Annales Mathematicae et Informaticae
Accepted manuscript
DOI: 10.33039/ami.2026.05.001
URL: https://ami.uni-eszterhazy.hu

An experimental Hungarian language definition file for the Babel package in LaTeX

Tibor Tómács

Eszterházy Károly Catholic University,
Institute of Mathematics and Informatics, Eger, Hungary
tomacs.tibor@uni-eszterhazy.hu

Abstract. The paper reviews the historical background of Hungarian language support in the LaTeX Babel package and introduces the experimental language definition file `hungarian.ldf`. The new module aims to provide a smaller, more easily testable implementation built on modern LaTeX/Babel hooks, offering an explicit `hungarian` identity and more tightly controlled side effects while covering the most important requirements of Hungarian typography. We describe the option interface, the role of the active quotation mark and punctuation characters, date handling, and the integration of `huaz` for automatic definite article selection and numeric suffixation. Special attention is given to improving Hungarian localization for `hyperref` and `varioref`, as well as to the optional use of `rmathbr` for line breaking in inline mathematics. The `hungarian.ldf` and `magyar.ldf` modules are not compatible; the approach presented here is experimental and is expected to stabilize based on user feedback.

Keywords: LaTeX, Babel package, Hungarian typography

2020 *Mathematics Subject Classification:* 68U01, 68U15, 68U99

1. Historical introduction

The concept behind the Babel package for the LaTeX document preparation system (see [7]) is to allow switching – even within a single document – between multiple languages in such a way that language-dependent elements (captions and other fixed strings, date formats, hyphenation patterns, punctuation conventions, and

Submitted: February 26, 2026
Accepted: May 14, 2026
Published online: May 14, 2026

similar typographic norms) change in a controlled and coherent manner (see [1, 2]). In the classical Babel architecture, this functionality is typically provided by `.ldf` language modules (language definition files).

From this perspective, the Hungarian module is unusual: its filename is not the English `hungarian.ldf`, but rather derives from the Hungarian word *magyar*: `magyar.ldf`.

Hungarian is among the relatively few languages in which the name of a structural element and its number are conventionally written in the reverse order compared to English. For instance, instead of "*Chapter 1*", Hungarian requires "*1. fejezet*". Not only is the order inverted, but the ordinal number must also be followed by a full stop. Unfortunately, in most document classes (with a few exceptions) this behavior cannot be enabled via simple configuration; rather, it typically requires the redefinition of multiple nontrivial macros. The reversed order and the trailing full stop must be enforced consistently in titles and headings, running headers, the various lists (e.g., table of contents and lists of figures/tables), theorem-like environments, and also in the PDF bookmarks. Achieving this consistency is therefore a complex and error-prone undertaking.

The situation is further complicated by Hungarian morphology. In particular, suffixation depends on vowel harmony, which poses challenges for packages such as `varioref` and for `hyperref`'s **`\autoref`** and **`\autopageref`** commands. Another Hungarian-specific issue is the definite article (*a*/*az*), whose form depends on the initial sound of the following word. Automating these phenomena reliably constitutes a nontrivial programming task.

1.1. The early period: the roots of Hungarian support (1989–1996)

In the header of the current `magyar.ldf` file, the chain of authorship and copyright notices reaches back to the early days of Babel. The first versions are attributed to Johannes Braams (a key figure in the development of Babel); his name appears from 1989 onward, although the file is in fact the result of contributions by several authors. This period was characterized by the following:

- LaTeX language support was still comparatively limited in scope,
- Hungarian accented letters and the ecosystem of input/font encodings required substantially more manual care,
- only a restricted subset of Hungarian typographic conventions was automated by the system.

The header also records contributions by Árpád Bíró through 1996.

1.2. Stabilisation and the "legacy" magyar.ldf line: József Bérces and the v1.4c era (1996–2001)

The next name in this lineage is JÓZSEF BÉRCES. Version 1.4c (2001) includes substantial work by him. From a historical perspective, it is important to note that this period was characterised by the following:

- much of the behaviour still represented a compromise between Babel's general mechanisms and Hungarian-specific requirements;
- several "tricks" and ad hoc workarounds were necessary in order to interoperate with other packages;
- the prevailing mindset was not yet that "*the most faithful possible adherence to Hungarian typographic rules*" should be the primary goal; rather, the emphasis was on basic functionality and core localization.

1.3. The major turning point: Péter Szabó and the 2003 rewrite (v1.5)

The single most significant milestone in the history of `magyar.ldf` was its complete rewrite in the autumn of 2003 under the leadership of PÉTER SZABÓ, undertaken explicitly to achieve substantially better compliance with Hungarian typographic conventions [3, 11]. Why can this be regarded as a genuine paradigm shift?

Jump in size and functionality. The paper [11] provides an explicit comparison: the "old" `magyar.ldf` was on the order of ≈ 25 kB, whereas the rewritten version was larger by an order of magnitude (the article reports ~ 178 kB for the state at the time), indicating a substantial expansion of functionality.

A more rule-based encoding of Hungarian typography. The guiding philosophy of the rewrite was not merely to translate fixed strings, but to support recurrent Hungarian typographic and grammatical phenomena that arise throughout documents (e.g., quotation marks, spacing conventions, lists, heading formats, and the linguistic behavior of cross-references).

