

Babel, a multilingual package for use with L^AT_EX's standard document classes*

Johannes Braams
Kooiënswater 62
2715 AJ Zoetermeer
The Netherlands
JLBraams@cistron.nl

Printed May 21, 1998

Abstract

The standard distribution of L^AT_EX contains a number of document classes that are meant to be used, but also serve as examples for other users to create their own document classes. These document classes have become very popular among L^AT_EX users. But it should be kept in mind that they were designed for American tastes and typography. At one time they contained a number of hard-wired texts. This report describes **babel**, a package that makes use of the new capabilities of T_EX version 3 to provide an environment in which documents can be typeset in a non-american language or in more than one language.

Contents

	5.2	Support for saving macro definitions	9
1 The user interface	3	5.3	Support for extending macros 9
1.1 Languages supported by Babel	4	5.4	Macros common to a number of languages . . . 9
1.2 Workarounds	4		
2 Changes for L^AT_EX 2_ε	5	6 Compatibility with german.sty	10
3 Changes in Babel version 3.6	5	7 Compatibility with the french package	10
4 Changes in Babel version 3.5	6	8 Identification	10
5 The interface between the core of babel and the language definition files	7	9 The Package File	11
5.1 Support for active characters	8	10 The Kernel of Babel	13
		10.1 Multiple languages	14

*During the development ideas from Nico Poppelier, Piet van Oostrum and many others have been used. Bernd Raichle has provided many helpful suggestions.

10.2 Support for active characters	25	24 The French Language	85
10.3 Shorthands	26	24.1 About French typography	85
10.4 Support for saving macro definitions	33	24.2 T _E Xnical details	87
10.5 Support for extending macros	34	24.3 Captionnames and date .	89
10.6 Macros common to a number of languages . . .	34	24.4 Punctuation	90
10.7 Making glyphs available .	35	24.5 French quotation marks .	92
10.8 Quotation marks	35	24.6 French lists	93
10.9 Letters	36	24.7 French indentation of sections	94
10.10 Shorthands for quotation marks	37	24.8 Formatting numbers . . .	95
10.11 Umlauts and trema's . . .	38	24.9 Extra utilities	97
10.12 The redefinition of the style commands	39	25 The Italian language	100
10.12.1 Redefinition of macros	41	26 The Portuguese language	102
10.13 Cross referencing macros .	44	27 The Spanish language	107
11 Local Language Configuration	48	28 The Catalan language	112
12 Driver files for the documented source code	50	29 The Galician language	119
13 Conclusion	54	30 The Romanian language	123
14 Acknowledgements	54	31 The Danish language	125
15 The Esperanto language	55	32 The Norwegian language	128
16 The Dutch language	58	33 The Swedish language	131
17 The English language	63	34 The Finnish language	135
18 The German language	66	35 The Hungarian language	138
19 The Breton language	71	36 The Estonian language	141
20 The Welsh language	75	36.1 Implementation	141
21 The Irish language	77	37 The Croatian language	146
22 The Scottish language	79	38 The Czech language	148
23 The Greek language	81	39 The Polish language	150
		40 The Slovak language	156
		41 The Slovenian language	158
		42 The Russian language	161
		43 The Lower Sorbian language	168

44 The Upper Sorbian language	170	46 The Bahasa language	177
45 The Turkish language	174	47 Support for formats based on plain \TeX	179

1 The user interface

The user interface of this package is quite simple. It consists of a set of commands that switch from one language to another and a set of commands that deal with shorthands. It is also possible to find out what the current language is.

<code>\selectlanguage</code>	When a user wants to switch from one language to another he can do so using the macro <code>\selectlanguage</code> . This macro takes the language, defined previously by a language definition file, as its argument. It calls several macros that should be defined in the language definition files to activate the special definitions for the language chosen.
<code>otherlanguage</code>	The environment <code>otherlanguage</code> does basically the same as <code>\selectlanguage</code> , except the language change is local to the environment. For mixing left-to-right typesetting with right-to-left typesetting the use of this environment is a prerequisite. The language to switch to is specified as an argument to <code>\begin{otherlanguage}</code> .
<code>\foreignlanguage</code>	The command <code>\foreignlanguage</code> takes two arguments, the second argument is a phrase to be typeset according to the rules of the language named in its first argument. This command only switches the extra definitions and the hyphenation rules for the language, <i>not</i> the names and dates.
<code>otherlanguage*</code>	In the environment <code>otherlanguage*</code> only the typesetting is done according to the rules of the other language, but the text-strings such as ‘figure’, ‘table’, etc. are left as they were set outside this environment.
<code>\language</code>	The control sequence <code>\language</code> contains the name of the current language.
<code>\iflanguage</code>	If more than one language is used it might be necessary to know which language is active at a specific time. This can be checked by a call to <code>\iflanguage</code> . This macro takes three arguments. The first argument is the name of a language, the second and third arguments are the actions to take if the result of the test is <code>true</code> or <code>false</code> respectively.
<code>\useshorthands</code>	The command <code>\useshorthands</code> initiates the definition of user-defined shorthand sequences. It has one argument, the character which starts these personal shorthands.
<code>\defineshorthand</code>	The command <code>\defineshorthand</code> takes two arguments, the first of which is a one or two character sequence, the second argument is the code the shorthand should expand to.
<code>\aliasshorthand</code>	The command <code>\aliasshorthand</code> can be used to let another character perform the same functions as the default shorthand character. If one prefers for example to use the character / over " in typing polish texts this can be achieved by entering <code>\aliasshorthand{"}{/}</code> .
<code>\languageshorthands</code>	The command <code>\languageshorthands</code> can be used to switch the shorthands on the language level. It takes one argument, the name of a language. Note that for this to work the language should have been specified as an option when loading the <code>babel</code> package.

1.1 Languages supported by Babel

In the following table all the languages supported by **Babel** are listed, together with the names of the options with which you can load **babel** for each language.

Afrikaans	afrikaans
Bahasa	bahasa
Breton	breton
Catalan	catalan
Croatian	croatian
Czech	czech
Danish	danish
Dutch	dutch
English	english, USenglish, american, UKenglish, british
Esperanto	esperanto
Estonian	estonian
Finnish	finnish
French	french, francais
Galician	galician
German	austrian, german, germanb
Greek	greek
Hebrew	hebrew
Hungarian	magyar, hungarian
Irish Gaelic	irish
Italian	italian
Lower Sorbian	lowersorbian
Norwegian	norsk, nynorsk
Polish	polish
Portuguese	portuges, portuguese, brazilian, brazil
Romanian	romanian
Russian	russian
Scottish Gaelic	scottish
Spanish	spanish
Slovakian	slovak
Slovenian	slovene
Swedish	swedish
Turkish	turkish
Upper Sorbian	uppersorbian
Welsh	welsh

For some languages **babel** supports the options **activeacute** and **activegrave**; for typesetting russian texts **babel** knows about the options **LWN** and **LCY** to specify the fontencoding of the cyrillic font used. Currently only **LWN** is supported.

1.2 Workarounds

When you use the document class **book** *and* you use **\ref** inside the argument of **\chapter** you will experience the problem that **L^AT_EX** will keep complaining about an undefined label. The reason is that the argument of **\ref** is passed through **\uppercase** at some time during processing. To prevent such problems you could

revert to using uppercase labels, or you can use `\lowercase{\ref{foo}}` inside the argument of `\chapter`.

2 Changes for L^AT_EX 2_ε

With the advent of L^AT_EX 2_ε the interface to `babel` in the preamble of the document has changed. With L^AT_EX 2.09 one used to call up the `babel` system with a line such as:

```
\documentstyle[dutch,english]{article}
```

which would tell L^AT_EX that the document would be written in two languages, dutch and english and that english would be the first language in use.

The L^AT_EX 2_ε way of providing the same information is:

```
\documentclass{article}
\usepackage[dutch,english]{babel}
```

or, making `dutch` and `english` global options in order to let other packages detect and use them:

```
\documentclass[dutch,english]{article}
\usepackage{babel}
\usepackage{varioref}
```

In this last example the package `varioref` will also see the options and will be able to use them.

3 Changes in Babel version 3.6

In Babel version 3.6 a number of bugs that were found in version 3.5 are fixed. Also a number of changes and additions have occurred:

- A new environment `otherlanguage*` is introduced. it only switches the ‘specials’, but leaves the ‘captions’ untouched.
- The shorthands are no longer fully expandable. Some problems could only be solved by peeking at the token following an active character. The advantage is that `'{\a}` works as expected for languages that have the `'` active.
- Support for typesetting french texts is much enhanced; the file `francais.ldf` is now replaced by `frenchb.ldf` which is maintained by Daniel Flipo.
- Support for typesetting the russian language is again available. The language definition file was originally developed by Olga Lapko from cyrtug. The fonts needed to typeset the russian language are now part of the `babel` distribution. The support is not yet up to the level which is needed according to Olga, but this is a start.
- Support for typesetting greek texts is now also available. What is offered in this release is a first attempt; it will be enhanced later on by Yannis Haralambous.

- in babel 3.6j some hooks have been added for the development of support for Hebrew typesetting.
- Support for typesetting texts in Afrikaans (a variant of Dutch, spoken in South Africa) has been added to `dutch.ldf`.
- Support for typesetting welsh texts is now available.
- A new command `\aliasshorthand` is introduced. It seems that in Poland various conventions are used to type the necessary polish letters. It is now possible to use the character / as a shorthand character instead of the character " by issuing the command `\aliasshorthand{"}{/}`.
- The shorthand mechanism now deals correctly with characters that are already active.
- Shorthand characters are made active at `\begin{document}`, not earlier. This is to prevent problems with other packages.
- A *preambleonly* command `\substitutefontfamily` has been added to create .fd files on the fly when the font families of the latin text differ from the families used for the cyrillic or greek parts of the text.
- Three new commands `\LdfInit`, `\ldf@quit` and `\ldf@finish` are introduced that perform a number of standard tasks.

4 Changes in Babel version 3.5

In Babel version 3.5 a lot of changes have been made when compared with the previous release. Here is a list of the most important ones:

- the selection of the language is delayed until `\begin{document}`, this has the consequence that you need to add appropriate `\selectlanguage` commands if you include `\hyphenation` lists in the preamble of your document.
- babel now has a `language` environment and a new command `\foreignlanguage`;
- the way active characters are dealt with is completely changed. They are called ‘shorthands’; one can have three levels of shorthands: on the user level, the language level and on ‘system level’. A consequence of the new way of handling active characters is that they are now written to auxiliary files ‘verbatim’;
- A language change now also writes information in the .aux file as the change might also affect typesetting the table of contents. The consequence is that an .aux file generated by a LaTeX format with babel preloaded gives errors when read with a LaTeX format without babel, but I think this probably doesn’t occur;
- babel is now compatible with the `inputenc` and `fontenc` packages;
- the language definition files now have a new extension, `ldf`;
- the syntax of the file `language.dat` is extended to be compatible with the `french` package by Bernard Gaulle;

- each language definition file looks for a configuration file which has the same name, but the extension `.cfg`. It can contain any valid \LaTeX code.

5 The interface between the core of **babel** and the language definition files

In the core of the **babel** system two macros are defined that are to be used in language definition files. Their purpose is to make a new language known.

`\addlanguage` The macro `\addlanguage` is a non-outer version of the macro `\newlanguage`, defined in `plain.tex` version 3.x. For older versions of `plain.tex` and `lplain.tex` a substitute definition is used.

`\adddialect` The macro `\adddialect` can be used in the case where two languages can (or have to) use the same hyphenation patterns. This can also be useful when a user wants to use a language for which no patterns are preloaded in the format. In such a case the default behaviour of the **babel** system is to define this language as a ‘dialect’ of the language for which the patterns were loaded as `\language0`.

The language definition files have to conform to a number of conventions. The reason for this is that these files have to fill in the gaps left by the common code in `babel.def`, i.e., the definitions of the macros that produce texts. Also the language-switching possibility which has been built into the **babel** system has its implications.

The following assumptions are made:

- Some of the language-specific definitions might be used by plain \TeX users, so the files have to be coded such that they can be read by \LaTeX as well as by plain \TeX . The current format can be checked by looking at the value of the macro `\fmtname`.
- The common part of the **babel** system redefines a number of macros and environments (defined previously in the document style) to put in the names of macros that replace the previously hard-wired texts. These macros have to be defined in the language definition files.
- The language definition files define five macros, used to activate and deactivate the language-specific definitions. These macros are `\langle lang \rangle hyphenmins`, `\captions\langle lang \rangle`, `\date\langle lang \rangle`, `\extras\langle lang \rangle` and `\noextras\langle lang \rangle`; where `\langle lang \rangle` is either the name of the language definition file or the name of the \LaTeX option that is to be used. These macros and their functions are discussed below.
- When a language definition file is loaded, it can define `\l@\langle lang \rangle` to be a dialect of `\language0` when `\l@\langle lang \rangle` is undefined.
- The language definition files can be read in the preamble of the document, but also in the middle of document processing. This means that they have to function independently of the current `\catcode` of the `@` sign.

`\langhyphenmins` The macro `\langle lang \rangle hyphenmins` is used to store the values of the `\lefthyphenmin` and `\righthyphenmin`.

`\captionslang` The macro `\captions\langle lang \rangle` defines the macros that hold the texts to replace the original hard-wired texts.

<code>\datelang</code>	The macro <code>\date<lang></code> defines <code>\today</code> and
<code>\extraslang</code>	The macro <code>\extras<lang></code> contains all the extra definitions needed for a specific language.
<code>\noextraslang</code>	Because we want to offer the user the possibility to switch between languages and we do not know in what state \TeX might be after the execution of <code>\extras<lang></code> , a macro that brings \TeX into a predefined state is needed. It will be no surprise that the name of this macro is <code>\noextras<lang></code> .
<code>\main@language</code>	To postpone the activation of the definitions needed for a language until the beginning of a document, all language definition files should use <code>\main@language</code> instead of <code>\selectlanguage</code> . This will just store the name of the language and the proper language will be activated at the start of the document.
<code>\LdfInit</code>	The macro <code>\LdfInit</code> performs a couple of standard checks that have to be made at the beginning of a language definition file, such as checking the category code of the <code>@</code> -sign, preventing that the <code>.ldf</code> file is processed twice, etc.
<code>\ldf@quit</code>	The macro <code>\ldf@quit</code> performs a couple of tasks that need to be taken care of when a <code>.ldf</code> file was processed earlier. These tasks include the resetting of the category code of the <code>@</code> -sign, preparing the language to be activated at <code>\begin{document}</code> time and ending the input stream.
<code>\ldf@finish</code>	The macro <code>\ldf@finish</code> performs a couple of tasks that need to be taken care of at the end of each <code>.ldf</code> file. These tasks include the resetting of the category code of the <code>@</code> -sign, the loading of a local configuration file and preparing the language to be activated at <code>\begin{document}</code> time.
<code>\loadlocalcfg</code>	At the end of the processing of a language definition file \LaTeX can be instructed to load a local configuration file. This file can for instance be used to add strings to <code>\captions<lang></code> in order to support local document classes. The user will be informed of the fact that this configuration file is loaded. This macro is called by <code>\finish@ldf</code> .
<code>\substitute fontfamily</code>	This command takes three arguments, a font encoding and two font family names. It creates a font description file for the first font in the given encoding. This <code>.fd</code> file will instruct \LaTeX to use a font from the second family when a font from the first family in the given encoding seems to be needed.

5.1 Support for active characters

In quite a number of language definition files, active characters are introduced. To facilitate this, some support macros are provided.

<code>\initiate@active@char</code>	The internal macro <code>\initiate@active@char</code> is used in language definition files to instruct \LaTeX to give a character the category code ‘active’. When a character has been made active it will remain that way until the end of the document. Its definition may vary.
<code>\bbl@activate</code> <code>\bbl@deactivate</code>	The command <code>\bbl@activate</code> is used to change the way an active character expands. <code>\bbl@activate</code> ‘switches on’ the active behaviour of the character. <code>\bbl@deactivate</code> lets the active character expand to its former (mostly) non-active self.
<code>\declare@shorthand</code>	The macro <code>\declare@shorthand</code> is used to define the various shorthands. It takes three arguments, the name for the collection of shorthands this definition belongs to; the character (sequence) that makes up the shorthand i.i. <code>~</code> or <code>"a</code> and the code to be executed when the shorthand is encountered.
<code>\bbl@add@special</code> <code>\bbl@remove@special</code>	The \TeX book states: “Plain \TeX includes a macro called <code>\dospecials</code> that

is essentially a set macro, representing the set of all characters that have a special category code.” [1, p. 380] It is used to set text ‘verbatim’. To make this work if more characters get a special category code, you have to add this character to the macro `\dospecial`. L^AT_EX adds another macro called `\@sanitize` representing the same character set, but without the curly braces. The macros `\bbl@add@special⟨char⟩` and `\bbl@remove@special⟨char⟩` add and remove the character `⟨char⟩` to these two sets.

5.2 Support for saving macro definitions

Language definition files may want to *redefine* macros that already exist. Therefore a mechanism for saving (and restoring) the original definition of those macros is provided. We provide two macros for this¹.

`\babel@save` To save the current meaning of any control sequence the macro `\babel@save` is provided. It takes one argument, `⟨csequence⟩`, the control sequence for which the meaning has to be saved.

`\babel@savevariable` A second macro is provided to save the current value of a variable. In this context anything that is allowed after the `\the` primitive is considered to be a variable. The macro takes one argument, the `⟨variable⟩`.

The effect of the aforementioned macros is that a piece of code is appended to the current definition of `\originalTeX`. When `\originalTeX` is expanded this code restores the previous definition of the control sequence or the previous value of the variable.

5.3 Support for extending macros

`\addto` The macro `\addto{⟨control sequence⟩}{⟨TEX code⟩}` can be used to extend the definition of a macro. The macro need not be defined. This macro can, for instance, be used in adding instructions to a macro like `\extrasenglish`.

5.4 Macros common to a number of languages

`\allowhyphens` In a couple of european languages compound words are used. This means that when T_EX has to hyphenate such a compound word it only does that at the ‘-’ that is used in such words. To allow hyphenation in the rest of such a compound word the macro `\allowhyphens` can be used.

`\set@low@box` For some languages quotes need to be lowered to the baseline. For this purpose the macro `\set@low@box` is available. It takes one argument and puts that argument in an `\hbox`, at the baseline. The result is available in `\box0` for further processing.

`\save@sf@q` Sometimes it is necessary to preserve the `\spacefactor`. For this purpose the macro `\save@sf@q` is available. It takes one argument, saves the current spacefactor, executes the argument and restores the spacefactor.

`\bbl@frenchspacing`
`\bbl@nonfrenchspacing` The commands `\bbl@frenchspacing` and `\bbl@nonfrenchspacing` can be used to properly switch french spacing on and off.

¹This mechanism was introduced by Bernd Raichle.

6 Compatibility with `german.sty`

The file `german.sty` has been one of the sources of inspiration for the `babel` system. Because of this I wanted to include `german.sty` in the `babel` system. To be able to do that I had to allow for one incompatibility: in the definition of the macro `\selectlanguage` in `german.sty` the argument is used as the *number* for an `\ifcase`. So in this case a call to `\selectlanguage` might look like `\selectlanguage{\german}`.

In the definition of the macro `\selectlanguage` in `babel.def` the argument is used as a part of other macronames, so a call to `\selectlanguage` now looks like `\selectlanguage{german}`. Notice the absence of the escape character. As of version 3.1a of `babel` both syntaxes are allowed.

All other features of the original `german.sty` have been copied into a new file, called `germanb.sty`².

Although the `babel` system was developed to be used with \LaTeX , some of the features implemented in the language definition files might be needed by plain \TeX users. Care has been taken that all files in the system can be processed by plain \TeX .

7 Compatibility with the french package

It has been reported to me that the package `french` by Bernard Gaulle (gaulle@idris.fr) works together with `babel`.

Therefore, `babel` will first search for the file `french.ldf` when you give it the option `french`; then it will try to load `frenchb.ldf`. When you give `babel` the option `francais` it will only look for `frenchb.ldf`.

8 Identification

The file `babel.sty`³ is meant for \LaTeX 2_ϵ , therefore we make sure that the format file used is the right one.

The identification code for each file is something that was introduced in \LaTeX 2_ϵ . When the command `\ProvidesFile` does not exist, v3.6 dummy definition is provided.

```
8.1 \<!--package>
8.2 \ifx\ProvidesFile\@undefined
8.3   \def\ProvidesFile#1[#2 #3 #4]{%
8.4     \wlog{#4 #3 <#2>}}%
8.5 \<kernel & patterns>    \toks8{Babel <#3> and hyphenation patterns for }%
8.6   }
8.7 \<kernel & patterns>
8.8 \else
```

In this case we save the original definition of `\ProvidesFile` in `\bbl@tempa` and restore it after we have stored the version of the file in `\toks8`.

```
8.9   \let\bbl@tempa\ProvidesFile
8.10  \def\ProvidesFile#1[#2 #3 #4]{%
```

²The ‘b’ is added to the name to distinguish the file from Partls’ file.

³The file described in this section is called `babel.dtx`, has version number v3.6j and was last revised on 1998/03/24.

```

8.11 \toks8{Babel <#3> and hyphenation patterns for }%
8.12 \bbl@tempa{#1}[#2 #3 #4]%
8.13 \let\ProvidesFile\bbl@tempa}
8.14 </kernel & patterns>
8.15 \fi
8.16 </!package>

```

Identify each file that is produced from this source file.

```

8.17 <+package>\ProvidesPackage{babel}
8.18 <+core>\ProvidesFile{babel.def}
8.19 <+kernel & patterns>\ProvidesFile{hyphen.cfg}
8.20 <+kernel&!patterns>\ProvidesFile{switch.def}
8.21 <+driver&!user>\ProvidesFile{babel.drv}
8.22 <+driver & user>\ProvidesFile{user.drv}
8.23 [1998/03/24 v3.6j
8.24 <+package> The Babel package]
8.25 <+core> Babel common definitions]
8.26 <+kernel> Babel language switching mechanism]
8.27 <+driver>]

```

9 The Package File

In order to make use of the new features of $\text{\LaTeX} 2_{\epsilon}$, a new file is introduced to the **babel** system, **babel.sty**. This file is loaded by the `\usepackage` command and defines all the language options known in the **babel** system.

For all the languages supported we need to declare an option.

```

9.1 <*package>
9.2 \ifx\LdfInit\undefined\input{babel.def}\fi
9.3 \DeclareOption{afrikaans}{\input{dutch.ldf}}
9.4 \DeclareOption{american}{\input{english.ldf}}

Austrian is really a dialect of German.

9.5 \DeclareOption{austrian}{\input{germanb.ldf}}
9.6 \DeclareOption{bahasa}{\input{bahasa.ldf}}
9.7 \DeclareOption{brazil}{\input{portuges.ldf}}

9.8 \DeclareOption{brazilian}{\input{portuges.ldf}}
9.9 \DeclareOption{breton}{\input{breton.ldf}}

9.10 \DeclareOption{british}{\input{english.ldf}}
9.11 \DeclareOption{catalan}{\input{catalan.ldf}}
9.12 \DeclareOption{croatian}{\input{croatian.ldf}}
9.13 \DeclareOption{czech}{\input{czech.ldf}}
9.14 \DeclareOption{danish}{\input{danish.ldf}}
9.15 \DeclareOption{dutch}{\input{dutch.ldf}}

9.16 \DeclareOption{english}{\input{english.ldf}}
9.17 \DeclareOption{esperanto}{\input{esperant.ldf}}
9.18 \DeclareOption{estonian}{\input{estonian.ldf}}
9.19 \DeclareOption{finnish}{\input{finnish.ldf}}

```

The **babel** support for French used to be stored in **francais.ldf**; therefore the $\text{\LaTeX} 2.09$ option used to be **francais**. The hyphenation patterns may be loaded as either ‘french’ or as ‘francais’.

```
9.20 \DeclareOption{francais}{\input{frenchb.ldf}}
9.21 \DeclareOption{frenchb}{\input{frenchb.ldf}}
```

With L^AT_EX 2_ε we can now also use the option french and still call the file francais.ldf.

```
9.22 \IfFileExists{french.ldf}{%
9.23   \DeclareOption{french}{\input{french.ldf}}%
9.24 }{%
9.25   \DeclareOption{french}{\input{frenchb.ldf}}%
9.26 }
9.27 \DeclareOption{galician}{\input{galician.ldf}}
9.28 \DeclareOption{german}{\input{germanb.ldf}}
9.29 \DeclareOption{germanb}{\input{germanb.ldf}}

9.30 \DeclareOption{greek}{\input{greek.ldf}}
9.31 \DeclareOption{hebrew}{\input{rlbabel.def}\input{hebrew.ldf}}
```

hungarian is just a synonym for magyar

```
9.32 \DeclareOption{hungarian}{\input{magyar.ldf}}
9.33 \DeclareOption{irish}{\input{irish.ldf}}
9.34 \DeclareOption{italian}{\input{italian.ldf}}
9.35 \DeclareOption{lowersorbian}{\input{lsorbian.ldf}}
9.36 \DeclareOption{magyar}{\input{magyar.ldf}}
9.37 \DeclareOption{norsk}{\input{norsk.ldf}}
```

For Norwegian two spelling variants are provided.

```
9.38 \DeclareOption{nynorsk}{\input{norsk.ldf}}
9.39 \DeclareOption{polish}{\input{polish.ldf}}
9.40 \DeclareOption{portuges}{\input{portuges.ldf}}
9.41 \DeclareOption{portuguese}{\input{portuges.ldf}}
9.42 \DeclareOption{romanian}{\input{romanian.ldf}}
9.43 \DeclareOption{russian}{\input{russianb.ldf}}
9.44 \DeclareOption{scottish}{\input{scottish.ldf}}
9.45 \DeclareOption{slovak}{\input{slovak.ldf}}
9.46 \DeclareOption{slovene}{\input{slovene.ldf}}
9.47 \DeclareOption{spanish}{\input{spanish.ldf}}
9.48 \DeclareOption{swedish}{\input{swedish.ldf}}
9.49 \DeclareOption{turkish}{\input{turkish.ldf}}
9.50 \DeclareOption{uppersorbian}{\input{usorbian.ldf}}
9.51 \DeclareOption{welsh}{\input{welsh.ldf}}

9.52 \DeclareOption{UKenglish}{\input{english.ldf}}
9.53 \DeclareOption{USenglish}{\input{english.ldf}}
```

Apart from all the language options we also have a few options that influence the behaviour of language definition files.

The following options don't do anything themselves, they are just defined in order to make it possible for language definition files to check if one of them was specified by the user.

```
9.54 \DeclareOption{activeacute}{}
9.55 \DeclareOption{activegrave}{}

```

The next option tells babel to leave shorthand characters active at the end of processing the package. This is *not* the default as it can cause problems with other packages, but for those who want to use the shorthand characters in the preamble of their documents this can help.

```

9.56 \DeclareOption{KeepShorthandsActive}{%
9.57   \def\KeepShorthandsActive{}}

```

The options have to be processed in the order in which the user specified them:

```

9.58 \ProcessOptions*

```

`\substitutefontfamily` The command `\substitutefontfamily` creates an `.fd` file on the fly. The first argument is an encoding mnemonic, the second and third arguments are font family names.

```

9.59 \def\substitutefontfamily#1#2#3{%
9.60   \immediate\openout15=#1#2.fd\relax
9.61   \immediate\write15{%
9.62     \string\ProvidesFile{#1#2.fd}%
9.63     [\the\year/\two@digits{\the\month}/\two@digits{\the\day}
9.64     \space generated font description file]^^J
9.65     \string\DeclareFontFamily{#1}{#2}{^^J
9.66     \string\DeclareFontShape{#1}{#2}{m}{n}{<->ssub * #3/m/n}{^^J
9.67     \string\DeclareFontShape{#1}{#2}{m}{it}{<->ssub * #3/m/it}{^^J
9.68     \string\DeclareFontShape{#1}{#2}{m}{sl}{<->ssub * #3/m/sl}{^^J
9.69     \string\DeclareFontShape{#1}{#2}{m}{sc}{<->ssub * #3/m/sc}{^^J
9.70     \string\DeclareFontShape{#1}{#2}{b}{n}{<->ssub * #3/bx/n}{^^J
9.71     \string\DeclareFontShape{#1}{#2}{b}{it}{<->ssub * #3/bx/it}{^^J
9.72     \string\DeclareFontShape{#1}{#2}{b}{sl}{<->ssub * #3/bx/sl}{^^J
9.73     \string\DeclareFontShape{#1}{#2}{b}{sc}{<->ssub * #3/bx/sc}{^^J
9.74     }%
9.75   \closeout15
9.76 }

```

This command should only be used in the preamble of a document.

```

9.77 \@onlypreamble\substitutefontfamily
9.78 </package>

```

10 The Kernel of Babel

The kernel of the `babel` system is stored in either `hyphen.cfg` or `switch.def` and `babel.def`. The file `hyphen.cfg` is a file that can be loaded into the format, which is necessary when you want to be able to switch hyphenation patterns. The file `babel.def` contains some `TeX` code that can be read in at run time. When `babel.def` is loaded it checks if `hyphen.cfg` is in the format; if not the file `switch.def` is loaded.

Because plain `TeX` users might want to use some of the features of the `babel` system too, care has to be taken that plain `TeX` can process the files. For this reason the current format will have to be checked in a number of places. Some of the code below is common to plain `TeX` and `LaTeX`, some of it is for the `LaTeX` case only.

When the command `\AtBeginDocument` doesn't exist we assume that we are dealing with a plain-based format. In that case the file `plain.def` is needed.

```

10.1 <*kernel | core>
10.2 \ifx\AtBeginDocument\undefined
10.3   \input plain.def\relax
10.4 \fi
10.5 </kernel | core>

```

Check the presence of the command `\iflanguage`, if it is undefined read the file `switch.def`.

```
10.6 <*core>
10.7 \ifx\iflanguage\@undefined
10.8   \input switch.def\relax
10.9 \fi
10.10 </core>
```

10.1 Multiple languages

With \TeX version 3.0 it has become possible to load hyphenation patterns for more than one language. This means that some extra administration has to be taken care of. The user has to know for which languages patterns have been loaded, and what values of `\language` have been used.

Some discussion has been going on in the \TeX world about how to use `\language`. Some have suggested to set a fixed standard, i.e., patterns for each language should *always* be loaded in the same location. It has also been suggested to use the ISO list for this purpose. Others have pointed out that the ISO list contains more than 256 languages, which have *not* been numbered consecutively.

I think the best way to use `\language`, is to use it dynamically. This code implements an algorithm to do so. It uses an external file in which the person who maintains a \TeX environment has to record for which languages he has hyphenation patterns *and* in which files these are stored⁴. When hyphenation exceptions are stored in a separate file this can be indicated by naming that file *after* the file with the hyphenation patterns.

This “configuration file” can contain empty lines and comments, as well as lines which start with an equals (=) sign. Such a line will instruct \LaTeX that the hyphenation patterns just processed have to be known under an alternative name. Here is an example:

```
% File      : language.dat
% Purpose   : tell iniTeX what files with patterns to load.
english     english.hyphenations
=british

dutch       hyphen.dutch exceptions.dutch % Nederlands
german      hyphen.ger
```

As the file `switch.def` needs to be read only once, we check whether it was read before. If it was, the command `\iflanguage` is already defined, so we can stop processing.

```
10.11 <*kernel>
10.12 <!*patterns>
10.13 \expandafter\ifx\csname iflanguage\endcsname\relax \else
10.14 \expandafter\endinput
10.15 \fi
10.16 </!patterns>
```

⁴This is because different operating systems sometimes use *very* different filenames conventions.

`\language` Plain TeX version 3.0 provides the primitive `\language` that is used to store the current language. When used with a pre-3.0 version this function has to be implemented by allocating a counter.

```

10.17 \ifx\language\@undefined
10.18   \csname newcount\endcsname\language
10.19 \fi

```

`\last@language` Another counter is used to store the last language defined. For pre-3.0 formats an extra counter has to be allocated,

```

10.20 \ifx\newlanguage\@undefined
10.21   \csname newcount\endcsname\last@language

```

plain TeX version 3.0 uses `\count 19` for this purpose.

```

10.22 \else
10.23   \countdef\last@language=19
10.24 \fi

```

`\addlanguage` To add languages to TeX's memory plain TeX version 3.0 supplies `\newlanguage`, in a pre-3.0 environment a similar macro has to be provided. For both cases a new macro is defined here, because the original `\newlanguage` was defined to be `\outer`.

For a format based on plain version 2.x, the definition of `\newlanguage` can not be copied because `\count 19` is used for other purposes in these formats. Therefore `\addlanguage` is defined using a definition based on the macros used to define `\newlanguage` in plain TeX version 3.0.

```

10.25 \ifx\newlanguage\@undefined
10.26   \def\addlanguage#1{%
10.27     \global\advance\last@language \@ne
10.28     \ifnum\last@language<\@cclvi
10.29       \else
10.30         \errmessage{No room for a new \string\language!}%
10.31       \fi
10.32     \global\chardef#1\last@language
10.33     \wlog{\string#1 = \string\language\the\last@language}}

```

For formats based on plain version 3.0 the definition of `\newlanguage` can be simply copied, removing `\outer`.

```

10.34 \else
10.35   \def\addlanguage{\alloc@9\language\chardef\@cclvi}
10.36 \fi

```

`\adddialect` The macro `\adddialect` can be used to add the name of a dialect or variant language, for which an already defined hyphenation table can be used.

```

10.37 \def\adddialect#1#2{%
10.38   \global\chardef#1#2\relax
10.39   \wlog{\string#1 = a dialect from \string\language#2}}

```

`\iflanguage` Users might want to test (in a private package for instance) which language is currently active. For this we provide a test macro, `\iflanguage`, that has three arguments. It checks whether the first argument is a known language. If so, it compares the first argument with the value of `\language`. Then, depending on the result of the comparison, it executes either the second or the third argument.

```

10.40 \def\iflanguage#1#2#3{%
10.41   \expandafter\ifx\csname l@#1\endcsname\relax
10.42     \nolater{#1}%
10.43   \else
10.44     \ifnum\csname l@#1\endcsname=\language #2%
10.45     \else#3\fi
10.46   \fi}

```

`\selectlanguage` The macro `\selectlanguage` checks whether the language is already defined before it performs its actual task, which is to update `\language` and activate language-specific definitions.

To allow the call of `\selectlanguage` either with a control sequence name or with a simple string as argument, we have to use a trick to delete the optional escape character.

To convert a control sequence to a string, we use the `\string` primitive. Next we have to look at the first character of this string and compare it with the escape character. Because this escape character can be changed by setting the internal integer `\escapechar` to a character number, we have to compare this number with the character of the string. To do this we have to use TeX's backquote notation to specify the character as a number.

If the first character of the `\string`'ed argument is the current escape character, the comparison has stripped this character and the rest in the 'then' part consists of the rest of the control sequence name. Otherwise we know that either the argument is not a control sequence or `\escapechar` is set to a value outside of the character range 0–255.

If the user gives an empty argument, we provide a default argument for `\string`. This argument should expand to nothing.

```

10.47 \edef\selectlanguage{%
10.48   \noexpand\protect
10.49   \expandafter\noexpand\csname selectlanguage \endcsname
10.50 }

```

Because the command `\selectlanguage` could be used in a moving argument it expands to `\protect\selectlanguage␣`. Therefore, we have to make sure that a macro `\protect` exists. If it doesn't it is `\let` to `\relax`.

```

10.51 \ifx\@undefined\protect\let\protect\relax\fi

```

As L^AT_EX 2.09 writes to files *expanded* whereas L^AT_EX 2_ε takes care *not* to expand the arguments of `\write` statements we need to be a bit clever about the way we add information to `.aux` files. Therefore we introduce the macro `\xstring` which should expand to the right amount of `\string`'s.

```

10.52 \ifx\documentclass\@undefined
10.53   \def\xstring{\string\string\string}
10.54 \else
10.55   \let\xstring\string
10.56 \fi

10.57 \expandafter\def\csname selectlanguage \endcsname#1{%
10.58   \edef\language#1{%
10.59     \ifnum\escapechar=\expandafter'\string#1\@empty
10.60     \else \string#1\@empty\fi}%
10.61   \select@language{\language#1}%

```


We also write a command to change the current language in the auxiliary files.

```

10.62 \if@filesw
10.63   \protected@write\@auxout{}\string\select@language{\language}%
10.64   \addtocontents{toc}{\xstring\select@language{\language}}%
10.65   \addtocontents{lof}{\xstring\select@language{\language}}%
10.66   \addtocontents{lot}{\xstring\select@language{\language}}%
10.67 \fi}

```

First, check if the user asks for a known language. If so, update the value of `\language` and call `\originalTeX` to bring \TeX in a certain pre-defined state.

```

10.68 \def\select@language#1{%
10.69   \expandafter\ifx\csname date#1\endcsname\relax
10.70     \nolatererr{#1}%
10.71   \else
10.72     \language=\csname l@#1\endcsname\relax
10.73     \originalTeX

```

The name of the language is stored in the control sequence `\language`. The contents of this control sequence could be tested in the following way:

```

\edef\tmp{\string english}
\ifx\language\tmp
...
\else
...
\fi

```

The construction with `\string` is necessary because `\language` returns the name with characters of category code 12 (other). Then we have to *redefine* `\originalTeX` to compensate for the things that have been activated. To save memory space for the macro definition of `\originalTeX`, we construct the control sequence name for the `\noextras<lang>` command at definition time by expanding the `\csname` primitive.

```

10.74 \expandafter\def\expandafter\originalTeX
10.75   \expandafter{\csname noextras#1\endcsname
10.76               \let\originalTeX\@empty}%
10.77 \languageshorthands{none}%
10.78 \babel@beginsave

```

Now activate the language-specific definitions. This is done by constructing the names of three macros by concatenating three words with the argument of `\selectlanguage`, and calling these macros.

```

10.79 \csname captions#1\endcsname
10.80 \csname date#1\endcsname
10.81 \csname extras#1\endcsname\relax

```

The switching of the values of `\lefthyphenmin` and `\righthyphenmin` is somewhat different. First we save their current values, then we check if `\<lang>hyphenmins` is defined. If it is not we set default values (2 and 3), otherwise the values in `\<lang>hyphenmins` will be used.

```

10.82 \babel@savevariable\lefthyphenmin
10.83 \babel@savevariable\righthyphenmin
10.84 \expandafter\ifx\csname #1hyphenmins\endcsname\relax

```

```

10.85     \lefthyphenmin\tw@\righthyphenmin\thr@@\relax
10.86   \else
10.87     \expandafter\expandafter\expandafter\set@hyphenmins
10.88       \csname #1hyphenmins\endcsname\relax
10.89   \fi
10.90 \fi}

```

otherlanguage The `otherlanguage` environment can be used as an alternative to using the `\selectlanguage` declarative command. When you are typesetting a document which mixes left-to-right and right-to-left typesetting you have to use this environment in order to let things work as you expect them to.

The first thing this environment does is store the name of the language in `\language`; it then calls `\selectlanguage` to switch on everything that is needed for this language. The `\ignorespaces` command is necessary to hide the environment when it is entered in horizontal mode.

```

10.91 \long\def\otherlanguage#1{%
10.92   \def\language{#1}%
10.93   \csname selectlanguage \endcsname{#1}%
10.94   \ignorespaces
10.95 }

```

The `\endotherlanguage` part of the environment calls `\originalTeX` to restore (most of) the settings and tries to hide itself when it is called in horizontal mode.

```

10.96 \long\def\endotherlanguage{%
10.97   \originalTeX
10.98   \global\@ignoretrue\ignorespaces
10.99 }

```

otherlanguage* The `otherlanguage` environment is meant to be used when a large part of text from a different language needs to be typeset, but without changing the translation of words such as ‘figure’.

This environment makes use of `\foreign@language`.

```

10.100 \expandafter\def\csname otherlanguage*\endcsname#1{%
10.101   \foreign@language{#1}%
10.102 }

```

At the end of the environment we need to switch off the extra definitions. The grouping mechanism of the environment will take care of resetting the correct hyphenation rules.

```

10.103 \expandafter\def\csname endotherlanguage*\endcsname{%
10.104   \csname noextras\language\endcsname
10.105 }

```

\foreignlanguage The `\foreignlanguage` command is another substitute for the `\selectlanguage` command. This command takes two arguments, the first argument is the name of the language to use for typesetting the text specified in the second argument.

Unlike `\selectlanguage` this command doesn’t switch *everything*, it only switches the hyphenation rules and the extra definitions for the language specified. It does this within a group and assumes the `\extras⟨lang⟩` command doesn’t make any `\global` changes. The coding is very similar to part of `\selectlanguage`.

```

10.106 \def\foreignlanguage{\protect\csname foreignlanguage \endcsname}
10.107 \expandafter\def\csname foreignlanguage \endcsname#1#2{%
10.108   \begingroup

```

```

10.109 \foreign@language{#1}%
10.110 #2%
10.111 \csname noextras#1\endcsname
10.112 \endgroup
10.113 }

```

`\foreign@language` This macro does the work for `\foreignlanguage` and the `otherlanguage*` environment.

```

10.114 \def\foreign@language#1{%
10.115 %   First we need to store the name of the language and check that it
10.116 %   is a known language.
10.117 %   \begin{macrocode}
10.118 \def\language#1{%
10.119 \expandafter\ifx\csname l@#1\endcsname\relax
10.120 \nolatererr{#1}%
10.121 \else

```

If it is we can select the proper hyphenation table and switch on the extra definitions for this language.

```

10.122 \language=\csname l@#1\endcsname\relax
10.123 \languageshorthands{none}%

```

Then we set the left- and right hyphenmin variables.

```

10.124 \csname extras#1\endcsname
10.125 \expandafter\ifx\csname #1hyphenmins\endcsname\relax
10.126 \lefthyphenmin\tw@\righthyphenmin\thr@@\relax
10.127 \else
10.128 \expandafter\expandafter\expandafter\set@hyphenmins
10.129 \csname #1hyphenmins\endcsname\relax
10.130 \fi
10.131 \fi
10.132 }

```

`\set@hyphenmins` This macro sets the values of `\lefthyphenmin` and `\righthyphenmin`. It expects two values as its argument.

```

10.133 \def\set@hyphenmins#1#2{\lefthyphenmin#1\righthyphenmin#2}

```

`\LdfInit` This macro is defined in two versions. The first version is to be part of the ‘kernel’ of `babel`, ie. the part that is loaded in the format; the second version is defined in `babel.def`. The version in the format just checks the category code of the ampersand and then loads `babel.def`.

```

10.134 \def\LdfInit{%
10.135 \chardef\atcatcode=\catcode'\@
10.136 \catcode'\@=11\relax
10.137 \input babel.def\relax

```

The category code of the ampersand is restored and the macro calls itself again with the new definition from `babel.def`

```

10.138 \catcode'\@=\atcatcode \let\atcatcode\relax
10.139 \LdfInit}
10.140 </kernel>

```

The second version of this macro takes two arguments. The first argument is the name of the language that will be defined in the language definition file; the second argument is either a control sequence or a string from which a control sequence

should be constructed. The existence of the control sequence indicates that the file has been processed before.

At the start of processing a language definition file we always check the category code of the ampersand. We make sure that it is a ‘letter’ during the processing of the file.

```
10.141 <*core>
10.142 \def\LdfInit#1#2{%
10.143   \chardef\atcatcode=\catcode'\@
10.144   \catcode'\@=11\relax
```

Now we check whether we should perhaps stop the processing of this file. To do this we first need to check whether the second argument that is passed to `\LdfInit` is a control sequence. We do that by looking at the first token after passing `#2` through `string`. When it is equal to `\@backslashchar` we are dealing with a control sequence which we can compare with `\@undefined`.

```
10.145   \let\bbl@tempa\relax
10.146   \expandafter\if\expandafter\@backslashchar
10.147           \expandafter\@car\string#2\@nil
10.148   \ifx#2\@undefined
10.149   \else
```

If so, we call `\ldf@quit` (but after the end of this `\if` construction) to set the main language, restore the category code of the `@`-sign and call `\endinput`.

```
10.150       \def\bbl@tempa{\ldf@quit{#1}}
10.151   \fi
10.152 \else
```

When `#2` was *not* a control sequence we construct one and compare it with `\relax`.

```
10.153   \expandafter\ifx\csname#2\endcsname\relax
10.154   \else
10.155       \def\bbl@tempa{\ldf@quit{#1}}
10.156   \fi
10.157 \fi
10.158 \bbl@tempa
```

Finally we check `\originalTeX`.

```
10.159   \ifx\originalTeX\@undefined
10.160   \let\originalTeX\@empty
10.161 \else
10.162   \originalTeX
10.163 \fi}
```

`\ldf@quit` This macro interrupts the processing of a language definition file.

```
10.164 \def\ldf@quit#1{%
10.165   \expandafter\main@language\expandafter{#1}%
10.166   \catcode'\@=\atcatcode \let\atcatcode\relax
10.167   \endinput
10.168 }
```

`\ldf@finish` This macro takes one argument. It is the name of the language that was defined in the language definition file.

We load the local configuration file if one is present, we set the main language (taking into account that the argument might be a control sequence that needs to be expanded) and reset the category code of the `@`-sign.

```

10.169 \def\ldf@finish#1{%
10.170   \loadlocalcfg{#1}
10.171   \expandafter\main@language\expandafter{#1}%
10.172   \catcode'\@=\atcatcode \let\atcatcode\relax
10.173 }

```

After the preamble of the document the commands `\LdfInit`, `\ldf@quit` and `\ldf@finish` are no longer needed. Therefore they are turned into warning messages in \LaTeX .

```

10.174 \@onlypreamble\LdfInit
10.175 \@onlypreamble\ldf@quit
10.176 \@onlypreamble\ldf@finish

```

`\main@language` This command should be used in the various language definition files. It stores its argument in `\bbl@main@language`; to be used to switch to the correct language at the beginning of the document.

```

10.177 \def\main@language#1{%
10.178   \def\bbl@main@language{#1}%
10.179   \let\language\bbl@main@language
10.180   \language=\csname l@\language\endcsname\relax
10.181 }

```

The default is to use English as the main language.

```

10.182 \ifx\l@english\undefined
10.183   \let\l@english\z@
10.184 \fi
10.185 \main@language{english}

```

We also have to make sure that some code gets executed at the beginning of the document.

```

10.186 \AtBeginDocument{%
10.187   \expandafter\selectlanguage\expandafter{\bbl@main@language}}
10.188 \</core>

```

`\originalTeX` The macro `\originalTeX` should be known to \TeX at this moment. As it has to be expandable we `\let` it to `\@empty` instead of `\relax`.

```

10.189 \*kernel
10.190 \ifx\originalTeX\@undefined\let\originalTeX\@empty\fi

```

Because this part of the code can be included in a format, we make sure that the macro which initialises the save mechanism, `\babel@beginsave`, is not considered to be undefined.

```

10.191 \ifx\babel@beginsave\@undefined\let\babel@beginsave\relax\fi

```

`\@nolanerr` The `babel` package will signal an error when a documents tries to select a language that hasn't been defined earlier. When a user selects a language for which no hyphenation patterns were loaded into the format he will be given a warning about that fact. We revert to the patterns for `\language=0` in that case. In most formats that will be (US)english, but it might also be empty.

When the format knows about `\PackageError` it must be $\text{\LaTeX 2}_{\epsilon}$, so we can safely use its error handling interface. Otherwise we'll have to 'keep it simple'.

```

10.192 \ifx\PackageError\@undefined
10.193   \def\@nolanerr#1{%

```

```

10.194 \errhelp{Your command will be ignored, type <return> to proceed}%
10.195 \errmessage{You haven't defined the language #1\space yet}}
10.196 \def\@nopatterns#1{%
10.197 \message{No hyphenation patterns were loaded for}
10.198 \message{the language '#1'}
10.199 \message{I will use the patterns loaded for \string\language=0
10.200 instead}}
10.201 \def\@activated#1{%
10.202 \wlog{Package babel Info: Making #1 an active character}}
10.203 \else
10.204 \newcommand*\@nolanerr}[1]{%
10.205 \PackageError{babel}%
10.206 {You haven't defined the language #1\space yet}%
10.207 {Your command will be ignored, type <return> to proceed}}
10.208 \newcommand*\@nopatterns}[1]{%
10.209 \PackageWarningNoLine{babel}%
10.210 {No hyphenation patterns were loaded for\MessageBreak
10.211 the language '#1'\MessageBreak
10.212 I will use the patterns loaded for \string\language=0
10.213 instead}}
10.214 \newcommand*\@activated}[1]{%
10.215 \PackageInfo{babel}%
10.216 Making #1 an active character}}
10.217 \fi

```

The following code is meant to be read by `iniTeX` because it should instruct `TeX` to read hyphenation patterns. To this end the `docstrip` option `patterns` can be used to include this code in the file `hyphen.cfg`.

```

10.218 (*patterns)

```

`\process@line` Each line in the file `language.dat` is processed by `\process@line` after it is read. The first thing this macro does is to check whether the line starts with `=`. When the first token of a line is an `=`, the macro `\process@synonym` is called; otherwise the macro `\process@language` will continue.

```

10.219 \def\process@line#1#2/{%
10.220 \ifx=#1
10.221 \process@synonym#2/
10.222 \else
10.223 \process@language#1#2/%
10.224 \fi
10.225 }

```

`\process@synonym` This macro takes care of the lines which start with an `=`.

```

10.226 \def\process@synonym#1 /{%
10.227 \ifnum\last@language=\m@ne

```

When no languages have been loaded yet the name following the `=` will be a synonym for hyphenation register 0.

```

10.228 \expandafter\global
10.229 \expandafter\chardef\csname l@#1\endcsname0\relax
10.230 \wlog{\string\l@#1=\string\language0}
10.231 \else

```

Otherwise the name will be a synonym for the language loaded last.

```

10.232 \expandafter\global
10.233 \expandafter\chardef\csname l@#1\endcsname\last@language
10.234 \wlog{\string\l@#1=\string\language\the\last@language}
10.235 \fi
10.236 }

```

\process@language The macro `\process@language` is used to process a non-empty line from the ‘configuration file’. It has three arguments, each delimited by white space. The third argument is optional, therefore a / character is expected to delimit the last argument. The first argument is the ‘name’ of a language, the second is the name of the file that contains the patterns. The optional third argument is the name of a file containing hyphenation exceptions.

The first thing to do is call `\addlanguage` to allocate a pattern register and to make that register ‘active’.

```

10.237 \def\process@language#1 #2 #3/{%
10.238 \expandafter\addlanguage\csname l@#1\endcsname
10.239 \expandafter\language\csname l@#1\endcsname

```

Then the ‘name’ of the language that will be loaded now is added to the token register `\toks8`. and finally the pattern file is read.

```

10.240 \global\toks8\expandafter{\the\toks8#1, }%

```

Some pattern files contain assignments to `\lefthyphenmin` and `\righthyphenmin`. \TeX does not keep track of these assignments. Therefore we try to detect such assignments and store them in the `\(lang)hyphenmins` macro. When no assignments were made we provide a default setting.

```

10.241 \lefthyphenmin@m@ne
10.242 \input #2\relax
10.243 \ifnum\lefthyphenmin=\m@ne
10.244 \lefthyphenmin\tw@
10.245 \righthyphenmin\thr@@
10.246 \fi

```

When the hyphenation patterns have been processed we need to see if a file with hyphenation exceptions needs to be read. This is the case when the third argument is not empty and when it does not contain a space token.

```

10.247 \def\bbl@tempa{#3}
10.248 \ifx\bbl@tempa@empty
10.249 \else
10.250 \ifx\bbl@tempa\space
10.251 \else
10.252 \input #3\relax
10.253 \fi
10.254 \fi

```

Finally we store the settings of `\lefthyphenmin` and `\righthyphenmin`.

```

10.255 \expandafter\edef\csname #1hyphenmins\endcsname{%
10.256 \the\lefthyphenmin\the\righthyphenmin}}

```

\readconfigfile The configuration file can now be opened for reading.

```

10.257 \openin1 = language.dat

```

See if the file exists, if not, use the default hyphenation file `hyphen.tex`. The user will be informed about this.

```
10.258 \ifeof1
10.259   \message{I couldn't find the file language.dat,\space
10.260             I will try the file hyphen.tex}
10.261   \input hyphen.tex\relax
10.262 \else
```

Pattern registers are allocated using count register `\last@language`. Its initial value is 0. The definition of the macro `\newlanguage` is such that it first increments the count register and then defines the language. In order to have the first patterns loaded in pattern register number 0 we initialize `\last@language` with the value `-1`.

```
10.263   \last@language\m@ne
```

We now read lines from the file until the end is found

```
10.264   \loop
```

While reading from the input it is useful to switch off recognition of the end-of-line character. This saves us stripping off spaces from the contents of the controlsequence.

```
10.265     \endlinechar\m@ne
10.266     \read1 to \bbl@line
10.267     \endlinechar'\^^M
```

Empty lines are skipped.

```
10.268     \ifx\bbl@line\@empty
10.269     \else
```

Now we add a space and a / character to the end of `\bbl@line`. This is needed to be able to recognize the third, optional, argument of `\process@language` later on.

```
10.270       \edef\bbl@line{\bbl@line\space/}
10.271       \expandafter\process@line\bbl@line
10.272   \fi
```

Check for the end of the file. To avoid a new `if` control sequence we create the necessary `\iftrue` or `\iffalse` with the help of `\csname`. But there is one complication with this approach: when skipping the `loop...repeat` T_EX has to read `\if/\fi` pairs. So we have to insert a ‘dummy’ `\iftrue`.

```
10.273     \iftrue \csname fi\endcsname
10.274     \csname if\ifeof1 false\else true\fi\endcsname
10.275   \repeat
```

Reactivate the default patterns,

```
10.276   \language=0
10.277 \fi
```

and close the configuration file.

```
10.278 \closein1
```

Also remove some macros from memory

```
10.279 \let\process@language\@undefined
10.280 \let\process@synonym\@undefined
10.281 \let\process@line\@undefined
```



```

10.282 \let\bbl@tempa\@undefined
10.283 \let\bbl@tempb\@undefined
10.284 \let\bbl@eq\@undefined
10.285 \let\bbl@line\@undefined

```

We add a message about the fact that babel is loaded in the format and with which language patterns to the `\everyjob` register.

```

10.286 \ifx\addto@hook\@undefined
10.287 \else
10.288   \expandafter\addto@hook\expandafter\everyjob\expandafter{%
10.289     \expandafter\typeout\expandafter{\the\toks8 loaded.}}
10.290 \fi

```

Here the code for `iniTeX` ends.

```

10.291 \</patterns>
10.292 \</kernel>

```

10.2 Support for active characters

`\bbl@add@special` The macro `\bbl@add@special` is used to add a new character (or single character control sequence) to the macro `\dospecials` (and `\@sanitize` if `LATEX` is used).

To keep all changes local, we begin a new group. Then we redefine the macros `\do` and `\@makeother` to add themselves and the given character without expansion.

```

10.293 \*core | shorthands
10.294 \def\bbl@add@special#1{\begingroup
10.295   \def\do{\noexpand\do\noexpand}%
10.296   \def\@makeother{\noexpand\@makeother\noexpand}%

```

To add the character to the macros, we expand the original macros with the additional character inside the redefinition of the macros. Because `\@sanitize` can be undefined, we put the definition inside a conditional.

```

10.297   \edef\x{\endgroup
10.298     \def\noexpand\dospecials{\dospecials\do#1}%
10.299     \expandafter\ifx\csname @sanitize\endcsname\relax \else
10.300       \def\noexpand\@sanitize{\@sanitize\@makeother#1}%
10.301     \fi}%

```

The macro `\x` contains at this moment the following:

`\endgroup\def\dospecials{old contents \do⟨char⟩}`.

If `\@sanitize` is defined, it contains an additional definition of this macro. The last thing we have to do, is the expansion of `\x`. Then `\endgroup` is executed, which restores the old meaning of `\x`, `\do` and `\@makeother`. After the group is closed, the new definition of `\dospecials` (and `\@sanitize`) is assigned.

```

10.302 \x}

```

`\bbl@remove@special` The companion of the former macro is `\bbl@remove@special`. It is used to remove a character from the set macros `\dospecials` and `\@sanitize`.

To keep all changes local, we begin a new group. Then we define a help macro `\x`, which expands to empty if the characters match, otherwise it expands to its nonexpandable input. Because `TEX` inserts a `\relax`, if the corresponding `\else` or `\fi` is scanned before the comparison is evaluated, we provide a ‘stop sign’ which should expand to nothing.

```

10.303 \def\bbl@remove@special#1{\begingroup
10.304   \def\x##1##2{\ifnum'#1='##2\noexpand\@empty
10.305     \else\noexpand##1\noexpand##2\fi}%
      With the help of this macro we define \do and \make@other.
10.306   \def\do{x\do}%
10.307   \def\@makeother{x\@makeother}%
      The rest of the work is similar to \bbl@add@special.
10.308   \edef\x{\endgroup
10.309     \def\noexpand\dospecials{\dospecials}%
10.310     \expandafter\ifx\csname @sanitize\endcsname\relax \else
10.311       \def\noexpand\@sanitize{\@sanitize}%
10.312     \fi}%
10.313   \x}

```

10.3 Shorthands

\initiate@active@char A language definition file can call this macro to make a character active. This macro takes one argument, the character that is to be made active. When the character was already active this macro does nothing. Otherwise, this macro defines the control sequence `\normal@char⟨char⟩` to expand to the character in its ‘normal state’ and it defines the active character to expand to `\normal@char⟨char⟩` by default (⟨char⟩ being the character to be made active). Later its definition can be changed to expand to `\active@char⟨char⟩` by calling `\bbl@activate{⟨char⟩}`.

For example, to make the double quote character active one could have the following line in a language definition file:

```
\initiate@active@char{"}
```

\bbl@afterelse Because the code that is used in the handling of active characters may need to look ahead, we take extra care to ‘throw’ it over the `\else` and `\fi` parts of an `\if`-statement⁵.

```

10.314 \def\bbl@afterelse#1\else#2\fi{\fi#1}
10.315 \def\bbl@afterfi#1\fi{\fi#1}

```

\peek@token In order to prevent error messages when a shorthand, which normally takes an argument sees a `\par`, or `}`, or similar tokens we need to be able to ‘peek’ at what is coming up next in the input stream. Depending on the category code of the token that is seen we need to either continue the code for the active character, or insert the non-active version of that character in the output. The macro `\peek@token` therefore takes two arguments, with which it constructs the control sequence to expand next. It `\let`’s `\bbl@nexta` and `\bbl@nextb` to the two possible macro’s. This is necessary for `\bbl@test@token` to take the right decision.

```

10.316 \def\peek@token#1#2{%
10.317   \expandafter\let\expandafter\bbl@nexta\csname #1\string#2\endcsname
10.318   \expandafter\let\expandafter\bbl@nextb
10.319     \csname system@active\string#2\endcsname
10.320   \futurelet\bbl@token\bbl@test@token}

```

⁵This code is based on code presented in TUGboat vol. 12, no2, June 1991 in “An expansion Power Lemma” by Sonja Maus.

