The l3pdffield-action module
Commands to handle actions of form fields
LaTeX PDF management testphase bundle

The LaTeX Project*

Version 0.96n, released 2024-10-27

1 l3pdffield-action Introduction

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

A: ☐ B: ☐

Reset All | Reset B

Please keep in mind

- Not every PDF viewer supports form fields.
- Even if they support it, that doesn't mean that they support actions like Submit.
- The handling can depend on settings in the PDF viewer.
- Standards like pdf/A disable features of form fields too (as you typically can't change the PDF).

2 Actions

Actions must be handled in three places: In the action dictionary (`/A`) of the annotation(s), in the additional action dictionary of the field (`/AA`), and in the additional action dictionary (`/AA`) of the annotation(s). The distinction between the two `/AA` dictionaries is relevant as field actions can only be added when the field is initialized, so typically with the first field command for a specific name, while annotation actions can be added for example only to one specific instance of a checkbox or to pushbuttons belonging to the same field.

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

2.1 The additional action dictionaries (AA)

The actions in the `/AA` dictionaries are JavaScript (ECMAScript) actions. An example is the `calculate/AA/C` action, which is added with the `/C` key:

```
Netto~\pdffield_textfield:n{name=netto}~
\pdf_object_unnamed_write:ne{stream}
  {
    {}
    {
     var~f_netto = this.getField("netto");
     event.value = 1.19*f_netto.value;
    }
  }
\tl_set:Ne\l_tmpa_tl{\pdf_object_ref_last:}
Brutto~\pdffield_textfield:n{name=brutto,AA/C={\l_tmpa_tl}}
```

The main problem with scripts is to get the escaping right. While it is possible to pass the script as string, it is less error prone to include it as (file) stream and to reference the object number as done in the example.

2.2 The action dictionary (A)

The action dictionary can contain much more action types. The PDF reference lists 20 different types[1]: GoTo, GoToR, GoToE, GoToDp, Launch, Thread, URI, Sound, Movie, Hide, Named, SubmitForm, ResetForm, ImportData, SetOCGState, Rendition, Trans, GoTo3DView, JavaScript, RichMediaExecute.

And action dictionary typically looks like this:

```
/A << /Type /Action
      /S    ... % type name, e.g. /URI
      /Next ... % optional, next action
      ...   ... % more keys
   >>
```

The `/Next` key allows to chain actions. The value can be a reference to a single action, or an array of actions.

It depends on the action type which other keys should be used.

While almost all actions are usable in the annotations of a field, the three actions `SubmitForm`, `ResetForm` and `ImportData` are specific to fields and need explicit support. This support is the main task of this module.

2.2.1 ResetForm

The `ResetForm` action resets the fields of a form. The action is typically attached to a pushbutton.

In simple cases the action only need to contain the subtype entry and then resets everything (sadly with some side-effect on the appearance of the pushbutton, sigh).

[1] Sound and Movie are deprecated in PDF 2.0

```
\group_begin:
\pdfannot_dict_put:nnn{widget}{A}{<</Type/Action/S/ResetForm>>}
\pdffield_pushbutton:n{name=R,caption=Reset}
\group_end:
```

A `ResetForm` action knows two optional keys: `/Fields` allows to describe or restrict the fields which should be reset. Depending on the setting of the `/Flags` key the value of `/Fields` describes the fields to reset or the fields to leave unchanged.

The commands of this module allow to store actions under a name which can then be used in the key-val list of a field annotation.

A reset action can be defined and used like this

```
\pdffield_reset_new:nn
  {myreset}
  {
    fields= {A,B,C},
    exclude     % or include
  }
\pdffield_pushbutton:n{name=r,caption=Reset,reset=myreset}
```

`myreset` is the name the action can be referred to with the `reset` key.

It is possible to define and use a reset action before all fields it refers too have been created. The list of names can contain names that are actually not used by the form.

The reset action `all` is predefined.

