Implementation tricks in the Hungarian babel module

Szabó Péter
<pts@inf.bme.hu>

September 2004

presented at TUG 2004
Xanthi, Greece

Recent LaTeX localization attempts

In year 2003 and 2004 there were many attempts to localize LaTeX to better obey the Hungarian traditions. A collection of some of these attempts is the MagyarLaTeX package.

- *magyar.ldf*. This is the Hungarian babel module. Version 1.4 in the official babel, is 20 kB. Version 1.5, developed recently, is 195 kB – longer than any other language module ever.
- *huplain.bst*. A BibTeX style file with Hungarian typography, input encoding and sorting.
- *husort.pl*. A makeindex replacement.
- *magyar*.xdy*. Various Xindy style files not stable enough yet.
- *huhyphf.tex* and *huhyphc.tex*. Hyphenation patterns with PatGen.
- *huhyphn.tex*. Hyphenation patterns generated by PatGen, based on a dictionary and automatic suffix removal.
- *lafmtgen.pl*. A Perl script that helps LaTeX format generation.
- *ccs_extract.pl*. Finds long double consonants for hyphenation.

Load options

It is possible to give load options to the new magyar.ldf, similarly to the `\hypersetup` command of hyperref.sty.

Load options are parsed when magyar.ldf is loaded (the parsing is similar to `\@for` and keyval.sty, but extra sentinel tokens are added to better detect common syntax errors). The specified load options make unneeded parts of the rest of the file to be skipped, and they also customize the behaviour of magyar.ldf.

The benefits of load options are:

- *compatibility mode* with old versions
- *flexible* for the typographer
- *less conflicts* with other LaTeX packages
- the *flexible active character* causes less conflicts
- easy narrowing of *bug locations*
- experimental *dual load* with somewhat different sets

Default option sets

Since magyar.ldf has more than 50 load options, default sets are provided so the user can select quickly.

- `defaults=over-1.4`. Full `.tex` source compatibility with version 1.4, with some minor typographical enhancements and the new commands enabled. This is the default.
- `defaults=compat-1.4`. Full feature compatibility with version 1.4, but avoiding package conflicts.
- `defaults=safest`. All features are turned off. Useful for finding the location of a bug or compatibility issue in magyar.ldf.
- `defaults=prettiest`. Includes all features, dangerous too.
- `defaults=hu-min`. A minimal set of features are turned on so the document obeys the Hungarian traditions. Avoids dangerous features. This is recommended: `\PassOptionsToPackage{defaults=hu-min}{magyar.ldf}`.

See the new magyar.dtx for a more detailed description of the sets and the individual options.

Skipping parts of the input file

Wrapping a long code into `\@gobble{...}` would fill the input stack, wrapping it to `\iffalse...\fi` needs balanced conditionals and it even consumes hash memory for control sequences never used. magyar.ldf uses the following construct:

```
\@gobble\iftrue
\def\skiplong#1{\fi
  \bgroup% so ^} would close it
  \catcode\string‘^13 \lccode\string‘~=\string‘^
  \lowercase{\let~\fi}%
  \catcode\string‘\\14 % comment, save hash memory
  \catcode\string‘$14
  \iffalse}
\@gobble\fi
```

After that, code can be skipped with the construct

```
\ifnum\MyFeature<1 \skiplong\fi
  ...
  \@gobble
{^}
```

Hyphenation

The hyphenation rules of simple Hungarian words can be directly translated to TEX patterns (PatGen is not needed). However, compound words need to be hyphenated at subword boundary. Finding that boundary correctly without a dictionary is very hard.

- *huhyph3.tex*. no PatGen, short exception list, 24 kB
- *huhyphf.tex*. with PatGen, based on a short dictionary without suffixes, hyphenates foreign workds phonetically: *szink-ron*, 38 kB
- *huhyphc.tex*. like huhyphf.tex, but *szin-kron*
- *huhyphn.tex*. Based on huge number of Hungarian words gathered from web pages, suffixes removed before semi-automatic subword detection, suffix placed back before pattern generation. 94 kB.

There is also a problem with long double consonants, e.g. *poty-tyan* melds to *pottyan* if not hyphenated. magyar.ldf has an active character, so `lepo‘ttyantam` translates to `lepot\nobreak\discretionary{y-}{}{}\nobreak\hskip\z@skip tyantam`. ccs_extract.pl helps inserting ‘. Special VF and TFM files would be need for an automatic solution.

Footnotes

magyar.ldf implements the `\footnotestyle` command that customizes the appearance of footnotes.

It is possible to number footnotes with stars (*, **, ***), restarting at each page. By the time `\footnote` is called the page layout of the footnotes is unknown. magyar.ldf uses places a `\label` to each footnote, and at the next run of LaTeX, it check whether the page number of the current and the previous footnotes are the same – if they differ, renumbering starts at '*'.

