1 The gn-logic style option

Description of Version 1.5 (5/2026) by Gerd Neugebauer

The `gn-logic` style option provides a facility to typeset logical formulas of a certain kind. This style option provides an environment like `eqnarray`, an extended `newtheorem` environment and several macros.

1.1 Mathematical Symbols

The following macros provide better usage of the junctors and quantifiers. Especially the spacing is improved.

Symbol	Macro		Example
$\wedge$	`\AND`	`A\AND B`	$A \wedge B$
$\vee$	`\OR`	`A\OR B`	$A \vee B$
$\dot{\vee}$	`\XOR`	`A\XOR B`	$A \mathbin{\dot{\vee}} B$
$\rightarrow$	`\IMPLIES`	`A\IMPLIES B`	$A \rightarrow B$
$\rightarrow$	`\IMPL`	`A\IMPL B`	$A \rightarrow B$
$\leftarrow$	`\IF`	`A\IF B`	$A \leftarrow B$
$\leftrightarrow$	`\IFF`	`A\IFF B`	$A \leftrightarrow B$
$\stackrel{\text{def}}{\longleftrightarrow}$	`\IFFdef`	`A\IFFdef B`	$A \stackrel{\text{def}}{\longleftrightarrow} B$
$\wedge\ldots\wedge$	`\ANDdots`	`A_1\ANDdots A_n`	$A_1 \wedge\ldots\wedge A_n$
$\vee\ldots\vee$	`\ORdots`	`A_1\ORdots A_n`	$A_1 \vee\ldots\vee A_n$
$\setminus$	`\is`	`x\is y`	$x \setminus y$
$\mathbb{N}$	`\Nat`	`n\in\Nat`	$n \in \mathbb{N}$
$\forall$	`\Forall`	`\Forall x P(x)`	$\forall x\ P(x)$
$\exists$	`\Exists`	`\Exists y P(x)`	$\exists y\ P(x)$

The `\AND` Macro

This macro can be used for the logical conjunction. In addition to the `\wedge` macro it adds more space and the formulas tend to be better readable. Compare

`x=1\AND y=x`	produces	$x = 1 \wedge y = x$
`x=1\wedge y=x`	produces	$x = 1 \wedge y = x$
`x=1\land y=x`	produces	$x = 1 \wedge y = x$

The `\OR` Macro

This macro can be used for the logical disjunction. In addition to the `\vee` macro it adds more space. Compare

`x=1\OR y=x`	produces	$x = 1 \vee y = x$
`x=1\vee y=x`	produces	$x = 1 \vee y = x$
`x=1\lor y=x`	produces	$x = 1 \vee y = x$

The \XOR Macro

This macro can be used for the exclusive disjunction. It has no common counterpart. The spacing is like in all junctor macros.

`x=1\XOR y=x` produces $x = 1 \dot{\vee} y = x$

The \IMPL and the \IMPLIES Macros

These macros can be used for the logical implication. In addition to the `\rightarrow` macro it adds more space. Compare

`x=1\IMPL y=x`	produces	$x = 1 \rightarrow y = x$
`x=1\IMPLIES y=x`	produces	$x = 1 \rightarrow y = x$
`x=1\rightarrow y=x`	produces	$x = 1 \rightarrow y = x$

The \IF Macro

This macro can be used for the logical implication written in reverse order. In addition to the `\leftarrow` macro it adds more space. Compare

`x=1\IF y=x`	produces	$x = 1 \leftarrow y = x$
`x=1\leftarrow y=x`	produces	$x = 1 \leftarrow y = x$

The \IFF Macro

This macro can be used for the logical equivalence. In addition to the `\leftrightarrow` macro it adds more space. Compare

`x=1\IFF y=x`	produces	$x = 1 \leftrightarrow y = x$
`x=1\leftrightarrow y=x`	produces	$x = 1 \leftrightarrow y = x$

The \IFFdef Macro

Like above but with a small "def" above the arrow.

`x=1\IFFdef y=x` produces $x = 1 \stackrel{\text{def}}{\longleftrightarrow} y = x$

The \is Macro

This macro is for typesetting unifiers. In this case the predefined `\setminus` produces too much space.

