

The propositions package

Cian Dorr
ciandorr@gmail.com

v0.9: 2026/08/02

Abstract

The **propositions** package provides a key-value driven system for labelling propositions, theses, and premises in academic papers. Items may be given names like ‘(P)’ or ‘Physicalism’, or auto-numbered using different counters; all carry robust cross-references with configurable formatting. The package integrates with **amsmath**, **hyperref**, and **cleveref**.

Contents

1	Introduction	2
2	History	2
3	Basic usage	2
4	Advanced usage	3
5	Basic environments and commands	5
6	Keys	6
6.1	Item-level keys	6
6.2	Keys for changing the list geometry	9
6.3	Other environment-level keys	11
6.4	Global key	13
7	Cross-referencing commands	13
7.1	How cross-referencing works	16
8	Defining and modifying styles	16
9	Built-in styles	17
10	Compatibility	24
11	Known issues	25
12	Release notes	25

1 Introduction

In some academic disciplines (such as philosophy), it is common to have displayed propositions (examples, theses, premises, . . .) with various kinds of labels. A thesis might be referred to as ‘(P)’ or ‘Physicalism’; the premises of an argument might be numbered as ‘P1’, ‘P2’, ‘P3’, . . . ; or examples might be numbered consecutively over the course of a whole article. Standard L^AT_EX environments like `enumerate` can handle some of these cases, but cross-referencing is awkward: `\ref` produces a bare number or letter, and the author must manually add parentheses or other formatting at every point of reference. The standard `description` environment, meanwhile, does not allow cross-referencing at all.

The `propositions` package solves this by attaching formatting information to each label. With an appropriate choice of style, `\item[P]` (inside `prop`^{P.5}) is displayed as “(P)” and `\ref` automatically produces “(P)” as well—complete with parentheses and hyperlink. The full key-value interface supports named items, numbered items, custom counters, glosses, shorthands, per-item format changes, changes to the list geometry, and styles.

The `\ptag`^{P.6} command (which requires `amsmath`) extends this to displayed math environments: an equation can be tagged with a proposition label instead of (or using) its equation number, and `\ref` to an equation will pick up its name or number with specified formatting.

2 History

I wrote the ancestor to this package in the 90s while finishing my Ph.D. thesis, and used it in my own work but never documented or shared it. This new version is a thorough re-implementation in L^AT_EX3, written in 2026 with very extensive help from Claude Code. I hope others will find it as useful as I have.

3 Basic usage

Load with `\usepackage{propositions}` (see subsection 6.4 below for valid package options).

The `prop` environment generates a list of propositions, each introduced by `\item`. `\item` with an optional argument gives a `description`-like label:

```
\begin{prop}
  \item[Physicalism] Everything is physical. \label{phys}
  \item[Idealism] Everything is mental. \label{ideal}
\end{prop}
```

Physicalism Everything is physical.

Idealism Everything is mental.

Unlike the standard `description` environment, one can refer back to these propositions using the standard `\ref` command (or with new cross-referencing commands described in section 7):

```
\ref{phys} is more plausible than \ref{ideal}.
```

Physicalism is more plausible than **Idealism**.

With no optional argument, `\item` will by default generate numbered items similar to `enumerate`, but with numbering that persists across the document:

```
\begin{prop}
  \item Every atom is physical. \label{atoms}
\end{prop}
\ref{phys} follows from the conjunction of \ref{atoms} and
\begin{prop}
  \item Everything is an atom. \label{atomism}
\end{prop}
```

(1) Every atom is physical.

Physicalism follows from the conjunction of (1) and

(2) Everything is an atom.

As with `enumerate`, the counter and formatting depend on the nesting level:

```
\begin{prop}
  \item \label{dual}
  \begin{prop}
    \item Some things are physical. \label{some}
    \item Some things are not physical. \label{notall}
  \end{prop}
\end{prop}
Without \ref{some}, \ref{dual} would be consistent
with \ref{ideal}.
```

(3) a. Some things are physical.

b. Some things are not physical.

Without (3a), (3) would be consistent with **Idealism**.

4 Advanced usage

The format of the proposition labels, and of subsequent references, are both configurable using a key=value syntax (see [section 6](#) for the possible keys):

```
\begin{prop}
  \item[No Overlap,
        align=flush,
        display format=\textsc{#1},
        ref format=\textit{#1}]
    Nothing mental is physical. \label{incomp}
\end{prop}
Is \ref{incomp} consistent with \ref{phys}?
```

No OVERLAP Nothing mental is physical.
 Is *No Overlap* consistent with **Physicalism**?

The prop environment can also take an optional argument with a list of keys:

```
\begin{prop}[leftmargin=5em, format=#1]
  \item Every mental thing is physical.
\end{prop}
```

[4] Every mental thing is physical.

Preset styles can be declared and used in place of a set of key-value pairs. The package loads with a range of predefined styles (see section 9).

```
\begin{prop}
  \item[Nihilism, style=thesis] There is nothing. \label{nihilism}
\end{prop}
Does \ref{nihilism} imply \ref{phys}, \ref{dual}, or both? Discuss.
```

NIHILISM There is nothing.
 Does *Nihilism* imply **Physicalism**, (3), or both? Discuss.

Additional cross-referencing commands (see section 7) include `\nref`^{P.13} (strips formatting from reference) and `\lastref`^{P.15} (refers to the most recent proposition, even if it lacked a label):

```
\begin{prop}
  \item[Mentality] \label{mental} Everything is mental.
  \item[\nref{mental}*] Many things are mental.
\end{prop}
One may think \lastref\ more reasonable than \ref{mental}.
```

Mentality Everything is mental.
Mentality* Many things are mental.
 One may think **Mentality*** more reasonable than **Mentality**.

When the package is loaded with `\usepackage[equations]{propositions}`, a first-level `\item` within `prop`^{P.5} will use the counter as equations. (This looks better with the `leqno` option to `\documentclass`.)

```
\begin{equation}
  \exists x (\text{Mental}(x) \wedge \text{Physical}(x)) \label{overlap}
\end{equation}
\ref{overlap} can be stated in English as \ref{overlap2}:
\begin{prop}
  \item Some things are both mental and physical. \label{overlap2}
\end{prop}
```

$$(5) \quad \exists x(\text{Mental}(x) \wedge \text{Physical}(x))$$

(5) can be stated in English as (6):

(6) Some things are both mental and physical.

