Nuspell: the new spell checker

FOSS spell checker implemented in C++14 with aid of Mozilla.

Sander van Geloven

FOSDEM, Brussels

February 2, 2019

Nuspell

Nuspell is

- spell checker
- free and open source software with LGPL
- library and command-line tool
- written in C++14

Nuspell – Team

Our team currently consists of

- Dimitrij Mijoski
  - lead software developer
  - github.com/dimztimz
- Sander van Geloven
  - information analyst
  - hellebaard.nl
  - linkedin.com/in/svgeloven
  - github.com/PanderMusubi

Nuspell – Spell Checking

Spell checking is not trivial

- much more than searching an exhaustive word list
- dependent of language, character encoding and locale
- involves case conversion, affixing, compounding, etc.
- suggestions for spelling, typing and phonetic errors
- long history over decades with `spell`, `ispell`, `aspell`, `myspell`, `hunspell` and now `nuspell`

See also my talk at FOSDEM 2016 archive.fosdem.org/2016/schedule/event/integrating_spell_and_grammar_checking

Nuspell – Goals

Nuspell's goals are

- a drop-in replacement for browsers, office suites, etc.
- backwards compatibility MySpell and Hunspell format
- improved maintainability
- minimal dependencies
- maximum portability
- improved performance
- suitable for further optimizations

Realized with an object-oriented C++ implementation.

Nuspell – Features

Nuspell supports

- many character encodings
- compounding
- affixing
- complex morphology
- suggestions
- personal dictionaries
- 167 (regional) languages via 89 existing dictionaries

Nuspell – Support

Mozilla Open Source Support (MOSS) funded in 2018 the creation of Nuspell. Thanks to Gerv Markham† and Mehan Jayasuriya. See mozilla.org/moss for more information.

moz://a

Verification Hunspell has a mean precision of 1.000 and accuracy of 0.997. Perfect match 70% of tested languages. On average checking 30% faster and suggestions 8x faster.

Workings – Spell Checking

Spell checking is highly complex and unfortunately not suitable for a lightning talk. It mainly concerns

- searching strings
- using simple regular expressions
- locale-dependent case detection and conversion
- finding and using break patterns
- performing input and output conversions
- matching, stripping and adding (multiple) affixes, mostly in reverse
- compounding in several ways, mostly in reverse
- locale-dependent tokenization of plain text

Workings – Case Conversion

Examples of non-trivial case detection and conversion

- `to_title("istanbul")` → English `"Istanbul"`
  Turkish `"İstanbul"`

  `to_upper("Diyarbakır")` → English `"DIYARBAKIR"`
  Turkish `"DİYARBAKIR"`
- `to_upper("σίγμα")` → Greek `"ΣΙΓΜΑ"`
  `to_upper("ςίγμα")` → Greek `"ΣΙΓΜΑ"`
  `to_lower("ΣΙΓΜΑ")` → Greek `"ςίγμα"`
- `to_upper("Straße"` → English `Straße"`
  `to_upper("Straße"` → German `STRASSE"`
- `to_title("ijsselmeeer")` → English `"Ijsselmeer"`
  `to_title("ijsselmeeer")` → Dutch `"IJsselmeer"`

Workings – Suggestions

Suggestions are currently found in the following order

1. replacement table	h[ëê]llo → hello
2. mapping table	hełło$ → hello
3. extra character	hhello → hello
4. keyboard layout	hrllo → hello
5. bad character	hellø → hello
6. forgotten character	hllo → hello
7. phonetic mapping	ˆello → hello

Workings – Initialization

Initialize Nuspell in four steps in C++

- find, get and load dictionary

```
auto find = Finder::search_all_dirs_for_dicts();
auto path = find.get_dictionary_path("en_US");
auto dic = Dictionary::load_from_path(path);
```

- associate currently active locale

```
boost::locale::generator gen;
auto loc = gen("");
dic.imbue(loc);
```

These steps are more simple when using the API.

Workings – Usage

Use Nuspell by simply calling to

- check spelling

```
auto spelling = false;
spelling = dic.spell(word);
```

- find suggestions

```
auto suggestions = List_Strings();
dic.suggest(word, suggestions);
```

Technologies – Libraries

Libraries used in run-time

- C++14 library
  e.g. GNU Standard C++ Library
  libstdc++ ≥ 7.0
- Boost.Locale
  C++ facilities for localization
  boost-locale ≥ 1.62
- International Components for Unicode (ICU)
  a C++ library for Unicode and locale support
  icu ≥ 57.1

Technologies – Compilers

Currently supported compilers to build Nuspell

- GNU GCC compiler g++ ≥ 7.0
- LLVM Clang compiler clang ≥ 6.0

Upcoming supported compilers

- MinGW with MSYS mingw
- GNU GCC compiler 6.0 (backport)

Technologies – Tools

Tools used for development

- build tools such as Autoconf, Automake, Make, Libtool and pkg-config
- QtCreator for development and debugging, also possible with gdb and other command-line tools
- unit testing with Catch2
- continuous integration with Travis for GCC and Clang and coming soon AppVeyor for MinGW
- profiling with Callgrind, KCachegrind, Perf and Hotspot
- API documentation generation with Doxygen
- code coverage reporting with LCOV and genhtml

Upcoming – Next Version

Next version will have improved

- performance
- compounding
- suggestions
- API
- command-line tool
- documentation
- testing

Nuspell will then also be

- migrated to CMake
- integrated with web browsers
- offering ports and packages
- offering language bindings

Upcoming – Ports and Packages

Supported

- Ubuntu ≥ 18.04 LTS (Bionic Beaver)
- Debian ≥ 9 (Stretch)

Tested

- FreeBSD ≥ 11

Help wanted

- Android
- Arch Linux
- CentOS
- Fedora
- Gentoo
- iOS
- Linux Mint
- macOS
- NetBSD
- OpenBSD
- openSUSE
- Slackware
- Windows
- ...

Upcoming – Language Bindings

Supported

- C++
- C

Help wanted

- C#
- Go
- Java
- JavaScript
- Lua
- Objective-C
- Perl
- PHP
- Ruby
- Rust
- Python
- Scala
- ...

Upcoming – Miscellaneous

Other ways to help are

- fix bugs in dictionaries and word lists
- improve dictionaries and word lists
- contribute word lists with errors and corrections
- integrate Nuspell with IDEs, text editors and editors for HTML, XML, JSON, YAML, TEX, etc.
- integrate Nuspell with Enchant e.g. for GtkSpell
- sponsor our team
- join our team

Upcoming – Info and Contact

nuspell.github.io

twitter.com/nuspell1

facebook.com/nuspell

fosstodon.org/@nuspell

Big thank you to Dimitrij.

Contact us to support the development, porting and maintenance of Nuspell.

Thanks for your attention.