1 The Ukrainian language

The file `ukraineb.dtx`[1] defines all the language-specific macros for the Ukrainian language. It needs the file `cyrcod` for success documentation with Ukrainian encodings (see below).

For this language the character `"` is made active. In table 1 an overview is given of its purpose.

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

Table 1: The extra definitions made by `ukraineb`

The quotes in table 1 (see, also table **??**) can also be typeset by using the commands in table 2 (see, also table **??**).

`\cdash---`	Cyrillic emdash in plain text.
`\cdash--~`	Cyrillic emdash in compound names (surnames).
`\cdash--*`	Cyrillic emdash for denoting direct speech.
`\glqq`	for German left double quotes (looks like „).
`\grqq`	for German right double quotes (looks like ").
`\flqq`	for French left double quotes (looks like ≪).
`\frqq`	for French right double quotes (looks like ≫).
`\dq`	the original quotes character (`"`).

Table 2: More commands which produce quotes, defined by babel

[1] The file described in this section has version number v1.1l. This file was derived from the `russianb.dtx` version 1.1g.

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 Ukrainian and Russian languages were borrowed from other languages (e.g. French and German) so they keep their original names.

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{ukrainian}{captionsukrainian}
```

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

```
\ifx\l@ukrainian\@undefined
  \@nopatterns{Ukrainian}
  \adddialect\l@ukrainian0
\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!

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

The user may choose between different available Cyrillic encodings—e.g., X2, LCY, or LWN. Hopefully, X2 will eventually replace the two latter encodings (LCY and LWN). If the user wants to use another font encoding than the default (T2A), he has to load the corresponding file *before* ukraineb.sty. This may be done in the following way:

```
% override the default X2 encoding used in Babel
\usepackage[LCY,OT1]{fontenc}
\usepackage[english,ukrainian]{babel}
```

Note: for the Ukrainian language, the T2A encoding is better than 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 Ukrainian phrase or vice versa.

We parse the \cdp@list containing the encodings known to LaTeX 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.

```
\def\reserved@a#1#2{%
   \edef\reserved@b{#1}%
   \edef\reserved@c{#2}%
   \ifx\reserved@b\reserved@c
     \let\cyrillicencoding\reserved@c
```

```
   \fi}
\def\cdp@elt#1#2#3#4{%
   \reserved@a{#1}{OT2}%
   \reserved@a{#1}{LWN}%
   \reserved@a{#1}{LCY}%
   \reserved@a{#1}{X2}%
   \reserved@a{#1}{T2C}%
   \reserved@a{#1}{T2B}%
   \reserved@a{#1}{T2A}}
\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`).

```
\ifx\cyrillicencoding\undefined
  \IfFileExists{ot2enc.def}{\def\cyrillicencoding{OT2}}\relax
  \IfFileExists{lwnenc.def}{\def\cyrillicencoding{LWN}}\relax
  \IfFileExists{lcyenc.def}{\def\cyrillicencoding{LCY}}\relax
  \IfFileExists{x2enc.def}{\def\cyrillicencoding{X2}}\relax
  \IfFileExists{t2cenc.def}{\def\cyrillicencoding{T2C}}\relax
  \IfFileExists{t2benc.def}{\def\cyrillicencoding{T2B}}\relax
  \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.

```
  \ifx\cyrillicencoding\undefined
    \PackageError{babel}%
      {No Cyrillic encoding definition files were found}%
      {Your installation is incomplete.\MessageBreak
       You need at least one of the following files:\MessageBreak
       \space\space
       x2enc.def, t2aenc.def, t2benc.def, t2cenc.def,\MessageBreak
       \space\space
       lcyenc.def, lwnenc.def, ot2enc.def.}%
  \else
```

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

```
    \lowercase
      \expandafter{\expandafter\input\cyrillicencoding enc.def\relax}%
  \fi
\fi

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