The `\ptag`^{→ P. 6} command (requires `amsmath`) is a replacement for the standard `\tag` command that behaves just like an `\item` in a `prop` environment.

```
\begin{equation}
  \ptag[Monism, \format=\textsc{#1}] \label{mon}
  \exists x \forall y (y = x)
\end{equation}
Is \ref{mon} compatible with \ref{dual}?
```

Monism $\exists x \forall y (y = x)$

Is **Monism** compatible with (3)?

5 Basic environments and commands

```
\begin{prop}[<keys>]
  <environment content>
\end{prop}
```

Creates a displayed list of propositions. It is a standard L^AT_EX list, so by default its formatting will depend on the standard length parameters like `\itemsep` and `\topsep`, although these can be overridden by setting keys.

Within `prop`, `\item` creates the propositions (see below).

The optional `<keys>` argument can contain a list of keys, which will affect only the given environment (not any other `prop` environments that may be nested within it).

```
\begin{inlineprop}[<keys>]
  <environment content>
\end{inlineprop}
```

Like `prop`, but does not create a list. Allows `\item` to be used outside list environments, e.g. for generating numbers at the beginning of paragraphs. Steps the `prop` counter and increments the nesting level. Accepts the same optional `<keys>` as `prop`.

Within `prop` and `inlineprop`, `\item` is redefined to act as the proposition-item command. Its optional argument is a comma-separated list of `<key>=<value>` pairs (see section 6). A bare string without `=` is treated as a proposition name.

When used without an optional argument (or without setting `name`^{→ P. 7}, `counter`^{→ P. 7}, or `style`^{→ P. 7}), the style is determined by the `nameless style`^{→ P. 12} key (section 6). By default this is `numbered`, which dispatches by nesting depth to `levelone`, `leveltwo`, ... via `\proplevelchoice`^{→ P. 6}.

`\ptag[⟨keys⟩]`

(Available only when `amsmath` is loaded.) Works inside displayed math environments like `equation` and `align`. Accepts the same keys as `\item` within `prop`, except that `align`^{→P.9} has no effect (since positioning is controlled by the tag placement system).

Any display math environment (`equation`, `align`, etc.) used inside `prop`^{→P.5} or `inlineprop`^{→P.5} automatically increments the nesting level for its duration, so that `\ptag` inside such an environment behaves as if it were one level deeper.

`\proptions{⟨keys⟩}`

Sets default keys (see section 6), which take effect for all subsequent uses of `prop`^{→P.5}, `inlineprop`^{→P.5}, `\item`, and `\ptag` (within the current scope).

`\proplevelchoice{⟨item1, item2, ...⟩}`

Picks the entry from the comma-separated list depending on the current nesting depth (1-indexed; last element will be used if the list is too short; empty entries are skipped). At depth 0 (outside any `prop`^{→P.5} environment), returns the first entry.

This macro can be used, for example, to set different list dimensions for different nesting levels (see section 6.2) or to create styles that produce different results depending on the nesting depth where they are used (see section 8).

Further commands are described below, especially in section 7 (cross-referencing) and section 8 (defining styles).

6 Keys

6.1 Item-level keys

The keys listed in this subsection can be used in the following places:

- In the optional argument of `\item` or `\ptag`: the key applies to that item only.
- In the optional argument of `prop`^{→P.5} or `inlineprop`^{→P.5}: the key applies to every top-level `\item` within that environment, unless overridden by the `\item`'s own optional argument. (If another `prop`^{→P.5} or `inlineprop`^{→P.5} environment is used inside this one, the key will not apply to its `\items`.)
- In the argument of `\proptions`: the key will apply to every subsequent `\item` or `\ptag` (in the current group), unless overridden by that `\item`'s optional argument, or that of its parent environment.
- In the optional argument of `\usepackage{propositions}`: equivalent to `\proptions`, except that keys that require the `#` character cannot be set here, (due to a limitation in how L^AT_EX processes options)
- With `\SetPropStyle`^{→P.16} or `\DeclareNumberedStyle`^{→P.17}, to add the key to a given style.

`style=<style>` (no default)

A style, equivalent to a preset collection of keys: see section 8 for details.

`name=<text>` (no default)

The proposition's name. A bare string (without =) is equivalent to `name=<text>`. (To be precise: if the optional argument contains no = at all, the whole of it is the name; otherwise, it is read as a key list, and any entry of it without an = is the name, so a bare name can be combined with other keys, as in `\item[No Overlap, align=flush]`. In that case a name containing a comma must be braced—`\item[{Alpha, Beta}, align=flush]`—and a name containing an = needs the explicit `name={<text>}` form.)

`counter=<name>` (no default)

Counter to use. The counter is automatically stepped, and the item's `name` is set to `\the<name>` (though this can be overridden by `counter format` or `name`).

Any counter can be used with `counter format`. Special counters `numpropii`, `numpropiii`, `numpropiv`, `numpropv` are provided, whose values are automatically reset each time an `\item` is processed at a lower nesting level. (There is also a counter `numpropi` which does not automatically reset.)

`counter format=<template>` (no default)

How to display the counter value. Use #1 for the counter name, e.g. `counter format=\roman{#1}`. When both this key and `counter` are set, this is used instead of `\the<counter>` for generating the item's `name`.

`display format=<template>` (no default)

Format for displaying the name or counter value in the proposition's label. Use #1 for the argument, e.g. `display format=\textbf{#1}`. Does not affect cross-references.

`ref format=<template>` (no default)

Format for subsequent cross-references to this proposition. Use #1 for the argument, e.g. `ref format=\{(#1)\}`. Does not affect the display.

`format=<template>` (no default)

Shorthand for setting both `display format` and `ref format`.

