churchslavonic package — Church Slavonic Typography in \LaTeX

Aleksandr Andreev and Mike Kroutikov
version v0.2.4
June 24, 2023

Abstract
Package churchslavonic provides fonts, hyphenation patterns and supporting macros to typeset Church Slavonic texts.

Contents
1 How to use the package
   1.1 Options color, gray (= grey), and bw ....................... 2
   1.2 Underscore ............................................. 2

2 Numbers .................................................. 3
   2.1 \cuNum ................................................. 3

3 Dates ..................................................... 3
   3.1 \cuDate ............................................... 4
   3.2 \cuDateJulian ......................................... 4
   3.3 \cuDefineDateFormat .................................. 4
   3.4 \cuUseDateFormat ...................................... 5
   3.5 \cuMonthName ......................................... 5
   3.6 \cuDayName ........................................... 6
   3.7 \cuDayNameAccusative ................................ 6
   3.8 \cuToday .............................................. 6
   3.9 \cuTodayJulian ....................................... 6
   3.10 \cuAsJulian .......................................... 6
   3.11 \cuAsGregorian .................................... 6
Introduction

Church Slavonic (also called Church Slavic, Old Church Slavonic or Old Slavonic; ISO 639-2 code cu) is a literary language used by the Slavic peoples; presently it is used as a liturgical language by the Russian Orthodox Church, other local Orthodox Churches, as well as various Byzantine-Rite Catholic and Old Ritualist communities. The package churchslavonic provides fonts, hyphenation patterns and supporting macros to typeset Church Slavonic texts in \LaTeXe. The package is designed to support Unicode text encoded in UTF-8. Texts encoded in legacy codepages (such as HIP and UCS) may be converted to Unicode using a separate bundle of utilities. See the Slavonic Computing Initiative website for more information. To use the tools in this package, you will need a Unicode-aware \LaTeXe engine such as Xe\LaTeX or Lua\LaTeX.

1 How to use the package

To use the churchslavonic package one needs to include the following declarations into the document preamble:

\usepackage{polyglossia}
\setmainlanguage{churchslavonic}
\usepackage{churchslavonic}

This makes the Church Slavonic hyphenation patterns immediately available. After that, use the standard polyglossia commands to control current language. Church Slavonic fonts are provided by the fonts-churchslavonic package, which should have been installed automatically when you installed this package. See the fonts-churchslavonic documentation for information about fonts.
1.1 Options color, gray (= grey), and bw

These options control what color is actually being used for text coloring commands like \cuKinovar.

- **color** this is the default option; it indicates the original color (a shade of red).
- **gray** (grey) replaces the red color with gray – useful if you are printing on media that does not support color, but is capable of grayscale.
- **bw** replaces the red color with black (effectively turning off commands like \cuKinovar and \cuKinovarColor). Use this option to generate a document that will be printed in black-and-white.

Example:
\usepackage[gray]{churchslavonic}

1.2 Underscore

The underscore symbol (\_ U+005F Low Line) is a valid text symbol in Church Slavonic (it has roughly the same role as the hyphen in English). The package churchslavonic redefines the underscore in a way that it can be directly entered in text mode, for example:

Послѣ́дованїе моле́бнагѡ пѣ́нїѧ ст ҃ы̑мъ мч ҃камъ к ҃_гѡ вѣ́ка, въ С а́нктъ_Петербургстѣй дꙋхо́внѣй а ҆каде́мїи нача̑льствовавшимъ, ᲂу҆чи̑вшимъ и ҆ ᲂу҆чи̑вшимсѧ

**Attention:** if you have an older version of the fontspec package installed on your system, the redefined underscore symbol cannot be used in font names and font options in fontspec commands like \setXXXfont and \newfontfamily. Typically you need to set underscore as the hyphenation character for Church Slavonic fonts: \texttt{HyphenChar=\_}. With older versions of fontspec this will cause errors. This problem exists in \texttt{TeX Live 2013} and in fontspec v2.3c.

We recommend upgrading your \texttt{TeX} distribution to at least \texttt{TeX Live 2015}. Alternatively you can selectively upgrade the fontspec package to version v2.4c or better.

