The covington Package

Macros for Linguistics

Michael A. Covington Jürgen Spitzmüller*

Version 2.17, November 12, 2026

Abstract

This package, initially a collection of Michael A. Covington's private macros, provides numerous customizable LaTeX macros that are helpful for linguists, including macros that allow for multiple diacritics on the same letter, numbered linguistic examples, interlinear glosses (word-by-word translations), Discourse Representation Structures, phrase structure rules and disjunctions, linguistic feature structures, and for semantic markup commonly used in linguistic notation.

Contents

1 Introduction 2
2 Stacked diacritics 3
3 Numbered examples 4
3.1 Example numbers . . . 4
3.2 The **example** environment . . . 5
3.3 The **examples** environment . . . 7
3.4 The **subexamples** environment . . . 10
3.5 Examples of examples . . . 13
3.6 Referring to examples . . . 16
3.7 Numbering per chapter or continuously . . . 17
4 Glossing sentences word-by-word 17
4.1 Gloss macros . . . 17
4.2 Glossing with low-level commands . . . 21
4.3 Example glosses . . . 23
5 Phrase structure rules 26
6 Feature structures 26
7 Discourse Representation Structures 27
8 Exercises 29
9 Reference Lists 30
10 Semantic markup 31
11 Big curly brackets (disjunctions) 32
12 Release history 32

*Current maintainer. Please report issues via https://codeberg.org/jspitz/covington

1 Introduction

This is the documentation for version 2.17 of covington (November 12, 2026), which is a LaTeX package providing macros for typing some special notations common in linguistics.[1]

To use covington, load the package as usual by adding the command

```
\usepackage[<options>]{covington}
```

to your document preamble. The package has the following options to customize its behavior:

force=⟨**true|false**⟩ Default: *false*. Enforce the redefinition of environments that have already been defined by other packages or the class.

This applies to the **example**, **examples**, **subexamples** and **exercise** environments, which are by default not touched if they have already been defined before covington is loaded. See sec. 3.2, 3.3, 3.4 and 8 for details.

keeplayout=⟨**true|false**⟩ Default: *false*. Do not tweak the layout.

Covington sets `\raggedbottom` and redefines the value of the `\textfloatsep` length. This just follows the preferences of the original package author and is not necessary for the package's functionality. Yet for backwards compatibility reasons, we cannot change this. Thus, we provide the option described here to opt out this presetting. And we actually encourage you to use it.

noglossbreaks=⟨**true|false**⟩ Default: *false*. If this option is set to **true**, covington will try hard to prevent page breaks within glosses.

If this option is not set, page breaks can occur between interlinearized text and free translation of a gloss, as well as between gloss preamble and interlinearized text, which is usually not what you will want. Nonetheless the option is not set by default. This is for backwards compatibility reasons (in order to not change page breaking of existing documents). Note that page breaks might still occur in some cases even if the option is set. In order to prevent them definitely, you can put the gloss in a parbox or minipage.

owncounter=⟨**true|false**⟩ Default: *false*. Use an own counter for numbered examples.

By default, covington uses LaTeX's equation counter for example numbering, so that if you use equations and numbered examples in the same paper, you get a single continuous series of numbers. While some people (including the original author of this package) consider this a feature, others might prefer to number equations and linguistic examples separately. If you count to the latter sort, use this option. Note that the default appearance of the own counter and the equation counter differs if the document has chapters. See sec. 3.7 for details.

[1] The package has a long history. It started off as a collection of private macros back in the LaTeX 2.09 days and was initially released as `covingtn.sty` (following the old 8.3 FAT file name limit). In emTeX under MS-DOS, the file was distributed as `covingto.sty`. Eventually, it has been renamed to covington and adapted to LaTeX 2ε. The current version requires a reasonably recent version of LaTeX 2ε, as it employs some newer kernel features.

fnexamplecounter=⟨main|own|own-reset⟩ Default: *main*. Specifics of the counter for numbered examples in footnotes.

By default, covington numbers examples in footnotes in sequence with the numbering used in the main text (this conforms to the default value of this option, **fnexamplecounter=main**). If you set **fnexamplecounter=own**, examples in footnotes get a different numbering which is incremented throughout the whole footnote apparatus.[2] If you set **fnexamplecounter=own-reset**, examples in footnotes get an own counter as well, but this variant resets the counter at each footnote (similar to linguex and gb4e).[3] Both own counters use lowercase Roman numbering by default (see sec. 3.2 on how to change this)

Please note the following package-related caveats:

- If you are using covington and the uga (University of Georgia thesis style) package together, you should load uga before covington.
- If you are using covington with beamer-article, you should load beamer-article before covington.
- If you are using covington with the drs package, you should load drs before covington. See sec. 7.

In what follows we presume that you know how to use LaTeX and have access to LaTeX manuals.

2 Stacked diacritics

LaTeX provides a generous range of diacritics that can be placed on or below any letter, such as:

x̀ x́ x̂ ẍ x̃ x̄ x̋ x̌x x̧ x̣ x̱

which are typed, respectively, as:

```
\‘{x} \’{x} \^{x} \"{x} \~{x} \={x} \H{x} \t{xx} \c{x} \d{x} \b{x}
```

Out of the box, however, LaTeX doesn't give you a convenient way to put *two* diacritical marks on the same letter. To fill this gap, covington provides the following macros:

\twodias{<upper diac.>}{<lower diac.>}{<char>}
to combine any two diacritics, e. g., `\twodias{\~}{\=}{a}` = ã̄

\acm{...} for acute over macron, e. g., `\acm{a}` = ā́
\grm{...} for grave over macron, e. g., `\grm{a}` = ā̀
\cim{...} for circumflex over macron, e. g., `\cim{a}` = ā̂

[2] The previous option **ownfncounter** is still supported, but deprecated.
[3] The previous option **ownfncounter*** is still supported, but deprecated.

The first of these is the general case[4] and the latter three are special cases that are often used in Greek transcription. Now you can type *Koinḗ* with both accents in place.

The vertical distance between the two diacritics can be adjusted via the macro **\SetDiaOffset{<length>}** which lets you increase or decrease the vertical space that is currently in effect. If you'd use `\SetDiaOffset{-0.25ex}`, the above examples would come out as

\twodias{<upper diac.>}{<lower diac.>}{<char>}
to combine any two diacritics, e. g., `\twodias{\~}{\=}{a}` = ã

\acm{...} for acute over macron, e. g., `\acm{a}` = ā́
\grm{...} for grave over macron, e. g., `\grm{a}` = ā̀
\cim{...} for circumflex over macron, e. g., `\cim{a}` = ā̂

with a slightly better matching distance for the font used here.

Note that not all accent macros work in the `tabbing` environment. Use the `Tabbing` package or refer to [7] for alternative solutions.

3 Numbered examples

Linguistic papers often include numbered examples. With covington, generating those is straightforward. In this section, we describe how you can typeset a self-stepping example number (see section 3.1), a single numbered example (sec. 3.2), a consecutive range of numbered examples (sec. 3.3), and alpha-numerically labeled sub-examples (sec. 3.4).

Covington's numbered examples can be customized in many ways. For instance, you can make them by default be typeset in italics, as common in linguistics, you can adjust the way the numbers are displayed, and you can customize the spacing and indentation if the default settings do not suit you. This can be achieved via example options that can be set locally to a specific environment as optional arguments or globally via the **\setexampleoptions** macro (see sec. 3.2 for details).

All numbered examples can be referred to in the text via `\label` and `\ref` as usual, or with a convenience macro **\pxref** that adds the parentheses that are usually used in example references (see sec. 3.6 for details).

3.1 Example numbers

The macro **\exampleno** generates a new example number, stepped by 1. It can be used anywhere you want the number to appear. For example, to display a sentence with a number at the extreme right, do this:

```
\begin{flushleft}
This is a sentence. \hfill (\exampleno)
\end{flushleft}
```

[4] Alternatively, there's also the old syntax **\twoacc[<upper diac.>|<char with lower diacr.>],** e. g. `\twoacc[\~|\={a}]` to the same effect, which is however discouraged due to its rather odd form.

Here's what you get:

This is a sentence. (1)

If you want to output the (current) number without stepping it at the same time, the starred form **\exampleno*** will do that.

Normally, however, you do not need to manually place **\exampleno** yourselves, as in the example above. For the common case where example numbers in parentheses are placed left to the example, covington provides more convenient solutions. These are described in turn.