`shorthand=<text>` (no default)

An abbreviation displayed after the name. If present, the shorthand becomes the reference text: `\ref` produces the shorthand (formatted with `ref format`) rather than the full name.

```
\begin{prop}
  \item[Global Physical Supervenience, shorthand=GPS] \label{global}
  Every fact is entailed by some fact about
  the physical world.
\end{prop}
There are several interesting arguments for \ref{global}.
```

Global Physical Supervenience [GPS] Every fact is entailed by some fact about the physical world.

There are several interesting arguments for **GPS**.

shorthand *format*=*<template>* (initially ~[#1])

Format for displaying the shorthand in the label.

gloss=*<text>* (no default)

A parenthetical gloss displayed after the name. Does not affect cross-references.

```
\begin{prop}
  \item[Humean Supervenience, gloss=after David Lewis]
    Every fact is entailed by some fact about the
    fundamental properties and spatiotemporal relations
    of points.
\end{prop}
```

Humean Supervenience (after David Lewis) Every fact is entailed by some fact about the fundamental properties and spatiotemporal relations of points.

gloss *format*=*<template>* (initially ~(#1))

Format for displaying the gloss in the label.

label *format*=*<template>* (no default)

Format applied to the *entire* assembled label, i.e. the name (after **display format**^{→ P. 7}), shorthand, and gloss together. Use #1 for the whole assembled text. For example, **label format**=#1: appends a colon after the complete label.

ref=*<text>* (no default)

Explicitly set the reference text, overriding what would be derived from **name**^{→ P. 7}, **counter**^{→ P. 7}, or **shorthand**^{→ P. 7}.

label=*<label>* (no default)

Equivalent to a trailing **\label{<label>}**.

reset=*<boolean>* (initially **true**)

When **true** (the default), processing an **\item** resets the sub-level counter one nesting depth below the current level (e.g. **numpropii** at level 1). **reset=false** suppresses this behaviour, so that sub-item numbering continues from where it left off across consecutive parent items.

crefname=*<type>* (no default)

When **cleveref** is loaded, assigns an arbitrary reference type to this proposition. For example, **crefname=lemma** causes **\cref** to use the names defined by **\crefname{lemma}{...}{...}** instead of the default (private) **prop**^{→ P. 5} type. The *<type>* must be known to **cleveref**; new types can be declared with **\crefname**.

Four of the list-dimension keys documented in the next section—`labelwidth`^{P.10}, `labelsep`^{P.10}, `itemindent`^{P.10} and `labelindent`^{P.10}—may also be given to an individual `\item`, where they reposition the label of that item alone, overriding the value in force for the rest of the list. (The other dimension keys have no effect at item level.)

6.2 Keys for changing the list geometry

The keys in this section can be used in the following places:

- In the optional argument of `prop`^{P.5}: the key applies to that environment only, not to any other `prop`^{P.5} or `inlineprop`^{P.5} environments that may be nested within it.
- In the argument of `\proppositions`^{P.6}: the key will apply to every subsequent `prop`^{P.5} environment (in the current group), unless overridden by that environment’s optional argument.
- In the optional argument of `\usepackage{propositions}`: equivalent to `\proppositions`^{P.6}.
- With `\SetPropStyle`^{P.16} or `\DeclareNumberedStyle`^{P.17}, to add the key to a given style.

Five of the keys can also be used in the optional argument of `\item` to affect the geometry of that specific proposition (`align`, `labelwidth`^{P.10}, `labelsep`^{P.10}, `itemindent`^{P.10}, and `labelindent`^{P.10}). The other keys will have no effect in an `\item` (or a `\ptag`^{P.6}).

`align=left|right|center|flush|nextline|flush-nextline` (initially `left`)

How the label should be positioned. `left` is the standard left-aligned label, its offset controlled by `labelwidth`^{P.10} and `labelsep`^{P.10}; `right` is right-aligned within the label box, like `enumerate`; `center` is centred within it; `flush` aligns the label with the left margin of the item text; `nextline` puts the label on a line of its own, and `flush-nextline` does both. Has no effect inside `inlineprop`^{P.5} or `\ptag`^{P.6}.

```
\begin{prop}
  \item[L, align=left] Left aligned label.
  \item[C, align=center] Center aligned label.
  \item[R, align=right] Right aligned label.
  \item[Long label, align=center] As in standard \LaTeX\ lists,
    longer labels expand to fill the label box and then push
    the following text along to make room for themselves.
  \item[Flush label, align=flush] Flush aligned label.
  \item[Sometimes a long label will deserve its own line,
    align=nextline] Nextline aligned label.
  \item[Another rather long label,
    align=flush-nextline] Flush-nextline aligned label.
\end{prop}
```

L Left aligned label.
C Center aligned label.
R Right aligned label.
Long label As in standard L^AT_EX lists, longer labels expand to fill the label box and then push the following text along to make room for themselves.
Flush label Flush aligned label.
Sometimes a long label will deserve its own line
Nextline aligned label.
Another rather long label
Flush-nextline aligned label.

```
topsep=<length>
partopsep=<length>
itemsep=<length>
parsep=<length>
leftmargin=<length>
rightmargin=<length>
labelwidth=<length>
labelsep=<length>
itemindent=<length>
listparindent=<length>
labelindent=<length>
```

Override the standard L^AT_EX list dimensions. Accept the same values as `\setlength`, including rubber lengths (e.g. `itemsep=4pt plus 2pt`).

`labelindent` is not a standard L^AT_EX list dimension. It positions the left edge of the label box at `<length>` from the enclosing margin. This introduces a redundancy: any one of `labelindent`, `\labelwidth`, `\leftmargin`, `\labelsep`, or `\itemindent` can be calculated from the four others, via the expression

$$\text{labelindent} + \text{labelwidth} + \text{itemsep} = \text{leftmargin} + \text{itemindent}$$

Whichever of these is not explicitly set will be computed from the ones that are explicitly set (or from the four most recently explicitly set) in such a way as to guarantee this identity. Each dimension key also accepts the value `*`, which means *behave as if the key had not been set*: the dimension takes the document class's default, unless its value must shift to preserve the above identity. This behaviour is identical to that of package `enumitem`.

