The future of Language Converter

C. Scott Ananian <cscott@cscott.net>
Wikimedia Foundation

MediaWiki Developer Summit 2015
January 27, 2015

Why do we use Language Converter?

Languages are fun!

Some example pairs for English speakers:

- English language variants:
  - color/colour
  - ten million/one crore
- Brazillian Portuguese differs to about the same extent:
  - berinjela / beringela
- Pig latin
  - Igpay atinlay *(reversible)*
  - eightway *(is that “eight” or “weight”?)*

On wikipedia

See: https://meta.wikimedia.org/wiki/Wikipedias_in_multiple_writing_systems

Chinese Wiki: two major writing systems, various dialects.

- zh-cn (Mainland China), zh-tw (Taiwan), zh-hk (Hong Kong), zh-mo (Macao) and zh-sg (Singapore)

Serbian Wiki: two writing systems, and two dialects.

- Latin alphabet Ekavian, Cyrillic alphabet Ekavian, Latin alphabet Ijekavian, Cyrillic alphabet Ijekavian

Kazakh Wiki: three writing systems.

- Cyrillic, Latin, and Perso-Arabic (Central Asian branch) alphabets. (Conversion to Arabic read only.)

On Wikipedia, continued

Kurdish Wiki: three writing systems

- Latin (Turkey/Syria) <-> Arabic (Iraq/Iran)
- no support for Cyrillic (ex USSR)

Inuktitut Wiki: two writing systems.

- Inuktitut syllabics (Nuvavut) <-> Latin
  - □□□□□□ <=> Inuktitut
  - http://www.languagegeek.com/inu/inutext.html
- Syllabics lose case distinction from latin script

Anglo-Saxon Wiki: two writing systems

- Latin <-> Runic

On Wikipedia, even more

Tajik: (Cyrillic<->Latin, but not Arabic)

Uzbek: (Cyrillic<->Latin, but not Arabic)

Gan: (simplified<->traditional Gan Chinese, but not Romanized Gan)

Kyrgyz: (Cyrillic/Latin/Arabic, not yet deployed)

Uyghur: (Arabic/Cyrillic, not implemented?)

Chechen: (Cyrillic/Latin, not yet deployed)

...and 29 more, see the full list

What does Language Converter do?

Some example markup

Marking up text which has variants:

- Bidirectional rules:

  -{zh-hans:computer; zh-hant:ELECTRONICBRAIN;}-
- Unidirectional rules:

  -{HUGEBLOCK=>zh-cn:macro;}-
- Disable conversion:

  -{R|SimpTrad}-

Also a bunch of stateful options for adding/removing rules (more on this later), and for working around title limitations.

See https://www.mediawiki.org/wiki/Writing_systems/Syntax

Some example code (transliteration)

```
class SrConverter extends LanguageConverter {
    public $mToLatin = array(
        'а' => 'a', 'б' => 'b', 'в' => 'v', 'г' => 'g', 'д' => 'd',
        'ђ' => 'đ', 'е' => 'e', 'ж' => 'ž', 'з' => 'z', 'и' => 'i',
        [...]
        'Х' => 'H', 'Ц' => 'C', 'Ч' => 'Č', 'Џ' => 'Dž', 'Ш' => 'Š',
    );
    public $mToCyrillics = array(
        'a' => 'а', 'b' => 'б', 'c' => 'ц', 'č' => 'ч', 'ć' => 'ћ',
        'd' => 'д', 'dž' => 'џ', 'đ' => 'ђ', 'e' => 'е', 'f' => 'ф',
        [...]
        'Nj' => 'Њ', 'n!j' => 'нј', 'N!j' => 'Нј', 'N!J' => 'НЈ'
    );
```

Some example code (hant/hans)

```
$zh2Hant = array(
'佩' => '佩',
'佾' => '佾',
[...]
';克制' => ';剋制',
'？克制' => '？剋制',
);
$zh2Hans = array(
'偵' => '侦',
'傯' => '偬',
[...]
```

Some example code (templates!)

The zhwiki as a gadget installed named 'NoteTA'. See zh:Module:NoteTA and zh:MediaWiki:Gadget-noteTA.js.

This is used to display the current set of word conversions for a page.

A template with NoteTA is used to define a set of conversions specific to the page.

Some example code (templates!)

From [[:zh:鋼鐵人3]] ([[:en:Iron Man 3]]):

```
{{noteTA
|G1=Movie
|G2=Show
|G3=美国漫画
|1=zh-hans:罗伯特; zh-tw:勞勃; zh-hk:羅拔;
|2=zh-hans:奥德利奇·齊連安; zh-tw:奧德奇·齊禮安; zh-hk:奧德奇·齊禮安;
|3=zh-hans:羅德斯; zh-tw:羅德; zh-hk:羅德;
}}
```

Pros and cons

- For variants with lossless conversions the process seems to work well, expanding the set of readers for our content
- In some wikis, a dominant variant is used for authoring content
- But… in some cases editors are not typically fluent in both variants. Content is written in both, interspersed.
- Some issues with word boundaries.
- Some issues with lossy conversions.

Technical issues

Language Converter is awkwardly placed in the parsing pipeline.

- Leads to strange parsing issues because -{}- comes *after* other parsing steps
- Leads to security bugs because -{}- parsing comes *after* the sanitizer!
- Should be better integrated
  - But we don’t know what variant to use until the page is viewed
  - Requires splitting the parser cache by variant.
  - Probably not a trivial task!

Parsoid (for completeness)

!! wikitext

Taiwan is not China.

But -{A|zh:China;zh-tw:Taiwan}- is China,

(This-{-|zh:China;zh-tw:Taiwan}- should be stripped!)

and -{China}- is China.

!! html/parsoid

Taiwan is not China.

But <span typeof="mw:LanguageVariant" data-mw='{"add":true,"show":true,"
bidir":{"zh":"China","zh-tw":"Taiwan"}}'> is China,

(This<meta typeof="mw:LanguageVariant" data-mw='{"remove":true,"bidir":
{"zh":"China","zh-tw":"Taiwan"}}'/> should be stripped!)

and <span typeof="mw:LanguageVariant" data-mw='{"disabled":true,"text":"
China"}'> is China.

Parsoid/Visual Editor challenges

- Convert to stateless representation
  - RFC
- Parsoid and templates are not good friends
- HTML page views / read only representation
  - Implement conversion as post-processing
- Editing
  - Can we round-trip the conversion process?
  - Requires (automatically) marking up content where conversion is lossy
  - Editor affordances for adding glossaries
  - Editor affordances for editing variant text
  - Narrower selser to restrict editing scope.

What are some alternatives?

Other ways to handle variants

- Null option.
  - Pick a single preferred variant.
- Read-only language converter.
  - Content authored in a single preferred variant.
- Bidirectional language converter.
- Content Translation Tools.
- Split the wikis.
  - With better tools to maintain forked wikis?
  - Content Translation Tools one option here?
  - One-click synchronization between wikis?

Are they other good ideas here?

Scaling translation

We have two scaling axes in localization:

- Message size
  - Wikidata tags
  - Interface strings
  - …
  - Full articles
- Language divergence
  - American/British English
  - Brazillian/Portuguese
  - Romance languages
  - “Eastern Punjabi” (ISO PAN, in India)/“Western Punjabi” (ISO PNB, in Pakistan)
  - English/Mandarin

Questions

Can we make tools which scale well along both axes?

Should we?

How many tools should we have?

What are the sweet spots?

Ok, let’s discuss!

(thanks)