The gloss Package*

Jose Luis Díiaz
Javier Bezos†

July 26, 2002

Gloss is a package which allows the creation of glossaries using BibTeX. With this approach, the user writes a database of terms and definitions using a file format much like the bibliographic databases. Then he inserts in the LaTeX document a command `\gloss{`⟨*key*⟩`}` for selecting which entries he wants to appear in the glossary. These keys are written in an auxiliar file when LaTeX is run on the document. Then, running BibTeX these entries are extracted from the database and collected in alphabetical order in a file. The next run of LaTeX read this file and inserts it in the appropiate point of the document, typesetting the desired glossary or glossaries.

The process is much like the mechanism for including the bibliographic references. This approach has several advantages:

1. BibTeX is available in every platform where TeX is.
2. Glossary entries can be stored in databases, and you needn't rewrite the definitions.
3. There are a lot of tools for managing BibTeX databases.
4. BibTeX can sort the entries and group them; furthermore, BibTeX-8 can sort correctly the entries in non-English languages. (In fact, this is the main raison which the package was first developped for.)
5. BibTeX is a programable tool and you can define your own styles for acronyms, symbols, and so on, or changing its default behaviour. Gloss styles are pretty simples and creating new ones is not difficult.

Of course, there also disavantages—for example, you cannot select sorting entries word by word or letter by letter, at least at present.

The gloss bundle includes the gloss.sty package, a glsplain.bst style, a glsbase.bib database, and a `sample.tex` file with its `sample.bib` database (and, of course, the `readme` file and this manual).

*The gloss package is currently at version 2.19.

†For bug reports, comments and suggestions go to http://www.texytipografia.com/contact.php. New `.bst` styles are also welcome. English is not my strong point, so contact me when you find mistakes in the manual. Other packages by Javier Bezos: accents, tensind, esindex, titlesec, titletoc, dotlessi.

1 The data files

If you know BibTeX, you will find this section easy to understand. A data file contains a set of records defining terms, and its name must have the extension `.bib`. Every entry has the following format:

```
@gd{gnu,
  word = {gnu},
  definition = {Extrange animal}
}
```

Here

- `gnu` is a key identifying the record;
- `word` is the word which will be used as headword and in the text; and
- `definition` is the definition printed in the glossary.

Here is the description of the available fields, and when they are used

word Required. It should be given always as it would be written at the middle of a sentence. The basic style glsplain converts its first letter to uppercase for use after a period. In the generated glossary, entries with the same initial are grouped, preceded by a heading with that letter.

definition Required. The definition can be as long an you want, but you should note that implicit paragraphs (those with a blank line) are ignored and a `\par` would be necessary. The final period should be omitted because it is supplied later (and sometimes it will be replaced by a comma).

short Optional. A short form. It could be a symbol, an acronym, etc. depending on the nature of the glossary.

sort-word Optional. If present, this field will be used to sort the entries. It is useful in greek symbols and signs, for example.

group Optional. This field should consist in an uppercase letter and is intended for entries not beginning (or not containing) letters. Entries with the same `group` are gathered in the glossary under a single heading and sorted in the whole glossary using this letter, with entries without `group` placed as if they had a group key of `L`. Using both `sort-word` and this field allows grouping, say, greek symbols, numbers, signs and the like under seperate headings. More on that later.

heading Optional. It forces an entry to be listed under the given heading. Useful in non-English languages when letters with diacriticals are used (even with BibTeX-8):

```
@gd{gnu,
  word = "{\'A}nimo",
  definition = "...",
  heading = "A"}
```

This field is uncompatible with `group`.

The `@gd` entry type is the only available currently in `glsplain`. `@glossdef` is a synonymous.

The default `crossref` field is available, but in glossaries is mostly unpractical except in a few cases (for example, in a list of symbols with the same greek letter with many meanings).

2 Basic commands

`\makegloss`

This command in the preamble tells gloss to create a glossary.

`\gloss{`⟨*key*⟩`}`

Similar to `\cite`. It writes a "citation" to the auxiliary file `.gls.aux`. Sadly, this double extension is necessary because BibTeX requires the input file to be named with the `.aux` extension. Below is explained how MS-DOS users can work around that. Note that this file is not reread by the document, it just provides information to BibTeX of terms cited.

