

# translations

V1.0      2013/07/16

a simple translator

Clemens NIEDERBERGER

<https://github.com/cgnieder/translations/>  
[contact@mychemistry.eu](mailto:contact@mychemistry.eu)

English documentation

## Contents

		3.4.2    The ‘fallback’ language . . . . .	6
		3.5    Dictionaries . . . . .	6
		3.5.1    Own Dictionaries . . . . .	6
		3.5.2 <b>TRANSLATIONS</b> ’ Basic Dictionaries . . . . .	7
<b>1</b>	<b>Motivation</b>	<b>1</b>	
<b>2</b>	<b>License and Requirements</b>	<b>2</b>	
<b>3</b>	<b>Usage</b>	<b>2</b>	
3.1	Background . . . . .	2	
3.2	Available Commands . . . . .	3	
3.3	A Small Example . . . . .	4	
3.4	Usage in Packages . . . . .	5	
3.4.1	Basic Structure . . . . .	5	
		<b>4    Defined Languages</b>	<b>9</b>
		<b>5    Implementation</b>	<b>10</b>
		<b>Index</b>	<b>23</b>

## 1 Motivation

This package provides means for package authors to have an easy interface for internationalization of their packages. The functionality of this package is in many parts also covered by the package translator (part of the beamer<sup>1</sup> bundle). Internationalization is also possible with babel<sup>2</sup> and it’s `\addto\captions<language>` mechanism or KOMA-Script’s `\providecaptionname`. However, I believe that **TRANSLATIONS** is more flexible than all of these. Unlike translator it detects the used (babel or polyglossia<sup>3</sup>) language itself and provides expandable retrieving of the translated key. **TRANSLATIONS** also provides support for language dialects which means package authors can for example distinguish between British, Australian, Canadian and US English.

The first draft of the package was written since I missed an expandable version of translator’s `\translate` command. Once I had the package available I began using it in various of my other packages so it got extended to the needs I faced there.

---

<sup>1</sup> on CTAN: beamer    <sup>2</sup> on CTAN: babel    <sup>3</sup> on CTAN: polyglossia

## 2 License and Requirements

**TRANSLATIONS** is placed under the terms of the L<sup>A</sup>T<sub>E</sub>X Project Public License, version 1.3 or later (<http://www.latex-project.org/lppl.txt>). It has the status “maintained.”

**TRANSLATIONS** requires the packages etoolbox<sup>4</sup> and scrfile (part of the KOMA-Script bundle<sup>5</sup>).

## 3 Usage

### 3.1 Background

The **TRANSLATIONS** package enables the author of a package or a class (or a document) to declare translations in different languages of key words and fetch these translations in the document depending on the active language as set by babel or polyglossia. Since **TRANSLATIONS** checks which language is active it is generally not necessary to specify the language for which a translation should be fetched manually.

**TRANSLATIONS** knows of three types of languages: main languages (see table 2 on page 9), language aliases (see table 3 on page 9) and language dialects (see table 4 on page 10). For the commands declaring or fetching a translation base languages and language aliases are equivalent. Dialects are similar to aliases but there are a few important differences.

Figure 1 shows what happens if **TRANSLATIONS** is asked to fetch a translation for a given key.

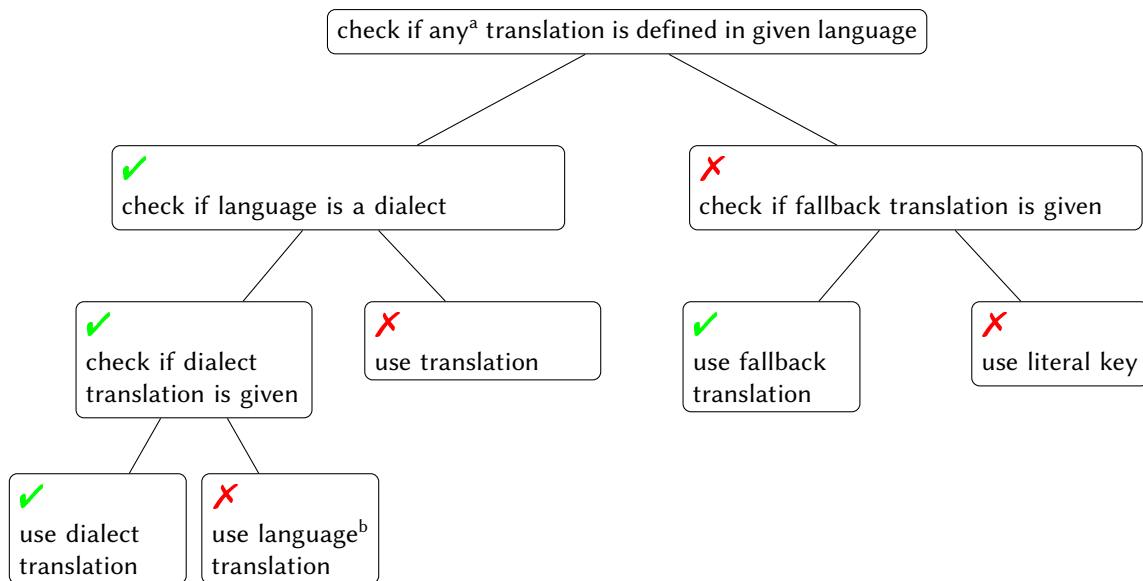


FIGURE 1: Schematic representation of **TRANSLATIONS**’ translating mechanism. Notes: <sup>a</sup> except for a possible fallback translation. <sup>b</sup> i.e., the base language of the dialect.

What happens if you declare a translation? There are three cases:

1. You declare a translation for a base language: this is the normal case where an internal macro is defined which can be fetched by the `\GetTranslation` command (see section 3.2).

