

The eqnlines Package

Niklas Beisert

Institut für Theoretische Physik
Eidgenössische Technische Hochschule Zürich
Wolfgang-Pauli-Strasse 27, 8093 Zürich, Switzerland

`nbeisert@itp.phys.ethz.ch`

2025/03/27, v0.6.1

<https://ctan.org/pkg/eqnlines>

– preview version –

Abstract

eqnlines is a $\text{\LaTeX} 2_{\epsilon}$ package providing a framework for typesetting single- and multi-line equations which extends the established equation environments of \LaTeX and the `amsmath` package with many options for convenient adjustment of the intended layout. In particular, the package adds flexible schemes for numbering, horizontal alignment and semi-automatic punctuation, and it improves upon the horizontal and vertical spacing options. The extensions can be used and adjusted through optional arguments and modifiers to the equation environments as well as global settings.

Contents

1	Introduction	3
2	Usage	4
2.1	Equations Environment	5
2.2	Numbering	6
2.3	Horizontal Adjustment	7
2.4	Punctuation	8
2.5	Math Classes at Alignment	9
2.6	Vertical Spacing	9
2.7	Further Environments	11
2.8	General Options	11
2.9	Feature Selection and Package Options	12
3	Information	12
3.1	Copyright	12
3.2	Credits	13
3.3	Files and Installation	13
3.4	Related CTAN Packages	14
3.5	Feature Suggestions	14
3.6	Revision History	14
A	Implementation	15

B	General Support	15
	B.1 Development Messages	15
	B.2 Supporting Definitions	15
	B.3 Dollardollar Abstraction	16
	B.4 Look-Ahead in Alignment	16
	B.5 Error Messages	18
	B.6 amsmath Integration	18
	B.7 Tagging Support	19
C	Parameters and Registers	20
	C.1 Parameters	20
	C.2 Registers	23
	C.3 Hooks	25
D	Punctuation	25
E	Math Classes at Alignment	27
F	Equations Box Environment	28
G	Equation Numbering	32
	G.1 Tag Formatting	32
	G.2 Showkeys Integration	33
	G.3 Labels	34
	G.4 Tags	34
	G.5 Anchors	35
	G.6 Tag Composition	35
	G.7 Tagbox Methods	37
	G.8 Numbering Schemes	38
	G.9 Numbering Framework	40
H	Subequation Numbering	44
	H.1 Definitions	44
	H.2 Environment	45
	H.3 Subequation Scheme	47
I	Display Equations Support	47
	I.1 Display Breaks	47
	I.2 General Initialisation	48
	I.3 halign Support	52
	I.4 Stack	53
J	Horizontal Spacing for Lines	55
	J.1 Supporting Definitionss	55
	J.2 Shape Schemes	56
	J.3 Adjustment Methods	58
	J.4 Central Alignment Layout	61
	J.5 Left Alignment Layout	65
K	Single-Line Equation	69
	K.1 Environment	69
	K.2 Native	70
	K.3 Adjustment	70
L	Multi-Line Support	71

L.1	Registers	71
L.2	Measure Support	72
L.3	Print Support	73
L.4	Line Breaks	74
L.5	Intertext	74
L.6	Main	76
M	Multi-Line Lines Mode	78
M.1	Measure	78
M.2	Print	80
N	Multi-Line Align Mode	81
N.1	Columns Processing	81
N.2	Measure	82
N.3	Print	84
N.4	Adjust	88
O	Interface	94
O.1	Scanning the Equation Body	94
O.2	Options Processing	97
O.3	Equations Environment	97
O.4	Independent Routines of amsmath	100
P	Options	100
P.1	Selection Tools	101
P.2	Declaration Code	102
P.3	Parameter Sets	102
P.4	Options Declarations	104
P.5	Component Selection	110
P.6	Global and Package Options	116

1 Introduction

Typesetting mathematical equations is an undisputed strength of $\text{T}_{\text{E}}\text{X}$. $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ improved the overall management of display equations, for instance by providing optional numbering. It also added elementary functionality for multi-lined equations with alignment. Some of its deficiencies were addressed by the multi-line equation environments of the package `amsmath` which have become an established standard for these purposes.

The package `eqnlines` builds upon and extends the functionality of the $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ and `amsmath` equation environments with some new features as well as convenient options to adjust the layout where needed. The main additions are as follows:

- Equation numbers can be assigned to individual lines (as for `align` and `gather`) or once for the multi-line equation block (as for `multiline`). In the former case, a sub-numbering scheme can be applied (as through `subequations`). In the latter case, the position can be assigned to a specific line (`first/middle/last/chosen`). Moreover, equation numbers can be turned on and off by commands, and they can be triggered by setting a label.
- The vertical spacing above and below single- and multi-line equations of $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ and `amsmath` can be somewhat variable, hard to control and even resistive in certain situations. The package implements clearer structures controlling the vertical spacing, including proper dependency on the text line above and ways to adjust the spacing.

- The framework introduces a scheme which semi-automatically inserts punctuation, e.g. ‘.’ or ‘;’, at the end of the following (or every) equation environment. Punctuation can also be inserted at every alignment column or equation line including the possibility to prepend a certain spacing.
- Next to `\[...\]` as an alias for the single-line `equation` environment, the package uses `\<...\>` as an alias multi-line equations.
- The horizontal alignment and indentation of equation lines can be adjusted via a scheme or on a line-by-line basis.
- The alignment marker can be placed before or after the equation signs while maintaining proper spacing to symbols before and after it. This simplifies the construction of continuing equations in an aligned context.
- Equation lines are subject to shrinking of space if the available space does not suffice (analogously to single-line equations).
- Most settings can be controlled via optional arguments and modifiers to the equation environment or via global settings. This includes switching between different types of equation environments, enabling or disabling numbering, adjusting vertical spacing, etc. This feature simplifies the adjustment and fine-tuning of equations towards the intended layout.
- Last but not least, the underlying `amsmath` code, originating from the `TEX` era and early `LATEX` years, has been redesigned with emphasis on clarity, readability, adjustability and maintainability (but at the cost of moderately higher resource consumption and moderately lower efficiency). Nevertheless, it remains `LATEX 2ε` code.

The package represents a stand-alone implementation of an equations environment which is largely compatible with the established `LATEX` and `amsmath` environments `equation`, `multline`, `gather`, `align` and their variants. Hence, the package can be used instead of `amsmath` with no or minor modifications to the `LATEX` sources for single- and multi-line equations. It can also be used alongside `amsmath` including the `mathtools` extensions to make use of the additional maths typesetting features provided by these packages. In the latter case, the equation environments of `LATEX` and `amsmath` are either replaced or left in place while the `eqnlines` environments can be accessed using the alternate name `equations`.

2 Usage

Notice regarding package version v0.6.1: Please note that this package is still in a development and testing stage in the present version. This mainly applies to the documentation of features and code: Currently, the documentation is basic and minimal without extensive coverage of all features and settings, and it lacks desirable illustrations and examples.

It is likely that some features of the package do not work to full extent, and that the package will not cooperate well with other packages. Therefore, please report any malfunctions that you may notice.

Therefore, it is likely that internal macros and mechanisms will change, It is also conceivable that the public interface will change in minor but relevant ways in order to accommodate for important adjustments or additional features. It is intended that such changes would only require minor adaption of document sources that use an early version of this package.

To use the `eqnlines` package add the command

```
\usepackage{eqnlines}
```

to the preamble of the L^AT_EX document. To use unrelated features of the `amsmath` package or of the `mathtools` extension, it makes sense to load these packages *before* `eqnlines`.

2.1 Equations Environment

`equations` (*env.*) The package supplies a main maths environment called `equations` which accepts a comma-separated list of optional parameters ‘`[opts]`’:

```
\begin{equations}[opts]_
...
\end{equations}
```

Furthermore, the environment accepts some modifiers (like the star modifier ‘`*`’ for many other L^AT_EX macros) which will be explained further below. These follow the scheme { `!t~ !t* !t! !o !e{@}` } according to the syntax of `\NewDocumentCommand`.

We note that the `equations` environment should be started with a whitespace character ‘`_`’ which provides a clear separation from optional arguments ‘`[opts]`’ and/or modifiers which must immediately follow the environment declaration `\begin{equations}` without whitespaces.

`single` (*key*) The environment has three principal modes of operation which are selected by setting an optional argument as follows:

`lines` (*key*)

`columns` (*key*)

purpose	single-line equation	stacked equation(s)	aligned equations
name	<code>single</code>	<code>lines</code>	<code>columns</code>
alt. names	<code>equation, eq, 1</code>	<code>gather, ga, ln, ~</code>	<code>align, al, col, @</code>
symbolic	<code>\[...]</code>	<code>\<~...></code>	<code>\<...></code>
<code>amsmath</code> env.	<code>equation</code>	<code>gather, multiline</code>	<code>align</code>
columns	—	single	multiple, aligned
alignment	adjustable	adjustable	alternating right/left
parsing	single, direct	two passes	two passes
numbering	on/off	off/single/multiple	off/single/multiple

The aligned mode more or less encompasses all three modes, and the stacked mode with only a single line is more or less just a single equation. However, the more complex forms also come along with some restrictions, hence, it makes sense to use the appropriate mode for the intended equation content. For instance, a single equation simply reads the equation input once, while the multi-line equation environments parse the environment body twice which can potentially disrupt some other functionality that is included in the body. Furthermore, the horizontal adjustment options are very restricted in aligned mode, and therefore the aligned form can automatically reduce to the stacked form (with right alignment) if only a single column is provided (no ‘`&`’s).

`\[...]` The package offers several alternative names for the same mode as well as a symbolic short `\<...>` form `\<...>` extending the L^AT_EX display equation form `\[...]` to multi-line equations.

`~` (*key*) Here, the tilde ‘`~`’ in `\<~...>` is a modifier character which acts as a short form for the

`spropt` (*key*) optional argument `lines` selecting the lines mode. Both short forms can be customised by

`angopt` (*key*) setting default arguments via the global options `spropt={opts}` and `angopt={opts}`. Both default arguments are preset to `nonumber` which disables equation numbering, see section 2.2.

`equation` (*env.*) The package also supplies or overwrites the `amsmath` environments `equation`, `multiline`,

`multiline` (*env.*) `gather`, `align` and `flalign` including their starred variants but neither the `alignat` alter-

`gather` (*env.*) native forms nor `split`. It is possible to define further equation environments *env* with a

`align` (*env.*) predefined set of options *opts* using:

`\[re]newenvironment{env}{\eqnadopt{opts}\equations}{\endequations}`

2.2 Numbering

`numberline` (*key*) The package extends the established interface of L^AT_EX and the `amsmath` package for labelling equations with numbers or with manually assigned tags. For multi-line equations, there are two distinct modes of operations: individual labelling of the equation lines or one overall number/tag for the whole block of equations. The modes are selected by an optional argument `numberline=mode` as follows:

name	alt.	description	preset
<code>none</code>	<code>n</code>		preset off
<code>all</code>	<code>a</code>	individual lines	preset on
<code>sub</code>	<code>s</code>		subequations (a, b, c, ...)
<code>first</code>	<code>f</code>		first line
<code>last</code>	<code>l</code>		last line
<code>middle</code>	<code>m</code>	single number	middle line
<code>out</code>	<code>o</code>		last/first line for right/left tags
<code>in</code>	<code>i</code>		first/last line for right/left tags
<code>here</code>	<code>h</code>		line indicated by <code>\numberhere</code>

`\nonumber` Numbering can be turned on and off (for individual lines or for the block as a whole depending on the mode) by means of:

`\nonumber` and `\donumber`

* (*key*) Alternatively, the numbering can be disabled or enabled for the block using modifiers (which ! (*key*) must be placed *before* further optional arguments):

`\[*_...]` and `\[!_...]`

This allows to define a default behaviour and specify exceptions where they may occur. The star modifier following directly the environment declaration replaces the starred form of environments (`equation*`, etc.).

`\numberhere` The placement of a single number for an equation block can be adjusted by:
`\numbernext`

`\numberhere` and `\numbernext`

The former macro overrides the position to the present line. The latter macro defers the number to the next line, e.g. if an equation is broken into several lines and the last one should receive the number tag.

`\label` Equation numbers can receive L^AT_EX labels as usual and they can be turned into manually assigned tags using the established macros:

`\label{label}` and `\tag*{tag}`

Note that a label and a tag will always apply to the next number that will be printed, and only a single label and/or tag may be specified for it. For example, if the present line has no numbering, but the following line does, `\label` or `\tag` will apply to the following line. The macros `\label` and `\tag` can also be instructed to automatically enable numbering for the present line or block (`\donumber`). By default, numbers are triggered for `\tag`, but not for `\label` reflecting the behaviour set forth by `amsmath`. By enabling triggering for `\label`, numbers will be produced only if they have a chance of being referenced.

`label` (*key*) The equations environment provides an alternative means to specify labels and tags within
`tag` (*key*) the optional arguments [*opts*] or via the modifier `@{label}` (which may follow further op-
`@` (*key*) tional arguments):

$$\text{label}=\{label\}, \quad \text{tag}[*]=\{tag\}, \quad \backslash[@\{label\} \dots \backslash]$$

In particular, in subequations mode (`sub`), the optional argument `label` can be used to assign a label to the parent number addressing the whole equation block.

`\raisetag`

`taglayout` (*key*) The typesetting of equation numbers and tags passes through two macros, one which defines
`tagform` (*key*) the layout and another one which adds a decoration (brackets). These two methods can be adjusted via the options:

$$\text{taglayout}[*]=\{code\} \quad \text{and} \quad \text{tagform}=\{l\{code\}r\} \quad \text{or} \quad \text{tagform}*\{code\}$$

Here, *code* is some macro code that references the argument ‘#1’ containing the number or tag, and *l* and *r* can be opening and closing brackets for the tag presentation.

`\eqref` The macro `\eqref` is the standard method for referring to equation numbers via their label. This method also uses the layout defined above.

2.3 Horizontal Adjustment

`margin` (*key*) The horizontal alignment of columns is fixed for aligned multi-line equations: Each pair of
`mincolsep` (*key*) subsequent columns forms a unit which is aligned at the intermediate alignment marker ‘&’.
`maxcolsep` (*key*) These columns are distributed evenly over the available horizontal space. Here, the outer space left and right of the set of columns is treated on equal footing to the space between the columns (option `margins=on`), but it can be eliminated so that the outer columns are pushed right to the margin (option `margins=off`). In addition, a minimum and maximum width can be specified for the column separation (`mincolsep=dimen` and `maxcolsep=dimen`). By default, no maximum column separation is set (`maxcolsep*`), and all horizontal space is used, otherwise a value of `2em` ($\equiv \backslashqqquad$) is suggested with the minimum separation set to `1em` ($\equiv \backslashquad$) by default.

For stacks of equations including single equations, there is just a single alignment column whose horizontal alignment can be adjusted via a shape scheme or by manually adjusting individual lines. A shape scheme determines the horizontal alignment for each line and it is specified by the optional argument `shape=mode` as follows:

name	alt.	shape	alignment
<code>default</code>	<code>def</code>	uniform	default
<code>left</code>	<code>l</code>		left
<code>center</code>	<code>c</code>	uniform	central
<code>right</code>	<code>r</code>		right
<code>first</code>	<code>indent, rc</code>	first/rest	first line indented
<code>hanging</code>	<code>outdent, lc</code>	first/rest	first line hanging
<code>steps</code>	<code>lcr</code>	first/intermediate/last	left/centre...centre/right

Note that the `steps` shape comes to use in the `amsmath` environment `multline`. The alignment preset can be adjusted for individual lines by the macros:

$$\backslashshoveleft[*]!|[*dimen*], \quad \backslashshovecenter, \quad \backslashshoveright$$

In contradistinction to `amsmath`, these macros do not require to specify the cell contents `indent` (*key*) as their argument (but there is no harm in doing so). The macro `\shoveleft` also accepts

the modifiers ‘*’ or ‘!’ for indentation or hanging indentation by the standard indentation amount (`indent=2em`) or an optional argument [*indent*] specifying a variable amount of indentation.

`padding` (*key*) Note that (hanging) indentation requires to add some padding around the equations block via `padleft` (*key*) the optional argument `padding|padleft|padright[={dimen}]` or `padmax` to extend padding `padright` (*key*) to the whole line. Note that `indent*={dimen}` sets the default indentation amount and the `padmax` (*key*) left padding at the same time.

`layout` (*key*) Finally, the overall layout can be adjusted between central and left alignment via `center` (*key*) `layout=center`, `layout=left` or `center`, `left` for short.

`left` (*key*) In central alignment layout, there is the option of imposing a tag margin `tagmargin={dimen}` `tagmargin` (*key*) which allocates some space to the tag such that equation content is centred in the remaining `tagmargin*` (*key*) horizontal space. The margin can also be set to the width of some text by `tagmargin*={text}` `tagmargincalc` (*key*) or it can be calculated as the maximum width of tags by `tagmargincalc`. The option `tagmarginratio` (*key*) `tagmarginratio={ratio}` uses the tag margin only for equation blocks with a ratio of tags to rows above the given (decimal) ratio (a value above 1 uses the tag margin only for single equations with tags). The option `tagmarginthreshold={threshold}` uses the tag margin only if the ratio of spacings is below the given (decimal) threshold. The latter two options together with some tag margin can produce a more appealing layout for equation blocks of mixed filling.

`leftmargin` (*key*) In left alignment layout, all equations are left aligned to a left margin (`leftmargin` is `leftmargin*` (*key*) initialised to the first level of enumerations and itemisations). It can be set to the width `minleftmargin` (*key*) of some text by `leftmargin*={text}`. Depending on the situation, the left margin may be `maxleftmargin` (*key*) reduced or extended to `minleftmargin` or `maxleftmargin`, respectively.

`marginbadness` (*key*) Finally, we note that within single and stacked equations, very long equations that do not `maxbadness` (*key*) fit the available horizontal space are subject to shrinking attempts. In other words, \TeX will attempt to shrink the glue contained in the equation line to make it fit. This shrinking can be controlled by the two parameters `marginbadness` and `maxbadness` accepting integer values. The former is used for trying to shrink onto certain horizontal margins which are otherwise reserved for tags; the latter is used for using the maximal horizontal space which also raises or lowers the equation tag if needed. Small values prevent shrinking and higher values allow for more compression.

2.4 Punctuation

Extending proper punctuation across equations is a delicate matter, and maintaining it while redacting the text certainly takes more attention to detail than many author are willing to afford. A contributing factor is that punctuation marks are harder to spot alongside equation context and somewhat out of place anyway.

`\eqnpunctmain` The package supplies a semi-automatic scheme by which equations are terminated by a `\eqnpunct` specific punctuation mark. Punctuation marks are set by:

`punct` (*key*)

```

\eqnpunctmain{punct}      \eqnlineset{punct={punct}}
\eqnpunct{punct}         \eqnaddopt{punct={punct}}
\[[punct={punct}] ... \]

```

The former two forms set and enable a default punctuation mark; the middle two forms set the punctuation mark for the next equation environment in line; the final form applies to the equation environment only. For example, one might declare ‘`\eqnpunctmain.`’ to terminate all equations with a period ‘.’. The default behaviour can be adjusted to a comma ‘,’ for an individual equation by declaring ‘`\eqnpunct,`’ before the equation (i.e. at the end of the textual phrase to which the punctuation mark belongs) or by using the optional argument [`punct={,}`]. Likewise, `\eqnpunct{}` and [`punct={}`] eliminate a preset punctuation.

`punctsep` (*key*) For convenience, one may also specify a desired space (or any other sequence) preceding the punctuation by [`punctsep={sep}`], e.g. `sep=\`, or `sep=_`.

`\eqnpunctcol` For multi-line equations, there are two further levels of default punctuation for terminating columns and lines which are specified via the macros `\eqnpunctcol` and `\eqnpunctline` or `punctcol` (*key*) the optional arguments `punctcol` and `punctline`. A punctuation item may also be handed on to the next lower level of punctuation via the starred forms `punct*` and `punctline*`.

2.5 Math Classes at Alignment

Alignment in multi-line equations breaks equations into components before and after the alignment position. Unfortunately, this also interrupts $\text{T}_{\text{E}}\text{X}$'s math spacing mechanism which is based on the math classes assigned to the characters, and there appears to be no direct way of determining the math class to the previous letter. Therefore, one has to make some assumptions on the letters that will surround the alignment marker ‘&’ in order to obtain the appropriate spacing also across the alignment.

The `amsmath` environment `align` assumes that the left column ends with an ordinary character. This leads to the correct spacing when an equation $a = b + c$ is broken before the equals relation as `a&=b+c`, and also if an equation sequence continues on the next line as `\&=d-e`. However, it is difficult to achieve the right spacing if the right-hand side is to be broken into several lines: For instance, `\&_+f` aligns the subordinate binary operation with the equals sign (which may be undesirable). Instead placing a phantom equals sign is an effort that somewhat disrupts the readability of the code.

`class` (*key*) The package implements a more flexible assignment of math classes at the alignment. The `ampeq` (*key*) above default behaviour is invoked by the optional argument `class=ampeq` (or `ampeq` for `eqamp` (*key*) short). The optional argument `class=eqamp` (or `eqamp` for short) imposes math classes at the alignment such that an equation sign should be placed just before the alignment. Concretely, it inserts `\mathrel{}` classes just before and after the alignment marker. Furthermore, in case of an empty left alignment cell, the leading math class is changed to `\mathord{}` so that a following binary operator is not interpreted as a unary one.

`classout` (*key*) Math classes just before and after alignment can be adjusted freely by the optional arguments:
`classin` (*key*)
`classlead` (*key*)

`classout={class}, classin={class}, classlead={class}.`

The parameter `classlead` determines the math class just after the alignment if the cell before alignment is empty.

For example, the following two expressions produce identical output:

```
\<[ampeq] a &= b+c \ \ &= d-e \ \ &\mathrel{\phantom{=}} +f \> |
\<[eqamp] a =& b+c \ \ =& d-e \ \ & +f \>
```

2.6 Vertical Spacing

Display equations in $\text{T}_{\text{E}}\text{X}$ are considered to be part of the surrounding text. Hence, the vertical spacing depends on the surrounding text, in particular on the width and depth of the last line of text. Due to this influence it can be difficult to control the spacing accurately. The package adds several options to control the vertical spacing, and it also implements a uniform behaviour for all types of equations.

The spacing of equations to the surrounding text is a combination of several aspects:

First, $\text{T}_{\text{E}}\text{X}$ inserts some interline spacing according to its rules. The amount depends on the depth/height of the surrounding text and the height/depth of the math content. The former

typically takes rather uniform values, whereas the latter can range wildly with the context (plain equations vs. fractions and matrices). As equations are normally surrounded by a relatively large amount of glue, it makes sense to reduce the dependency on the height/depth of math content. Therefore, the package makes equation environments appear to the surrounding text as a line with a fixed height and depth, and thus interline glue merely fills some potential gaps of the surrounding text. The apparent height and depth are defined by `displayheight` (*key*) and `displaydepth` (*key*) which default to the dimensions of a strut.

Second, the spacing of display equations depends on the width of the previous line of text. If the math content fits well into the available horizontal space, the display equation is called short and less glue is needed above the equation. The package implements this basic T_EX feature for all single- and multi-line equation environments. T_EX also reduces the amount of glue below short equations (potentially to make their spacing appear more uniform). The package allows to adjust the spacing for short equations via the global option `shortmode={mode}` where *mode* takes the values:

name	reduced glue
<code>off</code>	disabled
<code>above</code>	above short equations
<code>belowone</code>	also below short single-line equations
<code>belowall</code>	also below all short single-line equations

`short` (*key*) Short and long amounts of glue can also be enforced for individual equation environments via the optional arguments `short` and `long` taking the values `above`, `below` or `both`.

Third, the package provides several means to adjust the glue around equations:

`noskip` (*key*) Next to `short` and `long` the spacing above and below equation environments can be reduced to some other fixed smaller amount via `medskip` or removed altogether via `noskip`. These keys also take the values `above`, `below` or `both`.

`par` (*key*) The key `par` controls whether the equation environments end in horizontal mode (value `cont`) or in vertical mode (value `par`, default) with a dedicated amount of glue `belowparskip`. An environment can also be made to end in vertical mode without interline skip (value `top`) using the glue `belowtopskip`.

`...skip` (*key*) Variable amounts of skip can be set via `aboveskip` and `belowskip` or `skip` for both simultaneously. In addition, the package extends the `\vspace` mechanism of L^AT_EX to equation bodies where it adds vertical space below the next equation line or below the equation environment. Additional glue can be added above or below equation environments by means of the options `abovespace` and `belowspace`.

The package also maintains several global vertical space settings `aboveposskip` and `...skip` (*key*) `belowposskip` (sometimes `posskip` for both):

<code>...posskip</code>	both	description
<code>...long...</code>	<code>skip</code>	standard amount of glue
<code>...short...</code>	–	reduced glue for short equations
<code>...cont...</code>	–	glue when issued from an empty <code>\noindent</code> paragraph
<code>...par...</code>	–	glue when starting a paragraph (in vertical mode)
<code>...top...</code>	–	glue when issued at the top of vertical list
<code>...med...</code>	<code>medskip</code>	medium amount of glue
<code>...tag...</code>	<code>tagskip</code>	glue for outer raised/lowered tags
<code>...medtag...</code>	<code>medtagskip</code>	glue for outer raised/lowered tags with med
<code>...partag...</code>	–	glue for outer raised/lowered tags with par

`...mode` (*key*) The situations `cont`, `par` and `top` use the respective glue `above...skip` above the equations. By default they use `belowlongskip` below the equations, but this setting may be adjusted by the global option `posmode` with the values:

name	reduced glue
<code>long</code>	regular amount below equations
<code>short</code>	short amount below equations
<code>cont</code>	<code>cont</code> amount below equations
<code>par</code>	end the paragraph (only <code>parmode</code> and <code>topmode</code>)
<code>top</code>	end the paragraph without interline skip (<code>topmode</code> only)

`spread` (*key*) Likewise, the spacing between the lines of a multi-line equation environment can be adjusted via `spread={dimen}` which defaults to `\jot\equiv 3pt`. In addition, all equation lines and tags `strut` (*key*) are supplied with struts to ensure a minimum height and depth. The latter behaviour is controlled by the boolean switches `strut` and `strutttag` (*key*).

`displaybreak` (*key*) Finally, the breaking of multi-line equations across pages can be controlled as follows: The setting `allowdisplaybreaks` taking values 0 (never) through 4 (permissive) controls the permissivity of page breaks within multi-line equations. The optional argument `displaybreak` (*key*) taking values 0 (do not) through 4 (enforce) suggests a break just *above* the equation environment. The command `\displayatbreak` with values 0 through 4 suggests a break below the current line or below the equation environment.

2.7 Further Environments

The package supplies some additional environments:

`equationsbox` (*env.*) The package provides a boxed equation environment `equationsbox` which can be used within arbitrary math content. It works analogously to `equations` including optional arguments `margin` (*key*) and modifiers, but it offers a reduced range of functionality such as (evidently) no numbering. Additional arguments are given by `margin`, `marginleft`, `marginright` (*key*) which specify additional margin space around the equations box.

`subequations` (*env.*) The environment `subequations` group equations contained in the body with a common primary equation number and an extra level of numbering (typically: a, b, c, ...). The numbering layout can be controlled via `subeqtemplate` (*key*). For instance, the default behaviour of adding lowercase latin letters to the parent equation number (`\theparentequation`) is achieved by:

```
subeqtemplate={\theparentequation\alph{equation}}
```

`intertext` (*env.*) The environment `intertext` (or equivalently the macro `\intertext`) injects a (short) line of text into a multi-line equation while preserving the equation alignment across the text. The `intertext` environment must replace the end of line marker ‘\’ between two lines of the equation (to avoid blank lines). The environment accepts several of the vertical spacing adjustments as an optional argument.

2.8 General Options

`\eqnlineset` Options of general nature can be selected by the commands:

```
\usepackage[opts]{eqnlines}
or \PassOptionsToPackage{opts}{eqnlines}
or \eqnlineset{opts}
```

`\PassOptionsToPackage` must be used before `\usepackage`; `\eqnlineset` must be used afterwards. *opts* is a comma-separated list of options.

The package supplies the following general settings:

option	description
<code>defaults=classic</code>	mimic classic L ^A T _E X/amsmath (layout and dimensions)
<code>defaults=eqnlines</code>	eqnlines layout with fontsize-relative dimensions
<code>rescan</code>	rescan environment body to allow special commands (e.g. <code>\verb</code>)
<code>linesfallback</code>	single column in align mode reverts to lines mode
<code>ampproof</code>	equip optional argument parsing with protection for <code>&</code>
<code>crerror</code>	invoke an error when ‘ <code>\</code> ’ is used in a single equation

2.9 Feature Selection and Package Options

The following few settings can only be specified when loading the package, not via `\eqnlineset`:

option	type	description
<code>equation</code>	bool	provide/overwrite <code>equation</code> and <code>\[...]</code>
<code>amsmath</code>	bool	provide/overwrite <code>amsmath</code> environments and macros
<code>ang</code>	bool	provide <code>\<...></code>

If the above settings are explicitly disabled, the package will only supply the general purpose environment `equations` and its boxed cousin `equationsbox`. In that case, the specific equation environments and other features can be activated by the command:

$$\backslash eqnlinesprovide\{features\}$$

features is a comma-separated list of features:

feature	description
<i>env</i>	provide/overwrite environment <i>env</i> : <code>equation</code> , <code>gather</code> , <code>multline</code> , <code>align</code> , <code>flalign</code> , <code>multlined</code> , <code>gathered</code> , <code>aligned</code> , <code>subequations</code>
<i>env=name</i>	provide environment <i>env</i> as <i>name</i>
<code>sqr</code>	provide <code>\[...]</code>
<code>ang</code>	provide <code>\<...></code>
<i>cmd</i>	provide/overwrite macro <i>cmd</i> : <code>eqref</code> , <code>notag</code> , <code>thetag</code> , <code>allowdisplaybreaks</code> , <code>numberwithin</code>
<code>tagform</code>	provide/overwrite macro <code>\tagform@</code>
<code>maketag</code>	provide/overwrite macro <code>\maketag@@@</code>

3 Information

3.1 Copyright

Copyright © 2024–2025 Niklas Beisert

Based on the latex package `amsmath`: Copyright © 1995, 2000, 2013 American Mathematical Society; 2016–2024 LaTeX Project and American Mathematical Society.

