1 The Latin language

The file `latin.dtx`[1] 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.

Furthermore a third typesetting style `withprosodicmarks` is defined in order to use special shortcuts for inserting breves and macrons when typesetting grammars, dictionaries, teaching texts, and the like, where prosodic marks are important for the complete information on the words or the verses. The shortcuts, listed in table 1 and described in section 2, may interfere with other packages; therefore by default this third style is off and no interference is introduced. If this third style is used and interference is experienced, there are special commands for turning on and off the specific short hand commands of this style.

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 if 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 1; all of them are supposed to work with both spelling styles, except where the opposite is explicitly stated.

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

```
1 ⟨*code⟩
2 \LdfInit{latin}{captionslatin}
```

[1] The file described in this section has version number v2.0l and was last revised on 2008/07/06. The original author is Claudio Beccari with contributions by Krzysztof Konrad Żelechowski, (kkz@alfa.mimuw.edu.pl)

`^i`	inserts the breve accent as ĭ; valid also for the other lowercase vowels, but it does not operate on the medieval ligatures æ and œ.
`=a`	inserts the macron accent as ā; valid also for the other lowercase vowels, but it does not operate on the medieval ligatures æ and œ.
`"`	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).
`"|`	same as above, but operates also when the next character is a medieval ligature or an accented letter.

Table 1: Shortcuts defined for the Latin language. The characters `^` and `=` are active only when the language attribute `withprosodicmarks` has been declared, otherwise they are disabled; see section 2 for more details.

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

```
\ifx\l@latin\@undefined
    \@nopatterns{Latin}
    \adddialect\l@latin0\fi
```

Now we declare the `medieval` language attribute.

```
\bbl@declare@ttribute{latin}{medieval}{%
  \addto\captionslatin{\def\prefacename{Pr{\ae}fatio}}%
  \def\november{Nouembris}%
  \expandafter\addto\expandafter\extraslatin
  \expandafter{\extrasmedievallatin}%
  }
```

The third typesetting style `withprosodicmarks` is defined here

```
\bbl@declare@ttribute{latin}{withprosodicmarks}{%
  \expandafter\addto\expandafter\extraslatin
  \expandafter{\extraswithprosodicmarks}%
  }
```

It must be remembered that the `medieval` and the `withprosodicmarks` styles may be used together.

The next step consists of defining commands to switch to (and from) the Latin language[2].

`\captionslatin` The macro `\captionslatin` defines all strings used in the four standard document classes provided with LaTeX.

```
\@namedef{captionslatin}{%
  \def\prefacename{Praefatio}%
```

[2]Most of these names were kindly suggested by Raffaella Tabacco.

```
\def\refname{Conspectus librorum}%
\def\abstractname{Summarium}%
\def\bibname{Conspectus librorum}%
\def\chaptername{Caput}%
\def\appendixname{Additamentum}%
\def\contentsname{Index}%
\def\listfigurename{Conspectus descriptionum}%
\def\listtablename{Conspectus tabularum}%
\def\indexname{Index rerum notabilium}%
\def\figurename{Descriptio}%
\def\tablename{Tabula}%
\def\partname{Pars}%
\def\enclname{Adduntur}%   Or " Additur" ? Or simply Add.?
\def\ccname{Exemplar}%     Use the recipient's dative
\def\headtoname{\ignorespaces}% Use the recipient's dative
\def\pagename{Charta}%
\def\seename{cfr.}%
\def\alsoname{cfr.}% R.Tabacco never saw "cfr. atque" or similar forms
\def\proofname{Demonstratio}%
\def\glossaryname{Glossarium}%
}
```

In the above definitions there are some points that might change in the future or that require a minimum of attention from the typesetter.

1. the `\enclname` is translated by a passive verb, that literally means "(they) are being added"; if just one enclosure is joined to the document, the plural passive is not suited any more; nevertheless a generic plural passive might be incorrect but suited for most circumstances. On the opposite "Additur", the corresponding singular passive, might be more correct with one enclosure and less suited in general: what about the abbreviation "Add." that works in both cases, but certainly is less elegant?

2. The `\headtoname` is empty and gobbles the possible following space; in practice the typesetter should use the dative of the recipient's name; since nowadays not all such names can be translated into Latin, they might result indeclinable. The clever use of an appellative by the typesetter such as "Domino" or "Dominae" might solve the problem, but the header might get too impressive. The typesetter must make a decision on his own.

