

# THE LANGSELECT PACKAGE

V1.1.3      2026-09-18

Generate different language versions from a common source

Matthias WERNER<sup>1</sup>

<https://github.com/werner-matthias/osglecture>

This package supports the generation of documents in two or three language versions from a common source document. A corresponding user macro is generated, and there are several methods for determining the target language.

The package is part of the [osglecture](#) bundle, is integrated into the [osglecture](#) class and supported by the build script `ol1m`, but can also be used independently.

## Table of Contents

|          |                                  |          |            |                                        |           |
|----------|----------------------------------|----------|------------|----------------------------------------|-----------|
| <b>1</b> | <b>Introduction</b>              | <b>2</b> | <b>3.3</b> | <b>Individual Language Commands</b>    | <b>8</b>  |
| 1.1      | TL;DR . . . . .                  | 2        |            |                                        |           |
| 1.2      | Motivation . . . . .             | 2        | <b>4</b>   | <b>Interaction with Other Packages</b> | <b>8</b>  |
| 1.3      | Alternatives . . . . .           | 2        | 4.1        | Babel/Polyglossia . . . . .            | 8         |
| 1.4      | Languages . . . . .              | 3        | 4.2        | Csquotes . . . . .                     | 9         |
| <b>2</b> | <b>Usage</b>                     | <b>4</b> | <b>5</b>   | <b>FAQ</b>                             | <b>9</b>  |
| 2.1      | Selection of the Target Language | 4        | <b>6</b>   | <b>Implementation</b>                  | <b>10</b> |
| 2.2      | Options . . . . .                | 5        |            |                                        |           |
| <b>3</b> | <b>User Commands</b>             | <b>6</b> | <b>6</b>   | <b>Index</b>                           | <b>29</b> |
| 3.1      | Automatic Language Commands      | 6        |            |                                        |           |
| 3.2      | Starred Variant . . . . .        | 7        | <b>7</b>   | <b>Index</b>                           | <b>30</b> |

---

1. [matthias.werner@informatik.tu-chemnitz.de](mailto:matthias.werner@informatik.tu-chemnitz.de)

## 1 Introduction

### 1.1 TL;DR

Assume, you have a text with an English and a German version whilst you want to use a common source.

1. Include `\usepackage[languages={en,de}]{langselect}`
2. Use in the source `\lende{⟨texten⟩}{⟨textde⟩}`. `⟨texten⟩` will appear in the English version, whereas `⟨textde⟩` will appear in the German one.
  - Write text that appears in both versions outside the macro or use the star version so that you only have to write common data in the body text once, see Section 3.2.

3. Translate the document with

```
> latex "\def\olsTargetLanguage{en}\include{⟨document⟩}"
```

to get the English version and with

```
> latex "\def\olsTargetLanguage{de}\include{⟨document⟩}"
```

for the German version.

- There are several other ways to determine the target language, including via the job name and via `\DocumentMetadata`.

### 1.2 Motivation

Sometimes, multiple language versions of the same document are required. In the application area of the `oslecture` bundle, these are, for example, the lecture notes, lecture slides, and handouts for a course. It is often difficult to keep multiple parallel language versions consistent. This task is somewhat easier if all versions are contained in a common source document.

This package supports the generation of such language versions from a common L<sup>A</sup>T<sub>E</sub>X source document by providing commands for language variants and mechanisms for selecting the target language.

### 1.3 Alternatives

There are a number of alternatives to `langselect`.

**Ad hoc commands** It is relatively easy to create ad hoc solutions, such as the following

```
1 \newif\ifenglish
2 \englishtrue
3 \ifenglish
4     Welcome!
5 \else
6     ¡Bienvenido!
7 \fi
```

In fact, `langselect` basically follows this approach, but automates the creation of language commands, simplifies the sharing of non-language-dependent elements such as formulas, and enables the selection of the target language by different approaches.

**multilang** The sophisticated package `multilang`<sup>1</sup> by Richard GREWE creates language versions of given commands, whereby the active language is taken directly from `Babel` or `Polyglossia`. `langselect` has a similar goal, but a different (from the package author's point of view: more practical) interface. However, if you don't like the approach of `langselect` you should definitely take a look at `multilang`.

**comment** Victor EIJKHOUT wrote the package `comment`,<sup>2</sup> which makes it easy to output only certain sections of the document. This can also be used very well for language selection.

**bicaption/translations/translator/xtcpts** The packages `bicaption`<sup>3</sup>/`translations`<sup>4</sup>/`translator`<sup>5</sup>/`xtcpts`<sup>6</sup> are intended for programmers of L<sup>A</sup>T<sub>E</sub>X packages to internationalize fixed text elements such as the headings of tables of contents. These packages are only of limited use for document authors.

## 1.4 Languages

The concept of `langselect` involves the use of languages at different levels with different effects. To avoid confusion, we want to clearly define the terms here.

1. The set of *possible* target languages for the document. A document that uses `langselect` typically contains text sections in different target languages. The possible target languages are set via a package option, see Section 2.2. In the remaining part of this documentation, we refer to the *possible* target languages as *selectable languages*, since one of these languages is selected for the L<sup>A</sup>T<sub>E</sub>X translation.
2. The actual target language used in a L<sup>A</sup>T<sub>E</sub>X translation. This must be one of the languages from 1. Typically, the translation process is performed separately for each of the actual target languages. There are several methods for selecting the actual target language; see section 2.1.
3. The language in which text fragments or individual words are actually written, what is considered, e. g. for hyphenation. This is the concept of language used by packages such

---

1. on CTAN as `multilang`: <https://ctan.org/tex-archive/macros/latex/contrib/multilang>

2. on CTAN as `comment`: <https://ctan.org/tex-archive/macros/latex/contrib/comment>

3. on CTAN as `bicaption`: <https://ctan.org/tex-archive/macros/latex/contrib/bicaption>

4. on CTAN as `translations`: <https://ctan.org/tex-archive/macros/latex/contrib/translations>

5. on CTAN as `translator`: <https://ctan.org/tex-archive/macros/latex/contrib/translator>

6. on CTAN as `xtcpts`: <https://ctan.org/tex-archive/macros/latex/contrib/xtcpts>

as Babel or Polyglossia.<sup>7</sup> Please note that a multilingual document from the perspective of `langselect` can be a monolingual document from the perspective of Babel. Frequently, it is the other way around, and more languages than the selectable languages are used: For example, a document with the selectable languages German and English may contain Latin words or sentences in the text. This selection is made using the usual mechanisms such as `\selectlanguage`, `\foreignlanguage` or `\text{language}`. When Babel or Polyglossia is loaded, which can also be done directly via `langselect` if desired, the target language is selected as the active language.

Languages are designated in `langselect` using ISO 639-1 codes (“en”, “fr”, “de” ...). If a specific dialect or variant is desired for language processing at the Babel/Polyglossia level, mapping can be performed.

## 2 Usage

The package is loaded in the usual way:

```
\usepackage[options]{langselect}
```

Since the target language can be selected both via options and in other ways, these different methods are described here first. Further options, such as setting the selectable languages, are discussed in Section 2.2.

### 2.1 Selection of the Target Language

There are various ways to determine a target language for a LaTeX translation run. The order in the following list also reflects the order of evaluation. Once a target language has been determined, further evaluation is canceled.

1. Definition of a macro `\olsTargetLanguage` *before* loading the package. This makes it possible, for example, to use

```
> latexmk -usepretex="\def\olsTargetLanguage{fr}" document.tex
```

to set the target language French for the translation run.

2. Setting of the packet option `targetlang`. Various values are possible here:

```
targetlang = <ISO code>
```

The target language is specified directly by the code.

```
targetlang = {job=<n>}
```

The target language is determined from the  $n$ th element of the job name. An element is a part of the job name separated by “-” (minus), see the documentation for the package `varsfromjobname`,<sup>8</sup> which is loaded for this purpose. For example, with `targetlang = {job=2}` one can set Russian as target language using

7. However, in these packages, “language” also refers to several different and, in some cases, independent concepts such as hyphenation patterns, fonts to be used, etc.

