

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

Johannes Braams
Kersengarde 33
2723 BP Zoetermeer
The Netherlands
JLBraams@cistron.nl

Printed 3rd April 2001

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	8 Compatibility with ngerman.sty	10
1 The user interface	2 9 Compatibility with the french package	10
1.1 Languages supported by Babel	3	
1.2 Workarounds	4	10 Conclusion
2 Changes for L^AT_EX 2_ε	4 11 Acknowledgements	11
3 Changes in Babel version 3.7	5 12 The Esperanto language	12
4 Changes in Babel version 3.6	6 13 The Dutch language	12
5 Changes in Babel version 3.5	7 14 The English language	13
6 The interface between the core of babel and the language definition files	7 15 The German language	13
6.1 Support for active characters	9	16 The German language – new orthography
6.2 Support for saving macro definitions	9	13
6.3 Support for extending macros	9	17 The Breton language
6.4 Macros common to a number of languages	9	14
7 Compatibility with german.sty	10 19 The Irish language	14

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

20	The Scottish language	14	39	The Croatian language	33
21	The Greek language	15	40	The Czech language	34
	21.1 Typing conventions	15			
	21.2 Greek numbering	15	41	The Polish language	34
22	The French language	16	42	The Serbocroatian language	34
23	The Italian language	20	43	The Slovak language	35
24	The Latin language	22	44	The Slovenian language	35
25	The Portuguese language	23	45	The Russian language	36
26	The Spanish language	23	46	The Bulgarian language	37
27	The Catalan language	25	47	The Ukrainian language	38
28	The Galician language	25	48	The Lower Sorbian language	39
29	The Basque language	26	49	The Upper Sorbian language	39
30	The Romanian language	27	50	The Turkish language	39
31	The Danish language	27	51	The Hebrew language	39
32	The Icelandic language	27	51.1 Acknowledgement		40
	32.1 Overview	27	52	Hebrew input encodings	40
33	The Norwegian language	28	53	Hebrew font encodings	41
34	The Swedish language	29	54	Hebrew in L^AT_EX 2.09 compatibility mode	41
35	The North Sami language	29	54.1 The DOCSTRIP modules		42
	35.1 The code of <code>samin.dtx</code>	30	55	Writing Hebrew/English thesis with L^AT_EX 2_ε	42
36	The Finnish language	30	56	The Bahasa language	42
37	The Hungarian language	30			
38	The Estonian language	32			

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>hyphenrules</code>	The environment <code>hyphenrules</code> can be used to select <i>only</i> the hyphenation rules to be used. This can for instance be used to select ‘nohyphenation’, provided that in <code>language.dat</code> the ‘language’ nohyphenation is defined by loading <code>serohyph.tex</code> .
<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>\usesshorthands</code>	The command <code>\usesshorthands</code> initiates the definition of user-defined shorthand sequences. It has one argument, the character which starts these personal shorthands.
<code>\definesshorthand</code>	The command <code>\definesshorthand</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 babel package.
<code>\shorthandon</code>	It is sometimes necessary to be able to switch a shorthand character off temporarily because it needs to be used in an entire different way. For this purpose the user commands <code>\shorthandoff</code> and <code>\shorthandon</code> are provided. They each take a list of characters as their arguments. The command <code>\shorthandoff</code> sets the <code>\catcode</code> for each of the characters in its argument to other (12); the command <code>\shorthandon</code> sets the <code>\catcode</code> to active (13). Both commands only perform their duty on ‘known’ shorthand characters. If a character is not known to be a shorthand character its category code will be left unchanged.
<code>\shorthandoff</code>	
<code>\languageattribute</code>	This is a user-level command, to be used in the preamble of a document (after <code>\usepackage[...]{babel}</code> which declares which attributes are to be used for a given language). It takes two arguments, the first being the name of the language, the second a (list of) attribute(s) to used. The command checks whether the language is known in this document and whether the attribute(s) are known for this language.

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.

Language	Option(s)
Afrikaans	afrikaans
Bahasa	bahasa
Basque	basque
Breton	breton
Bulgarian	bulgarian
Catalan	catalan
Croatian	croatian
Czech	czech
Danish	danish

Language	Option(s)
Dutch	dutch
English	english, USenglish, american, UKenglish, british, canadian
Esperanto	esperanto
Estonian	estonian
Finnish	finnish
French	french, francais, canadien, acadian
Galician	galician
German	austrian, german, germanb, ngerman, naustrian
Greek	greek, polutonikogreek
Hebrew	hebrew
Hungarian	magyar, hungarian
Icelandic	icelandic
Irish Gaelic	irish
Italian	italian
Latin	latin
Lower Sorbian	lowersorbian
North Sami	samin
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
Serbian	serbian
Turkish	turkish
Ukrainian	ukrainian
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 \LaTeX 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 \LaTeX 2 ϵ

With the advent of \LaTeX 2 ϵ the interface to babel in the preamble of the document has changed. With \LaTeX 2.09 one used to call up the babel system with a line such as:

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

which would tell \LaTeX 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 3.7

In Babel version 3.7 a number of bugs that were found in version 3.6 are fixed. Also a number of changes and additions have occurred:

- Shorthands are expandable again. The disadvantage is that one has to type ‘`{}`’a when the acute accent is used as a shorthand character. The advantage is that a number of other problems (such as the breaking of ligatures, etc.) have vanished.
- Two new commands, `\shorthandon` and `\shorthandoff` have been introduced to enable to temporarily switch off one or more shorthands.
- Support for typesetting Greek has been enhanced. Code from the `kdgreek` package (suggested by the author) was added and `\greeknumeral` has been added.
- Support for typesetting Basque is now available thanks to Juan Aguirregabiria.
- Support for typesetting Serbian with latin script is now available thanks to Dejan Muhamedagić and Jankovic Slobodan.
- Support for typesetting Hebrew (and potential support for typesetting other right-to-left written languages) is now available thanks to Rama Porrat and Boris Lavva.
- Support for typesetting Bulgarian is now available thanks to Georgi Boshnakov.
- Support for typesetting Latin is now available, thanks to Claudio Beccari and Krzysztof Konrad Żelechowski.
- Support for typesetting North Sami is now available, thanks to Regnor Jernsletten.
- The options `canadian`, `candien` and `acadien` have been added for Canadian English and French use.
- A language attribute has been added to the `\mark...` commands in order to make sure that a greek header line comes out right on the last page before a language switch.
- Hyphenation pattern files are now read *inside a group*; therefore any changes a pattern file needs to make to lowercase codes, uppercase codes and category codes are kept local to that group. If they are needed for the language, these changes will need to be repeated and stored in `\extras...`
- The concept of language attributes is introduced. It is intended to be used for giving the user some control over the use of the features a language definition file provides. It’s first use is for the Greek language where the user can select to use the *πολυτονικό* (“Polutoniko” or multiaccented) Greek way of typesetting texts. These attributes will possibly find wider use in future releases.

- The environment `hyphenrules` is introduced.
- The syntax of the file `language.dat` has been extended in order to be able to (optionally) specify the font encoding to be used while processing the patterns file.
- The command `\providehyphenmins` should now be used in language definition files in order to be able to keep any settings provided by the pattern file.

4 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.
- In babel 3.6k the language Ukrainian has been added and the support for russian typesetting has been adapted to the package ‘cyrillic’ to be released with the december 1998 release of L^AT_EX 2_ε.

5 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.

6 The interface between the core of babel and the language definition files

In the core of the babel system a number of 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 usefull 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 `\<lang>hyphenmins`, `\captions<lang>`, `\date<lang>`, `\extras<lang>` and `\noextras<lang>`; where `<lang>` 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@<lang>` to be a dialect of `\language0` when `\l@<lang>` 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.

<code>\providehyphenmins</code>	The macro <code>\providehyphenmins</code> should be used in the language definition files to set the <code>\lefthyphenmin</code> and <code>\righthyphenmin</code> . This macro will check whether these parameters were provided by the hyphenation file before it takes any action.
<code>\langhyphenmins</code>	The macro <code>\<lang>hyphenmins</code> is used to store the values of the <code>\lefthyphenmin</code> and <code>\righthyphenmin</code> .
<code>\captionslang</code>	The macro <code>\captions<lang></code> defines the macros that hold the texts to replace the original hard-wired texts.
<code>\date<lang></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>\bbl@declare@ttribute</code>	This is a command to be used in the language definition files for declaring a language attribute. It takes three arguments, the name of the language, the attribute to be defined, and the code to be executed when the attribute is to be used.
<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>\ProvidesLanguage</code>	The macro <code>\ProvidesLanguage</code> should be used to identify the language definition files. Its syntax is similar to the syntax of the \LaTeX command <code>\ProvidesPackage</code> .
<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>\ldf@finish</code> .
<code>\substitutefontfamily</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.

6.1 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.e. <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 <code>\dospecial</code> . \LaTeX adds another macro called <code>@sanitize</code> representing the same character set, but without the curly braces. The macros <code>\bbl@add@special⟨char⟩</code> and <code>\bbl@remove@special⟨char⟩</code> add and remove the character <code>⟨char⟩</code> to these two sets.

