

1 Package header

```
1 <*package>
2 <@@=yoin>
```

Necessary packages: First, \LaTeX 3 stuff.

```
3 \RequirePackage{expl3,l3keys2e,l3regex,xparse}
```

From zref bundle, for computing the total number of pages of an article.

```
4 \RequirePackage{zref-totpages}
```

We need the absolute paths. This also means we need `-recorder` option to `pdflatex`.

```
5 \RequirePackage[abspath]{currfile}
```

For including PDF files.

```
6 \RequirePackage{pdfpages}
```

Package header.

```
7 \ProvidesExplPackage{yoin}{2016/02/28}{v0.0.1}{Joining articles into issues}
```

2 General macros

Macros not necessarily related to the package; moreorless an addition to \LaTeX 3.

`\yoin_seq_gappend_clist:Nn` Globally append `clist` #2 to `seq` #1.

```
8 \seq_new:N \l__yoin_seq_tmpa_seq
9 \cs_new_protected:Nn \yoin_seq_gappend_clist:Nn {
10   \seq_set_from_clist:Nn \l__yoin_seq_tmpa_seq { #2 }
11   \seq_gconcat:NNN #1 #1 \l__yoin_seq_tmpa_seq
12 }
```

`\yoin_keys_set_from_file:nn` Read a file #2 containing a key–value list and set the keys for #1. No checks are done here, nothing like comments could be used, the keys should be separated by a comma (and spaces of course as needed).

```
13 \tl_new:N \l__yoin_keys_tmpa_tl
14 \cs_new_protected:Nn \yoin_keys_set_from_file:nn {
15   \tl_set_from_file:Nnn \l__yoin_keys_tmpa_tl { } { #2 }
16   \keys_set:nV { #1 } \l__yoin_keys_tmpa_tl
17 }
18 \cs_generate_variant:Nn \keys_set:nn { nV }
```

`\yoin_keyval_parse_from_file:nn` Read a file #2 containing a key-value list and set the keys for #1. No checks are done here, nothing like comments could be used, the keys should be separated by a comma (and spaces of course as needed).

```
19 \cs_new_protected:Nn \yoin_keyval_parse_from_file:NNn {
20   \tl_set_from_file:Nnn \l__yoin_keys_tmpa_tl { } { #3 }
21   \keyval_parse:NNV #1 #2 \l__yoin_keys_tmpa_tl
22 }
23 \cs_generate_variant:Nn \keyval_parse:NNn { NNV }
```

3 Key-value interface for the package setup

First, we define the variables to store the keys.

`\g_yoin_subprocess_bool` Booleans:

```
\g_yoin_article_bool 24 \bool_new:N \g_yoin_subprocess_bool
\g_yoin_dryrun_bool 25 \bool_new:N \g_yoin_article_bool
\g_yoin_onlyflags_bool 26 \bool_new:N \g_yoin_dryrun_bool
\g_yoin_onlytags_bool 27 \bool_new:N \g_yoin_onlyflags_bool
28 \bool_new:N \g_yoin_onlytags_bool
```

`\g_yoin_flags_seq` Sequences for flags, tags and their filtering:

```
\g_yoin_tags_seq 29 \seq_new:N \g_yoin_flags_seq
\g_yoin_onlyflags_seq 30 \seq_new:N \g_yoin_tags_seq
\g_yoin_onlytags_seq 31 \seq_new:N \g_yoin_onlyflags_seq
32 \seq_new:N \g_yoin_onlytags_seq
```

`msg: unknown-flag` Two messages, for unknown flags and unknown tags.

```
msg: unknown-tag 33 \msg_new:nnnn { yoin } { unknown-flag }
34 { The ~ flag ~ '#1' ~ is ~ unknown ~ to ~ 'yoin'. }
35 { You ~ either ~ misspelled ~ it ~ or ~forgot ~ to ~ declare ~ it. }

36 \msg_new:nnnn { yoin } { unknown-tag }
37 { The ~ tag ~ '#1' ~ is ~ unknown ~ to ~ 'yoin'. }
38 { You ~ either ~ misspelled ~ it ~ or ~forgot ~ to ~ declare ~ it. }
```