Documentation and the MagyarLaTeX ecosystem. As noted in [11], the version of `magyar.ldf` current at the time was distributed as part of the *MagyarLaTeX* bundle.

From the 1.5 branch onward, `magyar.ldf` has been explicitly designed around the idea that many of its features can be enabled or disabled via options, and that distinct "defaults" profiles are available. These are discussed in detail in the user documentation [12] and [17, Chapter 9], which also explain how the corresponding settings must be passed to Babel prior to loading it.

This period is historically significant because, from this point on, `magyar.ldf` is no longer "a single fixed behavior" but rather a configurable system that navigates the trade-offs between Hungarian typographic conventions and the practical compatibility constraints of the LaTeX ecosystem.

A key part of `magyar.ldf`'s reputation consists in features that automate phenomena specific to Hungarian. One prominent example is the automatic handling of the definite article (a/az) and the associated cross-referencing commands. Historically, this is noteworthy because it goes beyond "localization" in the narrow sense and instead integrates linguistic logic into the typesetting system: the appropriate Hungarian article is inserted before numbers and labels generated by LaTeX in contexts where Hungarian usage requires it.

1.4. Maintainer transition and modernization: 2025–2026

In 2025, the package underwent a change of maintainer, with Tibor Tómács becoming the primary maintainer. Version 1.6a of `magyar.ldf` (2026-02-03) introduces several modernization steps, including a revised option-handling mechanism and the gradual retirement of legacy/deprecated options. From a historical perspective, this marks a new chapter: the emphasis is no longer solely on typographic tricks, but also on improving the package's integration with the modern LaTeX/Babel infrastructure (see [1, 2, 13]).

2. A new avenue for Hungarian language support within the Babel system

The existing `magyar.ldf` is a mature, historically layered Babel module: it attempts to support a wide variety of document classes and packages through a large number of switches, compatibility branches, and "guard rails". However, this legacy also comes with a set of typical consequences:

Maintenance burden and growth in complexity. The `magyar.ldf` module is on the order of 224 kB / 5261 lines, and it handles 63 options via multiple key–value associations. This is a rational design choice if one aims to accommodate both the behavior of legacy documents and the requirements of modern typography, yet it also makes systematic, transparent testing of changes nearly impossible, and complicates the isolation of regressions and compatibility issues.

Global side effects and "defensive" interventions. From the very beginning, the code in `magyar.ldf` must account for the fact that other languages/states can "break" category codes; a clear symptom of this is the abundance of `\catcode`-level interventions throughout the file. This typically indicates that the module is not merely expected to enable Hungarian-specific behaviour, but must also protect/restore part of the global LaTeX/Babel state.

Identity and naming duality: magyar vs. hungarian. From a Babel user perspective, the language is typically called "*hungarian*", while historically the module file name is `magyar.ldf`, which itself relies on dynamic mechanisms such as '`\ProvidesLanguage{\CurrentOption}`'. This duality is not an error per se, but it can lead to subtle "name resolution" problems in a number of packages.

An overly large "surface API": many switches, many combinations. One of the key strengths of `magyar.ldf` is its configurability (default profiles, multiple compatibility paths), but this simultaneously implies an explosion of the testing space: many option combinations do not merely add features, but also introduce distinct side effects.

Among other considerations, these facts motivated the author of the present paper, Tibor Tómács, to develop – on an experimental basis – a completely new `.ldf` file under the name `hungarian.ldf`.

While `magyar.ldf` is valuable and stable, its historical layering has made it both overly large and, in many places, globally invasive. The goal of `hungarian.ldf` is to establish a new foundation that leverages the modern LaTeX/Babel infrastructure, provides an explicit "hungarian" identity, exposes a smaller surface, and yields more controlled side effects.

The purpose of the rewrite was not to reproduce all historical options of the legacy module immediately, but rather to provide a compact, clean, readily comprehensible, and extensible core on top of which functionality can later be layered and expanded with due caution.

The `magyar.ldf` module relies heavily on `\AtBeginDocument`, which often "works well enough" in practice, yet in certain package combinations it typically leads to load-order–dependent issues. By contrast, `hungarian.ldf` makes substantially greater use of LaTeX kernel hooks, Babel hooks, and package hooks. With a hook-based approach, patches can be applied precisely where and when they are most stable – namely, once the target package is guaranteed to be available – which in turn reduces the likelihood of regressions in the long run.

The `hungarian.ldf` declares captions and date strings using Babel's newer interfaces:

```
\StartBabelCommands*{hungarian}{captions} + \SetString{...}
\StartBabelCommands*{hungarian}{date} + \SetStringLoop{...}
```

From a technical standpoint, this approach scales substantially better than the "hand-assembled `\@namedef{captions...}{...}`" idiom, because the language strings can be treated as data. This makes it easier for other packages to build upon them, and it is more robust under language switching (rather than degenerating into a collection of ad hoc definitions). In this way, it provides a foundation for a consistent and extensible localization layer – particularly in the long term, as LaTeX/Babel internals evolve.

Accordingly, `hungarian.ldf` offers an alternative implementation of Hungarian language support in Babel alongside `magyar.ldf`. It is currently in an experimental phase; the official `babel-hungarian` package (https://ctan.org/pkg/bab

el-hungarian) does not include it. Under the proposed development strategy, `magyar.ldf` remains the established, widely used solution, while `hungarian.ldf` serves as a new baseline where more substantial architectural decisions can be made without jeopardizing the large corpus of existing documents; furthermore, users of the legacy setup are not forced into immediate migration.