`bbl@test@token` When the result of peeking at the next token has yielded a token with category ‘letter’, ‘other’ or ‘active’ it is safe to proceed with evaluating the code for the shorthand. When a token is found with any other category code proceeding is unsafe and therefore the original shorthand character is inserted in the output. The macro that calls `\bbl@test@token` needs to setup `\bbl@nexta` and `\bbl@nextb` in order to achieve this.

```

10.321 \def\bbl@test@token{%
10.322   \let\bbl@next\bbl@nexta
10.323   \ifcat\noexpand\bbl@token a%
10.324   \else
10.325     \ifcat\noexpand\bbl@token=%
10.326     \else
10.327       \ifcat\noexpand\bbl@token\noexpand\bbl@next
10.328       \else
10.329         \let\bbl@next\bbl@nextb
10.330       \fi
10.331     \fi
10.332   \fi
10.333   \bbl@next}

```

Note that the definition of `\initiate@active@char` needs an active character, for this the `~` is used. Some of the changes we need, do not have to become available later on, so we do it inside a group.

```

10.334 \begingroup
10.335   \catcode'\~\active
10.336   \def\x{\endgroup
10.337     \def\initiate@active@char##1{%

```

If the character is already active we provide the default expansion under this shorthand mechanism.

```

10.338       \ifcat\noexpand##1\noexpand~\relax
10.339       \expandafter\edef\csname normal@char\string##1\endcsname{##1}%
10.340       \expandafter\gdef
10.341       \expandafter##1%
10.342       \expandafter{%
10.343       \expandafter\active@prefix\expandafter##1%
10.344       \csname normal@char\string##1\endcsname}
10.345     \else

```

Otherwise we write a message in the transcript file,

```

10.346       \@activated{##1}%

```

and define `\normal@char⟨char⟩` to expand to the character in its default state.

```

10.347       \@namedef{normal@char\string##1}{##1}%

```

If we are making the right quote active we need to change `\pr@m@s` as well.

```

10.348       \ifx##1'%
10.349       \let\pr@m@s\bbl@pr@m@s
10.350     \fi

```

To prevent problems with the loading of other packages after `babel` we reset the catcode of the character at the end of the package.

```

10.351     \ifx\KeepShorthandsActive\@undefined
10.352     \edef\bbl@tempa{\catcode'\noexpand##1\the\catcode'\##1}
10.353     \expandafter\AtEndOfPackage\expandafter{\bbl@tempa}%
10.354   \fi

```

Now we set the lowercase code of the `~` equal to that of the character to be made active and execute the rest of the code inside a `\lowercase` ‘environment’.

```
10.355      \@tempcnta=\lccode'\~
10.356      \lccode'\~='##1%
10.357      \lowercase{%
```

Make the character active and add it to `\dospecials` and `\@sanitize`.

```
10.358      \catcode'\~\active
10.359      \expandafter\bb1\add@special
10.360      \csname \string##1\endcsname
```

Also re-activate it again at `\begin{document}`.

```
10.361      \AtBeginDocument{\catcode'\##1\active}%
```

Define the character to expand to

```
\active@prefix <char> \normal@char<char>
```

(where `\active@char<char>` is *one* control sequence!).

```
10.362      \expandafter\gdef
10.363      \expandafter~%
10.364      \expandafter{%
10.365      \expandafter\active@prefix\expandafter##1%
10.366      \csname normal@char\string##1\endcsname}}%
10.367      \lccode'\~\@tempcnta
10.368      \fi
```

We define the first level expansion of `\active@char<char>` to check the status of the `@safe@actives` flag. If it is set to true we expand to the ‘normal’ version of this character, otherwise we call `\@active@char<char>`.

```
10.369      \@namedef{active@char\string##1}{%
10.370      \if@safe@actives
10.371      \bb1\afterelse\csname normal@char\string##1\endcsname
10.372      \else
10.373      \bb1\afterfi\csname user@active\string##1\endcsname
10.374      \fi}%
```

The next level of the code checks whether a user has defined a shorthand for himself with this character. First we check for a single character shorthand. If that doesn’t exist we check for a shorthand with an argument.

```
10.375      \@namedef{user@active\string##1}{%
10.376      \expandafter\ifx
10.377      \csname \user@group @sh@\string##1\endcsname
10.378      \relax
10.379      \bb1\afterelse\csname @sh@\string##1@sel\endcsname
10.380      {user@active@arg\string##1}{language@active\string##1}%
10.381      \else
10.382      \bb1\afterfi\csname \user@group @sh@\string##1\endcsname
10.383      \fi}%
```

When there is also no user-level shorthand with an argument we will check whether there is a language defined shorthand for this active character. Before the next token is absorbed as argument we need to make sure that this is safe. Therefore `\peek@token` is called to decide that.

```
10.384      \@namedef{user@active@arg\string##1}{%
10.385      \peek@token{@user@active@arg}{##1}}
```

```

10.386 \long\@namedef{@user@active@arg\string##1}####1{%
10.387 \expandafter\ifx
10.388 \csname \user@group @sh@\string##1\string####1@endcsname
10.389 \relax
10.390 \bbl@afterelse
10.391 \csname language@active\string##1@endcsname####1%
10.392 \else
10.393 \bbl@afterfi
10.394 \csname \user@group @sh@\string##1\string####1@%
10.395 \endcsname
10.396 \fi}%

```

Like the shorthands that can be defined by the user, a language definition file can also define shorthands with and without an argument, so we need two more macros to check if they exist.

```

10.397 \@namedef{language@active\string##1}{%
10.398 \expandafter\ifx
10.399 \csname \language@group @sh@\string##1@endcsname
10.400 \relax
10.401 \bbl@afterelse\csname @sh@\string##1@sel@endcsname
10.402 {language@active@arg\string##1}{system@active\string##1}%
10.403 \else
10.404 \bbl@afterfi
10.405 \csname \language@group @sh@\string##1@endcsname
10.406 \fi}%

10.407 \@namedef{language@active@arg\string##1}{%
10.408 \peek@token{@language@active@arg}{##1}}
10.409 \long\@namedef{@language@active@arg\string##1}####1{%
10.410 \expandafter\ifx
10.411 \csname \language@group @sh@\string##1\string####1@endcsname
10.412 \relax
10.413 \bbl@afterelse
10.414 \csname system@active\string##1@endcsname####1%
10.415 \else
10.416 \bbl@afterfi
10.417 \csname \language@group @sh@\string##1\string####1@%
10.418 \endcsname
10.419 \fi}%

```

And the same goes for the system level.

```

10.420 \@namedef{system@active\string##1}{%
10.421 \expandafter\ifx
10.422 \csname \system@group @sh@\string##1@endcsname
10.423 \relax
10.424 \bbl@afterelse\csname @sh@\string##1@sel@endcsname
10.425 {system@active@arg\string##1}{normal@char\string##1}%
10.426 \else
10.427 \bbl@afterfi\csname \system@group @sh@\string##1@endcsname
10.428 \fi}%

```

When no shorthands were found the ‘normal’ version of the active character is inserted.

```

10.429 \@namedef{system@active@arg\string##1}{%
10.430 \peek@token{@system@active@arg}{##1}}

```

```

10.431 \long\@namedef{@system@active@arg\string##1}####1{%
10.432 \expandafter\ifx
10.433 \csname \system@group @sh@\string##1\string####1\endcsname
10.434 \relax
10.435 \bbl@afterelse\csname normal@char\string##1\endcsname####1%
10.436 \else
10.437 \bbl@afterfi
10.438 \csname \system@group @sh@\string##1\string####1\endcsname
10.439 \fi}%
10.440 }%
10.441 }\x

```

\active@prefix The command **\active@prefix** which is used in the expansion of active characters has a function similar to **\OT1-cmd** in that it **\protects** the active character whenever **\protect** is *not* **\@typeset@protect**.

```

10.442 \def\active@prefix#1{%
10.443 \ifx\protect\@typeset@protect
10.444 \else
10.445 \bbl@afterfi\protect#1\@gobble
10.446 \fi}

```

\if@safe@actives In some circumstances it is necessary to be able to change the expansion of an active character on the fly. For this purpose the switch **@safe@actives** is available. This setting of this switch should be checked in the first level expansion of **\active@char***<char>*.

```

10.447 \newif\if@safe@actives
10.448 \@safe@activesfalse

```

\bbl@activate This macro takes one argument, like **\initiate@active@char**. The macro is used to change the definition of an active character to expand to **\active@char***<char>* instead of **\normal@char***<char>*.

```

10.449 \def\bbl@activate#1{%
10.450 \expandafter\def
10.451 \expandafter#1\expandafter{%
10.452 \expandafter\active@prefix
10.453 \expandafter#1\csname active@char\string#1\endcsname}%
10.454 }

```

\bbl@deactivate This macro takes one argument, like **\bbl@ctivate**. The macro doesn't really make a character non-active; it changes its definition to expand to **\normal@char***<char>*.

```

10.455 \def\bbl@deactivate#1{%
10.456 \expandafter\def
10.457 \expandafter#1\expandafter{%
10.458 \expandafter\active@prefix
10.459 \expandafter#1\csname normal@char\string#1\endcsname}%
10.460 }

```

\bbl@firstcs These macros have two arguments. They use one of their arguments to build a **\bbl@scndcs** control sequence from.

```

10.461 \def\bbl@firstcs#1#2{\csname#1\endcsname}
10.462 \def\bbl@scndcs#1#2{\csname#2\endcsname}

```

`\declare@shorthand` The command `\declare@shorthand` is used to declare a shorthand on a certain level. It takes three arguments:

1. a name for the collection of shorthands, i.e. ‘system’, or ‘dutch’;
2. the character (sequence) that makes up the shorthand, i.e. `~` or `"a`;
3. the code to be executed when the shorthand is encountered.

```

10.463 \def\declare@shorthand#1#2{\@decl@short{#1}#2\@nil}
10.464 \def\@decl@short#1#2#3\@nil#4{%
10.465   \def\bbl@tempa{#3}%
10.466   \ifx\bbl@tempa\@empty
10.467     \expandafter\let\csname @sh@\string#2@sel\endcsname\bbl@scndcs
10.468   \else
10.469     \expandafter\let\csname @sh@\string#2@sel\endcsname\bbl@firstcs
10.470   \fi
10.471   \@namedef{#1@sh@\string#2\string#3@}{#4}}

```

`\textormath` Some of the shorthands that will be declared by the language definition files have to be useable in both text and mathmode. To achieve this the helper macro `\textormath` is provided.

```

10.472 \def\textormath#1#2{%
10.473   \ifmmode
10.474     \bbl@afterelse#2%
10.475   \else
10.476     \bbl@afterfi#1%
10.477   \fi}

```

`\user@group` The current concept of ‘shorthands’ supports three levels or groups of shorthands.
`\language@group` For each level the name of the level or group is stored in a macro. The default
`\system@group` is to have no user group; use language group ‘english’ and have a system group called ‘system’.

```

10.478 \def\user@group{}
10.479 \def\language@group{english}
10.480 \def\system@group{system}

```

`\useshorthands` This is the user level command to tell L^AT_EX that user level shorthands will be used in the document. It takes one argument, the character that starts a shorthand.

```

10.481 \def\useshorthands#1{%
10.482   \def\user@group{user}%
10.483   \initiate@active@char{#1}%
10.484   \bbl@activate{#1}}

```

`\defineshorthand` Currently we only support one group of user level shorthands, called ‘user’.

```

10.485 \def\defineshorthand{\declare@shorthand{user}}

```

`\languageshorthands` A user level command to change the language from which shorthands are used.

```

10.486 \def\languageshorthands#1{\def\language@group{#1}}

```

`\aliasshorthand` Because we deal with active characters here we need to use the `\lccode` trick. Therefore we save the current `\lccode` of the `~`-character and restore it later. Then we make the new character active and `\let` it be equal to the original.

```

10.487 \def\aliasshorthand#1#2{%
10.488   \@tempcnta\lccode'\~
10.489   \lccode'\~='#2%
10.490   \lowercase{\catcode'\~\active\let~#1\catcode'\#112\relax}%
10.491   \lccode'\~\@tempcnta}

```

To prevent problems with constructs such as `\char"01A` when the double quote is made active, we define a shorthand on system level.

```

10.492 \declare@shorthand{system}{"}{\csname normal@char\string"\endcsname}

```

When the right quote is made active we need to take care of handling it correctly in mathmode. Therefore we define a shorthand at system level to make it expand to a non-active right quote in textmode, but expand to its original definition in mathmode. (Note that the right quote is ‘active’ in mathmode because of its mathcode.)

```

10.493 \declare@shorthand{system}{'}{%
10.494   \textormath{\csname normal@char\string'\endcsname}%
10.495   {\sp\bgroup\prim@s}}

```

When the left quote is made active we need to take care of handling it correctly when it is followed by for instance an open brace token. Therefore we define a shorthand at system level to make it expand to a non-active left quote.

```

10.496 \declare@shorthand{system}{'}{\csname normal@char\string'\endcsname}

```

`\bbl@pr@m@s` One of the internal macros that are involved in substituting `\prime` for each right quote in mathmode is `\pr@m@s`. This checks if the next character is a right quote. When the right quote is active, the definition of this macro needs to be adapted to look for an active right quote.

```

10.497 \begingroup
10.498   \catcode'\'\active\let'\relax
10.499   \def\x{\endgroup
10.500     \def\bbl@pr@m@s{%
10.501       \ifx'\@let@token
10.502         \expandafter\pr@@@s
10.503       \else
10.504         \ifx~\@let@token
10.505           \expandafter\expandafter\expandafter\pr@@@t
10.506         \else
10.507           \egroup
10.508         \fi
10.509       \fi}%
10.510   }
10.511 \x
10.512 </core | shorthands>

```

Normally the `~` is active and expands to `\penalty\@M__`. When it is written to the `.aux` file it is written expanded. To prevent that and to be able to use the character `~` as a start character for a shorthand, it is redefined here as a one character shorthand on system level.

```

10.513 % \changes{babel~3.5f}{1996/04/02}{No need to reset the category code
10.514 %   ofg the tilde as \cs{initiate@active@char now cooreclty deals
10.515 %   with active characters}
10.516 < *core>

```



```

10.517 \initiate@active@char{~}
10.518 \declare@shorthand{system}{~}{\penalty\@M\ }
10.519 \bbl@activate{~}

```

`\OT1dqpos` The position of the double quote character is different for the OT1 and T1 encodings. It will later be selected using the `\f@encoding` macro. Therefore we define two macros here to store the position of the character in these encodings.

```

10.520 \expandafter\def\csname OT1dqpos\endcsname{127}
10.521 \expandafter\def\csname T1dqpos\endcsname{4}

```

When the macro `\f@encoding` is undefined (as it is in plain `TeX`) we define it here to expand to `OT1`

```

10.522 \ifx\f@encoding\undefined
10.523   \def\f@encoding{OT1}
10.524 \fi

```

10.4 Support for saving macro definitions

To save the meaning of control sequences using `\babel@save`, we use temporary control sequences. To save hash table entries for these control sequences, we don't use the name of the control sequence to be saved to construct the temporary name. Instead we simply use the value of a counter, which is reset to zero each time we begin to save new values. This works well because we release the saved meanings before we begin to save a new set of control sequence meanings (see `\selectlanguage` and `\originalTeX`).

`\babel@savecnt` The initialization of a new save cycle: reset the counter to zero.

```

\babel@beginsave{0.525 \def\babel@beginsave{\babel@savecnt\z@}

```

Before it's forgotten, allocate the counter and initialize all.

```

10.526 \newcount\babel@savecnt
10.527 \babel@beginsave

```

`\babel@save` The macro `\babel@save{csname}` saves the current meaning of the control sequence `csname` to `\originalTeX`⁶. To do this, we let the current meaning to a temporary control sequence, the restore commands are appended to `\originalTeX` and the counter is incremented.

```

10.528 \def\babel@save#1{%
10.529   \expandafter\let\csname babel@number\babel@savecnt\endcsname #1\relax
10.530   \begingroup
10.531     \toks@\expandafter{\originalTeX \let#1=}
10.532     \edef\x{\endgroup
10.533       \def\noexpand\originalTeX{\the\toks@ \expandafter\noexpand
10.534         \csname babel@number\babel@savecnt\endcsname\relax}}
10.535     \x
10.536     \advance\babel@savecnt\@ne}

```

`\babel@savevariable` The macro `\babel@savevariable{variable}` saves the value of the variable. `variable` can be anything allowed after the `\the` primitive.

```

10.537 \def\babel@savevariable#1{\begingroup
10.538   \toks@\expandafter{\originalTeX #1=}

```

⁶`\originalTeX` has to be expandable, i. e. you shouldn't let it to `\relax`.

```

10.539 \edef\x{\endgroup
10.540 \def\noexpand\originalTeX{\the\toks@ \the#1\relax}}%
10.541 \x}

```

`\bbl@frenchspacing` Some languages need to have `\frenchspacing` in effect. Others don't want that.
`\bbl@nonfrenchspacing` The command `\bbl@frenchspacing` switches it on when it isn't already in effect and `\bbl@nonfrenchspacing` switches it off if necessary.

```

10.542 \def\bbl@frenchspacing{%
10.543 \ifnum\the\sfcodes'\.=\@m
10.544 \let\bbl@nonfrenchspacing\relax
10.545 \else
10.546 \frenchspacing
10.547 \let\bbl@nonfrenchspacing\nonfrenchspacing
10.548 \fi}
10.549 \let\bbl@nonfrenchspacing\nonfrenchspacing

```

10.5 Support for extending macros

`\addto` For each language four control sequences have to be defined that control the language-specific definitions. To be able to add something to these macro once they have been defined the macro `\addto` is introduced. It takes two arguments, a *control sequence* and T_EX-code to be added to the *control sequence*.

If the *control sequence* has not been defined before it is defined now.

```

10.550 \def\addto#1#2{%
10.551 \ifx#1\@undefined
10.552 \def#1{#2}
10.553 \else

```

The control sequence could also expand to `\relax`, in which case a circular definition results. The net result is a stack overflow.

```

10.554 \ifx#1\relax
10.555 \def#1{#2}
10.556 \else

```

Otherwise the replacement text for the *control sequence* is expanded and stored in a token register, together with the T_EX-code to be added. Finally the *control sequence* is redefined, using the contents of the token register.

```

10.557 {\toks@\expandafter{#1#2}}%
10.558 \xdef#1{\the\toks@}%
10.559 \fi
10.560 \fi
10.561 }

```

10.6 Macros common to a number of languages

`\allowhyphens` This macro makes hyphenation possible. Basically its definition is nothing more than `\nobreak \hskip 0pt plus 0pt`⁷.

```

10.562 \def\allowhyphens{\penalty\@M \hskip\z@skip}

```

⁷T_EX begins and ends a word for hyphenation at a glue node. The penalty prevents a linebreak at this glue node.

`\set@low@box` The following macro is used to lower quotes to the same level as the comma. It prepares its argument in box register 0.

```
10.563 \def\set@low@box#1{\setbox\tw@\hbox{,}\setbox\z@\hbox{#1}%
10.564 \dimen\z@\ht\z@ \advance\dimen\z@ -\ht\tw@%
10.565 \setbox\z@\hbox{\lower\dimen\z@ \box\z@}\ht\z@\ht\tw@ \dp\z@\dp\tw@}
```

`\save@sf@q` The macro `\save@sf@q` is used to save and reset the current space factor.

```
10.566 \def\save@sf@q#1{{\ifhmode
10.567 \edef\@SF{\spacefactor\the\spacefactor}\else
10.568 \let\@SF\empty \fi \leavevmode #1\@SF}}
```

`\bbl@disc` For some languages the macro `\bbl@disc` is used to ease the insertion of discretionary for letters that behave ‘abnormally’ at a breakpoint.

```
10.569 \def\bbl@disc#1#2{%
10.570 \penalty\@M\discretionary{#2-}{#1}\allowhyphens}
```

10.7 Making glyphs available

The file `babel.dtx`⁸ makes a number of glyphs available that either do not exist in the OT1 encoding and have to be ‘faked’, or that are not accessible through `T1enc.def`.

10.8 Quotation marks

`\quotedblbase` In the T1 encoding the opening double quote at the baseline is available as a separate character, accessible via `\quotedblbase`. In the OT1 encoding it is not available, therefore we make it available by lowering the normal open quote character to the baseline.

```
10.571 \ProvideTextCommand{\quotedblbase}{OT1}{%
10.572 \save@sf@q{\set@low@box{\textquotedblright\}%
10.573 \box\z@\kern-.04em\allowhyphens}}
```

Make sure that when an encoding other than OT1 or T1 is used this glyph can still be typeset.

```
10.574 \ProvideTextCommandDefault{\quotedblbase}{%
10.575 \UseTextSymbol{OT1}{\quotedblbase}}
```

`\quotesinglbase` We also need the single quote character at the baseline.

```
10.576 \ProvideTextCommand{\quotesinglbase}{OT1}{%
10.577 \save@sf@q{\set@low@box{\textquoteright\}%
10.578 \box\z@\kern-.04em\allowhyphens}}
```

Make sure that when an encoding other than OT1 or T1 is used this glyph can still be typeset.

```
10.579 \ProvideTextCommandDefault{\quotesinglbase}{%
10.580 \UseTextSymbol{OT1}{\quotesinglbase}}
```

⁸The file described in this section has version number v3.6j, and was last revised on 1998/03/24.

`\guillemotleft` The guillemot characters are not available in OT1 encoding. They are faked.

```

\guillemotright%0.581 \ProvideTextCommand{\guillemotleft}{OT1}{%
    10.582 \ifmmode
    10.583 \ll
    10.584 \else
    10.585 \save@sf@q{\penalty\@M
    10.586 \raise.2ex\hbox{$\scriptscriptstyle\ll$}\allowhyphens}%
    10.587 \fi}
    10.588 \ProvideTextCommand{\guillemotright}{OT1}{%
    10.589 \ifmmode
    10.590 \gg
    10.591 \else
    10.592 \save@sf@q{\penalty\@M
    10.593 \raise.2ex\hbox{$\scriptscriptstyle\gg$}\allowhyphens}%
    10.594 \fi}

    Make sure that when an encoding other then OT1 ot T1 is used these glyphs can
    still be typeset.

    10.595 \ProvideTextCommandDefault{\guillemotleft}{%
    10.596 \UseTextSymbol{OT1}{\guillemotleft}}
    10.597 \ProvideTextCommandDefault{\guillemotright}{%
    10.598 \UseTextSymbol{OT1}{\guillemotright}}

```

`\guilsinglleft` The single guillemots are not available in OT1 encoding. They are faked.

```

\guilsinglright%0.599 \ProvideTextCommand{\guilsinglleft}{OT1}{%
    10.600 \ifmmode
    10.601 <%
    10.602 \else
    10.603 \save@sf@q{\penalty\@M
    10.604 \raise.2ex\hbox{$\scriptscriptstyle<$}\allowhyphens}%
    10.605 \fi}
    10.606 \ProvideTextCommand{\guilsinglright}{OT1}{%
    10.607 \ifmmode
    10.608 >%
    10.609 \else
    10.610 \save@sf@q{\penalty\@M
    10.611 \raise.2ex\hbox{$\scriptscriptstyle>$}\allowhyphens}%
    10.612 \fi}

    Make sure that when an encoding other then OT1 ot T1 is used these glyphs can
    still be typeset.

    10.613 \ProvideTextCommandDefault{\guilsinglleft}{%
    10.614 \UseTextSymbol{OT1}{\guilsinglleft}}
    10.615 \ProvideTextCommandDefault{\guilsinglright}{%
    10.616 \UseTextSymbol{OT1}{\guilsinglright}}

```

10.9 Letters

`\ij` The dutch language uses the letter ‘ij’. It is available in T1 encoded fonts, but not `\IJ` in the OT1 encoded fonts. Therefore we fake it for the OT1 encoding.

```

    10.617 \DeclareTextCommand{\ij}{OT1}{%
    10.618 \allowhyphens i\kern-0.02em j\allowhyphens}
    10.619 \DeclareTextCommand{\IJ}{OT1}{%
    10.620 \allowhyphens I\kern-0.02em J\allowhyphens}

```

```

10.621 \DeclareTextCommand{\ij}{T1}{\char188}
10.622 \DeclareTextCommand{\IJ}{T1}{\char156}

```

Make sure that when an encoding other than OT1 or T1 is used these glyphs can still be typeset.

```

10.623 \ProvideTextCommandDefault{\ij}{%
10.624   \UseTextSymbol{OT1}{\ij}}
10.625 \ProvideTextCommandDefault{\IJ}{%
10.626   \UseTextSymbol{OT1}{\IJ}}

```

`\dj` The croatian language needs the letters `\dj` and `\DJ`; they are available in the T1 encoding, but not in the OT1 encoding by default.

Some code to construct these glyphs for the OT1 encoding was made available to me by Stipcevic Mario, (stipcevic@olimp.irb.hr).

```

10.627 \def\crrtic@{\hrule height0.1ex width0.3em}
10.628 \def\crttic@{\hrule height0.1ex width0.33em}
10.629 %
10.630 \def\ddj@{%
10.631   \setbox0\hbox{d}\dimen@=\ht0
10.632   \advance\dimen@1ex
10.633   \dimen@.45\dimen@
10.634   \dimen@ii\expandafter\rem@pt\the\fontdimen\@ne\font\dimen@
10.635   \advance\dimen@ii.5ex
10.636   \leavevmode\rlap{\raise\dimen@\hbox{\kern\dimen@ii\vbox{\crrtic@}}}}
10.637 \def\DDJ@{%
10.638   \setbox0\hbox{D}\dimen@=.55\ht0
10.639   \dimen@ii\expandafter\rem@pt\the\fontdimen\@ne\font\dimen@
10.640   \advance\dimen@ii.15ex % correction for the dash position
10.641   \advance\dimen@ii-.15\fontdimen7\font % correction for cmtt font
10.642   \dimen\thr@@\expandafter\rem@pt\the\fontdimen7\font\dimen@
10.643   \leavevmode\rlap{\raise\dimen@\hbox{\kern\dimen@ii\vbox{\crttic@}}}}
10.644 %
10.645 \DeclareTextCommand{\dj}{OT1}{\ddj@ d}
10.646 \DeclareTextCommand{\DJ}{OT1}{\DDJ@ D}

```

Make sure that when an encoding other than OT1 or T1 is used these glyphs can still be typeset.

```

10.647 \ProvideTextCommandDefault{\dj}{%
10.648   \UseTextSymbol{OT1}{\dj}}
10.649 \ProvideTextCommandDefault{\DJ}{%
10.650   \UseTextSymbol{OT1}{\DJ}}

```

10.10 Shorthands for quotation marks

Shorthands are provided for a number of different quotation marks, which make them useable both outside and inside mathmode.

`\glq` The ‘german’ single quotes.

```

\grq10.651 \DeclareRobustCommand{\glq}{%
10.652   \textormath{\quotesinglbase}{\mbox{\quotesinglbase}}}
10.653 \DeclareRobustCommand{\grq}{%
10.654   \textormath{\kern-.0125em\textquoteleft\kern.07em}%
10.655     {\mbox{\textquoteleft}}}

```

`\glqq` The ‘german’ double quotes.

```

\grqq10.656 \DeclareRobustCommand{\glqq}{%
10.657   \textormath{\quotedblbase}{\mbox{\quotedblbase}}}
10.658 \DeclareRobustCommand{\grqq}{%
10.659   \textormath{\save@sf@q{\kern-.07em\textquotedblleft\kern.07em}}%
10.660   {\mbox{\textquotedblleft}}}
10.661 }
```

`\flq` The ‘french’ single guillemets.

```

\frq10.662 \DeclareRobustCommand{\flq}{%
10.663   \textormath{\guilsinglleft}{\mbox{\guilsinglleft}}}
10.664 \DeclareRobustCommand{\frq}{%
10.665   \textormath{\guilsinglright}{\mbox{\guilsinglright}}}
```

`\flqq` The ‘french’ double quillemts.

```

\frqq10.666 \DeclareRobustCommand{\flqq}{%
10.667   \textormath{\guillemotleft}{\mbox{\guillemotleft}}}
10.668 \DeclareRobustCommand{\frqq}{%
10.669   \textormath{\guillemotright}{\mbox{\guillemotright}}}
```

10.11 Umlauts and trema’s

The command `\` needs to have a different effect for different languages. For German for instance, the ‘umlaut’ should be positioned lower than the default position for placing it over the letters a, o, u, A, O and U. When placed over an e, i, E or I it can retain its normal position. For Dutch the same glyph is always placed in the lower position.

`\umlauthigh` To be able to provide both positions of `\` we provide two commands to switch
`\umlautlow` the positioning, the default will be `\umlauthigh` (the normal positioning).

```

10.670 \def\umlauthigh{%
10.671   \def\bbl@umlauta##1{%
10.672     \expandafter\accent\csname\fontencoding dqpos\endcsname
10.673     ##1\allowhyphens}}%
10.674   \let\bbl@umlaute\bbl@umlauta}
10.675 \def\umlautlow{%
10.676   \def\bbl@umlauta{\protect\lower@umlaut}}
10.677 \def\umlaute\low{%
10.678   \def\bbl@umlaute{\protect\lower@umlaut}}
10.679 \umlauthigh
```

`\lower@umlaut` The command `\lower@umlaut` is used to position the `\` closer the the letter.

We want the umlaut character lowered, nearer to the letter. To do this we need an extra *dimen* register.

```

10.680 \expandafter\ifx\csname U@D\endcsname\relax
10.681   \csname newdimen\endcsname\U@D
10.682 \fi
```

The following code fools T_EX’s `make_accent` procedure about the current x-height of the font to force another placement of the umlaut character.

```

10.683 \def\lower@umlaut#1{%
```

First we have to save the current x-height of the font, because we'll change this font dimension and this is always done globally.

```
10.684 {\U@D 1ex%
```

Then we compute the new x-height in such a way that the umlaut character is lowered to the base character. The value of `.45ex` depends on the METAFONT parameters with which the fonts were built. (Just try out, which value will look best.)

```
10.685 {\setbox\z@\hbox{%
10.686     \expandafter\char\csname\f@encoding dqpos\endcsname}%
10.687     \dimen@ -.45ex\advance\dimen@\ht\z@
```

If the new x-height is too low, it is not changed.

```
10.688 \ifdim 1ex<\dimen@ \fontdimen5\font\dimen@ \fi}%
```

Finally we call the `\accent` primitive, reset the old x-height and insert the base character in the argument.

```
10.689 \expandafter\accent\csname\f@encoding dqpos\endcsname
10.690 \fontdimen5\font\U@D #1}}
```

For all vowels we declare `\` to be a composite command which uses `\bbl@umlauta` or `\bbl@umlaute` to position the umlaut character. We need to be sure that these definitions override the ones that are provided when the package `fontenc` with option `OT1` is used. Therefore these declarations are postponed until the beginning of the document.

```
10.691 \AtBeginDocument{%
10.692   \DeclareTextCompositeCommand{"}{OT1}{a}{\bbl@umlauta{a}}%
10.693   \DeclareTextCompositeCommand{"}{OT1}{e}{\bbl@umlaute{e}}%
10.694   \DeclareTextCompositeCommand{"}{OT1}{i}{\bbl@umlaute{i}}%
10.695   \DeclareTextCompositeCommand{"}{OT1}{\i}{\bbl@umlaute{i}}%
10.696   \DeclareTextCompositeCommand{"}{OT1}{o}{\bbl@umlauta{o}}%
10.697   \DeclareTextCompositeCommand{"}{OT1}{u}{\bbl@umlauta{u}}%
10.698   \DeclareTextCompositeCommand{"}{OT1}{A}{\bbl@umlauta{A}}%
10.699   \DeclareTextCompositeCommand{"}{OT1}{E}{\bbl@umlaute{E}}%
10.700   \DeclareTextCompositeCommand{"}{OT1}{I}{\bbl@umlaute{I}}%
10.701   \DeclareTextCompositeCommand{"}{OT1}{O}{\bbl@umlauta{O}}%
10.702   \DeclareTextCompositeCommand{"}{OT1}{U}{\bbl@umlauta{U}}%
10.703 }
```

10.12 The redefinition of the style commands

The rest of the code in this file can only be processed by L^AT_EX, so we check the current format. If it is plain T_EX, processing should stop here. But, because of the need to limit the scope of the definition of `\format`, a macro that is used locally in the following `\if` statement, this comparison is done inside a group. To prevent T_EX from complaining about an unclosed group, the processing of the command `\endinput` is deferred until after the group is closed. This is accomplished by the command `\aftergroup`.

```
10.704 {\def\format{plain}
10.705 \ifx\fmtname\format
10.706 \else
10.707   \def\format{LaTeX2e}
10.708   \ifx\fmtname\format
```

```

10.709 \else
10.710 \aftergroup\endinput
10.711 \fi
10.712 \fi}

```

Now that we're sure that the code is seen by L^AT_EX only, we have to find out what the main (primary) document style is because we want to redefine some macros. This is only necessary for releases of L^AT_EX dated before december 1991. Therefore this part of the code can optionally be included in `babel.def` by specifying the `docstrip` option `names`.

```

10.713 (*names)

```

The standard styles can be distinguished by checking whether some macros are defined. In table 1 an overview is given of the macros that can be used for this purpose.

article	:	both the <code>\chapter</code> and <code>\opening</code> macros are undefined
report and book	:	the <code>\chapter</code> macro is defined and the <code>\opening</code> is undefined
letter	:	the <code>\chapter</code> macro is undefined and the <code>\opening</code> is defined

Table 1: How to determine the main document style

The macros that have to be redefined for the `report` and `book` document styles happen to be the same, so there is no need to distinguish between those two styles.

`\doc@style` First a parameter `\doc@style` is defined to identify the current document style. This parameter might have been defined by a document style that already uses macros instead of hard-wired texts, such as `artikel1.sty` [6], so the existence of `\doc@style` is checked. If this macro is undefined, i. e., if the document style is unknown and could therefore contain hard-wired texts, `\doc@style` is defined to the default value '0'.

```

10.714 \ifx\@undefined\doc@style
10.715 \def\doc@style{0}%

```

This parameter is defined in the following `if` construction (see table 1):

```

10.716 \ifx\@undefined\opening
10.717 \ifx\@undefined\chapter
10.718 \def\doc@style{1}%
10.719 \else
10.720 \def\doc@style{2}%
10.721 \fi
10.722 \else
10.723 \def\doc@style{3}%
10.724 \fi%
10.725 \fi%

```


10.12.1 Redefinition of macros

Now here comes the real work: we start to redefine things and replace hard-wired texts by macros. These redefinitions should be carried out conditionally, in case it has already been done.

For the `figure` and `table` environments we have in all styles:

```
10.726 \@ifundefined{figurename}{\def\fnun@figure{\figurename} \thefigure}}{}
10.727 \@ifundefined{tablename}{\def\fnun@table{\tablename} \thetable}}{}
```

The rest of the macros have to be treated differently for each style. When `\doc@style` still has its default value nothing needs to be done.

```
10.728 \ifcase \doc@style\relax
10.729 \or
```

This means that `babel.def` is read after the `article` style, where no `\chapter` and `\opening` commands are defined⁹.

First we have the `\tableofcontents`, `\listoffigures` and `\listoftables`:

```
10.730 \@ifundefined{contentsname}%
10.731   {\def\tableofcontents{\section*{\contentsname\mkboth
10.732     {\uppercase{\contentsname}}{\uppercase{\contentsname}}}%
10.733     \@starttoc{toc}}}{}}
10.734
10.735 \@ifundefined{listfigurename}%
10.736   {\def\listoffigures{\section*{\listfigurename\mkboth
10.737     {\uppercase{\listfigurename}}{\uppercase{\listfigurename}}}%
10.738     \@starttoc{lof}}}{}}
10.739
10.740 \@ifundefined{listtablename}%
10.741   {\def\listoftables{\section*{\listtablename\mkboth
10.742     {\uppercase{\listtablename}}{\uppercase{\listtablename}}}%
10.743     \@starttoc{lot}}}{}}
```

Then the `\thebibliography` and `\theindex` environments.

```
10.744 \@ifundefined{refname}%
10.745   {\def\thebibliography#1{\section*{\refname
10.746     \mkboth{\uppercase{\refname}}{\uppercase{\refname}}}%
10.747     \list{[\arabic{enumi}]}{\settowidth\labelwidth{[#1]}%
10.748       \leftmargin\labelwidth
10.749       \advance\leftmargin\labelsep
10.750       \usecounter{enumi}}}%
10.751     \def\newblock{\hskip.11em plus.33em minus.07em}%
10.752     \sloppy\clubpenalty4000\widowpenalty\clubpenalty
10.753     \sfcode'\.=1000\relax}}{}
10.754
10.755 \@ifundefined{indexname}%
10.756   {\def\theindex{\@restonecoltrue\if@twocolumn\@restonecolfalse\fi
10.757     \columnseprule \z@
10.758     \columnsep 35pt\twocolumn[\section*{\indexname}}%
10.759     \mkboth{\uppercase{\indexname}}{\uppercase{\indexname}}}%
10.760     \thispagestyle{plain}%
10.761     \parskip\z@ plus.3pt\parindent\z@\let\item\@idxitem}}{}}
```

⁹A fact that was pointed out to me by Nico Poppelier and was already used in Piet van Oostrum's document style option `nl`.

The abstract environment:

```

10.762 \@ifundefined{abstractname}%
10.763     {\def\abstract{\if@twocolumn
10.764         \section*{\abstractname}%
10.765         \else \small
10.766         \begin{center}%
10.767         {\bf \abstractname\vspace{-.5em}\vspace{\z@}}%
10.768         \end{center}%
10.769         \quotation
10.770         \fi}}{}

```

And last but not least, the macro \part:

```

10.771 \@ifundefined{partname}%
10.772 {\def\@part[#1]#2{\ifnum \c@secnumdepth >\m@ne
10.773     \refstepcounter{part}%
10.774     \addcontentsline{toc}{part}{\thepart
10.775     \hspace{1em}#1}\else
10.776     \addcontentsline{toc}{part}{#1}\fi
10.777     {\parindent\z@ \raggedright
10.778     \ifnum \c@secnumdepth >\m@ne
10.779     \Large \bf \partname{} \thepart
10.780     \par \nobreak
10.781     \fi
10.782     \huge \bf
10.783     #2\markboth{}{}\par}%
10.784     \nobreak
10.785     \vskip 3ex\@afterheading}%
10.786 }{}

```

This is all that needs to be done for the `article` style.

10.787 \or

The next case is formed by the two styles `book` and `report`. Basically we have to do the same as for the `article` style, except now we must also change the `\chapter` command.

The tables of contents, figures and tables:

```

10.788 \@ifundefined{contentsname}%
10.789     {\def\tableofcontents{\@restonecolfalse
10.790         \if@twocolumn\@restonecoltrue\onecolumn
10.791         \fi\chapter*{\contentsname\@mkboth
10.792             {\uppercase{\contentsname}}{\uppercase{\contentsname}}}%
10.793         \@starttoc{toc}%
10.794         \csname if@restonecol\endcsname\twocolumn
10.795         \csname fi\endcsname}}{}
10.796
10.797 \@ifundefined{listfigurename}%
10.798     {\def\listoffigures{\@restonecolfalse
10.799         \if@twocolumn\@restonecoltrue\onecolumn
10.800         \fi\chapter*{\listfigurename\@mkboth
10.801             {\uppercase{\listfigurename}}{\uppercase{\listfigurename}}}%
10.802         \@starttoc{lof}%
10.803         \csname if@restonecol\endcsname\twocolumn
10.804         \csname fi\endcsname}}{}
10.805

```

```

10.806 \ifundefined{listtablename}
10.807     {\def\listoftables{\@restonecolfalse
10.808         \if@twocolumn\@restonecoltrue\onecolumn
10.809         \fi\chapter*{\listtablename\@mkboth
10.810             {\uppercase{\listtablename}}{\uppercase{\listtablename}}}%
10.811         \@starttoc{lot}%
10.812         \csname if@restonecol\endcsname\twocolumn
10.813         \csname fi\endcsname}}{}

```

Again, the bibliography and index environments; notice that in this case we use `\bibname` instead of `\refname` as in the definitions for the `article` style. The reason for this is that in the `article` document style the term ‘References’ is used in the definition of `\thebibliography`. In the `report` and `book` document styles the term ‘Bibliography’ is used.

```

10.814 \ifundefined{bibname}
10.815     {\def\thebibliography#1{\chapter*{\bibname
10.816         \@mkboth{\uppercase{\bibname}}{\uppercase{\bibname}}}%
10.817         \list{\arabic{enumi}}{\settowidth\labelwidth{[#1]}%
10.818         \leftmargin\labelwidth \advance\leftmargin\labelsep
10.819         \usecounter{enumi}}}%
10.820         \def\newblock{\hspace{11em plus.33em minus.07em}%
10.821         \sloppy\clubpenalty4000\widowpenalty\clubpenalty
10.822         \sfcode'\.=1000\relax}}{}
10.823
10.824 \ifundefined{indexname}
10.825     {\def\theindex{\@restonecoltrue\if@twocolumn\@restonecolfalse\fi
10.826         \columnseprule \z@
10.827         \columnsep 35pt\twocolumn[\@makeschapterhead{\indexname}]%
10.828         \@mkboth{\uppercase{\indexname}}{\uppercase{\indexname}}}%
10.829         \thispagestyle{plain}%
10.830         \parskip\z@ plus.3pt\parindent\z@ \let\item\@idxitem}}{}

```

Here is the abstract environment:

```

10.831 \ifundefined{abstractname}
10.832     {\def\abstract{\titlepage
10.833         \null\vfil
10.834         \begin{center}%
10.835         {\bf \abstractname}%
10.836         \end{center}}}{}}

```

And last but not least the `\chapter`, `\appendix` and `\part` macros.

```

10.837 \ifundefined{chaptername}{\def\@chapapp{\chaptername}}{}
10.838 %
10.839 \ifundefined{appendixname}
10.840     {\def\appendix{\par
10.841         \setcounter{chapter}{0}%
10.842         \setcounter{section}{0}%
10.843         \def\@chapapp{\appendixname}%
10.844         \def\thechapter{\Alph{chapter}}}}{}
10.845 %
10.846 \ifundefined{partname}
10.847     {\def\@part[#1]#2{\ifnum \c@secnumdepth >-2\relax
10.848         \refstepcounter{part}%
10.849         \addcontentsline{toc}{part}{\the part
10.850         \hspace{1em}#1}\else

```

```

10.851         \addcontentsline{toc}{part}{#1}\fi
10.852     \markboth{}{}%
10.853     {\centering
10.854         \ifnum \c@secnumdepth >-2\relax
10.855             \huge\bf \partname{} \thepart
10.856         \par
10.857         \vskip 20pt \fi
10.858         \Huge \bf
10.859         #1\par}\@endpart}}{}%
10.860 \or

```

Now we address the case where `babel.def` is read after the `letter` style. The `letter` document style defines the macro `\opening` and some other macros that are specific to `letter`. This means that we have to redefine other macros, compared to the previous two cases.

First two macros for the material at the end of a letter, the `\cc` and `\encl` macros.

```

10.861 \@ifundefined{ccname}%
10.862     {\def\cc#1{\par\noindent
10.863         \parbox[t]{\textwidth}%
10.864         {\@hangfrom{\rm \ccname : }\ignorespaces #1\strut}\par}}{}
10.865
10.866 \@ifundefined{enclname}%
10.867     {\def\encl#1{\par\noindent
10.868         \parbox[t]{\textwidth}%
10.869         {\@hangfrom{\rm \enclname : }\ignorespaces #1\strut}\par}}{}

```

The last thing we have to do here is to redefine the `headings` pagestyle:

```

10.870 \@ifundefined{headtoname}
10.871     {\def\ps@headings{%
10.872         \def\@oddhead{\sl \headtoname{} \ignorespaces\toname \hfil
10.873             \@date \hfil \pagename{} \thepage}%
10.874         \def\@oddfoot{}}{}

```

This was the last of the four standard document styles, so if `\doc@style` has another value we do nothing and just close the `if` construction.

```

10.875 \fi

```

Here ends the code that can be optionally included when a version of \LaTeX is in use that is dated *before* december 1991.

```

10.876 </names>
10.877 </core>

```

10.13 Cross referencing macros

The \LaTeX book states:

The *key* argument is any sequence of letters, digits, and punctuation symbols; upper- and lowercase letters are regarded as different.

When the above quote should still be true when a document is typeset in a language that has active characters, special care has to be taken of the category codes of these characters when they appear in an argument of the cross referencing macros.

When a cross referencing command processes its argument, all tokens in this argument should be character tokens with category ‘letter’ or ‘other’.

The only way to accomplish this in most cases is to use the trick described in the T_EXbook [1] (Appendix D, page 382). The primitive `\meaning` applied to a token expands to the current meaning of this token. For example, ‘`\meaning\A`’ with `\A` defined as ‘`\def\A#1{\B}`’ expands to the characters ‘`macro:#1->\B`’ with all category codes set to ‘other’ or ‘space’.

\bbl@redefine To redefine a command, we save the old meaning of the macro. Then we redefine it to call the original macro with the ‘sanitized’ argument. The reason why we do it this way is that we don’t want to redefine the L^AT_EX macros completely in case their definitions change (they have changed in the past).

Bacuse we need to redefine a number of commands we define the command `\bbl@redefine` which takes care of this. It creates a new control sequence, `\org@...`

```
10.878 < *core | shorthands >
10.879 \def\bbl@redefine#1{%
10.880   \edef\bbl@tempa{\expandafter\@gobble\string#1}%
10.881   \expandafter\let\csname org@\bbl@tempa\endcsname#1
10.882   \expandafter\def\csname\bbl@tempa\endcsname}
```

This command should only be used in the preamble of the document.

```
10.883 \@onlypreamble\bbl@redefine
```

\bbl@redefine@long This version of `\babel@redefine` can be used to redefine `\long` commands such as `\ifthenelse`.

```
10.884 \def\bbl@redefine@long#1{%
10.885   \edef\bbl@tempa{\expandafter\@gobble\string#1}%
10.886   \expandafter\let\csname org@\bbl@tempa\endcsname#1
10.887   \expandafter\long\expandafter\def\csname\bbl@tempa\endcsname}
10.888 \@onlypreamble\bbl@redefine@long
```

\bbl@redefineroobust For commands that are redefined, but which *might* be robust we need a slightly more intelligent macro. A robust command `foo` is defined to expand to `\protect\foo_`. So it is necessary to check whether `\foo_` exists.

```
10.889 \def\bbl@redefineroobust#1{%
10.890   \edef\bbl@tempa{\expandafter\@gobble\string#1}%
10.891   \expandafter\ifx\csname\bbl@tempa\space\endcsname\relax
10.892     \expandafter\let\csname org@\bbl@tempa\endcsname#1
10.893     \expandafter\edef\csname\bbl@tempa\endcsname{\noexpand\protect
10.894       \expandafter\noexpand\csname\bbl@tempa\space\endcsname}%
10.895   \else
10.896     \expandafter\let\csname org@\bbl@tempa\expandafter\endcsname
10.897       \csname\bbl@tempa\space\endcsname
10.898   \fi
```

The result of the code above is that the command that is being redefined is always robust afterwards. Therefore all we need to do now is define `\foo_`.

```
10.899   \expandafter\def\csname\bbl@tempa\space\endcsname}
```

This command should only be used in the preamble of the document.

```
10.900 \@onlypreamble\bbl@redefineroobust
```

`\newlabel` The macro `\label` writes a line with a `\newlabel` command into the `.aux` file to define labels.

```

10.901 \bbl@redefine\newlabel#1#2{%
10.902   \@safe@activetrue\org@newlabel{#1}{#2}\@safe@activfalse}

```

`\@testdef` An internal L^AT_EX macro used to test if the labels that have been written on the `.aux` file have changed. It is called by the `\enddocument` macro. This macro needs to be completely rewritten, using `\meaning`. The reason for this is that in some cases the expansion of `\#1#2` contains the same characters as the `#3`; but the character codes differ. Therefore L^AT_EX keeps reporting that the labels may have changed.

```

10.903 \def\@testdef #1#2#3{%
10.904   \expandafter\let\expandafter\bbl@tempa\csname #1#2\endcsname
10.905   \def\bbl@tempb{#3}%
10.906   \ifx\bbl@tempa\relax\else
10.907     \edef\bbl@tempa{\expandafter\strip@prefix\meaning\bbl@tempa}\fi
10.908     \edef\bbl@tempb{\expandafter\strip@prefix\meaning\bbl@tempb}%
10.909     \ifx \bbl@tempa \bbl@tempb
10.910       \else \@tempwattrue \fi}

```

`\ref` The same holds for the macro `\ref` that references a label and `\pageref` to reference a page. So we redefine `\ref` and `\pageref`. While we change these macros, we make them robust as well (if they weren't already) to prevent problems if they should become expanded at the wrong moment.

```

10.911 \bbl@redefineroobust\ref#1{%
10.912   \@safe@activetrue\org@ref{#1}\@safe@activfalse}
10.913 \bbl@redefineroobust\pageref#1{%
10.914   \@safe@activetrue\org@pageref{#1}\@safe@activfalse}

```

`\@citex` The macro used to cite from a bibliography, `\cite` uses an internal macro, `\@citex`. It is this internal macro that picks up the argument, so we redefine this internal macro and leave `\cite` alone.

```

10.915 \bbl@redefine\@citex[#1]#2{%
10.916   \@safe@activetrue\org@@citex[#1]{#2}\@safe@activfalse}

```

`\nocite` The macro `\nocite` which is used to instruct BiB_TE_X to extract uncited references from the database.

```

10.917 \bbl@redefine\nocite#1{%
10.918   \@safe@activetrue\org@nocite{#1}\@safe@activfalse}

```

`\bibcite` The macro that is used in the `.aux` file to define citation labels.

```

10.919 %\bbl@redefine\bibcite#1#2{%
10.920 %   \@safe@activetrue\org@bibcite{#1}{#2}\@safe@activfalse}

```

`\@bibitem` One of the two internal L^AT_EX macros called by `\bibitem` that write the citation label on the `.aux` file.

```

10.921 \bbl@redefine\@bibitem#1{%
10.922   \@safe@activetrue\org@@bibitem{#1}\@safe@activfalse}

```

`\@lbibitem` The other of the two internal L^AT_EX macros called by `\bibitem` that write the citation label on the `.aux` file.

```

10.923 \bbl@redefine\@lbibitem[#1]#2{%

```

```

10.924 \@safe@activetrue\org@@lbibitem[#1]{#2}\@safe@activesfalse}
10.925 </core | shorthands>

```

`\ifthenelse` Sometimes a document writer wants to create a special effect depending on the page a certain fragment of text appears on. This can be achieved by the following piece of code:

```

\ifthenelse{\isodd{pageref{some:label}}}{
  {code for odd pages}
}{code for even pages}

```

In order for this to work the argument of `\isodd` needs to be fully expandable. With the above redefinition of `\pageref` it is not in the case of this example. To overcome that we add some code to the definition of `\ifthenelse` to make things work.

The first thing we need to do is check if the package `ifthen` is loaded. This should be done at `\begin{document}` time.

```

10.926 (*package)
10.927 \AtBeginDocument{%
10.928 \ifpackageloaded{ifthen}{%

```

Then we can redefine `\ifthenelse`:

```

10.929 \bbl@redefine@long\ifthenelse#1#2#3{%

```

We want to revert the definition of `\pageref` to its original definition for the duration of `\ifthenelse`, so we first need to store its current meaning.

```

10.930 \let\bbl@tempa\pageref
10.931 \let\pageref\org@pageref

```

Then we can set the `\@safe@actives` switch and call the original `\ifthenelse`.

```

10.932 \@safe@activetrue\org@ifthenelse{#1}{#2}{#3}%
10.933 \@safe@activesfalse

```

Now we need to re-install the stored definition of `\pageref`.

```

10.934 \let\pageref\bbl@tempa
10.935 }%

```

When the package wasn't loaded we do nothing.

```

10.936 }{}%
10.937 }

```

`\@@vpageref` When the package `varioref` is in use we need to modify its internal command `\@@vpageref` in order to prevent problems when an active character ends up in the argument of `\vref`.

```

10.938 \AtBeginDocument{%
10.939 \ifpackageloaded{varioref}{%
10.940 \bbl@redefineroobust\@@vpageref#1[#2]#3{%
10.941 \@safe@activetrue
10.942 \org@@vpageref{#1}{#2}{#3}%
10.943 \@safe@activesfalse}%
10.944 }{}%
10.945 }

```

`\hhline` Dealing the activation of the shorthand charactes has introduced a problem with the `hhline` package. The reason is that it uses the ‘:’ character which is made active by the french support in `babel`. Therefore we need to *reload* the package when the ‘:’ is an active character.

So at `\begin{document}` we check whether `hhline` is loaded.

```
10.946 \AtBeginDocument{%
10.947   \ifpackageloaded{hhline}
```

Then we check whether the expansion of `\normal@char:` is not equal to `\relax`.

```
10.948   {\expandafter\ifx\cname normal@char:\endcsname\relax
10.949   \else
```

In that case we simply reload the package. Note that this happens *after* the category code of the @-sign has been changed to other, so we need to temporarily change it to letter again.

```
10.950     \makeatletter
10.951     \def\@currname{hhline}\input{hhline.sty}\makeatother
10.952     \fi}
10.953   {}}
10.954 \end{package}
```

`\nfss@catcodes` L^AT_EX’s font selection scheme sometimes wants to read font definition files in the middle of processing the document. In order to guard against any characters having the wrong `\catcode`’s it always calls `\nfss@catcodes` before loading a file. Unfortunately, the characters “ and ’ are not dealt with. Therefore we have to add them untill L^AT_EX does that herself.

```
10.955 \core | shorthands
10.956 \ifx\nfss@catcodes\undefined
10.957 \else
10.958   \addto\nfss@catcodes{%
10.959     \@makeother\’%
10.960     \@makeother\"%
10.961   }
10.962 \fi
10.963 \core | shorthands
```

11 Local Language Configuration

`\loadlocalcfg` At some sites it may be necessary to add site specific actions to a language definition file. This can be done by creating a file with the same name as the language definition file, but with the extension `.cfg`. For instance the file `norsk.cfg` will be loaded when the language definition file `norsk.ldf` is loaded.

11.1 `\core`

For plain based formats we don’t want to override the definition of `\loadlocalcfg` from `plain.def`.

```
11.2 \ifx\loadlocalcfg\undefined
11.3   \def\loadlocalcfg#1{%
11.4     \InputIfFileExists{#1.cfg}
11.5     {\typeout{*****^J%
11.6               * Local config file #1.cfg used^J%
```



```

11.7          *}%
11.8      }
11.9      {}
11.10 \fi

```

Just to be compatible with L^AT_EX 2.09 we add a few more lines of code:

```

11.11 \ifx\@unexpandable@protect\@undefined
11.12   \def\@unexpandable@protect{\noexpand\protect\noexpand}
11.13   \long\def \protected@write#1#2#3{%
11.14       \begingroup
11.15       \let\thepage\relax
11.16       #2%
11.17       \let\protect\@unexpandable@protect
11.18       \edef\reserved@a{\write#1{#3}}%
11.19       \reserved@a
11.20   \endgroup
11.21   \if@nobreak\ifvmode\nobreak\fi\fi
11.22 }
11.23 \fi
11.24 </core>

```

12 Driver files for the documented source code

Since **babel** version 3.4 all source files that are part of the **babel** system can be typeset separately. But in order to typeset them all in one document the file **babel.drv** can be used. If you only want the information on how to use the **babel** system and what goodies are provided by the language specific files you can run the file **user.drv** through \LaTeX to get a user guide.

```

12.1 (*driver)
12.2 \documentclass{ltxdoc}
12.3 \DoNotIndex{!,\',\,,\.,\-,\:,\;,\?,\/,\\,\^,\',\@M}
12.4 \DoNotIndex{@,\@one,\@m,\@afterheading,\@date,\@endpart}
12.5 \DoNotIndex{@changefrom,\@idxitem,\@makeschapterhead,\@mkboth}
12.6 \DoNotIndex{@oddfont,\@oddfont,\@restonecolfalse,\@restonecoltrue}
12.7 \DoNotIndex{@starttoc,\@unused}
12.8 \DoNotIndex{\accent,\active}
12.9 \DoNotIndex{\addcontentsline,\advance,\Alph,\arabic}
12.10 \DoNotIndex{\baselineskip,\begin,\beginningof,\bf,\box,\c@secnumdepth}
12.11 \DoNotIndex{\catcode,\centering,\char,\chardef,\clubpenalty}
12.12 \DoNotIndex{\columnsep,\columnseprule,\crrc,\csname}
12.13 \DoNotIndex{\day,\def,\dimen,\discretionary,\divide,\dp,\do}
12.14 \DoNotIndex{\edef,\else,\@empty,\end,\endgroup,\endcsname,\endinput}
12.15 \DoNotIndex{\errhelp,\errmessage,\expandafter,\fi,\filedate}
12.16 \DoNotIndex{\fileversion,\fmtname,\fnum@figure,\fnum@table,\fontdimen}
12.17 \DoNotIndex{\gdef,\global}
12.18 \DoNotIndex{\hbox,\hidewidth,\hfil,\hskip,\hspace,\ht,\Huge,\huge}
12.19 \DoNotIndex{\ialign,\if@twocolumn,\ifcase,\ifcat,\ifhmode,\ifmmode}
12.20 \DoNotIndex{\ifnum,\ifx,\immediate,\ignorespaces,\input,\item}
12.21 \DoNotIndex{\kern}
12.22 \DoNotIndex{\labelsep,\Large,\large,\labelwidth,\lccode,\leftmargin}
12.23 \DoNotIndex{\lineskip,\leavevmode,\let,\list,\ll,\long,\lower}
12.24 \DoNotIndex{\m@ne,\mathchar,\mathaccent,\markboth,\month,\multiply}
12.25 \DoNotIndex{\newblock,\newbox,\newcount,\newdimen,\newif,\newwrite}
12.26 \DoNotIndex{\nobreak,\noexpand,\noindent,\null,\number}
12.27 \DoNotIndex{\onecolumn,\or}
12.28 \DoNotIndex{\p@,\par,\parbox,\parindent,\parskip,\penalty}
12.29 \DoNotIndex{\protect,\ps@headings}
12.30 \DoNotIndex{\quotation}
12.31 \DoNotIndex{\raggedright,\raise,\refstepcounter,\relax,\rm,\setbox}
12.32 \DoNotIndex{\section,\setcounter,\settowidth,\scriptscriptstyle}
12.33 \DoNotIndex{\sfcode,\sl,\sloppy,\small,\space,\spacefactor,\strut}
12.34 \DoNotIndex{\string}
12.35 \DoNotIndex{\textwidth,\the,\thechapter,\thefigure,\thepage,\thepart}
12.36 \DoNotIndex{\thetable,\thispagestyle,\titlepage,\tracingmacros}
12.37 \DoNotIndex{\tw@,\twocolumn,\typeout,\uppercase,\usecounter}
12.38 \DoNotIndex{\vbox,\vfil,\vskip,\vspace,\vss}
12.39 \DoNotIndex{\widowpenalty,\write,\xdef,\year,\z@,\z@skip}

```

Here `\dlqq` is defined so that an example of " can be given.

```

12.40 \makeatletter
12.41 \gdef\dlqqf{\setbox\tw@=\hbox{,}\setbox\zc@=\hbox{' '}%
12.42 \dimen\zc@=\ht\zc@ \advance\dimen\zc@-\ht\tw@
12.43 \setbox\zc@=\hbox{\lower\dimen\zc@\box\zc@}\ht\zc@=\ht\tw@
12.44 \dp\zc@=\dp\tw@ \box\zc@\kern-.04em}}

```

The code lines are numbered within sections,

```
12.45 <!*user>
12.46 \@addtoreset{CodelineNo}{section}
12.47 \renewcommand\theCodelineNo{%
12.48   \reset@font\scriptsize\thesection.\arabic{CodelineNo}}
```

which should also be visible in the index; hence this redefinition of a macro from `doc.sty`.

```
12.49 \renewcommand\codeline@wrindex[1]{\if@filesw
12.50   \immediate\write\@indexfile
12.51   {\string\indexentry{#1}%
12.52   {\number\c@section.\number\c@CodelineNo}}\fi}
```

The glossary environment is used or the change log, but its definition needs changing for this document.

```
12.53 \renewenvironment{theglossary}{%
12.54   \glossary@prologue%
12.55   \GlossaryParms \let\item\@idxitem \ignorespaces}%
12.56 {}
12.57 </!user>
12.58 \makeatother
```

A few shorthands used in the documentation

```
12.59 \font\manual=logo10 % font used for the METAFONT logo, etc.
12.60 \newcommand*\MF{{\manual META}\-{\manual FONT}}
12.61 \newcommand*\TeXhax{\TeX hax}
12.62 \newcommand*\babel{\textsf{babel}}
12.63 \newcommand*\Babel{\textsf{Babel}}
12.64 \newcommand*\m[1]{\mbox{$\langle\it#1\rangle$}}
12.65 \newcommand*\langvar{\m{lang}}
```

Some more definitions needed in the documentation.

```
12.66 %\newcommand*\note[1]{\textbf{#1}}
12.67 \newcommand*\note[1]{}
12.68 \newcommand*\bsl{\protect\bslash}
12.69 \newcommand*\Lopt[1]{\textsf{#1}}
12.70 \newcommand*\file[1]{\texttt{#1}}
12.71 \newcommand*\cls[1]{\texttt{#1}}
12.72 \newcommand*\pkg[1]{\texttt{#1}}
12.73 \newcommand*\langdeffile[1]{%
12.74 <-user> \clearpage
12.75   \DocInput{#1}}
```

When a full index should be generated uncomment the line with `\EnableCrossres`.

Beware, processing may take some time. Use `\DisableCrossrefs` when the index is ready.

```
12.76 % \EnableCrossrefs
12.77 \DisableCrossrefs
```

Include the change log.

```
12.78 <-user> \RecordChanges
```

The index should use the linenumbers of the code.

```
12.79 <-user> \CodelineIndex
```

Set everything in `\MacroFont` instead of `\AltMacroFont`

```
12.80 \setcounter{StandardModuleDepth}{1}
```

For the user guide we only want the description parts of all the files.

12.81 `<+user>\OnlyDescription`

Here starts the document

12.82 `\begin{document}`

12.83 `\DocInput{babel.dtx}`

All the language definition files.

12.84 `<+user>\clearpage`

12.85 `\langdeffile{esperant.dtx}`

12.86 `\langdeffile{dutch.dtx}`

12.87 `\langdeffile{english.dtx}`

12.88 `\langdeffile{germanb.dtx}`

12.89 `%`

12.90 `\langdeffile{breton.dtx}`

12.91 `\langdeffile{welsh.dtx}`

12.92 `\langdeffile{irish.dtx}`

12.93 `\langdeffile{scottish.dtx}`

12.94 `%`

12.95 `\langdeffile{greek.dtx}`

12.96 `%`

12.97 `\langdeffile{frenchb.dtx}`

12.98 `\langdeffile{italian.dtx}`

12.99 `\langdeffile{portuges.dtx}`

12.100 `\langdeffile{spanish.dtx}`

12.101 `\langdeffile{catalan.dtx}`

12.102 `\langdeffile{galician.dtx}`

12.103 `\langdeffile{romanian.dtx}`

12.104 `%`

12.105 `\langdeffile{danish.dtx}`

12.106 `\langdeffile{norsk.dtx}`

12.107 `\langdeffile{swedish.dtx}`

12.108 `%`

12.109 `\langdeffile{finnish.dtx}`

12.110 `\langdeffile{magyar.dtx}`

12.111 `\langdeffile{estonian.dtx}`

12.112 `%`

12.113 `\langdeffile{croatian.dtx}`

12.114 `\langdeffile{czech.dtx}`

12.115 `\langdeffile{polish.dtx}`

12.116 `\langdeffile{slovak.dtx}`

12.117 `\langdeffile{slovene.dtx}`

12.118 `\langdeffile{russianb.dtx}`

12.119 `%`

12.120 `\langdeffile{lsorbian.dtx}`

12.121 `\langdeffile{usorbian.dtx}`

12.122 `\langdeffile{turkish.dtx}`

12.123 `%`

12.124 `\langdeffile{bahasa.dtx}`

12.125 `\clearpage`

12.126 `\DocInput{bbplain.dtx}`

Finally print the index and change log (not for the user guide).

12.127 `<!*user>`

12.128 `\clearpage`

```
12.129 \def\filename{index}  
12.130 \PrintIndex  
12.131 \clearpage  
12.132 \def\filename{changes}  
12.133 \PrintChanges  
12.134 \!/user  
12.135 \end{document}  
12.136 \!/driver
```

13 Conclusion

A system of document options has been presented that enable the user of \LaTeX to adapt the standard document classes of \LaTeX to the language he or she prefers to use. These options offer the possibility to switch between languages in one document. The basic interface consists of using one option, which is the same for *all* standard document classes.

In some cases the language definition files provide macros that can be of use to plain \TeX users as well as to \LaTeX users. The `babel` system has been implemented in such a way that it can be used by both groups of users.

14 Acknowledgements

I would like to thank all who volunteered as β -testers for their time. I would like to mention Julio Sanchez who supplied the option file for the Spanish language and Maurizio Codogno who supplied the option file for the Italian language. Michel Goossens supplied contributions for most of the other languages. Nico Poppelier helped polishing the text of the documentation and supplied parts of the macros for the Dutch language. Paul Wackers and Werenfried Spit helped finding and repairing bugs.

During the further development of the `babel` system I received much help from Bernd Raichle, for which I am grateful.

References

- [1] Donald E. Knuth, *The \TeX book*, Addison-Wesley, 1986.
- [2] Leslie Lamport, *\LaTeX , A document preparation System*, Addison-Wesley, 1986.
- [3] K.F. Treebus. *Tekstwijzer, een gids voor het grafisch verwerken van tekst*. SDU Uitgeverij ('s-Gravenhage, 1988). A Dutch book on layout design and typography.
- [4] Hubert Partl, *German \TeX* , *TUGboat* 9 (1988) #1, p. 70–72.
- [5] Leslie Lamport, in: \TeX hax Digest, Volume 89, #13, 17 februari 1989.
- [6] Johannes Braams, Victor Eijkhout and Nico Poppelier, *The development of national \LaTeX styles*, *TUGboat* 10 (1989) #3, p. 401–406.
- [7] Joachim Schrod, *International \LaTeX is ready to use*, *TUGboat* 11 (1990) #1, p. 87–90.

15 The Esperanto language

The file `esperant.dtx`¹⁰ defines all the language-specific macros for the Esperanto language.

For this language the character `^` is made active. In table 2 an overview is given of its purpose.

<code>^c</code>	gives <code>ĉ</code> with hyphenation in the rest of the word allowed, this works for <code>c</code> , <code>C</code> , <code>g</code> , <code>G</code> , <code>H</code> , <code>J</code> , <code>s</code> , <code>S</code> , <code>z</code> , <code>Z</code>
<code>^h</code>	prevents <code>ĥ</code> from becoming too tall
<code>^j</code>	gives <code>ĵ</code>
<code>^u</code>	gives <code>ŭ</code> , with hyphenation in the rest of the word allowed
<code>^U</code>	gives <code>Ŭ</code> , with hyphenation in the rest of the word allowed
<code>^ </code>	inserts a <code>\discretionary{-}{-}{-}</code>

Table 2: The functions of the active character for Esperanto.

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the `@` sign, etc.

```
15.1 (*code)
15.2 \LdfInit{esperanto}\captionesperanto
```

When this file is read as an option, i.e. by the `\usepackage` command, `esperanto` will be an ‘unknown’ language in which case we have to make it known. So we check for the existence of `\l@esperanto` to see whether we have to do something here.

```
15.3 \ifx\l@esperanto\undefined
15.4 \nopatterns{Esperanto}
15.5 \adddialect\l@esperanto0\fi
```

The next step consists of defining commands to switch to the Esperanto language. The reason for this is that a user might want to switch back and forth between languages.