If upgrading is not an option, you can work around this problem by either specifying the hexadecimal code for the hyphenation character: \texttt{Hyphen-Char="005F}, or by declaring all fonts before loading churchslavonic.
2 Numbers

The Church Slavonic numbering system (Cyrillic numerals) is based on Greek Ionian numerals and uses letters as digits. For more information on the implementation, consult the appropriate section of [1].

2.1 \cuNum

Use this command to typeset a Cyrillic numeral. The command takes a single argument that should expand to a number.

\cuNum{1} а
\cuNum{12} ві
\cuNum{123} ркг
\cuNum{1234} ї
\cuNum{10345} ї
\cuNum{12345} ї
\cuNum{123456} ї
\cuNum{800456} ї
\cuNum{1234567} ї
\cuNum{1500567} ї
\cuNum{12345678} ї
\cuNum{123456789} ї

3 Dates

\cuDate{2016-4-21} кв а прілл ї а, лє́та вѕ і
\cuDateJulian{2016-4-21} і прілл ї а, лє́та вѕ і
\cuDate{\cuToday} квд і їн ї а, лє́та вкг

3.1 \cuDate

This command formats the date (according to the current format). The argument is a triplet of numbers YYYY-MM-DD specifying the date. The output will be something like this: κα ληρίλαδι, μήτα λєйі.

Note that YYYY-MM-DD values are not being normalized or interpreted in any way. Thus, it is totally fine to call \cuDate{2016-4-32} even though April 32 is not a valid date. It will be formatted and printed as April 32. This makes it possible to use this macro in a phrase like "the date \cuDate{2016-4-32} is not a valid date in any calendar".
However, if your date format uses \cuDOW (day of the week) or \cuYEARAM (year Anno Mundi), the later quantities are computed by interpreting the date as a Gregorian calendar date. In this case, if the input date is not a valid date, it will be normalized via extrapolation. For example, April 32 will be interpreted as May 2 for the purpose of determining values of the day of the week and year Anno Mundi.

If your format uses \cuDOW or \cuYEARAM, and you specify a date according to the Julian, not Gregorian, calendar, you must use \cuDateJulian to correctly format days of the week and year Anno Mundi.

The best practice is to always use \cuDate with Gregorian calendar dates and use \cuDateJulian with Julian calendar dates regardless of the current date format. This way you can switch the date formatting style without worrying about getting the wrong output.

3.2 \cuDateJulian

Formats the date, just like \cuDate does, but the argument is interpreted as a date on the Julian calendar (“old style”) instead of the Gregorian calendar. This makes a difference only if your format is using symbolic names \cuDOW and/or \cuYEARAM.

3.3 \cuDefineDateFormat

This command allows you to define your own date format. It does not change how \cuDate formats its output (for that, use \cuUseDateFormat). Example:

\begin{verbatim}
\cuDefineDateFormat{long}{%
  \cuDayName{\cuDOW},
  \cuNum{\cuDAY}\textunderscore гѡ
  \cuMonthName{\cuMONTH},~%
  лѣ́та ѿ сотворе́нїѧ мі ́ ра ~%
  \cuNum{\cuYEARAM}%
}\}
\end{verbatim}

defines a format with name \texttt{long}. If we use this format to print the same date as before, we will get: пѧто́къ, к ҃в_гѡ а ҆при ́ лл Ѵа, лѣ́та ѿ сотворе́нїѧ мі ́ ра ҂зфк ҃є.

The following symbolic names can be used when formatting the date:

- \texttt{\cuYEAR} — the year part of a date (a number, like 2016)
- \texttt{\cuYEARAM} — the year Anno Mundi, that is, since the creation of the world according to the Byzantine reckoning (aka “the Byzantine era”; a number, like 7525)
• \texttt{\cuMONTH} — the month part of a date (a number from 1 to 12, with January set to 1)
• \texttt{\cuDAY} — the day of the month
• \texttt{\cuDOW}\(^1\) — the day of the week (number from 0 to 6, where 0 means “Sunday”)
• \texttt{\cuINDICATION} — the indication\(^2\) (a number from 1 to 15)

3.4 \texttt{\cuUseDateFormat}
This command sets the date format to be used by the subsequent \texttt{\cuDate} and \texttt{\cuDateJulian}.

