# The l3pdffield-checkbox module
# Commands to create checkbox form fields
# LaTeX PDF management testphase bundle

The LaTeX Project*

Version 0.95u, released 2023-02-07

## 1 l3pdffield-checkbox Introduction

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

Please keep in mind

- Not every PDF viewer supports checkboxes.
- 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 Checkboxes

Click me: 

### 2.1 Commands

`\pdffield_checkbox:n` `\pdffield_checkbox:n{`⟨*key val list*⟩`}`

This creates a checkbox to check and uncheck. The list of allowed keys is described below. The ⟨*key val list*⟩ should at least set the name, without it the default name `checkbox` is used. Checkboxes with the same name belong to the same field and are checked and unchecked together. The default appearance is a quadratic frame with a `\texttimes` in it for the checked case. The default appearance is setup at the first use and will use the font family active at that time.

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

## 2.2 Keys

The new checkbox 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 checkbox specific behaviour, other keys have other defaults than with the basic commands. Additionally there are a small number of keys specific to a checkbox. For convenience a number of important keys are documented here too, even if they are already in the document from l3pdffield.

Disabled keys are

- `V`, `DV`, `AS`: use `checked` instead.
- `FT` is overwritten.
- For checkboxes only the field flags `ReadOnly`, `Required` and `NoExport` make sense. `Radio`, `Pushbotton` are set automatically automatically by the code as this is required for a checkbox.

---

`preset-checkbox` `preset-checkbox = {`⟨*key-val-list*⟩`}`

---

This allows to set default keys for a checkbox.

---

`name` `name = `⟨*partial name*⟩
`T` `T = `⟨*partial name*⟩

---

This sets the partial name of the field. 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 checkboxes with the same partial name are annotations with the same field as parent and are checked and unchecked together—this what is typically expected. 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.

---

`fieldID` `fieldID = `⟨*field ID*⟩

---

*For experts only!* This allows to give the checkbox field a specific ID. This is only useful in the context of a larger fieldset, if for example you want to use the same partial name for more than one field, 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`.

---

`altname` `altname = `⟨*string*⟩
`TU` `TU = `⟨*string*⟩

---

This is sets an alternative name for user interaction. This name can only be set at the first checkbox instance, when the field is initialized.

mappingname `mappingname = ⟨string⟩`
TM `TM = ⟨string⟩`

This is sets an alternative name for export. This name can only be set at the first checkbox instance, when the field is initialized.

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

These keys allow to set the dimensions of checkbox instance. 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/checkbox/default` for the normal appearance and shows a `\texttimes`. The other appearances are not set by default.

checked `checked = true|false`

This is a boolean key which allows to set if the checkbox should be initially checked or not. It sets the `/V` and `/DV` key of the field and the `/AS` key of the annotation instance. It is possible to use different values for different instances, if one wants to confuse the user.

value `value = Yes|Off`
default `default = Yes|Off`

With checkboxes this two key are simply an alternative input for `checked`.

## 2.3 Using with hyperref

The `\CheckBox` command from hyperref also prints a label, something that the command here doesn't do. A redefinition like the following should allow `\CheckBox` to use the commands of this module. Be aware that the behaviour will not be identical! Not every setting and key from hyperref has been copied.

```
\ExplSyntaxOn\makeatletter
\def\@CheckBox[#1]#2{\LayoutCheckField{#2}{\pdffield_checkbox:n {name=#2,#1}}}
\ExplSyntaxOff\makeatother
```

## 2.4 Some background

For some general background about fieldsets, fields and field annotations, please check l3pdffield. Here are only some remarks about the special case of checkboxes.

A checkbox consist of a field along with one or more field annotations. The annotations can appear on more than one page or locations and if one instance is checked all other instances follows and are checked too.

A checkbox has two different looks: checked and unchecked. The hyperref implementation uses symbolic names for the two states and adds some values with the /MK key and lets the PDF viewer create a look from them. But this doesn't work reliably and is one of the reasons why a reimplementation is needed. Also newer PDF versions deprecate the /NeedAppearances setting and require that such a look, an "appearance", is given as form XObjects. So the code forces the use of two appearances.