3. The same holds true for the copy recipient's name in the "Cc" field of `\ccname`.

`\datelatin` The macro `\datelatin` redefines the command `\today` to produce Latin dates; the choice of faked small caps Latin numerals is arbitrary and may be changed in the future. For medieval latin the spelling of 'Novembris' should be *Nouembris*. This is taken care of by using a control sequence which can be redefined when the attribute 'medieval' is selected.

```
\def\datelatin{%
```

```
\def\november{Novembris}%
\def\today{%
  {\check@mathfonts\fontsize\sf@size\z@\math@fontsfalse\selectfont
    \uppercase\expandafter{\romannumeral\day}}~\ifcase\month\or
  Ianuarii\or Februarii\or Martii\or Aprilis\or Maii\or Iunii\or
  Iulii\or Augusti\or Septembris\or Octobris\or \november\or
  Decembris\fi
  \space{\uppercase\expandafter{\romannumeral\year}}}}
```

`\romandate` Thomas Martin Widmann (`viralbus@daimi.au.dk`) developed a macro originally named `\latindate` (but to be renamed `\romandate` so as not to conflict with the standard babel conventions) that should compute and translate the current date into a date *ab urbe condita* with days numbered according to the kalendae and idus; for the moment this is a placeholder for Thomas' macro, waiting for a self standing one that keeps local all the intermediate data, counters, etc. If he succeeds, here is the place to add his macro.

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

```
\providehyphenmins{\CurrentOption}{\tw@\tw@}
```

`\extraslatin` Lower the chance that clubs or widows occur.

`\noextraslatin`

```
\addto\extraslatin{%
  \babel@savevariable\clubpenalty
  \babel@savevariable\@clubpenalty
  \babel@savevariable\widowpenalty
  \clubpenalty3000\@clubpenalty3000\widowpenalty3000}
```

Never ever break a word between the last two lines of a paragraph in latin texts.

```
\addto\extraslatin{%
  \babel@savevariable\finalhyphendemerits
  \finalhyphendemerits50000000}
```

With medieval Latin we need the suitable correspondence between upper case V and lower case u, since in that spelling there is only one sign, and the u shape is the (uncial) version of the capital V. Everything else is identical with Latin.

```
\addto\extrasmedievallatin{%
  \babel@savevariable{\lccode`\V}%
  \babel@savevariable{\uccode`\u}%
  \lccode`\V=`\u \uccode`\u=`\V}
```

`\SetLatinLigatures` We need also the lccodes for æ and œ; since they occupy different positions in the OT1 TEX-fontencoding compared to the T1 one, we must save the lc- and the uccodes for both encodings, but we specify the new lc- and uccodes separately as it appears natural not to change encoding while typesetting the same language. The encoding is assumed to be set before starting to use the Latin language, so that if Latin is the default language, the font encoding must be chosen before requiring the babel package with the `latin` option, in any case before any `\selectlanguage` or `\foreignlanguage` command.

All this fuss is made in order to allow the use of the medieval ligatures æ and œ while typesetting with the medieval spelling; I have my doubts that the medieval spelling should be used at all in modern books, reports, and the like; the uncial 'u' shape of the lower case 'v' and the above ligatures were fancy styles of the copyists who were able to write faster with those rounded glyphs; with typesetting there is no question of handling a quill penn... Since my (CB) opinion may be wrong, I managed to set up the instruments and it is up to the typesetter to use them or not.

```
\addto\extrasmedievallatin{%
  \babel@savevariable{\lccode`\^^e6}% T1   \ae
  \babel@savevariable{\uccode`\^^e6}% T1   \ae
  \babel@savevariable{\lccode`\^^c6}% T1   \AE
  \babel@savevariable{\lccode`\^^f7}% T1   \oe
  \babel@savevariable{\uccode`\^^f7}% T1   \OE
  \babel@savevariable{\lccode`\^^d7}% T1   \OE
  \babel@savevariable{\lccode`\^^1a}% OT1  \ae
  \babel@savevariable{\uccode`\^^1a}% OT1  \ae
  \babel@savevariable{\lccode`\^^1d}% OT1  \AE
  \babel@savevariable{\lccode`\^^1b}% OT1  \oe
  \babel@savevariable{\uccode`\^^1b}% OT1  \OE
  \babel@savevariable{\lccode`\^^1e}% OT1  \OE
  \SetLatinLigatures}
\providecommand\SetLatinLigatures{%
  \def\@tempA{T1}\ifx\@tempA\f@encoding
    \catcode`\^^e6=11 \lccode`\^^e6=`\^^e6 \uccode`\^^e6=`\^^c6 % \ae
    \catcode`\^^c6=11 \lccode`\^^c6=`\^^e6 % \AE
    \catcode`\^^f7=11 \lccode`\^^f7=`\^^f7 \uccode`\^^f7=`\^^d7 % \oe
    \catcode`\^^d7=11 \lccode`\^^d7=`\^^f7 % \OE
  \else
    \catcode`\^^1a=11 \lccode`\^^1a=`\^^1a \uccode`\^^1a=`\^^1d % \ae
    \catcode`\^^1d=11 \lccode`\^^1d=`\^^1a % \AE (^^])
    \catcode`\^^1b=11 \lccode`\^^1b=`\^^1b \uccode`\^^1b=`\^^1e % \oe
    \catcode`\^^1e=11 \lccode`\^^1e=`\^^1b % \OE (^^^)
  \fi
  \let\@tempA\@undefined
  }
```