```
\DeclareRobustCommand{\Ukrainian}{%
  \fontencoding\cyrillicencoding\selectfont
  \let\encodingdefault\cyrillicencoding
  \expandafter\set@hyphenmins\ukrainianhyphenmins
  \language\l@ukrainian}%
\DeclareRobustCommand{\English}{%
  \fontencoding\latinencoding\selectfont
  \let\encodingdefault\latinencoding
  \expandafter\set@hyphenmins\englishhyphenmins
  \language\l@english}%
\let\Ukr\Ukrainian
\let\Eng\English
\let\cyrillictext\Ukrainian
\let\cyr\Ukrainian
```

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

```
\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).

```
  \def\@alph#1{{\fontencoding{\latinencoding}\selectfont
    \ifcase#1\or
      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 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\@ctrerr\fi}}%
  \def\@Alph#1{{\fontencoding{\latinencoding}\selectfont
    \ifcase#1\or
      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 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\@ctrerr\fi}}%
```

Unfortunately, the commands `\AA` and `\aa` are not encoding dependent in LaTeX (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`).

```
  \DeclareTextSymbolDefault{\AA}{OT1}
  \DeclareTextSymbolDefault{\aa}{OT1}
  \DeclareTextCommand{\aa}{OT1}{\r a}
  \DeclareTextCommand{\AA}{OT1}{\r A}
\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.

```
% \begingroup\catcode`\"=12
% % uppercase greek letters:
% \def\@tempa#1{\expandafter\@tempb\meaning#1\relax\relax\relax\relax
%   "0000\@nil#1}
% \def\@tempb#1"#2#3#4#5#6\@nil#7{%
%   \ifnum"#2=7 \count@"1#3#4#5\relax
%     \ifnum\count@<"1000 \else \global\mathchardef#7="0#3#4#5\relax \fi
%   \fi}
% \@tempa\Gamma\@tempa\Delta\@tempa\Theta\@tempa\Lambda\@tempa\Xi
% \@tempa\Pi\@tempa\Sigma\@tempa\Upsilon\@tempa\Phi\@tempa\Psi
% \@tempa\Omega
% % some accents:
% \def\@tempa#1#2\@nil{\def\@tempc{#1}}\def\@tempb{\mathaccent}
% \expandafter\@tempa\hat\relax\relax\@nil
% \ifx\@tempb\@tempc
%   \def\@tempa#1\@nil{#1}%
%   \def\@tempb#1{\afterassignment\@tempa\mathchardef\@tempc=}%
%   \def\do#1"#2{}
%   \def\@tempd#1{\expandafter\@tempb#1\@nil
%     \ifnum\@tempc>"FFF
%       \xdef#1{\mathaccent"\expandafter\do\meaning\@tempc\space}%
%     \fi}
%   \@tempd\hat\@tempd\check\@tempd\tilde\@tempd\acute\@tempd\grave
%   \@tempd\dot\@tempd\ddot\@tempd\breve\@tempd\bar
% \fi
% \endgroup
```

The user must 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.

```
\@ifpackageloaded{inputenc}{}{%
  \def\reserved@a{LWN}%
  \ifx\reserved@a\cyrillicencoding\else
    \def\reserved@a{OT2}%
    \ifx\reserved@a\cyrillicencoding\else
      \PackageWarning{babel}%
        {No input encoding specified for Ukrainian language}
  \fi\fi}
```

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

`\cyrillictext` `\latintext`

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.

```
%\DeclareRobustCommand{\latintext}{%
%  \fontencoding{\latinencoding}\selectfont
%  \def\encodingdefault{\latinencoding}}
\let\lat\latintext
```

`\textcyrillic` `\textlatin` These commands take an argument which is then typeset using the requested font encoding.

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

We make the TEX

```
%\ifx\ltxTeX\undefined\let\ltxTeX\TeX\fi
%\ProvideTextCommandDefault{\TeX}{\textlatin{\ltxTeX}}
```

and LATEX logos encoding independent.

