The l3pdffield-radiobutton module
Commands to create radio form fields
LaTeX PDF management testphase bundle

The LaTeX Project*

Version 0.95i, released 2021-08-28

1   l3pdffield-radiobutton Introduction

This is the documentation for radio buttons fields, for general information about form fields check the documentation l3pdffield.

Choose your favorite

Please keep in mind

- Not every PDF viewer supports radio buttons.

- The handling can depend on settings in the PDF viewer. In adobe reader for example I had to disable an option to avoid that it tries to create an appearance itself

- Standards like pdf/A disable features of form fields too (as you typically can't change the PDF).
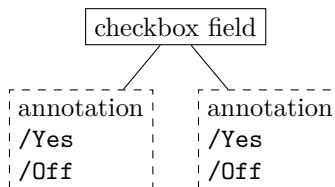
2   Radio buttons

Click me:

Radio buttons are similar to checkboxes and they have like checkboxes two "states": checked and unchecked.

The difference is that multiple checkboxes are either "clones" of each other which are checked and unchecked together, or they are independent of each other. Radio buttons on the other side build *groups* where checking one of the buttons unchecks all other buttons.

---

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

The elements of a such a radio button group are annotations of one field which has a flag set which makes it into a radio button group.
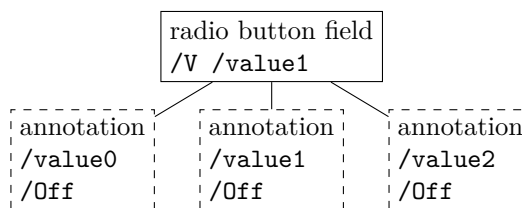
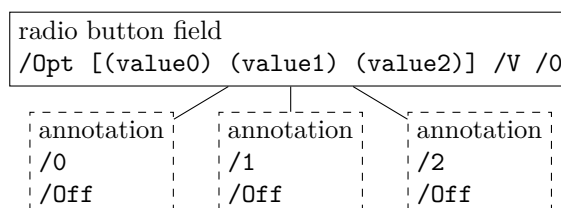In a checkbox field the two states have the fix names `/Yes` and `/Off`.

```
        ┌──────────────┐
        │checkbox field│
        └──────────────┘
         ╱            ╲
┌──────────────┐  ┌──────────────┐
│ annotation   │  │ annotation   │
│ /Yes         │  │ /Yes         │
│ /Off         │  │ /Off         │
└──────────────┘  └──────────────┘
```

In radio buttons the off state should still always have the name `/Off`[1] but the on state should be a specific value for every button. The field dictionary should then set in the `/V` key as the value of one of the button. This button is then set as selected and all the other are deselected when the PDF is opened or reset.

It is theoretically possible to control the start appearance state for every button so e.g. all buttons could have the "selected" state when the PDF is opened. But as soon as one button is clicked you get one selected button and the other are unselected. You can't select or deselect all buttons. The PDF reference mentions an flag `NoToggleToOff` but this doesn't do anything, at least not in the PDF viewers I tried. For this reason setting special start states is currently not supported: it complicates the code and the input for no real gain.

The values of the individual buttons can be coded as annotation state names:

```
        ┌────────────────────┐
        │ radio button field │
        │ /V /value1         │
        └────────────────────┘
       ╱          │          ╲
┌───────────┐ ┌───────────┐ ┌───────────┐
│ annotation│ │ annotation│ │ annotation│
│ /value0   │ │ /value1   │ │ /value2   │
│ /Off      │ │ /Off      │ │ /Off      │
└───────────┘ └───────────┘ └───────────┘
```

But it can be awkward to have to use the values also as names of appearances states, it makes it for example difficult to use unicode for the value names, so there is another option: One can set up an `/Opt` array which contains the values as strings, and used "named numbers" as appearance state name: The numbers `/0`, `/1` point then to the index position in the array.

```
┌──────────────────────────────────────────────┐
│ radio button field                           │
│ /Opt [(value0) (value1) (value2)] /V /0      │
└──────────────────────────────────────────────┘
       ╱          │          ╲
┌───────────┐ ┌───────────┐ ┌───────────┐
│ annotation│ │ annotation│ │ annotation│
│ /0        │ │ /1        │ │ /2        │
│ /Off      │ │ /Off      │ │ /Off      │
└───────────┘ └───────────┘ └───────────┘
```

This method is clearly more flexible, and so it is used in this module.