`\yoin_if_tag_defined:n` Conditionals for checking whether a tag/flag was defined.

```
\yoin_if_flag_defined:n 39 \prg_new_protected_conditional:Nnn \yoin_if_tag_defined:n { T, F, TF } {
40   \seq_if_in:NnTF \g_yoin_tags_seq { #1 } { \prg_return_true: } { \prg_return_false: }
41 }
```

```

42 \prg_new_protected_conditional:Nnn \yoin_if_flag_defined:n { T, F, TF } {
43   \seq_if_in:NnTF \g_yoin_flags_seq { #1 } { \prg_return_true: } { \prg_return_false: }
44 }

```

_yoin_error_if_tag_undefined:n Check whether a tag/flag is defined, if not, issue an error.

```

\_yoin_error_if_flag_undefined:n 45 \cs_new_protected:Nn \_yoin_error_if_tag_undefined:n {
46   \yoin_if_tag_defined:nF { #1 } { \msg_error:nnn { yoin } { unknown-tag } { #1 } }
47 }

48 \cs_new_protected:Nn \_yoin_error_if_flag_undefined:n {
49   \yoin_if_flag_defined:nF { #1 } { \msg_error:nnn { yoin } { unknown-flag } { #1 } }
50 }

```

yoin / general The keys themselves:

```

51 \keys_define:nn { yoin / general } {

```

Booleans:

```

52   dryrun .bool_gset:N = \g_yoin_dryrun_bool,
53   dryrun .initial:n = { false },

54   article .bool_gset:N = \g_yoin_article_bool,
55   article .initial:n = { false },

56   subprocess .bool_gset:N = \g_yoin_subprocess_bool,
57   subprocess .initial:n = { false },

```

Keys whose clist values are appended to a seq:

```

58   defineflags .code:n = \yoin_seq_gappend_clist:Nn \g_yoin_flags_seq { #1 },
59   definetags .code:n = \yoin_seq_gappend_clist:Nn \g_yoin_tags_seq { #1 },

```

A clist key is stored in a seq, also, a corresponding bool is set true. (The point is, if onlyflags/onlytags is not ever set up, we want to know it since we treat it as if we use all flags/tags.)

```

60   onlyflags .code:n =
61     \seq_gset_from_clist:Nn \g_yoin_onlyflags_seq { #1 }
62     \bool_gset_true:N \g_yoin_onlyflags_bool
63   ,

64   onlytags .code:n =
65     \seq_gset_from_clist:Nn \g_yoin_onlytags_seq { #1 }
66     \bool_gset_true:N \g_yoin_onlytags_bool
67   ,

```

```
68 }
```

`\ProcessKeysPackageOptions` Process key options given to the package. We *do not want to process any options given to the class*. Whence `\ProcessKeysPackageOptions` and not `\ProcessKeysOptions`.

```
69 \ProcessKeysPackageOptions { yoin / general }
```

`\yoin_setup:n` Allow keys to be set later. We define both a \LaTeX 3 interface and an xparse UI wrapper.

```
\yoinSetup 70 \cs_new_protected:Nn \yoin_setup:n {
71   \keys_set:nn { yoin / general } { #1 }
72 }
```

```
73 \NewDocumentCommand \yoinSetup { R[]{} } {
74   \yoin_setup:n { #1 }
75 }
```

4 `\yoinAdd` macro — adding articles to the issue

The key–value interface. In this case, we basically only store the keys for each article in a prop. First, an interface for setting the keys for the articles. `\yoin_yoinadd_prop:n` returns the name of the prop for the given article; *no check for existence is done at this place*.

`\g_yoin_yoinadd_seq` A sequence for storing the list of the existing articles.

```
76 \seq_new:N \g_yoin_yoinadd_seq
```

`\yoin_yoinadd_prop:n` `\yoin_yoinadd_prop:V` `\yoin_yoinadd_prop:nn` `\yoin_yoinadd_prop_item:nn` `\yoin_yoinadd_prop_item:Vn` `\yoin_yoinadd_prop:n` returns the name of the prop for the given article; *no check for existence is done at this place*. `\yoin_yoinadd_prop:nn` returns property #2 of article #1, or `\q_no_value` if the property is not set.

```
77 \cs_new:Nn \yoin_yoinadd_prop:n {
78   g__yoin_article_#1_prop
79 }
80 \cs_generate_variant:Nn \yoin_yoinadd_prop:n { V }

81 \cs_new:Nn \yoin_yoinadd_prop_item:nn {
82   \prop_item:cn { \yoin_yoinadd_prop:n { #1 } } { #2 }
83 }
84 \cs_generate_variant:Nn \yoin_yoinadd_prop_item:nn { V }
```