A dimension key can be set to any macro that expands to a dimension. For example, one can set different dimensions for different nesting levels by using `\proplevelchoice`^{→ P. 6}:

```
\proppoptions{leftmargin = \proplevelchoice{2.5em, 0em, *}}
```

(Note that without the `*`, the last entry `0em` would apply at every level below the second.)

`tightspacing` (no value)

Sets all vertical spacing to the compact defaults that the standard document classes use for level-three lists (`\topsep` and `\itemsep` to 2pt with stretch/shrink, `\parsep` to 0pt, `\partopsep` to 1pt).

`nosep` (no value)

Sets `\topsep`, `\itemsep`, and `\parsep` all to zero.

`\propformlabel`

Expands to the formatted label of the (approximately) `items`-th item in this environment, computed at `\begin{prop}` *before* the list dimensions are applied, so it can be used directly in dimension expressions. It keeps that value for the whole environment: a dimension key is read again for each item's label area, and a prediction that changed as the list went along would size each item's label box to its predecessor's label. Typical use:

```
\begin{prop}[items=20,
  leftmargin=\widthof{\propformlabel}+0.5em]
```

sizes the left margin to accommodate labels up to the 20th item.

`\propwidestlabel`

The same as `\propformlabel`, but with every digit replaced by the digit that makes the label widest (an 8 in many fonts). Using it in place of `\propformlabel` in a dimension expression makes the dimension depend on how many digits the label has rather than on which digits they are, so that a run of numbered environments keeps a steady margin instead of shifting at every change of number. (The substitution is used for measurement only; the labels themselves are untouched.) (Used by the built-in `fitmargin` and `outerfit` styles.)

`items=<n>` (initially 1)

Indicates that approximately $\langle n \rangle$ items are expected in this environment. The only effect of this key is in conjunction with `\propformlabel`, to compute a preview label wide enough for the $\langle n \rangle$ th expected item, so that dimension expressions such as `leftmargin=\widthof{\propformlabel}+0.5em` reserve enough space for the widest label. Has no effect on actual item processing or counter stepping.

`continue=<boolean>` (initially true)

When a `\propP.5` environment with `continue=true` is immediately followed by another such environment, (with no intervening paragraph text), the normal `\topsep`-based inter-list space is replaced with `\itemsep + \parsep`, giving the visual appearance of a single continuous list.

6.3 Other environment-level keys

The keys in this section have effect in both the `\propP.5` and `\inlinepropP.5` environments. They can also be used with `\propoptionsP.6`, `\usepackage`, and `\SetPropStyleP.16`, just like the keys in the previous subsection.

`named style=<style>` (initially proposition)

The style when an `\item` or `\ptag`^{P.6} has a value for `name`^{P.7} but no explicit `style`^{P.7}.

`named ptag style=<style>` (initially empty)

If set, overrides `named style` for `\ptag`^{P.6} items only.

`nameless style=<style>` (initially numbered)

The style used when `\item` or `\ptag`^{P.6} has no value for `name`^{P.7} and no explicit `style`^{P.7}. A `counter`^{P.7} does not affect the choice: it says how the item is numbered, and this is the style meant to look right around a number.

`nameless ptag style=<style>` (initially empty)

If set, overrides `nameless style` for `\ptag`^{P.6} items only.

The choice between them turns on `name`^{P.7} alone, read from the whole key list—`\propoptions`^{P.6}, then the environment argument, then the item's own argument. An item counts as named if a name reaches it from any of those, so `\begin{prop}[name=foo]\item bar` gives the same result as `\begin{prop}\item[name=foo] bar`. An explicit `style`^{P.7} overrides the choice wherever it is set.

`wrapper begin=<code>`

`wrapper end=<code>`

Insert arbitrary `<code>` immediately before the beginning and after the end of the environment, so that the whole list can (for example) be wrapped in another environment.

Any value that contains a comma (such as a comma-separated list of options to an environment) must be enclosed in braces, so that the comma is not read as a key separator:

```
\begin{prop}[
  wrapper begin = {\begin{tcolorbox}[
    colframe = red, colback = white]},
  wrapper end   = {\end{tcolorbox}}]
\item[Red] A proposition in a red frame.
\item And a second one, in the same frame.
\end{prop}
```

Red A proposition in a red frame.

(7) And a second one, in the same frame.

The wrapping environment must be one that can be split into separate begin and end code—that is, it must not read its body as a macro argument via `\collect@body` or similar.

The following command is provided for use with `wrapper begin`:

`\propoperativeleftmargin`

A read-only length giving the `leftmargin`^{→P.10} a `prop`^{→P.5} environment (with no optional argument) would use, given the current defaults set by `\proptions`^{→P.6} and the current nesting level. It is recomputed at every `\begin{prop}`, so it is valid inside a wrapper (before the environment’s own `leftmargin`^{→P.10} has taken effect). For example, the built-in `framed` style uses this command in the value of `wrapper begin` to make the margins of framed environments match those of regular environments.

6.4 Global key

The following key can only be set in the optional argument of `\usepackage`, or in the preamble with `\proptions`^{→P.6}.

`equations` (no value, **global only**)

Installs hooks so that the L^AT_EX and `amstex` displayed equation environments (such as `equation` and `align`) use the same formatting as the special `equation` prop style. The `equation` style’s `display format`^{→P.7} and `ref format`^{→P.7} keys are used for generating equation numbers and cross-references to them.

Redefining the `equation` style, e.g. with

```
\SetPropStyle{equation}{format = [#1]}
```

will automatically update the hooks.

The `equations` key also sets `nameless style=eqnum`, so that top-level `\items` with no name will also use the `equation` style, while lower-level `\items` will use other kinds of numbering: for details, see section 9 below.

7 Cross-referencing commands

Labels placed after `\item` items (within `prop`^{→P.5}) work with the standard `\label/\ref` mechanism. The key difference from ordinary L^AT_EX references is that `\ref` produces *formatted* output: for example, `\textbf` might be applied to the name, or the number might be wrapped in parentheses. The formatting is controlled by the `format` key (or separately by `display format` and `ref format`).