As radio buttons build a group of buttons, there is more interaction going on, and more values have to be set. Also the first command, which initializes the field, has to set the default value of the group.

This means a typical setup should do something like this

---

[1] the PDF reference doesn't say anything about this, but various tests showed that one better should stick to this name, with other names the buttons disappeared.

```
\pdffield_radio:n
  {
    group   = A,          % required, can also be given as name=A or T=A
    value   = button1,    % required,  on-state of this button
    default = button4     % default of the group,
                          % if not given choice (button1) is used
                          % should refer to an existing button!
                          % It will set the V and the DV key
                          % button4 will be checked
  }
\pdffield_radio:n
  {
    group   = A,          % required, can also be given as name=A or T=A
    value   = button2     % required,  export value of this button
  }
\pdffield_radio:n
  {
    group   = A,          % required, can also be given as name=A or T=A
    value   = button3,    % required,  choice/export value of this button
  }
\pdffield_radio:n
  {
    group   = A,          % required, can also be given as name=A or T=A
    value   = button4     % required,  choice/export value of this button
  }
```

If two radio field annotations use the same `value` they are selected and unselected together, like checkboxes with the same `/Yes` state. This can be used to build radio groups which works "in unison". The flag `RadiosInUnison` is neither needed for this (but doesn't harm either) nor does it change the behaviour, at least again not in the PDF viewers I tried.

2.1   Commands

\pdffield_radio:n   \pdffield_radio:n{⟨*key val list*⟩}

This creates a radio button to check and uncheck. The list of allowed keys is described below. The ⟨*key val list*⟩ should at least set the group name, without it the default group `radio` is used. Radiobuttons with the same group name belong to the same field and if checked, the others are unchecked. The default appearance is a circle frame with a black bullet in it for the checked case. The default appearance is setup at the first use. Its geometry is quadratic.

The first radio button setups the field and should also set the button which should be shown as selected when the PDF is opened.

2.2   Keys

The new radio command accept all field and annot keys from l3pdffield. A few keys are disabled or are forced to specific values. The `appearance` keys have a more radio specific

behaviour, other keys have other defaults than with the basic commands. Additionally there are a small number of keys specific to a radio button. `value` and `default` have a special meaning.

Disabled keys are

- `V`, `DV`, `AS`: they are set by the other keys.

- `FT` is overwritten.

- For radio buttons only the field flags `ReadOnly`, `Required`, `NoExport`, `NoToggleToOff` and `RadiosInUnison` make sense. The last two are as mentioned above probably useless. `Radio` is set automatically automatically by the code as this is required for a radio button set.

---

preset-radiobutton **preset-radiobutton = {⟨*key-val-list*⟩}**

This allows to set default keys for a radio button.

---

group **group = ⟨*partial name*⟩**
name   **name = ⟨*partial name*⟩**
T   **T = ⟨*partial name*⟩**

These keys set the partial name of the field. They all do the same thing, use the one you are more comfortable with. The value shouldn't contain a period, be not empty and sensibly consist of simple chars. Additionally the value is used to create the field ID. This means that radio buttons with the same partial name are annotations with the same field as parent and so build a radio button group. The field ID is then internal and can not be used to attach another annotation. For explicit control of the field ID use the `fieldID` key.

---

value **value = ⟨*string*⟩**

With this key you set the export value name for the individual buttons. Every button should have a value. buttons with the same value are checked in unison. The value can use unicode.

---

default **default = ⟨*string*⟩**

With this key you set the button which is checked when the PDF is opened. The key must be used in the command of the first button. If it is missing the value of the first button is used as default. ⟨*string*⟩ should be identical to a `value` of one of the buttons of the group. This is not checked, buttons can be created on various pages, so it would require a label/ref system to keep track of the names.