This work may be distributed and/or modified under the conditions of the L^AT_EX Project Public License, either version 1.3 of this license or (at your option) any later version. The latest version of this license is in <https://www.latex-project.org/lppl.txt> and version 1.3c or later is part of all distributions of L^AT_EX version 2008 or later.

This work has the LPPL maintenance status ‘maintained’.

The Current Maintainer of this work is Niklas Beisert.

This work consists of the files `README.txt`, `eqnlines.ins` and `eqnlines.dtx` as well as the derived files `eqnlines.sty` and `eqnlines.pdf`.

3.2 Credits

This package is based on the L^AT_EX package `amsmath` (initially named `amstex`) which in turn is based on the T_EX macro system `amstex` written by Michael Spivak. The initial work of porting `amstex` to L^AT_EX was done in 1988–1989 by Frank Mittelbach and Rainer Schöpf. In 1994 David M. Jones added the support for flush-left layout and did extensive improvements to the `align` family of environments and to the equation number handling in general. Michael Downes at the AMS served as coordinator for the efforts of Mittelbach, Schöpf, and Jones, and has contributed various bug fixes and additional refinements over time. Since 2016, the package has been maintained by the L^AT_EX Project with contributions by the above and David Carlisle.

This package has been forked from `amsmath` in accordance with the LPPL, particularly paragraph 6. The original package `amsmath` is available at CTAN within `latex-amsmath`. It uses the basic mechanisms for processing numbered multi-line equations as developed in `amsmath` (environments `equation`, `align`, `gather`, `multline`, `gathered`, `aligned` and related), as well as code implementing these mechanisms. It differs from `amsmath` in the following aspects:

- The implementations of `split` and methods unrelated to multi-line equations and equation numbering have been dropped.
- Code has been restructured, macros have been renamed and extended.
- Numbering and horizontal adjustment schemes have been unified and extended.
- Options for math classes surrounding the alignment have been added.
- A punctuation scheme has been added.
- Vertical spacing has been redesigned.
- Optional parameters have been added to environments.
- Various configuration options and layout settings have been added.
- Cooperation with `hyperref`, `showkeys` and `amsmath` has been included into the package.

3.3 Files and Installation

The package consists of the files:

<code>README.txt</code>	readme file
<code>eqnlines.ins</code>	installation file
<code>eqnlines.dtx</code>	source file
<code>eqnlines.sty</code>	package file
<code>eqnlines-dev.sty</code>	package file (development version)
<code>eqnlines.pdf</code>	manual

The distribution consists of the files `README.txt`, `eqnlines.ins` and `eqnlines.dtx`.

- Run (pdf)L^AT_EX on `eqnlines.dtx` to compile the manual `eqnlines.pdf` (this file).
- Run L^AT_EX on `eqnlines.ins` to create the package `eqnlines.sty` and the developers version `eqnlines-dev.sty`. Copy the file `eqnlines.sty` to an appropriate directory of your L^AT_EX distribution, e.g. `texmf-root/tex/latex/eqnlines`.

3.4 Related CTAN Packages

The package is related to other packages available at CTAN:

- This package uses the package `keyval` to process the options for the package, environments and macros. Compatibility with the `keyval` package has been tested with v1.15 (2022/05/29).
- This package reproduces the math environments functionality of the package `amsmath`. The present code is based on `amsmath` v2.17t (2024/11/05). Compatibility with the `amsmath` package is maintained whether `eqnlines` is loaded before or after `amsmath`. By default, `eqnlines` will rename the math environments of `amsmath` with a prefix `ams...` and overwrite them with its own implementations. Alternatively, `eqnlines` may assign individual names to the maths environments and preserve the ones of `amsmath`. The other features provided by `amsmath` can be used.
- The package `mathtools` is a popular extension of the `amsmath` package. This package incorporates some of the features and improvements provided by the `mathtools` package. Compatibility with the `mathtools` package has been tested with v1.31 (2024/10/04), and it is maintained whether `eqnlines` is loaded before or after `mathtools`. Some features like adding a box and emphasising equations via `empheq` does not (yet) work.
- This package cooperates with the package `hyperref` to create anchors and references within the electronic document. Compatibility with the `hyperref` package has been tested with v7.011 (2024/11/05).
- This package supports the display of labels and references through the package `showkeys`. Compatibility with the `showkeys` package has been tested with v3.21 (2024/05/23).

3.5 Feature Suggestions

The following is a list of features for consideration towards future versions of this package. Their potential use may range between useful and niche; and their difficulty between easy and impossible:

- a proper manual
- complete code documentation
- sample document
- version of `\eqref` that can handle multiple references and ranges enclosed by the default formatting for equation tags
- switch to allow scanning of `\par` within body
- numbering scheme to place a number where there is the most available space

3.6 Revision History

v0.6.1: 2025/03/27

- `\eqnpunct` can place punctuation within the current equation field
- `numberline=none` now acts as `numberline=all` and `nonumber`
- fixed and extended `tagmargin` with `tagmargincalc`, `tagmarginratio` and `tagmarginthreshold`
- padding now applies to single-line equations as well

v0.6: 2025/03/11

- preliminary pdf tagging support (<https://latex3.github.io/tagging-project/>); `amsmath` *must* be loaded *before* `eqnlines` to avoid errors
- classic L^AT_EX/`amsmath` vs. `eqnlines` presets
- changed vertical spacing schemes and added further options
- supplied dimensions processed by `\glueexpr`
- more independent of `amsmath` structures
- internal reorganisations

v0.5: 2025/02/25

- preview version published on CTAN

A Implementation

The appendix documents the various components of the present package.

The code for the package is based on the `amsmath` package, see section 3.1 and section 3.2. It was forked at version v2.17t dated 2024/11/05. Most of the code was substantially redesigned (macros renamed, reshuffled, enhanced), but many of the underlying mechanisms were preserved. The documentation thus contains excerpts from the `amsmath` package documentation explaining some details of the implementation.

Please note that the documentation is completed only for few sections in the present version. Various open issues are remarked.

B General Support

In the following we describe general purpose supporting routines.

B.1 Development Messages

The package offers a version `eqnlines-dev` for development and debugging purposes. It outputs extra information on the current location within the code in order to track progress. The extra lines for the development version are indicated as ‘`<dev>`’ in the implementation documentation:

```
1 <dev>\def\eql@dev#1{\PackageInfo{eqnlines-dev}{#1}}
2 <dev>\def\eql@dev@start#1{\eql@dev{starting \string#1}}
3 <dev>\def\eql@dev@enter#1{\eql@dev{entering \string#1}}
4 <dev>\def\eql@dev@leave#1{\eql@dev{ leaving \string#1}}
5 <dev>\def\eql@dev@enterenv{\eql@dev{entering \@currentenv}}
6 <dev>\def\eql@dev@leaveenv{\eql@dev{ leaving \@currentenv}}
7 <dev>\def\eql@dev@in#1#2{\eql@dev{ \space within \string#1 #2}}
```

B.2 Supporting Definitions

`\eql@false` (*bool*) Rather than the standard L^AT_EX scheme of `\xxxfalse`, `\xxxtrue` and `\ifxxx` for boolean variables *xxx*, we use a scheme where `\xxx` is either undefined or defined (to an empty

macro) and is tested against by the ε -TeX conditional `\ifdefined\xxx`. In order to make the scheme more tangible, we define the two expected values for boolean variables:

```
8 \let\eql@false\@undefined
9 \let\eql@true\@empty
```

B.3 Dollardollar Abstraction

`\dollar@begin` As of 2025 L^AT_EX defines `\dollar@begin` and `\dollar@end` to represent `\eq@dollar@end` (and adjust) the beginning and end of bare T_EX display equations (`‘$$’`). For the time being, we make sure to revert to `‘$$’` if these macros are not yet available:

```
10 \ifdefined\dollar@begin
11 \def\eql@dollar@begin{\dollar@begin}
12 \def\eql@dollar@end{\dollar@end}
13 \else
14 \def\eql@dollar@begin{$$}
15 \def\eql@dollar@end{$$}
16 \fi
```

B.4 Look-Ahead in Alignment

Scanning for optional arguments [...] or modifiers such as `‘*’` using the L^AT_EX `\@ifnextchar` mechanism has two challenges within aligned equations: a square bracket or star may well be part of the intended mathematical expression and the look-ahead could trip upon an alignment character `‘&’` which inadvertently triggers to enter the next alignment column.

`\eq@ifnextchar@loose` To address the first challenge, we can force the special characters to follow immediately the `\eq@ifnextchar@tight` macro invocation. For clarity, we copy L^AT_EX’s original `\@ifnextchar` in `\kernel@ifnextchar` which skips over spaces as `\eq@ifnextchar@loose`. We replicate the `amsgen` version `\new@ifnextchar` that does not skip over spaces as `\eq@ifnextchar@loose`. The space before `#1` allows to look-ahead for spaces as well:

```
17 \let\eq@ifnextchar@loose\kernel@ifnextchar
18 \long\def\eq@ifnextchar@tight#1#2#3{%
19 \let\reserved@d= #1%
20 \def\reserved@a{#2}%
21 \def\reserved@b{#3}%
22 \futurelet\@let@token\eq@ifnch@tight
23 }
24 \def\eq@ifnch@tight{%
25 \ifx\@let@token\reserved@d
26 \let\reserved@b\reserved@a
27 \fi
28 \reserved@b
29 }
```

`\eql@atxii` Capture `‘@’` as a character (catcode 12) rather than a letter (catcode 11) as `\eql@atxii` so that we can look-ahead for `‘@’` with both `\makeatother` and `\makeatletter` modes:

```
30 \begingroup
31 \makeatother
32 \let\tmp=@%
33 \makeatletter
34 \global\let\eql@atxii\tmp
35 \endgroup
```


`\eql@ifnextgobble@...` We introduce a collection of look-ahead macros which do or do not skip over spaces. The macros `\eql@ifstar@...` and `\eql@testopt@...` replicate the L^AT_EX counterparts `\@ifstar` and `\@testopt`. The macros `\eql@ifnextgobble@...` work like `\@ifnextchar`, but also gobble the specific character if found; one might define `\eql@ifstar@...` as `\eql@ifnextgobble@...*`. The macros `\eql@teststaropt@...` tests for combinations of ‘*’ and optional arguments [...]:

```

36 \long\def\eql@ifnextgobble@loose#1#2{\eql@ifnextchar@loose#1{\@firstoftwo{#2}}}
37 \long\def\eql@ifnextgobble@tight#1#2{\eql@ifnextchar@tight#1{\@firstoftwo{#2}}}
38 \long\def\eql@ifstar@loose#1{\eql@ifnextchar@loose*\@firstoftwo{#1}}
39 \long\def\eql@ifstar@tight#1{\eql@ifnextchar@tight*\@firstoftwo{#1}}
40 \long\def\eql@ifat@loose#1#2{\eql@ifnextgobble@loose{#1}{%
41   \eql@ifnextgobble@loose\eql@atxii{#1}{#2}}}
42 \long\def\eql@ifat@tight#1#2{\eql@ifnextgobble@tight{#1}{%
43   \eql@ifnextgobble@tight\eql@atxii{#1}{#2}}}
44 \long\def\eql@testopt@loose#1#2{\eql@ifnextchar@loose[#{1}{#1}[#{2}]}]}
45 \long\def\eql@testopt@tight#1#2{\eql@ifnextchar@tight[#{1}{#1}[#{2}]}]}
46 \long\def\eql@teststaropt@loose#1#2#3{%
47   \eql@ifstar@loose{\eql@testopt@loose{#1}{#3}}{\eql@testopt@loose{#2}{#3}}
48 \long\def\eql@teststaropt@tight#1#2#3{%
49   \eql@ifstar@tight{\eql@testopt@tight{#1}{#3}}{\eql@testopt@tight{#2}{#3}}

```

`\eql@spbgroup` The second challenge is addressed by enclosing the look-ahead in spurious groups¹ which protect against triggering ‘&’. The macros `\eql@spbgroup` and `\eql@spegroup` open and close a spurious group. For some reason, the look-ahead mechanism requires further protections by inserting `\relax` at the beginning and by resetting `\let@token` at the end. These adjustments are included in the macros `\eql@srbgroup` and `\ers@spegroup`:

```

50 \def\eql@spbgroup{\iffalse{\fi\ifnum0=‘}\fi}
51 \def\eql@spegroup{\ifnum0=‘{\fi\iffalse}\fi}
52 \def\eql@srbgroup{\relax\iffalse{\fi\ifnum0=‘}\fi}
53 \def\eql@sregroup{\let\let@token\relax\ifnum0=‘{\fi\iffalse}\fi}

```

`\eql@ampprotect` The macros `\eql@ampprotect` and `\eql@ampprotecttwo` inject the opening and closing of spurious groups into the look-ahead mechanism:

```

54 \long\def\eql@ampprotect#1#2{\eql@srbgroup#1{\eql@sregroup#2}}
55 \long\def\eql@ampprotecttwo#1#2#3{%
56   \eql@srbgroup#1{\eql@sregroup#2}{\eql@sregroup#3}}

```

`...@ampsafe` We introduce a collection of ‘&’-safe look-ahead macros:

```

57 \def\eql@ifnextchar@loose@ampsafe#1{%
58   \eql@ampprotecttwo{\eql@ifnextchar@loose#1}}
59 \def\eql@ifnextchar@tight@ampsafe#1{%
60   \eql@ampprotecttwo{\eql@ifnextchar@tight#1}}
61 \def\eql@ifstar@loose@ampsafe{\eql@ampprotecttwo\eql@ifstar@loose}
62 \def\eql@ifstar@tight@ampsafe{\eql@ampprotecttwo\eql@ifstar@tight}
63 \def\eql@testopt@loose@ampsafe{\eql@ampprotect\eql@testopt@loose}
64 \def\eql@testopt@tight@ampsafe{\eql@ampprotect\eql@testopt@tight}
65 \def\eql@teststaropt@loose@ampsafe{\eql@ampprotecttwo\eql@teststaropt@loose}
66 \long\def\eql@teststaropt@tight@ampsafe{%
67   \eql@ampprotecttwo\eql@teststaropt@tight}

```

`\eql@amproof` We may want to replace L^AT_EX’s definitions `\@ifnextchar`, `\@ifstar` and `\@testopt` to respect ‘&’ characters within aligned equations. This might make unrelated definitions with

¹See <https://www.latex-project.org/cgi-bin/ltbugs2html?pr=latex/3040>, <https://www.latex-project.org/cgi-bin/ltbugs2html?pr=amslatex/1834> and <https://tex.stackexchange.com/questions/9897/showcase-of-brace-tricks-egroup-iffalse-fi-etc>.

optional arguments and starred variants more robust in this context. The macro `\eql@amproof` overwrites the original definitions, and `\eql@amprevert` reverts the changes:

```

68 \let\eql@ifnextchar@org@ifnextchar
69 \let\eql@ifstar@org@ifstar
70 \let\eql@testopt@org@testopt
71 \def\eql@amprevert{%
72   \let@ifnextchar\eql@ifnextchar@org
73   \let@testopt\eql@testopt@org
74   \let@ifstar\eql@ifstar@org
75 }
76 \def\eql@ampproof{%
77   \let@ifnextchar\eql@ifnextchar@loose@ampsafe
78   \let@testopt\eql@testopt@loose@ampsafe
79   \let@ifstar\eql@ifstar@loose@ampsafe
80 }

```

B.5 Error Messages

`\eql@error` Main error and warning message function for the package:
`\eql@warning`

```

81 \def\eql@error#1{\PackageError{eqnlines}{#1}{}}
82 \def\eql@warning{\PackageWarning{eqnlines}}

```

`\eql@error@nomathmode` Error messages concerning math mode:

```

\eql@error@mathmode
83 \def\eql@error@nomathmode#1{\eql@error{#1 allowed only in math mode}}
84 \def\eql@error@mathmode#1{\eql@error{#1 allowed only in paragraph mode}}

```

`\eql@warn@label@unused` Warning messages concerning unused and multiply declared labels and tags:

```

\eql@warn@label@multiple
85 \def\eql@warn@label@unused{\eql@warning{Unused equation \string\label:
86   label '\eql@nextlabel' will be lost}}
87 \def\eql@warn@label@multiple#1{\eql@warning{Multiple equation \string\label's:
88   previous label '#1' will be lost}}
89 \def\eql@warn@tag@unused{\eql@warning{Unused equation \string>tag:
90   tag declaration will be lost}}
91 \def\eql@warn@tag@multiple{\eql@warning{Multiple equation \string>tag's:
92   previous tag declaration will be lost}}

```

B.6 amsmath Integration

`\eql@amsmath@after` We need to overwrite certain macros from `amsmath`. The method `\eql@amsmath@after` executes argument #1 after loading `amsmath` is loaded. It also runs the code if `amsmath` has already been loaded. Furthermore, loading `amsmath` requires certain macros to be undefined. To this end `\eql@amsmath@before` will execute argument #1 before any future loading of `amsmath`. `\eql@amsmath@undefine` undefines a macro in this way and `\eql@amsmath@let` overwrites a macro of `\ctanpkg{amsmath}`:

```

93 \def\eql@amsmath@after#1{\AddToHook{package/amsmath/after}{#1}}
94 \def\eql@amsmath@before#1{%
95   \ifpackageloaded{amsmath}{\AddToHook{package/amsmath/before}{#1}}
96   \def\eql@amsmath@undefine#1{\eql@amsmath@before{\let#1\undefined}}
97 \def\eql@amsmath@let#1#2{\eql@amsmath@undefine#1\let#1#2}

```

B.7 Tagging Support

`\eql@tagging@...` Proper PDF tagging² support requires a L^AT_EX version at least of 2025. For the time being, we define an abstraction layer so that the package will collaborate with L^AT_EX versions around 2020:

```

98 \let\eql@tagging@on\eql@false
99 \IfFormatAtLeastTF{2025-06-01}{%
100 \csname tag_if_active:T\endcsname{\let\eql@tagging@on\eql@true}}{}
101 \ifdefined\eql@tagging@on
102 \def\eql@tagging@mathsave{%
103 \UseTaggingSocket{math/luamml/save/nNn}{\displaystyle{mtd}}}
104 \def\eql@tagging@mathaddlast{%
105 \UseTaggingSocket{math/luamml/mtable/finalizecol}{last}}
106 \def\eql@tagging@tagbegin{%
107 \UseTaggingSocket{math/display/tag/begin}}
108 \def\eql@tagging@tagend{%
109 \UseTaggingSocket{math/display/tag/end}}
110 \def\eql@tagging@tagsave{%
111 \UseTaggingSocket{math/luamml/mtable/tag/save}}
112 \def\eql@tagging@tagaddbox{%
113 \setbox\z@\copy\eql@tagbox%
114 \UseTaggingSocket{math/luamml/mtable/tag/set}}
115 \def\eql@tagging@tablesaveinner{%
116 \UseExpandableTaggingSocket{math/luamml/mtable/innertable/save}}
117 \def\eql@tagging@tableaddinner{%
118 \UseTaggingSocket{math/luamml/mtable/innertable/finalize}}
119 \def\eql@tagging@tablesavelines{%
120 \UseExpandableTaggingSocket{math/luamml/mtable/finalize}{gather}}
121 \def\eql@tagging@tablesavealign{%
122 \UseExpandableTaggingSocket{math/luamml/mtable/finalize}{align}}
123 \def\eql@tagging@alignleft{%
124 \UseTaggingSocket{math/luamml/mtable/aligncol}{left}}
125 \def\eql@tagging@aligncenter{%
126 \UseTaggingSocket{math/luamml/mtable/aligncol}{center}}
127 \def\eql@tagging@alignright{%
128 \UseTaggingSocket{math/luamml/mtable/aligncol}{right}}

```

We need to get hold of the equation body in all cases so that we can feed it into the tagging mechanism:

```

129 \let\eql@single@doscan\eql@true
130 \let\eql@scan@body\eql@scan@body@rescan

```

`\eql@tagging@start` We need to activate tagging for display equations for environments and for enclosures
`\eql@tagging@end` `\[...]` and `\<...>`. The tagging interface registration macro `\RegisterMathEnvironment` will work only partially for our cases, hence we replicate code from `\math_register_halign_env:nn`. Make sure collection is not yet active (`\l_math_collected_bool`). Then feed collected environment name, options and body into `_math_process:nn`. Indicate the start of a display equation:

```

131 \ExplSyntaxOn
132 \def\eql@tagging@start{%
133 \bool_if:NF\l_math_collected_bool{%
134 \toks@\expandafter{\eql@tagging@opt}%
135 \edef\eql@tmp{\@currentenv}{\the\toks@} \the\eql@scan@reg}}%
136 \expandafter\_math_process:nn\eql@tmp

```

²see <https://latex3.github.io/tagging-project/>

```

137     \@kernel@math@registered@begin
138     \bool_set_true:N\l__math_collected_bool
139   }%
140 }
141 \def\eql@tagging@end{}
142 \ExplSyntaxOff
143 \else
144   \def\eql@tagging@mathsave{}
145   \def\eql@tagging@mathaddlast{}
146   \def\eql@tagging@tagbegin{}
147   \def\eql@tagging@tagend{}
148   \def\eql@tagging@tagsave{}
149   \def\eql@tagging@tagaddbox{}
150   \def\eql@tagging@tablesaveinner{}
151   \def\eql@tagging@tableaddinner{}
152   \def\eql@tagging@tablesavealign{}
153   \def\eql@tagging@tablesavealign{}
154   \def\eql@tagging@alignleft{}
155   \def\eql@tagging@aligncenter{}
156   \def\eql@tagging@alignright{}
157   \def\eql@tagging@start{}
158   \def\eql@tagging@end{}
159 \fi

```

C Parameters and Registers

In the following, we collect parameter and register definitions.

C.1 Parameters

TODO: describe

TODO: maybe sort parameters into sections **TODO:** or sort parameters in sections here

`\eql@tagsleft` (*bool*) The boolean parameter `\eql@tagsleft` specifies whether the tags are placed at the left or right margin:

```
160 \let\eql@tagsleft\eql@false
```

`\eql@flushleft` (*bool*) The boolean parameter `\eql@flushleft` specifies whether to use left or central alignment layout:

```
161 \let\eql@flushleft\eql@false
```

`\eql@flushleftmargin@` (*dimen*) The default width of the left margin in left alignment layout is specified by

`\eql@flushleftmargin@val` `\eql@flushleftmargin@`. It may be pushed down to `\eql@flushleftmarginmin@` and up

to `\eql@flushleftmarginmax@`:

`\eql@flushleftmarginmin@` (*dimen*)

```

162 \newdimen\eql@flushleftmargin@
163 \newdimen\eql@flushleftmarginmin@
164 \newdimen\eql@flushleftmarginmax@
165 \def\eql@flushleftmargin@val{\leftmargini}
166 \eql@flushleftmarginmin@\z@
167 \eql@flushleftmarginmax@.5\maxdimen

```

`\eql@tagmargin@` (*dimen*) The intended margin width for tags in central alignment layout is stored in `\eql@tagmargin@` which is sourced by `\eql@tagmargin@val`. An undefined `\eql@tagmargin@val` will compute the margin width as the maximum width of tags (without separation). `\eql@tagmargin@ratio@` describes the maximum ratio of lines with tags to total number of lines for which `\eql@tagmargin@` is set to zero: **TODO**: threshold

```
168 \newdimen\eql@tagmargin@
169 \let\eql@tagmargin@val\@undefined
170 \newdimen\eql@tagmargin@ratio@
171 \eql@tagmargin@ratio@\p@
172 \def\eql@tagmargin@threshold{0.5}
```

`\eql@indent@` (*dimen*) The currently selected indentation width is specified by `\eql@indent@`. This dimension register is set to the macro `\eql@indent@val` when entering the equation environments:

```
173 \newdimen\eql@indent@
174 \def\eql@indent@val{2em}
```

`\eql@paddingleft@` (*dimen*) The padding of an equation (column) is specified by `\eql@paddingleft@` and `\eql@paddingright@`. These dimension registers are set to the macros `\eql@paddingleft@val` and `\eql@paddingright@val`, respectively, when entering the equation environments:

```
175 \newdimen\eql@paddingleft@
176 \newdimen\eql@paddingright@
177 \def\eql@paddingleft@val{0pt}
178 \def\eql@paddingright@val{0pt}
```

`\eql@paddingmax` (*bool*) The boolean register `\eql@paddingmax` specifies whether the full line should be used for padding:

```
179 \let\eql@paddingmax\eql@false
```

`\eql@box@marginleft` and `\eql@box@marginright` The macros `\eql@box@marginleft` and `\eql@box@marginright` specify the margin surrounding equation boxes:

```
180 \def\eql@box@marginleft{\z@skip}
181 \def\eql@box@marginright{\z@skip}
```

`\eql@box@colsep` The macro `\eql@box@colsep` specifies the intercolumn separation for equation boxes:

```
182 \def\eql@box@colsep{2em}
```

`\eql@spread` The extra spread of equation lines is specified by `\eql@spread`:

```
183 \def\eql@spread{\jot}
184 \newdimen\eql@spread@amount@
```

`\eql@tagfuzz@` (*dimen*) The value `\eql@tagfuzz@` specifies the margin of error for comparing whether a tag fits a given equation line. We should not expect rounding errors in the fixed point arithmetic of \TeX , nevertheless: **TODO**: probably do not need this due to fixed point arithmetic.

```
185 \newdimen\eql@tagfuzz@
186 \eql@tagfuzz@16sp\relax
```

`\eql@display@height` and `\eql@display@depth` An equation will appear to the surrounding text with a fixed apparent height and depth specified by `\eql@display@height` and `\eql@display@depth`, respectively. By default it appears as a strut for equations:

```

187 \def\eqldisplay@height{\ht\eql@strutbox@}
188 \def\eqldisplay@depth{\dp\eql@strutbox@}

```

`\mode@short@` (*counter*) The setting `\eql@skip@mode@short@` specifies when a reduced amount of glue should be used around equations in case the text line above the equation fits in the space that is left available in the first equation line. Value 0 turns this feature off, value 1 reduces the glue above the equation, value 2 furthermore reduces the glue below a single equation line and value 3 also reduces the glue below multi-line equations:

```

189 \newcount\eql@skip@mode@short@
190 \eql@skip@mode@short@ \tw@

191 \newcount\eql@skip@mode@cont@
192 \eql@skip@mode@cont@ \z@

193 \newcount\eql@skip@mode@par@
194 \eql@skip@mode@par@ \z@

195 \newcount\eql@skip@mode@top@
196 \eql@skip@mode@top@ \z@

197 \newcount\eql@skip@mode@leave@
198 \let\eql@skip@force@leave@\@undefined

```

`\eql@skip@force@above` 0: short, 1: long, 2: cont, 3: par, 4: top, 5: no, 6: med, 7: custom

```

\eql@skip@force@below
\mode@above@ (counter)
\mode@below@ (counter)
199 \newcount\eql@skip@mode@above@
200 \newcount\eql@skip@mode@below@
201 \let\eql@skip@force@above@\@undefined
202 \let\eql@skip@force@below@\@undefined
203 \let\eql@skip@custom@above@\@undefined
204 \let\eql@skip@custom@below@\@undefined

```

`\eql@skip@cont@above` The glue when an equation is at the top of a horizontal list is specified by `\eql@skip@cont@above`:

`\eql@skip@top@above` The glue when an equation is at the top of a vertical list is specified by `\eql@skip@top@above` and `\eql@skip@top@below`:

`\eql@skip@par@above` The glue when an equation starts a paragraph is specified by `\eql@skip@par@above`:

`\eql@skip@med@above` The surrounding glue for an equation with reduced spacing is given by `\eql@skip@med@above` and `\eql@skip@med@below`:

```

205 \def\eql@skip@long@above{\abovedisplayskip}
206 \def\eql@skip@long@below{\belowdisplayskip}
207 \def\eql@skip@short@above{\abovedisplayshortskip}
208 \def\eql@skip@short@below{\belowdisplayshortskip}
209 \def\eql@skip@cont@above{\eql@skip@short@above}
210 \def\eql@skip@cont@below{\eql@skip@short@below}
211 % \TODD: parabove plus parskip?
212 \def\eql@skip@par@above{\eql@skip@long@above}
213 \def\eql@skip@par@below{\eql@skip@long@below}
214 \def\eql@skip@top@above{\eql@skip@long@above}
215 \def\eql@skip@top@below{\eql@skip@long@below}
216 \def\eql@skip@med@above{\abovedisplayskip/2}
217 \def\eql@skip@med@below{\belowdisplayskip/2}
218 \def\eql@skip@tag@above{\z@skip}

```

```

219 \def\eq@skip@tag@below{\z@skip}
220 \def\eq@skip@partag@above{\z@skip}
221 \def\eq@skip@partag@below{\z@skip}
222 \def\eq@skip@medtag@above{\z@skip}
223 \def\eq@skip@medtag@below{\z@skip}

```

`\eq@colsepmin@` (*dimen*) The minimum intercolumn separation is specified by `\eq@colsepmin@`. This dimension register is set to `\eq@colsepmin@val` when entering the equation environments to allow font-dependent values. Furthermore, `\eq@colsepmax@val` specifies the maximum intercolumn separation:

```

224 \newdimen\eq@colsepmin@
225 \def\eq@colsepmin@val{1em}
226 \def\eq@colsepmax@val{.5\maxdimen}

```

`\eq@tagwidthmin@` (*dimen*) The minimum tag width is specified by `\eq@tagwidthmin@`:

```

227 \newdimen\eq@tagwidthmin@
228 \eq@tagwidthmin@\z@

```

`\eq@tagsepmin@` (*dimen*) The minimum separation between an equation and its tag is given by `\eq@tagsepmin@`. \TeX 's built-in value is half a quad³ in font number 2. As the tag is processed in text mode, we use 0.5em instead.

```

229 \newdimen\eq@tagsepmin@
230 \def\eq@tagsepmin@val{.5\fontdimen6\textfont\tw@}

```

`\eq@equations@sqr@opt` and `\eq@equations@ang@opt` store the default arguments for `\[...]` and `\<...>`, respectively:

```

231 \def\eq@equations@sqr@opt{equation,nonumber}
232 \def\eq@equations@ang@opt{align,nonumber}

```

Multi-Line Align Mode.

```

233 \let\eq@align@margins\eq@true

```

C.2 Registers

TODO: describe

General.

