1 The Polish language

The file `polish.dtx`\(^1\) defines all the language-specific macros for the Polish language.

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

<table>
<thead>
<tr>
<th>Character</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>\aob, for tailed-a (like ä)</td>
</tr>
<tr>
<td>A</td>
<td>\Aob, for tailed-A (like Å)</td>
</tr>
<tr>
<td>e</td>
<td>\eob, for tailed-e (like é)</td>
</tr>
<tr>
<td>E</td>
<td>\Eob, for tailed-E (like Æ)</td>
</tr>
<tr>
<td>c</td>
<td>\’c, for accented c (like ã), same with uppercase letters and n,o,s</td>
</tr>
<tr>
<td>l</td>
<td>\lpb{}, for l with stroke (like łą)</td>
</tr>
<tr>
<td>L</td>
<td>\Lpb{}, for L with stroke (like Ł)</td>
</tr>
<tr>
<td>r</td>
<td>\zkb{}, for pointed z (like ż), cf. pronunciation</td>
</tr>
<tr>
<td>R</td>
<td>\Zkb{}, for pointed Z (likeŻ)</td>
</tr>
<tr>
<td>z</td>
<td>\’z, for accented z</td>
</tr>
<tr>
<td>Z</td>
<td>\’Z, for accented Z</td>
</tr>
<tr>
<td>&quot;</td>
<td>disable ligature at this position.</td>
</tr>
<tr>
<td>-</td>
<td>an explicit hyphen sign, allowing hyphenation in the rest of the word.</td>
</tr>
<tr>
<td>&quot;&quot;</td>
<td>like &quot;-&quot;, but producing no hyphen sign (for compound words with hyphen, e.g. x=&quot;&quot;y).</td>
</tr>
<tr>
<td>&quot;‘</td>
<td>for German left double quotes (looks like ,,).</td>
</tr>
<tr>
<td>&quot;’</td>
<td>for German right double quotes.</td>
</tr>
<tr>
<td>&quot;&lt;</td>
<td>for French left double quotes (similar to &lt;&lt;).</td>
</tr>
<tr>
<td>&quot;&gt;</td>
<td>for French right double quotes (similar to &gt;&gt;).</td>
</tr>
</tbody>
</table>

Table 1: The extra definitions made by `polish.sty`

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

\begin{verbatim}
1 (+code)
2 \LdfInit{polish}\captionspolish

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

3 \ifx\@polish\@undefined
4 \@nopatterns{Polish}
5 \adddialect{\@polish}\fi
\end{verbatim}

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

\(^1\)The file described in this section has version number v1.2l and was last revised on 2005/03/31.
\captionspolish  The macro \captionspolish defines all strings used in the four standard document classes provided with \LaTeX.
\begin{verbatim}
6 \addto\captionspolish{%
7 \def\prefacename{Przedmowa}%
8 \def\refname{Literatura}%
9 \def\abstractname{Streszczenie}%
10 \def\bibname{Bibliografia}%
11 \def\chaptername{Rozdzia\l}%
12 \def\appendixname{Dodatek}%
13 \def\contentsname{Spis tre\'sci}%
14 \def\listfigurename{Spis rysunk\'ow}%
15 \def\listtablename{Spis tablic}%
16 \def\indexname{Indeks}%
17 \def\figurename{Rysunek}%
18 \def\tablename{Tablica}%
19 \def\partname{Cz\'eb\'osc}%
20 \def\enclname{Za\l\'acznik}%
21 \def\ccname{Kopie:}%
22 \def\headtoname{Do}%
23 \def\pagename{Strona}%
24 \def\seename{Por\'ownaj}%
25 \def\alsoname{Por\'ownaj tak\'.ze}%
26 \def\proofname{Dow\'od}%
27 \def\glossaryname{Glossary}% <-- Needs translation
28 }
\end{verbatim}

\datepolish  The macro \datepolish redefines the command \today to produce Polish dates.
\begin{verbatim}
29 \def\datepolish{%
30 \def\today{\number\day~\ifcase\month\or
31 stycznia\or lutego\or marca\or kwietnia\or maja\or czerwca\or lipca\or
32 sierpnia\or wrze\'nia\or pa\'dziernika\or listopada\or grudnia\fi
33 \space\number\year}%
34 }
\end{verbatim}

\extraspolish \noextraspolish  The macro \extraspolish will perform all the extra definitions needed for the Polish language. The macro \noextraspolish is used to cancel the actions of \extraspolish.

For Polish the " character is made active. This is done once, later on its definition may vary. Other languages in the same document may also use the " character for shorthands; we specify that the polish group of shorthands should be used.
\begin{verbatim}
35 \initiate@active@char{"}
36 \addto\extraspolish{\languageshorthands{polish}}
37 \addto\extraspolish{\bbl@activate{"}}
\end{verbatim}

Don’t forget to turn the shorthands off again.
\begin{verbatim}
38 \addto\noextraspolish{\bbl@deactivate{"}}
\end{verbatim}
The code above is necessary because we need an extra active character. This character is then used as indicated in table 1.

If you have problems at the end of a word with a linebreak, use the other version without hyphenation tricks. Some TeX wizard may produce a better solution with forecasting another token to decide whether the character after the double quote is the last in a word. Do it and let us know.