3.2 The `example` environment

The **example** environment (alias **covexample**[5]) displays a single example with a generated example number to the left of it. If you type

```
\begin{example}
This is a sentence.
\end{example}
```

or

```
\begin{covexample}
This is a sentence.
\end{covexample}
```

you get:

(2) This is a sentence.

The example can be of any length; it can consist of many lines (separated by \\), or even whole paragraphs.

The **example** environment provides the following options which also allow for customization of their appearance:

fs={⟨font settings⟩} Default: `{\normalfont\upshape}`. Adjusts the font settings of the example text. Valid values are LaTeX font switches such as `\itshape`, `\bfseries` etc.

fsno={⟨font settings⟩} Default: `{\normalfont\upshape}`. Adjusts the font settings of the example number. Valid values are LaTeX font switches such as `\itshape`, `\bfseries` etc.

[5] As of version 1.1, covington checks if there is already an `example` environment defined (e. g., by the class). If so, covington does not define its own one. The alias environment **covexample** which can be used in order to produce `covington's` example, however, is always defined and can be used in such cases. If you use the package option **force**, covington will override existing `example` environments. In any case, the package will issue a warning if `example` is already defined (this is the case, for instance, if you use covington with the `beamer` class).

judge={⟨arbitrary text⟩} Insert grammaticality judgments (such as * or ?) to an example. This will be inserted before the example text with a small space and uses its own markup. For convenience, the following shorthand options are also provided: **`*`**, **`?`**, **`*?`**, **`??`**, and **`#`** (as alias to **`judge=*`** etc.).

fsjudge={⟨font settings⟩} Default: `{\normalfont\upshape}`. Adjusts the font settings of the grammaticality judgment. Valid values are LaTeX font switches such as `\itshape`, `\bfseries` etc.

preamble={⟨arbitrary text⟩} Arbitrary text that is inserted on an own line after the example number (preceding the example text). This might be useful, for instance, to give context information, to specify the language or the source in case of cited examples. The advantage over just adding a line manually in the example is that you can globally or locally set the markup with the **fspreamble** option. Also, judgment markers are properly set after the preamble.

postamble={⟨arbitrary text⟩} Arbitrary text that is appended to the example (on the same line after a space). This might be useful to add a reference or an explanation to the translation. The advantage over just adding text manually in the example is that that you can globally or locally set the markup with the **fspostamble** option.

postamble*={⟨arbitrary text⟩} A variant of **postamble** that does not add a space. This might be preferred if you want to add a footnote. Note that **postamble** and **postamble*** are mutually exclusive.

If you want the postamble text to be on a line of its own, simply add a line break (i. e., `postamble={\\Text}` or, to the same effect, `postamble*={\\Text}`).

fspreamble={⟨font settings⟩} Default: `{\normalfont\upshape}`. Adjusts the font settings of the preamble text. Valid values are LaTeX font switches such as `\itshape`, `\bfseries` etc.

fspostamble={⟨font settings⟩} Default: `{\normalfont\upshape}`. Adjusts the font settings of the postamble text. Valid values are LaTeX font switches such as `\itshape`, `\bfseries` etc.

leftmargin={⟨length⟩} Default: `{0pt}`. Lets you change the indentation of examples (including number). Positive values will increase, negative values will decrease the indentation of the example.

addnumbersep={⟨length⟩} Default: `{0pt}`. Increases or (with negative values) decreases the spacing between example number and example.

judgewidth={⟨text⟩} Sets the width that is reserved for the judgment marker. This is automatically set to the input of **judge**. By default and notably in the **examples** and **examples** environments which use an angular **\item** argument rather than the **judge** option, this is preset to `{?}`.

addjudgesep={⟨length⟩} Default: `{0pt}`. Increases or (with negative values) decreases the spacing between judgment mark and example (which amounts to 0.2 em by default).

numberformat={⟨template⟩} Default: `{(1)}`. The format of the example number. The option takes templates as known from the enumerate package [2]. Arabic numbering is represented by ⟨1⟩, Roman by ⟨i⟩ and ⟨I⟩, alphabetic by ⟨a⟩ and ⟨A⟩. Any text before and after the number are inserted as is.

For instance, `numberformat={[a.]}` results in lower alphabetic numbering followed by a dot, embraced in brackets.

Note that if the equation counter is used with chapters, this resets the separator between chapter number and example number to the default (a dot). To override this, you might redefine the macro **\covchapexsep**, which is predefined to `{.}`.

fnnumberformat={⟨template⟩} Default: `{(i)}`. The format of the example number if **fnexamplecounter=own** or **fnexamplecounter=own-reset** and example is in a footnote. For the template format, see **numberformat**.

If you pass them to an environment (as optional argument of the environment), the options will only apply to that very environment. If you want to make a global change, you can use the macro

```
\setexampleoptions{<options>}
```

and pass either of the above options to it (as usual separated by comma). This will apply to all subsequent examples (the **example** environment described here as well as **examples**, **subexamples** and glosses with the **ex** option) until further global change and unless an option is altered locally via optional argument.

Please refer to sec. 3.5 for exemplifications of the options.

3.3 The examples environment

To display a series of examples together, each with its own example number, use **examples** (or **covexamples**[6]) instead of **example** or **covexample**. There can be more than one example with this environment, and each of them has to be introduced by `\item`, like this:

```
\begin{examples}
\item This is the first sentence.
\item This is the second sentence.
\end{examples}
```

or, respectively:

```
\begin{covexamples}
\item This is the first sentence.
\item This is the second sentence.
\end{covexamples}
```

[6]The fallback mechanism described in footnote 5 for `example` apply here, too.

This prints as:

(3) This is the first sentence.

(4) This is the second sentence.

Besides to the normal optional item argument (see below), the examples environment provides an optional angular argument which allows you to pass a judgment marker. Consider:

```
\begin{examples}
\item<*> This sentence not is well-formed.
\item This is a well-formed sentence.
\end{examples}
```

This is printed as:

(5) *This sentence not is well-formed.

(6) This is a well-formed sentence.

If you use a wider marker such as ⟨??⟩, you need to adjust the width of the reserved space for the marker. To this end, the option **judgewidth={⟨text⟩}** is provided (e. g., `judgewidth={??}`). With even wider markers, you might also need to adjust **addnumbersep** to prevent the marker from coming too close to the example number.

The **examples** environments accepts the same options than the **example** environment (please refer to sec. 3.2 for details). However, there are some notable deviations and exceptions due to the fact that we are dealing with multiple examples in a row here:

- **judge={⟨arbitrary text⟩}** and its shorthand aliases (*****, **?**, ***?**, **??**, and **#**) set the default judgment marker for all items in the environment. It also sets the appropriate **judgewidth** for the selected marker. Use the aforementioned angular **\item** option to overwrite the preset marker or to set one for indivual examples only.
- **preamble** and **postamble[*]** do not have any effect at all. Preambles and postambles, after all, belong to individual examples in the set rather than to the whole set of examples (or their first and last item, respectively). Instead of these options, we thus provide per-item solutions, which are documented in what follows.

For preambles and postambles, we provide specific macros, namely **\expreamble** and **\expostamble**, which can be placed where they should appear in the example, and which take the same markup than the texts passed via **preamble** or **postamble** option in the **example** environment.

Hence, also here, the advantage over just adding text as is in the example is that an own, globally settable markup is used. This is particularly relevant if you globally set examples to be italicized via the **fs** option (as we have done it in our examples in sec. 3.5). Also, judgment markers are properly set after the preamble.

Consider:

```
\begin{examples}
\item \expreamble{Here is Jakobson's famous poetic example:}
      I like Ike!
\item \expreamble{Here is the less poetic variant:}
      I am in favor of Mr. Eisenhower.
\end{examples}
```

This prints as follows:

(7) Here is Jakobson's famous poetic example:
I like Ike!

(8) Here is the less poetic variant:
I am in favor of Mr. Eisenhower.

To see the advantage more clearly, let us locally set the markup of the example text:

```
\begin{examples}[fs={\itshape}]
\item \expreamble{Here is Jakobson's famous poetic example:}
      I like Ike!
\item \expreamble{Here is the less poetic variant:}
      I am in favor of Mr. Eisenhower.
\end{examples}
```

As you can see, the preamble text is untouched:

(9) Here is Jakobson's famous poetic example:
I like Ike!

(10) Here is the less poetic variant:
I am in favor of Mr. Eisenhower.

