1 User documentation

User documentation would appear here.

2 Package header

We load the required packages needed for LaTeX3, and the package header.

```
⟨*package⟩
⟨@@=yoin⟩
```

Necessary packages: First, LaTeX3 stuff.

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

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

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

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

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

Package header.

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

3 General macros

Macros not necessarily related to the package; moreorless an addition to LaTeX3.

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

```
\cs_new_protected:Nn \yoin_seq_gappend_clist:Nn {
   \seq_set_from_clist:Nn \l__yoin_tmpa_seq { #2 }
   \seq_gconcat:NNN #1 #1 \l__yoin_tmpa_seq
}
```

(*End definition for* `\yoin_seq_gappend_clist:Nn`*. This function is documented on page* **??**.)

4 Key–value interface for the package setup

First, we define the variables to store the keys. Booleans:

```
\bool_new:N \g_yoin_subprocess_bool
\bool_new:N \g_yoin_article_bool
\bool_new:N \g_yoin_dryrun_bool
\bool_new:N \g_yoin_onlyflags_bool
\bool_new:N \g_yoin_onlytags_bool
```

Sequences for flags, tags and their filtering:

```
\seq_new:N \g_yoin_flags_seq
\seq_new:N \g_yoin_tags_seq
\seq_new:N \g_yoin_onlyflags_seq
\seq_new:N \g_yoin_onlytags_seq
```

msg: unknown-flag
msg: unknown-tag

Two messages, for unknown flags and unknown tags.

```
\msg_new:nnnn { yoin } { unknown-flag }
   { The ~ flag ~ ‘#1’ ~ is ~ unknown ~ to ~ ‘yoin’. }
   { You ~ either ~ misspelled ~ it ~ or ~forgot ~ to ~ declare ~ it. }

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

(*End definition for* `msg: unknown-flag` *and* `msg: unknown-tag`*. These functions are documented on page* ??.)

keys: yoin / general

The keys themselves:

```
\keys_define:nn { yoin / general } {
```

Booleans:

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

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

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

Keys whose `clist` values are appended to a `seq`:

```
  defineflags .code:n = \yoin_seq_gappend_clist:Nn \g_yoin_flags_seq { #1 },
  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.

```
  onlyflags .code:n =
    \seq_gset_from_clist:Nn \g_yoin_onlyflags_seq { #1 }
    \bool_gset_true:N \g_yoin_onlyflags_bool
    ,
  onlytags .code:n =
    \seq_gset_from_clist:Nn \g_yoin_onlytags_seq { #1 }
    \bool_gset_true:N \g_yoin_onlytags_bool
    ,
}
```

(*End definition for* `keys: yoin / general`*. This function is documented on page* ??.)

\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`.

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

(*End definition for* `\ProcessKeysPackageOptions`*. This function is documented on page* ??.)

\yoin_setup:n
\yoinSetup

Allow keys to be set later. We define both a LaTeX3 interface and an xparse UI wrapper.

```
\cs_new_protected:Nn \yoin_setup:n {
  \keys_set:nn { yoin / general } { #1 }
}
\NewDocumentCommand \yoinSetup { R[]{} } {
  \yoin_setup:n { #1 }
}
```

(*End definition for* `\yoin_setup:n` *and* `\yoinSetup`*. These functions are documented on page* ??.)

5 \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. \g_yoin_articles_seq stores the list of the existing articles. \yoin_yoinadd_prop:n returns the name of the prop for the given article; *no check for existence is done at this place.*

```
\seq_new:N \g_yoin_articles_seq

\cs_new:Nn \yoin_yoinadd_prop:n {
   g__yoin_article_#1_prop
}
\cs_generate_variant:Nn \yoin_yoinadd_prop:n { V }
\cs_new:Nn \yoin_yoinadd_prop_item:nn {
   \prop_item:cn { \yoin_yoinadd_prop:n { #1 } } { #2 }
}
\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 l3keys 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.