6.2 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.

<code>\babel@save</code>	To save the current meaning of any control sequence the macro <code>\babel@save</code> is provided. It takes one argument, <code>⟨curname⟩</code> , the control sequence for which the meaning has to be saved.
<code>\babel@savevariable</code>	A second macro is provided to save the current value of a variable. In this context anything that is allowed after the <code>\the</code> primitive is considered to be a variable. The macro takes one argument, the <code>⟨variable⟩</code> . The effect of the aforementioned macros is that a piece of code is appended to the current definition of <code>\originalTeX</code> . When <code>\originalTeX</code> is expanded this code restores the previous definition of the control sequence or the previous value of the variable.

6.3 Support for extending macros

<code>\addto</code>	The macro <code>\addto{⟨control sequence⟩}{⟨\TeX code⟩}</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 <code>\extrasenglish</code> .
---------------------	---

6.4 Macros common to a number of languages

<code>\allowhyphens</code>	In a couple of European languages compound words are used. This means that when \TeX 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 <code>\allowhyphens</code> can be used.
<code>\set@low@box</code>	For some languages quotes need to be lowered to the baseline. For this purpose the macro <code>\set@low@box</code> is available. It takes one argument and puts that argument in an <code>\hbox</code> , at the baseline. The result is available in <code>\box0</code> for further processing.
<code>\save@sf@q</code>	Sometimes it is necessary to preserve the <code>\spacefactor</code> . For this purpose the macro

¹This mechanism was introduced by Bernd Raichle.

`\save@sf@q` is available. It takes one argument, saves the current spacefactor, executes the argument and restores the spacefactor.

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

7 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 .

8 Compatibility with `ngerman.sty`

When used with the options `ngerman` or `naustrian` `babel` will provide all features of the package `ngerman`. There is however one exception: The commands for special hyphenation of double consonants ("ff etc.) and ck ("ck), which are no longer required with the new German orthography, are undefined. With the `ngerman` package, however, these commands will generate appropriate warning messages only.

9 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`.

Therefor, `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`.

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

10 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.

11 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.

12 The Esperanto language

The file `esperanto.dtx`³ defines all the language-specific macros for the Esperanto language.

For this language the character `^` is made active. In table 1 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 1: The functions of the active character for Esperanto.

13 The Dutch language

The file `dutch.dtx`⁴ defines all the language-specific macros for the Dutch language and the ‘Afrikaans’ version⁵ of it.

For this language the character `"` is made active. In table 2 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’`.

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

Table 2: The extra definitions made by `dutch.ldf`

³The file described in this section has version number ? and was last revised on ?. 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.

⁴The file described in this section has version number v3.8h, and was last revised on 2001/01/30.

⁵contributed by Stoffel Lombard (lombc@b31pc87.up.ac.za)

14 The English language

The file `english.dtx`⁶ 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.

15 The German language

The file `germanb.dtx`⁷ defines all the language definition macros for the German language as well as for the Austrian dialect of this language⁸.

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 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 in table 3 can also be

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

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

typeset by using the commands in table 4.

16 The German language – new orthography

The file `ngermanb.dtx`⁹ defines all the language definition macros for the German language with the ‘new orthography’ introduced in August 1998. This includes also the Austrian dialect of this language.

As with the ‘traditional’ German orthography, the character `"` is made active, and the commands in table 3 can be used, except for `"ck` and `"ff` etc., which are no longer required.

⁶The file described in this section has version number v3.3k and was last revised on 2001/02/07.

⁷The file described in this section has version number v2.6k and was last revised on 2001/01/26.

⁸This file is a re-implementation of Hubert Partl’s `german.sty` version 2.5b, see [4].

⁹The file described in this section has version number v2.6m and was last revised on 2001/02/06.

<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 4: More commands which produce quotes, defined by `german.ldf`

The internal language names are `ngerman` and `naustrian`.

17 The Breton language

The file `breton.dtx`¹⁰ 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.

18 The Welsh language

The file `welsh.dtx`¹¹ 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.

19 The Irish language

The file `irish.dtx`¹² defines all the language definition macros for the Irish language.

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

20 The Scottish language

The file `scottish.dtx`¹³ defines all the language definition macros for the Scottish language.

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

¹⁰The file described in this section has version number v1.0h and was last revised on 2001/02/19.

¹¹The file described in this section has version number v1.0c and was last revised on 2001/01/31.

¹²The file described in this section has version number v1.0h and was last revised on 2001/01/30. A contribution was made by Marion Gunn.

¹³The file described in this section has version number v1.0g and was last revised on 2001/01/30. A contribution was made by Fraser Grant (FRASER@CERNVM).

21 The Greek language

The file `greek.dtx`¹⁴ defines all the language definition macros for the Greek language, i.e., as it used today with only one accent, and the attribute *πολυτονικό* (“Polutoniko”) for typesetting greek text with all accents. This separation arose out of the need to simplify things, for only very few people will be really interested to typeset polytonic Greek text.

`\greektext` The commands `\greektext` and `\latintext` can be used to switch to greek or latin fonts. These are declarations.

`\latintext` The commands `\textgreek` and `\textlatin` both take one argument which is then typeset using the requested font encoding. The command `\greekol` switches to the greek outline font family, while the command `\textol` typesets a short text in outline font. A number of extra greek characters are made available through the added text commands `\stigma`, `\qoppa`, `\sampi`, `\ddigamma`, `\Digamma`, `\euro`, `\permill`, and `\vardigamma`.

21.1 Typing conventions

Entering greek text can be quite difficult because of the many diacritical signs that need to be added for various purposes. The fonts that are used to typeset Greek make this a lot easier by offering a lot of ligatures. But in order for this to work, some characters need to be considered as letters. These characters are `<`, `>`, `~`, `'`, `,`, `"` and `|`. Therefore their `\lccode` is changed when Greek is in effect. In order to let `\uppercase` give correct results, the `\uccode` of these characters is set to a non-existing character to make them disappear. Of course not all characters are needed when typesetting “modern” *μονοτονικό*. In that case we only need the `'` and `"` symbols which are treated in the proper way.

21.2 Greek numbering

The Greek alphabetical numbering system, like the Roman one, is still used in everyday life for short enumerations. Unfortunately most Greeks don’t know how to write Greek numbers bigger than 20 or 30. Nevertheless, in official editions of the last century and beginning of this century this numbering system was also used for dates and numbers in the range of several thousands. Nowadays this numbering system is primary used by the Eastern Orthodox Church and by certain scholars. It is hence necessary to be able to typeset any Greek numeral up to 999 999. Here are the conventions:

- There is no Greek numeral for any number less than or equal to 0.
- Numbers from 1 to 9 are denoted by letters alpha, beta, gamma, delta, epsilon, stigma, zeta, eta, theta, followed by a mark similar to the mathematical symbol “prime”. (Nowadays instead of letter stigma the digraph sigma tau is used for number 6. Mainly because the letter stigma is not always available, so people opt to write down the first two letters of its name as an alternative. In our implementation we produce the letter stigma, not the digraph sigma tau.)
- Decades from 10 to 90 are denoted by letters iota, kappa, lambda, mu, nu, xi, omikron, pi, qoppa, again followed by the numeric mark. The qoppa used for this purpose has a special zig-zag form, which doesn’t resemble at all the original ‘q’-like qoppa.
- Hundreds from 100 to 900 are denoted by letters rho, sigma, tau, upsilon, phi, chi, psi, omega, sampi, followed by the numeric mark.

¹⁴The file described in this section has version number v1.3j and was last revised on 2001/02/22. The original author is Apostolos Syropoulos (apostolo@platon.ee.duth.gr), code from `kdgreek.sty` by David Kastrup dak@neuroinformatik.ruhr-uni-bochum.de was used to enhance the support for typesetting greek texts.

- Any number between 1 and 999 is obtained by a group of letters denoting the hundreds decades and units, followed by a numeric mark.
- To denote thousands one uses the same method, but this time the mark is placed in front of the letter, and under the baseline (it is inverted by 180 degrees). When a group of letters denoting thousands is followed by a group of letters denoting a number under 1000, then both marks are used.

`\greeknumeral` Using these conventions one obtains numbers up to 999 999. The command `\greeknumeral` makes it possible to typeset Greek numerals. There is also an `\Greeknumeral` “uppercase” version of this macro: `\Greeknumeral`.

Another system which was in wide use only in Athens, could express any positive number. This system is implemented in package `athnum`.

22 The French language

The file `frenchb.dtx`¹⁵, 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 $\text{Plain}\text{\TeX}$ formats. If you are still using \LaTeX 2.09 , you *should* consider switching to $\text{\LaTeX 2}\epsilon$!

