

Babel support for the Greek language

Apostolos Syropoulos, Günter Milde

December 3, 2013

Babel-greek is a contributed package providing support for the Greek language and script via the [Babel](#) system. See the [README](#) file for an overview of the `babel-greek` package and links to requirements and related packages.

The file `babel-greek.dtx`¹ defines all the language definition macros for the Greek language, i.e., as it used today with only one accent, and the attribute `polutoniko` for typesetting greek text with all accents (classical and polytonic Greek).

1 Requirements

Typesetting Greek with Babel requires (of course) the [Babel core](#), support for Greek font encodings ([greek-fontenc](#)) and a font supporting the Greek script.

The [CB Greek fonts](#) created by CLAUDIO BECCARI² are a complete set of 8-bit TeX fonts matching KNUTH's Computer Modern. The package `cbfonts-fd` sets them up as Greek substitute for the Computer Modern and Latin Modern font families. The `substitutefont` package can be used to set up Greek supplements for other font families (like Times or Palatino).

The [Latin Modern](#) OpenType font that is the default font for XeTeX/LuaTeX does not support the Greek script. The user needs to set up an alternative font like Linux Libertine or DejaVu with [fontspec](#).

2 Usage

To activate Greek support with babel, specify the option `greek`, either as global option or as option to the `babel` package. Remember, that the *last* language option determines the document language, e.g.

```
\usepackage[greek,english]{babel}
```

¹The file described in this section has version number v1.8a and was last revised on 2013/12/03. 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.

²Apostolos Syropoulos wishes to thank Claudio Beccari for his patience, collaboration, comments and suggestions.

activates support for Greek in an English document while

```
\usepackage[english,greek]{babel}
\languageattribute{greek}{polutoniko}
```

sets the document language to Greek with “polytonic” spelling of auto-generated strings.

```
\selectlanguage
\foreignlanguage
```

The Babel core provides two commands to switch the active language: The declaration `\selectlanguage{greek}` switches to the Greek languages. The macro `\foreignlanguage{greek}{<some text>}` sets its second argument in the Greek language. This is intended for short text parts. For details see the [Babel](#) documentation.

For backwards compatibility, “polytonic” spelling can also be selected via the dummy language `polutonikogreek`. In this case, the language name `polutonikogreek` must also be used for language switches with `\selectlanguage` or `\foreignlanguage`. However, it is not possible to use both options, `greek` and `polutonikogreek` in one document.

2.1 Input of Greek text

There are several alternatives for input of Greek text.

- With the packages `inputenc` and [greek-inputenc](#), literal Greek characters can be input using the utf-8, iso-8859-7, or macgreek encoding.
Literal input using the utf-8 encoding is also the standard input method if compiling with the XeTeX and LuaTeX engines.
- The Latin transliteration defined by the LGR font encoding is explained in the file [usage.pdf](#).
- The package [greek-fontenc](#) defines *LaTeX internal character representation* (LICR) macros for Greek letters and text symbols. It is required by `babel-greek`, so the `\textAlpha` ... `\textomega` macros are a save but cumbersome method to input Greek characters.
- The `alphabetalpha` package bundled with [greek-fontenc](#) makes the short macro names `\Alpha` ... `\omega` available in text mode, too.

The [greek-fontenc](#) bundle has details and examples for all these input methods.

2.2 Greek vs. Latin script

When switching the language to Greek, `babel-greek` ensures that the Greek script is supported. With 8-bit LaTeX, Greek text fonts use a non-standard font encoding³ (LGR) that misses Latin letters. Latin characters in the source are instead interpreted as a transcription for Greek symbols. The following macros allow the use of Greek vs. Latin script without changing the active language. The

```
\greekscript
\greektext
```

³LaTeX font encodings are described in the *LaTeX font guide* (fntguide.pdf).

*TextCommand*⁴ `\greekscript` switches to a font encoding supporting the Greek script. The declaration `\greektext` always switches the font encoding to LGR. Both declarations do not change the active language. `\latintext` (defined by the Babel core) can be used to switch back to an encoding supporting the Latin script. The macros `\ensuregreek` and `\textgreek` take one argument which is typeset using a font encoding supporting the Greek script. While `\ensuregreek` only switches the font encoding if required, `\textgreek` always uses the LGR font encoding. The Babel core defines a corresponding `\textlatin` macro.

