Introductory User's Guide to *Regular* GELLMU

Version 0.8.4 release, July 2007

William F. Hammond

Dept. of Mathematics & Statistics
University at Albany
Albany, New York 12222 (USA)

revision of July 6, 2007

Table of Contents

1 Introduction

Regular GELLMU is the part of the GELLMU Project under which source language markup most closely resembles actual LaTeX source markup. Much of the markup vocabulary is the same as that of actual LaTeX. The differences — and, indeed, the whole reason for the GELLMU Project — arises from the author's conclusion in 1998 that it would not be possible in a robust way to write rules for actual LaTeX source markup so that the collection of documents in

the *article* document class prepared under such rules would admit fully reliable automatic translation to other formats[1].

The version 0.8 release of the GELLMU Project was the first release providing an output stream for the XML form of HTML extended by the World Wide Web Consortium's Mathematical Markup Language (MATHML)[2]. The driver script `mmkg` provides translations of regular GELLMU articles to (1) actual LaTeX (and from there to DVI and PDF), (2) classic HTML, and (3) XHTML+MATHML, while the driver script `mkg` omits XHTML+MATHML. Though the PDF output may be read on a screen, it is intended primarily for printing by end consumers.

While one could provide a landscape version of the PDF output with live anchors for screen consumption, the XHTML+MATHML version would still be superior for screen reading since, as with ordinary HTML, the text is re-formatted to conform naturally both to magnification and window-resizing actions by the end user. The quality of the rendering of mathematics in the XHTML+MATHML version is very nearly as good as that in the PDF version.

Indeed the author, as an author, finds that the best venue for proof reading what he writes is XHTML+MATHML when displayed in a suitable web browser[3] with larger than normal font size and narrower than normal line length.

Moreover, the XHTML+MATHML version is fully compliant with the Accessibility Guidelines[4] of the World Wide Web Consortium[5].

This document is an introductory user guide. The relation of regular GELLMU to the GELLMU Project overall is explained in the Manual, and advanced user information on regular GELLMU is found there as well. This document covers fundamentals not mentioned in the manual. It is available in PDF[6], XHTML+MATHML[7], and classical HTML[8] versions. Online copies of this Introductory Guide and of the Manual may be found at CTAN[9] and the GELLMU web site[10].

To use this document as an example for learning refer to its source markup[11] and to use the project manual as an example refer to the manual's source markup.

2 First Steps

2.1 Acquiring and Installing

GELLMU is implemented as cross-platform free software licensed under the GNU General

---

[1]This should not be construed to mean, however, that standard LaTeX processing could not incorporate this type of source markup at some time in the future.

[2]URI: http://www.w3.org/Math/

[3]The author prefers the freely available cross-platform browser *Firefox* (`http://www.mozilla.com/`) from the *Mozilla* Project.

[4]URI: http://www.w3.org/WAI/

[5]URI: http://www.w3.org/

[6]URI: userdoc.pdf

[7]URI: userdoc.xhtml

[8]URI: userdoc.html

[9]URI: http://www.tex.ac.uk/tex-archive/help/Catalogue/entries/gellmu.html

[10]URI: http://www.albany.edu/~hammond/gellmu/

[11]URI: userdoc.glm

Public License[12]. Its package is available from the Comprehensive TeX Archive Network[13] (CTAN) and from the GELLMU web site.

The package requires a number of other cross-platform free software packages including GNU Emacs, *Perl*, and some standard items of SGML/XML software, chiefly *Open SP*. Locations may be found in the Manual (key `"requiredsoftware"`).

Linux: The required packages are generally part of a full GNU/*Linux* distribution. The package should be installed in `/usr/local/gellmu` and symlinks to driver scripts should be made from a suitable place in one's command path.

MacOS-X and other Unix variants: The only difference from GNU/*Linux* is that some of the supporting packages may need to be acquired and installed.

MS Windows: The best strategy is to acquire and install a **full** *Cygwin*[14] distribution. Then proceed as with *Linux*.

Additional information may be found in the Manual.

