

1 The Bulgarian language

The file `bulgarian.dtx`¹ 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 Note: automatic switch to Bulgarian quotation is withdrawn for the moment and may not be reintroduced at all.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 1 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 1: The extra definitions made by `bulgarian`

The quotes in table 1 can also be typeset by using the commands in table 2.

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

¹The file described in this section has version number v1.0g and was last revised on 2008/03/21. 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).

| | |
|------------------------|---|
| <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 2: More commands which produce quotes, defined by `babel`

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{bulgarian}{captionsbulgarian}
```

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

```
3 \ifx\l@bulgarian\undefined
4   \@nopatterns{Bulgarian}
5   \adddialect\l@bulgarian0
6 \fi
```

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

```
7 \let\latinencoding\cf@encoding
```

The user may choose between different available Cyrillic encodings—e.g., `X2`, `LCY`, or `LWN`. If the user wants to use a font encoding other than the default (`T2A`), he has to load the corresponding file *before* `bulgarian.sty`. This may be done in the following way:

```
\usepackage[LCY,OT1]{fontenc}      %overwrite the default encoding;
\usepackage[english,bulgarian]{babel}
```

Note: most people would prefer the `T2A` to `X2`, because `X2` does not contain Latin letters, and users should be very careful to switch the language every time they want to typeset a Latin word inside a Bulgarian phrase or vice versa. On the other hand, switching the language is a good practice anyway. With a decent text processing program it does not involve more work than switching between the Bulgarian and English keyboard. Moreover that the far most common disruption occurs as a result of forgetting to switch back to cyrillic keyboard.

We parse the `\cdp@list` containing the encodings known to L^AT_EX in the order they were loaded. We set the `\cyrillicencoding` to the *last* loaded encoding in the list of supported Cyrillic encodings: OT2, LWN, LCY, X2, T2C, T2B, T2A, if any.

```

8 \def\reserved@a#1#2{%
9   \edef\reserved@b{#1}%
10  \edef\reserved@c{#2}%
11  \ifx\reserved@b\reserved@c
12    \let\cyrillicencoding\reserved@c
13  \fi}
14 \def\cdp@elt#1#2#3#4{%
15   \reserved@a{#1}{OT2}%
16   \reserved@a{#1}{LWN}%
17   \reserved@a{#1}{LCY}%
18   \reserved@a{#1}{X2}%
19   \reserved@a{#1}{T2C}%
20   \reserved@a{#1}{T2B}%
21   \reserved@a{#1}{T2A}}
22 \cdp@list

```

Now, if `\cyrillicencoding` is undefined, then the user did not load any of supported encodings. So, we have to set `\cyrillicencoding` to some default value. We test the presence of the encoding definition files in the order from less preferable to more preferable encodings. We use the lowercase names (i.e., `lcyenc.def` instead of `LCYenc.def`).

```

23 \ifx\cyrillicencoding\undefined
24   \IfFileExists{ot2enc.def}{\def\cyrillicencoding{OT2}}\relax
25   \IfFileExists{lwnenc.def}{\def\cyrillicencoding{LWN}}\relax
26   \IfFileExists{lcyenc.def}{\def\cyrillicencoding{LCY}}\relax
27   \IfFileExists{x2enc.def}{\def\cyrillicencoding{X2}}\relax
28   \IfFileExists{t2cenc.def}{\def\cyrillicencoding{T2C}}\relax
29   \IfFileExists{t2benc.def}{\def\cyrillicencoding{T2B}}\relax
30   \IfFileExists{t2aenc.def}{\def\cyrillicencoding{T2A}}\relax

```

If `\cyrillicencoding` is still undefined, then the user seems not to have a properly installed distribution. A fatal error.

