The mathspec package

Font selection for mathematics with XeLaTeX

version 0.2b

Andrew Gilbert Moschou*
andmos@gmail.com

THURSDAY, 22 DECEMBER 2016

TABLE OF CONTENTS

1 PREAMBLE

This document describes the mathspec package, a package that provides an interface to select ordinary text fonts for typesetting mathematics with XeLaTeX. It relies on fontspec to work and familiarity with fontspec is advised. I thank Will Robertson for his useful advice and suggestions!

The package is developmental and later versions might to be incompatible with this version. This version is incompatible with earlier versions. The package requires at least version 0.9995 of XeTeX.

*v0.2b update by Will Robertson (will.robertson@latex-project.org).

Should you be using this package? If you are using another LaTeX package for some mathematics font, then you should not (unless you know what you are doing). If you want to use Asana Math or Cambria Math (or the final release version of the STIX fonts) then you should be using `unicode-math`.

Some paragraphs in this document are marked ADVANCED. Such paragraphs may be safely ignored by basic users.

2 INTRODUCTION

Since Jonathan Kew released XeTeX, an extension to TeX that permits the inclusion of system wide Unicode fonts and modern font technologies in TeX documents, users have been able to easily typeset documents using readily available fonts such as Hoefler Text and Times New Roman (This document is typeset using Sabon LT Std). Will Robertson's XeLaTeX package `fontspec` provides an automatic font selection process for such fonts. Still, mathematics typesetting has not benefited from this development to the same extent as text typesetting, which is not surprising given the font requirements that are demanded.

Will Robertson has in development a package `unicode-math` that is used to typeset mathematics using some font that has an OpenType mathematics table. Currently, the only two that are available are Cambria Math, a new design, by Microsoft and Asana Math, ultimately based on Hermann Zapf's Palatino, by Apostolos Syropoulos. It is expected that the STIX fonts, based on Times, may also be used.

Thus, it is evident that a book designer is confronted with such narrow variety in mathematics typefaces, even considering already established LaTeX fonts such as AMS Euler (and Computer Modern, of course!) that are dedicated to providing mathematics typefaces; and the book designer is challenged to find a suitable pair of matching typefaces for mathematics and body text.

This package provides mathematics alphabets in any of the same typefaces that are available to XeTeX, using `fontspec` as a back end. Other mathematics symbols such as arrows and operators, whose designs are largely independent of an alphabetic typeface, can be taken from collections like MnSymbol or Computer Modern and are not covered by the scope of this package.

3 IMPLEMENTATION

`\usepackage[`⟨*mathspec and fontspec options*⟩`]{mathspec}`

To use the package, put `\usepackage{mathspec}` in the preamble of the document. It is not necessary to put `\usepackage{fontspec}` because `mathspec` will ensure that `fontspec`

is loaded anyway. To load `fontspec` with some options, parse them through `mathspec`, for example, `\usepackage[quiet]{mathspec}` is equivalent to:

```
\usepackage[quiet]{fontspec}
\usepackage{mathspec}
```

Actually, `mathspec` ordinarily loads `fontspec` with the `no-math` option. To cancel this, explicitly use the `math` option: e.g. `\usepackage[math]{mathspec}`.

4 SETTING FONTS

4.1 *Letters and Digits*

`\setmathsfont(`⟨*sets*⟩`)[`⟨*shapes, font features*⟩`]{`⟨*font name*⟩`}`
`\setmathfont(`⟨*sets*⟩`)[`⟨*shapes, font features*⟩`]{`⟨*font name*⟩`}`

This single command is used to entirely describe the desired font for some use. For each character set (Digits, Latin, Greek), there is a (possibly shared) `\setmathsfont` command. The command can be used only in the preamble. Then, there can be only one typeface for each character set in a single document.

For basic use, `(`⟨*sets*⟩`)` is mandatory. `[`⟨*shapes, font features*⟩`]` is optional and may be omitted.

⟨*sets*⟩ A comma separated value list of any of the following: `Digits`, `Latin`, `Greek`.

ADVANCED ⟨*sets*⟩ *may also take the value* `Special`. `(`⟨*sets*⟩`)` *is actually optional and if it is omitted, then* `(Special)` *is asumed. For basic use, it is safe to ignore* `Special`, *so* `(`⟨*sets*⟩`)` *would effectively be a mandatory argument.*

If the value of ⟨*sets*⟩ *is* `Special`, *then* `\eu@ScopeSet@Special[`⟨*font features*⟩`]{`⟨*font name*⟩`}` *is executed, but this command is provided by* `mathspec` *to gobble its arguments (that is, it does nothing). The command* `\eu@ScopeSet@Special` *maybe predefined before* `mathspec` *is loaded (or redefined after* `mathspec` *is loaded) to cause the command to do something else. This effectively means that* `\setmathsfont(`⟨*sets*⟩`)[`⟨*font features*⟩`]{`⟨*font name*⟩`}` *is a* `mathspec` *command, but* `\setmathsfont[`⟨*font features*⟩`]{`⟨*font name*⟩`}` *is an external command which another package might use.*

⟨*shapes*⟩ A comma separated value list of keys and their values. The permitted keys and their values are:

`Uppercase`=`Regular`, `Italic`, `Plain`

`Lowercase`=`Regular`, `Italic`, `Plain`

`Arabic`=`Regular`, `Italic`, `Plain`

To set the symbols in an upright font, choose `Regular`. To set the symbols in an italic font, choose `Italic`. Choose `Plain` to indicate that no font is assignment is to be performed

TABLE 1 The default shapes for the symbol sets.

Set	Key	Default value
`Digits`	`Arabic`	`Regular`
`Latin`	`Uppercase`	`Italic`
`Latin`	`Lowercase`	`Italic`
`Greek`	`Uppercase`	`Regular`
`Greek`	`Lowercase`	`Italic`

by `mathspec`, so that the symbols will remain with their default font (usually Computer Modern). If a value's key is omitted, its default value, which depends on the ⟨*set*⟩, as shown in table 1 is chosen.

The default values are very good and you would be wise not to change the shapes for the `Latin` and `Digit` sets. The `Greek` shapes may be changed, if needed, to adhere to some particular style, for example all `Italic` or all `Regular`.

⟨*font features*⟩ ⟨*font name*⟩ ⟨*font features*⟩ and ⟨*font name*⟩ follow directly from `fontspec` to select the font and its features. See `fontspec`'s documentation for details.

Note that if ⟨*sets*⟩ contains multiple sets, the command is iterated over each set. Each of these iterations share the same ⟨*shapes and font features*⟩ and ⟨*font name*⟩. If you require that the different sets have individual options and font names, they must be specified in separate commands. However, if a value's key is omitted, the individual default values are still applied.

These same command options follow through to the other commands of this package.

4.2 *Symbols*

There is currently no way to set the font for general mathematical symbols such as:

$$=, \times, \mapsto, \partial, \varnothing, \in, \int, \subset$$

You can try the package `MnSymbol`, which has greater (and more uniform) coverage, with the package option `MnSymbol`, e.g. `\usepackage[MnSymbol]{mathspec}`. Note that `MnSymbol` is a third party package by Achim Blumensath, which can only be used if additionally installed.

4.3 *Examples*

The following command:

```
\setmathsfont(Digits,Latin,Greek)
    [Numbers={Lining,Proportional}]{Minion Pro}
```

is equivalent to:

```
\setmathsfont(Digits)[Numbers={Lining,Proportional}]{Minion Pro}
\setmathsfont(Latin)[Numbers={Lining,Proportional}]{Minion Pro}
\setmathsfont(Greek)[Numbers={Lining,Proportional}]{Minion Pro}
```

and indicates that all digits, Latin and Greek characters are to be set in Minion Pro with lining, proportional digits, with digits and uppercase Greek in regular and Latin and lowercase Greek in italic, which are the default shapes.

The following command:

```
\setmathsfont(Digits,Greek)
    [Uppercase=Plain,Lowercase=Regular,Scale=MatchLowercase]
    {GFS Porson}
```

is equivalent to:

```
\setmathsfont(Digits)
    [Uppercase=Plain,Lowercase=Regular,Scale=MatchLowercase]{GFS Porson}
\setmathsfont(Greek)
    [Uppercase=Plain,Lowercase=Regular,Scale=MatchLowercase]{GFS Porson}
```

and indicates that all digits and lowercase Greek charaters are to be set in GFS Porson, scaled so that its x-height matches the main font's, in regular. Uppercase Greek characters remain unchanged (probably from Computer Modern).

4.4 Declaring alphabets

The mathematics alphabets like `\mathrm` and `\mathcal` can be set using these commands.

`\setmathrm[`⟨*font features*⟩`]{`⟨*font name*⟩`}`

This command defines `\mathrm`, `\mathit`, `\mathbf` and the font for operators like sin and log.

`\setmathsf[`⟨*font features*⟩`]{`⟨*font name*⟩`}`

This command defines `\mathsf`.

`\setmathtt[`⟨*font features*⟩`]{`⟨*font name*⟩`}`

This command defines `\mathtt`.

`\setmathcal[`⟨*font features*⟩`]{`⟨*font name*⟩`}`

This command defines `\mathcal`.

`\setmathbb[`⟨*font features*⟩`]{`⟨*font name*⟩`}`

This command defines `\mathbb`.

`\setmathfrak[`⟨*font features*⟩`]{`⟨*font name*⟩`}`

This command defines `\mathfrak`.