`\{y\setminus x, z\setminus 4\}`	produces	$\{y \setminus x, z \setminus 4\}$
`\{y\is x, z\is 4\}`	produces	$\{y\is x, z\is 4\}$
`\{y\backslash x, z\backslash 4\}`	produces	$\{y\backslash x, z\backslash 4\}$

The Number Macros

These are macros for those who have no access to the $\mathcal{AMS}$-TEX fonts. It makes the symbols for the natural numbers, integers, rationals, reals and complex numbers. The usual magnification commands apply to it as well.

	`\tiny`			...		`\normalsize`			...	`\Huge`	`X_X`
`\bbB`	$\mathbb{B}$	$\mathbb{B}$	$\mathbb{B}$	$\mathbb{B}$	$\mathbb{B}$	$\mathbb{B}$	$\mathbb{B}$	$\mathbb{B}$	$\mathbb{B}$	$\mathbb{B}$	$\mathbb{B}_{\mathbb{B}}$
`\Complex\bbC`	$\mathbb{C}$	$\mathbb{C}$	$\mathbb{C}$	$\mathbb{C}$	$\mathbb{C}$	$\mathbb{C}$	$\mathbb{C}$	$\mathbb{C}$	$\mathbb{C}$	$\mathbb{C}$	$\mathbb{C}_{\mathbb{C}}$
`\bbD`	$\mathbb{D}$	$\mathbb{D}$	$\mathbb{D}$	$\mathbb{D}$	$\mathbb{D}$	$\mathbb{D}$	$\mathbb{D}$	$\mathbb{D}$	$\mathbb{D}$	$\mathbb{D}$	$\mathbb{D}_{\mathbb{D}}$
`\bbE`	$\mathbb{E}$	$\mathbb{E}$	$\mathbb{E}$	$\mathbb{E}$	$\mathbb{E}$	$\mathbb{E}$	$\mathbb{E}$	$\mathbb{E}$	$\mathbb{E}$	$\mathbb{E}$	$\mathbb{E}_{\mathbb{E}}$
`\bbF`	$\mathbb{F}$	$\mathbb{F}$	$\mathbb{F}$	$\mathbb{F}$	$\mathbb{F}$	$\mathbb{F}$	$\mathbb{F}$	$\mathbb{F}$	$\mathbb{F}$	$\mathbb{F}$	$\mathbb{F}_{\mathbb{F}}$
`\bbG`	$\mathbb{G}$	$\mathbb{G}$	$\mathbb{G}$	$\mathbb{G}$	$\mathbb{G}$	$\mathbb{G}$	$\mathbb{G}$	$\mathbb{G}$	$\mathbb{G}$	$\mathbb{G}$	$\mathbb{G}_{\mathbb{G}}$
`\bbH`	$\mathbb{H}$	$\mathbb{H}$	$\mathbb{H}$	$\mathbb{H}$	$\mathbb{H}$	$\mathbb{H}$	$\mathbb{H}$	$\mathbb{H}$	$\mathbb{H}$	$\mathbb{H}$	$\mathbb{H}_{\mathbb{H}}$
`\bbI`	$\mathbb{I}$	$\mathbb{I}$	$\mathbb{I}$	$\mathbb{I}$	$\mathbb{I}$	$\mathbb{I}$	$\mathbb{I}$	$\mathbb{I}$	$\mathbb{I}$	$\mathbb{I}$	$\mathbb{I}_{\mathbb{I}}$