2.2 Writing an Article

The minimal form of a regular GELLMU article is this:

```
\documenttype{article}
\title{Some title}

\begin{document}

First Paragraph.

Second Paragraph.

. . .

\end{document}
```

An article must begin with `\documenttype{article}`. The initial part of an article up to the required `\begin{document}` is the article's *preamble*, and the rest of the article is the article's *body*.

Almost everything in the *body* of an article must be part of a paragraph. Most typically a paragraph is begun with a blank line. Paragraphs themselves may be parts of sectional units.

Beware special characters. For example '`%`' converts the rest of its line to comment status. To use '`%`' in text write `"\%"`. The special characters are with small exceptions the same as those in LATEX. Note, however, that in dealing with multiple downstream formats any character that is not alpha-numeric becomes potentially special. Therefore, there is a named form for every non-alphanumeric ASCII character. For example, `"\tld;"` is the name of ASCII '`~`', which is sometimes handy when forming a URL.[15]

---

[12]URI: http://www.gnu.org/copyleft/
[13]URI: http://www.ctan.org
[14]URI: http://www.cygwin.com/
[15]In GELLMU source, as in LATEX source '`~`' is markup for "non-breaking space". Note further that ASCII

The names of special characters may be found by browsing the SGML document type definition in the file `gellmu.dtd` at the top of the distribution tree.[16]

3 Basic Markup Within Sentences

Although there is no markup for a sentence as a unit, there is markup for the end of a sentence or a "full stop". The normal way to indicate the end of a sentence is to use an ASCII period '.' followed by either a newline or two or more blank spaces. An end of sentence is marked up as the SGML empty element *eos*. This is automatically generated under the condition indicated above. For most purposes it will not matter if an ASCII period is followed by only a single blank space, but in that case no *eos* will be generated.

The foregoing paragraph provides several instances of markup within sentences other than *eos*. There are (1) a quoted phrase, (2) two abbreviations, (3) a quoted character, and (4) two instances of emphasized text. The source for that paragraph is:

```
Although there is no markup for a sentence as a unit, there is markup
for the end of a sentence or a ''full stop''.  The normal way to
indicate the end of a sentence is to use an \abbr{ASCII} period
\quochar{.}  followed by either a newline or two or more blank spaces.
An end of sentence is marked up as the \sgml empty element \emph{eos}.
This is automatically generated under the condition indicated above.
For most purposes it will not matter if an \ascii period is followed
by only a single blank space, but in that case no \emph{eos} will be
generated.
```

The `''full stop''` is a marked up quoted phrase. The named form of a quoted phrase is *quophrase*. The markup `\quophrase{full stop}` is the equivalent expanded form. An abbreviation is indicated with the command *abbr*. There is, of course, no requirement that an abbreviation be marked up; it is merely good practice.

The name of markup for emphasized text is *emph*. A common alternate form of emphasis is called *bold*. One way the two differ is that *emph* has order two effect, i.e., *emph* within *emph* removes emphasis while *bold* is not permitted within itself. Note that **bold** may be emphasized.

In the cited marked up text above the instances `\sgml` and `\ascii` are user-defined *newcommands* occurring in the preamble (or at least before the invocations):

```
\newcommand{\ascii}{\abbr{ASCII}}
\newcommand{\sgml}{\abbr{SGML}}
```

The command *newcommand* is a meta-command in GELLMU source. All *newcommand* instances are fully expanded by the syntactic translator before it writes any output.

The markup `\quochar{x}` produces a quoted character: in this case 'x'. There is no compelling reason why *quophrase* could not have been used instead of *quochar*, but it is customary in technical documentation to use a different presentation style for quoted characters than for

---

tilde `"\tld;"` should not be confused with either the accent command `"\tilde"` or the math command `"\sim"` (rendered as $\sim$) that is typically used for mathematical operators.