The experimental `hungarian.ldf` module described in this paper has version v0.1alpha (2026-02-25). The minimum required LaTeX kernel date is 2022-06-01, and the minimum required Babel version is v26.3. It works with pdfLaTeX, LuaLaTeX, and XeLaTeX, and provides Hungarian language support for the following classes and packages:

Classes: `article`, `report`, `book`, `extarticle`, `extreport`, `extbook`, `amsart`, `amsbook`, `memoir`, `scrartcl`, `scrreprt`, `scrbook`

Packages: `titlesec`, `fancyhdr`, `amsthm`, `ntheorem`, `hyperref`, `bookmark`, `listings`, `caption`, `csquotes`, `eukdate`, `footmisc`, `varioref`

We note that `magyar.ldf` does not support the `memoir` class (see [18, 19]) or the KOMA-Script classes (`scrartcl`, `scrreprt`, `scrbook`) (see [6]). On the other hand, support for the AMS classes (see [4]) is more stable in `hungarian.ldf`.

Testing methodology. The compatibility claims above are based on compilation tests covering the three major engines (pdfLaTeX, LuaLaTeX, XeLaTeX) and a representative set of document classes and package combinations. In particular, we verified (i) the number–name order and the trailing full stop in headings and generated lists (toc/lof/lot), (ii) the behavior of `hyperref`/`bookmark` PDF strings, and (iii) the correctness of Hungarian definite article and suffixation logic in cross-references.

- Engines tested: pdfLaTeX, LuaLaTeX, XeLaTeX.
- Classes tested: `article`, `report`, `book`, `extarticle`, `extreport`, `extbook`, `amsart`, `amsbook`, `memoir`, `scrartcl`, `scrreprt`, `scrbook`.
- Critical package sets:
  - Heading and list formatting: `titlesec`, `fancyhdr`, `caption`, `footmisc`.
  - Theorem-like environments: `amsthm`, `ntheorem`.
  - Cross-references and PDF strings: `hyperref`, `bookmark`, and `varioref`.
  - Quotation and verbatim-like contexts: `csquotes`, `listings`.
  - Date handling: `eukdate`.
- Regression checkpoints:
  - Number–name order and trailing full stop in headings and in the generated lists (toc/lof/lot), with consistent behavior across sectioning levels.

– Running headers produced by the class and by `fancyhdr`, including correct number–name order and punctuation.

– PDF bookmarks and other PDF strings produced by `hyperref` and `bookmark`: correct number–name order, punctuation, and the absence of artefacts caused by active characters.

– Cross-references: correctness of Hungarian definite article selection and numeric suffixation logic in references, including **`\Az`**, **`\az`** with **`\autoref`**, **`\autopageref`** (and their starred variants), as well as the corresponding `varioref` commands.

– Active characters: the three functions of the active quotation mark (digraph/trigraph splitting, hyphenated compounds, and name–name en-dash spacing) and the insertion of thin spacing before `!?;:` in text mode, with special attention to robustness in `listings` and other verbatim-like contexts.

– Decimal comma behavior in math mode (`decimalcomma`), distinguishing decimal commas from punctuation commas (e.g., `$1,2$` vs. `$a,b$`).

– Date commands and formats: **`\today`**, **`\ontoday`**, **`\weekday`**, **`\hudate`** and **`\huondate`** for multiple output formats.

– Optional inline-math line breaking (`mathbrk`): compilation and output checks when `rmathbr` is enabled, including interaction scenarios with active punctuation.

Table 1. High-level comparison of the legacy **`magyar.ldf`** module and the experimental **`hungarian.ldf`** baseline.

Aspect	**`magyar.ldf` v1.6a (legacy)**	**`hungarian.ldf` v0.1alpha (experimental)**
Status	Mature, widely used, historically layered	Experimental baseline; intended to stabilize through feedback
Size / option surface	∼224 kB / 5261 lines; 63 options and multiple profiles	Designed as a smaller, more testable core; a limited option set via **`\SetHungarian`**
Identity	Historical naming duality (**`magyar`** vs. **`hungarian`**)	Explicit **`hungarian`** identity and more controlled name resolution
Implementation style	Heavy use of global interventions and **`\AtBeginDocument`**-style patching	Relies on LaTeX kernel, Babel, and package hooks; aims to minimize global side effects

Continued on the next page.

