Package 'glossr'

August 19, 2022

Type Package

Title Use Interlinear Glosses in R Markdown

Version 0.6.0

Description Read examples with interlinear glosses from files or from text and print them in a way compatible with both Latex and HTML outputs.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports tibble, dplyr, knitr, magrittr, purrr, rlang, stringr, flextable, tidyr, cli

RoxygenNote 7.2.1

Config/testthat/edition 3

VignetteBuilder knitr

Suggests bookdown, officedown, testthat, rmarkdown, officer, htmltools

Depends R (>= 2.10)

URL https://montesmariana.github.io/glossr/, https://github.com/montesmariana/glossr

Language en-GB

NeedsCompilation no

Author Mariana Montes [aut, cre] (<https://orcid.org/0000-0002-3869-3207>), Benjamin Chauvette [cph] (Author of included leipzig.js library)

Maintainer Mariana Montes <montesmariana@gmail.com>

Repository CRAN

Date/Publication 2022-08-18 23:00:02 UTC

R topics documented:

`as_gloss` *Helper to create* `gloss` *objects*

Description

Based on a character vectors and up to three label arguments, create an object where those arguments are attributes. These are:

- **source**: Where the text comes from. This will be printed in the first line of the example, without word alignment.
- **translation**: Free translation. This will be printed as the last line of the example, without word alignment and in quotation marks if so desired.
- **label**: Named label of the example, for cross-references.
- **lengths**: This is computed within the function, not provider, and it's the number of items identified in each gloss line.

Usage

```
as_gloss(
  ...,
  source = NULL,
  translation = NULL,
  label = NULL,
  trans_quotes = getOption("glossr.trans.quotes", "\""),
  output_format = getOption("glossr.output", "latex"),
  numbering = getOption("glossr.numbering", TRUE)
)
```

Arguments

...	Lines for glossing
source	(Optional) Source of example
translation	(Optional) Free translation
label	(Optional) Example label
trans_quotes	(Optional) Quotes to surround the free translation with.
output_format	(Optional) Whether it will use latex, word or html format.
numbering	(Optional) Whether the gloss should be numbered (in Word and HTML).

Value

Object of class gloss, ready to be printed based on the chosen output format, and with a gloss_data object as data attribute (or, in the case of calls via gloss_df(), the original input asdata).

Examples

```
ex_sp <- "Un ejemplo en español"
ex_gloss <- "DET.M.SG example in Spanish"
ex_trans <- "An example in Spanish"
my_gloss <- as_gloss(ex_sp, ex_gloss, translation = ex_trans, label="ex1")

# check the gloss data
attr(my_gloss, "data")
```

---

gloss *Reference gloss*

Description

Latex output uses \@ref(label) to reference examples, whereas HTML output is based on pandoc examples, i.e. (@label). `r gloss(label)`, written inline in the text, will return the appropriate reference based on the selected output.

Usage

```
gloss(label)
```

Arguments

label	Label for reference

Value

Character string with label reference

glosses *Examples of glosses*

Description

A dataset containing five glossing examples extracted from Koptjevskaja-Tamm (2015)'s *The Linguistics of Temperature* and chapters within.

Usage

```
glosses
```

Format

A `tibble::tibble` with 5 rows and 6 variables:

original The text in the original language.

parsed The text with translations to English or morphological annotation per word or expression, with LaTeX formatting.

translation Free translation to English.

label Label for referencing the example.

language Original language of the text.

Source Where the example was taken from (published paper).

gloss_data `gloss_data` *class*

Description

Based on a character vectors and up to three label arguments, create an object where those arguments are attributes. These are:

- **source**: Where the text comes from. This will be printed in the first line of the example, without word alignment.
- **translation**: Free translation. This will be printed as the last line of the example, without word alignment and in quotation marks if so desired.
- **label**: Named label of the example, for cross-references.
- **lengths**: This is computed within the function, not provider, and it's the number of items identified in each gloss line.