<sup>4</sup> on CTAN: etoolbox    <sup>5</sup> on CTAN: koma-script

2. You declare a translation for a language alias: this is the very same as the first case since the same internal macro is defined.
3. You declare a translation for a dialect: this is two-fold. Either a translation for the base language exists so only the translation for the dialect is saved. If the translation for the base language does not exist it is defined to be the same as the one for the dialect.

### 3.2 Available Commands

Below the commands provided by **TRANSLATIONS** are explained. The symbol ★ means that the command is expandable, ★ means that it isn't.

- ★ **\DeclareLanguage**{<lang>}  
Declare a language that can be used by **TRANSLATIONS**. If the language already exists it will be silently redefined. This command can only be used in the preamble.
- ★ **\DeclareLanguageAlias**{<lang2>}{<lang1>}  
Declares <lang2> to be an alias of <lang1>. If <lang1> doesn't exist yet a warning will be raised and it will be defined. This command can only be used in the preamble.
- ★ **\DeclareLanguageDialect**{<dialect>}{<lang>}  
Declares <dialect> to be a dialect of language <lang>. If a translation for <dialect> is provided it is used by the translation macros. If there is none the corresponding translation for <lang> is used instead.
- ★ **\NewTranslation**{<lang>}{<key>}{<translation>}  
Defines a translation of key <key> for the language <lang>. An error will be raised if a translation of <key> already exists. This command can only be used in the preamble.
- ★ **\RenewTranslation**{<lang>}{<key>}{<translation>}  
Redefines a translation of key <key> for the language <lang>. An error will be raised if no translation of <key> exists. This command can only be used in the preamble.
- ★ **\DeclareTranslation**{<lang>}{<key>}{<translation>}  
Defines a translation of key <key> for the language <lang>. No error will be raised if a translation of <key> already exists. This command can only be used in the preamble.
- ★ **\DeclareTranslationFallback**{<key>}{<fallback>}  
Defines a fallback translation for key <key> that is used in case no translation of <key> for the currently active language has been provided. No error will be raised if a fallback for <key> already exists. This command can only be used in the preamble.
- ★ **\GetTranslationFor**{<lang>}{<key>}  
Fetches and prints the translation of <key> for the language <lang>. This command is expandable.
- ★ **\GetTranslation**{<key>}  
Fetches and prints the translation of <key> for the currently active language (as for example set by babel). This command is expandable.

★ `\GetTranslationForWarn{<lang>}{<key>}`

Introduced in version 1.0 Fetches and prints the translation of <key> for the language <lang>. Issues a warning if no translation is available.

★ `\GetTranslationWarn{<key>}`

Introduced in version 1.0 Fetches and prints the translation of <key> for the currently active language (as for example set by babel). Issues a warning if no translation is available.

★ `\SaveTranslationFor{<cmd>}{<lang>}{<key>}`

Fetches and saves the translation of <key> for the language <lang> in the macro <cmd>.

★ `\SaveTranslation{<cmd>}{<key>}`

Fetches and saves the translation of <key> for the currently active language (as for example set by babel) in the macro <cmd>.

★ `\LoadDictionary{<name>}`

Loads a file named <name>-<lang>.trsl where <lang> corresponds to the lowercase name of the current language as defined with `\DeclareLanguage`. This file should contain the translations for the specified language.

★ `\LoadDictionaryFor{<lang>}{<name>}`

Loads a file named <name>-<lang>.trsl.

★ `\DeclareDictTranslation{<key>}{<translation>}`

This command is to be used in a dictionary file and picks up the language of that file, see section 3.5 for an example.

★ `\ProvideDictionaryFor{<lang>}{<name>}[<date>]`

Needs to be in a dictionary file. This command tells **TRANSLATIONS** that the file indeed is a dictionary and also sets the language for the dictionary which is used by `\DeclareDictTranslation`.

★ `\PrintDictionaryFor{<lang>}{<name>}{<pre>}{<mid>}{<post>}`

Introduced in version 1.0 Prints all entries of dictionary <name> in language <lang> in the order the entries have been declared. For every entry the code

```
<pre><key><mid><translation><post>
```

is printed. The dictionary must have been loaded of course.

### 3.3 A Small Example

This section demonstrates with two short examples how the macros are used. The first example covers the basics: declaring of translations and then retrieving and typesetting them.

```

1 % in the preamble:
2 % \DeclareTranslation{English}{Kueche}{kitchen}
3 % \DeclareTranslation{German}{Kueche}{K\"uche}
4 % \DeclareTranslation{Spanish}{Kueche}{cocina}
5 % \DeclareTranslation{French}{Kueche}{cuisine}
6
7 \GetTranslation{Kueche}
8 \SaveTranslation\kitchen{Kueche}
9 \SaveTranslationFor\cuisine{french}{Kueche}
10
11 \selectlanguage{ngerman}
12 \GetTranslation{Kueche} \kitchen\ \GetTranslationFor{spanish}{Kueche}
13 \cuisine

kitchen
Küche kitchen cocina cuisine

```

The next example demonstrates the use of dialects and how they fall back to the translation for the main language if no extra translation was declared:

```

1 % in the preamble:
2 % \DeclareTranslation{English}{farbe}{color}
3 % \DeclareTranslation{British}{farbe}{colour}
4
5 \GetTranslationFor{English}{farbe} \
6 \GetTranslationFor{British}{farbe} \
7 \GetTranslationFor{American}{farbe}

color
colour
color

```

### 3.4 Usage in Packages

#### 3.4.1 Basic Structure

A typical usage in a package would look as follows:

```

1 \RequirePackage{translations}
2 \DeclareTranslationFallback{mypackage-title}{Nice Title}
3 \DeclareTranslation{English}{mypackage-title}{Nice Title}
4 \DeclareTranslation{French}{mypackage-title}{Beau Titre}
5 \DeclareTranslation{German}{mypackage-title}{Sch\"o{ner Titel}
6 ...
7 \def\mypackage@title{\GetTranslation{mypackage-title}}

```

That is, a package defines some unique key for an expression and at least defines a fallback translation. Additionally translations for as many languages as the author wants are defined. A user then may add `\DeclareTranslation{<language>}{<translation>}` if they find their translation missing.