[16]The SGML document type definition is more elaborately commented than its XML mirror found in the file `xml/xgellmu.dtd`.

quoted phrases. Beyond that for non-traditional forms of processing it can be very useful to provide for semantic distinctions such as the distinction between a quoted phrase in normal discourse and a quoted string in technical discourse. More particularly, it is significant that in some languages for computer programming there is an important distinction between a character and a string of unit length. Some of the markup commands useful with technical documentation are

command	example in source	example rendered
quochar	`use \quochar{\%} to comment`	use '%' to comment
quostr	`\quostr{unsigned long num;}`	`unsigned long num;`
qquostr	`type \qquostr{done} to exit`	type "done" to exit
path	`\path{C:\bsl;TEX\bsl;TEXMF}`	`C:\TEX\TEXMF`
urlanch	`see \urlanch{userdoc.glm}`	see `userdoc.glm`

With regular LaTeX standard practice is to use *texttt* for the examples just cited involving *quochar*, *quostr*, *path*, and, apart from web-related considerations, *urlanch*. The name *url* has been used variously in regular LaTeX; therefore, *url* has not been introduced as a name in regular GELLMU so that an author may safely use it, if desired, as a macro name.

4 Sectioning

Unless one wants fancy non-default behavior, one may provide sections, subsections, and sub-subsections as in LaTeX. Thus, for example,

```
\documenttype{article}
\title{Some title}

\begin{document}

\section{Title of first section}
First Paragraph.

Second Paragraph.

. . .


\section{Title of second section}
First Paragraph.

Second Paragraph.

. . .

\end{document}
```

With this style of markup one does not indicate the end of a section although there is a way described in the Manual (key `"secusage"`) to gain finer control over sectional units.

5 Lists

As with LaTeX GELLMU *article* provides lists with the names *description*, *enumerate*, and *itemize*. For example, the markup

```
\begin{enumerate}
\item bird
\item cat
\item dog
\end{enumerate}
```

yields:

1. bird
2. cat
3. dog

while the markup

```
\begin{description}
\item[Parrot] a type of bird
\item[Persian] a type of house cat
\item[Pointer] a type of dog
\end{description}
```

yields:

Parrot a type of bird

Persian a type of house cat

Pointer a type of dog

A list is normally part of a paragraph, but a list may occur outside of a paragraph in a sectional unit or even at the top level of an article's body.

An *item* in a list may contain paragraphs but, as the foregoing examples show, may contain loose text.

While LaTeX provides an excellent construction facility for customized lists through its *list* environment, which may be used in conjunction with its *newenvironment* command, there is no facility for the provision of on-the-fly markup in SGML. Therefore, the *article* document type provides several other kinds of list.

The lists called *menu* and *Menu* consist of *items*. These lists have no item labels. They differ only in intention for spacing in presentation formats.

It often makes sense for *menu* to be used inside an *item* of another list. Moreover, a single item *menu* will render in either the LaTeX or HTML formattings as "hanging indentation".

A *defnlist* is intended to be more like the list called *dl* in HTML than like *description*. It must consist of one or more *term*, *desc* pairs with *term* mandatory and *desc* optional.

There are also other lists that will not be mentioned here except to point out that when the name *verbatim* is enabled as a metacommand, rather than as a command, by calling the syntactic translator with a non-default function[17] such as `verblist` or `latex-faq`, the verbatim material

---

[17]The most convenient way to call the syntactic translator with a non-default function is to make the first line

is organized as a *verblist*.

For more information about lists browse the document type definitions `gellmu.dtd` and `xml/xgellmu.dtd` being mindful that default processing in the didactic production system sometimes does a small amount of list manipulation between the SGML and XML versions of an *article*.

6 *Tabular* Environment Emulation

Most users having a LaTeX background will find that *tabular* with column arguments limited to '`l`', '`r`', '`c`', and '`p`' is reasonably functional. Improvement made during 2006 were made possible by the increased availability to users of web browsers supporting version 2 of the cascading style sheets (CSS) specification.