`\eq@fieldbox@` (*box*) The box `\eq@fieldbox@` holds the present alignment component and `\eq@tagbox@` the tag for the present line. The corresponding dimensions `\eq@fieldwidth@` and `\eq@tagwidth@` hold their widths:

```

234 \newbox\eq@fieldbox@
235 \newbox\eq@tagbox@
236 \newdimen\eq@fieldwidth@
237 \newdimen\eq@tagwidth@

```

³another half of a quad is left empty at the other end of the line.

`\eq@totalwidth@` (*dimen*)
`\eq@tagwidth@max@` (*dimen*)
238 `\newdimen\eq@totalwidth@`
239 `\newdimen\eq@tagwidth@max@`

`\eq@line@height@` (*dimen*) The dimension registers `\eq@line@height@` and `\eq@line@depth@` keep track of the
`\eq@line@depth@` (*dimen*) height and depth of the present line in an alignment:
240 `\newdimen\eq@line@height@`
241 `\newdimen\eq@line@depth@`

`\eq@line@width@` (*dimen*)
`\eq@line@avail@` (*dimen*)
`\eq@line@pos@` (*dimen*)
242 `\newdimen\eq@line@width@`
243 `\newdimen\eq@line@avail@`
244 `\newdimen\eq@line@pos@`

Rows and Columns.

`\eq@row@` (*counter*) `\eq@row@` counts the present row (1-based) and `\eq@totalrows@` holds the total number
`\eq@totalrows@` (*counter*) of rows:
`\eq@tagrows@` (*counter*)
245 `\newcount\eq@row@`
246 `\newcount\eq@totalrows@`
247 `\newcount\eq@tagrows@`

`\eq@column@`
`\eq@totalcolumns@`
248 `\newcount\eq@column@`
249 `\newcount\eq@totalcolumns@`

`\eq@colsep@` (*dimen*) The dimension of the intercolumn separation for align environments is stored in
`\eq@colsep@`:
250 `\newdimen\eq@colsep@`

`\align@inter@` (*counter*)
251 `\newcount\eq@align@inter@`

Vertical Spacing Adjustments.

`\eq@firstavail@` (*dimen*) The unused space on the first line of an alignment is stored in
`\eq@display@firstavail@set` `\eq@display@firstavail@` for comparison against `\predisplaysize` and determining
short skip mode of display equations. It is convenient to set it via
`\eq@display@firstavail@set` provided that we are on the first line:
252 `\newdimen\eq@display@firstavail@`
253 `\def\eq@display@firstavail@set#1{%`
254 `\ifnum\eq@row@=\@ne`
255 `\global\eq@display@firstavail@#1%`
256 `\fi`
257 `}`

`\eq@firstlast@` (*counter*) The counter stores whether the tag one first/last line is raised/lowered as 1/2 (or 3 for
both). This implies a different `vskip` corresponding to the mostly empty line:
258 `\newcount\eq@raisetag@firstlast@`

Shared Registers.

`\ifmeasuring@` (*bool*) All display environments get typeset twice – once during a “measuring” phase and then again during a “production” phase. We reuse the original `amsmath` definition `\ifmeasuring@` to determine which case we’re in, so we and other packages may take appropriate action. It does not hurt to define this conditional in any case. We should tell `hyperref` about measuring processes as we’re not `amsmath` and not being catered for:

```
259 \ifdefined\measuring@true\else
260   \expandafter\newif\csname ifmeasuring@\endcsname
261 \fi
262 \AddToHook{package/hyperref/after}{
263   \ifdefined\Hy@ifnotmeasuring
264     \renewcommand\Hy@ifnotmeasuring[1]{\ifmeasuring@\else#1\fi}
265   \fi
266 }
```

`\if@display` (*bool*) `amsmath` defines the conditional `\if@display` to test whether we’re in a display equation including the inner math parts of equation blocks. We provide it in case `amsmath` is absent, and initialise it:

```
267 \ifdefined\@displaytrue\else
268   \expandafter\newif\csname if@display\endcsname
269   \everydisplay\expandafter{\the\everydisplay\@displaytrue}
270 \fi
```

C.3 Hooks

`\eql@hook@...` For what it’s worth, we define a couple of entry points where one might hook into the equations typesetting framework. The \LaTeX hook framework would be more versatile, but as the purpose of these hooks is rather unclear at the moment, we make this as efficient as it could get: **TODO:** may add a few more hooks

```
271 \let\eql@hook@blockbefore\@empty
272 \let\eql@hook@blockafter\@empty
273 \let\eql@hook@blockin\@empty
274 \let\eql@hook@blockout\@empty
275 \let\eql@hook@linein\@empty
276 \let\eql@hook@lineout\@empty
277 \let\eql@hook@colin\@empty
278 \let\eql@hook@colout\@empty
279 \let\eql@hook@eqin\@empty
280 \let\eql@hook@eqout\@empty
281 \let\eql@hook@innerleft\@empty
282 \let\eql@hook@innerright\@empty
283 \let\eql@hook@innerlead\@empty
```

D Punctuation

The equations environments supply an automatic punctuation scheme which allows to define a default punctuation at the end of each column, line and equation block.

`\eql@punct@col` These macros store the punctuation character for columns, lines and blocks. A value `\eql@punct@line` `\relax` indicates that the punctuation should be handed down to the next lower level:
`\eql@punct@block`

```

284 \let\eqlopunct@col\empty
285 \let\eqlopunct@line\relax
286 \let\eqlopunct@block\relax

```

`\eqlopunct@sep` This macro stores the separation to be applied before the punctuation (unless it is empty):

```
287 \let\eqlopunct@sep\relax
```

`\eqnpunctcol` Set the punctuation for columns, lines and blocks. Note that the macro `\eqnpunct` sets the punctuation for the following equation block or for the current field. Starred versions clear the punctuation for the respectively levels:

```

\eqnpunct
288 \def\eqnpunctcol{\eqlopunct@tight\eqlopunct@col@setrelax\eqlopunct@col@set}
289 \def\eqlopunct@col@set#1{\def\eqlopunct@col{#1}\ignorespaces}
290 \def\eqlopunct@col@setrelax{\let\eqlopunct@col\empty\ignorespaces}
291 \def\eqnpunctline{\eqlopunct@tight\eqlopunct@line@setrelax\eqlopunct@line@set}
292 \def\eqlopunct@line@set#1{\def\eqlopunct@line{#1}\ignorespaces}
293 \def\eqlopunct@line@setrelax{\let\eqlopunct@line\relax\ignorespaces}
294 \def\eqnpunctmain{\eqlopunct@tight\eqlopunct@main@setrelax\eqlopunct@main@set}
295 \def\eqlopunct@main@set#1{\def\eqlopunct@main{#1}\ignorespaces}
296 \def\eqlopunct@main@setrelax{\let\eqlopunct@main\relax\ignorespaces}
297 \def\eqnpunct{\eqlopunct@tight\eqlopunct@next@setrelax\eqlopunct@next@set}
298 \def\eqlopunct@next@set#1{%
299   \ifmmode
300     \def\eqlopunct@col{#1}%
301     \def\eqlopunct@line{#1}%
302     \def\eqlopunct@block{#1}%
303   \else
304     \eqnadopt{punct={#1}}%
305   \fi
306   \ignorespaces}
307 \def\eqlopunct@next@setrelax{%
308   \ifmmode
309     \let\eqlopunct@block\relax
310   \else
311     \eqnadopt{punct*}%
312   \fi
313   \ignorespaces}

```

`\eqlopunct@apply@col` Output the punctuation for the present column. If non-empty, prepend some separation. Clear the punctuation so that no further column punctuation is output within the current group:

```

314 \def\eqlopunct@apply@col{%
315   \ifx\eqlopunct@col\empty\else
316     \eqlopunct@sep
317     \eqlopunct@col
318     \let\eqlopunct@col\empty
319   \fi
320 }

```

Output the punctuation currently set for lines unless disabled. Alike `\eqlopunct@apply@col` prevent further output of punctuations for lines and columns within the current group:

`\eqlopunct@apply@line`

```

321 \def\eqlopunct@apply@line{%
322   \ifx\eqlopunct@line\relax

```

```

323 % \TODO hand down immediately?
324 \else
325   \ifx\eq@punct@line\@empty\else
326     \eq@punct@sep
327     \eq@punct@line
328   \fi
329   \let\eq@punct@line\relax
330   \let\eq@punct@col\@empty
331 \fi
332 }

```

`\eq@punct@apply@block` Outputs the punctuation for the current equation block unless disabled in analogy to `\eq@punct@apply@line`:

```

333 \def\eq@punct@apply@block{%
334   \ifx\eq@punct@block\relax
335 % \TODO hand down immediately?
336 \else
337   \ifx\eq@punct@block\@empty\else
338     \eq@punct@sep
339     \eq@punct@block
340   \fi
341   \let\eq@punct@block\relax
342   \let\eq@punct@line\relax
343   \let\eq@punct@col\@empty
344 \fi
345 }

```

E Math Classes at Alignment

The following describes the adjustment of math classes surrounding the alignment marker.

`\eq@class@innerright@sel@` Select between `\eq@class@innerlead` and `\eq@class@innerright` depending on whether the left part of the aligned column is empty:

```

346 \def\eq@class@innerright@sel@{%
347   \ifdim\eq@fieldwidth@=\z@
348     \eq@class@innerlead
349   \else
350     \eq@class@innerright
351 \fi
352 }

```

`\eq@class@innerleft@set` Set the left, right and leading math classes. Setting the right math class disables the leading math class, so the leading math class must be specified after the right one:
`\eq@class@innerright@set`
`\eq@class@innerlead@set`

```

353 \def\eq@class@innerleft@set#1{%
354   \def\eq@class@innerleft{#1}%
355 }
356 \def\eq@class@innerright@set#1{%
357   \def\eq@class@innerright{#1}%
358   \let\eq@class@innerright@sel\eq@class@innerright
359 }
360 \def\eq@class@innerlead@set#1{%
361   \def\eq@class@innerlead{#1}%
362   \let\eq@class@innerright@sel\eq@class@innerright@sel@
363 }

```

`\eql@class@ampeq` We define two standard combinations of math classes intended to be used with ‘&=’
`\eql@class@eqamp` (ampeq) or ‘=&’ (eqamp). The default setting is ‘&=’ (ampeq):

```

364 \def\eql@class@ampeq{%
365   \eql@class@innerleft@set{}%
366   \eql@class@innerright@set{}%
367 }
368 \def\eql@class@eqamp{%
369   \eql@class@innerleft@set{\mathrel{}%
370   \eql@class@innerright@set{\mathrel{}%
371   \eql@class@innerlead@set{}%
372 }
373 \eql@class@ampeq

```

F Equations Box Environment

TODO: describe

TODO: fixed width version (works only towards intercolumn stretch)?

`\eql@box@cr`

```

374 \protected\def\eql@box@cr{%
375   \eql@ampprotect\eql@testopt@tight\eql@box@cr@z@
376 }
377 \def\eql@box@cr@[#1]{%
378   \eql@punct@apply@line
379   \eql@hook@lineout
380   \eql@box@lastfield
381   \cr
382   \noalign{%
383     \vskip\glueexpr#1\relax
384   }%
385 }

386 \let\eql@box@box\vcenter

387 \def\eql@box@lastfield@odd{%
388   &\omit
389   \kern-\wd\eql@fieldbox@
390   \box\eql@fieldbox@
391   \hfil
392   &\omit\kern-\eql@colsep@
393 }%
394 \def\eql@box@lastfield@even{&\omit\kern-\eql@colsep@}
395 \def\eql@box@lastfield@lines{&\omit\kern-2\eql@colsep@}

396 \def\eql@box@open@align{%
397 % \TODO templates
398   \let\eql@box@lastfield\@empty
399   \eql@halign@init{%
400 (dev)\eql@dev{starting new line}%
401   }%
402   \tabskip\z@skip
403   \halign\bgroup
404     &%
405     \let\eql@box@lastfield\eql@box@lastfield@odd
406     \global\setbox\eql@fieldbox@\hbox{%

```

```

407     \eql@strut@field
408     \@lign
409     $\m@th\displaystyle
410     \eql@hook@colin
411     ##%
412     \eql@class@innerleft
413     \eql@hook@innerleft
414     \eql@tagging@mathsave
415     $%
416     \eql@tagging@mathaddlast
417 }%
418 \hfil
419 \kern\wd\eql@fieldbox@
420 \tabskip\z@skip
421 &%
422 \eql@fieldwidth@\wd\eql@fieldbox@
423 \kern-\eql@fieldwidth@
424 \box\eql@fieldbox@
425 \let\eql@box@lastfield\eql@box@lastfield@even
426 \llap{\unhbox\eql@fieldbox@}%
427 \hbox{%
428     \eql@strut@field
429     \@lign
430     $\m@th\displaystyle
431     \eql@hook@innerright
432     \eql@class@innerright@sel
433     ##%
434     \eql@punct@apply@col
435     \eql@hook@colout
436     \eql@tagging@mathsave
437     $%
438     \eql@tagging@mathaddlast
439 }%
440 \hfil
441 \tabskip\eql@colsep@\relax
442 \crrc
443 \noalign{%
444     \eql@hook@blockbefore
445 }%
446 \eql@hook@blockin
447 }

448 \def\eql@box@open@lines{%
449 % \TODO templates
450 \let\shoveleft\eql@adjust@shoveleft
451 \let\shovecenter\eql@adjust@shovecenter
452 \let\shoveright\eql@adjust@shoveright
453 \let\eql@box@lastfield\eql@box@lastfield@lines
454 \eql@halign@init{%
455 (dev)\eql@dev{starting line \the\eql@row@}%
456 \global\advance\eql@row@\@ne
457 }%
458 \tabskip\z@skip
459 \halign\bgroup
460 &%
461     \eql@shape@pos@\m@ne
462     \setbox\eql@fieldbox@\hbox{%
463         \eql@strut@field
464         \@lign

```

```

465     $\m@th\displaystyle
466     \eql@hook@colin
467     ##%
468     \eql@punct@apply@col
469     \eql@hook@colout
470     \eql@tagging@mathsave
471     $%
472     \eql@tagging@mathaddlast
473     }%
474     \ifnum\eql@shape@pos@=\m@ne
475     \eql@shape@eval
476     \fi
477     \ifcase\eql@shape@pos@
478     \kern\eql@shape@amount@
479     \box\eql@fieldbox@
480     \skip@\@flushglue
481     \advance\skip@\eql@paddingleft@\relax
482     \advance\skip@\eql@paddingright@\relax
483     \advance\skip@-\eql@shape@amount@\relax
484     \hskip\skip@
485     \eql@tagging@alignleft
486     \or
487     \skip@\@flushglue
488     \advance\skip@\eql@paddingleft@\relax
489     \hskip\skip@
490     \box\eql@fieldbox@
491     \skip@\@flushglue
492     \advance\skip@\eql@paddingright@\relax
493     \hskip\skip@
494     \eql@tagging@aligncenter
495     \or
496     \skip@\@flushglue
497     \advance\skip@\eql@paddingleft@\relax
498     \advance\skip@\eql@paddingright@\relax
499     \hskip\skip@
500     \box\eql@fieldbox@
501     \eql@tagging@alignright
502     \fi
503     \tabskip\eql@colsep@\relax
504     \crr
505     \noalign{%
506     \eql@hook@blockbefore
507     }%
508     \eql@hook@blockin
509 }

510 \def\eql@box@close{%
511     \ifvmode\else
512     \global\eql@totalrows@\eql@row@
513     \eql@punct@apply@block
514     \eql@box@cr@[ \z@skip]%
515     \fi
516     \crr
517     \noalign{%
518     \eql@hook@blockafter
519     }%
520     \eql@tagging@tablesaveinner
521 \egroup
522 }

```

`\eql@box@start`

```
523 \def\eql@box@start{%
524   \relax
525   \ifmmode
526     \let\eql@box@endmath\@empty
527   \else
528     $\let\eql@box@endmath=$%
529   \fi
530   \eql@nextopt@process{equationsbox}%
531   \let\eql@punct@block\eql@punct@main
532   \let\eql@punct@main\relax
533   \eql@colsep@\glueexpr\eql@box@colsep\relax
534   \eql@paddingleft@\glueexpr\eql@paddingleft@val\relax
535   \eql@paddingright@\glueexpr\eql@paddingright@val\relax
536   \eql@indent@\glueexpr\eql@indent@val\relax
537   \eql@stack@save@boxed
538   \let\eql@flushleft\eql@false
539   \eql@row@z@
540   \eql@totalrows@\@M
541   \eql@shape@sel
542   \hskip\glueexpr\eql@box@marginleft\relax
543   \eql@box@box\bgroup
544     \eql@display@leave
545     \let\\eql@box@cr
546     \ifdefined\eql@box@mode@lines
547       \expandafter\eql@box@open@lines
548     \else
549       \expandafter\eql@box@open@align
550     \fi
551 }
```

`\eql@box@end`

```
552 \newcommand{\eql@box@end}{%
553   \eql@box@close
554   \egroup
555   \eql@tagging@tableaddinner
556   \hskip\glueexpr\eql@box@marginright\relax
557   \eql@stack@restore
558   \eql@box@endmath
559 }
```

`equationsbox (env.)`

```
560 \newenvironment{equationsbox}{%
561   (dev)\eql@dev@enterenv
562   \eql@ampprotect\eql@box@testall\eql@box@start
563 }{%
564   \eql@box@end
565   (dev)\eql@dev@leaveenv
566 }

567 \def\eql@box@testall{\eql@box@testtilde}
568 \def\eql@box@testtilde#1{%
569   \eql@ifnextgobble@tight~%
570   {\eqnadopt{lines}\eql@box@testopt{#1}}%
571   {\eql@box@testopt{#1}}}
572 \def\eql@box@testopt#1{%
```

```

573 \eql@ifnextchar@tight[%]
574   {\eql@box@adopt{#1}}%
575   {#1}}
576 \def\eql@box@adopt#1[#2]{\eqnadopt{#2}#1}

577 \def\eql@mode@aligned{\let\eql@box@mode@lines\eql@false}
578 \def\eql@mode@lined{\let\eql@box@mode@lines\eql@true}
579 \eql@mode@aligned

```

G Equation Numbering

TODO: describe

G.1 Tag Formatting

TODO: describe

```

580 \def\eql@tag@setbox#1{%
581   \def\eql@tag@box##1{#1}%
582 }
583 \def\eql@tag@setbox#1{%
584   \def\eql@tag@box##1{\hbox{\m@th\normalfont#1}}%
585 }

```

TODO: describe

```

586 \def\eql@tag@setform#1{%
587   \def\eql@tag@form##1{#1}%
588 }
589 \def\eql@tag@setform#1#2#3{%
590   \def\eql@tag@form##1{#1\ignorespaces#2\unskip\@italiccorr#3}%
591 }

592 \eql@tag@setbox{#1}
593 \eql@tag@setform({#1})
594 \def\eql@tag@boxedform#1{\eql@tag@box{\eql@tag@form{#1}}}

```

Raise Tags.

`\eql@tag@amount@` (*dimen*)

```
595 \newdimen\eql@raisetag@amount@
```

`\raisetag`

```

596 \def\eql@raisetag@default{%
597   \eql@warning{\string\raisetag\space not allowed here}
598   \@gobble
599 }

```

TODO: describe

```
600 \eql@amsmath@let\raisetag\eql@raisetag@default
```

TODO: maybe introduce a star form to enforce raise?

```
601 \def\eql@raisetag#1{\global\eql@raisetag@amount@\glueexpr#1\relax}%
```


G.2 Showkeys Integration

TODO: describe

```
602 \let\eqL@SK@loaded\eqL@false
603 \let\eqL@SK@label\@gobble
604 \let\eqL@SK@clearlabel\@empty
605 \let\eqL@SK@setlabel\@gobble
606 \let\eqL@SK@printlabel@right\@empty
607 \let\eqL@SK@printlabel@left\@empty
608 \let\eqL@SK@printlabel@line\@empty
609 \def\eqL@label@clean{\eqL@label@org}
610 \AddToHook{package/showkeys/after}{
611   \let\eqL@SK@loaded\eqL@true
612   \def\eqL@SK@label#1{\SK@\SK@@label#1}
613   \def\eqL@SK@clearlabel{\let\eqL@SK@lab\relax}
614   \eqL@SK@clearlabel
615   \def\eqL@SK@@label#1>#2\SK@{%
616     \def\eqL@SK@lab{\smash{\SK@labelcolor\showkeyslabelformat{#2}}}%
617   }
618   \def\eqL@SK@setlabel#1{\SK@\eqL@SK@@label#1}
619   \def\eqL@SK@printlabel@right{%
620     \ifx\eqL@SK@lab\relax\else
621       \rlap{\kern\marginparsep\eqL@SK@lab}%
622       \eqL@SK@clearlabel
623     \fi
624   }
625   \def\eqL@SK@printlabel@left{%
626     \ifx\eqL@SK@lab\relax\else
627       \llap{\eqL@SK@lab\kern\marginparsep}%
628       \eqL@SK@clearlabel
629     \fi
630   }
631   \def\eqL@SK@printlabel@line{%
632     \ifx\eqL@SK@lab\relax\else
633       \dimen@\prevdepth
634       \nointerlineskip
635       \ifdefined\eqL@tagsleft
636         \llap{%
637           \eqL@SK@lab
638           \kern\marginparsep
639         }%
640       \eqL@SK@clearlabel
641     \else
642       \rlap{%
643         \dimen@\displaywidth
644         \advance\dimen@\marginparsep
645         \kern\dimen@
646         \eqL@SK@lab
647       }%
648     \fi
649     \eqL@SK@clearlabel
650     \prevdepth\dimen@
651   \fi
652 }
653 \let\eqL@label@org\label
654 \def\eqL@label@clean{\let\SK@\@gobbletwo\eqL@label@org}
655 }
```

G.3 Labels

TODO: describe

```
656 % \TODO implement (via label[] or labelname similar to label/tag)
657 \let\eq@nextlabel\@undefined
658 \def\eq@labelname@default{[equation]}
```

\eq@label@org

```
659 \let\eq@label@org\label
```

G.4 Tags

TODO: describe

```
660 \let\eq@nexttag\@undefined
```

\eq@tag@default

```
661 \def\eq@tag@default{%
662   \eq@error{\string\tag\space not allowed here}{}\eq@tag@gobble}
663 \let\tag\eq@tag@default
```

\eq@tag@gobble **TODO:** ifnextchar, gobbletwo?

```
664 \def\eq@tag@gobble@[#1]#2{}
665 \def\eq@tag@gobble{%
666   \eq@ampprotecttwo\eq@teststaropt@tight\eq@tag@gobble@\eq@tag@gobble@{}}
```

\eq@nexttag **TODO:** can amsmath handle also counter refstepcounter in tags?

\eq@tag@makenext
hyperref anchors

\eq@tag@makenext@@
\eq@tag@makenext@@@

```
667 \let\eq@Hy@anchor@gobble
668 \AddToHook{package/hyperref/after}{
669   \def\eq@Hy@anchor#1{%
670     \Hy@raisedlink{\hyper@anchor{#1}}%
671   }%
672 }
```

```
673 \def\eq@tag@makenext{%
674   \eq@ampprotecttwo\eq@teststaropt@tight
675   \eq@tag@makenext@star\eq@tag@makenext@\eq@tag@text
676 }
```

TODO: not sure about \protected@edef\eq@tag@text was \def only

```
677 \def\eq@tag@makenext@star[#1]#2{%
678   \global\def\eq@nexttag{%
679     \let\eq@tag@tool\@firstofone
680     \protected@edef\eq@tag@text{#2}%
681     \protected@edef\eq@tag@label{#1}%
682   }%
683 }
684 \def\eq@tag@makenext@[#1]#2{%
685   \global\def\eq@nexttag{%
686     \let\eq@tag@tool\eq@tag@form
687     \protected@edef\eq@tag@text{#2}%
```

```

688 \protected@edef\eql@tag@label{#1}%
689 \protected@edef\eql@tag@label{\p@equation\eql@tag@label}%
690 }%
691 }

```

G.5 Anchors

TODO: describe

`\eql@refcount@` (*counter*)

```

692 \newcount\eql@numbering@refcount@
693 \eql@numbering@refcount@z@
694 \def\eql@numbering@ref{equation.eql-\the\eql@numbering@refcount@}
695 \def\eql@numbering@refstep{\global\advance\eql@numbering@refcount@\@ne}

```

TODO: describe

```

696 \def\eql@numbering@makeblockanchor{%
697 \eql@numbering@refstep
698 \global\edef\eql@label@currentHref{\eql@numbering@ref}%
699 \eql@Hy@anchor\eql@label@currentHref
700 \global\edef\eql@label@thepage{\thepage}%
701 }
702 \def\eql@numbering@setblockanchor{%
703 \let\thepage\eql@label@thepage
704 \let\@currentHref\eql@label@currentHref
705 }

```

G.6 Tag Composition

TODO: describe

```

\eql@compose@anchor
\eql@compose@tag
\eql@compose@label
706 \def\eql@compose@anchor{%
707 \ifdefined\eql@nexttag
708 \eql@nexttag
709 \def\@currentcounter{equation}%
710 \let\@currentlabel\eql@tag@label
711 \eql@numbering@refstep
712 \edef\@currentHref{\eql@numbering@ref}%
713 \eql@Hy@anchor\@currentHref
714 \global\let\eql@nexttag\@undefined
715 \else
716 \refstepcounter{equation}%
717 \let\eql@tag@tool\eql@tag@form
718 \edef\eql@tag@text{\theequation}%
719 \fi
720 }

721 \def\eql@compose@label{%
722 \ifmeasuring@\else
723 \eql@SK@clearlabel
724 \ifdefined\eql@nextlabel
725 \ifnum
726 \ifnum\eql@numbering@target@<z@

```

```

727     \eql@row@
728     \else
729     \eql@numbering@target@
730     \fi=\eql@row@
731     \eql@compose@label@
732     \fi
733     \fi
734 \fi
735 }

```

TODO: describe

```

736 \def\eql@compose@label@{%
737 \eql@SK@setlabel\eql@nextlabel
738 \begingroup
739 \ifnum\eql@numbering@target@=\eql@row@
740 \eql@numbering@setblockanchor
741 \fi
742 \let\@currentlabelname\eql@labelname@default
743 \expandafter\eql@label@clean\expandafter{\eql@nextlabel}%
744 \global\let\eql@nextlabel\@undefined
745 \endgroup
746 }

```

TODO: describe

```

747 \def\eql@compose@tag{%
748 \eql@tagging@tagbegin
749 \eql@tag@box{%
750 \eql@tag@tool\eql@tag@text
751 \eql@tagging@tagsave
752 }%
753 \eql@tagging@tagend
754 }

```

TODO: describe

```

755 \def\eql@compose@print{%
756 \eql@compose@anchor
757 \eql@compose@label
758 \ifdefined\eql@tagsleft
759 \eql@SK@printlabel@left
760 \eql@compose@tag
761 \else
762 \eql@compose@tag
763 \eql@SK@printlabel@right
764 \fi
765 }

```

TODO: describe

```

766 \def\eql@compose@measure{%
767 \ifdefined\eql@nexttag
768 \eql@nexttag
769 \eql@tag@box{\eql@tag@tool\eql@tag@text}%
770 \else
771 \stepcounter{equation}%
772 \eql@tag@box{\eql@tag@form\theequation}%
773 \fi
774 \ifnum\eql@numbering@target@<\z@
775 \global\let\eql@nextlabel\@undefined

```

```

776   \global\let\eql@nexttag\@undefined
777   \fi
778 }

```

G.7 Tagbox Methods

TODO: describe

TODO: one might still compare width to zero and pretend the tag is absent??

```

779 \def\eql@tagbox@make#1{%
780   \setbox\eql@tagbox@\hbox{\eql@strut@tag@\lign#1}%
781   \eql@tagwidth@\wd\eql@tagbox@
782   \ifdim\eql@tagwidth@<\eql@tagwidthmin@
783     \eql@tagwidth@\eql@tagwidthmin@
784   \fi
785   \advance\eql@tagwidth@\eql@tagsepmin@
786 }

```

TODO: describe

```

787 \def\eql@tagbox@print@right{%
788   \kern-\wd\eql@tagbox@
789   \box\eql@tagbox@
790 }

```

TODO: describe

```

791 \def\eql@tagbox@print@left{%
792   \wd\eql@tagbox@\z@
793   \box\eql@tagbox@
794 }

```

TODO: describe

```

795 \def\eql@tagbox@print@right@raise{%
796   \ifnum\eql@row@=\eql@totalrows@
797     \global\advance\eql@raisetag@firstlast@\tw@
798   \fi
799   \llap{\vtop{%
800     \vskip-\eql@raisetag@amount@
801     \normalbaselines
802     \setbox\@ne\hbox{}}%
803     \dp\@ne\eql@line@depth@
804     \box\@ne
805     \box\eql@tagbox@
806   }}%
807 }
808 \def\eql@tagbox@print@left@raise{%
809   \ifnum\eql@row@=\@ne
810     \global\advance\eql@raisetag@firstlast@\@ne
811   \fi
812   \rlap{\vbox{%
813     \normalbaselines
814     \box\eql@tagbox@
815     \vbox to\eql@line@height@{}}%
816     \vskip\eql@raisetag@amount@
817   }}%
818 }

```

TODO: describe

```
819 \def\eq@numbering@printsubeqlabel{%
820   \ifdefined\eq@parentlabel
821     \eq@numbering@makeblockanchor
822     \eq@SK@setlabel\eq@parentlabel
823   \begingroup
824     \def\@currentcounter{equation}%
825     \eq@numbering@setblockanchor
826     \let\@currentlabelname\eq@labelname@default
827     \protected@edef\@currentlabel{\p@equation\theparentequation}%
828     \expandafter\eq@label@clean\expandafter{\eq@parentlabel}%
829   \endgroup
830   \eq@SK@printlabel@line
831 \fi
832 }
```