`\Ref{\label}`

`\Ref*{\label}`

Titlecase variants of `\ref` and `\ref*`: uppercases the first letter of the formatted output. (For example, if `\ref{thesis}` produces ‘the Identity Theory’, then `\Ref{thesis}` produces ‘The Identity Theory’.) The starred form suppresses the hyperlink. Note: `\Ref` only affects references produced by this package (which are stored via `\propapply`^{→P.16}); for other references, it behaves like `\ref`.

`\nref{\label}`

`\nref*{\label}`

`\Nref{\label}`

`\Nref*{\label}`

“Naked ref.” Outputs the bare reference content with all formatting stripped. If `\ref{premise}` produces ‘(P1)’, then `\nref{premise}` produces ‘P1’. The

starred form suppresses the hyperlink. `\Nref` is the titlecase variant: it up-percases the first letter of the bare content.

`\nref` can be useful in the argument of `\item`, when the the name of one proposition should depend on that of another:

```
\SetPropStyle{proposition}{format=(\textbf{#1})}
\begin{prop}
  \item[Phys] Everything is physical. \label{phys2}
  \item[\nref{phys2}*] Almost everything is physical.
  \label{newphys2}
  \item[\ref{phys2}*] This one has two sets of parentheses,
  which is probably not desired! Note that
  the previous proposition \ref{newphys2} avoided this by using
  |\nref| in the optional argument
  of |\item|.
\end{prop}
```

(**Phys**) Everything is physical.
 (**Phys***) Almost everything is physical.
 ((**Phys**)*) This one has two sets of parentheses, which is probably not desired!
 Note that the previous proposition (**Phys***) avoided this by using `\nref` in
 the optional argument of `\item`.

Warning: documents where the name of one item includes a reference to that of another, and there are further references to that item, will require multiple $\LaTeX$ runs to resolve all references. To save time, it is better to avoid long chains of dependencies of this sort.

```
\oref[⟨prefix⟩][⟨suffix⟩]{⟨label⟩}
\oref*[⟨prefix⟩][⟨suffix⟩]{⟨label⟩}
\Oref[⟨prefix⟩][⟨suffix⟩]{⟨label⟩}
\Oref*[⟨prefix⟩][⟨suffix⟩]{⟨label⟩}
```

“Ref with options.” Extends `\ref` by injecting a prefix and/or suffix *inside* the formatting. With one optional argument, `⟨suffix⟩` is appended; with two, `⟨prefix⟩` is prepended and `⟨suffix⟩` appended. For instance, if `\ref{premise}` produces ‘(P1)’, then `\oref[*]{premise}` produces ‘(P1*)’ and `\oref[old~]{premise}` produces ‘(old~P1*)’.

`\Oref` is the titlecase variant; the starred forms suppress the hyperlink.

`\oref` can also be useful in the name of `\items`, if one wants the display format for the modified item to depend on that originally used

```
\begin{prop}
  \item[style=plain, name=\oref[$^\dag$]{phys2}]
  This will use boldface and parentheses because the
  original referenced item did.
\end{prop}
```

(**Phys[†]**) This will use boldface and parentheses because the original referenced item did.

Another handy use for `\oref` is in combination with `\nref` to refer to ranges:

The first two numbered examples in this document were `\oref[--\nref{atomism}]{atoms}`.

The first two numbered examples in this document were (1–2).

`\lastref[⟨prefix⟩]{⟨suffix⟩}`

`\Lastref[⟨prefix⟩]{⟨suffix⟩}`

Produces a reference (with no hyperlink) to the most recently processed `\item` or `\ptag→P.6`, even without a `\label`. Useful for back-references in running text. With one argument, `⟨suffix⟩` is appended; with two, `⟨prefix⟩` is also prepended. Use `\lastref{}` for a plain reference. `\Lastref` is the titlecase variant.

`\nlastref`

`\nLastref`

Like `\lastref{}`, but returns the bare content without formatting. `\nLastref` is the titlecase variant.

`\parentref[⟨prefix⟩]{⟨suffix⟩}`

`\Parentref[⟨prefix⟩]{⟨suffix⟩}`

Inside a nested `prop→P.5` (or `inlineprop→P.5`), produces a formatted reference to the most recent item of the enclosing level. Same argument convention as `\lastref`. `\Parentref` is the titlecase variant.

`\nparentref`

`\nParentref`

Like `\parentref{}` but returns the bare content without formatting. Takes no arguments; simply output any desired suffix directly afterwards. `\nParentref` is the titlecase variant.

`\parentref` and `\nparentref` are useful for making subitems whose names derive from their parent's:

```
\begin{prop}
  \item[P1, ref format=(#1)] \label{claim}
  \begin{prop}[counter format=\alph{#1},
    display format=\textit{#1.}, ref format=\parentref{#1}]
    \item \label{positive}
      Some things are physical.
    \item \label{negative}
      Some things are not physical.
  \end{prop}
\end{prop}
```

Of the two parts of `\ref{claim}`, `\ref{positive}` is far more controversial than `\ref{negative}`.

P1 *a.* Some things are physical.
 b. Some things are not physical.

Of the two parts of (P1), (P1a) is far more controversial than (P1b). Thus, we will mostly be considering part b.

Many of the built-in numbered styles use `\parentref` in their `ref format`^{→P.7}, to achieve this sort of composite effect.

7.1 How cross-referencing works

`\propapply{<template>}{<content>}`

Internally, each reference is stored in the `.aux` file as `\propapply{<template>}{<content>}`. The `<template>` contains formatting with the placeholder `\propfmtarg` where content appears. At reference time, `\propapply` evaluates the template with `\propfmtarg` bound to `<content>`. The `\oref`^{→P.14} and `\nref`^{→P.13} commands work by locally redefining `\propapply`. In normal use, you need not interact with `\propapply` directly.

`\propfmtarg`

Placeholder used inside templates; expands to the content argument of the enclosing `\propapply`.

8 Defining and modifying styles

Styles, equivalent to bundles of key-value settings, can be defined, and used freely in the optional arguments of `\item`, `\ptag`^{→P.6}, `prop`^{→P.5}, `inlineprop`^{→P.5}, and the argument of `\propoptions`^{→P.6}. Styles can freely be combined, and the definition of one style can reference another (in which case redefining the latter style will change the effect of the former style.)