# 3 l3pdffield-checkbox Implementation

```
⟨*package⟩
⟨@@=pdffield⟩
```

## 3.1 Variables

There are no specific variables.

## 3.2 Messages

There are no specific messages.

## 3.3 Appearances

The default appearances are a quadratic frame with cross (`\texttimes`) if checked. User appearances should have two versions and follow the naming module/⟨*name*⟩/Yes and module/⟨*name*⟩/Off.

__pdffield_checkbox_default_appearances:
pdffield/checkbox/default/Yes
pdffield/checkbox/default/Off

This defines the standard appearance. It is setup at the first use of a checkbox, and will adapt to the font family in use then.

```
\cs_new_protected:Npn \__pdffield_checkbox_default_appearances:
  {
    \pdffield_appearance:nn {pdffield/checkbox/default/Yes}
      {
        \normalsize
        \fboxsep 0pt
        \framebox
          [ \dim_eval:n { \box_ht:N\strutbox+\box_dp:N\strutbox } ]
          { \texttimes \strut }
      }
    \pdffield_appearance:nn {pdffield/checkbox/default/Off}
      {
        \normalsize
        \fboxsep 0pt
        \framebox
          [ \dim_eval:n { \box_ht:N\strutbox+\box_dp:N\strutbox } ]
          { \phantom{\texttimes} \strut }
```

```
      }
    \cs_gset_eq:NN \__pdffield_checkbox_default_appearances: \prg_do_nothing:
  }
```

(*End definition for* `\__pdffield_checkbox_default_appearances:`, `pdffield/checkbox/default/Yes`, *and* `pdffield/checkbox/default/Off`. *These functions are documented on page* **??**.)

## 3.4 Creating the field

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

`\__pdffield_checkbox_field:n`

```
\cs_new_protected:Npn \__pdffield_checkbox_field:n #1 %name
  {
    \pdf_object_if_exist:nF {__pdffield/field/__pdffield/checkbox/#1}
      {
        \__pdffield_field:n { __pdffield/checkbox/#1 }
      }
    \keys_set:nn {pdffield}{parent=__pdffield/checkbox/#1}
  }
\cs_generate_variant:Nn \__pdffield_checkbox_field:n {V}
```

(*End definition for* `\__pdffield_checkbox_field:n`.)

## 3.5 Assembling the checkbox

`\__pdffield_checkbox:n`

```
\cs_new_protected:Npn \__pdffield_checkbox:n #1
  {
    \group_begin:
    \__pdffield_checkbox_default_appearances:
    \cs_set_eq:NN\__pdffield_appearance_handler:nnn \__pdffield_checkbox_appearance_handler:n
    \cs_set_eq:NN\__pdffield_value_handler:n   \__pdffield_checkbox_value_handler:n
    \cs_set_eq:NN\__pdffield_default_handler:n \__pdffield_checkbox_default_handler:n
```

Setting up the defaults.

```
    \keys_set:nn {pdffield}
      {
        fieldID=,
        name=checkbox,
        appearance = pdffield/checkbox/default,
        checked=false,
        width  = \normalbaselineskip,
        height = \normalbaselineskip,
      }
```

Value keys should be undefined.

```
    \__pdffield_key_disable:nnn{checkbox}{V}{checked}
    \__pdffield_key_disable:nnn{checkbox}{DV}{checked}
    \__pdffield_key_disable:nnn{checkbox}{AS}{checked}
    \keys_set:nn { pdffield }{__pdffield/preset/checkbox,#1}
    \keys_set:nn { pdffield }
      {
        ,unsetFf={Radio,Pushbutton}
        ,FT= Btn
```

```
      }
    \tl_if_empty:NT\l__pdffield_fieldID_tl
      {
        \pdfdict_get:nnN {l__pdffield/field}{T}\l__pdffield_fieldID_tl
        \tl_put_left:Nn \l__pdffield_fieldID_tl {__pdffield/checkbox/}
      }
    \__pdffield_checkbox_field:V\l__pdffield_fieldID_tl
    \__pdffield_annot:
    \group_end:
  }
```

(*End definition for* `\__pdffield_checkbox:n`.)