Usage

```
new_gloss_data(
  gloss_lines,
  source = NULL,
  translation = NULL,
  label = NULL,
  trans_quotes = getOption("glossr.trans.quotes", "\"")
)
```

Arguments

gloss_lines	Lines for glossing, as a list
source	(Optional) Source of example
translation	(Optional) Free translation
label	(Optional) Example label
trans_quotes	(Optional) Quotes to surround the free translation with.

Details

This function is mostly for internal use, but may be useful for debugging or checking the output of specific calls. Normally, it's best to use `as_gloss()` or `gloss_df()`. Note that, unlike `as_gloss()`, `new_gloss_data` requires a list of gloss lines.

Value

Object of class `gloss_data`

---

gloss_df *Render gloss from a dataframe*

Description

Render gloss from a dataframe

Usage

```
gloss_df(
  df,
  output_format = getOption("glossr.output", "latex"),
  numbering = getOption("glossr.numbering", TRUE)
)
```

Arguments

df	Dataframe one row per gloss. Columns translation, source and label have special meaning (see as_gloss()); all the others will be interpreted as lines to align in the order given.
output_format	(Optional) Whether it will use latex, word or html format.
numbering	(Optional) Whether the gloss should be numbered (in Word and HTML).

Value

Object of class gloss with the original input as data attribute.

Examples

```
my_gloss <- data.frame(
  first_line = "my first line",
  second_line = "my second line",
  translation = "Translation of my example",
  label = "label"
)
gloss_df(my_gloss)
```

---

gloss_factory *Function factory to print glosses from dataframe*

Description

This function takes a dataframe with glosses and returns another function that takes either an id or list of ids (if use_conditionals is FALSE) or a conditional statement (if TRUE) and runs gloss_df() on the filtered dataframe.

Usage

```
gloss_factory(
  glosses,
  use_conditionals = FALSE,
  id_column = "label",
  ignore_columns = NULL,
  validate = TRUE
)
```

Arguments

glosses	Dataframe with gloss data.
use_conditionals	Boolean. If TRUE, the returned function will use conditional statements to filter the dataframe. Otherwise, it will use ids and match them to the values in the id_column.

id_column	Name of the column with ids for filtering, if use_conditionals is FALSE.
ignore_columns	Optional character vector with names of columns that could be used for filtering but should not be provided to gloss_df().
validate	Boolean. If TRUE, running gloss_factory() will print a few informative messages about how glossr is reading the dataframe.

Value

A function.

If use_conditionals is FALSE (the default), the returned function will take a character vector or a series of character vectors with id's to filter. If id_column is "label", running that function will be the equivalent to filtering glosses based on the values in the label column.

If use_conditionals is TRUE, the returned function will take the same conditions that a dplyr::filter() would.

Examples

```
my_glosses <- dplyr::select(glosses, -language)
by_label <- gloss_factory(my_glosses)

by_label("heartwarming-jp")

by_label("heartwarming-jp", "languid-jp")

by_cond <- gloss_factory(my_glosses, use_conditional = TRUE)
by_cond(stringr::str_ends(label, "jp"))
```

---

gloss_format_words *Apply latex formatting to many words*

Description

Facilitates applying the same latex formatting to different words in a row.

Usage

```
gloss_format_words(text, formatting)
```

Arguments

text	Character vector of length 1.
formatting	Latex formatting code, e.g. textit or textsc.

Value

Reformatted string

Examples

```
gloss_format_words("Many words to apply italics on.", "textit")
```

---

gloss_linetooltip *Apply tooltip to a full gloss*

Description

Apply tooltip to a full gloss

Usage

```
gloss_linetooltip(original, parsed)
```

Arguments

original	Text to show in the tooltip rendering.
parsed	Text to show as tooltip when hovering

Value

List of shiny.tag

Examples

```
ex_sp <- "Un ejemplo en español"
ex_gloss <- "DET.M.SG example in Spanish"
gloss_linetooltip(ex_sp, ex_gloss)
```

---

gloss_list *Sublist glosses*

Description

Takes a series of glosses from gloss_render() and puts them in a list within one example for PDF output.