`\captionesperanto` The macro `\captionesperanto` defines all strings used in the four standard documentclasses provided with L^AT_EX.

```
15.6 \addto\captionesperanto{%
15.7 \def\prefacename{Anta\u{u}parolo}%
15.8 \def\refname{Cita\^{}oj}%
15.9 \def\abstractname{Resumo}%
15.10 \def\bibname{Bibliografio}%
15.11 \def\chaptername{{\^{}C}apitro}%
15.12 \def\appendixname{Apendico}%
15.13 \def\contentsname{Enhavo}%
15.14 \def\listfigurename{Listo de figuroj}%
15.15 \def\listtablename{Listo de tabeloj}%
15.16 \def\indexname{Indekso}%
```

¹⁰The file described in this section has version number v1.4j and was last revised on 1997/01/06. A contribution was made by Ruiz-Altava Marti (ruizaltb@cernvm.cern.ch). Code from the file `esperant.sty` by Jörg Knappen (knappen@vkpmzd.kph.uni-mainz.de) was included.

```

15.17 \def\figurename{Figuro}%
15.18 \def\tablename{Tabelo}%
15.19 \def\partname{Parto}%
15.20 \def\enclname{Aldono(j)}%
15.21 \def\ccname{Kopie al}%
15.22 \def\headtoname{Al}%
15.23 \def\pagename{Pa\^go}%
15.24 \def\subjectname{Temo}%
15.25 \def\seename{vidu}% a~u: vd.
15.26 \def\alsoname{vidu anka\u{u}}% a~u vd. anka\u{u}
15.27 \def\proofname{Pruvo}%
15.28 }

```

`\dateesperanto` The macro `\dateesperanto` redefines the command `\today` to produce Esperanto dates.

```

15.29 \def\dateesperanto{%
15.30 \def\today{\number\day{--a}~de~\ifcase\month\or
15.31 januaro\or februaro\or marto\or aprilo\or majo\or junio\or
15.32 julio\or a\u{u}gusto\or septembro\or oktobro\or novembro\or
15.33 decembro\fi,\space \number\year}}

```

`\extrasesperanto` The macro `\extrasesperanto` performs all the extra definitions needed for the Esperanto language. The macro `\noextrasesperanto` is used to cancel the actions of `\extrasesperanto`.

For Esperanto the `^` character is made active. This is done once, later on its definition may vary.

```

15.34 \initiate@active@char{^}
15.35 \addto\extrasesperanto{\languageshorthands{esperanto}}
15.36 \addto\extrasesperanto{\bbl@activate{^}}
15.37 \addto\noextrasesperanto{\bbl@deactivate{^}}

```

And here are the uses of the active `^`:

```

15.38 \declare@shorthand{esperanto}{^c}{\^c}\allowhyphens}
15.39 \declare@shorthand{esperanto}{^C}{\^C}\allowhyphens}
15.40 \declare@shorthand{esperanto}{^g}{\^g}\allowhyphens}
15.41 \declare@shorthand{esperanto}{^G}{\^G}\allowhyphens}
15.42 \declare@shorthand{esperanto}{^h}{h\llap{\^}}\allowhyphens}
15.43 \declare@shorthand{esperanto}{^H}{\^H}\allowhyphens}
15.44 \declare@shorthand{esperanto}{^j}{\^j}\allowhyphens}
15.45 \declare@shorthand{esperanto}{^J}{\^J}\allowhyphens}
15.46 \declare@shorthand{esperanto}{^s}{\^s}\allowhyphens}
15.47 \declare@shorthand{esperanto}{^S}{\^S}\allowhyphens}
15.48 \declare@shorthand{esperanto}{^u}{\u u\allowhyphens}
15.49 \declare@shorthand{esperanto}{^U}{\u U\allowhyphens}
15.50 \declare@shorthand{esperanto}{^|}{\discretionary{-}{|}{}}\allowhyphens}

```

`\Esper` In `esperant.sty` Jörg Knappen provides the macros `\esper` and `\Esper` that can be used instead of `\alph` and `\Alph`. These macros are available in this file as well.

Their definition takes place in three steps. First the toplevel.

```

15.51 \def\esper#1{\@esper{\@nameuse{c#1}}}
15.52 \def\Esper#1{\@Esper{\@nameuse{c#1}}}

```


Then the first five occasions that are probably used the most.

```
15.53 \def\@esper#1{\ifcase#1\or a\or b\or c\or \^c\or d\else\@iesper{#1}\fi}
15.54 \def\@Esper#1{\ifcase#1\or A\or B\or C\or \^C\or D\else\@Iesper{#1}\fi}
```

And the 33 other cases.

```
15.55 \def\@iesper#1{\ifcase#1\or \or \or \or \or \or e\or f\or g\or \^g\or
15.56     h\or h\llap{\^{\}}\or i\or j\or \^j\or k\or l\or m\or n\or o\or
15.57     p\or s\or \^s\or t\or u\or \u{u}\or v\or z\else\@ctrerr\fi}

15.58 \def\@Iesper#1{\ifcase#1\or \or \or \or \or \or E\or F\or G\or \^G\or
15.59     H\or \^H\or I\or J\or \^J\or K\or L\or M\or N\or O\or
15.60     P\or S\or \^S\or T\or U\or \u{U}\or V\or Z\else\@ctrerr\fi}
```

`\hodiaun` In `esperant.sty` Jörg Knappen provides two alternative macros for `\today`, `\hodiaun` and `\hodiaun`. The second macro produces an accusative version of the date in Esperanto.

```
15.61 \addto\dateesperanto{\def\hodiaun{la \today}}
15.62 \def\hodiaun{la \number\day --an~de~\ifcase\month\or
15.63     januaro\or februaro\or marto\or aprilo\or majo\or junio\or
15.64     julio\or a\u{u}gusto\or septembro\or oktobro\or novembro\or
15.65     decembro\fi, \space \number\year}
```

The macro `\ldf@finish` takes care of looking for a configuration file, setting the main language to be switched on at `\begin{document}` and resetting the category code of `@` to its original value.

```
15.66 \ldf@finish{esperanto}
15.67 \code
```

16 The Dutch language

The file `dutch.dtx`¹¹ defines all the language-specific macros for the Dutch language and the ‘Afrikaans’ version¹² of it.

For this language the character " is made active. In table 3 an overview is given of its purpose. One of the reasons for this is that in the Dutch language a word with a dieresis can be hyphenated just before the letter with the umlaut, but the dieresis has to disappear if the word is broken between the previous letter and the accented letter.

In [3] the quoting conventions for the Dutch language are discussed. The preferred convention is the single-quote Anglo-American convention, i.e. ‘This is a quote’. An alternative is the slightly old-fashioned Dutch method with initial double quotes lowered to the baseline, „This is a quote”, which should be typed as “‘This is a quote’”.

"a	\ "a which hyphenates as -a; also implemented for the other letters.
"y	puts a negative kern between i and j
"Y	puts a negative kern between I and J
"	disable ligature at this position.
"-	an explicit hyphen sign, allowing hyphenation in the rest of the word.
"~	to produce a hyphencharacter without the following <code>\discretionary{}{}{}</code> .
" "	to produce an invisible ‘breakpoint’.
"‘	lowered double left quotes (see example below).
"’	normal double right quotes.
\-	like the old \-, but allowing hyphenation in the rest of the word.

Table 3: The extra definitions made by `dutch.ldf`

```

16.1 % \changes{dutch-3.8a}{1996/10/04}{made check dependant on
16.2 %   \cs{CurrentOption}}
16.3 %
16.4 %   The macro |\LdfInit| takes care of preventing that this file is
16.5 %   loaded more than once, checking the category code of the
16.6 %   \texttt{@} sign, etc.
16.7 % \changes{dutch-3.8a}{1996/10/30}{Now use \cs{LdfInit} to perform
16.8 %   initial checks}
16.9 %   \begin{macrocode}
16.10 < *code>
16.11 \LdfInit\CurrentOption{captions\CurrentOption}

```

When this file is read as an option, i.e. by the `\usepackage` command, `dutch` could be an ‘unknown’ language in which case we have to make it known. So we check for the existence of `\l@dutch` or `\l@afrikaans` to see whether we have to do something here.

¹¹The file described in this section has version number v3.8c, and was last revised on 1997/01/14.

¹²contributed by Stoffel Lombard (lombc@b31pc87.up.ac.za)

First we try to establish with which option we are being processed.

```

16.12 \def\bbl@tempa{dutch}
16.13 \ifx\CurrentOption\bbl@tempa

    If it is dutch then we first check if the Dutch hyphenation patterns wer loaded,
16.14 \ifx\l@dutch\undefined
    if no we issue a warning and make dutch a ‘dialect’ of either the hyphenation
    patterns that were loaded in slot 0 or of ‘afrikaans’ when it is available.

16.15 \nopatterns{Dutch}
16.16 \ifx\l@afrikaans\undefined
16.17 \addialect\l@dutch0
16.18 \else
16.19 \addialect\l@dutch\l@afrikaans
16.20 \fi
16.21 \fi

```

The next step consists of defining commands to switch to (and from) the Dutch language.

`\captionsdutch` The macro `\captionsdutch` defines all strings used in the four standard document classes provided with L^AT_EX.

```

16.22 \begingroup
16.23 \catcode'\\"active
16.24 \def\x{\endgroup
16.25 \def\captionsdutch{%
16.26 \def\prefacename{Voorwoord}%
16.27 \def\refname{Referenties}%
16.28 \def\abstractname{Samenvatting}%
16.29 \def\bibname{Bibliografie}%
16.30 \def\chaptername{Hoofdstuk}%
16.31 \def\appendixname{B"ylage}%
16.32 \def\contentsname{Inhoudsopgave}%
16.33 \def\listfigurename{L"yst van figuren}%
16.34 \def\listtablename{L"yst van tabellen}%
16.35 \def\indexname{Index}%
16.36 \def\figurename{Figuur}%
16.37 \def\tablename{Tabel}%
16.38 \def\partname{Deel}%
16.39 \def\enclname{B"ylage(n)}%
16.40 \def\ccname{cc}%
16.41 \def\headtoname{Aan}%
16.42 \def\pagename{Pagina}%
16.43 \def\seename{zie}%
16.44 \def\alsoname{zie ook}%
16.45 \def\proofname{Bew"ys}%
16.46 }
16.47 }\x

```

`\datedutch` The macro `\datedutch` redefines the command `\today` to produce Dutch dates.

```

16.48 \def\datedutch{%
16.49 \def\today{\number\day~\ifcase\month\or
16.50 januari\or februari\or maart\or april\or mei\or juni\or
16.51 juli\or augustus\or september\or oktober\or november\or

```

```

16.52     december\fi
16.53     \space \number\year}}

```

When the option with which this file is being process was not `dutch` we assume it was `afrikaans`. We perform a similar check on the availability of the hyphenation patterns.

```

16.54 \else
16.55   \ifx\l@afrikaans\undefined
16.56     \@nopatterns{Afrikaans}
16.57   \ifx\l@dutch\undefined
16.58     \addialect\l@afrikaans0
16.59   \else
16.60     \addialect\l@afrikaans\l@dutch
16.61   \fi
16.62 \fi

```

`\caption safrikaans` Now is the time to define the words for ‘Afrikaans’.

```

16.63   \def\caption safrikaans{%
16.64     \def\prefacename{Voorwoord}%
16.65     \def\refname{Verwysings}%
16.66     \def\abstractname{Samevatting}%
16.67     \def\bibname{Bibliografie}%
16.68     \def\chaptername{Hoofstuk}%
16.69     \def\appendixname{Bylae}%
16.70     \def\contentsname{Inhoudsopgawe}%
16.71     \def\listfigurename{Lys van figure}%
16.72     \def\listtablename{Lys van tabelle}%
16.73     \def\indexname{Inhoud}%
16.74     \def\figurename{Figuur}%
16.75     \def\tablename{Tabel}%
16.76     \def\partname{Deel}%
16.77     \def\enclname{Bylae(n)}%
16.78     \def\ccname{a. a.}%
16.79     \def\headtoname{Aan}%
16.80     \def\pagename{Bladsy}%
16.81     \def\seename{sien}%
16.82     \def\alsoname{sien ook}%
16.83     \def\proofname{Bewys}%
16.84   }

```

`\date afrikaans` Here is the ‘Afrikaans’ version of the date macro.

```

16.85   \def\date afrikaans{%
16.86     \def\today{\number\day~\ifcase\month\or
16.87       Januarie\or Februarie\or Maart\or April\or Mei\or Junie\or
16.88       Julie\or Augustus\or September\or Oktober\or November\or
16.89       Desember\fi
16.90     \space \number\year}}
16.91 \fi

```

`\extrasdutch` The macros `\extrasdutch` and `\caption safrikaans` will perform all the extra definitions needed for the Dutch language. The macros `\noextrasdutch` and `noextrasafrikaans` is used to cancel the actions of `\extrasdutch` and `\caption safrikaans`.

For Dutch the " character is made active. This is done once, later on its definition may vary. Other languages in the same document may also use the " character for shorthands; we specify that the dutch group of shorthands should be used.

16.92 \initiate@active@char{"}

Both version of the language use the same set of shorthand definitions although the 'ij' is not used in Afrikaans.

16.93 \@namedef{extras\CurrentOption}{\languageshorthands{dutch}}

16.94 \expandafter\addto\csname extras\CurrentOption\endcsname{%

16.95 \bbl@activate{"}}

The 'umlaut' character should be positioned lower on *all* vowels in Dutch texts.

16.96 \expandafter\addto\csname extras\CurrentOption\endcsname{%

16.97 \umlautlow\umlautelow}

16.98 \@namedef{noextras\CurrentOption}{%

16.99 \umlauthigh}

\dutchhyphenmins The dutch hyphenation patterns can be used with \lefthyphenmin set to 2 and
\afrikaanshyphenmins \righthyphenmin set to 3.

16.100 \def\dutchhyphenmins{\tw@\thr@@}

16.101 \def\afrikaanshyphenmins{\tw@\thr@@}

\@trema In the Dutch language vowels with a trema are treated specially. If a hyphenation occurs before a vowel-plus-trema, the trema should disappear. To be able to do this we could first define the hyphenation break behaviour for the five vowels, both lowercase and uppercase, in terms of \discretionary. But this results in a large \if-construct in the definition of the active ". Because we think a user should not use " when he really means something like '' we chose not to distinguish between vowels and consonants. Therefore we have one macro \@trema which specifies the hyphenation break behaviour for all letters.

16.102 \def\@trema#1{\allowhyphens\discretionary{-}{#1}{\{"#1}\allowhyphens}

Now we can define the doublequote macros: the tremas,

16.103 \declare@shorthand{dutch}{a}{\textormath{\@trema a}{\ddot a}}

16.104 \declare@shorthand{dutch}{e}{\textormath{\@trema e}{\ddot e}}

16.105 \declare@shorthand{dutch}{i}{\textormath

16.106 {\allowhyphens\discretionary{-}{i}{\{"i}\allowhyphens}%

16.107 {\ddot \imath}}

16.108 \declare@shorthand{dutch}{o}{\textormath{\@trema o}{\ddot o}}

16.109 \declare@shorthand{dutch}{u}{\textormath{\@trema u}{\ddot u}}

dutch quotes,

16.110 \declare@shorthand{dutch}{"'}{%

16.111 \textormath{\quotedblbase{}}{\mbox{\quotedblbase}}}

16.112 \declare@shorthand{dutch}{"'}{%

16.113 \textormath{\textquotedblright{}}{\mbox{\textquotedblright}}}

and some additional commands:

16.114 \declare@shorthand{dutch}{"-}{\allowhyphens-\allowhyphens}

16.115 \declare@shorthand{german}{"~}{\textormath{\leavevmode\hbox{-}}{-}}

16.116 \declare@shorthand{dutch}{"|}{%

16.117 \textormath{\discretionary{-}{|}{\kern.03em}}{}}

```

16.118 \declare@shorthand{dutch}{""}{\hskip\z@skip}
16.119 \declare@shorthand{dutch}{\textit{y}}{\textit{ij}}{\ddot y}
16.120 \declare@shorthand{dutch}{\textit{Y}}{\textit{IJ}}{\ddot Y}

```

- \- All that is left now is the redefinition of \-. The new version of \- should indicate an extra hyphenation position, while allowing other hyphenation positions to be generated automatically. The standard behaviour of T_EX in this respect is very unfortunate for languages such as Dutch and German, where long compound words are quite normal and all one needs is a means to indicate an extra hyphenation position on top of the ones that T_EX can generate from the hyphenation patterns.

```

16.121 \expandafter\addto\csname extras\CurrentOption\endcsname{%
16.122   \babel@save\-\}
16.123 \expandafter\addto\csname extras\CurrentOption\endcsname{%
16.124   \def\-\{\allowhyphens\discretionary{-}{-}{\allowhyphens}\}

```

The macro \ldf@finish takes care of looking for a configuration file, setting the main language to be switched on at \begin{document} and resetting the category code of @ to its original value.

```

16.125 \ldf@finish\CurrentOption
16.126 \code

```

17 The English language

The file `english.dtx`¹³ defines all the language definition macros for the English language as well as for the American version of this language.

For this language currently no special definitions are needed or available.

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the `@` sign, etc.

```
17.1 (*code)
17.2 \LdfInit\CurrentOption{date\CurrentOption}
```

When this file is read as an option, i.e. by the `\usepackage` command, `english` could be an ‘unknown’ language in which case we have to make it known. So we check for the existence of `\l@english` to see whether we have to do something here.

We allow for the british english patterns to be loaded as either ‘english’, ‘british’, or ‘UKenglish’

```
17.3 \ifx\l@english\@undefined
17.4   \ifx\l@UKenglish\@undefined
17.5     \ifx\l@british\@undefined
17.6       \nopatterns{English}
17.7       \addialect\l@english0
17.8     \else
17.9       \let\l@english\l@british
17.10    \fi
17.11  \else
17.12    \let\l@english\l@UKenglish
17.13  \fi
17.14 \fi
```

Because we allow ‘british’ to be used as the babel option we need to make sure that it will be recognised by `\selectlanguage`. In the code above we have made sure that `\l@english` has a sensible value; now we make `\l@british` equal to that.

```
17.15 \ifx\l@british\@undefined
17.16   \let\l@british\l@english
17.17 \fi
```

‘American’ is a version of ‘English’ which can have its own hyphenation patterns. The default english patterns are in fact for american english. We allow for the patterns to be loaded as ‘english’ ‘american’ or ‘USenglish’.

```
17.18 \ifx\l@american\@undefined
17.19   \ifx\l@USenglish\@undefined
```

When the patterns are not known as ‘american’ or ‘USenglish’ we add a “dialect”.

```
17.20     \addialect\l@american\l@english
17.21   \else
17.22     \let\l@american\l@USenglish
17.23   \fi
17.24 \fi
```

The next step consists of defining commands to switch to (and from) the English language.

¹³The file described in this section has version number v3.3h and was last revised on 1996/12/23.

`\captionseenglish` The macro `\captionseenglish` defines all strings used in the four standard document classes provided with L^AT_EX.

```

17.25 \@namedef{captions\CurrentOption}{%
17.26   \def\prefacename{Preface}%
17.27   \def\refname{References}%
17.28   \def\abstractname{Abstract}%
17.29   \def\bibname{Bibliography}%
17.30   \def\chaptername{Chapter}%
17.31   \def\appendixname{Appendix}%
17.32   \def\contentsname{Contents}%
17.33   \def\listfigurename{List of Figures}%
17.34   \def\listtablename{List of Tables}%
17.35   \def\indexname{Index}%
17.36   \def\figurename{Figure}%
17.37   \def\tablename{Table}%
17.38   \def\partname{Part}%
17.39   \def\enclname{encl}%
17.40   \def\ccname{cc}%
17.41   \def\headtoname{To}%
17.42   \def\pagename{Page}%
17.43   \def\seename{see}%
17.44   \def\alsoname{see also}%
17.45   \def\proofname{Proof}%
17.46 }

```

`\dateenglish` The macro `\dateenglish` redefines the command `\today` to produce English dates.

```

17.47 \@namedef{date\CurrentOption}{%
17.48 \def\today{\ifcase\day\or
17.49   1st\or 2nd\or 3rd\or 4th\or 5th\or
17.50   6th\or 7th\or 8th\or 9th\or 10th\or
17.51   11th\or 12th\or 13th\or 14th\or 15th\or
17.52   16th\or 17th\or 18th\or 19th\or 20th\or
17.53   21st\or 22nd\or 23rd\or 24th\or 25th\or
17.54   26th\or 27th\or 28th\or 29th\or 30th\or
17.55   31st\fi~\ifcase\month\or
17.56   January\or February\or March\or April\or May\or June\or
17.57   July\or August\or September\or October\or November\or December\fi
17.58   \space \number\year}}

```

`\dateamerican` The macro `\dateamerican` redefines the command `\today` to produce American dates.

```

17.59 \def\dateamerican{%
17.60 \def\today{\ifcase\month\or
17.61   January\or February\or March\or April\or May\or June\or
17.62   July\or August\or September\or October\or November\or December\fi
17.63   \space\number\day, \number\year}}

```

`\extrasenglish` The macro `\extrasenglish` will perform all the extra definitions needed for the English language. The macro `\extrasenglish` is used to cancel the actions of `\extrasenglish`. For the moment these macros are empty but they are defined for compatibility with the other language definition files.

```

17.64 \@namedef{extras\CurrentOption}{}

```


17.65 `\@namedef{noextras\CurrentOption}{}{}`

The macro `\ldf@finish` takes care of looking for a configuration file, setting the main language to be switched on at `\begin{document}` and resetting the category code of `@` to its original value.

17.66 `\ldf@finish\CurrentOption`

17.67 `\</code>`

18 The German language

The file `germanb.dtx`¹⁴ defines all the language definition macros for the German language as well as for the Austrian dialect of this language¹⁵.

For this language the character " is made active. In table 4 an overview is given of its purpose. One of the reasons for this is that in the German language some character combinations change when a word is broken between the combination. Also the vertical placement of the umlaut can be controlled this way. The quotes

"a	\a, also implemented for the other lowercase and uppercase vowels.
"s	to produce the German ß (like \ss{ }).
"z	to produce the German ß (like \ss{ }).
"ck	for ck to be hyphenated as k-k.
"ff	for ff to be hyphenated as ff-f, this is also implemented for l, m, n, p, r and t
"S	for SS to be \uppercase{"s}.
"Z	for SZ to be \uppercase{"z}.
"	disable ligature at this position.
"-	an explicit hyphen sign, allowing hyphenation in the rest of the word.
"	like "-", but producing no hyphen sign (for compound words with hyphen, e.g. x-"y).
"~	for a compound word mark without a breakpoint.
"=	for a compound word mark with a breakpoint, allowing hyphenation in the composing words.
"‘	for German left double quotes (looks like „).
"’	for German right double quotes.
"<	for French left double quotes (similar to <<).
">	for French right double quotes (similar to >>).

Table 4: The extra definitions made by `german.ldf`

in table 4 can also be typeset by using the commands in table 5.

When this file was read through the option `germanb` we make it behave as if `german` was specified.

```

18.1 \def\bbl@tempa{germanb}
18.2 \ifx\CurrentOption\bbl@tempa
18.3   \def\CurrentOption{german}
18.4 \fi

```

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the `@` sign, etc.

```

18.5 (*code)
18.6 \LdfInit\CurrentOption{captions\CurrentOption}

```

When this file is read as an option, i.e., by the `\usepackage` command, `german` will be an ‘unknown’ language, so we have to make it known. So we check for the

¹⁴The file described in this section has version number v2.6d and was last revised on 1996/12/23.

¹⁵This file is a re-implementation of Hubert Partl’s `german.sty` version 2.5b, see [4].

<code>\glqq</code>	for German left double quotes (looks like „).
<code>\grqq</code>	for German right double quotes (looks like “).
<code>\glq</code>	for German left single quotes (looks like ‚).
<code>\grq</code>	for German right single quotes (looks like ’).
<code>\flqq</code>	for French left double quotes (similar to <<).
<code>\frqq</code>	for French right double quotes (similar to >>).
<code>\flq</code>	for (French) left single quotes (similar to <).
<code>\frq</code>	for (French) right single quotes (similar to >).
<code>\dq</code>	the original quotes character (“).

Table 5: More commands which produce quotes, defined by `german.ldf`

existence of `\l@german` to see whether we have to do something here.

```

18.7 \ifx\l@german\@undefined
18.8   \@nopatterns{German}
18.9   \adddialect\l@german0
18.10 \fi

```

For the Austrian version of these definitions we just add another language.

```

18.11 \adddialect\l@austrian\l@german

```

The next step consists of defining commands to switch to (and from) the German language.

`\captionsgerman` Either the macro `\captionsgerman` or the macro `\captionsaustrian` will define all strings used in the four standard document classes provided with L^AT_EX.

```

18.12 \@namedef{captions\CurrentOption}{%
18.13   \def\prefacename{Vorwort}%
18.14   \def\refname{Literatur}%
18.15   \def\abstractname{Zusammenfassung}%
18.16   \def\bibname{Literaturverzeichnis}%
18.17   \def\chaptername{Kapitel}%
18.18   \def\appendixname{Anhang}%
18.19   \def\contentsname{Inhaltsverzeichnis}%      % oder nur: Inhalt
18.20   \def\listfigurename{Abbildungsverzeichnis}%
18.21   \def\listtablename{Tabellenverzeichnis}%
18.22   \def\indexname{Index}%
18.23   \def\figurename{Abbildung}%
18.24   \def\tablename{Tabelle}%                  % oder: Tafel
18.25   \def\partname{Teil}%
18.26   \def\enclname{Anlage(n)}%                 % oder: Beilage(n)
18.27   \def\ccname{Verteiler}%                   % oder: Kopien an
18.28   \def\headtoname{An}%
18.29   \def\pagename{Seite}%
18.30   \def\seename{siehe}%
18.31   \def\alsoname{siehe auch}%
18.32   \def\proofname{Beweis}%
18.33 }

```

`\dategerman` The macro `\dategerman` redefines the command `\today` to produce German dates.

```

18.34 \def\month@german{\ifcase\month\or

```

```

18.35 Januar\or Februar\or M"arz\or April\or Mai\or Juni\or
18.36 Juli\or August\or September\or Oktober\or November\or Dezember\fi}
18.37 \def\dategerman{\def\today{\number\day.\~\month@german
18.38 \space\number\year}}

```

`\dateaustrian` The macro `\dateaustrian` redefines the command `\today` to produce Austrian version of the German dates.

```

18.39 \def\dateaustrian{\def\today{\number\day.\~\ifnum1=\month
18.40 J"anner\else \month@german\fi \space\number\year}}

```

`\extrasgerman` Either the macro `\extrasgerman` or the macros `\extrasaustrian` will perform all the extra definitions needed for the German language. The macro `\noextrasgerman` is used to cancel the actions of `\extrasgerman`.
`\noextrasaustrian` For German (as well as for Dutch) the " character is made active. This is done once, later on its definition may vary.

```

18.41 \initiate@active@char{"}
18.42 \@namedef{extras\CurrentOption}{%
18.43 \languageshortands{german}}
18.44 \expandafter\addto\csname extras\CurrentOption\endcsname{%
18.45 \bbl@activate{}}
18.46 %\addto\noextrasgerman{\bbl@deactivate{}}

```

In order for T_EX to be able to hyphenate German words which contain ‘ß’ (in the OT1 position $\sim Y$) we have to give the character a nonzero `\lccode` (see Appendix H, the T_EXbook).

```

18.47 \expandafter\addto\csname extras\CurrentOption\endcsname{%
18.48 \babel@savevariable{\lccode25}%
18.49 \lccode25=25}

```

The umlaut accent macro `\` is changed to lower the umlaut dots. The redefinition is done with the help of `\umlautlow`.

```

18.50 \expandafter\addto\csname extras\CurrentOption\endcsname{%
18.51 \babel@save"\umlautlow}
18.52 \@namedef{noextras\CurrentOption}{\umlauthigh}

```

The german hyphenation patterns can be used with `\lefthyphenmin` and `\righthyphenmin` set to 2.

```

18.53 \def\germanhyphenmins{\tw@\tw@}

```

The code above is necessary because we need an extra active character. This character is then used as indicated in table 4.

To be able to define the function of " , we first define a couple of ‘support’ macros.

`\dq` We save the original double quote character in `\dq` to keep it available, the math accent `\` can now be typed as " .

```

18.54 \begingroup \catcode'\12
18.55 \def\x{\endgroup
18.56 \def\@SS{\mathchar"7019 }
18.57 \def\dq{"}}
18.58 \x

```

Now we can define the doublequote macros: the umlauts,

```

18.59 \declare@shorthand{german}{a}{\textormath{"{a}\allowhyphens}{\ddot a}}

```

```

18.60 \declare@shorthand{german}{\o}{\textormath{\{o\}\allowhyphens}{\ddot o}}
18.61 \declare@shorthand{german}{\u}{\textormath{\{u\}\allowhyphens}{\ddot u}}
18.62 \declare@shorthand{german}{\A}{\textormath{\{A\}\allowhyphens}{\ddot A}}
18.63 \declare@shorthand{german}{\O}{\textormath{\{O\}\allowhyphens}{\ddot O}}
18.64 \declare@shorthand{german}{\U}{\textormath{\{U\}\allowhyphens}{\ddot U}}

tremas,
18.65 \declare@shorthand{german}{\e}{\textormath{\{e\}\allowhyphens}{\ddot e}}
18.66 \declare@shorthand{german}{\E}{\textormath{\{E\}\allowhyphens}{\ddot E}}
18.67 \declare@shorthand{german}{\i}{\textormath{\{i\}\allowhyphens}{\ddot i}}%
18.68 \ddot{\imath}}
18.69 \declare@shorthand{german}{\I}{\textormath{\{I\}\allowhyphens}{\ddot I}}

german es-zet (sharp s),
18.70 \declare@shorthand{german}{\s}{\textormath{\ss}}{\@SS}}
18.71 \declare@shorthand{german}{\S}{SS}
18.72 \declare@shorthand{german}{\z}{\textormath{\ss}}{\@SS}}
18.73 \declare@shorthand{german}{\Z}{SZ}

german and french quotes,
18.74 \declare@shorthand{german}{\'}{\glqq}
18.75 \declare@shorthand{german}{\''}{\grqq}
18.76 \declare@shorthand{german}{\<}{\flqq}
18.77 \declare@shorthand{german}{\>}{\frqq}

discretionary commands
18.78 \declare@shorthand{german}{\c}{\textormath{\bbl@disc ck}{c}}
18.79 \declare@shorthand{german}{\C}{\textormath{\bbl@disc CK}{C}}
18.80 \declare@shorthand{german}{\f}{\textormath{\bbl@disc f{ff}}{f}}
18.81 \declare@shorthand{german}{\F}{\textormath{\bbl@disc F{FF}}{F}}
18.82 \declare@shorthand{german}{\l}{\textormath{\bbl@disc l{ll}}{l}}
18.83 \declare@shorthand{german}{\L}{\textormath{\bbl@disc L{LL}}{L}}
18.84 \declare@shorthand{german}{\m}{\textormath{\bbl@disc m{mm}}{m}}
18.85 \declare@shorthand{german}{\M}{\textormath{\bbl@disc M{MM}}{M}}
18.86 \declare@shorthand{german}{\n}{\textormath{\bbl@disc n{nn}}{n}}
18.87 \declare@shorthand{german}{\N}{\textormath{\bbl@disc N{NN}}{N}}
18.88 \declare@shorthand{german}{\p}{\textormath{\bbl@disc p{pp}}{p}}
18.89 \declare@shorthand{german}{\P}{\textormath{\bbl@disc P{PP}}{P}}
18.90 \declare@shorthand{german}{\r}{\textormath{\bbl@disc r{rr}}{r}}
18.91 \declare@shorthand{german}{\R}{\textormath{\bbl@disc R{RR}}{R}}
18.92 \declare@shorthand{german}{\t}{\textormath{\bbl@disc t{tt}}{t}}
18.93 \declare@shorthand{german}{\T}{\textormath{\bbl@disc T{TT}}{T}}

and some additional commands:
18.94 \declare@shorthand{german}{\~}{\penalty\@M-\allowhyphens}
18.95 \declare@shorthand{german}{\|}{\%}
18.96 \textormath{\penalty\@M\discretionary{-}{-}{\kern.03em}%
18.97 \allowhyphens}{}}
18.98 \declare@shorthand{german}{\~}{\hskip\z@skip}
18.99 \declare@shorthand{german}{\~}{\textormath{\leavevmode\hbox{-}{-}{-}}}
18.100 \declare@shorthand{german}{\~}{\penalty\@M-\hskip\z@skip}

```

\mdqon All that's left to do now is to define a couple of commands for reasons of compatibility with `german.sty`.

```

\ck
18.101 \def\mdqon{\bbl@activate{}}
18.102 \def\mdqoff{\bbl@deactivate{}}
18.103 \def\ck{\allowhyphens\discretionary{k-}{k}{ck}\allowhyphens}

```

The macro `\ldf@finish` takes care of looking for a configuration file, setting the main language to be switched on at `\begin{document}` and resetting the category code of `@` to its original value.

18.104 `\ldf@finish\CurrentOption`

18.105 `\code`

19 The Breton language

The file `breton.dtx`¹⁶ defines all the language-specific macros for the Breton language.

There are not really typographic rules for the Breton language. It is a local language (it's one of the celtic languages) which is spoken in Brittany (West of France). So we have a synthesis between french typographic rules and english typographic rules. The characters `:`, `;`, `!` and `?` are made active in order to get a whitespace automatically before these characters.

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the `@` sign, etc.

```
19.1 (*code)
19.2 \LdfInit{breton}\captionsbreton
```

When this file is read as an option, i.e. by the `\usepackage` command, `breton` will be an 'unknown' language in which case we have to make it known. So we check for the existence of `\l@breton` to see whether we have to do something here.

```
19.3 \ifx\l@breton\@undefined
19.4     \nopatterns{Breton}
19.5     \adddialect\l@breton0\fi
```

The next step consists of defining commands to switch to the English language. The reason for this is that a user might want to switch back and forth between languages.

`\captionsbreton` The macro `\captionsbreton` defines all strings used in the four standard document classes provided with L^AT_EX.

```
19.6 \addto\captionsbreton{%
19.7     \def\prefacename{Rakskrid}%
19.8     \def\refname{Daveenno\'u}%
19.9     \def\abstractname{Dvierra\~n}%
19.10 \def\bibname{Lennadurezh}%
19.11 \def\chaptername{Pennad}%
19.12 \def\appendixname{Stagadenn}%
19.13 \def\contentsname{Taolenn}%
19.14 \def\listfigurename{Listenn ar Figurenno\'u}%
19.15 \def\listtablename{Listenn an taolenno\'u}%
19.16 \def\indexname{Meneger}%
19.17 \def\figurename{Figurenn}%
19.18 \def\tablename{Taolenn}%
19.19 \def\partname{Lodenn}%
19.20 \def\enclname{Diello\'u kevret}%
19.21 \def\ccname{Eilskrid da}%
19.22 \def\headtoname{evit}
19.23 \def\pagename{Pajenn}%
19.24 \def\seename{Gwelout}%
19.25 \def\alsoname{Gwelout ivez}%
19.26 \def\proofname{Proof}% <-- needs translation
19.27 }
```

¹⁶The file described in this section has version number v1.0e and was last revised on 1996/12/23.

`\datebreton` The macro `\datebreton` redefines the command `\today` to produce Breton dates.

```
19.28 \def\datebreton{%
19.29 \def\today{\ifnum\day=1\relax 1\/$^{\rm a\tilde{n}}$\else
19.30 \number\day\fi \space a\space viz\space\ifcase\month\or
19.31 Genver\or C’hwevrer\or Meurzh\or Ebrel\or Mae\or Mezheven\or
19.32 Gouere\or Eost\or Gwengolo\or Here\or Du\or Kerzu\fi
19.33 \space\number\year}}
```

`\extrasbreton` The macro `\extrasbreton` will perform all the extra definitions needed for the Breton language. The macro `\noextrasbreton` is used to cancel the actions of `\extrasbreton`.

The category code of the characters `:`, `;`, `!` and `?` is made `\active` to insert a little white space.

```
19.34 \initiate@active@char{:}
19.35 \initiate@active@char{;}
19.36 \initiate@active@char{!}
19.37 \initiate@active@char{?}
```

We specify that the breton group of shorthands should be used.

```
19.38 \addto\extrasbreton{\languageshorthands{breton}}
```

These characters are ‘turned on’ once, later their definition may vary.

```
19.39 \addto\extrasbreton{%
19.40 \bbl@activate{:}\bbl@activate{;}%
19.41 \bbl@activate{!}\bbl@activate{?}}
19.42 %\addto\noextrasbreton{%
19.43 % \bbl@deactivate{:}\bbl@deactivate{;}%
19.44 % \bbl@deactivate{!}\bbl@deactivate{?}}
```

The last thing `\extrasbreton` needs to do is to make sure that `\frenchspacing` is in effect. If this is not the case the execution of `\noextrasbreton` will switch it of again.

```
19.45 \addto\extrasbreton{\bbl@frenchspacing}
19.46 \addto\noextrasbreton{\bbl@nonfrenchspacing}
```

`\breton@sh@;` We have to reduce the amount of white space before `;`, `:` and `!` when the user types a space in front of these characters. This should only happen outside mathmode, hence the test with `\ifmmode`.

```
19.47 \declare@shorthand{breton}{;}{;%
19.48 \ifmmode
19.49 \string;\space
19.50 \else\relax
```

In horizontal mode we check for the presence of a ‘space’ and replace it by a `\thinspace`.

```
19.51 \ifhmode
19.52 \ifdim\lastskip>\z@
19.53 \unskip\penalty\@M\thinspace
19.54 \fi
19.55 \fi
19.56 \string;\space
19.57 \fi}%
```


\breton@sh@:@ Because these definitions are very similar only one is displayed in a way that the
 \breton@sh@!@ definition can be easily checked.

```

19.58 \declare@shorthand{breton}{:}{%
19.59   \ifmmode\string:\space
19.60   \else\relax
19.61   \ifhmode
19.62     \ifdim\lastskip>\z@\unskip\penalty\@M\thinspace\fi
19.63   \fi
19.64   \string:\space
19.65 \fi}
19.66 \declare@shorthand{breton}{!}{%
19.67   \ifmmode\string!\space
19.68   \else\relax
19.69   \ifhmode
19.70     \ifdim\lastskip>\z@\unskip\penalty\@M\thinspace\fi
19.71   \fi
19.72   \string!\space
19.73 \fi}

```

\breton@sh@?@ For the question mark something different has to be done. In this case the amount
 of white space that replaces the space character depends on the dimensions of the
 font.

```

19.74 \declare@shorthand{breton}{?}{%
19.75   \ifmmode
19.76     \string?\space
19.77   \else\relax
19.78   \ifhmode
19.79     \ifdim\lastskip>\z@
19.80       \unskip
19.81       \kern\fontdimen2\font
19.82       \kern-1.4\fontdimen3\font
19.83     \fi
19.84   \fi
19.85   \string?\space
19.86 \fi}

```

All that is left to do now is provide the breton user with some extra utilities.
 Some definitions for special characters.

```

19.87 \DeclareTextSymbol{\at}{OT1}{64}
19.88 \DeclareTextSymbol{\at}{T1}{64}
19.89 \DeclareTextSymbolDefault{\at}{OT1}
19.90 \DeclareTextSymbol{\boi}{OT1}{92}
19.91 \DeclareTextSymbol{\boi}{T1}{16}
19.92 \DeclareTextSymbolDefault{\boi}{OT1}
19.93 \DeclareTextSymbol{\circonflexe}{OT1}{94}
19.94 \DeclareTextSymbol{\circonflexe}{T1}{2}
19.95 \DeclareTextSymbolDefault{\circonflexe}{OT1}
19.96 \DeclareTextSymbol{\tild}{OT1}{126}
19.97 \DeclareTextSymbol{\tild}{T1}{3}
19.98 \DeclareTextSymbolDefault{\tild}{OT1}
19.99 \DeclareTextSymbol{\degre}{OT1}{23}
19.100 \DeclareTextSymbol{\degre}{T1}{6}
19.101 \DeclareTextSymbolDefault{\degre}{OT1}

```

The following macros are used in the redefinition of `\^` and `\"` to handle the letter i.

```
19.102 \AtBeginDocument{%
19.103   \DeclareTextCompositeCommand{\^}{OT1}{i}{\^i}
19.104   \DeclareTextCompositeCommand{\"}{OT1}{i}{\"i}}
```

And some more macros for numbering.

```
19.105 \def\kentan{1\/$}\rm a\tilde{n}}$}
19.106 \def\eil{2\/$}\rm l}$}
19.107 \def\re{\/$}\rm re}$}
19.108 \def\trede{3\re}
19.109 \def\pevare{4\re}
19.110 \def\vet{\/$}\rm vet}$}
19.111 \def\pempvet{5\vet}
```

The macro `\ldf@finish` takes care of looking for a configuration file, setting the main language to be switched on at `\begin{document}` and resetting the category code of `@` to its original value.

```
19.112 \ldf@finish{breton}
19.113 \code}
```

20 The Welsh language

The file `welsh.dtx`¹⁷ defines all the language definition macros for the Welsh language as well as for the `¡Dialect¿` version of this language.

For this language currently no special definitions are needed or available.

The macro `\ldf@init` takes care of preventing that this file is loaded more than once, checking the category code of the `@` sign, etc.

```
20.1 (*code)
20.2 \LdfInit{welsh}{captionswelsh}
```

When this file is read as an option, i.e. by the `\usepackage` command, `welsh` could be an ‘unknown’ language in which case we have to make it known. So we check for the existence of `\l@welsh` to see whether we have to do something here.

```
20.3 \ifx\undefined\l@welsh
20.4   \@nopatterns{welsh}
20.5   \adddialect\l@welsh0\fi
```

The next step consists of defining commands to switch to (and from) the Welsh language.

`\welshhyphenmins` This macro is used to store the correct values of the hyphenation parameters `\lefthyphenmin` and `\righthyphenmin`.

```
20.6 \def\welshhyphenmins{23}
```

`\captionswelsh` The macro `\captionswelsh` defines all strings used in the four standard documentclasses provided with L^AT_EX.

```
20.7 \def\captionswelsh{%
20.8   \def\prefacename{Rhagair}%
20.9   \def\refname{Cyfeiriadau}%
20.10  \def\abstractname{Crynodeb}%
20.11  \def\bibname{Llyfryddiaeth}%
20.12  \def\chaptername{Pennod}%
20.13  \def\appendixname{Atodiad}%
20.14  \def\contentsname{Cynnwys}%
20.15  \def\listfigurename{Rhestr Ddarluniau}%
20.16  \def\listtablename{Rhestr Dablau}%
20.17  \def\indexname{Mynegai}%
20.18  \def\figurename{Darlun}%
20.19  \def\tablename{Taflen}%
20.20  \def\partname{Rhan}%
20.21  \def\enclname{amgae"edig}%
20.22  \def\ccname{cop\"i au}%
20.23  \def\headtoname{At}% % ‘at’ on letters meaning ‘to ( a person)’
20.24                      % ‘to (a place)’ is ‘i’ in Welsh
20.25  \def\pagename{tudalen}%
20.26  \def\seename{gweler}%
20.27  \def\alsoname{gweler hefyd}%
20.28  \def\proofname{Prawf}%
20.29 }
```

¹⁷The file described in this section has version number v1.0a and was last revised on 1996/12/23.

`\datewelsh` The macro `\datewelsh` redefines the command `\today` to produce welsh dates.

```
20.30 \def\datewelsh{%
20.31   \def\today{\ifnum\day=1\relax 1\/$^{\mathrm{a\tilde{n}}}}$\else
20.32     \number\day\fi \space a\space viz\space \ifcase\month\or
20.33     Ionawr\or Chwefror\or Mawrth\or Ebrill\or
20.34     Mai\or Mehefin\or Gorffennaf\or Awst\or
20.35     Medi\or Hydref\or Tachwedd\or Rhagfyr\fi
20.36   \space\number\year}}
```

`\extraswelsh` The macro `\extraswelsh` will perform all the extra definitions needed for the
`\noextraswelsh` welsh language. The macro `\noextraswelsh` is used to cancel the actions of
`\extraswelsh`. For the moment these macros are empty but they are defined for
compatibility with the other language definition files.

```
20.37 \addto\extraswelsh{}
20.38 \addto\noextraswelsh{}
```

The macro `\ldf@finish` takes care of looking for a configuration file, setting
the main language to be switched on at `\begin{document}` and resetting the
category code of `@` to its original value.

```
20.39 \ldf@finish{welsh}
20.40 \code
```

21 The Irish language

The file `irish.dtx`¹⁸ defines all the language definition macros for the Irish language.

For this language currently no special definitions are needed or available.

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the `@` sign, etc.

```
21.1 (*code)
21.2 \LdfInit{irish}\captionsirish
```

When this file is read as an option, i.e. by the `\usepackage` command, `irish` could be an ‘unknown’ language in which case we have to make it known. So we check for the existence of `\l@irish` to see whether we have to do something here.

```
21.3 \ifx\l@irish\@undefined
21.4   \@nopatterns{irish}
21.5   \adddialect\l@irish0\fi
```

The next step consists of defining commands to switch to (and from) the Irish language.

`\captionsirish` The macro `\captionsirish` defines all strings used in the four standard documentclasses provided with L^AT_EX.

```
21.6 \addto\captionsirish{%
21.7   \def\prefacename{Preface}%      <-- needs translation
21.8   \def\refname{Tagairt\'}{\i}%
21.9   \def\abstractname{Achoimre}%
21.10  \def\bibname{Leabharliosta}%
21.11  \def\chaptername{Caibidil}%
21.12  \def\appendixname{Aguis\'}{\i n}%
21.13  \def\contentsname{Cl\'ar \'}{Abhair}%
21.14  \def\listfigurename{L\'ear\'aid\'}{\i}%
21.15  \def\listtablename{T\'abla\'}{\i}%
21.16  \def\indexname{Inn\'eacs}%
21.17  \def\figurename{L\'ear\'aid}%
21.18  \def\tablename{T\'abla}%
21.19  \def\partname{Cuid}%
21.20  \def\enclname{faoi iamh}%
21.21  \def\ccname{cc}%                abrv. ‘c\'oip chuig’
21.22  \def\headtoname{Go}%
21.23  \def\pagename{Leathanach}%
21.24  \def\seename{see}%      <-- needs translation
21.25  \def\alsosome{see also}%    <-- needs translation
21.26  \def\proofname{Proof}%    <-- needs translation
21.27 }
```

`\dateirish` The macro `\dateirish` redefines the command `\today` to produce Irish dates.

```
21.28 \def\dateirish{\def\today{%
21.29   \number\day\space \ifcase\month\or
21.30   Ean\'air\or Feabhra\or M\'arta\or Aibre\'an\or
21.31   Bealtaine\or Meitheamh\or I\'uil\or L\'unasa\or
21.32   Me\'an F\'omhair\or Deireadh F\'omhair\or
```

¹⁸The file described in this section has version number v1.0f and was last revised on 1998/05/05. A contribution was made by Marion Gunn.

```

21.33 M'\{i} na Samhna\or M'\{i} na Nollag\fi
21.34 \space \number\year}}

```

`\extrasirish` The macro `\extrasirish` will perform all the extra definitions needed for the Irish language. The macro `\noextrasirish` is used to cancel the actions of `\extrasirish`. For the moment these macros are empty but they are defined for compatibility with the other language definition files.

```

21.35 \addto\extrasirish{}
21.36 \addto\noextrasirish{}

```

The macro `\ldf@finish` takes care of looking for a configuration file, setting the main language to be switched on at `\begin{document}` and resetting the category code of `@` to its original value.

```

21.37 \ldf@finish{irish}
21.38 \code

```

22 The Scottish language

The file `scottish.dtx`¹⁹ defines all the language definition macros for the Scottish language.

For this language currently no special definitions are needed or available.

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the `@` sign, etc.

```
22.1 (*code)
22.2 \LdfInit{scottish}\captionsscottish
```

When this file is read as an option, i.e. by the `\usepackage` command, `scottish` could be an ‘unknown’ language in which case we have to make it known. So we check for the existence of `\l@scottish` to see whether we have to do something here.

```
22.3 \ifx\l@scottish\@undefined
22.4 \@nopatterns{scottish}
22.5 \adddialect\l@scottish0\fi
```

The next step consists of defining commands to switch to (and from) the Scottish language.

`\captionsscottish` The macro `\captionsscottish` defines all strings used in the four standard documentclasses provided with L^AT_EX.

```
22.6 \addto\captionsscottish{%
22.7   \def\prefacename{Preface}%    <-- needs translation
22.8   \def\refname{Iomraidh}%
22.9   \def\abstractname{Br\‘{i}gh}%
22.10  \def\bibname{Leabhraichean}%
22.11  \def\chaptername{Caibideil}%
22.12  \def\appendixname{Ath-sgr‘{i}obhadh}%
22.13  \def\contentsname{Cl‘ar-obrach}%
22.14  \def\listfigurename{Liosta Dhealbh }%
22.15  \def\listtablename{Liosta Chl‘ar}%
22.16  \def\indexname{Cl‘ar-innse}%
22.17  \def\figurename{Dealbh}%
22.18  \def\tablename{Cl‘ar}%
22.19  \def\partname{Cuid}%
22.20  \def\enclname{a-staigh}%
22.21  \def\ccname{lethbhreac gu}%
22.22  \def\headtoname{gu}%
22.23  \def\pagename{t.d.}%          abrv. ‘taobh duilleag’
22.24  \def\seenname{see}%    <-- needs translation
22.25  \def\alsoname{see also}%    <-- needs translation
22.26  \def\proofname{Proof}%    <-- needs translation
22.27 }
```

`\datescottish` The macro `\datescottish` redefines the command `\today` to produce Scottish dates.

```
22.28 \def\datescottish{%
22.29   \number\day\space \ifcase\month\or
22.30   am Faoilteach\or an Gearran\or am M‘art\or an Giblean\or
```

¹⁹The file described in this section has version number v1.0d and was last revised on 1996/12/23. A contribution was made by Fraser Grant (FRASER@CERNVM).

```

22.31 an C\‘eitean\or an t-\‘Og mhios\or an t-Iuchar\or
22.32 L\‘unasdal\or an Sultuine\or an D\‘amhar\or
22.33 an t-Samhainn\or an Dubhlachd\fi
22.34 \space \number\year}}

```

`\extrasscottish` The macro `\extrasscottish` will perform all the extra definitions needed for the Scottish language. The macro `\noextrasscottish` is used to cancel the actions of `\extrasscottish`. For the moment these macros are empty but they are defined for compatibility with the other language definition files.

```

22.35 \addto\extrasscottish{}
22.36 \addto\noextrasscottish{}

```

The macro `\ldf@finish` takes care of looking for a configuration file, setting the main language to be switched on at `\begin{document}` and resetting the category code of `@` to its original value.

```

22.37 \ldf@finish{scottish}
22.38 \code

```


23 The Greek language

The file `greek.dtx`²⁰ defines all the language definition macros for the Greek language.

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the `@` sign, etc.

```
23.1 (*code)
23.2 \LdfInit{greek}\captionsgreek
```

When this file is read as an option, i.e. by the `\usepackage` command, `greek` could be an ‘unknown’ language in which case we have to make it known. So we check for the existence of `\l@greek` to see whether we have to do something here.

```
23.3 \ifx\l@greek\@undefined
23.4   \@nopatterns{greek}
23.5   \adddialect\l@greek0\fi
```

Tysetting Greek texts implies that a special set of fonts needs to be used. The current support for greek uses a set of fonts that more or less ‘match’ with the computer modern fonts. Therefore we define the Local GRreek encoding (LGR, see the file `greek.fdd`). We make sure that this encoding is known to L^AT_EX.

```
23.6 \input{LGRenc.def}
```

`\latinencoding` We need to know the encoding for text that is supposed to be typeset in latin text. We assume that it will be the encoding which is active at the end of the `babel` package. If the `fontenc` package is loaded later, then... too bad!

```
23.7 \AtEndOfPackage{\edef\latinencoding{\cf@encoding}}
```

Now we define two commands that offer the possibility to switch between greek and roman encodings.

`\greektext` The command `\greektext` will switch from latin font encoding to the greek font encoding, the command `\latintext` switches back. This assumes that the ‘normal’ font encoding is a latin one. These commands are *declarations*, for shorter peaces of text the commands `\textlatin` and `\textgreek` can be used.

```
23.8 \DeclareRobustCommand{\greektext}{%
23.9   \fontencoding{LGR}\selectfont
23.10  \def\encodingdefault{LGR}}
23.11 \DeclareRobustCommand{\latintext}{%
23.12   \fontencoding{\latinencoding}\selectfont
23.13   \def\encodingdefault{\latinencoding}}
```

`\textgreek` These commands take an argument which is then typeset using the requested font encoding. In order to avoid many encoding switches both commands operate in a local scope.

```
23.14 \DeclareRobustCommand{\textgreek}[1]{\greektext #1}
23.15 \DeclareRobustCommand{\textlatin}[1]{\latintext #1}
```

`\textol` A last aspect of the set of fonts provided with this version of support for typesetting Greek texts is that it contains an outline font. In order to make it available we define the command `\textol`.

²⁰The file described in this section has version number v1.0b and was last revised on 1996/12/23. The original author is Apostolos Syropoulos (`apostolo@platon.ee.duth.gr`).

```

23.16 \def\outlfamily{\usefont{LGR}{cmro}{m}{n}}
23.17 \DeclareTextFontCommand{\textol}{\outlfamily}

```

The next step consists of defining commands to switch to (and from) the Greek language.

\greekhyphenmins This macro is used to store the correct values of the hyphenation parameters **\lefthyphenmin** and **\righthyphenmin**.

```

23.18 \def\greekhyphenmins{23}

```

\captionsgreek The macro **\captionsgreek** defines all strings used in the four standard documentclasses provided with L^AT_EX.

```

23.19 \addto\captionsgreek{%
23.20   \def\prefacename{Pr'ologoc}%
23.21   \def\refname{Anafor'ec}%
23.22   \def\abstractname{Per'ilhyh}%
23.23   \def\bibname{Bibliograf'ia}%
23.24   \def\chaptername{Kef'alaio}%
23.25   \def\appendixname{Par'arthma}%
23.26   \def\contentsname{Perieq'omena}%
23.27   \def\listfigurename{Kat'alogoc Sqhm'atwn}%
23.28   \def\listtablename{Kat'alogoc Pin'akwn}%
23.29   \def\indexname{Euret'hrio}%
23.30   \def\figurename{Sq'hma}%
23.31   \def\tablename{P'inakac}%
23.32   \def\partname{M'eroc}%
23.33   \def\enclname{Sunhmm'ena}%
23.34   \def\ccname{Koinopo'ihsh}%
23.35   \def\headtoname{Proc}%
23.36   \def\pagename{Sel'ida}%
23.37   \def\seename{bl'epe}%
23.38   \def\alsoname{bl'pe ep'ishc}%
23.39 }

```

\dategreek The macro **\dategreek** redefines the command **\today** to produce greek dates.

```

23.40 \def\dategreek{%
23.41   \def\today{\number\day \space%
23.42     \ifcase\month\or
23.43     Ianouar'iou\or Febrouar'iou\or Mart'iou\or April'iou\or
23.44     Ma'"iou\or Ioun'iou\or Ioul'iou\or Augo'ustou\or
23.45     Septembr'iou\or Oktobr'iou\or Noembr'iou\or Dekembr'iou\fi
23.46     \space \number\year}}

```

\extrasgreek The macro **\extrasgreek** will perform all the extra definitions needed for the Greek language. The macro **\noextrasgreek** is used to cancel the actions of **\extrasgreek**. For the moment these macros switch the fontencoding used and the definition of the internal macros **\@alph** and **\@Alph** because the greek alphabetisation differs considerably from the roman alphabetisation..

```

23.47 \addto\extrasgreek{\greektext}
23.48 \addto\noextrasgreek{\latintext}

```

\greek@alph With the latin transcription used to create documents that will be typeset in the greek language we need to adopt the alphabetisation considerably. Therefore we have to redefine the internal L^AT_EX commands **\@alph** and **\@Alph**.

We need to be able to switch between the original definitions and the greek ones, so we first ‘save’ the original definitions.

```
23.49 \let\latin@alph\@alph
23.50 \let\latin@Alph\@Alph
```

Then we define the greek versions

```
23.51 \def\greek@alph#1{%
23.52   \ifcase #1\or a\or b\or g\or d\or e\or st\or z\or h\or j\or i\or
23.53     ia\or ib\or ig\or id\or ie\or ist\or iz\or ih\or ij\or k\or ka\or
23.54     kb\or kg\or \kd \or ke\or kst
23.55   \else
23.56     \@ctrerr
23.57   \fi
23.58   $'$_}
23.59 \def\greek@Alph#1{%
23.60   \ifcase #1\or A\or B\or G\or D\or E\or St\or Z\or H\or J\or I\or
23.61     IA\or IB\or IG\or ID\or IE\or IST\or IZ\or IH\or IJ\or K\or KA\or
23.62     KB\or KG\or \KD \or KE\or KST
23.63   \else
23.64     \@ctrerr
23.65   \fi
23.66   $'$_}
```

Now we can set up the switching.

```
23.67 \addto\extrasgreek{%
23.68   \let\@alph\@greek@alph
23.69   \let\@Alph\@greek@Alph}
23.70 \addto\noextrasgreek{%
23.71   \let\@alph\latin@alph
23.72   \let\@Alph\latin@Alph}
```

\greek@roman To prevent roman numerals being typeset in greek letters we need to adopt the **\greek@Roman** internal L^AT_EX commands \@roman and \@Roman.

```
23.73 \let\latin@roman\@roman
23.74 \let\latin@Roman\@Roman
23.75 \def\greek@roman#1{\textlatin{\latin@roman{#1}}}
23.76 \def\greek@Roman#1{\textlatin{\latin@Roman{#1}}}
23.77 \addto\extrasgreek{%
23.78   \let\@roman\@greek@roman
23.79   \let\@Roman\@greek@Roman}
23.80 \addto\noextrasgreek{%
23.81   \let\@roman\latin@roman
23.82   \let\@Roman\latin@Roman}
```

We provide access to a few extra greek characters. They are only available in one particular font, therefore we first define a ‘helper’ macro to select the correct font.

```
23.83 \def\greek@char#1{%
23.84   \fontfamily\rmdefault
23.85   \fontseries\mddefault
23.86   \fontshape\scdefault
23.87   \selectfont\char#1}}
23.88 \DeclareTextCommand{\tao}{LGR}{\greek@char{"7F}}
23.89 \DeclareTextCommand{\Qoppa}{LGR}{\greek@char{"43}}
```

```

23.90 \DeclareTextCommand{\qoppa}{LGR}{\greek@char{"5B}}
23.91 \DeclareTextCommand{\varqoppa}{LGR}{\greek@char{"5C}}
23.92 \DeclareTextCommand{\Sampi}{LGR}{\greek@char{"5F}}
23.93 \DeclareTextCommand{\sampi}{LGR}{\greek@char{"5E}}
23.94 \DeclareTextCommand{\Digamma}{LGR}{\greek@char{"17}}

```

The amssymb package defines a `\digamma` command, so in order to avoid problems we spell here digamma intentionally erroneous with a double d.

```

23.95 \DeclareTextCommand{\ddigamma}{LGR}{\greek@char{"60}}
23.96 \DeclareTextCommand{\vardigamma}{LGR}{\greek@char{"5D}}

```

Now make sure that these commands can also be used outside of the greek font encoding.

```

23.97 \ProvideTextCommandDefault{\tao}{\UseTextSymbol{LGR}{\tao}}
23.98 \ProvideTextCommandDefault{\Qoppa}{\UseTextSymbol{LGR}{\Qoppa}}
23.99 \ProvideTextCommandDefault{\qoppa}{\UseTextSymbol{LGR}{\qoppa}}
23.100 \ProvideTextCommandDefault{\varqoppa}{\UseTextSymbol{LGR}{\varqoppa}}
23.101 \ProvideTextCommandDefault{\Sampi}{\UseTextSymbol{LGR}{\Sampi}}
23.102 \ProvideTextCommandDefault{\sampi}{\UseTextSymbol{LGR}{\sampi}}
23.103 \ProvideTextCommandDefault{\Digamma}{\UseTextSymbol{LGR}{\Digamma}}
23.104 \ProvideTextCommandDefault{\ddigamma}{\UseTextSymbol{LGR}{\ddigamma}}
23.105 \ProvideTextCommandDefault{\vardigamma}{\%
23.106   \UseTextSymbol{LGR}{\vardigamma}}

```

The macro `\ldf@finish` takes care of looking for a configuration file, setting the main language to be switched on at `\begin{document}` and resetting the category code of `@` to its original value.

```

23.107 \ldf@finish{greek}
23.108 \code

```

24 The French Language

24.1 About French typography

The file `frenchb.dtx`²¹, derived from `frenchy.sty`, defines all the language definition macros for the French language.

Customization for the French language is achieved following the book “Lexique des règles typographiques en usage à l’Imprimerie nationale” troisième édition (1994), ISBN-2-11-081075-0.

This file has been designed to be used with $\text{\LaTeX 2}_{\epsilon}$, \LaTeX -2.09 and PlainTeX formats. If you are still using \LaTeX -2.09, you *should* consider switching to $\text{\LaTeX 2}_{\epsilon}$!

Any of the commands `\selectlanguage{french}`, `\selectlanguage{français}`, or `\selectlanguage{frenchb}` switches to the French language with the following effects:

1. French hyphenation patterns are made active;
2. ‘double punctuation’ is made active for correct spacing in French;
3. `\today` prints the date in French;
4. the caption names are translated into French (\LaTeX only);
5. the list items are set to ‘–’ instead of • (\LaTeX only);
6. the vertical spacing in lists is shortened (\LaTeX only);
7. the first paragraph of each section is indented (\LaTeX only);
8. French quotation marks can be typeset using the commands `\og` and `\fg` which work in $\text{\LaTeX 2}_{\epsilon}$, \LaTeX -2.09 and PlainTeX , their appearance depending on what is available to draw them; if you use $\text{\LaTeX 2}_{\epsilon}$ with T1-encoding you can also enter them as `<<~French quotation marks~>>` but then *don’t forget* the unbreakable spaces, (`\og` and `\fg` provide for correct line breaks);
9. a command `\up` is provided to typeset superscripts like `M\up{me}` (abbreviation for “Madame”), `1\up{er}` (for “premier”);
10. family names should be typeset in small capitals and never be hyphenated, the macro `\bsc` (boxed small caps) does this, e.g., `Leslie~\bsc{Lamport}` will produce Leslie LAMPORT;
11. commands `\primo`, `\secundo`, `\tertio` and `\quarto` may be used to enumerate in lists;
12. abbreviations for “Numéro” and “numéro” are obtained via the commands `\No`, `\no`;
13. two commands are provided to typeset abbreviations for “degré”: `\degre` prints the raw character and `\degres` should be used to typeset temperatures (e.g., `“20~\degres C”` with an unbreakable space), or for alcohols’ strengths (e.g., `“45\degres”` with *no* space in French);

²¹The file described in this section has version number v1.2a and was last revised on 1997/01/11.

