

ucharclasses

Mike "Pomax" Kamermans

October 19, 2009

Contents

1	introduction	2
2	use	2
2.1	overriding ucharclass transitions	3
3	commands	3
3.1	\setTransitionTo[2]	3
3.2	\setTransitionFrom[2]	3
3.3	\setTransitions[3]	4
3.4	\setTransitionsForXXXX[2]	4
3.5	\setDefaultTransitions[2]	4
4	Code	4
5	Unicode blocks	6


```

\setTransitionsForLatin{\fontspec{Palatino Linotype}}{}
\setTransitionsForArabic{\fontspec{Tahoma}\setRTL}{\setLTR}
\setTransitionsForCJK{\fontspec{SimSun}}{}
\setTransitionsForJapanese{\fontspec{Ume Mincho}}{}

\setTransitionTo{CJKUnifiedIdeographsExtensionB}{\fontspec{SimSun-ExtB}}
\setTransitionTo{Hebrew}{\fontspec{Times New Roman}}
\setTransitionTo{Thai}{\fontspec{IrisUPC}}
\setTransitionTo{Sinhala}{\fontspec{Iskoola Pota}}
\setTransitionTo{Malayalam}{\fontspec{Arial Unicode MS}}
\setTransitionTo{DominoTiles}{\fontspec{FreeSerif}}
\setTransitionTo{MahjongTiles}{\fontspec{FreeSerif}}

```

By default, `ucharclasses` is agnostic with regards to what you want to at the start or end of unicode blocks, so while using it for fontswitching is the most obvious application, you could also use it for far more creative purposes. An example of this is the Arabic block, which doesn't just need a different font, but also needs the text direction reversed, as it is read right-to-left, rather than left-to-right.

2.1 overriding ucharclass transitions

If you need to "override" `ucharclass` transition rules (for instance, you want a custom font for a bit of cross-unicode-block text), you will want to temporarily disable and reenabled XeTeX's `interchartoks` state. You can do this in three ways:

1. call `\XeTeXinterchartokstate=0` before, and `\XeTeXinterchartokstate=1` after you're done,
2. call the macros `\disableTransitionRules` before, and `\enableTransitionRules` after you're done, or
3. call `\uccoff` before, and `\uccon` after you're done.

This last option is mainly there because it's nice and short, and is more convenient in a scoped environment `{\uccoff such as this\uccon}` where you only want to override the transition behaviour within a paragraph. If you need it disabled for a few blocks of text instead, the full name commands are probably a better choice, because it makes your `.tex` more readable. As the base XeTeX command uses the `unLATEXy "... = ..."` construction, it's best to avoid it outside of the preamble (and when using `ucharclasses`, should not be in the preamble at all).

3 commands

3.1 `\setTransitionTo[2]`

This command has two arguments:

1. The name of the unicode class to which the transition should apply (see 'Unicode blocks' list)
2. The code you want used when entering this unicode block

3.2 `\setTransitionFrom[2]`

This command has two arguments:

1. The name of the unicode class to which the transition should apply (see 'Unicode blocks' list)
2. The code you want used when exiting this unicode block

3.3 `\setTransitions[3]`

This command has three arguments:

1. The name of the unicode class to which the transition should apply (see 'Unicode blocks' list)
2. The code you want used when entering this unicode block
3. The code you want used when exiting this unicode block

3.4 `\setTransitionsForXXXX[2]`

There are a number of these commands, pertaining to particular "informal groups": collections of unicode blocks which can be considered part of a single meta-block. Available informal groups (the names of which replace the XXXX in the section-stated command) are:

- Arabic
- Chinese
- CJK
- Cyrillic
- Diacritics
- Greek
- Korean
- Japanese
- Latin
- Mathematics
- Phonetics
- Punctuation
- Symbols
- Yi
- Other

Furthermore, these commands have two arguments:

1. The code you want used when entering blocks from the command's informal group
2. The code you want used when exiting blocks from the command's informal group

3.5 `\setDefaultTransitions[2]`

This is a blanket command that lets you set up the same to and from transition rules for all blocks in one go. It has (fairly obviously) two arguments:

