

# textglos package documentation\*

Natalie Weber  
natalie.a.weber@gmail.com

2012/06/27

## Abstract

The `textglos` package provides a set of macros for in-line linguistic examples (as opposed to interlinear glossing, set apart from the main text). It prevents hyphenated examples from breaking across lines and consistently formats phonemic examples, orthographic examples, and more.

## 1 Introduction

`\gl` Linguists often need to use examples in the line of text like Language *example* ‘gloss’  
`\xo` or proto-Language \*reconstruction ‘reconstructed semantics’. The following macros are  
`\xt` provided:

<code>\xm</code>		
<code>\xv</code>	(1) <b>This code</b>	<b>Prints</b>
<code>\xh</code>	<code>\gl{gloss}</code>	‘gloss’
	<code>\xo{orthography}</code>	<orthography>
	<code>\xt{phonetic}</code>	[phonetic]
	<code>\xm{phonemic}</code>	/phonemic/
	<code>\xv{vernacular}</code>	<i>vernacular</i>
	<code>\xh{reconstruction}</code>	*reconstruction

These macros are much more useful than simply typing each example by hand in a number of ways. First, you do not need to type the various opening and closing brackets or apostrophes (an easy place to introduce typos). Second, hyphens (for affixes) and equals signs (for clitics) never cause linebreaks and are always typeset in an upright font. Third, the style of each macro is fully customizable using a template in the preamble of your document, and users can redefine the default macros, or define new macros of their own.

---

\*This document corresponds to `textglos` v1.0, dated 2012/06/27.

## 2 Basic usage

All of the macros also have two optional arguments. When just one optional argument is specified, it is printed as a gloss following the mandatory argument. The gloss uses `\gl` so it will be printed with single quote marks automatically<sup>1</sup>. When two optional arguments are specified, the first one will print as a prefix plus a non-breaking space<sup>2</sup> in front of the mandatory argument, and the second one will print as a gloss following the mandatory argument. Therefore, if you only want to print a prefix, you need to use empty brackets in place of the gloss.

(2)	<b>This code</b>	<b>Prints</b>
	<code>\xv [Lang] [gloss] {word}</code>	Lang <i>word</i> ‘gloss’
	<code>\xh [Lang] [] {word}</code>	Lang *word
	<code>\xo [gloss] {word}</code>	<word> ‘gloss’

## 3 Typesetting features

The macros replace all hyphens with a non-breaking hyphen, so that morpheme breaks are not interpreted by  $\TeX$  as a place to break a line. You can suggest a hyphenation point by using `\-` in your example, as normal. Note that some of the macros prevent the normal line-breaking mechanism in  $\LaTeX$  while others do not; see section (4) for details.

The macros also replace all equals signs with a shorter version of the equals sign (for clitics), as long as the package `graphicx` exists on your system (otherwise a normal equals sign is used). The default math symbol is much longer than the hyphen, so that clitics look out of place next to affixes unless you use this shorter version.

Finally, hyphens (for morpheme breaks) and equals signs (for clitics) will always be typed in upright text, even if the surrounding text is italicized, as it is for `\xv{}`<sup>3</sup>. The macros use  $\LaTeX$ 's `\textup{}` command, which also take care of kerning when moving from italicized text to an upright punctuation mark and back. Notice that in the example in (3) typeset with `\textit`, the vertical edges of both the hyphen and the equals sign are slanted. (You may have to zoom in to see this.) The top and bottom bar of the equals sign are not aligned, and they are closer together than the upright version of the equals sign. The `\xv` versions are upright and squared off, and the clitic marker is shorter:

---

<sup>1</sup>By default; see section (4) for how to alter this behavior.

<sup>2</sup>By default; see section (4) for how to alter this behavior.

<sup>3</sup>This will happen no matter what the surrounding font shape is. This may be undesirable behavior for, e.g. smallcaps text.



```
(7) \lingexample{xh}{*\bgroup\itshape X\egroup}
    or
    \lingexample{xh}{*\textit\bgroup X\egroup}
```

You may want phonetic or phonemic examples to be typed in IPA. You can simply redefine `\xt` and `\xm` to use `\tipaencoding` (from the `tipa` package) by putting the following into your document preamble:

```
(8) \lingexample{xt}{\tipaencoding[X]}
    \lingexample{xm}{\tipaencoding/X/}
```

Of course you must have `\usepackage{tipa}` in your preamble to do this. This syntax means that both your example and the surrounding square brackets are typeset using `tipa`, which allows `tipa` to correctly handle the kerning between the brackets and the IPA symbols.

By default, the macros do *not* prevent the normal line-breaking mechanism in L<sup>A</sup>T<sub>E</sub>X. This means that multiword examples like *It was Colonel Mustard with the knife in the dining room*<sup>4</sup> will break between words as normal. Unfortunately, if you work with a highly agglutinative language with long words (or long morphemes), then L<sup>A</sup>T<sub>E</sub>X may try to hyphenate the words and break them across multiple lines. (It also may be undesirable to have transcriptions in `\xt{}` and `\xm{}` break across lines.) One solution would be to create a new `\lingexample` macro to be used with single-word examples which sets the example in an `\hbox`. This example will not break across lines:

```
(9) \lingexample{xvword}{\hbox\bgroup\itshape X\egroup}
```

## 5 The Code

The package `graphicx` is required to create `\shorreq`.

```
1 \IfFileExists{graphicx}{\RequirePackage{graphicx}}{}
```