For processing `\yoinAdd`, we first set up a `\tl` to contain the name of the article, then create the prop, and finally use `\keys` to fill in the prop. Note that if an article is added twice, an error is issued, if the error is ignored, the article is not added but the properties are set.

`\l_yoin_yoinadd_currentarticle_tl` A `tl` that stores the name of the article that is being processed by `\yoinAdd`.

```
85 \tl_new:N \l_yoin_yoinadd_currentarticle_tl
```

`__yoin_yoinadd_storekey:nn` Internal macro for storing a key in the prop. The one-parameter variant sets the value of the key empty.

```
\__yoin_yoinadd_storekey:n 86 \cs_new_protected:Nn \__yoin_yoinadd_storekey:nn {
87   \prop_gput:cnn { \yoin_yoinadd_prop:V \l_yoin_yoinadd_currentarticle_tl } { #1 } { #2 }
88 }
89 \cs_new_protected:Nn \__yoin_yoinadd_storekey:n {
90   \prop_gput:cnn { \yoin_yoinadd_prop:V \l_yoin_yoinadd_currentarticle_tl } { #1 } { }
91 }
```

`\yoin_yoinadd:nn` The macro `\yoinAdd` itself. We first set `\l_@@_yoinadd_currentarticle_tl`, then check whether the same article has not been processed before (issuing an error in that case and finishing). Then, the article is added in `\g_yoin_yoinadd_seq`, the prop created, the article's name added in the prop with key `article` and the keys are set. If the article has a `.yoin` file in its sub-directory, the key-values in it is added to the prop. If the file does not exist, it means things are wrong (the article should first be set up, before being added to its issue by `\yoinAdd`).

`\yoinAdd`

```
92 \cs_new_protected:Nn \yoin_yoinadd:nn {
93   \tl_set:Nn \l_yoin_yoinadd_currentarticle_tl { #1 }
94   \seq_if_in:NnTF \g_yoin_yoinadd_seq { #1 } {
95     \msg_error:nnn { yoin } { yoinadd-duplicatearticle } { #1 }
96   } {
97     \seq_gput_right:Nn \g_yoin_yoinadd_seq { #1 }
98     \prop_new:c { \yoin_yoinadd_prop:n { #1 } }
99     \__yoin_yoinadd_storekey:nn { article } { #1 }
100    \keys_set:nn { yoin / yoinadd } { #2 }
101    \file_if_exist:nTF { #1 / #1 .yoin } {
102      \yoin_keyval_parse_from_file:NNn
103        \__yoin_yoinadd_storekey:n
104        \__yoin_yoinadd_storekey:nn
105        { #1 / #1 .yoin }
106    } {
107      \msg_error:nnn { yoin } { yoinadd-dotyoinmissing } { #1 }
108    }
109  }
110 }

111 \NewDocumentCommand \yoinAdd { m O{} } {
112   \yoin_yoinadd:nn { #1 } { #2 }
113 }
```

`msg: yoinadd-duplicatearticle` The error messages: for adding a duplicate article and for adding an article with no #1/#1.yoin file.

```
msg: yoinadd-dotyoinmissing 114 \msg_new:nnn { yoin } { yoinadd-duplicatearticle }
                             115   { The ~ article ~ "#1" ~ has ~ been ~ already ~ processed ~ by ~ \token_to_str:N \yoinAdd ~.}
                             116 \msg_new:nnn { yoin } { yoinadd-dotyoinmissing }
                             117   { The ~ article ~ "#1" ~ has ~ no ~ file "#1/#1.yoin" ~ and ~ was ~ not ~ properly ~ set ~ up.}
```