```
%\ifx\ltxLaTeX\undefined\let\ltxLaTeX\LaTeX\fi
%\ProvideTextCommandDefault{\LaTeX}{\textlatin{\ltxLaTeX}}
```

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

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

```
\addto\captionsukrainian{%
  \def\prefacename{{\cyr\CYRV\cyrs\cyrt\cyru\cyrp}}%
% \def\prefacename{{\cyr\CYRP\cyre\cyrr\cyre\cyrd\cyrm\cyro\cyrv\cyra}}%
  \def\refname{%
    {\cyr\CYRL\cyrii\cyrt\cyre\cyrr\cyra\cyrt\cyru\cyrr\cyra}}%
%  \def\refname{%
%    {\cyr\CYRP\cyre\cyrr\cyre\cyrl\cyrii\cyrk
%         \ \cyrp\cyro\cyrs\cyri\cyrl\cyra\cyrn\cyrsftsn}}%
  \def\abstractname{%
    {\cyr\CYRA\cyrn\cyro\cyrt\cyra\cyrc\cyrii\cyrya}}%
%  \def\abstractname{{\cyr\CYRR\cyre\cyrf\cyre\cyrr\cyra\cyrt}}%
  \def\bibname{%
    {\cyr\CYRB\cyrii\cyrb\cyrl\cyrii\cyro\cyrgup\cyrr\cyra\cyrf\cyrii\cyrya}}%
% \def\bibname{{\cyr\CYRL\cyrii\cyrt\cyre\cyrr\cyra\cyrt\cyru\cyrr\cyra}}%
  \def\chaptername{{\cyr\CYRR\cyro\cyrz\cyrd\cyrii\cyrl}}%
%  \def\chaptername{{\cyr\CYRG\cyrl\cyra\cyrv\cyra}}%
  \def\appendixname{{\cyr\CYRD\cyro\cyrd\cyra\cyrt\cyro\cyrk}}%
  \def\contentsname{{\cyr\CYRZ\cyrm\cyrii\cyrs\cyrt}}%
  \def\listfigurename{{\cyr\CYRP\cyre\cyrr\cyre\cyrl\cyrii\cyrk
        \ \cyrii\cyrl\cyryu\cyrs\cyrt\cyrr\cyra\cyrc\cyrii\cyrishrt}}%
  \def\listtablename{{\cyr\CYRP\cyre\cyrr\cyre\cyrl\cyrii\cyrk
        \ \cyrt\cyra\cyrb\cyrl\cyri\cyrc\cyrsftsn}}%
  \def\indexname{{\cyr\CYRP\cyro\cyrk\cyra\cyrzh\cyrch\cyri\cyrk}}%
  \def\authorname{{\cyr\CYRII\cyrm\cyre\cyrn\cyrn\cyri\cyrishrt
        \ \cyrp\cyro\cyrk\cyra\cyrzh\cyrch\cyri\cyrk}}%
  \def\figurename{{\cyr\CYRR\cyri\cyrs.}}%
```

```
% \def\figurename{\cyr\CYRR\cyri\cyrs\cyru\cyrn\cyro\cyrk}}%
  \def\tablename{{\cyr\CYRT\cyra\cyrb\cyrl.}}%
% \def\tablename{\cyr\CYRT\cyra\cyrb\cyrl\cyri\cyrc\cyrya}}%
  \def\partname{{\cyr\CYRCH\cyra\cyrs\cyrt\cyri\cyrn\cyra}}%
  \def\enclname{{\cyr\cyrv\cyrk\cyrl\cyra\cyrd\cyrk\cyra}}%
  \def\ccname{{\cyr\cyrk\cyro\cyrp\cyrii\cyrya}}%
  \def\headtoname{{\cyr\CYRD\cyro}}%
  \def\pagename{{\cyr\cyrs.}}%
% \def\pagename{{\cyr\cyrs\cyrt\cyro\cyrr\cyrii\cyrn\cyrk\cyra}}%
  \def\seename{{\cyr\cyrd\cyri\cyrv.}}%
  \def\alsoname{{\cyr\cyrd\cyri\cyrv.\ \cyrt\cyra\cyrk\cyro\cyrzh}}
  \def\proofname{{\cyr\CYRD\cyro\cyrv\cyre\cyrd\cyre\cyrn\cyrn\cyrya}}%
  \def\glossaryname{{\cyr\CYRS\cyrl\cyro\cyrv\cyrn\cyri\cyrk\ %
                     \cyrt\cyre\cyrr\cyrm\cyrii\cyrn\cyrii\cyrv}}%
  }
```

`\dateukrainian` The macro `\dateukrainian` redefines the command `\today` to produce Ukrainian dates.