This code originally Sven Siegmund's, created with help of XeTeX mailing list, to detect optional argument. See <http://xelatex.blogspot.com/2008/03/newcommand-with-optional-argument.html>. I use it constantly.

```
2 \global\long\def\tlist@if@empty@nTF #1{%
3 \expandafter\ifx\expandafter\\\detokenize{#1}\%
4 \expandafter\@firstoftwo%
5 \else%
6 \expandafter\@secondoftwo%
7 \fi%
8 }%
```

---

<sup>4</sup>Code: `\xv{It was Colonel Mustard with the knife in the dining room}`

`\nbrpunct` Within a linguistic example we do not want TeX to break the line at explicit hyphens (as these are generally morpheme breaks). We cannot use penalties, as they are only set for the paragraph as a whole, and the linguistic environment will be closed before the end of the paragraph. This solution uses eTeX `scantokens` to make hyphens and equal signs active *only within the linguistic environment* and `let` them to a non-breaking upright counterpart. It is courtesy Alan Munn on Ling-TeX and some `comp.text.tex` discussion.

`\shorteq` First we use the `graphicx` package to define an equals sign (for clitics) that is shorter than the default math sign. I find that 1.5 times the length of a hyphen (for affixes) looks good.

```

9 \@ifpackageloaded{graphicx}{%
10  \newcommand{\shorteq}{%
11    \settowidth{\@tempdima}{-}% Width of hyphen
12    \resizebox{1.5\@tempdima}{\height}{=}%
13  }%
14 }{}%
```

`\nbrhyph, \nbreq` Then we define non-breaking punctuation.

```

15 \def\nbrhyph{\textup{\hbox{-}}}%
16 \ifx\shorteq\relax
17   \def\nbreq{\textup{\hbox{=}}}%
18 \else
19   \def\nbreq{\textup{\hbox{\shorteq}}}%
20 \fi
```

We define the macro `nbrpunct` which takes the linguistic example as its argument. This macro opens a group, makes punctuation marks active, and uses `scantokens` to `let` them to their nonbreaking counterparts, defined above. Then it closes the group.

```

21 \def\nbrpunct#1{%
22   \bgroup
23   \catcode'\-=\active
24   \catcode'\==\active
25 %   \hyphenchar\font=-1\relax % uncomment to disable automatic hyphenation
26   \scantokens{\let-\nbrhyph
27     \let=\nbreq#1\ignorespaces}
28   \egroup
29 }%
```

`\gl` The macro `\gl` is simply for glosses. It can be used on its own or in combination with an example, where it is hard-coded in.

```

30 \providecommand*\gl[1]{‘#1’}%
```

`\lingexample` The macro `\lingexample` takes two arguments (a user-defined name and template). First it calls `\@lingsub`, which defines a macro based on the user-defined name called

`\@@lingsub@<name>`. This macro takes one argument and sandwiches it between the beginning part of the template and the last part of the template. This trick is from John Frampton's `expex` package.

Second, it creates a new macro `\<name>` that the author can use in the text either alone or with two optional arguments. We pass the `<name>` along as the first parameter so that we can use it later.

```

31 \def\lingexample#1#2{% #1 = name, #2 = template
32   \@@lingsub(#1)#2\@nil % \@nil as delimiter
33   \expandafter\gdef\csname #1\endcsname{% \def\name
34     \@ifnextchar[{\@lingexample(#1)}{\@lingexample(#1) []}%
35   }%
36 }%
37
38 \def\@lingsub(#1)#2X#3\@nil{%
39   \expandafter\gdef\csname @@lingsub@#1\endcsname##1{\nbrpunct{#2##1#3}}%
40 }%

```

We use some trickery from the `natbib` package to switch around the optional arguments in the case that there is only one.

```

41 \def\@lingexample(#1) [#2]{\@ifnextchar [%
42   {\@@lingexample(#1) [#2]}{\@lingexample(#1) [] [#2]}%
43 }%

```

Finally, we check for math mode and use `\tlist@if@empty@nTF` to print the optional prefix *only* in the case that it is non-empty, then the mandatory argument, using `\@@lingsub@<name>` defined above to follow the desired template, and final the optional gloss *only* in the case that it is non-empty. The macros `\lingprestyle` and `\lingpoststyle` are defined below.

```

44 \def\@@lingexample(#1) [#2] [#3] #4{%
45   \ifmmode
46     $\tlist@if@empty@nTF{#2}{-}{#2~}%
47     {\csname @@lingsub@#1\endcsname{#4}}%
48     \tlist@if@empty@nTF{#3}{-}{ \gl{#3}}$%
49   \else
50     \tlist@if@empty@nTF{#2}{-}{\lingprestyle{#2}}%
51     {\csname @@lingsub@#1\endcsname{#4}}%
52     \tlist@if@empty@nTF{#3}{-}{\lingpoststyle{#3}}%
53   \fi
54 }%

```

`\lingprestyle`    The optional prefix and optional postfix are printed with `\lingprestyle` and `\lingpoststyle`  
`\lingpoststyle`    respectively.

```

55 \providecommand{\lingprestyle}[1]{#1~}
56 \providecommand{\lingpoststyle}[1]{ \gl{#1}}

```

```

\xo,\xt,\xm The textglos package pre-defines some in-text examples that will be useful to linguists.
\ xv,\xh 57 \lingexample{xo}{\ensuremath{<}X\ensuremath{>}}
58 \lingexample{xt}{[X]} % use \lingexample{xt}{\tipaencoding[X]}
59 % if you want IPA
60 \lingexample{xm}{/X/}
61 \lingexample{xv}{\itshape X}
62 \lingexample{xh}{*X} % use \lingexample{xh}{*\textit\bgroup X\egroup}
63 % if you want italicized

```