReadOnly`, `Locked`, `ToggleNoView`, `LockedContents` which correspond to the names used in the PDF references.

1.4 Widget annotations

Widget annotations are quite important for form fields, as they are used to build the actually instance of such fields.

As they can contain meaningful content hooks are probably needed to allow tagging and other manipulations, so like with link special commands are provided. Widget are normally in a box and line and page breaks are not relevant, so the command is offered as box command.

\pdfannot_widget_box:nnn

New: 2021-03-02

`\pdfannot_widget_box:nnn  {⟨width⟩} {⟨height⟩} {⟨depth⟩}`

This creates an **/Type/Annot** object with the given dimensions. The annotation doesn't occupy space. It will insert the attribute dictionary of the widget type (which is prefilled with **/Subtype/Widget**). The hooks `pdfannot/widget/before` and `pdfannot/widget/after` are executed before and after the widget. The widget has four subdirectories, `widget/AA`, `widget/AP`, `widget/MK` and `widget/BS` which can be filled with `\pdfannot_dict_put:nnn` and will be used if not empty.

2   l3pdfannot implementation

```
1  ⟨@@=pdfannot⟩
2  ⟨*header⟩
3  \ProvidesExplPackage{l3pdfannot}{2021-06-14}{0.95e}
4    {PDF-annotations}
5  \RequirePackage{l3pdfdict}
6  ⟨/header⟩
```

Annotations have a /F flag, we provide a public bitset for it.

```
7  ⟨*package⟩
8  \RequirePackage{l3bitset}
9  \bitset_new:Nn \l_pdfannot_F_bitset
10   {
11     Invisible     = 1,
12     Hidden        = 2,
13     Print         = 3,
14     NoZoom        = 4,
15     NoRotate      = 5,
16     NoView        = 6,
17     ReadOnly      = 7,
18     Locked        = 8,
19     ToggleNoView  = 9,
20     LockedContents = 10
21   }
```

2.1   General Annotations

\g__pdfannot_use_lastlink_bool

The pdf engines have two different primitive commands to refer to the last created annotation: one for links, one for boxed annotation. We use a boolean to decide which one should be used, so that only one user command is needed.

```
22  \bool_new:N \g__pdfannot_use_lastlink_bool
```

(*End definition for* \g__pdfannot_use_lastlink_bool.)

\pdfannot_box:nnnn
\pdfannot_box:nnnx
\pdfannot_box_ref_last:

```
23  \cs_new_protected:Npn \pdfannot_box:nnnn #1 #2 #3 #4
24    {
25      \__pdf_backend_annotation:nnnn {#1}{#2}{#3}{#4}
26      \bool_gset_false:N\g__pdfannot_use_lastlink_bool
27    }
28  \cs_generate_variant:Nn \pdfannot_box:nnnn {nnnx}
29  \cs_new:Npn \pdfannot_box_ref_last:
30    {
31      \__pdf_backend_annotation_last:
32    }
33
```

(*End definition for* \pdfannot_box:nnnn *and* \pdfannot_box_ref_last:. *These functions are documented on page 1.*)

2.2 Annotations, subtype Widget

Widgets are typically boxes, so we provide a box command. A local dictionary `l_-@@/Widget` is used. It contains like the other dictionaries the subtype setting (the /Type is added by the backend).

```
34  \pdfdict_new:n   { l__pdfannot/widget }
35  \pdfdict_new:n   { l__pdfannot/widget/AA }
36  \pdfdict_new:n   { l__pdfannot/widget/AP }
37  \pdfdict_new:n   { l__pdfannot/widget/MK }
38  \pdfdict_new:n   { l__pdfannot/widget/BS }
39  \pdfdict_put:nnn { l__pdfannot/widget }{ Subtype }{ /Widget }
40  \hook_new_pair:nn
41    {pdfannot/widget/before}
42    {pdfannot/widget/after}
43  \cs_new_protected:Npn \pdfannot_widget_box:nnn #1 #2 #3
44    {
45      \hook_use:n { pdfannot/widget/before }
46      \group_begin:
47      \pdfmeta_standard_verify:nT
48        {annot_widget_no_AA}
49        {
50          \pdfdict_if_empty:nF { l__pdfannot/widget/AA }
51            {
52              \pdf_object_unnamed_write:nx {dict}{\pdfdict_use:n{l__pdfannot/widget/AA}}
53              \pdfdict_put:nnx { l__pdfannot/widget }
54                {AA}
55                {\pdf_object_ref_last:}
56            }
57        }
58      \pdfdict_if_empty:nF { l__pdfannot/widget/AP }
59        {
60          \pdf_object_unnamed_write:nx {dict}{\pdfdict_use:n{l__pdfannot/widget/AP}}
61          \pdfdict_put:nnx { l__pdfannot/widget }
62              {AP}
63              {\pdf_object_ref_last:}
64        }
65      \pdfdict_if_empty:nF { l__pdfannot/widget/MK }
66        {
67          \pdf_object_unnamed_write:nx {dict}{\pdfdict_use:n{l__pdfannot/widget/MK}}
68          \pdfdict_put:nnx { l__pdfannot/widget }
69              {MK}
70              {\pdf_object_ref_last:}
71        }
72      \pdfdict_if_empty:nF { l__pdfannot/widget/BS }
73        {
74          \pdf_object_unnamed_write:nx {dict}{\pdfdict_use:n{l__pdfannot/widget/BS}}
75          \pdfdict_put:nnx { l__pdfannot/widget }
76              {BS}
77              {\pdf_object_ref_last:}
78        }
79      \pdfannot_box:nnnx {#1}{#2}{#3}
80        {
81          \pdfdict_use:n { l__pdfannot/widget}
82        }
```

```
83    \hook_use:n { pdfannot/widget/end }
84    \group_end:
85    \bool_gset_false:N\g__pdfannot_use_lastlink_bool
86  }
```