4.5 Shorthands

These commands are useful to save typing the same information multiple times, if the same font is used for different purposes.

`\setallmainfonts(`⟨*sets*⟩`)[`⟨*shapes, font features*⟩`]{`⟨*font name*⟩`}`

This command is equivalent to:

`\setmainfont[`⟨*font features*⟩`]{`⟨*font name*⟩`}`
`\setmathsfont(`⟨*sets*⟩`)[`⟨*shapes, font features*⟩`]{`⟨*font name*⟩`}`
`\setmathrm[`⟨*font features*⟩`]{`⟨*font name*⟩`}`

If `(`⟨*sets*⟩`)` is omitted, then `(Digits,Latin,Greek)` is assumed.

`\setprimaryfont[`⟨*shapes, font features*⟩`]{`⟨*font name*⟩`}`

This command is equivalent to:

`\setallmainfonts(Digits,Latin)[`⟨*shapes, font features*⟩`]{`⟨*font name*⟩`}`

`\setallsansfonts[`⟨*font features*⟩`]{`⟨*font name*⟩`}`

This command is equivalent to:

`\setsansfont[`⟨*font features*⟩`]{`⟨*font name*⟩`}`
`\setmathsf[`⟨*font features*⟩`]{`⟨*font name*⟩`}`

`\setallmonofonts`[⟨*font features*⟩]{⟨*font name*⟩}

This command is equivalent to:

`\setmonofont`[⟨*font features*⟩]{⟨*font name*⟩}
`\setmathtt`[⟨*font features*⟩]{⟨*font name*⟩}

4.6 *A further example*

This document used to be typeset with the following:

```
\setmainfont[Numbers=OldStyle]{Sabon LT Std}
\setallsansfonts[Numbers={OldStyle,Proportional},Scale=MatchLowercase]{Candara}
\setallmonofonts[Numbers=OldStyle,Scale=MatchLowercase]{Consolas}
\setmathsfont(Digits,Latin)[Scale=MatchLowercase]{Bembo MT}
\setmathsfont(Greek)[Scale=MatchLowercase]{STIXGeneral}
\setmathrm{Sabon LT Std}
\exchangeforms{phi}
\setminwhitespace[750]
```

The main text font is Sabon LT Std with old style figures. The sans serif font is Candara with old style, proportional figures and the monospaced font is Consolas with old style figures, both scaled to match Sabon LT Std in x-height.

The mathematics font for digits and Latin symbols is Bembo MT and for Greek symbols STIXGeneral, both scaled to match Sabon LT Std in x-height.

The `\mathrm`, `\mathit` and `\mathbf` alphabets are set in Sabon LT Std.

Finally, the normal and variant forms of Greek lowercase phi are exchanged, see section 5 Greek symbols, and the minimum white space on each side of a spaced character is 750 mmu, see section 6 Glyph bounds.

5 GREEK SYMBOLS

For reference, the Greek alphabet and variant letter forms are given in table 2. Despite its name, `\varsigma` ς is not a variant form of `\sigma` σ, it is the final form. Digamma F$\digamma$ is an obsolete letter of the alphabet, originally placed between epsilon Eε and zeta Zζ.

ADVANCED *Initially, `mathspec` defines control sequences for the Greek characters that are absent in Computer Modern, because they are similar in form to Latin characters, from ASCII, thus control sequences like `\Alpha` and `\omicron` are defined. Any variant forms that are absent become equivalent to the normal forms. The meanings of these control sequences are preserved by other control sequences of the same names prefixed with `eu@cm@` (e.g. `\eu@cm@alpha`), which may be recalled after the Greek mathematics font is changed.*

`\exchangeforms`{⟨*list*⟩}

TABLE 2 The Greek alphabet and variant letter forms with control sequences.

Uppercase		Lowercase		Uppercase		Lowercase	
A	\Alpha	α	\alpha	Ξ	\Xi	ξ	\xi
B	\Beta	β	\beta	O	\Omicron	o	\omicron
Γ	\Gamma	γ	\gamma	Π	\Pi	π	\pi
Δ	\Delta	δ	\delta	P	\Rho	ρ	\rho
E	\Epsilon	ε	\epsilon	Σ	\Sigma	ς	\varsigma
Z	\Zeta	ζ	\zeta			σ	\sigma
H	\Eta	η	\eta	T	\Tau	τ	\tau
Θ	\Theta	θ	\theta	Υ	\Upsilon	υ	\upsilon
I	\Iota	ι	\iota	Φ	\Phi	ϕ	\phi
K	\Kappa	κ	\kappa	X	\Chi	χ	\chi
Λ	\Lambda	λ	\lambda	Ψ	\Psi	ψ	\psi
M	\Mu	μ	\mu	Ω	\Omega	ω	\omega
N	\Nu	ν	\nu	F	\Digamma	$\digamma$	\digamma

Normal form	Variant form		Normal form	Variant form	
β	$\varbeta$	\varbeta	π	ϖ	\varpi
ε	ϵ	\varepsilon	ρ	ϱ	\varrho
θ	ϑ	\vartheta	ϕ	φ	\varphi
κ	$\varkappa$	\varkappa	Θ	$\varTheta$	\varTheta

⟨*list*⟩ A comma separated value list of any of the names for the Greek symbols which have variant forms: `beta`, `epsilon`, `theta`, `kappa`, `pi`, `rho`, `phi`, `Theta`.

Some authors might prefer the normal and variant forms of a symbol to be exchanged. For example, I prefer `\phi` to print the orthotic phi 'ϕ' and `\varphi` the cursive phi 'φ', contrary to many text fonts. To exchange the forms of any symbol, include its name in the list.

`\normalisevarforms[`⟨*list*⟩`]`
`\normalizevarforms[`⟨*list*⟩`]`

⟨*list*⟩ As above, a comma separated value list of any of the names for the Greek symbols which have variant forms: `beta`, `epsilon`, `theta`, `kappa`, `pi`, `rho`, `phi`, `Theta`.

If `[`⟨*list*⟩`]` is omitted, then `[beta,epsilon,theta,kappa,pi,rho,phi,Theta]`, that is every such symbol, is assumed.

Since not all fonts contain all variant forms, there might be the odd variant letter that remains in Computer Modern while other Greek letters have changed font. For these characters, the command `\normalisevarforms` (or `\normalizevarforms`) will cause the listed symbols that are absent from the font to be equivalent to their corresponding normal forms. If a symbol is listed which is included in the font, then it is ignored.

Note that this command only adjusts the characters that are absent in the font. Of course, this might mean that, for example, `\vartheta` and `\theta` would print the same glyph, which would be dubious if they are to be used in the same document, with different meanings.

It should be noted that `\epsilon` prints the lunate epsilon 'ϵ' and `\varpepsilon` prints the usual (double loop) epsilon 'ε' in Plain TEX and LATEX. This is in contrast to Unicode text fonts, which contain the lunate style epsilon separately from the Greek alphabet among variant letter forms and symbols (where Unicode assigns 'Greek Lunate Epsilon Symbol'). The definition of `\LaTeXe`, which prints 'LATEX 2_ε' uses `\varepsilon` and care should be taken to ensure that `\LaTeXe` still uses the usual epsilon either by using one of the methods that were presented in this section or by redefining `\LaTeXe` (possibly using my `metalogo` package), because 'LATEX 2_ϵ' is not as good as 'LATEX 2_ε'.

6 GLYPH BOUNDS

In using this package to set text fonts for mathematics, there are issues with glyph bounds (similar to italic correction), as illustrated in the equations:

```
\begin{align}
 f(x) &= \sum_{n = -\infty}^\infty c_{n}e^{jnx}            \\
c_{n} &= \frac {1}{2}\int_  {-}^{}f(x)e^{-jnx}\,\mathrm dx
\end{align}
```

$$f(x) = \sum_{n=-\infty}^{\infty} c_n e^{jnx} \tag{1}$$

$$c_n = \frac{1}{2\pi} \int_{-\pi}^{\pi} f(x) e^{-jnx} \, \mathrm{d}x \tag{2}$$

where the function f is to close the the parenthesis (and the exponent jnx is too close to its base e that there is in fact a collision. The reason is that the font has metrics that are suitable for use in text, but not for mathematics.

Comparing the alphabets typeset in Computer Modern italic, in table 3, it is shown that many characters have greater space around them in the mathematics version, which does not exist in the text version.

TABLE 3 Computer Modern Italic in text and mathematics.

Text	*abcdefghijklmnopqrstuvwxyz*
Mathematics	$abcdefghijklmnopqrstuvwxyz$

TABLE 4 All valid characters for use with " and \"…".

0123456789
$ABCDEFGHIJKLMNOPQRSTUVWXYZ$
$abcdefghijklmnopqrstuvwxyz$
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩF
$\alpha\beta\gamma\delta\varepsilon\zeta\eta\theta\iota\kappa\lambda\mu\nu\xi o\pi\rho\varsigma\sigma\tau\upsilon\phi\chi\psi\omega_F$
$\eth\vartheta\varphi\varpi\varkappa\varrho\epsilon\Theta$

"⟨*character*⟩

It is necessary to indicate exactly where additional space needs to be inserted. By putting " before a character, the character will be typeset with additional space inserted on both sides of it.

\"⟨*string*⟩"

For multiple adjacent characters, each need not have a " in front of it, but it the characters may be surrounded by \" before and " after. That is, for example \"abcde" is equivalent to "a"b"c"d"e.