`\bbJ`	$\mathbb{J}$	$\mathbb{J}$	$\mathbb{J}$	$\mathbb{J}$	$\mathbb{J}$	$\mathbb{J}$	$\mathbb{J}$	$\mathbb{J}$	$\mathbb{J}$	$\mathbb{J}$	$\mathbb{J}_{\mathbb{J}}$
`\bbK`	$\mathbb{K}$	$\mathbb{K}$	$\mathbb{K}$	$\mathbb{K}$	$\mathbb{K}$	$\mathbb{K}$	$\mathbb{K}$	$\mathbb{K}$	$\mathbb{K}$	$\mathbb{K}$	$\mathbb{K}_{\mathbb{K}}$
`\bbL`	$\mathbb{L}$	$\mathbb{L}$	$\mathbb{L}$	$\mathbb{L}$	$\mathbb{L}$	$\mathbb{L}$	$\mathbb{L}$	$\mathbb{L}$	$\mathbb{L}$	$\mathbb{L}$	$\mathbb{L}_{\mathbb{L}}$
`\bbM`	$\mathbb{M}$	$\mathbb{M}$	$\mathbb{M}$	$\mathbb{M}$	$\mathbb{M}$	$\mathbb{M}$	$\mathbb{M}$	$\mathbb{M}$	$\mathbb{M}$	$\mathbb{M}$	$\mathbb{M}_{\mathbb{M}}$
`\Nat \bbN`	$\mathbb{N}$	$\mathbb{N}$	$\mathbb{N}$	$\mathbb{N}$	$\mathbb{N}$	$\mathbb{N}$	$\mathbb{N}$	$\mathbb{N}$	$\mathbb{N}$	$\mathbb{N}$	$\mathbb{N}_{\mathbb{N}}$
`\bbO`	$\mathbb{O}$	$\mathbb{O}$	$\mathbb{O}$	$\mathbb{O}$	$\mathbb{O}$	$\mathbb{O}$	$\mathbb{O}$	$\mathbb{O}$	$\mathbb{O}$	$\mathbb{O}$	$\mathbb{O}_{\mathbb{O}}$
`\bbP`	$\mathbb{P}$	$\mathbb{P}$	$\mathbb{P}$	$\mathbb{P}$	$\mathbb{P}$	$\mathbb{P}$	$\mathbb{P}$	$\mathbb{P}$	$\mathbb{P}$	$\mathbb{P}$	$\mathbb{P}_{\mathbb{P}}$
`\Rat \bbQ`	$\mathbb{Q}$	$\mathbb{Q}$	$\mathbb{Q}$	$\mathbb{Q}$	$\mathbb{Q}$	$\mathbb{Q}$	$\mathbb{Q}$	$\mathbb{Q}$	$\mathbb{Q}$	$\mathbb{Q}$	$\mathbb{Q}_{\mathbb{Q}}$
`\Real \bbR`	$\mathbb{R}$	$\mathbb{R}$	$\mathbb{R}$	$\mathbb{R}$	$\mathbb{R}$	$\mathbb{R}$	$\mathbb{R}$	$\mathbb{R}$	$\mathbb{R}$	$\mathbb{R}$	$\mathbb{R}_{\mathbb{R}}$
`\Int \bbZ`	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}_{\mathbb{Z}}$
`\bbOne`	$\mathbb{1}$	$\mathbb{1}$	$\mathbb{1}$	$\mathbb{1}$	$\mathbb{1}$	$\mathbb{1}$	$\mathbb{1}$	$\mathbb{1}$	$\mathbb{1}$	$\mathbb{1}$	$\mathbb{1}_{\mathbb{1}}$