2.3 Annotations, subtype Link

The code assumes that there will be different link types (currently URI, GoToR, Launch, GoTo, Named, hyperref uses the names url,file,run,link,menu) and that links of the same type share the *attr spec* and also the same begin/end code. The list of link types need to stay restricted and well documented so that all packages know which types they have to handle. It is stored in a constant seq.

\c_pdfannot_link_types_seq   This constant sequence contains the list of currently supported link types for which hooks and dictionaries exist.

(*End definition for* \c_pdfannot_link_types_seq. *This variable is documented on page 2.*)

link/TYPE   These setup the dictionary and the hook pairs.
pdfannot/link/TYPE/before
pdfannot/link/TYPE/begin
pdfannot/link/TYPE/end
pdfannot/link/TYPE/after

```
87 \seq_const_from_clist:Nn \c_pdfannot_link_types_seq { URI , GoToR , Launch , GoTo, Named }
88 \seq_map_inline:Nn \c_pdfannot_link_types_seq
89   {
90     \pdfdict_new:n { l__pdfannot/link/#1 }
91     \pdfdict_put:nnn { l__pdfannot/link/#1 }{ Subtype }{ /Link }
92     \hook_new_pair:nn
93       {pdfannot/link/#1/before}
94       {pdfannot/link/#1/after}
95     \hook_new_pair:nn
96       {pdfannot/link/#1/begin}
97       {pdfannot/link/#1/end}
98   }
```

(*End definition for* link/TYPE *and others. These variables are documented on page 2.*)

2.3.1 Annotations, subtype Link /management

\pdfannot_link:nnn
\pdfannot_link:nxn

```
99 \cs_new_protected:Nn \pdfannot_link:nnn %#1 type (URI, GoTo etc),
100                                        %#2 action spec, #3 link text
101  {
102    \hook_use:n { pdfannot/link/#1/before}
103    \mode_leave_vertical:
104    \exp_args:Nxx %xetex needs expansion
105    \__pdf_backend_link_begin_user:nnw
106      {
107        \pdfdict_if_exist:nT { l__pdfannot/link/#1 }
108        {
109          \pdfdict_use:n { l__pdfannot/link/#1}
110        }
111      }
112      {
113        #2 %exp_not?
114      }
115    \bool_gset_true:N \g__pdfannot_use_lastlink_bool
```

8

```
116     \hook_use:n { pdfannot/link/#1/begin}
117     #3
118     \hook_use:n { pdfannot/link/#1/end}
119     \__pdf_backend_link_end:
120     \bool_gset_true:N \g__pdfannot_use_lastlink_bool
121     \hook_use:n { pdfannot/link/#1/after}
122   }
123 \cs_generate_variant:Nn \pdfannot_link:nnn {nxn}
```

(*End definition for* `\pdfannot_link:nnn`. *This function is documented on page 3.*)