G.8 Numbering Schemes

TODO: describe

```
833 \def\eq@numbering@tab@first{first}
834 \def\eq@numbering@tab@last{last}
835 \def\eq@numbering@tab@middle{middle}
836 \def\eq@numbering@tab@here{here}
837 \def\eq@numbering@tab@in{in}
838 \def\eq@numbering@tab@out{out}
839 \def\eq@numbering@tab@sub{sub}
840 \def\eq@numbering@tab@all{all}
841 \def\eq@numbering@tab@none{none}
```

TODO: describe

```
842 \let\eq@numbering@tab@f\eq@numbering@tab@first
843 \let\eq@numbering@tab@l\eq@numbering@tab@last
844 \let\eq@numbering@tab@m\eq@numbering@tab@middle
845 \let\eq@numbering@tab@mid\eq@numbering@tab@middle
846 \let\eq@numbering@tab@o\eq@numbering@tab@out
847 \let\eq@numbering@tab@outside\eq@numbering@tab@out
848 \let\eq@numbering@tab@i\eq@numbering@tab@in
849 \let\eq@numbering@tab@inside\eq@numbering@tab@in
850 \let\eq@numbering@tab@within\eq@numbering@tab@in
851 \let\eq@numbering@tab@h\eq@numbering@tab@here
852 \let\eq@numbering@tab@s\eq@numbering@tab@sub
853 \let\eq@numbering@tab@subeq\eq@numbering@tab@sub
854 \let\eq@numbering@tab@subequation\eq@numbering@tab@sub
855 \let\eq@numbering@tab@subequations\eq@numbering@tab@sub
856 \let\eq@numbering@tab@a\eq@numbering@tab@all
857 \let\eq@numbering@tab@n\eq@numbering@tab@none
858 \expandafter\let\csname eq@numbering@tab@!\endcsname\eq@numbering@tab@all
859 \expandafter\let\csname eq@numbering@tab@*\endcsname\eq@numbering@tab@none
860 \expandafter\let\csname eq@numbering@tab@1\endcsname\eq@numbering@tab@first

861 \let\eq@numbering@mode\eq@numbering@tab@all

862 \def\eq@numbering@set#1{%
863   \ifcsname eq@numbering@tab@#1\endcsname
864     \expandafter\let\expandafter\eq@numbering@mode
865     \csname eq@numbering@tab@#1\endcsname
```

```

866 \ifx\eql@numbering@mode\eql@numbering@tab@none
867 \let\eql@numbering@mode\eql@numbering@tab@all
868 \let\eql@numbering@active\eql@false
869 \fi
870 \else
871 \eql@error{numbering mode '#1' unknown: setting to 'all'}%
872 \let\eql@numbering@mode\eql@numbering@tab@all
873 \fi
874 }

```

ring@target@ (*counter*)

```

875 \let\eql@numbering@active\eql@true
876 \newcount\eql@numbering@target@

877 \def\eql@numbering@mode@all{%
878 \eql@numbering@target@\m@ne}
879 \def\eql@numbering@mode@sub{%
880 \eql@numbering@target@\m@ne
881 \let\eql@numbering@subeq@use\eql@true}
882 \def\eql@numbering@mode@first{%
883 \eql@numbering@target@\@ne}
884 \def\eql@numbering@mode@last{%
885 \eql@numbering@target@\@MM}
886 \def\eql@numbering@mode@here{%
887 \eql@numbering@target@\z@}

```

TODO: describe

```

888 \def\eql@numbering@mode@in{%
889 \ifdefined\eql@tagsleft
890 \eql@numbering@mode@last
891 \else
892 \eql@numbering@mode@first
893 \fi
894 }

```

TODO: describe

```

895 \def\eql@numbering@mode@out{%
896 \ifdefined\eql@tagsleft
897 \eql@numbering@mode@first
898 \else
899 \eql@numbering@mode@last
900 \fi
901 }

```

TODO: describe

```

902 \def\eql@numbering@mode@middle{%
903 \eql@numbering@target@\z@
904 \let\eql@numbering@eval@target\eql@numbering@eval@middle}
905 \def\eql@numbering@eval@middle{%
906 \ifnum\eql@numbering@target@=\z@
907 \count@\eql@row@
908 \advance\count@\@ne
909 \divide\count@\tw@
910 \global\eql@numbering@target@\count@
911 \fi
912 }

```

TODO: describe

```
913 \def\eq@numbering@eval@mode{%
914   \let\eq@numbering@eval@target@\undefined
915   \let\eq@numbering@subeq@use\eq@false
916   \csname eq@numbering@mode@\eq@numbering@mode\endcsname
917   \ifdefined\eq@numbering@active
918     \let\eq@numbering@eqnswinit\@eqnswtrue
919   \else
920     \let\eq@numbering@eqnswinit\@eqnswfalse
921   \fi
922   \let\eq@numbering@active\eq@false
923 }
```

TODO: reconsider operation

`\numberhere`

```
924 \def\numberhere{%
925   \ifmeasuring@
926     \ifnum\eq@numbering@target@<\z@\else
927       \global\eq@numbering@target@\eq@row@
928     \fi
929   \fi
930 }
```

TODO: describe

`\numbernext`

```
931 \def\numbernext{%
932   \ifmeasuring@
933     \ifnum\eq@numbering@target@<\z@\else
934       \ifnum\eq@numbering@target@=\eq@row@
935         \global\advance\eq@numbering@target@\@ne
936       \fi
937     \fi
938   \fi
939 }
```

G.9 Numbering Framework

TODO: describe

```
940 \let\eq@numbering@autolabel\eq@false
941 \let\eq@numbering@autotag\eq@true
942 \let\eq@numbering@blocklabel@\undefined
943 \let\eq@numbering@blocktag@\undefined

944 \eq@amsmath@after{
945   \let\eq@print@eqnum@default\print@eqnum
946   \let\eq@incr@eqnum@default\incr@eqnum
947 }
```

TODO: describe

```
948 \def\donumber{%
949   \if@eqnsw\else
950     \global\@eqnswtrue
951     \ifx\print@eqn@\empty
```



```

952     \global\let\print@eqn\eql@print@eqnum@default
953     \fi
954     \ifx\incr@eqn\@empty
955         \global\let\incr@eqn\eql@incr@eqnum@default
956         \fi
957     \fi
958 }

```

TODO: describe

```

959 \def\eql@label@warn{%
960     \ifdefined\eql@numbering@autolabel
961         \global\@eqnswtrue
962     \fi
963     \ifdefined\eql@nextlabel
964         \eql@warn@label@multiple\eql@nextlabel
965     \fi
966     \global\edef\eql@nextlabel
967 }

```

TODO: describe

```

968 \def\eql@tag@warn{%
969     \ifdefined\eql@numbering@autotag
970         \global\@eqnswtrue
971     \fi
972     \ifdefined\eql@nexttag
973         \eql@warn@tag@multiple
974     \fi
975     \eql@tag@makenext
976 }

```

TODO: describe

```

977 \def\eql@label@nowarn{%
978     \ifdefined\eql@numbering@autolabel
979         \global\@eqnswtrue
980     \fi
981     \global\edef\eql@nextlabel
982 }

```

TODO: describe

```

983 \def\eql@tag@nowarn{%
984     \ifdefined\eql@numbering@autotag
985         \global\@eqnswtrue
986     \fi
987     \eql@tag@makenext
988 }

```

TODO: describe

```

989 \def\eql@blocklabel@set#1{%
990     \ifdefined\eql@blocklabel
991         \eql@warn@label@multiple\eql@blocklabel
992     \fi
993     \edef\eql@blocklabel{#1}%
994 }

```

TODO: describe

```

995 \def\eql@blocktag@set#1{%

```

```

996 \ifdefined\eql@blocktag
997   \eql@warn@tag@multiple
998 \fi
999 \def\eql@blocktag{#{1}}%
1000 }

```

TODO: describe

```

1001 \def\eql@blocktag@setstar#1{%
1002   \ifdefined\eql@blocktag
1003     \eql@warn@tag@multiple
1004   \fi
1005 \def\eql@blocktag{*{#1}}%
1006 }

```

Single Line. **TODO:** describe

```

1007 \def\eql@numbering@single@init{%
1008   \let\label\eql@label@warn
1009   \let\tag\eql@tag@warn
1010   \let\raisetag\eql@raisetag
1011   \eql@numbering@target@&m@ne
1012   \let\eql@nextlabel\eql@blocklabel
1013   \ifdefined\eql@blocktag
1014     \expandafter\eql@tag@makenext\eql@blocktag
1015   \else
1016     \let\eql@nexttag\@undefined
1017   \fi
1018   \eql@numbering@eqnswinit
1019   \ifdefined\eql@numbering@autolabel
1020     \ifdefined\eql@nextlabel
1021       \@eqnswtrue
1022     \fi
1023   \fi
1024   \ifdefined\eql@numbering@autotag
1025     \ifdefined\eql@nexttag
1026       \@eqnswtrue
1027     \fi
1028   \fi
1029 \global\eql@raisetag@amount\z@
1030 }

```

Multi-Line Measuring Pass. **TODO:** describe

```

1031 \def\eql@numbering@measure@init{%
1032   \let\label\eql@label@warn
1033   \let\tag\eql@tag@warn
1034   \let\raisetag\eql@raisetag
1035   \global\let\eql@nextlabel\eql@blocklabel
1036   \ifdefined\eql@blocktag
1037     \expandafter\eql@tag@makenext\eql@blocktag
1038   \else
1039     \global\let\eql@nexttag\@undefined
1040   \fi
1041   \ifnum\eql@numbering@target@<\z@\else
1042     \eql@numbering@eqnswinit
1043     \ifdefined\eql@numbering@autolabel
1044       \ifdefined\eql@nextlabel

```

```

1045     \@eqnswtrue
1046     \fi
1047   \fi
1048 \fi
1049 }

```

TODO: describe

```

1050 \def\eq@numbering@measure@line@begin{%
1051   \ifnum\eq@numbering@target@<\z@
1052     \global\eq@numbering@eqnswinit
1053   \fi
1054 }

```

TODO: describe

```

1055 \def\eq@numbering@measure@eval{%
1056   \ifdefined\eq@numbering@eval@target
1057     \eq@numbering@eval@target
1058   \fi
1059   \ifnum\eq@numbering@target@>\eq@row@
1060     \global\eq@numbering@target@\eq@row@
1061   \fi
1062   \ifnum\eq@numbering@target@>\z@
1063     \if@eqnsw\else
1064       \global\eq@numbering@target@\z@
1065     \fi
1066   \fi
1067   \ifnum\eq@numbering@target@<\@ne
1068     \ifdefined\eq@nextlabel
1069       \eq@warn@label@unused
1070       \global\let\eq@nextlabel\@undefined
1071     \fi
1072     \ifdefined\eq@nexttag
1073       \eq@warn@tag@unused
1074       \global\let\eq@nexttag\@undefined
1075     \fi
1076   \fi
1077 }

```

Multi-Line Print Pass. **TODO:** describe

```

1078 \def\eq@numbering@print@init{%
1079   \ifnum\eq@numbering@target@<\z@
1080     \let\label\eq@label@nowarn
1081     \let\tag\eq@tag@nowarn
1082     \let\raisetag\eq@raisetag
1083     \let\eq@nextlabel\eq@blocklabel
1084     \ifdefined\eq@blocktag
1085       \expandafter\eq@tag@makenext\eq@blocktag
1086     \else
1087       \let\eq@nexttag\@undefined
1088     \fi
1089   \else
1090     \let\label\@gobble
1091     \let\tag\eq@tag@gobble
1092     \let\raisetag\eq@gobble
1093   \fi
1094 }

```

TODO: describe

```
1095 \def\eq@numbering@print@block@begin{%
1096   \ifnum\eq@numbering@target@>\z@
1097     \eq@numbering@makeblockanchor
1098   \fi
1099   \ifdefined\eq@numbering@subeq@use
1100     \eq@numbering@printsbeqlabel
1101   \fi
1102 }
```

TODO: describe

```
1103 \def\eq@numbering@print@line@begin{%
1104   \ifnum\eq@numbering@target@<\z@
1105     \global\eq@numbering@eqnswinit
1106     \global\eq@raisetag@amount@\z@
1107   \fi
1108 }
```

TODO: describe

```
1109 \def\eq@numbering@print@line@eval{%
1110   \ifnum\eq@numbering@target@<\z@\else
1111     \ifnum\eq@numbering@target@=\eq@row@
1112       \global\@eqnswtrue
1113     \else
1114       \global\@eqnswfalse
1115     \fi
1116   \fi
1117 }
```

H Subequation Numbering

We replicate the `amsmath` functionality to number a block of equations with a common number and a sub-numbering.

H.1 Definitions

`parentequation` (*counter*) We define a counter to store the main equation number while in subequation mode. It makes sense to share this definition with `amsmath` as `parentequation`, and we need to undefine it when `amsmath` is loaded at a later stage:

```
1118 \eq@amsmath@undefine\c@parentequation
1119 \eq@amsmath@undefine\theparentequation
1120 \ifdefined\c@parentequation\else
1121 \newcounter{parentequation}
1122 \fi
```

`subequations@template` We store a template which will be installed as `\theequation` in subequations mode: **TODO:** need to protect something?!

```
1123 \def\eq@subequations@template{\theparentequation\alph{equation}}
```

`@subequations@active` A boolean register which tells whether subequations are in use and thus must not be invoked again:

```
1124 \let\eq@subequations@active\eq@false
```

`\eql@subequations@init` Low-level initialise the subequations mode. Store the equation counter in `\eql@subequations@restorecounter` for the case that no equation numbers will be used. Step the equation counter, copy it to `parentequation` and initialise `\theparentequation` (and its hyperref counterpart) with the expanded current value of `\theequation`; fill with tag data instead if a tag has been specified. Reset the equation counter and use the template for `\theequation`:

```

1125 \def\eql@subequations@init{%
1126   \edef\eql@subequations@restorecounter{%
1127     \global\c@equation\the\c@equation\relax}%
1128   \ifdefined\eql@blocktag
1129     \expandafter\eql@tag@makenext\eql@blocktag
1130     \eql@nexttag
1131     \eql@numbering@refstep
1132     \protected@edef\theHparentequation{\eql@numbering@ref}%
1133     \protected@edef\theparentequation{\eql@tag@text}%
1134   \else
1135     \advance\c@equation\@ne
1136     \protected@edef\theparentequation{\theequation}%
1137     \ifdefined\theHequation
1138       \protected@edef\theHparentequation{\theHequation}%
1139     \fi
1140   \fi
1141   \global\c@parentequation\c@equation
1142   \global\c@equation\z@
1143   \let\theequation\eql@subequations@template
1144   \def\theHequation{\theHparentequation.\arabic{equation}}}%
1145 }

```

`\eql@subequations@close` Low-level close the subequations mode. If no number has been used, return to the original equation counter, otherwise use the value stored in `parentequation`. Note that we cannot use `\setcounter` here because the calc version would involve actions which are not allowed after `\halign` within a display equation:

```

1146 \def\eql@subequations@close{%
1147   \ifnum\c@equation=\z@
1148     \eql@subequations@restorecounter
1149   \else
1150     \global\c@equation\c@parentequation
1151   \fi
1152 }

```

H.2 Environment

`\eql@subequations@start` Start the subequations environment with optional parameters in #1. Enter subequations mode and set an anchor for subsequent `\label`'s. Manually print the showkeys tag:
TODO: join with other similar anchor routines `\eql@numbering@printsubeqlabel`

```

1153 \def\eql@subequations@start{%
1154   \let\eql@blocktag\undefined
1155   \let\eql@blocklabel\undefined
1156   \eql@nextopt@process{subequations}%
1157   \eql@subequations@init
1158   \eql@numbering@refstep
1159   \edef\eql@subequations@currentHref{\eql@numbering@ref}%
1160   \eql@Hy@anchor\eql@subequations@currentHref
1161   \edef\eql@subequations@thepage{\thepage}%

```

```

1162 \def\@currentcounter{equation}%
1163 \let\@currentHref\@eql@subequations@currentHref
1164 \protected@edef\@currentlabel{\p@equation\theparentequation}%
1165 \let\@currentlabelname\@eql@labelname@default
1166 \let\@eql@subequations@active\@eql@true
1167 \ifdefined\@eql@blocklabel
1168   \eql@SK@label\@eql@blocklabel
1169 \fi
1170 \ignorespaces
1171 }

```

`\eql@subequations@end` End the subequations environment. Issue the label if one has been specified and an equation number has actually been used. Then close subequations mode:

```

1172 \def\@eql@subequations@end{%
1173   \ifnum\c@equation>\z@
1174     \ifdefined\@eql@blocklabel
1175       \begingroup
1176         \def\@currentcounter{equation}%
1177         \let\thepage\@eql@subequations@thepage
1178         \let\@currentHref\@eql@subequations@currentHref
1179 % \TODO how about tag* ?! also within equations!
1180         \protected@edef\@currentlabel{\p@equation\theparentequation}%
1181         \let\@currentlabelname\@eql@labelname@default
1182         \expandafter\@eql@label@clean\expandafter{\@eql@blocklabel}%
1183       \endgroup
1184     \fi
1185   \fi
1186   \@eql@subequations@close
1187   \ignorespacesafterend
1188 }

```

`\subequations (env.)` The subequations environment tests for optional parameters and passes on to the start and end routines:

```

1189 \newenvironment{eql@subequations}{%
1190 (dev)\eql@dev@enterenv
1191 \eql@subequations@testall\eql@subequations@start%
1192 }{%
1193 \eql@subequations@end
1194 (dev)\eql@dev@leaveenv
1195 }

```

TODO: describe

```

1196 \def\@eql@subequations@testall{\eql@subequations@testopt}
1197 \def\@eql@subequations@testopt#1{%
1198   \eql@ifnextchar@tight[%]
1199     {\eql@subequations@adopt{\eql@subequations@testat{#1}}}%
1200     {\eql@subequations@testat{#1}}}
1201 \def\@eql@subequations@adopt#1[#2]{\eqnadopt{#2}#1}
1202 \def\@eql@subequations@testat#1{%
1203   \eql@ifat@tight%
1204     {\eql@subequations@addlabel{#1}}%
1205     {#1}}
1206 \def\@eql@subequations@addlabel#1#2{\eqnadopt{label={#2}}#1}

```

H.3 Subequation Scheme

TODO: describe

```
1207 \def\eq@numbering@subeq@init{%
1208   \let\eq@save@theequation\theequation
1209   \let\eq@save@theHequation\theHequation
1210   \eq@subequations@init
1211   \let\eq@parentlabel\eq@blocklabel
1212   \let\eq@parenttag\eq@blocktag
1213   \let\eq@blocklabel\@undefined
1214   \let\eq@blocktag\@undefined
1215 }
```

TODO: describe

```
1216 \def\eq@numbering@subeq@test{%
1217   \ifnum\c@equation<\tw@
1218     \let\eq@numbering@subeq@use\@ne
1219   \fi
1220 }
```

TODO: describe

```
1221 \def\eq@numbering@subeq@revert{%
1222   \let\eq@blocklabel\eq@parentlabel
1223   \let\eq@blocktag\eq@parenttag
1224   \let\eq@numbering@subeq@use\eq@false
1225   \let\theequation\eq@save@theequation
1226   \let\theHequation\eq@save@theHequation
1227   \eq@subequations@restorecounter
1228 }
```

TODO: describe

```
1229 % \TODO note must not use setcounter here (when calc is loaded)
1230 \def\eq@numbering@subeq@close{%
1231   \eq@subequations@close
1232 }
```

I Display Equations Support

TODO: describe

I.1 Display Breaks

TODO: describe

erdisplaylinepenalty

```
1233 \interdisplaylinepenalty\@M
```

\eq@getdsp@pen **TODO:** isn't this the opposite order than \@getpen?!

```
1234 \def\eq@getdsp@pen#1{%
1235   \ifcase #1\@M \or 9999 \or 6999 \or 2999 \or \z@\fi
1236 }
```

TODO: allow a displaybreak before equations

```
1237 \DeclareRobustCommand{\eql@displaybreak@default}[1][4]{%
1238   \eql@warning{Invalid use of \string\displaybreak}{}}
1239 \eql@amsmath@after{\let\eql@displaybreak@default\displaybreak}
1240 \eql@amsmath@let\displaybreak\eql@displaybreak@default

1241 \newcount\eql@displaybreak@pen@
1242 \newcount\eql@displaybreak@prepen@

1243 \protected\def\eql@displaybreak@print{%
1244   \eql@ampprotect\eql@testopt@tight\eql@displaybreak@print@{4}%
1245 }
```

TODO: describe

```
1246 \def\eql@displaybreak@print@{#1}{%
1247   \ifnum#1<\z@
1248     \global\eql@displaybreak@pen@\@MM
1249   \else
1250     \global\eql@displaybreak@pen@-\@getpen{#1}\relax
1251   \fi
1252 }
```

TODO: describe

```
1253 \def\eql@displaybreak@pre#1{%
1254   \ifnum#1<\z@
1255     \global\eql@displaybreak@prepen@\@MM
1256   \else
1257     \global\eql@displaybreak@prepen@-\@getpen{#1}\relax
1258   \fi
1259 }
```

TODO: describe

```
1260 \protected\def\eql@displaybreak@measure{%
1261   \eql@ampprotect\eql@testopt@tight\eql@displaybreak@measure@{4}%
1262 }
1263 \def\eql@displaybreak@measure@{#1}{}
```

I.2 General Initialisation

TODO: describe

`\eql@vspaceskip@` (*skip*) **TODO:** add a proper star variant?!

`\eql@abovespace@` (*skip*)

`\eql@belowspace@` (*skip*)

```
1264 \newskip\eql@vspaceskip@
1265 \newskip\eql@abovespace@
1266 \newskip\eql@belowspace@
1267 \let\eql@vspace@org\vspace
1268 \def\eql@vspacef{\eql@ifstar@loose\eql@vspace@\eql@vspace@}
1269 \def\eql@vspace@#1{%
1270   \global\advance\eql@vspaceskip@\glueexpr#1\relax}
```

`\eql@display@init`

```
1271 \def\eql@display@init{%
1272   \eql@display@firstavail@\z@
1273   \eql@raisetag@firstlast@\z@}
```



```

1274 \let\displaybreak\eqldisplaybreak@print
1275 \eqldisplaybreak@open@\@MM
1276 \eqlvspaceskip@\z@skip
1277 \let\eqlvspace@org\vspace
1278 \let\vspace\eqlvspace
1279 }

```

`\eqldisplay@close` **TODO:** there seems to be an offset of 1em in predisplaysize towards actual content, nice.
TODO: must not use setlength or setcounter when calc is loaded

```

1280 \def\eqldisplay@close{%
1281 \ifdim\eqldisplay@firstavail@<\z@
1282 \eqldisplay@firstavail@\z@
1283 \fi
1284 \eqldisplay@mode@leave@\z@
1285 \ifdim\eqldisplay@prevdepth@=\maxdimen
1286 \ifdim\predisplaysize@=\maxdimen
1287 \eqldisplay@mode@above@\tw@
1288 \eqldisplay@mode@below@\eqldisplay@mode@cont@
1289 \else
1290 \eqldisplay@mode@above@\z@
1291 \eqldisplay@mode@below@\z@
1292 \advance\eqldisplay@firstavail@\displayindent
1293 \ifdim\eqldisplay@firstavail@>\predisplaysize
1294 % \TODO: process this here or in below ifcase for short?
1295 \ifcase\eqldisplay@mode@short@
1296 \or
1297 \eqldisplay@mode@above@\@ne
1298 \or
1299 \eqldisplay@mode@above@\@ne
1300 \ifnum\eqldisplay@row@=\tw@
1301 \eqldisplay@mode@below@\@ne
1302 \fi
1303 \or
1304 \eqldisplay@mode@above@\@ne
1305 \eqldisplay@mode@below@\@ne
1306 \fi
1307 \fi
1308 \fi
1309 \else
1310 \ifdim\eqldisplay@prevdepth@=-\@m@p@
1311 \eqldisplay@mode@above@4\relax
1312 \eqldisplay@mode@below@\eqldisplay@mode@top@
1313 \eqldisplay@mode@leave@\z@
1314 \predisplaysizepenalty\z@
1315 \ifcase\eqldisplay@mode@top@
1316 \or
1317 \or
1318 \or
1319 \eqldisplay@mode@leave@\@ne
1320 \or
1321 \postdisplaysizepenalty\z@
1322 \eqldisplay@mode@leave@\tw@
1323 \fi
1324 \else
1325 \eqldisplay@mode@above@\thr@@
1326 \eqldisplay@mode@below@\eqldisplay@mode@par@
1327 \eqldisplay@mode@leave@\z@
1328 \predisplaysizepenalty\z@

```

```

1329     \ifnum\eq@skip@mode@par@=\thr@@
1330     \eq@skip@mode@leave@\@ne
1331     \fi
1332   \fi
1333 \fi
1334 \ifdefined\eq@skip@force@above
1335   \eq@skip@mode@above\eq@skip@force@above\relax
1336 \fi
1337 \ifdefined\eq@skip@force@below
1338   \eq@skip@mode@below\eq@skip@force@below\relax
1339 \fi
1340 \ifdefined\eq@skip@force@leave
1341   \eq@skip@mode@leave\eq@skip@force@leave\relax
1342 \fi
1343 \ifodd\eq@raisetag@firstlast@
1344   \ifcase\eq@skip@mode@above@
1345     \abovedisplayskip\glueexpr\eq@skip@tag@above\relax
1346   \or
1347     \abovedisplayskip\glueexpr\eq@skip@tag@above\relax
1348   \or
1349     \abovedisplayskip\glueexpr\eq@skip@tag@above\relax
1350   \or
1351     \abovedisplayskip\glueexpr\eq@skip@partag@above\relax
1352   \or
1353     \abovedisplayskip\glueexpr\eq@skip@partag@above\relax
1354   \or
1355     \abovedisplayskip\z@skip
1356   \or
1357     \abovedisplayskip\glueexpr\eq@skip@medtag@above\relax
1358   \or
1359     \abovedisplayskip\glueexpr\eq@skip@custom@above\relax
1360   \fi
1361 \else
1362   \ifcase\eq@skip@mode@above@
1363     \abovedisplayskip\glueexpr\eq@skip@long@above\relax
1364   \or
1365     \abovedisplayskip\glueexpr\eq@skip@short@above\relax
1366   \or
1367     \abovedisplayskip\glueexpr\eq@skip@cont@above\relax
1368   \or
1369     \abovedisplayskip\glueexpr\eq@skip@par@above\relax
1370   \or
1371     \abovedisplayskip\glueexpr\eq@skip@top@above\relax
1372   \or
1373     \abovedisplayskip\z@skip
1374   \or
1375     \abovedisplayskip\glueexpr\eq@skip@medtag@above\relax
1376   \or
1377     \abovedisplayskip\glueexpr\eq@skip@custom@above\relax
1378   \fi
1379 \fi
1380 \ifnum\eq@raisetag@firstlast@>\@ne
1381   \ifcase\eq@skip@mode@below@
1382     \belowdisplayskip\glueexpr\eq@skip@tag@below\relax
1383   \or
1384     \belowdisplayskip\glueexpr\eq@skip@tag@below\relax
1385   \or
1386     \belowdisplayskip\glueexpr\eq@skip@tag@below\relax

```

```

1387 \or
1388 \belowdisplayskip\glueexpr\eq\skip@partag@below\relax
1389 \or
1390 \belowdisplayskip\glueexpr\eq\skip@partag@below\relax
1391 \or
1392 \belowdisplayskip\z\skip
1393 \or
1394 \belowdisplayskip\glueexpr\eq\skip@medtag@below\relax
1395 \or
1396 \belowdisplayskip\glueexpr\eq\skip@custom@below\relax
1397 \fi
1398 \else
1399 \ifcase\eq\skip@mode@below@
1400 \belowdisplayskip\glueexpr\eq\skip@long@below\relax
1401 \or
1402 \belowdisplayskip\glueexpr\eq\skip@short@below\relax
1403 \or
1404 \belowdisplayskip\glueexpr\eq\skip@cont@below\relax
1405 \or
1406 \belowdisplayskip\glueexpr\eq\skip@par@below\relax
1407 \or
1408 \belowdisplayskip\glueexpr\eq\skip@top@below\relax
1409 \or
1410 \belowdisplayskip\z\skip
1411 \or
1412 \belowdisplayskip\glueexpr\eq\skip@medtag@below\relax
1413 \or
1414 \belowdisplayskip\glueexpr\eq\skip@custom@below\relax
1415 \fi
1416 \fi
1417 \ifnum\eq\displaybreak@pen@=\@MM\else
1418 \postdisplaypenalty\eq\displaybreak@pen@
1419 \fi
1420 \ifnum\eq\displaybreak@prepen@=\@MM\else
1421 \predisplaypenalty\eq\displaybreak@prepen@
1422 \fi
1423 \advance\abovedisplayskip\eq\abovespace@
1424 \advance\belowdisplayskip\eq\belowspace@
1425 \advance\belowdisplayskip\eq\vspaceskip@
1426 \count@\prevgraf
1427 \advance\count@\eq\row@
1428 \advance\count@-\tw@
1429 \prevgraf\count@
1430 \global\eq\skip@mode@leave@\eq\skip@mode@leave@
1431 % \TODO temporary fix
1432 \ifdefined\eq\tagging@on
1433 \ifdefined\dollardollar@begin\else
1434 \belowdisplayskip-\belowdisplayskip
1435 \fi
1436 \fi
1437 }

1438 \def\eq\display@leave{%
1439 \let\label\eq\label@org
1440 \let\tag\eq\tag@default
1441 \let\raisetag\eq\raisetag@default
1442 \let\displaybreak\eq\displaybreak@default
1443 \let\intertext\eq\intertext@default
1444 \let\vspace\eq\vspace@org

```

```

1445 }
1446 \expandafter\def\expandafter\@arrayparboxrestore\expandafter{%
1447   \@arrayparboxrestore
1448   \eql@display@leave
1449   \ifdefined\eql@ampproof@active
1450     \eql@amprevert
1451   \fi
1452   \@displayfalse
1453 }