For example, the following markup underlies a small *lcc* table in the section on mathematics.

```
\display{\begin{tabular}{lcc}
\bold{Name} & \bold{Opener}      & \bold{Closer}   \\
math        & \quostr{\bsl;(}    & \quostr{\bsl;)} \\
tmath       & \quostr{\$}        & \quostr{\$}     \\
displaymath & \quostr{\bsl\lsb;} & \quostr{\bsl\rsb;}
\end{tabular}}
```

(In this markup *bsl* is a name for the backslash character and *lsb*, *rsb* the names of the square brackets.)

There is presently no concept of "float".

More information is available in the Manual (key `"tabular"`).

7 Labels, Cross References, and Anchors

The basic idea is very simple. Use

$$\texttt{\textbackslash label}\{\textit{mylabelkey}\}$$

to **mark** a location that is associated with the named label key, and use

$$\texttt{\textbackslash ref}\{\textit{mylabelkey}\}$$

or one of several other methods to reference that location. The command *ref* may be regarded as the default method. In an online version, however, it has **not** been designed to create a selectable anchor but rather simply to indicate section number.

For example, in this document there is a label with the key `"math"` at the beginning of the section (8) on mathematics. The previous sentence contains a reference with the markup `\ref{math}` to that section. This markup results in the number of that section appearing in the first sentence. For clarity the markup for the entire first sentence of this paragraph is:

---

of the source file point to the name of the special function. For example, the function "gellmu-verblist" will be called if the first line of the source file is `"%!verblist"`.

```
For example, in this document there is a label with the key
\qquostr{math} at the beginning of the section (\ref{math}) on
mathematics.
```

If, on the other hand, one wants an online version to contain a selectable anchor to section 8 to the section on math, then in the preamble of the document one might place the newcommand definition

```
\newcommand{\iref}[2]{\anch[iref="#1"]{#2}}
```

and then, as in the foregoing portion of this sentence use the markup

```
\iref{math}{selectable anchor to section \ref{math}} .
```

Note that this markup uses the label key `"math"` twice: once as the first argument to the newcommand called *iref* and once as the sole argument of the command *ref* that itself appears in the second argument of *iref*.

On the question of whether to use *ref* or one of several other commands to make a reference, there are several considerations:

1. How should the reference be indicated in an online format that will be viewed in an ebook or a web browser?

2. How should the reference be indicated in a typeset format where there is neither a cursor nor the possibility of automatic scrolling?

More details on this subject may be found in the Manual (key `"labelref"`).

8 Mathematics

GELLMU *article* provides 5 containers for mathematical content: *math*, *tmath*, *displaymath*, *equation*, *eqnarray*. All may be referenced by name. However, LaTeX-like markup is supported:

Name	Opener	Closer
math	\(	\)
tmath	$	$
displaymath	\[	\]

The containers *math* and *tmath* are essentially equivalent non-displayed containers. The current LaTeX formatter provides a tiny amount of extra horizontal space around *math* but not around *tmath*.

Just as with ordinary LaTeX there can be difficulty locating a missing '$' if one uses these for *tmath*. Unbalanced '$' characters usually lead the syntactic translator to complain (correctly, though not with as much user-friendliness as one might wish) about not being able to balance a brace. In this case if *tmath* is seen in the displayed surrounding text, a missing '$' is likely.

MathML rendering support (with MathML's presentation markup) in HTML versions, i.e., the modern XML form of HTML extended with MATHML, is new in the summer of 2004.

A program to generate an output stream providing content-level MathML markup, i.e., semantic markup transparent for computer algebra systems, has not yet been written.

Those who use mathematics may find the following illustrations instructive.

The markup

```
Just as with \tex; or \latex; it is easy to provide a mathematical
formula, such as $y = x^3/(x^2 + 1)$\aoc either inline or displayed.
```

produces:

Just as with TEX or LATEX it is easy to provide a mathematical formula, such as $y = x^3/(x^2 + 1)$, either inline or displayed.

This markup

```
Here the same formula is displayed.
\[ y = \frac{x^3}{x^2 + 1} \eos\]
```

yields:

Here the same formula is displayed.

$$y \;=\; \frac{x^3}{x^2 + 1} \;\;.$$

The *eos* corresponds to an empty SGML element for "end of sentence" (or "full stop"). Normally one simply uses '.', but inside a displayed math zone there is semantic-level risk of confusion with various possible uses of '.' as a mathematical symbol.

