Tooling for internationalisation of R help pages

Elio Campitelli and Renata Hirota

2023-09-28

Signatories

Project team

- Elio Campitelli: Ph.D. student at the University of Buenos Aires in atmospheric sciences and an R package developer. They maintain several open-source R packages (e.g., ggnewscale; metR) and contribute to other packages, such as data.table and ggplot2. They contributed to the translation of the book R for Data Science to Spanish, and to the translation of rOpenSci materials, including rOpenSci Packages: Development, Maintenance, and Peer Review.
- Renata Hirota

Contributors

- Maëlle Salmon: research software engineer at rOpenSci, previously in charge of four ISC funded projects (Catalyzing R-hub adoption through R package developer advocacy, HTTP testing in R, R-Ladies organizational guidance, Tooling and Guidance for Translations of Markdown-Based R Content (Quarto, R Markdown)) funded by the ISC, R blogger, volunteer editor for rOpenSci's system of package peer-review.
- Yanina Bellini Saibene: Community Manager at rOpenSci, R-Ladies Project Lead, Member of The Carpentries Executive Council and RForwards Core Team. Co-founder of MetaDocencia, LatinR, and R-Ladies Santa Rosa. She lead the collaborative translation of Teaching Teach Together to Spanish and MetaDocencia educational materials to Portuguese. She was involved in the translation to Spanish of R for Data Science, R-Ladies's Rules and Guidelines, some lessons by The Carpentries and severals RStudio Cheat Sheets. She is leading the Multilingual Publishing project at rOpenSci

Consulted

- Yanina Bellini Saibene

The Problem

English is currently the de-facto international language and this is reflected in R function and variable names (the `mean()` function is not called `promedio()` or `Mittelwert()`) and documentation language. And while contributed packages have a "Language" field and can be documented in other languages, the vast majority of contributed packages are documented in English.

There is a small number of packages documented in other languages, presumably in accordance to their target audience. For example, labstatR is a companion package for the Italian book Laboratorio Di Statistica Con R and is documented in Italian. The chilemapas package provides various simplified maps for Chile and it's documentation and function names are in Spanish.

These packages can be more accessible for their intended audience, but they are much less accessible to the wider community. Useful functions/algorithms implemented in those packages would be hard to use for the rest of the community. So package authors face the decision of either making their package inaccessible to their target demographic or being isolated from the wider ecosystem.

Real cases of this issue exists. For example, the utilsIPEA package, which implements functions used by the Brazilian Instituto de Pesquisa Economica Aplicada, is documented in English and its functions names are a mix of English and Portuguese. The author publicly expressed the need for bilingual documentation:

> I am writing a package to facilitate importing Brazilian socio-economic microdata sets (Census, PNAD, etc). I foresee two distinct groups of users of the package:
>
> - Users in Brazil, who may feel more at ease with the documentation in Portuguese. The probably can understand English to some extent, but a foreign language would probably make the package feel less "ergonomic".
> - The broader international users community, from whom English documentation may be a necessary condition.
>
> Is it possible to write a package in a way that the documentation is "bilingual" (English and Portuguese), and that the language shown to the user will depend on their country/language settings?

Moreover, CRAN hosts at least two packages that have a secondary package version with documentation in another language. The ExpDes package has the companion package ExpDes.pt with documentation in Portuguese. Similarly, the orloca package is documented in Spanish in the orloca.es package.

Needless to say, this method of bilingual documentation is not recommended, as it's very hard to maintain and doesn't scale to other languages. A better alternative would be for R to allow packages to have multilingual documentation.

The proposal

Overview

The idea for this proposal was born at the R Project Sprint 2023 in conversation with R Core members Martin Mächler, Deepayan Sarcar and Michael Lawrence.

The idea is to extend the R help system to allow for multiple help pages for the same function in different languages by installing translation modules. By default, `help()` would show the documentation in the preferred language of the user, if available, and fall-back to the canonical documentation otherwise (most likely, in English). It would also include a link to the canonical documentation and warnings if translations are out of date.

Detail

- Original packages have their "canonical" help pages written in the original languages as usual (e.g. `mein_paket` is documented in German).
- Translated documentation to a language is hosted in a translation module (e.g. `mein_paket.en` would provide English documentation for `mein_paket`).
  - These modules are R packages that use the PackageType field to indicate that they are a translation module (e.g. `PackageType: translation`). The Depends field is used to indicate the package being translated and the minimum version supported.
  - They store .po files with the translated strings.
  - The user can install translation modules using `install.packages()` to get the documentation in that language.

  - Translation modules are not necessarily maintained by the original package authors. In principle, multiple translation modules for the same package and the same language could exist and the user can chose which version they install.
  - These modules could be hosted on CRAN or a CRAN-like repository. Optionally, CRAN could develop quality checks on translation modules to ensure that they are up-to-date, such as a maximum percentage of missing strings or maximum percentage of fuzzy matches.

- When a translation module is installed, the .Rd files of the original package are parsed and translated using `gettext()` and the .po files in the translation module, and serialised into binary help pages (like regular packages have).
- When loading a package, R will also search for installed translations and load them too.
- `help()` gains an new "language" argument which defaults to `Sys.getenv("LANGUAGE")`.
- `help()` searches for the loaded topics. If any translation is available, then it would use the `language` argument to disambiguate.
- Help pages would be modified to include include a link to the original (canonical) documentation and add warnings if strings are untranslated/outdated.