The command `\selectlanguage{french}` switches to the French language¹⁶, 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 default items in `itemize` environment are set to ‘—’ instead of •, and no vertical spacing and no glue is added, a hook to reset standard \LaTeX spacing is provided (`\FrenchItemizeSpacingfalse`); it is possible to change ‘—’ to something else (‘—’ for instance) by redefining `\FrenchLabelItem`; apart from the global hook `\FrenchLabelItem`, it is also possible to change the ‘labelitems’ at any level (1 to 4) in French, using the standard \LaTeX syntax, for instance: `\renewcommand{\labelitemii}{\textbullet}`; in order to be effective in French, the redefinitions have to be made when French is the current language (i.e. *after* the `\begin{document}`), the changes are saved when switching to another language and will be remembered of, when switching back to French;
6. vertical spacing in general \LaTeX lists is shortened, a hook to reset standard \LaTeX settings is provided (`\FrenchListSpacingfalse`);
7. the first paragraph of each section is indented (\LaTeX only);
8. the space after `\dots` is removed in French.

Some commands are provided in `frenchb` to make typesetting easier:

¹⁵The file described in this section has version number ? and was last revised on ?.

¹⁶`\selectlanguage{français}` and `\selectlanguage{frenchb}` are kept for backward compatibility but should no longer be used.

1. French quotation marks can be entered 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 $\text{\LaTeX 2}_{\epsilon}$ and 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 and better horizontal spacing). `\og` and `\fg` can be used outside French, they provide then English quotes “ and ”.
2. A command `\up` is provided to typeset superscripts like `M\up{me}` (abbreviation for “Madame”), `1\up{er}` (for “premier”).
3. 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.
4. Commands `\primo`, `\secundo`, `\tertio` and `\quarto` may be used to enumerate in lists.
5. Abbreviations for “Numéro” and “numéro” are obtained via the commands `\No`, `\no`.
6. Two commands are provided to typeset the symbol 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).
7. In math-mode the comma has to be surrounded with braces to avoid a spurious space being inserted after it, in decimal numbers for instance (see the \TeXbook p. 134). The command `\DecimalMathComma` makes the comma be an ordinary character *in French only* (no space added); as a counterpart, if `\DecimalMathComma` is active, an explicit space has to be added in lists and intervals: `$(0,\ 1)$`, `$(x,\ y)$`. `\StandardMathComma` switches back to the standard behaviour of the comma.
8. A command `\nombre` is provided to easily typeset 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): each slice of three digits will be separated either with a comma in English or with a space in French (if you prefer a thin space instead of a normal space, just add the command `\ThinSpaceInFrenchNumbers` to the preamble of your document, or to `frenchb.cfg`). The command `\nombre` is a contribution of Vincent Jalby using ideas of David Carlisle in `comma.sty`.
9. `frenchb` has been designed to take advantage of the `xspace` package if present: adding `\usepackage{xspace}` in the preamble will force macros like `\fg`, `\ier`, `\ieme`, `\dots`, ..., to respect the spaces you type after them, for instance typing `‘1\ier juin’` will print ‘1^{er} juin’ (no need for a forced space after `1\ier`).
10. Two commands, `\FrenchLayout` and `\StandardLayout` (to be used *only in the preamble*) are provided to unify the layout of multilingual documents (it mainly concerns lists) regardless the current language (see section ?? for details).

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 (see the example configuration file below in section ??), 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 be 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 behaviour of `frenchb`.

Once you have built your format, a good precaution would be to perform some basic tests about hyphenation in French. For $\LaTeX 2_{\epsilon}$ I suggest this:

- 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).

```
%% Test file for French hyphenation.
\documentclass{article}
\usepackage[my-encoding]{inputenc}
\usepackage[T1]{fontenc} % Use EC fonts for French
%\usepackage{mltex}      % Uncomment this line and
                        % comment out the preceeding one
                        % to use standard CM fonts.
\usepackage[francais]{babel}
\begin{document}
\showhyphens{signal container \`ev\`enement alg\`ebre}
\showhyphens{signal container évènement algèbre}
\end{document}
```

- check the hyphenations proposed by \TeX 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 US-English, not for French;
- you get no hyphen at all in évé-ne-ment, this probably means that you are using CM fonts and the macro `\accent` to produce accented characters. Using EC fonts with built-in accented characters or \LaTeX with CM fonts avoids this type of mismatch.

`frenchb` has been improved using helpful suggestions from many people, the main contributions came from Vincent Jalby. Thanks to all of them!

Changes

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 \LaTeX 2_ε); also removed aliases `\french` and `\english` (`frenchb.cfg` is a better place for these).

Changes in version 1.3:

- The ‘`xspace`’ package, when present, now controls spacing after all (sensible) macros, formerly ‘`xspace`’ worked on `\fg` (suggested by Vincent Jalby);
- spacing after opening and before closing guillemets improved as suggested by Thierry Bouche;
- a replacement for poor looking guillemets in OT1 encoding (*math* fonts are used to emulate them) is provided if the file `ot2wncyr.fd` is found: this file defines an OT2 encoding using AMS Cyrillic fonts; these have built-in guillemets, they are *text* fonts, are free, and are distributed as METAFONT and Type1 fonts (good for pdf-tex!), the replacement was suggested by Gérard Degrez; if the files `ot2wncyr.fd`, `wncy*.mf`, `wncy*.tfm`, and `wncy*.pfb` aren't available on your system, you *should* get them from CTAN; if your system complains about missing `wncy*.tfm` fonts, it means your \TeX system is *incomplete*, as a quick fix, you can either remove `ot2wncyr.fd` or add the command `\LasyGuillemets` to the preamble of your document or to `frenchb.cfg`, then `frenchb` will behave as it did in the previous versions.

In case Cyrillic guillemets do not fit, it is possible to pick French guillemets from any font using the command `\FrenchGuillemetsFrom` which requires 4 arguments `\FrenchGuillemetsFrom{coding}{font}{char-left}{char-right}`, for instance, add `\FrenchGuillemetsIn{T1}{ppl}{19}{20}` to the preamble of your document or to `frenchb.cfg`, to pick French guillemets from Palatino (font-name=ppl, char 19 and 20 are left and right guillemets in T1-encoding). This was suggested by Michel Bovani.

- the environment ‘`itemize`’ has been redesigned in French according to specifications from Jacques André and Thierry Bouche; it is possible to switch back to the settings of version 1.2: add `\FrenchItemizeSpacingfalse` *after* loading `frenchb` in the preamble (this can be useful to process old documents);
- two switches have been added to go back to standard \LaTeX list spacing `\FrenchItemizeSpacingfalse` and `\FrenchListSpacingfalse`;

- `\nombre` now properly handles signs in $\text{\LaTeX 2}\epsilon$;
- definition of `\dots` changed in French;
- in French, with the standard \LaTeX classes, captions in figures and tables are printed with an endash instead of a colon.

Changes in version 1.4:

- the redefinition of `\@makecaption` is changed not to overwrite the changes made by some classes (`koma-script`, `amsart`, `ua-thesis`...) as pointed out by Werner Lemberg;
- a hook, `\FrenchGuillemetsFrom`, is provided to pick French guillemets from any font (suggested by Michel Bovani);
- a hook, `\FrenchLabelItem`, is provided to enable marks other than ‘–’ (`\textendash`) in French lists (also suggested by Michel Bovani);
- `\DecimalMathComma`, `\StandardMathComma`, `\ThinSpaceInFrenchNumbers` added;
- `\FrenchLayout` and `\StandardLayout` added;
- an example of customization file `frenchb.cfg` is included in `frenchb.dtx`.

Changes in version 1.5:

- The settings for spacing in French lists are no longer tuned at the `\@trivlist` level but within `\list`; this enables the users to provide their own settings for the lists they define with `\list` (suggested by P. Pichureau). Although the layout of the standard lists *has not changed*, some lists, based directly on `\@trivlist` or `\trivlist`, could possibly be typeset differently when upgrading from version 1.4 to 1.5. The command `\FrenchOldTrivlisttrue` could then, be useful to process older documents.
- Apart from the global hook `\FrenchLabelItem`, it is now also possible to change the ‘labelitems’ at any level (1 to 4) in French, using the standard \LaTeX syntax, for instance: `\renewcommand{\labelitemii}{\textbullet}`.
- The internal name for the French language has been changed from `frenchb` to `french`, it means that `\captionfrench`, `\datefrench`, `\extrasfrench`, `\noextrasfrench`, are to be used now instead of `\captionfrenchb`, `\datefrenchb`, `\extrasfrenchb`, `\noextrasfrenchb`.

23 The Italian language

The file `italian.dtx`¹⁷ defines all the language-specific macros for the Italian language. The features of this language definition file are the following:

1. The Italian hyphenation is invoked, provided that file `ithyph.tex` was loaded when the $\text{\LaTeX 2}\epsilon$ format was built; in case it was not, read the information coming with your implementation of the \TeX software and the `babel` documentation.
2. The language dependent fixed words to be inserted by such commands as `\chapter`, `\caption`, `\tableofcontents`, etc. are redefined in accordance with the Italian typographical practice.