### 3.4.2 The ‘fallback’ language

If a user has neither loaded `babel` nor `polyglossia` `TRANSLATIONS` will use English as language and translate to English if the translation was provided. If the user *has* loaded one of the language packages but has chosen a language for which no translation is defined the language ‘fallback’ will be used, i.e., the translation provided with `\DeclareTranslationFallback`. If no fallback translation is provided either the translation will expand to the literal string.

The following three examples should make this concept clear:

```

1 \documentclass{article}
2 \DeclareTranslation{German}{foo-literal}{bar}
3 \begin{document}
4 \GetTranslation{foo-literal} => `foo-literal'
5 \end{document}

```

```

1 \documentclass{article}
2 \DeclareTranslationFallback{foo-literal}{foo}
3 \DeclareTranslation{German}{foo-literal}{bar}
4 \begin{document}
5 \GetTranslation{foo-literal} => `foo'
6 \end{document}

```

```

1 \documentclass{article}
2 \usepackage[ngerman]{babel}
3 \DeclareTranslation{German}{foo-literal}{bar}
4 \begin{document}
5 \GetTranslation{foo-literal} => `bar'
6 \end{document}

```

## 3.5 Dictionaries

### 3.5.1 Own Dictionaries

A typical dictionary file should look as follows:

```

1 % this is file housing-german.trsl
2 \ProvideDictionaryFor{German}{housing}[-<version info>]
3 \DeclareDictTranslation{kitchen (housing)}{K\"uche}
4 \DeclareDictTranslation{bathroom (housing)}{Bad}
5 \DeclareDictTranslation{living room (housing)}{Wohnzimmer}
6 \DeclareDictTranslation{bedroom (housing)}{Schlafzimmer}
7 ...
8 \endinput

```

The usage is similar to the one in a package: unique keys are given translations, this time for the language the dictionary file is declared for only.

### 3.5.2 TRANSLATIONS' Basic Dictionaries

**TRANSLATIONS** already provides a basic dictionary for the languages English, French, German and Spanish. This dictionary is loaded automatically if the document language is one of these four. If you'd like to contribute and add the basic dictionary in your language this is more than welcome and highly appreciated! The easiest way to do this would be to copy one of the existing files `translations-basic-dictionary-<lang>.trsl` and modify the file accordingly. You can then send me the file via email and I'll add it to **TRANSLATIONS**.

Table 1 lists all words provided by the basic dictionary for German.

TABLE 1: All entries of **TRANSLATIONS**' basic dictionary in German.

key	translation
Abstract	Zusammenfassung
Addresses	Adressen
addresses	Adressen
Address	Adresse
address	Adresse
and	und
Appendix	Anhang
Authors	Autoren
authors	Autoren
Author	Autor
author	Autor
Bibliography	Literaturverzeichnis
cc	Verteiler
Chapters	Kapitel
chapters	Kapitel
Chapter	Kapitel
chapter	Kapitel
Conclusion	Zusammenfassung
conclusion	Zusammenfassung
Contents	Inhaltsverzeichnis
Continuation	Fortsetzung
continuation	Fortsetzung
cont	Forts
encl (plural)	Anlagen
encl (singular)	Anlage
encl	Anlage(n)
Figures	Abbildungen
figures	Abbildungen
Figure	Abbildung

*continues*

key	translation
figure	Abbildung
From	Von
from	von
Glossary	Glossar
Index	Index
Introduction	Einleitung
introduction	Einleitung
List of Figures and Tables	Abbildungs- und Tabellenverzeichnis
List of Figures	Abbildungsverzeichnis
List of Tables	Tabellenverzeichnis
or	oder
Outline	Gliederung
Overview	Übersicht
Pages	Seiten
pages	Seiten
Page	Seite
page	Seite
Paragraphs	Absätze
paragraphs	Absätze
Paragraph	Absatz
paragraph	Absatz
Parts	Teile
parts	Teile
Part	Teil
part	Teil
Preface	Vorwort
Proofs	Beweise
proofs	Beweise
Proof	Beweis
proof	Beweis
References	Literatur
Related work	Verwandte Arbeiten
Related Work	Verwandte Arbeiten
Sections	Abschnitte
sections	Abschnitte
Section	Abschnitt
section	Abschnitt
See also	Siehe auch
see also	siehe auch
See	Siehe
see	siehe
Sketch of Proofs	Beweisskizzen
Sketch of proofs	Beweisskizzen

*continues*



key	translation
Sketch of Proof	Beweisskizze
Sketch of proof	Beweisskizze
Subsections	Unterabschnitte
subsections	Unterabschnitte
Subsection	Unterabschnitt
subsection	Unterabschnitt
Summary	Zusammenfassung
Tables	Tabellen
tables	Tabellen
Table	Tabelle
table	Tabelle
To	An
to	an

## 4 Defined Languages

Quite a number of languages already are defined, either directly or via an alias. So, before you define a language you should take a look at the tables below if the language doesn't already exist. Table 2 lists all base languages, "fallback" being a dummy language used for fallback translations.

TABLE 2: Base languages defines by **TRANSLATIONS**.

fallback	albanian	bulgarian	catalan
croatian	czech	danish	dutch
english	finnish	french	german
greek	hebrew	hungarian	icelandic
italian	norwegian	polish	portuges
romanian	russian	serbocroatian	slovak
slovenian	spanish	swedish	turkish
ukrainian	canadien	american	australian
british	canadian	austrian	naustrian
magyar	brazil	swissgerman	

To every one of these languages at least one alias exists, the uppercase variant. This is due to the fact that it is common to write language names uppercased. All defined aliases are listed in table 3.