```

I.3 halign Support

TODO: describe

`\eql@strut` Next follows a special internal strut which is supposed to match the height and the depth `\eql@strutbox@` of a normal `\strut` minus `\normallineskiplimit` according to M. Spivak.

```

1454 \newbox\eql@strutbox@
1455 \def\eql@strut{\copy\eql@strutbox@}
1456 \let\eql@strut@field\eql@strut
1457 \let\eql@strut@tag\eql@strut
1458 \def\eql@strut@make{%
1459   \setbox\eql@strutbox@\hbox{%
1460     \@tempdima\normalbaselineskip
1461     \advance\@tempdima-\normallineskiplimit
1462     \@tempdimb.3\normalbaselineskip
1463     \advance\@tempdimb.5\normallineskiplimit
1464     \advance\@tempdima-\@tempdimb
1465     \vrule\@height\@tempdima\@depth\@tempdimb\@width\z@
1466   }
1467 }
1468 \AtBeginDocument{\eql@strut@make}

```

TODO: describe **TODO:** note on “`spread@equation`”

```

1469 \def\eql@halign@spread{%
1470   \eql@spread@amount@\glueexpr\eql@spread\relax
1471   \advance\eql@spread@amount@\normalbaselineskip
1472   \advance\eql@spread@amount@-\baselineskip
1473   \ifdim\eql@spread@amount@>\z@
1474     \openup\eql@spread@amount@
1475     \ifdefined\spread@equation
1476       \let\spread@equation\@empty
1477     \fi
1478   \fi
1479 }

```

`gn@prevdepth@` (*dimen*)

```

1480 \newdimen\eql@halign@prevdepth@
1481 \def\eql@halign@catchprevdepth{%
1482   \ifvmode
1483     \eql@halign@prevdepth@\prevdepth
1484     \nointerlineskip
1485     \noindent
1486   \else
1487     \eql@halign@prevdepth@\maxdimen
1488   \fi

```

```

1489 }

1490 \def\eql@halign@leave{%
1491   \ifcase\eql@skip@mode@leave@
1492   \or
1493     \endgraf
1494   \or
1495     \endgraf
1496     \prevdepth-\@m\p@
1497   \fi
1498 }

```

TODO: : how about penalty here? not for entry into display

```

1499 \def\eql@halign@before{%
1500   \ifdim\eql@halign@prevdepth@=\maxdimen\else
1501     \prevdepth\eql@halign@prevdepth@
1502   \fi
1503   \ifdim\prevdepth=-\@m\p@\else
1504     \ifdefined\eql@display@height
1505       \skip@\baselineskip
1506       \advance\skip@-\glueexpr\eql@display@height\relax
1507       \advance\skip@-\prevdepth\relax
1508       \ifdim\skip@<\lineskiplimit
1509         \skip@\lineskip
1510       \fi
1511       \advance\skip@-\eql@spread@amount@\relax
1512       \vskip\skip@
1513       \nointerlineskip
1514     \else
1515       \vskip-\eql@spread@amount@\relax
1516     \fi
1517   \fi
1518 }

```

TODO: describe

```

1519 \def\eql@halign@after{%
1520   \ifdefined\eql@display@depth
1521     \prevdepth\glueexpr\eql@display@depth\relax
1522   \fi
1523 }

```

TODO: describe

```

1524 \def\eql@halign@init#1{%
1525   \eql@halign@spread
1526   \eql@strut@make
1527   \everycr{\noalign{#1}}%
1528 }

```

I.4 Stack

TODO: describe

```

1529 \def\eql@stack@enable{%
1530   \let\eql@stack@save@single\eql@stack@save@single@
1531   \let\eql@stack@save@multi\eql@stack@save@multi@
1532   \let\eql@stack@save@boxed\eql@stack@save@boxed@
1533 }

```

TODO: describe

```
1534 \let\eqL@stack@save@single\eqL@stack@enable
1535 \let\eqL@stack@save@multi\eqL@stack@enable
1536 \let\eqL@stack@save@boxed\eqL@stack@enable
1537 \let\eqL@stack@restore\@empty
```

TODO: describe

```
1538 \def\eqL@stack@save@reg#1{\global#1\the#1\relax}
1539 \def\eqL@stack@save@let#1#2{\global\let\noexpand#2\noexpand#1}
```

TODO: describe

```
1540 \def\eqL@stack@save@single@{%
1541   \let\eqL@stack@nextlabel\eqL@nextlabel
1542   \let\eqL@stack@nexttag\eqL@nexttag
1543   \edef\eqL@stack@restore{%
1544     \global\if@eqnsw\noexpand\@eqnswtrue\else\noexpand\@eqnswfalse\fi
1545     \eqL@stack@save@let\eqL@stack@nextlabel\eqL@nextlabel
1546     \eqL@stack@save@let\eqL@stack@nexttag\eqL@nexttag
1547     \eqL@stack@save@reg\eqL@displaybreak@pen@
1548     \eqL@stack@save@reg\eqL@vspaceskip@
1549     \eqL@stack@save@reg\eqL@shape@pos@
1550     \eqL@stack@save@reg\eqL@shape@amount@
1551     \eqL@stack@save@reg\eqL@display@firstavail@
1552     \eqL@stack@save@reg\eqL@raisetag@amount@
1553     \eqL@stack@save@reg\eqL@raisetag@firstlast@
1554   }%
1555 }
```

TODO: describe

```
1556 \def\eqL@stack@save@multi@{%
1557   \let\eqL@stack@nextlabel\eqL@nextlabel
1558   \let\eqL@stack@nexttag\eqL@nexttag
1559   \let\eqL@stack@tagwidth@tab\eqL@tagwidth@tab
1560   \let\eqL@stack@fieldlength@tab\eqL@fieldlength@tab
1561   \let\eqL@stack@colwidth@tab\eqL@colwidth@tab
1562   \let\eqL@stack@label@thepage\eqL@label@thepage
1563   \let\eqL@stack@label@currentHref\eqL@label@currentHref
1564   \edef\eqL@stack@restore{%
1565     \global\if@eqnsw\noexpand\@eqnswtrue\else\noexpand\@eqnswfalse\fi
1566     \eqL@stack@save@let\eqL@stack@nextlabel\eqL@nextlabel
1567     \eqL@stack@save@let\eqL@stack@nexttag\eqL@nexttag
1568     \eqL@stack@save@let\eqL@stack@tagwidth@tab\eqL@tagwidth@tab
1569     \eqL@stack@save@let\eqL@stack@fieldlength@tab\eqL@fieldlength@tab
1570     \eqL@stack@save@let\eqL@stack@colwidth@tab\eqL@colwidth@tab
1571     \eqL@stack@save@let\eqL@stack@label@thepage\eqL@label@thepage
1572     \eqL@stack@save@let\eqL@stack@label@currentHref\eqL@label@currentHref
1573     \eqL@stack@save@reg\eqL@displaybreak@pen@
1574     \eqL@stack@save@reg\eqL@vspaceskip@
1575     \eqL@stack@save@reg\eqL@shape@pos@
1576     \eqL@stack@save@reg\eqL@shape@amount@
1577     \eqL@stack@save@reg\eqL@display@firstavail@
1578     \eqL@stack@save@reg\eqL@raisetag@amount@
1579     \eqL@stack@save@reg\eqL@raisetag@firstlast@
1580     \eqL@stack@save@reg\eqL@column@
1581     \eqL@stack@save@reg\eqL@totalcolumns@
1582     \eqL@stack@save@reg\eqL@line@avail@
1583     \eqL@stack@save@reg\eqL@line@pos@
```

```

1584 \eql@stack@save@reg\eql@line@width@
1585 \eql@stack@save@reg\eql@line@depth@
1586 \eql@stack@save@reg\eql@line@height@
1587 \eql@stack@save@reg\eql@tagwidth@max@
1588 \eql@stack@save@reg\eql@numbering@target@
1589 \eql@stack@save@reg\eql@row@
1590 \eql@stack@save@reg\eql@tagrows@
1591 }%
1592 }
1593 \def\eql@stack@save@boxed@{%
1594 \edef\eql@stack@restore{%
1595 \eql@stack@save@reg\eql@row@
1596 \eql@stack@save@reg\eql@totalrows@
1597 \eql@stack@save@reg\eql@shape@pos@
1598 \eql@stack@save@reg\eql@shape@amount@
1599 }%
1600 }

```

J Horizontal Spacing for Lines

The following code adjusts individual lines of equations for the equation and lines mode according to the selected layout and shape.

J.1 Supporting Definitionss

`\inf@bad` The `\inf@bad` constant is for testing overfull boxes:

```

1601 \ifdefined\inf@bad\else%
1602 \newcount\inf@bad
1603 \inf@bad1000000\relax
1604 \fi

```

`\eql@restore@hfuzz` We need to change the value of `\hfuzz` temporarily. The method `\eql@save@hfuzz` stores the value for recovery through `\eql@restore@hfuzz`:

```

1605 \let\eql@restore@hfuzz\empty
1606 \def\eql@save@hfuzz{\edef\eql@restore@hfuzz{\hfuzz\the\hfuzz\relax}}

```

`\eql@shape@pos@` (*dimen*) The registers `\eql@shape@pos@` and `\eql@shape@amount@` specify the currently selected horizontal alignment (0 for left, 1 for center, 2 for right) and the indentation amount, respectively:

```

1607 \newcount\eql@shape@pos@
1608 \newdimen\eql@shape@amount@

```

`\eql@marginleft@` (*dimen*) The registers `\eql@marginleft@` and `\eql@marginright@` store the intended left and right margin for the equation lines:

```

1609 \newdimen\eql@marginleft@
1610 \newdimen\eql@marginright@

```

`\eql@marginbadness@` The registers `\eql@marginbadness@` and `\eql@maxbadness@` store the allowable badness threshold for shrinking equation lines to the intended margin or to fit into the line at all before the tag is raised or lowered:

```

1611 \newcount\eql@marginbadness@
1612 \newcount\eql@maxbadness@
1613 \eql@marginbadness@\inf@bad
1614 \eql@maxbadness@\inf@bad

```

J.2 Shape Schemes

The horizontal alignment of each line is specified by a shape scheme.

`\eql@shape@tab@...` We select the scheme through a `\csname` selector with the following names:

```

1615 \def\eql@shape@tab@default{default}
1616 \def\eql@shape@tab@left{left}
1617 \def\eql@shape@tab@center{center}
1618 \def\eql@shape@tab@right{right}
1619 \def\eql@shape@tab@first{first}
1620 \def\eql@shape@tab@hanging{hanging}
1621 \def\eql@shape@tab@steps{steps}

```

For convenience, we add further alias names for the schemes:

```

1622 \let\eql@shape@tab@def\eql@shape@tab@default
1623 \let\eql@shape@tab@\eql@shape@tab@default
1624 \let\eql@shape@tab@l\eql@shape@tab@left
1625 \let\eql@shape@tab@c\eql@shape@tab@center
1626 \let\eql@shape@tab@r\eql@shape@tab@right
1627 \let\eql@shape@tab@rc\eql@shape@tab@first
1628 \let\eql@shape@tab@indent\eql@shape@tab@first
1629 \let\eql@shape@tab@hang\eql@shape@tab@hanging
1630 \let\eql@shape@tab@lc\eql@shape@tab@hanging
1631 \let\eql@shape@tab@outdent\eql@shape@tab@hanging
1632 \let\eql@shape@tab@lcr\eql@shape@tab@steps

```

`\eql@shape@mode` The currently selected scheme is stored in `\eql@shape@mode`. It is set to default:

```

1633 \let\eql@shape@mode\eql@shape@tab@default

```

`\eql@shape@set` Set the scheme via the translation table:

```

1634 \def\eql@shape@set#1{%
1635   \ifcsname eql@shape@tab@#1\endcsname
1636     \expandafter\let\expandafter\eql@shape@mode
1637     \csname eql@shape@tab@#1\endcsname
1638   \else
1639     \eql@error{shape ‘#1’ unknown: setting to default}%
1640     \let\eql@shape@mode\eql@shape@tab@default
1641   \fi
1642 }

```

`\eql@shape@center@...` Define the uniform shape schemes `left`, `center`, `right` and `default` for the central and

`\eql@shape@left@...` left alignment layout. The scheme functions determine the desired alignment and indentation for the current row:

```

1643 \def\eql@shape@center@left{\eql@shape@pos@z@\eql@shape@amount@z@}
1644 \def\eql@shape@center@center{\eql@shape@pos@ne\eql@shape@amount@z@}
1645 \def\eql@shape@center@right{\eql@shape@pos@tw@\eql@shape@amount@z@}
1646 \let\eql@shape@center@default\eql@shape@center@center
1647 \def\eql@shape@left@left{\eql@shape@pos@z@\eql@shape@amount@z@}

```



```

1648 \def\eqL@shape@left@center{\eqL@shape@pos@ \@ne\eqL@shape@amount@ \z@}
1649 \def\eqL@shape@left@right{\eqL@shape@pos@ \tw@ \eqL@shape@amount@ \z@}
1650 \let\eqL@shape@left@default\eqL@shape@left@left

```

The **first** scheme implements left alignment with indentation for the first line (unless there is only one line):

```

1651 \def\eqL@shape@center@first{%
1652   \eqL@shape@pos@ \z@
1653   \eqL@shape@amount@ \z@
1654   \ifnum\eqL@totalrows@ > \@ne
1655     \ifnum\eqL@row@ = \@ne
1656       \eqL@shape@amount@ \eqL@indent@
1657     \fi
1658   \fi
1659 }
1660 \def\eqL@shape@left@first{%
1661   \eqL@shape@pos@ \z@
1662   \eqL@shape@amount@ \z@
1663   \ifnum\eqL@totalrows@ > \@ne
1664     \ifnum\eqL@row@ = \@ne
1665       \eqL@shape@amount@ \eqL@indent@
1666     \fi
1667   \fi
1668 }

```

The **hanging** scheme implements left alignment with hanging indentation for the first line (unless there is only one line). In central alignment layout all but the first line are indented while in left aligned layout the first line has negative indentation:

```

1669 \def\eqL@shape@center@hanging{%
1670   \eqL@shape@pos@ \z@
1671   \eqL@shape@amount@ \eqL@indent@
1672   \ifnum\eqL@totalrows@ > \@ne
1673     \ifnum\eqL@row@ = \@ne
1674       \eqL@shape@amount@ \z@
1675     \fi
1676   \fi
1677 }
1678 \def\eqL@shape@left@hanging{%
1679   \eqL@shape@pos@ \z@
1680   \eqL@shape@amount@ \z@
1681   \ifnum\eqL@totalrows@ > \@ne
1682     \ifnum\eqL@row@ = \@ne
1683       \eqL@shape@amount@ - \eqL@indent@
1684     \fi
1685   \fi
1686 }

```

The **steps** scheme implements singles out the first and last lines which are shifted left and right, respectively. In central alignment layout the shift operates on the alignment whereas in left alignment layout the shift uses indentation:

```

1687 \def\eqL@shape@center@steps{%
1688   \eqL@shape@amount@ \z@
1689   \eqL@shape@pos@ \@ne
1690   \ifnum\eqL@totalrows@ > \@ne
1691     \ifnum\eqL@row@ = \@ne
1692       \eqL@shape@pos@ \z@
1693     \fi

```

```

1694 \ifnum\eqL@row@=\eqL@totalrows@
1695 \eqL@shape@pos@\tw@
1696 \fi
1697 \fi
1698 }
1699 \def\eqL@shape@left@steps{%
1700 \eqL@shape@pos@\z@
1701 \eqL@shape@amount@\z@
1702 \ifnum\eqL@totalrows@>\@ne
1703 \ifnum\eqL@row@=\@ne
1704 \eqL@shape@amount@-\eqL@indent@
1705 \fi
1706 \ifnum\eqL@row@=\eqL@totalrows@
1707 \eqL@shape@amount@\eqL@indent@
1708 \fi
1709 \fi
1710 }

```

`\eqL@shape@sel` Select the shape selector function for the current scheme `@\eqL@shape@mode` and layout `\eqL@shape@eval` and store it in `\eqL@shape@eval`:

```

1711 \let\eqL@shape@eval\@undefined
1712 \def\eqL@shape@sel{%
1713 \expandafter\let\expandafter\eqL@shape@eval
1714 \csname eqL@shape%
1715 @\ifdefined\eqL@flushleft left\else center\fi
1716 @\eqL@shape@mode\endcsname
1717 }

```

`\eqL@adjust@shoveleft` Adjust the alignment of the current equation line. For left alignment an optional argument `\eqL@adjust@shoveright` specifies the amount of indentation:
`\eqL@adjust@shovecenter`

```

1718 \def\eqL@adjust@shoveleft{%
1719 \global\eqL@shape@pos@\z@
1720 \eqL@srbgroup\eqL@ifstar@tight
1721 {\eqL@adjust@shoveleft@[ \eqL@indent@]}%
1722 {\eqL@ifnextgobble@tight{!}}%
1723 {\eqL@adjust@shoveleft@[- \eqL@indent@]}%
1724 {\eqL@testopt@tight\eqL@adjust@shoveleft@\z@}%
1725 }%
1726 }
1727 \def\eqL@adjust@shoveleft@[#1]{%
1728 \eqL@srbgroup\global\eqL@shape@amount@\glueexpr#1\relax}
1729 \def\eqL@adjust@shovecenter{%
1730 \global\eqL@shape@pos@\@ne\global\eqL@shape@amount@\z@}
1731 \def\eqL@adjust@shoveright{%
1732 \global\eqL@shape@pos@\tw@\global\eqL@shape@amount@\z@}

```

J.3 Adjustment Methods

`\eqL@adjust@try` Try to fit the current equation line in the available space. Argument `#1` specifies the amount of reserved space. Unpack the box `\eqL@fieldbox@`, replace the previous kerning with the new reserved space, and save the box back into `\eqL@fieldbox@`:

```

1733 \def\eqL@adjust@try#1{%
1734 \setbox\eqL@fieldbox@\hbox to\displaywidth{%
1735 \unhbox\eqL@fieldbox@\unkern\kern#1}%
1736 }

```

`\eql@adjust@print` We have found the final adjustment of the current line, so we typeset it with initial and final space adjustments #1 and #2, respectively. Restore the original value for `\hfuzz`:

```
1737 \def\eql@adjust@print#1#2{%
1738   \eql@restore@hfuzz
1739   \hbox to\displaywidth{%
1740     #1%
1741     \unhbox\eql@fieldbox@\unkern
1742     #2%
1743     \eql@tagging@mathaddlast
1744   }%
1745 }
```

`\just@print@alignleft` Fit the current equation line with the selected alignment within a given left and right `\st@print@aligncenter` margins #1 and #2. If we're on the first line, adjust `\eql@display@firstavail@` to the `\ust@print@alignright` minimum left available space we can guarantee:

```
1746 \def\eql@adjust@print@alignleft#1#2{%
1747   \ifnum\eql@row@=\@one
1748     \global\eql@display@firstavail@#1%
1749   \fi
1750   \eql@adjust@print{\kern#1}{\kern#2}%
1751 }
1752 \def\eql@adjust@print@alignright#1#2{%
1753   \ifnum\eql@row@=\@one
1754     \eql@display@firstavail@\displaywidth
1755     \advance\eql@display@firstavail@-\eql@fieldwidth@
1756     \global\advance\eql@display@firstavail@-#2%
1757   \fi
1758   \eql@adjust@print{\kern#1\hfil}{\unskip\kern#2}%
1759 }
1760 \def\eql@adjust@print@aligncenter#1#2{%
1761   \ifnum\eql@row@=\@one
1762     \eql@display@firstavail@\displaywidth
1763     \advance\eql@display@firstavail@-\eql@fieldwidth@
1764     \advance\eql@display@firstavail@#1%
1765     \advance\eql@display@firstavail@-#2%
1766     \global\divide\eql@display@firstavail@\tw@
1767   \fi
1768   \eql@adjust@print{\kern#1\hfil}{\kern#2}%
1769 }
```

`\eql@adjust@init` Initialise the horizontal adjustment framework. Turn off overfull box messages temporarily – otherwise there would be unwanted extra ones emitted during our measuring operations. Select the shape scheme:

```
1770 \def\eql@adjust@init{%
1771   \eql@save@hfuzz
1772   \hfuzz\maxdimen
1773   \eql@shape@sel
1774 }
```

`\eql@adjust@sel@tag` Select the appropriate adjustment method depending on the selected layout, selected tag `\eql@adjust@sel@notag` placement, current alignment position and on whether a tag is present or not:

```
1775 \def\eql@adjust@sel@tag{%
1776   \eql@tagging@tagaddbox
1777   \ifcase\eql@shape@pos@
1778     \eql@tagging@alignleft
```

```

1779 \or
1780 \eql@tagging@aligncenter
1781 \or
1782 \eql@tagging@alignright
1783 \fi
1784 \csname eql@adjust%
1785 @\ifdefined\eql@flushleft flushleft\else center\fi
1786 @\ifdefined\eql@tagsleft tagsleft\else tagsright\fi
1787 @\ifcase\eql@shape@pos@ shoveleft\or shovecenter\or shoveright\fi
1788 @tag\endcsname
1789 }
1790 \def\eql@adjust@sel@notag{%
1791 \eql@tagging@tagaddbox
1792 \ifcase\eql@shape@pos@
1793 \eql@tagging@alignleft
1794 \or
1795 \eql@tagging@aligncenter
1796 \or
1797 \eql@tagging@alignright
1798 \fi
1799 \csname eql@adjust%
1800 @\ifdefined\eql@flushleft flushleft\else center\fi
1801 @\ifdefined\eql@tagsleft tagsleft\else tagsright\fi
1802 @\ifcase\eql@shape@pos@ shoveleft\or shovecenter\or shoveright\fi
1803 @notag\endcsname
1804 }

```

\eql@adjust@calc **TODO:** any init needed for left alignment layout? marginleft is used per line!

```

1805 \def\eql@adjust@calc{%
1806 \ifdefined\eql@flushleft
1807 \eql@flushleftmargin@\glueexpr\eql@flushleftmargin@val\relax
1808 \else
1809 \eql@align@inter@\z@
1810 \eql@adjust@tagmargin
1811 \ifdefined\eql@paddingmax
1812 \eql@marginleft@\z@
1813 \eql@marginright@\z@
1814 \else
1815 \dimen@\displaywidth
1816 \advance\dimen@-\eql@totalwidth@
1817 \advance\dimen@-\eql@tagmargin@
1818 \divide\dimen@\tw@
1819 \eql@marginleft@\dimen@
1820 \advance\eql@marginleft@-\glueexpr\eql@paddingleft@val\relax
1821 \ifdim\eql@marginleft@<\z@
1822 \eql@marginleft@\z@
1823 \fi
1824 \eql@marginright@\dimen@
1825 \advance\eql@marginright@-\glueexpr\eql@paddingright@val\relax
1826 \ifdim\eql@marginright@<\z@
1827 \eql@marginright@\z@
1828 \fi
1829 \fi
1830 \fi
1831 }

```

J.4 Central Alignment Layout

TODO: describe

TODO: check all these!!

```
1832 \def\eq@adjust@center@tagsright@shovecenter@notag{%
1833   \dimen@\displaywidth
1834   \advance\dimen@-\eq@fieldwidth@
1835   \ifdim\dimen@>\eq@tagmargin@
1836     \eq@adjust@print@aligncenter\z@\eq@tagmargin@
1837   \else
1838     \eq@adjust@print@alignleft\z@\z@
1839   \fi
1840 }
```

TODO: describe

```
1841 \def\eq@adjust@center@tagsright@shovecenter@tag{%
1842   \dimen@\displaywidth
1843   \ifdim\eq@tagwidth@<\eq@tagmargin@
1844     \advance\dimen@-\eq@tagmargin@
1845   \else
1846     \advance\dimen@-2\eq@tagwidth@
1847     \advance\dimen@\eq@tagmargin@
1848   \fi
1849   \ifdim\eq@fieldwidth@<\dimen@
1850     \eq@adjust@print@aligncenter\z@\eq@tagmargin@
1851     \eq@tagbox@print@right
1852   \else
1853     \eq@adjust@try\eq@tagwidth@
1854     \ifnum\badness<\eq@maxbadness@
1855       \ifdim\eq@tagwidth@<\eq@tagmargin@
1856         \eq@adjust@print@alignleft\z@\eq@tagwidth@
1857       \else
1858         \eq@adjust@print@alignright\z@\eq@tagwidth@
1859       \fi
1860     \eq@tagbox@print@right
1861   \else
1862     \eq@adjust@center@tagsright@shovecenter@notag
1863     \eq@tagbox@print@right@raise
1864   \fi
1865 \fi
1866 }
```

TODO: describe

```
1867 \def\eq@adjust@center@tagsleft@shovecenter@notag{%
1868   \dimen@\displaywidth
1869   \advance\dimen@-\eq@tagmargin@
1870   \ifdim\eq@fieldwidth@<\dimen@
1871     \eq@adjust@print@aligncenter\eq@tagmargin@\z@
1872   \else
1873     \eq@adjust@print@alignright\z@\z@
1874   \fi
1875 }
```

TODO: describe

```
1876 \def\eq@adjust@center@tagsleft@shovecenter@tag{%
1877   \dimen@\displaywidth
```

```

1878 \ifdim\eq\tagwidth@<\eq\tagmargin@
1879 \advance\dimen@-\eq\tagmargin@
1880 \else
1881 \advance\dimen@-2\eq\tagwidth@
1882 \advance\dimen@\eq\tagmargin@
1883 \fi
1884 \ifdim\eq\fieldwidth@<\dimen@
1885 \eq\tagbox@print@left
1886 \eq\adjust@print@aligncenter\eq\tagmargin@\z@
1887 \else
1888 \eq\adjust@try\eq\tagwidth@
1889 \ifnum\badness<\eq\maxbadness@
1890 \eq\tagbox@print@left
1891 \ifdim\eq\tagwidth@<\eq\tagmargin@
1892 \eq\adjust@print@alignright\eq\tagwidth@\z@
1893 \else
1894 \eq\adjust@print@alignleft\eq\tagwidth@\z@
1895 \fi
1896 \else
1897 \eq\tagbox@print@left@raise
1898 \eq\adjust@center@tagsleft@shovecenter@notag
1899 \fi
1900 \fi
1901 \eq\display@firstavail@set\z@
1902 }

```

TODO: describe

```

1903 \def\eq\adjust@center@tagsright@shoveleft@notag{%
1904 \dimen@\displaywidth
1905 \advance\dimen@-\eq\marginleft@
1906 \advance\dimen@-\eq\shape@amount@
1907 \ifdim\eq\fieldwidth@<\dimen@
1908 \dimen@\eq\marginleft@
1909 \advance\dimen@\eq\shape@amount@
1910 \eq\adjust@print@alignleft\dimen@\z@
1911 \else
1912 \eq\adjust@print@alignright\z@\z@
1913 \fi
1914 }

```

TODO: describe

```

1915 \def\eq\adjust@center@tagsright@shoveleft@tag{%
1916 \dimen@\eq\marginleft@
1917 \advance\dimen@\eq\shape@amount@
1918 \advance\dimen@\eq\tagwidth@
1919 \eq\adjust@try\dimen@
1920 \ifnum\badness<\eq\marginbadness@
1921 \dimen@\eq\marginleft@
1922 \advance\dimen@\eq\shape@amount@
1923 \eq\adjust@print@alignleft\dimen@\eq\tagwidth@
1924 \eq\tagbox@print@right
1925 \else
1926 \ifdim\eq\marginleft@>-\eq\shape@amount@
1927 \eq\adjust@try\eq\tagwidth@
1928 \fi
1929 \ifnum\badness<\eq\maxbadness@
1930 \eq\adjust@print@alignright\z@\eq\tagwidth@
1931 \eq\tagbox@print@right

```

```

1932   \else
1933     \eql@adjust@center@tagsright@shoveleft@notag
1934     \eql@tagbox@print@right@raise
1935   \fi
1936 \fi
1937 }

```

TODO: describe

```

1938 \def\eql@adjust@center@tagsleft@shoveright@notag{%
1939   \dimen@ \displaywidth
1940   \advance\dimen@-\eql@tagmargin@
1941   \advance\dimen@-\eql@marginright@
1942   \ifdim\eql@fieldwidth@<\dimen@
1943     \eql@adjust@print@alignright\z@\eql@marginright@
1944   \else
1945     \eql@adjust@print@alignleft\z@\z@
1946   \fi
1947 }

```

TODO: describe

```

1948 \def\eql@adjust@center@tagsleft@shoveright@tag{%
1949   \dimen@ \eql@marginright@
1950   \advance\dimen@ \eql@tagwidth@
1951   \eql@adjust@try\dimen@
1952   \ifnum\badness<\eql@marginbadness@
1953     \eql@tagbox@print@left
1954     \eql@adjust@print@alignright\eql@tagwidth@\eql@marginright@
1955   \else
1956     \ifdim\eql@marginright@>\z@
1957       \eql@adjust@try\eql@tagwidth@
1958     \fi
1959     \ifnum\badness<\eql@maxbadness@
1960       \eql@tagbox@print@left
1961       \eql@adjust@print@alignleft\eql@tagwidth@\z@
1962     \else
1963       \eql@tagbox@print@left@raise
1964       \eql@adjust@center@tagsleft@shoveright@notag
1965     \fi
1966   \fi
1967 \eql@display@firstavail@set\z@
1968 }

```

TODO: describe

```

1969 \def\eql@adjust@center@tagsright@shoveright@notag{%
1970   \dimen@ \displaywidth
1971   \advance\dimen@-\eql@tagmargin@
1972   \advance\dimen@-\eql@marginright@
1973   \ifdim\eql@fieldwidth@<\dimen@
1974     \dimen@ \eql@tagmargin@
1975     \advance\dimen@ \eql@marginright@
1976     \eql@adjust@print@alignright\z@\dimen@
1977   \else
1978     \eql@adjust@print@alignleft\z@\z@
1979   \fi
1980 }