Along the same line, consider:

```
\begin{examples}[fs={\itshape}]
\item I like Ike! \expostamble{(Jakobson 1960: 357)}
\item I am in favor of Mr. Eisenhower.\expostamble{\footnote{He actually wasn't.}}
\end{examples}
```

which results in:

(11) *I like Ike!* (Jakobson 1960: 357)

(12) *I am in favor of Mr. Eisenhower.*[7]

Like in any other LaTeX list, the numbering can be freely set via the optional argument of `\item`. For a somewhat realistic example, observe how:

[7] He actually wasn't.

```
\begin{examples}
\item A proposition.
\item[\covexnumber{\exampleno*'}] An alternative proposition.
\end{examples}
```

results in:

(13) A proposition.

(13') An alternative proposition.

If you need an angular (judgment) argument and an optional argument, pass the judgment argument first (i. e., `\item<judgment>[custom label]`).

3.4 The `subexamples` environment

Sometimes a set of (paradigmatic) sub-examples gets only one main example number with alphabetic sub-numbering, as in (15 a) below. An ad-hoc way to achieve this is to use `itemize` or `enumerate` within an example with customized items, like this:

```
\begin{example}
\begin{itemize}
\item[(a)] This is the first sentence.
\item[(b)] This is the second sentence.
\end{itemize}
\end{example}
```

This prints as:

(14) (a) This is the first sentence.
(b) This is the second sentence.

However, the **`subexamples`** (or **`covsubexamples`**[8]) environment, described in this section, is usually more convenient for this task. As opposed to **`examples`**/**`covexamples`**, this environment sub-numbers its items. Thus

```
\begin{subexamples}
\item This is the first sentence.
\item This is the second sentence.
\end{subexamples}
```

or, respectively:

```
\begin{covsubexamples}
\item This is the first sentence.
\item This is the second sentence.
\end{covsubexamples}
```

prints as:

[8] The fallback mechanism described in footnote 5 for `example` apply here, too.

(15) (a) This is the first sentence.

(b) This is the second sentence.

Similar to the **examples** environment, the **subexamples** environment's item has an angular argument to pass grammaticality judgment markers, as in:

(16) (a) This is a well-formed sentence.

(b) *This not well-formed is.

which was typed as:

```
\begin{subexamples}
\item This is a well-formed sentence.
\item<*> This not well-formed is.
\end{subexamples}
```

The settings of the judgment markers (spacing and markup) is done via the options described in sec. 3.2.

The **subexamples**/**covsubexamples** environment provides the same options than the **example** and **examples** environment (see sec. 3.2) with the following deviations and additions:

judge={⟨arbitrary text⟩} and its shorthand aliases (*****, **?**, ***?**, **??**, and **#**) set the default judgment marker for all items in the environment. It also sets the appropriate **judgewidth** for the selected marker. Use the aforementioned angular **\item** option to overwrite the preset marker or to set one for indivual subexamples only.

preamble={⟨arbitrary text⟩} Arbitrary text is inserted here on the first line (after the main number and before the first sub-example, which then follows on a new line). This might be useful, for instance, to give context information, to specify the language or the source in case of cited sub-examples.

postamble={⟨arbitrary text⟩} Arbitrary text is appended here after all sub-examples (on a new line). This might be useful to add a reference or an explanation to the whole set.

addsubnumbersep={⟨length⟩} Default: `{0pt}`. Increases or (with negative values) decreases the spacing between example subnumber and example text.

subnumberformat={⟨template⟩} Default: `{(a)}`. The format of the sub-example number. For the template format, consult the documentation of **numberformat** in sec. 3.2.

subnumberrefsep={⟨text or command⟩} Default: `\thinspace`. The separator (space or characters) between example number and subnumber in cross-references (use `{}` to get no separation at all).

Also note that, if you set options globally via **\setexampleoptions**, the following derivating options should be used to distinguish the **subexample** from the **example** case.

subjudge={⟨text⟩} sets the default judge marker for subexamples.

subpreamble={⟨arbitrary text⟩} For a global preamble presetting in subexamples.

subpostamble={⟨arbitrary text⟩} For a global postamble presetting in subexamples.

fssubpreamble={⟨font settings⟩} Default: `{\normalfont\upshape}`. Adjusts the font settings of subexample's preamble text and the **\subexpreamble** macro described below. Valid values are LaTeX font switches such as `\itshape`, `\bfseries` etc.

fssubpostamble={⟨font settings⟩} Default: `{\normalfont\upshape}`. Tweaks the subexample's postamble text font settings and the **\subexpostamble** macro described below. Valid values are LaTeX font switches such as `\itshape`, `\bfseries` etc.

For convenience, all these options actually also work (as aliases) in optional arguments passed to **subexamples** environment.

To add additional text preceding or following specific subitems, you can principally use the **\expreamble** and **\expostamble** macros desribed in sec. 3.3. To allow for separating the markup of such texts in subexample from the example case, we also provide specific macros, **\subexpreamble** and **\subexpostamble** which use their own font settings (and the same than the **preamble** and **postamble** options of this environment). With all these macros, judgment markers are properly set after the preamble.

For example:

```
\begin{covsubexamples}[preamble={Here are two sentences:}]
\item \subexpreamble{English:} This is the first sentence.
\item \subexpreamble{German:} Das ist der zweite Satz.
\end{covsubexamples}
```

(17) Here are two sentences:

(a) English:
This is the first sentence.

(b) German:
Das ist der zweite Satz.

Analoguously, with:

```
\begin{subexamples}
\item This is the first sentence. \subexpostamble{(Miller 2022: 15)}
\item This is the second sentence.\subexpostamble{\footnote{My own.}}
\end{subexamples}
```

you will get:

(18) (a) This is the first sentence. (Miller 2022: 15)

(b) This is the second sentence.[9]

More customization possibilities of examples and subexamples are elaborated in the following section.

[9] My own.

3.5 Examples of examples

As already noted, all example environments can be customized in many aspects via the example options. If passed as an optional argument to the respective environment, the change only applies to this very environment. If passed via **\setexampleoptions**, the change will apply to any subsequent **example**, **examples**, or **subexamples** environment unless a local option overrides a setting.

For this section and the purpose of demonstration, we globally set the markup of example sentences to italics, as common in linguistics. As you now already know, this is very easy to do, via:

```
\setexampleoptions{fs=\itshape}
```

which applies from now on. If we now enter:

```
\begin{example}[preamble={Consider this example:},
                postamble={(German)}]
Das ist ein Satz.
\end{example}
```

We will get:

(19) Consider this example:

Das ist ein Satz. (German)

And with a similar subexample:

```
\begin{subexamples}[preamble={Here are two sentences},
                    postamble={(Doe 1974: 12)}]
\item This is the first sentence.
\item This is the second sentence.
\end{subexamples}
```

we get the following output:

(20) Here are two sentences

(a) *This is the first sentence.*

(b) *This is the second sentence.*

(Doe 1974: 12)

Let us briefly demonstrate the judgment feature. If we want to mark an example as being ungrammatical, we use the conventional asterisk mark, like this:

```
\begin{example}[judge=*]
Dies ein ungrammatischem Satz ist.
\end{example}
```

or, shorter:

```
\begin{example}[*]
Dies ein ungrammatischem Satz ist.
\end{example}
```

to get:

(21) * *Dies ein ungrammatischem Satz ist.*

With the **examples** and **subexamples** environment, we need to pass the marker to the item's angular argument instead, as already demonstrated in sec. 3.3:

```
\begin{examples}
\item<*> Dies ein ungrammatischem Satz ist.
\item Dies ist ein grammatisch wohlgeformter Satz.
\end{examples}
```

(22) * *Dies ein ungrammatischem Satz ist.*

(23) *Dies ist ein grammatisch wohlgeformter Satz.*

With wider markers, we need to adjust the marker width via **judgewidth**, and since we have a larger count of examples now, we also adjust **addnumbersep** to have a good space to the number:

```
\begin{examples}[judgewidth={??},addnumbersep={1ex}]
\item<??> Kann ich ein gratis Bier haben?
\item Kann ich ein Bier gratis haben?
\end{examples}
```

(24) ?? *Kann ich ein gratis Bier haben?*

(25) *Kann ich ein Bier gratis haben?*

Covington takes care that judgment markers are also properly set with preambles (with the **\expreamble** and **\subexpreamble** macros, this requires an extra LaTeX pass). Observe how:

```
\begin{examples}
\item \expreamble{Compare this:} Oh, I am so well-formed!
\item<*> \expreamble{with this:} Lucky are you!
\end{examples}
```

comes out as expected:

(26) Compare this:
Oh, I am so well-formed!