In Polish texts some letters get special diacritical marks. Leszek Holenderski designed the following code to position the diacritics correctly for every font in every size. These macros need a few extra dimension variables.

\newdimen\pl@left
\newdimen\pl@down
\newdimen\pl@right
\newdimen\pl@temp

The macro \sob is used to put the ‘ogonek’ in the right place.

\def\sob#1#2#3#4#5{%parameters: letter and fractions hl,ho,vl,vo
\setbox0\hbox{#1}\setbox1\hbox{$_\mathchar'454$}\setbox2\hbox{p}%
pl@right=#2\wd0 \advance pl@right by-#3\wd1
pl@down=#5\ht1 \advance pl@down by-#4\ht0
pl@left=pl@right \advance pl@left by\wd1
pl@temp=-pl@down \advance pl@temp by\dp2 \dp1=pl@temp
\leavevmode\kern pl@right\lower pl@down\box1\kern-\pl@left #1}

The ogonek is placed with the letters ‘a’, ‘A’, ‘e’, and ‘E’.

\DeclareTextCommand{\aob}{OT1}{\sob a{.66}{.20}{0}{.90}}
\DeclareTextCommand{\Aob}{OT1}{\sob A{.80}{.50}{0}{.90}}
\DeclareTextCommand{\eob}{OT1}{\sob e{.50}{.35}{0}{.93}}
\DeclareTextCommand{\Eob}{OT1}{\sob E{.60}{.35}{0}{.90}}

For the ‘new’ T1 encoding we can provide simpler definitions.

\DeclareTextCommand{\aob}{T1}{\k a}
\DeclareTextCommand{\Aob}{T1}{\k A}
\DeclareTextCommand{\eob}{T1}{\k e}
\DeclareTextCommand{\Eob}{T1}{\k E}

Construct the characters by default from the OT1 encoding.

\ProvideTextCommandDefault{\aob}{\UseTextSymbol{OT1}{\aob}}
\ProvideTextCommandDefault{\Aob}{\UseTextSymbol{OT1}{\Aob}}
\ProvideTextCommandDefault{\eob}{\UseTextSymbol{OT1}{\eob}}
\ProvideTextCommandDefault{\Eob}{\UseTextSymbol{OT1}{\Eob}}

The macro \spb is used to put the ‘poprzeczka’ in the right place.

\def\spb#1#2#3#4#5{%parameters: letter and fractions hl,ho,vl,vo
\setbox0\hbox{#1}\setbox1\hbox{\char'023}%
pl@right=#2\wd0 \advance pl@right by-#3\wd1
pl@down=#5\ht1 \advance pl@down by-#4\ht0
pl@left=pl@right \advance pl@left by\wd1
ht1=pl@down \dp1=-\pl@down
The macro \skb is used to put the ‘kropka’ in the right place.

\def\skb#1#2#3#4#5{\setbox0\hbox{#1} \setbox1\hbox{\char'056}\pl@right=#2\wd0 \advance\pl@right by-#3\wd1 \pl@down=#5\ht1 \advance\pl@down by-#4\ht0 \pl@left=\pl@right \advance\pl@left by\wd1 \leavevmode \kern\pl@right\lower\pl@down\box1\kern-\pl@left #1}

For the ‘poprzeczka’ and the ‘kropka’ in text fonts we don’t need any special coding, but we can (almost) use what is already available.

\def\textpl{\def\lpb{\plll} \def\Lpb{\pLLL} \def\zkb{.z} \def\Zkb{.Z}}

Initially we assume that typesetting is done with text fonts.

\let\lll=\l \let\LLL=\L \def\plll{\lll} \def\pLLL{\LLL}

But for the ‘teletype’ font in ‘OT1’ encoding we have to take some special actions, involving the macros defined above.

\def\telepl{\def\lpb{\spb l{.45}{.5}{.4}{.8}} \def\Lpb{\spb L{.23}{.5}{.4}{.8}} \def\zkb{\skb z{.5}{.5}{1.2}{0}} \def\Zkb{\skb Z{.5}{.5}{1.1}{0}}}

To activate these codes the font changing commands as they are defined in \TeX\ are modified. The same is done for plain \TeX’s font changing commands.

When \selectfont is undefined the current format is supposed to be either plain (based) or \LaTeX\ 2.09.
When \selectfont exists we assume LTEX $\epsilon_2$.

Currently we support the OT1 and T1 encodings. For T1 we don’t have to make a difference between typewriter fonts and other fonts, they all have the same glyphs.

For OT1 we need to check the current font family, stored in \f@family. Unfortunately we need a hack as \ttdefault is defined as a \long macro, while \f@family is not.

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

Now we can define the doublequote macros for diacritics,

The command \polishrz defines the shorthands "r, "z and "x to produce pointed z, accented z and "x. This is the default as these shorthands were defined by this language definition file for quite some time.
The command `\polishzx` switches to a different set of shorthands, "z", "x" and "r" to produce pointed z, accented z and "r; a different shorthand notation also in use.

Then we define access to two forms of quotation marks, similar to the german and french quotation marks.

And we want to have a shorthand for disabling a ligature.

All that’s left to do now is to define a couple of commands for reasons of compatibility with `polish.tex`.

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.