```

31 \ifx\cyrillicencoding\undefined
32   \PackageError{babel}%
33   {No Cyrillic encoding definition files were found}%
34   {Your installation is incomplete. \MessageBreak
35   You need at least one of the following files: \MessageBreak
36   \space\space
37   x2enc.def, t2aenc.def, t2benc.def, t2cenc.def, \MessageBreak
38   \space\space
39   lcyenc.def, lwnenc.def, ot2enc.def.}%
40 \else

```

We avoid `\usepackage[\cyrillicencoding]{fontenc}` because we don't want to force the switch of `\encodingdefault`.

```

41   \lowercase

```

```

42     \expandafter{\expandafter\input\cyrillicencoding enc.def\relax}%
43   \fi
44 \fi

```

```

    \PackageInfo{babel}
      {Using '\cyrillicencoding' as a default Cyrillic encoding}%

45 \DeclareRobustCommand{\Bulgarian}{%
46   \fontencoding\cyrillicencoding\selectfont
47   \let\encodingdefault\cyrillicencoding
48   \expandafter\set@hyphenmins\bulgarianhyphenmins
49   \language\l@bulgarian}
50 \DeclareRobustCommand{\English}{%
51   \fontencoding\latinencoding\selectfont
52   \let\encodingdefault\latinencoding
53   \expandafter\set@hyphenmins\englishhyphenmins
54   \language\l@english}
55 \let\Bul\Bulgarian
56 \let\Bg\Bulgarian
57 \let\cyrillictext\Bulgarian
58 \let\cyr\Bulgarian
59 \let\Eng\English
60 \def\selectenglange{\selectlanguage{english}}
61 \def\selectbglange{\selectlanguage{bulgarian}}

```

Since the X2 encoding does not contain Latin letters, we should make some redefinitions of L^AT_EX macros which implicitly produce Latin letters.

```

62 \expandafter\ifx\csname T@X2\endcsname\relax\else

```

We put `\latinencoding` in braces to avoid problems with `\@alph` inside minipages (e.g., footnotes inside minipages) where `\@alph` is expanded and we get for example `'\fontencoding OT1'` (`\fontencoding` is robust).

```

63 \def\@Alph@eng#1{\fontencoding{\latinencoding}\selectfont
64   \ifcase#1\or A\or B\or C\or D\or E\or F\or G\or H\or I\or J\or
65   K\or L\or M\or N\or O\or P\or Q\or R\or S\or T\or U\or V\or W\or
66   X\or Y\or Z\else \ctrerr\fi}%
67 \def\@alph@eng#1{\fontencoding{\latinencoding}\selectfont
68   \ifcase#1\or a\or b\or c\or d\or e\or f\or g\or h\or i\or j\or
69   k\or l\or m\or n\or o\or p\or q\or r\or s\or t\or u\or v\or w\or
70   x\or y\or z\else \ctrerr\fi}%
71 \let\@Alph\@Alph@eng
72 \let\@alph\@alph@eng

```

Unfortunately, the commands `\AA` and `\aa` are not encoding dependent in L^AT_EX (unlike e.g., `\oe` or `\DH`). They are defined as `\r{A}` and `\r{a}`. This leads to unpredictable results when the font encoding does not contain the Latin letters 'A' and 'a' (like X2).

```

73 \DeclareTextSymbolDefault{\AA}{OT1}
74 \DeclareTextSymbolDefault{\aa}{OT1}

```

```

75 \DeclareTextCommand{\AA}{OT1}{\r A}
76 \DeclareTextCommand{\aa}{OT1}{\r a}
77 \fi

```

The following block redefines the character class of uppercase Greek letters and some accents, if it is equal to 7 (variable family), to avoid incorrect results if the font encoding in some math family does not contain these characters in places of OT1 encoding. The code was taken from `amsmath.dtx`. See comments and further explanation there.

