1 Babel support for the Greek language

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

`\textgreek` `\textlatin` `\textol` 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` typests 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`.

1.1 Typing conventions

The file `usage.pdf` explains the several input conventions for Greek.

1.2 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

[1] The file described in this section has version number v1.5 and was last revised on 2013/06/21. 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.

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` `\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 "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`.

1.3 Implementation

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

```
⟨*code⟩
\LdfInit\CurrentOption{captions\CurrentOption}
```

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

```
\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.

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

Now we declare the `polutoniko` language attribute.

```
\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.

```
  \expandafter\addto\expandafter\extrasgreek
  \expandafter{\extraspolutonikogreek}%
  \let\captionsgreek\captionspolutonikogreek
  \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.

```
  \let\l@polutonikogreek\l@greek
  \let\datepolutonikogreek\dategreek
  \let\extraspolutonikogreek\extrasgreek
  \let\noextraspolutonikogreek\noextrasgreek
  }
```

Next, we check if LuaLaTeX or XeLaTeX is running. If so, we set boolean key `\if@babel@greek@unicode` to true. (The test code is taken from russian.ldf. We hope that a future version of `babel.sty` will do the check and provide a boolean.

We don't load the `ifluatex` or `ifxetex` package because `\RequirePackage` is not allowed at the stage of processing options (note that babel loads this file right when it processes its own options) but we borrow code from these packages.

The test requires the e-TeX extensions (provided by all TeX engines in contemporary TeX distributions).

```
\ifdefined\if@bbl@greek@unicode
  \PackageError{babel}{if@bbl@greek@unicode already defined.\MessageBreak
    Please contact author of greek.ldf}
  \relax
\fi
\newif\if@bbl@greek@unicode
\ifdefined\luatexversion \@bbl@greek@unicodetrue \else
\ifdefined\XeTeXrevision \@bbl@greek@unicodetrue \fi\fi
```

Typesetting Greek texts requires a font containing Greek letters. With the modern XeTeX or LuaTeX engines, the user should ensure that the choosen font contains the required glyphs. Examples for suitable fonts are the “Deja Vu” or the “Linux Libertine” OpenType fonts.

For 8-bit LaTeX, this package uses fonts with the LGR font encoding (see the ‘greek-fontenc’ package http://www.ctan.org/pkg/greek-fontenc). The `cb` fonts created by Claudio Beccari [2] are a complete set of LGR encoded Greek text fonts matching Computer Modern.

```
\if@bbl@greek@unicode
```

No font encoding switch required with Unicode fonts. Define dummy commands for `\textgreek` and `\greektext` (cf. below):

```
  \let\greektext\relax
  \let\textgreek\@firstofone
\else
  \InputIfFileExists{lgrenc.def}{}{%
    \errhelp{I can't find the lgrenc.def file for the Greek fonts}%
    \errmessage{Since I do not know the LGR font encoding^^J
      I can't typeset Greek with 8-bit TeX.^^J
      Please install the "greek-fontenc" package or use XeTeX/LuaTeX.}\@@end
   }
```

[2] Apostolos Syropoulos wishes to thank him for his patience, collaboration, comments and suggestions.

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.

```
\ProvideTextCommand{\textcopyright}{LGR}{\textlatin{\textcopyright}}
\ProvideTextCommand{\textregistered}{LGR}{\textlatin{\textregistered}}
\ProvideTextCommand{\texttrademark}{LGR}{\textlatin{\texttrademark}}
\let\ltx@amp\&
\DeclareTextCommandDefault{\&}{\ltx@amp}
\DeclareTextCommand{\&}{LGR}{\textlatin{\ltx@amp}}
\ProvideTextCommand{\SS}{LGR}{\textlatin{\SS}}
```

`\greek@roman` `\greek@Roman` To prevent roman numerals being typeset in greek letters we need to adopt the internal LaTeX commands `\@roman` and `\@Roman`. **The pre-1.5 definition caused errors where `roman` ends up in a situation where it needs to be expanded; Check whether problems persist with the AMS document classes!**

```
\let\latin@roman\@roman
\DeclareTextCommandDefault{\@roman}[1]{\latin@roman{#1}}
\DeclareTextCommand{\@roman}{LGR}[1]{\textlatin{\latin@roman{#1}}}
\let\latin@Roman\@Roman
\DeclareTextCommandDefault{\@Roman}[1]{\latin@Roman{#1}}
\DeclareTextCommand{\@Roman}{LGR}[1]{\textlatin{\latin@Roman{#1}}}
```

Later in this file, the charcter No 159 is defined as uppercase of `<`, `>`, `~`, `` ` ``, `'`, `"` and `|`. 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 the greek-fontenc package for details). As a the TextCompositeCommand mechanism compares the unexpanded literal strings, we need the literal character in the declarations, too.