8. on CTAN as `varsfromjobname`: <https://ctan.org/tex-archive/macros/latex/contrib/varsfromjobname>

```
> latex -jobname doc-ru doc.tex
```

```
targetlang = babel|polyglossia
```

To use this key value, Babel or Polyglossia must be loaded *before* `langselect`. The target language is then set to the (main) language set by the language packet.<sup>9</sup>

```
targetlang = meta
```

Uses the language specified in `\DocumentMetadata` as the target language.

3. If neither `\olsTargetLanguage` is defined nor the option `targetlang` is set, `langselect` attempts to determine the target language on its own by first trying the option `targetlang = babel` and then (if no language package was found) the option `targetlang = meta`.
4. If everything else fails, the first selectable languages is set as the target language.

## 2.2 Options

The following options are available:

`languages = {⟨list of selectable languages⟩}` (required)

Provides two or three selectable languages. The order matters for the automatic generation of the language commands (cf. Section 3).

`targetlang = ⟨language⟩|{job=⟨n⟩}|babel|polyglossia|meta`

See Section 2.1

`prefix = {⟨prefix⟩}`

Default: 1

Sets a prefix for multilingual commands. An empty prefix can also be set.

`auto = true|false`

Default: true

The actual language commands are created automatically, see Section 3. If `auto = false` is set, only the commands for generating the language commands are provided, not the language commands themselves.

`trim = true|false`

Default: true

Removes whitespace at the beginning and end of language macro arguments.

`load babel = {⟨Babel options⟩}`

`load polyglossia = {⟨Polyglossia options⟩}`

Loads either Babel or Polyglossia so that the target language of `langselect` is set as the main language. The options passed here are forwarded to the respective language package. If any option is provided, no language package must be loaded before `langselect`.

`map = {⟨language⟩=⟨variant⟩,...}`

Defines a mapping of languages to a language variant. When `langselect` interacts with other packages, the specific language variant is passed instead of the ISO code.

---

9. Both values have exactly the same effect: Since Babel and Polyglossia offer the same interface, it doesn't matter which of the two packages is loaded. Thus, `targetlang = polyglossia` will also find Babel.

For example, if you load Babel from `langselect` and your document explicitly uses *British* English and *Swiss* German, use the following code to ensure that Babel receives the correct options.

```

1 \documentclass{article}
2 \usepackage[
3   languages={en,de},
4   load=babel,
5   map={en=british,de=nswissgerman}
6 ]
7 \begin{document}
8   \title{\lende{Hallo}{Grüezi}}
9 
```

`unified shorthands = true|false` Default: true  
 Defines cross-language shorthands to support `csquotes`

`bcp47 = true|false` Default: true  
 When Babel is active, `langselect` makes the ISO codes used by `langselect` work as selector names for `\selectlanguage`. In case of `bcp47 = false`, Babel's target language must be selected through `map` or by hand. This option has no effect if Babel is not loaded.

## 3 User Commands

### 3.1 Automatic Language Commands

If you have not set the option `auto = false`, a multilanguage command will be created. Its names is determined by the list of selection languages. Depending on whether you have specified two or more selection languages, the name is

- $\backslash\langle\textit{prefix}\rangle\langle\textit{language}_1\rangle\langle\textit{language}_2\rangle\{\langle\textit{text}_1\rangle\}\{\langle\textit{text}_2\rangle\}$

or

- $\backslash\langle\textit{prefix}\rangle\langle\textit{language}_1\rangle\langle\textit{language}_2\rangle\langle\textit{language}_3\rangle\{\langle\textit{text}_1\rangle\}\{\langle\textit{text}_2\rangle\}\{\langle\textit{text}_3\rangle\}$

For example, for this documentation, the package was loaded with the following options:

```

1 \usepackage[
2   languages={de,en},
3   targetlang={job=2}
4 ]{langselect}
```

This makes the command `\ldeen` available.

If the target language “de” is now selected (here via the job name, but there are also other options, see Section 2.1), only  $\langle \text{Text}_1 \rangle$  is transferred to the PDF document and  $\langle \text{Text}_2 \rangle$  is discarded. With the target language “de”, it is the other way around.

### Attention!

The automatic generation of language commands can sometimes cause problems. For example, Abkhaz (code: “ab”) as the first language and Greek (Code: “el”) as the second language results in the automatic name `\label`, which leads to error messages. In such cases, use the features described in section 3.3.

## 3.2 Starred Variant

A common problem with multiple language versions is the consistency of non-language-dependent data, especially if this data changes frequently (think, for example, of timetables).

For such cases, `langselect` provides a starred variant of the language commands. In this starred variant, the text arguments can contain placeholders of the form “@ $\langle n \rangle$ ” can be used, where  $\langle n \rangle$  represents a value between 1 and 9.

Additional arguments must then be specified after the command, namely as many as correspond to the highest  $\langle n \rangle$  used. The starred variant replaces the placeholders with the corresponding arguments.

```

1 \documentclass{article}
2 \usepackage[languages={en,fr},targetlang=fr]{langselect}
3 \begin{document}
4 \lenfr*{
5   The formula @1 generally refers to the Pythagorean theorem.
6 }{
7   La formule @1 fait généralement référence au théorème de
8   Pythagore.
9 }{$a^{2}+b^{2}=c^{2}$}
10
11 \lenfr*{
12   Calculate the length of the third side if one side is @1 units long
13   and the hypotenuse is @2 units long.
14 }{
15   Calculez la longueur du troisième
16   côté si un côté mesure @1 unités et l'hypoténuse @2 unités.
17 }{$4$}{$5$}
18 \end{document}
```

La formule  $a^2+b^2 = c^2$  fait généralement référence au théorème de Pythagore.

Calculez la longueur du troisième côté si un côté mesure 4 unités et l'hypoténuse 5 unités.

The starred variant of the language macros is not fully expandable and should therefore not be used in contexts where full expandability is required. This applies in particular to PDF strings, such as those required by [hyperref](#) for the metadata of a PDF document. Although [langselect](#) selects the active language version there, placeholders such as “@<n>” are not replaced, and the corresponding replacement arguments remain in the input stream. Therefore, use only the unstarred form in PDF strings.

### 3.3 Individual Language Commands

If you do not like the generated names of the language commands or if they even lead to collisions, you can change them. The easiest way is to change the package option `prefix` so that, for example, with `prefix = mv`, the language command for the German/English combination is called `\mvdeen`.

If this is not sufficient, the following two commands give you complete control over the naming.

```
\olsMakeBilingualMacro[<prefix>]{<lang1>}{<lang2>}[<base name>]
```

```
\olsMakeTrilingualMacro[<prefix>]{<lang1>}{<lang2>}{<lang3>}[<base name>]
```

These commands create a bilingual or trilingual language command for the specified languages. If `<base name>` is specified, the name of the language command is `\<prefix><base name>`. If the `<prefix>` argument is used, it determines the prefix; otherwise, the package options apply.

## 4 Interaction with Other Packages

### 4.1 Babel/Polyglossia

The [langselect](#) package is initially independent of language packages such as Babel or Polyglossia. It can be used without or together with these packages. However, if you take the trouble to create a multilingual document, you will usually also want the correct hyphenation rules, font settings, date formats, etc. to be used for the respective language.<sup>10</sup>

Please note that as long as you do not use a different language *within* a text (perhaps the language of the other language version), your document is, from the perspective of Babel, a

10. In the following, I will refer exclusively to Babel. However, what has been said also applies accordingly to Polyglossia.

*monolingual* document. If you load Babel in the usual way with all the languages of your language version, Babel would pick the last one listed (or the one set via `main`) as its active language – not necessarily the target language of `langselect`. Without a countermeasure you could end up with incorrect hyphenation and similar errors.