Markup:

```
The square root of a compound fraction:
\[ \sqrt{\frac{\frac{a}{b}}{\frac{c}{d}}} \eos \]
```

Rendered:

The square root of a compound fraction:

$$\sqrt{\frac{\frac{a}{b}}{\frac{c}{d}}} \;\;.$$

In the current XHTML+MATHML formatting by default radical signs in source markup are converted by default to fractional exponents because of an intellectual property issue with fonts available for use by the open source *Mozilla* browser and its spinoffs including *Firefox*, *Galeon*, and *Nautilus*.[18]

Using the markup for numbered equations

```
The quadratic equation
```

---

[18]Similar intellectual property issues pertain more widely to fonts. In particular, for this reason by default in the XHTML+MATHML output character coloring is used to cover possible font failures for the blackboard bold, calligraphic, and fraktur math fonts; the colors corresponding colors are, respectively, red, blue, and green.

```
\begin{equation} a x^2 + b x + c = 0 \end{equation}
has roots
\begin{equation}
 x = \frac{-b \pm \sqrt{b^2 - 4 a c}}{2 a} \ \eos
\end{equation}
```

one obtains the rendering:

The quadratic equation

$$(1) \qquad\qquad ax^2 + bx + c \;=\; 0$$

has roots

$$(2) \qquad\qquad x \;=\; \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \quad.$$

Rule for subscripts and superscripts: A subscript or superscript must be braced if key-boarding it involves more than one character. Thus, one *may* use `$C^2$` or `$C^{2}$` to produce C^2, but one *must* use `$C^{\infty}$` to generate C^∞.

The Taylor series of the *exponential* function is

$$e^t \;=\; \sum_{k=0}^{\infty} \frac{t^k}{k!} \quad.$$

Note here that the *sum* markup \sum requires a formal terminator \sum: (likewise with \int for integral and \prod for product).

```
\[ e^t = \sum_{k = 0}^{\infty} \, \frac{t^k}{k!} \sum: \ \eos \]
```

The markup

```
The continued fraction expansion of the \emph{golden mean}:
\[\frac{1 + \sqrt{5}}{2} =
    1 + \frac{1}{1+\frac{1}{1+\frac{1}{1+\frac{1}{1+\ldots;}}}} \eos \]
```

produces:

The continued fraction expansion of the *golden mean*:

$$\frac{1 + \sqrt{5}}{2} \;=\; 1 + \cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\ldots}}}} \quad.$$

while the markup

```
Mixed function application and multiplication:
\[ {\func{sin}\aF;{ax}}\itimes;{\func{cos}\aF;{bx}} \eos \]
```

yields

Mixed function application and multiplication:

$$\sin ax\cos bx \quad.$$

Markup:

```
Newton's binomial series:
\[ (1 + t)^r = \sum_{k=0}^{\infty} \,
   \frac{r(r - 1)(r - 2)\ldots(r - k + 1)}{k!}\, t^k \sum: \eos \]
```

rendering:

Newton's binomial series:

$$(1+t)^r \;=\; \sum_{k=0}^{\infty} \frac{r(r-1)(r-2)\ldots(r-k+1)}{k!} \, t^k \quad.$$

Markup:

```
A differential equation:
\[ D^2 y - 3 x (D y)^2 = x \func{cos} x \ \eos \]
```

rendering:

A differential equation:

$$D^2 y - 3x(Dy)^2 \;=\; x\cos x \quad.$$

The statement of Stokes's Theorem in space

$$\int\!\!\int_S (\mathbf{curl\,F} \,\cdot\, \mathbf{N})\,d\sigma \;=\; \int_{\partial S} (\mathbf{F}\,\cdot\,\mathbf{T})\,ds \quad.$$

may be generated by the markup

```
\[ \int\int_S \, (\mbox{\bold{curl}}
 \, \regch{\bold{F}} \,\cdot\, \regch{\bold{N}})\, d\sigma;
\int:\int:
 =
\int_{\partial S} \, (\regch{\bold{F}}
\,\cdot\, \regch{\bold{T}})\, ds \int:\, \eos \]
```

Note that each \int integral opener must be followed eventually by a \int: closer (or braces could be used, i.e., \int{ ... }, or \begin{int} ... \end{int} also could be used.

