Code documentation to the physics2 package

Zhang Tingxuan

2024/01/10 v1.0.1

Contents

File I

The bare physics2

1 The physics2 package

```
1 ⟨*package⟩
2 \NeedsTeXFormat{LaTeX2e}[2020/10/01]
3 \ProvidesPackage{physics2}
4   [2024/01/10 v1.0.1 Tools for typesetting math for physics.]
```

1.1 Common variables

\phy@temp.. `\phy@temp⟨register type⟩⟨a or b⟩`

Some LaTeX 2_ε variables starting with "\phy@temp". These variables can be shared by any module of physics2.

```
5 \newcount  \phy@tempcnta
6 \newdimen  \phy@tempdima
7 \newdimen  \phy@tempdimb
8 \newskip   \phy@tempskipa
9 \newmuskip \phy@tempmuskipa
10 \newbox    \phy@tempboxa
11 \newif     \ifphy@tempswa
12 \newtoks   \phy@toksa
```

1.2 Package requirements and module-loading methods

physics2 requires keyval (part of the graphics bundle) to process options of modules.

```
13 \RequirePackage{keyval}
14 \def\phy@true{true}
15 \def\phy@false{false}
```

\phy@define@key `\phy@define@key {⟨module⟩} {⟨key⟩} [⟨default value⟩] {⟨code⟩}`
\phy@setkeys `\phy@setkeys {⟨module⟩} {⟨key-val list⟩}`
\phy@processkeyopt `\phy@processkeyopt {⟨module⟩}`

The position of \phy@processkeyopt in a physics2 module is just the same as the position of \ProcessOptions in a regular LaTeX package.

```
16 \long\def\phy@define@key#1{\define@key{phy-#1}}
17 \long\def\phy@setkeys#1{\setkeys{phy-#1}}
18 \def\phy@processkeyopt#1{\let\reserved@a\@empty%
19   \edef\reserved@a{\@ptionlist{\@currname.\@currext}}%
20   \edef\reserved@a{\noexpand\phy@setkeys{#1}{\reserved@a}}%
21   \reserved@a% the next line thanks to `geometry'
22   \AtEndOfPackage{\let\@unprocessedoptions\relax}}
```

We use almost the same way to load physics2 modules as LaTeX 2_ε kernel does. We use a lot of kernel commands in LaTeX 2_ε.

\usephysicsmodule `\usephysicsmodule [⟨key-val options⟩] {⟨module⟩} [⟨key-val options⟩]`
\phy@requiremodule `\phy@requiremodule [⟨key-val options⟩] {⟨module⟩} [⟨key-val options⟩]`

\usephysicsmodule is a user command, and \phy@requiremodule is a developer command.

```
\def\usephysicsmodule{\phy@FWoptions\@pkgextension}
\let\phy@requiremodule\usephysicsmodule
\@onlypreamble\usephysicsmodule
\def\phy@FWoptions#1{\@ifnextchar[%]
  {\phy@FW@ptions#1}{\phy@FW@ptions#1[]}}
\@onlypreamble\phy@FWoptions
\def\phy@FW@ptions#1[#2]#3{\@ifnextchar[%]
  {\phy@FW@pti@ns#1[{#2}]#3}{\phy@FW@pti@ns#1[{#2}]#3[]}}
\@onlypreamble\phy@FW@ptions
\def\phy@FW@pti@ns#1[#2]#3[#4]{%
  \def\reserved@b##1,{%
    \ifx\@nnil##1\relax\else
      \ifx\@nnil##1\@nnil\else
       \noexpand\@onefilewithoptions{phy-##1}[{\unexpanded{#2}}][{#4}]%
       \noexpand\@pkgextension
      \fi
      \expandafter\reserved@b
    \fi}%
    \edef\reserved@a{\zap@space#3 \@empty}%
    \edef\reserved@a{\expandafter\reserved@b\reserved@a,\@nnil,}%
  \reserved@a}
\@onlypreamble\phy@FW@pti@ns
```

1.3 The (used to be) common module

The code below used to be the automatically-loaded common module, but now we load it together with physics2's code. This change may bring better performance in Windows system.

Check if unicode-math loaded and (re)define the vert symbols. The \relax's at the ends of \vert and \Vert's definitions must not be removed. They are for \ifx to compare. unicode-math sets these symbols \fam1, \symoperators is equal to 1 in LaTeX 2_ε kernel. Moreover, we make \mid as a delimiter but it may not work.

```
\AtBeginDocument{\ifcsname symrm\endcsname
  \protected\def\|{\Udelimiter 0 \symoperators "2016 }%
  \protected\def\vert{\Udelimiter 0 \symoperators "007C\relax}%
  \protected\def\Vert{\Udelimiter 0 \symoperators "2016\relax}%
  \protected\def\mid{\Udelimiter 3 \symoperators "007C }%
\fi}
\protected\def\Vert{\delimiter"026B30D\relax}
\protected\def\mid{\delimiter"326A30C }
```

\delopen `\delopen ⟨left delimiter⟩`
\delclose `\delclose ⟨right delimiter⟩`

Actually in TeX, \left and \right will enclose the subformula as "inner", but \delopen and \delclose will make the subformula an empty open node and a non-empty close node.

```
\DeclareRobustCommand\delopen{\mathopen{}\mathclose\bgroup\left}
\DeclareRobustCommand\delclose{\aftergroup\egroup\right}
% Extension to 2e kernel's or amsmath's biggggg commands.
```

\bBigg@ is a command from amsmath. The code below should update with amsmath together.

```
\ifdefined\bBigg@
  \DeclareRobustCommand\biggg{\bBigg@{3}}
  \DeclareRobustCommand\Biggg{\bBigg@{3.5}}
\else
  \DeclareRobustCommand\biggg[1]{\leavevmode@ifvmode
    {\hbox{$\left#1\vbox to20.5\p@{}\right.\n@space$}}}
  \DeclareRobustCommand\Biggg[1]{\leavevmode@ifvmode
    {\hbox{$\left#1\vbox to23.5\p@{}\right.\n@space$}}}
  \AtBeginDocument{\ifdefined\bBigg@
    \DeclareRobustCommand\biggg{\bBigg@{3}}%
    \DeclareRobustCommand\Biggg{\bBigg@{3.5}}%
  \fi}
\fi
\DeclareRobustCommand\bigggl{\mathopen\biggg}
\DeclareRobustCommand\bigggm{\mathrel\biggg}
\DeclareRobustCommand\bigggr{\mathclose\biggg}
\DeclareRobustCommand\Bigggl{\mathopen\Biggg}
\DeclareRobustCommand\Bigggm{\mathrel\Biggg}
\DeclareRobustCommand\Bigggr{\mathclose\Biggg}
```