```

78 \begingroup\catcode'\="12
79 % uppercase greek letters:
80 \def\@tempa#1{\expandafter\@tempb\meaning#1\relax\relax\relax\relax
81 "0000\@nil#1}
82 \def\@tempb#1"#2#3#4#5#6\@nil#7{%
83 \ifnum"#2=7 \count@"1#3#4#5\relax
84 \ifnum\count@<"1000 \else \global\mathchardef#7="0#3#4#5\relax \fi
85 \fi}
86 \@tempa\Gamma\@tempa\Delta\@tempa\Theta\@tempa\Lambda\@tempa\Xi
87 \@tempa\Pi\@tempa\Sigma\@tempa\Upsilon\@tempa\Phi\@tempa\Psi
88 \@tempa\Omega
89 % some accents:
90 \def\@tempa#1#2\@nil{\def\@tempc{#1}}\def\@tempb{\mathaccent}
91 \expandafter\@tempa\hat\relax\relax\@nil
92 \ifx\@tempb\@tempc
93 \def\@tempa#1\@nil{#1}%
94 \def\@tempb#1{\afterassignment\@tempa\mathchardef\@tempc=}%
95 \def\do#1"#2{}
96 \def\@tempd#1{\expandafter\@tempb#1\@nil
97 \ifnum\@tempc>"FFF
98 \xdef#1{\mathaccent"\expandafter\do\meaning\@tempc\space}%
99 \fi}
100 \@tempd\hat\@tempd\check\@tempd\tilde\@tempd\acute\@tempd\grave
101 \@tempd\dot\@tempd\ddot\@tempd\breve\@tempd\bar
102 \fi
103 \endgroup

```

The user should use the `inputenc` package when any 8-bit Cyrillic font encoding is used, selecting one of the Cyrillic input encodings. We do not assume any default input encoding, so the user should explicitly call the `inputenc` package by `\usepackage{inputenc}`. We also removed `\AtBeginDocument`, so `inputenc` should be used before `babel`.

```

104 \@ifpackageloaded{inputenc}{\fi}%
105 \def\reserved@a{LWN}%
106 \ifx\reserved@a\cyrillicencoding\else
107 \def\reserved@a{OT2}%
108 \ifx\reserved@a\cyrillicencoding\else
109 \PackageWarning{babel}%
110 {No input encoding specified for Bulgarian language}\fi\fi}

```

Now we define two commands that offer the possibility to switch between Cyrillic and Roman encodings.

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

We comment out `\latintext` since it is defined in the core of babel (babel.def). We add the shorthand `\lat` for `\latintext`. Note that `\cyrillictext` has been defined above.

```
111 % \DeclareRobustCommand{\latintext}{%
112 % \fontencoding{\latinencoding}\selectfont
113 % \def\encodingdefault{\latinencoding}}
114 \let\lat\latintext
```

`\textcyrillic` These commands take an argument which is then typeset using the requested font encoding. `\textlatin` is commented out since it is defined in the core of babel. (It is defined there with `\DeclareRobustCommand` instead.)

```
115 \DeclareTextFontCommand{\textcyrillic}{\cyrillictext}
116 % \DeclareTextFontCommand{\textlatin}{\latintext}
```

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

`\captionsbulgarian` The macro `\captionsbulgarian` defines all strings used in the four standard document classes provided with L^AT_EX. The two commands `\cyr` and `\lat` activate Cyrillic resp. Latin encoding.