TABLE 3: All language aliases defined by **TRANSLATIONS**.

alias	language	alias	language
Fallback	fallback	Albanian	albanian
Bulgarian	bulgarian	Catalan	catalan

*continues*

## 5 Implementation

alias	language	alias	language
Croatian	croatian	Czech	czech
Danish	danish	Dutch	dutch
Finnish	finnish	francais	french
Francais	francais	Canadien	canadien
French	french	American	american
Australian	australian	British	british
Canadian	canadian	English	english
UKenglish	british	USenglish	american
Austrian	austrian	German	german
germanb	german	ngerman	german
Greek	greek	polutonikogreek	greek
Hebrew	hebrew	Hungarian	hungarian
Magyar	magyar	Icelandic	icelandic
Italian	italian	norsk	norwegian
Norsk	norsk	Norwegian	norwegian
nynorsk	norwegian	Nynorsk	nynorsk
Polish	polish	Brazil	brazil
brazilian	brazil	Brazilian	brazilian
Portuges	portuges	portuguese	portuges
Portuguese	portuguese	Romanian	romanian
Russian	russian	Serbocroatian	serbocroatian
Slovak	slovak	Slovenian	slovenian
Spanish	spanish	Swedish	swedish
Swiss	swissgerman	Swissgerman	swissgerman
Turkish	turkish	Ukrainian	ukrainian

**TRANSLATIONS** also defines a few dialects. They are listed in table 4.

TABLE 4: All dialects defined by **TRANSLATIONS**.

dialect	language	dialect	language
canadien	french	american	english
australian	english	british	english
canadian	english	austrian	german
naustrian	austrian	magyar	hungarian
brazil	portuges	swissgerman	german

These languages should cover all languages which are currently covered by babel and polyglossia.

## 5 Implementation

In the following code the lines 1–30 have been omitted. They only repeat the license statement which has already been mentioned in section 2.

## 5 Implementation

```

31 \def\@trnslt@date{2013/07/16}
32 \def\@trnslt@version{v1.0}
33
34 \ProvidesPackage{translations}[\@trnslt@date\space \@trnslt@version\space a simple
    translator]
35 \RequirePackage{etoolbox,scrfile}
36
37 % -----
38 % message handling
39 \def\@trnslt@error@message{%
40     For details have a look at the `translations' manual.}
41
42 \def\@trnslt@create@message#1{%
43     \ifstrequal{#1}{Error}
44     {%
45         \lowercase{\csdef{\@trnslt@#1}}##1{%
46             \csuse{Package#1}{translations}{##1}{\@trnslt@error@message}}%
47     }{%
48         \lowercase{\csdef{\@trnslt@#1}}##1{%
49             \csuse{Package#1}{translations}{##1}}%
50     }}
51 \@trnslt@create@message{Error}
52 \@trnslt@create@message{Warning}
53 \@trnslt@create@message{WarningNoLine}
54 \@trnslt@create@message{Info}
55
56 \def\@trnslt@err@unknown@lang#1{%
57     \@trnslt@error{Unknown language `#1'}}
58
59 \def\@trnslt@warn@unknown@lang#1{%
60     \@trnslt@warning{Unknown language `#1'}}
61
62 \def\@trnslt@err@already@defined#1#2{%
63     \@trnslt@error{The #2 translation for `#1' is already defined.}}
64
65 \def\@trnslt@err@not@defined#1#2{%
66     \@trnslt@error{The \@trnslt@language{#2} translation for `#1' is not defined yet.}}
67
68 % -----
69 % check if babel or polyglossia is used
70 \AtEndPreamble{
71     \ifpackageloaded{babel}{%
72         \ifpackageloaded{polyglossia}{%
73             {\@trnslt@warning{No language package found. I am going to use `english'
74                 as default language.}}
75         }
76     \ifdef\language{}
77         {\def\language{english}}
78     \def\@trnslt@current@language{\language}
79     \ifdef\bb@afterfi{}
80         {\long\def\bb@afterfi#1\fi{\fi#1}}
81 }

```

## 5 Implementation

```

82
83 % -----
84 % book keeping: the following macros will be used as `etoolbox' lists that
85 % keep record of defined languages, dialects and aliases
86 \def\@trnslt@languages{}% all languages
87 \def\@trnslt@aliases@pair{}% all aliases and their base
88 \def\@trnslt@aliases@single{}% all aliases
89 \def\@trnslt@dialects@pair{}% all dialects and their base
90 \def\@trnslt@dialects@single{}% all dialects
91
92 % -----
93 % define \@trnslt@if@<name> conditionals that don't leave the checked macro as
94 % \relax behind and check for \@trnslt@<name>@#1. These conditionals should
95 % also be expandable in an \edef-like context:
96 \def\@trnslt@newif#1{%
97   \csdef{@trnslt@if@#1}##1{%
98     \expandafter\expandafter\expandafter\expandafter
99     \expandafter\expandafter\expandafter
100     \@firstofone
101     {\expandafter\expandafter\expandafter}%
102     \ifcsname @trnslt@#1@##1\endcsname
103     \expandafter\@firstoftwo
104     \else
105     \expandafter\@secondoftwo
106     \fi
107   }%
108 }
109
110 % -----
111 % \DeclareLanguage
112 % #1: language
113 \newrobustcmd*\DeclareLanguage[1]{%
114   \@trnslt@declare@language{#1}}
115 \@onlypreamble\DeclareLanguage
116
117 \def\@trnslt@declare@language#1{%
118   \@trnslt@if@language{#1}
119   }{%
120     \csdef{@trnslt@language@#1}{#1}%
121     \listeadadd\@trnslt@languages{#1}%
122   }%
123 }
124
125 \def\@trnslt@language#1{%
126   \csuse{@trnslt@language@#1}}
127
128 \@trnslt@newif{language}
129
130 % -----
131 % \DeclareLanguageDialect
132 % #1: dialect
133 % #2: language