```
\def\dateukrainian{%
  \def\today{\number\day~\ifcase\month\or
    \cyrs\cyrii\cyrch\cyrn\cyrya\or
    \cyrl\cyryu\cyrt\cyro\cyrg\cyro\or
    \cyrb\cyre\cyrr\cyre\cyrz\cyrn\cyrya\or
    \cyrk\cyrv\cyrii\cyrt\cyrn\cyrya\or
    \cyrt\cyrr\cyra\cyrv\cyrn\cyrya\or
    \cyrch\cyre\cyrr\cyrv\cyrn\cyrya\or
    \cyrl\cyri\cyrp\cyrn\cyrya\or
    \cyrs\cyre\cyrr\cyrp\cyrn\cyrya\or
    \cyrv\cyre\cyrr\cyre\cyrs\cyrn\cyrya\or
    \cyrzh\cyro\cyrv\cyrt\cyrn\cyrya\or
    \cyrl\cyri\cyrs\cyrt\cyro\cyrp\cyra\cyrd\cyra\or
    \cyrg\cyrr\cyru\cyrd\cyrn\cyrya\fi
    \space\number\year~\cyrr.}}
```

`\extrasukrainian` The macro `\extrasukrainian` will perform all the extra definitions needed for the Ukrainian language. The macro `\noextrasukrainian` is used to cancel the actions of `\extrasukrainian`.

The first action we define is to switch on the selected Cyrillic encoding whenever we enter 'ukrainian'.

```
\addto\extrasukrainian{\cyrillictext}
```

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

```
\addto\noextrasukrainian{\latintext}
```

Next we must allow hyphenation in the Ukrainian words with apostrophe whenever we enter 'ukrainian'. This solution was proposed by Vladimir Volovich

¡vvv@vvv.vsu.ru¿

```
\addto\extrasukrainian{\lccode`\'=`\'}
\addto\noextrasukrainian{\lccode`\'=0}
```

`\verbatim@font` In order to get both Latin and Cyrillic letters in verbatim text we need to change the definition of an internal LaTeX command somewhat:

```
%\def\verbatim@font{%
%  \let\encodingdefault\latinencoding
%  \normalfont\ttfamily
%  \expandafter\def\csname\cyrillicencoding-cmd\endcsname##1##2{%
%    \ifx\protect\@typeset@protect
%      \begingroup\UseTextSymbol\cyrillicencoding##1\endgroup
%    \else\noexpand##1\fi}}
```

The category code of the characters ':', ';', '!', and '?' is made `\active` to insert a little white space.

For Ukrainian (as well as for Russian and German) the " character also is made active.

Note: It is *very* questionable whether the Russian typesetting tradition requires additional spacing before those punctuation signs. Therefore, we make the corresponding code optional. If you need it, then define the `frenchpunct` docstrip option in `babel.ins`.

Borrowed from french. Some users dislike automatic insertion of a space before 'double punctuation', and prefer to decide themselves whether a space should be added or not; so a hook `\NoAutoSpaceBeforeFDP` is provided: if this command is added (in file `ukraineb.cfg`, or anywhere in a document) `ukraineb` will respect your typing, and introduce a suitable space before 'double punctuation' *if and only if* a space is typed in the source file before those signs.

The command `\AutoSpaceBeforeFDP` switches back to the default behavior of `ukraineb`.

```
⟨*frenchpunct⟩
\initiate@active@char{:}
\initiate@active@char{;}
⟨/frenchpunct⟩
⟨*frenchpunct | spanishligs⟩
\initiate@active@char{!}
\initiate@active@char{?}
⟨/frenchpunct | spanishligs⟩
\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 Ukrainian group of shorthands should be used.

```
\addto\extrasukrainian{\languageshorthands{ukrainian}}
```

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

```
\addto\extrasukrainian{%
```

```
⟨frenchpunct⟩  \bbl@activate{:}\bbl@activate{;}%
⟨frenchpunct | spanishligs⟩  \bbl@activate{!}\bbl@activate{?}%
  \bbl@activate{"}}
\addto\noextrasukrainian{%
⟨frenchpunct⟩  \bbl@deactivate{:}\bbl@deactivate{;}%
⟨frenchpunct | spanishligs⟩  \bbl@deactivate{!}\bbl@deactivate{?}%
  \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.