## 3.6 Keys

Most keys are inherited simply the ones from the generic field and annot keys. A key to decide if the box is initially checked or not. We stay in the same family so that we can build a style.

`checked` This is a key specific for checkbox, it sets both field and annotation keys.

```
\keys_define:nn { pdffield  }
 {
   ,checked .choice:
   ,checked / false .code:n =
     {
       \pdfdict_put:nnx { l__pdffield/field }{V} { /Off }
       \pdfdict_put:nnx { l__pdffield/field }{DV}{ /Off }
       \pdfannot_dict_put:nnn {widget}{AS}{ /Off }
     }
   ,checked / true .code:n =
     {
       \pdfdict_put:nnx { l__pdffield/field }{V} { /Yes }
       \pdfdict_put:nnx { l__pdffield/field }{DV}{ /Yes }
       \pdfannot_dict_put:nnn {widget}{AS}{ /Yes }
     }
   ,checked .default:n = {true}
   ,checked .groups:n  = {checkbox}
 }
\keys_define:nn { pdffield  }
 {
   ,__value .choice:
   ,__value / Off .code:n =
     {
       \pdfdict_put:nnx { l__pdffield/field }{V} { /Off }
       \pdfdict_put:nnx { l__pdffield/field }{DV}{ /Off }
       \pdfannot_dict_put:nnn {widget}{AS}{ /Off }
     }
   ,__value / Yes .code:n =
     {
       \pdfdict_put:nnx { l__pdffield/field }{V} { /Yes }
       \pdfdict_put:nnx { l__pdffield/field }{DV}{ /Yes }
       \pdfannot_dict_put:nnn {widget}{AS}{ /Yes }
     }
 }
```

(*End definition for* `checked`. *This function is documented on page 3.*)

`\__pdffield_checkbox_value_handler:n`
`\__pdffield_checkbox_default_handler:n`

`value` and `default` do the same as checked.

```
\cs_new_protected:Npn \__pdffield_checkbox_value_handler:n #1
  {
    \keys_set:nn{pdffield}{__value={#1}}
  }
\cs_new_protected:Npn \__pdffield_checkbox_default_handler:n #1
  {
    \keys_set:nn{pdffield}{__value={#1}}
  }
```

(*End definition for* `\__pdffield_checkbox_value_handler:n` *and* `\__pdffield_checkbox_default_handler:n`.)

`\__pdffield_checkbox_appearance_handler:nnn`

Appearances must create a dictionary, so we define a special handler. {⟨*name*⟩} is the xform name without the /Yes, /Off, {⟨*type*⟩} is N, R, or D, {⟨*text*⟩} is a word for the error message.

```
\cs_new_protected:Npn \__pdffield_checkbox_appearance_handler:nnn #1 #2 #3 %name, type, text
  {
    \pdfxform_if_exist:nTF {  #1/Yes }
      {
        \pdf_object_if_exist:nF {__pdffield/checkbox/AP/#1}
          {
            \pdf_object_new:n {__pdffield/checkbox/AP/#1}
            \pdf_object_write:nnx
              {__pdffield/checkbox/AP/#1} { dict }
              {
                /Yes ~ \pdfxform_ref:n { #1/Yes}
                /Off ~ \pdfxform_ref:n { #1/Off}
              }
          }
        \pdfannot_dict_put:nnx {widget/AP}{#2}{\pdf_object_ref:n{__pdffield/checkbox/AP/#1}}
      }
      {
         \msg_error:nnnn{pdffield}{appearance-missing}{#1}{#3}
      }
   }

```

(*End definition for* `\__pdffield_checkbox_appearance_handler:nnn`.)

## 3.7 user commands

`\pdffield_checkbox:n`

```
\cs_set_eq:NN \pdffield_checkbox:n \__pdffield_checkbox:n
⟨/package⟩
```

(*End definition for* `\pdffield_checkbox:n`. *This function is documented on page 1.*)

# 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.