`\printgloss{`⟨*databases*⟩`}`

Similar to `\bibliography`. It prints the glossary stored into the `.gls.bbl` file generated by BibTeX from `.gls.aux`.

In this basic interface the glsplain style is used always.

3 The generated glossary

The steps to generate and use the glossary are:

- LaTeX the document (let's call it `file.tex`),
- BibTeX the `.gls.aux` file (i.e., `bibtex file.gls`),
- LaTeX the document again, and
- LaTeX again if there are unresolved cross references.

Once LaTeXed the document, and BibTeXed the `.gls.aux` file, you will get the glossary in the `.gls.bbl` with the following TeX format:

- the whole glossary is enclosed in the `thegloss` environment, which just prints the sectioning heading whith the `\glossname` title;
- a series of `\glossheading`s (or `\glossgroup`s) commands and `glossitem`s environments. Note that gloss items are not commands but environments. The `glossitem*` environment means that the definition of the entry ends with a period.

4 The whole thing

Most of the formatting is done by the package and not by BibTeX. The glsplain style provides two forms, sometimes three, of terms: the basic form as given in `word` to be used in the text, and probably as headword, the second one is `word` with the first letter uppercased to be used at the beginning of a sentence; and only if `short` was included, a short form.

Not surprisingly, the syntax of the gloss commands is very alike to that of bibliographies with some touches from that of indexes.

4.1 Package options

`refpages`

The number of the first page where the term is referred to, is appended to the gloss entry.

4.2 Multiple glossaries

`\makegloss`
`\newgloss{`⟨*name*⟩`}{`⟨*suffix*⟩`}{`⟨*title*⟩`}{`⟨*style*⟩`}`

For defining a new glossary use `\newgloss`. The `\makegloss` command is just a synonymous with

```
\newgloss{default}{.gls}{\glossname}{glsplain}
```

Note that the suffix does include the dot.

MS-DOS users must use `\newgloss` instead of `\makegloss`, and a document name with at most seven letters. For instance:

```
\newgloss{default}{G}{\glossname}{glsplain}
```

Note that, in this case, the suffix does not include the dot (this way we avoid double extensions).

`\printgloss[`⟨*name*⟩`]{`⟨*databases*⟩`}`

Prints the ⟨*name*⟩ glossary. By default, the `default` one is printed.

4.3 The `\gloss` command

`\gloss[`⟨*options*⟩`]{`⟨*key*⟩`}`

Possible options are:

- `nocite` makes the command behave in the same fashion as `\nocite`. For example, with `\gloss[nocite]{*}` all entries of the databases are included in the glossary.[1]
- `refpage` tells gloss to ignore previous references to pages. Sometimes you say things like "...at the end of the chapter, we will introduce the concept of..."; when the concept is actually introduced you should use this option.
- ⟨*name*⟩ of the glossary file where the key is written, as defined by `\newgloss`. The key will be written into that glossary. If there is no ⟨*name*⟩ it defaults to `default`.

The following options control the format of the term in the text.

- `word` prints the term exactly as given in the `word` field.

[1] The command `\onlygloss` is a deprecated synonymous with `\gloss[nocite]`.

- `Word` prints the term as given in the `word` field, but with the first letter uppercased.[2]

  ```
  ...discovered. \gloss[Word]{spectroscopy} became one of the most...
  ```

- `short` prints the short form, provided the bib file defines it (if not a warning is reported).
- `Long` prints a combination of `Word` and `short`: "Word (short)".
- `long` prints a combination of `word` and `short`: "word (short)". For example, the very first time an acronym is used:

  ```
  ...and the proposals made by the \gloss[long]{iupac} provide...
  ```

  Of course, you may want the following references to be in the short form; just define

  ```
  \newcommand{\acronym}[2][]{\gloss[short,#1]{#2}}
  ```

The former (`nocite` and `refpage`) are built in, ⟨*name*⟩s are created by `\newgloss`, and the latter are created by the package with `\setglosstext`; if no option defined by `\setglosstext` is included, it defaults to `word`.