---

fieldID **fieldID = ⟨*field ID*⟩**

For experts only! This allows to give the radio field a specific ID. This is only useful in the context of a larger fieldset or if you want to attach another annotation to the field with `\pdffield_annot:n`. If used wrongly you can easily create invalid fieldset. It allows you to create to fields with the same partial name, but if you want to see both you need to ensure that their full names are different—for example by adding some parent fields.

parent	parent = ⟨*field ID*⟩

This is only needed if the field should be part of a larger fieldset. The value should be a field ID of a field created previously with `\pdffield_field:nn`.

width	width = ⟨*dim expression*⟩
height	height = ⟨*dim expression*⟩
depth	depth = ⟨*dim expression*⟩

These keys allow to set the dimensions of radio button. The value should be a dimension expression. By default `width` and `height` use `\normalbaselineskip`, the `depth` is zero.

AP/N	AP/N = ⟨*partial appearance name*⟩
appearance	appearance = ⟨*partial appearance name*⟩
AP/R	AP/R = ⟨*partial appearance name*⟩
rollover-appearance	rollover-appearance = ⟨*partial appearance name*⟩
AP/D	AP/D = ⟨*partial appearance name*⟩
down-appearance	down-appearance = ⟨*partial appearance name*⟩

This keys sets the normal appearance, the rollover appearance (when the mouse hovers over the checkbox) and the down appearance (when the mouse clicks). They take as value a ⟨*partial appearance name*⟩ and expects that *two* form Xobjects ⟨*partial appearance name*⟩/`Yes` and ⟨*partial appearance name*⟩/`Off` has been created. The initial value is `pdffield/radio/default` for the normal appearance and shows a button. The down appearance enlarges the middle button a bit to give a visual feed back.

2.3 Using with hyperref

Radio buttons in hyperref are created with `\ChoiceMenu` command, and they also print a label. choices are actually a different field type and so created with different commands. It is not quite clear yet, how this could be mapped.

3 l3pdffield-radiobutton Implementation

1 ⟨*package⟩
2 ⟨@@=pdffield⟩
3 `\RequirePackage{l3draw}`

3.1 Variables

\l__pdffield_radio_value_tl
\l__pdffield_radio_default_tl
\l__pdffield_radio_value_num_int
\l__pdffield_radio_appearance_code_tl

variables to hold the value, its index numbers and the default value. radio buttons can setup the appearance only after the value is known, so the code is stored and executed later.

4 `\tl_new:N  \l__pdffield_radio_value_tl`
5 `\tl_new:N  \l__pdffield_radio_default_tl`
6 `\int_new:N \l__pdffield_radio_value_num_int`
7 `\tl_new:N  \l__pdffield_radio_appearance_code_tl`

(*End definition for* `\l__pdffield_radio_value_tl` *and others.*)

3.2 Appearances

The default appearances are a circle with button in it. Every appearance should have two versions and follow the naming module/⟨*name*⟩/Yes and module/⟨*name*⟩/Off.

This defines the standard appearances. They are setup at the first use of a radiobutton.

```
8 \cs_new_protected:Npn \__pdffield_radio_default_appearances:
9   {
10     \pdffield_appearance:nn {pdffield/radio/default/Yes}
11       {
12         \normalsize
13         \draw_begin:
14         \draw_path_circle:nn {0pt,0pt}{0.5\normalbaselineskip}
15         \draw_path_use_clear:n { stroke }
16         \draw_path_circle:nn {0pt,0pt}{0.2\normalbaselineskip}
17         \draw_path_use_clear:n { fill }
18         \draw_end:
19       }
20      \pdffield_appearance:nn {pdffield/radio/default/Off}
21       {
22         \normalsize
23         \draw_begin:
24         \draw_path_circle:nn {0pt,0pt}{0.5\normalbaselineskip}
25         \draw_path_use_clear:n { stroke }
26         \draw_end:
27       }
28
29     \pdffield_appearance:nn {pdffield/radio/defaultdown/Yes}
30       {
31         \normalsize
32         \draw_begin:
33         \draw_path_circle:nn {0pt,0pt}{0.5\normalbaselineskip}
34         \draw_path_use_clear:n { stroke }
35         \draw_path_circle:nn {0pt,0pt}{0.25\normalbaselineskip}
36         \draw_path_use_clear:n { fill }
37         \draw_end:
38       }
39      \pdffield_appearance:nn {pdffield/radio/defaultdown/Off}
40       {
41         \normalsize
42         \draw_begin:
43         \draw_path_circle:nn {0pt,0pt}{0.5\normalbaselineskip}
44         \draw_path_use_clear:n { stroke }
45         \draw_path_circle:nn {0pt,0pt}{0.25\normalbaselineskip}
46         \draw_path_use_clear:n { fill }
47         \draw_end:
48       }
49     \cs_gset_eq:NN \__pdffield_radio_default_appearances: \prg_do_nothing:
50   }
```