To avoid this, `langselect` offers several options:

1. Unless the target language is derived from Babel anyway, `langselect` selects it as the active language itself at the start of the document. When Babel is active it enables Babel’s BCP 47 mapping for this (option `bcp47`), so that the ISO codes used by `langselect` suffice as selector names and the selectable languages can simply be loaded under their Babel names. A `map` entry is then only needed for named variants.
2. As mentioned above, you can use the package option `targetlang = babel`. Then the target language is determined by the main language of Babel. However, you then have to forego the option of controlling the target language “from outside” (i.e., via the job name or with `\olsTargetLanguage`).
3. In turn, `langselect` loads Babel or Polyglossia using the options `load babel/load polyglossia` and sets the target language as the main language. Additional options for the language packages are passed as option arguments.
4. Finally, the hooks `langselect/language/before` and `langselect/language/after` can be used to insert any code before or after the arguments of the language macro. E.g., `\foreignlanguage` may be used here. The before hook has the (mapped) target language as a parameter.

## 4.2 Csquotes

Different languages have different conventions for punctuation. This is particularly problematic with quotation marks, as these should follow the target language, but may also appear outside language macros or in arguments of the asterisk variant. The `csquotes`<sup>11</sup> package offers a solution for this by providing `\enquote` and other macros that follow the quotation convention of the current language.

For added convenience, `langselect` supports the use of correct quotation marks with language-independent shorthands: If the option `unified shorthands` is true, `csquotes` is loaded and the shorthands `"`` and `"'` are set to `\openautoquote` and `\closeautoquote`, respectively.

## 5 FAQ

### • Is `langselect` stable?

Although the package has now reached version 1<sup>12</sup> and is in heavy use—at least by the author—I cannot promise that it is bug-free enough to deserve the “stable” label.

11. on CTAN as `csquotes`: <https://ctan.org/tex-archive/macros/latex/contrib/csquotes>

12. Incidentally, it is the only package in the `osglecture` bundle

Please report any bugs you find via <https://github.com/werner-matthias/osglecture/issues>.

- **What does the “o” in the `\ols` prefix stand for?**

The package belongs to the `osglecture` bundle, from which the “o” is inherited.

- **Are there any application examples that go beyond a toy Hello World?**

Yes, a few. Take a look at the source code of this documentation, for example.

- **I have an `@1` in the output where an argument substitution should take place.**

This usually has one of the following two causes:

1. You are using an old version ( $\leq$  vo.6.0) of `langselect`. In that version, different catcodes—for example, those generated by `\makeatletter`—could cause problems.
2. You forgot the asterisk in the language macro.

- **Can I omit an `@n` parameter in the starred version for one language?**

No. Besides, then the argument would obviously no longer be shared.

- **Can I use `@n` parameters in mixed order or more than once?**

Yes.

- **I want to use `@`, but not as a parameter, in a star variant of the language macro.**

Just do it.

- **But I need `@42`.**

Use `@{ }42`.

- **I use Russian, but the font is incorrect.**

This is not a problem with `langselect` (which does not deal with the font at all), but probably with Babel, `fontenc`, `fontspec`, etc. But maybe you find the hooks useful, cf. section 4.1.

- **My question does not appear here.**

You may ask at <https://github.com/werner-matthias/osglecture/issues>.

## 6 Implementation

```

1 \NeedsTeXFormat{LaTeX2e}[2022/06/01]
2 \def\packagename{langselect}
3 \ProvidesExplPackage{langselect}
4 {2026-09-18}
5 {v1.1.3}
6 {support for different languages versions from a common source}
7 \ExplSyntaxOff
8 \ExplSyntaxOn

```

## 6 Implementation

`\olsIsoTags` contains all valid ISO 639-1 code tags.

```

9 \seq_const_from_clist:cn{olsIsoTags}{aa,ab,ae,af,ak,am,an,ar,as,av,ay,az,ba,be,
10   bg,bh,bi,bm,bn,bo,br,bs,ca,ce,ch,co,cr,cs,cu,cv,cy,da,de,dv,dz,ee,el,en,eo,es,
11   et,eu,fa,ff,fi,fj,fo,fr,fy,ga,gd,gl,gn,gu,gv,ha,he,hi,ho,hr,ht,hu,hy,hz,ia,id,
12   ie,ig,ii,ik,io,is,it,iu,ja,jv,ka,kg,ki,kj,kl,km,kn,ko,kr,ks,ku,kv,kw,ky,la,
13   lb,lg,li,ln,lo,lt,lu,lv,mg,mh,mi,mk,ml,mn,mr,ms,mt,my,na,nb,nd,ne,ng,nl,nn,no,
14   nr,nv,ny,oc,oj,om,or,os,pa,pi,pl,ps,pt,qu,rm,rn,ro,ru,rw,sa,sc,sd,se,sg,si,sk,
15   sl,sm,sn,so,sq,sr,ss,st,su,sv,sw,ta,te,tg,th,ti,tk,tl,tn,to,tr,ts,tt,tw,ty,ug,
16   uk,ur,uz,ve,vi,vo,wa,wo,xh,yi,yo,za,zh,zu}
17
18 \newcommand\IfTagIsValidF[2]{%
19   \seq_if_in:ceF{olsIsoTags}{#1}{#2}
20 }
21 \RequirePackage{varsfromjobname}[2025/08/03]
```

The results of `\varsfromjobname` and `\str_range:Nnn` have catcode 12, but ISO tags and options have catcode 11. We standardize using a wrapper on catcode 11.

```

22 \newcommand\olsChangeCatEleven[1]{
23   \def\tmpa{#1}
24   \edef\tmpb{
25     \noexpand\scantokens{
26       \noexpand\edef\noexpand\olsCatElven{\tmpa}
27     }
28   }
29   \tmpb
30 }
```

If necessary, we map language codes to language variants. When interacting with external code, the language variant is then passed on.

```

31 \cs_new:Npn \ols_use_map:n #1 {
32   \cs_if_exist_use:cF { olslangmap / #1 }{ #1 }
33 }
```

`\ols_babel_name_from_ini:nN` reads the field `name.babel` from `babel-⟨Code⟩.ini`, which may contain multiple Babel names for the same language. Often, these are simply alternative names for the same thing. For German, however, there are two distinct orthographies (`german` = old, `ngerman` = reformed orthography). If `ngerman` is among the candidates, it is therefore preferred (today's standard for German). Otherwise, the last entry is used. If anyone who is more familiar with languages than I am discovers another necessary special case: Please open a corresponding issue.

```

34 \ior_new:N \g_ols_ini_ior
35 \seq_new:N \ols_ini_seq
36 \seq_new:N \ols_ini_names_seq
```

## 6 Implementation

```

37 \cs_new_protected:Npn \ols_babel_name_from_ini:nN #1 #2
38 {
39   \tl_clear:N #2
40   \ior_open:NnT \g_ols_ini_ior { babel- #1 .ini }
41   {
42     \ior_str_map_inline:Nn \g_ols_ini_ior
43     {
44       \str_if_eq:eeT { \str_range:nnn {##1} {1} {10} } { name.babel }
45       {
46         \seq_set_split:Nnn \ols_ini_seq { = } { ##1 }
47         \seq_set_split:Nne \ols_ini_names_seq { ~ }
48         { \tl_trim_spaces:e { \seq_item:Nn \ols_ini_seq {2} } } }
49         \seq_if_empty:NF \ols_ini_names_seq
50         {
51           \seq_if_in:NeTF \ols_ini_names_seq { \tl_to_str:n {ngerman}
52             ↪ }
53           { \tl_set:Nn #2 { ngerman } }
54           { \tl_set:Ne #2 { \seq_item:Nn \ols_ini_names_seq {-1} } } }
55         }
56         \ior_map_break:
57       }
58     }
59   }

```

Lines from `\ior_str_map_inline` have catcode 12. Babel compares language options with `\ifx` (catcode sensitive), so we turn the name back into letters with standard catcodes.

```

60   \tl_if_empty:NF #2 { \tl_set_rescan:NnV #2 { } #2 }
61 }

```

Babel has a list of language names (sorted differently in the various versions). That is why we turn to the “competition” and use `\ols_babel_name_from_polyglossia:nN` `polyglossias` `gloss-⟨Code⟩.ldf` files as our data source: for languages with multiple Babel names (such as German), these contain an explicit `babelname` field with a single, unique value. `polyglossia` itself is not loaded for this; its text files are just read line by line, the same way `\ols_babel_name_from_ini:nN` already does.

```

62 \tl_new:N \ols_ini_module_tl
63 \seq_new:N \ols_ini_regex_seq
64 \cs_new_protected:Npn \ols_babel_name_from_polyglossia:nN #1 #2
65 {
66   \tl_clear:N #2
67   \tl_clear:N \ols_ini_module_tl
68   \ior_open:NnT \g_ols_ini_ior { gloss- #1 .ldf }
69   {