Note that ⟨*character*⟩ must be one of the symbols that are listed in table 4. ⟨*string*⟩ must one or more such characters adjacent to each other. So \"xyz" and \sin"x is okay, but \"\sin x" is not because \sin is not a valid character. Also note that with Greek symbols, they can only be used if typed literally, not via control sequences, so \cos" is okay but \cos"\theta is not. Use of control sequences for Greek symbols is still perfectly acceptable because the spacings are built into the defintions of each (for example \alpha is defined to expand to {" } automatically).

Use of " and \" in mathematics mode does not interfere with their use in text mode for quotation marks and umlauts or diæreses: ‘’’ coöperates, even with “Mapping=tex-text”.

\setminwhitespace[⟨*number*⟩]

Use this command to change the minimum allowed white space around such a spaced character. The unit of ⟨*number*⟩ is millimu (mmu), that is $\frac{1}{1000}$ mu. Recall 18 mu = 1 em.

TABLE 5 Libre Baskerville in text and spaced mathematics.

Text	*abcdefghijklmnopqrstuvwxyz*
Mathematics	$abcdefghijklmnopqrstuvwxyz$

The default ⟨*number*⟩ is `500` which corresponds to $500\,\mathrm{mmu} = \frac{500}{1000}\,\mathrm{mu} = 0.5\,\mathrm{mu} = \frac{1}{36}\,\mathrm{em}$.

Note that this value corresponds to the inserted spaces on both sides for each spaced character. If two adjacent characters are spaced, then the total minimum white space between the two characters is twice this value. The effect is shown in table 5.

Now, the spacing in equations (1, 2) are improved by:

```
\begin{align}
"f\left("x\right) &= \sum_{"n = -\infty}^\infty"c_{"n}"e^{\"jnx"} \\
"c_{"n} &= \frac {1}{2"}\int_  {-"}^""f\bigl("x\bigr)"e^{-\"jnx"}\,\mathrm d"x
\end{align}
```

$$f\left(x\right) = \sum_{n=-\infty}^{\infty} c_n e^{jnx} \tag{3}$$

$$c_n = \frac{1}{2\pi}\int_{-\pi}^{\pi} f(x) e^{-jnx}\,\mathrm{d}x \tag{4}$$

Also note that the spacing is improved by replacing `(x)` with `\bigl(x\bigr)` (which you should be doing anyway).

7 COMPATABILITY

If `amsmath` is required, it must be loaded earlier than `mathspec`.