With the above definitions we are sure that `\MakeUppercase` works properly and `\MakeUppercase{C{\ae}sar}` correctly 'yields 'CÆSAR"; correspondingly `\MakeUppercase{Heluetia}` correctly yields "HELVETIA".

2 Latin shortcuts

For writing dictionaries or didactic texts (in modern spelling only) we defined a third language attribute, or a third typesetting style, a couple of other active characters are defined: `^` for marking a vowel with the breve sign, and `=` for marking a vowel with the macro sign. Please take notice that neither the OT1 font encoding,

nor the T1 one for most vowels, contain directly the marked vowels, therefore hyphenation of words containing these "accents" may become problematic; for this reason the above active characters not only introduce the required accent, but also an unbreakable zero skip that in practice does not introduce a discretionary break, but allows breaks in the rest of the word.

It must be remarked that the active characters ^ and = may have other meanings in other contexts. For example the equals sign is used by the graphic extensions for specifying keyword options for handling the graphic elements to be included in the document. At the same time, as mentioned in the previous paragraph, diacritical marking in Latin is used only for typesetting certain kind of documents, such as grammars and dictionaries. It is reasonable that the breve and macron active characters are switched on and off at will, and in particular that they are off by default if the attribute `withprosodicmarks` has not been set.

`\ProsodicMarksOn` `\ProsodicMarksOff`

We begin by adding to the normal typesetting style the definitions of the new commands `\ProsodicMarksOn` and `\ProsodicMarksOff` that should produce error messages when the third style is not declared:

```
\addto\extraslatin{\def\ProsodicMarksOn{%
    \GenericError{(latin)\@spaces\@spaces\@spaces\@spaces}%
            {Latin language error: \string\ProsodicMarksOn\space
            is defined by setting the\MessageBreak
            language attribute to 'withprosodicmarks'\MessageBreak
            If you continue you are likely to encounter\MessageBreak
            fatal errors that I can't recover}%
            {See the Latin language description in the babel
            documentation for explanation}{\@ehd}}}
\addto\extraslatin{\let\ProsodicMarksOff\relax}
```

Then we temporarily set the caret and the equals sign to active characters so that they can receive their definitions. But first we store their current category codes to restore them later on.

```
\@tempcnta=\catcode`\=
\@tempcntb=\catcode`\^
\catcode`\= \active
\catcode`\^ \active
```

Now we can add the necessary declarations to the macros that are being activated when the Latin language and its typesetting styles are declared:

```
\addto\extraslatin{\languageshorthands{latin}}%
\addto\extraswithprosodicmarks{\bbl@activate{^}}%
\addto\extraswithprosodicmarks{\bbl@activate{=}}%
\addto\noextraswithprosodicmarks{\bbl@deactivate{^}}%
\addto\noextraswithprosodicmarks{\bbl@deactivate{=}}%
\addto\extraswithprosodicmarks{\ProsodicMarks}
```

`\ProsodicMarks`

Next we define the defining macro for the active characters

```
\def\ProsodicMarks{%
  \def\ProsodicMarksOn{\catcode`\^ \active\catcode`\= \active}%
  \def\ProsodicMarksOff{\catcode`\^ 7\catcode`\= 12\relax}%
```