3 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.
- 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

⁴For a discussion of TextCommands, see the *LaTeX font guide* (fntguide.pdf), too.

`\greeknumeral` makes it possible to typeset Greek numerals. There is also an “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`.

4 Implementation

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\CurrentOption{captions\CurrentOption}
```

When the option `polutonikogreek` was used, redefine `\CurrentOption` to prevent problems later on.

```
3 \gdef\CurrentOption{greek}%
```

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

```
4 \ifx\l@greek\@undefined
5   \@nopatterns{greek}
6   \adddialect\l@greek0\fi
```

Now we declare the `polutoniko` language attribute.

```
7 \bbl@declare@ttribute{greek}{polutoniko}{%
```

This code adds the expansion of `\extraspolutonikogreek` to `\extrasgreek` and changes the definition of `\today` for Greek to produce polytonic month names.

```
8   \expandafter\addto\expandafter\extrasgreek
9   \expandafter{\extraspolutonikogreek}%
10  \let\captionsgreek\captionspolutonikogreek
11  \let\gr@month\gr@c@month
```

We need to take some extra precautions in order not to break older documents which still use the old `polutonikogreek` option.

```
12  \let\l@polutonikogreek\l@greek
13  \let\datepolutonikogreek\dategreek
14  \let\extraspolutonikogreek\extrasgreek
15  \let\noextraspolutonikogreek\noextrasgreek
16  }
```

5 Font setup

Typesetting Greek texts requires a font with Greek characters:

For 8-bit LaTeX, this package uses fonts with the [LGR font encoding](#). With the XeTeX or LuaTeX engines and Unicode fonts, the user must ensure that the selected font contains the required glyphs. LGR-encoded fonts can also be used

alongside Unicode fonts with XeTeX/LuaTeX to enable the input of Greek letters via the Latin transcription provided by this font encoding.

`\greekfontencoding` We test for available font encodings and set `\greekfontencoding` and the internal macro `\bbl@greek@fontencdef` with the name of the font definition file with Greek LICR macros. The fallback is LGR.

```

17 \@ifl@aded{def}{eu1enc}{
18   \providecommand*{\greekfontencoding}{EU1}
19   \providecommand*{\bbl@greek@fontencdef}{greek-euenc}
20   \renewcommand*{\LastDeclaredEncoding}{EU1}
21 }{}
22 \@ifl@aded{def}{eu2enc}{
23   \providecommand*{\greekfontencoding}{EU2}
24   \providecommand*{\bbl@greek@fontencdef}{greek-euenc}
25   \renewcommand*{\LastDeclaredEncoding}{EU2}
26 }{}
27 \providecommand*{\greekfontencoding}{LGR}
28 \providecommand*{\bbl@greek@fontencdef}{lgrenc}

```

Ensure that the Greek LICR macros are available:

```

29 \@ifl@aded{def}{\bbl@greek@fontencdef}{}{
30   \InputIfFileExists{\bbl@greek@fontencdef .def}{}{
31     \errhelp{I can't find the \bbl@greek@fontencdef .def file
32       for the Greek fonts}%
33     \errmessage{Font support for the Greek script missing.^^J
34       babel-greek can't typeset Greek.^^J
35       Install the "greek-fontenc" package^^J
36       or use XeTeX/LuaTeX with polyglossia.}%
37     \@@end
38   }
39 }

```

`\greekscript` The TextCommand `\greekscript` is a declaration that switches the font encoding to `\greekfontencoding` if the current font encoding does not provide a (typically empty) local variant.

```

40 \ProvideTextCommandDefault{\greekscript}{%
41   \fontencoding{\greekfontencoding}\selectfont
42   \def\encodingdefault{\greekfontencoding}}

```

`\ensuregreek` The TextCommand `\ensuregreek` sets its argument in `\greekfontencoding` if the current font encoding does not provide a (typically empty) local variant.

```

43 \ProvideTextCommandDefault{\ensuregreek}[1]{%
44   \leavevmode{\greekscript #1}}

```

`\latinencoding` The babel core defines the macro `\latinencoding` and sets it to either OT1 or (if available) T1. We extend the choice with the Unicode encodings EU1 and EU2 so that `\latintext` under XeTeX/LuaTeX does not prevent the use of Unicode fonts.