```

## 5 Implementation

```

134 \newrobustcmd*\DeclareLanguageDialect[2]{%
135   \@trnslt@declare@language@dialect{#1}{#2}}
136 \@onlypreamble\DeclareLanguageDialect
137
138 \def\@trnslt@declare@language@dialect#1#2{%
139   \@trnslt@if@language{#2}
140   {}{%
141     \@trnslt@warn@unknown@lang{#2}%
142     \@trnslt@declare@language{#2}%
143   }%
144   \@trnslt@if@dialect{#1}
145   {% => ist schon als dialect definiert => irgendwelche weiteren checks?
146   }
147   {%
148     \@trnslt@if@alias{#2}
149     {%
150       \csedef{\@trnslt@dialect@#1}{\@trnslt@alias{#2}}{#1}}%
151       \@trnslt@declare@language{#1}%
152       \listead\@trnslt@dialects@single{#1}%
153       \listead\@trnslt@dialects@pair{{#1}{\@trnslt@alias{#2}}}%
154     }
155     {%
156       \csdef{\@trnslt@dialect@#1}{{#2}{#1}}%
157       \@trnslt@declare@language{#1}%
158       \listead\@trnslt@dialects@single{#1}%
159       \listead\@trnslt@dialects@pair{{#1}{#2}}%
160     }%
161   }%
162 }
163
164 \def\@trnslt@dialect#1{%
165   \csuse{\@trnslt@dialect@#1}}
166
167 % this macros fetches the base language for a given dialect, expandably:
168 \def\@trnslt@dialect@of#1{%
169   \@trnslt@if@dialect{#1}
170   {%
171     \expandafter\expandafter\expandafter
172     \@firstoftwo
173     \csname \@trnslt@dialect@#1\endcsname
174   }{%
175   }
176
177 \@trnslt@newif{dialect}
178
179 % -----
180 % \DeclareLanguageAlias
181 % #1: alias
182 % #2: language
183 \newrobustcmd*\DeclareLanguageAlias[2]{%
184   \@trnslt@declare@language@alias{#1}{#2}}
185 \@onlypreamble\DeclareLanguageAlias

```

## 5 Implementation

```

186
187 \def\@trnslt@declare@languagealias#1#2{%
188   \@trnslt@if@language{#2}
189     {}{%
190       \@trnslt@warn@unknown@lang{#2}%
191       \@trnslt@declare@language{#2}%
192     }%
193   \csletcs{@trnslt@language@#1}{@trnslt@language@#2}%
194   \@trnslt@if@dialect{#2}
195     {\csletcs{@trnslt@dialect@#1}{@trnslt@dialect@#2}}
196     {}%
197   \ifinlist{#1}\@trnslt@aliases@single
198     {}{%
199       \csdef{@trnslt@alias@#1}{#2}%
200       \listadd\@trnslt@aliases@pair{{#1}{#2}}%
201       \listadd\@trnslt@aliases@single{#1}%
202     }%
203 }
204
205 \def\@trnslt@alias#1{%
206   \csuse{@trnslt@alias@#1}}
207
208 \@trnslt@newif{alias}
209
210 % -----
211 % dummy language: `fallback':
212 \DeclareLanguage{fallback}
213 \DeclareLanguageAlias{Fallback}{fallback}
214
215 % -----
216 % \DeclareTranslation, \NewTranslation and \RenewTranslation
217 % #1: language
218 % #2: word
219 % #3: replacement
220 \newrobustcmd*\DeclareTranslation[3]{%
221   \@trnslt@declare@translation{#2}{#1}{#3}}
222 \@onlypreamble\DeclareTranslation
223
224 \newrobustcmd*\DeclareTranslationFallback[2]{%
225   \@trnslt@declare@translation{#1}{fallback}{#2}}
226 \@onlypreamble\DeclareTranslationFallback
227
228 \newrobustcmd*\NewTranslation[3]{%
229   \@trnslt@new@translation{#2}{#1}{#3}}
230 \@onlypreamble\NewTranslation
231
232 \newrobustcmd*\RenewTranslation[3]{%
233   \@trnslt@renew@translation{#2}{#1}{#3}}
234 \@onlypreamble\RenewTranslation
235
236 % #1: word
237 % #2: language

```

## 5 Implementation

```

238 % #3: replacement
239 \def\@trnslt@declare@translation#1#2#3{%
240   \@trnslt@if@language{#2}
241   {%
242     \@trnslt@if@dialect{#2}
243     {%
244       \csdef{@trnslt@word@#1@\@trnslt@dialect{#2}}{#3}%
245       \@trnslt@if@word\@trnslt@dialect@of{#1}{#2}
246       {}
247       {\csdef{@trnslt@word@#1@\@trnslt@dialect@of{#2}}{#3}}%
248     }
249     {\csdef{@trnslt@word@#1@\@trnslt@language{#2}}{#3}}%
250     % save the <word> as <word>:
251     \csdef{@trnslt@word@#1@literal}{#1}%
252   }
253   {\@trnslt@err@unknown@lang{#2}}%
254 }
255
256 \def\@trnslt@new@translation#1#2#3{%
257   \@trnslt@if@translation{#1}{#2}
258   {\@trnslt@err@already@defined{#1}{#2}}
259   {\@trnslt@declare@translation{#1}{#2}{#3}}
260 }
261
262 \def\@trnslt@renew@translation#1#2#3{%
263   \@trnslt@if@translation{#1}{#2}
264   {\@trnslt@declare@translation{#1}{#2}{#3}}
265   {\@trnslt@err@not@defined{#1}{#2}}
266 }
267
268 % -----
269 % now let's go through some trouble to check if a translation exists:
270 \def\@trnslt@if@word#1#2#3{%
271   \expandafter\expandafter\expandafter\expandafter
272   \expandafter\expandafter\expandafter
273   \@firstofone
274   {\expandafter\expandafter\expandafter}%
275   \ifcsname @trnslt@word@#2@#1{#3}\endcsname
276   \expandafter\@firstoftwo
277   \else
278   \expandafter\@secondoftwo
279   \fi
280 }%
281
282 \def\@trnslt@if@translation#1#2{%
283   \@trnslt@if@word\@trnslt@language{#1}{#2}
284   {\expandafter\@firstoftwo}
285   {%
286     \@trnslt@if@dialect{#2}
287     {%
288       \@trnslt@if@word\@trnslt@dialect{#1}{#2}
289       {\expandafter\@firstoftwo}
290       {%
291         \@trnslt@if@word\@trnslt@dialect@of{#1}{#2}