```
\DeclareTextCompositeCommand{\"}{LGR}{}{\accdialytika}
\DeclareTextCompositeCommand{\'}{LGR}{}{\LGR@hiatus}
\DeclareTextCompositeCommand{\`}{LGR}{}{\LGR@hiatus}
\DeclareTextCompositeCommand{\<}{LGR}{}{\LGR@hiatus}
\DeclareTextCompositeCommand{\>}{LGR}{}{\LGR@hiatus}
```

Now we define two commands that offer the possibility to switch to the LGR font encoding. Babel defines corresponding macros to switch to a Latin font encoding.

`\greektext` The command `\greektext` will switch to a Greek font encoding. This command is a *declaration*, for shorter pieces of text the command `\textgreek` should be used.

```
\DeclareRobustCommand{\greektext}{%
  \fontencoding{LGR}\selectfont
  \def\encodingdefault{LGR}}
```

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

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

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

```
  \def\outlfamily{\usefont{LGR}{cmro}{m}{n}}
  \DeclareTextFontCommand{\textol}{\outlfamily}
\fi  % End of LGR fixes not required with XeTeX/LuaTeX.
```

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`.

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

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

```
\addto\captionsgreek{%
  \def\prefacename{\textPi\textrho\'\textomicron\textlambda\textomicron%
    \textgamma\textomicron\textfinalsigma}%
  \def\refname{\textAlpha\textnu\textalpha%
    \textphi\textomicron\textrho\'\textepsilon\textfinalsigma}%
  \def\abstractname{\textPi\textepsilon\textrho\'\textiota%
    \textlambda\texteta\textpsi\texteta}%
  \def\bibname{\textBeta\textiota\textbeta\textlambda\textiota\textomicron%
    \textgamma\textrho\textalpha\textphi\'\textiota\textalpha}%
  \def\chaptername{\textKappa\textepsilon\textphi\'\textalpha%
    \textlambda\textalpha\textiota\textomicron}%
  \def\appendixname{\textPi\textalpha\textrho\'\textalpha\textrho%
    \texttau\texteta\textmu\textalpha}%
  \def\contentsname{\textPi\textepsilon\textrho\textiota%
    \textepsilon\textchi\'\textomicron\textmu\textepsilon\textnu\textalpha}%
  \def\listfigurename{\textKappa\textalpha\texttau\'\textalpha%
    \textlambda\textomicron\textgamma\textomicron\textfinalsigma{}
    \textSigma\textchi\texteta\textmu\'\textalpha\texttau\textomega\textnu}%
  \def\listtablename{\textKappa\textalpha\texttau\'\textalpha%
    \textlambda\textomicron\textgamma\textomicron\textfinalsigma{}
    \textPi\textiota\textnu\'\textalpha\textkappa\textomega\textnu}%
  \def\indexname{\textEpsilon\textupsilon\textrho\textepsilon%
    \texttau\'\texteta\textrho\textiota\textomicron}%
  \def\figurename{\textSigma\textchi\'\texteta\textmu\textalpha}%
  \def\tablename{\textPi\'\textiota\textnu\textalpha%
    \textkappa\textalpha\textfinalsigma}%
  \def\partname{\textMu\'\textepsilon\textrho\textomicron\textfinalsigma}%
  \def\enclname{\textSigma\textupsilon\textnu\texteta\textmu%
    \textmu\'\textepsilon\textnu\textalpha}%
  \def\ccname{\textKappa\textomicron\textiota\textnu\textomicron%
    \textpi\textomicron\'\textiota\texteta\textsigma\texteta}%
  \def\headtoname{\textPi\textrho\textomicron\textfinalsigma}%
  \def\pagename{\textSigma\textepsilon\textlambda\'\textiota%
    \textdelta\textalpha}%
  \def\seename{\textbeta\textlambda\'\textepsilon\textpi\textepsilon}%
  \def\alsoname{\textbeta\textlambda\'\textepsilon\textpi\textepsilon{}
    \textepsilon\textpi\'\textiota\textsigma\texteta\textfinalsigma}%
  \def\proofname{\textAlpha\textpi\'\textomicron%
    \textdelta\textepsilon\textiota\textxi\texteta}%
  \def\glossaryname{\textGamma\textlambda\textomega\textsigma%
    \textsigma\'\textalpha\textrho\textiota}%
  }
```

\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.