```

45 \AtBeginDocument{
46   \@ifl@aded{def}{eu1enc}{\xdef\latinencoding{EU1}}{}% XeTeX

```

```

47 \ifl@aded{def}{eu2enc}{\xdef\latinencoding{EU2}}{}% LuaTeX
48 }

```

5.1 Fixes for the LGR font encoding

```

49 \ifl@aded{def}{lgrenc}{%

```

We redefine a few commands in the LGR encoding to work around problems because LGR is no *standard text encoding* but has Greek letters in the places of Latin ones.

```

50 \ProvideTextCommand{\textcopyright}{LGR}{\textlatin{\textcopyright}}
51 \ProvideTextCommand{\textregistered}{LGR}{\textlatin{\textregistered}}
52 \ProvideTextCommand{\texttrademark}{LGR}{\textlatin{\texttrademark}}
53 \let\ltx@amp\&
54 \DeclareTextCommandDefault{\&}{\ltx@amp}
55 \DeclareTextCommand{\&}{LGR}{\textlatin{\ltx@amp}}
56 \ProvideTextCommand{\SS}{LGR}{\textlatin{\SS}}

```

`\greek@roman` To prevent roman numerals being typeset in greek letters we need to adopt the internal L^AT_EX commands `\@roman` and `\@Roman`. *Up to version 1.5, the definition caused errors where `\roman` ends up in a situation where the argument needs to be expanded (e.g. Roman page numbers in the ToC).*

```

57 \def\@roman#1{\expandafter\textlatin\expandafter{\romannumeral#1}}
58 \def\@Roman#1{\expandafter\textlatin\expandafter{%
59 \expandafter\@slowromancap\romannumeral#1@}}

```

Now we define two commands that switch to the LGR font encoding. The Babel core defines corresponding commands to switch to a Latin font encoding.

`\greektext` The declaration `\greektext` switches to LGR. For shorter pieces of text the `\textgreek` macro should be used.

```

60 \DeclareRobustCommand{\greektext}{%
61 \fontencoding{LGR}\selectfont
62 \def\encodingdefault{LGR}}

```

`\textgreek` This command takes an argument which is then typeset using the LGR font encoding. In order to avoid many encoding switches it operates in a local scope.

```

63 \DeclareRobustCommand{\textgreek}[1]{\leavevmode{\greektext #1}}

```

`\textol` The [CB Greek fonts](#) contain an outline family. In order to make it available, we define the command `\textol`. (This font-specific macro does not fit in a language definition file and is only kept for backwards compatibility.)

```

64 \def\outlfamily{\usefont{LGR}{cmro}{m}{n}}
65 \DeclareTextFontCommand{\textol}{\outlfamily}
66 }{} % End of LGR fixes.

```

6 Definitions for the Greek language

The next step consists in defining commands to switch to (and from) the Greek language.

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

```

67 % Yannis Haralambous has suggested this value
68 \providehyphenmins{\CurrentOption}{\@ne\@ne}

```

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