`yoin / yoinadd` The keys here are pretty simple; each defined key just stores its value in the prop. We recall that #1 is the key and ##1 is the value.

```
118 \clist_map_inline:nn { forceopenany, forceopenright, ignore } {
119   \keys_define:nn { yoin / yoinadd } {
120     #1 .code:n = \__yoin_yoinadd_storekey:nn { #1 } { ##1 },
121   }
122 }
```

However, for the tag key, we additionally check that the tag exists.

```
123 \keys_define:nn { yoin / yoinadd } {
124   tag .code:n =
125     \__yoin_error_if_tag_undefined:n { #1 }
126     \__yoin_yoinadd_storekey:nn { tag } { #1 }
127   ,
128 }
```

5 Environment yoinshell

`\l_yoin_yoinshell_ignore_bool` A boolean for storing the ignore key's value.

`yoin / yoinshell` Key-value interface to yoinshell.

```
129 \keys_define:nn { yoin / yoinshell } {
```

If flag is set and onlyflags is set but the flag is not amongst them, the whole yoinshell is ignored (by setting the ignore key).

```
130   flag .code:n =
131     \__yoin_error_if_flag_undefined:n { #1 }
132     \bool_if:NT \g_yoin_onlyflags_bool {
133       \seq_if_in:NnF \g_yoin_onlyflags_seq { #1 } {
134         \keys_set:nn { yoin / yoinshell } {
135           ignore = true
136         }
137       }
138     }
139   ,
```

The ignore key sets a boolean

```
140   ignore .bool_set:N = \l_yoin_yoinshell_ignore_bool,
141   ignore .initial:n = { false },
142 }
```

`shellesc.sty` A reasonable shell escape that should work in both pdf_latex and lua_latex in T_EX Live 2016.

```
\ShellEscape 143 \file_if_exist:nTF { shellesc.sty } {
\__yoin_yoinshell_shellescape:n 144   \RequirePackage { shellesc }
145 } {
146   \def \ShellEscape #1 { \immediate \write 18 { #1 } }
147 }
148 \cs_new_protected:Nn \__yoin_yoinshell_shellescape:n {
149   \ShellEscape { #1 }
150 }
```

`__yoin_yoinshell_begin:n` Environment yoinshell (one key-value argument). We perform some local definitions that should stay local, so we put everything in a group. The keys are set. Then we define the macros — “shell commands”. If ignore is set, these macros are declared to do nothing, otherwise they are simply wrappers to the L^AT_EX3 counterparts.

```
151 \cs_new_protected:Nn \__yoin_yoinshell_begin:n {
152   \group_begin:
153   \keys_set:nn { yoin / yoinshell } { #1 }
154   \bool_if:NTF \l_yoin_yoinshell_ignore_bool {
155     \DeclareDocumentCommand \RunForEach { 0{} m } { }
156     \DeclareDocumentCommand \Run { 0{} m } { }
157   } {
158     \DeclareDocumentCommand \RunForEach { 0{} m } { \yoin_yoinshell_runforeach:nn { ##1 } { ##2 } }
159     \DeclareDocumentCommand \Run { 0{} m } { \yoin_yoinshell_run:nn { ##1 } { ##2 } }
160   }
161 }

162 \cs_new_protected:Nn \__yoin_yoinshell_end: {
163   \group_end:
164 }

165 \NewDocumentEnvironment { yoinshell } { 0{} } {
166   \__yoin_yoinshell_begin:n { #1 }
167 } {
168   \__yoin_yoinshell_end:
169 }
```

The yoinshell command \RunForEach.

\l__yoin_yoinshell_runforarticle_tag_tl First, two tls that will store tags: One for the tag of the article, one that could be passed to \RunForEach that is initially set to
\l__yoin_yoinshell_runforeach_onlytag_tl \q_no_value.

```
170 \tl_new:N \l__yoin_yoinshell_runforarticle_tag_tl
171 \tl_new:N \l__yoin_yoinshell_runforeach_onlytag_tl
172 \tl_set:Nn \l__yoin_yoinshell_runforeach_onlytag_tl { \q_no_value }
```

yoin / runforeach So far, the only key-val passable to \RunForEach is onlytag, which tests for the tag to be declared and passes it to \l__yoin_yoinshell_runforeach_onlytag_tl.

```
173 \keys_define:nn { yoin / runforeach } {
174   onlytag .code:n =
175     \__yoin_error_if_tag_undefined:n { #1 }
176     \tl_set:Nn \l__yoin_yoinshell_runforeach_onlytag_tl { #1 }
177   ,
178 }
```