```
117 \addto\captionsbulgarian{%
118   \def\prefacename{%
119     {\cyr\CYRP\cyrr\cyre\cyrd\cyrg\cyro\cyrv\cyro\cyrr}}%
120   \def\refname{%
121     {\cyr\CYRL\cyri\cyrt\cyre\cyrr\cyra\cyrt\cyru\cyrr\cyra}}%
122   \def\abstractname{%
123     {\cyr\CYRA\cyrb\cyrs\cyrt\cyrr\cyra\cyrk\cyrt}}%
124   \def\bibname{%
125     {\cyr\CYRB\cyri\cyrb\cyrl\cyri\cyro\cyrg\cyrr\cyra\cyrf\cyri\cyrya}}%
126   \def\chaptername{%
127     {\cyr\CYRG\cyrl\cyra\cyrv\cyra}}%
128   \def\appendixname{%
129     {\cyr\CYRP\cyrr\cyri\cyrl\cyro\cyrz\cyre\cyrn\cyri\cyre}}%
130   \def\contentsname{%
131     {\cyr\CYRS\cyrhdsn\cyrd\cyrhdsn\cyrr\cyrz\cyra\cyrn\cyri\cyre}}%
132   \def\listfigurename{%
133     {\cyr\CYRS\cyrp\cyri\cyrs\cyrhdsn\cyrk\ \cyrn\cyra\ \cyrf\cyri\cyrg\cyru\cyrr\cyri\cyrt}}%
134   \def\listtablename{%
135     {\cyr\CYRS\cyrp\cyri\cyrs\cyrhdsn\cyrk\ \cyrn\cyra\ \cyrt\cyra\cyrb\cyrl\cyri\cyrc\cyri}}%
136   \def\indexname{%
```

```

137     {\cyr\CYRA\cyrz\cyrb\cyru\cyrch\cyre\cyrn\ \cyru\cyrk\cyra\cyrz\cyra\cyrt\cyre\cyrl}}%
138 \def\authorname{%
139     {\cyr\CYRI\cyrn\cyre\cyrn\cyre\cyrn\ \cyru\cyrk\cyra\cyrz\cyra\cyrt\cyre\cyrl}}%
140 \def\figurename{%
141     {\cyr\CYRF\cyri\cyrg\cyru\cyrr\cyra}}%
142 \def\tablename{%
143     {\cyr\CYRT\cyra\cyrb\cyrl\cyri\cyrc\cyra}}%
144 \def\partname{%
145     {\cyr\CYRCH\cyra\cyrs\cyrt}}%
146 \def\enclname{%
147     {\cyr\CYRP\cyrr\cyri\cyrl\cyro\cyrzh\cyre\cyrn\cyri\cyrya}}%
148 \def\ccname{%
149     {\cyr\cyrk\cyro\cyrp\cyri\cyrya}}%
150 \def\headtoname{%
151     {\cyr\CYRZ\cyra}}%
152 \def\pagename{%
153     {\cyr\CYRS\cyrt\cyrr.}}%
154 \def\seename{%
155     {\cyr\cyrv\cyrzh.}}%
156 \def\alsoname{%
157     {\cyr\cyrv\cyrzh.\ \cyrs\cyrhrdsn\cyrshch\cyro\ \cyri}}%
158 \def\proofname{Proof}% <-- Needs translation
159 \def\glossaryname{Glossary}% <-- Needs translation
160 }

```

\datebulgarian The macro `\datebulgarian` redefines the command `\today` to produce Bulgarian dates. It also provides the command `\todayRoman` which produces the date with the month in capital roman numerals, a popular format for dates in Bulgarian.

```

161 \def\datebulgarian{%
162     \def\month@bulgarian{\ifcase\month\or
163         \cyrya\cyrn\cyru\cyra\cyrr\cyri\or
164         \cyrf\cyre\cyrv\cyrr\cyru\cyra\cyrr\cyri\or
165         \cyrm\cyra\cyrr\cyrt\or
166         \cyra\cyrp\cyrr\cyri\cyrl\or
167         \cyrm\cyra\cyrishrt\or
168         \cyryu\cyrn\cyri\or
169         \cyryu\cyrl\cyri\or
170         \cyra\cyrv\cyrg\cyru\cyrs\cyrt\or
171         \cyrs\cyre\cyrp\cyrt\cyre\cyrm\cyrv\cyrr\cyri\or
172         \cyro\cyrk\cyrt\cyro\cyrm\cyrv\cyrr\cyri\or
173         \cyrn\cyro\cyre\cyrm\cyrv\cyrr\cyri\or
174         \cyrd\cyre\cyrk\cyre\cyrm\cyrv\cyrr\cyri\fi}%
175 \def\month@Roman{\expandafter\@Roman\month}%
176 \def\today{\number\day~\month@bulgarian\ \number\year~\cyrg.}%
177 \def\todayRoman{\number\day.\,\month@Roman.\,\number\year~\cyrg.}%
178 }

```

\todayRoman The month is often written with roman numbers in Bulgarian dates. Here we define date in this format:

```

179 \def\Romannumeral#1{\uppercase\expandafter{\romannumeral #1}}
180 \def\todayRoman{\number\day.\Romannumeral{\month}.\number\year~\cyrg.}

```

`\extrasbulgarian` The macro `\extrasbulgarian` will perform all the extra definitions needed for the Bulgarian language. The macro `\noextrasbulgarian` is used to cancel the actions of `\extrasbulgarian`.

The first action we define is to switch on the selected Cyrillic encoding whenever we enter ‘bulgarian’.

```

181 \addto\extrasbulgarian{\cyrillictext}

```

When the encoding definition file was processed by L^AT_EX the current font encoding is stored in `\latinencoding`, assuming that L^AT_EX uses T1 or OT1 as default. Therefore we switch back to `\latinencoding` whenever the Bulgarian language is no longer ‘active’.

```

182 \addto\noextrasbulgarian{\latintext}

```

For Bulgarian the " character also is made active.

```

183 \initiate@active@char{"}

```

The code above is necessary because we need extra active characters. The character " is used as indicated in table 1. We specify that the Bulgarian group of shorthands should be used.

```

184 \addto\extrasbulgarian{\languageshorthands{bulgarian}}

```

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

```

185 \addto\extrasbulgarian{%
186   \bbl@activate{"}}
187 \addto\noextrasbulgarian{%
188   \bbl@deactivate{"}}

```

The X2 and T2* encodings do not contain `spanish_shriek` and `spanish_query` symbols; as a consequence, the ligatures ‘?’ and ‘!’ do not work with them (these characters are useless for Cyrillic texts anyway). But we define the shorthands to emulate these ligatures (optionally).

We do not use `\latinencoding` here (but instead explicitly use OT1) because the user may choose T2A to be the primary encoding, but it does not contain these characters.

```

189 <*spanishlig>
190 \declare@shorthand{bulgarian}{?}{\UseTextSymbol{OT1}\textquestiondown}
191 \declare@shorthand{bulgarian}{!}{\UseTextSymbol{OT1}\textexclamdown}
192 </spanishlig>

```

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

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

```

193 \begingroup \catcode'\12

```



```

194 \def\reserved@a{\endgroup
195   \def\@SS{\mathchar"7019}
196   \def\dq{"}}
197 \reserved@a

```

Now we can define the doublequote macros: german and french quotes. We use definitions of these quotes made in babel.sty. The french quotes are contained in the T2* encodings.

```

198 \declare@shorthand{bulgarian}{"'}{\glqq}
199 \declare@shorthand{bulgarian}{"'}{\grqq}
200 \declare@shorthand{bulgarian}{"<"}{\flqq}
201 \declare@shorthand{bulgarian}{">"}{\frqq}

```

Some additional commands:

```

202 \declare@shorthand{bulgarian}{""}{\hskip\z@skip}
203 \declare@shorthand{bulgarian}{""}{\textormath{\leavevmode\hbox{-}}{-}}
204 \declare@shorthand{bulgarian}{"="}{\nobreak-\hskip\z@skip}
205 \declare@shorthand{bulgarian}{"}{|}{%
206 \textormath{\nobreak\discretionary{-}{-}{\kern.03em}%
207 \allowhyphens}{}}

```

The next two macros for "- and "--- are somewhat different. We must check whether the second token is a hyphen character:

```

208 \declare@shorthand{bulgarian}{"-}{%

```

If the next token is '-', we typeset an emdash, otherwise a hyphen sign:

```

209   \def\bulgarian@sh@tmp{%
210     \if\bulgarian@sh@next-\expandafter\bulgarian@sh@emdash
211     \else\expandafter\bulgarian@sh@hyphen\fi
212   }%

```

T_EX looks for the next token after the first '-': the meaning of this token is written to \bulgarian@sh@next and \bulgarian@sh@tmp is called.