```

69 \addto\captionsgreek{%
70   \def\prefacename{\textPi\textrho\acconos\textomicron\textlambda
71     \textomicron\textgamma\textomicron\textfinalsigma}%
72   \def\refname{\textAlpha\textnu\textalpha
73     \textphi\textomicron\textrho\acconos\textepsilon\textfinalsigma}%
74   \def\abstractname{\textPi\textepsilon\textrho\acconos\textiota
75     \textlambda\texteta\textpsi\texteta}%
76   \def\bibname{\textBeta\textiota\textbeta\textlambda\textiota\textomicron
77     \textgamma\textrho\textalpha\textphi\acconos\textiota\textalpha}%
78   \def\chaptername{\textKappa\textepsilon\textphi\acconos\textalpha
79     \textlambda\textalpha\textiota\textomicron}%
80   \def\appendixname{\textPi\textalpha\textrho\acconos\textalpha\textrho
81     \texttau\texteta\textmu\textalpha}%
82   \def\contentsname{\textPi\textepsilon\textrho\textiota
83     \textepsilon\textchi\acconos\textomicron\textmu\textepsilon
84     \textnu\textalpha}%
85   \def\listfigurename{\textKappa\textalpha\texttau\acconos\textalpha
86     \textlambda\textomicron\textgamma\textomicron\textfinalsigma}%
87   \def\listtablename{\textKappa\textalpha\texttau\acconos\textalpha
88     \textlambda\textomicron\textgamma\textomicron\textfinalsigma}%
89   \def\indexname{\textEpsilon\textupsilon\textrho\textepsilon
90     \texttau\acconos\texteta\textrho\textiota\textomicron}%
91   \def\figurename{\textSigma\textchi\acconos\texteta\textmu\textalpha}%
92   \def\tablename{\textPi\acconos\textiota\textnu\textalpha
93     \textkappa\textalpha\textfinalsigma}%
94   \def\partname{\textMu\acconos\textepsilon\textrho\textomicron
95     \textfinalsigma}%
96   \def\enclname{\textSigma\textupsilon\textnu\texteta\textmu
97     \textmu\acconos\textepsilon\textnu\textalpha}%
98   \def\ccname{\textKappa\textomicron\textiota\textnu\textomicron
99     \textpi\textomicron\acconos\textiota\texteta\textsigma\texteta}%
100   \def\headtoname{\textPi\textrho\textomicron\textfinalsigma}%
101   \def\pagename{\textSigma\textepsilon\textlambda\acconos\textiota
102     \textdelta\textalpha}%

```

```

106 \def\seename{\textbeta\textlambda\acctonos\textepsilon\textpi\textepsilon}%
107 \def\alsoname{\textbeta\textlambda\acctonos\textepsilon\textpi\textepsilon{}
108 \textepsilon\textpi\acctonos\textiota\textsigma\texteta\textfinalsigma}%
109 \def\proofname{\textAlpha\textpi\acctonos\textomicron
110 \textdelta\textepsilon\textiota\textxi\texteta}%
111 \def\glossaryname{\textGamma\textlambda\textomega\textsigma
112 \textsigma\acctonos\textalpha\textrho\textiota}%
113 }

```

`\captionspolutonikogreek` For texts written in the *πολυτονικό* (polytonic greek) the translations are the same as above, but some words are spelled differently. For now we just add extra definitions to `\captionsgreek` in order to override the earlier definitions.

```

114 \let\captionspolutonikogreek\captionsgreek
115 \addto\captionspolutonikogreek{%
116 \def\refname{>\textAlpha\textnu\textalpha
117 \textphi\textomicron\textrho\accvaria\textepsilon\textfinalsigma}%
118 \def\indexname{\textEpsilon<\textupsilon\textrho\textepsilon
119 \texttau\acctonos\texteta\textrho\textiota\textomicron}%
120 \def\figurename{\textSigma\textchi\accperispomeni\texteta\textmu\textalpha}%
121 \def\headtoname{\textPi\textrho\accvaria\textomicron\textfinalsigma}%
122 \def\alsoname{\textbeta\textlambda\acctonos\textepsilon\textpi\textepsilon{}
123 >\textepsilon\textpi\acctonos\textiota\textsigma\texteta\textfinalsigma}%
124 \def\proofname{>\textAlpha\textpi\acctonos\textomicron
125 \textdelta\textepsilon\textiota\textxi\texteta}%
126 }

```

`\gr@month` The macro `\dategreek` redefines the command `\today` to produce greek dates.
`\dategreek` The name of the month is now produced by the macro `\gr@month` since it is needed in the definition of the macro `\Grtoday`.

```

127 \def\gr@month{%
128 \ifcase\month\or
129 \textIota\textalpha\textnu\textomicron\textupsilon\textalpha
130 \textrho\acctonos\textiota\textomicron\textupsilon \or
131 \textPhi\textepsilon\textbeta\textrho\textomicron\textupsilon
132 \textalpha\textrho\acctonos\textiota\textomicron\textupsilon \or
133 \textMu\textalpha\textrho\texttau\acctonos\textiota\textomicron\textupsilon
134 \or \textAlpha\textpi\textrho\textiota\textlambda\acctonos\textiota
135 \textomicron\textupsilon \or
136 \textMu\textalpha\'"\textiota\textomicron\textupsilon \or
137 \textIota\textomicron\textupsilon\textnu\acctonos\textiota
138 \textomicron\textupsilon \or
139 \textIota\textomicron\textupsilon\textlambda\acctonos\textiota
140 \textomicron\textupsilon \or
141 \textAlpha\textupsilon\textgamma\textomicron\acctonos\textupsilon
142 \textsigma\texttau\textomicron\textupsilon \or
143 \textSigma\textepsilon\textpi\texttau\textepsilon\textmu
144 \textbeta\textrho\acctonos\textiota\textomicron\textupsilon \or
145 \textOmicron\textkappa\texttau\textomega\textbeta
146 \textrho\acctonos\textiota\textomicron\textupsilon \or

```