14. an command `\nombre` is provided to ease the typesetting of numbers: it works both in text and in math-mode: inputting `\nombre{3141,592653}` will format this number properly according to the current language (French or non-French)²². The command `\nombre` is a contribution of Vincent Jalby using ideas of David Carlisle in `comma.sty`.

All commands previously available in `francais.ldf` have been included in `frenchb.ldf` for compatibility, sometimes with updated definitions.

The `french` package, by Bernard GAULLE, was not designed to run with `babel` (although the latest versions claim to be `babel` compatible), but rather as a stand-alone package for the French language. It provides many more functionalities (like `\lettrine`, `\sommaire...`) not available in `frenchb`, at the cost of a much greater complexity and possible incompatibilities with other languages.

As `french` is known to produce the best layout available for French typography, I have borrowed many ideas from Bernard's file. I did my best to help users of both packages (`french` and `frenchb`) to exchange their sources files easily, with one exception which affects the way French quotation marks are entered: `frenchb` uses *macros* (`\og` and `\fg`) while `french` uses active characters (`<<` and `>>`).

French typographic rules specify that some white space should be present before 'double punctuation' characters. These characters are ; ! ? and :. In order to get this white space automatically, the category code of these characters is made `\active`. In French, the user *should* input these four characters preceded with a space, but as many people forget about it (even among native French writers!), the default behaviour of `frenchb` is to automatically add a `\thinspace` before ' ; ' ! ' ? ' and a normal (unbreakable) space before ' : ' (this is the rule in French typography). It's up to the user to add or not a space *after* 'double punctuation' characters: usually a space is necessary, but not always (before a full point or a closing brace for instance), so this cannot done automatically.

In (rare) cases where no space should be added before a 'double punctuation', either use `\string; \string: \string! \string?` instead of `; : ! ?`, or switch locally to `english`. For instance you can type `C\string:TEX` or `\begin{otherlanguage}{english}{C:TEX}\end{otherlanguage}` to avoid the space before `:` in a MS-DOS path.

Some users dislike this automatic insertion of a space before 'double punctuation', and prefer to decide themselves whether a space should be added or not; so a hook `\NoAutoSpaceBeforeFDP` is provided: if this command is added (in file `frenchb.cfg`, or anywhere in a document) `frenchb` will respect your typing, and introduce a suitable space before 'double punctuation' *if and only if* a space is typed in the source file before those signs.

The command `\AutoSpaceBeforeFDP` switches back to the default behavior of `frenchb`.

Once you have built your format, a good precaution would be to perform some basic tests about hyphenation in French. For $\text{\LaTeX 2}_{\epsilon}$ I suggest this:

²²In math-mode the comma has to be surrounded with braces to avoid a spurious space being inserted after it (see the `TEXbook` p. 134). Besides this, each slice of three digits should be separated either with a comma in English or with a space in French.

- run the following file, with the encoding suitable for your machine (*my-encoding* will be `latin1` for UNIX machines and PCs running Windows, `applemac` for Macintoshes, or `cp850` for PCs running DOS. If you are using MIT_εX together with CMR fonts, comment out the line `\usepackage[my-encoding]{inputenc}`.

```
%% Test file for French hyphenation.
\documentclass{article}
\usepackage[my-encoding]{inputenc}
\usepackage[francais]{babel}
\begin{document}
\showhyphens{signal, container, \’ev\’enement, alg\’ebre}
\showhyphens{signal, container, évènement, algèbre}
\end{document}
```

- check the hyphenations proposed by T_εX in your log-file; in French you should get with both 7-bit and 8-bit encodings `si-gnal, contai-ner, évé-ne-ment, al-gèbre`. Do not care about how accented characters are displayed in the log-file, what matters is the position of the ‘-’ hyphen signs *only*.

If they are all correct, your installation (probably) works fine, if one (or more) is (are) wrong, ask a local wizard to see what’s going wrong and perform the test again (or e-mail me about what happens).

Frequent mismatches:

- you get `sig-nal, con-tainer`, this probably means that the hyphenation patterns you are using are for USenglish, not for French;
- you get no hyphen at all in `évé-ne-ment`, this probably means that you are using CMR fonts and the macro `\accent` to produce accented characters. Consider switching to DC/EC fonts and T1-encoding or use MIT_εX.

`frenchb` has been improved using helpful suggestions from many people, the main contributions came from Vincent Jalby. Thanks to all of them!

First version released: 1.1 as of 1996/05/31 part of `babel-3.6beta`.

Changes in version 1.1b: update for `babel-3.6`.

Changes in version 1.2: new command `\nombre` to format numbers; removed command `\fup` borrowed from the `french` package (`\up` does a better job in L^AT_εX 2_ε); also removed aliases `\french` and `\english` (`frenchb.cfg` is a better place for these).

24.2 T_εXnical details

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the `@` sign, etc.

```
24.1 (*code)
24.2 %% Please report errors to: Daniel Flipo, GUTenberg
24.3 %%                        Daniel.Flipo@univ-lille1.fr
24.4 %%
24.5 \LdfInit{frenchb}\NoAutoSpaceBeforeFDP
```

Check if hyphenation patterns for the French language have been loaded in language.dat: requested name ‘french’ or ‘français’.

```
24.6 \ifx\l@french\@undefined
24.7 \ifx\l@français\@undefined
24.8 \nopatterns{French}
24.9 \adddialect\l@french0
24.10 \fi
24.11 \fi
```

Regardless of `\CurrentOption` the internal name for the French language will be ‘frenchb’; ‘français’ and ‘french’ will be synonymous for ‘frenchb’: first let both names use the same hyphenation patterns. Later we will have to set aliases for `\captionsfrenchb`, `\datefrenchb`, `\extrasfrenchb` and `\noextrasfrenchb`. As French uses the standard values of `\lefthyphenmin`(2) and `\righthyphenmin`(3), no special setting is required here.

```
24.12 \def\CurrentOption{frenchb}
24.13 \ifx\l@français\@undefined
24.14 \let\l@français\l@french
24.15 \else
24.16 \let\l@french\l@français
24.17 \fi
24.18 \let\l@frenchb\l@french
```

To check the format in use (plain or LaTeX), we’ll need macros to hold the names of the plain and L^AT_EX 2_ε formats.

```
24.19 \def\PlainFmtName{plain}
24.20 \def\LaTeXeFmtName{LaTeX2e}
```

`\if@Two@E` We will need a new ‘if’: `\if@Two@E` is true if and only if L^AT_EX 2_ε is running *not* in compatibility mode. It is used in the definitions of the command `\nombre` and `\up`. The definition is somewhat complicated, due to the fact that `\if@compatibility` is not recognized as a `\if` in L^AT_EX-2.09 based formats.

```
24.21 \newif\if@Two@E \@Two@Etrue
24.22 \def\@FI@{\fi}
24.23 \ifx\@compatibilitytrue\@undefined
24.24 \@Two@Efalse \def\@FI@{\relax}
24.25 \else
24.26 \if@compatibility \@Two@Efalse \fi
24.27 \@FI@
```

`\extrasfrenchb` The macro `\extrasfrenchb` will perform all the extra definitions needed for the French language. The macro `\noextrasfrenchb` is used to cancel the actions of `\extrasfrenchb`.

In French “apostrophe” is used in hyphenation in expressions like l’**ambulance** (French patterns provide entries for this kind of words). This means that the `\lccode` of “apostrophe” has to be non null in French for proper hyphenation of those expressions, and to be reset to null when exiting French.

```
24.28 \@namedef{extras\CurrentOption}{\lccode'\=''}
24.29 \@namedef{noextras\CurrentOption}{\lccode'\=0}
24.30 \def\extrasfrançais{\extrasfrenchb}
24.31 \def\extrasfrench{\extrasfrenchb}
24.32 \def\noextrasfrançais{\noextrasfrenchb}
24.33 \def\noextrasfrench{\noextrasfrenchb}
```


It is best to use L^AT_EX 2_ε's font changing commands, and to emulate those we need when they are not available, as in PlainT_EX or L^AT_EX-2.09. Be aware that old commands `\sc`, `\it`, *etc.* exist in L^AT_EX 2_ε, but they behave like they did in L^AT_EX-2.09 (i.e., they switch back to `\normalfont` instead of keeping the other font attributes unchanged).

```

24.34 \ifx\scshape\@undefined
24.35   \ifx\sc\@undefined
24.36     \let\scshape\relax
24.37   \else
24.38     \let\scshape\sc
24.39   \fi
24.40 \fi
24.41 \ifx\emph\@undefined
24.42   \ifx\em\@undefined
24.43     \let\emph\relax
24.44   \else
24.45     \def\emph#1{\em #1}
24.46   \fi
24.47 \fi

```

24.3 Captionnames and date

The next step consists of defining the French equivalents for the L^AT_EX captionnames.

`\captionsfrenchb` The macro `\captionsfrenchb` defines all strings used in the four standard document classes provided with L^AT_EX. Some authors do not like some of these names; it is easy to change them in the preamble *after* loading `frenchb` (or in your file `frenchb.cfg`), e.g. `\addto\captionsfrenchb{\def\figurename{Figure}}` will print ‘Figure’ in roman instead of ‘FIG.’.

```

24.48 \ifx\fmtname\PlainFmtName
24.49 \else
24.50 \@namedef{captions\CurrentOption}{%
24.51   \def\refname{R\ref'ereferences}%
24.52   \def\abstractname{R'esum'e}%
24.53   \def\bibname{Bibliographie}%
24.54   \def\prefacename{Pr'eface}%
24.55   \def\chaptername{Chapitre}%
24.56   \def\appendixname{Annexe}%
24.57   \def\contentsname{Table des mati'eres}%
24.58   \def\listfigurename{Table des figures}%
24.59   \def\listtablename{Liste des tableaux}%
24.60   \def\indexname{Index}%
24.61   \def\figurename{{\scshape Fig.}}%
24.62   \def\tablename{{\scshape Tab.}}%

```

“Première partie” instead of “Part I”

```

24.63   \def\partname{\protect\Fpt partie}%
24.64   \def\Fpt{{\ifcase\value{part}\or Premi'ere\or Deuxi'eme\or
24.65   Troisi'eme\or Quatri'eme\or Cinqui'eme\or Sixi'eme\or
24.66   Septi'eme\or Huiti'eme\or Neuvi'eme\or Dixi'eme\or Onzi'eme\or
24.67   Douzi'eme\or Treizi'eme\or Quatorzi'eme\or Quinzi'eme\or
24.68   Seizi'eme\or Dix-septi'eme\or Dix-huiti'eme\or Dix-neuvi'eme\or

```

```

24.69 \Vingt\‘eme\fi}\space\def\thepart{}}}%
24.70 \def\pagename{page}%
24.71 \def\seename{{\emph{voir}}}%
24.72 \def\alsoname{{\emph{voir aussi}}}%
24.73 \def\enclname{P.~J. }%
24.74 \def\ccname{Copie \‘a }%
24.75 \def\headtoname{}}%
24.76 \def\proofname{D\‘emonstration}% for AMS- $\LaTeX$ 
24.77 }
24.78 \def\captionsfrench{\captionsfrenchb}
24.79 \def\captionsfrançais{\captionsfrenchb}
24.80 \fi

```

`\datefrenchb` The macro `\datefrenchb` redefines the command `\today` to produce French dates.

```

24.81 \@namedef{date\CurrentOption}{%
24.82   \def\today{\number\day
24.83     \ifnum1=\day \ier\fi
24.84     \space \ifcase\month
24.85       \or janvier\or f\‘evrier\or mars\or avril\or mai\or juin\or
24.86       juillet\or ao\‘ut\or septembre\or octobre\or novembre\or
24.87       d\‘ecembre\fi
24.88     \space \number\year}}
24.89 \def\datefrench{\datefrenchb}
24.90 \def\datefrançais{\datefrenchb}

```

24.4 Punctuation

The ‘double punctuation’ characters (; ! ? and :) have to be made `\active` for an automatic control of the amount of space to insert before them.

```

24.91 \initiate@active@char{:}
24.92 \initiate@active@char{;}
24.93 \initiate@active@char{!}
24.94 \initiate@active@char{?}

```

We specify that the French group of shorthands should be used.

```

24.95 \expandafter\addto\csname extras\CurrentOption\endcsname{%
24.96   \languageshorthands{frenchb}}

```

These characters are ‘turned on’ once, later their definition may vary.

```

24.97 \expandafter\addto\csname extras\CurrentOption\endcsname{%
24.98   \bbl@activate{:}\bbl@activate{;}%
24.99   \bbl@activate{!}\bbl@activate{?}}
24.100 \expandafter\addto\csname noextras\CurrentOption\endcsname{%
24.101   \bbl@deactivate{:}\bbl@deactivate{;}%
24.102   \bbl@deactivate{!}\bbl@deactivate{?}}

```

One more thing `\extrasfrenchb` needs to do is to make sure that `\frenchspacing` is in effect. If this is not the case the execution of `\noextrasfrenchb` will switch it off again.

```

24.103 \expandafter\addto\csname extras\CurrentOption\endcsname{%
24.104   \bbl@frenchspacing}
24.105 \expandafter\addto\csname noextras\CurrentOption\endcsname{%
24.106   \bbl@nonfrenchspacing}

```

`\frenchb@sh@;` We have to tune the amount of white space before `;` `!` `?` and `:`. This should only happen in horizontal mode, hence the test `\ifhmode`. In horizontal mode, if a space has been typed before `;` we remove it and put an unbreakable `\thinspace` instead. If no space has been typed, we add `\FDP@thinspace` which will be defined, up to the user's wishes, as an automatic added `\thinspace`, or as `\@empty`.

```
24.107 \declare@shorthand{\frenchb}{;}{\%
24.108   \ifhmode
24.109     \ifdim\lastskip>\z@
24.110       \unskip\penalty\@M\thinspace
24.111     \else
24.112       \FDP@thinspace
24.113     \fi
24.114   \fi
```

Now we can insert a `;` character.

```
24.115   \string;}
```

`\frenchb@sh@!` Because these definitions are very similar only one is displayed in a way that the
`\frenchb@sh@?` definition can be easily checked.

```
24.116 \declare@shorthand{\frenchb}{!}{\%
24.117   \ifhmode
24.118     \ifdim\lastskip>\z@
24.119       \unskip\penalty\@M\thinspace
24.120     \else
24.121       \FDP@thinspace
24.122     \fi
24.123   \fi
24.124   \string!}

24.125 \declare@shorthand{\frenchb}{?}{\%
24.126   \ifhmode
24.127     \ifdim\lastskip>\z@
24.128       \unskip\penalty\@M\thinspace
24.129     \else
24.130       \FDP@thinspace
24.131     \fi
24.132   \fi
24.133   \string?}
```

`\frenchb@sh@:` The `:` requires a normal space before it, instead of a `\thinspace`.

```
24.134 \declare@shorthand{\frenchb}{:}{\%
24.135   \ifhmode
24.136     \ifdim\lastskip>\z@
24.137       \unskip\penalty\@M\
24.138     \else
24.139       \FDP@space
24.140     \fi
24.141   \fi
24.142   \string:}
```

`\AutoSpaceBeforeFDP` `\FDP@thinspace` and `\FDP@space` are defined as unbreakable spaces by
`\NoAutoSpaceBeforeFDP` `\AutoSpaceBeforeFDP` or as `\@empty` by `\NoAutoSpaceBeforeFDP`.
Default is `\AutoSpaceBeforeFDP`.

```

24.143 \def\AutoSpaceBeforeFDP{%
24.144     \def\FDP@thinspace{\penalty\@M\thinspace}%
24.145     \def\FDP@space{\penalty\@M\ }%
24.146 \def\NoAutoSpaceBeforeFDP{\let\FDP@thinspace\empty
24.147     \let\FDP@space\empty}%
24.148 \AutoSpaceBeforeFDP

```

`\system@sh@:` When the active characters appear in an environment where their French behaviour is not wanted they should give an ‘expected’ result. Therefore we define `\system@sh@?` shorthands at system level as well.

```

\system@sh@;
24.149 \declare@shorthand{system}{:}{\string:}
24.150 \declare@shorthand{system}{!}{\string!}
24.151 \declare@shorthand{system}{?}{\string?}
24.152 \declare@shorthand{system}{;}{\string;}

```

24.5 French quotation marks

Several shapes of French quotation marks are provided for use with CMR or EC/DC fonts, or PostScript fonts. CMR fonts have no quotation marks built-in, so we have to emulate them using math symbols, either L^AT_EX’s ‘lasy’ font if available, or T_EX symbols `\ll` and `\gg` otherwise. EC/DC fonts and PostScript fonts have built-in quotation marks, so we will of course use them.

The following definitions will take care for correct spacing of French quotation marks (a white space precedes and follows quotation marks but no line break is allowed neither *after* the opening one, nor *before* the closing one).

`\oPlainGuill` For *PlainT_EX*, we define `\oPlainGuill` and `\fPlainGuill` using math symbols `\ll` and `\gg`. In order to have the word following opening guillemets hyphenated properly we have to end the definitions with the T_EX equivalent for `\allowhyphens` which is `\penalty\@M\hskip\z@skip`.

```

24.153 \def\oPlainGuill{\leavevmode\raise0.25ex%
24.154     \hbox{$\scriptscriptstyle\ll$\kern 0.15em}%
24.155     \penalty\@M\hskip\z@skip}
24.156 \def\fPlainGuill{\ifdim\lastskip>\z@\unskip\penalty\@M\fi
24.157     \leavevmode\raise0.25ex%
24.158     \hbox{\kern 0.15em$\scriptscriptstyle\gg$}}

```

`\oLasyGuill` In L^AT_EX 2_ε better looking quotation marks are available via the ‘lasy’ font (‘lasy’ stands for L^AT_EX Symbol).

```

24.159 \ifx\fmtname\LaTeXeFmtName
24.160     \def\oLasyGuill{\leavevmode
24.161         \hbox{\fontencoding{U}\fontfamily{lasy}\selectfont
24.162             (\kern-0.20em(\kern 0.20em)\allowhyphens}
24.163     \def\fLasyGuill{\ifdim\lastskip>\z@\unskip\penalty\@M\fi\leavevmode
24.164         \hbox{\kern0.20em%
24.165             \fontencoding{U}\fontfamily{lasy}\selectfont
24.166             )\kern-0.20em)}}
24.167 \fi

```

`\oECGuill` Now let’s define French quotation marks for T1 encoding.

```

\fECGuill
24.168 \def\oECGuill{\leavevmode\hbox{\guillemotleft\kern 0.15em}%
24.169     \allowhyphens}

```

```

24.170 \def\fECGuill{\ifdim\lastskip>\z@ \unskip \penalty \@M \fi
24.171          \leavevmode \hbox{\kern 0.15em \guillemotright}}

\og Now let's define which kind of French quotation marks will be used. The top
\fg macros for quotation marks will be called \og (“ouvrez guillemets”) and \fg
\bb1@frenchguillemets (“fermez guillemets”).
\bb1@nonfrenchguillemets Make the top level definitions for French quotation marks available through the
\extrasfrenchb \noextrasfrenchb mechanism.
As \DeclareTextCommand cannot be used after the \begin{document} we intro-
duce internal definitions \begin@guill and \end@guill.
We'll try to be smart to users of D. CARLISLE's xspace package: if this package
is loaded there will be no need for {} or \ to get a space after \fg.
In LATEX 2ε we provide a dummy definition for \og and \fg, just to display an
error message in case \og or \fg have been defined elsewhere.

24.172 \ifx\fmtname\LaTeXeFmtName
24.173   \newcommand{\og}{\@empty}
24.174   \newcommand{\fg}{\@empty}
24.175   \DeclareTextCommand{\begin@guill}{T1}{\oECGuill}
24.176   \DeclareTextCommand{\end@guill}{T1}{\fECGuill}
24.177   \DeclareTextCommand{\begin@guill}{OT1}{\oLasyGuill}
24.178   \DeclareTextCommand{\end@guill}{OT1}{\fLasyGuill}
24.179   \DeclareTextSymbolDefault{\begin@guill}{OT1}
24.180   \DeclareTextSymbolDefault{\end@guill}{OT1}
24.181 \else
24.182   \let\begin@guill\oPlainGuill
24.183   \let\end@guill\fPlainGuill
24.184 \fi
24.185 \def\bb1@frenchguillemets{\ifx\xspace\@undefined \let\xspace\relax \fi
24.186           \def\og{\begin@guill}%
24.187           \def\fg{\end@guill\xspace}}
24.188 \def\bb1@nonfrenchguillemets{\def\og{‘}%
24.189           \def\fg{\ifdim\lastskip>\z@ \unskip \fi ’}}
24.190 \expandafter\addto\csname extras\CurrentOption\endcsname{%
24.191   \bb1@frenchguillemets}
24.192 \expandafter\addto\csname noextras\CurrentOption\endcsname{%
24.193   \bb1@nonfrenchguillemets}

```

24.6 French lists

\bb1@frenchitems French lists are different from USenglish ones: the • is never used (long dash ‘—’
\bb1@nonfrenchitems is preferred for all levels), and vertical spacing between items, before and after
\bb1@frenchlistspacing the list, should be shorter in French texts than the defaults provided by L^AT_EX.
\bb1@nonfrenchlistspacing Note that the easy way, just changing values of vertical spacing parameters when
entering French and restoring them to their defaults on exit would not work, so
we have to redefine \@trivlist.

The amount of vertical space before and after a list is given by \topsep +
\parskip (+ \partopsep if the list starts a new paragraph). IMHO, \parskip
should be added *only* when the list starts a new paragraph, so I subtract \parskip
from \topsep and add it back to \partopsep; this will normally make no difference
because \parskip's default value is 0pt, but will be noticeable when \parskip is
not null.

I would appreciate feedback from experts in French typography, about the (some-what experimental) values set here for `\partopsep`, `\topsep`, `\itemsep` and `\parsep`.

Of course, this code is only for L^AT_EX.

```

24.194 \ifx\fmtname\PlainFmtName
24.195 \else
24.196   \let\@ltiORI\labelitemi
24.197   \let\@ltiiORI\labelitemii
24.198   \let\@ltiiiORI\labelitemiii
24.199   \let\@ltivORI\labelitemiv
24.200   \def\bbl@frenchitems{%
24.201     \def\labelitemi{--}%
24.202     \def\labelitemii{--}%
24.203     \def\labelitemiii{--}%
24.204     \def\labelitemiv{--}}
24.205   \def\bbl@nonfrenchitems{%
24.206     \let\labelitemi\@ltiORI
24.207     \let\labelitemii\@ltiiORI
24.208     \let\labelitemiii\@ltiiiORI
24.209     \let\labelitemiv\@ltivORI}
24.210   \expandafter\addto\csname extras\CurrentOption\endcsname{%
24.211     \bbl@frenchitems}
24.212   \expandafter\addto\csname noextras\CurrentOption\endcsname{%
24.213     \bbl@nonfrenchitems}
24.214   \let\@trivlistORI\@trivlist
24.215   \def\bbl@frenchlistspacing{%
24.216     \def\@trivlist{\setlength{\itemsep}{0.4ex plus 0.2ex minus 0.2ex}%
24.217       \setlength{\parsep}{0.4ex plus 0.2ex minus 0.2ex}%
24.218       \setlength{\topsep}{0.8ex plus 0.4ex minus 0.4ex}%
24.219       \setlength{\partopsep}{0.4ex plus 0.2ex minus 0.2ex}%
24.220       \addtolength{\topsep}{-\parskip}%
24.221       \addtolength{\partopsep}{-\parskip}%
24.222       \@trivlistORI}}
24.223   \def\bbl@nonfrenchlistspacing{\let\@trivlist\@trivlistORI}
24.224   \expandafter\addto\csname extras\CurrentOption\endcsname{%
24.225     \bbl@frenchlistspacing}
24.226   \expandafter\addto\csname noextras\CurrentOption\endcsname{%
24.227     \bbl@nonfrenchlistspacing}
24.228 \fi

```

24.7 French indentation of sections

`\bbl@frenchindent` In French the first paragraph of each section should be indented, this is another difference with USenglish. Add this code only in L^AT_EX.

```

24.229 \ifx\fmtname\PlainFmtName
24.230 \else
24.231   \let\@aifORI\@afterindentfalse
24.232   \def\bbl@frenchindent{\let\@afterindentfalse\@afterindenttrue
24.233     \@afterindenttrue}
24.234   \def\bbl@nonfrenchindent{\let\@afterindentfalse\@aifORI
24.235     \@afterindentfalse}
24.236   \expandafter\addto\csname extras\CurrentOption\endcsname{%
24.237     \bbl@frenchindent}

```

```

24.238 \expandafter\addto\csname noextras\CurrentOption\endcsname{%
24.239 \bbl@nonfrenchindent}
24.240 \fi

```

24.8 Formatting numbers

In English the decimal part starts with a point and thousands should be separated by a comma: an approximation of 1000π should be inputed as `$3{,}141.592{,}653$` in math-mode and as `3,141.592,653` in text. In French the decimal part starts with a comma and thousands should be separated by a space; the same approximation of 1000π should be inputed as `$3\;141{,}592\;653$` in math-mode and as something like `3~141,592~653` in text. Remember braces are mandatory around the comma in math-mode, the reason is mentioned in the T_EXbook p. 134: the comma is of type `\mathpunct` (thus normally followed by a space) while the point is of type `\mathord` (no space added).

Thierry Bouche suggested that a second type of comma, of type `\mathord` would be useful in math-mode, and proposed to introduce a command (named `\decimalsep` in this package), the expansion of which would depend on the current language.

Vincent Jalby suggested a command `\nombre` to conveniently typeset numbers: inputting `\nombre{3141,592653}` either in text or in math-mode will format this number properly according to the current language (French or non-French).

`\nombre` accepts an optional argument which happens to be useful with the extension ‘dcolumn’, it specifies the decimal separator used in the *source code*:

```

\newcolumntype{d}{D{,}\decimalsep}{-1}}
\begin{tabular}{d}\hline
  3,14 \\\
  \nombre[,]{123,4567} \\\
  \nombre[,]{9876,543}\hline
\end{tabular}

```

will print a column of numbers aligned on the decimal point (comma or point depending on the current language), each slice of 3 digits being separated by a space or a comma according to the current language.

`\decimalsep` We need a internal definition, valid in both text and math-mode, for the comma
`\thousandsep` (`\@comma@`) and another one for the unbreakable fixed length space (no glue) used in French (`\f@thousandsep`).

The commands `\decimalsep` and `\thousandsep` get default definitions (for the English language) when `frenchb` is loaded; these definitions will be updated when the current language is switched to or from French.

```

24.241 \mathchardef\m@comma="013B
24.242 \def\@comma@{\ifmmode\m@comma\else,\fi}
24.243 \def\f@thousandsep{\ifmmode\mskip5.5mu\else\penalty\@M\kern.3em\fi}
24.244 \newcommand{\decimalsep}{.}
24.245 \newcommand{\thousandsep}{\@comma@}
24.246 \expandafter\addto\csname extras\CurrentOption\endcsname{%
24.247 \def\decimalsep{\@comma@}%
24.248 \def\thousandsep{\f@thousandsep}}
24.249 \expandafter\addto\csname noextras\CurrentOption\endcsname{%
24.250 \def\decimalsep{.}%
24.251 \def\thousandsep{\@comma@}}

```

`\nombre` The decimal separator used when *inputing* a number with `\nombre` has to be a comma. `\nombre` splits the inputed number into two parts: what comes before the first comma will be formatted by `\@integerpart` while the rest (if not empty) will be formatted by `\@decimalpart`. Both parts, once formatted separately will be merged together with between them, either the decimal separator `\decimalsep` or (in L^AT_EX 2_ε only) the optional argument of `\nombre`.

```

24.252 \if@Two@E
24.253   \newcommand{\nombre}[2][\decimalsep]{%
24.254       \def\@decimalsep{#1}\@nombre#2\@empty,\@empty,\@nil}
24.255 \else
24.256   \newcommand{\nombre}[1]{%
24.257       \def\@decimalsep{\decimalsep}\@nombre#1\@empty,\@empty,\@nil}
24.258 \fi
24.259 \def\@nombre#1,#2,#3\@nil{%
24.260     \ifx\@empty#2%
24.261         \@integerpart{#1}%
24.262     \else
24.263         \@integerpart{#1}\@decimalsep\@decimalpart{#2}%
24.264     \fi}

```

The easiest bit is the decimal part: We attempt to read the first four digits of the decimal part, if it has less than 4 digits, we just have to print them, otherwise `\thousandsep` has to be appended after the third digit, and the algorithm is applied recursively to the rest of the decimal part.

```

24.265 \def\@decimalpart#1{\@@decimalpart#1\@empty\@empty\@empty}
24.266 \def\@@decimalpart#1#2#3#4{#1#2#3%
24.267     \ifx\@empty#4%
24.268     \else
24.269         \thousandsep\expandafter\@@decimalpart\expandafter#4%
24.270     \fi}

```

Formatting the integer part is more difficult because the slices of 3 digits start from the *bottom* while the number is read from the top! This (tricky) code is borrowed from David Carlisle's comma.sty.

```

24.271 \def\@integerpart#1{\@@integerpart{#1}\@empty\@empty\@empty}
24.272 \def\@@integerpart#1#2#3#4{%
24.273     \ifx\@empty#2%
24.274         \@addthousandsep#1\relax
24.275     \else
24.276         \ifx\@empty#3%
24.277             \@addthousandsep\@empty\@empty#1#2\relax
24.278         \else
24.279             \ifx\@empty#4%
24.280                 \@addthousandsep\@empty#1#2#3\relax
24.281             \else
24.282                 \@integerpartafterfi{#1#2#3#4}%
24.283             \fi
24.284         \fi
24.285     \fi}
24.286 \def\@integerpartafterfi#1\fi\fi\fi{\fi\fi\fi\@integerpart{#1}}
24.287 \def\@addthousandsep#1#2#3#4{#1#2#3%
24.288     \if#4\relax
24.289     \else

```



```

24.290 \thousandsep\expandafter\@addthousandsep\expandafter#4%
24.291 \fi}

```

24.9 Extra utilities

All that is left to do now is to provide the French user with some extra utilities.

`\up` `\up` eases the typesetting of superscripts like ‘1^{er}’. `\up` relies on `\textsuperscript` when available (i. e., in L^AT_EX 2_ε).

`\up@size` The internal macro `\up@size` holds the size at which the superscript will be typeset. The reason for this is that we have to specify it differently for different formats.

```

24.292 \ifx\sevenrm\@undefined
24.293 \ifx\@ptsize\@undefined
24.294 \let\up@size\small
24.295 \else
24.296 \ifx\selectfont\@undefined

```

In this case the format is the original L^AT_EX-2.09:

```

24.297 \ifcase\@ptsize
24.298 \let\up@size\ixpt\or
24.299 \let\up@size\xpt\or
24.300 \let\up@size\xipt
24.301 \fi

```

When `\selectfont` is defined we probably have NFSS available:

```

24.302 \else
24.303 \ifcase\@ptsize
24.304 \def\up@size{\fontsize\@ixpt{10pt}\selectfont}\or
24.305 \def\up@size{\fontsize\@xpt{11pt}\selectfont}\or
24.306 \def\up@size{\fontsize\@xipt{12pt}\selectfont}
24.307 \fi
24.308 \fi
24.309 \fi
24.310 \else

```

If we end up here it must be a plain based T_EX format, so:

```

24.311 \let\up@size\sevenrm
24.312 \fi

```

Now we can define `\up`. When L^AT_EX 2_ε runs in compatibility mode (L^AT_EX-2.09 emulation), `\textsuperscript` is also defined, but does no good job, so we give two different definitions for `\up` using `\if@Two@E`.

```

24.313 \if@Two@E
24.314 \DeclareRobustCommand*\up}[1]{\textsuperscript{#1}}
24.315 \else
24.316 \DeclareRobustCommand*\up}[1]{\leavevmode\raiselex\hbox{\up@size#1}}
24.317 \fi

```

`\ieme` is provided for compatibility with `francais.sty`, the other 5 for compatibility with `french.sty`:

```

24.318 \def\ieme{\up{\lowercase{e}}}
24.319 \def\iemes{\up{\lowercase{es}}}
24.320 \def\ier{\up{\lowercase{er}}}

```

```

24.321 \def\iers{\up{\lowercase{ers}}}
24.322 \def\iere{\up{\lowercase{re}}}
24.323 \def\ieres{\up{\lowercase{res}}}

\No And some more macros for numbering, first two support macros.
\No
24.324 \DeclareRobustCommand*\FrenchEnumerate[1]{%
\prim24.325 #1\up{\lowercase{o}}\kern+.3em}
\frimo24.326 \DeclareRobustCommand*\FrenchPopularEnumerate[1]{%
24.327 #1\up{\lowercase{o}}\kern+.3em}

Typing \primo should result in ‘1°’,
24.328 \def\primo{\FrenchEnumerate1}
24.329 \def\secundo{\FrenchEnumerate2}
24.330 \def\tertio{\FrenchEnumerate3}
24.331 \def\quatro{\FrenchEnumerate4}

while typing \frimo gives ‘1°’.
24.332 \def\frimo{\FrenchPopularEnumerate1}
24.333 \def\secundo{\FrenchPopularEnumerate2}
24.334 \def\tertio{\FrenchPopularEnumerate3}
24.335 \def\quatro{\FrenchPopularEnumerate4}

Let’s provide two macros for the common abbreviations of “Numéro”.
24.336 \DeclareRobustCommand*\No{N\up{\lowercase{o}}\kern+.2em}
24.337 \DeclareRobustCommand*\no{n\up{\lowercase{o}}\kern+.2em}

\bsc As family names should be written in small capitals and never be hyphen-
ated, we provide a command (its name comes from Boxed Small Caps) to input
them easily; this is a simpler implementation of commands \fsc and \lsc from
french.sty : no automatic uppercase/lowercase conversion is performed. Usage:
Jean~\bsc{Duchemin}.
24.338 \DeclareRobustCommand*\bsc[1]{\leavevmode\hbox{\scshape #1}}

Some definitions for special characters. The first eight are mandatory for \oe
etc. to work properly in moving arguments, the others just for convenience. We
won’t define \tilde as a Text Symbol not to conflict with the macro \tilde for
math-mode and use the name \tild instead. Note that \boi may not be used
in math-mode, its name in math-mode is \backslash. \degre needs a special
treatment: it is \char6 in T1-encoding and \char23 in OT1-encoding.
24.339 \ifx\fmtname\LaTeXeFmtName
24.340 \DeclareTextSymbol{\ae}{T1}{230}
24.341 \DeclareTextSymbol{\ae}{OT1}{26}
24.342 \DeclareTextSymbol{\oe}{T1}{247}
24.343 \DeclareTextSymbol{\oe}{OT1}{27}
24.344 \DeclareTextSymbol{\AE}{T1}{198}
24.345 \DeclareTextSymbol{\AE}{OT1}{29}
24.346 \DeclareTextSymbol{\OE}{T1}{215}
24.347 \DeclareTextSymbol{\OE}{OT1}{30}
24.348 \DeclareTextSymbol{\degre}{T1}{6}
24.349 \DeclareTextSymbol{\degre}{OT1}{23}
24.350 \DeclareTextSymbol{\boi}{T1}{92}
24.351 \DeclareTextCommand{\boi}{OT1}{\backslash$}
24.352 \DeclareTextSymbol{\at}{T1}{64}

```

```

24.353 \DeclareTextSymbol{\at}{OT1}{64}
24.354 \DeclareTextSymbol{\circonflexe}{T1}{94}
24.355 \DeclareTextSymbol{\circonflexe}{OT1}{94}
24.356 \DeclareTextSymbol{\tild}{T1}{126}
24.357 \DeclareTextSymbol{\tild}{OT1}{126}
24.358 \else
24.359 \def\T@one{T1}
24.360 \ifx\F@encoding\T@one
24.361 \newcommand{\degree}{\char6}
24.362 \else
24.363 \newcommand{\degree}{\char23}
24.364 \fi
24.365 \newcommand{\at}{\char64}
24.366 \newcommand{\circonflexe}{\char94}
24.367 \newcommand{\tild}{\char126}
24.368 \newcommand{\boi}{\{$\backslash$\}}
24.369 \fi

```

\degrees Macro for typesetting the abbreviation for ‘degrees’ (as in ‘degrees Celsius’). As the bounding box of the character ‘degree’ has *very* different widths in CMR/DC and PostScript fonts, we fix the width of the bounding box of `\degrees` to 0.3em, this lets the symbol ‘degree’ stick to the preceding (e.g., `45\degrees`) or following character (e.g., `20~\degrees C`).

```

24.370 \DeclareRobustCommand*{\degrees}{%
24.371 \leavevmode\hbox to 0.3em{\hss\degree\hss}}

```

The following macros are used in the redefinition of `\^` and `\"` to handle the letter i: they allow users to type simply `\^i` and `\"i` instead of `\^{i}` and `\{i}`. \LaTeX ’s macros dealing with accents conflict with those of $\text{\LaTeX} 2_{\epsilon}$, so we check whether `\csubinverse` is defined or not. If `\csubinverse` is *defined*, we are in \LaTeX .

```

24.372 \ifx\fmtname\LaTeXeFmtName
24.373 \AtBeginDocument{%
24.374 \ifx\csubinverse\@undefined
24.375 \DeclareTextCompositeCommand{\^}{OT1}{i}{\^{i}}%
24.376 \DeclareTextCompositeCommand{\"}{OT1}{i}{\{i}}%
24.377 \fi}
24.378 \fi

```

Finally the macrospace used by some control sequences we do not need any longer, is freed.

```

24.379 \let\T@one\relax
24.380 \let\@FI@\relax
24.381 \let\PlainFmtName\relax
24.382 \let\LaTeXeFmtName\relax

```

The macro `\ldf@finish` takes care of looking for a configuration file, setting the main language to be switched on at `\begin{document}` and resetting the category code of `@` to its original value. The config file searched for will always be ‘frenchb.cfg’. Remember that `\CurrentOption` has been set to ‘frenchb’, and that ‘francais’ and ‘french’ are aliases for ‘frenchb’.

```

24.383 \ldf@finish\CurrentOption
24.384 \code

```

25 The Italian language

The file `italian.dtx`²³ It defines all the language-specific macros for the Italian language.

For this language the `\clubpenalty`, `\widowpenalty` and `\finalhyphendemerits` are set to rather high values.

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the `@` sign, etc.

25.1 `(*code)`

25.2 `\LdfInit{italian}{captionsitalian}`

When this file is read as an option, i.e. by the `\usepackage` command, `italian` will be an ‘unknown’ language in which case we have to make it known. So we check for the existence of `\l@italian` to see whether we have to do something here.

25.3 `\ifx\l@italian\@undefined`

25.4 `\@nopatterns{Italian}`

25.5 `\adddialect\l@italian0\fi`

The next step consists of defining commands to switch to (and from) the Italian language.

`\captionsitalian` The macro `\captionsitalian` defines all strings used in the four standard documentclasses provided with L^AT_EX.

25.6 `\addto\captionsitalian{%`

25.7 `\def\prefacename{Prefazione}%`

25.8 `\def\refname{Riferimenti bibliografici}%`

25.9 `\def\abstractname{Sommario}%`

25.10 `\def\bibName{Bibliografia}%`

25.11 `\def\chaptername{Capitolo}%`

25.12 `\def\appendixname{Appendice}%`

25.13 `\def\contentsname{Indice}%`

25.14 `\def\listfigurename{Elenco delle figure}%`

25.15 `\def\listtablename{Elenco delle tabelle}%`

25.16 `\def\indexname{Indice analitico}%`

25.17 `\def\figurename{Figura}%`

25.18 `\def\tablename{Tabella}%`

25.19 `\def\partname{Parte}%`

25.20 `\def\enclname{Allegati}%`

25.21 `\def\ccname{e~p.~c.}%`

25.22 `\def\headtoname{Per}%`

25.23 `\def\pagename{Pag.}% % in Italian abbreviation is preferred`

25.24 `\def\seename{vedi}%`

25.25 `\def\alsoname{vedi anche}%`

25.26 `\def\proofname{Dimostrazione}%`

25.27 `}`

`\dateitalian` The macro `\dateitalian` redefines the command `\today` to produce Italian dates.

25.28 `\def\dateitalian{%`

25.29 `\def\today{\number\day~\ifcase\month\or`

²³The file described in this section has version number v1.2j and was last revised on 1996/12/29. The original author is Maurizio Codogno, (mau@beatles.cselt.stet.it).

```

25.30 gennaio\or febbraio\or marzo\or aprile\or maggio\or giugno\or
25.31 luglio\or agosto\or settembre\or ottobre\or novembre\or dicembre\fi
25.32 \space \number\year}}

```

`\italianhyphenmins` The italian hyphenation patterns can be used with both `\lefthyphenmin` and `\righthyphenmin` set to 2.

```

25.33 \def\italianhyphenmins{\tw@\tw@}

```

`\extrasitalian` Lower the chance that clubs or widows occur.

```

\noextrasitalian25.34 \addto\extrasitalian{%
25.35 \babel@savevariable\clubpenalty
25.36 \babel@savevariable\widowpenalty
25.37 \clubpenalty3000\widowpenalty3000}

```

Never ever break a word between the last two lines of a paragraph in italian texts.

```

25.38 \addto\extrasitalian{%
25.39 \babel@savevariable\finalhyphendemerits
25.40 \finalhyphendemerits50000000}

```

In order to enable the hyphenation of words such as “begl’italiani” we give the ’ a non-zero lower case code. When we do that T_EX finds the following hyphenation points be-gl’i-ta-lia-ni instead of none.

```

25.41 \addto\extrasitalian{%
25.42 \lccode‘’=‘’}
25.43 \addto\noextrasitalian{%
25.44 \lccode‘’=0}

```

The macro `\ldf@finish` takes care of looking for a configuration file, setting the main language to be switched on at `\begin{document}` and resetting the category code of `@` to its original value.

```

25.45 \ldf@finish{italian}
25.46 \code

```

26 The Portuguese language

The file `portuges.dtx`²⁴ defines all the language-specific macros for the Portuguese language as well as for the Brazilian version of this language.

For this language the character " is made active. In table 6 an overview is given of its purpose.

"	disable ligature at this position.
"-	an explicit hyphen sign, allowing hyphenation in the rest of the word.
""	like "-", but producing no hyphen sign (for words that should break at some sign such as "entrada/salida."
"<	for French left double quotes (similar to <<).
">	for French right double quotes (similar to >>).
\-	like the old \-, but allowing hyphenation in the rest of the word.

Table 6: The extra definitions made by `portuges.ldf`

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the `@` sign, etc.

26.1 `{*code}`

26.2 `\LdfInit\CurrentOption{captions\CurrentOption}`

When this file is read as an option, i.e. by the `\usepackage` command, `portuges` will be an 'unknown' language in which case we have to make it known. So we check for the existence of `\l@portuges` to see whether we have to do something here. Since it is possible to load this file with any of the following four options to babel: `portuges`, `portuguese`, `brazil` and `brazilian` we also allow that the hyphenation patterns are loaded under any of these four names. We just have to find out which one was used.

```

26.3 \ifx\l@portuges\@undefined
26.4   \ifx\l@portuguese\@undefined
26.5     \ifx\l@brazil\@undefined
26.6       \ifx\l@brazilian\@undefined
26.7         \@nopatterns{Portuguese}
26.8         \adddialect\l@portuges0
26.9       \else
26.10        \let\l@portuges\l@brazilian
26.11      \fi
26.12    \else
26.13      \let\l@portuges\l@brazil
26.14    \fi
26.15  \else
26.16    \let\l@portuges\portuguese
26.17  \fi
26.18 \fi

```

By now `\l@portuges` is defined. When the language definition file was loaded under a different name we make sure that the hyphenation patterns can be found.

²⁴The file described in this section has version number v1.2j and was last revised on 1996/12/23. Contributions were made by Jose Pedro Ramalheite (JRAMALHE@CERNVM or Jose-Pedro_Ramalheite@MACMAIL) and Arnaldo Viegas de Lima arnaldo@VNET.IBM.COM.

```

26.19 \expandafter\ifx\csname l@\CurrentOption\endcsname\relax
26.20 \expandafter\let\csname l@\CurrentOption\endcsname\l@portuges
26.21 \fi

```

Now we have to decide whether this language definition file was loaded for Portuguese or Brazilian use. This can be done by checking the contents of `\CurrentOption`. When it doesn't contain either 'portuges' or 'portuguese' we make `\bbl@tempa` empty.

```

26.22 \def\bbl@tempa{portuges}
26.23 \ifx\CurrentOption\bbl@tempa
26.24 \let\bbl@tempb\@empty
26.25 \else
26.26 \def\bbl@tempa{portuguese}
26.27 \ifx\CurrentOption\bbl@tempa
26.28 \let\bbl@tempb\@empty
26.29 \else
26.30 \def\bbl@tempb{brazil}
26.31 \fi
26.32 \fi
26.33 \ifx\bbl@tempb\@empty

```

The next step consists of defining commands to switch to (and from) the Portuguese language.

`\captionsportuges` The macro `\captionsportuges` defines all strings used in the four standard documentclasses provided with L^AT_EX.

```

26.34 \@namedef{captions\CurrentOption}{%
26.35 \def\prefacename{Pref\'acio}%
26.36 \def\refname{Refer\'encias}%
26.37 \def\abstractname{Resumo}%
26.38 \def\bibname{Bibliografia}%
26.39 \def\chaptername{Cap\'itulo}%
26.40 \def\appendixname{Ap\'endice}%

```

Some discussion took place around the correct translations for 'Table of Contents' and 'Index'. the translations differ for Portuguese and Brazilian based the following history:

The whole issue is that some books without a real index at the end misused the term 'Índice' as table of contents. Then, what happens is that some books appeared with 'Índice' at the beginning and a 'Índice Remissivo' at the end. Remissivo is a redundant word in this case, but was introduced to make up the difference. So in Brasil people started using 'Sumário' and 'Índice Remissivo'. In Portugal this seems not to be very common, therefore we chose 'Índice' instead of 'Índice Remissivo'.

```

26.41 \def\contentsname{Conte\'udo}%
26.42 \def\listfigurename{Lista de Figuras}%
26.43 \def\listtablename{Lista de Tabelas}%
26.44 \def\indexname{\'}Índice}%
26.45 \def\figurename{Figura}%
26.46 \def\tablename{Tabela}%
26.47 \def\partname{Parte}%
26.48 \def\enclname{Anexo}%

```

```

26.49 \def\ccname{Com c\’opia a}%
26.50 \def\headtoname{Para}%
26.51 \def\pagename{P\’agina}%
26.52 \def\seename{ver}%
26.53 \def\alsoname{ver tamb\’em}%

```

An alternate term for ‘Proof’ could be ‘Prova’.

```

26.54 \def\proofname{Demonstra\c{c}\~ao}%
26.55 }

```

`\dateportuges` The macro `\dateportuges` redefines the command `\today` to produce Portuguese dates.

```

26.56 \@namedef{date\CurrentOption}{%
26.57 \def\today{\number\day\space de\space\ifcase\month\or
26.58 Janeiro\or Fevereiro\or Mar\c{c}o\or Abril\or Maio\or Junho\or
26.59 Julho\or Agosto\or Setembro\or Outubro\or Novembro\or Dezembro
26.60 \fi
26.61 \space de\space\number\year}}
26.62 \else

```

For the Brazilian version of these definitions we just add a “dialect”.

```

26.63 \expandafter\adddialect\csname l@CurrentOption\endcsname\l@portuges

```

`\captionsbrazil` The “captions” are different for both versions of the language, so we define the macro `\captionsbrazil` here.

```

26.64 \@namedef{captions\CurrentOption}{%
26.65 \def\prefacename{Pref\’acio}%
26.66 \def\refname{Refer\^encias}%
26.67 \def\abstractname{Resumo}%
26.68 \def\bibname{Refer\^encias Bibliogr\’aficas}%
26.69 \def\chaptername{Cap\’itulo}%
26.70 \def\appendixname{Ap\^endice}%
26.71 \def\contentsname{Sum\’ario}%
26.72 \def\listfigurename{Lista de Figuras}%
26.73 \def\listtablename{Lista de Tabelas}%
26.74 \def\indexname{\’Indice Remissivo}%
26.75 \def\figurename{Figura}%
26.76 \def\tablename{Tabela}%
26.77 \def\partname{Parte}%
26.78 \def\enclname{Anexo}%
26.79 \def\ccname{C\’opia para}%
26.80 \def\headtoname{Para}%
26.81 \def\pagename{P\’agina}%
26.82 \def\seename{veja}%
26.83 \def\alsoname{veja tamb\’em}%
26.84 \def\proofname{Demonstra\c{c}\~ao}%
26.85 }

```

`\datebrazil` The macro `\datebrazil` redefines the command `\today` to produce Brazilian dates, for which the names of the months are not capitalized.

```

26.86 \@namedef{date\CurrentOption}{%
26.87 \def\today{\number\day\space de\space\ifcase\month\or
26.88 janeiro\or fevereiro\or mar\c{c}o\or abril\or maio\or junho\or
26.89 julho\or agosto\or setembro\or outubro\or novembro\or dezembro

```



```

26.90      \fi
26.91      \space de\space\number\year}}
26.92 \fi

```

`\portugeshyphenmins` Set correct values for `\lefthyphenmin` and `\righthyphenmin`.

```
26.93 \@namedef{\CurrentOption hyphenmins}{\tw@\tw@}
```

`\extrasportuges` The macro `\extrasportuges` will perform all the extra definitions needed for the Portuguese language. The macro `\noextrasportuges` is used to cancel the actions of `\extrasportuges`.

For Portuguese the " character is made active. This is done once, later on its definition may vary. Other languages in the same document may also use the " character for shorthands; we specify that the portuguese group of shorthands should be used.

```

26.94 \initiate@active@char{"}
26.95 \@namedef{extras\CurrentOption}{\languageshorthands{portuges}}
26.96 \expandafter\addto\csname extras\CurrentOption\endcsname{%
26.97   \bbl@activate{}}
26.98 %\addto\noextrasportuges{\bbl@deactivate{}}

```

First we define access to the guillemets for quotations,

```

26.99 \declare@shorthand{portuges}{"<"}{%
26.100   \textormath{\guillemotleft}}{\mbox{\guillemotleft}}}
26.101 \declare@shorthand{portuges}{">"}{%
26.102   \textormath{\guillemotright}}{\mbox{\guillemotright}}}

```

then we define two shorthands to be able to specify hyphenation breakpoints that behave a little different from `\-`.

```

26.103 \declare@shorthand{portuges}{"-"}{\allowhyphens-\allowhyphens}
26.104 \declare@shorthand{portuges}{""}{\hskip\z@skip}

```

And we want to have a shorthand for disabling a ligature.

```

26.105 \declare@shorthand{portuges}{"|"}{%
26.106   \textormath{\discretionary{-}{-}{\kern.03em}}{}}

```

`\-` All that is left now is the redefinition of `\-`. The new version of `\-` should indicate an extra hyphenation position, while allowing other hyphenation positions to be generated automatically. The standard behaviour of \TeX in this respect is very unfortunate for languages such as Dutch and German, where long compound words are quite normal and all one needs is a means to indicate an extra hyphenation position on top of the ones that \TeX can generate from the hyphenation patterns.

```

26.107 \expandafter\addto\csname extras\CurrentOption\endcsname{%
26.108   \babel@save\-\}
26.109 \expandafter\addto\csname extras\CurrentOption\endcsname{%
26.110   \def\-\{\allowhyphens\discretionary{-}{-}{\allowhyphens}}

```

`\ord` We also provide an easy way to typeset ordinals, both in the male (`\ord` or `\ro`) and the female (`orda` or `\ra`) form.

```

\ord26.111 \def\ord{${\rm o}$}
\ra26.112 \def\orda{${\rm a}$}
26.113 \let\ro\ord\let\ra\orda

```

The macro `\ldf@finish` takes care of looking for a configuration file, setting the main language to be switched on at `\begin{document}` and resetting the category code of `@` to its original value.

26.114 `\ldf@finish\CurrentOption`

26.115 `\code`

27 The Spanish language

The file `spanish.dtx`²⁵ defines all the language definition macro's for the Spanish²⁶ language.

This file²⁷ incorporates the result of discussions held in the Spanish-TeX²⁸ electronic mail list.

For this language the characters ' ~ and " are made active. In table 7 an overview is given of their purpose. These active accent characters behave according

'a	an accent that allows hyphenation. Valid for all vowels uppercase and lowercase.
'n	a n with a tilde. This is included to improve compatibility with FTC. Works for uppercase too.
"	disable ligature at this position.
"-	an explicit hyphen sign, allowing hyphenation in the rest of the word.
""	like "-", but producing no hyphen sign (for words that should break at some sign such as "entrada/salida."
\-	like the old \-, but allowing hyphenation in the rest of the word.
"u	a u with dieresis allowing hyphenation.
"a	feminine ordinal as in 1 ^a .
"o	masculine ordinal as in 1 ^o .
"<	for French left double quotes (similar to <<).
">	for French right double quotes (similar to >>).
~n	a n with tilde. Works for uppercase too.

Table 7: The extra definitions made by `spanish.ldf`

to their original definitions if not followed by one of the characters indicated in that table.

This option includes support for working with extended, 8-bit fonts, if available. Old versions of this file based this support on the existence of special macros with names as in Ferguson's ML-TeX. This is no longer the case. Support is now based on providing an appropriate definition for the accent macros on entry to the Spanish language. This is automatically done by L^AT_EX 2_ε or NFSS2. If T1 encoding is chosen, and provided that adequate hyphenation patterns²⁹ exist, it

²⁵The file described in this section has version number v3.4g and was last revised on 1997/01/15. The original author is Julio Sánchez, (jsanchez@gmv.es).

²⁶Catalan used to be part of this file but is now on its own file.

²⁷In writing this file, many ideas and actual coding solutions have been taken from a number of sources. The language definition files `dutch.sty` and `germanb.sty` are the main contributors and are not explicitly mentioned in the sequel. J. L. Braams and Bernd Raichle have given helpful advice. Another source of inspiration is the experience gained in the use of FTC, a software package written by José A. Mañas. The members of the Spanish-TeX list have helped clarify a number of issues. Other sources are explicitly acknowledged when used. If you think that you contributed something and you are not mentioned, please let me (jsanchez@gmv.es) know. I humbly apologize for any omission.

²⁸spanish-tex@goya.eunet.es, subscription requests can be sent to the address listserv@goya.eunet.es. This list is devoted to discussions on support in TeX for Spanish. Comments on this language option are welcome there or directly to jsanchez@gmv.es.

²⁹One source for such patterns is the archive at [ftp.eunet.es](ftp://ftp.eunet.es) that can be accessed by anonymous FTP or electronic mail to ftpmail@goya.eunet.es. They are in the `info` direc-

is possible to get better hyphenation for Spanish than before. The easiest way to use the new encoding with L^AT_EX 2_ε to load the package `t1enc` with `\usepackage`. This must be done before loading `babel`.

If the combination of keyboard and T_EX version that the user has is able to produce the accented characters in the T1 encoding, the user could see the accented characters in the editor, greatly improving the readability of the document source. As of today, this is not a recommended method for producing documents for distribution, although it is possible to mechanically translate the document so that the receiver can make use of it. If care is taken to define the encoding needed by the document, the results are pretty portable.

This option file will automatically detect if the T1 encoding is being used and behave appropriately. If any other encoding is being used, the accent macros will be redefined to allow hyphenation on the accented words.

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the `@` sign, etc.

```
27.1 (*code)
27.2 \LdfInit{spanish}\captionsspanish
```

When this file is read as an option, i.e. by the `\usepackage` command, `spanish` could be an ‘unknown’ language in which case we have to make it known. So we check for the existence of `\l@spanish` to see whether we have to do something here.

```
27.3 \ifx\l@spanish\@undefined
27.4   \@nopatterns{Spanish}
27.5   \adddialect\l@spanish0
27.6 \fi
```

The next step consists of defining commands to switch to (and from) the Spanish language.

`\captionsspanish` The macro `\captionsspanish` defines all strings³⁰ used in the four standard documentclasses provided with L^AT_EX.

```
27.7 \addto\captionsspanish{%
27.8   \def\prefacename{Prefacio}%
27.9   \def\refname{Referencias}%
27.10  \def\abstractname{Resumen}%
27.11  \def\bibname{Bibliograf'\{i}a}%
27.12  \def\chaptername{Cap'\{i}tulo}%
27.13  \def\appendixname{Ap'\endice}%
27.14  \def\contentsname{'Indice General}%
27.15  \def\listfigurename{'Indice de Figuras}%
27.16  \def\listtablename{'Indice de Tablas}%
27.17  \def\indexname{'Indice de Materias}%
27.18  \def\figurename{Figura}%
27.19  \def\tablename{Tabla}%
27.20  \def\partname{Parte}%
27.21  \def\enclname{Adjunto}%
```

tory `src/TeX/spanish`. The list of Frequently Asked Questions with Answers about T_EX for Spanish is kept there as well. That list is meant to be a summary of the discussions held in the Spanish-T_EX mail list. Warning: It is in Spanish.

³⁰The accent on the uppercase ‘I’ is intentional, following the recommendation of the *Real Academia de la Lengua* in *Esbozo de una Nueva Gramática de la Lengua Española*, *Comisión de Gramática*, Espasa-Calpe, 1973.

```

27.22 \def\ccname{Copia a}%
27.23 \def\headtoname{A}%
27.24 \def\pagename{P\'agina}%
27.25 \def\seename{v\'ease}%
27.26 \def\alsoname{v\'ease tambi\'en}%
27.27 \def\proofname{Demostraci\'on}%
27.28 }%

```

`\datespanish` The macro `\datespanish` redefines the command `\today` to produce Spanish³¹ dates.

```

27.29 \def\datespanish{%
27.30 \def\today{\number\day~de\space\ifcase\month\or
27.31 enero\or febrero\or marzo\or abril\or mayo\or junio\or
27.32 julio\or agosto\or septiembre\or octubre\or noviembre\or diciembre\fi
27.33 \space de~\number\year}}

```

`\extrasspanish` The macro `\extrasspanish` will perform all the extra definitions needed for the Spanish language. The macro `\noextrasspanish` is used to cancel the actions of `\extrasspanish`. For Spanish, some characters are made active or are redefined. In particular, the " character, the ' character and the ~ character receive new meanings. Therefore these characters have to be treated as 'special' characters.

```

27.34 \addto\extrasspanish{\languageshorthands{spanish}}
27.35 \initiate@active@char{"}
27.36 \initiate@active@char{~}
27.37 \addto\extrasspanish{%
27.38 \bbl@activate{"}%
27.39 \bbl@activate{~}}

27.40 \ifpackagewith{babel}{activeacute}{%
27.41 \initiate@active@char{'}}{}
27.42 \ifpackagewith{babel}{activeacute}{%
27.43 \addto\extrasspanish{\bbl@activate{'}}{}
27.44 %\addto\noextrasspanish{
27.45 % \bbl@deactivate{"}\bbl@deactivate{~}\bbl@deactivate{'}}

```

Apart from the active characters some other macros get a new definition.

Therefore we store the current one to be able to restore them later.

```

27.46 \addto\extrasspanish{%
27.47 \babel@save\"%
27.48 \babel@save\~%
27.49 \def"\{\protect\@umlaut}%
27.50 \def\~{\protect\@tilde}}
27.51 \ifpackagewith{babel}{activeacute}{%
27.52 \babel@save\'
27.53 \addto\extrasspanish{\def'\{\protect\@acute}}
27.54 }{}

```

`\spanishhyphenmins` Spanish hyphenation uses `\lefthyphenmin` and `\righthyphenmin` both set to 2.

```

27.55 \def\spanishhyphenmins{\tw@\tw@}

```

³¹Months are written lowercased. This has been cause of some controversy. This file follows *Diccionario de Uso de la Lengua Española*, María Moliner, 1990, that is in agreement with the most common practice.

`\dieresis` The original definition of `\"` is stored as `\dieresis`, because the we do not know
`\textacute` what is its definition, since it depends on the encoding we are using or on special
`\texttilde` macros that the user might have loaded. The expansion of the macro might use
the \TeX `\accent` primitive using some particular accent that the font provides
or might check if a combined accent exists in the font. These two cases happen
with respectively OT1 and T1 encodings. For this reason we save the definition of
`\"` and use that in the definition of other macros. We do likewise for `\'` and `\~`.
The present coding of this option file is incorrect in that it can break when the
encoding changes. We do not use `\acute` or `\tilde` as the macro names because
they are already defined as `\mathaccent`.

```
27.56 \let\dieresis\"
27.57 \let\texttilde\~
27.58 \@ifpackagewith{babel}{activeacute}{\let\textacute\'}{}
```

`\@umlaut` We check the encoding and if not using T1, we make the accents expand but
`\@acute` enabling hyphenation beyond the accent. If this is the case, not all break positions
`\@tilde` will be found in words that contain accents, but this is a limitation in \TeX . An
unsolved problem here is that the encoding can change at any time. The definitions
below are made in such a way that a change between two 256-char encodings
are supported, but changes between a 128-char and a 256-char encoding are not
properly supported. We check if T1 is in use. If not, we will give a warning and
proceed redefining the accent macros so that \TeX at least finds the breaks that
are not too close to the accent. The warning will only be printed to the log file.

```
27.59 \ifx\DeclareFontShape\undefined
27.60   \wlog{Warning: You are using an old LaTeX}
27.61   \wlog{Some word breaks will not be found.}
27.62   \def\@umlaut#1{\allowhyphens\dieresis{#1}\allowhyphens}
27.63   \def\@tilde#1{\allowhyphens\texttilde{#1}\allowhyphens}
27.64   \@ifpackagewith{babel}{activeacute}{%
27.65     \def\@acute#1{\allowhyphens\textacute{#1}\allowhyphens}}{}
27.66 \else
27.67   \edef\next{T1}
27.68   \ifx\f@encoding\next
27.69     \let\@umlaut\dieresis
27.70     \let\@tilde\texttilde
27.71     \@ifpackagewith{babel}{activeacute}{%
27.72       \let\@acute\textacute}{}
27.73   \else
27.74     \wlog{Warning: You are using encoding \f@encoding\space
27.75       instead of T1.}
27.76     \wlog{Some word breaks will not be found.}
27.77     \def\@umlaut#1{\allowhyphens\dieresis{#1}\allowhyphens}
27.78     \def\@tilde#1{\allowhyphens\texttilde{#1}\allowhyphens}
27.79     \@ifpackagewith{babel}{activeacute}{%
27.80       \def\@acute#1{\allowhyphens\textacute{#1}\allowhyphens}}{}
27.81   \fi
27.82 \fi
```

Now we can define our shorthands: the umlauts,

```
27.83 \declare@shorthand{spanish}{\"u}{\@umlaut u}
27.84 \declare@shorthand{spanish}{\"U}{\@umlaut U}
```

french quotes,

```

27.85 \declare@shorthand{spanish}{"<"}{%
27.86   \textormath{\guillemotleft}}{\mbox{\guillemotleft}}
27.87 \declare@shorthand{spanish}{">"}{%
27.88   \textormath{\guillemotright}}{\mbox{\guillemotright}}

ordinals32,

27.89 \declare@shorthand{spanish}{"o"}{%
27.90   \leavevmode\raise1ex\hbox{\underbar{\scriptsize o}}}
27.91 \declare@shorthand{spanish}{"a"}{%
27.92   \leavevmode\raise1ex\hbox{\underbar{\scriptsize a}}}

acute accents,

27.93 \ifpackagewith{babel}{activeacute}{%
27.94   \declare@shorthand{spanish}{'a'}{\textormath{\@acute a}{\prime a}}
27.95   \declare@shorthand{spanish}{'e'}{\textormath{\@acute e}{\prime e}}
27.96   \declare@shorthand{spanish}{'i'}{\textormath{\@acute i}{\prime i}}
27.97   \declare@shorthand{spanish}{'o'}{\textormath{\@acute o}{\prime o}}
27.98   \declare@shorthand{spanish}{'u'}{\textormath{\@acute u}{\prime u}}
27.99   \declare@shorthand{spanish}{'A'}{\textormath{\@acute A}{\prime A}}
27.100  \declare@shorthand{spanish}{'E'}{\textormath{\@acute E}{\prime E}}
27.101  \declare@shorthand{spanish}{'I'}{\textormath{\@acute I}{\prime I}}
27.102  \declare@shorthand{spanish}{'O'}{\textormath{\@acute O}{\prime O}}
27.103  \declare@shorthand{spanish}{'U'}{\textormath{\@acute U}{\prime U}}

the acute accent,

27.104   \declare@shorthand{spanish}{' '}{%
27.105     \textormath{\textquotedblright}{\sp\bgroup\prim@s'}}

tildes,

27.106   \declare@shorthand{spanish}{'n'}{\textormath{\~n}{\prime n}}
27.107   \declare@shorthand{spanish}{'N'}{\textormath{\~N}{\prime N}}
27.108   }{}
27.109 \declare@shorthand{spanish}{\~n}{\textormath{\~n}{\@tilde n}}
27.110 \declare@shorthand{spanish}{\~N}{\textormath{\~N}{\@tilde N}}

and some additional commands:

27.111 \declare@shorthand{spanish}{"-}{\allowhyphens\-\allowhyphens}
27.112 \declare@shorthand{spanish}{"|"}{%
27.113   \textormath{\penalty\@M\discretionary{-}{-}{\kern.03em}%
27.114     \allowhyphens}{}}
27.115 \declare@shorthand{spanish}{""}{\hskip\z@skip}

The macro \ldf@finish takes care of looking for a configuration file, setting
the main language to be switched on at \begin{document} and resetting the
category code of @ to its original value.

27.116 \ldf@finish{spanish}
27.117 \code

```

³²The code for the ordinals was taken from the answer provided by Raymond Chen (raymond@math.berkeley.edu) to a question by Joseph Gil (yogi@cs.ubc.ca) in `comp.text.tex`.

