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.

`"a`	or `\aob`, for tailed-a (like ą)
`"A`	or `\Aob`, for tailed-A (like Ą)
`"e`	or `\eob`, for tailed-e (like ę)
`"E`	or `\Eob`, for tailed-E (like Ę)
`"c`	or `\'c`, for accented c (like ć), same with uppercase letters and n,o,s
`"l`	or `\lpb{}`, for l with stroke (like ł)
`"L`	or `\Lpb{}`, for L with stroke (like Ł)
`"r`	or `\zkb{}`, for pointed z (like ż), cf. pronounciation
`"R`	or `\Zkb{}`, for pointed Z (like Ż)
`"z`	or `\'z`, for accented z
`"Z`	or `\'Z`, for accented Z
`"|`	disable ligature at this position.
`"-`	an explicit hyphen sign, allowing hyphenation in the rest of the word.
`""`	like `"-`, but producing no hyphen sign (for compund words with hyphen, e.g. `x-""y`).
``"` ``	for German left double quotes (looks like ,,).
`"'`	for German right double quotes.
`"<`	for French left double quotes (similar to <<).
`">`	for French right double quotes (similar to >>).

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.

```
⟨*code⟩
\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 `\l@polish` to see whether we have to do something here.

```
\ifx\l@polish\@undefined
  \@nopatterns{Polish}
  \adddialect\l@polish0\fi
```

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 documentclasses provided with LaTeX.

```
\addto\captionspolish{%
  \def\prefacename{Przedmowa}%
  \def\refname{Literatura}%
  \def\abstractname{Streszczenie}%
  \def\bibname{Bibliografia}%
  \def\chaptername{Rozdzia\l}%
  \def\appendixname{Dodatek}%
  \def\contentsname{Spis tre\'sci}%
  \def\listfigurename{Spis rysunk\'ow}%
  \def\listtablename{Spis tablic}%
  \def\indexname{Indeks}%
  \def\figurename{Rysunek}%
  \def\tablename{Tablica}%
  \def\partname{Cz\eob{}\'s\'c}%
  \def\enclname{Za\l\aob{}cznik}%
  \def\ccname{Kopie:}%
  \def\headtoname{Do}%
  \def\pagename{Strona}%
  \def\seename{Por\'ownaj}%
  \def\alsoname{Por\'ownaj tak\.ze}%
  \def\proofname{Dow\'od}%
  \def\glossaryname{Glossary}% <-- Needs translation
}
```

`\datepolish`

The macro `\datepolish` redefines the command `\today` to produce Polish dates.

```
\def\datepolish{%
  \def\today{\number\day~\ifcase\month\or
  stycznia\or lutego\or marca\or kwietnia\or maja\or czerwca\or lipca\or
  sierpnia\or wrze\'snia\or pa\'zdziernika\or listopada\or grudnia\fi
  \space\number\year}%
}
```

`\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.

```
\initiate@active@char{"}
\addto\extraspolish{\languageshorthands{polish}}
\addto\extraspolish{\bbl@activate{"}}
```

Don't forget to turn the shorthands off again.

```
\addto\noextraspolish{\bbl@deactivate{"}}
```

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 forcasting 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
```

`\sob` 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}
```

`\aob` `\Aob` `\eob` `\Eob` 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}}
```

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

```
\def\spb#1#2#3#4#5{%
  \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
  \leavevmode
  \kern\pl@right\lower\pl@down\box1\kern-\pl@left #1}
```

`\skb` 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}
```

`\textpl` 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.

```
\textpl
```

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

`\telepl` 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 LaTeX are modified. The same is done for plain TeX’s font changing commands.

When `\selectfont` is undefined the current format is spposed to be either plain (based) or LaTeX 2.09.

```
\ifx\selectfont\@undefined
  \ifx\prm\@undefined \addto\rm{\textpl}\else \addto\prm{\textpl}\fi
  \ifx\pit\@undefined \addto\it{\textpl}\else \addto\pit{\textpl}\fi
  \ifx\pbf\@undefined \addto\bf{\textpl}\else \addto\pbf{\textpl}\fi
  \ifx\psl\@undefined \addto\sl{\textpl}\else \addto\psl{\textpl}\fi
  \ifx\psf\@undefined                  \else \addto\psf{\textpl}\fi
  \ifx\psc\@undefined                  \else \addto\psc{\textpl}\fi
  \ifx\ptt\@undefined \addto\tt{\telepl}\else \addto\ptt{\telepl}\fi
\else
```

When `\selectfont` exists we assume LaTeX 2ε.