```
⟨*spanishligs⟩
\declare@shorthand{ukrainian}{?‘}{\UseTextSymbol{OT1}\textquestiondown}
\declare@shorthand{ukrainian}{!‘}{\UseTextSymbol{OT1}\textexclamdown}
⟨/spanishligs⟩
```

\ukrainian@sh@;@
\ukrainian@sh@:@
\ukrainian@sh@!@
\ukrainian@sh@?@

We have to reduce the amount of white space before ;, : and !. This should only happen in horizontal mode, hence the test with \ifhmode.

```
⟨*frenchpunct⟩
\declare@shorthand{ukrainian}{;}{%
  \ifhmode
```

In horizontal mode we check for the presence of a 'space', 'unskip' if it exists and place a 0.1em kerning.

```
    \ifdim\lastskip>\z@
      \unskip\nobreak\kern.1em
    \else
```

If no space has been typed, we add \FDP@thinspace which will be defined, up to the user's wishes, as an automatic added thinspace, or as \@empty.

```
      \FDP@thinspace
    \fi
  \fi
```

Now we can insert a ';' character.

```
  \string;}
```

The other definitions are very similar.

```
\declare@shorthand{ukrainian}{:}{%
  \ifhmode
    \ifdim\lastskip>\z@
      \unskip\nobreak\kern.1em
    \else
      \FDP@thinspace
    \fi
```

```
  \fi
  \string:}
\declare@shorthand{ukrainian}{!}{%
  \ifhmode
    \ifdim\lastskip>\z@
      \unskip\nobreak\kern.1em
    \else
        \FDP@thinspace
    \fi
  \fi
  \string!}
\declare@shorthand{ukrainian}{?}{%
  \ifhmode
    \ifdim\lastskip>\z@
      \unskip\nobreak\kern.1em
    \else
        \FDP@thinspace
    \fi
  \fi
  \string?}
```

`\AutoSpaceBeforeFDP` `\NoAutoSpaceBeforeFDP` `\FDP@thinspace`

`\FDP@thinspace` is defined as unbreakable spaces if `\AutoSpaceBeforeFDP` is activated or as `\@empty` if `\NoAutoSpaceBeforeFDP` is in use. The default is `\AutoSpaceBeforeFDP`.

```
\def\AutoSpaceBeforeFDP{%
      \def\FDP@thinspace{\nobreak\kern.1em}}
\def\NoAutoSpaceBeforeFDP{\let\FDP@thinspace\@empty}
\AutoSpaceBeforeFDP
```

`\FDPon` `\FDPoff`

The next macros allow to switch on/off activeness of double punctuation signs.

```
\def\FDPon{\bbl@activate{:}%
        \bbl@activate{;}%
        \bbl@activate{?}%
        \bbl@activate{!}}
\def\FDPoff{\bbl@deactivate{:}%
        \bbl@deactivate{;}%
        \bbl@deactivate{?}%
        \bbl@deactivate{!}}
```

`\system@sh@:@` `\system@sh@!@` `\system@sh@?@` `\system@sh@;@`

When the active characters appear in an environment where their Ukrainian behaviour is not wanted they should give an 'expected' result. Therefore we define shorthands at system level as well.

```
\declare@shorthand{system}{:}{\string:}
\declare@shorthand{system}{;}{\string;}
⟨/frenchpunct⟩
⟨*frenchpunct&!spanishligs⟩
\declare@shorthand{system}{!}{\string!}
\declare@shorthand{system}{?}{\string?}
⟨/frenchpunct&!spanishligs⟩
```

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 '"'.