```

## 6 Implementation

```

70     \ior_str_map_inline:Nn \g_ols_ini_ior
71     {
72         \regex_extract_once:nnN { InheritGlossFile\{([a-zA-Z]+)\} } {##1}
73         ↪ \ols_ini_regex_seq
74         \seq_if_empty:NF \ols_ini_regex_seq
75         {
76             \tl_set:Ne \ols_ini_module_tl { \seq_item:Nn \ols_ini_regex_seq
77             ↪ {2} }
78             \ior_map_break:
79         }
80     }
81     \ior_close:N \g_ols_ini_ior
82 }

```

The module name itself is the Babel name unless the target file defines its own `babelname` – true for most languages, where the Polyglossia and Babel names coincide anyway.

```

81     \tl_if_empty:NF \ols_ini_module_tl
82     {
83         \ior_open:NnT \g_ols_ini_ior { gloss- \ols_ini_module_tl .ldf }
84         {
85             \ior_str_map_inline:Nn \g_ols_ini_ior
86             {
87                 \regex_extract_once:nnN { babelname\s*=\s*([a-zA-Z]+) } {##1}
88                 ↪ \ols_ini_regex_seq
89                 \seq_if_empty:NF \ols_ini_regex_seq
90                 {
91                     \tl_set:Ne #2 { \seq_item:Nn \ols_ini_regex_seq {2} }
92                     \ior_map_break:
93                 }
94             }
95             \ior_close:N \g_ols_ini_ior
96         }
97         \tl_if_empty:NT #2 { \tl_set_eq:NN #2 \ols_ini_module_tl }
98     }
99     \tl_if_empty:NF #2 { \tl_set_rescan:NnV #2 { } #2 }

```

`\ols_babel_name:nN` tries `polyglossia`’s unambiguous entry first, and only falls back – e.g. because `polyglossia` isn’t installed at all, or a language has no gloss file of its own – to Babel’s own, sometimes ambiguous `.ini` list.

```

100 \cs_new_protected:Npn \ols_babel_name:nN #1 #2
101 {
102     \ols_babel_name_from_polyglossia:nN {#1} #2
103     \tl_if_empty:NT #2 { \ols_babel_name_from_ini:nN {#1} #2 }
104 }

```

## 6 Implementation

We define some options. All except `languages` are optional. We will store some results in variables that we first create.

```
105 \bool_new:N\ols_load_babel
106 \bool_new:N\ols_load_polyglossia
```

True when the target language was derived from the active language package. Its main language then already is the target language and must not be selected again (possibly with a code instead of a language name).

```
107 \bool_new:N\ols_target_from_langpkg
108 \DeclareKeys{
109   languages.clist_gset:N = \ols_langs,
110   languages.usage=load,
111   targetlang.code = {
```

The option `targetlang` will be only evaluated in case, the target language isn't already set by definition of `\olsTargetLanguage`.

```
112   \cs_if_exist:NF\olsTargetLanguage{
```

Is an element of the jobname given?

```
113   \bool_if:nT{
114     \str_if_eq_p:ee{\str_range:nnn{#1}{1}{3}}{job}
115   }{%
116     \olsChangeCatEleven{\getfromjobname{\str_range:Nnn{#1}{-1}{-1}}}
117     \xdef\olsTargetLanguage{\olsCatEleven}
118   }
```

Should the target language be derived from `babel` or `polyglossia`?

```
119   \bool_if:nT{
120     \str_if_eq_p:nn{#1}{babel} || \str_if_eq_p:nn{#1}{polyglossia}
121   }{
122     \str_if_eq:eeF{\BCPdata{main.language}}{ }{
123       \edef\olsTargetLanguage{\BCPdata{main.language}}
124       \bool_set_true:N\ols_target_from_langpkg
125     }
126   }
```

Should the language of the meta data be used?

```
127   \str_if_eq:nnTF{#1}{meta}{
128     \IfDocumentMetadataTF{
129       \edef\tmpa{\GetDocumentProperty{document/lang}}
130       \edef\tmpb{\str_range:Nnn\tmpa{1}{2}}
131       \olsChangeCatEleven{\tmpb}
```

```

132     \edef\olsTargetLanguage{\olsCatElven}
133   }{
134     \PackageWarningNoLine{\packagename}{No~meta~data~provided}
135   }
136 }
137 }

```

If no language is assigned by now, the argument is assumed to be the language. We check later, whether it is valid.

```

138   \cs_if_exist:NF \olsTargetLanguage {
139     \edef\olsTargetLanguage{#1}
140   }
141 },

```

Prefix of the user language commands.

```

142   targetlang.usage=load,
143   prefix.store=\ols_sprefix,
144   prefix.initial:n=l,
145   prefix.usage=load,

```

Automatic generation of the language commands?

```

146   auto.bool_set:N=\ols_generate,
147   auto.usage=load,
148   auto.initial:n=true,

```

Automatic removal of heading and trailing spaces?

```

149   trim.bool_set:N=\ols_trim,
150   trim.usage=load,
151   trim.initial:n=true,

```

Mapping to language variants.

```

152   map.code:n = {
153     \prop_new:N \ols_lang_pl
154     \prop_clear:N \ols_lang_pl
155     \prop_set_from_keyval:Nn \ols_lang_pl {#1}

```

Since property lists cannot be expanded, we convert the entries to ordinary macros.

```

156     \prop_map_inline:Nn \ols_lang_pl
157     { \cs_gset:cpn { olslangmap / \tl_to_str:n {##1} } {##2} }
158   },

```

Load `babel` or `polyglossia` with options. An empty value (load babel without =) **must not** be forwarded as an empty option.

## 6 Implementation

```

159   load~babel .code:n = {
160     \bool_set_true:N \ols_load_babel
161     \tl_if_empty:nF {#1} { \PassOptionsToPackage{#1}{babel} }
162   },
163   load~polyglossia .code:n = {
164     \bool_set_true:N \ols_load_polyglossia
165     \tl_if_empty:nF {#1} { \PassOptionsToPackage{#1}{polyglossia} }
166   },
167   load.usage = load,
168   unified~shorthands.bool_set:N=\ols_shorthands,
169   unified~shorthands.initial:n=true,
170   unified~shorthands.usage=load,

```

Should the ISO codes used by `langselect` be accepted as BCP 47 selector names by an active `babel`?

```

171   bcp47.bool_set:N=\ols_bcptags,
172   bcp47.initial:n=true,
173   bcp47.usage=load
174 }%
175 \ProcessKeyOptions\relax

```

If no target language has been assigned via `\olsTargetLanguage` or `targetlang`, we check again whether a language can be identified via language packages or `\DocumentMetadata`.

```

176 \cs_if_exist:NF \olsTargetLanguage {
177   \str_if_eq:eeF{\BCPdata{main.language}}{}{
178     \edef\olsTargetLanguage{\BCPdata{main.language}}
179     \bool_set_true:N\ols_target_from_langpkg
180   }
181 }
182 \cs_if_exist:NF \olsTargetLanguage {
183   \IfDocumentMetadataT{
184     \edef\tmpa{\GetDocumentProperty{document/lang}}
185     \edef\tmpb{\str_range:Nnn\tmpa{1}{2}}
186     \olsChangeCatEleven{\tmpb}
187     \edef\olsTargetLanguage{\olsCatElven}
188   }
189 }

```

We are doing some checks. Are at least two selectable languages specified?

```

190 \int_compare:nNtT{\clist_count:N\ols_langs} < {2}{
191   \PackageError{\packagename}{No~sufficient~number~of~selectable
192     \MessageBreak~languages~provided}{
193     You~have~to~provide~a~list~of~two~or~three~\MessageBreak
194     languages~via~'language'~option.}