```

213   \futurelet\bulgarian@sh@next\bulgarian@sh@tmp}

```

Here are the definitions of hyphen and emdash. First the hyphen:

```

214 \def\bulgarian@sh@hyphen{\nobreak-\bbl@allowhyphens}

```

For the emdash definition, there are the two parameters: we must 'eat' two last hyphen signs of our emdash ...:

```

215 \def\bulgarian@sh@emdash#1#2{\cdash-#1#2}

```

\cdash ... these two parameters are useful for another macro: \cdash:

```

216 \ifx\cdash\undefined % should be defined earlier
217 \def\cdash#1#2#3{\def\tempx@{#3}%
218 \def\tempa@{-}\def\tempb@{~}\def\tempc@{*}%
219 \ifx\tempx@\tempa@\@Acdash\else
220 \ifx\tempx@\tempb@\@Bcdash\else
221 \ifx\tempx@\tempc@\@Ccdash\else
222   \errmessage{Wrong usage of cdash}\fi\fi\fi}

```

second parameter (or third for `\cdash`) shows what kind of emdash to create in next step

--- ordinary (plain) Cyrillic emdash inside text: an unbreakable thinspace will be inserted before only in case of a *space* before the dash (it is necessary for dashes after display maths formulae: there could be lists, enumerations etc. started with “—where *a* is ...” i.e., the dash starts a line). (Firstly there were planned rather soft rules for user:he may put a space before the dash or not. But it is difficult to place this thinspace automatically, i.e., by checking modes because after display formulae T_EX uses horizontal mode. Maybe there is a misunderstanding? Maybe there is another way?) After a dash a breakable thinspace is always placed;

```
223 % What is more grammatically: .2em or .2\fontdimen6\font?
224 \def\@Acddash{\ifdim\lastskip>\z@\unskip\nobreak\hskip.2em\fi
225 \cyrdash\hskip.2em\ignorespaces}%
```

--~ emdash in compound names or surnames (like Mendeleev–Klaepiron); this dash has no space characters around; after the dash some space is added `\exhyphenalty`

```
226 \def\@Bcdash{\leavevmode\ifdim\lastskip>\z@\unskip\fi
227 \nobreak\cyrdash\penalty\exhyphenpenalty\hskip\z@skip\ignorespaces}%
```

---* for denoting direct speech (a space like `\enskip` must follow the emdash);

```
228 \def\@Ccdash{\leavevmode
229 \nobreak\cyrdash\nobreak\hskip.35em\ignorespaces}%
230 %\fi
```

`\cyrdash` Finally the macro for “body” of the Cyrillic emdash. The `\cyrdash` macro will be defined in case this macro hasn’t been defined in a fontenc file. For T2*fonts, `cyrdash` will be placed in the code of the English emdash thus it uses ligature ---.

```
231 % Is there an IF necessary?
232 \ifx\cyrdash\undefined
233 \def\cyrdash{\hbox to.8em{--\hss--}}
234 \fi
```

Here a really new macro—to place thinspace between initials. This macro used instead of `\,`, allows hyphenation in the following surname.

```
235 \declare@shorthand{bulgarian}{",}{\nobreak\hskip.2em\ignorespaces}
```

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

```
236 \providehyphenmins{\CurrentOption}{\tw@\tw@}
237 \fi
```

Now the action `\extrasbulgarian` has to execute is to make sure that the command `\frenchspacing` is in effect. If this is not the case the execution of `\noextrasbulgarian` will switch it off again.

```

238 \addto\extrasbulgarian{\bbl@frenchspacing}
239 \addto\noextrasbulgarian{\bbl@nonfrenchspacing}

```

Make the double quotes produce the traditional quotes used in Bulgarian texts (these are the German quotes).