`\setglosstext{`⟨*option-name*⟩`}{`⟨*format*⟩`}` (3 parameters)
`\ifglossshort{`⟨*format*⟩`}` `\ifglossshort*{`⟨*format*⟩`}`

`\setglosstext` sets how entries are printed in the main text by `\gloss`, where ⟨*option-name*⟩ is the name to be used in the optional argument of `\gloss`. There are five predefined formats, described above, which you may redefine or complement with new defined ones. In ⟨*format*⟩ there are three available arguments, which are defined implicitly: `#1` is `word`, `#2` is `word` with its initial uppercased, and `#3` is the `short` field. Thus, the package does the following:

```
\setglosstext{word}{#1}
\setglosstext{Word}{#2}
\setglosstext{short}{\ifglossshort*{#3}{}}
\setglosstext{long}{#1\ifglossshort*{ (#3)}{}}
\setglosstext{Long}{#2\ifglossshort*{ (#3)}{}}
```

Use is made of `\ifglossshort` which takes the first argument if the short form exists, and the second one if does not exist. In the latter case, that is done silently in the unstarred version, but with an error in the starred one.

4.4 Glossary layout

`\glossheading{`⟨*format*⟩`}`

Sets how the headings are formatted; it is redefined with `\renewcommand`. For example:

```
\renewcommand\glossheading[1]{%
  \stopglosslist
  \subsection*{#1}}
```

[2] The command `\Gloss` is a deprecated synonymous with `\gloss[Word]`.

`\setglossgroup{`⟨*group*⟩`}{`⟨*heading*⟩`}`

Sets the heading corresponding to entries grouped by glsplain under the same `group` key (and preceded by `\glossgroup`, which in turn calls `\glossheading`).

```
\setglossgroup{C}{Signs}
```

`\setglosslabel{`⟨*format*⟩`}` (3 parameters)

Sets which of the three forms are printed as label in the gloss items and some other optional formatting.

The package does:

```
\setglosslabel{\sffamily\bfseries#1\ifglossshort{ (#3)}{}}
```

`thegloss`

By default, the main environment just prints the gloss title. You may change its definition.

`glosslist`
`\stopglosslist`

You usually won't see the `glosslist` environnment. It is automatically started by `glossitem` if necessary. You may stop it with the `\stopglosslist`. That's so done to interact with the format of the heading for each letter group. The above example of `\glossheading` uses it because sectioning commands cannot be used inside lists; this way, the list is stopped, the title is printed, and the following `glossitem` restarts the list. If you say:

```
\renewcommand{\glossheading}[1]{}
```

the whole glossary is printed in a single list (with no unwanted space between letter groups).

Gloss provides its own format for `glosslist` (simply because the authors like it) with a `\glosshang` length to adjust the left margin, but you may change its definition.

`glossitem`
`glossitem*`

Its `\begin` consists of an `\item` and some additional stuff. Its `\end` adds a period (except in the starred version) or the page number. You should not modify this environment, except if you want a format not based in a list environment.

Here is an example of how to modify the layout of the glossary:

```
\setglosslabel{#2}

\renewcommand{\glossheading}[1]{%
  \stopglosslist  % -- Don't forget that!
  \vspace{1pc}%
  {\large\centering\bfseries#1\par}}

\renewenvironment{glosslist}
  {\begin{description}}
  {\end{description}}
```

`\glosspage`
`\xglosspage`

This command is used to print the page at the end of the gloss entry with `refpages`. If the entry ends with a period, `\xglosspage` is used, which by default just maps to `\glosspage`. You may redefine them with `\renewcommand`:

```
\renewcommand{\glosspage}[1]{. (See page~#1)}
\renewcommand{\xglosspage}[1]{ (See page~#1)}
```

5 Order of items

Now we explain how BibTeX sorts and groups entries. Firstly, the necessary values are assigned, if necessary. The `group` field is used, as stated above, for entries consisting of non alphabetical terms, and differents steps are followed depending on whether this field exists or not.

If `group` is not present, then

- `sort-word`, if omitted, is `word` lowercased with non alphabetical signs removed.
- `heading`, if omitted, is the first letter in `word`.

If `group` is present, then

- `sort-word`, if omitted, is `word` lowercased with non alphabetical signs *not* removed.
- `heading` is not used.