Aspect	**magyar.ldf v1.6a**	**hungarian.ldf v0.1alpha**
Captions / date strings	Legacy mechanisms and option-driven configuration	Uses Babel's newer declarative interfaces for strings and dates
Distribution today	Officially shipped in `babel-hungarian`; current redirect setup makes both `magyar` and `hungarian` load `magyar.ldf`	Not yet shipped as the default; requires a local test setup until official integration
Supported classes (notable points)	No support for `memoir` and KOMA-Script classes (as noted in this paper)	Supports `memoir` and KOMA-Script; AMS-class support reported as more stable
Definite article / suffixation	Historically includes automation of Hungarian-specific phenomena in cross-references	Loads `huaz` and extends automation to `hyperref`/`varioref`
`hyperref` (`\autoref`, `\autopageref`)	Number–name order not implemented	Correct number–name order and suffixation, with definite article selection
`varioref` localization	(Depends on legacy mechanisms / not discussed here in detail)	Provides Hungarian localization with definite article and suffixation logic
Inline-math line breaking	(Not addressed here)	Optional `rmathbr`-based support (`mathbrk`); possible interaction issues with active punctuation
Migration story	Stable for existing documents; broad legacy coverage	Migration guide and an explicit compatibility matrix planned

2.1. Loading hungarian.ldf

In current TEX distributions, the `magyar` and `hungarian` options of the Babel package both resolve to `magyar.ldf`. This behaviour is implemented through the shipped redirect files `babel-magyar.tex` and `babel-hungarian.tex`, each of which points to `magyar.ldf`.

Accordingly, the experimental `hungarian.ldf` can presently be used only in a local test setup. In practice, the user places `hungarian.ldf` next to the document source and substitutes `babel-hungarian.tex` with a modified version that redirects to `hungarian.ldf`. A test bundle containing the v0.1alpha (2026-02-25) sources of `hungarian.ldf` and the corresponding modified `babel-hungarian.tex` redirect file is available at:

https://tibortomacs.github.io/latex-tutorial-hu/hungarianldf.zip

Once `hungarian.ldf` is officially released as part of the babel-hungarian bundle, the above workaround will no longer be required. At that point, the so-called *dual* loading of `magyar.ldf` will also disappear; using `magyar` together with `hungarian` will no longer trigger two consecutive loads governed by different option sets. I do not regard this as a disadvantage: in my experience, it is very difficult to design two such option systems so that they do not interfere with each other, and I have not found a useful practical application for this dual-loading mechanism.

If one uses pdfLaTeX or LaTeX engine, the `fontenc` package must also be loaded with the `T1` option. For example

```
\documentclass{book}
\usepackage[T1]{fontenc}
\usepackage[hungarian]{babel}
\usepackage{hulipsum} % for Hungarian dummy text (\hulipsum)
\title{Cím} % Cím = Title
\author{Szerző} % Szerző = Author
\begin{document}

\maketitle
\tableofcontents
\chapter{Fejezet címe} % Fejezet címe = Chapter title
\section{Szakasz címe} % Szakasz címe = Section title
\hulipsum

\end{document}
```

2.2. Options for the hungarian.ldf

The options for `hungarian.ldf` can be specified as follows after loading Babel:

\SetHungarian{⟨*options*⟩}

The ⟨*options*⟩ are as follows:

`activeqmark`=⟨*boolean*⟩ (default: `true`)

If its value is set to `true`, the `"` character becomes active. Its role is discussed in detail in Subsection 2.4.

`activespace`=⟨*boolean*⟩ (default: `true`)

When this option is set to `true`, the exclamation mark, question mark, semicolon, and colon characters become active. In this case, in text mode, a short horizontal spacing (0.1 em), as recommended by Hungarian typographic conventions, is inserted before these punctuation marks (see [16, p. 29]).

`booknum`={⟨*code*⟩} (default: {**\thebook.\space**})

In the `memoir` class, there is an additional structural division above **\part**, namely **\book**. The ⟨*code*⟩ determines how the book number is typeset before the "könyv" (book) label.

booknumspell=⟨*boolean*⟩ (default: false)

If the value is true, the number of the book is rendered in words within the running text. E.g., "Első könyv" (First book) instead of "1. könyv" (Book 1). This option has no effect on lists (toc, lof, lot), headers, or the PDF bookmarks.

chapternum={⟨*code*⟩} (default: {**\thechapter.\space**})

The ⟨*code*⟩ determines how the chapter number is typeset before the "fejezet" (chapter) label.

chapternumspell=⟨*boolean*⟩ (default: false)

If the value is true, the number of the chapter is rendered in words within the running text. E.g., "Első fejezet" (First chapter) instead of "1. fejezet" (Chapter 1). This option has no effect on lists (toc, lof, lot), headers, or the PDF bookmarks.

decimalcomma=⟨*boolean*⟩ (default: true)

By default, in mathematical mode the comma is treated as a punctuation symbol. Consequently, for example, the output of $1,2$ is $1,2$ (i.e., a small amount of space is inserted after the comma). In Hungarian, however, this behavior is undesirable due to the use of the decimal comma. If the value is true, then in mathematical mode no space is inserted after a comma provided that the comma is immediately followed by a digit; in all other cases, spacing is applied as usual. Thus, the output of $1,2$ becomes 1,2 (whereas $1, 2$ yields $1, 2$, and a,b results in a, b). If the value is false, then the decimal comma must be entered explicitly as {,}, e.g., $1{,}2$.

footnoterule=⟨*boolean*⟩ (default: true)

If the value is true, a horizontal rule is placed above the footnotes; its length equals one quarter of the text block width. If the value is false, no rule is placed above the footnotes. If the footmisc package (see [5]) is also loaded with the norule option, then no rule will be produced even when footnoterule=true.

hyphenmins={⟨*digit1*⟩⟨*digit2*⟩} (default: {22})

The value ⟨*digit1*⟩ specifies the minimum number of characters that must precede the hyphenation point when a word is hyphenated, while ⟨*digit2*⟩ specifies the minimum number of characters that must follow it. In LaTeX, the default setting is 23; however, **hungarian.ldf** changes this to 22.

mathbrk=⟨*boolean*⟩ (default: true)

Hungarian typographic conventions permit line breaks in inline mathematics around binary relations and binary operators (with the exception of the multiplication sign and the slash); however, if a line break occurs, the operator or relation must be repeated at the beginning of the following line. If the value is true, then **hungarian.ldf** loads the rmathbr package (see [10]), which automates this line-breaking rule. Further details are provided in Subsection 2.9.