¹⁷The file described in this section has version number v1.2o and was last revised on 2001/01/19. The original author is Maurizio Codogno, (`mau@beatles.cselt.stet.it`). It has been largely revised by Johannes Braams and Claudio Beccari

3. Since Italian can be easily hyphenated and Italian practice allows to break a word before the last two letters, hyphenation parameters have been set accordingly, but a very high demerit value has been set in order to avoid word breaks in the penultimate line of a paragraph. Specifically the `\clubpenalty`, and the `\widowpenalty` are set to rather high values and `\finalhyphendemerits` is set to such a high value that hyphenation is prohibited between the last two lines of a paragraph.
4. Some language specific shortcuts have been defined so as to allow etymological hyphenation, specifically `"` inserts a break point in any word boundary that the typesetter chooses, provided it is not followed by an accented letter (very unlikely in Italian, where compulsory accents fall only on the last and ending vowel of a word, but may take place with compound words that include foreign roots), and `"|` when the desired break point falls before an accented letter.
5. The shortcut `"` introduces the raised (English) opening double quotes; this shortcut proves its usefulness when one reminds that the Italian keyboard misses the backtick key, and the backtick on a Windows based platform may be obtained only by pressing the `Alt` key while inputting the numerical code 0096; very, very annoying!
6. The shortcuts `"<` and `">` insert the French guillemets, sometimes used in Italian typography; with the T1 font encoding the ligatures `<<` and `>>` should insert such signs directly, but not all the virtual fonts that claim to follow the T1 font encoding actually contain the guillemets; with the OT1 encoding the guillemets are not available and must be faked in some way. By using the `"<` and `">` shortcuts (even with the T1 encoding) the necessary tests are performed and in case the suitable glyphs are taken from other fonts normally available with any good, modern \LaTeX distribution.
7. The accent internal macros used with the OT1 encoding are redefined so as to avoid elaborate input sequences such as `\‘{\i}` that can be replaced with the much simpler and readable `\‘i`, as it happens with the T1 encoding; the redefinition is valid for all accents.
8. Three new specific commands `\unit`, `\ped`, and `\ap` are introduced so as to enable the correct composition of technical mathematics according to the ISO 31/XI recommendations.

For this language a limited number of shortcuts has been defined, table 5, some of which are used to overcome certain limitations of the Italian keyboard; in section ?? there are other comments and hints in order to overcome some other keyboard limitations.

<code>"</code>	inserts a compound word mark where hyphenation is legal; it allows etymological hyphenation which is recommended for technical terms, chemical names and the like; it does not work if the next character is represented with a control sequence or is an accented character.
<code>" </code>	the same as the above without the limitation on characters represented with control sequences or accented ones.
<code>"</code>	inserts open quotes “.
<code>"<</code>	inserts open guillemets.
<code>"></code>	inserts closed guillemets.
<code>"/</code>	equivalent to <code>\slash</code>

Table 5: Shortcuts for the Italian language

References

- [1] Beccari C., “Computer Aided Hyphenation for Italian and Modern Latin”, TUGboat vol. 13, n. 1, pp. 23-33 (1992).
- [2] Beccari C., “Typesetting mathematics for science and technology according to ISO 31/XI”, TUGboat vol. 18, n. 1, pp. 39-48 (1997).

24 The Latin language

The file `latin.dtx`¹⁸ defines all the language-specific macros for the Latin language both in modern and medieval spelling.

For this language the `\clubpenalty`, `\widowpenalty` are set to rather high values and `\finalhyphendemerits` is set to such a high value that hyphenation is prohibited between the last two lines of a paragraph.

For this language two “styles” of typesetting are implemented: “regular” or modern-spelling Latin, and medieval Latin. The medieval Latin specific commands can be activated by means of the language attribute `medieval`; the medieval spelling differs from the modern one by the systematic use of the lower case ‘u’ also where in modern spelling the letter ‘v’ is used; when typesetting with capital letters, on the opposite, the letter ‘V’ is used also in place of ‘U’. Medieval spelling also includes the ligatures `\ae` (æ), `\oe` (œ), `\AE` (Æ), and `\OE` (Œ) that are not used in modern spelling, nor were used in the classical times.

For what concerns `babel` and typesetting with \LaTeX , the differences between the two styles of spelling reveal themselves in the strings used to name for example the “Preface” that becomes “Praefatio” or “Præfatio” respectively. Hyphenation rules are also different, but the hyphenation pattern file `lahyph.tex` takes care of both versions of the language. Needless to say that such patterns must be loaded in the \LaTeX format by running `initex` (or whatever the name of the initializer) on `latex.ltx`.

The name strings for chapters, figures, tables, etcetera, are suggested by prof. Raffaella Tabacco, a classicist of the University of Turin, Italy, to whom we address our warmest thanks. The names suggested by Krzysztof Konrad Żelechowski, when different, are used as the names for the medieval variety, since he made a word and spelling choice more suited for this variety.

For this language some shortcuts are defined according to table 6; all of them are supposed to work with both spelling styles, except where the opposite is explicitly stated.

<code>~i</code>	inserts the breve accent as <i>ï</i> ; valid also for the other lowercase vowels, but it does not operate on the medieval ligatures æ and œ.
<code>=a</code>	inserts the macron accent as <i>ā</i> ; valid also for the other lowercase vowels, but it does not operate on the medieval ligatures æ and œ.
<code>"</code>	inserts a compound word mark where hyphenation is legal; the next character must not be a medieval ligature æ or œ, nor an accented letter (foreign names).
<code>" </code>	same as above, but operates also when the next character is a medieval ligature or an accented letter.

Table 6: Shortcuts defined for the Latin language

¹⁸The file described in this section has version number v2.0c and was last revised on 2001/02/19. The original author is Claudio Beccari with contributions by Krzysztof Konrad Żelechowski, (kkz@alfa.mimuw.edu.pl)

25 The Portuguese language

The file `portuges.dtx`¹⁹ 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 7 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 7: The extra definitions made by `portuges.ldf`

26 The Spanish language

The file `spanish.dtx`²⁰ defines all the language-specific macros for the Spanish language.

Customization is made following mainly the books on the subject by José Martínez de Sousa. By typesetting `spanish.dtx` directly you will get the full documentation (regrettably is in Spanish only, but it is pretty long). References in this part refers to that document. There are several additional features documented in the Spanish version only.

This style provides:

- Translations following the International L^AT_EX conventions, as well as `\today`.
- Shorthands listed in Table 8. Examples in subsection 3.4 are illustrative. Note that "~ has a special meaning in spanish different to other languages, and is used mainly in linguistic contexts.
- `\deactivatetilden` deactivates the ~n and ~N shorthands.
- *In math mode* a dot followed by a digit is replaced by a decimal comma.
- Spanish ordinals and abbeviations with `\sptext` as, for instance, `1\sptext{er}`. The preceptive dot is included.
- Accented functions: lím, máx, mín, mód. You may globally omit the accents with `\unaccentedoperators`. Spaced functions: arc cos, etc. You may globally kill that space with `\unspacedoperators`. `\dotlessi` is provided for use in math mode.
- A quoting environment and a related pair of shorthands << and >>. See subsection 3.7 for examples. The command `\deactivatequoting` deactivates these shorthand in case you want to use < and > in some AMS commands and numerical comparisons.

¹⁹The file described in this section has version number v1.2o and was last revised on 2001/02/16. Contributions were made by Jose Pedro Ramallete (JRAMELHE@CERNVM or Jose-Pedro_Ramallete@MACMAIL) and Arnaldo Viegas de Lima arnaldo@VNET.IBM.COM.

²⁰The file described in this section has version number v4.1c and was last revised on 2001/01/30. The original author from v4.0 on is Javier Bezos. Previous versions were by Julio Sánchez, whose hyphenation patterns are still strongly recommended.

'a	acute accented a. Also for: e, i, o, u (both lowercase and uppercase).
'n	ñ (also uppercase).
~n	ñ (also uppercase).
"u	ü (also uppercase).
"a	Ordinal numbers (also "A, "o, "O).
"c	ç (also uppercase).
"rr	rr, but -r when hyphenated
"-	Like \-, but allowing hyphenation in the rest the word.
"=	Like -, but allowing hyphenation in the rest the word.
"~	The hyphen is repeated at the very beginning of the next line if the word is hyphenated at this point.
" "	Like " - but producing no hyphen sign.
~-	Like - but with no break after the hyphen. Also for: en-dashes (~--) and em-dashes (~---).
"/	A slash slightly lowered, if necessary.
"	disable ligatures at this point.
"<	Left guillemets.
">	Right guillemets.
<<	\begin{quoting}. (See text.)
>>	\end{quoting}. (See text.)

Table 8: Extra definitions made by file `spanish.ldf`

- The command `\selectspanish` selects the spanish language *and* its shorthands. (Intended for the preamble.)
- `\frenchspacing` is used.
- Ellipsis are best typed ... or, within a sentence, \...
- There is a small space before \%.
- `\lsc` provides lowercase small caps. (See subsection 3.10.)