(*End definition for* __pdffield_radio_default_appearances: *and others. These functions are documented on page* **??**.)

3.3 Creating the field

A field should be created if the name doesn't exist yet

```
51 \cs_new_protected:Npn \__pdffield_radio_field:n #1 %name
52   {
53     \pdf_object_if_exist:nF {__pdffield/field/__pdffield/radio/#1}
54       {
```

We need an object and a seq for the Opt array. The object is written at the end of the document.

```
55         \pdf_object_new:nn {__pdffield/field/__pdffield/radio-Opt/#1}{array}
56         \pdfdict_put:nnx { l__pdffield/field }{Opt} { \pdf_object_ref:n {__pdffield/field/__p
57         \seq_new:c { g__pdffield_radio_opt_#1_seq }
58         \hook_gput_code:nnn {shipout/lastpage}{pdffield/radio}
59           {
60             \pdf_object_write:nx
61               {__pdffield/field/__pdffield/radio-Opt/#1}
62               {\seq_use:cn {g__pdffield_radio_opt_#1_seq}{~}}
63           }
```

The default value is always the first in the array. If a default has been set, it is added to the left of the seq.

```
64         \tl_if_eq:NnF \l__pdffield_radio_default_tl{()}
65           {
66             \seq_gput_left:cV { g__pdffield_radio_opt_#1_seq }\l__pdffield_radio_default_tl
67           }
68         \pdfdict_put:nnx { l__pdffield/field }{V}  { /0 }
69         \pdfdict_put:nnx { l__pdffield/field }{DV} { /0 }
```

now we create the field and set it as parent for the following annotation.

```
70         \__pdffield_field:n { __pdffield/radio/#1 }
71       }
72     \keys_set:nn {pdffield}{parent=__pdffield/radio/#1}
73   }
74 \cs_generate_variant:Nn \__pdffield_radio_field:n {V}
```

(*End definition for* __pdffield_radio_field:n.)

3.4 Assembling the radio

The argument are key-val settings. At first we map the handlers. To setup the appearance we need the Opt array, so the appearance handler only stores the code.

```
75 \cs_new_protected:Npn \__pdffield_radio:n #1
76   {
77     \group_begin:
78     \cs_set_eq:NN\__pdffield_appearance_handler:nnn \__pdffield_radio_appearance_handler:nnn
79     \cs_set_eq:NN\__pdffield_value_handler:n       \__pdffield_radio_value_handler:n
80     \cs_set_eq:NN\__pdffield_default_handler:n     \__pdffield_radio_default_handler:n
```

Setting up the defaults.

```
81     \tl_set:Nn\l__pdffield_radio_appearance_code_tl{}
82     \keys_set:nn {pdffield}
83       {
84         fieldID=
```

```
85          ,name=radio
86          ,width  = \normalbaselineskip
87          ,height = \normalbaselineskip
88          ,__pdffield/preset/radiobutton
89          ,#1
90          ,unsetFf={Pushbutton}
91          ,setFf={Radio}
92          ,FT= Btn
93        }
```

If the fieldID has not been set explicitly, we use the name/T key

```
94      \tl_if_empty:NT\l__pdffield_fieldID_tl
95        {
96          \pdfdict_get:nnN {l__pdffield/field}{T}\l__pdffield_fieldID_tl
97          \tl_put_left:Nn \l__pdffield_fieldID_tl {__pdffield/radio/}
98        }
```