8 THE PACKAGE

The package style file is printed in this section.

```
\NeedsTeXFormat{LaTeX2e}[2005/12/01]
\ProvidesPackage{mathspec}
  [2016/12/22 v0.2b LaTeX Package (Mathematics font selection for XeLaTeX)]

%% Requirements:
\RequirePackage{etoolbox}
\RequirePackage{amstext}

\RequirePackage{ifxetex}
\RequireXeTeX
\ifcsundef{XeTeXglyphbounds}
  {\PackageError{mathspec}
     {mathspec requires a more recent version of XeTeX}
     {Your current vesion of XeTeX is \the\XeTeXversion\XeTeXrevision.\MessageBreak
      Update your version of XeTeX to at least 0.9995.}}
  {\relax}

%% Booleans are created automatically on demand.
\newcommand\eu@booltrue[1]{\providebool{#1}\booltrue{#1}}
\newcommand\eu@gbooltrue[1]{\providebool{#1}\global\booltrue{#1}}
\newcommand\eu@boolfalse[1]{\providebool{#1}\boolfalse{#1}}
\newcommand\eu@ifbool[1]{\providebool{#1}\ifbool{#1}}

%% Some extras:
%%   \eu@ifbooltrue{<bool>}{<true>}
%%   \eu@ifboolfalse{<bool>}{<false>}
%%   \eu@ifsomebooltrue{<bool_1>,...,<bool_n>}{<true>}
%%                                          (if <bool_i> is true for some i)
%%   \eu@ifnumis{x}{x_1,...,x_n}{<true>}{<false>} (if x = x_i for some i)
%%   \eu@ifnumin{x}{a,b}{<true>}{<false>}         (if a   x   b)
\newcommand\eu@ifbooltrue[2]{\eu@ifbool{#1}{#2}{\relax}}
\newcommand\eu@ifboolfalse[2]{\eu@ifbool{#1}{\relax}{#2}}
\newcommand\eu@ifsomebooltrue[1]{%
  \eu@boolfalse{temp}%
  \def\do##1{\eu@ifbooltrue{##1}{\eu@booltrue{temp}}}%
  \docsvlist{#1}%
  \eu@ifbooltrue{temp}}
\newcommand\eu@ifnumis[2]{%
  \providebool{temp}%
  \boolfalse{temp}%
  \def\do##1{\ifnumcomp{#1}{=}{##1}{\booltrue{temp}}{\relax}}%
  \docsvlist{#2}%
  \ifbool{temp}}
\newcommand\eu@ifnumin[2]{\@eu@ifnumin{#1}#2\@nil}
\def\@eu@ifnumin #1#2,#3\@nil{%
```

```
  \providebool{temp}%
  \booltrue{temp}%
  \ifnumcomp{#1}{<}{#2}{\boolfalse{temp}}{\relax}%
  \ifnumcomp{#3}{<}{#1}{\boolfalse{temp}}{\relax}%
  \ifbool{temp}}

%% Options:
\DeclareOption{normalskips}
  {\PackageWarning{mathspec}
     {Package option `normalskips' is deprecated}}
\def\eu@zf@math{no-math}
\DeclareOption{math}{\def\eu@zf@math{math}}
\DeclareOption{no-math}{\relax}
\DeclareOption{MnSymbol}{\eu@booltrue{MnSymbol}}
\DeclareOption*{\PassOptionsToPackage{\CurrentOption}{fontspec}}
\ProcessOptions\relax

%% Requires packages:
%%   fontspec, xkeyval, mathstyle, etoolbox and maybe MnSymbol
\RequirePackage[\eu@zf@math]{fontspec}[2008/08/09]
\RequirePackage{xkeyval}
\eu@ifbooltrue{MnSymbol}{\RequirePackage{MnSymbol}}

\providecommand\currentmathstyle{\relax}

%% @-namespace fontspec variable:
\ExplSyntaxOn
\def\eu@enc{\g_fontspec_encoding_tl}
\ExplSyntaxOff
%% This is needed to transition to the TU encoding instead of EU1 (WSPR)

%% Deprecated commands in fontspec:
\ExplSyntaxOn
\tl_set:Nn \zf@enc { \g_fontspec_encoding_tl }
\cs_set:Npn \zf@fontspec #1 #2
 {
  \fontspec_select:nn {#1} {#2}
  \tl_set:Nn \zf@family { \l_fontspec_family_tl }
  \tl_set:Nn \zf@basefont { \l_fontspec_font }
 }
\ExplSyntaxOff
%% These are still defined in fontspec at time of writing but maybe be removed in the future. (WSPR)

%% Define the undefined Greek letters. Include all variant forms (same as normal
%% forms if variant forms are unavailable). Latin F for digamma is better than
%% nothing. If Digamma is available, then Capital and Lowercase the are same.
\DeclareMathSymbol{\Alpha}{\mathalpha}{operators}{"41}
\DeclareMathSymbol{\Beta}{\mathalpha}{operators}{"42}
\DeclareMathSymbol{\Epsilon}{\mathalpha}{operators}{"45}
```

```
\DeclareMathSymbol{\Zeta}{\mathalpha}{operators}{"5A}
\DeclareMathSymbol{\Eta}{\mathalpha}{operators}{"48}
\DeclareMathSymbol{\Iota}{\mathalpha}{operators}{"49}
\DeclareMathSymbol{\Kappa}{\mathalpha}{operators}{"4B}
\DeclareMathSymbol{\Mu}{\mathalpha}{operators}{"4D}
\DeclareMathSymbol{\Nu}{\mathalpha}{operators}{"4E}
\DeclareMathSymbol{\Omicron}{\mathalpha}{operators}{"4F}
\DeclareMathSymbol{\Rho}{\mathalpha}{operators}{"50}
\DeclareMathSymbol{\Tau}{\mathalpha}{operators}{"54}
\DeclareMathSymbol{\Chi}{\mathalpha}{operators}{"58}
\DeclareMathSymbol{\omicron}{\mathord}{letters}{"6F}
\let\varbeta\beta
\ifdef{\varkappa}{\relax}
  {\let\varkappa\kappa}
\ifdef{\varTheta}{\let\eu@cm@varTheta\varTheta}
  {\relax}
\let\varTheta\Theta
\ifdef{\digamma}{\let\Digamma\digamma}
  {\DeclareMathSymbol{\Digamma}{\mathalpha}{operators}{"46}
   \DeclareMathSymbol{\digamma}{\mathord}{letters}{"46}}

%% Preserve old definitions of all Greek letters. \eu@cm@alpha etc.
\def\do#1{\csletcs{eu@cm@#1}{#1}}
\docsvlist{
  Alpha,Beta,Gamma,Delta,Epsilon,Zeta,Eta,Theta,Iota,Kappa,Lambda,Mu,Nu,
  Xi,Omicron,Pi,Rho,Sigma,Tau,Upsilon,Phi,Chi,Psi,Omega,Digamma,alpha,
  beta,gamma,delta,epsilon,zeta,eta,theta,iota,kappa,lambda,mu,nu,xi,
  omicron,pi,rho,varsigma,sigma,tau,upsilon,phi,chi,psi,omega,digamma,
  varTheta,varbeta,varepsilon,vartheta,varkappa,varpi,varrho,varphi}
%% varTheta is done separately because amsmath defines it differently.

%% Can define/redefine any command using the syntax of
%% \newcommand/\renewcommand without error. Some helpers.
\newcommand\ernewcommand[1]{\ifdef{#1}{\renewcommand{#1}}{\newcommand{#1}}}
\providecommand\expanded[1]{\edef\@tempa{#1}\@tempa}
\newcommand\eu@setkeys[3][]{
  \ifblank{#1}{\relax}{\presetkeys[eu]{#2}{#1}{}}
  \expanded{\noexpand\setkeys*[eu]{#2}{\expandonce#3\@empty}}}
\newcommand\eu@fontspec[2]{
  \expanded{\noexpand\zf@fontspec{\expandonce#1\@empty}{\expandonce#2\@empty}}}

%% Deprecated commands.
%%   \+:              Use \"..." or " instead.
%%   \plaindigits:    Digits are no longer selected automatically from either
%%                    the Latin or the Greek mathematics font. Instead, they
%%                    must explicitely be stated. If you want plain digits, just
%%                    don't say you want digits.
%5   \normalvarforms: Now called \normalisevarforms
%%   \varforms:       Now called \exchangeforms
```

```
%% Eventually, documents that use these will compile with error.
\newcommand\+{
  \PackageWarning{mathspec}
    {\protect\+\space is deprecated, recommend to use\MessageBreak\protect\"..."
     or "}
  \eu@plus}
\newcommand\setsansfonts{
  \PackageWarning{mathspec}
    {\protect\setsansfonts\space is deprecated, recommend to use \protect
     \setallsansfonts.}
  \setallsansfonts}
\@onlypreamble\setsansfonts
\newcommand\setmonofonts{
  \PackageWarning{mathspec}
    {\protect\setmonofonts\space is deprecated, recommend to use \protect
     \setallmonofonts.}
  \setallsansfonts}
\@onlypreamble\setmonofonts
\newcommand\plaindigits{
  \PackageWarning{mathspec}{\protect\plaindigits\space is deprecated}}
\@onlypreamble\plaindigits
\newcommand\normalvarforms{
  \PackageWarning{mathspec}
    {\protect\normalvarforms\space is deprecated, recommend to use
     \protect\normalisevarforms\space or \protect\normalizevarforms}
  \normalisevarforms}
\@onlypreamble\normalvarforms
\newcommand\varforms{
  \PackageWarning{mathspec}
    {\protect\varforms\space is deprecated, recommend to use
     \protect\exchangeforms}
  \exchangeforms}
\@onlypreamble\varforms

%% I don't want this clogging up my sty file. It will be gone eventually.
\def\@ifnext#1#2#3{%
  \let\@tempd=#1%
  \def\@tempa{#2}%
  \def\@tempb{#3}%
  \futurelet\@tempc\@ifnexta}
\def\@ifnexta{%
  \ifx\@tempc\@tempd%
  \let\@tempb\@tempa%
  \fi\@tempb}
\def\eu@DeclareRobustCommand{\@star@or@long\eu@declare@robustcommand}
\def\eu@declare@robustcommand#1{%
  \ifx#1\@undefined\else\ifx#1\relax\else
    \@latex@info{Redefining \string#1}%
  \fi\fi
```

```
  \edef\reserved@a{\string#1}%
  \def\reserved@b{#1}%
  \edef\reserved@b{\expandafter\strip@prefix\meaning\reserved@b}%
  \edef#1{%
    \ifx\reserved@a\reserved@b
      \noexpand\x@protect
      \noexpand#1%
    \fi
    \noexpand\protect
    \expandafter\noexpand\csname\expandafter\@gobble\string#1\endcsname}%
  \let\@ifdefinable\@rc@ifdefinable
  \expandafter\eu@new@command\csname\expandafter\@gobble\string#1\endcsname}
\def\eu@new@command#1{\eu@@testopt{\eu@@newcommand#1}0}
\def\eu@@newcommand#1[#2]{\@ifnext[{\@xargdef#1[#2]}{\@argdef#1[#2]}}
\long\def\eu@@testopt#1#2{\@ifnext[{#1}{#1[{#2}]}}
\eu@DeclareRobustCommand\eu@plus[1][]{}

%% The main user command (comes in two spellings)
%%   \setmathsfont(<sets>)[<shapes, font features>]{<font name>}
%%   \setmathfont(<sets>)[<shapes, font features>]{<font name>}
%%
%%     <set> is a CSV list of any of: Special, Latin, Greek, Digits, Symbols. If
%%         (<set>) is omitted, then (Special) is assumed. Special is provided to
%%         hook to some external code (e.g. potentially unicode-math). That is,