\phy@mathvphantom

\phy@mathvphantom {⟨*math mode material*⟩}

This command is just like \vphantom in LaTeX 2ε kernel but only works in math mode.

```
\def\phy@mathvphantom#1{\setbox\phy@tempboxa=\hbox{}%
  \mathchoice
    {\setbox\@tempboxa\hbox{$\displaystyle#1$}%
      \ht\phy@tempboxa=\ht\@tempboxa
      \dp\phy@tempboxa=\dp\@tempboxa
      \box\phy@tempboxa}
    {\setbox\@tempboxa\hbox{$\textstyle#1$}%
      \ht\phy@tempboxa=\ht\@tempboxa
      \dp\phy@tempboxa=\dp\@tempboxa
      \box\phy@tempboxa}
    {\setbox\@tempboxa\hbox{$\scriptstyle#1$}%
      \ht\phy@tempboxa=\ht\@tempboxa
      \dp\phy@tempboxa=\dp\@tempboxa
      \box\phy@tempboxa}
    {\setbox\@tempboxa\hbox{$\scriptscriptstyle#1$}%
      \ht\phy@tempboxa=\ht\@tempboxa
      \dp\phy@tempboxa=\dp\@tempboxa
      \box\phy@tempboxa}%
}
```

1.4 The (used to be) explsetup module

Some common variables and functions for experimental LaTeX3 syntax.

```
⟨@@=phy⟩
```

```
\ExplSyntaxOn
\int_new:N \l__phy_tmpa_int
\int_new:N \l__phy_tmpb_int
\tl_new:N \l__phy_tmpa_tl
\tl_new:N \l__phy_tmpb_tl
```

The function that can gobble one token.

```
\cs_new:Npn \__phy_gobble_i:n #1 { }
\ExplSyntaxOff
⟨@@=⟩
⟨/package⟩
```

File II
Modules written in LaTeX 2ε syntax

1 The ab module

⟨*gibberish⟩

This module is important but the code is hard to read. One of the motivations I manage physics2 with DocStrip is that, when I tried to write a new module based on ab after 5 months when I maintained physics2 the last time, I found that I could not understand the code I wrote at all! Therefore, it's significant to comment out the alien code in ab.

⟨/gibberish⟩

```
⟨*ab⟩
\ProvidesFile{phy-ab.sty}
  [2023/10/24 `ab' (autobraces) module of physics2]
```

If you don't know when to use \phy@define@key, \phy@setkeys and \phy@processkeyopt in a module, see ahead. In ab, the tightbraces option can control if the automatically-sized braces are tight or not. Do you remember \delopen and \delclose?

```
\phy@define@key{ab}{tightbraces}[true]{\def\@phy@abtight{#1}}
\phy@setkeys{ab}{tightbraces=true}
\phy@processkeyopt{ab}
```

\phy@abopen `\phy@abopen ⟨left delimiter⟩`
\phy@abclose `\phy@abclose ⟨right delimiter⟩`

They are defined either {\delopen, \delclose} or {\left, \right}. If a module requires ab, these two commands are likely to be used.

```
\ifx\@phy@abtight\phy@true
  \let\phy@abopen\delopen
  \let\phy@abclose\delclose
\else
  \let\phy@abopen\left
  \let\phy@abclose\right
\fi
```

1.1 The implementation of \ab

This is the alienest part of ab. It's better to draw something rather than write boring comments. First let's take a look at \ab's syntax. After \ab should be a pair of delimiters; take () as an example. Between \ab and "(" can be a biggg command or star, or even nothing. \ab is defined as follows:

```
\ab ← begindef
        \phy@d@lx {mb} {ab}
      enddef
```

where ab is the branch name of \ab(), and mb is the branch name of \ab\big() and \ab*(). Then let's see the syntax of \phy@d@lx.

```
\phy@d@lx {⟨biggg or star branch name⟩} {⟨automatic branch name⟩} {#3}
```

Here exists an #3. #3 is one token immediately following \ab, which can be { a biggg command or a star } or a "(", under our assumption.

\phy@d@lx is defined as follows:

```
\phy@d@lx ← begindef (#1: biggg or star branch name, ⟨mb⟩; #2:
              automatic branch name, ⟨ab⟩; #3, the token after
              \ab)
              if #3 == biggg or #3 == star (⇔ csname
                {phy@del\string#3} is defined) then
                let ⟨next cs⟩ = csname {phy@d@lx⟨mb⟩}
              else
                let ⟨next cs⟩ = csname {phy@d@lx⟨ab⟩}
              endif
              ⟨next cs⟩ #3
            enddef
```

The condition should be true when #3 is \big or *, and it should be false when #3 is "(". Accordingly, in math mode,

```
\ab \big (   →   \phy@d@lxmb \big (
\ab      (   →   \phy@d@lxab      (
```

Then we meet two new commands — \phy@d@lxmb and \phy@d@lxab. Syntax is as follows.

```
\phy@d@lxmb ⟨biggg or *⟩ ⟨left delimiter⟩ ⟨subformula⟩ ⟨right delimiter⟩
\phy@d@lxab              ⟨left delimiter⟩ ⟨subformula⟩ ⟨right delimiter⟩
```

Notice that ab and mb in the above commands are names of \ab's two branches — they are like namespaces. \phy@d@lxmb and \phy@d@lxab are defined by the following two lines:

```
\phy@d@l@genxm{mb}
\phy@d@l@genxa{ab}
```

\phy@d@l@genxm and \phy@d@l@genxa are defined as follows:

```
\phy@d@l@genxm ← begindef (#1: biggg or star branch name, ⟨mb⟩)
                  \phy@d@lx⟨mb⟩ ← begindef (##1: biggg or star;
                                            ##2: left delimiter)
                                    \begingroup
                                    if ##1 == star then
                                      ⟨temp⟩ ← \relax
                                    else
                                      ⟨temp⟩ ← ##1
                                    endif
                                    csname {phy@⟨mb⟩@\string##2}
                                      ⟨temp⟩ ##2
                                    % requires an \endgroup af-
                                      ter the right delimiter
                                  enddef
                  enddef
\phy@d@l@genxa ← begindef (#1: automatic branch name, ⟨ab⟩)
                  \phy@d@lx⟨ab⟩ ← begindef (##1: left delimiter)
                                    csname {phy@⟨ab⟩@\string##1}
                                      ##1
                                  enddef
                  enddef
```

So we can get

```
\ab \big (   →   \begingroup csname {phy@mb@(} \big   (
\ab *     (   →   \begingroup csname {phy@mb@(} \relax (
\ab       (   →                csname {phy@ab@(} (
```

The csnames above (\phy@mb@(and \phy@ab@() are generated with \phy@AB@gen.