```

## 5 Implementation

```

290         {\expandafter\@firstoftwo}
291         {\expandafter\@secondoftwo}%
292     }
293 }
294 {\expandafter\@secondoftwo}%
295 }%
296 }
297
298 % -----
299 % \GetTranslationFor and \GetTranslation
300 % these need to be expandable!
301 % #1: language
302 % #2: word
303 \newcommand*\GetTranslationFor[2]{%
304     \@trnslt@checkandget@translation@for{#2}{#1}}
305
306 \newcommand*\GetTranslation[1]{%
307     \@trnslt@checkandget@translation@for{#1}{\@trnslt@current@language}}
308
309 % unexpandable version of the commands that raise a warning if no translation
310 % is available:
311 \newcommand*\GetTranslationForWarn[2]{%
312     \@trnslt@getandwarn@translation@for{#2}{#1}}
313
314 \newcommand*\GetTranslationWarn[1]{%
315     \@trnslt@getandwarn@translation@for{#1}{\@trnslt@current@language}}
316
317 % #1: word #2: language
318 \def\@trnslt@get@translation@for#1#2{%
319     \@trnslt@if@dialect{#2}
320     {%
321         \ifcsdef{@trnslt@word@#1@\@trnslt@dialect{#2}}
322         {\csuse{@trnslt@word@#1@\@trnslt@dialect{#2}}}
323         {\csuse{@trnslt@word@#1@\@trnslt@dialect@of{#2}}}%
324     }
325     {\csuse{@trnslt@word@#1@\@trnslt@language{#2}}}%
326 }
327
328 \def\@trnslt@checkandget@translation@for#1#2{%
329     \@trnslt@if@translation{#1}{#2}
330     {\@trnslt@get@translation@for{#1}{#2}}
331     {%
332         \@trnslt@if@translation{#1}{fallback}
333         {\csuse{@trnslt@word@#1@fallback}}
334         {\csuse{@trnslt@word@#1@literal}}}%
335     }%
336 }
337
338 % this is not expandable!
339 \protected\def\@trnslt@getandwarn@translation@for#1#2{%
340     \@trnslt@if@translation{#1}{#2}
341     {\@trnslt@get@translation@for{#1}{#2}}

```



## 5 Implementation

```

342 {%
343   \@trnslt@warning{Translation for `#1' in #2 unknown. You may try to use
344   \string\DeclareTranslation{#2}{#1}{ ... } in your preamble.}%
345   \@trnslt@if@translation{#1}{fallback}
346   {%
347     \@trnslt@info{Using fallback translation for `#1'}%
348     \csuse{@trnslt@word@#1@fallback}
349   }
350   {\csuse{@trnslt@word@#1@literal}}}%
351 }%
352 }
353
354 % -----
355 % \SaveTranslationFor and \SaveTranslation
356 \newrobustcmd*\SaveTranslationFor[3]{%
357   \@trnslt@save@translation@for{#1}{#3}{#2}}
358
359 \newrobustcmd*\SaveTranslation[2]{%
360   \@trnslt@save@translation@for{#1}{#2}{\@trnslt@current@language}}
361
362 \def\@trnslt@save@translation@for#1#2#3{%
363   \edef#1{%
364     \@trnslt@if@translation{#2}{#3}
365     {\csuse{@trnslt@word@#2@\@trnslt@language{#3}}}
366     }%
367   }}
368
369 % -----
370 % \LoadDictionary and \LoadDictionaryFor
371 \newrobustcmd*\LoadDictionary[1]{%
372   \@trnslt@load@dictionary@for{#1}{\@trnslt@current@language}}
373 \@onlypreamble\LoadDictionary
374
375 \newrobustcmd*\LoadDictionaryFor[2]{%
376   \@trnslt@load@dictionary@for{#2}{#1}}
377 \@onlypreamble\LoadDictionaryFor
378
379 % #1: name
380 % #2: lang
381 \def\@trnslt@load@dictionary@for#1#2{%
382   \AtBeginDocument{%
383     \InputIfFileExists{#1-\@trnslt@language{#2}.trsl}
384     {\@trnslt@check@dictionary{#1}{#2}}
385     {\@trnslt@warning{dictionary file `#1-\@trnslt@language{#2}.trsl' not
386       found.}}}%
387   }}
388
389 \def\@trnslt@check@dictionary#1#2{%
390   \AfterFile{#1-\@trnslt@language{#2}.trsl}
391   {%
392     \ifcsdef{@trnslt@dictionary@#1@\@trnslt@language{#2}}
393     {\@trnslt@info{loading dictionary `#1' for `#2'.}}