```
\tl_new:N \l__yoin_yoinadd_currentarticle_tl

\cs_new_protected:Nn \__yoin_yoinadd_storekey:nn {
   \prop_gput:cnn { \yoin_yoinadd_prop:V \l__yoin_yoinadd_currentarticle_tl } { #1 } { #2 }
}
\cs_generate_variant:Nn \__yoin_yoinadd_storekey:nn { x , V }

\cs_new_protected:Nn \yoin_yoinadd:nn {
   \tl_set:Nn \l__yoin_yoinadd_currentarticle_tl { #1 }
   \seq_if_in:NnTF \g_yoin_articles_seq { #1 } {
      \msg_error:nnn { yoin } { yoinadd / duplicatearticle } { #1 }
   } {
      \seq_gput_right:Nn \g_yoin_articles_seq { #1 }
      \prop_new:c { \yoin_yoinadd_prop:n { #1 } }
      \__yoin_yoinadd_storekey:nn { article } { #1 }
      \keys_set:nn { yoin / yoinadd } { #2 }
      \file_if_exist:nT { #1 / #1 .yoin } {
         \yoin_keys_set_from_file:nn { yoin / yoinaddfromarticle } { #1 / #1 .yoin }
      }
   }
}
```

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

The error message for adding a duplicate article.

```
\msg_new:nnn { yoin } { yoinadd / duplicatearticle }
  { The ~ article ~ "#1" ~ has ~ been ~ already ~ processed ~ by ~ \token_to_str:N \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.

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

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

```
\keys_define:nn { yoin / yoinadd } {
  tag .code:n =
    \__yoin_error_if_tag_undefined:n { #1 }
    \__yoin_yoinadd_storekey:nn { tag } { #1 }
    ,
}
\keys_define:nn { yoin / yoinaddfromarticle } {
  unknown .code:n =
    \__yoin_yoinadd_storekey:Vn \l_keys_key_tl { #1 }
    ,
}
```

6 Conditionals for checking the existence of tags and flags

```
\prg_new_protected_conditional:Nnn \yoin_if_tag_defined:n { T, F, TF } {
  \seq_if_in:NnTF \g_yoin_tags_seq { #1 } { \prg_return_true: } { \prg_return_false: }
}
\prg_new_protected_conditional:Nnn \yoin_if_flag_defined:n { T, F, TF } {
  \seq_if_in:NnTF \g_yoin_flags_seq { #1 } { \prg_return_true: } { \prg_return_false: }
}
\cs_new_protected:Nn \__yoin_error_if_tag_undefined:n {
  \yoin_if_tag_defined:nF { #1 } { \msg_error:nnn { yoin } { unknown-tag } { #1 } }
}
\cs_new_protected:Nn \__yoin_error_if_flag_undefined:n {
   \yoin_if_flag_defined:nF { #1 } { \msg_error:nnn { yoin } { unknown-flag } { #1 } }
}
```

7 Environment yoinshell

Key–value interface to `yoinshell`

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

   flag .code:n =
      \__yoin_error_if_flag_undefined:n { #1 }
      \bool_if:NT \g_yoin_onlyflags_bool {
         \seq_if_in:NnF \g_yoin_onlyflags_seq { #1 } {
            \keys_set:nn { yoin / yoinshell } {
               ignore = true
            }
         }
      }
      ,

   ignore .bool_set:N = \l_yoin_yoinshell_ignore_bool,
   ignore .initial:n = { false },

}
```

A reasonable shell escape (we use its expandable variant

```
\file_if_exist:nTF { shellesc.sty } {
   \RequirePackage { shellesc }
} {
   \def \ShellEscape #1 { \immediate \write 18 { #1 } }
}
\cs_new_protected:Nn \__yoin_yoinshell_shellescape:n {
   \ShellEscape { #1 }
}
\cs_generate_variant:Nn \__yoin_yoinshell_shellescape:n { x }
```

Environment `yoinshell`:

```
\cs_new_protected:Nn \yoin_yoinshell_begin:n {
   \group_begin:
   \keys_set:nn { yoin / yoinshell } { #1 }
   \bool_if:NTF \l_yoin_yoinshell_ignore_bool {
      \DeclareDocumentCommand \RunForEach { O{} m } { }
      \DeclareDocumentCommand \Run { O{} m } { }
   } {
      \DeclareDocumentCommand \RunForEach { O{} m } { \yoin_yoinshell_runforeach:nn { ##1 } { ##2 } }
      \DeclareDocumentCommand \Run { O{} m } { \yoin_yoinshell_run:nn { ##1 } { ##2 } }
   }
}

\cs_new_protected:Nn \yoin_yoinshell_end: {
   \group_end:
}

\NewDocumentEnvironment { yoinshell } { O{} } {
   \yoin_yoinshell_begin:n { #1 }
} {
   \yoin_yoinshell_end:
}
```

\RunForEach

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

\keys_define:nn { yoin / runforeach } {

   onlytag .code:n =
      \__yoin_error_if_tag_undefined:n { #1 }
      \tl_set:Nn \l__yoin_yoinshell_runforeach_onlytag_tl { #1 }
      ,

}

\tl_new:N \__yoin_yoinshell_runforarticle_tmpa_tl
\cs_new_protected:Nn \__yoin_yoinshell_runforarticle_keyfromprop:nNn {
   \prop_get:cnN { \yoin_yoinadd_prop:n { #1 } } { #3 } \l__yoin_yoinshell_runforarticle_tmpa_tl
   \let #2 \l__yoin_yoinshell_runforarticle_tmpa_tl
}

\cs_new_protected:Nn \__yoin_yoinshell_runforarticle:nn {
   \prop_get:cnN { \yoin_yoinadd_prop:n { #1 } } { tag } \l__yoin_yoinshell_runforarticle_tag_tl
   \bool_if:nT {
      \quark_if_no_value_p:N \l__yoin_yoinshell_runforarticle_tag_tl
      ||
      \quark_if_no_value_p:N \l__yoin_yoinshell_runforeach_onlytag_tl
      ||
      \tl_if_eq_p:NN \l__yoin_yoinshell_runforeach_onlytag_tl \l__yoin_yoinshell_runforarticle_tag_tl
   }{
      \group_begin:
      \__yoin_yoinshell_runforarticle_keyfromprop:nNn { #1 } \Article { article }
      \__yoin_yoinshell_runforarticle_keyfromprop:nNn { #1 } \Jobname { jobname }
      \__yoin_yoinshell_shellescape:n { #2 }
      \group_end:
   }
}

\cs_new_protected:Nn \yoin_yoinshell_runforeach:nn {
   \group_begin:
   \keys_set:nn { yoin / runforeach } { #1 }
   \seq_map_inline:Nn \g_yoin_articles_seq { \__yoin_yoinshell_runforarticle:nn { ##1 } { #2 } }
   \group_end:
}
```