(27) with this:
* *Lucky are you!*

In what follows, we demonstrate further adjustments of the spacing of examples. The space between the example number and the text (or the subnumber in case of subexamples) can be adjusted, as documented above, by means of the **addnumbersep** example option. So, for instance,

```
\begin{example}[addnumbersep={2em}]
Look, I am shifted right!
\end{example}
```

will come out like this:

(28) *Look, I am shifted right!*

The indentation (left margin) of examples and subexamples is controlled by the **leftmargin** example option. Doing **leftmargin=1em**, for instance, will increase the indentation by 1 em, negative values will decrease it accordingly. For the latter, consider:

```
\begin{example}[leftmargin={-2.6em}]
This is a marginal case.
\end{example}
```

which will come out like this:

(29) *This is a marginal case.*

In case of **subexamples**, the distance between example subnumber and text can be changed via the **addsubnumbersep** option. Again, a positive value will increase, a negative value will decrease the distance. Doing

```
\begin{subexamples}[addnumbersep={-0.4em},addsubnumbersep={0.4em}]
\item My numbers have been moved.
\item Mine as well.
\end{subexamples}
```

for instance, will yield:

(30) (a) *My numbers have been moved.*

(b) *Mine as well.*

The example also demonstrates how, in subexamples, **addnumbersep** alters (here: decreases) the spacing between main number and subnumber.

Next, we showcase how to change the display of the example number or subnumber via the **numberformat** and **subnumberformat** options. As described in sec. 3.2, these take a 'mini template' as known from the enumerate package [2]. In these templates, the first occurrence of the characters ⟨1⟩, ⟨i⟩, ⟨I⟩, ⟨a⟩, and ⟨A⟩ represents the numeric system where the respective character marks the first positive value. All the rest of the template (including following occurrences of the mentioned characters) is interpreted literally. For instance, ⟨(1)⟩ is an alphabetic number in parentheses, ⟨a.⟩ a lower alphabetic value followed by a dot, etc.

So to equip the examples with a dot after the number, we use **numberformat={(1.)}** To remove the parentheses from the subexample letter, for instance, we can use **subnumberformat={a}**. The following example demonstrates both options:

```
\begin{subexamples}[numberformat={(1.)},subnumberformat={a}]
\item This is a statement.
\item This is an adjacent statement.
\end{subexamples}
```

And we get what we want:

(31.) a *This is a statement.*

b *This is an adjacent statement.*

Note that the **numberformat** option by default also changes the numbers of examples in footnotes, even if **fnexamplecounter=own[-reset]** is used. This does not apply to the counter format, however. So **numberformat={[1.])}** will also make the example numbers in footnote appear in brackets and with the dot, but it will still use the lower Roman numbering. Use **fnnumberformat** to fully change the appearance.

With the last examples, we return to example markup. Remember, we have globally set the markup of example sentences to italics at the beginning of this section. Now if we want to have it bold instead for one single occurence, we can reset it for the one case via the environment options (if we wanted to change it globally from now on, we would use **\setexampleoptions** once more). Observe how:

```
\begin{example}[fs=\bfseries,preamble={Note:},
                postamble={(in any respect)}]
This is a bold statement!
\end{example}
```

results in:

(32) Note:

This is a bold statement! (in any respect)

As outlined in sec. 3.2 and 3.4, all elements of examples and subexamples (number, actual example text, pre- and postamble) can be marked up independently via **fsno**, **fspreamble**, **fspostamble**, etc. Hence, many useful (and also lots of nonsensical) example formats are easily achievable.

3.6 Referring to examples

References to examples and sub-examples can be made the usual way via the **\ref** command (which refers to a **\label** that is placed in the respective example paragraph). The references do not have parentheses by default, i. e., a reference to the example in section 3.2 would be printed as 2, a reference to the sub-example in section 3.4 as 15 a. For convenience, though, covington provides a command **\pxref** that also prints the parentheses, as in (2) and (15 a). It is defined as follows and can be redefined if needed:

```
\providecommand*\pxref[1]{(\ref{#1})}
```

The spacing between example number and subnumber can be customized via the option **subnumberrefsep** of **\setexampleoptions** or the **subexamples** environment (see sec. 3.4).

3.7 Numbering per chapter or continuously

With the default (equation) counter which is used for examples, examples automatically get numbered per chapter if the document has chapters (e.g., (2.1) for the first example in chapter 2). If you want continuously numbered examples instead, use

```
\counterwithout{equation}{chapter}
```

which also will change the equation numbering behavior.

With **owncounter**, on the other hand, examples are always continuously numbered by default. If you want examples numbered per chapter, do:

```
\counterwithin*{covex}{chapter}
\renewcommand\ownexcounterprep{\thechapter.}
```

4 Glossing sentences word-by-word

To gloss a sentence is to annotate it word-by-word. Most commonly, a sentence in a foreign language is followed by a word-for-word translation (with the words lined up vertically) and then a smooth translation (not lined up), like this:

Dit	*is*	*een*	*Nederlands*	*voorbeeld*
This	is	a	Dutch	example

'This is an example in Dutch.'

Covington provides different ways to typeset such glosses. The most convenient way is via gloss macros (see sec. 4.1), an alternative (and the traditional) way is via a set of commands (see sec. 4.2). Both are described in turn.

4.1 Gloss macros

Covington provides two gloss macros:

```
\digloss[<options>]
        {<gloss line 1>}[<optional comment line 1>]
        {<gloss line 2>}[<optional comment line 2>]
        {<free translation>}
```

typesets two-line glosses with a translation line, and optionally comments to the right side of the gloss lines (the macro name puns on Greek *diglossía*, lit. 'two tongues', 'bilingualism').

```
\trigloss[<options>]
         {<gloss line 1>}[<optional comment line 1>]
         {<gloss line 2>}[<optional comment line 2>]
         {<gloss line 3>}[<optional comment line 3>]
         {<free translation>}
```

typesets three-line[10] glosses with a translation line and optional comments (cf. Greek *triglossía*, lit. 'three tongues', 'trilingualism').

The example given above would thus be typed as:

```
\digloss{Dit is een Nederlands voorbeeld}
        {This is a Dutch example}
        {This is an example in Dutch.}
```

Note that:

- The ⟨free translation⟩ argument can be left empty. In this case, no translation line is added (and no extra space taken).
- The macros automatically markup the lines. By default, the first gloss line is in italics, subsequent lines are set upright, and the free translation in single quotation marks (using the language-sensitive csquotes [6] macros if this package is loaded). This can be customized, though, via the macro options or globally via the macro **\setglossoptions** which will be documented below.
- By default, page breaks might occur within glosses. In order to prevent it, the package option **noglossbreaks** (see sec. 1) will help in many cases. If it doesn't, you can wrap the whole gloss into a minipage or parbox.
- The words do not have to be typed lining up; TEX counts and aligns them.
- On the other hand, multiple blanks are ignored, so you can, but do not need to, use the space key to line up the words to your liking in the TEX file. The example above could thus also have been input like this, with no change to the output:

  ```
  \digloss{Dit  is een Nederlands voorbeeld}
          {This is a   Dutch      example}
          {This is an example in Dutch.}
  ```

- If the words in the two languages do not correspond one-to-one, you can use curly brackets to group words and a pair of empty curly brackets to mark null forms. For example, to print

Dit	*is*	*een*	*voorbeeldje*	*in*	*het*	*Nederlands*
This	is	a	little example	in		Dutch

  'This is a little example in Dutch.'

[10] The additional line can be useful for instance to gloss cited forms, morphology, or an additional translation. See sec. 4.3 for examples.

you would type:

```
\digloss{Dit  is een voorbeeldje      in het Nederlands}
        {This is a   {little example} in {}  Dutch}
        {This is a little example in Dutch.}
```

The following ⟨**options**⟩ (key-value pairs) are provided for the two gloss macro:[11]

ex=⟨**true|false**⟩ Default: *false*. Wraps the gloss in an example environment (i. e., it is numbered). In this case, any example settings done via **\setexampleoptions** (sec. 3.2) apply.

tlr=⟨**true|false**⟩ Default: *false*. If set to true, the translation line (content of the ⟨free translation⟩ argument) is set right to the gloss lines, rather than into a new line below, and shifted towards the right margin. In contrast to the normal translation line, the content is not enquoted. Since the gloss itself is set in a box, the ⟨free translation⟩ will appear lined up with the first line of the gloss. This can be useful when no translation, but an aligned number or something similar, is to be inserted right to the gloss (please refer to sec. 4.3 for an example).