%%         for mathspec purposes, (<sets>) is mandatory. Omit it so that it
%%         behaves like a different command. See below.
%%     <shapes> is the keyval list of font shapes for the subset of <set>.
%%         Valid keys are: Uppercase, Lowercase, Arabic.
%%         Valid values are: Regular, Italic, Plain.
%%     <font features>, <font name> follow directly from fontspec.
\providecommand\setmathsfont{\eu@setmathsfont}
\let\setmathfont\setmathsfont
\newcommand\eu@setmathsfont{
  \@ifnextchar(
    {\@eu@setmathsfont}
    {\@eu@setmathsfont(Special)}}
\def\@eu@setmathsfont(#1){
  \edef\eu@setmathsfont@Set{#1}
  \@@eu@setmathsfont}
\newcommand\@@eu@setmathsfont[2][]{
  \@for\i@for:=\eu@setmathsfont@Set\do
    {\csname eu@ScopeSet@\i@for\endcsname[#1]{#2}}}

%% The FIRST branch \setmathsfont(Special)
%%   Predefine this command before mathspec is loaded (or redefine it) to get
%%
%%       \setmathsfont[<font features>]{<font name>}
%%
%%   to do something else.
```

```
%%
%% Essentially, \setmathsfont(<sets>)[<font features>]{<font name>} is a mathspec
%% command, but \setmathsfont[<font features>]{<font name>} is an external
%% command.
\providecommand\eu@ScopeSet@Special[2][]{\relax}

%% The SECOND branch \setmathsfont(Digits)
\newcommand\eu@ScopeSet@Digits[2][]{
  \eu@setkeys[Arabic=Regular]{Digits}{#1}
  \eu@fontspec{\XKV@rm}{#2}
 \ifcase\eu@DigitsArabic@@value %% If Digits Regular
    \ernewcommand\eu@DigitsArabic@symfont{Digits:m:n}
    \let\eu@Digitsmathsfont\zf@family
    \eu@booltrue{Digits}
  \or %% If Digits Italic
    \ernewcommand\eu@DigitsArabic@symfont{Digits:m:it}
    \let\eu@Digitsmathsfont\zf@family
    \eu@booltrue{Digits}
  \or %% If Digits Plain
    \eu@boolfalse{Digits}
  \fi
  \eu@ifsomebooltrue{Digits}
    {\DeclareSymbolFont{Digits:m:n}{\eu@enc}{\eu@Digitsmathsfont}{m}{n}}
  \eu@ifbooltrue{Digits}
    {\fontfamily\eu@Digitsmathsfont\selectfont
     \DeclareMathSymbol{0}{\mathord}{\eu@DigitsArabic@symfont}{`0}
     \DeclareMathSymbol{1}{\mathord}{\eu@DigitsArabic@symfont}{`1}
     \DeclareMathSymbol{2}{\mathord}{\eu@DigitsArabic@symfont}{`2}
     \DeclareMathSymbol{3}{\mathord}{\eu@DigitsArabic@symfont}{`3}
     \DeclareMathSymbol{4}{\mathord}{\eu@DigitsArabic@symfont}{`4}
     \DeclareMathSymbol{5}{\mathord}{\eu@DigitsArabic@symfont}{`5}
     \DeclareMathSymbol{6}{\mathord}{\eu@DigitsArabic@symfont}{`6}
     \DeclareMathSymbol{7}{\mathord}{\eu@DigitsArabic@symfont}{`7}
     \DeclareMathSymbol{8}{\mathord}{\eu@DigitsArabic@symfont}{`8}
     \DeclareMathSymbol{9}{\mathord}{\eu@DigitsArabic@symfont}{`9}}}

%% The THIRD branch \setmathsfont(Latin)
\newcommand\eu@ScopeSet@Latin[2][]{
  \eu@setkeys[Uppercase=Italic,Lowercase=Italic]{Latin}{#1}
  \eu@fontspec{\XKV@rm}{#2}
  \ifcase\eu@LatinUppercase@@value %% If Latin Uppercase Regular
    \ernewcommand\eu@LatinUppercase@symfont{Latin:m:n}
    \let\eu@Latinmathsfont\zf@family
    \eu@booltrue{LatinUppercase}
  \or %% If Latin Uppercase Italic
    \ernewcommand\eu@LatinUppercase@symfont{Latin:m:it}
    \let\eu@Latinmathsfont\zf@family
    \eu@booltrue{LatinUppercase}
  \or %% If Latin Uppercase Plain
```

```
    \eu@boolfalse{LatinUppercase}
  \fi
  \ifcase\eu@LatinLowercase@@value %% If Latin Lowercase Regular
    \ernewcommand\eu@LatinLowercase@symfont{Latin:m:n}
    \let\eu@Latinmathsfont\zf@family
    \eu@booltrue{LatinLowercase}
  \or %% If Latin Lowercase Italic
    \ernewcommand\eu@LatinLowercase@symfont{Latin:m:it}
    \let\eu@Latinmathsfont\zf@family
    \eu@booltrue{LatinLowercase}
  \or %% If Latin Lowercase Plain
    \eu@boolfalse{LatinLowercase}
  \fi
  \eu@ifsomebooltrue{LatinUppercase,LatinLowercase}
    {\DeclareSymbolFont{Latin:m:n}{\eu@enc}{\eu@Latinmathsfont}{m}{n}
     \DeclareSymbolFont{Latin:m:it}{\eu@enc}{\eu@Latinmathsfont}{m}{it}}
  \eu@ifbooltrue{LatinUppercase}
    {\fontfamily\eu@Latinmathsfont\selectfont
     \DeclareMathSymbol{A}{\mathalpha}{\eu@LatinUppercase@symfont}{`A}
     \DeclareMathSymbol{B}{\mathalpha}{\eu@LatinUppercase@symfont}{`B}
     \DeclareMathSymbol{C}{\mathalpha}{\eu@LatinUppercase@symfont}{`C}
     \DeclareMathSymbol{D}{\mathalpha}{\eu@LatinUppercase@symfont}{`D}
     \DeclareMathSymbol{E}{\mathalpha}{\eu@LatinUppercase@symfont}{`E}
     \DeclareMathSymbol{F}{\mathalpha}{\eu@LatinUppercase@symfont}{`F}
     \DeclareMathSymbol{G}{\mathalpha}{\eu@LatinUppercase@symfont}{`G}
     \DeclareMathSymbol{H}{\mathalpha}{\eu@LatinUppercase@symfont}{`H}
     \DeclareMathSymbol{I}{\mathalpha}{\eu@LatinUppercase@symfont}{`I}
     \DeclareMathSymbol{J}{\mathalpha}{\eu@LatinUppercase@symfont}{`J}
     \DeclareMathSymbol{K}{\mathalpha}{\eu@LatinUppercase@symfont}{`K}
     \DeclareMathSymbol{L}{\mathalpha}{\eu@LatinUppercase@symfont}{`L}
     \DeclareMathSymbol{M}{\mathalpha}{\eu@LatinUppercase@symfont}{`M}
     \DeclareMathSymbol{N}{\mathalpha}{\eu@LatinUppercase@symfont}{`N}
     \DeclareMathSymbol{O}{\mathalpha}{\eu@LatinUppercase@symfont}{`O}
     \DeclareMathSymbol{P}{\mathalpha}{\eu@LatinUppercase@symfont}{`P}
     \DeclareMathSymbol{Q}{\mathalpha}{\eu@LatinUppercase@symfont}{`Q}
     \DeclareMathSymbol{R}{\mathalpha}{\eu@LatinUppercase@symfont}{`R}
     \DeclareMathSymbol{S}{\mathalpha}{\eu@LatinUppercase@symfont}{`S}
     \DeclareMathSymbol{T}{\mathalpha}{\eu@LatinUppercase@symfont}{`T}
     \DeclareMathSymbol{U}{\mathalpha}{\eu@LatinUppercase@symfont}{`U}
     \DeclareMathSymbol{V}{\mathalpha}{\eu@LatinUppercase@symfont}{`V}
     \DeclareMathSymbol{W}{\mathalpha}{\eu@LatinUppercase@symfont}{`W}
     \DeclareMathSymbol{X}{\mathalpha}{\eu@LatinUppercase@symfont}{`X}
     \DeclareMathSymbol{Y}{\mathalpha}{\eu@LatinUppercase@symfont}{`Y}
     \DeclareMathSymbol{Z}{\mathalpha}{\eu@LatinUppercase@symfont}{`Z}}
  \eu@ifbooltrue{LatinLowercase}
    {\fontfamily\eu@Latinmathsfont\selectfont
     \DeclareMathSymbol{a}{\mathalpha}{\eu@LatinLowercase@symfont}{`a}
     \DeclareMathSymbol{b}{\mathalpha}{\eu@LatinLowercase@symfont}{`b}
     \DeclareMathSymbol{c}{\mathalpha}{\eu@LatinLowercase@symfont}{`c}
```

```
      \DeclareMathSymbol{d}{\mathalpha}{\eu@LatinLowercase@symfont}{`d}
      \DeclareMathSymbol{e}{\mathalpha}{\eu@LatinLowercase@symfont}{`e}
      \DeclareMathSymbol{f}{\mathalpha}{\eu@LatinLowercase@symfont}{`f}
      \DeclareMathSymbol{g}{\mathalpha}{\eu@LatinLowercase@symfont}{`g}
      \DeclareMathSymbol{h}{\mathalpha}{\eu@LatinLowercase@symfont}{`h}
      \DeclareMathSymbol{i}{\mathalpha}{\eu@LatinLowercase@symfont}{`i}
      \DeclareMathSymbol{j}{\mathalpha}{\eu@LatinLowercase@symfont}{`j}
      \DeclareMathSymbol{k}{\mathalpha}{\eu@LatinLowercase@symfont}{`k}
      \DeclareMathSymbol{l}{\mathalpha}{\eu@LatinLowercase@symfont}{`l}
      \DeclareMathSymbol{m}{\mathalpha}{\eu@LatinLowercase@symfont}{`m}
      \DeclareMathSymbol{n}{\mathalpha}{\eu@LatinLowercase@symfont}{`n}
      \DeclareMathSymbol{o}{\mathalpha}{\eu@LatinLowercase@symfont}{`o}
      \DeclareMathSymbol{p}{\mathalpha}{\eu@LatinLowercase@symfont}{`p}
      \DeclareMathSymbol{q}{\mathalpha}{\eu@LatinLowercase@symfont}{`q}
      \DeclareMathSymbol{r}{\mathalpha}{\eu@LatinLowercase@symfont}{`r}
      \DeclareMathSymbol{s}{\mathalpha}{\eu@LatinLowercase@symfont}{`s}
      \DeclareMathSymbol{t}{\mathalpha}{\eu@LatinLowercase@symfont}{`t}
      \DeclareMathSymbol{u}{\mathalpha}{\eu@LatinLowercase@symfont}{`u}
      \DeclareMathSymbol{v}{\mathalpha}{\eu@LatinLowercase@symfont}{`v}
      \DeclareMathSymbol{w}{\mathalpha}{\eu@LatinLowercase@symfont}{`w}
      \DeclareMathSymbol{x}{\mathalpha}{\eu@LatinLowercase@symfont}{`x}
      \DeclareMathSymbol{y}{\mathalpha}{\eu@LatinLowercase@symfont}{`y}
      \DeclareMathSymbol{z}{\mathalpha}{\eu@LatinLowercase@symfont}{`z}}}

%% The FOURTH branch \setmathsfont(Greek)
\newcommand\eu@ScopeSet@Greek[2][]{
  \eu@setkeys[Uppercase=Regular,Lowercase=Italic]{Greek}{#1}
  \eu@fontspec{\XKV@rm}{#2}
  \ifcase\eu@GreekUppercase@@value %% If Greek Uppercase Regular
    \ernewcommand\eu@GreekUppercase@symfont{Greek:m:n}
    \let\eu@Greekmathsfont\zf@family
    \eu@booltrue{GreekUppercase}
  \or %% If Greek Uppercase Italic
    \ernewcommand\eu@GreekUppercase@symfont{Greek:m:it}
    \let\eu@Greekmathsfont\zf@family
    \eu@booltrue{GreekUppercase}
  \or %% If Greek Uppercase Plain
    \eu@boolfalse{GreekUppercase}
  \fi
  \ifcase\eu@GreekLowercase@@value %% If Greek Lowercase Regular
    \ernewcommand\eu@GreekLowercase@symfont{Greek:m:n}
    \let\eu@Greekmathsfont\zf@family
    \eu@booltrue{GreekLowercase}
  \or %% If Greek Lowercase Italic
    \ernewcommand\eu@GreekLowercase@symfont{Greek:m:it}
    \let\eu@Greekmathsfont\zf@family
    \eu@booltrue{GreekLowercase}