```
\begingroup \catcode`\"12
\def\reserved@a{\endgroup
  \def\@SS{\mathchar"7019 }
  \def\dq{"}}
\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.

```
\declare@shorthand{ukrainian}{"`}{\glqq}
\declare@shorthand{ukrainian}{"'}{\grqq}
\declare@shorthand{ukrainian}{"<}{\flqq}
\declare@shorthand{ukrainian}{">}{\frqq}
```

Some additional commands:

```
\declare@shorthand{ukrainian}{""}{\hskip\z@skip}
\declare@shorthand{ukrainian}{"~}{\textormath{\leavevmode\hbox{-}}{-}}
\declare@shorthand{ukrainian}{"=}{\nobreak-\hskip\z@skip}
\declare@shorthand{ukrainian}{"|}{%
  \textormath{\nobreak\discretionary{-}{}{\kern.03em}%
              \allowhyphens}{}}
```

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

```
\declare@shorthand{ukrainian}{"-}{%
```

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

```
  \def\ukrainian@sh@tmp{%
    \if\ukrainian@sh@next-\expandafter\ukrainian@sh@emdash
    \else\expandafter\ukrainian@sh@hyphen\fi
  }%
```

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

```
  \futurelet\ukrainian@sh@next\ukrainian@sh@tmp}
```

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

```
\def\ukrainian@sh@hyphen{%
  \nobreak\-\bbl@allowhyphens}
```

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

```
\def\ukrainian@sh@emdash#1#2{\cdash-#1#2}
```

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

```
%\ifx\cdash\undefined % should be defined earlier
\def\cdash#1#2#3{\def\tempx@{#3}%
\def\tempa@{-}\def\tempb@{~}\def\tempc@{*}%
 \ifx\tempx@\tempa@\@Acdash\else
  \ifx\tempx@\tempb@\@Bcdash\else
   \ifx\tempx@\tempc@\@Ccdash\else
    \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 TEX uses horizontal mode. Maybe there is a misunderstanding? Maybe there is another way?) After a dash a breakable thinspace is always placed;

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

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

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

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

```
\def\@Ccdash{\leavevmode
 \nobreak\cyrdash\nobreak\hskip.35em\ignorespaces}%
%\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 ---.

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

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

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

\mdqon All that's left to do now is to define a couple of commands for ".
\mdqoff
```
\def\mdqon{\bbl@activate{"}}
\def\mdqoff{\bbl@deactivate{"}}
```

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

```
\providehyphenmins{\CurrentOption}{\tw@\tw@}
% temporary hack:
\ifx\englishhyphenmins\undefined
  \def\englishhyphenmins{\tw@\thr@@}
\fi
```

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

```
\addto\extrasukrainian{\bbl@frenchspacing}
\addto\noextrasukrainian{\bbl@nonfrenchspacing}
```

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

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

```
\def\Asbuk#1{\expandafter\@Asbuk\csname c@#1\endcsname}
\def\@Asbuk#1{\ifcase#1\or
  \CYRA\or\CYRB\or\CYRV\or\CYRG\or\CYRD\or\CYRE\or\CYRIE\or
  \CYRZH\or\CYRZ\or\CYRI\or\CYRII\or\CYRYI\or\CYRISHRT\or
  \CYRK\or\CYRL\or\CYRM\or\CYRN\or\CYRO\or\CYRP\or\CYRR\or
  \CYRS\or\CYRT\or\CYRU\or\CYRF\or\CYRH\or\CYRC\or\CYRCH\or
  \CYRSH\or\CYRSHCH\or\CYRYU\or\CYRYA\else\@ctrerr\fi}
```

\asbuk The macro `\asbuk` is similar to `\alph`; it produces lowercase Ukrainian letters.

```
\def\asbuk#1{\expandafter\@asbuk\csname c@#1\endcsname}
\def\@asbuk#1{\ifcase#1\or
  \cyra\or\cyrb\or\cyrv\or\cyrg\or\cyrd\or\cyre\or\cyrie\or
  \cyrzh\or\cyrz\or\cyri\or\cyrii\or\cyryi\or\cyrishrt\or
  \cyrk\or\cyrl\or\cyrm\or\cyrn\or\cyro\or\cyrp\or\cyrr\or
  \cyrs\or\cyrt\or\cyru\or\cyrf\or\cyrh\or\cyrc\or\cyrch\or
  \cyrsh\or\cyrshch\or\cyryu\or\cyrya\else\@ctrerr\fi}
```

