`smultiling.sty`: Multilingiality Support for sTeX

Michael Kohlhase
Jacobs University, Bremen
`http://kwarc.info/kohlhase`

April 20, 2014

Abstract

The `smultiling` package is part of the sTeX collection, a version of TeX/LaTeX that allows to markup TeX/LaTeX documents semantically without leaving the document format, essentially turning TeX/LaTeX into a document format for mathematical knowledge management (MKM).

The `smultiling` package adds multilinguality support for sTeX.

Contents

1 Introduction

The smultiling package adds multilinguiality support for sTeX, it is essentially a wrapper around the babel package but allows specification of languages by their ISO 639 language codes.

2 The User Interface

The smultiling package accepts all options of the babel.sty and just passes them on to it. The options specify which languages can be used in the sTeX language bindings.

3 Implementation

3.1 Class Options

To initialize the `smultiling` class, we pass on all options to `babel.cls` and record which languages are loaded by defining `\smul@`⟨*language*⟩`@loaded` macros.[1] EdN:1

```
⟨*sty⟩
\DeclareOption*{\PassOptionsToPackage{\CurrentOption}{babel}
\@namedef{smul@\CurrentOption @loaded}{yes}}
\ProcessOptions
⟨/sty⟩
⟨*ltxml⟩
# -*- CPERL -*-
package LaTeXML::Package::Pool;
use strict;
use LaTeXML::Package;
DeclareOption(undef,sub {PassOptions('babel','sty',ToString(Digest(T_CS('\CurrentOption')))); }
ProcessOptions();
⟨/ltxml⟩
```

We load `babel.sty`

```
⟨*sty⟩
\RequirePackage{etoolbox}
\RequirePackage{babel}
⟨/sty⟩
⟨*ltxml⟩
RequirePackage('babel');
⟨/ltxml⟩
```

3.2 Handling Languages

`\smg@select@language` This macro selects one of the registered languages by its langauage code by setting the internal `\smg@lang` macro to the argument and then runs the actual selection code in `\smg@select@lang`. This internal code register is only initialized there, the code is generated by the `\smg@register@language` macro below.

```
⟨ltxml⟩RawTeX('
⟨*sty | ltxml⟩
\newcommand\smg@select@lang{}
\newcommand\smg@select@language[1]{\def\smg@lang{#1}\smg@select@lang}
```

`\smg@register@language` `\smg@register@language{`⟨*lang*⟩`}{`⟨*babel*⟩`}` registers the `babel` language name ⟨*babel*⟩ with its ISO 639 languge code ⟨*lang*⟩ by extending the `\smg@select@language` macro.

```
\newcommand\smg@register@language[2]%
{\@ifundefined{smul@#1@loaded}{}{\appto\smg@select@lang%
{\expandafter\ifstrequal\expandafter\smg@lang{#1}{\selectlanguage{#2}}{}}}}
```

[1] EdNote: @DG: We also want to do that in LaTeXML

Now we register a couple of languages for which we have `babel` support. Maybe we have to extend this list with others. But then we have to extend the mechanisms.

```
\smg@register@language{af}{afrikaans}
\smg@register@language{de}{ngerman}
\smg@register@language{fr}{french}%
\smg@register@language{he}{hebrew}
\smg@register@language{hu}{hungarian}
\smg@register@language{id}{indonesian}
\smg@register@language{ms}{malay}
\smg@register@language{nn}{nynorsk}
\smg@register@language{pt}{portuguese}
\smg@register@language{ru}{russian}
\smg@register@language{uk}{ukrainian}
\smg@register@language{en}{english}
\smg@register@language{es}{spanish}
\smg@register@language{sq}{albanian}
\smg@register@language{bg}{bulgarian}
\smg@register@language{ca}{catalan}
\smg@register@language{hr}{croatian}
\smg@register@language{cs}{czech}
\smg@register@language{da}{danish}
\smg@register@language{nl}{dutch}
\smg@register@language{eo}{esperanto}
\smg@register@language{et}{estonian}
\smg@register@language{fi}{finnish}
\smg@register@language{ka}{georgian}
\smg@register@language{el}{greek}
\smg@register@language{is}{icelandic}
\smg@register@language{it}{italian}
\smg@register@language{la}{latin}
\smg@register@language{no}{norsk}
\smg@register@language{pl}{polish}
\smg@register@language{sr}{serbian}
\smg@register@language{sk}{slovak}
\smg@register@language{sl}{slovenian}
\smg@register@language{sv}{swedish}
\smg@register@language{th}{thai}
\smg@register@language{tr}{turkish}
\smg@register@language{vi}{vietnamese}
\smg@register@language{cy}{welsh}
\smg@register@language{hi}{hindi}
```

3.3 Language Bindings

`modsig:*`

```
\addmetakey*{modsig}{title}
\addmetakey*{modsig}{creators}
\addmetakey*{modsig}{contributors}
```

modsig The modsig environment is just a layer over the module environment.

```
\newenvironment{modsig}[2][]{\metasetkeys{modsig}{#1}% to check
\def\@test{#1}\ifx\@test\@empty\begin{module}[id=#2]\else\begin{module}[id=#2,#1]\fi}
{\end{module}}
```

modnl:*

```
\addmetakey{modnl}{load}
\addmetakey*{modnl}{title}
\addmetakey*{modnl}{creators}
\addmetakey*{modnl}{contributors}
```

modnl The modnl environment is just a layer over the module environment with the keys and language suitably adapted.

```
\newenvironment{modnl}[3][]{\def\@test{#1}%
\ifx\@test\@empty\begin{module}[id=#2.#3]\else\begin{module}[id=#2.#3,#1]\fi%
\ifx\modnl@load\@empty\importmodule{#2}\else\importmodule[load=\modnl@load]{#2}%
\smg@select@language{#3}}
{\end{module}}
```