2.2.2 ImportData

The `ImportData` action allows to import field data from an external file, for example a `.fdf` created with the submit action. The action is typically attached to a pushbutton, its only value is a file name.

2.2.3 SubmitForm

`SubmitForm` is the most challenging action. There are more keys in the dictionary, 13 flags, and there are four export options, which require each some flag settings. And the main argument is an url, which can contain special chars like # or % and so need special handling.

A submit action can be defined like this

```
\pdffield_submit_new:nnn
  {mysubmit}
  {
    export = fdf,
    fields= {A,B,C},
    exclude,     % or include
    setsubmitflags={...}
    ...
  }
  {URL}
\pdffield_pushbutton:n{name=s,caption={Submit~A,B,C},submit=mysubmit}
```

The export options are `html`, `pdf`, `fdf` and `xfdf`. The `html` export uses my default the `POST` method, the `GET` method can be forced by setting the `GetMethod` flag—at least according to the reference, I wasn't able yet to test it.

The following tabular describes the relation between these export options and the flags. `0` and `1` in the tabular means that the value is set by the code. `*` can be set by the user with the `setFlags` or `setsubmitflags` key.

Flag	html	pdf	fdf	xfdf
Include/Exclude	*	0	*	*
IncludeNoValueFields	*	0	*	*
ExportFormat	1	0	0	0
GetMethod	*	0	0	0
SubmitCoordinates	*	0	0	0
XFDF	0	0	0	1
IncludeAppendSaves	0	0	*	0
IncludeAnnotations	0	0	*	0
SubmitPDF	0	1	0	0
CanonicalFormat	*	0	*	*
ExclNonUserAnnots	0	0	*	0
ExclFKey	0	0	*	0
EmbedForm	0	0	*	0

2.3 Commands

`\pdffield_reset_new:nn` `\pdffield_reset_new:nn{⟨name⟩}{⟨key val list⟩}`

This defines an new reset action with the name ⟨*name*⟩. The keys are described below.

`\pdffield_submit_new:nnn` `\pdffield_submit_new:nn{⟨name⟩}{⟨key val list⟩}{⟨URL⟩}`

This defines an submit action with the name ⟨*name*⟩. The keys are described below. ⟨*URL*⟩ should be given verbatim. That means # and % should not be escaped. The URL should be correctly percent encoded, or the `urlencode` key should be used then the code will create the percent encoding.

⟨*URL*⟩ can be a mail address and should then start with `mailto:`. A subject can be attached with `?subject=...`. This normally works quite fine as it only needs a correctly working mail program. (But for some unknown reason my mail program don't like mail addresses with hyphens in them).

Alternatively it should be an URL of a script, which should return the correct answer (which one this is, is rather unclear).

2.4 Keys for the commands to create actions

`fields` `fields = {⟨comma list of fully qualified field names⟩}`

`fields` is a comma list of *fully qualified* field names, typically this can be the short name set with `name` in a field declaration, but in complex fieldsets it is needed to include the parents in the name too, separated by periods. If `exclude` is set, this list describes the fields that should be excluded from the reset or the submission. The default is `include`: the list describes the fields to reset or submit. Descendant of fields are reset or submitted too!

The key is relevant for reset and submit actions.

`include` `include`
`exclude` `exclude`

These keys are shorthands to unset/set the `Include/Exclude` flag and decide if the list of fields in the `fields` key are included or excluded. The default is `include`: the list describes the fields to reset or submit.

The keys are relevant for reset and submit actions.

`urlencode` `urlencode = true|false`

When this is true the code will percent encode the submission URL. This is needed if the URL contains for example non-ascii chars or spaces.

`setsubmitflags` `setsubmitflags = {⟨comma list of flags⟩}`
`setFlags` `setFlags = {⟨comma list of flags⟩}`
`unsetsubmitflags` `unsetsubmitflags = all | {⟨comma list of flags⟩}`
`unsetFlags` `unsetFlags = all | {⟨comma list of flags⟩}`

These keys accept a list of flag names and then sets or unsets them, the resulting value is then used with the `/Flags` key of a submit action. `Include/Exclude` is also used by a reset action andcan also be set with the `exclude` and `include` keys. Depending on the export type of the submit action some flags are set by the code. See the tabular above for details. The flag name can be given in PDF spelling (`IncludeNoValueFields`), in lowercase (`includenovaluefields`), and as number. `unsetFlags` and its alias `unsetsubmitflags` know the special value `all` which clears all the fields.