\phy@AB@gen {⟨*branch name*⟩} ⟨*left delimiter*⟩ {⟨*arg spec*⟩} {⟨*definition*⟩}

If ⟨*branch name*⟩ is mb, {⟨*arg spec*⟩} should be mr(), where m is for biggg or star; If ⟨*branch name*⟩ is ab, {⟨*arg spec*⟩} should be r().

Note: The "(" in the example above must not be replaced by a subformula braced by a pair of {}.

\phy@AB@gen \phy@AB@gen {⟨*branch name*⟩} ⟨*left delimiter*⟩ {⟨*arg spec*⟩} {⟨*definition*⟩}

```
\def\phy@AB@gen#1#2{\expandafter\DeclareDocumentCommand\csname phy@#1@\string#2\endcsname}
\phy@AB@gen{ab}({r()}{\phy@abopen(#1\phy@abclose)}
\phy@AB@gen{ab}[{r[]}{\phy@abopen[#1\phy@abclose]}
\phy@AB@gen{ab}\{{r\{\}}{\phy@abopen\{#1\phy@abclose\}}
\phy@AB@gen{ab}|{r||}{\phy@abopen|#1\phy@abclose|}
```

```
\phy@AB@gen{ab}\|{r\|\|}{\phy@abopen\|#1\phy@abclose\|}
\phy@AB@gen{ab}<{r<>}{\phy@abopen<#1\phy@abclose>}
\phy@AB@gen{ab}\lbrace{r\lbrace\rbrace}{\phy@abopen\lbrace#1\phy@abclose\rbrace}
\phy@AB@gen{ab}\vert{r\vert\vert}{\phy@abopen\vert#1\phy@abclose\vert}
\phy@AB@gen{ab}\Vert{r\Vert\Vert}{\phy@abopen\Vert#1\phy@abclose\Vert}
\phy@AB@gen{ab}\langle{r\langle\rangle}{\phy@abopen\langle#1\phy@abclose\rangle}
```

\endgroup's in the end of the following definitions are corresponding to \begingroup's in the definition of \phy@d@l@genxm.

```
\phy@AB@gen{mb}({mr()}{\mathopen#1(#2\mathclose#1)\endgroup}
\phy@AB@gen{mb}[{mr[]}{\mathopen#1[#2\mathclose#1]\endgroup}
\phy@AB@gen{mb}\{{mr\{\}}{\mathopen#1\lbrace#2\mathclose#1\rbrace\endgroup}
\phy@AB@gen{mb}|{mr||}{\mathopen#1\vert#2\mathclose#1\vert\endgroup}
\phy@AB@gen{mb}\|{mr\|\|}{\mathopen#1\Vert#2\mathclose#1\Vert\endgroup}
\phy@AB@gen{mb}<{mr<>}{\mathopen#1\langle#2\mathclose#1\rangle\endgroup}
\phy@AB@gen{mb}\lbrace{mr\lbrace\rbrace}{\mathopen#1\lbrace#2\mathclose#1\rbrace\endgroup}
\phy@AB@gen{mb}\vert{mr\vert\vert}{\mathopen#1\vert#2\mathclose#1\vert\endgroup}
\phy@AB@gen{mb}\Vert{mr\Vert\Vert}{\mathopen#1\Vert#2\mathclose#1\Vert\endgroup}
\phy@AB@gen{mb}\langle{mr\langle\rangle}{\mathopen#1\langle#2\mathclose#1\rangle\endgroup}
```

\phy@del\string. The syntax seems not important. These following lines seems only for \ifcsname to judge if the commands are defined.

```
\def\phy@del#1#2#3{\phy@abopen#1#3\phy@abclose#2}
\expandafter\def\csname phy@del\string*\endcsname#1#2#3{\mathopen#1#3\mathclose#2}
\expandafter\def\csname phy@del\string\big\endcsname#1#2#3{\bigl#1#3\bigr#2}
\expandafter\def\csname phy@del\string\Big\endcsname#1#2#3{\Bigl#1#3\Bigr#2}
\expandafter\def\csname phy@del\string\bigg\endcsname#1#2#3{\biggl#1#3\biggr#2}
\expandafter\def\csname phy@del\string\Bigg\endcsname#1#2#3{\Biggl#1#3\Biggr#2}
\expandafter\def\csname phy@del\string\biggg\endcsname#1#2#3{\bigggl#1#3\bigggr#2}
\expandafter\def\csname phy@del\string\Biggg\endcsname#1#2#3{\Bigggl#1#3\Bigggr#2}
```

\phy@d@lx \phy@d@lx {⟨*biggg or star branch name*⟩} {⟨*automatic branch name*⟩} {#3}

```
\def\phy@d@lx#1#2#3{%
  \ifcsname phy@del\string#3\endcsname
    \def\reserved@a{#1}% #3 is star or \biggg
  \else
    \def\reserved@a{#2}% #3 is delimiter
  \fi
  \csname phy@d@lx\reserved@a\endcsname#3}
```

\phy@d@l@genxm \phy@d@l@genxm {⟨*biggg or star branch name*⟩}
\phy@d@l@genxa \phy@d@l@genxa {⟨*automatic branch name*⟩}

```
\def\phy@d@l@genxm#1{%
  \expandafter\def\csname phy@d@lx#1\endcsname##1##2{%
    \begingroup % \endgroup is at the end of #4 of \phy@AB@gen
    \ifx##1*\let\phy@tempa=\relax\else\let\phy@tempa=##1\fi
    \csname phy@#1@\string##2\endcsname\phy@tempa##2}}
\def\phy@d@l@genxa#1{%
  \expandafter\def\csname phy@d@lx#1\endcsname##1{%
    \csname phy@#1@\string##1\endcsname##1}}
```

\phy@d@lxmb \phy@d@lxmb ⟨*biggg or **⟩ ⟨*left delimiter*⟩ ⟨*subformula*⟩ ⟨*right delimiter*⟩
\phy@d@lxab \phy@d@lxab ⟨*left delimiter*⟩ ⟨*subformula*⟩ ⟨*right delimiter*⟩

```
\phy@d@l@genxm{mb}
\phy@d@l@genxa{ab}
```

\ab The users' command \ab.

```
\DeclareRobustCommand\ab{\phy@d@lx{mb}{ab}}
```

1.2 \pab-like commands

This is so simple. No need to comment a lot.