3.5 \texttt{\cuMonthName}
This command expands a numeric argument (month number) into textual representation. It is typically used when defining a date format. For example, a date format named \texttt{default} is defined as:

\begin{verbatim}
\cuDefineDateFormat{default}{
  \cuNum{\cuDAY}~\cuMonthName{\cuMONTH},
  лѣ́та~\cuNum{\cuYEAR}
}
\end{verbatim}

3.6 \texttt{\cuDayName}
Expands a numeric argument into a textual representation of the day of the week in the nominative case.

3.7 \texttt{\cuDayNameAccusative}
Expands a numeric argument into a textual representation of the day of the week in the accusative case.

3.8 \texttt{\cuToday}
This macro expands to a triplet \texttt{YYYY-MM-DD}. The date is generated according to the Gregorian calendar.

\(^1\)If your format uses this value, make sure that you format the date with the correct macro: you must use \texttt{\cuDate} for dates on the Gregorian calendar and \texttt{\cuDateJulian} for dates on the Julian calendar.

\(^2\)See \url{https://en.wikipedia.org/wiki/Indiction}. 
3.9 \texttt{\todayJulian}

This macro expands to a triplet YYYY-MM-DD. The date is generated according to the Julian calendar.

It is a shortcut for \texttt{\AsJulian{\today}}.

3.10 \texttt{\AsJulian}

Converts a date on the Gregorian calendar to a date on the Julian calendar. Input and output use numeric triplet format YYYY-MM-DD.

Useful when the same date needs to be formatted both according to the Gregorian and Julian calendars.

3.11 \texttt{\AsGregorian}

Converts a date according to the Julian calendar to a date according to the Gregorian calendar. Input and output use numeric triplet format YYYY-MM-DD.

4 Kinovar

Printed and hand-written Church Slavonic texts often use color to highlight sectional and paragraph structure and to indicate liturgical rubrics, section names, comments, and marginal notes. The first letter of each paragraph is also often colored red.

4.1 \texttt{\kinovar}

Takes a single argument and prints it using red color. For example, explicitly specifying its argument one gets the expected result:

\begin{verbatim}
\kinovar{ли́къ: гдⷭ҇и поми́лꙋй.}
\end{verbatim}

If one uses the \TeX{} mechanism of implicit argument detection, then the first character of the text after this command will be printed in red. However, a non-trivial feature of this command is that it will also “collect” all of the diacritical marks that belong to this first character, and thus all accents will also be colored in red! Use this command in the indirect parameter mode to print in red the first letter of each paragraph.\footnote{You can also experiment with the \TeX{} command \texttt{\everypar} to automate this, but the success or failure of this technique critically depends on the \TeX{} class used and packages loaded. We found it very fragile and thus the \texttt{churchslavonic} package does not offer any automation for this task. It}
4.2 \texttt{\textbackslash cuKinovarColor}

Switches the current color to red. One would typically use this command inside a group that limits the scope of red text, unless you want all subsequent text to be colored red.

\textit{Information:} The shade of red used by the \texttt{\textbackslash cuKinovar} is declared in the package as a new color called kinovar, and is set to (205, 8, 3) in the RGB color space, which has a hexadecimal representation of \#CC0502.

5 Miscellaneous

5.1 \texttt{\textbackslash cuMarginMark}, \texttt{\textbackslash cuMarginMarkSkip}, \& \texttt{\textbackslash cuMarginMarkText}

The command \texttt{\textbackslash cuMarginMark} is used to place short text in the margin at the same level as the line where the macro is placed. Example:

\begin{verbatim}
\cuMarginMark{\text{k}}\cuKinovar Блгословѝ дꙋш’ моѧ гдⷭ҇а и не забыкй крфъ коззлйш иўв. \cuMarginMark{\text{i}}\cuKinovar Ъчпимйшго кй кззаккйнїа ткд, Ѵћкђлйцѧго кй кддїгн ткд: \cuMarginMark{\text{k}}\cuKinovar Ъзкљлйцѧго ہ Ѵћкђлйцѧ жињкть ткд, Ѵћкђлйцѧго тѧ мдтїв ۰ Ѵћдђлдѧмн: \cuMarginMark{\text{i}}\cuKinovar Ъпоплйлѧгѧго ко Ѵћгїх жәлвѧ ткд, Ѵћкђлдѧ ткдно ткд.
\end{verbatim}