__yoin_yoinshell_runforarticle_keyfromprop:nnN This macro lets #3 to the value of property #2 of article #1. It makes it an empty definition if the property is unset.

```
179 \tl_new:N \l__yoin_yoinshell_runforarticle_tmpa_tl
180 \cs_new_protected:Nn \__yoin_yoinshell_runforarticle_keyfromprop:nnN {
181   \prop_get:cnN { \yoin_yoinadd_prop:n { #1 } } { #2 } \l__yoin_yoinshell_runforarticle_tmpa_tl
182   \quark_if_no_value:NTF \l__yoin_yoinshell_runforarticle_tmpa_tl {
183     \def #3 {}
184   } {
185     \let #3 \l__yoin_yoinshell_runforarticle_tmpa_tl
186   }
187 }
```

__yoin_yoinshell_runforeach:nn \RunForEach itself just sets the keys (in a group to make things local) and then calls __yoin_yoinshell_runforarticle:nn on each article.

```
188 \cs_new_protected:Nn \yoin_yoinshell_runforeach:nn {
189   \group_begin:
190   \keys_set:nn { yoin / runforeach } { #1 }
191   \seq_map_inline:Nn \g_yoin_yoinadd_seq { \__yoin_yoinshell_runforarticle:nn { ##1 } { #2 } }
192   \group_end:
193 }
```


`__yoin_yoinshell_runforarticle:nn` If the tag passed to `onlytag` of `\RunForEach` is identical to the tag of the article or if any of them is not set, we do what should be done, otherwise nothing is done (the tags do not match). We only extract the prop to publically available macros like `\Article`, `\Jobname` etc. (in a group to make this local), and then run the command in shell escape.

```

194 \cs_new_protected:Nn \__yoin_yoinshell_runforarticle:nn {
195   \prop_get:cnN { \yoin_yoinadd_prop:n { #1 } } { tag } \l__yoin_yoinshell_runforarticle_tag_tl
196   \bool_if:nT {
197     \quark_if_no_value_p:N \l__yoin_yoinshell_runforarticle_tag_tl
198     ||
199     \quark_if_no_value_p:N \l__yoin_yoinshell_runforeach_onlytag_tl
200     ||
201     \tl_if_eq_p:NN \l__yoin_yoinshell_runforeach_onlytag_tl \l__yoin_yoinshell_runforarticle_tag_tl
202   }{
203     \group_begin:
204     \__yoin_yoinshell_runforarticle_keyfromprop:nnN { #1 } { article } \Article
205     \__yoin_yoinshell_runforarticle_keyfromprop:nnN { #1 } { jobname } \Jobname
206     \__yoin_yoinshell_shellescape:n { #2 }
207     \group_end:
208   }
209 }

```

6 Article setting stuff (undocumented)

Information to be stored in an auxiliary file.

```

210 \cs_new_protected:Nn \__yoin_article_write:n {
211   \immediate \write \@auxout { \token_to_str:N \@writefile { yoin } { #1 } }
212 }
213
214 \cs_new_protected:Nn \__yoin_article_write_keyval:nn {
215   \__yoin_article_write:n { #1 ~ = ~ #2 , }
216 }
217 \cs_generate_variant:Nn \__yoin_article_write_keyval:nn { nx }
218
219 \cs_new_protected:Nn \yoin_article_write_meta:nn {
220   \__yoin_article_write_keyval:nn { meta-#1 } { #2 }
221 }
222
223 \cs_new_protected:Nn \yoin_article_writekeys: {
224   \__yoin_article_write_keyval:nx { jobname } { \jobname }