\phy@d@l@geny \phy@d@l@geny ⟨*command*⟩ ⟨*left delimiter*⟩ ⟨*right delimiter*⟩

This command used to define commands like \pab.

```
\def\phy@d@l@geny#1#2#3{%
  \DeclareDocumentCommand#1{som}{% ##1: star; ##2: bigg (csname); ##3: subformula.
    \IfBooleanTF{##1}%
      {#2##3#3}%
      {\IfValueTF{##2}%
        {\csname##2l\endcsname#2##3\csname##2r\endcsname#3}%
        {\phy@abopen#2##3\phy@abclose#3}%
      }%
  }%
}
\phy@d@l@geny\pab()
\phy@d@l@geny\bab[]
\phy@d@l@geny\Bab\lbrace\rbrace
\phy@d@l@geny\vab\vert\vert
\phy@d@l@geny\aab\langle\rangle
\phy@d@l@geny\Vab\Vert\Vert
⟨/ab⟩
```

2 The ab.braket module

```
⟨*ab.braket⟩
\ProvidesFile{phy-ab.braket.sty}
  [2023/10/24 `ab.braket' module of physics2]
```

This module requires \phy@abopen and \phy@abclose from ab. This module may have conflict with braket.

```
\phy@requiremodule{ab}
\ifdefined\phy@bra@@
  \PackageWarning{physics2}{You cannot load `ab.braket' and `braket'
    modules together.\MessageBreak Only `ab.braket' module works now.}
\fi
```

\bra \bra < ⟨*subformula*⟩ |

```
\phy@AB@gen{br.m}<{mr<|}{\mathopen#1\langle#2\mathclose#1\vert\endgroup}
\phy@AB@gen{br.a}<{r<|}{\phy@abopen\langle#1\phy@abclose\vert}
\phy@d@l@genxm{br.m}
\phy@d@l@genxa{br.a}
\DeclareRobustCommand\bra{\phy@d@lx{br.m}{br.a}}
```

\ket \ket | ⟨*subformula*⟩ >

```
\phy@AB@gen{kt.m}|{mr|>}{\mathopen#1\vert#2\mathclose#1\rangle\endgroup}
\phy@AB@gen{kt.a}|{r|>}{\phy@abopen\vert#1\phy@abclose\rangle}
\phy@d@l@genxm{kt.m}
\phy@d@l@genxa{kt.a}
\DeclareRobustCommand\ket{\phy@d@lx{kt.m}{kt.a}}
```

\braket \braket < ⟨*subformula 1*⟩ | ⟨*subformula 2*⟩ [| ⟨*subformula 3*⟩ …] >

```
\begingroup
\catcode`\|=\active
\gdef\phy@@mb@bk#1#2{\begingroup
  \mathcode`\|="8000\def|{\egroup#1\vert\bgroup}%
  \def\<{\mathrel{<}}\def\>{\mathrel{>}}%
  \mathopen#1\langle\bgroup#2\egroup\mathclose#1\rangle\endgroup}
\gdef\phy@@ab@bk#1{\begingroup
  \mathcode`\|="8000\def|{\egroup\phy@abb@bkv\bgroup}%
  \def\<{\mathrel{<}}\def\>{\mathrel{>}}%
  \phy@abopen\langle\bgroup#1\egroup\phy@abclose\rangle\endgroup}
\endgroup
\def\phy@abb@bkv{\middle\vert}
\phy@AB@gen{bk.m}<{mr<>}{\phy@@mb@bk#1{#2}\endgroup}
\phy@AB@gen{bk.a}<{r<>}{\phy@@ab@bk{#1}}
\phy@d@l@genxm{bk.m}
\phy@d@l@genxa{bk.a}
\DeclareRobustCommand\braket{\phy@d@lx{bk.m}{bk.a}}
```

\ketbra \braket | ⟨*subformula 1*⟩ > ⟨*subformula 2*⟩ < ⟨*subformula 3*⟩ |

```
\begingroup
\catcode`\<=\active
\catcode`\>=\active
\gdef\phy@@mb@kb#1#2{\begingroup
  \mathcode`\<="8000 \mathcode`\>="8000%
  \def<{#1\langle}\def>{#1\rangle}%
  \def\<{\phy@abb@l}\def\>{\phy@abb@r}%
  \mathopen#1\vert#2\mathclose#1\vert\endgroup}
\endgroup
\gdef\phy@@ab@kb#1>#2<#3\phy@@end{\begingroup
  \def\<{\phy@abb@l}\def\>{\phy@abb@r}%
  \phy@abopen\vert\mathopen{\phy@mathvphantom{#3}}#1\phy@abclose\rangle#2%
  \phy@abopen\langle#3\mathclose{\phy@mathvphantom{#1}}\phy@abclose\vert
\endgroup}
\AtBeginDocument{\ifcsname symbf\endcsname
  \def\phy@abb@l{\Umathchar 3 \symoperators "003C }%
```

```
  \def\phy@abb@r{\Umathchar 3 \symoperators "003E }%
\fi}
\def\phy@abb@l{\mathchar"313C }
\def\phy@abb@r{\mathchar"313E }
\phy@AB@gen{kb.m}|{mr||}{\phy@@mb@kb#1{#2}\endgroup}
\phy@AB@gen{kb.a}|{r||}{\phy@@ab@kb#1\phy@@end}
\phy@d@l@genxm{kb.m}
\phy@d@l@genxa{kb.a}
\DeclareRobustCommand\ketbra{\phy@d@lx{kb.m}{kb.a}}
⟨/ab.braket⟩
```

3 The braket module

```
⟨*braket⟩
\ProvidesFile{phy-braket.sty}
  [2023/10/24 `braket' module of physics2]
```

This module requires \phy@abopen and \phy@abclose from ab. This module may have conflict with ab.braket.

```
\phy@requiremodule{ab}
\ifdefined\phy@abb@bkv
  \PackageWarning{physics2}{You cannot load `ab.braket' and `braket'
    modules together.\MessageBreak Only `braket' module works now.}
\fi
```