Set up default Cyrillic math alphabets. The math groups for cyrillic letters are defined in the encoding definition files. First, declare a new alphabet for symbols, `\cyrmathrm`, based on the symbol font for Cyrillic letters defined in the encoding definition file. Note, that by default Cyrillic letters are taken from upright font in math mode (unlike Latin letters).

```
%\RequirePackage{textmath}
```

```
\@ifundefined{sym\cyrillicencoding letters}{}{%
\SetSymbolFont{\cyrillicencoding letters}{bold}\cyrillicencoding
  \rmdefault\bfdefault\updefault
\DeclareSymbolFontAlphabet\cyrmathrm{\cyrillicencoding letters}
```

And we need a few commands to be able to switch to different variants.

```
\DeclareMathAlphabet\cyrmathbf\cyrillicencoding
  \rmdefault\bfdefault\updefault
\DeclareMathAlphabet\cyrmathsf\cyrillicencoding
  \sfdefault\mddefault\updefault
\DeclareMathAlphabet\cyrmathit\cyrillicencoding
  \rmdefault\mddefault\itdefault
\DeclareMathAlphabet\cyrmathtt\cyrillicencoding
  \ttdefault\mddefault\updefault
%
\SetMathAlphabet\cyrmathsf{bold}\cyrillicencoding
  \sfdefault\bfdefault\updefault
\SetMathAlphabet\cyrmathit{bold}\cyrillicencoding
  \rmdefault\bfdefault\itdefault
}
```

Some math functions in Ukrainian and Russian math books have other names: e.g., `sinh` in Russian is written as `sh` etc. So we define a number of new math operators.

`\sinh`:

```
\def\sh{\mathop{\operator@font sh}\nolimits}
```

`\cosh`:

```
\def\ch{\mathop{\operator@font ch}\nolimits}
```

`\tan`:

```
\def\tg{\mathop{\operator@font tg}\nolimits}
```

`\arctan`:

```
\def\arctg{\mathop{\operator@font arctg}\nolimits}
```

arcctg:

```
\def\arcctg{\mathop{\operator@font arcctg}\nolimits}
```

The following macro conflicts with `\th` defined in Latin 1 encoding:

`\tanh`:

```
\addto\extrasrussian{%
  \babel@save{\th}%
  \let\ltx@th\th
  \def\th{\textormath{\ltx@th}%
                      {\mathop{\operator@font th}\nolimits}}%
  }
```

`\cot`:

```
\def\ctg{\mathop{\operator@font ctg}\nolimits}
```

`\coth`:

```
\def\cth{\mathop{\operator@font cth}\nolimits}
```

\csc:

```
\def\cosec{\mathop{\operator@font cosec}\nolimits}
```

And finally some other Ukrainian and Russian mathematical symbols:

```
\def\Prob{\mathop{\kern\z@\mathsf{P}}\nolimits}
\def\Variance{\mathop{\kern\z@\mathsf{D}}\nolimits}
\def\nsd{\mathop{\cyrmathrm{\cyrn.\cyrs.\cyrd.}}\nolimits}
\def\nsk{\mathop{\cyrmathrm{\cyrn.\cyrs.\cyrk.}}\nolimits}
\def\NSD{\mathop{\cyrmathrm{\CYRN\CYRS\CYRD}}\nolimits}
\def\NSK{\mathop{\cyrmathrm{\CYRN\CYRS\CYRK}}\nolimits}
  \def\nod{\mathop{\cyrmathrm{\cyrn.\cyro.\cyrd.}}\nolimits}    % ??????
  \def\nok{\mathop{\cyrmathrm{\cyrn.\cyro.\cyrk.}}\nolimits}    % ??????
  \def\NOD{\mathop{\cyrmathrm{\CYRN\CYRO\CYRD}}\nolimits}       % ??????
  \def\NOK{\mathop{\cyrmathrm{\CYRN\CYRO\CYRK}}\nolimits}       % ??????
\def\Proj{\mathop{\cyrmathrm{\CYRP\cyrr}}\nolimits}
```

This is for compatibility with older Ukrainian packages.

```
\DeclareRobustCommand{\No}{%
   \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.

```
\ldf@finish{ukrainian}
```

⟨/code⟩