```

```

195 \aftergroup\endinput
196 }

```

Currently, `langselect` can't process more than three languages. Thus, we issue a warning if the list of selectable languages contains more than three languages.

```

197 \int_compare:nNt{\clist_count:N\ols_langs} > {3}{
198   \PackageWarningNoLine{\packagename}{Too~many~selectable~languages~provided.
199     \MessageBreak I~will~ignore~the~superfluous~languages}
200 }

```

Next, we test the validity of the selectable languages. They should form ISO 639-1 codes.

```

201 \clist_map_inline:Nn\ols_langs {
202   \IfTagIsValidF{#1}{
203     \PackageError{\packagename}{Couldn't~resolve~selectable~language~'#1'}{
204       Use~valid~ISO-639-1~code~in~option~'languages'.}
205     \aftergroup\endinput
206   }
207 }

```

Now we check whether a valid target language could be determined.

```

208 \cs_if_exist:NF \olsTargetLanguage {
209   \PackageWarningNoLine{\packagename}{Can't~identify~any~target~language.
210     \MessageBreak Falling~back~to~'\clist_item:Nn \ols_langs {1}'}
211   \edef\olsTargetLanguage{\clist_item:Nn \ols_langs {1}}
212   \IfTagIsValidF{\olsTargetLanguage}{
213     \PackageWarningNoLine{\packagename}{Do~not~recognize~language~
214       ↪ '\olsTargetLanguage'.
215     \MessageBreak Take~'\clist_item:Nn \ols_langs {1}'~as~replacement}
216     \edef\olsTargetLanguage{\clist_item:Nn \ols_langs {1}}
217   }

```

The list of selectable languages should include the target language.

```

217 \clist_if_in:NVF\ols_langs{\olsTargetLanguage}{
218   \PackageError{\packagename}{Target~language~'\olsTargetLanguage'~is~
219     ↪ not~\MessageBreak
220     in~the~list~of~selectable~languages}{
221     Check~option~'language'~and~'targetlang'.}
222 }

```

Babel and Polyglossia cannot both be loaded.

```

222 \bool_if:nT{
223   \ols_load_babel && \ols_load_polyglossia
224 }{

```

## 6 Implementation

```

225 \bool_set_false:N \ols_load_babel
226 \bool_set_false:N \ols_load_polyglossia
227 \PackageError{\packagename}{The~options~'load~babel'~and~'load~polyglossia'
228 \MessageBreak
    ↳ are~mutual~exclusive,~you~can't~use~both}{Use~one~load~option,~only.}
229 }

```

If a language package is already active, we don't load `babel`/`polyglossia` ourselves. The target language is activated later.

```

230 \bool_if:nT{
231   ( \cs_if_exist_p:c{ver@babel.sty} || \cs_if_exist_p:c{ver@polyglossia.sty} )
232   &&
233   ( \ols_load_babel || \ols_load_polyglossia )
234 }{
235   \bool_set_false:N \ols_load_babel
236   \bool_set_false:N \ols_load_polyglossia
237   \PackageWarningNoLine{\packagename}{Can't~load~babel/polyglossia~for
238     ~you,since\MessageBreak
239     a~language~package~is~already~loaded.\MessageBreak
240     I'll~ignore~that~option}
241 }

```

`\ols_mapped_target_lang` is the name of the target language as it is passed on to external code, if applicable, i.e. `\DocumentMetadata`, language packs, and hooks.

```

242 \edef\ols_mapped_target_lang{\ols_use_map:n{\olsTargetLanguage}}

```

The target language should now be determined. We write the information into the log.

```

243 \PackageNoteNoLine{\packagename}{Target~language~is~'\olsTargetLanguage'~}
    ↳ (\ols_mapped_target_lang)}

```

If needed, we correct the language metadata set by `\DocumentMetadata`. Now, Babel (for example) does not derive a wrong main language.

```

244 \IfDocumentMetadataT{
245   \edef\olsDocLang{\exp_args:Nc\str_range:nnn{\GetDocumentProperty{
    ↳ document/lang}}{1}{2}}
246   \bool_if:nF{
247     \str_if_eq_p:ee{\olsDocLang}{\olsTargetLanguage}
248   }{
249     \PackageWarningNoLine{\packagename}{Target~language~'\olsTargetLanguage'~
250       doesn't~comply~with\MessageBreak \c_backslash_str
251       DocumentMetadata~'\GetDocumentProperty{document/lang}'.\MessageBreak
252       I'll~try~to~overwrite~the~meta~data}
253     \AddToDocumentProperties[document]{lang}{\olsTargetLanguage}
254   }
255 }

```

Since the name of the language macro is unknown, patching or generic hooks are more complicated. We therefore provide our own hooks.

```

256 \NewHookWithArguments{langselect/language/before}{1}
257 \NewReversedHook{langselect/language/after}
258 \NewHookWithArguments{langselect/argument}{2}

```

A Boolean to indicate whether more than two selectable languages have been specified.

```

259 \bool_new:c{ols_trilang}
260 \int_compare:nNnTF{\clist_count:N{\ols_langs}} > {2}{
261   \bool_set_true:c{ols_trilang}
262 }{
263   \bool_set_false:c{ols_trilang}
264 }
265 \def\olsFrstLanguage{\clist_item:Nn \ols_langs{1}}
266 \def\olsScndLanguage{\clist_item:Nn \ols_langs{2}}
267 \bool_if:cT{ols_trilang}{
268   \def\olsThrdLanguage{\clist_item:Nn \ols_langs{3}}
269 }
270 \bool_if:NnTF{ols_trim}{
271   \let\ols_trim:n=\tl_trim_spaces:n
272 }{
273   \let\ols_trim:n=\relax
274 }

```

If `\DeclarePresInitialBranchingCommand` exists (i.e., the `presitemize` environment is defined), the language macro is registered for it.

```

275 \cs_new_protected:Npn \ols_register_pres_initial_branching:cn #1#2
276 {
277   \cs_if_exist:NnTF \DeclarePresInitialBranchingCommand
278   {
279     \exp_args:Nc \DeclarePresInitialBranchingCommand {#1} {#2}
280   }
281   {
282     \AddToHook { package/osglecture-presitemize/after }
283     {
284       \exp_args:Nc \DeclarePresInitialBranchingCommand {#1} {#2}
285     }
286   }
287 }
288

```

For PDF strings, we register a simplified version of every language macro with `hyperref`. It also consumes a star, but only selects the active language branch and runs neither hooks nor replacements.

## 6 Implementation

```

289 \cs_new_protected:Npn \ols_register_pdfstring_bilingual:cn #1#2
290 {
291   \AddToHook { package/hyperref/after }
292   {
293     \pdfstringdefDisableCommands
294     {
295       \int_compare:nNnTF {#2} = {1}
296       {
297         \ExpandArgs{c}\RenewExpandableDocumentCommand {#1}
298         {s +m +m} {##2}
299       }
300       {
301         \ExpandArgs{c}\RenewExpandableDocumentCommand {#1}
302         {s +m +m} {##3}
303       }
304     }
305   }
306 }
307 \cs_new_protected:Npn \ols_register_pdfstring_trilingual:cn #1#2
308 {
309   \AddToHook { package/hyperref/after }
310   {
311     \pdfstringdefDisableCommands
312     {
313       \int_case:nn {#2}
314       {
315         {1}
316         {
317           \ExpandArgs{c}\RenewExpandableDocumentCommand {#1}
318           {s +m +m +m} {##2}
319         }
320         {2}
321         {
322           \ExpandArgs{c}\RenewExpandableDocumentCommand {#1}
323           {s +m +m +m} {##3}
324         }
325         {3}
326         {
327           \ExpandArgs{c}\RenewExpandableDocumentCommand {#1}
328           {s +m +m +m} {##4}
329         }
330       }
331     }
332   }

```

333 }

The commands `\olsMakeBilingualMacro` and `\olsMakeTrilingualMacro`, resp., are used to generate the actual language commands.