\bra \bra * [⟨*biggg*⟩] {⟨*subformula*⟩}

```
\DeclareDocumentCommand\bra{ s o m }{%
  \IfBooleanTF{#1}
    {\mathopen\langle#3\mathclose\vert}
    {\IfValueTF{#2}
      {\csname#2l\endcsname\langle#3\csname#2r\endcsname\vert}
      {\phy@abopen\langle#3\phy@abclose\vert}%
    }%
}
```

\ket \ket * [⟨*biggg*⟩] {⟨*subformula*⟩}

```
\DeclareDocumentCommand\ket{ s o m }{%
  \IfBooleanTF{#1}
    {\mathopen\vert#3\mathclose\rangle}
    {\IfValueTF{#2}
      {\csname#2l\endcsname\vert#3\csname#2r\endcsname\rangle}
      {\phy@abopen\vert#3\phy@abclose\rangle}%
    }%
}
```

\braket \ket * [⟨*biggg*⟩, $n \in \{1, 2, 3\}$] {⟨*subformula 1*⟩} … {⟨*subformula n*⟩}

```
\DeclareDocumentCommand\braket{ s O{} }{%
  \IfBooleanTF{#1}%
    {%
      \gdef\@phy@bk@argnum{ii}%
```

```
      \phy@bk@doopt{#2}%
      \gdef\@phy@bk@l{mathopen}%
      \gdef\@phy@bk@m{mathord}%
      \gdef\@phy@bk@r{mathclose}%
    }%
    {%
      \gdef\@phy@bk@argnum{ii}%
      \gdef\@phy@bk@l{phy@abopen}%
      \gdef\@phy@bk@m{middle}%
      \gdef\@phy@bk@r{phy@abclose}%
      \phy@bk@doopt{#2}%
    }%
  \csname phy@bk@in@\@phy@bk@argnum\endcsname%
}
```

\phy@bk@in@i
\phy@bk@in@ii
\phy@bk@in@iii

\phy@bk@in@⟨*n.roman*⟩ {⟨*subformula 1*⟩} … {⟨*subformula n*⟩}

⟨*n.roman*⟩ is n in roman lowercase, where $n \in \{1, 2, 3\}$.

```
\def\phy@bk@in@i#1{%
  \csname\@phy@bk@l\endcsname\langle{#1}%
  \csname\@phy@bk@r\endcsname\rangle}
\def\phy@bk@in@ii#1#2{%
  \csname\@phy@bk@l\endcsname\langle{#1}%
  \csname\@phy@bk@m\endcsname\vert{#2}%
  \csname\@phy@bk@r\endcsname\rangle}
\def\phy@bk@in@iii#1#2#3{%
  \csname\@phy@bk@l\endcsname\langle{#1}%
  \csname\@phy@bk@m\endcsname\vert{#2}%
  \csname\@phy@bk@m\endcsname\vert{#3}%
  \csname\@phy@bk@r\endcsname\rangle}
```

\phy@bk@doopt
\phy@bk@do@pt

\phy@bk@doopt {⟨*clist*⟩}

Parse the optional argument of \braket. This will add 3 entries to hash.

```
\def\@phy@bk@do@pt#1,{\ifx#1\relax\@empty\else
  \edef\reserved@a{\zap@space#1 \@empty}%
  \ifx\reserved@a\@empty\else
    \ifcsname phy@del\expandafter\string\csname\reserved@a\endcsname\endcsname
      \xdef\@phy@bk@l{\reserved@a l}%
      \xdef\@phy@bk@m{\reserved@a}% but not m (m stands for \mathrel)
      \xdef\@phy@bk@r{\reserved@a r}%
    \else
      \ifnum\reserved@a>3%
        \PackageError{physics2}{\string\braket\space can only take 3
          mandatory arguments at most}{Check if you had written a number
          more than 3 in the [optional] argument.}%
      \fi
      \xdef\@phy@bk@argnum{\romannumeral\reserved@a}%
    \fi
  \fi
  \expandafter\@phy@bk@do@pt\fi}
\def\phy@bk@doopt#1{\@phy@bk@do@pt#1,\relax,}
```

\ketbra `\ketbra * [⟨biggg⟩] {⟨subformula 1⟩} [⟨between 1 and 2⟩] {⟨subformula 2⟩}`

```
\DeclareDocumentCommand\ketbra{ s o m O{} m }{%
  \IfBooleanTF{#1}%
    {\mathopen\vert#3\mathclose\rangle#4\mathopen\langle#5\mathclose\vert}%
    {\IfValueTF{#2}%
      {\csname#2l\endcsname\vert#3\csname#2r\endcsname\rangle#4%
        \csname#2l\endcsname\langle#5\csname#2r\endcsname\vert}%
      {\begingroup
        \phy@abopen\vert\mathopen{\phy@mathvphantom{#5}}#3\phy@abclose\rangle#4%
        \phy@abopen\langle#5\mathclose{\phy@mathvphantom{#3}}\phy@abclose\vert
      \endgroup}%
    }%
}
⟨/braket⟩
```

4 The doubleprod module

```
⟨*doubleprod⟩
\ProvidesFile{phy-doubleprod.sty}
  [2023/10/24 `doubleprod' (vertically stacked binary operators) module of physics2]
```

Boolean options.

```
\phy@define@key{doubleprod}{crosssymbol}{\def\@phy@dbl@c{#1}}
\phy@define@key{doubleprod}{dotsymbol}{\def\@phy@dbl@d{#1}}
\phy@define@key{doubleprod}{crossscale}{\def\@phy@dbl@sc{#1}}
\phy@define@key{doubleprod}{dotscale}{\def\@phy@dbl@sd{#1}}
\phy@define@key{doubleprod}{crossopenup}{\def\@phy@dbl@oc{#1}}
\phy@define@key{doubleprod}{dotopenup}{\def\@phy@dbl@od{#1}}
\phy@setkeys{doubleprod}{crosssymbol=\times,dotsymbol=\ldotp,
  crossscale=0.8,dotscale=1,crossopenup=.02,dotopenup=.2}
\phy@processkeyopt{doubleprod}
\def\phy@dbl@gen#1#2#3#4{%
  \DeclareRobustCommand#1{\mathbin{\vcenter{\baselineskip\z@skip%
    \lineskip#4\phy@dblcurrf@size%
    \setbox\@tempboxa=\hbox{\fontsize{#2\phy@dblcurrf@size}\z@$#3$}%
    \copy\@tempboxa\box\@tempboxa}}}}
\def\phy@dblcurrf@size{\dimexpr\f@size pt\relax}
\phy@dbl@gen\doublecross\@phy@dbl@sc\@phy@dbl@c\@phy@dbl@oc
\phy@dbl@gen\doubledot\@phy@dbl@sd\@phy@dbl@d\@phy@dbl@od
⟨/doubleprod⟩
```