8 Article setting stuff

Information to be stored in an auxiliary file.

```
\cs_new_protected:Nn \__yoin_article_write:n {
   \immediate \write \@auxout { \token_to_str:N \@writefile { yoin } { #1 } }
}

\cs_new_protected:Nn \__yoin_article_write_keyval:nn {
   \__yoin_article_write:n { #1 ~ = ~ #2 , }
```

```
}
\cs_generate_variant:Nn \__yoin_article_write_keyval:nn { nx }

\cs_new_protected:Nn \yoin_article_write_meta:nn {
   \__yoin_article_write_keyval:nn { meta-#1 } { #2 }
}

\cs_new_protected:Nn \yoin_article_write_keys: {
   \__yoin_article_write_keyval:nx { jobname } { \jobname }
   \__yoin_article_write_keyval:nx { totpages } { \ztotpages }
   \__yoin_article_write_keyval:nx { currdir } { \l_yoin_article_currdir_tl }
   \__yoin_article_write_keyval:nx { firstpage } { \int_use:N \l_yoin_article_firstpage_int }
}

\prop_new:N \l__yoin_article_readkeys_prop

\cs_new_protected:Nn \yoin_article_set_readkey:nn {
   \prop_put:Nnn \l__yoin_article_readkeys_prop { #1 } { #2 }
}

\int_new:N \l_yoin_article_firstpage_int
\int_set:Nn \l_yoin_article_firstpage_int { 1 }

\keys_define:nn { yoin / toarticle } {
   firstpage .code:n =
      \int_set:Nn \l_yoin_article_firstpage_int { #1 }
      \yoin_article_set_readkey:nn { firstpage } { #1 }
      ,

   unknown .code:n =
      \yoin_article_set_readkey:nn { \l_keys_key_tl } { #1 }
      ,
}

\cs_new_protected:Nn \yoin_article_read_keys: {
   \file_if_exist:nT { ../ \l_yoin_article_currdir_tl .yoin } {
      \yoin_keys_set_from_file:nn { yoin / toarticle } { ../ \l_yoin_article_currdir_tl .yoin }
   }
}

\tl_new:N \l__yoin_tmpa_tl
\seq_new:N \l__yoin_tmpa_seq
\tl_new:N \l_yoin_article_currdir_tl
\cs_new_protected:Nn \yoin_article_getcurrdir:N {
   \tl_set:Nx \l__yoin_tmpa_tl { \currfileabsdir }
   \cs_generate_variant:Nn \regex_extract_once:nnNF { nV }
   \regex_extract_once:nVNF { /([^/]+)/\Z } \l__yoin_tmpa_tl \l__yoin_tmpa_seq { \error }
   \seq_get_right:NN \l__yoin_tmpa_seq #1
}

\AtBeginDocument{ \yoin_atbegindocument: }

\cs_new_protected:Nn \yoin_atbegindocument: {
   \expandafter \newwrite \csname tf@yoin\endcsname
   \bool_if:NTF \g_yoin_article_bool {
      \yoin_article_getcurrdir:N \l_yoin_article_currdir_tl
      \immediate \openout \csname tf@yoin\endcsname \l_yoin_article_currdir_tl .yoin\relax
      \yoin_article_read_keys:
      \setcounter { page } { \l_yoin_article_firstpage_int }
      \yoin_article_write_keys:
   } {
      \immediate \openout \csname tf@yoin\endcsname \jobname .yoin\relax
   }
}
```

9 yoinProcess

```
\int_new:N \g_yoin_page_int
\iow_new:N \g__yoin_yoinprocess_stream
\cs_new_protected:Nn \yoin_yoinprocess:n {
   \keys_set:nn { yoin / yoinprocess } { #1 }
   \seq_map_inline:Nn \g_yoin_articles_seq {
      \cleardoublepage
      \int_gset:Nn \g_yoin_page_int { \value { page } }
      \includepdf [ pages = - ] { ##1 / \yoin_yoinadd_prop_item:nn { ##1 } { jobname } .pdf }
      \iow_open:Nn \g__yoin_yoinprocess_stream { ##1 .yoin }
      \iow_now:Nx \g__yoin_yoinprocess_stream { firstpage ~ = ~ \int_use:N \g_yoin_page_int , }
      \iow_close:N \g__yoin_yoinprocess_stream
      \int_gadd:Nn \g_yoin_page_int { \yoin_yoinadd_prop_item:nn { ##1 } { totpages } }
   }
}
\DeclareDocumentCommand \yoinProcess { O{} } { \yoin_yoinprocess:n { #1 } }
```

10 Experimental

```
\cs_new_protected:Nn \yoin_keys_set_from_file:nn {
   \tl_set_from_file:Nnn \l_tmpa_tl { } { #2 }
   \keys_set:nV { #1 } \l_tmpa_tl
}
\cs_generate_variant:Nn \keys_set:nn { nV }
```

\bla

```
\cs_new:Nn \yoin_blabla: {
   Blabla
}

```

(*End definition for* \bla*. This function is documented on page* ??.)

```
⟨/package⟩
```