#1: prefix, #2,#3: selectable languages, #4: base name

```

334 \NewDocumentCommand{\olsMakeBilingualMacro}{0{\ols_sprefix} m m o}{
335   \str_if_eq:eeT{\olsTargetLanguage}{#2} {
336     \IfValueTF{#4}{
337       \ExpandArgs{c}\NewExpandableDocumentCommand{#1#4}{s +m +m}{
338         \UseHook{langselect/language/before}{\ols_mapped_target_lang}
339         \IfBooleanTF{##1}{
340           \ols_trim:n{\olsProcessArgs{##2}}
341         }{
342           \ols_trim:n{##2}
343           \UseHook{langselect/language/after}
344         }
345       }
346     }{
347       \ExpandArgs{c}\NewExpandableDocumentCommand{#1#2#3}{s +m +m}{
348         \UseHook{langselect/language/before}{\ols_mapped_target_lang}
349         \IfBooleanTF{##1}{
350           \ols_trim:n{\olsProcessArgs{##2}}
351         }{
352           \ols_trim:n{##2}
353           \UseHook{langselect/language/after}
354         }
355       }
356     }
357   }
358 }
359 \str_if_eq:eeT{\olsTargetLanguage}{#3}{
360   \IfValueTF{#4}{
361     \ExpandArgs{c}\NewExpandableDocumentCommand{#1#4}{s +m +m}{
362       \UseHook{langselect/language/before}{\ols_mapped_target_lang}
363       \IfBooleanTF{##1}{
364         \ols_trim:n{\olsProcessArgs{##3}}
365       }{
366         \ols_trim:n{##3}
367         \UseHook{langselect/language/after}
368       }
369     }
370   }{
371     \ExpandArgs{c}\NewExpandableDocumentCommand{#1#2#3}{s +m +m}{
372       \UseHook{langselect/language/before}{\ols_mapped_target_lang}

```

## 6 Implementation

```

373     \IfBooleanTF{##1}{
374         \ols_trim:n{\olsProcessArgs{##3}}
375     }{
376         \ols_trim:n{##3}\UseHook{langselect/language/after}
377     }
378 }
379 }
380 }
381 \IfValueTF{#4}
382 {
383     \ols_register_pres_initial_branching:cn {#1#4} {2}
384     \str_if_eq:eeTF {\olsTargetLanguage} {#2}
385     { \ols_register_pdfstring_bilingual:cn {#1#4} {1} }
386     { \ols_register_pdfstring_bilingual:cn {#1#4} {2} }
387 }
388 {
389     \ols_register_pres_initial_branching:cn {#1#2#3} {2}
390     \str_if_eq:eeTF {\olsTargetLanguage} {#2}
391     { \ols_register_pdfstring_bilingual:cn {#1#2#3} {1} }
392     { \ols_register_pdfstring_bilingual:cn {#1#2#3} {2} }
393 }
394 }

```

Now do the same thing again for *three* languages.

#1: prefix, #2...#4: selectable languages, #5: base name

```

395 \NewDocumentCommand{\olsMakeTrilingualMacro}{0{\ols_sprefix} m m m o}{
396     \str_if_eq:eeT{\olsTargetLanguage}{#2} {
397         \IfValueTF{#5}{
398             \ExpandArgs{c}\NewExpandableDocumentCommand{#1#5}{s +m +m +m}{
399                 \UseHook{langselect/language/before}{\ols_mapped_target_lang}
400                 \IfBooleanTF{##1}{
401                     \ols_trim:n{\olsProcessArgs{##2}}
402                 }{
403                     \ols_trim:n{##2}
404                     \UseHook{langselect/language/after}
405                 }
406             }
407         }{
408             \ExpandArgs{c}\NewExpandableDocumentCommand{#1#2#3#4}{s +m +m +m}{
409                 \UseHook{langselect/language/before}{\ols_mapped_target_lang}
410                 \IfBooleanTF{##1}{
411                     \ols_trim:n{\olsProcessArgs{##2}}
412                 }{
413                     \ols_trim:n{##2}

```

## 6 Implementation

```
414         \UseHook{langselect/language/after}
415     }
416 }
417 }
418 }
419 \str_if_eq:eeT{\olsTargetLanguage}{#3} {
420     \IfValueTF{#5}{
421         \ExpandArgs{c}\NewExpandableDocumentCommand{#1#5}{s +m +m +m}{
422             \UseHook{langselect/language/before}{\ols_mapped_target_lang}
423             \IfBooleanTF{##1}{
424                 \ols_trim:n{\olsProcessArgs{##3}}
425             }{
426                 \ols_trim:n{##3}
427                 \UseHook{langselect/language/after}
428             }
429         }
430     }{
431         \ExpandArgs{c}\NewExpandableDocumentCommand{#1#2#3#4}{s +m +m +m}{
432             \UseHook{langselect/language/before}{\ols_mapped_target_lang}
433             \IfBooleanTF{##1}{
434                 \ols_trim:n{\olsProcessArgs{##3}}
435             }{
436                 \ols_trim:n{##3}
437                 \UseHook{langselect/language/after}
438             }
439         }
440     }
441 }
442 \str_if_eq:eeT{\olsTargetLanguage}{#4} {
443     \IfValueTF{#5}{
444         \ExpandArgs{c}\NewExpandableDocumentCommand{#1#5}{s +m +m +m}{
445             \UseHook{langselect/language/before}{\ols_mapped_target_lang}
446             \IfBooleanTF{##1}{
447                 \ols_trim:n{\olsProcessArgs{##4}}
448             }{
449                 \ols_trim:n{##4}
450                 \UseHook{langselect/language/after}
451             }
452         }
453     }{
454         \ExpandArgs{c}\NewExpandableDocumentCommand{#1#2#3#4}{s +m +m +m}{
455             \UseHook{langselect/language/before}{\ols_mapped_target_lang}
456             \IfBooleanTF{##1}{
457                 \ols_trim:n{\olsProcessArgs{##4}}
```

## 6 Implementation

```

458     }{
459         \ols_trim:n{##4}
460         \UseHook{langselect/language/after}
461     }
462 }
463 }
464 }
465 \IfValueTF{#5}
466 {
467     \ols_register_pres_initial_branching:cn {#1#5} {3}
468     \str_if_eq:eeTF {\olsTargetLanguage} {#2}
469     { \ols_register_pdfstring_trilingual:cn {#1#5} {1} }
470     {
471         \str_if_eq:eeTF {\olsTargetLanguage} {#3}
472         { \ols_register_pdfstring_trilingual:cn {#1#5} {2} }
473         { \ols_register_pdfstring_trilingual:cn {#1#5} {3} }
474     }
475 }
476 {
477     \ols_register_pres_initial_branching:cn {#1#2#3#4} {3}
478     \str_if_eq:eeTF {\olsTargetLanguage} {#2}
479     { \ols_register_pdfstring_trilingual:cn {#1#2#3#4} {1} }
480     {
481         \str_if_eq:eeTF {\olsTargetLanguage} {#3}
482         { \ols_register_pdfstring_trilingual:cn {#1#2#3#4} {2} }
483         { \ols_register_pdfstring_trilingual:cn {#1#2#3#4} {3} }
484     }
485 }
486 }

```

If automatic generation is not disabled, the specific language commands are created. The names are  $\backslash\langle prefix \rangle\langle language\_1 \rangle\langle language\_2 \rangle$  or  $\backslash\langle prefix \rangle\langle language\_1 \rangle\langle language\_2 \rangle\langle language\_3 \rangle$ .

```

487 \bool_if:cTF{ols_generate}{
488     \bool_if:cTF{ols_trilang}{
489         \olsMakeTrilingualMacro{\olsFrstLanguage}{\olsScndLanguage}{
490             ↪ \olsThrdLanguage}
491     }{
492         \olsMakeBilingualMacro{\olsFrstLanguage}{\olsScndLanguage}
493     }
494 }{
495     \PackageInfoNoLine{\packagename}{No~language~macros~generated.\MessageBreak
496     Use~\string\olsMakeBilingualMacro~or~\string\olsMakeTrilingualMacro~to~
497     ↪ generate.}
498 }