Note that this option defuncts the optional gloss line comments.

tlr*=⟨**true|false**⟩ Default: *false*. A variant of **tlr** which is suitable for setting real free translations (enquoted) right to the gloss (after a small space). Both the gloss and the translation line are placed in a box with limited width in this case. The maximum width of the translation line can be adjusted via the **glosscommentwidth** option described below, the maximum width of the gloss and the space between gloss and translation line via **glosswidth** and **glosssep**, respectively.

fsi={⟨**font settings**⟩**}** . Default: `{\normalfont\itshape}`. Adjusts the font settings of the first gloss line. Valid values are LaTeX font switches such as `\itshape`, `\bfseries` etc.

fsii={⟨**font settings**⟩**}** Default: `{\normalfont\upshape}`. Adjusts the font settings of the second gloss line. Valid values are LaTeX font switches such as `\itshape`, `\bfseries` etc.

fsiii={⟨**font settings**⟩**}** Default: `{\normalfont\upshape}`. Adjusts the font settings of the third gloss line (in case of **\trigloss**). Valid values are LaTeX font switches such as `\itshape`, `\bfseries` etc.

fstl={⟨**font settings**⟩**}** Default: `{\normalfont\upshape}`. Adjusts the font settings of the translation line. Valid values are LaTeX font switches such as `\itshape`, `\bfseries` etc.

enquotetl=⟨**true|false**⟩ Default: *true*. By default, the translation line is embraced in single quotation marks. Set this option to ⟨false⟩ to omit this.

[11] Please consult sec. 4.3 below for examples that showcase these options.

Note that for **enquotetl=true**, covington checks at document begin whether the csquotes [6] package is loaded. If so, it uses its language-sensitive **\enquote*** macro for enquoting the translation. If not, a fallback quotation (using English single quotation marks) is used. The usage of csquotes is highly recommended!

addlinesepi={⟨length⟩} . Default: `{0pt}`. Increases or (with negative values) decreases the vertical spacing after the first gloss line.

addlinesepii={⟨length⟩} Default: `{0pt}`. Increases or (with negative values) decreases the vertical spacing after the second gloss line (in **\digloss**, this is only done if a translation line follows).

addlinesepiii={⟨length⟩} Default: `{0pt}`. Increases or (with negative values) decreases the vertical spacing after the third gloss line if a translation line follows (in case of **\trigloss**).

judge={⟨arbitrary text⟩} Insert grammaticality judgments (such as * or ?) to the gloss. This will be prepended to the first gloss line with a small space and uses its own markup. For convenience, the following shorthand options are also provided: *****, **?**, ***?**, **??**, and **#** (as alias to **judge=*** etc.).

fsjudge={⟨font settings⟩} Default: `{\normalfont\upshape}`. Adjusts the font settings of the grammaticality judgment. Valid values are LaTeX font switches such as `\itshape`, `\bfseries` etc.

addjudgesep={⟨length⟩} Default: `{0pt}`. Increases or (with negative values) decreases the spacing between judgment mark and gloss (which amounts to 0.2 em by default).

preamble={⟨arbitrary text⟩} Arbitrary text that is inserted on an own line before the interlinearized gloss. This might be useful, for instance, to give context information, to specify the language or the source in case of cited glosses.

The advantages over just adding a line manually above the gloss are that you can locally and globally set the markup with the **fspreamble** option, and that such lines are kept on the same page than the gloss with the option **noglossbreaks** (at least as long as preamble does not exceed one line). Furthermore, this option is the only way to add such text when the **ex** option is used.

postamble={⟨arbitrary text⟩} Arbitrary text that is appended to the translation line, but after the closing quotation mark. This might be useful to add a footnote or some additional text to the translation line, after the actual translation.

fspreamble={⟨font settings⟩} Default: `{\normalfont\upshape}`. Adjusts the font settings of the preamble line. Valid values are LaTeX font switches such as `\itshape`, `\bfseries` etc.

fspostamble={⟨font settings⟩} Default: `{\normalfont\upshape}`. Adjusts the font settings of the postamble line. Valid values are LaTeX font switches such as `\itshape`, `\bfseries` etc.

glosswidth={⟨length⟩} Default: `{0.6\textwidth}`. Sets the maximum width the gloss takes if optional comments or **tlr*=true** are used.

glosssep={⟨length⟩} Default: `{1em}`. Sets space between gloss and optional comments, or between gloss and free translation in case of **tlr*=true**, respectively.

glosscommentwidth={⟨length⟩} Sets the maximum width of gloss comments, or of free translation in case of **tlr*=true**, respectively. By default this is automatically calculated from the above two lengths.

fscomments={⟨font settings⟩} Default: `{\normalfont\upshape}`. Adjusts the font settings of the gloss comments. Valid values are LaTeX font switches such as `\itshape`, `\bfseries` etc.

If passed to a **\digloss** or **\trigloss** macro, the options will only apply to this very gloss. If you want to make a permanent change, you can use the macro

```
\setglossoptions{<options>}
```

and pass either of the above options to it. This will apply to all subsequent glosses (until further global change and unless the setting is altered locally via macro option).

4.2 Glossing with low-level commands

The gloss macros described above build on low-level commands[12] which can also be used directly (this was actually the only way up to covington 2.0 which introduced the macros). Low-level commands can be useful if you want to do fancy things in a gloss. Note, though, that using them also has limitations: Some commands cannot be used inside macros (such as footnotes), and the markup is not done automatically in all cases (as documented in what follows); furthermore, you cannot make use of the options the macros have. Thus, we strongly suggest to use the macros, unless you have very good reasons not to do that.

The following is a complete list of all low-level glossing commands:

\gll introduces two lines of words vertically aligned, and activates an environment very similar to `flushleft`. The two lines are separated by a normal line break (carriage return). This is possible since the command makes the line-ending character active. As a consequence, however, this command does not work inside macros (such as `\footnote`).

\glll is like **\gll** except that it introduces *three* lines of lined-up words.

\xgll is similar to **\gll** except that it does not make the line ending active. It thus works inside macros such as footnotes but requires explicit gloss line termination via **\xgle**.

\xglll is similar to **\glll** except that it does not make the line ending active. It thus works inside macros such as footnotes but requires explicit gloss line termination via **\xgle**.

[12] The commands are adapted with permission from `gloss.tex`, by Marcel R. van der Goot.

\xgle is a gloss line ending marker to be used with **\xgll** and **\xglll**.

\glt ends the set of lined-up lines and introduces a line (or more) of translation. Note that this command does not markup the translation line (no automatic enquoting). Also, it outputs an empty line if no text follows.

\gln is like **\glt** but does not start a new line (useful when no translation follows but you want to put a number on the right).

\glot{⟨**free translation**⟩**}** is an alternative to **\glt** and a smarter way to insert a translation line. Other than **\glt**, it marks up (by default: enquotes) the translation line. Also, it does not add an empty line if the translation is empty. This macro also takes an optional argument where a judgment marker can be passed. This is for nice alignment with the gloss lines if such a marker is used.

The markup can be customized by redefining the following macro:

```
\newcommand*\glosslinetrans[1]{%
        \bgroup\cov@fstl\ifcov@enquote@tl\covenquote{#1}\else#1\fi\egroup}
```

Note that for **\covenquote**, as used in the default definition, covington checks at document begin whether the csquotes [6] package is loaded. If so, it uses its language-sensitive **\enquote*** macro for enquoting the translation. If not, a fallback quotation (using English single quotation marks) is used. The usage of csquotes is highly recommended!

The **\ifcov@enquote@tl** boolean checks for the **enquotetl** option of the gloss macro. The macro **\cov@fstl** holds the font settings as set via the **fstl** gloss option. These are both relevant also for low-level commands if you set the respective options via **\setglossoptions**.

In general, please take care when redefining this macro that the functioning of some gloss environment options relies on some of the macros contained in the default definition.

\glosspreamble{⟨**arbitrary text**⟩**}** lets you enter text that is printed immediately before the interlinearized gloss (on a line of its own). This might be useful, for instance, to give context information, to specify the language or the source in case of cited glosses. The advantages over just adding a line manually above the gloss are that you can globally set the markup and that such lines are kept on the same page than the gloss with the option **noglossbreaks**. Note, however, that page breaks might occur if this text spans multiple lines. In this case, you can wrap the whole gloss into a minipage. This command has been introduced in covington 2.1.

\glend ends the special `flushleft`-like environment.