```

## 5 Implementation

```

394     {%
395         \@trnslt@warning{file `#1-\@trnslt@language{#2}.trsl' does not
396             appear to be a dictionary}%
397     }%
398 }%
399 }
400
401 \def\@trnslt@load@dictionary@silent@for#1#2{%
402     \AtBeginDocument{\InputIfFileExists{#1-\@trnslt@language{#2}.trsl}{}}{}%
403 }
404 \newrobustcmd*\ProvideDictionaryFor[2]{%
405     \@trnslt@provide@dictionary@for{#1}{#2}%
406     \@onlypreamble\ProvideDictionaryFor
407 }
408 \def\@trnslt@provide@dictionary@for#1#2{%
409     \def\@trnslt@dictionary@name{#2}%
410     \edef\@trnslt@dictionary@lang{\@trnslt@language{#1}}%
411     % this macro can be used to check if we have a dictionary and will also be
412     % used as a list for the dictionary entries:
413     \csdef{\@trnslt@dictionary@\@trnslt@dictionary@name \@trnslt@dictionary@lang}{%
414         \ifnextchar[
415             {\@trnslt@provide@dictionary@version}
416             {
417                 \ProvidesFile
418                     {#2-\@trnslt@dictionary@lang.trsl}%
419                     [(\@trnslt@dictionary@lang\space translation file `#2')]
420             }%
421         }
422     }
423 \def\@trnslt@provide@dictionary@version[#1]{%
424     \ProvidesFile
425         {\@trnslt@dictionary@name-\@trnslt@dictionary@lang.trsl}%
426         [(\@trnslt@dictionary@lang\space translation file ` \@trnslt@dictionary@name') #1]}
427
428 % \@trnslt@dictionary@language
429 \newrobustcmd*\DeclareDictTranslation[2]{%
430     \listcsadd
431         {\@trnslt@dictionary@\@trnslt@dictionary@name \@trnslt@dictionary@lang}
432         {{#1}{#2}}%
433     \@trnslt@declare@translation{#1}{\@trnslt@dictionary@lang}{#2}%
434 }
435 \@onlypreamble\DeclareDictTranslation
436
437 % \PrintDictionaryFor
438 % #1: lang
439 % #2: name
440 % #3: pre
441 % #4: mid
442 % #5: post
443 \newcommand*\PrintDictionaryFor[5]{%
444     \@trnslt@print@dictionary@for{#1}{#2}{#3}{#4}{#5}%
445 }

```

## 5 Implementation

```
446 % #1: lang
447 % #2: name
448 % #3: pre
449 % #4: mid
450 % #5: post
451 \def\@trnslt@print@dictionary@for#1#2#3#4#5{%
452   \forlistcsloop
453     {\@trnslt@print@dictionary@entry{#3}{#4}{#5}}
454     {\@trnslt@dictionary@#2@\@trnslt@language{#1}}%
455 }
456
457 % #1: pre
458 % #2: mid
459 % #3: post
460 % #4: {key}{translation}
461 \def\@trnslt@print@dictionary@entry#1#2#3#4{%
462   \@trnslt@print@dictionary@entry@aux{#1}{#2}{#3}#4}
463
464 % #1: pre
465 % #2: mid
466 % #3: post
467 % #4: key
468 % #5: translation
469 \def\@trnslt@print@dictionary@entry@aux#1#2#3#4#5{#1#4#2#5#3}
470
471 % -----
472 % predefined languages
473 \DeclareLanguage{albanian}
474 \DeclareLanguage{bulgarian}
475 \DeclareLanguage{catalan}
476 \DeclareLanguage{croatian}
477 \DeclareLanguage{czech}
478 \DeclareLanguage{danish}
479 \DeclareLanguage{dutch}
480 \DeclareLanguage{english}
481 \DeclareLanguage{finnish}
482 \DeclareLanguage{french}
483 \DeclareLanguage{german}
484 \DeclareLanguage{greek}
485 \DeclareLanguage{hebrew}
486 \DeclareLanguage{hungarian}
487 \DeclareLanguage{icelandic}
488 \DeclareLanguage{italian}
489 \DeclareLanguage{norwegian}
490 \DeclareLanguage{polish}
491 \DeclareLanguage{portuges}
492 \DeclareLanguage{romanian}
493 \DeclareLanguage{russian}
494 \DeclareLanguage{serbocroatian}
495 \DeclareLanguage{slovak}
496 \DeclareLanguage{slovenian}
497 \DeclareLanguage{spanish}
```

## 5 Implementation

```
498 \DeclareLanguage{swedish}
499 \DeclareLanguage{turkish}
500 \DeclareLanguage{ukrainian}
501
502 % -----
503 % aliases and dialects:
504 \DeclareLanguageAlias {Albanian}{albanian}
505 \DeclareLanguageAlias {Bulgarian}{bulgarian}
506 \DeclareLanguageAlias {Catalan}{catalan}
507 \DeclareLanguageAlias {Croatian}{croatian}
508 \DeclareLanguageAlias {Czech}{czech}
509 \DeclareLanguageAlias {Danish}{danish}
510 \DeclareLanguageAlias {Dutch}{dutch}
511 \DeclareLanguageAlias {Finnish}{finnish}
512 \DeclareLanguageAlias {français}{french}
513 \DeclareLanguageAlias {Français}{français}
514 \DeclareLanguageDialect{canadien}{french}
515 \DeclareLanguageAlias {Canadien}{canadien}
516 \DeclareLanguageAlias {French}{french}
517 \DeclareLanguageDialect{american}{english}
518 \DeclareLanguageAlias {American}{american}
519 \DeclareLanguageDialect{australian}{english}
520 \DeclareLanguageAlias {Australian}{australian}
521 \DeclareLanguageDialect{british}{english}
522 \DeclareLanguageAlias {British}{british}
523 \DeclareLanguageDialect{canadian}{english}
524 \DeclareLanguageAlias {Canadian}{canadian}
525 \DeclareLanguageAlias {English}{english}
526 \DeclareLanguageAlias {UKenglish}{british}
527 \DeclareLanguageAlias {USenglish}{american}
528 \DeclareLanguageDialect{austrian}{german}
529 \DeclareLanguageAlias {Austrian}{austrian}
530 \DeclareLanguageAlias {German}{german}
531 \DeclareLanguageAlias {germanb}{german}
532 \DeclareLanguageDialect{naustrian}{austrian}
533 \DeclareLanguageAlias {ngerman}{german}
534 \DeclareLanguageAlias {Greek}{greek}
535 \DeclareLanguageAlias {polutonikogreek}{greek}
536 \DeclareLanguageAlias {Hebrew}{hebrew}
537 \DeclareLanguageAlias {Hungarian}{hungarian}
538 \DeclareLanguageDialect{magyar}{hungarian}
539 \DeclareLanguageAlias {Magyar}{magyar}
540 \DeclareLanguageAlias {Icelandic}{icelandic}
541 \DeclareLanguageAlias {Italian}{italian}
542 \DeclareLanguageAlias {norsk}{norwegian}
543 \DeclareLanguageAlias {Norsk}{norsk}
544 \DeclareLanguageAlias {Norwegian}{norwegian}
545 \DeclareLanguageAlias {nynorsk}{norwegian}
546 \DeclareLanguageAlias {Nynorsk}{nynorsk}
547 \DeclareLanguageAlias {Polish}{polish}
548 \DeclareLanguageDialect{brazil}{portuges}
549 \DeclareLanguageAlias {Brazil}{brazil}
```