Notice that with the above redefinitions of the commands `\ProsodicMarksOn` and `\ProsodicMarksOff`, the operation of the newly defined shortcuts may be switched on and off at will, so that even if a picture has to be inserted in the document by means of the commands and keyword options of the `graphicx` package, it suffices to switch them off before invoking the picture including command, and switched on again afterwards; or, even better, since the picture very likely is being inserted within a `figure` environment, it suffices to switch them off within the environment, being conscious that their deactivation remains local to the environment.

```
  \initiate@active@char{^}%
  \initiate@active@char{=}%
  \declare@shorthand{latin}{^a}{%
    \textormath{\u{a}\bbl@allowhyphens}{\hat{a}}}%
  \declare@shorthand{latin}{^e}{%
    \textormath{\u{e}\bbl@allowhyphens}{\hat{e}}}%
  \declare@shorthand{latin}{^i}{%
    \textormath{\u{\i}\bbl@allowhyphens}{\hat{\imath}}}%
  \declare@shorthand{latin}{^o}{%
    \textormath{\u{o}\bbl@allowhyphens}{\hat{o}}}%
  \declare@shorthand{latin}{^u}{%
    \textormath{\u{u}\bbl@allowhyphens}{\hat{u}}}%
%
  \declare@shorthand{latin}{=a}{%
    \textormath{\={a}\bbl@allowhyphens}{\bar{a}}}%
  \declare@shorthand{latin}{=e}{%
    \textormath{\={e}\bbl@allowhyphens}{\bar{e}}}%
  \declare@shorthand{latin}{=i}{%
    \textormath{\={\i}\bbl@allowhyphens}{\bar{\imath}}}%
  \declare@shorthand{latin}{=o}{%
    \textormath{\={o}\bbl@allowhyphens}{\bar{o}}}%
  \declare@shorthand{latin}{=u}{%
    \textormath{\={u}\bbl@allowhyphens}{\bar{u}}}%
}
```

Notice that the short hand definitions are given only for lower case letters; it would not be difficult to extend the set of definitions to upper case letters, but it appears of very little use in view of the kind of documents where prosodic marks are supposed to be used. Nevertheless in those rare cases when it's required to set some uppercase letters with their prosodic marks, it is always possible to use the standard LaTeX commands such as `\u{I}` for typesetting Ĭ, or `\={A}` for typesetting Ā.

Finally we restore the caret and equals sign initial default category codes.

```
\catcode`\= \@tempcnta
\catcode`\^ \@tempcntb
```

so as to avoid conflicts with other packages or other babel options.

`\LatinMarksOn` `\LatinMarksOff`

We define two new commands so as to switch on and off the breve and macron shortcuts.

```
\addto\extraswithprosodicmarks{\let\LatinMarksOn\ProsodicMarksOn}
```

```
\addto\extraswithprosodicmarks{\let\LatinMarksOff\ProsodicMarksOff}
```

It must be understood that by using the above prosodic marks, line breaking is somewhat impeached; since such prosodic marks are used almost exclusively in dictionaries, grammars, and poems (only in school textbooks), this shouldn't be of any importance for what concerns the quality of typesetting.

3 Etymological hyphenation

In order to deal in a clean way with prefixes and compound words to be divided etymologically, the active character " is given a special definition so as to behave as a discretionary break with hyphenation allowed after it. Most of the code for dealing with the active " is already contained in the core of babel, but we are going to use it as a single character shorthand for Latin.

```
\initiate@active@char{"}%
\addto\extraslatin{\bbl@activate{"}%
}
```

A temporary macro is defined so as to take different actions in math mode and text mode: specifically in the former case the macro inserts a double quote as it should in math mode, otherwise another delayed macro comes into action.

```
\declare@shorthand{latin}{"}{%
  \ifmmode
    \def\lt@@next{''}%
  \else
    \def\lt@@next{\futurelet\lt@temp\lt@cwm}%
  \fi
  \lt@@next
}%
```

In text mode the `\lt@next` control sequence is such that upon its execution a temporary variable `\lt@temp` is made equivalent to the next token in the input list without actually removing it. Such temporary token is then tested by the macro `\lt@cwm` and if it is found to be a letter token, then it introduces a compound word separator control sequence `\lt@allowhyphens` whose expansion introduces a discretionary hyphen and an unbreakable space; in case the token is not a letter, the token is tested again to find if it is the character `|`, in which case it is gobbled and a discretionary break is introduced.

```
\def\lt@allowhyphens{\nobreak\discretionary{-}{}{}\nobreak\hskip\z@skip}
\newcommand*{\lt@cwm}{\let\lt@n@xt\relax
  \ifcat\noexpand\lt@temp a%
    \let\lt@n@xt\lt@allowhyphens
  \else
    \if\noexpand\lt@temp\string|%
        \def\lt@n@xt{\lt@allowhyphens\@gobble}%
    \fi
  \fi
  \lt@n@xt}%
```

Attention: the category code comparison does not work if the temporary control sequence `\lt@temp` has been let equal to an implicit character, such as `\ae`; therefore this etymological hyphenation facility does not work with medieval Latin spelling when `"` immediately precedes a ligature. In order to overcome this drawback the shorthand `"|` may be used in such cases; it behaves exactly as `"`, but it does not test the implicit character control sequence. An input such as `super"|{\ae}quitas`[3] gets hyphenated as `su-per-æqui-tas` instead of `su-pe-ræ-qui-tas`.

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

```
\ldf@finish{latin}
⟨/code⟩
```

[3]This word does not exist in "regular" Latin, and it is used just as an example.