The list of flags are: `Include/Exclude`, `IncludeNoValueFields`, `ExportFormat`, `GetMethod`, `SubmitCoordinates`, `XFDF`, `IncludeAppendSaves`, `IncludeAnnotations`, `SubmitPDF`, `CanonicalFormat`, `ExclNonUserAnnots`, `ExclFKey`, `EmbedForm`

`next` `next = ⟨object reference⟩`

This allows to add a follow up action to the `/Next` key. The value is expanded, so can be given as `\pdf_object_ref:n{name}` and point to a suitable object.

2.5 Keys for fields

With the following keys actions can be attached to the annotation of a form field, for example a pushbutton.

reset reset = ⟨name⟩
submit submit = ⟨name⟩

This adds the reset or submit action ⟨*name*⟩ as action to the annotation. The action should be been defined first. The actions are mutually exclusive, only one action can be added. If more actions are needed use the `next` key. For reset the default value `all` exists and it is used if the key is used without value.

import import = ⟨*file name*⟩

This adds an import action to the annotation. ⟨*file name*⟩ should be for example the name of a `.fdf` file. If the name contains a path use a slash as separator. non-ascii chars should work, but ascii is safer. The action is mutually exclusive to submit and reset.

3 l3pdffield-action Implementation

```
⟨*package⟩
⟨@@=pdffield⟩
```

3.1 Messages

```
\msg_new:nnn {pdffield}{action-name-undefined}
  {
    The~'#1'~action~name~'#2'~is~not~defined~and~
    will~be~ignored.
  }

```

3.2 Variables

\l__pdffield_action_export_tl
\l__pdffield_action_Fields_seq
\l__pdffield_action_next_tl

We need to store the export, fields and next value into variables.

```
\tl_new:N  \l__pdffield_action_export_tl
\seq_new:N \l__pdffield_action_Fields_seq
\tl_new:N  \l__pdffield_action_next_tl
```

(*End of definition for* `\l__pdffield_action_export_tl`, `\l__pdffield_action_Fields_seq`, *and* `\l__pdffield_action_next_tl`.)

3.3 dictionaries

l__pdffield/ResetForm
l__pdffield/SubmitForm

We define two dictionaries to handle the reset and submit code.

```
\pdfdict_new:n   {l__pdffield/ResetForm}
\pdfdict_put:nnn {l__pdffield/ResetForm}{Type}{/Action}
\pdfdict_put:nnn {l__pdffield/ResetForm}{S}{/ResetForm}
\pdfdict_new:n   {l__pdffield/SubmitForm}
\pdfdict_put:nnn {l__pdffield/SubmitForm}{Type}{/Action}
\pdfdict_put:nnn {l__pdffield/SubmitForm}{S}{/SubmitForm}
```

(*End of definition for* `l__pdffield/ResetForm` *and* `l__pdffield/SubmitForm`.)

3.4 bitset for submit action

\l__pdffield_Flags_bitset

A bitset for the submit (and the one reset) flags:

```
\bitset_new:Nn \l__pdffield_Flags_bitset
  {
     Include/Exclude      = 1
    ,IncludeNoValueFields = 2
    ,ExportFormat         = 3
    ,GetMethod            = 4
    ,SubmitCoordinates    = 5
    ,XFDF                 = 6
    ,IncludeAppendSaves   = 7
    ,IncludeAnnotations   = 8
    ,SubmitPDF            = 9
    ,CanonicalFormat      = 10
    ,ExclNonUserAnnots    = 11
    ,ExclFKey             = 12
    ,EmbedForm            = 14
    ,include/exclude      = 1
    ,includenovaluefields = 2
    ,exportformat         = 3
    ,getmethod            = 4
    ,submitcoordinates    = 5
    ,xfdf                 = 6
    ,includeappendsaves   = 7
    ,includeannotations   = 8
    ,submitpdf            = 9
    ,canonicalformat      = 10
    ,exclnonuserannots    = 11
    ,exclfkey             = 12
    ,embedform            = 14
  }
```

(*End of definition for* `\l__pdffield_Flags_bitset`.)