```

TODO: describe

```

1981 \def\eql@adjust@center@tagsright@shoveright@tag{%
1982   \dimen@ \eql@tagmargin@
1983   \advance\dimen@ \eql@marginright@
1984   \ifdim\eql@tagwidth@ < \dimen@
1985     \eql@adjust@try\dimen@%
1986     \ifnum\badness < \eql@marginbadness@
1987       \eql@adjust@print@alignright\z@ \dimen@
1988       \eql@tagbox@print@right
1989     \else
1990       \eql@adjust@try\eql@tagwidth@
1991       \ifnum\badness < \eql@maxbadness@
1992         \eql@adjust@print@alignleft\z@ \eql@tagwidth@
1993         \eql@tagbox@print@right
1994       \else
1995         \eql@adjust@print@alignleft\z@ \z@
1996         \eql@tagbox@print@left@raise
1997       \fi
1998     \fi
1999   \else
2000     \eql@adjust@try\eql@tagwidth@
2001     \ifnum\badness < \eql@maxbadness@
2002       \eql@adjust@print@alignright\z@ \eql@tagwidth@
2003       \eql@tagbox@print@right
2004     \else
2005       \eql@adjust@center@tagsright@shoveright@notag
2006       \eql@tagbox@print@right@raise
2007     \fi
2008   \fi
2009 }

```

TODO: describe

```

2010 \def\eql@adjust@center@tagsleft@shoveleft@notag{%
2011   \dimen@ \displaywidth
2012   \advance\dimen@ - \eql@tagmargin@
2013   \advance\dimen@ - \eql@marginleft@
2014   \advance\dimen@ - \eql@shape@amount@
2015   \ifdim\eql@fieldwidth@ < \dimen@
2016     \dimen@ \eql@tagmargin@
2017     \advance\dimen@ \eql@marginleft@
2018     \advance\dimen@ \eql@shape@amount@
2019     \eql@adjust@print@alignleft\dimen@\z@
2020   \else
2021     \eql@adjust@print@alignright\z@\z@
2022   \fi
2023 }

```

TODO: describe

```

2024 \def\eql@adjust@center@tagsleft@shoveleft@tag{%
2025   \dimen@ \eql@tagmargin@
2026   \advance\dimen@ \eql@marginleft@
2027   \advance\dimen@ \eql@shape@amount@
2028   \ifdim\eql@tagwidth@ < \dimen@
2029     \eql@adjust@try\dimen@%
2030     \ifnum\badness < \eql@marginbadness@
2031       \eql@tagbox@print@left
2032       \eql@adjust@print@alignleft\dimen@\z@
2033     \else
2034       \eql@adjust@try\eql@tagwidth@

```



```

2035     \ifnum\badness<\eql@maxbadness@
2036     \eql@tagbox@print@left
2037     \eql@adjust@print@alignright\eql@tagwidth@\z@
2038     \else
2039     \eql@tagbox@print@left@raise
2040     \eql@adjust@print@alignright\z@\z@
2041     \fi
2042     \fi
2043     \else
2044     \eql@adjust@try\eql@tagwidth@
2045     \ifnum\badness<\eql@maxbadness@
2046     \eql@tagbox@print@left
2047     \eql@adjust@print@alignleft\eql@tagwidth@\z@
2048     \else
2049     \eql@tagbox@print@left@raise
2050     \eql@adjust@center@tagsleft@shoveleft@notag
2051     \fi
2052     \fi
2053     \eql@display@firstavail@set\z@
2054 }

```

eql@adjust@tagmargin

```

2055 \def\eql@adjust@tagmargin{%
2056   \ifdefined\eql@tagmargin@val
2057     \eql@tagmargin@\glueexpr\eql@tagmargin@val\relax
2058   \else
2059     \eql@tagmargin@\eql@tagwidth@max@
2060     \ifdim\eql@tagmargin@>\z@
2061       \advance\eql@tagmargin@-\eql@tagsepmin@
2062     \fi
2063   \fi

2064   \dimen@\eql@tagrows@p@
2065   \ifnum\eql@totalrows@=\@ne
2066     \ifnum\eql@tagrows@=\@ne
2067       \advance\dimen@1sp\relax
2068     \fi
2069   \fi

2070   \ifdim\dimen@>\eql@totalrows@\eql@tagmargin@ratio@\else
2071     \eql@tagmargin@\z@
2072   \fi

2073   \@tempdima\displaywidth
2074   \advance\@tempdima-\eql@totalwidth@
2075   \advance\@tempdima-\eql@align@inter@\eql@colsepmin@
2076   \@tempdimb\@tempdima
2077   \advance\@tempdimb-\tw@\eql@tagmargin@
2078   \ifdim\@tempdimb>\z@
2079     \ifdim\eql@tagmargin@threshold\@tempdima<\@tempdimb
2080       \eql@tagmargin@\z@
2081     \fi
2082   \fi
2083 }

```

J.5 Left Alignment Layout

TODO: describe

```

2084 \def\eq@adjust@flushleft@shoveleft{%
2085   \eq@marginleft@\eq@flushleftmargin@
2086   \advance\eq@marginleft@\eq@shape@amount@
2087   \ifdim\eq@marginleft<\eq@flushleftmarginmin@
2088     \eq@marginleft@\eq@flushleftmarginmin@
2089   \fi
2090   \ifdim\eq@marginleft>\eq@flushleftmarginmax@
2091     \eq@marginleft@\eq@flushleftmarginmax@
2092   \fi
2093 }

```

TODO: perform checks based on unstretched dimension?! **TODO:** mention alternatives to fill; emphasis is on good left margin with ragged right (allow space between tag and equation in close case)

```

2094 \def\eq@adjust@flushleft@shoveleft@notag{%
2095   \ifdim\eq@flushleftmarginmin<\eq@marginleft@
2096     \eq@adjust@try\eq@marginleft@
2097     \ifnum\badness<\eq@marginbadness@
2098       \eq@adjust@print@alignleft\eq@marginleft@z@
2099     \else
2100       \eq@adjust@print@alignleft\eq@flushleftmarginmin@z@
2101     \fi
2102   \else
2103     \eq@adjust@print@alignleft\eq@marginleft@z@
2104   \fi
2105 }

```

TODO: describe

```

2106 \def\eq@adjust@flushleft@tagsright@shoveleft@notag{%
2107   \eq@adjust@flushleft@shoveleft
2108   \eq@adjust@flushleft@shoveleft@notag
2109 }
2110 \let\eq@adjust@flushleft@tagsleft@shoveleft@notag
2111 \eq@adjust@flushleft@tagsright@shoveleft@notag

```

TODO: what is worse, extend into margin or raise tag? this assumes raise tag, but other option might be better **TODO:** mention alternatives to fill; emphasis is on good left margin with ragged right (allow space between tag and equation in close case)

```

2112 \def\eq@adjust@flushleft@tagsright@shoveleft@tag{%
2113   \eq@adjust@flushleft@shoveleft
2114   \dimen@\eq@marginleft@
2115   \advance\dimen@\eq@tagwidth@
2116   \eq@adjust@try\dimen@
2117   \ifnum\badness<\eq@marginbadness@
2118     \eq@adjust@print@alignleft\eq@marginleft@\eq@tagwidth@
2119     \eq@tagbox@print@right
2120   \else
2121     \ifdim\eq@flushleftmarginmin<\eq@marginleft@
2122       \dimen@\eq@flushleftmarginmin@
2123       \advance\dimen@\eq@tagwidth@
2124       \eq@adjust@try\dimen@
2125     \fi
2126     \ifnum\badness<\eq@maxbadness@
2127       \eq@adjust@print@alignleft\eq@flushleftmarginmin@\eq@tagwidth@
2128       \eq@tagbox@print@right
2129     \else
2130       \eq@adjust@flushleft@shoveleft@notag

```

```

2131     \eql@tagbox@print@right@raise
2132     \fi
2133 \fi
2134 }

2135 \def\eql@adjust@flushleft@tagsleft@shoveleft@tag{%
2136 \eql@adjust@flushleft@shoveleft
2137 \ifdim\eql@tagwidth@<\eql@flushleftmarginmin@
2138 \eql@tagbox@print@left
2139 \eql@adjust@flushleft@notag
2140 \else
2141 \ifdim\eql@tagwidth@<\eql@marginleft@
2142 \eql@adjust@try\eql@marginleft@
2143 \ifnum\badness<\eql@marginbadness@
2144 \eql@tagbox@print@left
2145 \eql@adjust@print@alignleft\eql@marginleft@ \z@
2146 \else
2147 \eql@adjust@try\eql@tagwidth@
2148 \ifnum\badness<\eql@maxbadness@
2149 \eql@tagbox@print@left
2150 \eql@adjust@print@alignleft\eql@tagwidth@ \z@
2151 \else
2152 \eql@tagbox@print@left@raise
2153 \eql@adjust@print@alignleft\eql@flushleftmarginmin@ \z@
2154 \fi
2155 \fi
2156 \else
2157 \ifdim\eql@tagwidth@>\eql@flushleftmarginmax@
2158 \eql@tagbox@print@left@raise
2159 \eql@adjust@flushleft@shoveleft@notag
2160 \else
2161 \eql@adjust@try\eql@tagwidth@
2162 \ifnum\badness<\eql@maxbadness@
2163 \eql@tagbox@print@left
2164 \eql@adjust@print@alignleft\eql@tagwidth@ \z@
2165 \else
2166 \eql@tagbox@print@left@raise
2167 \eql@adjust@flushleft@shoveleft@notag
2168 \fi
2169 \fi
2170 \fi
2171 \fi
2172 \eql@display@firstavail@set \z@
2173 }

```

TODO: describe

```

2174 \def\eql@adjust@flushleft@shoveright@notag{%
2175 \eql@marginleft@\eql@flushleftmargin@
2176 \ifdim\eql@flushleftmarginmin@<\eql@marginleft@
2177 \eql@adjust@try\eql@marginleft@
2178 \ifnum\badness<\eql@marginbadness@
2179 \eql@adjust@print@alignright\eql@marginleft@ \z@
2180 \else
2181 \eql@adjust@print@alignright\eql@flushleftmarginmin@ \z@
2182 \fi
2183 \else
2184 \eql@adjust@print@alignright\eql@marginleft@ \z@
2185 \fi

```

```

2186 }
2187 \let\eql@adjust@flushleft@tagsright@shoveright@notag
2188 \eql@adjust@flushleft@shoveright@notag
2189 \let\eql@adjust@flushleft@tagsleft@shoveright@notag
2190 \eql@adjust@flushleft@shoveright@notag

```

TODO: describe

```

2191 \def\eql@adjust@flushleft@tagsright@shoveright@tag{%
2192 \dimen@ \eql@marginleft@
2193 \advance \dimen@ \eql@tagwidth@
2194 \eql@adjust@try \dimen@
2195 \ifnum \badness < \eql@marginbadness@
2196 \eql@adjust@print@alignright \eql@marginleft@ \eql@tagwidth@
2197 \eql@tagbox@print@right
2198 \else
2199 \ifdim \eql@flushleftmarginmin@ < \eql@marginleft@
2200 \dimen@ \eql@flushleftmarginmin@
2201 \advance \dimen@ \eql@tagwidth@
2202 \eql@adjust@try \dimen@
2203 \fi
2204 \ifnum \badness < \eql@maxbadness@
2205 \eql@adjust@print@alignright \eql@flushleftmarginmin@ \eql@tagwidth@
2206 \eql@tagbox@print@right
2207 \else
2208 \eql@adjust@flushleft@shoveright@notag
2209 \eql@tagbox@print@right@raise
2210 \fi
2211 \fi
2212 }

```

TODO: describe

```

2213 \def\eql@adjust@flushleft@tagsleft@shoveright@tag{%
2214 \ifdim \eql@tagwidth@ < \eql@flushleftmarginmin@
2215 \eql@tagbox@print@left
2216 \eql@adjust@flushleft@shoveright@notag
2217 \else
2218 \ifdim \eql@tagwidth@ < \eql@marginleft@
2219 \eql@adjust@try \eql@marginleft@
2220 \ifnum \badness < \eql@marginbadness@
2221 \eql@tagbox@print@left
2222 \eql@adjust@print@alignright \eql@marginleft@ \z@
2223 \else
2224 \eql@adjust@try \eql@tagwidth@
2225 \ifnum \badness < \eql@maxbadness@
2226 \eql@tagbox@print@left
2227 \eql@adjust@print@alignright \eql@tagwidth@ \z@
2228 \else
2229 \eql@tagbox@print@left@raise
2230 \eql@adjust@print@alignright \eql@flushleftmarginmin@ \z@
2231 \fi
2232 \fi
2233 \else
2234 \ifdim \eql@tagwidth@ > \eql@flushleftmarginmax@
2235 \eql@tagbox@print@left@raise
2236 \eql@adjust@flushleft@shoveright@notag
2237 \else
2238 \eql@adjust@try \eql@tagwidth@
2239 \ifnum \badness < \eql@maxbadness@

```

```

2240         \eql@tagbox@print@left
2241         \eql@adjust@print@alignright\eql@tagwidth@\z@
2242     \else
2243         \eql@tagbox@print@left@raise
2244         \eql@adjust@flushleft@shoveright@notag
2245     \fi
2246 \fi
2247 \fi
2248 \fi
2249 \eql@display@firstavail@set\z@
2250 }

2251 \def\eql@adjust@flushleft@shovecenter{%
2252 \eql@error{shove center not implemented for left alignment}%
2253 }
2254 \let\eql@adjust@flushleft@tagsright@shovecenter@notag
2255 \eql@adjust@flushleft@shovecenter
2256 \let\eql@adjust@flushleft@tagsright@shovecenter@tag
2257 \eql@adjust@flushleft@shovecenter
2258 \let\eql@adjust@flushleft@tagsleft@shovecenter@notag
2259 \eql@adjust@flushleft@shovecenter
2260 \let\eql@adjust@flushleft@tagsleft@shovecenter@tag
2261 \eql@adjust@flushleft@shovecenter

```

K Single-Line Equation

TODO: describe

K.1 Environment

```

2262 \def\eql@single@cr{%
2263 \eql@error{Cannot use '\string\\' within display equation.
2264 Please switch to equations environment}%
2265 }

```

TODO: describe

```

2266 \def\eql@single@start{%
2267 \eql@halign@catchprevdepth
2268 \eql@tagging@start
2269 \eql@dollar@dollar@begin
2270 \eql@numbering@eval@mode
2271 \let\eql@numbering@subeq@use\eql@false
2272 \eql@stack@save@single

```

TODO: make other display environments push these!?

```

2273 \eql@numbering@single@init
2274 \ifdefined\eql@single@native
2275 \let\eql@single@start@sel\eql@single@start@native
2276 \let\eql@single@end@sel\eql@single@end@native
2277 \let\raisetag\eql@raisetag@default
2278 \else
2279 \let\eql@single@start@sel\eql@single@start@adjust
2280 \let\eql@single@end@sel\eql@single@end@adjust
2281 \fi
2282 \ifdefined\eql@single@crerror\else
2283 \let\\eql@single@cr

```

```

2284 \fi
2285 \eql@single@start@sel
2286 }

2287 \def\eql@single@end{%
2288 \eql@punct@apply@block
2289 \eql@hook@eqout
2290 \eql@single@end@sel
2291 \eql@stack@restore
2292 \eql@dollar@dollar@end
2293 \eql@tagging@end
2294 \eql@halign@leave
2295 }

```

TODO: : try to feed in tagging after catchprevdepth

```

2296 \def\eql@single@main{%
2297 \expandafter\eql@single@start
2298 \eql@scan@body
2299 \eql@single@end
2300 }

```

K.2 Native

```

2301 \def\eql@single@start@native{%
2302 % \mathopen{}%
2303 \eql@hook@eqin
2304 }%

```

TODO: describe

```

2305 \def\eql@single@end@native{%
2306 % \mathclose{}%
2307 \if@eqnsw
2308 \ifdefined\eql@tagsleft
2309 \leqno
2310 \else
2311 \eqno
2312 \fi
2313 \eql@compose@print
2314 \fi
2315 \ifnum\eql@displaybreak@pen@=\@MM\else
2316 \postdisplaypenalty\eql@displaybreak@pen@
2317 \fi
2318 }%

```

K.3 Adjustment

```

2319 \def\eql@single@start@adjust{%
2320 \eql@totalrows@\@ne
2321 \eql@row@\z@
2322 \eql@display@init
2323 \let\shoveleft\eql@adjust@shoveleft
2324 \let\shovecenter\eql@adjust@shovecenter
2325 \let\shoveright\eql@adjust@shoveright
2326 \eql@adjust@init
2327 \eql@shape@eval
2328 \eql@halign@init{}%

2329 \eql@row@\@ne
2330 \setbox\eql@fieldbox@\hbox\bgroup
2331 \eql@restore@hfuzz

```

```

2332 \eql@strut@field
2333 $\m@th\displaystyle%$
2334 \eql@hook@eqin
2335 }

2336 \def\eql@single@end@adjust{%
2337 \eql@tagging@mathsave
2338 $%$
2339 \hfil
2340 \kern\z@
2341 \egroup

2342 \eql@fieldwidth@\wd\eql@fieldbox@
2343 \eql@line@height@\ht\eql@fieldbox@
2344 \eql@line@depth@\dp\eql@fieldbox@
2345 \eql@totalwidth@\eql@fieldwidth@
2346 \eql@totalrows@\@ne

2347 \if@eqnsw
2348 \eql@tagbox@make\eql@compose@print
2349 \eql@tagrows@\@ne
2350 \else
2351 \eql@tagwidth@\z@
2352 \eql@tagrows@\z@
2353 \fi
2354 \eql@tagwidth@max@\eql@tagwidth@

2355 \eql@adjust@calc

2356 \halign{##\cr
2357 \noalign{\eql@halign@before}%
2358 \if@eqnsw
2359 \eql@adjust@sel@tag
2360 \else
2361 \eql@adjust@sel@notag
2362 \fi
2363 \cr
2364 \noalign{\eql@halign@after}%
2365 \eql@tagging@tablesavelines
2366 }%
2367 \eql@row@tw@
2368 \eql@display@close
2369 }

```

L Multi-Line Support

TODO: describe

L.1 Registers

TODO: can we unite `\eql@fieldlength@tab` and `\eql@tagwidth@tab` **TODO:** then process sequentially not using ifcase?

```

\eql@fieldlength@tab
\eql@fieldlength@save
\eql@fieldlength@get
2370 \let\eql@fieldlength@tab\@empty
2371 \def\eql@fieldlength@save#1{%
2372 \beginngroup
2373 \let\or\relax
2374 \global\edef\eql@fieldlength@tab{%

```

```

2375     \eql@fieldlength@tab
2376     \ifnum#1=\@ne
2377     \or
2378     \else
2379     ,%
2380     \fi
2381     \the\wd\eql@fieldbox@
2382     }%
2383 \endgroup
2384 }
2385 \def\eql@fieldlength@get#1{%
2386 \ifcase\expandafter#1\eql@fieldlength@tab\fi
2387 }

```

\eql@tagwidth@get
\eql@tagwidth@save

```

2388 \let\eql@tagwidth@tab\@empty
2389 \def\eql@tagwidth@get#1{%
2390 \ifcase\expandafter#1\eql@tagwidth@tab\fi
2391 }
2392 \def\eql@tagwidth@save{%
2393 \begingroup
2394 \let\or\relax
2395 \global\edef\eql@tagwidth@tab{\eql@tagwidth@tab\or\the\eql@tagwidth@}%
2396 \endgroup
2397 }
2398 \def\eql@tagwidth@savezero{%
2399 \begingroup
2400 \let\or\relax
2401 \global\edef\eql@tagwidth@tab{\the\eql@tagwidth@\eql@tagwidth@tab}%
2402 \endgroup
2403 }

```

\eql@colwidth@tab

```

2404 \let\eql@colwidth@tab\@empty

```

\eql@align@colwidth@get

```

2405 \def\eql@align@colwidth@get#1{%
2406 \ifcase\expandafter#1\eql@colwidth@tab\else\z@\fi
2407 }
2408 \def\eql@align@colwidth@save{%
2409 \begingroup
2410 \let\or\relax
2411 \global\edef\eql@colwidth@tab{\or\the\wd\thr@@\eql@colwidth@tab}%
2412 \endgroup
2413 }

```

L.2 Measure Support

TODO: describe

```

2414 \def\eql@measure@init#1{%
2415 \measuring@true
2416 \eql@row@\z@
2417 \let\displaybreak\eql@displaybreak@measure
2418 \tabskip\z@skip

```



```

2419 \everycr{%
2420   \noalign{%
2421     \global\advance\eq@row@\@ne
2422     #1%
2423   }%
2424 }%
2425 }

```

sure@restorecounters
measure@savecounters

```

2426 \let\eq@measure@restorecounters@\@empty
2427 \def\eq@measure@savecounters{%
2428   \begingroup
2429   \def\@elt##1{%
2430     \global\csname c@##1\endcsname\the\csname c@##1\endcsname}%
2431   \global\edef\@gtempa{%
2432     \cl@@ckpt
2433     \let\noexpand\eq@measure@restorecounters\noexpand@\@empty
2434   }%
2435   \endgroup
2436   \let\eq@measure@restorecounters@\@gtempa
2437 }

```

L.3 Print Support

TODO: describe

eq@print@inithalign

```

2438 \def\eq@print@init#1{%
2439   \eq@row@z@
2440   \eq@halign@init{%
2441     \global\eq@displaybreak@open@\@MM
2442     \global\advance\eq@row@\@ne
2443     #1%
2444   }%
2445 }

```

```

2446 \def\eq@print@overfull{%
2447   \dimen@\eq@line@width@
2448   \advance\dimen@-\hfuzz
2449   \ifdim\dimen@>\displaywidth
2450     \setbox\z@\hbox to\displaywidth{\hbox to\eq@line@width@{\hfil}}%
2451     \wd\z@\z@
2452     \ht\z@\eq@line@height@
2453     \dp\z@\eq@line@depth@
2454     \box\z@
2455   \fi
2456 }

```

l@tagbox@print@multi

```

2457 \def\eq@tagbox@print@multi{%
2458   \advance\eq@tagwidth@-\eq@tagfuzz@
2459   \ifdefined\eq@tagsleft
2460     \eq@display@firstavail@set\z@
2461     \ifdim\eq@tagwidth@>\eq@line@avail@
2462       \eq@tagbox@print@left@raise

```

```

2463 \else
2464 \eql@tagbox@print@left
2465 \fi
2466 \kern\displaywidth
2467 \else
2468 \kern\displaywidth
2469 \advance\eql@tagwidth@\eql@line@width@
2470 \ifdim\eql@tagwidth@>\displaywidth
2471 \eql@tagbox@print@right@raise
2472 \else
2473 \eql@tagbox@print@right
2474 \fi
2475 \fi
2476 }

```

L.4 Line Breaks

TODO: describe

```

\eql@math@cr
2477 \protected\def\eql@math@cr{%
2478 \eql@ampprotecttwo\eql@teststaropt@tight
2479 {\global\eql@displaybreak@pen@\@M\eql@math@cr@}\eql@math@cr@\z@}

\eql@math@cr@
2480 \def\eql@math@cr@[#1]{%
2481 \eql@math@cr@@@
2482 \cr
2483 \noalign{%
2484 \ifnum\eql@displaybreak@pen@=\@MM
2485 \penalty\interdisplaylinepenalty
2486 \else
2487 \penalty\eql@displaybreak@pen@
2488 \fi
2489 \advance\eql@vspaceskip@\glueexpr#1\relax%
2490 \vskip\eql@vspaceskip@
2491 \global\eql@vspaceskip@\z@skip
2492 }%
2493 }

\eql@let@cr
2494 \def\eql@let@cr#1{%
2495 \let\\\eql@math@cr
2496 \let\eql@math@cr@@@#1%
2497 }

```

L.5 Intertext

TODO: describe

TODO: revert in everymath?

```

2498 \def\eql@intertext@default{\eql@error{Invalid use of \string\intertext}}
2499 \eql@amsmath@let\intertext\eql@intertext@default

```

TODO: why does it fail in measuring? total width?! determine total width otherwise!?

```
2500 \def\eq@intertext@process{%
2501   \eq@math@cr@@@
2502   \cr
2503   \ifmeasuring@
2504     \expandafter\@gobble
2505   \else
2506     \expandafter\eq@intertext@print
2507   \fi
2508 }
```

TODO: describe **TODO:** prevgraf **TODO:** prevdepth **TODO:** does this have to be in a vbox? **TODO:** vskip and penalty opposite order **TODO:** can we handle short?

```
2509 \def\eq@intertext@print#1{%
2510   \noalign{%
2511     \eq@halign@after
2512     \let\eq@skip@force@below\z@
2513     \let\eq@skip@force@above\z@
2514     \eq@setkeys{intertext}\eq@intertext@opt
2515     \openup-\eq@spread@amount@
2516     \penalty\postdisplaypenalty
2517     \ifcase\eq@skip@force@below\relax
2518       \advance\eq@vspaceskip@\glueexpr\eq@skip@long@below\relax
2519     \or
2520       \advance\eq@vspaceskip@\glueexpr\eq@skip@short@below\relax
2521     \or
2522       \advance\eq@vspaceskip@\glueexpr\eq@skip@cont@below\relax
2523     \or
2524       \advance\eq@vspaceskip@\glueexpr\eq@skip@par@below\relax
2525     \or
2526       \advance\eq@vspaceskip@\glueexpr\eq@skip@top@below\relax
2527     \or
2528       \advance\eq@vspaceskip@\z@skip
2529     \or
2530       \advance\eq@vspaceskip@\glueexpr\eq@skip@med@below\relax
2531     \or
2532       \advance\eq@vspaceskip@\glueexpr\eq@skip@custom@below\relax
2533     \fi
2534     \vskip\eq@vspaceskip@
2535     \global\eq@vspaceskip@\z@skip
2536     \vbox{%
2537       \@parboxrestore
2538       \ifdim
2539         \ifdim\@totalleftmargin=\z@\linewidth\else-\maxdimen\fi=\columnwidth
2540       \else
2541         \parshape\@ne
2542         \@totalleftmargin\linewidth
2543       \fi
2544       \noindent
2545       \ignorespaces
2546       #1%
2547     \par
2548   }%
2549   \penalty\predisplaypenalty
2550   \ifcase\eq@skip@force@above\relax
2551     \vskip\glueexpr\eq@skip@long@above\relax
2552   \or
```

```

2553     \vskip\glueexpr\eq\@skip@short@above\relax
2554 \or
2555     \vskip\glueexpr\eq\@skip@cont@above\relax
2556 \or
2557     \vskip\glueexpr\eq\@skip@par@above\relax
2558 \or
2559     \vskip\glueexpr\eq\@skip@top@above\relax
2560 \or
2561     \vskip\z@skip
2562 \or
2563     \vskip\glueexpr\eq\@skip@med@above\relax
2564 \or
2565     \vskip\glueexpr\eq\@skip@custom@above\relax
2566 \fi
2567 % \eq\@halign@prevdepth@\maxdimen
2568 \eq\@halign@prevdepth@\z@
2569 \eq\@halign@before
2570 }
2571 }

```

TODO: describe

```

2572 \newenvironment{eq\@intertext}{%
2573 \eq\@testopt@tight\eq\@intertext@{}}%
2574 }{%
2575 \aftergroup\eq\@intertext@after
2576 \ignorespacesafterend
2577 }

```

TODO: describe

```

2578 \def\eq\@intertext@env{intertext}
2579 \def\eq\@intertext@[#1]{%
2580 \global\def\eq\@intertext@opt{#1}%
2581 \ifx\@currenvir\eq\@intertext@env
2582 \expandafter\eq\@scan@env\expandafter\eq\@intertext@inject
2583 \else
2584 \expandafter\eq\@intertext@process
2585 \fi
2586 }

```

TODO: describe

```

2587 \def\eq\@intertext@inject{%
2588 \global\edef\eq\@intertext@after{%
2589 \noexpand\eq\@intertext@process{%
2590 \ifx\eq\@scan@body\eq\@scan@body@dump
2591 \eq\@scan@body@dump
2592 \else
2593 \noexpand\scantokens{\eq\@scan@body@dump}%
2594 \fi
2595 }%
2596 }%
2597 }

```

L.6 Main

TODO: note that switching from align to lines mode, the width can be incorrect due to different formatting (punctuation only?!). only minor discrepancies expected and lines can

adjust

`\eql@multi@main`

```
2598 \let\eql@multi@mode@lines\eql@false
2599 \def\eql@multi@main{%
2600   \eql@halign@catchprevdepth
2601   \eql@tagging@start
2602   \eql@dollar@begin
2603   \eql@numbering@eval@mode
2604   \eql@stack@save@multi
2605   \ifdefined\eql@subequations@active
2606     \let\eql@numbering@subeq@use\eql@false
2607   \fi
2608   \ifdefined\eql@numbering@subeq@use
2609     \eql@numbering@subeq@init
2610   \fi
2611   \let\intertext\eql@intertext
2612   \let\endintertext\endeql@intertext
2613   \let\shoveleft\eql@adjust@shoveleft
2614   \let\shovecenter\eql@adjust@shovecenter
2615   \let\shoveright\eql@adjust@shoveright
2616   \ifdefined\eql@multi@mode@lines
2617     \expandafter\eql@lines@measure
2618   \else
2619     \ifdefined\eql@ampproof@active
2620       \eql@ampproof
2621     \fi
2622     \expandafter\eql@align@measure
2623   \fi
2624   \ifx\eql@numbering@subeq@use\@ne
2625     \eql@numbering@subeq@revert
2626   \fi
2627   \ifdefined\eql@multi@mode@lines\else
2628     \ifdefined\eql@multi@lines@fallback
2629       \ifnum\eql@totalcolumns@=\@ne
2630         \let\eql@multi@mode@lines\eql@true
2631         \eql@shape@set{r}%
2632       \eql@lines@measure
2633     \fi
2634   \fi
2635   \fi
2636   \ifdefined\eql@multi@mode@lines
2637     \expandafter\eql@lines@print
2638   \else
2639     \expandafter\eql@align@print
2640   \fi
2641   \ifdefined\eql@numbering@subeq@use
2642     \eql@numbering@subeq@close
2643   \fi
2644   \eql@stack@restore
2645   \eql@dollar@end
2646   \eql@tagging@end
2647   \eql@halign@leave
2648 }
```