## 5 Implementation

```
550 \DeclareLanguageAlias {brazilian}{brazil}
551 \DeclareLanguageAlias {Brazilian}{brazilian}
552 \DeclareLanguageAlias {Portuges}{portuges}
553 \DeclareLanguageAlias {portuguese}{portuges}
554 \DeclareLanguageAlias {Portuguese}{portuguese}
555 \DeclareLanguageAlias {Romanian}{romanian}
556 \DeclareLanguageAlias {Russian}{russian}
557 \DeclareLanguageAlias {Serbocroatian}{serbocroatian}
558 \DeclareLanguageAlias {Slovak}{slovak}
559 \DeclareLanguageAlias {Slovenian}{slovenian}
560 \DeclareLanguageAlias {Spanish}{spanish}
561 \DeclareLanguageAlias {Swedish}{swedish}
562 \DeclareLanguageDialect{swissgerman}{german}
563 % this maybe should be a language of it's own:
564 \DeclareLanguageAlias {Swiss}{swissgerman}
565 \DeclareLanguageAlias {Swissgerman}{swissgerman}
566 \DeclareLanguageAlias {Turkish}{turkish}
567 \DeclareLanguageAlias {Ukrainian}{ukrainian}
568
569 % -----
570 % load basic dictionary if available
571 \AtBeginDocument{%
572   \@trnslt@load@dictionary@silent@for
573     {translations-basic-dictionary}
574     {\@trnslt@current@language}%
575 }
576
577 \endinput
578
579 % -----
580 % HISTORY:
581 2012/09/30 v0.2beta - first version (as part of the `exsheets' bundle)
582 2012/10/05 v0.2    - \LoadDictionary and \LoadDictionaryFor added and loads of
583                    languages defined.
584 2013/03/10 v0.8    - basic dictionaries for English, German, French and Spanish
585                    - new command \DeclareDictTranslation
586 2013/04/04 v0.8a   - bug fix in \DeclareDictTranslation
587 2013/04/07 v0.9    - slightly improved messages
588 2013/04/08 v0.9a   - changed fallback warning into info
589                    - synchronized version number with `exsheets' until now but
590                      won't any more
591 2013/06/22 v0.9b   - added Swiss
592 2013/06/28 v0.10   - declaring aliases of dialects now works as expected
593                    - declarings dialects of an alias now correctly declares
594                      the dialect to the correct base language
595                    - corrected a few erroneous language declarations
596 2013/07/12 v0.10a  - \GetTranlation gets two-folded fallback: use
597                    fallback-translation if no translation for the current
598                    language has been defined; use literal string if /no/
599                    language is used - this should never happen but /will/
600                    happen if neither `babel' nor `polyglossia' have been
601                    loaded, i.e., no language has been chosen /and/ the
```

## 5 *Implementation*

```
602 package writer did not provide an English translation
603 2013/07/16 v1.0 - removed from `exsheets' bundle - `translations' should
604 be a package of it's own
605 - load basic dictionary automatically if available
606 - rudimentary check in \LoadDictionary if loaded file is a
607 dictionary
608 - new command \PrintDictionaryFor
609 - redefined conditionals; they still seemed to make
610 trouble in some cases
```

# Index

Section titles are indicated **bold**, packages sans serif and commands \brown.

<b>B</b>			
babel	1-4, 6		
beamer	1		
<b>C</b>			
\captions<language>	1		
<b>D</b>			
\DeclareDictTranslation	4		
\DeclareLanguage	3 f.		
\DeclareLanguageAlias	3		
\DeclareLanguageDialect	3		
\DeclareTranslation	3, 5		
\DeclareTranslationFallback	3, 6		
<b>Dictionaries</b>	6-9		
basic	7		
own	6		
<b>E</b>			
etoolbox	2		
<b>G</b>			
\GetTranslation	2 f.		
		\GetTranslationFor	3
		\GetTranslationForWarn	4
		\GetTranslationWarn	4
<b>I</b>			
<b>Implementation</b>	10		
<b>L</b>			
<b>Languages</b>	9 f.		
\LoadDictionary	4		
\LoadDictionaryFor	4		
<b>M</b>			
<b>Motivation</b>	1		
<b>N</b>			
\NewTranslation	3		
<b>P</b>			
polyglossia	1 f., 6		
\PrintDictionaryFor	4		
\providecaptionname	1		
\ProvideDictionaryFor	4		
<b>R</b>			
\RenewTranslation	3		
<b>Requirements</b>	2		
<b>S</b>			
\SaveTranslation	4		
\SaveTranslationFor	4		
scrfile	2		
<b>T</b>			
\translate	1		
translator	1		
<b>U</b>			
<b>Usage</b>	2-9		
Background	2		
Commands	3		
Dictionaries	6		
Example	4		
Packages	5		
fallback	6		
structure	5		