```

186 \addto\extrasgreek{\greekscript}
187 \addto\noextrasgreek{\latintext}
188 % \addto\noextrasgreek{\RestoreFontEncoding}

\gr@ill@value When the argument of \greeknumeral has a value outside of the acceptable
               bounds ( $0 < x < 999999$ ) a warning will be issued (and nothing will be printed).
189 \def\gr@ill@value#1{%
190   \PackageWarningNoLine{babel}{Illegal value (#1) for greeknumeral}}

\anw@true When a large number with three trailing zero's is to be printed those zeros and
\anw@false the numeric mark need to be discarded. As each 'digit' is processed by a separate
\anw@print macro and because the processing needs to be expandable we need some helper
               macros that help remember to not print the numeric mark (\anwtonos).

               The command \anw@false switches the printing of the numeric mark off by
               making \anw@print expand to nothing. The command \anw@true (re)enables the
               printing of the numeric marc. These macro's need to be robust in order to prevent
               improper expansion during writing to files or during \uppercase.
191 \DeclareRobustCommand\anw@false{%
192   \DeclareRobustCommand\anw@print{}}
193 \DeclareRobustCommand\anw@true{%
194   \DeclareRobustCommand\anw@print{\textdexiakeraia}} % \anwtonos
195 \anw@true

\greeknumeral The command \greeknumeral needs to be fully expandable in order to get the
               right information in auxiliary files. Therefore we use a big \if-construction to
               check the value of the argument and start the parsing at the right level.
196 \def\greeknumeral#1{%
               If the value is negative or zero nothing is printed and a warning is issued.
197   \ifnum#1<\@ne\space\gr@ill@value{#1}%
198   \else
199     \ifnum#1<10\expandafter\gr@num@i\number#1%
200     \else
201       \ifnum#1<100\expandafter\gr@num@ii\number#1%
202       \else
               We use the available shorthands for 1.000 (\@m) and 10.000 (\@M) to save a few
               tokens.
203         \ifnum#1<\@m\expandafter\gr@num@iii\number#1%
204         \else
205           \ifnum#1<\@M\expandafter\gr@num@iv\number#1%
206           \else
207             \ifnum#1<100000\expandafter\gr@num@v\number#1%
208             \else
209               \ifnum#1<1000000\expandafter\gr@num@vi\number#1%
210               \else
               If the value is too large, nothing is printed and a warning is issued.
211               \space\gr@ill@value{#1}%
212               \fi

```

```

213         \fi
214     \fi
215     \fi
216     \fi
217     \fi
218     \fi
219 }

```

`\Greeknatural` The command `\Greeknatural` prints uppercase greek numerals. The parsing is performed by the macro `\greeknumeral`.

```

220 \def\Greeknatural#1{%
221     \expandafter\MakeUppercase\expandafter{\greeknumeral{#1}}

```

`\greek@alph` In the previous release of this language definition the commands `\greek@aplh` and `\greek@Alph` were kept just for reasons of compatibility. Here again they become meaningful macros. They are defined in a way that even page numbering with greek numerals is possible. Since the macros `\@alph` and `\@Alph` will lose their original meaning while the Greek option is active, we must save their original value. macros `\@alph`

```

222 \let\latin@alph\@alph
223 \let\latin@Alph\@Alph

```

Then we define the Greek versions; the additional `\expandafters` are needed in order to make sure the table of contents will be correct, e.g., when we have appendixes.

```

224 \def\greek@alph#1{\expandafter\greeknumeral\expandafter{\the#1}}
225 \def\greek@Alph#1{\expandafter\Greeknatural\expandafter{\the#1}}

```

Now we can set up the switching.