Just in case spanish is the main language, the group `\layoutspanish` is activated, which modifies the standard classes through the whole document (it cannot be deactivated) in the following way (see section 4):

- Both `enumerate` and `itemize` are adapted to Spanish rules.
- Both `\alph` and `\Alph` include ñ after *n*.
- Symbol footmarks are one, two, three, etc., asteriscs.
- OT1 guillemets are generated with two lasy symbols instead of small \ll and \gg.
- `\roman` is redefined to write small caps roman numerals, since lowercase roman numerals are not allowed. However, *MakeIndex* rejects entries containing pages in that format. The `.idx` file must be preprocessed if the document has this kind of entries with the provided `romanidx.tex` tool—just \TeX it and follow the instructions.
- There is a dot after section numbers in titles and toc.

This group is ignored if you write `\selectspanish*` in the preamble.

Some additional commands are provided to be used in the `spanish.cfg` file:

- With `\es@activeacute` acute accents are always active, overriding the default babel behaviour.

- `\es@enumerate` sets the labels to be used by `enumerate`. The same applies to `\es@itemize` and `itemize`.
- `\es@operators` stores the operator commands (subsection 3.6). All of them are canceled with

```
\let\es@operators\relax
```

The commands `\deactivatequoting`, `\deactivatetilden` and `\selectspanish` may be used in this file, too.

A subset of these commands is provided for use in Plain TeX (with `\input spanish.sty`).

27 The Catalan language

The file `catalan.dtx`²¹ 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 9 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 10 also become available²².

<code>\l.l</code>	geminated-l digraph (similar to l-l). <code>\L.L</code> produces the uppercase version.
<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 9: Extra definitions made by file `catalan.ldf` (activated by default)

These active accents characters behave according to their original definitions if not followed by one of the characters indicated in that table.

28 The Galician language

The file `galician.dtx`²³ defines all the language definition macros for the Galician language.

²¹The file described in this section has version number v2.2n and was last revised on 2001/02/19.

²²Please 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`.

²³The file described in this section has version number v1.2k and was last revised on 2001/02/20.

- 'e acute accented a, allowing hyphenation in the rest of the word. Valid for the following vowels: e, i, o, u (both lowercase and uppercase).
- 'a grave accented a, allowing hyphenation in the rest of the word. Valid for the following vowels: a, e, o (both lowercase and uppercase).

Table 10: Extra definitions made by file `catalan.ldf` (activated only when using the options `activeacute` and `activegrave`)

For this language the characters ' ~ and " are made active. In table 11 an overview is given of their purpose. These active accents character behave according to their original

- "| 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.
- '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.
- "u a u with dieresis allowing hyphenation.
- "a feminine ordinal as in 1^a.
- "o masculine ordinal as in 1^o.
- ~n a n with tilde. Works for uppercase too.

Table 11: The extra definitions made by `galician.ldf`

definitions if not followed by one of the characters indicated in that table.

29 The Basque language

The file `basque.dtx`²⁴ defines all the language definition macro's for the Basque language.

For this language the characters ~ and " are made active. In table 12 an overview is given of their purpose. These active accent characters behave according to their original

- "| 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 French left double quotes (similar to <<).
- "> for French right double quotes (similar to >>).
- ~n a n with tilde. Works for uppercase too.

Table 12: The extra definitions made by `basque.ldf`

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. Support is based on providing an appropriate definition for the accent macros on entry to the

²⁴The file described in this section has version number v1.0e and was last revised on 2001/01/30. The original author is Juan M. Aguirregabiria, (`wtpagagj@lg.ehu.es`) and is based on the Spanish file by Julio Sánchez, (`jsanchez@gmv.es`).

Basque language. This is automatically done by $\text{\LaTeX 2}_{\epsilon}$ or NFSS2. If T1 encoding is chosen, and provided that adequate hyphenation patterns²⁵ are available. The easiest way to use the new encoding with $\text{\LaTeX 2}_{\epsilon}$ is to load the package `t1enc` with `\usepackage`. This must be done before loading `babel`.

30 The Romanian language

The file `romanian.dtx`²⁶ defines all the language-specific macros for the Romanian language.

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

31 The Danish language

The file `danish.dtx`²⁷ defines all the language definition macros for the Danish 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.
" "	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 13: The extra definitions made by `danish.ldf`

32 The Icelandic language

32.1 Overview

The file `iceland.dtx`²⁸ defines all the language definition macros for the Icelandic language

Customization for the Icelandic language was made following several official and semiofficial publications [2, 3, 1, 6, 5]. These publications do not always agree and we indicate those instances.

For this language the character " is made active. In table 14 an overview is given of its purpose. The shorthands in table 14 can also be typeset by using the commands in table 15.

References

- [1] Alþingi. *Reglur um frágang þingskjala og prentun umræðna*, 1988.

²⁵One source for such patterns is the archive at `lcp07.lc.ehu.es` that can be accessed by anonymous FTP or in <http://lcp07.lc.ehu.es/Basque.html>

²⁶The file described in this section has version number v1.2k and was last revised on 2001/01/30. A contribution was made by Umstatter Horst (`hhu@cernvm.cern.ch`).

²⁷The file described in this section has version number v1.3n and was last revised on 2001/01/30. A contribution was made by Henning Larsen (`larsen@cernvm.cern.ch`)

²⁸The file described in this section has version number ? and was last revised on ?.

"	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 Icelandic left double quotes (looks like „).
"’	for Icelandic right double quotes.
">	for Icelandic ‘french’ left double quotes (similar to >>).
"<	for Icelandic ‘french’ right double quotes (similar to <<).
"o	for old Icelandic o
"O	for old Icelandic O
"ó	for old Icelandic ó
"Ó	for old Icelandic Ó
"e	for old Icelandic e
"E	for old Icelandic E
"é	for old Icelandic é
"É	for old Icelandic É
\tala	for typesetting numbers
\grada	for the ‘degree’ symbol
\gradur	for ‘degrees’, e.g. 5 °C
\upp	for textsuperscript

Table 14: The shorthands and extra definitions made by `icelandic.ldf`

- [2] Auglýsing um greinarmerkjasetningu. Stj.tíð B, nr. 133/1974, 1974.
- [3] Auglýsing um breyting auglýsingu nr. 132/1974 um íslenska stafsetningu. Stj.tíð B, nr. 261/1977, 1977.
- [4] Einar Haugen, editor. *First Grammatical Treatise*. Longman, London, 2 edition, 1972.
- [5] Staðlaráð Íslands og Fagráð í upplýsingatækni, Reykjavík. *Forstaðall FS 130:1997*, 1997.
- [6] STRÍ Staðlaráð Íslands. *SI - kerfið*, 2 edition, 1994.

33 The Norwegian language

The file `norsk.dtx`²⁹ defines all the language definition macros for the Norwegian language as well as for an alternative variant ‘nynorsk’ of this language.

For this language the character " is made active. In table 16 an overview is given of its purpose.

Rune Kleveland distributes a Norwegian dictionary for `ispell` (570000 words). It can be found at <http://www.uio.no/~runekl/dictionary.html>.

²⁹The file described in this section has version number v2.0h and was last revised on 2001/01/12. 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); the shorthand definitions were provided by Rune Kleveland (runekl@math.uio.no).

<code>\ilqq</code>	for Icelandic left double quotes (looks like „).
<code>\irqq</code>	for Icelandic right double quotes (looks like “).
<code>\ilq</code>	for Icelandic left single quotes (looks like ,).
<code>\irq</code>	for Icelandic right single quotes (looks like ‘).
<code>\iflqq</code>	for Icelandic ‘french’ left double quotes (similar to >>).
<code>\ifrqq</code>	for Icelandic ‘french’ right double quotes (similar to <<).
<code>\ifrq</code>	for Icelandic ‘french’ right single quotes (similar to <).
<code>\iflq</code>	for Icelandic ‘french’ left single quotes (similar to >).
<code>\dq</code>	the original quotes character (").
<code>\oob</code>	for old Icelandic o
<code>\Oob</code>	for old Icelandic O
<code>\ooob</code>	for old Icelandic ó
<code>\OOob</code>	for old Icelandic Ó
<code>\eob</code>	for old Icelandic e
<code>\Eob</code>	for old Icelandic E
<code>\eeob</code>	for old Icelandic é
<code>\EEob</code>	for old Icelandic É

Table 15: Commands which produce quotes and old Icelandic diacritics, defined by `icelandic.ldf`

This dictionary supports the spellings `spi"sslede` for ‘spisslede’ (hyphenated spisslede) and other such words, and also suggest the spelling `spi"sslede` for ‘spisslede’ and ‘spisslede’.

34 The Swedish language

The file `swedish.dtx`³⁰ defines all the language-specific macros for the Swedish language. This file has borrowed heavily from `finnish.dtx` and `germanb.dtx`.

For this language the character " is made active. In table 17 an overview is given of its purpose. The vertical placement of the "umlaut" in some letters can be controlled this way.

Two variations for formatting of dates are added. `\datesymd` makes `\today` output dates formatted as YYYY-MM-DD, which is commonly used in Sweden today. `\datesdmy` formats the date as D/M YYYY, which is also very common in Sweden. These commands should be issued after `\begindocument`.

35 The North Sami language

The file `samin.dtx`³¹ defines all the language definition macros for the North Sami language.