Unfortunately the macros `\bbC`, `\bbG`, `\bbO`, and `\bbQ` do not scale properly when used in subscripts or superscripts of formulae. The following example shows how the sizing can be achieved manually

`\bbQ_{\mbox{\scriptsize \bbQ}}` produces $\mathbb{Q}_{\mathbb{Q}}$

The `\Forall` and the `\Exists` Macros

The general problem with quantifiers is that after the quantified variable the following formula is not automatically separated with a small space. This can be overcome by the following macros.

The `\Forall` and the `\Exists` macros take one argument. They typeset the respective quantifier followed by the argument (i.e. the variable) and finally a small space. As usual the argument has to be enclosed in braces if it consists of more than one character. Otherwise the braces can be omitted. This allows an elegant notation of short quantified formulas.

`\Forall x P(x)`	produces	$\forall x\ P(x)$
`\Forall{x_1,\ldots,x_n}P(x_1,\ldots,x_n)`	produces	$\forall x_1, \ldots, x_n\ P(x_1, \ldots, x_n)$
`\Exists x P(x)`	produces	$\exists x\ P(x)$
`\Exists{x_1,\ldots,x_n}P(x_1,\ldots,x_n)`	produces	$\exists x_1, \ldots, x_n\ P(x_1, \ldots, x_n)$

1.2 The `Formula` Environment

This environment allows to typeset logical formulas. The main problem with the `eqnarray` environment was the numbering. In multiline formulas my intention was to have the number in the middle of the formula. Inside this environment several macros are valid.

`\begin{Formula}[`*label*`] \end{Formula}`
: Start the list of formulas. Optionally a label can be given. This label is used to reference the first formula.

`\=`
: Start a new line.

`\>`***level***
: Start a new line and indent to the given *level*. This indentation is done in quantities of `\FormulaIndent` which can be set with the `\setlength` command. The default value is `3em`.

`\Form[`*label*`]`
: Start a new formula. Optionally a *label* can be given. This *label* can be used to reference to the formula (see `\ref`).

Now let's have a look at some examples. First, we see a single two-line formula. Note that the number at the right side is centered between the two lines.

```
\begin{Formula}
    P(X) \IMPL
\=  Q(X) \IFF R_1(X) \OR R_2(X)
\end{Formula}
```

$$P(X) \rightarrow \\ Q(X) \leftrightarrow R_1(X) \vee R_2(X) \tag{1}$$

Next we will see an example of several formulas. The first formula is split to three lines and the third line is indented to level 1. Remark: `\=` is in reality an abbreviation for `\>0`.

```
\begin{Formula}[form:1]
    P(X) \IMPL
\=  Q(X) \IFF R_1(X)
\>1 \OR R_2(X)
\Form[form:2]
    S(X) \IMPL
\=  \neg Q(X) \IFF R_1(X) \OR R_2(X)
\end{Formula}
```

$$P(X) \rightarrow Q(X) \leftrightarrow R_1(X) \vee R_2(X) \tag{2}$$

$$S(X) \rightarrow \neg Q(X) \leftrightarrow R_1(X) \vee R_2(X) \tag{3}$$

1.3 The `NewTheorem` Environment

My experience with the `newtheorem` environment was that I had a certain scheme to use it. First, every theorem got a label. Thus, every *theorem* was followed by a `label` command. Optionally a *theorem* may have a name. This name is typeset right after the number. The body of the *theorem* always started in the next line. This led to the definition of an extended `NewTheorem` environment. The arguments are the same as those of the `newtheorem` environment. But the environment defined by this extended command takes two optional arguments. The first optional argument is a label to be assigned to the *theorem*. This argument has to be enclosed in parentheses. The second type of optional argument has to be enclosed in brackets. It is typeset in `\small` after the title text. The third argument is optional and enclosed in `<>`. It is typeset in `\small\bf` and surrounded by parentheses.

```
\NewTheorem{guess}{Conjecture}

\begin{guess}[Fermat](thm:fermat)
 There do not exist integers $n>2$,
 $x$, $y$, and $z$ such that
 $x^n+y^n=z^n$.
\end{guess}
```

Conjecture 1 *Fermat*
There do not exist integers $n > 2$, x, y, and z such that $x^n + y^n = z^n$.

The commands used to typeset some of the optional argument can be customized in the following way. The macros `\TheoremTitle` and `\TheoremName` are used to typeset their argument in `\small` and `\small\bf` and enclosed in parentheses respectively. These macros can be redefined using `\renewcommand` as shown in the following example:

```
\NewTheorem{theorem}{Theorem}
\renewcommand{\TheoremTitle}[1]{{\sf [#1]}}
\renewcommand{\TheoremName}[1]{{\small(#1)}}
\begin{theorem}[Fermat]<conjecture>(thm:f2)
 There do not exist integers ...
\end{theorem}
```

Theorem 1 Fermat *(conjecture)*
There do not exist integers $n > 2$, x, y, and z such that $x^n + y^n = z^n$.