The branched definition of the absolute value of a real number may be marked up with

```
\[ \absval{x} = \lbalbr{\begin{array}{rl}
              x & \text{\ if\ } x \geq 0 \\
              -x & \text{\ if\ } x < 0
              \end{array}} \ \eos \]
```

to yield

$$|x| \;=\; \begin{cases} x & \text{if } x \geq 0 \\ -x & \text{if } x < 0 \end{cases} \quad.$$

The LaTeX-like command *mathbf* is new in GELLMU *article*. It may only be used in math. The command *bold*, which indicates bold-style emphasis, parallel to *strong* in HTML and *textbf* in LaTeX, may be used inside mathematical content as well as outside. However, *bold* may not contain math-special symbols such as that given by \alpha, while *mathbf* may.

A certain amount of attention to semantics is required in order to have robust generation of the XHTML+MathML output. An absolutely basic part of this is having unambiguous

determination of what is a single symbol. By long convention in TeX the string "abcd" in math is understood as the product (in whatever sense of "product") of four symbols. If a string with more than one character — for example, "Hom" — is to be a single symbol, it may be marked up with *mbox* or, in GELLMU source, be assigned a symbol name having custom print representation using *mathsym* in the preamble. In GELLMU source *mathbf* should only contain a single symbol.

The markup

```
\display{$\nabla\times\vec{\mbox{\bold{B}}}$
        \ with ''mbox'' and ''bold'' versus
        \ $\nabla\times\vec{\mathbf{B}}$ \ with ''mathbf''\,.}
```

produces

$$\nabla \times \vec{\mathbf{B}} \quad \text{with "mbox" and "bold" versus} \quad \nabla \times \vec{\boldsymbol{B}} \quad \text{with "mathbf".}$$

Diagrams are done using *array*. They must, therefore, be rectangular. The markup

```
\newcommand{\ra}{\rightarrow}
\newcommand{\da}{\downarrow}
\[ \begin{array}{ccccccccc}
0 & \ra  & A'  & \ra  & A   & \ra & A''  & \ra & 0 \\
~ &      & \da &      & \da &     & \da  &     &   \\
0 & \ra  & B'  & \ra  & B   & \ra & B''  & \ra & 0
\end{array} \]
```

yields

$$
\begin{array}{ccccccccc}
0 & \to & A' & \to & A & \to & A'' & \to & 0 \\
 & & \downarrow & & \downarrow & & \downarrow & & \\
0 & \to & B' & \to & B & \to & B'' & \to & 0
\end{array}
$$

Comments about *array*, *eqnarray*, *tabular*

1. In this markup note the appearance of the character '~' at the beginning of the second row of the array. The modular design of the didactic production system requires that the syntactic translator may only deal with markup syntax. The '~', which is markup in GELLMU source as in LaTeX for "non-breaking space", in the first cell of the second row serves to keep the first cell from being empty. There is a technical error in the SGML output of the syntactic translator when the first cell in an *array* or *tabular* row is empty unless there is an explicit opentag for the first cell. The technical error will trigger a validation error though it should not interfere with the generation of a correct parsed output.[19]

2. Also in the foregoing markup note that the LaTeX-like use of the character '&' as the introducer of a new cell must in GELLMU source be followed by whitespace. Otherwise there is risk of confusion with the alternative markup use of '&' in GELLMU source (though not in LaTeX source) to initiate an SGML entity reference.

---

[19] Beyond that in certain situations some web browsers appear not to deal well with empty table cells, and placing an HTML non-breaking space in such a cell is often a good thing for that reason. Of course, the special character '~' for non-breaking space in GELLMU source gives rise in HTML output to the SGML non-breaking space character denoted with the entity reference " ".