```
\let\captionspolutonikogreek\captionsgreek
\addto\captionspolutonikogreek{%
  \def\refname{\>\textAlpha\textnu\textalpha%
    \textphi\textomicron\textrho\`\textepsilon\textfinalsigma}%
  \def\indexname{\textEpsilon\<\textupsilon\textrho\textepsilon%
    \texttau\'\texteta\textrho\textiota\textomicron}%
  \def\figurename{\textSigma\textchi\~\texteta\textmu\textalpha}%
  \def\headtoname{\textPi\textrho\`\textomicron\textfinalsigma}%
  \def\alsoname{\textbeta\textlambda\'\textepsilon\textpi\textepsilon{}
    \>\textepsilon\textpi\'\textiota\textsigma\texteta\textfinalsigma}%
  \def\proofname{\>\textAlpha\textpi\'\textomicron%
    \textdelta\textepsilon\textiota\textxi\texteta}%
}
```

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

```
\def\gr@month{%
  \ifcase\month\or
    \textIota\textalpha\textnu\textomicron\textupsilon\textalpha%
      \textrho\'\textiota\textomicron\textupsilon\or
    \textPhi\textepsilon\textbeta\textrho\textomicron\textupsilon%
      \textalpha\textrho\'\textiota\textomicron\textupsilon\or
    \textMu\textalpha\textrho\texttau\'\textiota\textomicron\textupsilon\or
    \textAlpha\textpi\textrho\textiota\textlambda\'\textiota%
      \textomicron\textupsilon\or
    \textMu\textalpha\'\"\textiota\textomicron\textupsilon\or
    \textIota\textomicron\textupsilon\textnu\'\textiota%
      \textomicron\textupsilon\or
    \textIota\textomicron\textupsilon\textlambda\'\textiota%
      \textomicron\textupsilon\or
    \textAlpha\textupsilon\textgamma\textomicron\'\textupsilon\textsigma%
      \texttau\textomicron\textupsilon\or
    \textSigma\textepsilon\textpi\texttau\textepsilon\textmu%
      \textbeta\textrho\'\textiota\textomicron\textupsilon\or
    \textOmicron\textkappa\texttau\textomega\textbeta%
      \textrho\'\textiota\textomicron\textupsilon\or
    \textNu\textomicron\textepsilon\textmu\textbeta%
      \textrho\'\textiota\textomicron\textupsilon\or
    \textDelta\textepsilon\textkappa\textepsilon\textmu\textbeta%
      \textrho\'\textiota\textomicron\textupsilon%
  \fi%
}
\def\dategreek{%
  \def\today{\number\day \space \gr@month\space \number\year}}
```

\gr@c@greek

```
\def\gr@c@month{%
  \ifcase\month\or
    \>\textIota\textalpha\textnu\textomicron\textupsilon\textalpha%
      \textrho\'\textiota\textomicron\textupsilon\or
    \textPhi\textepsilon\textbeta\textrho\textomicron\textupsilon%
      \textalpha\textrho\'\textiota\textomicron\textupsilon\or
    \textMu\textalpha\textrho\texttau\'\textiota\textomicron\textupsilon\or
    \>\textAlpha\textpi\textrho\textiota\textlambda%
      \'\textiota\textomicron\textupsilon\or
    \textMu\textalpha\"\'\textiota\textomicron\textupsilon\or
    \>\textIota\textomicron\textupsilon\textnu%
      \'\textiota\textomicron\textupsilon\or
    \>\textIota\textomicron\textupsilon\textlambda%
      \'\textiota\textomicron\textupsilon\or
    \textAlpha\>\textupsilon\textgamma\textomicron\'\textupsilon\textsigma%
      \texttau\textomicron\textupsilon\or
    \textSigma\textepsilon\textpi\texttau\textepsilon\textmu\textbeta%
      \textrho\'\textiota\textomicron\textupsilon\or
    \>\textOmicron\textkappa\texttau\textomega\textbeta%
      \textrho\'\textiota\textomicron\textupsilon\or
    \textNu\textomicron\textepsilon\textmu\textbeta%
      \textrho\'\textiota\textomicron\textupsilon\or
    \textDelta\textepsilon\textkappa\textepsilon\textmu%
      \textbeta\textrho\'\textiota\textomicron\textupsilon%
  \fi
}
```

\Grtoday The macro \Grtoday produces the current date, only that the month and the day are shown as greek numerals instead of arabic as it is usually the case.