```

```

225 \__yoin_article_write_keyval:nx { totpages } { \ztotpages }
226 \__yoin_article_write_keyval:nx { currdir } { \l_yoin_article_currdir_tl }
227 \__yoin_article_write_keyval:nx { firstpage } { \int_use:N \l_yoin_article_firstpage_int }
228 }
229
230 \prop_new:N \l__yoin_article_readkeys_prop
231
232 \cs_new_protected:Nn \yoin_article_set_readkey:nn {
233   \prop_put:Nnn \l__yoin_article_readkeys_prop { #1 } { #2 }
234 }
235
236 \int_new:N \l_yoin_article_firstpage_int
237 \int_set:Nn \l_yoin_article_firstpage_int { 1 }
238
239 \keys_define:nn { yoin / toarticle } {
240   firstpage .code:n =
241     \int_set:Nn \l_yoin_article_firstpage_int { #1 }
242     \yoin_article_set_readkey:nn { firstpage } { #1 }
243   ,
244
245   parent .code:n =
246     \file_if_exist:nT { ../ #1 .yoin } {
247       \yoin_keys_set_from_file:nn { yoin / toarticle } { ../ #1 .yoin }
248     }
249     \yoin_article_set_readkey:nn { parent } { #1 }
250   ,
251
252   unknown .code:n =
253     \yoin_article_set_readkey:nn { \l_keys_key_tl } { #1 }
254   ,
255 }
256
257 \bool_new:N \g__yoin_article_readkeys_bool
258 \bool_gset_true:N \g__yoin_article_readkeys_bool
259
260 \cs_new_protected:Nn \yoin_article_readkeys: {
261   \bool_if:NT \g__yoin_article_readkeys_bool {
262     \file_if_exist:nT { ../ \l_yoin_article_currdir_tl .yoin } {
263       \yoin_keys_set_from_file:nn { yoin / toarticle } { ../ \l_yoin_article_currdir_tl .yoin }
264     }

```

```

265     }
266     \bool_gset_false:N \g__yoin_article_readkeys_bool
267 }
268
269 \tl_new:N \l__yoin_article_tmpa_tl
270 \seq_new:N \l__yoin_article_tmpa_seq
271 \tl_new:N \l_yoin_article_currdir_tl
272 \cs_new_protected:Nn \yoin_article_getcurrdir:N {
273     \tl_set:Nx \l__yoin_article_tmpa_tl { \currfileabsdir }
274     \cs_generate_variant:Nn \regex_extract_once:nnNF { nV }
275     \regex_extract_once:nVNF { /([~/]+)/Z } \l__yoin_article_tmpa_tl \l__yoin_article_tmpa_seq { \error }
276     \seq_get_right:NN \l__yoin_article_tmpa_seq #1
277 }
278
279 \AtBeginDocument{ \yoin_atbegindocument: }
280
281 \cs_new_protected:Nn \yoin_atbegindocument: {
282     \expandafter \newwrite \csname tf@yoin\endcsname
283     \bool_if:NTF \g_yoin_article_bool {
284         \yoin_article_getcurrdir:N \l_yoin_article_currdir_tl
285         \immediate \openout \csname tf@yoin\endcsname \l_yoin_article_currdir_tl .yoin\relax
286         \yoin_article_readkeys:
287         \setcounter { page } { \l_yoin_article_firstpage_int }
288         \yoin_article_writekeys:
289     } {
290         \immediate \openout \csname tf@yoin\endcsname \jobname .yoin\relax
291     }
292 }

```

7 yoinProcess (undocumented)

```

293 \int_new:N \g_yoin_page_int
294 \iow_new:N \g__yoin_yoinprocess_stream
295 \cs_new_protected:Nn \yoin_yoinprocess:n {
296     \keys_set:nn { yoin / yoinprocess } { #1 }
297     \clearpage
298     \message{AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA}
299     \bool_if:NTF \l__yoin_yoinprocess_cleardoublepage_bool {
300         \bool_if:NTF \l__yoin_yoinprocess_setpagenumber_bool {