Several Sami dialects/languages are spoken in Finland, Norway, Sweden and on the Kola Peninsula (Russia). The alphabets differ, so there will eventually be a need for more .dtx files for e.g. Lule and South Sami. Hence the name `samin.dtx` (and not `sami.dtx` or the like) in the North Sami case.

There are currently no hyphenation patterns available for the North Sami language, but you might consider using the patterns for Finnish (`fi8hyph.tex`), Norwegian

³⁰The file described in this section has version number v2.3c and was last revised on 2001/03/12. Contributions were made by Sten Hellman (`HELLMAN@CERNVM.CERN.CH`) and Erik Östhols (`erik_osthols@yahoo.com`).

³¹The file described in this section has version number v1.0c and was last revised on 2000/10/04. It was written by Regnor Jernsletten, (`Regnor.Jernsletten@sami.uit.no`) or (`Regnor.Jernsletten@eunet.no`).

"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. (o"ppussing)
"ee	Hyphenate "ee as \ 'e-e. (komit"een)
"-	an explicit hyphen sign, allowing hyphenation in the composing words. Use this for compound words when the hyphenation patterns fail to hyphenate properly. (alpin"-anlegg)
"	Like "-", but inserts 0.03em space. Use it if the compound point is spanned by a ligature. (hoff" intriger)
""	Like "-", but producing no hyphen sign. (i""g\aa{}r)
"~	Like -, but allows no hyphenation at all. (E"~cup)
"=	Like -, but allowing hyphenation in the composing words. (marksistisk"=leninistisk)
"<	for French left double quotes (similar to <<).
">	for French right double quotes (similar to >>).

Table 16: The extra definitions made by norsk.sty

(nohyph.tex) or Swedish (sehyph.tex). Add a line for the samin language to the language.dat file, and rebuild the L^AT_EX format file. See the documentation for your L^AT_EX distribution.

A note on writing North Sami in L^AT_EX: The T_I encoding and EC fonts do not include the T WITH STROKE letter, which you will need a workaround for. My suggestion is to place the lines

```
\newcommand{\tx}{\mbox{t\hspace{-.35em}-}}
\newcommand{\txx}{\mbox{T\hspace{-.5em}-}}
in the preamble of your documents. They define the commands
\txx{} for LATIN CAPITAL LETTER T WITH STROKE and
\tx{} for LATIN SMALL LETTER T WITH STROKE.
```

35.1 The code of samin.dtx

36 The Finnish language

The file finnish.dtx³² defines all the language definition macros for the Finnish language.

For this language the character " is made active. In table 18 an overview is given of its purpose.

37 The Hungarian language

The file option magyar.dtx defines all the language definition macros for the Hungarian language.

The babel support for the Hungarian language until file version 1.3i was essentially changing the English document elements to Hungarian ones, but because of the differences between these two languages this was actually unusable ('Part I' was transferred to 'Rész I' which is not usable instead of 'I. rész'). To enhance the typesetting facilities for Hungarian the following should be considered:

- In Hungarian documents there is a period after the part, section, subsection etc. numbers.

³²The file described in this section has version number v1.3o and was last revised on 2001/03/12. A contribution was made by Mikko KANERVA (KANERVA@CERNVM) and Keranen Reino (KERANEN@CERNVM).

"a	Gives ä, also implemented for "A, "o and "O.
"w, "W	gives å and Å.
"ff	for ff to be hyphenated as ff-f. Used for compound words, such as stra"ffånge, which should be hyphenated as straff-fånge. This is also implemented for b, d, f, g, l, m, n, p, r, s, and t.
"	disable ligature at this position. This should be used for compound words, such as "stra"ffinrättning", which should not have the ligature "ffi".
"-	an explicit hyphen sign, allowing hyphenation in the rest of the word, such as e. g. in "x"-axeln".
" "	like "-", but producing no hyphen sign (for words that should break at some sign such as och/" "eller).
"~	for an explicit hyphen without a breakpoint; useful for expressions such as "2"~3 veckor" where no linebreak is desirable.
"=	an explicit hyphen sign allowing subsequent hyphenation, for expressions such as "studiebidrag och -lån".
\-	like the old \-, but allowing hyphenation in the rest of the word.

Table 17: The extra definitions made by `swedish.sty`

- In the part, chapter, appendix name the number (or letter) goes before the name, so 'Part I' translates to 'I. rész'.
- The same is true with captions ('Table 2.1' goes to '2.1. táblázat').
- There is a period after the caption name instead of a colon. ('Table 2.1:' goes to '2.1. táblázat.')
- There is a period at the end of the title in a run-in head (when `afterskip<0` in `\@startsection`).
- Special hyphenation rules must be applied for the so-called long double consonants (ccs, ssz,...).

"	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 18: The extra definitions made by `finnish.ldf`

- The opening quotation mark is like the German one (the closing is the same as in English).
- In Hungarian figure, table, etc. referencing a definite article is also incorporated. The Hungarian definite articles behave like the English indefinite ones ('a/an'). 'a' is used for words beginning with a consonant and 'az' goes for a vowel. Since some numbers begin with a vowel some others with a consonant some commands should be provided for automatic definite article generation.

Until file version 1.3i³³ the special typesetting rules of the Hungarian language mentioned above were not taken into consideration. This version (v1.4c)³⁴ enables babel to typeset 'good-looking' Hungarian texts.

<code>\ontoday</code>	The <code>\ontoday</code> command works like <code>\today</code> but produces a slightly different date format used in expressions such as 'on February 10th'.
<code>\Az</code>	The commands <code>\Az#1</code> and <code>\az#1</code> write the correct definite article for the argument and the argument itself (separated with a ~). The star-forms (<code>\Az*</code> and <code>\az*</code>) produce the article only.
<code>\Azr</code>	<code>\Azr#1</code> and <code>\azr#1</code> treat the argument as a label so expand it then write the definite article for <code>\r@#1</code> , a non-breakable space then the label expansion. The star-forms do not print the label expansion. <code>\Azr(#1</code> and <code>\azr(#1</code> are used for equation referencing with the syntax <code>\azr(label)</code> .
<code>\Aref</code>	There are two aliases <code>\Aref</code> and <code>\aref</code> for <code>\Azr</code> and <code>\azr</code> , respectively. During the preparation of a document it is not known in general, if the code <code>'a~\ref{label}'</code> or the code <code>'az~\ref{label}'</code> is the grammatically correct one. Writing <code>'\aref{label}'</code> instead of the previous ones solves the problem.
<code>\Azp</code>	<code>\Azp#1</code> and <code>\azp#1</code> also treat the argument as a label but use the label's page for definite article determination. There are star-forms giving only the definite article without the page number.
<code>\Apageref</code>	There are aliases <code>\Apageref</code> and <code>\apageref</code> for <code>\Azp</code> and <code>\azp</code> , respectively. The code <code>\apageref{label}</code> is equivalent either to <code>a~\pageref{label}</code> or to <code>az~\pageref{label}</code> .
<code>\Azc</code>	<code>\Azc</code> and <code>\azc</code> work like the <code>\cite</code> command but (of course) they insert the definite article. There can be several comma separated cite labels and in that case the definite article is given for the first one. They accept <code>\cite</code> 's optional argument. There are star-forms giving the definite article only.
<code>\Acite</code>	There are aliases <code>\Acite</code> and <code>\acite</code> for <code>\Azc</code> and <code>\azc</code> , respectively. For this language the character ' ' is made active. Table 19 shows the shortcuts. The main reason for the activation of the ' ' character is to handle the special hyphenation of the long double consonants.

38 The Estonian language

The file `estonian.dtx`³⁵ 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 20. These active accent characters behave according to

³³That file was last revised on 1996/12/23 with a contribution by the next authors: Attila Koppányi (attila@cernvm.cern.ch), Árpád Biró (JZP1104@HUSZEG11.bitnet), István Hamecz (hami@ursus.bke.hu) and Dezső Horváth (horvath@pisa.infn.it).

³⁴It was written by József Bérces (jozsi@docs4.mht.bme.hu) with some help from Ferenc Wettl (wettl@math.bme.hu) and an idea from David Carlisle (david@dcarlisle.demon.co.uk).

³⁵The file described in this section has version number v1.0h and was last revised on 2001/01/30. The original author is Enn Saar, (saar@aai.ee).

shortcut	explanation	example
‘ ‘	same as \glqq in babel, or \quotedblbase in T1 (opening quotation mark, like „)	‘ ‘id\’ezet’’→„idézet”
‘c, ‘C	ccs is hyphenated as cs-cs	lo‘ccsan→locs-csan
‘d, ‘D	ddz is hyphenated as dz-dz	e‘ddz\’unk→edz-dzünk
‘g, ‘G	ggy is hyphenated as gy-gy	po‘ggy\’asz→pogy-gyász
‘l, ‘L	lly is hyphenated as ly-ly	Kod\’a‘llyal→Kodály-lyal
‘n, ‘N	nny is hyphenated as ny-ny	me‘nnyei→meny-nyei
‘s, ‘S	ssz is hyphenated as sz-sz	vi‘ssza→visz-sza
‘t, ‘T	tty is hyphenated as ty-ty	po‘ttyan→poty-tyan
‘z, ‘Z	zsz is hyphenated as zs-zs	ri‘zzsel→rizzs-zsel

Table 19: The shortcuts defined in magyar.ldf

~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 20: The extra definitions made by estonian.ldf

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.