Now, entries can be sorted. First, they are ordered by `group`, and then, inside each group, by `sort-word`. In fields with `group` and no `sort-word` the ASCII codes are used. No further sortening is done. Finally, entries are grouped: first, consecutive entries with the same `group` field; then, consecutive entries whose `group` is "L" and with the same `heading` field. Note that `group` sorts and groups, `sort-word` just sorts, and `heading` just groups.

Now, let's answer the following simple question: When should I use `heading` and `sort-word`? If the word begins with a letter with diacritical mark alphabetized under the letter without diacritical mark (a fairly frequent case), use heading, does not matter you are using BibTeX or BibTeX8:

```
@gd{ecole,
  word = "{\'e}cole",
  definition = "...",
  heading = "E"
}
```

(`école` and `'ecole` are allowed, too).

If the word begins with a letter which is placed under a heading of its own, use `sort-word` in BibTeX and nothing in BibTeX8 (provided a correct sorting file is provided, which is not the case for many languages):

```
@gd{nname,
  word = "{\~n}ame",
  definition = "...",
  sort-word = "nzzame"
}
```

in 7-bits versions. In 8-bits version, you may set `word` as `ñame` and suppress the `sort-word` field.

Anyway, if you are using a 7-bits version you may want using both fields:

```
@gd{innigo,
  word = "{\'I}\~nigo",
  definition = "...",
  heading = "I",
  sort-word = "Inzzigo"
}
```

Finaly, gloss provides inside the `thegloss` environment the `\+zz+` command expanding to nothing, where `zz` is any text helping in sorting entries (usualy `zz`); this way, `sort-field` is not necessary in most of cases:

```
@gd{nname,
  word = "{\~n\+zz+}ame",
  definition = "..."
}
```

An example in Swedish is `\+zzx+\r{a}`, in Czech `{\v{c}\+zz+}`, and in Breton (8-bits) `\+n+\~n`. (Of course, BibTeX could translate from a readable form to one for alphabetizing. That should be done in a future.) Using either `sort-word` or `\+zz+` is a question of personal taste; One of us [JB] uses `sort-word` while the other [JLDA] prefers BibTeX8. This syntax is not compliant with the LaTeX interface guidelines (use braces or brackets always) but it's short, which was the main goal; this feature is mostly unsuportted, however.

6 Complements

6.1 The glsbase database

This database defines some useful strings which can be used in other databases. Currently, it only includes a set of strings named `alphasort`, `betasort`, etc. to be used in the `sort-word` field to provide the right order of greek symbols. (They are defined as `"01"` , `"02"`, etc.)

6.2 The glsshort style

This style is provided for acronym lists. It sorts and creates headings using the `short` filed. A new `sort-short` can be used to fine tune the order of entries (this field will be ignored in the glsplain style). However, note that the printed form still follows the conventions given above, and you should use the `short` specifier in `\cite`, and `\setglosslabel`.

6.3 Compatibility with hyperref

If the hyperref package is loaded, links from the word to the entry will be created. You can control the appearance of gloss links with the following macros: `\glosslinkborder`, `\glosslinkcolor`, `\glosslinkbordercolor` (which correspond to `pdfborder`, `linkcolor` and `linkbordercolor`); you can change them with `\renewcommand`.

6.4 The sample file

Once installed gloss you should be able to typeset the `sample.tex` file, which will enlighten the usage of this package — LaTeX to write the auxiliary file, BibTeX to create the glossary, LaTeX to define the labels, and LaTeX to see the final result. Disclaimer: its text is in Spanish.

6.5 Backward uncompatibility

Version 0.1 beta had a different syntax for the `\gloss` command. The old syntax `\gloss[`⟨*word*⟩`]{`⟨*key*⟩`}` does not work and ⟨*word*⟩`\gloss[nocite]{`⟨*key*⟩`}` should be used instead. The `\glossstyle` command has been removed and its functionality merged into `\newgloss`.

6.6 Language support

The following package options provide translation of the glossary heading and page abbreviation: `basque`, `catalan`, `danish`, `dutch`, `french`, `german`, `italian`, `portuguese` (and `brazilian`), `russian` (any encoding), `polish`, `spanish` and `swedish`. Translations to other languages are welcome.