Usage

```
gloss_list(glist, listlabel = NULL)
```

Arguments

glist	Concatenation of gloss objects, e.g. as output of gloss_df().
listlabel	Label for the full list (optional)

Value

Character vector including the frame for a list of glosses.

`gloss_render` *Render a gloss*

Description

These functions are output-specific and can be used to check the specific output of certain calls, but are not meant to be used in an R Markdown file. Instead, use `as_gloss()` or `gloss_df()`.

Usage

```
gloss_pdf(gloss)

gloss_html(gloss, numbering = TRUE)

gloss_tooltip(gloss, numbering = TRUE)

gloss_leipzig(gloss, numbering = TRUE)

gloss_word(gloss, numbering = TRUE)
```

Arguments

`gloss`	Object of class `gloss_data`
`numbering`	Whether the gloss should be numbered (in HTML and Word).

Value

Object of class `gloss`.

Functions

- `gloss_pdf()`: Render in PDF
- `gloss_html()`: Render in HTML
- `gloss_tooltip()`: Tooltip rendering for HTML
- `gloss_leipzig()`: Leipzig.js engine
- `gloss_word()`: Render in Word

Examples

```
ex_sp <- "Un ejemplo en español"
ex_gloss <- "DET.M.SG example in Spanish"
ex_trans <- "An example in Spanish"
my_gloss <- new_gloss_data(list(ex_sp, ex_gloss), translation = ex_trans, label="ex1")
gloss_pdf(my_gloss)

gloss_html(my_gloss)
```

---

knit_print.gloss *Print method for glosses*

Description

This method print `gloss` objects with `{knitr}`.

Usage

```
## S3 method for class 'gloss'
knit_print(x, ...)
```

Arguments

`x`	Object to print
`...`	Other options

---

new_gloss *gloss class*

Description

The `gloss` class contains how a gloss will be printed and its original input (Object of class `gloss_data`) as `data` attribute. It also has a `knitr::knit_print()` method for rendering in R Markdown (and Quarto).

Usage

```
new_gloss(input, output)
```

Arguments

`input`	A `gloss_data` object.
`output`	How the gloss must be printed, depending on the output.

Value

Object of class `gloss`.

`set_style_options` *Set general styling options*

Description

This is a helper function to set `options()` that control style characteristics for glosses across the full document. It is called within `use_glossr()` but can be overridden later but setting the appropriate options.

Usage

```
set_style_options(styling = list())
```

Arguments

`styling`	Named list of styling options for specific elements of glosses.

Details

There are two types of settings that can be provided in the list. First, `trans_quotes` sets the characters that must surround the free translation in a gloss. If no value is specified, it will be double quotes. There are no real restrictions for this value.

Second, the following elements can set general styling instructions for different sections of a gloss, formatting them completely in italics OR bold. The items with a `|` indicate that various names are possible.

- **source|preamble**: The line of the glosses where the `source` is rendered.
- **a|first**: The first line of the glosses, with the original language text.
- **b|second**: The second line of the glosses.
- **c|third**: The third line of the glosses if it exists.
- **ft|trans|translation**: The line of the glosses where the free `translation` is rendered. **numbering**: Whether the glosses should be numbered (in HTML and Word).

Each of these items can take one of a few values:

- `i`, `it`, `italics` and `textit` set italics.
- `b`, `bf`, `bold` and `textbf` set boldface.

Value

Set the appropriate options.

use_glossr *Use glossr*

Description

Call in a setup chunk.

Usage

```
use_glossr(html_format = NULL, styling = list())
```

Arguments

html_format	Whether the html output should use leipzig.js or tooltips.
styling	Named list of styling options for specific elements of glosses.

Value

Set options

Index