28 The Catalan language

The file `catalan.dtx`³³ defines all the language-specific macro's for the Catalan language.

For this language only the double quote character (") is made active by default. In table 8 an overview is given of the new macros defined and the new meanings of ". Additionally to that, the user can explicitly activate the acute accent or apostrophe (') and/or the grave accent (‘) characters by using the `activeacute` and `activegrave` options. In that case, the definitions shown in table 9 also become available³⁴.

<code>\lgem</code>	geminated-l digraph (similar to l·l). <code>\Lgem</code> produces the uppercase version.
<code>\up</code>	Macro to help typing raised ordinals, like 1 ^{er} . Takes one argument.
<code>\-</code>	like the old <code>\-</code> , but allowing hyphenation in the rest of the word.
<code>"i</code>	i with diaeresis, allowing hyphenation in the rest of the word. Valid for the following vowels: i, u (both lowercase and uppercase).
<code>"c</code>	c-cedilla (ç). Valid for both uppercase and lowercase c.
<code>"l</code>	geminated-l digraph (similar to l·l). Valid for both uppercase and lowercase l.
<code>"<</code>	French left double quotes (similar to <<).
<code>"></code>	French right double quotes (similar to >>).
<code>"-</code>	explicit hyphen sign, allowing hyphenation in the rest of the word.
<code>" </code>	disable ligature at this position.

Table 8: Extra definitions made by file `catalan.ldf` (activated by default)

<code>'e</code>	acute accented a, allowing hyphenation in the rest of the word. Valid for the following vowels: e, i, o, u (both lowercase and uppercase).
<code>‘a</code>	grave accented a, allowing hyphenation in the rest of the word. Valid for the following vowels: a, e, o (both lowercase and uppercase).

Table 9: Extra definitions made by file `catalan.ldf` (activated only when using the options `activeacute` and `activegrave`)

These active accents characters behave according to their original definitions if not followed by one of the characters indicated in that table.

³³The file described in this section has version number v2.2h and was last revised on 1997/01/08.

³⁴Please note that if the acute accent character is active, it is necessary to take special care of coding apostrophes in a way which cannot be confounded with accents. Therefore, it is necessary to type `l'{}estri` instead of `l'estri`.

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the `@` sign, etc.

```
28.1 (*code)
28.2 \LdfInit{catalan}\captionscatalan
```

When this file is read as an option, i.e. by the `\usepackage` command, `catalan` could be an ‘unknown’ language in which case we have to make it known. So we check for the existence of `\l@catalan` to see whether we have to do something here.

```
28.3 \ifx\l@catalan\@undefined
28.4   \@nopatterns{Catalan}
28.5   \adddialect\l@catalan0
28.6 \fi
```

The next step consists of defining commands to switch to (and from) the Catalan language.

`\captionscatalan` The macro `\captionscatalan` defines all strings used in the four standard documentclasses provided with L^AT_EX.

```
28.7 \addto\captionscatalan{%
28.8   \def\prefacename{Pr\‘oleg}%
28.9   \def\refname{Refer\‘encies}%
28.10  \def\abstractname{Resum}%
28.11  \def\bibName{Bibliografia}%
28.12  \def\chaptername{Cap\‘{i}tol}%
28.13  \def\appendixname{Ap\‘endix}%
28.14  \def\contentsname{\‘Index}%
28.15  \def\listfigurename{\‘Index de figures}%
28.16  \def\listtablename{\‘Index de taules}%
28.17  \def\indexname{\‘Index alfab\‘etic}%
28.18  \def\figurename{Figura}%
28.19  \def\tablename{Taula}%
28.20  \def\partname{Part}%
28.21  \def\enclname{Adjunt}%
28.22  \def\ccname{C\‘opies a}%
28.23  \def\headtoname{A}%
28.24  \def\pagename{P\‘agina}%
28.25  \def\seename{Vegeu}%
28.26  \def\alsoname{Vegeu tamb\‘e}%
28.27  \def\proofname{Demostraci\‘o}%
28.28 }
```

`\datecatalan` The macro `\datecatalan` redefines the command `\today` to produce Catalan dates. Months are written in lowercase³⁵.

```
28.29 \def\datecatalan{%
28.30   \def\today{\number\day~\ifcase\month\or
28.31     de gener\or de febrer\or de mar\c{c}\or d’abril\or de maig\or
28.32     de juny\or de juliol\or d’agost\or de setembre\or d’octubre\or
28.33     de novembre\or de desembre\fi
28.34     \space de~\number\year}}
```

³⁵This seems to be the common practice. See for example: E. Coromina, *El 9 Nou: Manual de redacció i estil*, Ed. Eumo, Vic, 1993

`\extrascatalan` The macro `\extrascatalan` will perform all the extra definitions needed for the Catalan language. The macro `\noextrascatalan` is used to cancel the actions of `\extrascatalan`.

To improve hyphenation we give the grave character (‘) a non-zero lower case code; when we do that T_EX will find more breakpoints in words that contain this character in its rôle as apostrophe.

```
28.35 \addto\extrascatalan{%
28.36   \lccode‘’=‘’}
28.37 \addto\noextrascatalan{%
28.38   \lccode‘’=0}
```

For Catalan, some characters are made active or are redefined. In particular, the " character receives a new meaning; this can also happen for the ’ character and the ‘ character when the options `activegrave` and/or `activeacute` are specified.

```
28.39 \addto\extrascatalan{\languageshorthands{catalan}}
28.40 \initiate@active@char{"}
28.41 \addto\extrascatalan{\bbl@activate{}}
28.42 \@ifpackagewith{babel}{activegrave}{%
28.43   \initiate@active@char{‘}{}}
28.44 \@ifpackagewith{babel}{activegrave}{%
28.45   \addto\extrascatalan{\bbl@activate{‘}}{}}
28.46 \@ifpackagewith{babel}{activeacute}{%
28.47   \initiate@active@char{’}{}}
28.48 \@ifpackagewith{babel}{activeacute}{%
28.49   \addto\extrascatalan{\bbl@activate{’}}{}}
28.50 %\addto\noextrascatalan{%
28.51 %  \bbl@deactivate{"}
28.52 %  \bbl@deactivate{‘}\bbl@deactivate{’}}
```

Apart from the active characters some other macros get a new definition. Therefore we store the current ones to be able to restore them later. When their current meanings are saved, we can safely redefine them.

We provide new definitions for the accent macros when one or both of the options `activegrave` or `activeacute` were specified.

```
28.53 \addto\extrascatalan{%
28.54   \babel@save\"%
28.55   \def\"{\protect\@umlaut}}%
28.56 \@ifpackagewith{babel}{activegrave}{%
28.57   \babel@save\‘%
28.58   \addto\extrascatalan{\def\‘{\protect\@grave}}
28.59 }{}
28.60 \@ifpackagewith{babel}{activeacute}{%
28.61   \babel@save\’%
28.62   \addto\extrascatalan{\def\’{\protect\@acute}}
28.63 }{}}
```

All the code above is necessary because we need a few extra active characters. These characters are then used as indicated in tables 8 and 9.

`\dieresis` The original definition of \" is stored as `\dieresis`, because the definition of \" might not be the default plain T_EX one. If the user uses POSTSCRIPT fonts with the Adobe font encoding the " character is not in the same position as in Knuth’s font encoding. In this case \" will not be defined as `\accent"7F 1`, but

as `\accent'310 #1`. Something similar happens when using fonts that follow the Cork encoding. For this reason we save the definition of `\'` and use that in the definition of other macros. We do likewise for `\'`, and `\'`.

```

28.64 \let\dieresis\"
28.65 \@ifpackagewith{babel}{activegrave}{\let\textgrave\'{}}
28.66 \@ifpackagewith{babel}{activeacute}{\let\textacute\'{}}

\@umlaut We check the encoding and if not using T1, we make the accents expand but
\@acute enabling hyphenation beyond the accent. If this is the case, not all break positions
\@grave will be found in words that contain accents, but this is a limitation in TEX. An
unsolved problem here is that the encoding can change at any time. The definitions
below are made in such a way that a change between two 256-char encodings
are supported, but changes between a 128-char and a 256-char encoding are not
properly supported. We check if T1 is in use. If not, we will give a warning and
proceed redefining the accent macros so that TEX at least finds the breaks that
are not too close to the accent. The warning will only be printed to the log file.

28.67 \ifx\DeclareFontShape\@undefined
28.68 \wlog{Warning: You are using an old LaTeX}
28.69 \wlog{Some word breaks will not be found.}
28.70 \def\@umlaut#1{\allowhyphens\dieresis{#1}\allowhyphens}
28.71 \@ifpackagewith{babel}{activeacute}{%
28.72 \def\@acute#1{\allowhyphens\textacute{#1}\allowhyphens}}{}
28.73 \@ifpackagewith{babel}{activegrave}{%
28.74 \def\@grave#1{\allowhyphens\textgrave{#1}\allowhyphens}}{}
28.75 \else
28.76 \edef\next{T1}
28.77 \ifx\f@encoding\next
28.78 \let\@umlaut\dieresis
28.79 \@ifpackagewith{babel}{activeacute}{%
28.80 \let\@acute\textacute}{}
28.81 \@ifpackagewith{babel}{activegrave}{%
28.82 \let\@grave\textgrave}{}
28.83 \else
28.84 \wlog{Warning: You are using encoding \f@encoding\space
28.85 instead of T1.}
28.86 \wlog{Some word breaks will not be found.}
28.87 \def\@umlaut#1{\allowhyphens\dieresis{#1}\allowhyphens}
28.88 \@ifpackagewith{babel}{activeacute}{%
28.89 \def\@acute#1{\allowhyphens\textacute{#1}\allowhyphens}}{}
28.90 \@ifpackagewith{babel}{activegrave}{%
28.91 \def\@grave#1{\allowhyphens\textgrave{#1}\allowhyphens}}{}
28.92 \fi
28.93 \fi

```

If the user setup has extended fonts, the Ferguson macros are required to be defined. We check for their existence and, if defined, expand to whatever they are defined to. For instance, `\'a` would check for the existence of a `\@ac@a` macro. It is assumed to expand to the code of the accented letter. If it is not defined, we assume that no extended codes are available and expand to the original definition but enabling hyphenation beyond the accent. This is as best as we can do. It is better if you have extended fonts or ML- $\text{T}_{\text{E}}\text{X}$ because the hyphenation algorithm can work on the whole word. The following macros are directly derived from

Now we can define our shorthands: the diaeresis and “ela geminada” support,

```
28.94 \declare@shorthand{catalan}{\i}{\textormath{\@umlaut\i}{\ddot\imath}}
28.95 \declare@shorthand{catalan}{\l}{\lgem{}}
28.96 \declare@shorthand{catalan}{\u}{\textormath{\@umlaut u}{\ddot u}}
28.97 \declare@shorthand{catalan}{\I}{\textormath{\@umlaut I}{\ddot I}}
28.98 \declare@shorthand{catalan}{\L}{\Lgem{}}
28.99 \declare@shorthand{catalan}{\U}{\textormath{\@umlaut U}{\ddot U}}
```

cedille,

```
28.100 \declare@shorthand{catalan}{\c}{\textormath{\c c}{\prime} c}}
28.101 \declare@shorthand{catalan}{\C}{\textormath{\c C}{\prime} C}}
```

‘french’ quote characters,

```
28.102 \declare@shorthand{catalan}{\<}{\%
28.103 \textormath{\guillemotleft}}{\mbox{\guillemotleft}}
28.104 \declare@shorthand{catalan}{\>}{\%
28.105 \textormath{\guillemotright}}{\mbox{\guillemotright}}
```

grave accents,

```
28.106 \@ifpackagewith{babel}{activegrave}{\%
28.107 \declare@shorthand{catalan}{\a}{\textormath{\@grave a}{\grave a}}
28.108 \declare@shorthand{catalan}{\e}{\textormath{\@grave e}{\grave e}}
28.109 \declare@shorthand{catalan}{\o}{\textormath{\@grave o}{\grave o}}
28.110 \declare@shorthand{catalan}{\A}{\textormath{\@grave A}{\grave A}}
28.111 \declare@shorthand{catalan}{\E}{\textormath{\@grave E}{\grave E}}
28.112 \declare@shorthand{catalan}{\O}{\textormath{\@grave O}{\grave O}}
28.113 \declare@shorthand{catalan}{\'}{\textquotedblleft}
28.114 }
```

acute accents,

```
28.115 \@ifpackagewith{babel}{activeacute}{\%
28.116 \declare@shorthand{catalan}{\a}{\textormath{\@acute a}{\prime} a}}
28.117 \declare@shorthand{catalan}{\e}{\textormath{\@acute e}{\prime} e}}
28.118 \declare@shorthand{catalan}{\i}{\textormath{\@acute i}{\prime} i}}
28.119 \declare@shorthand{catalan}{\o}{\textormath{\@acute o}{\prime} o}}
28.120 \declare@shorthand{catalan}{\u}{\textormath{\@acute u}{\prime} u}}
28.121 \declare@shorthand{catalan}{\A}{\textormath{\@acute A}{\prime} A}}
28.122 \declare@shorthand{catalan}{\E}{\textormath{\@acute E}{\prime} E}}
28.123 \declare@shorthand{catalan}{\I}{\textormath{\@acute I}{\prime} I}}
28.124 \declare@shorthand{catalan}{\O}{\textormath{\@acute O}{\prime} O}}
28.125 \declare@shorthand{catalan}{\U}{\textormath{\@acute U}{\prime} U}}
28.126 \declare@shorthand{catalan}{\'}{\%
28.127 \textormath{\csname normal@char\string'\endcsname}{\prime}}}
```

the acute accent,

```
28.128 \declare@shorthand{catalan}{\'}{\%
28.129 \textormath{\textquotedblright}{\sp\bgroup\prim's'}}
28.130 }
```

³⁶A problem is perceived here with these macros when used in a multilingual environment where extended hyphenation patterns are available for some but not all languages. Assume that no extended patterns exist at some site for French and that `french.sty` would adopt this scheme too. In that case, ‘e in French would produce the combined accented letter, but hyphenation around it would be suppressed. Both language options would need an independent method to know whether they have extended patterns available. The precise impact of this problem and the possible solutions are under study.

and finally, some support definitions

```
28.131 \declare@shorthand{catalan}{"-}{\allowhyphens-\allowhyphens}
28.132 \declare@shorthand{catalan}{"}{|}%
28.133 \textormath{\penalty\M\discretionary{-}{}{\kern.03em}%
28.134 \allowhyphens}{}}
```

- \- All that is left now is the redefinition of \-. The new version of \- should indicate an extra hyphenation position, while allowing other hyphenation positions to be generated automatically. The standard behaviour of T_EX in this respect is unfortunate for Catalan but not as much as for Dutch or German, where long compound words are quite normal and all one needs is a means to indicate an extra hyphenation position on top of the ones that T_EX can generate from the hyphenation patterns. However, the average length of words in Catalan makes this desirable and so it is kept here.

```
28.135 \addto\extrascatalan{%
28.136 \babel@save{\-}%
28.137 \def\-\{\allowhyphens\discretionary{-}{}{\allowhyphens}}
```

\lgem Here we define a macro for typing the catalan “ela geminada” (geminated l).

\Lgem The macros \lgem and \Lgem have been chosen for its lowercase and uppercase representation, respectively³⁷.

The code used in the actual macro used is a combination of the one proposed by Feruglio and Fuster³⁸ and the proposal³⁹ from Valiente presented at the T_EX Users Group Annual Meeting in 1995. This last proposal has not been fully implemented due to its limitation to CM fonts.

```
28.138 \newdimen\leftllkern \newdimen\rightllkern \newdimen\raiselldim
28.139 \def\lgem{%
28.140 \ifmmode
28.141 \csname normal@char\string\endcsname 1%
28.142 \else
28.143 \leftllkern=0pt\rightllkern=0pt\raiselldim=0pt%
28.144 \setbox0\hbox{1}\setbox1\hbox{1/}\setbox2\hbox{.}%
28.145 \advance\raiselldim by \the\fontdimen5\the\font
28.146 \advance\raiselldim by -\ht2%
28.147 \leftllkern=-.25\wd0%
28.148 \advance\leftllkern by \wd1%
28.149 \advance\leftllkern by -\wd0%
28.150 \rightllkern=-.25\wd0%
28.151 \advance\rightllkern by -\wd1%
28.152 \advance\rightllkern by \wd0%
28.153 \allowhyphens\discretionary{1-}{1}%
28.154 {\hbox{1}\kern\leftllkern\raise\raiselldim\hbox{.}%
28.155 \kern\rightllkern\hbox{1}}\allowhyphens
28.156 \fi
28.157 }
28.158 \def\Lgem{%
28.159 \ifmmode
28.160 \csname normal@char\string\endcsname L%
```

³⁷The macro names \ll and \LL were not taken because of the fact that \ll is already used in mathematical mode.

³⁸G. Valiente and R. Fuster, Typesetting Catalan Texts with T_EX, *TUGboat* **14**(3), 1993.

³⁹G. Valiente, Modern Catalan Typographical Conventions, *TUGboat* **16**(3), 1995.

```

28.161 \else
28.162 \leftllkern=0pt\rightllkern=0pt\raiselldim=0pt%
28.163 \setbox0\hbox{L}\setbox1\hbox{L/}\setbox2\hbox{.}%
28.164 \advance\raiselldim by .5\ht0%
28.165 \advance\raiselldim by -.5\ht2%
28.166 \leftllkern=-.125\wd0%
28.167 \advance\leftllkern by \wd1%
28.168 \advance\leftllkern by -\wd0%
28.169 \rightllkern=-\wd0%
28.170 \divide\rightllkern by 6%
28.171 \advance\rightllkern by -\wd1%
28.172 \advance\rightllkern by \wd0%
28.173 \allowhyphens\discretionary{L-}{L}%
28.174 {\hbox{L}\kern\leftllkern\raise\raiselldim\hbox{.}%
28.175 \kern\rightllkern\hbox{L}}\allowhyphens
28.176 \fi
28.177 }

```

\l.1 It seems to be the most natural way of entering the “ela geminda” to use the \L.L sequences \l.1 and \L.L. These are not really macro’s by themselves but the macros \l and \L with delimited arguments. Therefor we define two macros that check if the next character is a period. If not the “polish l” will be typeset, otherwise a “ela geminada” will be typeset and the next two tokens will be ‘eaten’.

```

28.178 \let\lslash\l
28.179 \let\Lslash\L
28.180 \DeclareRobustCommand\l{\@ifnextchar.\bbl@l\lslash}
28.181 \DeclareRobustCommand\L{\@ifnextchar.\bbl@L\Lslash}
28.182 \def\bbl@l#1#2{\lgem}
28.183 \def\bbl@L#1#2{\Lgem}

```

\up A macro for typesetting things like 1^{er} as proposed by Raymon Seroul⁴⁰.

```

28.184 \DeclareRobustCommand*\up}[1]{\textsuperscript{#1}}

```

The macro \ldf@finish takes care of looking for a configuration file, setting the main language to be switched on at \begin{document} and resetting the category code of @ to its original value.

```

28.185 \ldf@finish{catalan}
28.186 \code

```

⁴⁰This macro has been borrowed from francais.dtx

29 The Galician language

The file `galician.dtx`⁴¹ defines all the language definition macros for the Galician language.

For this language the characters `'`, `~` and `"` are made active. In table 10 an overview is given of their purpose. These active accents character behave according

<code>" </code>	disable ligature at this position.
<code>"-</code>	an explicit hyphen sign, allowing hyphenation in the rest of the word.
<code>\-</code>	like the old <code>\-</code> , but allowing hyphenation in the rest of the word.
<code>'a</code>	an accent that allows hyphenation. Valid for all vowels uppercase and lowercase.
<code>'n</code>	a n with a tilde. This is included to improve compatibility with FTC. Works for uppercase too.
<code>"u</code>	a u with dieresis allowing hyphenation.
<code>"a</code>	feminine ordinal as in 1 ^a .
<code>"o</code>	masculine ordinal as in 1 ^o .
<code>~n</code>	a n with tilde. Works for uppercase too.

Table 10: The extra definitions made by `galician.ldf`

to their original definitions if not followed by one of the characters indicated in that table.

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the `@` sign, etc.

```
29.1 (*code)
29.2 \LdfInit{galician}\captionsgalician
```

When this file is read as an option, i.e. by the `\usepackage` command, `galician` could be an ‘unknown’ language in which case we have to make it known. So we check for the existence of `\l@galician` to see whether we have to do something here.

```
29.3 \ifx\l@galician\@undefined
29.4 \@nopatterns{Galician}
29.5 \adddialect\l@galician0\fi
```

The next step consists of defining commands to switch to (and from) the Galician language.

`\captionsgalician` The macro `\captionsgalician` defines all strings used in the four standard documentclasses provided with L^AT_EX.

```
29.6 \addto\captionsgalician{%
29.7 \def\prefacename{Prefacio}%
29.8 \def\refname{Referencias}%
29.9 \def\abstractname{Resumo}%
29.10 \def\bibname{Bibliograf'\{i}a}%
29.11 \def\chaptername{Cap'\{i}tulo}%
29.12 \def\appendixname{Ap'\endice}%
```

⁴¹The file described in this section has version number v1.2g and was last revised on 1997/01/15.

```

29.13 \def\contentsname{\'Indice Xeral}%
29.14 \def\listfigurename{\'Indice de Figuras}%
29.15 \def\listtablename{\'Indice de T\'aboas}%
29.16 \def\indexname{\'Indice de Materias}%
29.17 \def\figurename{Figura}%
29.18 \def\tablename{T\'aboas}%
29.19 \def\partname{Parte}%
29.20 \def\enclname{Adxunto}%
29.21 \def\ccname{Copia a}%
29.22 \def\headtoname{A}%
29.23 \def\pagename{P\'axina}%
29.24 \def\seename{v\'exase}%
29.25 \def\alsoname{v\'exase tam\'en}%
29.26 \def\proofname{Proof}% <-- Needs Translation!
29.27 }

```

`\dategalician` The macro `\dategalician` redefines the command `\today` to produce Galician dates.

```

29.28 \def\dategalician{%
29.29   \def\today{\number\day~de\space\ifcase\month\or
29.30     xaneiro\or febreiro\or marzo\or abril\or maio\or xu~no\or
29.31     xullo\or agosto\or setembro\or outubro\or novembro\or decembro\fi
29.32     \space de~\number\year}}

```

`\extragalician` The macro `\extragalician` will perform all the extra definitions needed for the Galician language. The macro `\noextragalician` is used to cancel the actions of `\extragalician`.

For Galician, some characters are made active or are redefined. In particular, the " character and the ~ character receive new meanings this can also happen for the ' character when the option `activeacute` is specified.

```

29.33 \addto\extragalician{\languageshorthands{galician}}
29.34 \initiate@active@char{"}
29.35 \initiate@active@char{~}
29.36 \addto\extragalician{%
29.37   \bbl@activate{"}\bbl@activate{~}}
29.38 \@ifpackagewith{babel}{activeacute}{%
29.39   \initiate@active@char{'}}{}
29.40 \@ifpackagewith{babel}{activeacute}{%
29.41   \addto\extragalician{\bbl@activate{'}}{}
29.42 }\addto\noextragalician{%
29.43 % \bbl@deactivate{"}\bbl@deactivate{~}\bbl@deactivate{'}}

```

Apart from the active characters some other macros get a new definition. Therefore we store the current one to be able to restore them later.

```

29.44 \addto\extragalician{%
29.45   \babel@save{"\babel@save\~%
29.46   \def\{"{\protect\@umlaut}%
29.47   \def\~{\protect\@tilde}}
29.48 \@ifpackagewith{babel}{activeacute}{%
29.49   \babel@save\'%
29.50   \addto\extragalician{\def\'{\protect\@acute}}
29.51 }{}

```


All the code above is necessary because we need a few extra active characters. These characters are then used as indicated in table 10.

This option includes some support for working with extended, 8-bit fonts, if available. This assumes that the user has some macros predefined. For instance, if the user has a `\@ac@a` macro defined, the sequence `\'a` or `'a` will both expand to whatever `\@ac@a` is defined to expand, presumably á. The names of these macros are the same as those in Ferguson's ML- \TeX compatibility package on purpose. Using this method, and provided that adequate hyphenation patterns exist, it is possible to get better hyphenation for Galician than before. If the user has a terminal able to produce these codes directly, it is possible to do so. If the need arises to send the document to someone who does not have such support, it is possible to mechanically translate the document so that the receiver can make use of it.

To be able to define the function of the new accents, we first define a couple of 'support' macros.

`\dieresis` The original definition of `\"` is stored as `\dieresis`, because the definition of `\textacute` `\"` might not be the default plain \TeX one. If the user uses POSTSCRIPT fonts with the Adobe font encoding the `"` character is not in the same position as in Knuth's font encoding. In this case `\"` will not be defined as `\accent"7F #1`, but as `\accent'310 #1`. Something similar happens when using fonts that follow the Cork encoding. For this reason we save the definition of `\"` and use that in the definition of other macros. We do likewise for `\'` and `\~`.

```
29.52 \let\dieresis\"
```

```
29.53 \let\texttilde\~
```

```
29.54 \@ifpackagewith{babel}{activeacute}{\let\textacute\'}{}
```

`\@umlaut` If the user setup has extended fonts, the Ferguson macros are required to be defined. We check for their existence and, if defined, expand to whatever they are defined to. For instance, `\'a` would check for the existence of a `\@ac@a` macro. It is assumed to expand to the code of the accented letter. If it is not defined, we assume that no extended codes are available and expand to the original definition but enabling hyphenation beyond the accent. This is as best as we can do. It is better if you have extended fonts or ML- \TeX because the hyphenation algorithm can work on the whole word. The following macros are directly derived from ML- \TeX .⁴²

```
29.55 \def\@umlaut#1{\allowhyphens\dieresis{#1}\allowhyphens}
```

```
29.56 \def\@tilde#1{\allowhyphens\texttilde{#1}\allowhyphens}
```

```
29.57 \@ifpackagewith{babel}{activeacute}{%
```

```
29.58   \def\@acute#1{\allowhyphens\textacute{#1}\allowhyphens}}{}
```

Now we can define our shorthands: the umlauts,

```
29.59 \declare@shorthand{galician}{"-}{\allowhyphens-\allowhyphens}
```

```
29.60 \declare@shorthand{galician}{"|}{\discretionary{-}{ }\kern.03em}}
```

⁴²A problem is perceived here with these macros when used in a multilingual environment where extended hyphenation patterns are available for some but not all languages. Assume that no extended patterns exist at some site for French and that `french.sty` would adopt this scheme too. In that case, `'e` in French would produce the combined accented letter, but hyphenation around it would be suppressed. Both language options would need an independent method to know whether they have extended patterns available. The precise impact of this problem and the possible solutions are under study.

```

29.61 \declare@shorthand{galician}{\u}{\@umlaut{u}}
29.62 \declare@shorthand{galician}{\U}{\@umlaut{U}}
ordinals43,
29.63 \declare@shorthand{galician}{\o}{\%
29.64 \leavevmode\raise1ex\hbox{\underbar{\scriptsize o}}}
29.65 \declare@shorthand{galician}{\a}{\%
29.66 \leavevmode\raise1ex\hbox{\underbar{\scriptsize a}}}
acute accents,
29.67 \@ifpackagewith{babel}{activeacute}{\%
29.68 \declare@shorthand{galician}{\a}{\textormath{\@acute a}{\prime} a}}
29.69 \declare@shorthand{galician}{\e}{\textormath{\@acute e}{\prime} e}}
29.70 \declare@shorthand{galician}{\i}{\textormath{\@acute i}{\prime} i}}
29.71 \declare@shorthand{galician}{\o}{\textormath{\@acute o}{\prime} o}}
29.72 \declare@shorthand{galician}{\u}{\textormath{\@acute u}{\prime} u}}
29.73 \declare@shorthand{galician}{\A}{\textormath{\@acute A}{\prime} A}}
29.74 \declare@shorthand{galician}{\E}{\textormath{\@acute E}{\prime} E}}
29.75 \declare@shorthand{galician}{\I}{\textormath{\@acute I}{\prime} I}}
29.76 \declare@shorthand{galician}{\O}{\textormath{\@acute O}{\prime} O}}
29.77 \declare@shorthand{galician}{\U}{\textormath{\@acute U}{\prime} U}}
tildes,
29.78 \declare@shorthand{galician}{\n}{\textormath{\~n}{\prime} n}}
29.79 \declare@shorthand{galician}{\N}{\textormath{\~N}{\prime} N}}
the acute accent,
29.80 \declare@shorthand{galician}{\'}{\%
29.81 \textormath{\textquotedblright}{\sp\group\prim@s'}}
29.82 }{}
29.83 \declare@shorthand{galician}{\~n}{\textormath{\~n}{\@tilde n}}
29.84 \declare@shorthand{galician}{\~N}{\textormath{\~N}{\@tilde N}}

```

- \- All that is left now is the redefinition of \-. The new version of \- should indicate an extra hyphenation position, while allowing other hyphenation positions to be generated automatically. The standard behaviour of T_EX in this respect is unfortunate for Galician but not as much as for Dutch or German, where long compound words are quite normal and all one needs is a means to indicate an extra hyphenation position on top of the ones that T_EX can generate from the hyphenation patterns. However, the average length of words in Galician makes this desirable and so it is kept here.

```

29.85 \addto\extrasgalician{\%
29.86 \babel@save{\-}%
29.87 \def\-\{\allowhyphens\discretionary{-}{-}{\allowhyphens}}

```

The macro \ldf@finish takes care of looking for a configuration file, setting the main language to be switched on at \begin{document} and resetting the category code of @ to its original value.

```

29.88 \ldf@finish{galician}
29.89 \code

```

⁴³The code for the ordinals was taken from the answer provided by Raymond Chen (raymond@math.berkeley.edu) to a question by Joseph Gil (yogi@cs.ubc.ca) in comp.text.tex.

30 The Romanian language

The file `romanian.dtx`⁴⁴ defines all the language-specific macros for the Romanian language.

For this language currently no special definitions are needed or available.

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the `@` sign, etc.

```
30.1 (*code)
30.2 \LdfInit{romanian}\captionsromanian
```

When this file is read as an option, i.e. by the `\usepackage` command, `romanian` will be an ‘unknown’ language in which case we have to make it known. So we check for the existence of `\l@romanian` to see whether we have to do something here.

```
30.3 \ifx\l@romanian\@undefined
30.4     \nopatterns{Romanian}
30.5     \adddialect\l@romanian0\fi
```

The next step consists of defining commands to switch to (and from) the Romanian language.

`\captionsromanian` The macro `\captionsromanian` defines all strings used in the four standard documentclasses provided with L^AT_EX.

```
30.6 \addto\captionsromanian{%
30.7   \def\prefacename{Prefa\c{t}\u{a}}%
30.8   \def\refname{Bibliografie}%
30.9   \def\abstractname{Rezumat}%
30.10  \def\bibName{Bibliografie}%
30.11  \def\chaptername{Capitolul}%
30.12  \def\appendixname{Anexa}%
30.13  \def\contentsname{Cuprins}%
30.14  \def\listfigurename{List\u{a} de figuri}%
30.15  \def\listtablename{List\u{a} de tabele}%
30.16  \def\indexname{Glosar}%
30.17  \def\figurename{Figura}%      % sau Plan\c{s}a
30.18  \def\tablename{Tabela}%
30.19  \def\partname{Partea}%
30.20  \def\enclname{Anex\u{a}}%    % sau Anexe
30.21  \def\ccname{Copie}%
30.22  \def\headtoname{Pentru}%
30.23  \def\pagename{Pagina}%
30.24  \def\seenname{Vezi}%
30.25  \def\alsoname{Vezi de asemenea}%
30.26  \def\proofname{Demonstra\c{t}ie} %
30.27  }%
```

`\dateromanian` The macro `\dateromanian` redefines the command `\today` to produce Romanian dates.

```
30.28 \def\dateromanian{%
30.29 \def\today{\number\day~\ifcase\month\or
30.30   ianuarie\or februarie\or martie\or aprilie\or mai\or
```

⁴⁴The file described in this section has version number v1.2h and was last revised on 1996/12/23. A contribution was made by Umstatter Horst (hhu@cernvm.cern.ch).

```

30.31 iunie\or iulie\or august\or septembrie\or octombrie\or
30.32 noiembrie\or decembrie\fi
30.33 \space \number\year}}

```

`\extrasromanian` The macro `\extrasromanian` will perform all the extra definitions needed for the Romanian language. The macro `\noextrasromanian` is used to cancel the actions of `\extrasromanian`. For the moment these macros are empty but they are defined for compatibility with the other language definition files.

```

30.34 \addto\extrasromanian{}
30.35 \addto\noextrasromanian{}

```

The macro `\ldf@finish` takes care of looking for a configuration file, setting the main language to be switched on at `\begin{document}` and resetting the category code of `@` to its original value.

```

30.36 \ldf@finish{romanian}
30.37 \code

```

31 The Danish language

The file `danish.dtx`⁴⁵ defines all the language definition macros for the Danish language.

For this language the character " is made active. In table 11 an overview is given of its purpose.

"	disable ligature at this position.
"-	an explicit hyphen sign, allowing hyphenation in the rest of the word.
""	like "-", but producing no hyphen sign (for words that should break at some sign such as "entrada/salida.")
"‘	lowered double left quotes (looks like „)
"’	normal double right quotes
"<	for French left double quotes (similar to <<).
">	for French right double quotes (similar to >>).

Table 11: The extra definitions made by `danish.ldf`

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the `@` sign, etc.

31.1 `(*code)`

31.2 `\LdfInit{danish}\captionstdanish`

When this file is read as an option, i.e. by the `\usepackage` command, `danish` will be an ‘unknown’ language in which case we have to make it known. So we check for the existence of `\l@danish` to see whether we have to do something here.

31.3 `\ifx\l@danish\@undefined`

31.4 `\@nopatterns{Danish}`

31.5 `\adddialect\l@danish0\fi`

The next step consists of defining commands to switch to (and from) the Danish language.

`\captionstdanish` The macro `\captionstdanish` defines all strings used in the four standard documentclasses provided with L^AT_EX.

31.6 `\addto\captionstdanish{%`

31.7 `\def\prefacename{Forord}%`

31.8 `\def\refname{Litteratur}%`

31.9 `\def\abstractname{Resum\’e}%`

31.10 `\def\bibname{Litteratur}%`

31.11 `\def\chaptername{Kapitel}%`

31.12 `\def\appendixname{Bilag}%`

31.13 `\def\contentsname{Indhold}%`

31.14 `\def\listfigurename{Figurer}%`

31.15 `\def\listtablename{Tabeller}%`

31.16 `\def\indexname{Indeks}%`

31.17 `\def\figurename{Figur}%`

31.18 `\def\tablename{Tabel}%`

31.19 `\def\partname{Del}%`

⁴⁵The file described in this section has version number v1.3j and was last revised on 1996/12/23. A contribution was made by Henning Larsen (`larsen@cernvm.cern.ch`)

```

31.20 \def\enclname{Vedlagt}%
31.21 \def\ccname{Kopi til}% or Kopi sendt til
31.22 \def\headtoname{Til}% in letter
31.23 \def\pagename{Side}%
31.24 \def\seenname{Se}%
31.25 \def\alsoname{Se ogs{\aa}}%
31.26 \def\proofname{Bevis}%
31.27 }%

```

`\datedanish` The macro `\datedanish` redefines the command `\today` to produce Danish dates.

```

31.28 \def\datedanish{%
31.29 \def\today{\number\day.~\ifcase\month\or
31.30 januar\or februar\or marts\or april\or maj\or juni\or
31.31 juli\or august\or september\or oktober\or november\or december\fi
31.32 \space\number\year}}

```

`\extrasdanish` The macro `\extrasdanish` will perform all the extra definitions needed for the Danish language. The macro `\noextrasdanish` is used to cancel the actions of `\extrasdanish`.

Danish typesetting requires `\frenchspacing` to be in effect.

```

31.33 \addto\extrasdanish{\bbl@frenchspacing}
31.34 \addto\noextrasdanish{\bbl@nonfrenchspacing}

```

For Danish the " character is made active. This is done once, later on its definition may vary. Other languages in the same document may also use the " character for shorthands; we specify that the danish group of shorthands should be used.

```

31.35 \initiate@active@char{"}
31.36 \addto\extrasdanish{\languageshorthands{danish}}
31.37 \addto\extrasdanish{\bbl@activate{"}}
31.38 %\addto\noextrasdanish{\bbl@deactivate{"}}

```

First we define access to the low opening double quote and guillemets for quotations,

```

31.39 \declare@shorthand{danish}{"'}{%
31.40 \textormath{\quotedblbase{}}{\mbox{\quotedblbase}}}
31.41 \declare@shorthand{danish}{"'}{%
31.42 \textormath{\textquotedblleft{}}{\mbox{\textquotedblleft}}}
31.43 \declare@shorthand{danish}{"<"}{%
31.44 \textormath{\guillemotleft{}}{\mbox{\guillemotleft}}}
31.45 \declare@shorthand{danish}{">"}{%
31.46 \textormath{\guillemotright{}}{\mbox{\guillemotright}}}

```

then we define to be able to specify hyphenation breakpoints that behave a little different from \-.

```

31.47 \declare@shorthand{danish}{"-}{\allowhyphens-\allowhyphens}
31.48 \declare@shorthand{danish}{""}{\hskip\z@skip}
31.49 \declare@shorthand{danish}{ "~ }{\textormath{\leavevmode\hbox{-}}{-}}
31.50 \declare@shorthand{danish}{"=}{\penalty\@M-\hskip\z@skip}

```

And we want to have a shorthand for disabling a ligature.

```

31.51 \declare@shorthand{danish}{"|}{%
31.52 \textormath{\discretionary{-}{-}{\kern.03em}}{}}

```

The macro `\ldf@finish` takes care of looking for a configuration file, setting the main language to be switched on at `\begin{document}` and resetting the category code of `@` to its original value.

31.53 `\ldf@finish{danish}`

31.54 `\code`

32 The Norwegian language

The file `norsk.dtx`⁴⁶ defines all the language definition macros for the Norwegian language as well as for a new spelling variant ‘nynorsk’ for this language.

For this language currently no special definitions are needed or available.

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the `@` sign, etc.

```
32.1 (*code)
32.2 \LdfInit\CurrentOption{captions\CurrentOption}
```

When this file is read as an option, i.e. by the `\usepackage` command, `norsk` will be an ‘unknown’ language in which case we have to make it known. So we check for the existence of `\l@norsk` to see whether we have to do something here.

```
32.3 \ifx\l@norsk\undefined
32.4     \nopatterns{Norsk}
32.5     \addialect\l@norsk0\fi
```

`\norskhyphenmins` The Norwegian hyphenation patterns can be used with `\lefthyphenmin` set to 1 and `\righthyphenmin` set to 2. This is true for both ‘versions’ of the language.

```
32.6 \@namedef{\CurrentOption hyphenmins}{\@ne\tw@}
```

Now we have to decide which version of the captions should be made available. This can be done by checking the contents of `\CurrentOption`.

```
32.7 \def\bbl@tempa{norsk}
32.8 \ifx\CurrentOption\bbl@tempa
```

The next step consists of defining commands to switch to (and from) the Norwegian language.

`\captionsnorsk` The macro `\captionsnorsk` defines all strings used in the four standard documentclasses provided with L^AT_EX.

```
32.9 \def\captionsnorsk{%
32.10     \def\prefacename{Forord}%
32.11     \def\refname{Referanser}%
32.12     \def\abstractname{Sammendrag}%
32.13     \def\bibname{Bibliografi}%           or Litteraturoversikt
32.14 %                                     or Litteratur or Referanser
32.15     \def\chaptername{Kapittel}%
32.16     \def\appendixname{Tillegg}%         or Appendix
32.17     \def\contentsname{Innhold}%
32.18     \def\listfigurename{Figurer}%      or Figurliste
32.19     \def\listtablename{Tabeller}%      or Tabelliste
32.20     \def\indexname{Register}%
32.21     \def\figurename{Figur}%
32.22     \def\tablename{Tabell}%
32.23     \def\partname{Del}%
32.24     \def\enclname{Vedlegg}%
32.25     \def\ccname{Kopi sendt}%
32.26     \def\headtoname{Til}% in letter
```

⁴⁶The file described in this section has version number v1.2h and was last revised on 1996/12/23. Contributions were made by Haavard Helstrup (HAAVARD@CERNVM) and Alv Kjetil Holme (HOLMEA@CERNVM); the ‘nynorsk’ variant has been supplied by Per Steinar Iversen (iversen@vxcern.cern.ch) and Terje Engeset Petterst (TERJEEP@VSFYS1.FI.UIB.NO).


```

32.27 \def\pagename{Side}%
32.28 \def\seename{Se}%
32.29 \def\alsoname{Se ogs\aa{}}%
32.30 \def\proofname{Bevis}%
32.31 }
32.32 \else

```

For the ‘nynorsk’ version of these definitions we just add a “dialect”.

```

32.33 \addialect\l@nynorsk\l@norsk

```

`\captionesnynorsk` The macro `\captionesnynorsk` defines all strings used in the four standard documentclasses provided with L^AT_EX, but using a different spelling than in the command `\captionesnorsk`.

```

32.34 \def\captionesnynorsk{%
32.35 \def\prefacename{Forord}%
32.36 \def\refname{Referansar}%
32.37 \def\abstractname{Samandrag}%
32.38 \def\bibname{Litteratur}% or Litteraturoversyn
32.39 % % or Referansar
32.40 \def\chaptername{Kapittel}%
32.41 \def\appendixname{Tillegg}% or Appendiks
32.42 \def\contentsname{Innhald}%
32.43 \def\listfigurename{Figurar}% or Figurliste
32.44 \def\listtablename{Tabellar}% or Tabelliste
32.45 \def\indexname{Register}%
32.46 \def\figurename{Figur}%
32.47 \def\tablename{Tabell}%
32.48 \def\partname{Del}%
32.49 \def\enclname{Vedlegg}%
32.50 \def\ccname{Kopi sendt}%
32.51 \def\headtoname{Til}% in letter
32.52 \def\pagename{Side}%
32.53 \def\seename{Sj\aa{}}%
32.54 \def\alsoname{Sj\aa{} ogs\aa{}}%
32.55 \def\proofname{Bevis}%
32.56 }
32.57 \fi

```

`\datenorsk` The macro `\datenorsk` redefines the command `\today` to produce Norwegian dates.

```

32.58 \@namedef{date\CurrentOption}{%
32.59 \def\today{\number\day.\~\ifcase\month\or
32.60 januar\or februar\or mars\or april\or mai\or juni\or
32.61 juli\or august\or september\or oktober\or november\or desember
32.62 \fi
32.63 \space\number\year}}

```

`\extrasnorsk` The macro `\extrasnorsk` will perform all the extra definitions needed for the Norwegian language. The macro `\noextrasnorsk` is used to cancel the actions of `\extrasnorsk`.

Norwegian typesetting requires `\frenchspacing` to be in effect.

```

32.64 \@namedef{extras\CurrentOption}{\bbl@frenchspacing}
32.65 \@namedef{noextras\CurrentOption}{\bbl@nonfrenchspacing}

```

The macro `\ldf@finish` takes care of looking for a configuration file, setting the main language to be switched on at `\begin{document}` and resetting the category code of `@` to its original value.

32.66 `\ldf@finish\CurrentOption`

32.67 `\code`

33 The Swedish language

The file `swedish.dtx`⁴⁷ defines all the language-specific macros for the Swedish language.

For this language the character " is made active. In table 12 an overview is given of its purpose. The vertical placement of the "umlaut" in some letters can be controlled this way.

"a	\ "a, also implemented for Å, ö and Ö.
"w	gives å, also works for uppercase letters.
"ff	for ff to be hyphenated as ff-f, this is also implemented for b, d, f, g, l, m, n, p, r, s, and t.
"	disable ligature at this position.
"-	an explicit hyphen sign, allowing hyphenation in the rest of the word.
""	like "-", but producing no hyphen sign (for compound words with hyphen, e.g. x-"y).
"~	for a compound word mark without a breakpoint.
"=	for a compound word mark with a breakpoint, allowing hyphenation in the composing words.

Table 12: The extra definitions made by `swedish.sty`

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the @ sign, etc.

33.1 (*code)

33.2 `\LdfInit{swedish}\captionsswedish`

When this file is read as an option, i.e. by the `\usepackage` command, `swedish` will be an 'unknown' language in which case we have to make it known. So we check for the existence of `\l@swedish` to see whether we have to do something here.

33.3 `\ifx\l@swedish\@undefined`

33.4 `\@nopatterns{Swedish}`

33.5 `\adddialect\l@swedish0\fi`

The next step consists of defining commands to switch to the Swedish language. The reason for this is that a user might want to switch back and forth between languages.

`\captionsswedish` The macro `\captionsswedish` defines all strings used in the four standard documentclasses provided with L^AT_EX.

33.6 `\addto\captionsswedish{%`

33.7 `\def\prefacename{F"orord}%`

33.8 `\def\refname{Referenser}%`

33.9 `\def\abstractname{Sammanfattning}%`

33.10 `\def\bibname{Litteraturf"orteckning}%`

33.11 `\def\chaptername{Kapitel}%`

⁴⁷The file described in this section has version number v2.1 and was last revised on 1996/12/23. Contributions were made by Sten Hellman (HELLMAN@CERNVM.CERN.CH) and Erik Öshols (erik@ine.kfk.de).

```

33.12 \def\appendixname{Bilaga}%
33.13 \def\contentsname{Inneh\csname aa\endcsname ll}%
33.14 \def\listfigurename{Figurer}%
33.15 \def\listtablename{Tabeller}%
33.16 \def\indexname{Sakregister}%
33.17 \def\figurename{Figur}%
33.18 \def\tablename{Tabell}%
33.19 \def\partname{Del}%
33.20 \def\enclname{Bil}%
33.21 \def\ccname{Kopia f\ "or k\ "annedom}%
33.22 \def\headtoname{Till}% in letter
33.23 \def\pagename{Sida}%
33.24 \def\seename{se}%

33.25 \def\alsoname{se \ "aven}%

33.26 \def\proofname{Bevis}%
33.27 }%

```

`\dateswedish` The macro `\dateswedish` redefines the command `\today` to produce Swedish dates.

```

33.28 \def\dateswedish{%
33.29   \def\today{%
33.30     \number\day~\ifcase\month\or
33.31     januari\or februari\or mars\or april\or maj\or juni\or
33.32     juli\or augusti\or september\or oktober\or november\or
33.33     december\fi
33.34     \space\number\year}}

```

`\swedishhyphenmins` The swedish hyphenation patterns can be used with `\lefthyphenmin` set to 2 and `\righthyphenmin` set to 2.

```

33.35 \def\swedishhyphenmins{\tw@\tw@}

```

`\extrasswedish` The macro `\extrasswedish` performs all the extra definitions needed for the Swedish language. The macro `\noextrasswedish` is used to cancel the actions of `\extrasswedish`.

For Swedish texts `\frenchspacing` should be in effect. We make sure this is the case and reset it if necessary.

```

33.36 \addto\extrasswedish{\bbl@frenchspacing}
33.37 \addto\noextrasswedish{\bbl@nonfrenchspacing}

```

For Swedish the " character is made active. This is done once, later on its definition may vary.

```

33.38 \initiate@active@char{"}
33.39 \addto\extrasswedish{\languageshortands{swedish}}
33.40 \addto\extrasswedish{\bbl@activate{"}}
33.41 %\addto\noextrasswedish{\bbl@deactivate{"}}

```

The “umlaut” accent macro `\` is changed to lower the umlaut dots. The redefinition is done with the help of `\umlautlow`.

```

33.42 \addto\extrasswedish{\babel@save\ "\umlautlow}
33.43 \addto\noextrasswedish{\umlauthigh}

```

The code above is necessary because we need an extra active character. This character is then used as indicated in table 12.

To be able to define the function of " , we first define a couple of ‘support’ macros.

`\dq` We save the original double quote character in `\dq` to keep it available, the math accent `\"` can now be typed as `"`.

```
33.44 \begingroup \catcode'\ "12
33.45 \def\x{\endgroup
33.46 \def\@SS{\mathchar"7019 }
33.47 \def\dq{"}}
33.48 \x
```

Now we can define the doublequote macros: the umlauts and å,

```
33.49 \declare@shorthand{swedish}{"w}{\textormath{\aa}{\ddot w}}
33.50 \declare@shorthand{swedish}{"a}{\textormath{"{a}{\ddot a}}
33.51 \declare@shorthand{swedish}{"o}{\textormath{"{o}{\ddot o}}
33.52 \declare@shorthand{swedish}{"W}{\textormath{\AA}{\ddot W}}
33.53 \declare@shorthand{swedish}{"A}{\textormath{"{A}{\ddot A}}
33.54 \declare@shorthand{swedish}{"O}{\textormath{"{O}{\ddot O}}
```

discretionary commands

```
33.55 \declare@shorthand{swedish}{"b}{\textormath{\bbl@disc b{bb}}{b}}
33.56 \declare@shorthand{swedish}{"B}{\textormath{\bbl@disc B{BB}}{B}}
33.57 \declare@shorthand{swedish}{"d}{\textormath{\bbl@disc d{dd}}{d}}
33.58 \declare@shorthand{swedish}{"D}{\textormath{\bbl@disc D{DD}}{D}}
33.59 \declare@shorthand{swedish}{"f}{\textormath{\bbl@disc f{ff}}{f}}
33.60 \declare@shorthand{swedish}{"F}{\textormath{\bbl@disc F{FF}}{F}}
33.61 \declare@shorthand{swedish}{"g}{\textormath{\bbl@disc g{gg}}{g}}
33.62 \declare@shorthand{swedish}{"G}{\textormath{\bbl@disc G{GG}}{G}}
33.63 \declare@shorthand{swedish}{"l}{\textormath{\bbl@disc l{ll}}{l}}
33.64 \declare@shorthand{swedish}{"L}{\textormath{\bbl@disc L{LL}}{L}}
33.65 \declare@shorthand{swedish}{"m}{\textormath{\bbl@disc m{mm}}{m}}
33.66 \declare@shorthand{swedish}{"M}{\textormath{\bbl@disc M{MM}}{M}}
33.67 \declare@shorthand{swedish}{"n}{\textormath{\bbl@disc n{nn}}{n}}
33.68 \declare@shorthand{swedish}{"N}{\textormath{\bbl@disc N{NN}}{N}}
33.69 \declare@shorthand{swedish}{"p}{\textormath{\bbl@disc p{pp}}{p}}
33.70 \declare@shorthand{swedish}{"P}{\textormath{\bbl@disc P{PP}}{P}}
33.71 \declare@shorthand{swedish}{"r}{\textormath{\bbl@disc r{rr}}{r}}
33.72 \declare@shorthand{swedish}{"R}{\textormath{\bbl@disc R{RR}}{R}}
33.73 \declare@shorthand{swedish}{"s}{\textormath{\bbl@disc s{ss}}{s}}
33.74 \declare@shorthand{swedish}{"S}{\textormath{\bbl@disc S{SS}}{S}}
33.75 \declare@shorthand{swedish}{"t}{\textormath{\bbl@disc t{tt}}{t}}
33.76 \declare@shorthand{swedish}{"T}{\textormath{\bbl@disc T{TT}}{T}}
```

and some additional commands:

```
33.77 \declare@shorthand{swedish}{"-}{\penalty\@M-\allowhyphens}
33.78 \declare@shorthand{swedish}{"|}{%
33.79 \textormath{\penalty\@M\discretionary{-}{\kern.03em}%
33.80 \allowhyphens}{}}
33.81 \declare@shorthand{swedish}{"~}{\hskip\z@skip}
33.82 \declare@shorthand{swedish}{"~}{\textormath{\leavevmode\hbox{-}}{-}}
33.83 \declare@shorthand{swedish}{"=}{\penalty\@M-\hskip\z@skip}
```

The macro `\ldf@finish` takes care of looking for a configuration file, setting the main language to be switched on at `\begin{document}` and resetting the category code of `@` to its original value.

33.84 `\ldf@finish{swedish}`

33.85 `\end{code}`

34 The Finnish language

The file `finnish.dtx`⁴⁸ defines all the language definition macros for the Finnish language.

For this language the character " is made active. In table 13 an overview is given of its purpose.

"	disable ligature at this position.
"-	an explicit hyphen sign, allowing hyphenation in the rest of the word.
"=	an explicit hyphen sign for expressions such as "pakastekaapit ja -arkut".
""	like "-", but producing no hyphen sign (for words that should break at some sign such as "entrada/salida."
"‘	lowered double left quotes (looks like „)
"’	normal double right quotes
"<	for French left double quotes (similar to <<).
">	for French right double quotes (similar to >>).
\-	like the old \-, but allowing hyphenation in the rest of the word.

Table 13: The extra definitions made by `finnish.ldf`

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the `@` sign, etc.

34.1 `(*code)`

34.2 `\LdfInit{finnish}\captionsfinnish`

When this file is read as an option, i.e. by the `\usepackage` command, `finnish` will be an ‘unknown’ language in which case we have to make it known. So we check for the existence of `\l@finnish` to see whether we have to do something here.

34.3 `\ifx\l@finnish\@undefined`

34.4 `\@nopatterns{Finnish}`

34.5 `\adddialect\l@finnish0\fi`

The next step consists of defining commands to switch to the Finnish language. The reason for this is that a user might want to switch back and forth between languages.

`\captionsfinnish` The macro `\captionsfinnish` defines all strings used in the four standard documentclasses provided with L^AT_EX.

34.6 `\addto\captionsfinnish{%`

34.7 `\def\prefacename{Esipuhe}%`

34.8 `\def\refname{Viitteet}%`

34.9 `\def\abstractname{Tiivistelmä}`

34.10 `\def\bibname{Kirjallisuutta}%`

34.11 `\def\chaptername{Luku}%`

34.12 `\def\appendixname{Liite}%`

⁴⁸The file described in this section has version number v1.3j and was last revised on 1997/01/23. A contribution was made by Mikko KANERVA (`KANERVA@CERNVM`) and Keranen Reino (`KERANEN@CERNVM`).

```

34.13 \def\contentsname{Sis\alt{o}}% /* Could be "Sis\allys" as well */
34.14 \def\listfigurename{Kuvat}%
34.15 \def\listtablename{Taulukot}%
34.16 \def\indexname{Hakemisto}%
34.17 \def\figurename{Kuva}%
34.18 \def\tablename{Taulukko}%
34.19 \def\partname{Osa}%
34.20 \def\enclname{Liitteet}%
34.21 \def\ccname{Jakelu}%
34.22 \def\headtoname{Vastaanottaja}%
34.23 \def\pagename{Sivu}%
34.24 \def\seename{katso}%
34.25 \def\alsoname{katso my\os}%
34.26 \def\proofname{Todistus}%
34.27 }%

```

`\datefinnish` The macro `\datefinnish` redefines the command `\today` to produce Finnish dates.

```

34.28 \def\datefinnish{%
34.29 \def\today{\number\day.~\ifcase\month\or
34.30 tammikuuta\or helmikuuta\or maaliskuuta\or huhtikuuta\or
34.31 toukokuuta\or kes\akuuta\or hein\akuuta\or elokuuta\or
34.32 syyskuuta\or lokakuuta\or marraskuuta\or joulukuuta\fi
34.33 \space\number\year}}

```

`\extrasfinnish` Finnish has many long words (some of them compound, some not). For this reason hyphenation is very often the only solution in line breaking. For this reason the values of `\hyphenpenalty`, `\exhyphenpenalty` and `\doublehyphendemerits` should be decreased. (In one of the manuals of style Matti Rintala noticed a paragraph with ten lines, eight of which ended in a hyphen!)

Matti Rintala noticed that with these changes $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ handles Finnish very well, although sometimes the values of `\tolerance` and `\emergencystretch` must be increased. However, I don't think changing these values in `finnish.ldf` is appropriate, as the looseness of the font (and the line width) affect the correct choice of these parameters.

```

34.34 \addto\extrasfinnish{%
34.35 \babel@savevariable\hyphenpenalty\hyphenpenalty=30%
34.36 \babel@savevariable\exhyphenpenalty\exhyphenpenalty=30%
34.37 \babel@savevariable\doublehyphendemerits\doublehyphendemerits=5000%
34.38 \babel@savevariable\finalhyphendemerits\finalhyphendemerits=5000%
34.39 }
34.40 \addto\noextrasfinnish{}

```

Another thing `\extrasfinnish` needs to do is to make sure that `\frenchspacing` is in effect. If this is not the case the execution of `\noextrasfinnish` will switch it off again.

```

34.41 \addto\extrasfinnish{\bbl@frenchspacing}
34.42 \addto\noextrasfinnish{\bbl@nonfrenchspacing}

```

For Finnish the " character is made active. This is done once, later on its definition may vary. Other languages in the same document may also use the " character for shorthands; we specify that the finnish group of shorthands should be used.


```

34.43 \initiate@active@char{"}
34.44 \addto\extrasfinnish{\languageshorthands{finnish}}
34.45 \addto\extrasfinnish{\bbl@activate{}}
34.46 %\addto\noextrasfinnish{\bbl@deactivate{}}

```

The ‘umlaut’ character should be positioned lower on *all* vowels in Finnish texts.

```

34.47 \addto\extrasfinnish{\umlautlow\umlautelow}
34.48 \addto\noextrasfinnish{\umlauthigh}

```

First we define access to the low opening double quote and guillemets for quotations,

```

34.49 \declare@shorthand{finnish}{"'}{%
34.50   \textormath{\quotedblbase{}}{\mbox{\quotedblbase}}}
34.51 \declare@shorthand{finnish}{"'}{%
34.52   \textormath{\textquotedblright{}}{\mbox{\textquotedblright}}}
34.53 \declare@shorthand{finnish}{"<"}{%
34.54   \textormath{\guillemotleft{}}{\mbox{\guillemotleft}}}
34.55 \declare@shorthand{finnish}{">"}{%
34.56   \textormath{\guillemotright{}}{\mbox{\guillemotright}}}

```

then we define two shorthands to be able to specify hyphenation breakpoints that behave a little different from \-.

```

34.57 \declare@shorthand{finnish}{"-"}{\allowhyphens-\allowhyphens}
34.58 \declare@shorthand{finnish}{""}{\hskip\z@skip}
34.59 \declare@shorthand{finnish}{"="}{\hbox{-}\allowhyphens}

```

And we want to have a shorthand for disabling a ligature.

```

34.60 \declare@shorthand{finnish}{"|"}{%
34.61   \textormath{\discretionary{-}{-}{\kern.03em}}{}}

```

- \- All that is left now is the redefinition of \-. The new version of \- should indicate an extra hyphenation position, while allowing other hyphenation positions to be generated automatically. The standard behaviour of T_EX in this respect is very unfortunate for languages such as Dutch, Finnish and German, where long compound words are quite normal and all one needs is a means to indicate an extra hyphenation position on top of the ones that T_EX can generate from the hyphenation patterns.

```

34.62 \addto\extrasfinnish{\babel@save\-\}
34.63 \addto\extrasfinnish{\def\-\{\allowhyphens
34.64   \discretionary{-}{-}{\allowhyphens}}

```

\finishhyphenmins The finnish hyphenation patterns can be used with \lefthyphenmin set to 2 and \righthyphenmin set to 2.

```

34.65 \def\finnishhyphenmins{\tw@\tw@}

```

The macro \ldf@finish takes care of looking for a configuration file, setting the main language to be switched on at \begin{document} and resetting the category code of @ to its original value.

```

34.66 \ldf@finish{finnish}
34.67 \code

```

35 The Hungarian language

The file option `magyar.dtx`⁴⁹ defines all the language definition macros for the Hungarian language.

`\ontoday` For this language currently the only special definition that is added is the `\ontoday` command which works like `\today` but produces a slightly different date format used in expressions such as ‘on february 10th’.

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the `@` sign, etc.

```
35.1 (*code)
35.2 \LdfInit{magyar}{caption\CurrentOption}
```

When this file is read as an option, i.e. by the `\usepackage` command, `magyar` will be an ‘unknown’ language in which case we have to make it known. So we check for the existence of `\l@magyar` or `\l@hungarian` to see whether we have to do something here.

```
35.3 \ifx\l@magyar\@undefined
35.4   \ifx\l@hungarian\@undefined
35.5     \nopatterns{Magyar}
35.6     \adddialect\l@magyar0
35.7   \fi
35.8 \fi
35.9 \let\l@hungarian\l@magyar
```

An additional note about formatting Hungarian texts: One should invert the order of the number and text in things like chapter headings, page references etc. So one should write ‘I. rész’ instead of ‘Part I’, or ‘3. oldal’ for ‘page 3’.

For chapter headings this could be accomplished by a redefinition of the macros `\@makechapterhead` and `\@makeschapterhead`, for other instances this is a lot harder to accomplish. Therefore I think complete document classes should be written to accommodate the needed formatting.

The next step consists of defining commands to switch to (and from) the Hungarian language.