We need an expandable construct that emits n stars. David Kastrup has provided a brilliant solution to the problem: `\expandafter\mtostar\romannumeral\number`n`000A`, where `\mtostar` transforms `m`s to stars: `\def\mtostar#1{\if#1m*\expandafter\mtostar\fi}`. This solution is used in magyar.ldf.

The definite article

The numbers generated by `\ref` are often prefixed by the definite article ('the' in English). But the Hungarian definite article has two forms: *a/az*, depending on the pronounciation of the word following it. For nonnegative integers it is very easy to put the correct article:

```
char* aaz(unsigned i) { return i==1 || i==5 || i>49 && i<60
  || i>499 && i<600 || i>999 && aaz(i/1000) ? "az" : "a";}
```

The `\az` command in magyar.ldf does the following:

- works for numbers, words and single letters
- half-expands its argument, removes accents, braces and control sequences, finds the very first letter or digit
- half-expansion involves the redefinition of `\hbox`, `\j` etc.
- has special support for `\ref`, `\pageref` and `\cite`
- emits a `\hunnewlabel` with arabic numbers to the `.aux` file after each `\newlabel`, so the definite article can be based on the numeric value and not the roman letters

Suffix generation for numbers

The Hungarian language has suffixes instead of prepositions. The suffixes depend on the vowel harmony of the suffixed word, so when suffixing a generated number (possibly an equation number with `\ref`), the suffix has to be also generated. This is done by the `\told` command in magyar.ldf.

An example sentence: (1)-hez hozzáadva (2)-t és elosztva (3)-mal, és az eredményt kivonva (4)-ből kapunk egy (5)-nél nagyobb értéket, ami relatív prím (6)-hoz, ám ez (7)-ről nem mondható el.

`\told` divides integers into 23 paradigm classes based on the last nonzero digit and the number of zeroes following it. The proper suffix is then looked up in a table indexed by the suffix and the paradigm class. Half-expansion is used to find the last number in the argument. Classification is done by a finite automaton implemented in TEX macros.

Structural references

When using \ref the structural depth of the referred element must be typed directly, such as in 'from section~\ref{foo}'. \autoref in nameref.sty and hyperref.sty can generate section in the example above. \refstruc in magyar.ldf is similar, but it is possible to combine it \told and \az.

\autoref changes the syntax of the \newlabel command emitted to the .aux file. \refstruc uses this change to get the structure depth, but if the extra information is missing, \refstruc generates \thechapter, \thesection etc. with all numbers changed to one, and compares it with \@currentlabel. Disadvantages: is '5.6' in a book a section or a figure (\refstruc chooses section), extra workaround is needed for refs between the main text and the appendix.

Others (1)

- compatibility issues with the active character: `\shu`
- date generation and recognition
- accent lowering: `\umlautlow` vs virtual fonts; the ae fonts
- table and figure captions: '1. táblázat'; modify `\fnum@table`
- TOC flaw for language changes: forced `\select@language` before each `.toc` entry
- dot after section number: change `\numberline` instead of `\@sect` (etc.) to increase compatibility
- extra space before `!`, `?`, `:` and `;` – activate new characters
- nested in-paragraph quotations: `\textqq`
- formatting of `itemize` and `enumerate` environments

Others (2)

- inserting late begin-document hooks to `\@preamblecmds`
- theorem titles with/without theorem.sty, ntheorem.sty, amsthm.sty, `\theoremstyle{magyar-plain}`
- the decimal comma. One implementation: active math character, checks `\meaning` of the following token. Changes `\mathpunct` to `\mathord` if finds a digit.
- patching `\@sect` to remove full stop after section titles in AMS classes; pattern matching on a macro body, dirty
- repeat hyphenated operation or relation: substitute `=\nobreak \discretionary{}{\hbox{\(=\)}}{}` for = (etc.) in formulas
- spelled-out number generation: `\@huordinal` and `\@hunumeral`
- warning messages on load: missing tlenc.sty etc.

Conclusion and future work

- magyar.ldf is more compatible with other packages: graphic*.sty, hyperref.sty, ams*.cls, *theorem.sty and varioref.sty now work without problems.
- MagyarLaTeX is freely available from www.math.bme.hu/latex. The lincense is GNU GPL.
- magyar.dtx is waiting the approval of Johannes Braams
- Many of the features and ideas in magyar.ldf are useful in other language modules.
- MagyarLaTeX has been tested on a 100 page long thesis and a 750 pages long book.
- The design of the LaTeX core, the module system and babel is too simple. Compatibility checks and dirty fixups are needed in many cases. It is hoped that a new, more versatile localization model is invented soon, possibly in Ω.
- magyar.dtx contains several dozen improvement possibilities. Gyöngyi said, “There is a lot to do in the future.”