__pdffield_action_flags_pdf:
__pdffield_action_flags_html:
__pdffield_action_flags_fdf:
__pdffield_action_flags_xfdf:

The following commands sets the forced flags for the export options.

```
\cs_new_protected:Npn \__pdffield_action_flags_pdf:
  {
    \bitset_clear:N \l__pdffield_Flags_bitset
    \bitset_set_true:Nn \l__pdffield_Flags_bitset { SubmitPDF }
  }

\cs_new_protected:Npn \__pdffield_action_flags_html:
  {
    \bitset_set_true:Nn \l__pdffield_Flags_bitset { ExportFormat }
    \bitset_set_false:Nn \l__pdffield_Flags_bitset { XFDF }
    \bitset_set_false:Nn \l__pdffield_Flags_bitset { IncludeAppendSaves }
    \bitset_set_false:Nn \l__pdffield_Flags_bitset { IncludeAnnotations }
    \bitset_set_false:Nn \l__pdffield_Flags_bitset { SubmitPDF }
    \bitset_set_false:Nn \l__pdffield_Flags_bitset { ExclNonUserAnnots }
    \bitset_set_false:Nn \l__pdffield_Flags_bitset { ExclFKey }
    \bitset_set_false:Nn \l__pdffield_Flags_bitset { EmbedForm }
  }

```

```
\cs_new_protected:Npn \__pdffield_action_flags_fdf:
  {
    \bitset_set_false:Nn \l__pdffield_Flags_bitset { ExportFormat }
    \bitset_set_false:Nn \l__pdffield_Flags_bitset { GetMethod }
    \bitset_set_false:Nn \l__pdffield_Flags_bitset { SubmitCoordinates }
    \bitset_set_false:Nn \l__pdffield_Flags_bitset { XFDF }
    \bitset_set_false:Nn \l__pdffield_Flags_bitset { SubmitPDF }
  }

\cs_new_protected:Npn \__pdffield_action_flags_xfdf:
  {
    \bitset_set_false:Nn \l__pdffield_Flags_bitset { ExportFormat }
    \bitset_set_false:Nn \l__pdffield_Flags_bitset { GetMethod }
    \bitset_set_false:Nn \l__pdffield_Flags_bitset { SubmitCoordinates }
    \bitset_set_true:Nn \l__pdffield_Flags_bitset { XFDF }
    \bitset_set_false:Nn \l__pdffield_Flags_bitset { IncludeAppendSaves }
    \bitset_set_false:Nn \l__pdffield_Flags_bitset { IncludeAnnotations }
    \bitset_set_false:Nn \l__pdffield_Flags_bitset { SubmitPDF }
    \bitset_set_false:Nn \l__pdffield_Flags_bitset { ExclNonUserAnnots }
    \bitset_set_false:Nn \l__pdffield_Flags_bitset { ExclFKey }
    \bitset_set_false:Nn \l__pdffield_Flags_bitset { EmbedForm }
  }
```

(*End of definition for* `\__pdffield_action_flags_pdf:` *and others.*)