```
  \expandafter\addto\csname selectfont \endcsname{%
    \csname\f@encoding @pl\endcsname}
\fi
```

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.

```
\expandafter\let\csname T1@pl\endcsname\textpl
```

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.

```
\expandafter\def\csname OT1@pl\endcsname{%
  \long\edef\curr@family{\f@family}%
  \ifx\curr@family\ttdefault
    \telepl
  \else
    \textpl
  \fi}
```

`\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\x{\endgroup
  \def\dq{"}}
\x
```

Now we can define the doublequote macros for diacritics,

```
\declare@shorthand{polish}{"a}{\textormath{\aob}{\ddot a}}
\declare@shorthand{polish}{"A}{\textormath{\Aob}{\ddot A}}
\declare@shorthand{polish}{"c}{\textormath{\'c}{\acute c}}
\declare@shorthand{polish}{"C}{\textormath{\'C}{\acute C}}
\declare@shorthand{polish}{"e}{\textormath{\eob}{\ddot e}}
\declare@shorthand{polish}{"E}{\textormath{\Eob}{\ddot E}}
\declare@shorthand{polish}{"l}{\textormath{\lpb}{\ddot l}}
\declare@shorthand{polish}{"L}{\textormath{\Lpb}{\ddot L}}
\declare@shorthand{polish}{"n}{\textormath{\'n}{\acute n}}
\declare@shorthand{polish}{"N}{\textormath{\'N}{\acute N}}
\declare@shorthand{polish}{"o}{\textormath{\'o}{\acute o}}
\declare@shorthand{polish}{"O}{\textormath{\'O}{\acute O}}
\declare@shorthand{polish}{"s}{\textormath{\'s}{\acute s}}
\declare@shorthand{polish}{"S}{\textormath{\'S}{\acute S}}
```

`\polishrz` `\polishzx` 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.

```
\newcommand*{\polishrz}{%
  \declare@shorthand{polish}{"r}{\textormath{\zkb}{\ddot r}}%
  \declare@shorthand{polish}{"R}{\textormath{\Zkb}{\ddot R}}%
  \declare@shorthand{polish}{"z}{\textormath{\'z}{\acute z}}%
  \declare@shorthand{polish}{"Z}{\textormath{\'Z}{\acute Z}}%
  \declare@shorthand{polish}{"x}{\dq x}%
  \declare@shorthand{polish}{"X}{\dq X}%
  }
\polishrz
```

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.

```
\newcommand*{\polishzx}{%
  \declare@shorthand{polish}{"z}{\textormath{\zkb}{\ddot z}}%
  \declare@shorthand{polish}{"Z}{\textormath{\Zkb}{\ddot Z}}%
  \declare@shorthand{polish}{"x}{\textormath{\'z}{\acute x}}%
  \declare@shorthand{polish}{"X}{\textormath{\'Z}{\acute X}}%
  \declare@shorthand{polish}{"r}{\dq r}%
  \declare@shorthand{polish}{"R}{\dq R}%
  }
```

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

```
\declare@shorthand{polish}{"`}{%
  \textormath{\quotedblbase}{\mbox{\quotedblbase}}}
\declare@shorthand{polish}{"'}{%
  \textormath{\textquotedblright}{\mbox{\textquotedblright}}}
\declare@shorthand{polish}{"<}{%
  \textormath{\guillemotleft}{\mbox{\guillemotleft}}}
\declare@shorthand{polish}{">}{%
  \textormath{\guillemotright}{\mbox{\guillemotright}}}
```

then we define two shorthands to be able to specify hyphenation breakpoints that behavew a little different from `\-`.

```
\declare@shorthand{polish}{"-}{\nobreak-\bbl@allowhyphens}
\declare@shorthand{polish}{""}{\hskip\z@skip}
```

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

```
\declare@shorthand{polish}{"|}{%
  \textormath{\discretionary{-}{}{\kern.03em}}{}}
```

`\mdqon`
`\mdqoff`

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

```
\def\mdqon{\shorthandon{"}}
\def\mdqoff{\shorthandoff{"}}
```

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{polish}
⟨/code⟩
```