```

For the starred variants of the language commands, we create a command that processes the subsequent arguments.

```

497 \seq_new:N \ols_args_seq
498 \int_new:N \ols_num_of_args
499 \tl_new:N \ols_format_str_tl
500 \NewDocumentCommand \olsProcessArgs { +m } {
501   \tl_gset:Nn \ols_format_str_tl{#1\UseHook{langselect/language/after}}
502   \int_zero:N \ols_num_of_args

```

Find the highest value  $n$ , where  $@n$  is contained in the format string.

```

503 \int_step_inline:nnnn { 9 } { -1 } { 1 }
504 {
505   \regex_if_match:nnT {@##1}{ #1 }
506   {
507     \int_set:Nn \ols_num_of_args { ##1 } \prg_break:
508   }
509 }

```

If no placeholder character is found, output the text in original form. Otherwise, the replacement command is called.

```

510 \int_compare:nNnTF {\ols_num_of_args} = { 0 }
511 { #1 }{
512   \seq_clear:N \ols_args_seq
513   \ols_collect_replace:n \ols_num_of_args
514 }
515 }

```

`\ols_collect_replace` receives *one* argument, namely the number of remaining replacement texts. However, it consumes *two* arguments. It obtains the second one from the token stream following the command, where it finds the next replacement text. This is stored in a token sequence and `\ols_collect_replace` is called again.

```

516 \cs_new_protected:Npn \ols_collect_replace:n #1 #2{
517   \IfHookEmptyTF{langselect/argument}{
518     \seq_put_right:Nn \ols_args_seq { #2 }
519   }{
520     \seq_put_right:Nn \ols_args_seq {
521       \UseHook{langselect/argument}{\ols_mapped_target_lang}{#2}
522     }
523   }
524   \int_compare:nNnTF { #1 } = { 1 }
525   {

```

Once all replacement texts have been read in, they are applied one after the other to the placeholder expressions.

## 6 Implementation

```

526 \seq_map_indexed_inline:Nn \ols_args_seq
527 {
528   \tl_set:Nn \ols_replacement_tl { ##2 }
529
530   \regex_replace_all:nnN
531   { @##1 }
532   { \u{ols_replacement_tl} }
533   \ols_format_str_tl
534 }
535 \ols_format_str_tl
536 }{
537   \ols_collect_replace:n { \int_eval:n { #1 - 1 } }
538 }
539 }

```

If requested, `langselect` loads Babel and sets the target language as the main language. Babel expects language names, not ISO codes. For every selectable language without a `map` entry the name is therefore determined and stored as an implicit map entry – even if `load babel` itself isn't set but `bcp47` or `load polyglossia` are active, since the BCP-47 hookup and `\setdefaultlanguage` further below then need the same entries. Only if `load babel` or `load polyglossia` is set and no name can be determined is there an error; if only `bcp47` is set, an unresolvable language is simply left alone, and Babel decides as it would without `langselect`.

```

540 \tl_new:N \ols_babel_name_tl
541 \tl_new:N \ols_babel_opts_tl
542 \bool_if:nT { \ols_load_babel || \ols_load_polyglossia || \ols_bcptags } {
543   \clist_map_inline:Nn \ols_langs {
544     \cs_if_exist:cF{ olslangmap / \tl_to_str:n {#1} }{
545       \ols_babel_name:nN {#1} \ols_babel_name_tl
546       \tl_if_empty:NTF \ols_babel_name_tl {
547         \bool_if:nT { \ols_load_babel || \ols_load_polyglossia } {
548           \PackageError{\packagename}{Can't find a language name for '~#1'}{
549             Add a 'map' entry such as map={#1=...} to the package options,
550             \MessageBreak
551             ↪ or load babel/polyglossia yourself before ~\packagename.}
552         }
553       }{
554         \cs_gset:cpe { olslangmap / \tl_to_str:n {#1} } { \ols_babel_name_tl }
555       }
556     }
557   }
558 }

```

Recompute `\ols_mapped_target_lang` after the new map entries – @1 and @2 below both need the updated value. `\selectlanguage\setdefaultlanguage`

```

557 \edef\ols_mapped_target_lang{\ols_use_map:n{\olsTargetLanguage}}

```

```

558 }
559 \bool_if:NT \ols_load_babel{

```

Load all selectable languages under their Babel names, the target as the main language.

```

560   \tl_clear:N \ols_babel_opts_tl
561   \clist_map_inline:Nn \ols_langs
562     { \tl_put_right:Ne \ols_babel_opts_tl { \ols_use_map:n {#1} , } }
563   \tl_put_right:Ne \ols_babel_opts_tl { main = \ols_mapped_target_lang }
564   \exp_args:NV \PassOptionsToPackage \ols_babel_opts_tl { babel }
565   \RequirePackage { babel }
566   \bool_if:nT{
567     \ols_shorthands && !\str_if_eq_p:ee{\olsTargetLanguage}{de}}
568   {
569     \languageshorthands{ngerman}
570   }
571 }
572 \bool_if:NT\ols_load_polyglossia{
573   \RequirePackage{polyglossia}
574 }

```

For the shorthand logic, we need csquotes.

```

575 \bool_if:NT\ols_shorthands {
576   \RequirePackage[autostyle=true]{csquotes}
577 }

```

The initialization below can only happen once it is determined which language package (if any) is ultimately active. Since other packages or classes typically defer language decisions, everything is deferred using `\AddToHook[begindocument/before]` and `\AddToHook[begindocument]`, respectively.

```

578 \AddToHook{begindocument/before}{%

```

Activate BCP 47 before language selection so that `\selectlanguage` can resolve the code.

```

579   \bool_lazy_and:nnT{\ols_bcptags}{\cs_if_exist_p:N\babeladjust}{

```

`\bbl@bcp@map@⟨Code⟩` is pre-populated for every selectable language with a name determined (above) before `autoload.bcp47` is enabled. Its own tag resolution checks `\bbl@bcp@map@⟨Code⟩` first and uses it unchanged if present (in this way, language selection does not depend on Babel's own, release-dependent BCP-47 database). For languages that could not be determined there, `autoload.bcp47`'s own behavior is unchanged.

```

580   \clist_map_inline:Nn \ols_langs {
581     \cs_if_exist:cT{ olslangmap / \tl_to_str:n {#1} }{
582       \cs_gset:cpe { bbl@bcp@map@ #1 } { \ols_use_map:n {#1} }
583     }

```

```

584     }
585     \babeladjust{ bcp47.toname = on , autoload.bcp47 = on }
586   }
587   \bool_if:NT\ols_shorthands {

```

The shorthand macros need an active @.

```

588     \makeatletter

```

Babel's shorthand infrastructure is already available with Babel; with Polyglossia it has to be initialized explicitly, and without a language package it has to be loaded directly.

```

589     \@ifpackageloaded{babel}{}{%
590       \@ifpackageloaded{polyglossia}
591       {\InitializeBabelShorthands}
592       {\input{babelsh.def}}}%
593     }
594     \@ifpackageloaded{babel}
595     {\usesshorthands*{"}}
596     {\usesshorthands{"}}
597     \defineshorthand{"`"}{\openautoquote}
598     \defineshorthand{"'"}{\closeautoquote}
599     \defineshorthand{" /"}{\textormath{\bbl@allowhyphens
600       \discretionary{/}{/}{/}\bbl@allowhyphens}{}}
601     \makeatother
602   }
603 }

```

Language selection happens in the begindocument hook, i.e. *after* the language package has activated its main language.

```

604 \AddToHook{begindocument}{%
605   \bool_if:NF\ols_target_from_langpkg{
606     \cs_if_exist:cT{ver@babel.sty}{
607       \selectlanguage{\ols_mapped_target_lang}
608     }
609     \cs_if_exist:cT{ver@polyglossia.sty}{
610       \setdefaultlanguage{\ols_mapped_target_lang}
611     }
612   }
613 }