`partnum=`{⟨*code*⟩} (default: {**\thepart.\space**})

The ⟨*code*⟩ determines how the part number is typeset before the "rész" (part) label.

`partnumspell=`⟨*boolean*⟩ (default: `false`)

If the value is `true`, the number of the part is rendered in words within the running text. E.g., "Első rész" (First part) instead of "1. rész" (Part 1). This option has no effect on lists (toc, lof, lot), headers, or the PDF bookmarks.

`secnumdelim=`{⟨*code*⟩} (default: {**.\enskip**})

A ⟨*code*⟩ specifies the code inserted between the numbers of sections, subsections, etc., and their titles.

`tocnumdelim=`{⟨*code*⟩} (default: {.})

The ⟨*code*⟩ specifies the code that appears in lists (e.g., the table of contents) immediately after the numbers of structural units.

2.3. Spacing after periods

By default, LaTeX follows English typographic conventions by inserting a larger inter-sentence space after periods that end sentences than the normal inter-word space. In Hungarian typography, this is not permitted; therefore, **hungarian.ldf** activates **\frenchspacing**, which enforces normal spacing after periods as well.

2.4. The active role of the " character

The `activeqmark=true` option makes the " character active, enabling it to perform three functions.

2.4.1. Hyphenation of double digraph/trigraph consonant clusters

By default, hyphenation is not possible in the presence of double digraph/trigraph consonant clusters (ccs, ddz, ddzs, ggy, lly, nny, ssz, tty, zzs); for instance, the Hungarian word "mennyiség" ("quantity") can only be hyphenated as "mennyi-ség". If, in a given context, this does not yield optimal line breaking, the hyphenation of the double digraph (trigraph) consonant cluster can be specified locally:

```
me\discretionary{ny-}{ny}{nny}iség
```

When the `activeqmark=true` option is enabled, this can be achieved more simply:

```
"ccs "ddz "ddzs "ggy "lly "nny "ssz "tty "zzs
"CCS "DDZ "DDZS "GGY "LLY "NNY "SSZ "TTY "ZZS
```

For example,

```
me"nnyiség
```

the hyphenation will already follow the pattern “meny-nyi-ség”. Irrespective of the value of `activeqmark`, the following commands may also be used for the same purpose:

`\ccs \ddz \ddzs \ggy \lly \nny \ssz \tty \zzs`

`\CCS \DDZ \DDZS \GGY \LLY \NNY \SSZ \TTY \ZZS`

In accordance with LaTeX conventions, these commands must be inserted into running text in such a way that they are followed by either a space or a pair of curly braces; alternatively, the command itself should be enclosed in curly braces. For example,

```
me\nny iség  =  me\nny{}iség  =  me{\nny}iség
```

In the following example, the only Hungarian double triple consonant, “ddzs”, is also included. The first line can be used only when `activeqmark=true`.

```
E"ddzünk bri"ddzsel! % = Let's practice playing bridge!
E\ddz{}ünk bri\ddzs{}el!
E\discretionary{dz-}{dz}{ddz}ünk
bri\discretionary{dzs-}{dzs}{ddzs}el!
```

Of course, this is only worth doing if it results in more optimal line breaking at a given point.

2.4.2. Hyphenated compound words

Hyphenated compound words, such as “vízvezeték-szerelő” (in English: plumber), can, by default in LaTeX, be broken only at the hyphen. If this does not produce optimal line breaking at a given point, then – when the option `activeqmark=true` is enabled – insert the `"` character immediately before the hyphen:

`"-`

For example, after

```
vízvezeték"-szerelő
```

the hyphenation pattern “víz-ve-ze-ték-sze-re-lő” will be in effect. Of course, in hyphenated compounds the internal hyphen (i.e., the hyphen that is part of the compound itself) is always retained and remains visible, regardless of where the word is hyphenated at a line break.

`\hyp`

Irrespective of the value of the `activeqmark` option, the **`\hyp`** command may be used instead of `"-`:

```
vízvezeték\hyp szerelő
vízvezeték\hyp{}szerelő
vízvezeték{\hyp}szerelő
```

2.4.3. En dash between names

In the case of an en dash used between personal names, a more aesthetically pleasing result can be achieved by inserting a thin, non-breaking space before and after the dash. For example

```
Newton\,--\,Leibniz-tétel
```

When the `activeqmark=true` option is enabled, the following can also be used instead of `\,--\,`:

`"--`

For example

```
Newton"--Leibniz-tétel
```

`\hyp-`

Irrespective of the value of the `activeqmark` option, the **`\hyp-`** command may be used instead of `\,--\,`. For example

```
Newton\hyp-Leibniz-tétel
```

2.5. Dates

The following date-related commands are available:

`\today`

If, for instance, the LaTeX document is converted to PDF on February 26, 2026, then the output of **`\today`** is the Hungarian equivalent of that date: 2026. február 26.