TODO: describe

```
2649 \def\eql@mode@equation{%
2650   \let\eql@equations@mode@single\eql@true
```

```

2651 \ifdefined\eql@single@doscan
2652   \let\eql@equations@main\eql@single@main
2653   \let\eql@equations@end\@empty
2654 \else
2655   \let\eql@equations@main\@undefined
2656   \let\eql@equations@end\eql@single@end
2657 \fi
2658 }
2659 \def\eql@mode@align{%
2660   \let\eql@equations@mode@single\eql@false
2661   \let\eql@multi@mode@lines\eql@false
2662   \let\eql@equations@main\eql@multi@main
2663   \let\eql@equations@end\@empty
2664 }
2665 \def\eql@mode@lines{%
2666   \let\eql@equations@mode@single\eql@false
2667   \let\eql@multi@mode@lines\eql@true
2668   \let\eql@equations@main\eql@multi@main
2669   \let\eql@equations@end\@empty
2670 }
2671 \eql@mode@align

```

M Multi-Line Lines Mode

M.1 Measure

TODO: describe

```

2672 \def\eql@lines@measure@line@begin{%
2673 (dev)\eql@dev{starting line \the\eql@row}%
2674 \eql@numbering@measure@line@begin
2675 \eql@hook@linein
2676 }

```

TODO: describe

```

2677 \def\eql@lines@measure@line@end{%
2678 \eql@punct@apply@line
2679 \eql@hook@lineout
2680 }

```

TODO: describe

```

2681 \def\eql@lines@measure@field{%
2682 \kern\wd\eql@fieldbox@
2683 }

```

TODO: describe

```

2684 \def\eql@lines@measure@tag{%
2685 \ifnum\eql@numbering@target@<\z@
2686 \if@eqnsw
2687 \eql@tagbox@make\eql@compose@measure
2688 \ifdim\eql@tagwidth@>\eql@tagwidth@max@
2689 \global\eql@tagwidth@max@\eql@tagwidth@
2690 \fi
2691 \global\advance\eql@tagrows@\@ne
2692 \else

```

```

2693     \eql@tagwidth@z@
2694     \fi
2695 \fi
2696 }

```

\eql@lines@measure

```

2697 \def\eql@lines@measure{%
2698 (dev)\eql@dev@enter\eql@lines@measure
2699 \eql@tagwidth@max@z@
2700 \eql@tagrows@z@
2701 \eql@measure@savecounters
2702 \setboxz@vbox{%
2703 \eql@numbering@measure@init
2704 \eql@measure@init\eql@lines@measure@line@begin
2705 \eql@let@cr\eql@lines@measure@line@end
2706 \halign{%
2707 \setbox\eql@fieldbox@hbox{%
2708 \@lign
2709 $\m@th\displaystyle
2710 \eql@hook@colin
2711 ##%
2712 \eql@punct@apply@col
2713 \eql@hook@colout
2714 $%
2715 }%
2716 \eql@lines@measure@field
2717 \eql@lines@measure@tag
2718 \crr
2719 \noalign{%
2720 \eql@hook@blockbefore
2721 }%
2722 \eql@hook@blockin
2723 \eql@scan@body
2724 \ifvmode\else
2725 \eql@punct@apply@block
2726 \eql@hook@blockout
2727 \eql@lines@measure@line@end
2728 \cr
2729 \fi
2730 \omit
2731 \cr
2732 \noalign{%
2733 \eql@hook@blockafter
2734 }%
2735 }%
2736 \global\advance\eql@row@-\tw@
2737 \eql@numbering@measure@eval
2738 \ifnum\eql@numbering@target@>z@
2739 \eql@tagbox@make\eql@compose@measure
2740 \global\eql@tagwidth@max@\eql@tagwidth@
2741 \global\eql@tagrows@\@ne
2742 \fi
2743 }%
2744 \eql@totalrows@\eql@row@
2745 \ifdefined\eql@numbering@subeq@use
2746 \eql@numbering@subeq@test
2747 \fi
2748 \eql@measure@restorecounters

```

```

2749 \setbox\z@\vbox{%
2750 \unvbox\z@
2751 \unpenalty
2752 \global\setbox\@ne\lastbox
2753 }%
2754 \eq\@totalwidth@\wd\@ne

2755 (dev)\eq\@dev\@leave\eq\@lines\@measure
2756 }

```

M.2 Print

TODO: describe

mes@print@line@begin

```

2757 \def\eq\@lines\@print\@line\@begin{%
2758 (dev)\eq\@dev\@starting\line\@the\eq\@row{%
2759 \eq\@numbering\@print\@line\@begin
2760 \eq\@hook\@linein
2761 }

```

TODO: describe

```

2762 \def\eq\@lines\@print\@line\@end{%
2763 \eq\@punct\@apply\@line
2764 \eq\@hook\@lineout
2765 }

```

TODO: describe

```

2766 \def\eq\@lines\@print\@line\@adjust{%
2767 \eq\@numbering\@print\@line\@eval
2768 \eq\@fieldwidth@\wd\eq\@fieldbox@
2769 \eq\@line\@height@\ht\eq\@fieldbox@
2770 \eq\@line\@depth@\dp\eq\@fieldbox@
2771 \if\@eq\@sw
2772 \eq\@tagbox\@make\eq\@compose\@print
2773 \eq\@adjust\@sel\@tag
2774 \else
2775 \eq\@adjust\@sel\@notag
2776 \fi
2777 }

```

TODO: describe

```

2778 \def\eq\@lines\@print{%
2779 (dev)\eq\@dev\@enter\eq\@lines\@print
2780 \eq\@display\@init
2781 \eq\@adjust\@init
2782 \eq\@adjust\@calc
2783 \eq\@numbering\@print\@init
2784 \eq\@print\@init\eq\@lines\@print\@line\@begin
2785 \eq\@let\@cr\eq\@lines\@print\@line\@end
2786 \tabskip\z@skip
2787 \halign{%
2788 \eq\@shape\@eval
2789 \setbox\eq\@fieldbox@\hbox{%
2790 \eq\@restore\@hfuzz
2791 \eq\@strut\@field

```



```

2792     \@lign
2793     $\m@th\displaystyle
2794     \eql@hook@colin
2795     ##%
2796     \eql@punct@apply@col
2797     \eql@hook@colout
2798     \eql@tagging@mathsave
2799     $%
2800     \hfil
2801     \kern\z@
2802     }%
2803     \eql@lines@print@line@adjust
2804     \crr
2805     \noalign{%
2806     \eql@halign@before
2807     \eql@numbering@print@block@begin
2808     \eql@hook@blockbefore
2809     }%
2810 % \TODO relax? leavevmode?!
2811     \eql@hook@blockin
2812     \eql@scan@body
2813     \ifvmode\else
2814     \eql@punct@apply@block
2815     \eql@hook@blockout
2816     \eql@lines@print@line@end
2817     \cr
2818     \fi
2819     \noalign{%
2820     \eql@hook@blockafter
2821     \eql@halign@after
2822 (dev)\eql@dev@leave\eql@lines@print
2823     }%
2824     \eql@tagging@tablesavelines
2825     }%
2826     \eql@display@close
2827 }

```

N Multi-Line Align Mode

TODO: describe

N.1 Columns Processing

TODO: describe

```

\l@align@completerow
\l@align@add@amp
2828 \def\l@align@add@amp#1{\if m#1&\omit\expandafter\l@align@add@amp\fi}
2829 \def\l@align@completerow{%
2830 \count@\l@totalcolumns@
2831 \advance\count@-\l@column@
2832 \advance\count@\@ne
2833 \edef\l@tmp{%
2834 \expandafter\l@align@add@amp\romannumeral\number\count@ 000q}%
2835 \l@tmp
2836 }

```

N.2 Measure

TODO: describe

n@measure@line@begin

```
2837 \def\eq@align@measure@line@begin{%
2838 (dev)\eq@dev{starting line \the\eq@row}%
2839 \global\eq@column@z@
2840 \eq@numbering@measure@line@begin
2841 \eq@hook@linein
2842 }
```

```
2843 \def\eq@align@measure@field{%
2844 \eq@fieldlength@save\eq@column@
2845 \kern\wd\eq@fieldbox@
2846 }
```

ign@measure@line@end

```
2847 \def\eq@align@measure@line@end{%
2848 \eq@punct@apply@line
2849 \eq@hook@lineout
2850 &\omit
2851 \ifnum\eq@column@>\eq@totalcolumns@
2852 \global\eq@totalcolumns@\eq@column@
2853 \fi
2854 \eq@align@measure@tag
2855 }
```

ql@align@measure@tag

```
2856 \def\eq@align@measure@tag{%
2857 \ifnum\eq@numbering@target@<\z@
2858 \if@eqnsw
2859 \eq@tagbox@make\eq@compose@measure
2860 \ifdim\eq@tagwidth@>\eq@tagwidth@max@
2861 \global\eq@tagwidth@max@\eq@tagwidth@
2862 \fi
2863 \global\advance\eq@tagrows@\@ne
2864 \else
2865 \eq@tagwidth@z@
2866 \fi
2867 \eq@tagwidth@save
2868 \fi
2869 }
```

\eq@align@measure

```
2870 \def\eq@align@measure{%
2871 (dev)\eq@dev@center\eq@align@measure
2872 \eq@totalcolumns@z@
2873 \eq@tagwidth@max@z@
2874 \let\eq@tagwidth@tab@empty
2875 \let\eq@fieldlength@tab@empty
2876 \eq@tagrows@z@
2877 \eq@measure@savecounters
2878 \setbox\z@\vbox{%
2879 \eq@numbering@measure@init
```

```

2880 \eql@measure@init\eql@align@measure@line@begin
2881 \eql@let@cr\eql@align@measure@line@end
2882 \tabskip\z@skip
2883 \halign{%
2884   &%
2885   \global\advance\eql@column@\@ne
2886   \hfil
2887   \global\setbox\eql@fieldbox@\hbox{%
2888     \@lign
2889     $\m@th\displaystyle
2890     \eql@hook@colin
2891     ##%
2892     \eql@class@innerleft
2893     \eql@hook@innerleft
2894     $%
2895   }%
2896   \global\eql@fieldwidth@\wd\eql@fieldbox@
2897   \eql@align@measure@field
2898   &%
2899   \global\advance\eql@column@\@ne
2900   \setbox\eql@fieldbox@\hbox{%
2901     \@lign
2902     $\m@th\displaystyle
2903     \eql@hook@innerright
2904     \eql@class@innerright@sel
2905     ##%
2906     \eql@punct@apply@col
2907     \eql@hook@colout
2908     $%
2909   }%
2910   \eql@align@measure@field
2911   \hfil
2912   \crrc
2913   \noalign{%
2914     \eql@hook@blockbefore
2915   }%
2916   \eql@hook@blockin
2917   \eql@scan@body

```

TODO: test for vmode okay?!

```

2918   \ifvmode\else
2919     \eql@punct@apply@block
2920     \eql@hook@blockout
2921     \eql@align@measure@line@end
2922     \cr
2923   \fi
2924   \noalign{%
2925     \eql@hook@blockafter
2926   }%

```

TODO: should we enforce even columns already here?! **TODO:** should we guard against no columns at all?!

```

2927   \eql@align@completerow
2928   \cr
2929 }%
2930 \global\advance\eql@row@-\tw@
2931 \eql@numbering@measure@eval
2932 \ifnum\eql@numbering@target@>\z@

```

```

2933     \eql@tagbox@make\eql@compose@measure
2934     \global\eql@tagwidth@max@\eql@tagwidth@
2935     \global\eql@tagrows@\@ne
2936     \eql@tagwidth@savezero
2937     \fi
2938 }%
2939 \eql@totalrows@\eql@row@
2940 \ifdefined\eql@numbering@subeq@use
2941     \eql@numbering@subeq@test
2942     \fi
2943 \eql@measure@restorecounters
2944 % \eql@totalwidth@\wd\z@

2945 \setbox\z@\vbox{%
2946     \unvbox\z@
2947     \unpenalty
2948     \global\setbox\@ne\lastbox
2949 }%
2950 \eql@totalwidth@\wd\@ne

```

TODO: why not recycle box contents altogether?!

```

2951 \let\eql@colwidth@tab\@empty
2952 \loop
2953     \setbox\@ne\hbox{%
2954         \unhbox\@ne
2955         \unskip
2956         \global\setbox\thr@\@lastbox
2957     }%
2958 \ifhbox\thr@@
2959     \eql@align@colwidth@save
2960 \repeat

2961 (dev)\eql@dev@leave\eql@align@measure
2962 }

```

N.3 Print

TODO: describe

`ign@print@line@begin`

```

2963 \def\eql@align@print@line@begin{%
2964 (dev)\eql@dev{starting line \the\eql@row@}%
2965     \global\eql@column@\z@
2966     \global\eql@line@pos@\eql@marginleft@
2967     \global\eql@line@width@\z@
2968     \global\eql@line@avail@\eql@totalwidth@
2969     \global\eql@line@height@\z@
2970     \global\eql@line@depth@\z@
2971     \eql@numbering@print@line@begin
2972     \eql@hook@linein
2973 }

```

`ql@align@print@field`

```

2974 \def\eql@align@print@field{%

```

determine available and used space

```
2975 \dimen@eql@align@colwidth@get\eql@column@\relax
2976 \ifdim\wd\eql@fieldbox>\z@
2977 \ifdim\eql@line@width=\z@
2978 \eql@line@avail@\eql@line@pos@
2979 \ifodd\eql@column@
2980 \advance\eql@line@avail@\dimen@
2981 \advance\eql@line@avail@-\wd\eql@fieldbox@
2982 \fi
2983 \global\eql@line@avail@\eql@line@avail@
2984 \fi
2985 \eql@line@width@\eql@line@pos@
2986 \ifodd\eql@column@
2987 \advance\eql@line@width@\dimen@
2988 \else
2989 \advance\eql@line@width@\wd\eql@fieldbox@
2990 \fi
2991 \global\eql@line@width@\eql@line@width@
2992 \fi
2993 \advance\eql@line@pos@\dimen@
2994 \ifodd\eql@column@\else
2995 \advance\eql@line@pos@\eql@colsep@
2996 \fi
2997 \global\eql@line@pos@\eql@line@pos@
```

update height and depth

```
2998 \ifdim\ht\eql@fieldbox>\eql@line@height@
2999 \global\eql@line@height@\ht\eql@fieldbox@
3000 \fi
3001 \ifdim\dp\eql@fieldbox>\eql@line@depth@
3002 \global\eql@line@depth@\dp\eql@fieldbox@
3003 \fi
```

print box enforce given width: hopefully measure was correct, but need a precise width for tag placement

```
3004 %
3005 % \box\eql@fieldbox@
3006 %
3007 % \dimen@\eql@align@colwidth@get\eql@column@\relax
3008 % \advance\dimen@-\wd\eql@fieldbox@
3009 % \ifodd\eql@column@
3010 % \kern\dimen@
3011 % \box\eql@fieldbox@
3012 % \else
3013 % \box\eql@fieldbox@
3014 % \kern\dimen@
3015 % \fi
3016 %
3017 % \dimen@\eql@align@colwidth@get\eql@column@\relax
3018 % \ifodd\eql@column@
3019 % \kern\dimen@
3020 % \else
3021 % \advance\dimen@-\wd\eql@fieldbox@
3022 % \box\eql@fieldbox@
3023 % \kern\dimen@
3024 % \fi
3025 %
```

```

3026 }

3027 \def\eq@align@print@trailright{%
3028   &\omit
3029   \global\advance\eq@column@\@ne
3030   \setbox\eq@fieldbox@\hbox{%
3031     \kern-\wd\eq@fieldbox@\box\eq@fieldbox@
3032   }%
3033   \eq@align@print@field
3034 }

```

\eq@align@print@line@end

```

3035 \def\eq@align@print@line@end{%
3036   \eq@punct@apply@line
3037   \eq@hook@lineout
3038   % \TODO add an even column with empty stuff if box processing deferred
3039   \ifodd\eq@column@
3040     \expandafter\eq@align@print@trailright
3041   \fi
3042   \eq@align@completerow
3043   \eq@align@print@tag
3044 }

```

\eq@align@print@tag

```

3045 \def\eq@align@print@tag{%
3046   \dimen@\eq@totalwidth@
3047   \advance\dimen@\eq@colsep@
3048   \kern-\dimen@

```

determine first line available space

```

3049   \eq@display@firstavail@set\eq@line@avail@
3050   \eq@print@overfull
3051   \eq@numbering@print@line@eval
3052   \if@eqnsw
3053     \eq@tagbox@make\eq@compose@print
3054     \eq@tagging@tagaddbox
3055     \eq@tagbox@print@multi
3056   \else
3057     \eq@tagging@tagaddbox
3058     \kern\displaywidth
3059   \fi
3060 }

```

\eq@align@print

```

3061 \def\eq@align@print{%
3062 (dev)\eq@dev@enter\eq@align@print
3063   \eq@align@adjust
3064   \eq@display@init
3065   \eq@numbering@print@init
3066   \eq@print@init\eq@align@print@line@begin
3067   \eq@let@cr\eq@align@print@line@end
3068   \tabskip\eq@marginleft@
3069   \halign{%
3070     &%
3071     \global\advance\eq@column@\@ne
3072     \hfil

```

```

3073 \global\setbox\eqL@fieldbox@\hbox{%
3074 \eqL@strut@field
3075 \@lign
3076 $\m@th\displaystyle
3077 \eqL@hook@colin
3078 ##%
3079 \eqL@class@innerleft
3080 \eqL@hook@innerleft
3081 \eqL@tagging@mathsave
3082 $%
3083 \eqL@tagging@mathaddlast
3084 }%
3085 \global\eqL@fieldwidth@\wd\eqL@fieldbox@
3086 \eqL@align@print@field
3087 \tabskip\z@skip
3088 &%
3089 \global\advance\eqL@column@\@ne
3090 \setbox\eqL@fieldbox@\hbox{%
3091 % \TODO printing left field in right field
3092 \kern-\wd\eqL@fieldbox@
3093 \box\eqL@fieldbox@
3094 \eqL@strut@field
3095 \@lign
3096 $\m@th\displaystyle
3097 \eqL@hook@innerright
3098 \eqL@class@innerright@sel
3099 ##%
3100 \eqL@punct@apply@col
3101 \eqL@hook@colout
3102 \eqL@tagging@mathsave
3103 $%
3104 \eqL@tagging@mathaddlast
3105 }%
3106 \eqL@align@print@field
3107 \hfil
3108 \tabskip\eqL@colsep@\relax
3109 \crrc
3110 \noalign{%
3111 \eqL@halign@before
3112 \eqL@numbering@print@block@begin
3113 \eqL@hook@blockbefore
3114 }%
3115 \eqL@hook@blockin
3116 \eqL@scan@body
3117 \ifvmode\else
3118 \eqL@punct@apply@block
3119 \eqL@hook@blockout
3120 \eqL@align@print@line@end
3121 \cr
3122 \fi
3123 \noalign{%
3124 \eqL@hook@blockafter
3125 \eqL@halign@after
3126 (dev)\eqL@dev@leave\eqL@align@print
3127 }%
3128 \eqL@tagging@tablesavelign
3129 }%
3130 \eqL@display@close

```

3131 }

N.4 Adjust

TODO: describe **TODO:** does this respect the margin for numbers in centre mode?

```
3132 \def\eql@align@adjust{%
```

```
3133   \eql@colsepmin@\glueexpr\eql@colsepmin@val\relax
```

TODO: shouldn't we do this earlier for access to last column?

```
3134   \ifodd\eql@totalcolumns@
```

```
3135     \advance\eql@totalcolumns@\@ne
```

```
3136   \fi
```

TODO: should we guard against no columns?!

```
3137   \ifnum\eql@totalcolumns@<\thr@@
```

```
3138     \let\eql@align@margin\eql@true
```

```
3139   \fi
```

Determine the number of intercolumn spaces `\eql@align@inter@`:

```
3140   \eql@align@inter@\eql@totalcolumns@
```

```
3141   \divide\eql@align@inter@\tw@
```

```
3142   \advance\eql@align@inter@\m@ne
```

```
3143   \ifdefined\eql@flushleft
```

```
3144     \eql@flushleftmargin@\glueexpr\eql@flushleftmargin@val\relax
```

```
3145   \else
```

```
3146     \eql@adjust@tagmargin
```

```
3147   \fi
```

```
3148   \eql@colsep@\displaywidth
```

```
3149   \advance\eql@colsep@-\eql@totalwidth@
```

```
3150   \ifdefined\eql@flushleft
```

```
3151     \advance\eql@colsep@-\eql@flushleftmargin@
```

```
3152   \else
```

```
3153     \advance\eql@colsep@-\eql@tagmargin@
```

```
3154     \ifdefined\eql@align@margin\else
```

```
3155       \ifdim\eql@tagmargin@>\z@
```

```
3156         \advance\eql@colsep@-\eql@tagsepmin@
```

```
3157       \fi
```

```
3158     \fi
```

```
3159   \fi
```

```
3160   \count@\eql@align@inter@
```

```
3161   \ifdefined\eql@align@margin
```

```
3162     \ifdefined\eql@flushleft
```

```
3163       \advance\count@\@ne
```

```
3164     \else
```

```
3165       \advance\count@\tw@
```

```
3166     \fi
```

```
3167   \fi
```

```
3168   \divide\eql@colsep@\count@
```

TODO: here or above, this code does not make much sense if there is a single column. nevertheless it works using the following code. yet it could be cleaner to treat a single column separately (may be some distinctions based on flush left)


```

3169 \ifdim\eq\colsep@<\eq\colsepmin@
3170 \eq\colsep@\eq\colsepmin@
3171 \else
3172 \dimen@\glueexpr\eq\colsepmax@val\relax
3173 \ifdim\eq\colsep@>\dimen@
3174 \eq\colsep@\dimen@
3175 \fi
3176 \fi

```

Now determine the left margin `\eq\marginleft@` **TODO**: complete

```

3177 \ifdefined\eq\flushleft
3178 \ifdim\eq\colsep@=\eq\colsepmin@
3179 \eq\marginleft@\displaywidth
3180 \advance\eq\marginleft@-\eq\totalwidth@
3181 \advance\eq\marginleft@-\eq\align@inter@\eq\colsep@
3182 \ifdim\eq\marginleft@>\eq\flushleftmargin@
3183 \eq\marginleft@\eq\flushleftmargin@
3184 \else
3185 \ifdim\eq\marginleft@<\eq\flushleftmarginmin@
3186 \eq\marginleft@\eq\flushleftmarginmin@
3187 \fi
3188 \fi
3189 \else
3190 \eq\marginleft@\eq\flushleftmargin@
3191 \fi
3192 \else
3193 \ifdefined\eq\align@argins
3194 \eq\marginleft@\displaywidth
3195 \advance\eq\marginleft@-\eq\totalwidth@
3196 \advance\eq\marginleft@-\eq\align@inter@\eq\colsep@
3197 \ifdim\eq\marginleft@<\eq>tagmargin@
3198 \eq\marginleft@\z@
3199 \else
3200 \advance\eq\marginleft@-\eq>tagmargin@
3201 \divide\eq\marginleft@\tw@
3202 \fi
3203 \ifdefined\eq>tagleft
3204 \advance\eq\marginleft@\eq>tagmargin@
3205 \fi
3206 \else
3207 \ifdefined\eq>tagleft
3208 \eq\marginleft@\eq>tagmargin@
3209 \ifdim\eq>tagmargin@>\z@
3210 \advance\eq\marginleft@\eq>tagsepmin@
3211 \fi
3212 \else
3213 \eq\marginleft@\z@
3214 \fi
3215 \fi
3216 \fi

```

Loop through the rows and adjust the intercolumn and margin space to make the tags fit into the available space at the corresponding side as far as possible:

```

3217 \ifdefined\eq>tagleft
3218 \let\eq\align@adjust@test\eq\align@adjust@test>tagleft
3219 \else
3220 \let\eq\align@adjust@test\eq\align@adjust@test>tagright
3221 \fi

```

```

3222 \eql@row@\eql@totalrows@
3223 \loop\ifnum\eql@row@>\z@

```

Fetch the tag width for the current row depending on whether there are tags for individual rows or one overall tag. If a tag is present, compute the available space and try to adjust spaces if needed:

```

3224 \ifnum\eql@numbering@target@<\z@
3225 \eql@tagwidth@\eql@tagwidth@get\eql@row@\relax
3226 \else
3227 \ifnum\eql@numbering@target@=\eql@row@
3228 \eql@tagwidth@\eql@tagwidth@get\z@\relax
3229 \fi
3230 \fi
3231 \ifdim\eql@tagwidth@>\z@
3232 \eql@align@adjust@calc
3233 \eql@align@adjust@test
3234 \fi
3235 \advance\eql@row@\m@ne
3236 \repeat

```

From now on `\eql@totalwidth@` will include the left margin and the total intercolumn separation:

```

3237 \advance\eql@totalwidth@\eql@align@inter@\eql@colsep@
3238 \advance\eql@totalwidth@\eql@marginleft@
3239 }

```

Calc Space.

`\eql@align@adjust@calc` Compute the space that is available at the beginning and at the end of a particular row `\eql@row@`. The space available at the beginning is returned in `\eql@line@avail@` and `\@tempcnta` counts the available intercolumn spaces whose width is not contained in `\eql@line@avail@` because it is still flexible at this stage. The total used width is returned in `\eql@line@width@` and `\@tempcntb` describes the last used intercolumn space. The available space at the end of the row is given as the difference to `\eql@totalwidth@` and `\eql@align@inter@`:

```

3240 \def\eql@align@adjust@calc{%
3241 \eql@line@pos@\z@
3242 \eql@column@\z@
3243 \eql@line@avail@\eql@totalwidth@
3244 \@tempcnta\eql@totalcolumns@
3245 \eql@line@width@\z@
3246 \@tempcntb\z@
3247 \edef\@tempb{\eql@fieldlength@get\eql@row@}%
3248 \for\@tempa:=\@tempb\do
3249 \eql@align@adjust@calc@col
3250 \advance\@tempcnta\m@ne
3251 \divide\@tempcnta\tw@
3252 \advance\@tempcntb\m@ne
3253 \divide\@tempcntb\tw@
3254 }

```

`\eql@align@adjust@width@col` The macro `\eql@align@adjust@width@col` iterates over columns. When a non-blank field is encountered, the available space on the left will be fixed if it is still undetermined, and the total width is updated to the current position:

```

3255 \def\eql@align@adjust@calc@col{%
3256   \advance\eql@column@\@ne
3257   \@tempdima\@tempa\relax
3258   \dimen@\eql@align@colwidth@get\eql@column@\relax
3259   \ifdim\@tempdima>\z@
3260     \ifdim\eql@line@width@=\z@
3261       \eql@line@avail@\eql@line@pos@
3262       \@tempcnta\eql@column@
3263       \ifodd\eql@column@
3264         \advance\eql@line@avail@\dimen@
3265         \advance\eql@line@avail@-\@tempdima
3266       \fi
3267     \fi
3268     \eql@line@width@\eql@line@pos@
3269     \@tempcntb\eql@column@
3270     \ifodd\eql@column@
3271       \advance\eql@line@width@\dimen@
3272     \else
3273       \advance\eql@line@width@\@tempdima
3274     \fi
3275   \fi
3276   \advance\eql@line@pos@\dimen@
3277 }

```

Placement for Right Tags.

`\adjust@test@tagright` Test whether the spacing can be adjusted to make the current row fit:

```
3278 \def\eql@align@adjust@test@tagright{%
```

The register `\@tempdima` will hold the amount of available space.

```

3279   \@tempdima\displaywidth
3280   \advance\@tempdima-\eql@line@width@
3281   \advance\@tempdima-\eql@tagwidth@

```

Test whether the space at the end of the row is sufficient to hold the tag with the current settings.

```

3282   \dimen@\eql@marginleft@
3283   \advance\dimen@\@tempcntb\eql@colsep@
3284   \ifdim\dimen@>\@tempdima

```

If not, determine whether the row and tag may at all fit into the available space with minimal intercolumn spaces and minimal left margin (in left alignment layout).

```

3285     \dimen@\@tempcntb\eql@colsepmin@
3286     \ifdefined\eql@flushleft
3287       \advance\dimen@\eql@flushleftmarginmin@
3288     \fi
3289     \ifdim\dimen@>\@tempdima\else

```

If so, hand over to `\eql@align@adjust@modify@tagright`.

```

3290       \eql@align@adjust@modify@tagright
3291     \fi
3292   \fi
3293 }

```

`\adjust@modify@tagright` Adjust the intercolumn space and left margin to make the row fit.

```
3294 \def\eq@align@adjust@modify@tagright{%
```

If there are any intercolumn spaces that contribute to the available space, determine how much intercolumn separation would be needed while keeping the current left margin fixed (in left alignment layout). In central alignment layout, assume that the left margin will be adjusted to match the intercolumn separation by stepping the number of columns to divide by.

```
3295 \ifnum\@tempcntb>\z@
3296   \dimen@\@tempdima
3297   \count@\@tempcntb
3298   \ifdefined\eq@flushleft
3299     \advance\dimen@-\eq@marginleft@
3300   \else
3301     \ifdefined\eq@align@margin
3302       \advance\count@\@one
3303     \fi
3304   \fi
3305   \divide\dimen@\count@
```

If smaller, reduce the intercolumn separation, but make sure to not exceed the minimum allowed value.

```
3306   \ifdim\dimen@<\eq@colsep@
3307     \ifdim\dimen@<\eq@colsepmin@
3308       \dimen@\eq@colsepmin@
3309     \fi
3310     \eq@colsep@\dimen@
3311   \fi
3312 \fi
```

Now adjust the left margin as much as needed to fit the contents.

```
3313 \dimen@\@tempdima
3314 \advance\dimen@-\@tempcntb\eq@colsep@
3315 \ifdim\dimen@<\eq@marginleft@
3316   \eq@marginleft@\dimen@
3317 \fi
3318 }
```

Placement for Left Tags.