`\SetPropStyle{<name>}{<keys>}`

Defines or modifies a prop style for use with the `style`^{→P.7} key. All item-level keys (subsection 6.1) are accepted, plus the following:

`style=<parent>` (no default)

Inherit from a parent style. When an item is created, the parent's settings are loaded first (recursively, if the parent itself has a parent), then this style's own keys are applied on top.

The argument may be a macro which is expanded when the item is created: for example, a style with `style=\{<style1>\},\{<style2>\},\{<style3>\}` will resolve to `\{<style1>\}, \{<style2>\}, \{<style2>\}` depending on the nesting depth. The built-in `numbered` and `eqnum` styles use this mechanism.

`macro=<command>` (no default)

A new user macro, equivalent to `\item[style=<name>]`. Any further keys given to the macro are passed to `\item`.

If the style `<name>` already exists, `\SetPropStyle` modifies or adds keys. For example, `\SetPropStyle{proposition}{align=flush}` changes the alignment of the built-in `proposition` style while preserving its other settings.

```
\SetPropStyle{angle}{
  labelindent = 0em,
  display format = \textbf{${\langle\!\!\rangle\!$#1$\rangle\!$},
```

```

    ref format      = $\langle\angle\$\#1\$\rangle$,
    macro           = \angitem
}
\begin{prop}
  \angitem[Angle thesis] Everything is angular.
\end{prop}
No further discussion of \lastref{} is needed.

```

$\langle$ Angle thesis $\rangle$ Everything is angular.
 No further discussion of $\langle$ Angle thesis $\rangle$ is needed.

\DeclareNumberedStyle{ $\langle$ name $\rangle$ }[$\langle$ keys $\rangle$]

Creates a new L^AT_EX counter named $\langle$ name $\rangle$ and a matching prop style with `counter= $\langle$ name $\rangle$` . All `\SetPropStyle`^{→P.16} keys are accepted, plus:

parent= $\langle$ counter $\rangle$ (no default)

A parent counter; the new counter resets when the parent steps (same mechanism as `\numberwithin`). A dedicated `prop` counter (stepped by each `prop`^{→P.5} and `inlineprop`^{→P.5}) is available for non-persistent numbering.

```

\DeclareNumberedStyle{P}
\begin{prop}
  \item[counter=P] First premise. \label{p1}
  \item[counter=P] Second premise. \label{p2}
\end{prop}
From \ref{p1} and \ref{p2}\ldots

```

(P1) First premise.
 (P2) Second premise.
 From (P1) and (P2)...

9 Built-in styles

The following prop styles are predefined. Each entry shows the defining code (using user-facing commands), and each group of styles ends with a live example. All styles can be modified with `\SetPropStyle`^{→P.16}; the definitions are reproduced here to facilitate modification.

Text styles

plain Unformatted text label.

```
\SetPropStyle{plain}{format = #1}
```

proposition The standard style assigned to named propositions. By default, uses boldface for both the label and references.

```
\SetPropStyle{proposition}{format = \textbf{#1}}
```

thesis Intended for named propositions with a longer name. The name begins at the text margin (flush alignment), and is set in small caps; references use plain font.

```
\SetPropStyle{thesis}{display format = \textsc{#1},
  ref format=#1, align = flush}
```

vignette Intended for things like example vignettes and comments. Italic name at the text margin, and italic references, separated from the text by a colon.

```
\SetPropStyle{vignette}{ref format = \textit{#1},
  align = flush, label format = \textit{#1:}}
```

bullet Bullet symbol varying by depth (like `\itemize`).

```
\SetPropStyle{bullet}{align = center,
  name = \proplevelchoice{\textbullet,
    {\normalfont\textendash}, \textasteriskcentered,
    \textperiodcentered},
  display format = #1}
```

```
\begin{prop}
  \item[One, style=plain] \label{bs:plain}
  A proposition in style |plain|, cited as \ref{bs:plain}.
  \item[Two, style=proposition] \label{bs:prop}
  A proposition in style |proposition|, cited as \ref{bs:prop}.
  \item[Three, style=thesis] \label{bs:thesis}
  A proposition in style |thesis|, cited as \ref{bs:thesis}.
  \item[Four, style=vignette] \label{bs:vignette}
  A proposition in style |vignette|, cited as \ref{bs:vignette}.
  \item[style=bullet] \label{bs:bullet}
  A proposition in style |bullet|, cited as \ref{bs:bullet}---though
  it is not often that one would want to cite an item in this style.
\end{prop}
```

One A proposition in style `plain`, cited as `One`.

Two A proposition in style `proposition`, cited as `Two`.

THREE A proposition in style `thesis`, cited as `Three`.

Four: A proposition in style `vignette`, cited as *Four*.

- A proposition in style `bullet`, cited as `(•)`—though it is not often that one would want to cite an item in this style.

Numbered styles

numbered The default nameless style. Selects one of the helper styles `levelone`–`levelfive`, depending on nesting depth.

```
\SetPropStyle{levelone}{counter = numpropi, format = (#1)}
\SetPropStyle{leveltwo}{counter = numpropii,
```

```

display format = #1., ref format = \parentref{#1}}
\SetPropStyle{levelthree}{counter = numpropiii,
display format = (#1), ref format = \parentref{.#1}}
\SetPropStyle{levelfour}{counter = numpropiv,
display format = #1., ref format = \parentref{#1}}
\SetPropStyle{levelfive}{counter = numpropv,
display format = (#1), ref format = \parentref{.#1}}
\SetPropStyle{numbered}{style = \proplevelchoice{
levelone, leveltwo, levelthree, levelfour, levelfive}}

```

equation Uses the equation counter. Note that this style has a special behavior when the `equations` option is active: changing its `display format`^{→P.7} and `ref format`^{→P.7} keys (or both, by setting `format`^{→P.7}) will also change the corresponding hooks used to create the tags for numbered equations make the `equation` environment and `amstex` environments like `gather`.

```

\SetPropStyle{equation}{counter = equation, format = (#1)}