39 The Croatian language

The file croatian.dtx³⁶ defines all the language definition macros for the Croatian language.

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

³⁶The file described in this section has version number v1.3k and was last revised on 2001/01/30. A contribution was made by Alan Pać (paica@cernvm.cern.ch).

40 The Czech language

The file `czech.dtx`³⁷ 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 ř. The command `\w` is used to put the ring-accent which appears in ångström over the letters u and U.

41 The Polish language

The file `polish.dtx`³⁸ defines all the language-specific macros for the Polish language.

For this language the character " is made active. In table 21 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 21: The extra definitions made by `polish.sty`

42 The Serbocroatian language

The file `serbian.dtx`³⁹ defines all the language definition macros for the Serbian language, typeset in a latin script. In a future version support for typesetting in a cyrillic script may be added.

For this language the character " is made active. In table 22 an overview is given of its purpose. One of the reasons for this is that in the Serbian language some special characters are used.

³⁷The file described in this section has version number v1.3j and was last revised on 2001/01/30. Contributions were made by Milos Lokajicek (LOKAJICK@CERNVM).

³⁸The file described in this section has version number v1.2j and was last revised on 2001/01/30.

³⁹The file described in this section has version number v1.0d and was last revised on 2001/01/30. A contribution was made by Dejan Muhamedagić (dejan@unix.com).

"c	\ "c, also implemented for the lowercase and uppercase s and z.
"d	\dj, also implemented for "D
"-	an explicit hyphen sign, allowing hyphenation in the rest of the word.
"	disable ligature at this position
"	like "-", but producing no hyphen sign (for compound words with hyphen, e.g. x-"y).
"‘	for Serbian left double quotes (looks like „).
"’	for Serbian right double quotes.
"<	for French left double quotes (similar to <<).
">	for French right double quotes (similar to >>).

Table 22: The extra definitions made by `serbian.ldf`

Apart from defining shorthands we need to make sure that the first paragraph of each section is intended. Furthermore the following new math operators are defined (`\tg`, `\ctg`, `\arctg`, `\arcctg`, `\sh`, `\ch`, `\th`, `\cth`, `\arsh`, `\arch`, `\arth`, `\arcth`, `\Prob`, `\Expect`, `\Variance`).

43 The Slovak language

The file `slovak.dtx`⁴⁰ 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'.

44 The Slovenian language

The file `slovene.dtx`⁴¹ defines all the language-specific macros for the Slovenian language.

For this language the character " is made active. In table 23 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 23: The extra definitions made by `slovene.ldf`

⁴⁰The file described in this section has version number v1.2l and was last revised on 2001/01/30. It was written by Jana Chlebková (`chlebk@euromath.dk`).

⁴¹The file described in this section has version number v1.2m and was last revised on 2001/01/30. Contributions were made by Danilo Zavrtanik, University of Ljubljana (YU) and Leon Žlajpah (`leon.zlajpah@ijs.si`).

45 The Russian language

The file `russianb.dtx`⁴² defines all the language-specific macros for the Russian language. It needs the file `cyr.cod` for success documentation with Russian encodings (see below).

For this language the character " is made active. In table 24 an overview is given of its purpose.

"	disable ligature at this position.
"-	an explicit hyphen sign, allowing hyphenation in the rest of the word.
"---	Cyrillic emdash in plain text.
"--~	Cyrillic emdash in compound names (surnames).
"--*	Cyrillic emdash for denoting direct speech.
"	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.
",	thinspace for initials with a breakpoint in following surname.
"‘	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 24: The extra definitions made by `russianb`

The quotes in table 24 can also be typeset by using the commands in table 25.

<code>\cdash---</code>	Cyrillic emdash in plain text.
<code>\cdash--~</code>	Cyrillic emdash in compound names (surnames).
<code>\cdash--*</code>	Cyrillic emdash for denoting direct speech.
<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 25: More commands which produce quotes, defined by `babel`

The French quotes are also available as ligatures ‘<<’ and ‘>>’ in 8-bit Cyrillic font encodings (LCY, X2, T2*) and as ‘<’ and ‘>’ characters in 7-bit Cyrillic font encodings (OT2 and LWN).

The quotation marks traditionally used in Russian were borrowed from other languages (e.g., French and German) so they keep their original names.

⁴²The 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 \LaTeX 2.09), `russian.sty` version v2.5c (for \LaTeX 2 ϵ) and `francais.sty` version 4.5c and `germanb.sty` version 2.5c were added.

46 The Bulgarian language

The file `bulgarian.dtx`⁴³ provides the language-specific macros for the Bulgarian language.

Users should take note of the various “cyrillic” dashes available now (see below). These should remove many causes of headache. Also, although by default the Bulgarian quotation marks will appear automatically when typesetting in Bulgarian, it is better to use the new commands `\''` and `\''` which explicitly typeset them. Note: automatic switch to Bulgarian quotation is withdrawn for the moment and may not be reintroduced at all.

For this language the character `"` is made active. In table 26 an overview is given of its purpose.

<code>" </code>	disable ligature at this position.
<code>"-</code>	an explicit hyphen sign, allowing hyphenation in the rest of the word.
<code>"---</code>	Cyrillic emdash in plain text.
<code>"--~</code>	Cyrillic emdash in compound names (surnames).
<code>"--*</code>	Cyrillic emdash for denoting direct speech.
<code>"</code>	like <code>"-</code> , but producing no hyphen sign (for compound words with hyphen, e.g. <code>x-"y</code> or some other signs as “disable/enable”).
<code>"~</code>	for a compound word mark without a breakpoint.
<code>"=</code>	for a compound word mark with a breakpoint, allowing hyphenation in the composing words.
<code>" ,</code>	thinspace for initials with a breakpoint in following surname.
<code>"‘</code>	for German left double quotes (looks like „).
<code>"’</code>	for German right double quotes (looks like “).
<code>"<</code>	for French left double quotes (looks like <<).
<code>"></code>	for French right double quotes (looks like >>).

Table 26: The extra definitions made by `bulgarian`

The quotes in table 26 can also be typeset by using the commands in table 27.

<code>\cdash---</code>	Cyrillic emdash in plain text.
<code>\cdash--~</code>	Cyrillic emdash in compound names (surnames).
<code>\cdash--*</code>	Cyrillic emdash for denoting direct speech.
<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 (<code>"</code>).

Table 27: More commands which produce quotes, defined by `babel`

The French quotes are also available as ligatures `‘<<’` and `‘>>’` in 8-bit Cyrillic font encodings (LCY, X2, T2*) and as `‘<’` and `‘>’` characters in 7-bit Cyrillic font encodings (OT2 and LWN).

⁴³The file described in this section has version number ? and was last revised on ?. This file was initially derived from the August-1998 version of `russianb.dtx`.

It is (reasonably) backward compatible with the 1994/1996 (non-babel) `bulgarian` style (`bulgaria.sty`) by Georgi Boshnakov—files prepared for that style should compile successfully (with vastly improved appearance due to usage of standard fonts).

The quotation marks traditionally used in Bulgarian were borrowed from German o they keep their original names. French quotation marks may be seen as well in older books.

47 The Ukrainian language

The file `ukraineb.dtx`⁴⁴ defines all the language-specific macros for the Ukrainian language. It needs the file `cyr.cod` for success documentation with Ukrainian encodings (see below).

For this language the character " is made active. In table 28 an overview is given of its purpose.

"	disable ligature at this position.
" -	an explicit hyphen sign, allowing hyphenation in the rest of the word.
" ---	Cyrillic emdash in plain text.
" --~	Cyrillic emdash in compound names (surnames).
" --*	Cyrillic emdash for denoting direct speech.
" "	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.
" ,	thinspace for initials with a breakpoint in following surname.
" ‘	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 28: The extra definitions made by `ukraineb`

The quotes in table 28 (see, also table 24) can also be typeset by using the commands in table 29 (see, also table 25).

<code>\cdash---</code>	Cyrillic emdash in plain text.
<code>\cdash--~</code>	Cyrillic emdash in compound names (surnames).
<code>\cdash--*</code>	Cyrillic emdash for denoting direct speech.
<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 29: More commands which produce quotes, defined by `babel`

The French quotes are also available as ligatures ‘<<’ and ‘>>’ in 8-bit Cyrillic font encodings (LCY, X2, T2*) and as ‘<’ and ‘>’ characters in 7-bit Cyrillic font encodings (OT2 and LWN).

⁴⁴The file described in this section has version number ?. This file was derived from the `russianb.dtx` version 1.1g.

The quotation marks traditionally used in Ukrainian and Russian languages were borrowed from other languages (e.g. French and German) so they keep their original names.

48 The Lower Sorbian language

The file `lsorbian.dtx`⁴⁵ It defines all the language-specific macros for Lower Sorbian.