```

```

301      %^^A CDP + SPN
302      \message{(((CDP+SPN)))}
303      \int_if_odd:nT { \value { page } + \l__yoin_yoinprocess_setpagenumber_int } {
304          \bool_if:NT \l__yoin_yoinprocess_output_bool {
305              \hbox {} \newpage \if@twocolumn \hbox {} \newpage \fi
306          }
307      }
308      \setcounter { page } { \int_use:N \l__yoin_yoinprocess_setpagenumber_int }
309  } {
310      %^^A CDP
311      \message{(((CDP)))}
312      \bool_if:NT \l__yoin_yoinprocess_output_bool { \cleardoublepage }
313  }
314  } {
315      \bool_if:NTF \l__yoin_yoinprocess_setpagenumber_bool {
316          %^^A SPN
317          \message{(((SPN)))}
318          \setcounter { page } { \int_use:N \l__yoin_yoinprocess_setpagenumber_int }
319      } {
320          %^^A neither
321          \message{((( )))}
322      }
323  }
324  \int_gset:Nn \g_yoin_page_int { \value { page } }
325  \seq_map_inline:Nn \g_yoin_yoinadd_seq {
326      \bool_if:NT \l__yoin_yoinprocess_openright_bool {
327          \__yoin_yoinprocess_cleardoublepage:
328      }
329      \bool_if:NT \l__yoin_yoinprocess_output_bool {
330          \includepdf [ pages = - ] { ##1 / \yoin_yoinadd_prop_item:nn { ##1 } { jobname } .pdf }
331      }
332      \iow_open:Nn \g__yoin_yoinprocess_stream { ##1 .yoin }
333      \bool_if:NTF \l__yoin_yoinprocess_alwayspageone_bool {
334          \iow_now:Nx \g__yoin_yoinprocess_stream { firstpage ~ = ~ 1 , }
335      } {
336          \iow_now:Nx \g__yoin_yoinprocess_stream { firstpage ~ = ~ \int_use:N \g_yoin_page_int , }
337      }
338      \iow_now:Nx \g__yoin_yoinprocess_stream { parent ~ = ~ \jobname , }
339      \iow_close:N \g__yoin_yoinprocess_stream
340      \int_gadd:Nn \g_yoin_page_int { \yoin_yoinadd_prop_item:nn { ##1 } { totpages } }

```

```

341     }
342 }
343 \DeclareDocumentCommand \yoinProcess { 0{ } } { \yoin_yoinprocess:n { #1 } }
344
345 \cs_new_protected:Nn \__yoin_yoinprocess_cleardoublepage: {
346     \bool_if:NT \l__yoin_yoinprocess_output_bool { \cleardoublepage }
347     \int_if_even:nT { \g_yoin_page_int } { \int_gincr:N \g_yoin_page_int }
348 }
349
350 \bool_new:N \l__yoin_yoinprocess_cleardoublepage_bool
351 \bool_new:N \l__yoin_yoinprocess_output_bool
352 \bool_new:N \l__yoin_yoinprocess_openright_bool
353 \bool_new:N \l__yoin_yoinprocess_alwayspageone_bool
354 \bool_new:N \l__yoin_yoinprocess_setpagenumber_bool
355 \int_new:N \l__yoin_yoinprocess_setpagenumber_int
356 \keys_define:nn { yoin / yoinprocess } {
357
358     cleardoublepage .bool_set:N = \l__yoin_yoinprocess_cleardoublepage_bool ,
359     cleardoublepage .initial:n = { false },
360
361     output .bool_set:N = \l__yoin_yoinprocess_output_bool ,
362     output .initial:n = { true },
363
364     openright .bool_set:N = \l__yoin_yoinprocess_openright_bool ,
365     openany .bool_set_inverse:N = \l__yoin_yoinprocess_openright_bool ,
366     openright .initial:n = { false },
367
368     setpagenumber .code:n =
369         \str_if_eq:nnTF { #1 } { false } {
370             \bool_set_false:N \l__yoin_yoinprocess_setpagenumber_bool
371         } {
372             \bool_set_true:N \l__yoin_yoinprocess_setpagenumber_bool
373             \int_set:Nn \l__yoin_yoinprocess_setpagenumber_int { #1 }
374         }
375     ,
376     setpagenumber .initial:n = { false },
377
378     alwayspageone .bool_set:N = \l__yoin_yoinprocess_alwayspageone_bool ,
379     alwayspageone .initial:n = { false },
380

```

```
381 }  
382
```

8 Experimental

```
\bla
```

```
383 \cs_new:Nn \yoin_blabla: {  
384   Blabla  
385 }  
386  
387 \</package>
```