```

eqnum Like `numbered`, but uses `equation` instead of `levelone` at the outer level. The `equations` package option sets default `nameless style = eqnum`. Since this manual uses this option, its outer-level nameless items use the `equation` style.

```

\SetPropStyle{eqnum}{style = \proplevelchoice{
equation, leveltwo, levelthree, levelfour, levelfive}}

```

```

\begin{prop}
\item \label{outer}
This is an outermost numbered item. Since the default style for
nameless items is |eqnum|, it uses the |equation| style.
\begin{prop}
\item \label{second}
This is a second level item, using |leveltwo|.
\begin{prop}
\item \label{third}
This is a third level item, using |levelthree|.
\begin{prop}
\item \label{fourth}
This is a fourth level item, using |levelfour|.
\begin{prop}
\item \label{fifth}
This is a fifth level item, using |levelfive|.
\end{prop}
\item Another fourth level item.
\end{prop}
\item Another third level item.
\end{prop}
\item Another second level item.
\end{prop}
\end{prop}
We hope you enjoyed reading \ref{outer}, \ref{second}, \ref{third},

```

`\ref{fourth}`, and `\ref{fifth}`.

- (8) This is an outermost numbered item. Since the default style for nameless items is `eqnum`, it uses the `equation` style.
 - a. This is a second level item, using `leveltwo`.
 - (i) This is a third level item, using `levelthree`.
 - A. This is a fourth level item, using `levelfour`.
 - (I) This is a fifth level item, using `levelfive`.
 - B. Another fourth level item.
 - (ii) Another third level item.
 - b. Another second level item.

We hope you enjoyed reading (8), (8a), (8a.i), (8a.iA), and (8a.iA.I).

roman Roman numerals. At the outer level, they use their own counter (reset every section); inside nested lists, they use the counter appropriate to the nesting level, so one can easily make roman-numbered sublists.

```
\DeclareNumberedStyle{roman}[parent = section,
  counter format = \roman{#1}, format = (#1)]
\SetPropStyle{roman}{counter = \proplevelchoice{
  roman, numpropii, numpropiii, numpropiv, numpropv}}
```

alph Letters. Works the same as `roman`, but with letters. Uses its own dedicated counter at the outer level.

```
\DeclareNumberedStyle{alph}[parent = section,
  counter format = \alph{#1}, display format=#1.,
  ref format = (#1)]
\SetPropStyle{alph}{counter = \proplevelchoice{
  alph, numpropii, numpropiii, numpropiv, numpropv}}
```

```
\begin{prop}
\item[Basic classification, ref format={the \textit{#1}}]
There are three kinds of people. \label{complex}
\begin{prop}
\item[style=roman] \label{partone} Those who know how to count,
comprising in turn:
\begin{prop}
\item[style=alph] \label{parta} Those who know how to count up to
some number, but not beyond.
\item[style=alph] \label{partb} Those who can keep going indefinitely.
\end{prop}
\item[style=roman] \label{parttwo} Those who do not know how to count.
\end{prop}
\end{prop}
For \ref{complex} to serve its purpose, both \ref{partone} (and
its two components \ref{parta} and \ref{partb}) and
\ref{parttwo} are needed.
\begin{prop}
```

```

\item[style=roman]
The |roman| and |alph| styles are also useful for creating ad-hoc
numbered lists at the outer level.
\item[style=roman]
Like this one.
\end{prop}

```

Basic classification There are three kinds of people.

- (i) Those who know how to count, comprising in turn:
 - a. Those who know how to count up to some number, but not beyond.
 - b. Those who can keep going indefinitely.
- (ii) Those who do not know how to count.

For the *Basic classification* to serve its purpose, both (i) (and its two components (a) and (b)) and (ii) are needed.

- (i) The `roman` and `alph` styles are also useful for creating ad-hoc numbered lists at the outer level.
- (ii) Like this one.

hierarchical Produces 1, 1.1, 1.1.1... numbering. Defined via two helper styles:

```

\SetPropStyle{h-base}{counter = numpropi,
  counter format = \arabic{#1}, ref format = #1,
  display format = #1.}
\SetPropStyle{h-sub}{
  counter = \proplevelchoice{numpropi, numpropii,
    numpropiii, numpropiv, numpropv},
  counter format = \arabic{#1},
  format = \parentref{.#1}}
\SetPropStyle{hierarchical}{style = \proplevelchoice{
  h-base, h-sub}, tightspacing}

```

Note that since the optional arguments of `prop`^{P.5} and `inlineprop`^{P.5} only affect the `\items` in the *immediate* scope of that environment (not of any nested environments), styles like `hierarchical`, which one presumably wants to apply to apply an environment along with all its sub-environments, sub-sub-environments, etc., will need to be set using `\proptions`^{P.6}. (The scope of `\proptions` can be controlled by creating a `TeX` group.)

```

\begin{group}
\proptions{style = hierarchical, leftmargin=3em, labelindent=0em}
\begin{prop}
\item \label{theworld} The world is everything that is the case.
\end{prop}
\begin{prop}
\item \label{totality}
The world is the totality of facts, not of things.
\end{prop}
\item
The world is determined by the facts, and by their being all the

```

```

        facts.
      \item
        For the totality of facts determines what is the case, and also
        whatever is not the case.
    \end{prop}
\end{prop}
\end{prop}
Proposition \ref{totality} helps elucidate the meaning of
proposition \ref{theworld}.
\endgroup

```

1. The world is everything that is the case.
 - 1.1 The world is the totality of facts, not of things.
 - 1.1.1 The world is determined by the facts, and by their being all the facts.
 - 1.1.2 For the totality of facts determines what is the case, and also whatever is not the case.
- Proposition 1.1 helps elucidate the meaning of proposition 1.

Environment-level styles

These styles are designed to be used in the optional argument of `prop`^{→P.5} or `inlineprop`^{→P.5}.

`fitmargin` Fits the left margin to the predicted width of the label (computed using `\propwidestlabel`^{→P.11}), so that the label will never extend past the left margin of the following text.

```

\SetPropStyle{fitmargin}{
  labelindent = 0pt,
  leftmargin  = \widthof{\propwidestlabel} + \labelsep}

```

(Explanation: `\propwidestlabel`^{→P.11} computes the widest label the environment's items are expected to produce. By setting `labelindent`^{→P.10} to zero, we make sure that the `labelwidth`^{→P.10} will be sized to fit the computed `leftmargin`^{→P.10}.)