3.5 Keys

reset
submit
import

These are the three additional keys for the field annotations

```
\keys_define:nn { pdffield }
  {
    reset .code:n =
      {
        \cs_if_exist:cTF { __pdffield_action_reset_#1: }
          {
            \use:c { __pdffield_action_reset_#1: }
            \pdfannot_dict_put:nne{widget}
             {A}
             {\tl_use:c { c__pdffield_action_reset_#1_tl } }
          }
          {
            \msg_warning:nnnn{pdffield}{action-name-undefined}{reset}{#1}
          }
      }
   ,reset .default:n = all
  }

\keys_define:nn { pdffield }
  {
    submit .code:n =
      {
        \cs_if_exist:cTF { __pdffield_action_submit_#1: }
          {
            \use:c { __pdffield_action_submit_#1: }
            \pdfannot_dict_put:nne{widget}
```

```
            {A}
            {\tl_use:c { c__pdffield_action_submit_#1_tl } }
          }
          {
            \msg_warning:nnnn{pdffield}{action-name-undefined}{submit}{#1}
          }
      }
  }

\keys_define:nn { pdffield }
  {
    import .code:n =
      {
        \pdf_string_from_unicode:nnN {utf8/string}{#1}\l__pdffield_tmpa_str
        \pdf_object_unnamed_write:ne {dict}{/Type/Action/S/ImportData/F\l__pdffield_tmpa_str}
        \pdfannot_dict_put:nne{widget}
          {A}
          {\pdf_object_ref_last: }
      }
  }

```

(*End of definition for* `reset`, `submit`, *and* `import`. *These functions are documented on page* 6.)

fields
exclude
include
export
charset
urlencode
next

These are keys used when defining the actions.

```
\keys_define:nn { pdffield / action }
  {
    fields .code:n  =
      {
        \clist_map_inline:nn {#1}
          {
            \pdf_string_from_unicode:nnN {utf8/string}{##1}\l__pdffield_tmpa_str
            \seq_put_right:NV\l__pdffield_action_Fields_seq \l__pdffield_tmpa_str
          }
      }
    ,exclude .code:n = { \bitset_set_true:Nn \l__pdffield_Flags_bitset {Include/Exclude }}
    ,include .code:n = { \bitset_set_false:Nn \l__pdffield_Flags_bitset {Include/Exclude }}
    ,export .choices:nn = {pdf,fdf,html,xfdf}
      {
        \tl_set:Nn \l__pdffield_action_export_tl {#1}
      }
    ,export .initial:n = {html}
    ,charset .choices:nn =
      {utf-8, utf-16, Shift-JIS, BigFive, GBK, UHC}
      { \pdfdict_put:nnn { l__pdffield/SubmitForm }{#1} }
    ,urlencode .bool_set:N = \l__pdffield_url_encode_bool
    ,next .tl_set:N = \l__pdffield_action_next_tl
  }
```

(*End of definition for* `fields` *and others. These functions are documented on page* 5.)

setFlags
setsubmitflags
unsetFlags
unsetsubmitflags

```
\keys_define:nn { pdffield / action }
  {
    ,setFlags .code:n =
```

```
      {
        \clist_map_inline:nn {#1}
          {
            \bitset_set_true:Nn \l__pdffield_Flags_bitset {##1}
          }
      }
    ,setsubmitflags .meta:n = {setFlags={#1}}
    ,unsetFlags .multichoice:
    ,unsetFlags / all .code:n = { \bitset_clear:N \l__pdffield_Flags_bitset}
    ,unsetFlags / unknown .code:n =
      {
        \bitset_set_false:Nn \l__pdffield_Flags_bitset {#1}
      }
    ,unsetsubmitflags .meta:n = {unsetFlags={#1}}
  }
```

(*End of definition for* `setFlags` *and others. These functions are documented on page* 5.)