`\captionsmagyar` The macro `\captionsmagyar` defines all strings used in the four standard documentclasses provided with L^AT_EX.

```
35.10 \@namedef{captions\CurrentOption}{%
35.11   \def\prefacename{Előszó}%
```

For the list of references at the end of an article we have a choice between two words, ‘Referenciák’ (a Hungarian version of the English word) and ‘Hivatkozások’. The latter seems to be in more widespread use.

```
35.12   \def\refname{Hivatkozások}%
```

If you have a document with a summary instead of an abstract you might want to replace the word ‘Kivonat’ with ‘Összefoglaló’.

```
35.13   \def\abstractname{Kivonat}%
```

⁴⁹The file described in this section has version number v1.3h and was last revised on 1996/12/23. A contribution was made by Attila Koppanyi (attila@cernvm.cern.ch). Later updates and suggestions by Árpád Bíró (JZP1104@HUSZEG11.bitnet), Istvan Hamecz (hami@ursus.bke.hu) and Horvath Dezso (horvath@pisa.infn.it).

The Hungarian version of ‘Bibliography’ is ‘Bibliográfia’, but a more natural word to use is ‘Irodalomjegyzék’.

```

35.14 \def\bibName{Irodalomjegyz\'ek}%
35.15 \def\chaptername{Fejezet}%
35.16 \def\appendixname{F\'uggel\'ek}%
35.17 \def\contentsname{Tartalomjegyz\'ek}%
35.18 \def\listfigurename{\'Abr\'ak jegyz\'eke}%
35.19 \def\listtablename{T\'abl\'azatok jegyz\'eke}%
35.20 \def\indexname{T\'argymutat\'o}%
35.21 \def\figurename{\'abra}%
35.22 \def\tablename{T\'abl\'azat}%
35.23 \def\partname{R\'esz}%
35.24 \def\enclname{Mell\'eklet}%
35.25 \def\ccname{K\'orlev\'el--c\'i mzetek}%
35.26 \def\headtoname{C\'i mzett}%
35.27 \def\pagename{oldal}%
35.28 \def\seename{L\'asd}%
35.29 \def\alsoname{L\'asd m\'eg}%

```

Besides the Hungarian word for Proof, ‘Bizonyítás’ we can also name Corollary (Következmény), Theorem (Tétel) and Lemma (Lemma).

```

35.30 \def\proofname{Bizony\'i t\'as}%
35.31 }%

```

`\datemagyar` The macro `\datemagyar` redefines the command `\today` to produce Hungarian dates.

```

35.32 \@namedef{date\CurrentOption}{%
35.33 \def\today{\number\year.\~\ifcase\month\or
35.34 janu\'ar\or febru\'ar\or m\'arcius\or
35.35 \'aprilis\or m\'ajus\or j\'unius\or
35.36 j\'ulius\or augusztus\or szeptember\or
35.37 okt\'ober\or november\or december\fi
35.38 \space\ifcase\day\or
35.39 1.\or 2.\or 3.\or 4.\or 5.\or
35.40 6.\or 7.\or 8.\or 9.\or 10.\or
35.41 11.\or 12.\or 13.\or 14.\or 15.\or
35.42 16.\or 17.\or 18.\or 19.\or 20.\or
35.43 21.\or 22.\or 23.\or 24.\or 25.\or
35.44 26.\or 27.\or 28.\or 29.\or 30.\or
35.45 31.\fi}}

```

`\ondatemagyar` The macro `\ondatemagyar` produces Hungarian dates which have the meaning ‘*on this day*’. It does not redefine the command `\today`.

```

35.46 \@namedef{ondate\CurrentOption}{%
35.47 \number\year.\~\ifcase\month\or
35.48 janu\'ar\or febru\'ar\or m\'arcius\or
35.49 \'aprilis\or m\'ajus\or j\'unius\or
35.50 j\'ulius\or augusztus\or szeptember\or
35.51 okt\'ober\or november\or december\fi
35.52 \space\ifcase\day\or
35.53 1-j\'en\or 2-\'an\or 3-\'an\or 4-\'en\or 5-\'en\or
35.54 6-\'an\or 7-\'en\or 8-\'an\or 9-\'en\or 10-\'en\or
35.55 11-\'en\or 12-\'en\or 13-\'an\or 14-\'en\or 15-\'en\or
35.56 16-\'an\or 17-\'en\or 18-\'an\or 19-\'en\or 20-\'an\or

```

```

35.57 21-\'en\or 22-\'en\or 23-\'an\or 24-\'en\or 25-\'en\or
35.58 26-\'an\or 27-\'en\or 28-\'an\or 29-\'en\or 30-\'an\or
35.59 31-\'en\fi}

```

`\extrasmagyar` The macro `\extrasmagyar` will perform all the extra definitions needed for the Hungarian language. The macro `\noextrasmagyar` is used to cancel the actions of `\extrasmagyar`. For the moment these macros are nearly empty; only the user command `\ontoday` to access `\ondatemagyar` is defined.

```

35.60 \@namedef{extras\CurrentOption}{%
35.61   \expandafter\let\expandafter\ontoday
35.62     \csname ondate\CurrentOption\endcsname}
35.63 \@namedef{noextras\CurrentOption}{\let\ontoday\@undefined}

```

The macro `\ldf@finish` takes care of looking for a configuration file, setting the main language to be switched on at `\begin{document}` and resetting the category code of `@` to its original value.

```

35.64 \ldf@finish\CurrentOption
35.65 \code

```

36 The Estonian language

The file `estonian.dtx`⁵⁰ defines the language definition macro's for the Estonian language.

This file was written as part of the TWGML project, and borrows heavily from the `babel` German and Spanish language files `germanb.ldf` and `spanish.ldf`.

Estonian has the same umlauts as German (ä, ö, ü), but in addition to this, we have also õ, and two recent characters š and ž, so we need at least two active characters. We shall use " and ~ to type Estonian accents on ASCII keyboards (in the 7-bit character world). Their use is given in table 14. These active accent

~o	\~o, (and uppercase);
"a	\"a, (and uppercase);
"o	\"o, (and uppercase);
"u	\"u, (and uppercase);
~s	\v s, (and uppercase);
~z	\v z, (and uppercase);
"	disable ligature at this position;
"-	an explicit hyphen sign, allowing hyphenation in the rest of the word;
\-	like the old \-, but allowing hyphenation in the rest of the word;
"‘	for Estonian low left double quotes (same as German);
"’	for Estonian right double quotes;
"<	for French left double quotes (also rather popular)
">	for French right double quotes.

Table 14: The extra definitions made by `estonian.ldf`

characters behave according to their original definitions if not followed by one of the characters indicated in that table; the original quote character can be typed using the macro `\dq`.

We support also the T1 output encoding (and Cork-encoded text input). You can choose the T1 encoding by the command `\usepackage[T1]{fontenc}`. This package must be loaded before `babel`. As the standard Estonian hyphenation file `eehyph.tex` is in the Cork encoding, choosing this encoding will give you better hyphenation.

As mentioned in the Spanish style file, it may happen that some packages fail (usually in a `\message`). In this case you should change the order of the `\usepackage` declarations or the order of the style options in `\documentclass`.

36.1 Implementation

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the `@` sign, etc.

36.1 (*code)

⁵⁰The file described in this section has version number v1.0e and was last revised on 1996/12/23. The original author is Enn Saar, (saar@aai.ee).

36.2 `\LdfInit{estonian}\captionsestonian`

If Estonian is not included in the format file (does not have hyphenation patterns), we shall use English hyphenation.

36.3 `\ifx\l@estonian\@undefined`
 36.4 `\@nopatterns{Estonian}`
 36.5 `\adddialect\l@estonian0`
 36.6 `\fi`

Now come the commands to switch to (and from) Estonian.

`\captionsestonian` The macro `\captionsestonian` defines all strings used in the four standard documentclasses provided with L^AT_EX.

36.7 `\addto\captionsestonian{%`
 36.8 `\def\prefacename{Sissejuhatus}%`
 36.9 `\def\refname{Viited}%`
 36.10 `\def\bibname{Kirjandus}%`
 36.11 `\def\appendixname{Lisa}%`
 36.12 `\def\contentsname{Sisukord}%`
 36.13 `\def\listfigurename{Joonised}%`
 36.14 `\def\listtablename{Tabelid}%`
 36.15 `\def\indexname{Indeks}%`
 36.16 `\def\figurename{Joonis}%`
 36.17 `\def\tablename{Tabel}%`
 36.18 `\def\partname{Osa}%`
 36.19 `\def\enclname{Lisa(d)}%`
 36.20 `\def\ccname{Koopia(d)}%`
 36.21 `\def\headtoname{}%`
 36.22 `\def\pagename{Lk.}%`
 36.23 `\def\seename{vt.}%`
 36.24 `\def\alsoname{vt. ka}%`
 36.25 `\def\proofname{Korrektuur}%`
 36.26 `}`

These captions contain accented characters.

36.27 `\begingroup \catcode'\\"active`
 36.28 `\def\x{\endgroup`
 36.29 `\addto\captionsestonian{%`
 36.30 `\def\abstractname{Kokkuvõte}%`
 36.31 `\def\chaptername{Peat"ukk}}}`
 36.32 `\x`

`\dateestonian` The macro `\dateestonian` redefines the command `\today` to produce Estonian dates.

36.33 `\begingroup \catcode'\\"active`
 36.34 `\def\x{\endgroup`
 36.35 `\def\month@estonian{\ifcase\month\or`
 36.36 `jaanuar\or veebruar\or m"arts\or aprill\or mai\or juuni\or`
 36.37 `juuli\or august\or september\or oktoober\or november\or`
 36.38 `detsember\fi}}`
 36.39 `\x`
 36.40 `\def\dateestonian{\def\today{\number\day.\space\month@estonian`
 36.41 `\space\number\year.\space a.}}`

`\extrasestonian` The macro `\extrasestonian` will perform all the extra definitions needed for
`\noextrasestonian` Estonian. The macro `\noextrasestonian` is used to cancel the actions of
`\extrasestonian`. For Estonian, " is made active and has to be treated as 'special'
(~ is active already).

```
36.42 \initiate@active@char{"}
36.43 \initiate@active@char{~}
36.44 \addto\extrasestonian{\languageshorthands{estonian}}
36.45 \addto\extrasestonian{\bbl@activate{"}\bbl@activate{~}}
```

Store the original macros, and redefine accents.

```
36.46 \addto\extrasestonian{\babel@save"\umlaulow\babel@save"\tildelow}
```

Estonian does not use extra spaces after sentences.

```
36.47 \addto\extrasestonian{\bbl@frenchspacing}
36.48 \addto\noextrasestonian{\bbl@nonfrenchspacing}
```

`\estonianhyphenmins` For Estonian, `\lefthyphenmin` and `\righthyphenmin` are both 2.

```
36.49 \def\estonianhyphenmins{\tw@\tw@}
```

`\tildelow` The standard T_EX accents are too high for Estonian typography, we have to lower
`\gentilde` them (following the babel German style). For a detailed explanation see the file
`\newtilde` `glyphs.dtx`.

```
\newcheck.36.50 \def\tildelow{\def~{\protect\gentilde}}
36.51 \def\gentilde#1{\if#1o\newtilde{#1}\else\if#10\newtilde{#1}%
36.52 \else\newcheck{#1}%
36.53 \fi\fi}
36.54 \def\newtilde#1{\leavevmode\allowhyphens
36.55 {\U@D 1ex%
36.56 {\setbox\z@\hbox{\char126}\dimen@ -.45ex\advance\dimen@\ht\z@
36.57 \ifdim 1ex<\dimen@ \fontdimen5\font\dimen@ \fi}%
36.58 \accent126\fontdimen5\font\U@D #1}\allowhyphens}
36.59 \def\newcheck#1{\leavevmode\allowhyphens
36.60 {\U@D 1ex%
36.61 {\setbox\z@\hbox{\char20}\dimen@ -.45ex\advance\dimen@\ht\z@
36.62 \ifdim 1ex<\dimen@ \fontdimen5\font\dimen@ \fi}%
36.63 \accent20\fontdimen5\font\U@D #1}\allowhyphens}
```

We save the double quote character in `\dq`, and tilde in `\til`, and store the
original definitions of " and ~ as `\dieresis` and `\texttilde`.

```
36.64 \begingroup \catcode'"12
36.65 \edef\x{\endgroup
36.66 \def\noexpand\dq{"}
36.67 \def\noexpand\til{~}}
36.68 \x
36.69 \let\dieresis"
36.70 \let\texttilde~
```

This part follows closely `spanish.ldf`. We check the encoding and if it is T1,
we have to tell T_EX about our redefined accents.

```
36.71 \edef\next{T1}
36.72 \ifx\f@encoding\next
36.73 \let@umlaul\dieresis
```

```

36.74 \let\@tilde\texttilde
36.75 \DeclareTextComposite{\~}{T1}{s}{178}
36.76 \DeclareTextComposite{\~}{T1}{S}{146}
36.77 \DeclareTextComposite{\~}{T1}{z}{186}
36.78 \DeclareTextComposite{\~}{T1}{Z}{154}
36.79 \DeclareTextComposite{\~}{T1}{'}{17}
36.80 \DeclareTextComposite{\~}{T1}{'}{18}
36.81 \DeclareTextComposite{\~}{T1}{<}{19}
36.82 \DeclareTextComposite{\~}{T1}{>}{20}

```

If the encoding differs from T1, we expand the accents, enabling hyphenation beyond the accent. In this case T_EX will not find all possible breaks, and we have to warn people.

```

36.83 \else
36.84 \wlog{Warning: Hyphenation would work better for the T1 encoding.}
36.85 \let\@umlaut\newumlaut
36.86 \let\@tilde\gentilde
36.87 \fi

```

Now we define the shorthands.

```

36.88 \declare@shorthand{estonian}{a}{\textormath{"a}{\ddot a}}
36.89 \declare@shorthand{estonian}{A}{\textormath{"A}{\ddot A}}
36.90 \declare@shorthand{estonian}{o}{\textormath{"o}{\ddot o}}
36.91 \declare@shorthand{estonian}{O}{\textormath{"O}{\ddot O}}
36.92 \declare@shorthand{estonian}{u}{\textormath{"u}{\ddot u}}
36.93 \declare@shorthand{estonian}{U}{\textormath{"U}{\ddot U}}

```

german and french quotes,

```

36.94 \declare@shorthand{estonian}{'}{%
36.95 \textormath{\quotedblbase}}{\mbox{\quotedblbase}}
36.96 \declare@shorthand{estonian}{'}{%
36.97 \textormath{\textquotedblleft}}{\mbox{\textquotedblleft}}
36.98 \declare@shorthand{estonian}{<}{%
36.99 \textormath{\guillemotleft}}{\mbox{\guillemotleft}}
36.100 \declare@shorthand{estonian}{>}{%
36.101 \textormath{\guillemotright}}{\mbox{\guillemotright}}
36.102 \declare@shorthand{estonian}{~o}{\textormath{\@tilde o}{\tilde o}}
36.103 \declare@shorthand{estonian}{~O}{\textormath{\@tilde O}{\tilde O}}
36.104 \declare@shorthand{estonian}{~s}{\textormath{\@tilde s}{\check s}}
36.105 \declare@shorthand{estonian}{~S}{\textormath{\@tilde S}{\check S}}
36.106 \declare@shorthand{estonian}{~z}{\textormath{\@tilde z}{\check z}}
36.107 \declare@shorthand{estonian}{~Z}{\textormath{\@tilde Z}{\check Z}}

```

and some additional commands:

```

36.108 \declare@shorthand{estonian}{-}{\allowhyphens\-\allowhyphens}
36.109 \declare@shorthand{estonian}{|}{%
36.110 \textormath{\penalty\@M\discretionary{-}{-}{\kern.03em}%
36.111 \allowhyphens}}
36.112 \declare@shorthand{estonian}{"}{\dq}
36.113 \declare@shorthand{estonian}{~}{\til}

```

The macro \ldf@finish takes care of looking for a configuration file, setting the main language to be switched on at \begin{document} and resetting the category code of @ to its original value.

36.114 \ldf@finish{estonian}
36.115 \end{code}

37 The Croatian language

The file `croatian.dtx`⁵¹ defines all the language definition macros for the Croatian language.

For this language currently no special definitions are needed or available.

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the `@` sign, etc.

```
37.1 (*code)
37.2 \LdfInit{croatian}\captionscroatian
```

When this file is read as an option, i.e. by the `\usepackage` command, `croatian` will be an ‘unknown’ language in which case we have to make it known. So we check for the existence of `\l@croatian` to see whether we have to do something here.

```
37.3 \ifx\l@croatian\@undefined
37.4     \nopatterns{Croatian}
37.5     \adddialect\l@croatian0\fi
```

The next step consists of defining commands to switch to (and from) the Croatian language.

`\captionscroatian` The macro `\captionscroatian` defines all strings used in the four standard documentclasses provided with L^AT_EX.

```
37.6 \addto\captionscroatian{%
37.7     \def\prefacename{Predgovor}%
37.8     \def\refname{Literatura}%
37.9     \def\abstractname{Sa\v{z}etak}%
37.10    \def\bibname{Bibliografija}%
37.11    \def\chaptername{Glava}%
37.12    \def\appendixname{Dodatak}%
37.13    \def\contentsname{Sadr\v{z}aj}%
37.14    \def\listfigurename{Slike}%
37.15    \def\listtablename{Tablice}%
37.16    \def\indexname{Indeks}%
37.17    \def\figurename{Slika}%
37.18    \def\tablename{Tablica}%
37.19    \def\partname{Dio}%
37.20    \def\enclname{Prilozi}%
37.21    \def\ccname{Kopije}%
37.22    \def\headtoname{Prima}%
37.23    \def\pagename{Strana}%
37.24    \def\seenname{Vidi}%
37.25    \def\alsoname{Vidi tako\dj er}%
37.26    \def\proofname{Dokaz}%
37.27 }%
```

`\datecroatian` The macro `\datecroatian` redefines the command `\today` to produce Croatian dates.

```
37.28 \def\datecroatian{%
37.29     \def\today{\number\day .~\ifcase\month\or
37.30         sijev{c}nja\or velja\v{c}e\or o\v{z}ujka\or travnja\or svibnja\or
```

⁵¹The file described in this section has version number v1.3g and was last revised on 1996/12/23. A contribution was made by Alan Pać (paica@cernvm.cern.ch).

```

37.31 lipnja\or srpnja\or kolovoza\or rujna\or listopada\od studenog\or
37.32 prosinca\fi \space \number\year}}

```

`\extrascroatian` The macro `\extrascroatian` will perform all the extra definitions needed for the Croatian language. The macro `\noextrascroatian` is used to cancel the actions of `\extrascroatian`. For the moment these macros are empty but they are defined for compatibility with the other language definition files.

```

37.33 \addto\extrascroatian{}
37.34 \addto\noextrascroatian{}

```

The macro `\ldf@finish` takes care of looking for a configuration file, setting the main language to be switched on at `\begin{document}` and resetting the category code of `@` to its original value.

```

37.35 \ldf@finish{croatian}
37.36 \code

```

38 The Czech language

The file `czech.dtx`⁵² defines all the language definition macros for the Czech language.

For this language `\frenchspacing` is set and two macros `\q` and `\w` for easy access to two accents are defined.

The command `\q` is used with the letters (t, d, l, and L) and adds a ' to them to simulate a 'hook' that should be there. The result looks like `ṭ`. The command `\w` is used to put the ring-accent which appears in ångström over the letters u and U.

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the `@` sign, etc.

```
38.1 (*code)
38.2 \LdfInit{czech}\captionsczech
```

When this file is read as an option, i.e. by the `\usepackage` command, `czech` will be an 'unknown' language in which case we have to make it known. So we check for the existence of `\l@czec` to see whether we have to do something here.

```
38.3 \ifx\l@czec\@undefined
38.4     \@nopatterns{Czech}
38.5     \adddialect\l@czec0\fi
```

The next step consists of defining commands to switch to (and from) the Czech language.

`\captionsczech` The macro `\captionsczech` defines all strings used in the four standard documentclasses provided with L^AT_EX.

```
38.6 \addto\captionsczech{%
38.7   \def\prefacename{P\v redmluva}%
38.8   \def\refname{Reference}%
38.9   \def\abstractname{Abstrakt}%
38.10  \def\bibname{Literatura}%
38.11  \def\chaptername{Kapitola}%
38.12  \def\appendixname{Dodatek}%
38.13  \def\contentsname{Obsah}%
38.14  \def\listfigurename{Seznam obr\'azk\r{u}}%
38.15  \def\listtablename{Seznam tabulek}%
38.16  \def\indexname{Index}%
38.17  \def\figurename{Obr\'azek}%
38.18  \def\tablename{Tabulka}%
38.19  \def\partname{\v{C}\'}ast}%
38.20  \def\enclname{P\v{r}\'\{i\}loha}%
38.21  \def\ccname{Na v\v{e}dom\'\{i\}:}%
38.22  \def\headtoname{Komu}%
38.23  \def\pagename{Strana}%
38.24  \def\seename{viz}%
38.25  \def\alsoname{viz tak\'e}%
38.26  \def\proofname{D\r{u}kaz}%
38.27  }%
```

⁵²The file described in this section has version number v1.3h and was last revised on 1996/12/23. Contributions were made by Milos Lokajicek (LOKAJICK@CERNVM).

`\dateczech` The macro `\dateczech` redefines the command `\today` to produce Czech dates.

```
38.28 \def\dateczech{%
38.29 \def\today{\number\day.\~\ifcase\month\or
38.30 ledna\or \'unora\or b\v{r}ezna\or dubna\or kv\v{e}tna\or \v{c}ervna\or
38.31 \v{c}ervence\or srpna\or z\'a\v{r}\'\{i\}\or \v{r}\'\{i\}jna\or
38.32 listopadu\or prosince\fi
38.33 \space \number\year}}
```

`\extrasczech` The macro `\extrasczech` will perform all the extra definitions needed for the Czech language. The macro `\noextrasczech` is used to cancel the actions of `\extrasczech`. This means saving the meaning of two one-letter control sequences before defining them.

```
38.34 \addto\extrasczech{\babel@save\q\let\q\v}
38.35 \addto\extrasczech{\babel@save\w\let\w\r}
```

For Czech texts `\frenchspacing` should be in effect. We make sure this is the case and reset it if necessary.

```
38.36 \addto\extrasczech{\bbl@frenchspacing}
38.37 \addto\noextrasczech{\bbl@nonfrenchspacing}
```

`\v` L^AT_EX's normal `\v` accent places a caron over the letter that follows it (ö). This is not what we want for the letters d, t, l and L; for those the accent should change shape. This is achieved by the following.

```
38.38 \AtBeginDocument{%
38.39 \DeclareTextCompositeCommand{\v}{OT1}{t}{t}{%
38.40 t\kern-.23em\raise.24ex\hbox{'}}
38.41 \DeclareTextCompositeCommand{\v}{OT1}{d}{d}{%
38.42 d\kern-.13em\raise.24ex\hbox{'}}
38.43 \DeclareTextCompositeCommand{\v}{OT1}{l}{l}{\lcaron{}}
38.44 \DeclareTextCompositeCommand{\v}{OT1}{L}{L}{\Lcaron{}}}
```

`\lcaron` For the letters l and L we want to distinguish between normal fonts and monospaced fonts.

```
38.45 \def\lcaron{%
38.46 \setbox0\hbox{M}\setbox\tw@\hbox{i}%
38.47 \ifdim\wd0>\wd\tw@\relax
38.48 l\kern-.13em\raise.24ex\hbox{'}\kern-.11em%
38.49 \else
38.50 l\raise.45ex\hbox to\z@{\kern-.35em '\hss}%
38.51 \fi}
38.52 \def\Lcaron{%
38.53 \setbox0\hbox{M}\setbox\tw@\hbox{i}%
38.54 \ifdim\wd0>\wd\tw@\relax
38.55 L\raise.24ex\hbox to\z@{\kern-.28em '\hss}%
38.56 \else
38.57 L\raise.45ex\hbox to\z@{\kern-.40em '\hss}%
38.58 \fi}
```

The macro `\ldf@finish` takes care of looking for a configuration file, setting the main language to be switched on at `\begin{document}` and resetting the category code of `@` to its original value.

```
38.59 \ldf@finish{czech}
38.60 \code
```

39 The Polish language

The file `polish.dtx`⁵³ defines all the language-specific macros for the Polish language.

For this language the character " is made active. In table 15 an overview is given of its purpose.

"a	or \aob, for tailed-a (like ą)
"A	or \Aob, for tailed-A (like Ą)
"e	or \eob, for tailed-e (like ę)
"E	or \Eob, for tailed-E (like Ę)
"c	or \'c, for accented c (like ć), same with uppercase letters and n,o,s
"l	or \lpb{ }, for l with stroke (like ł)
"L	or \Lpb{ }, for L with stroke (like Ł)
"r	or \zkb{ }, for pointed z (like ż), cf. pronunciation
"R	or \Zkb{ }, for pointed Z (like Ż)
"z	or \'z, for accented z
"Z	or \'Z, for accented Z
"	disable ligature at this position.
"-	an explicit hyphen sign, allowing hyphenation in the rest of the word.
" "	like "-", but producing no hyphen sign (for compound words with hyphen, e.g. x-"y).
"‘	for German left double quotes (looks like „).
"’	for German right double quotes.
"<	for French left double quotes (similar to <<).
">	for French right double quotes (similar to >>).

Table 15: The extra definitions made by `polish.sty`

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the @ sign, etc.

39.1 `(*code)`

39.2 `\LdfInit{polish}\captionspolish`

When this file is read as an option, i.e. by the `\usepackage` command, `polish` could be an ‘unknown’ language in which case we have to make it known. So we check for the existence of `\l@polish` to see whether we have to do something here.

39.3 `\ifx\l@polish\undefined`

39.4 `\@nopatterns{Polish}`

39.5 `\adddialect\l@polish0\fi`

The next step consists of defining commands to switch to (and from) the Polish language.

`\captionspolish` The macro `\captionspolish` defines all strings used in the four standard documentclasses provided with L^AT_EX.

⁵³The file described in this section has version number v1.2d and was last revised on 1996/12/23.

```

39.6 \addto\captionspolish{%
39.7   \def\prefacename{Przedmowa}%
39.8   \def\refname{Bibliografia}%
39.9   \def\abstractname{Streszczenie}%
39.10  \def\bibname{Literatura}%
39.11  \def\chaptername{Rozdzia\l}%
39.12  \def\appendixname{Dodatek}%
39.13  \def\contentsname{Spis rzeczy}%
39.14  \def\listfigurename{Spis rysunk\ow}%
39.15  \def\listtablename{Spis tablic}%
39.16  \def\indexname{Indeks}%
39.17  \def\figurename{Rysunek}%
39.18  \def\tablename{Tablica}%
39.19  \def\partname{Cz\eob\}\s\c}%
39.20  \def\enclname{Za\l\ aob\}cznik}%
39.21  \def\ccname{Kopie:}%
39.22  \def\headtoname{Do}%
39.23  \def\pagename{Strona}%
39.24  \def\seename{Por\ownaj}%
39.25  \def\alsoname{Por\ownaj tak\ze}%
39.26  \def\proofname{Proof}%    <-- needs translation
39.27 }

```

\datepolish The macro `\datepolish` redefines the command `\today` to produce Polish dates.

```

39.28 \def\datepolish{%
39.29   \def\today{\number\day~\ifcase\month\or
39.30     stycznia\or lutego\or marca\or kwietnia\or maja\or czerwca\or lipca\or
39.31     sierpnia\or wrze\'snia\or pa\'zdziernika\or listopada\or grudnia\fi
39.32     \space\number\year}
39.33 }

```

\extrapolish The macro `\extrapolish` will perform all the extra definitions needed for the Polish language. The macro `\noextrapolish` is used to cancel the actions of `\extrapolish`.

For Polish the " character is made active. This is done once, later on its definition may vary. Other languages in the same document may also use the " character for shorthands; we specify that the polish group of shorthands should be used.

```

39.34 \initiate@active@char{"}
39.35 \addto\extrapolish{\languageshorthands{polish}}
39.36 \addto\extrapolish{\bbl@activate{}}
39.37 %\addto\noextrapolish{\bbl@deactivate{}}

```

The code above is necessary because we need an extra active character. This character is then used as indicated in table 15.

If you have problems at the end of a word with a linebreak, use the other version without hyphenation tricks. Some TeX wizard may produce a better solution with forecasting another token to decide whether the character after the double quote is the last in a word. Do it and let us know.

In Polish texts some letters get special diacritical marks. Leszek Holenderski designed the following code to position the diacritics correctly for every font in every size. These macros need a few extra dimension variables.

```

39.38 \newdimen\pl@left
39.39 \newdimen\pl@down
39.40 \newdimen\pl@right
39.41 \newdimen\pl@temp

```

\sob The macro **\sob** is used to put the ‘ogonek’ in the right place.

```

39.42 \def\sob#1#2#3#4#5{%parameters: letter and fractions hl,ho,vl,vo
39.43 \setbox0\hbox{#1}\setbox1\hbox{$\_mathchar'454$}\setbox2\hbox{p}%
39.44 \pl@right=#2\wd0 \advance\pl@right by-#3\wd1
39.45 \pl@down=#5\ht1 \advance\pl@down by-#4\ht0
39.46 \pl@left=\pl@right \advance\pl@left by\wd1
39.47 \pl@temp=-\pl@down \advance\pl@temp by\dp2 \dp1=\pl@temp
39.48 \leavevmode
39.49 \kern\pl@right\lower\pl@down\box1\kern-\pl@left #1}

```

\aob The ogonek is placed with the letters ‘a’, ‘A’, ‘e’, and ‘E’.

```

\Aob39.50 \DeclareTextCommand{\aob}{OT1}{\sob a{.66}{.20}{0}{.90}}
\EOB39.51 \DeclareTextCommand{\Aob}{OT1}{\sob A{.80}{.50}{0}{.90}}
\EOB39.52 \DeclareTextCommand{\eob}{OT1}{\sob e{.50}{.35}{0}{.93}}
39.53 \DeclareTextCommand{\Eob}{OT1}{\sob E{.60}{.35}{0}{.90}}

```

For the ‘new’ T1 encoding we can provide simpler definitions.

```

39.54 \DeclareTextCommand{\aob}{T1}{\k a}
39.55 \DeclareTextCommand{\Aob}{T1}{\k A}
39.56 \DeclareTextCommand{\eob}{T1}{\k e}
39.57 \DeclareTextCommand{\Eob}{T1}{\k E}

```

Construct the characters by default from the OT1 encoding.

```

39.58 \ProvideTextCommandDefault{\aob}{\UseTextSymbol{OT1}{\aob}}
39.59 \ProvideTextCommandDefault{\Aob}{\UseTextSymbol{OT1}{\Aob}}
39.60 \ProvideTextCommandDefault{\eob}{\UseTextSymbol{OT1}{\eob}}
39.61 \ProvideTextCommandDefault{\Eob}{\UseTextSymbol{OT1}{\Eob}}

```

\spb The macro **\spb** is used to put the ‘poprzeczka’ in the right place.

```

39.62 \def\spb#1#2#3#4#5{%
39.63 \setbox0\hbox{#1}\setbox1\hbox{\char'023}%
39.64 \pl@right=#2\wd0 \advance\pl@right by-#3\wd1
39.65 \pl@down=#5\ht1 \advance\pl@down by-#4\ht0
39.66 \pl@left=\pl@right \advance\pl@left by\wd1
39.67 \ht1=\pl@down \dp1=-\pl@down
39.68 \leavevmode
39.69 \kern\pl@right\lower\pl@down\box1\kern-\pl@left #1}

```

\skb The macro **\skb** is used to put the ‘kropka’ in the right place.

```

39.70 \def\skb#1#2#3#4#5{%
39.71 \setbox0\hbox{#1}\setbox1\hbox{\char'056}%
39.72 \pl@right=#2\wd0 \advance\pl@right by-#3\wd1
39.73 \pl@down=#5\ht1 \advance\pl@down by-#4\ht0
39.74 \pl@left=\pl@right \advance\pl@left by\wd1
39.75 \leavevmode
39.76 \kern\pl@right\lower\pl@down\box1\kern-\pl@left #1}

```


`\textpl` For the ‘poprzeczka’ and the ‘kropka’ in text fonts we don’t need any special coding, but we can (almost) use what is already available.

```
39.77 \def\textpl{%
39.78   \def\lpb{\p111}%
39.79   \def\Lpb{\pLLL}%
39.80   \def\zkb{\.z}%
39.81   \def\Zkb{\.Z}}
```

Initially we assume that typesetting is done with text fonts.

```
39.82 \textpl
39.83 \let\111=\1 \let\LLL=\L
39.84 \def\p111{\111}
39.85 \def\pLLL{\LLL}
```

`\telepl` But for the ‘teletype’ font in ‘OT1’ encoding we have to take some special actions, involving the macros defined above.

```
39.86 \def\telepl{%
39.87   \def\lpb{\spb 1{.45}{.5}{.4}{.8}}%
39.88   \def\Lpb{\spb L{.23}{.5}{.4}{.8}}%
39.89   \def\zkb{\skb z{.5}{.5}{1.2}{0}}%
39.90   \def\Zkb{\skb Z{.5}{.5}{1.1}{0}}}
```

To activate these codes the font changing commands as they are defined in \LaTeX are modified. The same is done for plain \TeX ’s font changing commands.

When `\selectfont` is undefined the current format is supposed to be either plain (based) or \LaTeX 2.09.

```
39.91 \ifx\selectfont@undefined
39.92   \ifx\prm@undefined \addto\rm{\textpl}\else \addto\prm{\textpl}\fi
39.93   \ifx\pit@undefined \addto\it{\textpl}\else \addto\pit{\textpl}\fi
39.94   \ifx\pbf@undefined \addto\bf{\textpl}\else \addto\pbf{\textpl}\fi
39.95   \ifx\psl@undefined \addto\sl{\textpl}\else \addto\psl{\textpl}\fi
39.96   \ifx\psf@undefined \else \addto\psf{\textpl}\fi
39.97   \ifx\psc@undefined \else \addto\psc{\textpl}\fi
39.98   \ifx\ptt@undefined \addto\tt{\telepl}\else \addto\ptt{\telepl}\fi
39.99 \else
```

When `\selectfont` exists we assume \LaTeX 2 ϵ .

```
39.100 \expandafter\addto\csname selectfont \endcsname{%
39.101   \csname f@encoding @pl\endcsname}
39.102 \fi
```

Currently we support the OT1 and T1 encodings. For T1 we don’t have to make a difference between typewriter fonts and other fonts, they all have the same glyphs.

```
39.103 \expandafter\let\csname T1@pl\endcsname\textpl
```

For OT1 we need to check the current font family, stored in `\f@family`. Unfortunately we need a hack as `\ttdefault` is defined as a `\long` macro, while `\f@family` is not.

```
39.104 \expandafter\def\csname OT1@pl\endcsname{%
39.105   \long\edef\curr@family{\f@family}%
39.106   \ifx\curr@family\ttdefault
39.107     \telepl
39.108   \else
39.109     \textpl
39.110   \fi}
```

`\dq` We save the original double quote character in `\dq` to keep it available, the math accent `\"` can now be typed as `"`.

```
39.111 \begingroup \catcode'\''12
39.112 \def\x{\endgroup
39.113   \def\dq{"}}
39.114 \x
```

Now we can define the doublequote macros for diacritics,

```
39.115 \declare@shorthand{polish}{a}{\textormath{\aob}{\ddot a}}
39.116 \declare@shorthand{polish}{A}{\textormath{\Aob}{\ddot A}}
39.117 \declare@shorthand{polish}{c}{\textormath{\'c}{\acute c}}
39.118 \declare@shorthand{polish}{C}{\textormath{\'C}{\acute C}}
39.119 \declare@shorthand{polish}{e}{\textormath{\eob}{\ddot e}}
39.120 \declare@shorthand{polish}{E}{\textormath{\Eob}{\ddot E}}
39.121 \declare@shorthand{polish}{l}{\textormath{\lpb}{\ddot l}}
39.122 \declare@shorthand{polish}{L}{\textormath{\Lpb}{\ddot L}}
39.123 \declare@shorthand{polish}{n}{\textormath{\'n}{\acute n}}
39.124 \declare@shorthand{polish}{N}{\textormath{\'N}{\acute N}}
39.125 \declare@shorthand{polish}{o}{\textormath{\'o}{\acute o}}
39.126 \declare@shorthand{polish}{O}{\textormath{\'O}{\acute O}}
39.127 \declare@shorthand{polish}{r}{\textormath{\zkb}{\ddot r}}
39.128 \declare@shorthand{polish}{R}{\textormath{\Zkb}{\ddot R}}
39.129 \declare@shorthand{polish}{s}{\textormath{\'s}{\acute s}}
39.130 \declare@shorthand{polish}{S}{\textormath{\'S}{\acute S}}
39.131 \declare@shorthand{polish}{z}{\textormath{\'z}{\acute z}}
39.132 \declare@shorthand{polish}{Z}{\textormath{\'Z}{\acute Z}}
```

Then we define access to two forms of quotation marks, similar to the german and french quotation marks.

```
39.133 \declare@shorthand{polish}{"'}{
39.134   \textormath{\quotedblbase{}}{\mbox{\quotedblbase}}}
39.135 \declare@shorthand{polish}{"'}{
39.136   \textormath{\textquotedblleft{}}{\mbox{\textquotedblleft}}}
39.137 \declare@shorthand{polish}{"<}{
39.138   \textormath{\guillemotleft{}}{\mbox{\guillemotleft}}}
39.139 \declare@shorthand{polish}{">}{
39.140   \textormath{\guillemotright{}}{\mbox{\guillemotright}}}
```

then we define two shorthands to be able to specify hyphenation breakpoints that behave a little different from `\-`.

```
39.141 \declare@shorthand{polish}{"-}{\allowhyphens-\allowhyphens}
39.142 \declare@shorthand{polish}{""}{\hskip\z@skip}
```

And we want to have a shorthand for disabling a ligature.

```
39.143 \declare@shorthand{polish}{"|}{
39.144   \textormath{\discretionary{-}{-}{\kern.03em}}{}}
```

`\mdqon` All that's left to do now is to define a couple of commands for reasons of compatibility with `polish.tex`.

```
39.145 \def\mdqon{\bbl@activate{}}
39.146 \def\mdqoff{\bbl@deactivate{}}
```

The macro `\ldf@finish` takes care of looking for a configuration file, setting the main language to be switched on at `\begin{document}` and resetting the category code of `@` to its original value.

39.147 \ldf@finish{polish}
39.148 \code

40 The Slovak language

The file `slovak.dtx`⁵⁴ defines all the language-specific macros for the Slovak language.

For this language the macro `\q` is defined. It is used with the letters (t, d, l, and L) and adds a ' to them to simulate a 'hook' that should be there. The result looks like t'.

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the `@` sign, etc.

```
40.1 (*code)
40.2 \LdfInit{slovak}\captionsslovak
```

When this file is read as an option, i.e. by the `\usepackage` command, `slovak` will be an 'unknown' language in which case we have to make it known. So we check for the existence of `\l@slovak` to see whether we have to do something here.

```
40.3 \ifx\l@slovak\@undefined
40.4     \nopatterns{Slovak}
40.5     \adddialect\l@slovak0\fi
```

The next step consists of defining commands to switch to (and from) the Slovak language.

`\captionsslovak` The macro `\captionsslovak` defines all strings used in the four standard documentclasses provided with L^AT_EX.

```
40.6 \addto\captionsslovak{%
40.7   \def\prefacename{\'Uvod}%
40.8   \def\refname{Referencia}%
40.9   \def\abstractname{Abstrakt}%
40.10  \def\bibname{Literat\'ura}%
40.11  \def\chaptername{Kapitola}%
40.12  \def\appendixname{Dodatok}%
40.13  \def\contentsname{Obsah}%
40.14  \def\listfigurename{Zoznam obr\'azkov}%
40.15  \def\listtablename{Zoznam tabuliek}%
40.16  \def\indexname{Index}%
40.17  \def\figurename{Obr\'azok}%
40.18  \def\tablename{Tabuľka}%%% special letter l with hook
40.19  \def\partname{\v{C}as\q t}%%% special letter t with hook
40.20  \def\enclname{Pr\'iloha}%
40.21  \def\ccname{CC}%
40.22  \def\headtoname{Komu}%
40.23  \def\pagename{Strana}%
40.24  \def\seenname{\vi\q d}%%% Special letter d with hook
40.25  \def\alsoname{\vi\q d tie\v z}%%% Special letter d with hook
40.26  \def\proofname{Proof}% <-- needs translation
40.27 }
```

`\dateslovak` The macro `\dateslovak` redefines the command `\today` to produce Slovak dates.

```
40.28 \def\dateslovak{%
40.29 \def\today{\number\day.\~\ifcase\month\or
40.30 janu\'ara\or febru\'ara\or marca\or apr\'{i}la\or m\'aja\or j\'una\or
```

⁵⁴The file described in this section has version number v1.2i and was last revised on 1996/12/23. It was written by Jana Chlebková (chlebk@euromath.dk).

```

40.31 j\'ula\or augusat\or septembra\or okt\'obra\or
40.32 novembra\or decembra\fi
40.33 \space \number\year}}

```

\extrasslovak The macro **\extrasslovak** will perform all the extra definitions needed for the Slovak language. The macro **\noextrasslovak** is used to cancel the actions of **\extrasslovak**. This currently means saving the meaning of one one-letter control sequence before defining it.

```

40.34 \addto\extrasslovak{\babel@save\q\let\q\v}

```

The slovak hyphenation patterns should be used with **\lefthyphenmin** set to 2 and **\righthyphenmin** set to 2.

```

40.35 \def\slovakhyphenmins{\tw@\tw@}

```

The macro **\ldf@finish** takes care of looking for a configuration file, setting the main language to be switched on at **\begin{document}** and resetting the category code of **@** to its original value.

```

40.36 \ldf@finish{slovak}
40.37 \code

```

41 The Slovenian language

The file `slovene.dtx`⁵⁵ defines all the language-specific macros for the Slovenian language.

For this language the character " is made active. In table 16 an overview is given of its purpose. One of the reasons for this is that in the Slovene language some special characters are used.

"c	\ "c, also implemented for the lowercase and uppercase s and z.
"-	an explicit hyphen sign, allowing hyphenation in the rest of the word.
" "	like "-", but producing no hyphen sign (for compound words with hyphen, e.g. x-"y).
"‘	for Slovene left double quotes (looks like „).
"’	for Slovene right double quotes.
"<	for French left double quotes (similar to <<).
">	for French right double quotes (similar to >>).

Table 16: The extra definitions made by `slovene.ldf`

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the @ sign, etc.

41.1 (*code)

41.2 `\LdfInit{slovene}\captionsslovene`

When this file is read as an option, i.e. by the `\usepackage` command, `slovene` will be an ‘unknown’ language in which case we have to make it known. So we check for the existence of `\l@slovene` to see whether we have to do something here.

41.3 `\ifx\l@slovene\@undefined`

41.4 `\@nopatterns{Slovene}`

41.5 `\adddialect\l@slovene0\fi`

The next step consists of defining commands to switch to the Slovenian language. The reason for this is that a user might want to switch back and forth between languages.

`\captionsslovene` The macro `\captionsslovene` defines all strings used in the four standard documentclasses provided with L^AT_EX.

41.6 `\addto\captionsslovene{%`

41.7 `\def\prefacename{Predgovor}%`

41.8 `\def\refname{Literatura}%`

41.9 `\def\abstractname{Povzetek}%`

41.10 `\def\bibname{Literatura}%`

41.11 `\def\chaptername{Poglavje}%`

41.12 `\def\appendixname{Dodatek}%`

41.13 `\def\contentsname{Kazalo}%`

⁵⁵The file described in this section has version number v1.2i and was last revised on 1996/12/23. Contributions were made by Danilo Zavrtanik, University of Ljubljana (YU) and Leon Žlajpah (leon.zlajpah@ijs.si).

```

41.14 \def\listfigurename{Slike}%
41.15 \def\listtablename{Tabele}%
41.16 \def\indexname{Stvarno kazalo}% used to be Indeks
41.17 \def\figurename{Slika}%
41.18 \def\tablename{Tabela}%
41.19 \def\partname{Del}%
41.20 \def\enclname{Priloge}%
41.21 \def\ccname{Kopije}%
41.22 \def\headtoname{Prejme}%
41.23 \def\pagename{Stran}%
41.24 \def\seename{glej}%
41.25 \def\alsoname{glej tudi}%
41.26 \def\proofname{Dokaz}%
41.27 }%

```

`\dateslovene` The macro `\dateslovene` redefines the command `\today` to produce Slovenian dates.

```

41.28 \def\dateslovene{%
41.29 \def\today{\number\day.\~\ifcase\month\or
41.30 januar\or februar\or marec\or april\or maj\or junij\or
41.31 julij\or avgust\or september\or oktober\or november\or december\fi
41.32 \space \number\year}}

```

`\extrasslovene` The macro `\extrasslovene` performs all the extra definitions needed for the Slovenian language. The macro `\noextrasslovene` is used to cancel the actions of `\extrasslovene`.

For Slovene the " character is made active. This is done once, later on its definition may vary. Other languages in the same document may also use the " character for shorthands; we specify that the slovenian group of shorthands should be used.

```

41.33 \initiate@active@char{"}
41.34 \addto\extrasslovene{\languageshorthands{slovene}}
41.35 \addto\extrasslovene{\bbl@activate{}}
41.36 %\addto\noextrasslovene{\bbl@deactivate{}}

```

First we define shorthands to facilitate the occurrence of letters such as č.

```

41.37 \declare@shorthand{slovene}{"c"}{\textormath{\v c}{\check c}}
41.38 \declare@shorthand{slovene}{"s"}{\textormath{\v s}{\check s}}
41.39 \declare@shorthand{slovene}{"z"}{\textormath{\v z}{\check z}}
41.40 \declare@shorthand{slovene}{"C"}{\textormath{\v C}{\check C}}
41.41 \declare@shorthand{slovene}{"S"}{\textormath{\v S}{\check S}}
41.42 \declare@shorthand{slovene}{"Z"}{\textormath{\v Z}{\check Z}}

```

Then we define access to two forms of quotation marks, similar to the german and french quotation marks.

```

41.43 \declare@shorthand{slovene}{"'"}{%
41.44 \textormath{\quotedblbase{}}{\mbox{\quotedblbase}}}
41.45 \declare@shorthand{slovene}{"'"}{%
41.46 \textormath{\textquotedblleft{}}{\mbox{\textquotedblleft}}}
41.47 \declare@shorthand{slovene}{"<"}{%
41.48 \textormath{\guillemotleft{}}{\mbox{\guillemotleft}}}
41.49 \declare@shorthand{slovene}{">"}{%
41.50 \textormath{\guillemotright{}}{\mbox{\guillemotright}}}

```

then we define two shorthands to be able to specify hyphenation breakpoints that behave a little different from `\-`.

```
41.51 \declare@shorthand{slovene}{"-}{\allowhyphens-\allowhyphens}
```

```
41.52 \declare@shorthand{slovene}{""}{\hskip\z@skip}
```

And we want to have a shorthand for disabling a ligature.

```
41.53 \declare@shorthand{slovene}{"}{%
```

```
41.54 \textormath{\discretionary{-}{-}{\kern.03em}}{}}
```

The macro `\ldf@finish` takes care of looking for a configuration file, setting the main language to be switched on at `\begin{document}` and resetting the category code of `@` to its original value.

```
41.55 \ldf@finish{slovene}
```

```
41.56 \endcode
```


42 The Russian language

The file `russianb.dtx`⁵⁶ defines all the language-specific macros for the Russian language. It needs the file `cyrnod` for success documentation with Russian encodings (see below).

For this language the character " is made active. In table 17 an overview is given of its purpose.

"	disable ligature at this position.
"-	an explicit hyphen sign, allowing hyphenation in the rest of the word.
""	like "-", but producing no hyphen sign (for compound words with hyphen, e.g. x-"y or some other signs as "disable/enable").
"~	for a compound word mark without a breakpoint.
"=	for a compound word mark with a breakpoint, allowing hyphenation in the composing words.
"‘	for German left double quotes (looks like „).
"’	for German right double quotes (looks like “).
" "<	for French left double quotes (looks like <<).
" ">	for French right double quotes (looks like >>).

Table 17: The extra definitions made by `russianb`

The quotes in table 17 can also be typeset by using the commands in table 18.

<code>\glqq</code>	for German left double quotes (looks like „).
<code>\grqq</code>	for German right double quotes (looks like “).
<code>" \flqq</code>	for French left double quotes (looks like <<).
<code>\frqq</code>	for French right double quotes (looks like >>).
<code>\dq</code>	the original quotes character (").

Table 18: More commands which produce quotes, defined by `babel`

The quotation marks traditionally used in Russian language were borrowed from other languages (e.g. French and German) so they keep their original names.

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the `@` sign, etc.

42.1 `{*code}`

42.2 `\LdfInit{russian}{captionsrussian}`

When this file is read as an option, i.e., by the `\usepackage` command, `russianb` will be an ‘unknown’ language, in which case we have to make it known. So we check for the existence of `\l@russian` to see whether we have to do something here.

⁵⁶The file described in this section has version number ? and was last revised on ?. This file was initially derived from the original version of `german.sty`, which has some definitions for Russian. Later the definitions from `russian.sty` version 1.0b (for L^AT_EX 2.09), `russian.sty` version v2.5c (for L^AT_EX 2e) and `francais.sty` version 4.5c and `germanb.sty` version 2.5c were added.

```

42.3 \ifx\l@russian\@undefined
42.4 \nopatterns{Russian}%
42.5 \addialect\l@russian0
42.6 \fi

```

Tysetting Russian texts implies that a special set of fonts needs to be used. The current support for Russian uses a set of fonts that more or less ‘match’ with the computer modern fonts. Therefore we define the Local Washington encoding (LWN, see the file `cyrillic.fdd`). We make sure that this encoding is known to L^AT_EX.

```

42.7 \input{LWNenc.def}

```

`\latinencoding` We need to know the encoding for text that is supposed to be typeset in latin text. We assume that it will be the encoding which is active at the end of the `babel` package. If the `fontenc` package is loaded later, then... too bad!

```

42.8 \AtEndOfPackage{\edef\latinencoding{cf@encoding}}

```

Now we define two commands that offer the possibility to switch between cyrillic and roman encodings.

`\cyrillictext` The command `\cyrillictext` will switch from latin font encoding to the cyrillic font encoding, the command `\latintext` switches back. This assumes that the ‘normal’ font encoding is a latin one. These commands are *declarations*, for shorter peaces of text the commands `\textlatin` and `\textcyrillic` can be used.

```

42.9 \DeclareRobustCommand{\cyrillictext}{%
42.10 \fontencoding{LWN}\selectfont
42.11 \def\encodingdefault{LWN}}
42.12 \DeclareRobustCommand{\latintext}{%
42.13 \fontencoding{\latinencoding}\selectfont
42.14 \def\encodingdefault{\latinencoding}}
42.15 \let\lat\latintext
42.16 \let\let\cyr\cyrillictext

```

`\textcyrillic` These commands take an argument which is then typeset using the requested font encoding.
`\textlatin`

The next step consists of defining commands to switch to (and from) the Russian language.

`\captionsrussian` The macro `\captionsrussian` defines all strings used in the four standard document classes provided with L^AT_EX. There are the two commands: `\cyr` and `\lat` which switch on the right (Cyrillic or Latin) encoding.

```

42.17 \addto\captionsrussian{%
42.18 \def\prefacename{%
42.19 {\cyr \CYRP\CYRr\CYRe\CYRd\CYRi\CYRs\CYRl\CYRo\CYRv\CYRi\CYRe}}%
42.20 %{\cyr \CYRV\CYRv\CYRe\CYRd\CYRe\CYRn\CYRi\CYRe}}%
42.21 \def\refname{%
42.22 {\cyr \CYRS\CYRp\CYRi\CYRs\CYRo\CYRk\space
42.23 \CYRl\CYRi\CYRt\CYRe\CYRr\CYRa\CYRt\CYRu\CYRr\CYRy}}%
42.24 \def\abstractname{%
42.25 {\cyr \CYRA\CYRn\CYRn\CYRo\CYRt\CYRa\CYRc\CYRi\CYRya}}%
42.26 \def\bibname{%
42.27 {\cyr\CYRB\CYRi\CYRb\CYRl\CYRi\CYRo\CYRg\CYRr\CYRa\CYRf\CYRi\CYRya}}%

```

```

42.28 \def\chaptername{%
42.29     {\cyr \CYRG\CYRl\CYRa\CYRv\CYRa}}%
42.30 \def\appendixname{%
42.31     {\cyr \CYRP\CYRr\CYRi\CYRl\CYRo\CYRzh\CYRe\CYRn\CYRi\CYRe}}%

    There are two names for the Table of Contents that are in use in Russian publi-
    cations.

42.32 \def\contentsname{%
    For books this one is appropriate:
42.33     {\cyr \CYRO\CYRg\CYRl\CYRa\CYRv\CYRl\CYRe\CYRn\CYRi\CYRe}}%
    but for proceedings the following is preferred:
42.34     {%\cyr \CYRS\CYRo\CYRd\CYRe\CYRr\CYRzh\CYRa\CYRn\CYRi\CYRe}}%
42.35 \def\listfigurename{%
42.36     {\cyr \CYRS\CYRp\CYRi\CYRs\CYRo\CYRk\space
42.37         \CYRi\CYRl\CYRl\CYRyu\CYRs\CYRt\CYRr\CYRa\CYRc\CYRi\CYRishrt}}%

    The List of Tables is not used so we provide an empty definition by default.
42.38 \def\listtablename{%
42.39     %\CYRS\CYRp\CYRi\CYRs\CYRo\CYRk\space
42.40     %\CYRt\CYRa\CYRb\CYRl\CYRi\CYRc}%
42.41 }
42.42 \def\indexname{%
42.43     {\cyr \CYRP\CYRr\CYRe\CYRd\CYRm\CYRe\CYRt\CYRn\CYRy\CYRishrt\space
42.44         \CYRu\CYRk\CYRa\CYRz\CYRa\CYRt\CYRe\CYRl\CYRssgn}}%
42.45 \def\authorname{%
42.46     {\cyr \CYRI\CYRm\CYRe\CYRn\CYRn\CYRo\CYRishrt\space
42.47         \CYRu\CYRk\CYRa\CYRz\CYRa\CYRt\CYRe\CYRl\CYRssgn}}%
42.48 \def\figurename{{\cyr \CYRR\CYRi\CYRs.}}%
42.49 \def\tablename{{\cyr \CYRT\CYRa\CYRb\CYRl\CYRi\CYRc\CYRa}}%
42.50 \def\partname{{\cyr \CYRCH\CYRa\CYRs\CYRt\CYRssgn}}%
42.51 \def\enclname{{\cyr \CYRv\CYRk\CYRl.}}%
42.52 \def\ccname{{\cyr \CYRi\CYRs\CYRh.}}%
42.53 \def\headtoname{{\cyr \CYRv\CYRh.}}%
42.54 \def\pagename{{\cyr \CYRs.}}%
42.55 \def\seename{{\cyr \CYRs\CYRm.}}%
42.56 \def\alsoname{{\cyr \CYRs\CYRm.\ \CYRt\CYRa\CYRk\CYRzh\CYRe}}

```

`\daterussian` The macro `\daterussian` redefines the command `\today` to produce Russian dates.

```

42.57 \def\month@russian{\ifcase\month\or
42.58     \CYRya\CYRn\CYRv\CYRa\CYRr\CYRya\or
42.59     \CYRf\CYRe\CYRv\CYRr\CYRa\CYRl\CYRya\or
42.60     \CYRm\CYRa\CYRr\CYRt\CYRa\or
42.61     \CYRa\CYRp\CYRr\CYRe\CYRl\CYRya\or
42.62     \CYRm\CYRa\CYRya\or
42.63     \CYRi\CYRyu\CYRn\CYRya\or
42.64     \CYRi\CYRyu\CYRl\CYRya\or
42.65     \CYRa\CYRv\CYRg\CYRu\CYRs\CYRt\CYRa\or
42.66     \CYRs\CYRe\CYRn\CYRt\CYRya\CYRb\CYRr\CYRya\or
42.67     \CYRo\CYRk\CYRt\CYRya\CYRb\CYRr\CYRya\or
42.68     \CYRn\CYRo\CYRya\CYRb\CYRr\CYRya\or
42.69     \CYRd\CYRe\CYRk\CYRa\CYRb\CYRr\CYRya\fi}
42.70 \def\daterussian{%
42.71     \def\today{\number\day~\month@russian\space\number\year~\CYRg.}}

```

`\extrasrussian` The macro `\extrasrussian` will perform all the extra definitions needed for the Russian language. The macro `\noextrasrussian` is used to cancel the actions of `\extrasrussian`.

The first action we define is to switch to the LWN encoding whenever we enter ‘russian’.

```
42.72 \addto\extrasrussian{\cyrillictext}
```

When the file `LWNenc.def` was processed by LaTeX it stores the current font encoding in `\latinencoding`, assuming that L^AT_EX uses T1 or OT1 as default. Therefore we switch back to `\latinencoding` whenever the russian language is no longer ‘active’.

```
42.73 \addto\noextrasrussian{\latintext}
```

`\verbatim@font` In order to get verbatim text in the latin alphabet we need to change the definition of an internal L^AT_EX command somewhat:

```
42.74 \def\verbatim@font{%
```

```
42.75   \normalfont
```

```
42.76   \fontencoding\latinencoding\ttfamily}
```

In order to be able to use cyrillic letters in mathematics we need to have the package `cyrmath` available.

```
42.77 \AtEndOfPackage{\RequirePackage{cyrmath}}
```

The category code of the characters `:`, `;`, `!` and `?` is made `\active` to insert a little white space.

For Russian (as well as for German) the `"` character also is made active.

```
42.78 \initiate@active@char{:}
```

```
42.79 \initiate@active@char{;}
```

```
42.80 \initiate@active@char{!}
```

```
42.81 \initiate@active@char{?}
```

```
42.82 \initiate@active@char{"}
```

The code above is necessary because we need extra active characters. The character `"` is used as indicated in table 17.

We specify that the russian group of shorthands should be used.

```
42.83 \addto\extrasrussian{\languageshorthands{russian}}
```

These characters are ‘turned on’ once, later their definition may vary.

```
42.84 \addto\extrasrussian{%
```

```
42.85   \bbl@activate{:}\bbl@activate{;}%
```

```
42.86   \bbl@activate{!}\bbl@activate{?}%
```

```
42.87   \bbl@activate{"}}
```

```
42.88 %\addto\noextrasrussian{%
```

```
42.89 %   \bbl@deactivate{:}\bbl@deactivate{;}%
```

```
42.90 %   \bbl@deactivate{!}\bbl@deactivate{?}%
```

```
42.91 %   \bbl@deactivate{"}}
```

`\russian@sh@;` We have to reduce the amount of white space before `;`, `:` and `!`. This should only happen in horizontal mode, hence the test with `\ifhmode`.

```
\russian@sh@!@42.92 \declare@shorthand{russian}{;}{;%
```

```
\russian@sh@?@42.93   \ifhmode
```

In horizontal mode we check for the presence of a ‘space’, ‘unskip’ if it exists and place a 0.1em kerning.

```
42.94 \ifdim\lastskip>\z@
42.95 \unskip\penalty\@M\thinspace
42.96 \else
42.97 \thinspace
42.98 \fi
42.99 \fi
```

Now we can insert a ; character.

```
42.100 \string;}
```

Because these definitions are very similar only one is displayed in a way that the definition can be easily checked.

```
42.101 \declare@shorthand{russian}{:}{\%
42.102 \ifhmode
42.103 \ifdim\lastskip>\z@
42.104 \unskip\penalty\@M\thinspace
42.105 \else
42.106 \thinspace
42.107 \fi
42.108 \fi
42.109 \string:}

42.110 \declare@shorthand{russian}{!}{\%
42.111 \ifhmode
42.112 \ifdim\lastskip>\z@
42.113 \unskip\penalty\@M\thinspace
42.114 \else
42.115 \thinspace
42.116 \fi
42.117 \fi
42.118 \string!}

42.119 \declare@shorthand{russian}{?}{\%
42.120 \ifhmode
42.121 \ifdim\lastskip>\z@
42.122 \unskip\penalty\@M\thinspace
42.123 \else
42.124 \thinspace
42.125 \fi
42.126 \fi
42.127 \string?}
```

\system@sh@: When the active characters appear in an environment where their Russian behaviour is not wanted they should give an ‘expected’ result. Therefore we define \system@sh@? shorthands at system level as well.

```
\system@sh@;
42.128 \declare@shorthand{system}{:}{\string:}
42.129 \declare@shorthand{system}{!}{\string!}
42.130 \declare@shorthand{system}{?}{\string?}
42.131 \declare@shorthand{system}{;}{\string;}
```

To be able to define the function of ", we first define a couple of ‘support’ macros.

`\dq` We save the original double quote character in `\dq` to keep it available, the math accent `\"` can now be typed as `"`.

```
42.132 \begingroup \catcode'\''12
42.133 \def\x{\endgroup
42.134 \def\@SS{\mathchar"7019 }
42.135 \def\dq{"}}
42.136 \x
```

Now we can define the doublequote macros: german and french quotes. The french quotes are made in Russian font so they are described in `lhr cod.sty`

```
42.137 \declare@shorthand{russian}{"'}{%
42.138 \textormath{\quotedblbase}{\mbox{\quotedblbase}}}
42.139 \declare@shorthand{russian}{"'}{%
42.140 \textormath{\kern-.07em\textquotedblleft}{\mbox{\textquotedblleft}}}
42.141 \declare@shorthand{russian}{"<}{%
42.142 \textormath{\flqq}{\mbox{\flqq}}}
42.143 \declare@shorthand{russian}{">}{%
42.144 \textormath{\frqq}{\mbox{\frqq}}}
```

and some additional commands:

```
42.145 \declare@shorthand{russian}{""}{\hskip\z@skip}
42.146 \declare@shorthand{russian}{""}{\textormath{\leavevmode\hbox{-}}{-}}
42.147 \declare@shorthand{russian}{""}{\penalty\@M-\hskip\z@skip}
42.148 \declare@shorthand{russian}{"|}{%
42.149 \textormath{\penalty\@M\discretionary{-}{-}{\kern.03em}%
42.150 \allowhyphens}{}}
```

The next two macros for `"-` and `"--` have some difference. We must check whether the second token is a hyphen character:

```
42.151 \declare@shorthand{russian}{"-}{%
```

If the next token is `-`, we typeset an emdash, else—hyphen sign:

```
42.152 \def\russian@sh@tmp{%
42.153 \if\russian@sh@next-\expandafter\russian@sh@emdash
42.154 \else\expandafter\russian@sh@hyphen\fi
42.155 }%
```

$\mathrm{T}_{\mathrm{E}}\mathrm{X}$ looks for the next sign after first `-`, the meaning of this sign it writes to `\russian@sh@next` and call `\russian@sh@tmp`

```
42.156 \futurelet\russian@sh@next\russian@sh@tmp}
```

There are the definitions of hyphen and emdash: hyphen definition:

```
42.157 \def\russian@sh@hyphen{%
42.158 \penalty\@M-\allowhyphens}
```

emdash definition, there are the two parameters: we must “eat” two last hyphen signs of our emdash:

```
42.159 \def\russian@sh@emdash#1#2{%
42.160 \ifdim\lastskip>\z@
42.161 \unskip
42.162 \fi
42.163 \penalty\@M
42.164 \hskip.2\fontdimen6\font
42.165 \hbox to.8\fontdimen6\font{--\hss--}%
42.166 \hskip.2\fontdimen6\font
42.167 \ignorespaces}
```

The russian hyphenation patterns can be used with `\lefthyphenmin` and `\righthyphenmin` set to 2.

```
42.168 \def\russianhyphenmins{\tw@\tw@}
```

Now the thing `\extrasrussian` needs to do is to make sure that `\frenchspacing` is in effect. If this is not the case the execution of `\noextrasrussian` will switch it off again.

```
42.169 \addto\extrasrussian{\bbl@frenchspacing}
```

```
42.170 \addto\noextrasrussian{\bbl@nonfrenchspacing}
```

Now we add a new enumeration style for Russian manuscripts with Cyrillic letters and later on we define some math operatornames in accordance with Russian typesetting traditions.