Now we build the field

```
99      \__pdffield_radio_field:V\l__pdffield_fieldID_tl
```

Retrieve the number for the label

```
100     \seq_if_in:cVF { g__pdffield_radio_opt_ \l__pdffield_fieldID_tl _seq }\l__pdffield_radio_
101       {
102         \seq_gput_right:cV { g__pdffield_radio_opt_ \l__pdffield_fieldID_tl _seq }\l__pdffiel
103       }
104     \int_zero:N \l__pdffield_radio_value_num_int
105     \exp_args:Nc
106     \seq_map_inline:Nn { g__pdffield_radio_opt_ \l__pdffield_fieldID_tl _seq }
107      {
108        \str_if_eq:nVTF { ##1 } \l__pdffield_radio_value_tl
109          {
110            \seq_map_break:
111          }
112          {
113            \int_incr:N \l__pdffield_radio_value_num_int
114          }
115      }
```

Annotations with the default value are set to on, the rest to off

```
116     \int_compare:nNnTF { \l__pdffield_radio_value_num_int } = 0
117       { \pdfannot_dict_put:nnx {widget}{AS}{/0} }
118       { \pdfannot_dict_put:nnx {widget}{AS}{/Off} }
```

Now we set the appearances, if no key has been use we take the default.

```
119     \tl_if_empty:NT\l__pdffield_radio_appearance_code_tl
120       {
121         \__pdffield_radio_default_appearances:
122         \keys_set:nn {pdffield}
123           {
124             appearance      = pdffield/radio/default,
125             down-appearance = pdffield/radio/defaultdown,
126           }
127       }
128     \l__pdffield_radio_appearance_code_tl
129     \__pdffield_annot:
130     \group_end:
131   }
```

(*End definition for* __pdffield_radio:n.)

3.5   Keys and handlers

Most keys are inherited simply the ones from the generic field and annot keys. We define a group key, as the name is better. The value key sets the export value. default the button which is checked on. At first the two handlers

__pdffield_radio_value_handler:n
__pdffield_radio_default_handler:n

```
132 \cs_new_protected:Npn \__pdffield_radio_value_handler:n #1
133   {
134     \pdf_string_from_unicode:nnN {utf8/string}{#1}\l__pdffield_radio_value_tl
135   }
136 \cs_new_protected:Npn \__pdffield_radio_default_handler:n #1
137   {
138     \pdf_string_from_unicode:nnN {utf8/string}{#1}\l__pdffield_radio_default_tl
139   }
```

(*End definition for* __pdffield_radio_value_handler:n *and* __pdffield_radio_default_handler:n.)

group

```
140 \keys_define:nn { pdffield }
141   {
142     group .meta:n = {T=#1}
143   }
```

(*End definition for* group. *This function is documented on page 4.*)

__pdffield_radio_appearance_handler:nnn   The handler for the appearances stores only the code as it must be executed rather late.

```
144 \cs_new_protected:Npn \__pdffield_radio_appearance_handler:nnn #1 #2 #3 %name, type, text
145   {
146     \tl_put_right:Nn \l__pdffield_radio_appearance_code_tl
147       {
148         \pdfxform_if_exist:nTF {  #1 / Yes }
149           {
150             \pdf_object_unnamed_write:nx
151               {dict}
152               {
153                 /\int_use:N \l__pdffield_radio_value_num_int
154                   \c_space_tl   \pdfxform_ref:n  { #1/Yes}
155                 /Off ~ \pdfxform_ref:n { #1/Off}
156               }
157             \pdfannot_dict_put:nnx {widget/AP}{#2}{\pdf_object_ref_last:}
158           }
159           {
160             \msg_error:nnnn{pdffield}{appearance-missing}{#1}{#3}
161           }
162       }
163   }
164
```

(*End definition for* __pdffield_radio_appearance_handler:nnn.)

3.6 User commands

\pdffield_radio:n

```
165 \cs_set_eq:NN \pdffield_radio:n \__pdffield_radio:n
166 ⟨/package⟩
```

(*End definition for* \pdffield_radio:n*. This function is documented on page* *3*.)

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.