3.6 New reset action

`\__pdffield_action_reset_new:nn`

The command defines a command which will setup the dictionary at first use, and then store the reference in a constant.

```
\cs_new_protected:Npn \__pdffield_action_reset_new:nn #1 #2 %#1 name, #2 keyval
  {
    \cs_new_protected:cpn {__pdffield_action_reset_#1:}
      {
        \group_begin:
        \seq_clear:N \l__pdffield_action_Fields_seq
        \keys_set:nn { pdffield / action }{ #2 }
        \pdf_object_unnamed_write:ne
          { array }
          { \seq_use:Nn \l__pdffield_action_Fields_seq {~} }
        \tl_if_empty:NF \l__pdffield_action_next_tl
          {
            \pdfdict_put:nne {l__pdffield/ResetForm}{Next}{\l__pdffield_action_next_tl}
          }
        \pdfdict_put:nne
          { l__pdffield/ResetForm }
          { Fields }
          { \pdf_object_ref_last: }
        \pdfdict_put:nne
          { l__pdffield/ResetForm }
          { Flags }
          { \bitset_item:Nn\l__pdffield_Flags_bitset{Include/Exclude} }
        \pdf_object_unnamed_write:ne
          { dict }
          { \pdfdict_use:n{l__pdffield/ResetForm} }
        \tl_const:ce { c__pdffield_action_reset_#1_tl } { \pdf_object_ref_last: }
        \cs_gset_eq:cN {__pdffield_action_reset_#1:} \prg_do_nothing:
        \group_end:
      }
  }

\__pdffield_action_reset_new:nn  {all}{fields={},exclude}
```

(*End of definition for* `\__pdffield_action_reset_new:nn`.)

3.7 New submit action

`\__pdffield_action_submit_new:nn`

Quite similar to the reset command, only that we have to use an auxiliary to grab the URL with suitable catcodes.

```
\cs_new_protected:Npn \__pdffield_action_submit_new:nn #1 #2 %#1 name, #2 keyval
  {
    \group_begin:
    \char_set_catcode_other:N \%
    \char_set_catcode_other:N \#
    \__pdffield_action_submit_new:nnn {#1}{#2}
  }
\cs_new_protected:Npn \__pdffield_action_submit_new:nnn #1 #2 #3 %#1 name, #2 keyval, #3 url
  {
    \group_end:
    \cs_new_protected:cpn {__pdffield_action_submit_#1:}
      {
        \group_begin:
        \seq_clear:N    \l__pdffield_action_Fields_seq
        \bitset_clear:N \l__pdffield_Flags_bitset
        \keys_set:nn {pdffield/action}{#2}
        \use:c{ __pdffield_action_flags_\l__pdffield_action_export_tl :}
        \pdfdict_put:nne
          { l__pdffield/SubmitForm }
          { Flags }
          { \bitset_to_arabic:N \l__pdffield_Flags_bitset }
        \tl_if_empty:NF \l__pdffield_action_next_tl
          {
            \pdfdict_put:nne {l__pdffield/SubmitForm}{Next}{\l__pdffield_action_next_tl}
          }
        \bool_if:NTF \l__pdffield_url_encode_bool
          { \pdf_string_from_unicode:nnN { utf8/URI }   {#3}\l__pdffield_tmpa_str }
          { \pdf_string_from_unicode:nnN { utf8/string }{#3}\l__pdffield_tmpa_str }
        \pdf_object_unnamed_write:ne {dict}
          {
            /FS/URL
            /F \l__pdffield_tmpa_str
          }
        \pdfdict_put:nne
          { l__pdffield/SubmitForm }
          { F }
          { \pdf_object_ref_last: }
        \pdf_object_unnamed_write:ne
          { dict }
          { \pdfdict_use:n{ l__pdffield/SubmitForm } }
        \tl_const:ce { c__pdffield_action_submit_#1_tl } { \pdf_object_ref_last: }
        \cs_gset_eq:cN { __pdffield_action_submit_#1: } \prg_do_nothing:
        \group_end:
      }
  }

```

(*End of definition for* `\__pdffield_action_submit_new:nn`.)

3.8 User commands

\pdffield_reset_new:nn
\pdffield_submit_new:nnn

```
\cs_set_eq:NN \pdffield_reset_new:nn   \__pdffield_action_reset_new:nn
\cs_set_eq:NN \pdffield_submit_new:nnn \__pdffield_action_submit_new:nn
⟨/package⟩
```

(*End of definition for* `\pdffield_reset_new:nn` *and* `\pdffield_submit_new:nnn`. *These functions are documented on page 4.*)

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.