The xr package*

David Carlisle†

2020-05-10

This package implements a system for eXternal References.

If one document needs to refer to sections of another, say `aaa.tex`, then this package may be loaded in the main file, and the command
`\externaldocument{aaa}`
given in the preamble.

Then you may use `\ref` and `\pageref` to refer to anything which has been given a `\label` in either `aaa.tex` or the main document. You may declare any number of such external documents.

If any of the external documents, or the main document, use the same `\label` then an error will occur as the label will be multiply defined. To overcome this problem `\externaldocument` has an optional argument. If you declare `\externaldocument[A-]{aaa}` Then all references from `aaa` are prefixed by `A-`. So for instance, if a section of `aaa` had `\label{intro}`, then this could be referenced with `\ref{A-intro}`. The prefix need not be `A-`, it can be any string chosen to ensure that all the labels imported from external files are unique. Note however that if your style declares certain active characters (`:` in French, `"` in German) then these characters can not usually be used in `\label`, and similarly may not be used in the optional argument to `\externaldocument`.

As first suggested in Enrico Gregorio's `xcite` package, the current version also allows `\cite` to reference `\bibitem` in the external document. For compatibility with `xcite`, `\externalcitedocument` is made available as an alias for `\externaldocument`

1 The macros

1 ⟨*package⟩

Check for the optional argument.

```
2 \def\externaldocument{\@ifnextchar[\XR@{\XR@[]}}
3 \let\externalcitedocument\externaldocument
```

Save the optional prefix. Start processing the first `aux` file.

```
4 \def\XR@[#1]#2{{%
```

*This file has version number v5.06, last revised 2020-05-10.

†The Author of Versions 1–4 was Jean-Pierre Drucbert

```
  \makeatletter
  \def\XR@prefix{#1}%
  \set@curr@file{#2}%
  \expandafter\XR@next\@curr@file.aux\relax\\}}
```

Process the next aux file in the list and remove it from the head of the list of files to process.

```
\def\XR@next#1\relax#2\\{%
  \edef\XR@list{#2}%
  \XR@loop{#1}}
```

Check whether the list of aux files is empty.

```
\def\XR@aux{%
  \ifx\XR@list\@empty\else\expandafter\XR@explist\fi}
```

Expand the list of aux files, and call \XR@next to process the first one.

```
\def\XR@explist{\expandafter\XR@next\XR@list\\}
```

If the aux file exists, loop through line by line, looking for \newlabel and \@input. Otherwise process the next file in the list.

```
\def\XR@loop#1{\openin\@inputcheck{#1}\relax
  \ifeof\@inputcheck
    \PackageWarning{xr}{^^JNo file #1^^JLABELS NOT IMPORTED.^^J}%
    \expandafter\XR@aux
  \else
    \PackageInfo{xr}{IMPORTING LABELS FROM #1}%
    \expandafter\XR@read\fi}
```

Read the next line of the aux file.

```
\def\XR@read{%
  \read\@inputcheck to\XR@line
```

The ... make sure that \XR@test always has sufficient arguments.

```
  \expandafter\XR@test\XR@line...\XR@}
```

Look at the first token of the line. If it is \newlabel, do the \newlabel. If it is \@input, add the filename to the list of files to process. Otherwise ignore. Go around the loop if not at end of file. Finally process the next file in the list.

2018 update: make sure the arguments are handled outside the \ifx test,

```
\long\def\XR@test#1#2#3#4\XR@{%
  \let\XR@tempa\@gobbletwo
  \ifx#1\newlabel
    \let\XR@tempa\@firstoftwo
  \else\ifx#1\bibcite
    \let\XR@tempa\@firstoftwo
  \else\ifx#1\@input
     \let\XR@tempa\@secondoftwo
  \fi\fi\fi
   \XR@tempa{#1{\XR@prefix#2}{#3}}{\edef\XR@list{\XR@list#2\relax}}%
  \ifeof\@inputcheck\expandafter\XR@aux
  \else\expandafter\XR@read\fi}
```

⟨/package⟩