Will result in:

\begin{verbatim}
К Блгословѝ дꙋш’ моѧ гдⷭ҇а и не забыкй крфъ коззлйш иўв. Ъчпимйшго кй кззаккйнїа ткд, Ѵћкђлйцѧ. К го кй кддїгн ткд: Ъзкљлйцѧго ہ Ѵћкђлйцѧ жињкть ткд, Ѵћкђлйцѧго тѧ мдтїв ۰ Ѵћдђлдѧмн: Ъпоплйлѧгѧго ко Ѵћгїх жәлвѧ ткд, Ѵћкђлдѧ ткдно ткд.
\end{verbatim}

\textit{may be easier and more robust to use the Find/Replace functionality embedded in any non-trivial text editor to just automatically place a \textbackslash cuKinovar command before every paragraph of the source text.}
The marginal mark is placed on the right margin for odd pages and on the left margin for even pages (e.g. the mark is placed on the “outer” margin, not the spine margin), which is usually the desired behavior.

The distance between the mark and the text is controlled by the value of \cuMarginMarkSkip. The default is:
\def\cuMarginMarkSkip{0.6em}

To globally customize the font and color of the margin mark use \cuMarginMarkText. For example, to make margin marks appear in red color, redefine \cuMarginMarkText in the preamble of your document like this:
\def\cuMarginMarkText#1{{\textcolor{red}{#1}}}

If you need to change the font/size/color just for a single mark, you can do it directly with \cuMarginMark:
\cuMarginMark{{\tiny *}}

5.2 Superscripts

The \cuSup macro can be used to place arbitrary superscript text above the baseline text, a feature that can be useful in editing Slavonic manuscripts. The macro takes two parameters: the first parameter is the superscripted text, the second parameter is the baseline text. For example:

\text{мол} \cuSup[\text{ва}]{\text{и́т}}

\text{бол} \cuSup[\text{лаже́}]{\text{ложи́}}

The macro accepts an optional parameter raise, which controls the vertical space between the base characters and the superscript. The formatting of the base and superscript texts can also be controlled directly:

\text{пологи́} \cuSup[raise=0.75ex]{\kern2em \cuKinovar{лага́етъ}}{\cuKinovar{ложи́}}

5.3 Dropcaps

The mechanism that the \cuKinovar macro is using to collect all accents (when the argument is specified implicitly) can be useful for many other purposes. One example is to typeset a droplet letter at the beginning of a chapter (this is often used in Church Slavonic texts). For this purpose, the standard \LaTeX package lettrine...
works just fine. The only nuisance is that one has to be careful to pass to \lettrine not just the first letter, but also any diacritical marks that attach to this letter. Naturally, we want to reuse the clever mechanism that \cukinovar uses, and automatically collect the diacritical marks!

Here is an example of how to accomplish this:

\def\culettrine{\lettrine[lines=3,findent=0pt,nindent=0pt]}
\def\culettrine{\cuktokenizeletter\culettrine}
\renewcommand{\LettrineFontHook}{\cukinovarColor}

Once this definition of \culettrine is created (somewhere in the preamble, between the declarations \makeatletter and \makeatother), you can create drop capitals like this:

\culettrine Êже дх̣а г̣лл к̣а н̣ео̣м̣н̣и̣ с̣о̣к̣ер̣ш̣а̣н̣̣г̣а...
Here is a Podchashie: \textchurchslavonic{\cuKruk{𜽰 𜼅}{Тво}} will produce:

Here is a Podchashie: $Тво$

The \cuKrukPara command is used to typeset longer, independent passages of music in Znamenny Notation. The command takes as an argument a line of neumes, followed by $\backslash \backslash$, followed by a line of lyrics. Groups of neumes are separated by spaces and syllables are separated by the hyphen (-) character. Here we use the command to typeset a piece in Znamenny Notation:

\begin{verbatim}
\cuKrukPara{𜽟 𜾆𜼱 𜼇 𜽗𜼆 𜽐𜼱𜼢 𜽖𜼅 𜽜𜼄𜼦 𜾖𜼆𜼤 𜽝𜼅𜼤𜽘 𜾆𜼱 𜼈 𜽞 𜾬 𜼈𜼤 𜽖𜼅 𜾒𜼰𜼺𜼆𜼣 𜾖𜼆𜼤 𜾋𜼅 𜾆𜼄 𜽝𜼄 𜽐𜼅 𜽗𜼄 𜽞 𜽐𜼱 𜼇 𜽜𜼆 𜾆𜼅 𜽞 𜽐𜼱赀 𜼈 𜽗𜼇 𜽐𜼱赀 𜽖𜼅 𜽗𜼅 𜽞 𜽐𜼱赀 𜼇 𜽜𜼇 𜾀 \!

млѧ̀, и҆ ве-се́-лї-емъ вос-по--йте лю́-дї-е, ꙗ҆́-кѡ про-сла́-ви-сѧ.}
\end{verbatim}

Note 1: the number of neume groups needs to equal the number of syllables, otherwise the \cuKrukPara command will return a compile-time error, such as: ! Too many kruk groups.

Note 2: In both the \cuKruk and \cuKrukPara commands, the ~ character can be used when a neume appears without any syllable below it. The command will draw an underline under the relevant neume. Leaving a syllable empty will produce the same result. An empty neume block or the ~ character can also be used to produce a syllable with no neume above it (in this case, the neume block is left blank). To produce a neume with empty space (and no underline) below it, use any other command that is rendered as a space (for example, \thinspace). The following examples demonstrate the functionality:
Note 3: fonts for Znamenny Notation usually provide color information for certain glyphs (such as the cinnabar marks) via data in the COLR and CPAL tables. However, COLR / CPAL technology is presently not supported in XeTeX or LuaTeX. When the churchslavonic package is loaded with the autocolormarks option, the \cuKruk and \cuKrukPara commands will automatically typeset the cinnabar marks in red by internally invoking the \cuKinovar macro. Additionally, if the churchslavonic package is loaded with the gray or bw options, the cinnabar marks will be typeset in grayscale or black, respectively.

Limitations: When a document is processed using XeTeX, attempting to render marks in color breaks normal OpenType glyph positioning rules. Loading the churchslavonic package with the autocolormarks option when XeTeX is used will produce a warning. Presently, you can only render marks in red and maintain correct positioning when using LuaTeX. When a document is processed using LuaTeX, the churchslavonic package by default loads with the autocolormarks option. If automatic coloring is not desired, it can be turned off in this case by loading the churchslavonic package with the noautocolormarks option.

The \cuKruk commands may be nested, which can be used to produce 'explanatory marks' («подобные пометы»):

\cuKruk{\cuKruk{\cuKinovar{\tiny 𜽗}}}
\{\Large 𜽮\}\{\textit{ла}}

6.2 Controlling the Appearance of Znamenny Notation

A number of parameters may be used to control the positioning and appearance of Znamenny neumes:
Specifies the font that is used to typeset the neumes

*specsSpace*

Specifies the amount of spacing around a syllable (default is 0.2em)

topMargin

Specifies the amount of space (margin) above the neumes (default is 0.3em)

kruskRaise

Specifies the amount of space between the neumes and lyrics (default is 1em)

sylRuleHeight

Specifies the thickness of the rule used when no syllable appears below a neume (default is 0.08em)

The parameters are passed as options to the \texttt{\textbackslash cuKruk} and \texttt{\textbackslash cuKrukPara} commands, separated by commas. For example, we can modify the spacing of neumes in the above example:

\texttt{\textbackslash cuKrukPara[krukRaise=1.5em,topMargin=0.6em,sylRuleHeight=0.02em]{ǝ́趔 ǝ́趟 ǝ́ FETCH}}

\begin{verbatim}
Хри-сто́съ ... \textbackslash
Хри-сто́съ ...
\end{verbatim}

References