  \or %% If Greek Lowercase Plain
    \eu@boolfalse{GreekLowercase}
```

```
  \fi
  \eu@ifsomebooltrue{GreekUppercase,GreekLowercase}
    {\DeclareSymbolFont{Greek:m:n}{\eu@enc}{\eu@Greekmathsfont}{m}{n}
     \DeclareSymbolFont{Greek:m:it}{\eu@enc}{\eu@Greekmathsfont}{m}{it}}
  \eu@ifbooltrue{GreekUppercase}
    {\fontfamily\eu@Greekmathsfont\selectfont
     \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}[\Alpha]
     \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}[\Beta]
     \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}[\Gamma]
     \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}[\Delta]
     \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}[\Epsilon]
     \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}[\Zeta]
     \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}[\Eta]
     \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}[\Theta]
     \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}[\Iota]
     \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}[\Kappa]
     \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}[\Lambda]
     \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}[\Mu]
     \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}[\Nu]
     \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}[\Xi]
     \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}[\Omicron]
     \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}[\Pi]
     \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}[\Rho]
     \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}[\Sigma]
     \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}[\Tau]
     \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}[\Upsilon]
     \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}[\Phi]
     \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}[\Chi]
     \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}[\Psi]
     \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}[\Omega]
     \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}[\Digamma]
     \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}[\varTheta]
     \eu@fixgreekcs {}{Alpha}
     \eu@fixgreekcs {}{Beta}
     \eu@fixgreekcs {}{Gamma}
     \eu@fixgreekcs {}{Delta}
     \eu@fixgreekcs {}{Epsilon}
     \eu@fixgreekcs {}{Zeta}
     \eu@fixgreekcs {}{Eta}
     \eu@fixgreekcs {}{Theta}
     \eu@fixgreekcs {}{Iota}
     \eu@fixgreekcs {}{Kappa}
     \eu@fixgreekcs {}{Lambda}
     \eu@fixgreekcs {}{Mu}
     \eu@fixgreekcs {}{Nu}
     \eu@fixgreekcs {}{Xi}
     \eu@fixgreekcs {}{Omicron}
     \eu@fixgreekcs {}{Pi}
     \eu@fixgreekcs {}{Rho}
```

```
      \eu@fixgreekcs {}{Sigma}
      \eu@fixgreekcs {}{Tau}
      \eu@fixgreekcs {}{Upsilon}
      \eu@fixgreekcs {}{Phi}
      \eu@fixgreekcs {}{Chi}
      \eu@fixgreekcs {}{Psi}
      \eu@fixgreekcs {}{Omega}
      \eu@fixgreekcs {}{Digamma}
      \eu@fixgreekcs {}{varTheta}}
  \eu@ifbooltrue{GreekLowercase}
    {\fontfamily\eu@Greekmathsfont\selectfont
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\alpha]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\beta]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\gamma]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\delta]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\epsilon]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\zeta]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\eta]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\theta]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\iota]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\kappa]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\lambda]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\mu]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\nu]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\xi]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\omicron]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\pi]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\rho]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\varsigma]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\sigma]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\tau]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\upsilon]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\phi]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\chi]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\psi]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\omega]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\digamma]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\varbeta]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\varepsilon]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\vartheta]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\varkappa]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\varpi]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\varrho]
      \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\varphi]
      \eu@fixgreekcs {}{alpha}
      \eu@fixgreekcs {}{beta}
      \eu@fixgreekcs {}{gamma}
      \eu@fixgreekcs {}{delta}
      \eu@fixgreekcs {}{epsilon}
```

```
      \eu@fixgreekcs {}{zeta}
      \eu@fixgreekcs {}{eta}
      \eu@fixgreekcs {}{theta}
      \eu@fixgreekcs {}{iota}
      \eu@fixgreekcs {}{kappa}
      \eu@fixgreekcs {}{lambda}
      \eu@fixgreekcs {}{mu}
      \eu@fixgreekcs {}{nu}
      \eu@fixgreekcs {}{xi}
      \eu@fixgreekcs {}{omicron}
      \eu@fixgreekcs {}{pi}
      \eu@fixgreekcs {}{rho}
      \eu@fixgreekcs {}{varsigma}
      \eu@fixgreekcs {}{sigma}
      \eu@fixgreekcs {}{tau}
      \eu@fixgreekcs {}{upsilon}
      \eu@fixgreekcs {}{phi}
      \eu@fixgreekcs {}{chi}
      \eu@fixgreekcs {}{psi}
      \eu@fixgreekcs {}{omega}
      \eu@fixgreekcs {}{digamma}
      \eu@fixgreekcs {}{varbeta}
      \eu@fixgreekcs {}{varepsilon}
      \eu@fixgreekcs {}{vartheta}
      \eu@fixgreekcs {}{varkappa}
      \eu@fixgreekcs {}{varpi}
      \eu@fixgreekcs {}{varrho}
      \eu@fixgreekcs {}{varphi}}}
\newcommand\eu@fixgreekcs[2]{
  \ifcsequal{#2}{eu@cm@#2}
    {\relax}
    {\expandafter\def\csname #2\endcsname{{"#1}}}}

%% The FIFTH branch \setmathsfont(Symbols)
%% The symbols are not for now.
%% \newcommand\eu@ScopeSet@Symbols[2][]{
%%   \eu@fontspec{#1}{#2}
%%   \def\eu@Symbols@symfont{Symbols:m:n}
%%   \let\eu@Symbolsmathsfont\zf@family
%%   \eu@booltrue{Symbols}}
%%
%% The FIFTH branch (REDEFINED) \setmathsfont(Symbols)
\newcommand\eu@ScopeSet@Symbols[2][]{\eu@boolfalse{Symbols}}

%% Just so we know, by "all variant forms" I mean:
%%   varbeta     varpi       cursive beta    "omega" style pi
%%   varepsilon  varrho      lunate epsilon  rho with tail hooked under
%%   vartheta    varphi      cursive theta   cursive phi (or orthotic phi)
%%   varkappa    varTheta    cursive kappa   Theta with horizontal bar connected
```

```
%%
%% Note that varsigma is not included in this list (varsigma is not a variant
%% form, it is a final form).

%% Some authors might prefer the normal and variant forms to be exchanged. For
%% example, I prefer \phi to print the orthotic phi and \varphi the cursive phi,
%% contrary to many text fonts.
%%
%% \exchangeforms{<list>}
%%     <list> is a CSV list of any of:
%%         beta, epsilon, theta, kappa, pi, rho, phi, Theta
\newcommand\exchangeforms[1]{
  \AtBeginDocument{
    \@for\i@for:=#1\do{\eu@booltrue{exchange\i@for forms}}
    \eu@ifbooltrue{GreekLowercase}
      {\eu@ifbooltrue{exchangebetaforms}
         {\XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\beta]
          \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\varbeta]}
       \eu@ifbooltrue{exchangeepsilonforms}
         {\XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\epsilon]
          \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\varepsilon]}
       \eu@ifbooltrue{exchangethetaforms}
         {\XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\theta]
          \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\vartheta]}
       \eu@ifbooltrue{exchangekappaforms}
         {\XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\kappa]
          \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\varkappa]}
       \eu@ifbooltrue{exchangepiforms}
         {\XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\pi]
          \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\varpi]}
       \eu@ifbooltrue{exchangerhoforms}
         {\XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\rho]
          \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\varrho]}