```
\def\Grtoday{%
  \expandafter\Greeknumeral\expandafter{\the\day}\space
  \gr@c@month \space
  \expandafter\Greeknumeral\expandafter{\the\year}}
```

\extrasgreek \noextrasgreek The macro \extrasgreek will perform all the extra definitions needed for the Greek language. The macro \noextrasgreek is used to cancel the actions of \extrasgreek. For the moment these macros switch the fontencoding (with 8-bit

TeX) and the definition of the internal macros \@alph and \@Alph because in Greek we do use the Greek numerals.

```
\if@bbl@greek@unicode
  \relax
\else
  \addto\extrasgreek{\greektext}
  \addto\noextrasgreek{\latintext}
\fi
```

\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).

```
\def\gr@ill@value#1{%
  \PackageWarning{babel}{Illegal value (#1) for greeknumeral}}
```

\anw@true
\anw@false
\anw@print When a a large number with three *trailing* zero's is to be printed those zeros *and* the numeric mark need to be discarded. As each 'digit' is processed by a separate 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.

```
\DeclareRobustCommand\anw@false{%
  \DeclareRobustCommand\anw@print{}}
\DeclareRobustCommand\anw@true{%
  \DeclareRobustCommand\anw@print{\anwtonos}}
\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.

```
\def\greeknumeral#1{%
```

If the value is negative or zero nothing is printed and a warning is issued.

```
  \ifnum#1<\@ne\space\gr@ill@value{#1}%
  \else
    \ifnum#1<10\expandafter\gr@num@i\number#1%
    \else
      \ifnum#1<100\expandafter\gr@num@ii\number#1%
      \else
```

We use the available shorthands for 1.000 (\@m) and 10.000 (\@M) to save a few tokens.

```
        \ifnum#1<\@m\expandafter\gr@num@iii\number#1%
        \else
          \ifnum#1<\@M\expandafter\gr@num@iv\number#1%
          \else
            \ifnum#1<100000\expandafter\gr@num@v\number#1%
            \else
```

```
              \ifnum#1<1000000\expandafter\gr@num@vi\number#1%
              \else
```

If the value is too large, nothing is printed and a warning is issued.

```
                \space\gr@ill@value{#1}%
              \fi
            \fi
          \fi
        \fi
      \fi
    \fi
  \fi
}
```

\Greeknumeral

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

```
\def\Greeknumeral#1{%
  \expandafter\MakeUppercase\expandafter{\greeknumeral{#1}}}
```

\greek@alph
\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 definited 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

```
\let\latin@alph\@alph
\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.

```
\def\greek@alph#1{\expandafter\greeknumeral\expandafter{\the#1}}
\def\greek@Alph#1{\expandafter\Greeknumeral\expandafter{\the#1}}
```

Now we can set up the switching.

```
\addto\extrasgreek{%
  \let\@alph\greek@alph
  \let\@Alph\greek@Alph}
\addto\noextrasgreek{%
  \let\@alph\latin@alph
  \let\@Alph\latin@Alph}
```

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

\gr@num@i
\gr@num@ii
\gr@num@iii

As there is no representation for 0 in this system the zeros are simply discarded. 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.

```
\def\gr@num@i#1{%
```

```
  \ifcase#1\or \textalpha\or \textbeta\or \textgamma\or \textdelta\or
  \textepsilon\or \textstigma\or \textzeta\or \texteta\or \texttheta\fi
  \ifnum#1=\z@\else\anw@true\fi\anw@print}
\def\gr@num@ii#1{%
  \ifcase#1\or \textiota\or \textkappa\or \textlambda\or \textmu\or
  \textnu\or \textxi\or \textomicron\or \textpi\or \textqoppa\fi
  \ifnum#1=\z@\else\anw@true\fi\gr@num@i}
\def\gr@num@iii#1{%
  \ifcase#1\or \textrho\or \textsigma\or \texttau\or \textupsilon\or
  \textphi\or \textchi\or \textpsi\or \textomega\or \textsampi\fi
  \ifnum#1=\z@\anw@false\else\anw@true\fi\gr@num@ii}
```

\gr@num@iv
\gr@num@v
\gr@num@vi

The first three ‘digits’ always have the numeric mark, except when one is discarded because it’s value is zero.