\pdfannot_link_begin:nnw
\pdfannot_link_begin:nxw
\pdfannot_link_end:n

```
124 \cs_new_protected:Npn \pdfannot_link_begin:nnw #1 #2 %#1 type, #2 action spec
125   {
126     \hook_use:n { pdfannot/link/#1/before}
127     \mode_leave_vertical:
128     \exp_args:Nxx %xetex needs expansion
129       \__pdf_backend_link_begin_user:nnw
130         {
131           \pdfdict_if_exist:nT { l__pdfannot/link/#1 }
132             {
133               \pdfdict_use:n { l__pdfannot/link/#1}
134             }
135         }
136         { #2 }
137       \bool_gset_true:N \g__pdfannot_use_lastlink_bool
138       \hook_use:n { pdfannot/link/#1/begin}
139   }
140
141 \cs_new_protected:Nn \pdfannot_link_end:n %#1 type, e.g. url
142   {
143     \hook_use:n { pdfannot/link/#1/end}
144     \__pdf_backend_link_end:
145     \bool_gset_true:N \g__pdfannot_use_lastlink_bool
146     \hook_use:n { pdfannot/link/#1/after}
147   }
148 \cs_generate_variant:Nn \pdfannot_link_begin:nnw {nxw}
```

(*End definition for* `\pdfannot_link_begin:nnw` *and* `\pdfannot_link_end:n`. *These functions are documented on page 4.*)

\pdfannot_link_goto_begin:nw
\pdfannot_link_goto_end:

```
149 \cs_new_protected:Npn \pdfannot_link_goto_begin:nw #1 %#1 destination
150   {
151     \pdfdict_remove:nn { l__pdfannot/link/GoTo } {Subtype}
152     \hook_use:n { pdfannot/link/GoTo/before} %the backend add it too
153     \mode_leave_vertical:
154     \exp_args:Nxx %xetex needs expansion
155     \__pdf_backend_link_begin_goto:nnw
156       {
157         \pdfdict_use:n { l__pdfannot/link/GoTo}
158       }
159       { #1 }
160       \bool_gset_true:N \g__pdfannot_use_lastlink_bool
161     \pdfdict_put:nnn { l__pdfannot/link/GoTo } {Subtype}{GoTo}
```

```
162        \hook_use:n { pdfannot/link/GoTo/begin}
163    }
164
165  \cs_new_protected:Nn \pdfannot_link_goto_end:
166    {
167      \hook_use:n { pdfannot/link/GoTo/end}
168      \__pdf_backend_link_end:
169      \bool_gset_true:N \g__pdfannot_use_lastlink_bool
170      \hook_use:n { pdfannot/link/GoTo/after}
171    }
```

(*End definition for* \pdfannot_link_goto_begin:nw *and* \pdfannot_link_goto_end:. *These functions are documented on page* 4.)

\pdfannot_link_ref_last:
\pdfannot_ref_last:

```
172  \cs_new:Nn \pdfannot_link_ref_last: { \__pdf_backend_link_last: }
```

```
173  \cs_new:Npn \pdfannot_ref_last:
174    {
175      \bool_if:NTF \g__pdfannot_use_lastlink_bool
176        {
177          \__pdf_backend_link_last:
178        }
179        {
180          \__pdf_backend_annotation_last:
181        }
182    }
```

(*End definition for* \pdfannot_link_ref_last: *and* \pdfannot_ref_last:. *These functions are documented on page* 4.)

\pdfannot_link_margin:n

```
183  \cs_new_protected:Npn \pdfannot_link_margin:n #1
184    {
185      \__pdf_backend_link_margin:n { #1 }
186    }
```

(*End definition for* \pdfannot_link_margin:n. *This function is documented on page* 4.)

\pdfannot_dict_put:nnn
\pdfannot_dict_put:nnx
\pdfannot_dict_remove:nn
\pdfannot_dict_show:n ␣␣␣␣␣\pdfannot_dict_use:n

```
187  \cs_new_protected:Npn \pdfannot_dict_put:nnn #1 #2 #3
188    {
189      \pdfdict_put:nnn { l__pdfannot/#1 } { #2 }{ #3 }
190    }
191  \cs_generate_variant:Nn \pdfannot_dict_put:nnn {nnx}
```

```
192  \cs_new_protected:Npn \pdfannot_dict_remove:nn #1 #2
193    {
194      \pdfdict_remove:nn { l__pdfannot/#1 } { #2 }
195    }
```

```
196  \cs_new_protected:Npn \pdfannot_dict_show:n #1
197    {
198      \pdfdict_show:n { l__pdfannot/#1 }
199    }
200
201  \cs_new:Npn \pdfannot_dict_use:n #1
```

```
202    {
203        \pdfdict_use:n { l__pdfannot/#1 }
204    }
205  ⟨/package⟩
```

(*End definition for* `\pdfannot_dict_put:nnn` *,* `\pdfannot_dict_remove:nn` *, and* `\pdfannot_dict_show:n`
`\pdfannot_dict_use:n`*. These functions are documented on page 5.*)

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.