       \eu@ifbooltrue{exchangephiforms}
         {\XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\phi]
          \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}[\varphi]}}
    \eu@ifbooltrue{GreekUppercase}
      {\eu@ifbooltrue{exhangeThetaforms}
        {\XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}[\Theta]
         \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}[\varTheta]}}}}
\@onlypreamble\exchangeforms

%% Some text fonts do not contain all variant forms. For those that don't,
%% after defining the Greek mathematics font, the ''absent variant forms will
%% still be typeset in Computer Modern (usually).
%%
%% \normalisevarforms[<list>]
%% \normalizevarforms[<list>]
%%     <list> is a CSV list of any of:
```

```
%%         beta, epsilon, theta, kappa, pi, rho, phi, Theta
%%
%% This command makes the listed variant forms of Greek symbols equivalent to
%% their normal forms, but only if they do not already exist in the font. For
%% example, if \varbeta and \eu@cm@varbeta are equivalent, then
%% \XeTeXDeclareMathSymbol failed to assign \varbeta to   because   does not
%% exist in the font.
\newcommand\normalisevarforms[1][beta,epsilon,theta,kappa,pi,rho,phi,Theta]{
  \AtBeginDocument{
    \@for\i@for:=#1\do{\eu@booltrue{normalisevar\i@for}}
    \eu@ifbooltrue{GreekLowercase}
      {\eu@ifbooltrue{normalisevarbeta}
        {\ifx\varbeta\eu@cm@varbeta
           \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}
           \let\varbeta\beta
         \fi}
       \eu@ifbooltrue{normalisevarepsilon}
        {\ifx\varepsilon\eu@cm@varepsilon
           \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}
           \let\varepsilon\epsilon
         \fi}
       \eu@ifbooltrue{normalisevartheta}
        {\ifx\vartheta\eu@cm@vartheta
           \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}
           \let\vartheta\theta
         \fi}
       \eu@ifbooltrue{normalisevarkappa}
        {\ifx\varkappa\eu@cm@varkappa
           \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}
           \let\varkappa\kappa
         \fi}
       \eu@ifbooltrue{normalisevarpi}
        {\ifx\varpi\eu@cm@varpi
           \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}
           \let\varpi\pi
         \fi}
       \eu@ifbooltrue{normalisevarrho}
        {\ifx\varrho\eu@cm@varrho
           \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}
           \let\varrho\rho
         \fi}
       \eu@ifbooltrue{normalisevarphi}
        {\ifx\varphi\eu@cm@varphi
           \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekLowercase@symfont }{`}
           \let\varphi\phi
         \fi}}
     \eu@ifbooltrue{GreekUppercase}{
       \eu@ifbooltrue{normalisevarTheta}
        {\ifx\varTheta\eu@cm@varTheta
```

```
            \XeTeXDeclareMathSymbol {}{\mathalpha}{\eu@GreekUppercase@symfont }{`}
            \let\varTheta\Theta
          \fi}}}}
\@onlypreamble\normalisevarforms
\let\normalizevarforms\normalisevarforms
\@onlypreamble\normalizevarforms

%% xkeyval things
\define@choicekey[eu]
  {Digits}{Arabic}[\eu@DigitsArabic@value\eu@DigitsArabic@@value]
  {Regular,Italic,Plain}[Regular]{\relax}
\define@choicekey[eu]
  {Latin}{Uppercase}[\eu@LatinUppercase@value\eu@LatinUppercase@@value]
  {Regular,Italic,Plain}[Italic]{\relax}
\define@choicekey[eu]
  {Latin}{Lowercase}[\eu@LatinLowercase@value\eu@LatinLowercase@@value]
  {Regular,Italic,Plain}[Italic]{\relax}
\define@choicekey[eu]
  {Greek}{Uppercase}[\eu@GreekUppercase@value\eu@GreekUppercase@@value]
  {Regular,Italic,Plain}[Regular]{\relax}
\define@choicekey[eu]
  {Greek}{Lowercase}[\eu@GreekLowercase@value\eu@GreekLowercase@@value]
  {Regular,Italic,Plain}[Italic]{\relax}

%% Also say Digits has Uppercase and Lowercase keys, and Latin and Greek both
%% have Arabic key. So that if a superfluous key is accidently sent through the
%% wrong command (possible with \setallmainfonts), then it doesn't blow up.
\define@key[eu]{Digits}{Uppercase}{\relax}
\define@key[eu]{Digits}{Lowercase}{\relax}
\define@key[eu]{Latin}{Arabic}{\relax}
\define@key[eu]{Greek}{Arabic}{\relax}

%% I hope Will doesn't mind if I patch fontspec. Why? Because these keys are
%% accidently sent through fontspec (e.g. \setallmainfonts does \setmainfont,
%% which is a fontspec command). So I allow fontspec to understand them, but do
%% nothing.
\define@key[zf]{options}{Arabic}{\relax}
\define@key[zf]{options}{Uppercase}{\relax}
\define@key[zf]{options}{Lowercase}{\relax}

%% Fonts have metrics suitable for text, not for mathematics. Need to kern each
%% letter individually. Previous version took a trial and error approach with
%% \+[<size>] where <size> was determined by trial and error. Now, there is a
%% XeTeX primitive \XeTeXglyphbounds which is used to automatically calculate
%% the required mkern.
%%
%% Some register definitions. \three@digits takes a number and, if it is less
%% than 100, inserts leading zeroes so that it has three digits.
\muskipdef\muskip@\z@
```

```
\muskipdef\muskip@i\@ne
\dimendef\dimen@iv 4\relax
\newcounter{eu@}
\newcounter{eu@i}
\providecommand\three@digits[1]{\ifnum#1<100 0\ifnum#1<10 0\fi\fi\number#1}

%% \setminwhitespace{<number>}
%%     Sets the minimum gap between adjacent characters in mathematics.
%%     <number> is in units of mmu (1/1000 mu). So 1 em = 18000 mmu.
%% \eu@minwhitespace is the minimum white space. It's default value is 500
%% which corresponds to 0.5 mu or 1/36 em.
\newcommand\setminwhitespace[1][500]{\def\eu@minwhitespace{#1}}
\setminwhitespace

%% \eu@mkern operates over some adjacent character tokens
%% \@eu@mkern operates over individual character tokens
%%
%% Unless I've missed something, it is really, really hard to get information
%% about the font that a mathematics symbol is typeset in, so I devised a series
%% of tests in text mode (\text!) (otherwise the received information will be
%% rubbish, related to the text font outside the mathematics). It would be
%% neater if I do it inside a box that's not printed, instead of an empty
%% \text. The \@eu@mkern algorithm to calculate the mkern might be neater if I
%% use e-TeX's \numexpr and \glueexpr. Maybe \gluetomu might be useful? What I
%% have here already ought to be sufficient but perhaps inefficient. I think
%% optical sizes follow through automatically because \text inherits the
%% surrounding size, is that right?
\newcommand\eu@mkern[1]{%
  \@tfor\i@tfor:=#1\do{\expandafter\@eu@mkern\i@tfor}}
\def\eu@get@familyseriesshape#1:#2:#3\@nil{%
    \expandafter\fontfamily\csname eu@#1mathsfont\endcsname\selectfont
    \ifstrequal{#2}{m}{\mdseries}{\relax}%
    \ifstrequal{#2}{bx}{\bfseries}{\relax}%
    \ifstrequal{#3}{n}{\upshape}{\relax}%
    \ifstrequal{#3}{it}{\itshape}{\relax}}
\def\eu@get@familyseriesshape#1:#2:#3\@nil{%
  \def\eu@family{#1}%
  \def\eu@series{#2}%
  \def\eu@shape{#3}}
\newcounter{mkern}
\newcommand\@eu@mkern[1]{%
    \setcounter{mkern}{-1}%
    \eu@boolfalse{domkern}%
    \eu@ifnumin{`#1}{`0,`9}{\setcounter{mkern}{0}}{\relax}%
    \eu@ifnumin{`#1}{`A,`Z}{\setcounter{mkern}{1}}{\relax}%
    \eu@ifnumin{`#1}{`a,`z}{\setcounter{mkern}{2}}{\relax}%
    \eu@ifnumin  {`#1}{`,`}{\setcounter{mkern}{3}}{\relax}%
    \eu@ifnumin  {`#1}{`,`}{\setcounter{mkern}{4}}{\relax}%
    \eu@ifnumis  {`#1}{`,`}{\setcounter{mkern}{3}}{\relax}%
```

```
\eu@ifnumis        {`#1}{`,`,`,`,`,`,`,`}{\setcounter{mkern}{4}}{\relax}%
\text{%
  \ifnum\the\c@mkern>\m@ne
    \eu@gbooltrue{domkern}%
  \fi
  \ifcase\the\c@mkern\relax % If Digits (0)
    \ifdef{\eu@DigitsArabic@symfont}%
      {\expandafter\eu@get@familyseriesshape\eu@DigitsArabic@symfont\@nil}%
      {\global\boolfalse{domkern}}%
  \or                       % If Latin Uppercase (1)
    \ifdef{\eu@LatinUppercase@symfont}%
      {\expandafter\eu@get@familyseriesshape\eu@LatinUppercase@symfont\@nil}%
      {\global\boolfalse{domkern}}%
  \or                       % If Latin Lowercase (2)
    \ifdef{\eu@LatinLowercase@symfont}%
      {\expandafter\eu@get@familyseriesshape\eu@LatinLowercase@symfont\@nil}%
      {\global\boolfalse{domkern}}%
  \or                       % If Greek Uppercase (3)
    \ifdef{\eu@GreekUppercase@symfont}%
      {\expandafter\eu@get@familyseriesshape\eu@GreekUppercase@symfont\@nil}%
      {\global\boolfalse{domkern}}%
  \or                       % If Greek Lowercase (4)
    \ifdef{\eu@GreekLowercase@symfont}%
      {\expandafter\eu@get@familyseriesshape\eu@GreekLowercase@symfont\@nil}%
      {\global\boolfalse{domkern}}%
  \fi
  \ifdef{\eu@family}%
    {\expandafter\fontfamily\csname eu@\eu@family mathsfont\endcsname\selectfont}%
