interlinear Package Documentation

Akpoué Kouamé Josué

August 24, 2024

1 Package Description

1.1 Purpose

The `interlinear` package provides a new interface for typesetting interlinear texts. It is based on `cgloss4e.sty` and is designed to avoid conflicts with other packages, such as `linguex`. This package allows for the simultaneous use of `linguex` and `interlinear` by renaming commands from `cgloss4e.sty` for better clarity and maintenance. The package is part of a larger set of tools to facilitate linguistic document preparation using LaTeX.

1.2 What's New

- **Toolbox Integration:** Support for SIL Linguists Field's Toolbox input (with some modifications). You can copy-paste the interlinear text from Toolbox directly into LaTeX.
- **Four-Line Option:** Adds support for a fourth line (Part of speech) in interlinear texts.
- **Markers:** New markers are provided for various parts of the example, including `\ct` (context), `\li` (language information), `\lt` (literal translation), and `\gj` (grammaticality judgments).
- **Environment:** Defines an `interlinear` environment with options for fine-grained control over element appearance.

2 Installation and Loading

To install, place `interlinear.sty` in a directory where LaTeX can find it. Load the package using:

```
\usepackage{interlinear}
```

Alternatively, use:

```
\input{path/to/interlinear.sty}
```

To ensure the proper functioning of the `interlinear` package, the following LaTeX packages are required:

- `marginnote`: This package is used for creating margin notes in your document.
- `xifthen`: Provides conditional commands for handling various logic in the package.
- `xkeyval`: Used to implement key=value options for customizing the behavior of the `interlinear` environment.
- `xparse`: Facilitates the definition of commands and environments with a more flexible interface.
- `etoolbox`: Supplies advanced tools for programming within LaTeX, such as patching and extending existing commands.
- `enumitem`: Allows customization of list environments (like `itemize`, `enumerate`, etc.), which may be used within `interlinear`.

Ensure that these packages are included in your LaTeX distribution.

3 Basic Usage

The main feature of `interlinear` is the `interlinear` environment. This environment allows you to structure interlinear texts using markers. Below is a comparison between `cgloss4e` and `interlinear` syntax.

3.1 Example 1: Two-Line Interlinear

cgloss4e code:

```
\gll lorem ipsum dolor\\
     sit amet consegur\\
\glt ‘Lorem ipsum dolor’
```

interlinear code:

```
\begin{interlinear}
\tx lorem ipsum dolor\\
\gl sit amet consegur\\
\ft ‘Lorem ipsum dolor’\\
\end{interlinear}
```

3.2 Example 2: Three-Line Interlinear

cgloss4e code:

```
\glll lorem ipsum\\
      dolor sit\\
      amet consegur\\
\glt ‘Lorem ipsum dolor’
```

interlinear code:

```
\begin{interlinear}
\tx lorem ipsum\\
\mb dolor sit\\
\gl amet consegur\\
\ft ‘Lorem ipsum dolor’\\
\end{interlinear}
```

3.3 Complete Example

```
\begin{interlinear}
\li Language\\
\rf Identifier of the data in the corpus/corpora\\
\ct Context\\
\gj Grammaticality judgment\\
\tx Object language text\\
\mb Morpheme breaks\\
\gl Glosses\\
\ps Part-of-speech\\
\ft Free translation\\
\lt Literal translation\\
\nt Extra note\\
\end{interlinear}
```

4 Setting Options

The `interlinear` environment can be customized using the options described below.

4.1 Interlinear Lines Number

The number of interlinear lines is controlled by the `linesnumber` option. Possible values are 0, 2, 3, 4 (default is 2).

4.2 Markers and Customization

You can define custom markers for each part of the interlinear text:

```
\li languageinfomarker
\rf referencemarker
\ct contextmarker
\gj gramjudgmarker
\tx textmarker
\mb morphbreakmarker
\gl glossmarker
\ps partofspeechmarker
\ft freetranslationmarker
\lt literaltranslationmarker
\nt extranotemarker
```

4.3 Appearance Customization

You can modify the appearance of different elements using the following options:

```
lineoneformat    % First line format
linetwoformat    % Second line format
linethreeformat  % Third line format
linefourformat   % Fourth line format
exformat         % Example text format (no-interlinear mode)
extranoteformat  % Extra note format
contextnameformat % Context label format
```

These options can control the font family, shape, series, size, and color. To specify multiple formatting options, enclose them in a group. For example, the command `lineoneformat={\itshape\bfseries\small}` will set the text on the first interlinear or gloss line to boldface italics in a small size. By default, all formats are set to `\normalfont`.

4.4 Other Settings

Various other elements can be customized:

- The label indicating the context of the example can be changed using the `contextname` option. For instance, to change `Context:` to `Situation:`, use `contextname=Situation:`.
- The space between the context and the body of the example can be specified via the `contextvoffset` option.
- By default, marginal notes are enclosed in parentheses `()`. To change this behavior, use the `extranoteleftpunct` and `extranoterightpunct` options. For example, to change the parentheses to square brackets `[]`, set `extranoteleftpunct=[` and `extranoterightpunct=]`. If you prefer no punctuation marks around your notes, set `extranoteleftpunct=` and `extranoterightpunct=`.

- The placement of language information for the linguistic example is controlled by the `languageinfopos` option, which accepts two values: `head` and `margin`. The default value, `margin`, places this information in parentheses in the margin of the first line of the example. The `head` value places it in a separate line at the very beginning of the example, even before the context. When the no-interlinear mode is enabled, `languageinfopos` should be set to `head` to avoid positioning conflicts with marginal notes.

4.5 Applying Options Outside the Environment

Options can also be specified outside the `interlinear` environment using the `\interlinearstyle` command, which takes one mandatory argument. This argument should contain the list of chosen options, and `\interlinearstyle` must be called before the environment.

For example:

```
\begin{interlinear}[linesnumber=3,linefourformat=\itshape]
(content)
\end{interlinear}
```

is equivalent to:

```
\interlinearstyle{linesnumber=3,linefourformat=\itshape}
\begin{interlinear}
(content)
\end{interlinear}
```

The `\interlinearstyle` command redefines the default values for the specified options. Therefore, each time this command is used, it updates the default values for the specified options.

4.6 Using Predefined and Custom Styles

One of the features of the `interlinear` package is the ability to define multiple styles and use them throughout the document without having to call `\interlinearstyle` each time. There are four predefined styles:

- `default`: Standard settings.
- `gsr`: Sets the first interlinear line in italics.
- `nointerlinear`: Configures the no-interlinear mode, changes the `\ft` marker to `\ot`, and sets `languageinfopos` to `head`. This style provides a safe way to configure the no-interlinear mode without worrying about the various parameters.
- `gsrnointerlinear`: Similar to `nointerlinear`, but the text of the example is italicized.

There are two ways to apply a style:

1. Using the `\UseInterlinearStyle{}` command before the `interlinear` environment, at the beginning of the document, or in the preamble.
2. Using the dedicated `style` option in the optional argument of the `interlinear` environment.

Example: `\begin{interlinear}[style=gsr]`.

It is recommended to specify the `style` option first when combining it with other options.

You can also specify additional options alongside the style to customize it as needed (e.g., if you want to temporarily modify a style parameter).

For example, the `gsr` style was declared using the following code:

```
\DeclareInterlinearStyle{gsr}{%
  linesnumber=2,%
  lineoneformat=\itshape,%
  linetwoformat=\normalfont%
}
```