49 The Upper Sorbian language

The file `usorbian.dtx`⁴⁶ It defines all the language-specific macros for Upper Sorbian.

50 The Turkish language

The file `turkish.dtx`⁴⁷ defines all the language definition macros for the Turkish language⁴⁸.

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 `\frenhspacing` is set.

51 The Hebrew language

The file `hebrew.dtx`⁴⁹ provides the following packages and files for Hebrew language support:

`hebrew.ldf` file defines all the language-specific macros for the Hebrew language.

`rlbabel.def` file is used by `hebrew.ldf` for bidirectional versions of the major `LATEX` commands and environments. It is designed to be used with other right-to-left languages, not only with Hebrew.

`hebcals.sty` package defines a set of macros for computing Hebrew date from Gregorian one.

Additional Hebrew input and font encoding definition files that should be included and used with `hebrew.ldf` are:

`hebinp.dtx` provides Hebrew input encodings, such as ISO 8859-8, MS Windows codepage 1255 or IBM PC codepage 862 (see Section 52 on page 40).

`hebrew.fdd` contains Hebrew font encodings, related font definition files and `hebfont` package that provides Hebrew font switching commands (see Section 53 on page 41 for further details).

⁴⁵The file described in this section has version number v1.0f and was last revised on 2001/01/30. It was written by Eduard Werner (edi@kaihh.hanse.de).

⁴⁶The file described in this section has version number v1.0i and was last revised on 2001/01/30. It was written by Eduard Werner (edi@kaihh.hanse.de).

⁴⁷The file described in this section has version number v1.2m and was last revised on 2001/02/19.

⁴⁸Mustafa Burc, z6001@rziris01.rrz.uni-hamburg.de provided the code for this file. It is based on the work by Pierre Mackay; Turgut Uyar, uyar@cs.itu.edu.tr supplied additional translations in version 1.2j and later

⁴⁹The Hebrew language support files described in this section have version number v2.0c and were last revised on 2001/02/16.

L^AT_EX 2.09 compatibility files are included with `heb209.dtx` and gives possibility to compile existing L^AT_EX 2.09 Hebrew documents with small (if any) changes (see Section 54 on page 41 for details).

Finally, optional document class `hebtech` may be useful for writing theses and dissertations in both Hebrew and English (and any other languages included with `babel`). It designed to meet requirements of the Graduate School of the Technion — Israel Institute of Technology. See more details in Section 55 on page 42.

51.1 Acknowledgement

The following people have contributed to Hebrew package in one way or another, knowingly or unknowingly. In alphabetical order: Irina Abramovici, Yaniv Bargury, Yael Dubinsky, Sergio Fogel, Dan Haran, Rama Porrat, Michail Rozman, Alon Ziv.

Tatiana Samoilov and Vitaly Surazhsky found a number of serious bugs in preliminary version of Hebrew package.

A number of other people have contributed comments and information. Specific contributions are acknowledged within the document.

I want to thank my wife, Vita, and son, Mishka, for their infinite love and patience.

If you made a contribution and I haven't mentioned it, don't worry, it was an accident. I'm sorry. Just tell me and I will add you to the next version.

52 Hebrew input encodings

Hebrew input encodings defined in file `hebinp.dtx`⁵⁰ should be used with `inputenc` L^AT_EX 2_ε package. This package allows the user to specify an input encoding from this file (for example, ISO Hebrew/Latin 8859-8, IBM Hebrew codepage 862 or MS Windows Hebrew codepage 1255) by saying:

```
\usepackage[encoding name]{inputenc}
```

The encoding can also be selected in the document with:

```
\inputencoding{encoding name}
```

The only practical use of this command within a document is when using text from several documents to build up a composite work such as a volume of journal articles. Therefore this command will be used only in vertical mode.

The encodings provided by this package are:

- si960 7-bit Hebrew encoding for the range 32–127. This encoding also known as “old-code” and defined by Israeli Standard SI-960.
- 8859-8 ISO 8859-8 Hebrew/Latin encoding commonly used in UNIX systems. This encoding also known as “new-code” and includes hebrew letters in positions starting from 224.
- cp862 IBM 862 code page commonly used by DOS on IBM-compatible personal computers. This encoding also known as “pc-code” and includes hebrew letters in positions starting from 128.
- cp1255 MS Windows 1255 (hebrew) code page which is similar to 8859-8. In addition to hebrew letters, this encoding contains also hebrew vowels and dots (nikud).

Each encoding has an associated `.def` file, for example `8859-8.def` which defines the behaviour of each input character, using the commands:

⁵⁰The files described in this section have version number v1.0a and were last revised on 2000/09/26.


```
\DeclareInputText{slot}{text}
\DeclareInputMath{slot}{math}
```

This defines the input character *slot* to be the *text* material or *math* material respectively. For example, 8859-8.def defines slots "EA (letter alef) and "B5 (μ) by saying:

```
\DeclareInputText{224}{\alef}
\DeclareInputMath{181}{\mu}
```

Note that the *commands* should be robust, and should not be dependent on the output encoding. The same *slot* should not have both a text and a math declaration for it. (This restriction may be removed in future releases of inputenc).

The .def file may also define commands using the declarations:

`\providecommand` or `\ProvideTextCommandDefault`. For example, 8859-8.def defines:

```
\ProvideTextCommandDefault{\textonequarter}{\ensuremath{\frac{1}{4}}}
\DeclareInputText{188}{\textonequarter}
```

The use of the ‘provide’ forms here will ensure that a better definition will not be overwritten; their use is recommended since, in general, the best definition depends on the fonts available.

See the documentation in `inputenc.dtx` for details of how to declare input definitions for various encodings.

53 Hebrew font encodings

The file `hebrew.fdd`⁵¹ contains the Local Hebrew Encoding (LHE) definition, the external font information needed to use the Hebrew 7-bit fonts (old code fonts) and `hebrewfont` package that provides Hebrew font switching commands.

Using this file as an input, `lheenc.def` encoding definition file, all .fd files (font definition files) and font switching package for available Hebrew fonts are generated. We chose to use 7-bit encoding as default font encoding, because:

1. There are many 7-bit encoded Hebrew fonts available, more then for any other encoding.
2. Available T_EX Hebrew fonts do not include latin alphabet, and we can safely map Hebrew glyphs to the ASCII positions (0 – 127).

Current definition of the LHE encoding supports only Hebrew letters (`\alef`–`\tav`), but not Hebrew points, such as `\dagesh`, `\qamats`, `\patah`, `\shindot`, etc. We are working now on such addition.

54 Hebrew in L^AT_EX 2.09 compatibility mode

`\documentstyle` command in the preamble of L^AT_EX document indicates that it is a L^AT_EX 2.09 document, and should be processed in *compatibility mode*. In such documents, one of the following three Hebrew style options can be included:

1. `hebrew_newcode` indicates that document will use UNIX ISO 8859-8 or Windows cp1255 input encoding, i.e. *Alef* letter will be represented as 224.
2. `hebrew_p` indicates that document is encoded with IBM PC cp862 encoding, i.e. *Alef* letter will be represented as 128.

⁵¹The files described in this section have version number v1.0b and were last revised on 2001/02/16.

3. `hebrew_oldcode` indicates that document uses old 7-bit encoding, as defined in Israeli Standard 960, i.e. *Alef* is character number 96.

Note, that other hebrew-related styles, such as `hebcsl` can be included *after* the above-named Hebrew style option, for example:

```
\documentstyle[12pt,hebrew_p,hebcsl]{report}.
```

Any Hebrew document which compiled under \LaTeX 2.09 should compile under compatibility mode, unless it uses low-level commands such as `\tenrm`.

54.1 The DOCSTRIP modules

The following modules are used in the implementation to direct DOCSTRIP in generating the external files:

<code>newcode</code>	produce <code>hebrew_newcode.sty</code>
<code>pccode</code>	produce <code>hebrew_p.sty</code>
<code>oldcode</code>	produce <code>hebrew_oldcode.sty</code>

55 Writing Hebrew/English thesis with \LaTeX 2 ϵ

The file `hebtech.dtx`⁵² can be useful for writing a thesis or dissertation in both Hebrew and English languages, at the Technion — Israel Institute of Technology. This package is an experimental meant to answer to the demands of the Graduate School of the Technion, but can be easily adjusted (I hope) for writing theses at other academic institutions at Israel.

Original version of this style (called `hebtech.sty`) for \LaTeX 2.09 was written in April 1994 by Irina Abramovici from the Taub Computer Center, Technion and was updated for \LaTeX 2 ϵ with `babel` support by Boris Lavva. The thesis class can be used for English documents with a number of Hebrew parts, and for Hebrew documents with a number of English parts. In addition, any other languages supported by `babel` can be used freely in thesis.

56 The Bahasa language

The file `bahasa.dtx`⁵³ 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.

⁵²The Hebrew/English thesis class for \LaTeX 2 ϵ described in this section have version number v.1.0a and were last revised on 1997/12/09.

⁵³The file described in this section has version number v1.0h and was last revised on 2001/01/30.