Using the low-level commands, the examples given in the previous section would be coded as follows:

```
\gll Dit is een Nederlands voorbeeld.
     This is a Dutch example.
\glt ‘This is an example in Dutch.’
```

```
\glend
```

```
\gll Dit is een voorbeeldje     in het Nederlands
     This is a {little example} in {}  Dutch
\glt ‘This is a little example in Dutch.’
\glend
```

Observe that you need to markup (i. e., enquote) the translation line yourself if you use **\glt**. This is not so if you use **\glot**:

```
\gll Dit is een Nederlands voorbeeld
     This is a Dutch example
\glot{This is an example in Dutch.}
\glend
```

The advantage of **\glot** is that you can easily customize the translation markup globally. Also, **\glot** uses the language-sensitive csquotes [6] macros if this package is loaded (see sec. 4.1).

Since **\gll** and **\glll** locally activate the end of line in glosses in order to identify the different lines of the gloss (via category code change), they do not work inside macros (e. g., if the gloss is in a footnote). To work around this, special versions of the **\gll** and **\glll** commands are provided that do without the character activation: **\xgll** and **\xglll**, respectively. These can also be used in macro arguments; however, the end of each gloss line needs to be explicitly specified by the **\xgle** command in this case. If you want to put the above gloss in a footnote, thus, you would type:

```
\xgll Dit is een voorbeeldje     in het Nederlands.\xgle
      This is a {little example} in {}  Dutch.\xgle
\glt ‘This is a little example in Dutch.’
\glend
```

Note, again, that the macros described in sec. 4.1 do not have this problem.

To sum up: With low-level commands, every glossed sentence begins with either **\gll**, **\xgll**, **\glll** or **\xglll**, then contains either **\glt** or **\gln**, and ends with **\glend**. Layout is critical in the part preceding **\glt** or **\gln**, and fairly free afterward.

4.3 Example glosses

This section gives some further examples and showcases some options. First, a sentence with three lines aligned, instead of just two:

Hoc	*est*	*aliud*	*exemplum*
N.SG.NOM	3SG	N.SG.NOM	N.SG.NOM
This	is	another	example

‘This is another example.’

In order to produce this, we use the **\trigloss** macro for a three-line gloss and pass the **fsii** option with the respective font switches in order to get small capitals in the second line (**\normalfont** is needed here to disable the italics set for the first line):

```
\trigloss[fsii={\normalfont\scshape}]
        {Hoc est aliud exemplum}
        {n.sg.nom 3sg n.sg.nom n.sg.nom}
        {This is another example}
        {This is another example.}
```

Next, an example with a gloss but no translation, with an example number at the right:

Hoc *habet* *numerum* (33)
This has number

That one was typed using the option **tlr**:

```
\digloss[tlr]{Hoc habet numerum}
            {This has number}
            {(\exampleno)}
```

Third, we showcase the **tlr*** option which is handy if you want the free translation to the right:

Tres *faciunt* *collegium* 'Three persons constitute a group.'
Three make society

This has been entered like this:

```
\digloss[tlr*,glosscommentwidth={.6\textwidth}]
       {Tres faciunt collegium}
       {Three make society}
       {Three persons constitute a group.}
```

As you note, we also used the **glosscommentwidth** option to make the box which holds the translation line in this case a bit wider (as the translation is rather long and the gloss itself rather short).

Now, we'll put a glossed sentence inside the `example` environment, which is a very common way of using it:

(34) *Hoc* *habet* *numerum* *praepositum*
This has number preposed
'This one has a number in front of it.'

This last example was, of course, typed as:

```
\digloss[ex]{Hoc habet numerum praepositum}
           {This has number preposed}
           {This one has a number in front of it.}
```

although you could also construct it manually as:

```
\begin{example}
\digloss{Hoc habet numerum praepositum}
        {This has number preposed}
        {This one has a number in front of it.}
\end{example}
```

Here is an example that uses the *Leipzig glossing rules* ([1], cited example: p. 2) and also exemplifies the use of **preamble**:

(35) Lezgian (Haspelmath 1993:207)
Gila abur-u-n ferma hamišaluǧ güǧüna amuq'-da-č.
now they-OBL-GEN farm forever behind stay-FUT-NEG
'Now their farm will not stay behind forever.'

This has been input as follows:

```
\digloss[ex,preamble={Lezgian (Haspelmath 1993:207)}]
        {Gila abur-u-n             ferma hamišalu\v{g} gü\v{g}üna amuq'-da-\v{c}.}
        {now  they-\textsc{obl-gen} farm  forever       behind     stay-\textsc{fut-neg}}
        {Now their farm will not stay behind forever.}
```

Of course, you would use `\cite` in a real document for the citation. Also, if you adhere to the *Leipzig glossing rules*, you might want to check out the leipzig LaTeX package [8] that facilitates the use of the gloss abbreviations that have been entered and marked-up manually here.

Next, we showcase grammatical judgment. Similar to how this is done in examples, we simply pass the respective **judge** option or shorthand:

```
\digloss[ex,??]
  {Hier werden                          Sie                 geholfen!}
  {Here become.\textsc{aux.pass.2sg.pst} you.\textsc{2sg.acc} help.\textsc{pass.ptcp}}
  {Here you will be helped!}
```

resulting in:

(36) ??*Hier werden Sie geholfen!*
Here become.AUX.PASS.2SG.PST you.2SG.ACC help.PASS.PTCP
'You will be helped here!'

Gloss line comments allow you to set arbitrary text right to specific gloss lines, as in:

(37) *Gut Ding will Weile haben* (German proverb)
Good thing want while have
'Slow and steady wins the race'

This was typed in with the help of the optional gloss arguments:

```
\digloss[ex]{Gut Ding will Weile haben}[(German proverb)]
           {Good thing want while have}
           {Slow and steady wins the race}
```

The final example exemplifies the use of **postamble**. In what follows we add a footnote to the translation line, but outside the translation (as marked-up by quotation marks):

(38) *Mein* *Luftkissenboot* *ist* *voller* *Aale*
My hovercraft is full of eels
'Do you have matches?'[13]

This has been input as follows:

```
\digloss[ex, postamble={\footnote{Yes, really!}}]
       {Mein Luftkissenboot ist voller    Aale}
       {My   hovercraft     is  {full of} eels}
       {Do you have matches?}
```

5 Phrase structure rules

To print phrase structure rules such as $S \rightarrow NP\ VP$ you can use `covington's` macro **\psr{⟨constituent⟩}{⟨sub-constituents⟩}** (for the given example, `\psr{S}{NP~VP}`).

6 Feature structures

To print a feature structure such as

$$\begin{bmatrix} case : nom \\ person : P \end{bmatrix}$$

you can type:

```
\fs{case:nom \\ person:P}
```

The feature structure can appear anywhere – in continuous text, in a displayed environment such as `flushleft`, or inside a phrase-structure rule, or even inside another feature structure.

To put a category label at the top of the feature structure, like this,

$$N \begin{bmatrix} case : nom \\ person : P \end{bmatrix}$$

here's what you type:

```
\lfs{N}{case:nom \\ person:P}
```

And here is an example of a PS-rule made of labeled feature structures:

$$S \begin{bmatrix} tense : T \end{bmatrix} \quad \rightarrow \quad NP \begin{bmatrix} case : nom \\ number : N \end{bmatrix} \quad VP \begin{bmatrix} tense : T \\ number : N \end{bmatrix}$$

[13] Yes, really!

which was obviously coded as:

```
\psr{\lfs{S}{tense:T}}
    {\lfs{NP}{case:nom \\  number:N}
     \lfs{VP}{tense:T \\ number:N} }
```

7 Discourse Representation Structures

Several macros in covington facilitate display of *Discourse Representation Structures* (DRSes) in the box notation introduced by Hans Kamp [5]. The simplest one is **\drs**, which takes two arguments: a list of discourse variables joined by ~, and a list of DRS conditions separated by \\. Nesting is permitted.

Note that the **\drs** macro itself does not give you a displayed environment; you must use `flushleft` or the like to display the DRS.