There is multiscript support. The command *mscript* takes 5 arguments. The first is its base.
The script order is counter-clockwise beginning with upper-left. For example,

$$\begin{matrix} ^{14}_{2}\boldsymbol{C}^{2}_{5+} & {}^{a}_{b}\mathrm{Hom}^{d}_{c}(X,Y) \end{matrix}$$

is produced by

```
\[
\mscript{\mathbf{C}}{14}{2}{5+}{2}\quad
\mscript{\mbox{Hom}}{a}{b}{c}{d}(X,Y)
\]
```

AMSMath-like overset and underset: \overset{A}{B} then \underset{A}{B}

$$\overset{A}{B} \quad \underset{A}{B}$$

The command *overset* is the *AMSMath* replacement in LaTeX for LaTeX's native *stackrel*.

```
A map may be labeled using \emph{overset}:
\[ X \overset{f}{\longrightarrow} Y \]
or using \emph{underset}:
\[ X \underset{f}{\longrightarrow} Y \]
```

A map may be labeled using *overset*:

$$X \overset{f}{\longrightarrow} Y$$

or using *underset*:

$$X \underset{f}{\longrightarrow} Y$$

Binary or pseudo-binary operators:

vee: $\vee$ setminus: $\backslash$ ll: $\ll$ gg: $\gg$ approx: $\approx$ asymp: $\asymp$

rightarrow: $\rightarrow$ leftarrow: $\leftarrow$ implies: $\Rightarrow$ revimplies: $\Leftarrow$ iff: $\Leftrightarrow$

Other operators:

uparrow: $\uparrow$ downarrow: $\downarrow$ triangle: $\triangle$ abuts: $\Rightarrow$

The command *vect* is for vector generation. Its usage is:

\vect[*open-close-separator spec*]{*coordinate*} ...

Thus, the markup \vect{x}{y}{z} produces (x,y,z). On the other hand the markup \vect[]{x}{y}{z}
produces xyz. This might be useful, for example, in providing reasonably semantic (but visibly
traditional) markup for a tensor field. For example

$$T^{i_1 i_2 \dots i_p}_{j_1 j_2 \dots j_q}$$

When *vect*'s option is empty, vector opener, closer, and coordinate separators are all nil. *vect* might be used to provide the stylized, though semantically dubious, inner product markup `\vect[<>;]{v}{w}` yielding

$$< v; w > \; ,$$

or with the markup `\vect[\lsb;\rsb;:]{a}{b}{c}` to customize notation for the homogeneous coordinate triple representing a point of $\boldsymbol{P}^2$:

$$[a : b : c] \quad .$$

Leray's spectral sequence:

If

$$X \; \longrightarrow \; B$$

with fibre F, then

$$E_2^{pq} \; = \; H^p(B, H^q(F)) \Rightarrow H^*(X) \quad .$$

is produced by

```
If
\[ X \ \longrightarrow \  B \]
with fibre $F$\@, then
\[ {\mscript{E}{}{}{2}{\vect[]{p}{q}} = H^p\aF(B, H^q\aF(F))}
   \abuts H^{*}\aF(X) \ \eos \]
```

In this markup the command `\aF`, is not strictly necessary. It provides a deliberate way to indicate that, in the second instance above, the symbol X is argument-like in relation to the symbol H^*. In MathML rendering *aF* gives rise to presentation-level MathML's invisible operator *ApplyFunction*.

9 Beyond this Introductory Guide

More information about GELLMU is available in the Manual. Some examples of topics to be found there include:

- Inclusion of graphic objects: Manual (key `"graphics"`).

- Equations and equation arrays: Manual (key `"eqn"`).

- Emulation of LaTeX's *newtheorem* environments: Manual (key `"assert"`).

- Emulation of LaTeX's counters: Manual (key `"counters"`).

- The flowchart for regular GELLMU processing: Manual (key `"flow"`).

The author tries to keep fresh information on the GELLMU web site.