File III
Modules written in LaTeX3 syntax

We use phy as the namespace for physics2 modules.

```
⟨@@=phy⟩
```

1 The diagmat module

```
⟨*diagmat⟩
```

```
\ProvidesExplFile{phy-diagmat.sty}{2024/01/10}{}
  {`diagmat' module of physics2}
\RequirePackage { amsmath }
\phy@define@key { diagmat } { empty } [ 0 ] { \tl_gset:Nn \l__phy_mat_empty_tl { #1 } }
```

This module requires some new variables.

```
\clist_new:N \l__phy_mat_diag_clist
\int_new:N \l__phy_mat_dim_int
\tl_new:N \l__phy_mat_line_tl
\tl_new:N \l__phy_diagmat_tl
\tl_new:N \l__phy_mat_empty_tl
\tl_gset:Nn \l__phy_mat_empty_tl { 0 }
\phy@processkeyopt { diagmat }
\keys_define:nn { phy/diagmat }
  {
    empty .tl_set:N = \l__phy_mat_empty_tl ,
  }
```

\diagmat \⟨*delimiter type*⟩diagmat [⟨*key-val list*⟩] {⟨*diagonal*⟩}

```
\DeclareDocumentCommand \diagmat { O{} m }
  { \__phy_diagmat_type:nnn { } { #1 } { #2 } }
\DeclareDocumentCommand \pdiagmat { O{} m }
  { \__phy_diagmat_type:nnn { p } { #1 } { #2 } }
\DeclareDocumentCommand \bdiagmat { O{} m }
  { \__phy_diagmat_type:nnn { b } { #1 } { #2 } }
\DeclareDocumentCommand \Bdiagmat { O{} m }
  { \__phy_diagmat_type:nnn { B } { #1 } { #2 } }
\DeclareDocumentCommand \vdiagmat { O{} m }
  { \__phy_diagmat_type:nnn { v } { #1 } { #2 } }
\DeclareDocumentCommand \Vdiagmat { O{} m }
  { \__phy_diagmat_type:nnn { V } { #1 } { #2 } }
```

__phy_diagmat_type:nnn __phy_diagmat_type:nnn {⟨*delimiter type*⟩} {⟨*key-val list*⟩} {⟨*diagonal*⟩}

```
\cs_new:Npn \__phy_diagmat_type:nnn #1#2#3
  {
    \group_begin:
    \clist_set:Nn \l__phy_mat_diag_clist { #3 }
    \int_set:Nn \l__phy_mat_dim_int { \clist_count:N \l__phy_mat_diag_clist }
    \int_compare:nNnT { \l__phy_mat_dim_int } > { \value { MaxMatrixCols } }
      { \setcounter { MaxMatrixCols } { \l__phy_mat_dim_int } }
    \keys_set:nn { phy/diagmat } { #2 }
    \tl_gclear:N \l__phy_diagmat_tl
    \int_step_inline:nnn { 1 } { \l__phy_mat_dim_int }
      {
        \int_step_inline:nnn { 1 } { \l__phy_mat_dim_int }
          {
            \int_compare:nNnTF { ##1 } = { ####1 }
              {
                \clist_gpop:NN \l__phy_mat_diag_clist \l__phy_tmpa_tl
                \tl_if_empty:NTF \l__phy_tmpa_tl
                  { \tl_gput_right:Nn \l__phy_mat_line_tl { \l__phy_mat_empty_tl } }
```

Maybe it's better to use \expandafter\scantokens\expandafter{\l_@@_tmpa_tl} in the next line.

```
                  { \tl_gput_right:No \l__phy_mat_line_tl { \l__phy_tmpa_tl } }
              }
              { \tl_gput_right:Nn \l__phy_mat_line_tl { \l__phy_mat_empty_tl } }
            \int_compare:nNnTF { ####1 } = { \l__phy_mat_dim_int }
              {
                \tl_gput_right:Nn \l__phy_mat_line_tl { \\ }
              }
              {
                \tl_gput_right:Nn \l__phy_mat_line_tl { & }
              }
          }
        \tl_gput_right:No \l__phy_diagmat_tl { \l__phy_mat_line_tl }
        \tl_gclear:N \l__phy_mat_line_tl
      }
    \begin { #1 matrix }
      \tl_use:N \l__phy_diagmat_tl
    \end { #1 matrix }
    \group_end:
  }
```

⟨/diagmat⟩

2 The xmat module

```
⟨*xmat⟩
\ProvidesExplFile{phy-xmat.sty}{2023/10/24}{}
  {`xmat' module of physics2}
\RequirePackage { amsmath }
\phy@define@key { xmat } { showtop }
  { \int_gset:Nn \l__phy_xmat_showtop_int { #1 } }
\phy@define@key { xmat } { showleft }
  { \int_gset:Nn \l__phy_xmat_showleft_int { #1 } }
```

This module requires some new variables.

```
\bool_new:N \l__phy_xmat_extra_vdots_bool
\bool_new:N \l__phy_xmat_extra_cdots_bool
\int_new:N \l__phy_xmat_showtop_int
\int_new:N \l__phy_xmat_showleft_int
\tl_new:N \l__phy_xmat_tl
\int_gset:Nn \l__phy_xmat_showtop_int { \value { MaxMatrixCols } - 2 }
\int_gset:Nn \l__phy_xmat_showleft_int { \value { MaxMatrixCols } - 2 }
\cs_new:Npn \__phy_xmat_entry_format:nnn #1#2#3
  {
    #1 \c_math_subscript_token { #2 #3 }
  }
\phy@processkeyopt { xmat }
\DeclareDocumentCommand \xmat { O{} m m m }
 { \__phy_xmat_type:nnnnn { } { #1 } { #2 } { #3 } { #4 } }
\DeclareDocumentCommand \pxmat { O{} m m m }
 { \__phy_xmat_type:nnnnn { p } { #1 } { #2 } { #3 } { #4 } }
\DeclareDocumentCommand \bxmat { O{} m m m }
 { \__phy_xmat_type:nnnnn { b } { #1 } { #2 } { #3 } { #4 } }
\DeclareDocumentCommand \Bxmat { O{} m m m }
 { \__phy_xmat_type:nnnnn { B } { #1 } { #2 } { #3 } { #4 } }
```

```
\DeclareDocumentCommand \vxmat { O{} m m m }
 { \__phy_xmat_type:nnnnn { v } { #1 } { #2 } { #3 } { #4 } }
\DeclareDocumentCommand \Vxmat { O{} m m m }
 { \__phy_xmat_type:nnnnn { V } { #1 } { #2 } { #3 } { #4 } }
\keys_define:nn { phy/xmat }
  {
    format  .cs_set:Np = \__phy_xmat_entry_format:nnn #1#2#3 ,
    showtop .int_set:N = \l__phy_xmat_showtop_int ,
    showleft.int_set:N = \l__phy_xmat_showleft_int ,
  }
```