```

226 \addto\extrasgreek{%
227     \let\@alph\greek@alph
228     \let\@Alph\greek@Alph}
229 \addto\noextrasgreek{%
230     \let\@alph\latin@alph
231     \let\@Alph\latin@Alph}

```

What is left now is the definition of a set of macros to produce the various digits.

`\gr@num@i` As there is no representation for 0 in this system the zeros are simply discarded.
`\gr@num@ii` When we have a large number with three *trailing* zero's also the numeric mark is discarded. Therefore these macros need to pass the information to each other about the (non-)translation of a zero.

```

232 \def\gr@num@i#1{%
233     \ifcase#1\or \textalpha\or \textbeta\or \textgamma\or \textdelta\or
234     \textepsilon\or \textsigma\or \textzeta\or \texteta\or \texttheta\fi
235     \ifnum#1=z\else\anw@true\fi\anw@print}
236 \def\gr@num@ii#1{%
237     \ifcase#1\or \textiota\or \textkappa\or \textlambda\or \textmu\or

```

```

238 \textnu\or \textxi\or \textomicron\or \textpi\or \textqoppa\fi
239 \ifnum#1=\z@\else\anw@true\fi\gr@num@i}
240 \def\gr@num@iii#1{%
241 \ifcase#1\or \textrho\or \textsigma\or \texttau\or \textupsilon\or
242 \textphi\or \textchi\or \textpsi\or \textomega\or \textsampi\fi
243 \ifnum#1=\z@\anw@false\else\anw@true\fi\gr@num@ii}

\gr@num@iv The first three ‘digits’ always have the numeric mark, except when one is discarded
\gr@num@v because it’s value is zero.
\gr@num@vi 244 \def\gr@num@iv#1{%
245 \ifnum#1=\z@\else\textaristerikeraia\fi
246 \ifcase#1\or \textalpha\or \textbeta\or \textgamma\or \textdelta\or
247 \textepsilon\or \stigma\or \textzeta\or \texteta\or \texttheta\fi
248 \gr@num@iii}
249 \def\gr@num@v#1{%
250 \ifnum#1=\z@\else\textaristerikeraia\fi
251 \ifcase#1\or \textiota\or \textkappa\or \textlambda\or \textmu\or
252 \textnu\or \textxi\or \textomicron\or \textpi\or \textqoppa\fi
253 \gr@num@iv}
254 \def\gr@num@vi#1{%
255 \textaristerikeraia
256 \ifcase#1\or \textrho\or \textsigma\or \texttau\or \textupsilon\or
257 \textphi\or \textchi\or \textpsi\or \textomega\or \textsampi\fi
258 \gr@num@v}

```

7 Character codes

Greek letters drop diacritics (except dialytika and sub-iota) in UPPERCASE. This is not cared for by the Unicode standard. The file `xgreek-fixes.def` from `polyglossia` contains the required `\lccode` and `\uccode` corrections from the `xgreek` package by Apostolos Syropoulos.

```

259 \@ifl@aded{def}{greek-euenc}{
260 \InputIfFileExists{xgreek-fixes.def}{}{\PackageWarningNoLine{babel}{%
261 Can't find the file "xgreek-fixes.def".\MessageBreak
262 For correct up/down-casing of Greek text,\MessageBreak
263 install the "polyglossia" package}%
264 }
265 \addto\extraspolutonikogreek{} % prevent command missing error.
266 }{}

```

The LGR encoded fonts that are used to typeset Greek with 8-bit TeX define ligatures for characters with diacritics. 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.

`\greek@tilde` The Greek script uses a number of characters with more than one accent. In LGR encoded fonts combined diacritics can be obtained using Knuth's ligature mechanism (see `usage.pdf`). Characters we need to have ligatures with are the tilde, the acute and grave accent characters, the rough and smooth breathings, the subscript, and the double quote character. In text input the `~` is normally used to produce an unbreakable space.