`\Asbuk` We begin by defining `\Asbuk` which functions like `\Alpha`, but produces (uppercase) cyrillic letters instead of latin ones.

```
42.171 \def\Asbuk#1{\expandafter\@Asbuk\csname c@#1\endcsname}
```

```
42.172 \def\@Asbuk#1{\ifcase#1\or
```

```
42.173 \CYRA\or \CYRB\or \CYRV\or \CYRG\or \CYRD\or \CYRE\or \CYRZH\or
```

```
42.174 \CYRZ\or \CYRI\or \CYRK\or \CYRL\or \CYRM\or \CYRN\or \CYRO\or
```

```
42.175 \CYRP\or \CYRR\or \CYRS\or \CYRT\or \CYRU\or \CYRF\or \CYRH\or
```

```
42.176 \CYRC\or \CYRCH\or \CYRSH\or \CYRSHCH\or \CYRErev\or \CYRYU\or
```

```
42.177 \CYRYA\else\@ctrerr\fi}%
```

`\asbuk` The macro `\asbuk` is similar to `\alph`, it produces lowercase Russian letters.

```
42.178 \def\asbuk#1{\expandafter\@asbuk\csname c@#1\endcsname}
```

```
42.179 \def\@asbuk#1{\ifcase#1\or
```

```
42.180 \CYRa\or \CYRb\or \CYRv\or \CYRg\or \CYRd\or \CYRe\or \CYRzh\or
```

```
42.181 \CYRz\or \CYRi\or \CYRk\or \CYRl\or \CYRm\or \CYRn\or \CYRo\or
```

```
42.182 \CYRp\or \CYRr\or \CYRs\or \CYRt\or \CYRu\or \CYRf\or \CYRh\or
```

```
42.183 \CYRc\or \CYRch\or \CYRsh\or \CYRshch\or \CYRerev\or \CYRyu\or
```

```
42.184 \CYRYa\else\@ctrerr\fi}
```

`\mathrussian` Some math functions in Russian math books have other names: e.g. `\sinh` in Russian is written as `sh` etc. So we define a number of new math operators.

```
42.185 \def\sh{\mathop{\operator@font sh}\nolimits} % same as \sinh
```

```
42.186 \def\ch{\mathop{\operator@font ch}\nolimits} % same as \cosh
```

```
42.187 \def\tg{\mathop{\operator@font tg}\nolimits} % same as \tan
```

```
42.188 \def\arctg{\mathop{\operator@font arctg}\nolimits} % same as \arctan
```

```
42.189 \def\arcctg{\mathop{\operator@font arcctg}\nolimits} %
```

```
42.190 \def\th{\mathop{\operator@font th}\nolimits} % same as \tanh
```

```
42.191 \def\ctg{\mathop{\operator@font ctg}\nolimits} % same as \cot
```

```
42.192 \def\cth{\mathop{\operator@font cth}\nolimits} % same as \coth
```

```
42.193 \def\cosec{\mathop{\operator@font cosec}\nolimits} % same as \csc
```

```
42.194 \def\Prob{\mathop{\hbox{\sfshape P}}\nolimits}
```

```
42.195 \def\nod{\mathop{\operator@font \CYRn.\CYRo.\CYRd.}\nolimits}
```

```
42.196 \def\nok{\mathop{\operator@font \CYRn.\CYRo.\CYRk.}\nolimits}
```

```
42.197 \def\Variance{\mathop{\hbox{\sfshape D}}\nolimits}
```

The macro `\ldf@finish` takes care of looking for a configuration file, setting the main language to be switched on at `\begin{document}` and resetting the category code of `@` to its original value.

```
42.198 \ldf@finish{russian}
```

```
42.199 \code
```

43 The Lower Sorbian language

The file `lsorbian.dtx`⁵⁷ It defines all the language-specific macros for Lower Sorbian.

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the `@` sign, etc.

```
43.1 (*code)
43.2 \LdfInit{lsorbian}\captionlsorbian
```

When this file is read as an option, i.e. by the `\usepackage` command, `lsorbian` will be an ‘unknown’ language in which case we have to make it known. So we check for the existence of `\l@lsorbian` to see whether we have to do something here.

```
43.3 \ifx\l@lsorbian\@undefined
43.4   \@nopatterns{Lsorbian}
43.5   \adddialect\l@lsorbian\l@usorbian\fi
```

The next step consists of defining commands to switch to (and from) the Lower Sorbian language.

`\captionlsorbian` The macro `\captionlsorbian` defines all strings used in the four standard documentclasses provided with L^AT_EX.

```
43.6 \addto\captionlsorbian{%
43.7   \def\prefacename{Zawod}%
43.8   \def\refname{Referency}%
43.9   \def\abstractname{Abstrakt}%
43.10  \def\bibname{Literatura}%
43.11  \def\chaptername{Kapitl}%
43.12  \def\appendixname{Dodawki}%
43.13  \def\contentsname{Wop\'simje\'se}%
43.14  \def\listfigurename{Zapis wobrazow}%
43.15  \def\listtablename{Zapis tabulkow}%
43.16  \def\indexname{Indeks}%
43.17  \def\figurename{Wobraz}%
43.18  \def\tablename{Tabulka}%
43.19  \def\partname{\`Z\`v el}%
43.20  \def\enclname{P\'si\`l oga}%
43.21  \def\ccname{CC}%
43.22  \def\headtoname{Komu}%
43.23  \def\pagename{Strona}%
43.24  \def\seename{gl.}%
43.25  \def\alsoname{gl.~teke}%
43.26  \def\proofname{Proof}% <-- needs translation
43.27  }%
```

`\newdatelsorbian` The macro `\newdatelsorbian` redefines the command `\today` to produce Lower Sorbian dates.

```
43.28 \def\newdatelsorbian{%
43.29 \def\today{\number\day.~\ifcase\month\or
43.30 januara\or februara\or m\`v erca\or apryla\or maja\or junija\or
43.31 julija\or awgusta\or septembra\or oktobra\or
```

⁵⁷The file described in this section has version number v1.0d and was last revised on 1996/12/23. It was written by Eduard Werner (edi@kaihh.hanse.de).


```

43.32 nowembra\or decembra\fi
43.33 \space \number\year}}

```

`\olddatelsorbian` The macro `\olddatelsorbian` redefines the command `\today` to produce old-style Lower Sorbian dates.

```

43.34 \def\olddatelsorbian{%
43.35 \def\today{\number\day.\~\ifcase\month\or
43.36 wjelikego ro\v zka\or
43.37 ma\l ego ro\v zka\or
43.38 nal\v etnika\or
43.39 jat\v sownika\or
43.40 ro\v zownika\or
43.41 sma\v znika\or
43.42 pra\v znika\or
43.43 \v znje\'nca\or
43.44 po\v znje\'nca\or
43.45 winowca\or
43.46 nazymnika\or
43.47 godownika\fi \space \number\year}}

```

The default will be the new-style dates.

```

43.48 \let\datelsorbian\newdatelsorbian

```

`\extraslsorbian` The macro `\extraslsorbian` will perform all the extra definitions needed for the
`\noextraslsorbian` lsorbian language. The macro `\noextraslsorbian` is used to cancel the actions of
`\extraslsorbian`. For the moment these macros are empty but they are defined
for compatibility with the other language definition files.

```

43.49 \addto\extraslsorbian{}
43.50 \addto\noextraslsorbian{}

```

The macro `\ldf@finish` takes care of looking for a configuration file, setting the main language to be switched on at `\begin{document}` and resetting the category code of `@` to its original value.

```

43.51 \ldf@finish{lsorbian}
43.52 \code

```

44 The Upper Sorbian language

The file `usorbian.dtx`⁵⁸ It defines all the language-specific macros for Upper Sorbian.

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the `@` sign, etc.

```
44.1 (*code)
44.2 \LdfInit{usorbian}\captionusorbian
```

When this file is read as an option, i.e. by the `\usepackage` command, `usorbian` will be an ‘unknown’ language in which case we have to make it known. So we check for the existence of `\l@usorbian` to see whether we have to do something here.

```
44.3 \ifx\l@usorbian\undefined
44.4     \nopatterns{Usorbian}
44.5     \adddialect\l@usorbian0\fi
```

The next step consists of defining commands to switch to (and from) the Upper Sorbian language.

`\captionusorbian` The macro `\captionusorbian` defines all strings used in the four standard documentclasses provided with L^AT_EX.

```
44.6 \addto\captionusorbian{%
44.7   \def\prefacename{Zawod}%
44.8   \def\refname{Referency}%
44.9   \def\abstractname{Abstrakt}%
44.10  \def\bibname{Literatura}%
44.11  \def\chaptername{Kapitl}%
44.12  \def\appendixname{Dodawki}%
44.13  \def\contentsname{Wobsah}%
44.14  \def\listfigurename{Zapis wobrazow}%
44.15  \def\listtablename{Zapis tabulkow}%
44.16  \def\indexname{Indeks}%
44.17  \def\figurename{Wobraz}%
44.18  \def\tablename{Tabulka}%
44.19  \def\partname{D\'z\ve l}%
44.20  \def\enclname{P\vr\l oha}%
44.21  \def\ccname{CC}%
44.22  \def\headtoname{Komu}%
44.23  \def\pagename{Strona}%
44.24  \def\seename{hl.}%
44.25  \def\alsoname{hl.~te\ve z}
44.26  \def\proofname{Proof}% <-- needs translation
44.27  }%
```

`\newdateusorbian` The macro `\newdateusorbian` redefines the command `\today` to produce Upper Sorbian dates.

```
44.28 \def\newdateusorbian{%
44.29 \def\today{\number\day.~\ifcase\month\or
44.30 januara\or februara\or m\vr erca\or apryla\or meje\or juniya\or
44.31 julija\or awgusta\or septembra\or oktobra\or
```

⁵⁸The file described in this section has version number v1.0e and was last revised on 1996/12/23. It was written by Eduard Werner (edi@kaihh.hanse.de).

```

44.32 nowembra\or decembra\fi
44.33 \space \number\year}}

```

`\olddateusorbian` The macro `\olddateusorbian` redefines the command `\today` to produce old-style Upper Sorbian dates.

```

44.34 \def\olddateusorbian{%
44.35 \def\today{\number\day.\~\ifcase\month\or
44.36 wulkeho r\'o\v zka\or ma\l eho r\'o\v zka\or nal\v etnika\or
44.37 jutrownika\or r\'o\v zownika\or sma\v znika\or pra\v znika\or
44.38 \v znjenca\or po\v znjenca\or winowca\or nazymnika\or
44.39 hodownika\fi \space \number\year}}

```

The default will be the new-style dates.

```

44.40 \let\dateusorbian\newdateusorbian

```

`\extrasusorbian` The macro `\extrasusorbian` will perform all the extra definitions needed for the Upper Sorbian language. It's pirated from `germanb.sty`. The macro `\noextrasusorbian` is used to cancel the actions of `\extrasusorbian`.

Because for Upper Sorbian (as well as for Dutch) the " character is made active. This is done once, later on its definition may vary.

```

44.41 \initiate@active@char{"}
44.42 \addto\extrasusorbian{\languageshorthands{usorbian}}
44.43 \addto\extrasusorbian{\bbl@activate{}}
44.44 %\addto\noextrasusorbian{\bbl@deactivate{}}

```

In order for \TeX to be able to hyphenate German Upper Sorbian words which contain 'ß' we have to give the character a nonzero `\lccode` (see Appendix H, the \TeX book).

```

44.45 \addto\extrasusorbian{\babel@savevariable{\lccode'\^^Y}%
44.46 \lccode'\^^Y'\^^Y}

```

The umlaut accent macro `\"` is changed to lower the umlaut dots. The redefinition is done with the help of `\umlautlow`.

```

44.47 \addto\extrasusorbian{\babel@save"\umlautlow}
44.48 \addto\noextrasusorbian{\umlauthigh}

```

The Upper Sorbian hyphenation patterns can be used with `\lefthyphenmin` and `\righthyphenmin` set to 2.

```

44.49 \def\usorbianhyphenmins{\tw@\tw@}

```

`\dq` We save the original double quote character in `\dq` to keep it available, the math accent `\"` can now be typed as `"`. Also we store the original meaning of the command `\"` for future use.

```

44.50 \begingroup \catcode'\ "12
44.51 \def\x{\endgroup
44.52 \def\@SS{\mathchar"7019 }
44.53 \def\dq{"}}
44.54 \x

```

Now we can define the doublequote macros: the umlauts,

```

44.55 \declare@shorthand{usorbian}{a}{\textormath{"a}{\ddot a}}
44.56 \declare@shorthand{usorbian}{o}{\textormath{"o}{\ddot o}}
44.57 \declare@shorthand{usorbian}{u}{\textormath{"u}{\ddot u}}

```

```

44.58 \declare@shorthand{usorbian}{\textmath{"A"}{\ddot A}}
44.59 \declare@shorthand{usorbian}{\textmath{"O"}{\ddot O}}
44.60 \declare@shorthand{usorbian}{\textmath{"U"}{\ddot U}}

tremas,
44.61 \declare@shorthand{usorbian}{\textmath{"e"}{\ddot e}}
44.62 \declare@shorthand{usorbian}{\textmath{"E"}{\ddot E}}
44.63 \declare@shorthand{usorbian}{\textmath{"i"}{\ddot i}{\imath}}
44.64 \declare@shorthand{usorbian}{\textmath{"I"}{\ddot I}}

usorbian es-zet (sharp s),
44.65 \declare@shorthand{usorbian}{\textmath{\ss}{\@SS}}
44.66 \declare@shorthand{usorbian}{\textmath{S}{SS}}

german and french quotes,
44.67 \declare@shorthand{usorbian}{\textmath{"'"}{}}
44.68 \textmath{\quotedblbase}{\mbox{\quotedblbase}}
44.69 \declare@shorthand{usorbian}{\textmath{"'"}{}}
44.70 \textmath{\textquotedblleft}{\mbox{\textquotedblleft}}
44.71 \declare@shorthand{usorbian}{\textmath{"<"}{}}
44.72 \textmath{\guillemotleft}{\mbox{\guillemotleft}}
44.73 \declare@shorthand{usorbian}{\textmath{">"}{}}
44.74 \textmath{\guillemotright}{\mbox{\guillemotright}}

discretionary commands
44.75 \declare@shorthand{usorbian}{\textmath{\bbl@disc ck}{c}}
44.76 \declare@shorthand{usorbian}{\textmath{\bbl@disc CK}{C}}
44.77 \declare@shorthand{usorbian}{\textmath{\bbl@disc f{ff}}{f}}
44.78 \declare@shorthand{usorbian}{\textmath{\bbl@disc F{FF}}{F}}
44.79 \declare@shorthand{usorbian}{\textmath{\bbl@disc l{ll}}{l}}
44.80 \declare@shorthand{usorbian}{\textmath{\bbl@disc L{LL}}{L}}
44.81 \declare@shorthand{usorbian}{\textmath{\bbl@disc m{mm}}{m}}
44.82 \declare@shorthand{usorbian}{\textmath{\bbl@disc M{MM}}{M}}
44.83 \declare@shorthand{usorbian}{\textmath{\bbl@disc n{nn}}{n}}
44.84 \declare@shorthand{usorbian}{\textmath{\bbl@disc N{NN}}{N}}
44.85 \declare@shorthand{usorbian}{\textmath{\bbl@disc p{pp}}{p}}
44.86 \declare@shorthand{usorbian}{\textmath{\bbl@disc P{PP}}{P}}
44.87 \declare@shorthand{usorbian}{\textmath{\bbl@disc t{tt}}{t}}
44.88 \declare@shorthand{usorbian}{\textmath{\bbl@disc T{TT}}{T}}

and some additional commands:
44.89 \declare@shorthand{usorbian}{\penalty\@M\allowhyphens}
44.90 \declare@shorthand{usorbian}{\textmath{\penalty\@M\discretionary{-}{\kern.03em}%
\allowhyphens}}{}}
44.91 \textmath{\penalty\@M\discretionary{-}{\kern.03em}%
\allowhyphens}}{}}
44.92 \allowhyphens}{}
44.93 \declare@shorthand{usorbian}{\hspace{0.5em}}

```

\mdqon All that's left to do now is to define a couple of commands for reasons of compatibility with german.sty.

```

\ck44.94 \def\mdqon{\bbl@activate{}}
44.95 \def\mdqoff{\bbl@deactivate{}}
44.96 \def\ck{\allowhyphens\discretionary{k-}{k}{ck}\allowhyphens}

```

The macro \ldf@finish takes care of looking for a configuration file, setting the main language to be switched on at \begin{document} and resetting the category code of @ to its original value.

```
44.97 \ldf@finish{usorbian}  
44.98 
```

45 The Turkish language

The file `turkish.dtx`⁵⁹ defines all the language definition macros for the Turkish language⁶⁰.

Turkish typographic rules specify that a little ‘white space’ should be added before the characters ‘:’, ‘!’ and ‘=’. In order to insert this white space automatically these characters are made ‘active’. Also `\frenhspace` is set.

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the `@` sign, etc.

45.1 `(*code)`

45.2 `\LdfInit{turkish}\captionsturkish`

When this file is read as an option, i.e. by the `\usepackage` command, `turkish` could be an ‘unknown’ language in which case we have to make it known. So we check for the existence of `\l@turkish` to see whether we have to do something here.

45.3 `\ifx\l@turkish\@undefined`

45.4 `\@nopatterns{Turkish}`

45.5 `\adddialect\l@turkish0\fi`

The next step consists of defining commands to switch to (and from) the Turkish language.

`\captionsturkish` The macro `\captionsturkish` defines all strings used in the four standard documentclasses provided with L^AT_EX.

```

45.6 \addto\captionsturkish{%
45.7   \def\prefacename{Preface}% <-- This needs translation!!
45.8   \def\refname{Ba\c svurulan Kitaplar}%
45.9   \def\abstractname{Konu}%
45.10  \def\bibName{Bibliografi}%
45.11  \def\chaptername{Anab\ol\um}%
45.12  \def\appendixname{Appendix}%
45.13  \def\contentsname{\.I\c cindekiler}%
45.14  \def\listfigurename{\c Sekiller Listesi}%
45.15  \def\listtablename{Tablolar\i{n} Listesi}%
45.16  \def\indexname{\.Index}%
45.17  \def\figurename{\c Sekiller}%
45.18  \def\tablename{Tablo}%
45.19  \def\partname{B\ol\um}%
45.20  \def\enclname{Ekler}%
45.21  \def\ccname{G\onderen}%
45.22  \def\headtoname{Al\i{c}i}%
45.23  \def\pagename{Sayfa}%
45.24  \def\subjectname{To}% <-- This needs translation!!
45.25  \def\seename{see}% <-- This needs translation!!
45.26  \def\alsosome{see also}% <-- This needs translation!!
45.27  \def\proofname{Proof}% <-- This needs translation!!
45.28 }%
```

⁵⁹The file described in this section has version number v1.2i and was last revised on 1996/12/23.

⁶⁰Mustafa Burc, `z6001@rziris01.rz.uni-hamburg.de` provided the code for this file. It is based on the work by Pierre Mackay

`\dateturkish` The macro `\dateturkish` redefines the command `\today` to produce Turkish dates.

```
45.29 \def\dateturkish{%
45.30   \def\today{\number\day.\~\ifcase\month\or
45.31     Ocak\or \c Subat\or Mart\or Nisan\or May\i{s}\or Haziran\or
45.32     Temmuz\or A\u gustos\or Eyl\"ul\or Ekim\or Kas\i{m}\or
45.33     Aral\i{k}\fi
45.34   \space\number\year}}
```

`\extrasturkish` The macro `\extrasturkish` will perform all the extra definitions needed for the Turkish language. The macro `\noextrasturkish` is used to cancel the actions of `\extrasturkish`.

Turkish typographic rules specify that a little ‘white space’ should be added before the characters ‘:’, ‘!’ and ‘=’. In order to insert this white space automatically these characters are made `\active`, so they have to be treated in a special way.

```
45.35 \initiate@active@char{:}
45.36 \initiate@active@char{!}
45.37 \initiate@active@char{=}
```

We specify that the turkish group of shorthands should be used.

```
45.38 \addto\extrasturkish{\languageshorthands{turkish}}
```

These characters are ‘turned on’ once, later their definition may vary.

```
45.39 \addto\extrasturkish{%
45.40   \bbl@activate{:}\bbl@activate{!}\bbl@activate{=}}
```

For Turkish texts `\frenchspacing` should be in effect. We make sure this is the case and reset it if necessary.

```
45.41 \addto\extrasturkish{\bbl@frenchspacing}
45.42 \addto\noextrasturkish{\bbl@nonfrenchspacing}
```

`\turkish@sh@!` The definitions for the three active characters were made using intermediate macros. These are defined now. The insertion of extra ‘white space’ should only happen outside math mode, hence the check `\ifmmode` in the macros.

```
45.43 \declare@shorthand{turkish}{:}{%
45.44   \ifmmode
45.45     \string:%
45.46   \else\relax
45.47     \ifhmode
45.48       \ifdim\lastskip>\z@
45.49         \unskip\penalty\@M\thinspace
45.50       \fi
45.51     \fi
45.52     \string:%
45.53   \fi}
45.54 \declare@shorthand{turkish}{!}{%
45.55   \ifmmode
45.56     \string!%
45.57   \else\relax
45.58     \ifhmode
45.59       \ifdim\lastskip>\z@
45.60         \unskip\penalty\@M\thinspace
45.61       \fi
```

```

45.62     \fi
45.63     \string!%
45.64     \fi}
45.65 \declare@shorthand{turkish}{=}{%
45.66     \ifmmode
45.67         \string=%
45.68     \else\relax
45.69         \ifhmode
45.70             \ifdim\lastskip>\z@
45.71                 \unskip\kern\fontdimen2\font
45.72                 \kern-1.4\fontdimen3\font
45.73             \fi
45.74         \fi
45.75         \string=%
45.76     \fi}

```

The macro `\ldf@finish` takes care of looking for a configuration file, setting the main language to be switched on at `\begin{document}` and resetting the category code of `@` to its original value.

```

45.77 \ldf@finish{turkish}
45.78 \code

```


46 The Bahasa language

The file `bahasa.dtx`⁶¹ defines all the language definition macros for the bahasa indonesia / bahasa melayu language. Bahasa just means ‘language’ in bahasa indonesia / bahasa melayu. Since both national versions of the language use the same writing, although differing in pronunciation, this file can be used for both languages.

For this language currently no special definitions are needed or available.

The macro `\LdfInit` takes care of preventing that this file is loaded more than once, checking the category code of the `@` sign, etc.

46.1 `(*code)`

46.2 `\LdfInit{bahasa}\captionsbahasa`

When this file is read as an option, i.e. by the `\usepackage` command, `bahasa` could be an ‘unknown’ language in which case we have to make it known. So we check for the existence of `\l@bahasa` to see whether we have to do something here.

46.3 `\ifx\l@bahasa\@undefined`

46.4 `\@nopatterns{Bahasa}`

46.5 `\adddialect\l@bahasa0\fi`

The next step consists of defining commands to switch to (and from) the Bahasa language.

`\captionsbahasa` The macro `\captionsbahasa` defines all strings used in the four standard documentclasses provided with L^AT_EX.

```
46.6 \addto\captionsbahasa{%
46.7   \def\prefacename{Pendahuluan}%
46.8   \def\refname{Pustaka}%
46.9   \def\abstractname{Ringkasan}% (sometime it's called 'intisari'
46.10                                % or 'ikhtisar')
46.11   \def\bibname{Bibliografi}%
46.12   \def\chaptername{Bab}%
46.13   \def\appendixname{Lampiran}%
46.14   \def\contentsname{Daftar Isi}%
46.15   \def\listfigurename{Daftar Gambar}%
46.16   \def\listtablename{Daftar Tabel}%
46.17 % Glossary: Daftar Istilah
46.18   \def\indexname{Indeks}%
46.19   \def\figurename{Gambar}%
46.20   \def\tablename{Tabel}%
46.21   \def\partname{Bagian}%
46.22 % Subject: Subyek
46.23 % From: Dari
46.24   \def\enclname{Lampiran}%
46.25   \def\ccname{cc}%
46.26   \def\headtoname{Kepada}%
46.27   \def\pagename{Halaman}%
46.28 % Notes (Endnotes): Catatan
46.29   \def\seename{lihat}%
46.30   \def\alsoname{lihat juga}%
46.31   \def\proofname{Bukti}%
```

⁶¹The file described in this section has version number v1.0e and was last revised on 1996/12/23.

46.32 }

`\datebahasa` The macro `\datebahasa` redefines the command `\today` to produce Bahasa dates.

```
46.33 \def\datebahasa{%
46.34   \def\today{\number\day~\ifcase\month\or
46.35     Januari\or Februari\or Maret\or April\or Mei\or Juni\or
46.36     Juli\or Agustus\or September\or Oktober\or Nopember\or Desember\fi
46.37   \space \number\year}}
```

`\extrasbahasa` The macro `\extrasbahasa` will perform all the extra definitions needed for the
`\noextrasbahasa` Bahasa language. The macro `\extrasbahasa` is used to cancel the actions of
`\extrasbahasa`. For the moment these macros are empty but they are defined for
compatibility with the other language definition files.

```
46.38 \addto\extrasbahasa{}
46.39 \addto\noextrasbahasa{}
```

`\bahasahyphenmins` The bahasa hyphenation patterns should be used with `\lefthyphenmin` set to 2
and `\righthyphenmin` set to 2.

```
46.40 \def\bahasahyphenmins{\tw@\tw@}
```

The macro `\ldf@finish` takes care of looking for a configuration file, setting
the main language to be switched on at `\begin{document}` and resetting the
category code of `@` to its original value.

```
46.41 \ldf@finish{bahasa}
46.42 \code
```

47 Support for formats based on plain $\mathrm{T}_{\mathrm{E}}\mathrm{X}$

The following code duplicates or emulates parts of $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X} 2_{\varepsilon}$ that are needed for `babel`.

```
47.1 (*code)
47.2 \ifx\adddialect\@undefined
```

When `\adddialect` is still undefined we are making a format. In that case only the first part of this file is needed.

```
47.3 \def\@empty{}
```

We need to define `\loadlocalcfg` for plain users as the $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ definition uses `\InputIfFileExists`.

```
47.4 \def\loadlocalcfg#1{%
47.5   \openin0#1.cfg
47.6   \ifeof0
47.7     \closein0
47.8   \else
47.9     \closein0
47.10    {\immediate\write16{*****}%
47.11     \immediate\write16{* Local config file #1.cfg used}%
47.12     \immediate\write16{*}%
47.13    }
47.14    \input #1.cfg\relax
47.15  \fi
```

We have to execute `\@endofldf` in this case

```
47.16 \endofldf
47.17 }
```

We want to add a message to the message $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ 2.09 puts in the `\everyjob` register. This could be done by the following code:

```
\let\orgeveryjob\everyjob
\def\everyjob#1{%
  \orgeveryjob{#1}%
  \orgeveryjob\expandafter{\the\orgeveryjob\immediate\write16{%
    hyphenation patterns for \the\loaded@patterns loaded.}}%
  \let\everyjob\orgeveryjob\let\orgeveryjob\@undefined}
```

The code above redefines the control sequence `\everyjob` in order to be able to add something to the current contents of the register. This is necessary because the processing of hyphenation patterns happens long before $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ fills the register. There are some problems with this approach though.

- When someone wants to use several hyphenation patterns with $\mathrm{S}^{\mathrm{U}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ the above scheme won't work. The reason is that $\mathrm{S}^{\mathrm{U}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ overwrites the contents of the `\everyjob` register with its own message.
- Plain $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ does not use the `\everyjob` register so the message would not be displayed.

To circumvent this a 'dirty trick' can be used. As this code is only processed when creating a new format file there is one command that is sure to be used,

\dump. Therefore the original \dump is saved in \orig@dump and a new definition is supplied.

```
47.18 \let\orig@dump=\dump
47.19 \def\dump{%
```

To make sure that L^AT_EX 2.09 executes the \@begindocumenthook we would want to alter \begin{document}, but as this done too often already, we add the new code at the front of \@preamblecmds. But we can only do that after it has been defined, so we add this peice of code to \dump.

```
47.20 \ifx\@ztryfc\undefined
47.21 \else
47.22 \toks0=\expandafter{\@preamblecmds}
47.23 \edef\@preamblecmds{\noexpand\@begindocumenthook\the\toks0}
47.24 \def\@begindocumenthook{}
47.25 \fi
```

This new definition starts by adding an instruction to write a message on the terminal and in the transcript file to inform the user of the preloaded hyphenation patterns.

```
47.26 \everyjob\expandafter{\the\everyjob%
47.27 \immediate\write16{\the\toks8 loaded.}}%
```

Then everything is restored to the old situation and the format is dumped.

```
47.28 \let\dump\orig@dump\let\orig@dump\undefined\dump}
47.29 \expandafter\endinput
47.30 \fi
```

The rest of this file is not processed by iniT_EX but during the normal document run. A number of L^AT_EX macro's that are needed later on.

```
47.31 \long\def\@firstofone#1{#1}
47.32 \long\def\@firstoftwo#1#2{#1}
47.33 \def\@ifstar#1{\@ifnextchar *{\@firstoftwo{#1}}}
47.34 \def\@star@or@long#1{%
47.35 \@ifstar
47.36 {\let\l@ngrel@x\relax#1}%
47.37 {\let\l@ngrel@x\long#1}}
47.38 \let\l@ngrel@x\relax
47.39 \def\@car#1#2\@nil{#1}
47.40 \def\@cdr#1#2\@nil{#2}
47.41 \let\@typeset@protect\relax
47.42 \long\def\@gobble#1{}
47.43 \edef\@backslashchar{\expandafter\@gobble\string\\}
47.44 \def\@strip@prefix#1>{}
47.45 \def\@g@addto@macro#1#2{%
47.46 \toks0\expandafter{#1#2}%
47.47 \xdef#1{\the\toks0}}
47.48 \def\@namedef#1{\expandafter\def\csname #1\endcsname}
```

L^AT_EX 2_ε has the command \@onlypreamble which adds commands to a list of commands that are no longer needed after \begin{document}.

```
47.49 \ifx\@preamblecmds\undefined
47.50 \def\@preamblecmds{}
47.51 \fi
47.52 \def\@onlypreamble#1{%
47.53 \expandafter\gdef\expandafter\@preamblecmds\expandafter{%
```

```

47.54 \preamblecmds\do#1}}
47.55 \@onlypreamble\@onlypreamble

Mimick LATEX's \AtBeginDocument; for this to work the user needs to add
\begindocument to his file.

47.56 \def\begindocument{%
47.57 \begindocumenthook
47.58 \global\let\begindocumenthook\@undefined
47.59 \def\do##1{\global\let ##1\@undefined}%
47.60 \preamblecmds
47.61 \global\let\do\noexpand
47.62 }

47.63 \ifx\begindocumenthook\@undefined
47.64 \def\begindocumenthook{}
47.65 \fi
47.66 \@onlypreamble\begindocumenthook
47.67 \def\AtBeginDocument{\g@addto@macro\begindocumenthook}

We also have to mimick LATEX's \AtEndOfPackage. Our replacement macro is
much simpler; it stores its argument in \@endoflfd.

47.68 \def\AtEndOfPackage#1{\g@addto@macro\endoflfd{#1}}
47.69 \@onlypreamble\AtEndOfPackage
47.70 \def\@endoflfd{}
47.71 \@onlypreamble\endoflfd

LATEX needs to be able to switch off writing to its auxiliary files; plain doesn't
have them by default.

47.72 \ifx\bye\@undefined\else
47.73 \expandafter\let\csname if@files\expandafter\csname iffalse\endcsname
47.74 \fi

Mimick LATEX's commands to define control sequences.

47.75 \def\newcommand{\@star@or@long\new@command}
47.76 \def\new@command#1{%
47.77 \testopt{\@newcommand#1}0}
47.78 \def\@newcommand#1[#2]{%
47.79 \ifnextchar [{\@xargdef#1[#2]}%
47.80 {\@argdef#1[#2]}}
47.81 \long\def\@argdef#1[#2]#3{%
47.82 \@yargdef#1\@ne{#2}{#3}}
47.83 \long\def\@xargdef#1[#2][#3]#4{%
47.84 \expandafter\def\expandafter#1\expandafter{%
47.85 \expandafter\@protected@testopt\expandafter #1%
47.86 \csname\string#1\expandafter\endcsname{#3}}%
47.87 \expandafter\@yargdef \csname\string#1\endcsname
47.88 \tw@{#2}{#4}}
47.89 \long\def\@yargdef#1#2#3{%
47.90 \@tempcnta#3\relax
47.91 \advance \@tempcnta \@ne
47.92 \let\@hash@\relax
47.93 \edef\reserved@a{\ifx#2\tw@ [\@hash@1]\fi}%
47.94 \@tempcntb #2%
47.95 \@whilenum\@tempcntb <\@tempcnta
47.96 \do{%
47.97 \edef\reserved@a{\reserved@a\@hash@\the\@tempcntb}%

```

```

47.98      \advance\@tempcntb \@ne}%
47.99      \let\@hash@###%
47.100     \l@ngrel@x\expandafter\def\expandafter#1\reserved@a}
47.101     \let\providecommand\newcommand
47.102
47.103     \def\DeclareRobustCommand{\@star@or@long\declare@robustcommand}
47.104     \def\declare@robustcommand#1{%
47.105         \edef\reserved@a{\string#1}%
47.106         \def\reserved@b{#1}%
47.107         \edef\reserved@b{\expandafter\strip@prefix\meaning\reserved@b}%
47.108         \edef#1{%
47.109             \ifx\reserved@a\reserved@b
47.110                 \noexpand\x@protect
47.111                 \noexpand#1%
47.112             \fi
47.113             \noexpand\protect
47.114             \expandafter\noexpand\csname
47.115                 \expandafter\@gobble\string#1 \endcsname
47.116         }%
47.117         \expandafter\new@command\csname
47.118             \expandafter\@gobble\string#1 \endcsname
47.119     }
47.120     \def\x@protect#1{%
47.121         \ifx\protect\@typeset@protect\else
47.122             \@x@protect#1%
47.123         \fi
47.124     }
47.125     \def\@x@protect#1\fi#2#3{%
47.126         \fi\protect#1%
47.127     }

```

L^AT_EX has a macro to check whether a certain package was loaded with specific options. The command has two extra arguments which are code to be executed in either the true or false case. This is used to detect whether the document needs one of the accents to be activated (`activegrave` and `activeacute`). For plain T_EX we assume that the user wants them to be active by default. Therefore the only thing we do is execute the third argument (the code for the true case).

```

47.128 \def\ifpackagewith#1#2#3#4{%
47.129     #3}

```

For the following code we need to make sure that the commands `\newcommand` and `\providecommand` exist with some sensible definition. They are not fully equivalent to their L^AT_EX 2_ε versions; just enough to make things work in plain T_EX environments.

```

47.130 \ifx\@tempcnta\@undefined
47.131     \csname newcount\endcsname\@tempcnta\relax
47.132 \fi
47.133 \ifx\@tempcntb\@undefined
47.134     \csname newcount\endcsname\@tempcntb\relax
47.135 \fi

```

To prevent wasting two counters in L^AT_EX 2.09 (because counters with the same name are allocated later by it) we reset the counter that holds the next free counter (`\count10`).

```

47.136 \ifx\bye\@undefined
47.137 \advance\count10 by -2\relax
47.138 \fi
47.139 \ifx\@ifnextchar\@undefined
47.140 \def\@ifnextchar#1#2#3{%
47.141 \let\reserved@d=#1%
47.142 \def\reserved@a{#2}\def\reserved@b{#3}%
47.143 \futurelet\@let@token\@ifnch}
47.144 \def\@ifnch{%
47.145 \ifx\@let@token\@sptoken
47.146 \let\reserved@c\@xifnch
47.147 \else
47.148 \ifx\@let@token\reserved@d
47.149 \let\reserved@c\reserved@a
47.150 \else
47.151 \let\reserved@c\reserved@b
47.152 \fi
47.153 \fi
47.154 \reserved@c}
47.155 \def\:{\let\@sptoken= } \: % this makes \@sptoken a space token
47.156 \def\:{\@xifnch} \expandafter\def\:{\futurelet\@let@token\@ifnch}
47.157 \fi
47.158 \def\@testopt#1#2{%
47.159 \@ifnextchar[#{#1}{#1[#2]}}
47.160 \def\@protected@testopt#1{%%
47.161 \ifx\protect\@typeset@protect
47.162 \expandafter\@testopt
47.163 \else
47.164 \@x@protect#1%
47.165 \fi}
47.166 \long\def\@whilenum#1\do #2{\ifnum #1\relax #2\relax\@iwhilenum{#1\relax
47.167 #2\relax}\fi}
47.168 \long\def\@iwhilenum#1{\ifnum #1\expandafter\@iwhilenum
47.169 \else\expandafter\@gobble\fi{#1}}

```

Code from `ltoutenc.dtx`, adapted for use in the plain $\text{T}_{\text{E}}\text{X}$ environment.

```

47.170 \def\DeclareTextCommand{%
47.171 \@dec@text@cmd\providecommand
47.172 }
47.173 \def\ProvideTextCommand{%
47.174 \@dec@text@cmd\providecommand
47.175 }
47.176 \def\DeclareTextSymbol#1#2#3{%
47.177 \@dec@text@cmd\chardef#1{#2}#3\relax
47.178 }
47.179 \def\@dec@text@cmd#1#2#3{%
47.180 \expandafter\def\expandafter#2%
47.181 \expandafter{%
47.182 \csname#3-cmd\expandafter\endcsname
47.183 \expandafter#2%
47.184 \csname#3\string#2\endcsname
47.185 }%
47.186 % \let\@ifdefinable\@rc@ifdefinable
47.187 \expandafter#1\csname#3\string#2\endcsname
47.188 }

```

```

47.189 \def\@current@cmd#1{%
47.190   \ifx\protect\@typeset@protect\else
47.191     \noexpand#1\expandafter\@gobble
47.192   \fi
47.193 }
47.194 \def\@changed@cmd#1#2{%
47.195   \ifx\protect\@typeset@protect
47.196     \expandafter\ifx\csname\cf@encoding\string#1\endcsname\relax
47.197       \expandafter\ifx\csname ?\string#1\endcsname\relax
47.198         \expandafter\def\csname ?\string#1\endcsname{%
47.199           \@changed@x@err{#1}%
47.200         }%
47.201       \fi
47.202     \global\expandafter\let
47.203     \csname\cf@encoding\string#1\expandafter\endcsname
47.204     \csname ?\string#1\endcsname
47.205   \fi
47.206   \csname\cf@encoding\string#1%
47.207   \expandafter\endcsname
47.208   \else
47.209     \noexpand#1%
47.210   \fi
47.211 }
47.212 \def\@changed@x@err#1{%
47.213   \errhelp{Your command will be ignored, type <return> to proceed}%
47.214   \errmessage{Command \protect#1 undefined in encoding \cf@encoding}}
47.215 \def\DeclareTextCommandDefault#1{%
47.216   \DeclareTextCommand#1?%
47.217 }
47.218 \def\ProvideTextCommandDefault#1{%
47.219   \ProvideTextCommand#1?%
47.220 }
47.221 \expandafter\let\csname OT1-cmd\endcsname\@current@cmd
47.222 \expandafter\let\csname?-cmd\endcsname\@changed@cmd
47.223 \def\DeclareTextAccent#1#2#3{%
47.224   \DeclareTextCommand#1{#2}[1]{\accent#3 ##1}
47.225 }
47.226 \def\DeclareTextCompositeCommand#1#2#3#4{%
47.227   \expandafter\let\expandafter\reserved@a\csname#2\string#1\endcsname
47.228   \edef\reserved@b{\string##1}%
47.229   \edef\reserved@c{%
47.230     \expandafter\@strip@args\meaning\reserved@a:-\@strip@args}%
47.231   \ifx\reserved@b\reserved@c
47.232     \expandafter\expandafter\expandafter\ifx
47.233     \expandafter\@car\reserved@a\relax\relax\@nil
47.234     \@text@composite
47.235   \else
47.236     \edef\reserved@b##1{%
47.237       \def\expandafter\noexpand
47.238       \csname#2\string#1\endcsname####1{%
47.239         \noexpand\@text@composite
47.240         \expandafter\noexpand\csname#2\string#1\endcsname
47.241         ####1\noexpand\@empty\noexpand\@text@composite
47.242         {##1}%

```



```

47.243         }%
47.244     }%
47.245     \expandafter\reserved@b\expandafter{\reserved@a{##1}}%
47.246     \fi
47.247     \expandafter\def\csname\expandafter\string\csname
47.248         #2\endcsname\string#1-\string#3\endcsname{#4}
47.249     \else
47.250         \errhelp{Your command will be ignored, type <return> to proceed}%
47.251         \errmessage{\string\DeclareTextCompositeCommand\space used on
47.252             inappropriate command \protect#1}
47.253     \fi
47.254 }
47.255 \def\@text@composite#1#2#3\@text@composite{%
47.256     \expandafter\@text@composite@x
47.257         \csname\string#1-\string#2\endcsname
47.258 }
47.259 \def\@text@composite@x#1#2{%
47.260     \ifx#1\relax
47.261         #2%
47.262     \else
47.263         #1%
47.264     \fi
47.265 }
47.266 %
47.267 \def\@strip@args#1:#2-#3\@strip@args{#2}
47.268 \def\DeclareTextComposite#1#2#3#4{%
47.269     \def\reserved@a{\DeclareTextCompositeCommand#1{#2}{#3}}%
47.270     \bgroup
47.271         \lccode'\@=#4%
47.272         \lowercase{%
47.273     \egroup
47.274         \reserved@a \@%
47.275     }%
47.276 }
47.277 %
47.278 \def\UseTextSymbol#1#2{%
47.279 %     \let\@curr@enc\cf@encoding
47.280 %     \@use@text@encoding{#1}%
47.281     #2%
47.282 %     \@use@text@encoding\@curr@enc
47.283 }
47.284 \def\UseTextAccent#1#2#3{%
47.285 %     \let\@curr@enc\cf@encoding
47.286 %     \@use@text@encoding{#1}%
47.287 %     #2{\@use@text@encoding\@curr@enc\selectfont#3}%
47.288 %     \@use@text@encoding\@curr@enc
47.289 }
47.290 \def\@use@text@encoding#1{%
47.291 %     \edef\f@encoding{#1}%
47.292 %     \xdef\font@name{%
47.293 %         \csname\curr@fontshape/\f@size\endcsname
47.294 %     }%
47.295 %     \pickup@font
47.296 %     \font@name

```

```

47.297 %    \@@enc@update
47.298 }
47.299 \def\DeclareTextSymbolDefault#1#2{%
47.300     \DeclareTextCommandDefault#1{\UseTextSymbol{#2}#1}%
47.301 }
47.302 \def\DeclareTextAccentDefault#1#2{%
47.303     \DeclareTextCommandDefault#1{\UseTextAccent{#2}#1}%
47.304 }
47.305 \def\cf@encoding{OT1}

```

Currently we only use the $\text{\LaTeX} 2_{\epsilon}$ method for accents for those that are known to be made active in *some* language definition file.

```

47.306 \DeclareTextAccent{"}{OT1}{127}
47.307 \DeclareTextAccent{'}{OT1}{19}
47.308 \DeclareTextAccent{`}{OT1}{94}
47.309 \DeclareTextAccent{\'}{OT1}{18}
47.310 \DeclareTextAccent{\~}{OT1}{126}

```

The following two control sequences are used in `babel.def` but are not defined for PLAIN \TeX .

```

47.311 \DeclareTextSymbol{\textquotedblright}{OT1}{'\'}
47.312 \DeclareTextSymbol{\textquoteright}{OT1}{'\'}
47.313 \DeclareTextSymbol{\i}{OT1}{16}
47.314 \DeclareTextSymbol{\ss}{OT1}{25}

```

For a couple of languages we need the \LaTeX -control sequence `\scriptsize` to be available. Because plain \TeX doesn't have such a sophisticated font mechanism as \LaTeX has, we just `\let` it to `\sevenrm`.

```

47.315 \ifx\scriptsize\@undefined
47.316     \let\scriptsize\sevenrm
47.317 \fi
47.318 \</code>

```

Index

Numbers written in *italic* refer to the page where the corresponding entry is described, the ones underlined to the code line of the definition, the rest to the code lines where the entry is used.

Symbols	
\- ...	<u>16.121</u> , <u>26.107</u> , <u>28.135</u> , <u>29.85</u> , <u>34.62</u>
\@vpageref ...	<u>10.938</u>
\@acute	<u>27.59</u> , <u>28.67</u> , <u>29.55</u>
\@bibitem	... <u>10.921</u>
\@citex	... <u>10.915</u>
\@grave	... <u>28.67</u>
\@lbibitem	... <u>10.923</u>
\@nolanerr	... <u>10.192</u>
\@nopatterns	... <u>10.192</u>
\@testdef	... <u>10.903</u>
\@tilde	... <u>27.59</u> , <u>29.55</u>
\@trema	... <u>16.102</u>
\@umlaut	... <u>27.59</u> , <u>28.67</u> , <u>29.55</u>
A	
\active@prefix	... <u>10.442</u>
\adddialect	... <u>7</u> , <u>10.37</u>
\addlanguage	... <u>7</u> , <u>10.25</u>
\addto	... <u>9</u> , <u>10.550</u>
\afrikaanshyphenmins	... <u>16.100</u>
\aliasshorthand	<u>3</u> , <u>10.487</u>
\allowhyphens	... <u>9</u> , <u>10.562</u>
\Aob	... <u>39.50</u>
\aob	... <u>39.50</u>
\Asbuk	... <u>42.171</u>
\asbuk	... <u>42.178</u>
\AutoSpaceBeforeFDP	... <u>24.143</u>
B	
\babel@beginsave	<u>10.525</u>
\babel@save	... <u>9</u> , <u>10.528</u>
\babel@savecnt	... <u>10.525</u>
\babel@savevariable	... <u>9</u> , <u>10.537</u>
\bahasahyphenmins	<u>46.40</u>
\bbl@activate	... <u>8</u> , <u>10.449</u>
\bbl@add@special	... <u>8</u> , <u>10.293</u>
\bbl@afterelse	... <u>10.314</u>
\bbl@afterfi	... <u>10.314</u>
\bbl@deactivate	<u>8</u> , <u>10.455</u>
\bbl@disc	... <u>10.569</u>
\bbl@firstcs	... <u>10.461</u>
\bbl@frenchguillemets	... <u>24.172</u>
\bbl@frenchindent	<u>24.229</u>
\bbl@frenchitems	<u>24.194</u>
\bbl@frenchlistspacing	... <u>24.194</u>
\bbl@frenchspacing	... <u>9</u> , <u>10.542</u>
\bbl@main@language	... <u>10.177</u>
\bbl@nonfrenchguillemets	... <u>24.172</u>
\bbl@nonfrenchindent	... <u>24.229</u>
\bbl@nonfrenchitems	... <u>24.194</u>
\bbl@nonfrenchlistspacing	... <u>24.194</u>
\bbl@nonfrenchspacing	... <u>9</u> , <u>10.542</u>
\bbl@pr@m@s	... <u>10.497</u>
\bbl@redefine	... <u>10.878</u>
\bbl@redefine@long	... <u>10.884</u>
\bbl@redefineroobust	... <u>10.889</u>
\bbl@remove@special	... <u>8</u> , <u>10.303</u>
\bbl@scndcs	... <u>10.461</u>
\bibcite	... <u>10.919</u>
bbl@test@token	... <u>10.321</u>
\breton@sh@:	... <u>19.58</u>
\breton@sh@;	... <u>19.47</u>
\breton@sh@?	... <u>19.74</u>
\breton@sh@@	... <u>19.58</u>
\bsc	... <u>24.338</u>
C	
\captionsafrikaans	<u>16.63</u>
\captionsaustrian	<u>18.12</u>
\captionsbahasa	... <u>46.6</u>
\captionsbrazil	... <u>26.64</u>
\captionsbreton	... <u>19.6</u>
\captionscatalan	... <u>28.7</u>
\captionscroatian	<u>37.6</u>
\captionsczech	... <u>38.6</u>
\captionsdanish	... <u>31.6</u>
\captionsdutch	... <u>16.22</u>
\captionsenglish	<u>17.25</u>
\captionsesperanto	<u>15.6</u>
\captionsestonian	<u>36.7</u>
\captionsestonian	<u>34.6</u>
\captionsfrenchb	<u>24.48</u>
\captionsgalician	<u>29.6</u>
\captionsgerman	... <u>18.12</u>
\captionsgreek	... <u>23.19</u>
\captionssirish	... <u>21.6</u>
\captionssitalian	... <u>25.6</u>
\captionsslang	... <u>7</u>
\captionsslsorbian	<u>43.6</u>
\captionssmagyar	... <u>35.10</u>
\captionssnorsk	... <u>32.9</u>
\captionssnynorsk	<u>32.34</u>
\captionsspolish	... <u>39.6</u>
\captionssportuges	<u>26.34</u>
\captionssromanian	<u>30.6</u>
\captionssrussian	<u>42.17</u>
\captionssscottish	<u>22.6</u>
\captionsslovak	... <u>40.6</u>
\captionsslovene	... <u>41.6</u>
\captionssspanish	... <u>27.7</u>
\captionssswedish	... <u>33.6</u>
\captionsssturkish	... <u>45.6</u>
\captionssusorbian	<u>44.6</u>
\captionsswelsh	... <u>20.7</u>
\ck	... <u>18.101</u> , <u>44.94</u>
\cyrillictext	... <u>42.9</u>
D	
\dateafrikaans	... <u>16.85</u>
\dateamerican	... <u>17.59</u>
\dateaustrian	... <u>18.39</u>
\datebahasa	... <u>46.33</u>
\datebrazil	... <u>26.86</u>
\datebreton	... <u>19.28</u>
\datecatalan	... <u>28.29</u>
\datecroatian	... <u>37.28</u>
\dateczech	... <u>38.28</u>
\datedanish	... <u>31.28</u>
\datedutch	... <u>16.48</u>
\dateenglish	... <u>17.47</u>
\dateesperanto	... <u>15.29</u>
\dateestonian	... <u>36.33</u>
\datefinnish	... <u>34.28</u>
\datefrenchb	... <u>24.81</u>
\dategalician	... <u>29.28</u>

N		O		<code>\spanishhyphenmins</code> 27.55
<code>\newcheck</code>	36.50	<code>\oECGuill</code>	24.168	<code>\spb</code> 39.62
<code>\newdatelsorbian</code>	43.28	<code>\og</code>	24.172	<code>\substitutefontfamily</code> 8
<code>\newdateusorbian</code>	44.28	<code>\oLasyGuill</code>	24.159	<code>\substitutefontfamily</code>
<code>\newlabel</code>	10.901	<code>\olddatelsorbian</code>	43.34 9.59
<code>\newtilde</code>	36.50	<code>\olddateusorbian</code>	44.34	<code>\swedishhyphenmins</code> 33.35
<code>\nfss@catcodes</code>	10.955	<code>\ondatemagyar</code>	35.46	<code>\system@group</code> 10.478
<code>\No</code>	24.324	<code>\ontoday</code>	138	<code>\system@sh@:</code>
<code>\no</code>	24.324	<code>\oPlainGuill</code>	24.153 24.149 , 42.128
<code>\NoAutoSpaceBeforeFDP</code>		<code>\ord</code>	26.111	<code>\system@sh@;</code>
.	24.143	<code>\orda</code>	26.111 24.149 , 42.128
<code>\nocite</code>	10.917	<code>\originalTeX</code>	10.189	<code>\system@sh@?@</code>
<code>\noextrasafrikaans</code> 16.92		<code>\OTtdqpos</code>	10.520 24.149 , 42.128
<code>\noextrasaustrian</code> 18.41		otherlanguage (envi-		<code>\system@sh@@</code>
<code>\noextrasbahasa</code> 46.38		ronment) 3, 10.91	 24.149 , 42.128
<code>\noextrasbreton</code> 19.34		otherlanguage* (envi-		
<code>\noextrascatalan</code> 28.35		ronment) 3, 10.100		
<code>\noextrascroatian</code> 37.33				T
<code>\noextrasczech</code> 38.34		P		<code>\Ttdqpos</code> 10.520
<code>\noextrasdanish</code> 31.33		<code>\pageref</code> 10.911		<code>\telepl</code> 39.86
<code>\noextrasdutch</code> 16.92		<code>\peek@token</code> 10.316		<code>\textacute</code>
<code>\noextrasenglish</code> 17.64		<code>\portugeshyphenmins</code>		27.56 , 28.64 , 29.52
<code>\noextrasesperanto</code> 15.34	 26.93		<code>\textcyrillic</code> 42.17
<code>\noextrasestonian</code> 36.42		<code>\primo</code>	24.324	<code>\textgrave</code> 28.64
<code>\noextrasfinnish</code> 34.34		<code>\process@language</code> 10.237		<code>\textgreek</code> 23.14
<code>\noextrasfrenchb</code> 24.28		<code>\process@line</code> 10.219		<code>\textlatin</code> 23.14 , 42.17
<code>\noextrasgalician</code> 29.33		<code>\process@synonym</code> 10.226		<code>\textol</code> 23.16
<code>\noextrasgerman</code> 18.41				<code>\textormath</code> 10.472
<code>\noextrasgreek</code> 23.47		Q		<code>\textpl</code> 39.77
<code>\noextrasirish</code> 21.35		<code>\quotedblbase</code> 10.571		<code>\texttilde</code> 27.56 , 29.52
<code>\noextrasitalian</code> 25.34		<code>\quotesinglbase</code> 10.576		<code>\thousandsep</code> 24.241
<code>\noextrasslang</code> 8		R		<code>\tildelow</code> 36.50
<code>\noextrassorbian</code> 43.49		<code>\ra</code>	26.111	<code>\turkish@sh@:</code> 45.43
<code>\noextrassmagyar</code> 35.60		<code>\readconfigfile</code> 10.257		<code>\turkish@sh@@</code> 45.43
<code>\noextrasspolish</code> 39.34		<code>\ref</code>	10.911	
<code>\noextrassportuges</code> 26.94		<code>\ro</code>	26.111	U
<code>\noextrassromanian</code> 30.34		<code>\russian@sh@:</code> 42.92		<code>\umlauthigh</code> 10.670
<code>\noextrasscottish</code> 22.35		<code>\russian@sh@;</code> 42.92		<code>\umlautlow</code> 10.670
<code>\noextrasslovak</code> 40.34		<code>\russian@sh@?@</code> 42.92		<code>\up</code> 24.292 , 28.184
<code>\noextrasslovene</code> 41.33		<code>\russian@sh@@</code> 42.92		<code>\up@size</code> 24.292
<code>\noextrasspanish</code> 27.34		S		<code>\user@group</code> 10.478
<code>\noextrasswedish</code> 33.36		<code>\save@sf@q</code> 9, 10.566		<code>\usesorthands</code> 3, 10.481
<code>\noextrassrussian</code> 45.35		<code>\selectlanguage</code> 3, 10.47		
<code>\noextrasswelsh</code> 20.37		<code>\set@hyphenmins</code> 10.133		V
<code>\nombre</code> 24.252		<code>\set@low@box</code> 9, 10.563		<code>\v</code> 38.38
<code>\norskhyphenmins</code> 32.6		<code>\skb</code>	39.70	<code>\verbatim@font</code> 42.74
		<code>\sob</code>	39.42	
				W
				<code>\welshhyphenmins</code> 20.6

Change History

babel 2.0a	
General: Added text about <code>german.sty</code>	1
babel 2.0b	
General: Changed order of code to prevent plain <code>T_EX</code> from seeing all of it	1
babel 2.1	
General: Modified user interface, <code>\langTeX</code> no longer necessary	1
babel 2.1a	
General: Incorporated Nico's comments	1
babel 2.1b	
General: rename <code>\language</code> to <code>\current@language</code>	1
babel 2.1c	
General: abstract for report fixed, missing <code>}</code> , found by Nicolas Brouard	1
babel 2.1d	
General: Missing right brace in definition of abstract environment, found by Werenfried Spit	1
babel 2.1e	
General: Incorporated more comments from Nico	1
babel 2.2	
General: Renamed <code>\newlanguage</code> to <code>\addlanguage</code>	1
babel 2.2a	
General: Modified the documentation somewhat	1
babel 3.0	
General: Moved part of the code to <code>hyphen.doc</code> in preparation for <code>T_EX 3.0</code>	1
babel 3.0a	
General: Updated comments in various places	1
<code>\iflanguage</code> : Added <code>\@bsphack</code> and <code>\@esphack</code>	15
<code>\selectlanguage</code> : Added <code>\@bsphack</code> and <code>\@esphack</code>	17
Replaced <code>\gdef</code> with <code>\def</code>	17
babel 3.0b	
General: Removed some problems in change log	1
babel 3.0c	
General: Renamed <code>babel.sty</code> and <code>latexhax.sty</code> to <code>.com</code>	1
<code>\iflanguage</code> : Added comment character after <code>#2</code>	15
<code>\selectlanguage</code> : Made <code>\selectlanguage</code> robust	16
babel 3.0d	
<code>\@nopatterns</code> : Added a percent sign to remove unwanted white space	21
General: Removed use of <code>\@ifundefined</code>	14
<code>\doc@style</code> : Removed use of <code>\@ifundefined</code>	40
<code>\iflanguage</code> : Removed space hacks and use of <code>\@ifundefined</code>	15
Removed superfluous <code>\expandafter</code>	15
<code>\process@language</code> : Added the collection of pattern names.	23
Reinserted <code>\expandafter</code>	23
Removed superfluous <code>\expandafter</code>	23
<code>\selectlanguage</code> : Removed space hacks and use of <code>\@ifundefined</code>	17
Removed superfluous <code>\expandafter</code>	17
babel 3.1	
General: Added the support for active characters and for extending a macro	1
Removed definition of <code>\if@restonecol</code>	40
Removed the need for <code>latexhax</code>	1
<code>\addto</code> : Added macro	34
<code>\readconfigfile</code> : Removed the extra <code>if</code> control sequence	24
Removed use of <code>\toks0</code>	24

\selectlanguage: \originalTeX should only be executed once	17
babel 3.2	
General: Some Changes by br	1
\adddialect: Added \relax	15
\addlanguage: Added a %, removed by	15
\babel@beginsave: Added macro	33
\babel@save: Added macro	33
\babel@savecnt: Added macro	33
\babel@savevariable: Added macro	33
\bb1@add@special: Added macro	25
\bb1@remove@special: Added macro	25
\iflanguage: Refrased \ifnum test	15
\selectlanguage: Modified to allow arguments that start with an escape character	16
babel 3.2a	
General: Fixups of the code and documentation	1
\originalTeX: Set \originalTeX to \empty, because it should be expandable.	21
\readconfigfile: Free macro space for \process@language	24
\selectlanguage: Added \relax as first command to stop an expansion if \protect is empty	17
Added three \expandafters to save macro space for \originalTeX	17
Moved definition of \originalTeX before \extras<lang>	17
Set \originalTeX to \empty, because it should be expandable.	17
Simplified the modification to allow the use in a \write command	16
babel 3.2b	
\allowhyphens: Moved macro from language definition files	34
\save@sf@q: Moved macro from language definition files	35
\set@low@box: Moved macro from language definition files	35
babel 3.2c	
\babel@save: missing backslash led to errors when executing \originalTeX ..	33
babel 3.2d	
\babel@save: saving in \babel@i and restoring from \@babel@i doesn't work very well...	33
babel 3.2e	
General: Added slovak	52
babel 3.3	
General: \headpagename should be \pagename	44
Added catalan and galician	52
Added turkish	52
Included driver file, and prepared for dsitribution	1
babel 3.4	
General: Added bahasa	52
Added language definition file for bahasa	1
Updated for L ^A T _E X 2 _ε	1
\addto: Changed to use toks register	34
babel 3.4b	
General: Added a small driver to be able to process just this file	1
Use the ltxdoc class instead of article	50
babel 3.4c	
General: lhyphen.cfg has become lthyphen.cfg	11
babel 3.4e	
\@nolanerr: Use \PackageError in L ^A T _E X 2 _ε mode	21
\@nopatterns: Macro added	21

General: Redid the identification code, provided dummy definition of <code>\ProvidesFile</code> for plain <code>TeX</code>	10
<code>\process@language</code> : Added code to detect assignments to left- and righthyphen- min in the patternfile.	23
babel 3.4g	
<code>\@testdef</code> : Moved the <code>\def</code> inside the macrocode environment	46
babel 3.5a	
<code>\@nopatterns</code> : Added <code>\@activated</code> to log active characters	21
General: Added a system shorthand for the right quote	32
Added breton, irish, scottish	52
Changed extension of language definition files to <code>ldf</code>	11
Provided common code to handle the active double quote	1
Replaced 16 system shorthands to deal with hex numbers by one	32
<code>\bbl@activate</code> : Added macro	30
<code>\bbl@deactivate</code> : Added macro	30
<code>\bbl@main@language</code> : Macro added	21
<code>\bbl@pr@ms</code> : Added macro	32
<code>\initiate@active@char</code> : Added a check for right quote and adapt <code>\pr@ms</code> if necessary	27
Added macro	26
<code>\main@language</code> : Macro added	21
<code>\selectlanguage</code> : write the language change to the auxiliary files	17
babel 3.5b	
General: Added brazilian as alternative for brazil	11
Added estonian option	11
Added lsorbian, usorbian	52
<code>lthyphen.cfg</code> has become <code>hyphen.cfg</code>	11
<code>\initiate@active@char</code> : Renamed macro	26
<code>\pageref</code> : Made <code>\ref</code> and <code>\pageref</code> robust (PR1353)	46
<code>\process@language</code> : Added optional reading of file with hyphenation exceptions	23
<code>\process@line</code> : added macro	22
<code>\process@synonym</code> : added macro	22
<code>\readconfigfile</code> : Now add a <code>\space</code> and a <code>/</code> character	24
<code>\selectlanguage</code> : Added an extra level of expansion to separate the switching mechanism from writing to aux files	16
Added default setting of <code>hyphenmin</code> parameters	17
Changed the name of the internal macro to <code>\selectlanguage</code>	16
Separated the setting of the <code>hyphenmin</code> values	17
Store the name of the current language in a control sequence instead of passing the whole macro construct to strip the escape character in the argument of <code>\selectlanguage</code>	16
babel 3.5c	
<code>\@nopatterns</code> : Added missing closing brace	21
General: Changed the order of including the language files somewhat (PR1652)	52
corrected a few typos (PR1652)	1
Repaired a typo (itlaic, PR1652)	35
babel 3.5d	
General: Added british as an alternative for ‘english’ with a preference for british hyphenation	11
Added options to influence behaviour of active acute and grave accents	12
Load <code>french.ldf</code> when it is found instead of <code>frenchb.ldf</code>	12
Load language definition files <i>after</i> the check for the hyphenation patterns	11
Merged <code>glyphs.dtx</code> into this file	1
<code>\active@prefix</code> : <code>\@protected@cmd</code> has vanished from <code>ltoutenc.dtx</code>	30