`\ontoday`

In the preceding example, this command yields: 2026. február 26-án (in English: on February 26, 2026).

`\weekday`

If, for instance, the LaTeX document is converted to PDF on Thursday, then the output of this command is: csütörtök.

`\hudate`{⟨*letter*⟩}{**`\today`**}

In place of ⟨*letter*⟩, the following letters may be used: `a`, `b`, `c`, `d`, `e`, `f`. Their respective outputs are:

`\hudate{a}{\today}`	2026-02-26
`\hudate{b}{\today}`	2026. február 26. (= `\today`)
`\hudate{c}{\today}`	2026. feb. 26.
`\hudate{d}{\today}`	2026. II. 26.
`\hudate{e}{\today}`	2026. 02. 26.
`\hudate{f}{\today}`	2026. február

`\hudate{`⟨*letter*⟩`}{`⟨*yyyy*⟩`-`⟨*mm*⟩`-`⟨*dd*⟩`}`

In place of ⟨*letter*⟩, the following letters may be used: `a`, `b`, `c`, `d`, `e`, `f`. In the placeholder ⟨*yyyy*⟩, the year is to be inserted; in ⟨*mm*⟩, the month number is to be given using one or two digits; and in ⟨*dd*⟩, the day number is to be given using one or two digits. For example

`\hudate{a}{1848-3-15}`	1848-03-15
`\hudate{b}{1848-3-15}`	1848. március 15.
`\hudate{c}{1848-3-15}`	1848. márc. 15.
`\hudate{d}{1848-03-15}`	1848. III. 15.
`\hudate{e}{1848-03-15}`	1848. 03. 15.
`\hudate{f}{1848-03-15}`	1848. március

`\huondate{`⟨*letter*⟩`}{\today}`

In place of ⟨*letter*⟩, the following letters may be used: `b`, `c`, `d`, `e`. Their respective outputs are:

`\huondate{b}{\today}`	2026. február 26-án (= `\ontoday`)
`\huondate{c}{\today}`	2026. feb. 26-án
`\huondate{d}{\today}`	2026. II. 26-án
`\huondate{e}{\today}`	2026. 02. 26-án

`\huondate{`⟨*letter*⟩`}{`⟨*yyyy*⟩`-`⟨*mm*⟩`-`⟨*dd*⟩`}`

In place of ⟨*letter*⟩, the following letters may be used: `b`, `c`, `d`, `e`. In the placeholder ⟨*yyyy*⟩, the year is to be inserted; in ⟨*mm*⟩, the month number is to be given using one or two digits; and in ⟨*dd*⟩, the day number is to be given using one or two digits. For example

`\huondate{b}{1848-3-15}`	1848. március 15-én
`\huondate{c}{1848-3-15}`	1848. márc. 15-én
`\huondate{d}{1848-03-15}`	1848. III. 15-én
`\huondate{e}{1848-03-15}`	1848. 03. 15-én

2.6. Automatic definite article and number suffixation

The `hungarian.ldf` automatically loads the `huaz` package (see [14, 15]), which can be used to automatically specify definite article and number suffixes.

2.7. The hyperref package's `\autoref` and `\autopageref` commands

The `hyperref` package (see [9]) translates the names of the structural elements produced by the `\autoref`, `\autoref*`, `\autopageref`, and `\autopageref*` commands into Hungarian; however, it does not implement the required inversion of

the ordinal number and the element name, and therefore the default output of these commands does not conform to Hungarian typographic conventions. The `hungarian.ldf` (unlike `magyar.ldf`) resolves this issue, and automatic definite article selection works for these references as well. For example, compiling the following minimal document

```
\documentclass{article}
\usepackage[T1]{fontenc}
\usepackage[hungarian]{babel}
\usepackage{hyperref}
\begin{document}

\section{Cím}\label{a}
\Az{\autoref*{a}}\\
\az{\autopageref*{a}}

\end{document}
```

yields the following output:

Az 1. szakasz
az 1. oldal

In English:

The section 1
the page 1

The `huaz` package provides the **\husuffix**, **\ahusuffix**, and **\Ahusuffix** commands for the automatic suffixation of numerals; however, `hungarian.ldf` extends this functionality to the **\autoref** and **\autopageref** commands and their starred variants as well:

\husuffix{**\autoref**{⟨*label*⟩}}{⟨*suffix*⟩}

\ahusuffix{**\autoref**{⟨*label*⟩}}{⟨*suffix*⟩}

\Ahusuffix{**\autoref**{⟨*label*⟩}}{⟨*suffix*⟩}

\husuffix{**\autopageref**{⟨*label*⟩}}{⟨*suffix*⟩}

\ahusuffix{**\autopageref**{⟨*label*⟩}}{⟨*suffix*⟩}

\Ahusuffix{**\autopageref**{⟨*label*⟩}}{⟨*suffix*⟩}

The admissible values of the ⟨*suffix*⟩ argument coincide with those of the `huaz` package (with the exception of `as` and `es`, since, for example, “1. szakaszos” or “1. ábrás” would be meaningless):

`at et ot` (what)

`val vel` (with what)

`tol` (from what)

`rol` (about what)

`bol` (from what)

`ba be` (into what)

`ban ben` (in what)

`ra re` (to/onto what)

`nak nek` (to/for what)

`nal nel` (at/than what)

`hoz hez` (to what)

The interrogative word associated with the suffix is given in parentheses. Any of the forms listed on the same line may be used and may even be written in capital letters. For example, compiling the following minimal document

```
\documentclass{article}
\usepackage[T1]{fontenc}
\usepackage[hungarian]{babel}
\usepackage{hyperref}
\begin{document}

\section{Cím}\label{a}
\Ahusuffix{\autoref*{a}}{val}

\end{document}
```

yields the following output (in English: “With Section 1”):

Az 1. szakasszal

These commands may be used only to a limited extent within sectioning commands (`\part`, `\chapter`, `\section`, etc.); they should preferably be avoided.