When a list will contain several numbered items, you can use the `items`^{→P.11} key to tell the environment how many `\items` it contains, so that it can properly anticipate how wide the widest label will be.

```

\begin{prop}[name=Fitted Margin, style=fitmargin]
  \item Notice that the |name| key needs to be set in the optional
  argument of |prop| rather than |item|, so that the style knows
  how much space needs to be reserved.
\end{prop}
\begin{prop}[items=2, style=fitmargin]
  \item
    The use of |items=2| is crucial here; without it, we would
    only reserve enough space for a one-digit number.
  \item

```

```

    This happens to be where the |equation| counter hits value 10.
\end{prop}
The |hierarchical| style arguably looks better combined with |fitmargin|:
\begin{group}
\propoptions{style=hierarchical, style=fitmargin}
\begin{prop}
  \item What is the case---a fact---is the existence of states of affairs.
  \begin{prop}
    \item A state of affairs (a state of things) is a combination
      of objects (things).
    \begin{prop}
      \item
        It is essential to things that they should be possible
        constituents of states of affairs.
      \item
        In logic nothing is accidental: if a thing can occur in a
        state of affairs, the possibility of the state of affairs
        must be written into the thing itself.
    \end{prop}
  \end{prop}
\end{prop}
\end{group}

```

Fitted Margin Notice that the `name` key needs to be set in the optional argument of `prop` rather than `item`, so that the style knows how much space needs to be reserved.

- (9) The use of `items=2` is crucial here; without it, we would only reserve enough space for a one-digit number.
- (10) This happens to be where the `equation` counter hits value 10.

The `hierarchical` style arguably looks better combined with `fitmargin`:

- 2. What is the case—a fact—is the existence of states of affairs.
 - 2.1 A state of affairs (a state of things) is a combination of objects (things).
 - 2.1.1 It is essential to things that they should be possible constituents of states of affairs.
 - 2.1.2 In logic nothing is accidental: if a thing can occur in a state of affairs, the possibility of the state of affairs must be written into the thing itself.

outerfit This style behaves similar to `fitmargin` at the outer nesting level (with a bit more space between label and text), but does nothing at other levels. Intended to be set once using `\propoptions`^{P. 6}, for a document that wants ‘`fitmargin`’-like behavior for its top-level propositions, but not for lower-level propositions (where adjusting to accommodate, e.g., roman numerals of different lengths would be a delicate task).

This matches the behaviour of the `\ex.` command in package `LINGUEX`, widely used in linguistics.

```

\SetPropStyle{outerfit}{
  labelindent = \proplevelchoice{0pt, *},
  leftmargin  = \proplevelchoice{

```

```
\widthof{\propwidestlabel} + 1.3em, *}}
```

(Note the use of `*` in the argument of `\proplevelchoice`^{→P.6}, which leaves the deeper levels as though neither key had been set.)

framed Puts an outline around a list (requires the `tcolorbox` package to be loaded). Change the padding with `\setlength\propframepad{...}`.

```
\SetPropStyle{framed}{
  align = flush, leftmargin = Opt, rightmargin = Opt,
  labelindent = {}, continue = false,
  wrapper begin = {\begin{tcolorbox}[
    colback = white, colframe = black,
    boxrule = 0.4pt, arc = Opt, boxsep = Opt,
    left skip =
      \dimexpr\propoperativeleftmargin-\propframepad\relax,
    right skip =
      \dimexpr\propoperativeleftmargin-\propframepad\relax,
    left = \propframepad, right = \propframepad,
    top = \propframepad, bottom = \propframepad]},
  wrapper end = {\end{tcolorbox}}},
}
```

(Explanation: `leftmargin=0` means that the left margin *inside* the frame is zero. But the use of `\propoperativeleftmargin`^{→P.12} indents the whole `tcolorbox` environment by the right amount for the the left margin of the enclosed material to appear at the same position that would have been used for a list without the `frame` style, with the frame rule one `\propframepad` further out.)

```
\begin{prop}[style=framed]
  \item[Important Claim]
  This claim is so important that it deserves to
  put in a special box!
\end{prop}
```

Important Claim This claim is so important that it deserves to put in a special box!

standard A style that resets all the environment-level keys back to their default value. (This is useful, since there is otherwise no easy way to ‘unset’ a globally-set style.) Item-level keys like `align`^{→P.9} and `display format`^{→P.7} are not affected, so these will still be set by the the operative `named style`^{→P.12} or `nameless style`^{→P.12} defaults. (To reset `align` along with the other keys, just set `[style=standard, align=left]`.)

10 Compatibility

The propositions package is designed to work with `hyperref`, `cleveref`, `amsmath`, and `tcolorbox`.

- `amsmath` is required for `\ptagP.6` and the `equations` option.
- With `hyperref` loaded, all cross-referencing commands generate hyperlinks, as usual.
- With `cleveref` loaded, all proposition items are assigned to a private `prop` “reference type”, which by default has no special name, so `\cref` and `\ref` will generate the same output. The `crefname` key can override this.
- The built-in `framed` style requires `tcolorbox` to be loaded.

None of these packages has to be loaded in any particular order relative to `propositions`: each is detected either at load time or at `\begin{document}`, whichever is needed. The usual external conventions still apply—`hyperref` late, and `cleveref` after `hyperref`—which gives:

```
\usepackage{amsmath}      % if using \ptag or the equations option
\usepackage{tcolorbox}    % if using the framed style
\usepackage{hyperref}
\usepackage{propositions}
\usepackage{cleveref}     % if used
```

11 Known issues

When using `\ptagP.6` with a named counter (e.g. `\ptag[counter=P]`) inside an `amsmath` equation environment, `hyperref` may emit warnings of the form:

```
pdfTeX warning: destination with the same
identifier (name{equation.N}) has been already
used, duplicate ignored
```

These warnings are harmless and do not affect the correctness of cross-references.

12 Release notes

0.9 Initial public release.