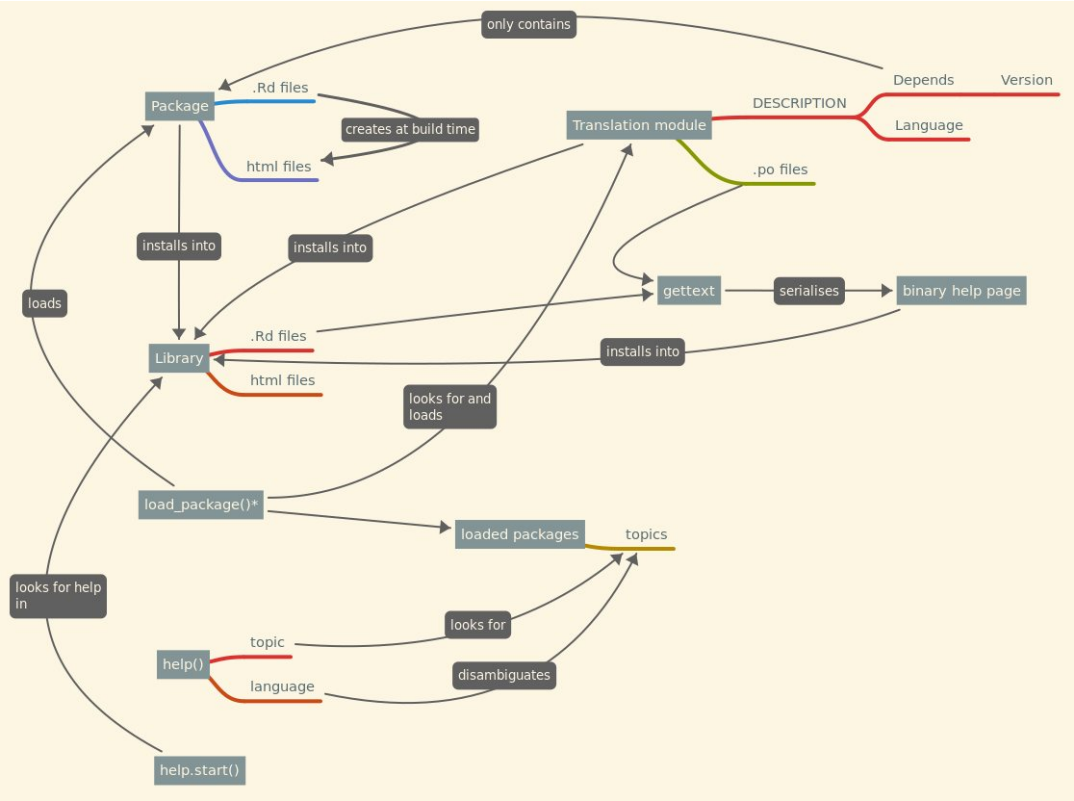

Figure 1: Concept map of the installation of translation modules and access to their translated help.

Project plan

Start-up phase

Before starting the project proper we will consult on the R help system internals, .Rd parsing, and .pot creation and translation. Depending on scheduling with the people involved, this could take around 2 weeks.

Technical delivery

Target date is realtive to after the start-up phase.

Delivery	Hours	Target date
.Rd parsing.	25	2 weeks
Creation of .pot files	10	3 weeks
Translation of .Rd files using .po files	10	4 weeks
Installation of translation modules	30	6 weeks
Rendering of help pages in selected language	30	8 weeks

Other aspects

We will promote the work via blogposts in the rOpenSci blog during development and announce the package when it's done. 16 hours in total.

We will also write an article for JOSS. 20 hours.

Requirements

People

Elio Campitelli and Renara Hirota would be responsible for implementation. Several other people would give their know as consultants: Michael Lawrence would serve as R Core contact, Gergely Daroczi and Michael Chirico (?) would add their expertise with .po files and the `gettext()` infrastructure, and Maëlle Salmon would help with their knowledge of translation and internationalisation workflows.

Processes

The package is covered by rOpenSci Code of Conduct.

Tools & Tech

We would use a GitHub repository for collaboration and code hosting.

Funding

Summary

Success

Definition of done

Success of the project will be to have one or more R packages that implement the following functionality:

1. Parsing .Rd files into .pot files.
2. Translation of .Rd files using .po files.
3. Installation of translation modules.
4. Access of translated documentation using a `help()`-like function.
5. Modification of rendered help pages to include links to the canonical documentation.

Additionally, dummy packages to showcase and test the functionality will be created.

Measuring success

Use of this functionality in a real-world package.

Future work

Test in real world packages and workflows and gather feedback to improve the system. Make changes in the help rendered by `help.start()` to support multiple languages. Once the project is mature, incorporate the functionality into R itself.

Key risks

.Rd files can be very complex and even dynamically created. Parsing them could be to hard or brittle to allow for robust creation of .pot files. Alternatively, the `gettext()` system is designed for short pieces of text and might not prove ergonomical. In those cases, an alternative system relying on whole .Rd files hosted in translation modules could be explored.