2.8. The varioref package

The Hungarian localization of the `varioref` package (see [8]) has been implemented using automatic definite article and numeric suffixation selection. For example, compiling the following minimal document

```
\documentclass{article}
\usepackage[T1]{fontenc}
\usepackage[hungarian]{babel,varioref}
\begin{document}

Lásd \vref{a}\\
Lásd \vpageref{a}\\
```

```
Lásd \vpageref{c}\\
\vpagerefrange{a}{b}\\
\vrefrange{a}{b}

\section{Cím}\label{a}
\newpage
\section{Cím}\label{b}
\newpage
\section{Cím}\label{c}

\end{document}
```

yields the following output:

Lásd 1
Lásd ezen az oldalon
Lásd a 3. oldalon
az 1–2. oldalakon
1-től 2-ig az 1–2. oldalakon

In English:

See 1
See on the current page
See on page 3
on pages 1–2
1 to 2 on pages 1–2

2.9. Line breaking of inline formulae

The `mathbrk=true` option loads the `rmathbr` package (see [10]), which enables the automatic repetition of binary relations and binary operators in inline mathematical formulae when a line break occurs, placing the repeated symbol at the beginning of the next line.

The `rmathbr` package is sensitive to cases where the colon and the semicolon are active characters (as with the `activespace=true` option). This is mitigated by **`hungarian.ldf`**, yet incompatibilities may still arise. In such situations, with `mathbrk=false`, you may locally use the

`\mbrk`

command immediately before the symbols that should be repeated at the break. For example

```
$a+b\mbrk=c+d$
```

This yields

$a + b = c + d$

however, if a line break is required at the equality sign:

$$a + b = \\ = c + d$$

With the `mathbrk=true` option, the **`\mbrk`** command is not required.

2.10. The tangent and cotangent functions

In Hungarian usage, the tangent and cotangent functions, as well as their inverse (arc) and hyperbolic variants, are denoted differently from the Anglophone convention. Therefore, the following function commands have been introduced:

`\tg`	tangent
`\th`	hyperbolic tangent
`\arctg`	arctangent
`\ctg`	cotangent
`\cth`	hyperbolic cotangent
`\arcctg`	arccotangent

These commands are provided as Hungarian language extras, i.e., they are defined when Hungarian is the active Babel language. When switching to another language, Babel restores any previous meanings of these macros.

3. Future directions

At present, **`hungarian.ldf`** is an experimental development. Our goal is a smaller, more transparent, and more easily testable Hungarian Babel module that relies on the modern LaTeX/Babel infrastructure and serves Hungarian typography with as few global side effects as possible. We regard the following development directions as the most important.

Official distribution and loading. The most immediate goal is to ensure that **`hungarian.ldf`** can be installed without requiring users to copy files manually or to modify `babel-hungarian.tex`. Technically, this entails making the `hungarian` option load **`hungarian.ldf`** unambiguously, while the `magyar` option loads **`magyar.ldf`**. This separation allows the two modules to coexist cleanly, each with a distinct identity.

A stable surface API. Finalize the option system built around **`\SetHungarian`** (naming, defaults, and documented side effects), and specify what is considered a supported interface that is intended to be maintained in the long term.

Regression tests and automated checking. A prerequisite for further extension of the module is an extensive test suite. The goal is a CI (Continuous Integration) workflow that automatically runs the tests on every change and flags breakages.

Extending the set of supported packages. Short-term candidates include additional elements of the referencing and labeling ecosystem (e.g. `cleveref`, `nameref`), as well as the typical Hungarian-language requirements of bibliographic and list-management packages.

Refining typographic details. The aim is not the immediate reproduction of the full historical option set of `magyar.ldf`, but the gradual, well-tested incorporation of the most frequently used features.

Migration and coexistence. In the longer term, `hungarian.ldf` will be genuinely useful only if it offers a clear migration path: which `magyar.ldf` options have counterparts, where deliberate differences exist, and in which cases it remains advisable to choose `magyar.ldf`. The goal is to maintain a short "migration guide" and a compatibility matrix.

Development would be greatly facilitated by user feedback and the collection of concrete issue reports – especially minimal examples exhibiting package conflicts, or anomalies at the level of headings and generated lists.

4. Summary

The paper provided a brief historical overview of Hungarian language support in Babel and of the main milestones in the evolution of `magyar.ldf`, and, in this context, introduced the experimental `hungarian.ldf` language module. The motivation is twofold: on the one hand, to reduce the complexity and maintenance burden arising from the historical stratification of `magyar.ldf`; on the other hand, to develop a solution built on a modern LaTeX/Babel infrastructure, organizing the required Hungarian typographic behavior around a smaller, more transparent, and more readily testable core.