Here are some examples:

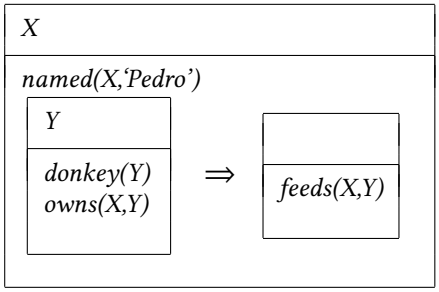
```
\begin{flushleft}
  \drs{X}
      {
       donkey(X)\\
       green(X)
      }
\end{flushleft}
```

prints as:

X
donkey(X) *green(X)*

```
\begin{flushleft}
  \drs{X}
      {
       named(X,'Pedro')\\
       \drs{Y}
           {
            donkey(Y)\\
            owns(X,Y)
           }
       ~~{\large $\Rightarrow$}~
       \drs{~}
           {feeds(X,Y)}
      }
\end{flushleft}
```

comes out as:



Note that the alignment of the input is fairly free, so you can also write the two arguments of **\drs** in one line, like:

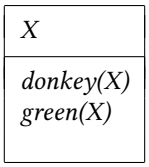
```
\drs{X}{donkey(X)\\green(X)}
```

To display a sentence above the DRS, use **\sdrs**, which has one extra argument for this purpose, as in:

```
\begin{flushleft}
  \sdrs{A donkey is green.}
       {X}
       {donkey(X)\\green(X)}
\end{flushleft}
```

which prints as:

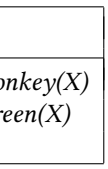
A donkey is green.



Some DRS connectives are also provided (normally for forming DRSes that are to be nested within other DRSes). The macro **\negdrs** forms a DRS preceded by a negation symbol, so

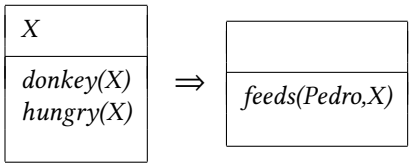
```
\negdrs{X}
       {
        donkey(X)\\
        green(X)
       }
```

comes out as

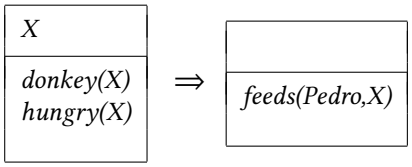


Finally, **\ifdrs** forms a pair of DRSes joined by a big arrow, like this:

```
\ifdrs{X}
      {
       donkey(X)\\
       hungry(X)
      }
      {~}
      {feeds(Pedro,X)}
```



If you have an "if"-structure appearing among ordinary predicates inside a DRS, you may prefer to use **\alifdrs**, which is just like **\ifdrs** but shifted slightly to the left for better alignment:



Note that for more extended DRS representations, dedicated packages are meanwhile available, most notably the drs [3] and the sdrt [4] package. Both packages actually draw on covington, add some additional features and, in some cases, tweak the layout to (what strikes those package authors) the better. If the rather basic DRS macros provided by covington do not suit you, please check if one of those packages does.

Note, though, that while sdrt introduces new (capitalized) macro naming which lets the package peacefully coexist with covington, drs simply re-uses covington's macro names, which makes the two packages incompatible. In order to fix this, covington checks whether the DRS macros are already defined when it is loaded; if so, it does not define its own ones. So if you want to use the DRS macros of the drs package together with covington's non-DRS features, you can do so, provided that drs is loaded *before* covington. In that case, covington's own DRS macros are disabled.

8 Exercises

The **exercise** environment (alias **covexercise**) generates an exercise numbered according to chapter, section, and subsection (suitable for use in a large book; in this example, the subsection number is going to come out as 0). Here is an example:

Exercise 8.0.1 (Project) *Prove that the above assertion is true.*

This was coded as

```
\begin{exercise}[Project]
Prove that the above assertion is true.
\end{exercise}
```

The argument (`[Project]` in the example) is optional.

Note that, as of version 1.1, covington checks if there is already an **exercise** environment defined (e. g., by the class). If so, covington does not define its own one. However, there is always the alias environment **covexercise** which can be used in order to produce `covington's` exercise. If you use the package option **force**, covington will override existing **exercise** environments. In any case, the package will issue a warning if **exercise** is already defined.

9 Reference Lists

To type a simple LSA-style hanging-indented reference list, you can use the **reflist** environment. (*Note:* **reflist** is not integrated with BibTEX or even the `bibliography` environment in any way, so it is not suitable for citing.[14]) For example,

```
\begin{reflist}
Barton, G. Edward; Berwick, Robert C.; and Ristad, Eric Sven.  1987.
Computational complexity and natural language.  Cambridge,
Massachusetts: MIT Press.

Chomsky, Noam.  1965.  Aspects of the theory of syntax.  Cambridge,
Massachusetts: MIT Press.

Covington, Michael.  1993.  Natural language processing for Prolog
programmers.  Englewood Cliffs, New Jersey: Prentice-Hall.
\end{reflist}
```

prints as:

Barton, G. Edward; Berwick, Robert C.; and Ristad, Eric Sven. 1987. Computational complexity and natural language. Cambridge, Massachusetts: MIT Press.

Chomsky, Noam. 1965. Aspects of the theory of syntax. Cambridge, Massachusetts: MIT Press.

Covington, Michael A. 1993. Natural language processing for Prolog programmers. Englewood Cliffs, New Jersey: Prentice-Hall.

[14] For BibTEX, there are several options: the LSA style, as used in the journal *Language*, can be obtained by means of the style files `lsalike.bst` (`http://www.icsi.berkeley.edu/ftp/pub/speech/jurafsky/lsalike.bst`) or `language.bst` (`http://ron.artstein.org/resources/language.bst`); the latter uses `natbib`. The so-called *Unified Style Sheet for Linguistics*, as proposed by the CELXJ (*Committee of Editors of Linguistics Journals*), which slightly differs from the LSA style, is followed by the style file `unified.bst` (available at `https://codeberg.org/jspitz/univie-ling/raw/branch/master/3rdparty/unified.bst`). A `biblatex` style file for the unified style is available on CTAN in form of the biblatex-unified package or as part of the univie-ling bundle.

By default, the references have a hanging indentation of 3 em. This can be globally changed by altering the length **\reflistindent**. Doing `\setlength\reflistindent{1.5em}`, for instance, will shorten the indentation by half. Likewise, the length **\reflistitemsep** (6 pt by default) and **\reflistparsep** (ca. 4 pt by default) can be adjusted to alter the vertical separation between items (`\itemsep`) and paragraphs (`\parsep`) of reference entries.

Notice that within the reference list, "French spacing" is in effect — that is, spaces after periods are no wider than normal spaces. Thus you do not have to do anything special to avoid excessive space after people's initials.

10 Semantic markup

The macro **\sentence** displays an italicized sentence (it is a combination of `flushleft` and `itshape`). If you type

```
\sentence{This is a sentence.}
```

you get:

This is a sentence.

The font shape can be modified by redefining the following macro:

```
\newcommand*\sentencefs{\itshape}
```

The following macros provide further markup options common in linguistics:

- **\lexp**{word} is used to mark word forms (italic by default, as in *word*)
- **\lcon**{concept} is used to mark concepts (small caps by default, as in CONCEPT)
- **\lmean**{meaning} is used to mark meaning (single quotes by default, as in 'meaning')

Note that for **\lmean**, covington checks at document begin whether the csquotes [6] package is loaded. If so, it uses its language-sensitive **\enquote*** macro for quoting. If not, a fallback quotation (using English single quotation marks) is used. The usage of csquotes is highly recommended!

Here are the definitions of the macros. They can be redefined via **\renewcommand**:

```
\providecommand*\lexp[1]{\textit{#1}}
\providecommand*\lcon[1]{\textsc{#1}}
\providecommand*\lmean[1]{\covenquote{#1}}
```

11 Big curly brackets (disjunctions)

Last of all, the two-argument macro **\either** expresses alternatives within a sentence or PS-rule:

`the \either{big}{large} dog` = the $\left\{ \begin{array}{c} \text{big} \\ \text{large} \end{array} \right\}$ dog

`\psr{A}{B~\either{C}{D}~E}` $= A \longrightarrow B \left\{ \begin{array}{c} C \\ D \end{array} \right\} E$

That's all there is for now. Suggestions for improving covington are welcome, and bug reports are actively solicited (via `https://github.com/jspitz/covington`). Please note, however, that this is free software, and the authors make no commitment to do any further work on it.

12 Release history

2.17 (2026 November 12)

- Fix end of **subexamples** with activated tagging.

2.16 (2025 December 30)

- Add **subnumberrefsep** option to **subexamples** environment (see sec. 3.4).
- Fix reference to **subexample** that is nested in an enumeration.

2.15 (2025 August 13)

- Enhance **reflist** documentation.
- Fix definition of covexercise if no chapters exist.

2.14 (2023 December 11)

- Add **\ownexcounterprep** macro to add a prefix (e.g., the chapter) to the (own-counter) example counter. See sec. 3.7.
- Document how to (not) number examples per chapter.