```

267 \ifl@aded{def}{\lgrenc}{%
268   \begingroup
269     \ifundefined{active@char\string!}{\catcode'\!=12\relax}
270     \catcode'\~=12
271     \lccode'\!='\~
272     \lowercase{\def\x{\endgroup
273       \def\greek@tilde{!}}\x}

```

In order to get correct hyphenation we need to set the lower case code of a number of characters. The 'v' character has a special usage for the `cb` fonts: in fact this ligature mechanism detects the end of a word and assures that a final sigma is typeset with the proper sign which is different from that of an initial or medial sigma; the 'v' after an *isolated* sigma fools the ligature mechanism in order to typeset σ in place of ς . Because of this we make sure its lowercase code is not changed. For "modern" greek we have to deal only with ' and " and so things are easy.

```

274 \addto\extrasgreek{%
275   \babel@savevariable{\lccode'v}\lccode'v='v%
276   \babel@savevariable{\lccode'\'}\lccode'\='\'%
277   \babel@savevariable{\lccode'\"}\lccode'\="=\}%
278 \addto\extrapolutonikogreek{%
279   \babel@savevariable{\lccode'\<}\lccode'\<='<%
280   \babel@savevariable{\lccode'\>}\lccode'\>='>%
281   \babel@savevariable{\lccode'\~}\lccode'\~='~%
282   \babel@savevariable{\lccode'\|}\lccode'\|='||%
283   \babel@savevariable{\lccode'\'}\lccode'\='\'%

```

And in order to get rid of all accents and breathings when a string is `\uppercased` we also change a number of uppercase codes.

```

284 \addto\extrasgreek{%
285   \babel@savevariable{\uccode'\"}\uccode'\="=\}%
286   \babel@savevariable{\uccode'\'}\uccode'\'=159} %% 159 == ^~9f
287 \addto\extrapolutonikogreek{%
288   \babel@savevariable{\uccode'\~}\uccode'\~=159%
289   \babel@savevariable{\uccode'\>}\uccode'\>=159%
290   \babel@savevariable{\uccode'\<}\uccode'\<=159%
291   \babel@savevariable{\uccode'\|}\uccode'\|=159%
292   \babel@savevariable{\uccode'\'}\uccode'\'=159}

```

For this to work we make the character `^~9f` a shorthand that expands to nothing. In order for this to work we need to make a character look like `^~9f` in $\text{T}_{\text{E}}\text{X}$'s eyes. The trick is to have another character and assign it a different lowercase code. The execute the macros needed in a `\lowercase` environment. Usually the tilde

~ character is used for such purposes. Before we do this we save it's original lowercase code to restore it once we're done.

```

293 \@tempcnta=\lccode'\~
294 \lccode'\~=159
295 \lowercase{%
296   \initiate@active@char{~}%
297   \declare@shorthand{greek}{~}{}}
298 \lccode'\~= \@tempcnta

```

Add composite commands, so that the dialytika is kept or put on the following character of a diphthong with `\MakeUppercase` (see `lgrdef.enc` from the `greek-fontenc` package for details).

```

299 \DeclareTextCompositeCommand{"}{LGR}{^^9F}{\accdialytika}
300 \DeclareTextCompositeCommand{'}{LGR}{^^9F}{\LGR@hiatus}
301 \DeclareTextCompositeCommand{\'}{LGR}{^^9F}{\LGR@hiatus}
302 \DeclareTextCompositeCommand{<}{LGR}{^^9F}{\LGR@hiatus}
303 \DeclareTextCompositeCommand{>}{LGR}{^^9F}{\LGR@hiatus}

```

We can also make the tilde character itself expand to a tilde with category code 12 to make the typing of texts easier.

```

304 \addto\extraspolutonikogreek{\languageshorthands{greek}}%
305 \declare@shorthand{greek}{~}{\greek@tilde}
306 }{} % End of LGR-specific code.

```

8 symbol name aliases

For backwards compatibility, we keep aliases for a few symbols.

```

307 \providecommand*\anwtonos{\textdexiakeraia}
308 \providecommand*\katwtonos{\textaristerikeraia}
309 \providecommand*\qoppa{\textqoppa}
310 \providecommand*\stigma{\textstigma}
311 \providecommand*\sampi{\textsampi}
312 \providecommand*\Digamma{\textDigamma}
313 \providecommand*\ddigamma{\textdigamma}
314 \providecommand*\vardigamma{\textvardigamma}
315 \providecommand*\euro{\texteuro}
316 \providecommand*\permill{\textperthousand}

```

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.

```

317 \ldf@finish{\CurrentOption}
318 </code>

```