    {\relax}%
  \ifdef{\eu@series}%
    {\expandafter\ifstrequal\expandafter{\eu@series}{m}%
       {\mdseries}%
       {\relax}%
     \expandafter\ifstrequal\expandafter{\eu@series}{bx}%
       {\bfseries}%
       {\relax}}%
    {\relax}%
  \ifdef{\eu@shape}%
    {\expandafter\ifstrequal\expandafter{\eu@shape}{n}%
       {\upshape}%
       {\relax}%
     \expandafter\ifstrequal\expandafter{\eu@shape}{it}%
       {\itshape}%
       {\relax}}%
    {\relax}%
  \global\dimen@\XeTeXglyphbounds\@ne\the\XeTeXcharglyph`#1\relax
  \global\dimen@iv\XeTeXglyphbounds\thr@@\the\XeTeXcharglyph`#1}
\dimen@ii 1em
\c@eu@\dimen@
```

```
    \c@eu@i\dimen@ii
    \multiply\c@eu@ 3000\relax
    \divide\c@eu@i 6\relax
    \divide\c@eu@\c@eu@i
    \c@eu@i\c@eu@
    \ifnum\number\c@eu@ <\eu@minwhitespace
      \c@eu@\eu@minwhitespace
    \fi
    \advance\c@eu@ -\c@eu@i
    \c@eu@i\c@eu@
    \divide\c@eu@i\@m
    \edef\eu@mkern@left{\number\c@eu@i}
    \multiply\c@eu@i\@m
    \advance\c@eu@ -\c@eu@i
    \edef\eu@mkern@left{\eu@mkern@left.\three@digits{\number\c@eu@}mu}
    \dimen@\dimen@iv
    \dimen@ii 1em
    \c@eu@\dimen@
    \c@eu@i\dimen@ii
    \multiply\c@eu@ 3000\relax
    \divide\c@eu@i 6\relax
    \divide\c@eu@\c@eu@i
    \c@eu@i\c@eu@
    \ifnum\number\c@eu@ <\eu@minwhitespace
      \c@eu@\eu@minwhitespace
    \fi
    \advance\c@eu@ -\c@eu@i
    \c@eu@i\c@eu@
    \divide\c@eu@i\@m
    \edef\eu@mkern@right{\number\c@eu@i}
    \multiply\c@eu@i\@m
    \advance\c@eu@ -\c@eu@i
    \edef\eu@mkern@right{\eu@mkern@right.\three@digits{\number\c@eu@}mu}
    \eu@ifbool{domkern}
      {\mkern\eu@mkern@left#1\mkern\eu@mkern@right}
      {#1}}

%% Redefine \" and " in maths mode only. Umlaut and quote definitions remain
%% in effect in text mode.
%%
%%   "       put the " before a character, e.g $"f$, and the character is
%%           printed with kerns on either side.
%%   \"..."  Surround a series of adjacent characters, e.g. $\"abcde"$ is
%%           equivalent to $"a"b"c"d"e$.
%% Note that the tokens that " and \" operate on MUST be characters because
%% they are sent through the XeTeX primitive \XeTeXcharglyph as in:
%%
%%   \XeTeXglyphbounds n \the\XeTeXcharglyph`#1
%%
```

```
%% where n = 1,2,3,4 and #1 is the character (If #1 is not a character, then
%% \XeTeXcharglyph`#1 doesn't make sense). Higher level tests are needed to
%% avoid this problem.
\let\eu@original@quote="
\let\eu@original@csquote=\"
\mathcode`\"="8000
\newcommand\eu@active@quote{%
  \ifmmode
    \expandafter\eu@new@quote
  \else
    \expandafter\eu@original@quote
  \fi}
\newcommand\eu@active@csquote{%
  \ifmmode
    \expandafter\eu@new@csquote
  \else
    \expandafter\eu@original@csquote
  \fi}
\begingroup
  \catcode`\"=\active
  \global\let"=\eu@active@quote
\endgroup
\let\"=\eu@active@csquote
\newcommand\eu@new@quote[1]{\eu@mkern{#1}}
\def\eu@new@csquote#1"{\eu@mkern{#1}}

%% Redefine LaTeX 2e kernel macros to do Unicode characters too. Add optional
%% fifth argument. This is a list of control sequences which will be let equal
%% to the symbol if the symbol exists in the font. If the symbol does not exist
%% in the font, then the fifth argument is ignored.
%%
%% e.g. \XeTeXDeclareMathSymbol {}{\mathrel}{font}{"2260}[\neq\ne]
\def\XeTeXDeclareMathSymbol#1#2#3#4{%
  \expandafter\in@\csname sym#3\expandafter\endcsname
    \expandafter{\group@list}%
  \ifin@
    \begingroup
      \if\relax\noexpand#1% is command?
        \edef\reserved@a{\noexpand\in@{\string\mathchar}{\meaning#1}}%
        \reserved@a
        \ifin@
          \expandafter\XeTeXset@mathsymbol
            \csname sym#3\endcsname#1#2{#4}%
          \@font@info{Redeclaring math symbol \string#1}%
        \else
          \expandafter\ifx
          \csname\expandafter\@gobble\string#1\endcsname
          \relax
          \expandafter\XeTeXset@mathsymbol
```

```
            \csname sym#3\endcsname#1#2{#4}%
          \else
            \@latex@error{Command `\string#1' already defined}\@eha
          \fi
        \fi
      \else
        \expandafter\XeTeXset@mathchar
          \csname sym#3\endcsname#1#2{#4}%
      \fi
    \endgroup
    \def\XeTeXDeclareMathSymbol@symbol{#1}%
    \def\XeTeXDeclareMathSymbol@slot{#4}%
    \expandafter\XeTeXDeclareMathSymbol@option
  \else
    \@latex@error{Symbol font `#3' is not defined}\@eha
    \expandafter\@gobbleoarg
  \fi}
\@onlypreamble\XeTeXDeclareMathSymbol
\def\XeTeXset@mathchar#1#2#3#4{%
  \global\XeTeXmathcode`#2="\mathchar@type#3#1#4\relax}
\@onlypreamble\XeTeXset@mathchar
\def\XeTeXset@mathsymbol#1#2#3#4{%
  \global\XeTeXmathchardef#2"\mathchar@type#3#1#4\relax}
\@onlypreamble\XeTeXset@mathsymbol
\newcommand\@gobbleoarg[1][]{}
\newcommand\XeTeXDeclareMathSymbol@option[1][]{%
  \ifnum\the\XeTeXcharglyph\XeTeXDeclareMathSymbol@slot>\z@
    \@tfor\i@tfor #1:=\do%
      {\expandafter\edef\i@tfor{\expandonce\XeTeXDeclareMathSymbol@symbol}}%
  \fi}

%% Some shorthands, so the same information isn't typed out more than once
%%
%% \setallmainfonts(<sets>)[<shapes, font features>]{<font name>}
%% \setprimaryfont[<shapes, font features>]{<font name>}
%% \setallsansfonts[<shapes, font features>]{<font name>}
%% \setallmonofonts[<shapes, font features>]{<font name>}
\newcommand\setallmainfonts{
  \@ifnextchar(
    {\eu@setallmainfonts}
    {\eu@setallmainfonts(Digits,Latin,Greek)}}
\def\eu@setallmainfonts(#1){
  \edef\eu@setmathsfont@Set{#1}
  \@eu@setallmainfonts}
\newcommand\@eu@setallmainfonts[2][]{
  \setmainfont[#1]{#2}
  \setmathsfont(\eu@setmathsfont@Set)[#1]{#2}
  \setmathrm[#1]{#2}}
\newcommand\setprimaryfont{\setallmainfonts(Digits,Latin)}
```

```
\newcommand\setallsansfonts[2][]{
  \setsansfont[#1]{#2}
  \setmathsf[#1]{#2}}
\newcommand\setallmonofonts[2][]{
  \setmonofont[#1]{#2}
  \setmathtt[#1]{#2}}

%% Set the particular mathematics alphabets
\ernewcommand\setmathrm[2][]{
  \zf@fontspec{#1}{#2}
  \let\eu@mathrm\zf@family
  \DeclareMathAlphabet{\mathrm}{\eu@enc}{\eu@mathrm}{m}{n}
  \SetMathAlphabet{\mathrm}{bold}{\eu@enc}{\eu@mathrm}{bx}{n}
  \DeclareMathAlphabet{\mathit}{\eu@enc}{\eu@mathrm}{m}{it}
  \DeclareMathAlphabet{\mathbf}{\eu@enc}{\eu@mathrm}{bx}{n}
  \SetMathAlphabet{\mathit}{bold}{\eu@enc}{\eu@mathrm}{bx}{it}
  \DeclareSymbolFont{Operators:m:n}{\eu@enc}{\eu@mathrm}{m}{n}
  \def\operator@font{\expandafter\mathgroup\csname symOperators:m:n\endcsname}}
\ernewcommand\setmathcal[2][]{
  \zf@fontspec{#1}{#2}
  \let\eu@mathcal\zf@family
  \DeclareMathAlphabet{\mathcal}{\eu@enc}{\eu@mathcal}{m}{n}}
\ernewcommand\setmathsf[2][]{
  \zf@fontspec{#1}{#2}
  \let\eu@mathsf\zf@family
  \DeclareMathAlphabet{\mathsf}{\eu@enc}{\eu@mathsf}{m}{n}
  \SetMathAlphabet{\mathsf}{bold}{\eu@enc}{\eu@mathsf}{bx}{n}}
\ernewcommand\setmathtt[2][]{
  \zf@fontspec{#1}{#2}
  \let\eu@mathtt\zf@family
  \DeclareMathAlphabet{\mathtt}{\eu@enc}{\eu@mathtt}{m}{n}}
\ernewcommand\setmathfrak[2][]{
  \zf@fontspec{#1}{#2}
  \let\eu@mathfrak\zf@family
  \DeclareMathAlphabet{\mathfrak}{\eu@enc}{\eu@mathfrak}{m}{n}}
\ernewcommand\setmathbb[2][]{
  \zf@fontspec{#1}{#2}
  \let\eu@mathbb\zf@family
  \DeclareMathAlphabet{\mathbb}{\eu@enc}{\eu@mathbb}{m}{n}}

%% If amsmath is loaded, it must be loaded before mathspec. Checking for its
%% existence \AtBeginDocument is too late because the damage is already done
%% (It attempted to define \varTheta when mathspec already defined it). Or
%% should I delay the definition of \varTheta until \AtBeginDocument?
\let\original@RequirePackage\RequirePackage
\renewcommand\RequirePackage[2][]{
  \ifstrequal{#2}{amsmath}
    {\PackageError{mathspec}
       {`amsmath' must be loaded earlier than `mathspec'}
```

```
        {Edit the document so that `amsmath' is required earlier than `mathspec'.}}
    {\relax}
  \original@RequirePackage[#1]{#2}}
\@onlypreamble\RequirePackage
\let\usepackage\RequirePackage
\@onlypreamble\usepackage

%% Any font changes that mathspec has done are reset by LaTeX  2 at
%% \begin{document} using \process@table.

\endinput

%% © Andrew Gilbert Moschou 2009
%%
%% This work may be distributed and/or modified under the
%% conditions of the LaTeX Project Public License, either version 1.3c
%% of this license or (at your option) any later version.
%% The latest version of this license is in:
%%   http://www.latex-project.org/lppl.txt
%% and version 1.3c or later is part of all recent distributions of LaTeX.
%%
%% This work has the LPPL maintenance status ``maintained.
%% The Current Maintainer of this work is Andrew Gilbert Moschou.
%% This work consists of the files mathspec.sty and mathspec.tex.
```

Exactly 1000 lines! How about that?

9 LICENSE