`\__phy_if_digits_only_p:n` ⋆ `\__phy_if_digits_only:nTF` {⟨*token list*⟩} {⟨*true code*⟩} {⟨*flase code*⟩}
`\__phy_if_digits_only:n`*TF* ⋆

Use LaTeX3 regular expression to tell if ⟨*token list*⟩ (the numbers of rows or columns) contain digits only.

```
\prg_new_conditional:Npnn \__phy_if_digits_only:n #1 { TF }
  {
    \regex_match:nnTF { \A [[:digit:]]* \Z } { #1 }
      { \prg_return_true: } { \prg_return_false: }
  }
```

`\__phy_xmat_type:nnnnn` `\__phy_xmat_type:nnnnn` {⟨*delimiter type*⟩} {⟨*key-val list*⟩} {⟨*common entry*⟩} {⟨*row number*⟩} {⟨*column number*⟩}

```
\cs_new:Npn \__phy_xmat_type:nnnnn #1#2#3#4#5
  {
    \group_begin:
    \tl_gclear:N \l__phy_xmat_tl
    \keys_set:nn { phy/xmat } { #2 } %
    \__phy_if_digits_only:nTF { #4 }
      {
        \int_compare:nNnTF { #4 } < { \l__phy_xmat_showtop_int + 1 }
          {
            \int_set:Nn \l__phy_xmat_showtop_int { #4 }
            \bool_set_false:N \l__phy_xmat_extra_vdots_bool
          }
          {
            \bool_set_true:N \l__phy_xmat_extra_vdots_bool
          }
      }
      {
        \bool_set_true:N \l__phy_xmat_extra_vdots_bool
      }
    \__phy_if_digits_only:nTF { #5 }
      {
        \int_compare:nNnTF { #5 } < { \l__phy_xmat_showleft_int + 1 }
          {
            \int_set:Nn \l__phy_xmat_showleft_int { #5 }
            \bool_set_false:N \l__phy_xmat_extra_cdots_bool
          }
          {
            \bool_set_true:N \l__phy_xmat_extra_cdots_bool
```

```
          }
      }
      {
        \bool_set_true:N \l__phy_xmat_extra_cdots_bool
      }
    \int_step_inline:nn { \l__phy_xmat_showtop_int }
      {
        \tl_put_right:Nn \l__phy_xmat_tl
          { \__phy_xmat_entry_format:nnn { #3 } { ##1 } { 1 } }
        \int_step_inline:nnn { 2 } { \l__phy_xmat_showleft_int }
          {
            \tl_put_right:Nn \l__phy_xmat_tl
              { & \__phy_xmat_entry_format:nnn { #3 } { ##1 } { ####1 } }
          }
        \bool_if:NT \l__phy_xmat_extra_cdots_bool
          {
            \tl_put_right:Nn \l__phy_xmat_tl
              { & \cdots & \__phy_xmat_entry_format:nnn { #3 } { ##1 } { #5 } }
          }
        \tl_put_right:Nn \l__phy_xmat_tl { \\ }
      }
    \bool_if:NT \l__phy_xmat_extra_vdots_bool
      {
        \tl_put_right:Nn \l__phy_xmat_tl { \vdots }
        \prg_replicate:nn { \l__phy_xmat_showleft_int - 1 }
          {
            \tl_put_right:Nn \l__phy_xmat_tl { & \vdots }
          }
        % Add \ddots if vdots_bool and cdots_bool be true simultaneously.
        \bool_if:NT \l__phy_xmat_extra_cdots_bool
          {
            \tl_put_right:Nn \l__phy_xmat_tl { & \ddots & \vdots }
          } % else relax
        \tl_put_right:Nn \l__phy_xmat_tl { \\ }
        % The last row.
        \tl_put_right:Nn \l__phy_xmat_tl
          { \__phy_xmat_entry_format:nnn { #3 } { #4 } { 1 } }
        \int_step_inline:nnn { 2 } { \l__phy_xmat_showleft_int }
          {
            \tl_put_right:Nn \l__phy_xmat_tl
              { & \__phy_xmat_entry_format:nnn { #3 } { #4 } { ##1 } }
          }
        \bool_if:NT \l__phy_xmat_extra_cdots_bool
          {
            \tl_put_right:Nn \l__phy_xmat_tl
              { & \cdots & \__phy_xmat_entry_format:nnn { #3 } { #4 } { #5 } }
          }
      } % else relax
    \begin { #1 matrix }
      \tl_use:N \l__phy_xmat_tl
    \end { #1 matrix }
    \group_end:
  }
⟨/xmat⟩
```

This part ends here.
126 ⟨@@=⟩

File IV
Legacy modules written in LaTeX 2ε syntax

1 The ab.legacy module

1 ⟨*ab.legacy⟩
```
\ProvidesFile{phy-ab.legacy.sty}
  [2023/10/24 `ab.legacy' module of physics2]
```
Requires ab's `tight` option.
```
\phy@requiremodule{ab}
\phy@define@key{ab.legacy}{order}[\mathcal{O}]{\def\phy@ab@ordersym{#1}}
\phy@setkeys{ab.legacy}{order}
\phy@processkeyopt{ab.legacy}
\phy@d@l@geny\abs\vert\vert
\phy@d@l@geny\norm\Vert\Vert
\DeclareDocumentCommand\order{som}{%
  \phy@ab@ordersym
  \IfBooleanTF{#1}
    {(#3)}
    {\IfValueTF{#2}
      {\csname#2l\endcsname(#3\csname#2r\endcsname)}
      {\phy@abopen(#3\phy@abclose)}%
    }%
}
\phy@d@l@geny\eval.\vert
\phy@d@l@geny\peval(\vert
\phy@d@l@geny\beval[\vert
```
⟨/ab.legacy⟩