2.13 (2023 December 2)

- Fix example numbering with chapters.

2.12 (2023 June 25)

- Fix issue with redefinitions of **\glosslineone**/**\glosslinetwo**/**\glosslinethree**, which could get lost along the text.

2.11 (2023 June 17)

- Add support for judgment markers in examples and glosses.
- Add support for customization of spacing between gloss lines.
- Add support for gloss comment lines via optional arguments.
- Add support for translation lines on the right side of glosses (via new **tlr*** option).

2.10 (2023 June 2)

- Use LaTeX 2$_\varepsilon$'s own keyval mechanism rather than xkeyval.
- Introduce a better interface to customize the appearance of examples. Now everything can be easily adjusted globally and locally via key-val options. This entails the following changes:
  - Allow to globally set example options via the new macro **\setexampleoptions**.
  - Add way to customize example and subexample number format (see sec. 3.4).
  - Add new example options: **fs**, **fsno**, **fspreamble**, **fspostamble**, **leftmargin**, **addnumbersep**, **addsubnumbersep**, **numberformat**, **subnumberformat**, as well as **fnnumberformat** (see sec. 3.2).
- Add new gloss options: **fspreamble**, **fspostamble**, **fstl**, and **enquotetl**. This allows for full customization of glosses via options, both locally and globally (see sec. 4).
- Package options now all have a key-value format as well.
- Introduce new package choice-option **fnexamplecounter** that supersedes the two boolean, mutually exclusive options **ownfncounter** and **ownfncounter*** (which are still supported for backwards compatibility reasons).
- Major documentation revisions.

2.9 (2023 May 26)

- Add **postamble** and **postamble*** options to **example** and **subexamples** environment (sec. 3.3 and 3.4).
- Add **\expreamble**, **\subexpreamble**, **\expostamble** and **\subexpostamble** macros (sec. 3.3 and 3.4).
- Some documentation fixes.

2.8 (2022 August 30)

- Add **ownfncounter** and **ownfncounter*** options to allow for separate counting of examples in footnotes. The starred option resets the counter at each footnote. See sec. 1.
- Add **preamble** option to **example** environment. See sec. 2.

2.7 (2021 September 1)

- Add **postamble** gloss macro option for arbitrary text that is appended to the free translation line (after closing quotation marks). See sec. 4.1.
- Add **\glosslinepostamble** command to markup this text.

2.6 (2021 May 19)

- New length **\exampleind** to adjust (increase/decrease) the indentation (left margin) of examples. See sec. 3.2 for details.

2.5 (2021 March 21)

- Fix metrics of stacked diacritics with XeTeX/LuaTeX.

2.4 (2020 January 2)

- Fix definition of covexercise theorem when no subsection counter is defined.

2.3 (2019 June 21)

- Add preamble option to subexamples environment. See sec. 3.4.
- Allow to use covington together with the drs package.
- Documentation fixes and restructuring.

2.2 (2019 June 4)

- Add new option **owncounter** that makes covington use an own counter for examples (rather than the equation counter).
- Add starred **\exampleno*** command that outputs the current example number value without stepping it. See sec. 3.1.
- Add macros **\covexamplefs** and **\covexamplenofs** for global setting of example text markup.

2.1 (2019 May 12)

- Add new option **noglossbreaks** that tries to prevent page breaks within glosses.
- Add **\glosspreamble** command and **preamble** gloss macro option for arbitrary text preceding glosses.

2.0 (2018 December 10)

- Add new gloss macros (**\digloss** and **\trigloss**) for a more convenient, flexible and robust gloss insertion. See sec. 4.1.
- Add **\glot** command for customizable gloss translation line, together with customization possibilities. See sec. 4.2.
- Add possibility to customize **\sentence** font setting. See sec. 10.
- Add **\lexp**, **\lcon** and **\lmean** markup macros. See sec. 10.

1.8 (2018 December 7)

- Fix font markup of second gloss line (do not force rm).
- Add possibility to customize gloss line font setting. See sec. 3.2.
- Add possibility to customize example number display. See sec. 3.2.

1.7 (2018 September 8)

- Fix alignment in **subexamples**.
- Improve manual.

1.6 (2018 September 7)

- Introduce new environment **subexamples** (see sec. 3.4).
- Introduce new command **\pxref** (see sec. 3.6).

1.5 (2018 August 24)

- Introduce new option **keeplayout** which allows to opt-out the layout presettings covington does (`\raggedbottom`, `\textfloatsep`).

1.4 (2017 May 23)

- Introduce a new macro **\twodias** that supersedes the rather odd **\twoacc** (which is kept for backwards compatibility). See sec. 2 for details.
- Introduce macro **\SetDiaOffset** for more convenient setting of vertical distance in stacked diacritics. See sec. 2 for details.
- LaTeX 2.09 is no longer officially supported (it might continue to work, but is not tested).

1.3 (2017 April 5)

- Gloss variants **\xgll** and **\xglll** that work inside macros (such as footnotes) but require explicit gloss line end markers (**\xgle**). See sec. 4 for details.
- New lengths **\reflistindent**, **\reflistparsep** and **\reflistitemsep** to globally adjust the indentation or vertical space, respectively, of reflist items. See sec. 9 for details.

1.2 (2016 August 26)

- New length **\examplenumbersep** to adjust (increase) the horizontal space between example number and example text. See sec. 3.2 for details.
- Add some more info about bibliography generation.

1.1a (2016 July 7)

- Fix encoding problem in documentation and some typos. No change in functionality.

1.1 (2016 July 6)

- The package now uses NFSS font commands if available (fallback for LaTeX 2.09 is still provided).
- Work around clash with classes/packages that define their own **example** and **examples** environments (most notably the `beamer` class) as well as **execise** environments. The covington package no longer blindly attempts to define these environments. By default, it does not define them if they are already defined (`covington's` own environments, however, are still available via aliases). By means of a new package option, a redefinition can also be forced. See sec. 3.2 and 3.3 for details.
- New length **\twoaccsep** allows for the adjustment of the distance between stacked accents (see sec. 2).
- Update manual.

- New maintainer: J. Spitzmüller.
- License has been changed to LPPL (in agreement with M. Covington)
- Introduce version numbers. Arbitrarily, we start with 1.1.

2014 May 16

- Patches by Robin Fairbairns:
  - Setting of `\textfloatsep` uses `\setlength` rather than `\renewcommand`
  - Style file converted to un*x line endings

2001 March 27

- It is no longer necessary to type `\it` to get proper italic type in feature structures.
- Instructions have been rewritten with LaTeX 2_ε users in mind.

Older versions

- Multiple accents on a single letter (e. g., $\acute{\tilde{a}}$) are supported.
- This package is now called covington (with the o) and is compatible with LaTeX 2_ε and NFSS as well as LaTeX 2.09.
- The vertical placement of labeled feature structures has been changed so that the category labels line up regardless of the size of the structures.

References

[1] Bickel, Balthasar, Bernard Comrie, and Martin Haspelmath: *The Leipzig glossing rules: Conventions for interlinear morpheme by morpheme glosses*. Revised version of February 2008. Department of Linguistics, Max Plank Institute for Evolutionary Anthropology. `http://www.eva.mpg.de/lingua/resources/glossing-rules.php`.

[2] Carlisle, David: *The enumerate package*. July 23, 2015. `https://www.ctan.org/pkg/enumerate`.

[3] Dimitriadis, Alexis: *drs – Typeset Discourse Representation Structures (DRS)*. June 10, 2010. `https://ctan.org/pkg/drs`.

[4] Isambert, Paul: *sdrt – Macros for Segmented Discourse Representation Theory*. May 13, 2007. `https://ctan.org/pkg/sdrt`.

[5] Kamp, Hans: A Theory of Truth and Semantic Representation. In Jeroen A. G. Groenendijk, Theo M. V. Janssen, and Martin J. B. Stokhof (eds.): Formal Methods in the Study of Language. Amsterdam: Mathematics Center, 1981, 277–322.

[6] Lehman, Philipp and Joseph Wright: *csquotes: Context sensitive quotation facilities*. April 4, 2018. https://www.ctan.org/pkg/csquotes.

[7] Pakin, Scott: The Comprehensive LaTeX Symbol List. November 30, 2015. https://www.ctan.org/pkg/comprehensive.

[8] Weber, Nathalie: *leipzig: Typeset and index linguistic gloss abbreviations*. June 16, 2017. https://ctan.org/pkg/leipzig.