```

# Index

|                                                        |                    |
|--------------------------------------------------------|--------------------|
| <b>A</b>                                               |                    |
| <code>auto</code> .....                                | 5 f.               |
| <b>B</b>                                               |                    |
| <code>babel</code> (package) .....                     | 14 ff., 18         |
| <code>bcp47</code> .....                               | 6, 9, 26           |
| <code>bicaption</code> (package) .....                 | 3                  |
| <b>C</b>                                               |                    |
| <code>\closeautoquote</code> .....                     | 9                  |
| <code>comment</code> (package) .....                   | 3                  |
| <code>csquotes</code> (package) .....                  | 6, 9, 27           |
| CTAN .....                                             | 3 f., 9            |
| <b>D</b>                                               |                    |
| <code>\DeclarePresInitialBranchingCommand</code> ..... | 19                 |
| <code>\DocumentMetadata</code> .....                   | 2, 5, 16, 18       |
| <b>E</b>                                               |                    |
| EIJKHOUT, Victor .....                                 | 3                  |
| <code>\enquote</code> .....                            | 9                  |
| <b>F</b>                                               |                    |
| <code>fontenc</code> (package) .....                   | 10                 |
| <code>fontspec</code> (package) .....                  | 10                 |
| <code>\foreignlanguage</code> .....                    | 4, 9               |
| <b>G</b>                                               |                    |
| GREWE, Richard .....                                   | 3                  |
| <b>H</b>                                               |                    |
| <code>hyperref</code> (package) .....                  | 8, 19              |
| <b>I</b>                                               |                    |
| <code>\include</code> .....                            | 2                  |
| <b>L</b>                                               |                    |
| <code>langselect</code> (package) .....                | 8, 10              |
| <code>languages</code> .....                           | 5, 14              |
| <code>load babel</code> .....                          | 5, 9, 26           |
| <code>load polyglossia</code> .....                    | 5, 9, 26           |
| <b>M</b>                                               |                    |
| <code>main</code> .....                                | 9                  |
| <code>\makeatletter</code> .....                       | 10                 |
| <code>map</code> .....                                 | 5 f., 9, 26        |
| <code>multilang</code> (package) .....                 | 3                  |
| <b>O</b>                                               |                    |
| <code>\ols_babel_name:nN</code> .....                  | 13                 |
| <code>\ols_babel_name_from_ini:nN</code> .....         | 11 f.              |
| <code>\ols_babel_name_from_polyglossia:nN</code> ..... | 12                 |
| <code>\ols_collect_replace</code> .....                | 25                 |
| <code>\ols_mapped_target_lang</code> .....             | 18, 26             |
| <code>\olsIsoTags</code> .....                         | 11                 |
| <code>\olsMakeBilingualMacro</code> .....              | 8, 21              |
| <code>\olsMakeTrilingualMacro</code> .....             | 8, 21              |
| <code>\olsTargetLanguage</code> .....                  | 2, 4 f., 9, 14, 16 |
| <code>\openautoquote</code> .....                      | 9                  |
| <code>osglecture</code> (bundle) .....                 | 9 f.               |
| <b>P</b>                                               |                    |
| <code>polyglossia</code> (package) .....               | 14 f., 18          |
| <code>prefix</code> .....                              | 5, 8               |
| <code>presitemize</code> (environment) .....           | 19                 |
| <b>S</b>                                               |                    |
| <code>\selectlanguage</code> .....                     | 4, 6, 26 f.        |
| <code>\setdefaultlanguage</code> .....                 | 26                 |
| <b>T</b>                                               |                    |
| <code>targetlang</code> .....                          | 4 f., 9, 14, 16    |
| <code>translations</code> (package) .....              | 3                  |
| <code>translator</code> (package) .....                | 3                  |
| <code>trim</code> .....                                | 5                  |
| <b>U</b>                                               |                    |
| <code>unified shorthands</code> .....                  | 6, 9               |
| <code>\usepackage</code> .....                         | 2, 4               |
| <b>V</b>                                               |                    |
| <code>\varsfromjobname</code> .....                    | 11                 |
| <code>varsfromjobname</code> (package) .....           | 4                  |
| <b>X</b>                                               |                    |
| <code>xtcpts</code> (package) .....                    | 3                  |

# Index

|                                                        |                    |
|--------------------------------------------------------|--------------------|
| <b>A</b>                                               |                    |
| <code>auto</code> .....                                | 5 f.               |
| <b>B</b>                                               |                    |
| <code>babel</code> (package) .....                     | 14 ff., 18         |
| <code>bcp47</code> .....                               | 6, 9, 26           |
| <code>bicaption</code> (package) .....                 | 3                  |
| <b>C</b>                                               |                    |
| <code>\closeautoquote</code> .....                     | 9                  |
| <code>comment</code> (package) .....                   | 3                  |
| <code>csquotes</code> (package) .....                  | 6, 9, 27           |
| CTAN .....                                             | 3 f., 9            |
| <b>D</b>                                               |                    |
| <code>\DeclarePresInitialBranchingCommand</code> ..... | 19                 |
| <code>\DocumentMetadata</code> .....                   | 2, 5, 16, 18       |
| <b>E</b>                                               |                    |
| EIJKHOUT, Victor .....                                 | 3                  |
| <code>\enquote</code> .....                            | 9                  |
| <b>F</b>                                               |                    |
| <code>fontenc</code> (package) .....                   | 10                 |
| <code>fontspec</code> (package) .....                  | 10                 |
| <code>\foreignlanguage</code> .....                    | 4, 9               |
| <b>G</b>                                               |                    |
| GREWE, Richard .....                                   | 3                  |
| <b>H</b>                                               |                    |
| <code>hyperref</code> (package) .....                  | 8, 19              |
| <b>I</b>                                               |                    |
| <code>\include</code> .....                            | 2                  |
| <b>L</b>                                               |                    |
| <code>langselect</code> (package) .....                | 8, 10              |
| <code>languages</code> .....                           | 5, 14              |
| <code>load babel</code> .....                          | 5, 9, 26           |
| <code>load polyglossia</code> .....                    | 5, 9, 26           |
| <b>M</b>                                               |                    |
| <code>main</code> .....                                | 9                  |
| <code>\makeatletter</code> .....                       | 10                 |
| <code>map</code> .....                                 | 5 f., 9, 26        |
| <code>multilang</code> (package) .....                 | 3                  |
| <b>O</b>                                               |                    |
| <code>\ols_babel_name:nN</code> .....                  | 13                 |
| <code>\ols_babel_name_from_ini:nN</code> .....         | 11 f.              |
| <code>\ols_babel_name_from_polyglossia:nN</code> ..... | 12                 |
| <code>\ols_collect_replace</code> .....                | 25                 |
| <code>\ols_mapped_target_lang</code> .....             | 18, 26             |
| <code>\olsIsoTags</code> .....                         | 11                 |
| <code>\olsMakeBilingualMacro</code> .....              | 8, 21              |
| <code>\olsMakeTrilingualMacro</code> .....             | 8, 21              |
| <code>\olsTargetLanguage</code> .....                  | 2, 4 f., 9, 14, 16 |
| <code>\openautoquote</code> .....                      | 9                  |
| <code>osglecture</code> (bundle) .....                 | 9 f.               |
| <b>P</b>                                               |                    |
| <code>polyglossia</code> (package) .....               | 14 f., 18          |
| <code>prefix</code> .....                              | 5, 8               |
| <code>presitemize</code> (environment) .....           | 19                 |
| <b>S</b>                                               |                    |
| <code>\selectlanguage</code> .....                     | 4, 6, 26 f.        |
| <code>\setdefaultlanguage</code> .....                 | 26                 |
| <b>T</b>                                               |                    |
| <code>targetlang</code> .....                          | 4 f., 9, 14, 16    |
| <code>translations</code> (package) .....              | 3                  |
| <code>translator</code> (package) .....                | 3                  |
| <code>trim</code> .....                                | 5                  |
| <b>U</b>                                               |                    |
| <code>unified shorthands</code> .....                  | 6, 9               |
| <code>\usepackage</code> .....                         | 2, 4               |
| <b>V</b>                                               |                    |
| <code>\varsfromjobname</code> .....                    | 11                 |
| <code>varsfromjobname</code> (package) .....           | 4                  |
| <b>X</b>                                               |                    |
| <code>xtcpts</code> (package) .....                    | 3                  |