2 The nabla.legacy module

1 ⟨*nabla.legacy⟩
```
\ProvidesFile{phy-nabla.legacy.sty}
  [2023/10/24 `nabla.legacy' module of physics2]
\phy@requiremodule{ab}
```
Requires fixdif version 2.x.
```
\RequirePackage{fixdif}[2023/01/31]
\letdif\phy@nl@nabla{nabla}
\AtBeginDocument{\ifcsname div\endcsname\let\divsymbol\div\fi
  \DeclareRobustCommand\grad{\phy@nl@nabla\ab}%
  \DeclareRobustCommand\div{\phy@nl@nabla\cdot\ab}%
  \DeclareRobustCommand\curl{\phy@nl@nabla\times\ab}%
  \DeclareRobustCommand\laplacian{\phy@nl@nabla^2\ab}%
}
```
⟨/nabla.legacy⟩

3 The op.legacy module

```
⟨*op.legacy⟩
\ProvidesFile{phy-op.legacy.sty}
  [2023/10/24 `op.legacy' module of physics2]
\phy@define@key{op.lega}{ReIm}[true]{\def\phy@reserveda{#1}}
\phy@define@key{op.lega}{PV}{\def\@phy@oplega@PV{#1}}
\phy@define@key{op.lega}{pv}{\def\@phy@oplega@pv{#1}}
\phy@setkeys{op.lega}{PV=\mathcal{P},pv={p.v.},ReIm=true}
\phy@processkeyopt{ab}
\DeclareRobustCommand\asin{\mathop{\operator@font asin}\nolimits}
\DeclareRobustCommand\acos{\mathop{\operator@font acos}\nolimits}
\DeclareRobustCommand\atan{\mathop{\operator@font atan}\nolimits}
\DeclareRobustCommand\acsc{\mathop{\operator@font acsc}\nolimits}
\DeclareRobustCommand\asec{\mathop{\operator@font asec}\nolimits}
\DeclareRobustCommand\acot{\mathop{\operator@font acot}\nolimits}
\DeclareRobustCommand\Tr{\mathop{\operator@font Tr}\nolimits}
\DeclareRobustCommand\tr{\mathop{\operator@font tr}\nolimits}
\DeclareRobustCommand\rank{\mathop{\operator@font rank}\nolimits}
\DeclareRobustCommand\erf{\mathop{\operator@font erf}\nolimits}
\DeclareRobustCommand\Res{\mathop{\operator@font Res}\nolimits}
\DeclareRobustCommand\res{\mathop{\operator@font res}\nolimits}
\DeclareRobustCommand\PV{\mathord{\@phy@oplega@PV}}
\DeclareRobustCommand\pv{\mathop{\operator@font\@phy@oplega@pv{}}\nolimits}
```

Restore \Re and \Im in \Resymbol and \Imsymbol. The \AtBeginDocument hook is used for the compatibility of unicode-math.

```
\ifx\phy@reserveda\phy@true
\AtBeginDocument{%
  \let\Resymbol\Re%
  \let\Imsymbol\Im%
  \DeclareRobustCommand\Re{\mathop{\operator@font Re}\nolimits}%
  \DeclareRobustCommand\Im{\mathop{\operator@font Im}\nolimits}%
}
\fi
⟨/op.legacy⟩
```

4 The qtext.legacy module

This module is written for the compatibility with the bad commands provided by physics only. The commands in this module should NEVER be used!

```
⟨*qtext.legacy⟩
\ProvidesFile{phy-qtext.legacy.sty}
  [2023/10/24 `qtext.legacy' module of physics2.sty]
\RequirePackage{amstext}
\def\phy@qtext@#1#2{#1\text{#2}\quad}
\DeclareRobustCommand\qqtext{\@ifstar{\phy@qtext@{}}{\phy@qtext@\quad}}
\DeclareRobustCommand\qq{\qqtext}
\DeclareRobustCommand\qcomma{,\quad}
\DeclareRobustCommand\qc{\qcomma}
\DeclareRobustCommand\qcc{\@ifstar{\phy@qtext@{}{c.c}}{\phy@qtext@\quad{c.c}}}
\def\phy@qtext@lega@gen@#1{%
  \expandafter\DeclareRobustCommand\csname q#1\endcsname%
    {\@ifstar{\phy@qtext@{}{#1}}{\phy@qtext@\quad{#1}}}}
```

```
\phy@qtext@lega@gen@{if}
\phy@qtext@lega@gen@{then}
\phy@qtext@lega@gen@{else}
\phy@qtext@lega@gen@{otherwise}
\phy@qtext@lega@gen@{unless}
\phy@qtext@lega@gen@{give}
\phy@qtext@lega@gen@{using}
\phy@qtext@lega@gen@{unless}
\phy@qtext@lega@gen@{assume}
\phy@qtext@lega@gen@{since}
\phy@qtext@lega@gen@{let}
\phy@qtext@lega@gen@{for}
\phy@qtext@lega@gen@{all}
\phy@qtext@lega@gen@{even}
\phy@qtext@lega@gen@{odd}
\phy@qtext@lega@gen@{integer}
\phy@qtext@lega@gen@{and}
\phy@qtext@lega@gen@{or}
\phy@qtext@lega@gen@{as}
\phy@qtext@lega@gen@{in}
⟨/qtext.legacy⟩
```

File V
Legacy modules written in LaTeX3 syntax

```
⟨@@=phy⟩
```

1 The bm-um.legacy module

```
⟨*bm-um.legacy⟩
\ProvidesExplFile{phy-bm-um.legacy.sty}{2023/10/24}{}
  {`bm-um.legacy' module of physics2}
\AtBeginDocument
  {
    \cs_if_exist:cF { symbf }
      {
        \PackageError { physics2 }
          {
            The ~ `bm-um.legacy' ~ module ~ requires ~
            `unicode-math' ~ package
          }
          {
            Have ~ you ~ used ~ `unicode-math' ~
            in ~ the ~ preamble?
          }
      }
  }
\DeclareDocumentCommand \bm { m }
  {
```

```
    \mode_if_math:TF
      {
        \tl_if_head_eq_catcode:nNTF { #1 } A
          {
            \symbfit { #1 }
          }
          {
            \symbf { #1 }
          }
      }
      {
        \PackageError { physics2 }
          {
            The ~ \string\bm\space command ~ should ~ be ~
            used ~ in ~ math ~ mode ~ only. \MessageBreak
            This ~ is ~ an ~ error ~ from ~ `bm-um.legacy' ~ module
          }
          {
            Check ~ if ~ any ~ `\string\bm' ~ is ~ out ~
            of ~ math ~ mode.
          }
      }
  }
```
⟨/bm-um.legacy⟩

This part ends here.

⟨@@=⟩