```
\def\gr@num@iv#1{%
  \ifnum#1=\z@\else\katwtonos\fi
  \ifcase#1\or \textalpha\or \textbeta\or \textgamma\or \textdelta\or
  \textepsilon\or \stigma\or \textzeta\or \texteta\or \texttheta\fi
  \gr@num@iii}
\def\gr@num@v#1{%
  \ifnum#1=\z@\else\katwtonos\fi
  \ifcase#1\or \textiota\or \textkappa\or \textlambda\or \textmu\or
  \textnu\or \textxi\or \textomicron\or \textpi\or \textqoppa\fi
  \gr@num@iv}
\def\gr@num@vi#1{%
  \katwtonos
  \ifcase#1\or \textrho\or \textsigma\or \texttau\or \textupsilon\or
  \textphi\or \textchi\or \textpsi\or \textomega\or \textsampi\fi
  \gr@num@v}
```

Handling of Diacritics differs between Unicode and LGR encoding.

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

```
\if@bbl@greek@unicode
  \InputIfFileExists{xgreek-fixes.def}{}{%
    \PackageWarning{babel}{Can't find the file "xgreek-fixes.def".\MessageBreak
      For correct up/downcasing of Greek text,
      install the "polyglossia" package.}%
    }
\else % 8-bit TeX
```

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.

```
\begingroup
  \@ifundefined{active@char\string!}{}{\catcode`!=12\relax}
  \catcode`\~=12
  \lccode`\!=`\~
  \lowercase{\def\x{\endgroup
      \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 wich 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.

```
\addto\extrasgreek{%
  \babel@savevariable{\lccode`v}\lccode`v=`v%
  \babel@savevariable{\lccode`\'}\lccode`\'=`\'%
  \babel@savevariable{\lccode`\"}\lccode`\"=`\"}
\addto\extraspolutonikogreek{%
  \babel@savevariable{\lccode`\<}\lccode`\<=`\<%
  \babel@savevariable{\lccode`\>}\lccode`\>=`\>%
  \babel@savevariable{\lccode`\~}\lccode`\~=`\~%
  \babel@savevariable{\lccode`\|}\lccode`\|=`\|%
  \babel@savevariable{\lccode`\`}\lccode`\`=`\`}
```

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

```
\addto\extrasgreek{%
  \babel@savevariable{\uccode`\"}\uccode`\"=`\"%
  \babel@savevariable{\uccode`\'}\uccode`\'=159} %% 159 == ^^9f
\addto\extraspolutonikogreek{%
  \babel@savevariable{\uccode`\~}\uccode`\~=159%
  \babel@savevariable{\uccode`\>}\uccode`\>=159%
  \babel@savevariable{\uccode`\<}\uccode`\<=159%
  \babel@savevariable{\uccode`\|}\uccode`\|=`\|%
  \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 TeX’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 tile `~` character is used for such purposes. Before we do this we save it's original lowercase code to restore it once we're done.

```
  \@tempcnta=\lccode`\~
  \lccode`\~=159
  \lowercase{%
    \initiate@active@char{~}%
    \declare@shorthand{greek}{~}{}}
  \lccode`\~=\@tempcnta
```

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

```
  \addto\extraspolutonikogreek{\languageshorthands{greek}}%
  \declare@shorthand{greek}{~}{\greek@tilde}
\fi  % End of LGR-specific code not required with XeTeX/LuaTeX.
```

We now define a few symbols which are used in the typesetting of greek numerals, as well as some other symbols which are usefull, such as the $\epsilon\upsilon\rho\omega$ symbol, etc.

```
\providecommand*{\anwtonos}{\textnumeralsigngreek}
\providecommand*{\katwtonos}{\textnumeralsignlowergreek}
\providecommand*{\qoppa}{\textqoppa}
\providecommand*{\stigma}{\textstigma}
\providecommand*{\sampi}{\textsampi}
\providecommand*{\Digamma}{\textDigamma}
\providecommand*{\ddigamma}{\textdigamma}
\providecommand*{\vardigamma}{\textvardigamma}
\providecommand*{\euro}{\texteuro}
\providecommand*{\permill}{\textperthousand}
```

Since the `~` cannot be used to produce an unbreakable white space we must redefine at least the commands `\fnum@figure` and `\fnum@table` so they do not produce a `~` instead of white space.

```
%\def\fnum@figure{\figurename\nobreakspace\thefigure}
%\def\fnum@table{\tablename\nobreakspace\thetable}
```

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{\CurrentOption}
⟨/code⟩
```