```

240 % \initiate@active@char{' }
241 % \initiate@active@char{' }
242 % \addto\extrasbulgarian{%
243 %   \bbl@activate{' }
244 % \addto\extrasbulgarian{%
245 %   \bbl@activate{' }
246 % \addto\noextrasbulgarian{%
247 %   \bbl@deactivate{' }
248 % \addto\noextrasbulgarian{%
249 %   \bbl@deactivate{' }
250 % \def\mlron{\bbl@activate{' }\bbl@activate{' }}
251 % \def\mlroff{\bbl@deactivate{' }\bbl@deactivate{' }}
252 % \declare@shorthand{bulgarian}{' }{\glqq}
253 % \declare@shorthand{bulgarian}{' }{\grqq}

```

Next we add a new enumeration style for Bulgarian manuscripts with Cyrillic letters, and later on we define some math operator names in accordance with Bulgarian typesetting traditions.

`\@Alph@bul` We begin by defining `\@Alph@bul` which works like `\@Alph`, but produces (uppercase) Cyrillic letters instead of Latin ones. The letters ISHRT, HRDSN and SFTSN are skipped, as usual for such enumeration.

```

254 \def\enumBul{\let\@Alph\@Alph@bul \let\@alph\@alph@bul}
255 \def\enumEng{\let\@Alph\@Alph@eng \let\@alph\@alph@eng}
256 \def\enumLat{\let\@Alph\@Alph@eng \let\@alph\@alph@eng}
257 \addto\extrasbulgarian{\enumBul}
258 \addto\noextrasbulgarian{\enumLat}
259 \def\@Alph@bul#1{%
260   \ifcase#1\or
261     \CYRA\or \CYRB\or \CYRV\or \CYRG\or \CYRD\or \CYRE\or \CYRZH\or
262     \CYRZ\or \CYRI\or \CYRK\or \CYRL\or \CYRM\or \CYRN\or \CYRO\or
263     \CYRP\or \CYRR\or \CYRS\or \CYRT\or \CYRU\or \CYRF\or \CYRH\or
264     \CYRC\or \CYRCH\or \CYRSH\or \CYRSHCH\or \CYRYU\or \CYRYA\else
265     \@ctrerrr\fi
266   }
267 \def\@Alph@eng#1{%
268   \ifcase#1\or
269     A\or B\or C\or D\or E\or F\or G\or H\or I\or J\or K\or L\or M\or
270     N\or O\or P\or Q\or R\or S\or T\or U\or V\or W\or X\or Y\or Z\else
271     \@ctrerrr\fi
272   }

```

`\@alph@bul` The macro `\@alph@bul` is similar to `\@Alph@bul`; it produces lowercase Bulgarian letters.


```

\arctan:
308 \def\arctg{\mathop{\operatorname{\arctg}}\nolimits}
\arccot:
309 \def\arcctg{\mathop{\operatorname{\arcctg}}\nolimits}
The following macro conflicts with \th defined in Latin 1 encoding: \tanh:
310 \addto\extrasrussian{%
311   \babel@save{\th}%
312   \let\ltx@th\th
313   \def\th{\textormath{\ltx@th}%
314             {\mathop{\operatorname{\th}}\nolimits}}%
315   }
\cot:
316 \def\ctg{\mathop{\operatorname{\ctg}}\nolimits}
\coth:
317 \def\cth{\mathop{\operatorname{\cth}}\nolimits}
\csc:
318 \def\cosec{\mathop{\operatorname{\cosec}}\nolimits}
This is for compatibility with older Bulgarian packages.
319 \DeclareRobustCommand{\No}{%
320   \ifmmode{\nfss@text{\textnumero}}\else\textnumero\fi}
The macro \ldf@finish takes care of looking for a configuration file, setting
the main language to be switched on at \begin{document} and resetting the
category code of @ to its original value.
321 \ldf@finish{bulgarian}
322 \end{code}

```