<code>\declare@shorthand</code> : Make a ‘note’ when a shorthand with an argument is defined.	31
<code>\foreignlanguage</code> : Macro added	18
<code>\initiate@active@char</code> : Skip the language-level active char with argument if no shorthands with arguments were defined	29
Skip the user-level active char with argument if no shorthands with arguments were defined	28
<code>\loadlocalcfg</code> : Added macro	48
<code>\pageref</code> : use a different control sequence while making <code>\ref</code> and <code>\pageref</code> robust	46
<code>otherlanguage</code> : environment added	18
babel 3.5e	
<code>otherlanguage</code> : changed name	18
babel 3.5f	
<code>\@bibitem</code> : Now use <code>\bbl@redefine</code>	46
<code>\@citex</code> : Now use <code>\bbl@redefine</code>	46
<code>\@lbibitem</code> : Now use <code>\bbl@redefine</code>	46
<code>\@testdef</code> : Complete rewrite of this macro as the same character ended up with different category codes in the labels that are being compared. Now use <code>\meaning</code>	46
Now use <code>\bbl@redefine</code>	46
Use <code>\strip@prefix</code> only on <code>\bbl@tempa</code> when it is not <code>\relax</code>	46
General: Added a system shorthand for the left quote	32
Added the <code>greek</code> option	12
Need to temporarily change the definition of <code>\ProvidesFile</code> for december 1995 release	10
Now use the file frenchb from Daniel Flipo for french support	11
repaired a typo	1
replaced <code>\tmp</code> , <code>\bbl@tmp</code> and <code>\bbl@temp</code> with <code>\bbl@tempa</code>	1
Store version in <code>\fileversion</code>	10
<code>\aliasshorthand</code> : New command	31
<code>\bbl@disc</code> : Macro moved from language definition files	35
<code>\bbl@redefine</code> : Macro added	45
<code>\bbl@redefineroast</code> : Define <code>*foo</code> instead of <code>\foo</code>	45
Macro added	45
<code>\bibcite</code> : Now use <code>\bbl@redefine</code>	46
<code>\bbl@test@token</code> : macro added	27
<code>\DJ</code> : New definition of <code>\dj</code> , see PR 2058	37
<code>\frq</code> : corrected spelling of <code>\quilsingl...</code>	38
now use <code>\textormath</code> in these definitions	38
<code>\frqq</code> : corrected spelling of <code>\quillemot...</code>	38
now use <code>\textormath</code> in these definitions	38
<code>\grq</code> : Added kerning to german right quote	37
now use <code>\textormath</code> in these definitions	37
<code>\grqq</code> : Added kerning to german right quote	38
now use <code>\textormath</code> in these definitions	38
<code>\initiate@active@char</code> : Deal correctly with already active characters, provide top level expansion and define all lower level expansion macro’s outside of the <code>\else</code> branch.	27
restore the <code>\lccode</code> of the tie	28
Restore the category code of a shorthand char at end of package	27
store the <code>\lccode</code> of the tie before changing it	28
use <code>\peek@token</code> to check wheter it is safe to proceed	28, 29
<code>\lower@umlaut</code> : Added a <code>\allowhyphens</code>	39

removed <code>\allowhyphens</code>	39
<code>\newlabel</code> : Now use <code>\bbl@redefine</code>	46
<code>\nocite</code> : Now use <code>\bbl@redefine</code>	46
<code>\pageref</code> : Now use <code>\bbl@redefinero bust</code>	46
redefine <code>*ref</code> if it exists instead of <code>\def</code>	46
redefine <code>\setref</code> instead of <code>\ref</code> and <code>\pageref</code> in L ^A T _E X 2 _ε	46
Reverse the previous change as it inhibits the use of active characters in labels	46
<code>\peek@token</code> : macro added	26
<code>\process@language</code> : Use <code>\empty</code> instead of <code>\@empty</code> as the latter is unknown in	
plain	23
<code>\readconfigfile</code> : Moved the fiddling with <code>\dump</code> to <code>bbplain.dtx</code> as it is no	
longer needed for L ^A T _E X	25
<code>\selectlanguage</code> : Added a missing percent character	16
Moved check for escape character one level down in the expansion	16
Now also define <code>\language name</code> at this level	16
<code>otherlanguage*</code> : environment added	18
babel 3.5g	
General: Added definition of <code>\Babel</code>	51
Added greek	52
Added option for ‘afrikaans’	11
Removed the use of <code>\patterns@loaded</code> altogether	22
replaced <code>\undefined</code> with <code>\@undefined</code> to be consistent with L ^A T _E X	1
Save a few csnames; use <code>\bbl@tempa</code> instead of <code>\ProvidesFile</code> and store mes-	
sage in <code>\toks8</code>	10
<code>\ifthenelse</code> : Redefinition of <code>\ifthenelse</code> added to circumvent problems with	
<code>\pageref</code> in the argument of <code>\isodd</code>	47
<code>\initiate@active@char</code> : Top level expansion of <code>\normal@char</code> <code>char</code> where <code>char</code>	
is already active, should be the expansion of the active character, not the	
active character itself as this causes an endless loop	27
<code>\nfss@catcodes</code> : Need to add the double quote and acute characters to <code>\nfss@catcodes</code>	
to prevent problems when reading in .fd files	48
<code>\process@line</code> : Simplified code, removing <code>\bbl@eq@</code>	22
babel 3.6a	
<code>\@@vpageref</code> : Redefinition of <code>\@@vpageref</code> added to circumvent problems with	
active : in the argument of <code>\vref</code> when <code>varioref</code> is used	47
General: Added welsh	52
Removed <code>\babel@core@loaded</code> , no longer needed whith the advent of <code>\LdfInit</code>	
.....	14
<code>\ldf@finish</code> : Macro added	20
<code>\ldf@quit</code> : Macro added	20
<code>\LdfInit</code> : Macro added	19
<code>\main@language</code> : <code>\main@language</code> now also sets <code>\language name</code> and <code>\l@language name</code>	
for use by other packages in the preamble of a document	21
<code>\selectlanguage</code> : Check for the existence of <code>\date...</code> instead of <code>\l@...</code>	17
babel 3.6b	
<code>\addto</code> : Also check if control sequence expands to <code>\relax</code>	34
babel 3.6c	
General: When <code>\LdfInit</code> is undefined we need to load <code>babel.def</code> from <code>babel.sty</code>	11
<code>\bbl@main@language</code> : When <code>hyphen.cfg</code> is not loaded in the format <code>\l@english</code>	
might not be defined; assume english is language 0	21
babel 3.6d	
<code>\foreign@language</code> : Added <code>\relax</code> to prevent disappearance of the first token	
after this command.	19
New macro	19

set the language shorthands to ‘none’ before switching on the extras	19
\foreignlanguage : Introduced \foreign@language	18
\selectlanguage : set the language shorthands to ‘none’ before switching on the extras	17
otherlanguage* : Introduced \foreign@language	18
babel 3.6e	
General: Added option ‘frenchb’ as an alias for ‘francais’	11
Added options ‘UKenglish’ and ‘USenglish’	12
babel 3.6f	
General: Added option KeepShorthandsActive	12
\bbl@redefine@long : Macro added	45
\ifthenelse : \ifthenelse needs to be long	47
\initiate@active@char : Made restoring of the category code of shorthand char- acters optional	27
babel 3.6h	
\readconfigfile : Added a couple of \expandafters to copy the contents of \toks8 into \everyjob instead of the reference	25
babel 3.6i	
\bbl@test@token : Renamed \test@token to \bbl@test@token to prevent clash with ArabTeX	27
\peek@token : Renamed \test@token to \bbl@test@token to prevent clash with ArabTeX	26
babel 3.6j	
General: Added the hebrew option	12
bahasa-0.9c	
General: Now use \@patterns to produce the warning	177
Removed the use of \filedate and moved identification after the loading of babel.def	177
bahasa-1.0b	
\captionsbahasa : Added \proofname for AMS-L ^A T _E X	177
bahasa-1.0d	
General: Replaced \undefined with \@undefined and \empty with \@empty for consistency with L ^A T _E X	177
\captionsbahasa : Replaced ‘Proof’ by ‘Bukti’ PR 2214	177
bahasa-1.0e	
General: Moved the definition of \atcatcode right to the beginning.	177
Now use \ldf@finish to wrap up	178
Now use \LdfInit to perform initial checks	177
\bahasahyphenmins : use \bahasahyphenmins to store the correct values	178
bbplain-0.1	
General: Added redefinition of \dump to add a message to \everyjob	179
bbplain-1.0c	
General: Add execution of \@beginindocumenthook to \@preamblecmds	180
Added definition of \loadlocalcfg	179
Moved the \dump code here from babel.dtx	179
bbplain-1.0d	
General: Also reset category codes after loading the configuration file as \AtEndOfPackage is undefined in this case	179
bbplain-1.0e	
General: Added the \newcommand code	182
Provide a more complete emulation of \DeclareRobustCommand and \newcommand	181
bbplain-1.0f	
General: added \@empty	179

Added <code>\textquotedblright</code> and <code>\textquoteright</code>	186
Added definition of <code>\scriptsize</code>	186
Consistently use <code>\@undefined</code> instead of <code>\undefined</code>	179
Use <code>\toks8</code> instead of <code>\patterns@loaded</code>	180
bbplain-1.0g	
General: Added <code>\ss</code> and <code>\i</code>	186
bbplain-1.0h	
General: Only load the necessary parts into the format, let this file be read again by <code>babel.def</code>	179
bbplain-1.0i	
General: <code>\document</code> is not a $\text{\LaTeX}2.09$ -only command; \AMSTeX defines it too; now use <code>\@ztryfc</code> to detect $\text{\LaTeX}2.09$	180
bbplain-1.0j	
General: <code>\@begindocumenthook</code> might already be defined	181
Add the definition of <code>\@begindocumenthook</code> to the $\text{\LaTeX}2.09$ format	180
bbplain-1.0k	
General: <code>\newcount</code> is an <code>\outer</code> command, can't use it inside an <code>\if</code> construct	182
missing <code>\@undefined</code> added	181
bbplain-1.0l	
General: Mixed up the definition of <code>\@tempcntb</code>	182
bbplain-v1.0m	
General: Set <code>\if@files</code> to <code>\iffalse</code> only for plain \TeX	181
breton-1.0	
General: First release	71
breton-1.0b	
<code>\captionubreton</code> : Added <code>\proofname</code> for \AMS-LaTeX	71
<code>\noextrasureton</code> : Use the new mechanism for dealing with active chars	72
breton-1.0c	
General: Postpone the <code>\DeclareTextCompositeCommands</code> until <code>\AtBeginDocument</code>	74
breton-1.0e	
General: Now use <code>\ldf@finish</code> to wrap up	74
Now use <code>\LdfInit</code> to perform initial checks	71
Replaced <code>\undefined</code> with <code>\@undefined</code> and <code>\empty</code> with <code>\@empty</code> for consistency with \LaTeX , moved the definition of <code>\atcatcode</code> right to the beginning.	71
catalan-1.1	
<code>\captionscatalan</code> : <code>\headpagename</code> should be <code>\pagename</code>	113
catalan-2.0	
General: Removed code to load <code>latexhax.com</code>	112
<code>\captionscatalan</code> : Added some names	113
<code>\extrascatalan</code> : Macro completely rewritten	114
<code>\noextrascatalan</code> : Macro completely rewritten	114
catalan-2.0b	
General: Incorporated the changes from <code>spanish.sty</code>	112
catalan-2.1	
General: Update for $\text{\LaTeX}2_{\epsilon}$	112
catalan-2.1d	
General: Now use <code>\@nopatterns</code> to produce the warning	113
Removed the use of <code>\filedate</code> and moved identification after the loading of <code>babel.def</code>	112
<code>\captionscatalan</code> : Added a few missing translations	113
<code>\textacute</code> : Renamed from <code>\acute</code> as that is a <code>\mathaccent</code>	114

catalan-2.2a	
General: All the code to deal with active characters is now in <code>babel.def</code>	116
<code>\extrascatalan</code> : Handling of active characters completely rewritten	114
<code>\noextrascatalan</code> : All the code for handling active characters is now moved to <code>babel.def</code>	114
catalan-2.2b	
General: Changed mathmode definition of acute shorthands to expand to a single prime followed by the next character in the input	116
Made the activation of the grave and acute accents optional	112
<code>\captionscatalan</code> : Added <code>\proofname</code> for AMS- \LaTeX	113
<code>\datecatalan</code> : Month names in lowercase	113
<code>\Lgem</code> : Added support for typing the catalan “ <i>ela geminada</i> ” with the macros <code>\lgem</code> and <code>\Lgem</code>	117
<code>\noextrascatalan</code> : Make activating the accent characters optional	114
<code>\up</code> : Added definition of macro <code>\up</code> , which can be used to type ordinals	118
catalan-2.2c	
General: Added “” as an axtra shorthand, removed ‘n as a shorthand	116
Added shorthands for guillemets	116
cedile now produced by double quote shorthand	116
Removed the use of the tilde for catalan	112
catalan-2.2d	
<code>\captionscatalan</code> : added translation of Proof	113
Translations revised	113
catalan-2.2e	
General: Added “ as an axtra shorthand	116
Added vertical bar as argument to active acute	116
<code>\L.L</code> : Added redefinition of <code>\l</code> and <code>\L</code>	118
<code>\noextrascatalan</code> : Need to split up the <code>\@ifpackagewith</code> statements	114
Now give the apostrophe a lowercase code	114
<code>\up</code> : Now use <code>\textsuperscript</code> and make <code>\up</code> robust	118
catalan-2.2f	
General: Replaced <code>\undefined</code> with <code>\@undefined</code> and <code>\empty</code> with <code>\@empty</code> for consistency with \LaTeX	112
<code>\Lgem</code> : Added a check for math mode as the use of <code>\lgem</code> and <code>\Lgem</code> in math mode is not sensible.	117
catalan-2.2g	
General: Moved the definition of <code>\atcatcode</code> right to the beginning.	112
Now use <code>\ldf@finish</code> to wrap up	118
Now use <code>\LdfInit</code> to perform initial checks	113
catalan-2.2h	
<code>\noextrascatalan</code> : Added some comment signs to prevent unwanted spaces in the output	114
croatian-1.0a	
General: Renamed <code>babel.sty</code> in <code>babel.com</code>	146
croatian-1.0b	
General: Removed use of <code>\@ifundefined</code>	146
croatian-1.0c	
General: Removed some typos	146
croatian-1.1	
General: Added a warning when no hyphenation patterns were loaded.	146
Brought up-to-date with babel 3.2a	146
<code>\captionscroatian</code> : Added <code>\seename</code> , <code>\alsoname</code> and <code>\prefacename</code>	146
croatian-1.2	
<code>\captionscroatian</code> : <code>\headpagename</code> should be <code>\pagename</code>	146

croatian-1.3	
General: Update for L ^A T _E X 2 _ε	146
croatian-1.3d	
\captionscroatian: Added a few translations	146
croatian-1.3e	
\captionscroatian: Added \proofname for AMS-L ^A T _E X	146
croatian-1.3f	
\captionscroatian: Added translation of Proof	146
\datecroatian: in croatian language, the genitive for the name of the month has to be used	146
croatian-1.3g	
General: Now use \ldf@finish to wrap up	147
Now use \LdfInit to perform initial checks	146
Replaced \undefined with \@undefined and \empty with \@empty for consis- tency with L ^A T _E X, moved the definition of \atcatcode right to the beginning.	146
czech-1.0a	
General: Renamed babel.sty in babel.com	148
czech-1.0b	
General: Removed use of \@ifundefined	148
czech-1.1	
General: Added a warning when no hyphenation patterns were loaded.	148
Brought up-to-date with babel 3.2a	148
\captionsczech: Added \seename, \alsoname and \prefacename	148
czech-1.1a	
\noextrasczech: Removed typo, \q was restored twice, once too many.	149
czech-1.2	
General: Included some features from Kasal's czech.sty	148
czech-1.3	
General: Update for L ^A T _E X 2 _ε	148
czech-1.3d	
General: Now use \@nopatterns to produce the warning	148
Removed the use of \filedate and moved identification after the loading of babel.def	148
czech-1.3e	
\noextrasczech: now use \bbl@frenchspacing and \bbl@nonfrenchspacing	149
Use L ^A T _E X's \v and \r accent commands	149
czech-1.3f	
\captionsczech: Added \proofname for AMS-L ^A T _E X	148
czech-1.3g	
\captionsczech: Fixed two errors and provided translation for 'proof'	148
czech-1.3h	
General: Now use \ldf@finish to wrap up	149
Now use \LdfInit to perform initial checks	148
Replaced \undefined with \@undefined and \empty with \@empty for consis- tency with L ^A T _E X, moved the definition of \atcatcode right to the beginning.	148
danish-1.0a	
General: Renamed babel.sty in babel.com	125
danish-1.0b	
General: Removed use of \@ifundefined	125
danish-1.1	
General: Added a warning when no hyphenation patterns were loaded.	125
Brought up-to-date with babel 3.2a	125
\captionsdanish: Added \seename, \alsoname and \prefacename	125

danish-1.2	
\captionsdanish: \headpagename should be \pagename	125
danish-1.2b	
\captionsdanish: Added a few translations	125
danish-1.3	
General: Update for L ^A T _E X 2 _ε	125
danish-1.3a	
\datedanish: Added ‘.’ to definition of \today	126
danish-1.3c	
\captionsdanish: Included some revisions from Peter Busk Larsen	125
danish-1.3f	
General: Now use \nopatterns to produce the warning	125
Removed the use of \filedate and moved identification after the loading of	
babel.def	125
danish-1.3g	
General: Added the active double quote character as suggested by Peter Busk	
Laursen	125
danish-1.3h	
\captionsdanish: Added \proofname for AMS-L ^A T _E X	125
\extrasdanish: Added \bbl@frenchspacing	126
\noextrasdanish: Added \bbl@nonfrenchspacing	126
danish-1.3i	
\captionsdanish: Added translation of ‘Proof’	125
danish-1.3j	
General: Now use \ldf@finish to wrap up	127
Replaced \undefined with \@undefined and \empty with \@empty for consis-	
tency with L ^A T _E X, moved the definition of \atcatcode right to the beginning.	125
\extrasdanish: Added definition of "~ and "=	126
Changed definition of " to print “ instead of ’	126
danish1.3j	
General: Now use \LdfInit to perform initial checks	125
dutch-2.0a	
General: Added checking of format	58
dutch-2.0b	
General: Added extrasdutch	58
dutch-2.0c	
General: Added grqq macros	58
dutch-2.1	
General: reflect change to version 2.1 in babel and changes in german v2.3	58
dutch-2.1a	
General: Incorporated Nico’s comments	58
dutch-2.1b	
General: Incorporated more comments by Nico	58
dutch-2.1c	
General: Fixed some typos	58
dutch-2.2	
General: Fixed problem with the use of " in moving arguments while " is active	58
dutch-2.3	
\@trema: \dieresis instead of \accent127	61
General: \dieresis instead of \accent127	61
When using PostScript fonts with the Adobe font-encoding, the dieresis-accent	
is located elsewhere, modified code	58
\noextrasafrikaans: Added \dieresis	60

dutch-2.3a	
General: Modified the documentation somewhat	58
dutch-3.0	
General: Modified for babel 3.0	58
Now use <code>\adddialect</code> if language undefined	59
dutch-3.0a	
General: Removed some problems in change log	58
dutch-3.0b	
<code>\extrasafrikaans</code> : added some comment chars to prevent white space	60
<code>\noextrasafrikaans</code> : added some comment chars to prevent white space	60
dutch-3.1	
General: Removed bug found by van der Meer	58
dutch-3.1a	
<code>\captionsdutch</code> : <code>\pagename</code> should be <code>\headpagename</code>	59
Removed <code>\global</code> definitions	59
<code>\datedutch</code> : Removed <code>\global</code> definitions	59
<code>\extrasafrikaans</code> : Removed <code>\global</code> definitions	60
<code>\noextrasafrikaans</code> : Removed <code>\global</code> definitions	60
dutch-3.2	
General: added case for "y and "Y	61
<code>\extrasafrikaans</code> : Save all redefined macros	60
<code>\noextrasafrikaans</code> : Try to restore everything to its former state	60
dutch-3.2a	
General: Added reset of catcode of @ before <code>\endinput</code> .	58
Renamed <code>babel.sty</code> in <code>babel.com</code>	58
dutch-3.2b	
General: removed typo (allowhpyhens)	61
dutch-3.2c	
General: Removed code to load <code>latexhax.com</code>	58
removed use of <code>\@ifundefined</code>	58, 59
dutch-3.3	
General: Rewritten parts of the code to use the new features of babel version 3.1	58
<code>\extrasafrikaans</code> : Macro complete rewritten	60
<code>\noextrasafrikaans</code> : Macro complete rewritten	60
dutch-3.3a	
<code>\@trema</code> : renamed <code>\@umlaut</code> to <code>\@trema</code>	61
General: Added <code>\save@sf@q</code> macro from <code>germanb</code> and rewrote all quote macros to use it	61
Moved code to the beginning of the file and added <code>\selectlanguage</code> call	58
<code>\captionsdutch</code> : added <code>\seename</code> and <code>\alsoname</code>	59
dutch-3.3b	
General: Added warning, if no dutch patterns loaded	59
<code>\captionsdutch</code> : added <code>\prefacename</code>	59
<code>\extrasafrikaans</code> : modified handling of <code>\dospecials</code> and <code>\@sanitize</code>	60
<code>\noextrasafrikaans</code> : modified handling of <code>\dospecials</code> and <code>\@sanitize</code>	60
dutch-3.4b	
General: moved definition of <code>\allowhyphens</code> , <code>\set@low@box</code> and <code>\save@sf@q</code> to <code>babel.com</code>	61
dutch-3.5	
<code>\captionsdutch</code> : <code>\headpagename</code> should be <code>\pagename</code>	59
dutch-3.6	
General: Update or LaTeX2e	58
dutch-3.6c	
General: Now use <code>\@nopatterns</code> to produce the warning	59

Removed the use of <code>\filedate</code> , moved identification after the loading of <code>babel.def</code>	58
dutch-3.7a	
General: Moved identification code to the top of the file	58
Moved the definition of <code>\ij</code> and <code>\IJ</code> to <code>glyphs.def</code>	61
moved the definition of the double quote character at the baseline to <code>glyphs.def</code>	61
Now use <code>\Declaredq dutch</code> to define the functions of the active double quote ..	61
Removed <code>\dlqq</code> , <code>\@dlqq</code> , <code>\drqq</code> , <code>\@drqq</code> and <code>\dieresis</code>	61
Rewrote the code with respect to the active double quote character	58
The support macros for the active double quote have been moved to <code>babel.def</code> ..	61
Use <code>\ddot</code> instead of <code>\@MATHUMLAUT</code>	61
Use more general mechanism of <code>\declare@shorthand</code>	61
<code>\afrikaanshyphenmins</code> : use <code>\dutchhyphenmins</code> to store the correct values ...	61
<code>\IJ</code> : Changed the kerning in the faked <code>ij</code> to match the dc-version of it	36
dutch-3.7b	
General: Added <code>"</code> shorthand	61
dutch-3.7c	
<code>\captionsdutch</code> : We need the <code>"</code> to be active while defining <code>\captionsdutch</code> ..	59
dutch-3.7d	
<code>\captionsdutch</code> : Added <code>\proofname</code> for AMS- \LaTeX	59
dutch-3.7f	
General: Replaced <code>\undefined</code> with <code>\@undefined</code> and <code>\empty</code> with <code>\@empty</code> for consistency with \LaTeX	58
dutch-3.8a	
General: Merged in the definitions for ‘afrikaans’	58
Now use <code>\ldf@finish</code> to wrap up	62
Now use <code>\LdfInit</code> to perform initial checks	141
this needs a more complicated check as ‘afrikaans’ may or may not have its own hyphenation patterns	59
<code>\noextrasafrikaans</code> : Made all definitions dependant on <code>\CurrentOption</code>	60
dutch-3.8b	
<code>\captionsdutch</code> : Use <code>Bew"ys</code> instead of <code>Bewijs</code>	59
dutch-3.8c	
General: Added the <code>"~</code> shorthand	61
dutch 3.8a	
General: Now use <code>\ldf@finish</code> to wrap up	140
english-2.0a	
General: Added checking of format	63
english-2.1	
General: Reflect changes in babel 2.1	63
english-2.1a	
General: Incorporated Nico’s comments	63
english-2.1b	
General: merged <code>USenglish.sty</code> into this file	63
english-2.1c	
General: fixed typo in definition for american language found by Werenfried Spit (<code>nspit@fys.ruu.nl</code>)	63
english-2.1d	
General: Fixed some typos	63
english-3.0	
General: Modified for babel 3.0	63
Now use <code>\adddialect</code> for american	63

Now use <code>\adddialect</code> if language undefined	63
english-3.0a	
General: Removed bug found by van der Meer	63
english-3.0b	
General: Removed <code>\global</code> definitions	63
<code>\captionseenglish: \pagename</code> should be <code>\headpagename</code>	64
Removed <code>\global</code> definitions	64
<code>\dateamerican: \global</code> definitions	64
<code>\dateenglish: \global</code> definitions	64
english-3.0c	
General: Renamed <code>babel.sty</code> in <code>babel.com</code>	63
english-3.0d	
General: removed use of <code>\@ifundefined</code>	63
english-3.1	
General: Rewrote parts of the code to use the new features of babel version 3.1	63
english-3.1a	
<code>\captionseenglish: added \seename and \alsoname</code>	64
english-3.1b	
<code>\captionseenglish: added \prefacename</code>	64
english-3.2	
<code>\captionseenglish: \headpagename</code> should be <code>\pagename</code>	64
english-3.3	
General: Update or $\text{\LaTeX} 2_{\epsilon}$	63
english-3.3c	
General: Now use <code>\@nopatterns</code> to produce the warning	63
Removed the use of <code>\filedate</code> and moved the identification after the loading	
of <code>babel.def</code>	63
english-3.3d	
General: Only define american as a dialect when no separate patterns have been	
loaded	63
english-3.3e	
<code>\captionseenglish: Added \proofname</code> for AMS- \LaTeX	64
english-3.3g	
General: Allow british as the name of the UK patterns	63
Allow USenglish as the name of the american patterns	63
Replaced <code>\undefined</code> with <code>\@undefined</code> and <code>\empty</code> with <code>\@empty</code> for consistency with \LaTeX	63
<code>\captionseenglish: Construct control sequence on the fly</code>	64
<code>\dateenglish: Construct control sequence on the fly</code>	64
<code>\noextrasenglish: Construct control sequences on the fly</code>	64
english-3.3h	
General: Moved the definition of <code>\atcatcode</code> right to the beginning.	63
Now use <code>\ldf@finish</code> to wrap up	65
Now use <code>\LdfInit</code> to perform initial checks	63
esperant-1.4j	
General: fixed typo in table caption (funtion instead of function)	55
esperanto-1.	
<code>\dateesperanto: Removed the capitals from \today</code>	56
esperanto-1.0a	
General: Renamed <code>babel.sty</code> in <code>babel.com</code>	55
esperanto-1.0b	
General: Removed use of <code>\makeatletter</code>	55
esperanto-1.1	
General: Added a warning when no hyphenation patterns were loaded.	55

Brought up-to-date with babel 3.2a	55
\captionesperanto: Added \seename, \alsoname and \prefacename	55
esperanto-1.2	
General: Included code from <code>esperant.sty</code>	55
esperanto-1.3	
\captionesperanto: \headpagename should be \pagename	55
Repaired a number of mistakes, indicated by D. Ederveen	55
esperanto-1.4a	
General: Updated for L ^A T _E X 2 _ε	55
\captionesperanto: added missing closing brace	55
esperanto-1.4d	
General: Removed the use of \filedate, moved Identification after loading of	
<code>babel.def</code>	55
Use \nopatterns for the warning	55
esperanto-1.4e	
General: Moved identification code to the top of the file	55
esperanto-1.4f	
General: Corrected typos (PR1652)	55
esperanto-1.4g	
\captionesperanto: Added \proofname for AMS-L ^A T _E X	55
esperanto-1.4h	
General: Added a few shorthands	56
esperanto-1.4i	
General: Moved the definition of \atcatcode right to the beginning.	55
Now use \ldf@finish to wrap up	57
Now use \LdfInit to perform initial checks	55
Replaced \undefined with \@undefined and \empty with \@empty for consis-	
tency with L ^A T _E X	55
\captionesperanto: Replaced ‘Proof’ by ‘Pruvo’ PR 2207	55
estonian-1.0b	
General: corrected typos	141
estonian-1.0c	
\captionsestonian: Added \proofname for AMS-L ^A T _E X	142
estonian-1.0d	
General: The second argument was missing in the definition of some of the double-	
quote shorthands	144
\captionsestonian: Added translation of ‘Proof’	142
\noextrasestonian: Removed the code that changes category, lower case, uper	
case and space factor codes	143
estonian-1.0e	
General: Now use \ldf@finish to wrap up	144
Replaced \undefined with \@undefined and \empty with \@empty for consis-	
tency with L ^A T _E X, moved the definition of \atcatcode right to the beginning.	141
finnish-1.0a	
General: Renamed <code>babel.sty</code> in <code>babel.com</code>	135
finnish-1.0b	
General: Removed use of \@ifundefined	135
finnish-1.1	
General: Added a warning when no hyphenation patterns were loaded.	135
Brought up-to-date with babel 3.2a	135
\captionsfinnish: \headpagename should be \pagename	135
Added \seename, \alsoname and \prefacename	135

finnish-1.1.2	
\captionsfinnish: Added translations	135
finnish-1.2	
General: Update for L ^A T _E X 2 _ε	135
finnish-1.3c	
General: Now use \nopatterns to produce the warning	135
Removed the use of \filedate and moved identification after the loading of	
babel.def	135
finnish-1.3d	
General: Removed a few references to babel.com	135
finnish-1.3e	
\datefinnish: Added a ‘.’ after the number of the day	136
finnish-1.3f	
\finishhyphenmins: use \finnishhyphenmins to store the correct values	137
\noextrasfinnish: Added the setting of \frenchspacing	136
Added the setting of more hyphenation parameters, according to PR1027	136
finnish-1.3g	
\:-: Added change of \-	137
\captionsfinnish: Added \proofname for AMS-L ^A T _E X	135
\noextrasfinnish: Added the active double quote	136
finnish-1.3h	
\captionsfinnish: Added finnish word for ‘Proof’	135
finnish-1.3i	
General: Now use \ldf@finish to wrap up	137
Now use \LdfInit to perform initial checks	135
Replaced \undefined with \@undefined and \empty with \@empty for consistency with L ^A T _E X, moved the definition of \atcatcode right to the beginning.	135
frenchb-1.1	
General: Add some font changing definitions.	89
Added \AllTeX.	98
Added \leavevmode in \bsc	98
Added a hook to insert space or not before ‘double punctuation’.	85, 91
Added command \bsc.	98
Added T1-encodings for œ, Œ, æ, Æ. <i>Do not</i> re-define these symbols outside L ^A T _E X 2 _ε .	98
Corrected definitions of \boi.	98
Do not redefine \^ and \" in M ^I T _E X, because it would break hyphenation. The correct place to redefine \^i and \"i is in the format itself, see MLTeX.cfg.	99
Do not use commands related to encodings outside L ^A T _E X 2 _ε .	98
Do this only in L ^A T _E X 2 _ε .	99
Special care has to be taken with M ^I T _E X.	99
Use \fmtname to check the format instead of \newcommand; define \PlainFmtName and \LaTeXeFmtName.	88
\bbl@nonfrenchguillemets: A warning is now issued if \og or \fg have been defined elsewhere in a L ^A T _E X 2 _ε document (suggested by Vincent Jalby).	93
\bbl@nonfrenchlistspacing: Save original definitions of label items, instead of hard coding them in \bbl@nonfrenchitems (suggested by Vincent Jalby).	94
Tune vertical spacing in French lists.	94
\captionsfrenchb: This code is useless in PlainT _E X, check the format before loading it.	89
\degres: Added \leavevmode in the \degres’s definition	99
Fixed width bounding box for correct spacing with both CMR/DC and PostScript fonts	99

\feccguill: \char 19 and \char 20 changed to \guillemotleft and \guillemotright as suggested by V. Jalby.	92
\flasyguill: Guillemets changed in L ^A T _E X-2.09 because of protection problems in moving arguments, now use the same guillemets as in plainT _E X.	92
\fprimo): Avoid using math superscripts in text mode (suggested by V. Jalby), use \up instead. The symbol ‘degree’ has nothing to do in \FrenchPopularEnumerate, replace it by a small ‘o’.	98
\frenchb@sh@;@: Added a hook to insert space or not before ‘double punctuation’.	91
\ieme: \Osize may not be undefined, i. e. in slides.cls.	97
Added 5 macros from french.sty and missing \lowercase.	97
Internal macro \upsize introduced by Johannes Braams to replace \small, too fragile in 2.09).	97
Use \textsuperscript in L ^A T _E X 2 _ε , as suggested by Vincent Jalby.	97
frenchb-1.1b	
General: Moved the definition of \atcatcode right to the beginning.	85
Now use \ldf@finish to wrap up	99
Now use \LdfInit to perform initial checks	87
Removed test for \l@english	88
Replaced \undefined with \@undefined and \empty with \@empty for consistency with L ^A T _E X	85
frenchb-1.2	
General: ‘french’ ‘frenchb’ and ‘français’ are synonymous regardless of \CurrentOption.	88
Check for hyphenation patterns	88
Removed \AllTeX.	98
The config file searched for is ‘frenchb.cfg’ regardless \CurrentOption.	99
\bbl@nonfrenchindent: Corrected typo \bbl@nonfrenchident.	94
\captionfrenchb: added aliases \captionfrench and \captionfrançais.	89
\datefrenchb: added aliases \datefrench and \datefrançais. Use \ier instead of \up.	90
\noextrsfrenchb: ‘french’ ‘frenchb’ and ‘français’ are synonymous regardless of \CurrentOption.	88
frenchb 1.1	
\if@Two@E: New test \if@Two@E.	88
frenchb 1.2	
General: New macros \nombre \decimalsep and \thousandsep added to format numbers.	95
galician-1.1	
General: Update for L ^A T _E X 2 _ε	119
galician-1.1c	
General: Now use \@nopatterns to produce the warning	119
Removed the use of \filedate and moved identification after the loading of babel.def	119
\textacute: Renamed from \acute as that is a \mathaccent	121
\texttilde: Renamed from \tilde as that is a \mathaccent	121
galician-1.1d	
\captionsgalician: Added a few missing translations	119
galician-1.2a	
\extragalician: Handling of active characters completely rewritten	120
\noextragalician: All the code for handling active characters is now moved to babel.def	120

galician-1.2b	
General: Changed mathmode definition of acute shorthands to expand to a single prime followed by the next character in the input	122
\captionsgalician: Added \proofname for AMS-L ^A T _E X	119
galician-1.2c	
\noextrasgalician: Make active accent optional	120
galician-1.2d	
General: Added " as an axtra shorthand	122
\noextrasgalician: Need to split up the \@ifpackagewith statement	120
galician-1.2e	
General: Now use \ldf@finish to wrap up	122
Now use \LdfInit to perform initial checks	119
Replaced \undefined with \@undefined and \empty with \@empty for consis- tency with L ^A T _E X, moved the definition of \atcatcode right to the beginning.	119
galician-1.2f	
General: Added \leavevmode to definitions of "a and "o	122
\noextrasgalician: Added some comment signs to prevent unwanted spaces in the output	120
galician1.1d	
\dategalician: Corrected the name of the month marzo from marzal	120
german-2.6a	
General: Moved all quotation characters to glyphs.dtx	68
Use \ddot instead of \@MATHUMLAUT	68
german-2.6b	
\captionsaustrian: Added \proofname for AMS-L ^A T _E X	67
german-2.6c	
General: added the \allowhyphens	68
german-2.6d	
General: Replaced \undefined with \@undefined and \empty with \@empty for consistency with L ^A T _E X	66
\captionsaustrian: Construct control sequence on the fly	67
\noextrasaustrian: Construct control sequence \extrasgerman or \extrasaustrian on the fly	68
germanb-1.0a	
General: Incorporated Nico's comments	66
germanb-1.0b	
General: fixed typo in definition for austrian language found by Werenfried Spit nspit@fys.ruu.nl	66
germanb-1.0c	
General: Fixed some typos	66
germanb-1.1	
General: When using PostScript fonts with the Adobe fontencding, the dieresis- accent is located elsewhere, modified code	66
\noextrasaustrian: Added \dieresis	68
germanb-1.1a	
General: Modified the documentation somewhat	66
germanb-2.0	
General: Modified for babel 3.0	66
Now use \adddialect for austrian	67
Now use \adddialect if language undefined	67
germanb-2.0a	
General: Removed some problems in change log	66
germanb-2.0b	
\extrasaustrian: added some comment chars to prevent white space	68

<code>\noextrasaustrian</code> : added some comment chars to prevent white space	68
germanb-2.1	
General: Removed bug found by van der Meer	66
germanb-2.2	
General: Removed global assignments, brought uptodate with <code>german.tex</code> v2.3d	66
<code>\captionsaustrian</code> : <code>\pagename</code> should be <code>\headpagename</code>	67
Removed <code>\global</code> definitions	67
<code>\extrasaustrian</code> : Save all redefined macros	68
<code>\noextrasaustrian</code> : Try to restore everything to its former state	68
germanb-2.2a	
General: Renamed <code>babel.sty</code> in <code>babel.com</code>	66
germanb-2.2d	
General: Removed use of <code>\@ifundefined</code>	67
germanb-2.3	
General: Rewritten parts of the code to use the new features of babel version 3.1	66
germanb-2.3e	
General: Added <code>\save@sf@q</code> macro and rewrote all quote macros to use it	68
Added warning, if no german patterns loaded	67
Brought up-to-date with <code>german.tex</code> v2.3e (plus some bug fixes) [br]	66
<code>\captionsaustrian</code> : Added <code>\prefacename</code> , <code>\seename</code> and <code>\alsoname</code>	67
<code>\dategerman</code> : Added <code>\month@german</code>	67
germanb-2.3h	
General: moved definition of <code>\allowhyphens</code> , <code>\set@low@box</code> and <code>\save@sf@q</code> to	
<code>babel.com</code>	68
germanb-2.4	
<code>\captionsaustrian</code> : <code>\headpagename</code> should be <code>\pagename</code>	67
germanb-2.5	
General: Update or \LaTeX 2\epsilon	66
germanb-2.5c	
General: Now use <code>\@nopatterns</code> to produce the warning	67
Removed the use of <code>\filedate</code> and moved the identification after the loading	
of <code>babel.def</code>	66
germanb-2.6a	
General: <code>\umlautlow</code> and <code>\umlauthigh</code> moved to <code>glyphs.dtx</code> , as well as <code>\newumlaut</code>	
(now <code>\lower@umlaut</code>	68
Moved the identification to the top of the file	66
Rewrote the code that handles the active double quote character	66
<code>\noextrasaustrian</code> : All the code to handle the active double quote has been	
moved to <code>babel.def</code>	68
Removed <code>\3</code> as it is no longer in <code>german.ldf</code>	68
use <code>\germanhyphenmins</code> to store the correct values	68
germanb-2.6c	
General: Moved <code>\german@dq@disc</code> to <code>babel.def</code> , calling it <code>\bbl@disc</code>	68
<code>\noextrasaustrian</code> : Use decimal number instead of hat-notation as the hat may	
be activated	68
germanb-2.6d	
General: Moved the definition of <code>\atcatcode</code> right to the beginning.	66
Now use <code>\ldf@finish</code> to wrap up	70
Now use <code>\LdfInit</code> to perform initial checks	66
greek-1.0b	
General: Moved the definition of <code>\atcatcode</code> right to the beginning.	81
Now use <code>\ldf@finish</code> to wrap up	84
Now use <code>\LdfInit</code> to perform initial checks	81

Replaced <code>\undefined</code> with <code>\@undefined</code> and <code>\empty</code> with <code>\@empty</code> for consistency with \LaTeX	81
<code>\textlatin</code> : Added a level of braces to keep encoding change local	81
irish-1.0b	
General: Corrected typo (PR1652)	77
irish-1.0c	
<code>\captionsirish</code> : Added <code>\proofname</code> for AMS- \LaTeX	77
irish-1.0e	
General: Now use <code>\ldf@finish</code> to wrap up	78
Now use <code>\LdfInit</code> to perform initial checks	77
Replaced <code>\undefined</code> with <code>\@undefined</code> and <code>\empty</code> with <code>\@empty</code> for consistency with \LaTeX , moved the definition of <code>\atcatcode</code> right to the beginning.	77
irish-1.0f	
<code>\dateirish</code> : Added missing <code>\def\today</code>	77
italian-0.99	
General: First version, from english.doc	100
italian-1.0	
General: Modified for babel 3.0	100
Now use <code>\adddialect</code> if language undefined	100
italian-1.0a	
General: removed typo	100
italian-1.0b	
General: Removed bug found by van der Meer	100
italian-1.0c	
<code>\captionsitalian</code> : <code>\pagename</code> should be <code>\headpagename</code>	100
Removed <code>\global</code> definitions	100
<code>\dateitalian</code> : Removed <code>\global</code> definitions	100
italian-1.0d	
<code>\captionsitalian</code> : ‘contine’ substituted by ‘Allegati’ as suggested by Marco Bozzo (BOZZO@CERNVM).	100
italian-1.0e	
General: Renamed <code>babel.sty</code> in <code>babel.com</code>	100
italian-1.0h	
General: Removed use of <code>\@ifundefined</code>	100
italian-1.1	
General: Added a warning when no hyphenation patterns were loaded.	100
Brought up-to-date with babel 3.2a	100
<code>\captionsitalian</code> : <code>\headpagename</code> should be <code>\pagename</code>	100
Added <code>\seename</code> , <code>\alsoname</code> and <code>\prefacename</code>	100
italian-1.2	
General: Update for LaTeX	100
italian-1.2b	
<code>\captionsitalian</code> : Changed some of the words following suggestions from Claudio Beccari	100
<code>\italianhyphenmins</code> : Added setting of left and righthyphenmin according to Claudio Beccari’s suggestion	101
<code>\noextrasitalian</code> : Added setting of club- and widowpenalty	101
Added setting of finallyhyphenemerits	101
italian-1.2e	
General: Now use <code>\@nopatterns</code> to produce the warning	100
Removed the use of <code>\filedate</code> and moved identification after the loading of <code>babel.def</code>	100

italian-1.2f	
General: Updated for babel 3.5	100
italian-1.2g	
<code>\captionsitalian</code> : Added <code>\proofname</code> for AMS- \LaTeX	100
italian-1.2h	
<code>\captionsitalian</code> : Added translation of ‘Proof’	100
<code>\noextrasitalian</code> : Now give the apostrophe a lowercase code	101
italian-1.2i	
General: Now use <code>\ldf@finish</code> to wrap up	101
Now use <code>\LdfInit</code> to perform initial checks	100
Replaced <code>\undefined</code> with <code>\@undefined</code> and <code>\empty</code> with <code>\@empty</code> for consistency with \LaTeX , moved the definition of <code>\atcatcode</code> right to the beginning.	100
italian-1.2j	
General: Added braces around second arg of <code>\LdfInit</code>	100
lsorbian-0.1	
General: First version	168
lsorbian-1.0b	
<code>\captionlsorbian</code> : Added <code>\proofname</code> for AMS- \LaTeX	168
lsorbian-1.0d	
General: Now use <code>\ldf@finish</code> to wrap up	169
Now use <code>\LdfInit</code> to perform initial checks	168
Replaced <code>\undefined</code> with <code>\@undefined</code> and <code>\empty</code> with <code>\@empty</code> for consistency with \LaTeX , moved the definition of <code>\atcatcode</code> right to the beginning.	168
maguar-1.3h	
General: Now use <code>\LdfInit</code> to perform initial checks	138
magyar-1.0a	
General: Renamed <code>babel.sty</code> in <code>babel.com</code>	138
magyar-1.0b	
General: Removed use of <code>\@ifundefined</code>	138
magyar-1.1	
General: Added a warning when no hyphenation patterns were loaded.	138
Brought up-to-date with babel 3.2a	138
<code>\captionsmagyar</code> : <code>\headpagename</code> should be <code>\pagename</code>	138
Added <code>\seename</code> , <code>\alsoname</code> and <code>\prefacename</code>	138
magyar-1.1.3	
<code>\captionsmagyar</code> : Added translations, fixed typos	138
<code>\ondatemagyar</code> : The date number should not be followed by a dot.	139
magyar-1.1.4	
General: Further spelling corrections	138
<code>\datemagyar</code> : Rewritten to produce the correct date format	139
<code>\ondatemagyar</code> : Renamed from <code>\datemagyar</code> ; no longer redefines <code>\today</code> .	139
magyar-1.1.5	
General: Still more spelling corrections	138
magyar-1.2	
General: Update for \LaTeX 2 ϵ	138
magyar-1.3c	
General: Now use <code>\@nopatterns</code> to produce the warning	138
Removed the use of <code>\filedate</code> and moved identification after the loading of <code>babel.def</code>	138
magyar-1.3e	
<code>\captionsmagyar</code> : Added <code>\proofname</code> for AMS- \LaTeX	138

magyar-1.3f	
\captionsmagyar: translated Proof and replaced some translations	138
magyar-1.3g	
General: Replaced \undefined with \@undefined and \empty with \@empty for consistency with L ^A T _E X	138
norsk-1.0a	
General: Renamed babel.sty in babel.com	128
norsk-1.0c	
General: Removed use of \@ifundefined	128
norsk-1.1	
General: Added a warning when no hyphenation patterns were loaded.	128
Brought up-to-date with babel 3.2a	128
\captionsnorsk: \headpagename should be \pagename	128
Added \seename, \alsoname and \prefacename	128
\captionsnynorsk: \headpagename should be \pagename	129
Added \seename, \alsoname and \prefacename	129
norsk-1.1.3	
General: Added a couple of translations (from Per Norman Oma, TeX@itk.unit.no)	128
norsk-1.2	
General: Update for L ^A T _E X 2 _ε	128
norsk-1.2d	
General: Now use \@nopatterns to produce the warning	128
Removed the use of \filedate and moved identification after the loading of babel.def	128
norsk-1.2f	
\captionsnorsk: Added \proofname for AMS-L ^A T _E X	128
\norskhyphenmins: Added setting of hyphenmin parameters	128
norsk-1.2g	
\captionsnorsk: Replaced ‘Proof’ by its translation	128
\captionsnynorsk: Replaced ‘Proof’ by its translation	129
norsk-1.2h	
General: Moved the definition of \atcatcode right to the beginning.	128
Now use \ldf@finish to wrap up	130
Now use \LdfInit to perform initial checks	128
Replaced \undefined with \@undefined and \empty with \@empty for consistency with L ^A T _E X	128
polish-1.0a	
\textpl: Initially execute ‘textpl’	153
polish-1.1c	
General: Now use \@nopatterns to produce the warning	150
Removed the use of \filedate and moved identification after the loading of babel.def	150
polish-1.1d	
General: The dqmacro for C used ‘c’	154
polish-1.2a	
General: Don’t modify \rm and friends for L ^A T _E X 2 _ε , take \selectfont instead	153
polish-1.2b	
\captionspolish: Added \proofname for AMS-L ^A T _E X	150
polish-1.2d	
General: Now use \ldf@finish to wrap up	154
Now use \LdfInit to perform initial checks	150

Replaced <code>\undefined</code> with <code>\@undefined</code> and <code>\empty</code> with <code>\@empty</code> for consistency with L ^A T _E X, moved the definition of <code>\atcatcode</code> right to the beginning.	150
<code>\Eob</code> : Use the constructed version of the characters only in OT1; use proper characters in T1.	152
<code>\skb</code> : This macro is meant to be used in horizontal mode; so leave vertical mode if necessary	152
<code>\sob</code> : This macro is meant to be used in horizontal mode; so leave vertical mode if necessary	152
<code>\spb</code> : This macro is meant to be used in horizontal mode; so leave vertical mode if necessary	152
portuges-1.0a	
General: Renamed <code>babel.sty</code> in <code>babel.com</code>	102
portuges-1.0b	
General: Removed use of <code>cs@ifundefined</code>	102
portuges-1.1	
General: Added a warning when no hyphenation patterns were loaded.	102
Brought up-to-date with babel 3.2a	102
<code>\captionportuges</code> : <code>\headpagename</code> should be <code>\pagename</code>	103
Added <code>\seename</code> , <code>\alsoname</code> and <code>\prefacename</code>	103
portuges-1.2	
General: Update for L ^A T _E X 2 _ε	102
portuges-1.2d	
General: Now use <code>\@nopatterns</code> to produce the warning	102
Removed the use of <code>\filedate</code> and moved identification after the loading of <code>babel.def</code>	102
portuges-1.2e	
<code>\captionportuges</code> : Added a few missing translations	103
portuges-1.2g	
General: Enhanced support for brasilian	102
<code>\captionbrazil</code> : The captions for brasilian and portuguese are different now	104
<code>\extraportuges</code> : Added the definition of some " shorthands	105
<code>\ord</code> : Added macro	105
<code>\orda</code> : Added macro	105
<code>\portugeshyphenmins</code> : Added setting of hyphenmin values	105
<code>\ra</code> : Added macro	105
<code>\ro</code> : Added macro	105
portuges-1.2h	
<code>\captionportuges</code> : Added <code>\proofname</code> for AMS-L ^A T _E X	103
portuges-1.2i	
<code>\captionbrazil</code> : Added <code>\proofname</code> for AMS-L ^A T _E X	104
<code>\captionportuges</code> : Substituted ‘Prova’ for ‘Proof’	103
portuges-1.2j	
General: Moved the definition of <code>\atcatcode</code> right to the beginning.	102
Now use <code>\LdfInit</code> to perform initial checks	102
ow use <code>\ldf@finish</code> to wrap up	106
Replaced <code>\undefined</code> with <code>\@undefined</code> and <code>\empty</code> with <code>\@empty</code> for consistency with L ^A T _E X	102
romanian-1.0a	
General: Renamed <code>babel.sty</code> in <code>babel.com</code>	123
romanian-1.0b	
General: Removed use of <code>\@ifundefined</code>	123
romanian-1.1	
General: Added a warning when no hyphenation patterns were loaded.	123

Brought up-to-date with babel 3.2a	123
<code>\captionsromanian: \headpagename</code> should be <code>\pagename</code>	123
Added <code>\seename</code> , <code>\alsoname</code> and <code>\prefacename</code>	123
Translation errors found by Robert Juhasz fixed	123
<code>\dateromanian</code> : Translation errors found by Robert Juhasz fixed	123
romanian-1.2	
General: Update for LaTeX2e	123
romanian-1.2d	
General: Now use <code>\@nopatterns</code> to produce the warning	123
Removed the use of <code>\filedate</code> and moved identification after the loading of babel.def	123
romanian-1.2e	
General: Updated for babel release 3.5	123
romanian-1.2f	
<code>\captionsromanian</code> : Added <code>\proofname</code> for AMS-L ^A T _E X	123
romanian-1.2g	
<code>\captionsromanian</code> : Added translation of ‘Proof’	123
romanian-1.2h	
General: Now use <code>\ldf@finish</code> to wrap up	124
Now use <code>\LdfInit</code> to perform initial checks	123
Replaced <code>\undefined</code> with <code>\@undefined</code> and <code>\empty</code> with <code>\@empty</code> for consis- tency with L ^A T _E X, moved the definition of <code>\atcatcode</code> right to the beginning.	123
russianb-1.1a	
<code>\extrasrussian</code> : Use <code>\ddot</code> instead of <code>\@MATHUMLAUT</code>	166
use <code>\russianhyphenmins</code> to store the correct values	167
Use the new mechanism for dealing with active chars	164
<code>\russian@sh@?@</code> : Use new <code>\DefineActiveNoArg</code>	164
Use the more general mechanism of <code>\declare@shorthand</code>	164
<code>\system@sh@;@</code> : Added system level shorthands	165
russianb-1.1b	
<code>\extrasrussian</code> : Added switch to L ^W N encoding	164
<code>\russian@sh@?@</code> : Updated to reflect the latest french definitions	164
<code>\verbatim@font</code> : Added changing of <code>\verbatim@font</code>	164
russianb-1.1c	
General: Replaced <code>\undefined</code> with <code>\@undefined</code> and <code>\empty</code> with <code>\@empty</code> for consistency with L ^A T _E X	161
russianb-1.1d	
General: Moved the definition of <code>\atcatcode</code> right to the beginning.	161
Now use <code>\ldf@finish</code> to wrap up	167
Now use <code>\LdfInit</code> to perform initial checks	161
russianb-1.1e	
General: Added closing brace to second argument of <code>\LdfInit</code>	161
russianb-1.1f	
<code>\extrasrussian</code> : added missing - in ”-	166
scottish-1.0b	
General: Corrected typos (PR1652)	79
scottish-1.0c	
<code>\captionsscottish</code> : Added <code>\proofname</code> for AMS-L ^A T _E X	79
scottish-1.0d	
General: Now use <code>\ldf@finish</code> to wrap up	80
Now use <code>\LdfInit</code> to perform initial checks	79
Replaced <code>\undefined</code> with <code>\@undefined</code> and <code>\empty</code> with <code>\@empty</code> for consis- tency with L ^A T _E X, moved the definition of <code>\atcatcode</code> right to the beginning.	79

slovak-1.0	
General: First version	156
slovak-1.2	
General: Update for L ^A T _E X 2 _ε	156
slovak-1.2b	
\noextrasslovak: Added setting of left- and righthyphenmin	157
slovak-1.2d	
General: Now use \@nopatterns to produce the warning	156
Removed the use of \filedate and moved identification after the loading of	
babel.def	156
slovak-1.2e	
\noextrasslovak: Now use \slovakhyphenmins	157
Use L ^A T _E X's \v accent command	157
slovak-1.2g	
\captionsslovak: Added \proofname for AMS-L ^A T _E X	156
slovak-1.2i	
General: Now use \ldf@finish to wrap up	157
Now use \LdfInit to perform initial checks	156
Replaced \undefined with \@undefined and \empty with \@empty for consistency with L ^A T _E X, moved the definition of \atcatcode right to the beginning.	156
slovene-1.0a	
General: Renamed babel.sty in babel.com	158
slovene-1.0b	
General: Removed use of \@ifundefined	158
slovene-1.1	
General: Added a warning when no hyphenation patterns were loaded.	158
Brought up-to-date with babel 3.2a	158
\captionsslovene: \headpagename should be \pagename	158
Added \seename, \alsoname and \prefacename	158
slovene-1.2	
General: Update for L ^A T _E X 2 _ε	158
slovene-1.2b	
\captionsslovene: Added extra translations from Josef Leydold, leydold@statix2.wu-wien.ac.at	158
slovene-1.2d	
General: Now use \@nopatterns to produce the warning	158
Removed the use of \filedate and moved identification after the loading of	
babel.def	158
slovene-1.2f	
\noextrasslovene: Introduced the active "	159
slovene-1.2g	
\captionsslovene: Added \proofname for AMS-L ^A T _E X	158
slovene-1.2h	
\captionsslovene: Added translation of 'Proof'	158
slovene-1.2i	
General: Now use \ldf@finish to wrap up	160
Now use \LdfInit to perform initial checks	158
Replaced \undefined with \@undefined and \empty with \@empty for consistency with L ^A T _E X, moved the definition of \atcatcode right to the beginning.	158
Replaced 'Slovakian' with correct 'Slovenian'	158
\noextrasslovene: removed shorthand for "L as it is not needed for slovenian	159
spanish-1.1	
General: Date format corrected. Wrong change history deleted	107

spanish-1.1a	
General: \I does not exist, modified	107
spanish-2.0	
General: Modified for babel 3.0	107
Now use \adddialect if language undefined	108
spanish-2.0a	
General: removed use of \setlanguage	107
spanish-2.0b	
General: New check before loading babel.sty	107
spanish-2.0c	
\captionsspanish: Removed \global definitions	108
\datespanish: Removed csglobal definitions	109
spanish-2.0d	
\datespanish: Capitalize months as suggested by E. Torrente (TORRENTE@CERNVM).	109
spanish-2.1	
General: Added catalan as a ‘dialect’	107
spanish-2.1a	
General: Renamed babel.sty in babel.com	107
spanish-3.0	
General: Catalan deleted	107
Major rewriting, new macros, active accents, catalan removed	107
\captionsspanish: Capitals are accented, some strings changed	108
\datespanish: Uncapitalize months, since that seems to be the correct, modern usage	109
\extrasspanish: Formerly empty, all code is new.	109
spanish-3.0a	
\@tilde: Added fix for \dotlessi	110
General: Text fixed	108
spanish-3.1	
General: Added warning, if no spanish patterns were loaded	108
Brought up-to-date with babel 3.2a	107
removed use of \@ifundefined	108
\captionsspanish: \headpagename should be \pagename	108
added \seename, and \alsoname and \prefacename	108
\extrasspanish: Rewrote the macro.	109
spanish-3.1a	
General: The accents had to be made active during their own definition. Changed address for goya.	107
spanish-3.1b	
General: Added address, phone and fax for Julio Sánchez. The definition of the active tilde was not being restored on exit.	107
spanish-3.2	
\@tilde: All this code is new	110
General: Active character definitions changed as in germanb.	107
Changed \acute to \textacute and \tilde to \texttilde because the old names were already used for math accents.	109
Update for L ^A T _E X 2 _ε	107
\captionsspanish: added translated strings for \seename \alsoname and \prefacename	108
\extrasspanish: Major rewrite. Now works like in germanb and dutch.	109
spanish-3.3d	
General: Now use \@nopatterns to produce the warning	108

Removed the use of <code>\filedate</code> and moved identification after the loading of <code>babel.def</code>	107
spanish-3.4a	
<code>\extrasspanish</code> : Yet another major rewrite	109
<code>\noextrasspanish</code> : All the code for handling active characters is now moved to <code>babel.def</code>	109
spanish-3.4b	
General: corrected typo (PR1652)	107
spanish-3.4c	
General: Added <code>”</code> as an extra shorthand, removed <code>’n</code> as a shorthand	111
Changed mathmode definition of acute shorthands to expand to a single prime followed by the next character in the input	111
made active acute optional	107
<code>\captionsspanish</code> : Added <code>\proofname</code> for AMS- \LaTeX	108
spanish-3.4d	
<code>\captionsspanish</code> : Added translation of ‘Proof’	108
<code>\noextrasspanish</code> : These two actions can not be combined in one <code>\ifpackagewith</code> statement for some reason	109
spanish-3.4e	
General: Now use <code>\ldf@finish</code> to wrap up	111
Now use <code>\LdfInit</code> to perform initial checks	108
Replaced <code>\undefined</code> with <code>\@undefined</code> and <code>\empty</code> with <code>\@empty</code> for consistency with \LaTeX , moved the definition of <code>\atcatcode</code> right to the beginning.	107
spanish-3.4f	
General: Added <code>\leavevmode</code> to definitions of <code>"a</code> and <code>"o</code>	111
<code>\noextrasspanish</code> : Removed two unwanted space tokens that turned up in the output	109
swedish-1.0a	
General: Renamed <code>babel.sty</code> in <code>babel.com</code>	131
swedish-1.0b	
<code>\captionsswedish</code> : added definition for <code>\enclname</code>	131
made definition of <code>\refname</code> pluralis	131
removed type in definition of <code>\contentsname</code>	131
swedish-1.0c	
General: Removed use of <code>\@ifundefined</code>	131
swedish-1.1	
General: Added a warning when no hyphenation patterns were loaded.	131
Brought up-to-date with babel 3.2a	131
<code>\captionsswedish</code> : <code>\headpagename</code> should be <code>\pagename</code>	131
Added <code>\seename</code> , <code>\alsoname</code> and <code>\prefacename</code>	131
swedish-1.1b	
<code>\captionsswedish</code> : Added translations	131
swedish-1.2	
General: Update for $\text{\LaTeX}2\epsilon$	131
swedish-1.3d	
General: Now use <code>\@nopatterns</code> to produce the warning	131
Removed the use of <code>\filedate</code> and moved identification after the loading of <code>babel.def</code>	131
<code>\captionsswedish</code> : Changed <code>\aa</code> to <code>\csname aa\endcsname</code> , to make <code>\uppercase</code> do the right thing	131
swedish-1.3e	
General: Update for release 3.5	131
<code>\captionsswedish</code> : Changed <code>\alsoname</code> from ‘se också’	132
<code>\extrasswedish</code> : Added <code>\bbl@frenchspacing</code>	132

\noextrasswedish: Added \bbl@nonfrenchspacing	132
\swedishhyphenmins: use \swedishhyphenmins to store the correct values	132
swedish-1.3f	
\captionsswedish: Added \proofname for AMS- \LaTeX	131
swedish-1.3g	
\captionsswedish: Replaced ‘Proof’ by its translation	132
swedish-2.0	
General: Introduced active double quote	131
\noextrasswedish: Added active double quote character	132
swedish-2.1	
General: Now use \ldf@finish to wrap up	134
Now use \LdfInit to perform initial checks	131
Replaced \undefined with \@undefined and \empty with \@empty for consistency with \LaTeX , moved the definition of \atcatcode right to the beginning.	131
turkish-1.1	
\captionsturkish: \headpagename should be \pagename	174
turkish-1.2	
General: Update for \LaTeX 2 ϵ	174
turkish-1.2b	
\captionsturkish: Added braces behind \i in strings	174
\dateturkish: Added braces behind \i in strings	175
turkish-1.2c	
General: Now use \@nopatterns to produce the warning	174
Removed the use of \filedate and moved identification after the loading of babel.def	174
turkish-1.2d	
\dateturkish: removed extra closing brace, \mont should be \month	175
\turkish@sh@: Added missing \def	175
turkish-1.2e	
\extrasturkish: Completely rewrote macro	175
\noextrasturkish: now use \bbl@frenchspacing and \bbl@nonfrenchspacing	175
\turkish@sh@: Use the new mechanism of \declare@shorthand	175
turkish-1.2f	
\captionsturkish: Added \proofname for AMS- \LaTeX	174
turkish-1.2h	
General: Replaced \undefined with \@undefined and \empty with \@empty for consistency with \LaTeX	174
turkish-1.2i	
General: Moved the definition of \atcatcode right to the beginning.	174
Now use \ldf@finish to wrap up	176
ow use \LdfInit to perform initial checks	174
usorbian-0.1	
General: First version	170
usorbian-0.1b	
General: Made it possible to run through \LaTeX ; added \MF and removed extra \endmacro	170
usorbian-0.1c	
\captionsusorbian: Removed two typos (Kapitel and Dodatki)	170
usorbian-1.0a	
General: Removed stuff that has been moved to babel.def	171
usorbian-1.0b	
\captionsusorbian: Added \proofname for AMS- \LaTeX	170

usorbian-1.0c	
General: Now use <code>\bbl@disc</code>	172
usorbian-1.0d	
General: Replaced <code>\undefined</code> with <code>\@undefined</code> and <code>\empty</code> with <code>\@empty</code> for consistency with L ^A T _E X	170
usorbian-1.0e	
General: Moved the definition of <code>\atcatcode</code> right to the beginning.	170
Now use <code>\ldf@finish</code> to wrap up	172
Now use <code>\LdfInit</code> to perform initial checks	170