A central design principle of `hungarian.ldf` is that, wherever possible, it should rely on more declarative Babel mechanisms and on LaTeX kernel/package hooks (instead of classical, global `\AtBeginDocument`-style patching), thereby making the timing of interventions more robust and reducing the likelihood of ordering regressions. The module provides a unified option interface via the `\SetHungarian` command, and addresses several details that are critical in Hungarian:

- making the spacing after a sentence-ending period conform to Hungarian conventions (`\frenchspacing`) and, optionally, inserting the thin space customary in Hungarian typography before `!?;:`;
- the active role of the quotation mark (`"`) (e.g. a concise marker for splitting double/triple consonants; see Subsection 2.4);

- support for multiple date formats and additional date-related commands;
- automatic selection of the definite article and suffixation of numerals via integration with the `huaz` package;
- improving the Hungarian localization of the `varioref` package and **`\autoref`**, **`\autopageref`** in `hyperref`, ensuring that both the number–name order and suffixation work correctly (see Subsections 2.7 and 2.8);
- optional support for line breaking in inline mathematics (`mathbrk`; see Subsection 2.9);
- introducing **`\tg`**, **`\ctg`** (and related) function macros in accordance with Hungarian mathematical notation.

The proposed approach does not aim at the immediate reproduction of the full historical option set and compatibility legacy of **`magyar.ldf`**; rather, it seeks to establish a foundation that can cooperate more predictably with modern document classes and typical package combinations. At present, **`hungarian.ldf`** remains experimental; based on practical experience and issue reports, the goal is gradual stabilization and, in the longer term, incorporation into the official package.

References

[1] J. Bezos, J. L. Braams: *Babel Code, Localization and internationalization*, The Comprehensive TEX Archive Network (2026), URL: https://ctan.org/pkg/babel.

[2] J. Bezos, J. L. Braams: *Babel User guide, Localization and internationalization*, The Comprehensive TEX Archive Network (2026), URL: https://ctan.org/pkg/babel.

[3] G. Bujdosó, F. Wettl: *On the localization of TEX in Hungary*, TUGboat 23.1 (2002), pp. 21–26, URL: https://www.tug.org/TUGboat/tb23-1/tb73complete.pdf.

[4] M. Downes, B. Beeton: *The amsart, amsproc, and amsbook document classes*, The Comprehensive TEX Archive Network (2020), URL: https://ctan.org/pkg/amscls.

[5] R. Fairbairns, F. Mittelbach: *footmisc — a portmanteau package for customizing footnotes in LATEX*, The Comprehensive TEX Archive Network (2025), URL: https://ctan.org/pkg/footmisc.

[6] M. Kohm: *The Guide KOMA-Script*, The Comprehensive TEX Archive Network (2025), URL: https://ctan.org/pkg/scrartcl.

[7] L. Lamport: *LATEX: A document preparation system, User's guide and reference manual*, 2nd edition, Addison Wesley, 1994, ISBN: 0201529831, 978-0201529838.

[8] F. Mittelbach: *The varioref package*, The Comprehensive TEX Archive Network (2025), URL: https://ctan.org/pkg/varioref.

[9] S. Rahtz, H. Oberdiek: *Hypertext marks in LATEX: a manual for hyperref*, The Comprehensive TEX Archive Network (2022), URL: https://ctan.org/pkg/hyperref.

[10] D. Ryabov: *The rmathbr package*, The Comprehensive TEX Archive Network (2020), URL: https://ctan.org/pkg/rmathbr.

[11] P. Szabó: *Implementation Tricks in the Hungarian Babel Module*, TUGboat 25.0 (2004), pp. 140–161, URL: https://tug.org/TUGboat/tb25-0/szabo.pdf.

[12] P. Szabó: *Typing Hungarian text with MagyarLaTeX*, Hungarian (2009), URL: https://math.bme.hu/latex/magyarldf-doc.pdf.

[13] The LaTeX3 Project Team: *LaTeX News, Issues 1–42*, The Comprehensive TeX Archive Network (2025), URL: https://www.latex-project.org/news/latex2e-news/ltnews.pdf.

[14] T. Tómács: *On a new LaTeX package for automatic Hungarian definite article*, Annales Mathematicae et Informaticae 55 (2022), pp. 208–221, DOI: 10.33039/ami.2022.10.001.

[15] T. Tómács: *The huaz package*, The Comprehensive TeX Archive Network (2026), URL: https://www.ctan.org/pkg/huaz.

[16] P. Virágvölgyi: *A tipográfia mestersége számítógéppel*, Budapest: Osiris Kiadó, 2004.

[17] F. Wettl, Gy. Mayer, P. Szabó: *LaTeX kézikönyv*, Hungarian, Budapest, Hungary: Panem, 2004, ISBN: 963 545 398 1.

[18] P. Wilson: *The memoir class*, The PracTeX Journal 3 (2006), URL: https://ctan.org/pkg/memoir.

[19] P. Wilson, L. Madsen: *The Memoir Class for Configurable Typesetting, User Guide*, The Comprehensive TeX Archive Network (2025), URL: https://ctan.org/pkg/memoir.