1. The code you want used when entering any unicode block
2. The code you want used when exiting any unicode block

4 Code

The code relies on individual tex files that define the unicode blocks, each starting with:

```
\newcommand{\UnicodeBlockNameClass}{somenumber}
```

The classes are automatically numbered, per run, by using the `\newXeTeXintercharclass` command. The code then simply iterates over the class numbers from lower bound to upper bound; lower bound is recorded by calling `\newXeTeXintercharclass` once, before loading all the unicode blocks. Every block that is loaded then increments the class counter by calling `\newXeTeXintercharclass` for its class, and after the loading is done `\newXeTeXintercharclass`

is called once more to get the iteration upper bound. This technically wastes two class numbers, but this is fairly irrelevant given the number of classes we use for all the different unicode blocks.

The block loading code is defined as follows:

```
\newcounter{glyphcounter}
\newcommand{\@defineUnicodeClass}[3]{
  \newXeTeXintercharclass#1
  \forloop{glyphcounter}{#2}{\value{glyphcounter}<#3}{
    \XeTeXcharclass\value{glyphcounter}=#1}
  \XeTeXcharclass#3=#1}

\newXeTeXintercharclass\@classstart
\@defineUnicodeClass{\AegeanNumbersClass}{65792}{65855}
...
\@defineUnicodeClass{\YijingHexagramSymbolsClass}{19904}{19967}
\newXeTeXintercharclass\@classend
```

And the transition commands are defined as follows:

```
\newcommand{\setTransitionsFor}[3]{
  \forloop{iclass}{he\@classstart}{\value{iclass} < \@nameuse{#1Class}}{
    \@transition{he\value{iclass}}{\@nameuse{#1Class}}{#2}}
    \@transition{\@nameuse{#1Class}}{he\value{iclass}}{#3}}
  \addtocounter{iclass}{2}
  \forloop{iclass}{\value{iclass}}{\value{iclass} < he\@classend}{
    \@transition{he\value{iclass}}{\@nameuse{#1Class}}{#2}}
    \@transition{\@nameuse{#1Class}}{he\value{iclass}}{#3}}
  % and a binding for the transitions to and from boundary characters
  \@transition{255}{\@nameuse{#1Class}}{#2}}
  \@transition{\@nameuse{#1Class}}{255}{#3}}

\newcommand{\setTransitionTo}[2]{
  \forloop{iclass}{\the\@classstart}{\value{iclass} < \@nameuse{#1Class}}{
    \@transition{\the\value{iclass}}{\@nameuse{#1Class}}{#2}}
  \addtocounter{iclass}{2}
  \forloop{iclass}{\value{iclass}}{\value{iclass} < \the\@classend}{
    \@transition{\the\value{iclass}}{\@nameuse{#1Class}}{#2}}
  % and a binding for the transition from boundary characters
  \@transition{255}{\@nameuse{#1Class}}{#2}}

\newcommand{\setTransitionFrom}[2]{
  \forloop{iclass}{\the\@classstart}{\value{iclass} < \@nameuse{#1Class}}{
    \@transition{\@nameuse{#1Class}}{\the\value{iclass}}{#2}}
  \addtocounter{iclass}{2}
  \forloop{iclass}{\value{iclass}}{\value{iclass} < \the\@classend}{
    \@transition{\@nameuse{#1Class}}{\the\value{iclass}}{#2}}
  % and a binding for the transition to boundary characters
  \@transition{\@nameuse{#1Class}}{255}{#2}}
```

The broad level `\setTransitionsFor(InformalGroupName)[2]` commands are essentially wrapper commands, calling `\setTransitionsFor` for each blocks that is in the informal group. For Arabic, for instance, the code is:

```
\newcommand{\setTransitionsForArabic}[2]{
```

```

\setTransitionsFor{Arabic}{#1}{#2}
\setTransitionsFor{ArabicPresentationFormsA}{#1}{#2}
\setTransitionsFor{ArabicPresentationFormsB}{#1}{#2}
\setTransitionsFor{ArabicSupplement}{#1}{#2}
}

```

5 Unicode blocks

The following unicode blocks are available:

- AegeanNumbers
- AlphabeticPresentationForms
- AncientGreekMusicalNotation
- AncientGreekNumbers
- AncientSymbols
- Arabic
- ArabicPresentationFormsA
- ArabicPresentationFormsB
- ArabicSupplement
- Armenian
- Arrows
- Balinese
- BasicLatin
- Bengali
- BlockElements
- Bopomofo
- BopomofoExtended
- BoxDrawing
- BraillePatterns
- Buginese
- Buhid
- ByzantineMusicalSymbols
- Carian
- Cham
- Cherokee
- CJKCompatibility
- CJKCompatibilityForms
- CJKCompatibilityIdeographs
- CJKCompatibilityIdeographsSupplement
- CJKRadicalsSupplement
- CJKStrokes
- CJKSymbolsandPunctuation
- CJKUnifiedIdeographs
- CJKUnifiedIdeographsExtensionA
- CJKUnifiedIdeographsExtensionB
- CombiningDiacriticalMarks
- CombiningDiacriticalMarksforSymbols
- CombiningDiacriticalMarksSupplement
- CombiningHalfMarks
- ControlPictures
- Coptic
- CountingRodNumerals
- Cuneiform
- CuneiformNumbersandPunctuation
- CurrencySymbols
- CypriotSyllabary
- Cyrillic
- CyrillicExtendedA
- CyrillicExtendedB
- CyrillicSupplement
- Deseret
- Devanagari
- Dingbats
- DominoTiles
- EnclosedAlphanumerics
- EnclosedCJKLettersandMonths
- Ethiopic
- EthiopicExtended
- EthiopicSupplement
- GeneralPunctuation
- GeometricShapes
- Georgian
- GeorgianSupplement
- Glagolitic
- Gothic
- GreekandCoptic
- GreekExtended
- Gujarati
- Gurmukhi
- HalfwidthandFullwidthForms
- HangulCompatibilityJamo
- HangulJamo
- HangulSyllables
- Hanunoo
- Hebrew
- Hiragana
- IdeographicDescriptionCharacters
- IPAExtensions
- Kanbun
- KangxiRadicals
- Kannada
- Katakana
- KatakanaPhoneticExtensions
- KayahLi
- Kharoshthi
- Khmer

- KhmerSymbols
- Lao
- LatinExtendedA
- LatinExtendedAdditional
- LatinExtendedB
- LatinExtendedC
- LatinExtendedD
- LatinSupplement
- Lepcha
- LetterlikeSymbols
- Limbu
- LinearBIdeograms
- LinearBSyllabary
- loadblocks.tex
- Lycian
- Lydian
- MahjongTiles
- Malayalam
- MathematicalAlphanumericSymbols
- MathematicalOperators
- MiscellaneousMathematicalSymbolsA
- MiscellaneousMathematicalSymbolsB
- MiscellaneousSymbolsandArrows
- MiscellaneousSymbols
- MiscellaneousTechnical
- ModifierToneLetters
- Mongolian
- moo.txt
- MusicalSymbols
- Myanmar
- NewTaiLue
- NKo
- NumberForms
- Ogham
- OldChiki
- OldItalic
- OldPersian
- OpticalCharacterRecognition
- Oriya
- Osmanya
- PhagsPa
- PhaistosDisc
- Phoenician
- PhoneticExtensions
- PhoneticExtensionsSupplement
- PrivateUseArea
- Rejang
- Runic
- Saurashtra
- Shavian
- Sinhala
- SmallFormVariants
- SpacingModifierLetters
- Specials
- SuperscriptsandSubscripts
- SupplementalArrowsA
- SupplementalArrowsB
- SupplementalMathematicalOperators
- SupplementalPunctuation
- SupplementaryPrivateUseAreaA
- SupplementaryPrivateUseAreaB
- SylotiNagri
- Syriac
- Tagalog
- Tagbanwa
- Tags
- TaiLe
- TaiXuanJingSymbols
- Tamil
- Telugu
- Thaana
- Thai
- Tibetan
- Tifinagh
- Ugaritic
- UnifiedCanadianAboriginalSyllabics
- Vai
- VariationSelectors
- VariationSelectorsSupplement
- VerticalForms
- YijingHexagramSymbols
- YiRadicals
- YiSyllables