`\adjust@test@tagleft` Test whether the spacing can be adjusted to make the current row fit:

```
3319 \def\eq@align@adjust@test@tagleft{%
```

The register `\@tempdima` will hold the deficit amount of space at the beginning of the row without adjustable space, and the register `\count@` will hold the number of intercolumn spaces that would contribute to space adjustments.

```
3320 \count@\eq@align@inter@
3321 \advance\count@-\@tempcnta
3322 \@tempdima\eq@tagwidth@
3323 \advance\@tempdima-\eq@line@avail@
```

Test whether the space at the beginning of the row is sufficient to hold the tag with the current settings.

```
3324 \dimen@\eq@marginleft@
3325 \advance\dimen@\@tempcnta\eq@colsep@
3326 \ifdim\dimen@<\@tempdima
```

If not, first verify that the tag will fit the line (or the maximal left margin in left alignment layout).

```

3327   \ifdefined\eq\flushleft
3328     \dimen@\eq\flushleftmarginmax@
3329   \else
3330     \dimen@\displaywidth
3331   \fi
3332   \ifdim\dimen@>\eq\tagwidth@

```

If so, determine whether the row and tag may at all fit into the available space with minimal intercolumn spaces.

```

3333     \dimen@\count@\eq\colsepmin@
3334     \advance\dimen@\eq\totalwidth@
3335     \advance\dimen@\@tempdima
3336     \ifdim\dimen@>\displaywidth\else

```

If so, hand over to `\eq\align@adjust@modify@tagleft`.

```

3337     \eq\align@adjust@modify@tagleft
3338   \fi
3339 \fi
3340 \fi
3341 }

```

TODO: implement a maximum shift (if tag+sep exceeds max, don't adjust) **TODO:** could this mechanism possibly shift any longer line past the margin?!

`\adjust@modify@tagleft` Adjust the intercolumn space and left margin to make the row fit.

```

3342 \def\eq\align@adjust@modify@tagleft{%

```

If there are any intercolumn spaces that contribute to the available space, determine how much intercolumn separation would be needed while keeping the current right margin fixed. In central alignment layout, assume that the right margin will be adjusted to match the intercolumn separation by stepping the number of columns to divide by.

```

3343   \ifnum\count@>\z@
3344     \dimen@\displaywidth
3345     \advance\dimen@-\eq\totalwidth@
3346     \advance\dimen@-\@tempdima
3347     \ifdefined\eq\align@margins
3348       \advance\count@\@ne
3349     \fi
3350     \divide\dimen@\count@

```

If smaller, reduce the intercolumn separation, but make sure to not exceed the minimum allowed value. Also adjust the left margin to keep the right margin fixed.

```

3351     \ifdim\dimen@<\eq\colsep@
3352       \ifdim\dimen@<\eq\colsepmin@
3353         \dimen@\eq\colsepmin@
3354       \fi
3355       \advance\dimen@-\eq\colsep@
3356       \advance\eq\marginleft@-\eq\align@inter@\dimen@
3357       \advance\eq\colsep@\dimen@
3358     \fi
3359 \fi

```

Now adjust the left margin as much as needed to fit the contents.

```

3360 \dimen@-\@tempcnta\eq@colsep@
3361 \advance\dimen@\@tempdima
3362 \ifdim\dimen@>\eq@marginleft@
3363   \eq@marginleft@\dimen@
3364 \fi
3365 }

```

O Interface

O.1 Scanning the Equation Body

The multi-line equation environment must scan its body twice: once to determine how wide the columns are and then to actually typeset them. This means that we must collect all text in this body before calling the environment macros. The mechanism and its description follows `amsmath` closely.

Token Register.

`\eq@scan@reg@` We start by defining a token register to hold the equation body.

```
3366 \newtoks\eq@scan@reg@
```

`\eq@scan@body@dump` The macro `\eq@scan@body@dump` dumps the equation body from the register so that we do not have to pass it around in arguments. The macro `\eq@scan@body@rescan` rescans the tokens so that special commands such as `\verb` can be processed properly. The register `\eq@scan@body` holds the currently selected mode of operation:

```

3367 \def\eq@scan@body@dump{\the\eq@scan@reg@}
3368 \def\eq@scan@body@rescan{%
3369   \expandafter\scantokens\expandafter{\the\eq@scan@reg@}}
3370 \let\eq@scan@body\eq@scan@body@dump

```

`\eq@scan@addto` We define a macro to append to the token register `\eq@scan@reg@`:

```
3371 \long\def\eq@scan@addto#1{\eq@scan@reg@\expandafter{\the\eq@scan@reg@#1}}
```

Environment Body. The following mechanism scans the contents of an environment taking into account nested environments that may be contained in the body.

`\eq@scan@env` The macro `\eq@scan@env` starts the scan for the `\end{...}` command of the current environment. The argument is a call-back macro to process the body in `\eq@scan@reg@`:

```

3372 \def\eq@scan@env#1{%
3373   (dev)\eq@dev@enter\eq@scan@env
3374   \def\eq@scan@end{#1\expandafter\end\expandafter{\@currenenv}}%
3375   \eq@scan@reg@{\def\eq@scan@stack{b}}%

```

We call `\eq@scan@env@iterate` which will scan until the next occurrence of `\end` and then count the number of occurrences of `\begin` before `\end` in `\eq@scan@stack`. If we simply called `\eq@scan@env@iterate` directly, the error message for an unwanted `\par` token (usually from a blank line) would refer to `\eq@scan@env@iterate` which would not be illuminating. We use a little finesse to get a more intelligible error message: We use the actual environment name as the name of the temporary function that is `\let` to `\eq@scan@env@iterate`:

```

3376 \edef\eql@scan@iterate{\expandafter\noexpand\csname\@currenvir\endcsname}%
3377 \expandafter\let\expandafter\eql@scan@org\eql@scan@iterate
3378 \expandafter\let\eql@scan@iterate\eql@scan@env@iterate
3379 \eql@scan@iterate
3380 }

```

`\eql@scan@env@iterate` `\eql@scan@env@iterate` takes two arguments: the first will consist of all text up to the next `\end` command, the second will be the `\end` command's argument. If there are any extra `\begin` commands in the body text, a marker is pushed onto a stack via `\eql@scan@env@count`. An empty state for this stack means that we have reached the `\end` that matches our original `\begin`. Otherwise we need to include the `\end` and its argument in the material that we are adding to our environment body accumulator:

```

3381 \def\eql@scan@env@iterate#1\end#2{%
3382 \edef\eql@scan@stack{%
3383 \eql@scan@env@count#1\begin\end\expandafter\@gobble\eql@scan@stack}%
3384 \ifx\@empty\eql@scan@stack
3385 \@checkend{#2}%
3386 \eql@scan@addto{#1}%
3387 \expandafter\let\eql@scan@iterate\eql@scan@env@org
3388 (dev)\eql@dev@leave\eql@scan@env
3389 \expandafter\eql@scan@end
3390 \else
3391 \eql@scan@addto{#1\end{#2}}%
3392 \expandafter\eql@scan@iterate
3393 \fi
3394 }

```

`\eql@scan@env@count` When adding a piece of the current environment's contents to `\eql@scan@reg@`, we scan it to check for additional `\begin` tokens, and add a 'b' to the stack for any that we find.

```

3395 \long\def\eql@scan@env@count#1\begin#2{%
3396 \ifx\end#2\else b\expandafter\eql@scan@env@count\fi
3397 }

```

The call-back macro `\eql@scan@env@cancel` ignores the body as well as the end clause for the environment:

```

3398 \def\eql@scan@env@cancel{%
3399 \@namedef{end\@currenvir}{\ignorespacesafterend}%
3400 }

```

Square Brackets. The following is a version of the above mechanism that scans for an equation body enclosed by `\[. . .\]` paying attention to potential further instances of the square bracket enclosures contained in the body.

`\eql@scan@sqr` Start scanning for `\]`:

```

3401 \def\eql@scan@sqr#1{%
3402 (dev)\eql@dev@enter\eql@scan@sqr
3403 \def\eql@scan@end{#1\]}%
3404 \eql@scan@reg@{\def\eql@scan@stack{b}%
3405 \let\eql@scan@sqr@org\[%\]
3406 \let\[%\eql@scan@sqr@iterate%\]
3407 \[%\]
3408 }

```

Iterate until we find a balanced pairing of square brackets. Then call the call-back macro:

```

3409 \def\eq@scan@sqr@iterate#1\]{%
3410 \edef\eq@scan@stack{%
3411 \eq@scan@sqr@count#1\[\]\expandafter\@gobble\eq@scan@stack}%
3412 \ifx\@empty\eq@scan@stack
3413 \let\[\eq@scan@sqr@org%\]
3414 \eq@scan@addto{#1}%
3415 (dev)\eq@dev@leave\eq@scan@sqr
3416 \expandafter\eq@scan@end
3417 \else
3418 \eq@scan@addto{#1\]}%
3419 \expandafter\[%\]
3420 \fi
3421 }

```

Push a ‘b’ for every encountered instance of ‘\[':

```

3422 \long\def\eq@scan@sqr@count#1\[#2{\%]
3423 \ifx\]#2\else b\expandafter\eq@scan@sqr@count\fi
3424 }

```

The call-back macro `\eq@scan@sqrang@cancel` ignores the body and the closing bracket:

```

3425 \def\eq@scan@sqrang@cancel{\expandafter\ignorespaces\@gobble}

```

Angle Brackets. The following is another version of the mechanism which scans for an equation body enclosed by `\<...>`.

`\eq@scan@ang` Start scanning for `\>`:

```

3426 \def\eq@scan@ang#1{%
3427 (dev)\eq@dev@enter\eq@scan@ang
3428 \def\eq@scan@end{#1\>}%
3429 \eq@scan@reg@{\def\eq@scan@stack{b}%
3430 \let\eq@scan@ang@org\<%\>
3431 \let\<\eq@scan@ang@iterate%\>
3432 \<%\>
3433 }

```

Iterate until we find a balanced pairing of angle brackets:

```

3434 \def\eq@scan@ang@iterate#1\>{%
3435 \edef\eq@scan@stack{%
3436 \eq@scan@ang@count#1\<\>\expandafter\@gobble\eq@scan@stack}%
3437 \ifx\@empty\eq@scan@stack
3438 \let\<\eq@scan@ang@org%\>
3439 \eq@scan@addto{#1}%
3440 (dev)\eq@dev@leave\eq@scan@ang
3441 \expandafter\eq@scan@end
3442 \else
3443 \eq@scan@addto{#1\>}%
3444 \expandafter\<%\>
3445 \fi
3446 }

```

Push a ‘b’ for every encountered instance of ‘\<’:

```

3447 \long\def\eq@scan@ang@count#1\<#2{\%>
3448 \ifx\>#2\else b\expandafter\eq@scan@ang@count\fi
3449 }

```


O.2 Options Processing

`\eql@equations@testall` The macro sequence started by `\eql@equations@testall` scans for optional arguments to the equation environments and appends them to the argument list using `\eqnaddopt`. The argument scheme is roughly `{ !t~ !t* !t! !o !e{@} }`. All arguments are scanned such that any spaces stop the scanning and such that any alignment markers ‘&’ cannot interfere:

```

3450 \def\eql@equations@testall{\eql@equations@testtilde}
3451 \def\eql@equations@testtilde#1{%
3452   \eql@ifnextgobble@tight~%
3453   {\eqnaddopt{lines}\eql@equations@testopt{#1}}%
3454   {\eql@equations@testopt{#1}}}
3455 \def\eql@equations@testopt#1{%
3456   \eql@ifnextchar@tight[%]
3457   {\eql@equations@addopt{\eql@equations@testexcl{#1}}}%
3458   {\eql@equations@testexcl{#1}}}
3459 \def\eql@equations@addopt#1[#2]{\eqnaddopt{#2}#1}
3460 \def\eql@equations@testexcl#1{%
3461   \eql@ifnextgobble@tight!%
3462   {\eqnaddopt{donumber}\eql@equations@testat{#1}}}%
3463   {\eql@equations@teststar{#1}}}
3464 \def\eql@equations@teststar#1{%
3465   \eql@ifstar@tight%
3466   {\eqnaddopt{nonumber}\eql@equations@testat{#1}}}%
3467   {\eql@equations@testat{#1}}}
3468 \def\eql@equations@testat#1{%
3469   \eql@ifat@tight
3470   {\eql@equations@addlabel{#1}}}%
3471   {#1}}
3472 \def\eql@equations@addlabel#1#2{\eqnaddopt{label={#2}}#1}

```

`\eql@equations@processopt` The macro `\eql@equations@processopt` processes the options received by `\eqnaddopt`. First, clear several non-persistent registers (labels, tags, direct vertical spacing). Then process the arguments. Finally evaluate `\eql@indent@val` and `\eql@tagsepmin@val` and prevent main punctuation from being passed to nested environments:

```

3473 \def\eql@equations@processopt{%
3474   \let\eql@blocklabel\@undefined
3475   \let\eql@blocktag\@undefined
3476   \let\eql@skip@force@above\@undefined
3477   \let\eql@skip@force@below\@undefined
3478   \let\eql@skip@force@leave\@undefined
3479   \eql@abovespace@\z@skip
3480   \eql@belowspace@\z@skip
3481   \eql@displaybreak@prepen@\@MM
3482   \eql@nextopt@process{equations}%
3483   \let\eql@punct@block\eql@punct@main
3484   \let\eql@punct@main\relax
3485   \eql@indent@\glueexpr\eql@indent@val\relax
3486   \eql@tagsepmin@\glueexpr\eql@tagsepmin@val\relax
3487 }

```

O.3 Equations Environment

We now declare the main environment and its symbolic versions.

Environment.

`equations` (*env.*) Declare the main equations environment. If already in math mode, fail and cancel the environment body. Otherwise scan for optional arguments and pass on to `\eql@equations@start`:

```
3488 \newenvironment{equations}{%
3489 <dev>\eql@dev@enterenv
3490 \ifmmode
3491 \eql@error@mathmode{\string\begin{\@currenvir}}%
3492 \expandafter\eql@scan@env\expandafter\eql@scan@env@cancel
3493 \else
3494 \expandafter\eql@ampprotect\expandafter\eql@equations@testall
3495 \expandafter\eql@equations@start
3496 \fi
3497 }{%
3498 \eql@equations@end
3499 \ignorespacesafterend
3500 <dev>\eql@dev@leaveenv
3501 }
```

`\eql@equations@start` The macro `\eql@equations@start` first processes the arguments. Depending on the chosen mode of operation, scan the environment body passing on to `\eql@equations@main` or process a single-line equation via `\eql@single@start`:

```
3502 \def\eql@equations@start{%
3503 \eql@equations@processopt
3504 \ifdefined\eql@equations@main
3505 \expandafter\eql@scan@env\expandafter\eql@equations@main
3506 \else
3507 \expandafter\eql@single@start
3508 \fi
3509 }
```

Square Brackets.

`equations@sqr` (*env.*) Define a pseudo-environment `equations@sqr` such that `\@currenvir` may point to it when needed:

```
3510 \newenvironment{equations@sqr}{}{}
```

`\eql@equations@sqr@open` The macro `\eql@equations@sqr@open` holds the definition for `\[`. If already in math mode, ignore the enclosed contents. Otherwise add the default arguments `\eql@equations@sqr@opt`, enter the pseudo-environment, scan for optional arguments, and pass on to `\eql@equations@sqr@start`:

```
3511 \protected\def\eql@equations@sqr@open{%
3512 \ifmmode
3513 \eql@error@mathmode{\string\[...\string\]}%
3514 \expandafter\eql@scan@sqr\expandafter\eql@scan@sqrang@cancel
3515 \else
3516 <dev>\eql@dev@enter{\string\[...\string\]}%
3517 \expandafter\eqnaddopt\expandafter{\eql@equations@sqr@opt}%
3518 \begin{equations@sqr}%
3519 \expandafter\eql@ampprotect\expandafter\eql@equations@testall
3520 \expandafter\eql@equations@sqr@start
3521 \fi
3522 }
```

`@equations@sqr@start` Process arguments. Depending on mode of operation, scan and process enclosed contents via `\eql@equations@main` or pass on to `\eql@single@start`:

```
3523 \def\eql@equations@sqr@start{%
3524   \eql@equations@processopt
3525   \ifdefined\eql@equations@main
3526     \expandafter\eql@scan@sqr\expandafter\eql@equations@main
3527   \else
3528     \expandafter\eql@single@start
3529   \fi
3530 }
```

`@equations@sqr@close` The macro `\eql@equations@sqr@close` holds the definition for ‘\’.

```
3531 \protected\def\eql@equations@sqr@close{%
3532   \eql@equations@end
3533 (dev)\eql@dev@leave{\[...\string\]}%
3534   \end{equations@sqr}%
3535   \ignorespaces
3536 }
```

Angle Brackets.

`equations@ang` (*env.*) Define a pseudo-environment `equations@ang`:

```
3537 \newenvironment{equations@ang}{}{}
```

`@equations@ang@open` The macro `\eql@equations@ang@open` holds the definition for ‘\<’.

```
3538 \protected\def\eql@equations@ang@open{%
3539   \ifmmode
3540     \eql@error@mathmode{\string\<...\string\>}%
3541     \expandafter\eql@scan@ang\expandafter\eql@scan@sqrang@cancel
3542   \else
3543 (dev)\eql@dev@enter{\<...\string\>}%
3544   \expandafter\eqnaddopt\expandafter{\eql@equations@ang@opt}%
3545   \begin{equations@ang}%
3546   \expandafter\eql@ampprotect\expandafter\eql@equations@testall
3547   \expandafter\eql@equations@ang@start
3548   \fi
3549 }
```

`@equations@ang@start` Process arguments and start handling the equation:

```
3550 \def\eql@equations@ang@start{%
3551   \eql@equations@processopt
3552   \ifdefined\eql@equations@main
3553     \expandafter\eql@scan@ang\expandafter\eql@equations@main
3554   \else
3555     \expandafter\eql@single@start
3556   \fi
3557 }
```

`@equations@ang@close` The macro `\eql@equations@ang@close` holds the definition for ‘\>’.

```
3558 \protected\def\eql@equations@ang@close{%
3559   \eql@equations@end
3560 (dev)\eql@dev@leave{\<...\string\>}%
3561   \end{equations@ang}%

```

```
3562 \ignorespaces
3563 }
```

O.4 Independent Routines of amsmath

`\eqref` amsmath defines the macro `\eqref` to refer to equation labels in a proper format. We provide it for completeness:

```
3564 \DeclareRobustCommand{\eq@eqref}[1]{\textup{\eq@tag@boxedform{\ref{#1}}}}
```

`\numberwithin` amsmath defines the macro `\numberwithin` to specify that equations should be numbered within sections. The L^AT_EX kernel contains a similar command `\counterwithin` (with a slightly extended syntax) that can be used as a drop-in replacement for `\numberwithin`:

```
3565 \DeclareRobustCommand{\eq@numberwithin}[3][\arabic]{%
3566   \@ifundefined{c@#2}{\@nocounterr{#2}}{%
3567     \@ifundefined{c@#3}{\@nocounterr{#3}}{%
3568       \ifdefined\HyOrg@addtoreset
3569         \HyOrg@addtoreset{#2}{#3}%
3570       \else
3571         \@addtoreset{#2}{#3}%
3572       \fi
3573       \expandafter\xdef\csname the#2\endcsname{%
3574         \expandafter\noexpand\csname the#3\endcsname.\noexpand#1{#2}}%
3575       \ifdefined\theHequation
3576         \expandafter\xdef\csname theH#2\endcsname{%
3577           \expandafter\noexpand
3578             \csname the\@ifundefined{theH#3}{H#3}\endcsname.%
3579           \noexpand\arabic{#2}}%
3580       \fi
3581     }%
3582   }%
3583 }
```

`\allowdisplaybreaks` amsmath defines the macro `\allowdisplaybreaks` which we also provide for completeness. The package uses the general setup mechanism instead:

```
3584 \DeclareRobustCommand{\eq@allowdisplaybreaks}[1][4]{%
3585   \eqnlineset{allowbreaks=#1}%
3586 }
```

`\notag` amsmath defines the alias `\notag` for `\nonumber` which we also provide for completeness:

```
3587 \newcommand{\eq@notag}{\nonumber}
```

`\thetag` amsmath defines `\thetag` to print a tag which we also provide for completeness:

```
3588 \newcommand{\eq@thetag}{\leavevmode\eq@tag@boxedform}
```

P Options

The package uses the `keyval` mechanism to parse key-value pairs to specify adjustments to the behaviour of the equations environments:

```
3589 \RequirePackage{keyval}
```

P.1 Selection Tools

`\eql@decide@select` Some parameter values take values in a given set, e.g. `true` vs. `false` or `left` vs. `right`. The macro `\eql@decide@select` is a general purpose selector. Arguments #1 and #2 describe the category and key which are used only towards error messages. Argument #3 contains the value and argument #4 is a list of values and corresponding actions in the format

$$\{ \{ \{ val1a, val1b, \dots \} \{ act1 \}, \{ \{ val2a, val2b, \dots \} \{ act2 \}, \dots \}.$$

If no corresponding value is found in the list, an error message is invoked. Single expansion is applied to the list of values:

```

3590 \def\eql@decide@select#1#2#3#4{%
3591   \def\@tempa{#3}%
3592   \let\@tempd\@undefined
3593   \@for\@tempc:=#4\do{%
3594     \ifdefined\@tempd\else
3595       \edef\@tempb{\noexpand\@tempb:=\expandafter\@firstoftwo\@tempc}%
3596       \expandafter\@for\@tempb\do{%
3597         \ifx\@tempa\@tempb
3598           \expandafter\expandafter\expandafter\def
3599             \expandafter\expandafter\expandafter\@tempd
3600               \expandafter\expandafter\expandafter{%
3601                 \expandafter\@secondoftwo\@tempc}%
3602         \fi
3603       }%
3604     \fi
3605   }%
3606   \ifdefined\@tempd
3607     \@tempd
3608   \else
3609     \eql@error{undefined value ‘#3’ for option ‘#2’ of ‘#1’}%
3610   \fi
3611 }
```

`\eql@decide@if` We will often have to decide between `true` and `false` or related pairs of values:

```

3612 \def\eql@decide@if#1#2#3#4#5{%
3613   \eql@decide@select{#1}{#2}{#3}{%
3614     {on,true,yes,enabled,1}{#4}},%
3615     {off,false,no,disabled,0}{#5}}}
```

`\eql@decide@bool` Boolean values frequently need to be stored into conditional registers:

```

3616 \def\eql@decide@bool#1#2#3#4{%
3617   \eql@decide@if{#1}{#2}{#3}{\let#4\eql@true}{\let#4\eql@false}}
```

`\eql@decide@abovebelow` **TODO:** describe

```

3618 \def\eql@decide@abovebelow#1#2#3#4#5{%
3619   \eql@decide@select{#1}{#2}{#3}{%
3620     {,abovebelow,both,tb}{#4#5},%
3621     {above,top,t}{#4},%
3622     {below,bottom,b}{#5}}%
3623 }
```

P.2 Declaration Code

`\eql@define@key` For convenience, we define a wrapper for `keyval`'s `\define@key` which accepts lists of categories and keys. We prepend the prefix `eql@` to all our categories so that it is hidden from the user in error messages:

```

3624 \def\eql@define@key#1#2{%
3625   \eql@ifnextchar@loose[%]
3626     {\eql@definekey@opt{#1}{#2}}%
3627     {\eql@definekey@noopt{#1}{#2}}%
3628 }
3629 \def\eql@definekey@noopt#1#2#3{\eql@definekey@for{#1}{#2}{#3}}
3630 \def\eql@definekey@opt#1#2[#3]#4{\eql@definekey@for{#1}{#2}{#3}{#4}}
3631 \def\eql@definekey@for#1#2#3{%
3632   \def\eql@for@fn##1##2##3{\define@key{eql@##3}{##2}##3}%
3633   \edef\eql@for@vara{\noexpand\eql@for@vara:=#1}%
3634   \expandafter\@for\eql@for@vara\do{%
3635     \edef\eql@for@varb{\noexpand\eql@for@varb:=#2}%
3636     \expandafter\@for\eql@for@varb\do{%
3637       \edef\eql@for@call##1{%
3638         \noexpand\eql@for@fn{##1}{\eql@for@varb}{\eql@for@vara}}%
3639       \eql@for@call{##1}%
3640     }%
3641   }%
3642 }

```

`\eql@setkeys` Our wrapper of `keyval`'s `\setkeys` prepends the prefix `eql@` to the category, and it expands the list argument once:

```

3643 \def\eql@setkeys#1#2{%
3644   \def\eql@tmp{\setkeys{eql@#1}}%
3645   \expandafter\eql@tmp\expandafter{#2}%
3646 }

```

`\eql@nextopt` It can be convenient to add arguments to the following equations environment, e.g. `\eql@nextopt@process` towards defining modifier macros:

`\eqnaddopt`

```

3647 \let\eql@nextopt\@empty
3648 \def\eql@nextopt@process#1{%
3649 (dev)\eql@dev@start\eql@nextopt@process
3650   \eql@setkeys{#1}\eql@nextopt
3651   \let\eql@tagging@opt\eql@nextopt
3652   \global\let\eql@nextopt\@empty
3653 }
3654 \newcommand{\eqnaddopt}[1]{%
3655   \expandafter\def\expandafter\eql@nextopt\expandafter{\eql@nextopt,#1}}

```

P.3 Parameter Sets

TODO: `par@above` plus `parskip`?

```

3656 \def\eql@defaults@classic{%
3657   \def\eql@tagsepmin@val{.5\fontdimen6\textfont\tw@}%
3658   \def\eql@colsepmax@val{.5\maxdimen}%
3659   \def\eql@spread{\jot}%
3660   \let\eql@tagmargin@val\@undefined
3661   \eql@tagmargin@ratio@p@
3662   \def\eql@tagmargin@threshold{0.5}%

```

```

3663 \def\eql@flushleftmargin@val{\leftmargini}%
3664 \let\eql@display@height\undefined
3665 \let\eql@display@depth\undefined
3666 \eql@skip@mode@short@\tw@
3667 \eql@skip@mode@cont@\@ne
3668 \eql@skip@mode@par@\z@
3669 \eql@skip@mode@top@\z@
3670 \def\eql@skip@long@above{\abovedisplayskip}%
3671 \def\eql@skip@long@below{\belowdisplayskip}%
3672 \def\eql@skip@short@above{\abovedisplaysshortskip}%
3673 \def\eql@skip@short@below{\belowdisplaysshortskip}%
3674 \def\eql@skip@cont@above{\eql@skip@short@above}%
3675 \def\eql@skip@cont@below{\eql@skip@short@below}%
3676 \def\eql@skip@par@above{\eql@skip@long@above}%
3677 \def\eql@skip@par@below{\eql@skip@long@below}%
3678 \def\eql@skip@top@above{\eql@skip@long@above}%
3679 \def\eql@skip@top@below{\eql@skip@long@below}%
3680 \def\eql@skip@med@above{\abovedisplayskip/2}%
3681 \def\eql@skip@med@below{\belowdisplayskip/2}%
3682 \def\eql@skip@medtag@above{\z@skip}%
3683 \def\eql@skip@medtag@below{\z@skip}%
3684 \def\eql@skip@tag@above{\z@skip}%
3685 \def\eql@skip@tag@below{\z@skip}%
3686 }

```

values based on 10pt vs 12pt

```

3687 \def\eql@defaults@eqnlines{%
3688 \def\eql@tagsepmin@val{.5em}%
3689 \def\eql@colsepmax@val{2em}%
3690 \def\eql@spread{0.2\normalbaselineskip}%
3691 \let\eql@tagmargin@val\undefined
3692 \eql@tagmargin@ratio@.334\p@
3693 \def\eql@tagmargin@threshold{0.5}%
3694 \def\eql@flushleftmargin@val{\leftmargini}%
3695 \def\eql@display@height{\ht\eql@strutbox}%
3696 \def\eql@display@depth{\dp\eql@strutbox}%
3697 \eql@skip@mode@short@\@ne
3698 \eql@skip@mode@cont@\z@
3699 \eql@skip@mode@par@\z@
3700 \eql@skip@mode@top@\z@
3701 \def\eql@skip@long@above{0.75\normalbaselineskip
3702 \@plus0.25\normalbaselineskip\@minus0.4\normalbaselineskip}%
3703 \let\eql@skip@long@below\eql@skip@long@above
3704 \def\eql@skip@short@above{0.0\normalbaselineskip
3705 \@plus0.25\normalbaselineskip}%
3706 \def\eql@skip@short@below{0.0\normalbaselineskip
3707 \@plus0.25\normalbaselineskip}%
3708 \def\eql@skip@med@above{0.4\normalbaselineskip
3709 \@plus0.2\normalbaselineskip\@minus0.2\normalbaselineskip}%
3710 \let\eql@skip@med@below\eql@skip@med@above
3711 \def\eql@skip@cont@above{\z@skip}%
3712 \def\eql@skip@cont@below{\eql@skip@long@below}%
3713 \def\eql@skip@par@above{\eql@skip@long@above}%
3714 \def\eql@skip@par@below{\eql@skip@long@below}%
3715 \def\eql@skip@top@above{\z@skip}%
3716 \def\eql@skip@top@below{\z@skip}%
3717 \def\eql@skip@tag@above{\z@skip}%
3718 \def\eql@skip@tag@below{\z@skip}%

```

```

3719 \def\eql@skip@partag@above{\z@skip}%
3720 \def\eql@skip@partag@below{\z@skip}%
3721 \def\eql@skip@medtag@above{\z@skip}%
3722 \def\eql@skip@medtag@below{\z@skip}%
3723 }
3724 \eql@defaults@eqnlines

```

P.4 Options Declarations

TODO: describe

Modes for Equations Box Environment. **TODO:** describe

```

3725 \eql@define@key{equationsbox}{gathered,gather,ga,lines,lined,ln,\string~}[]{}%
3726 \eql@mode@lined}
3727 \eql@define@key{equationsbox}{aligned,align,al,columns,col,@}[]{}%
3728 \eql@mode@aligned}
3729 \eql@define@key{equationsbox}{top,t}[]{\let\eql@box@box\vtop}
3730 \eql@define@key{equationsbox}{center,c}[]{\let\eql@box@box\vcenter}
3731 \eql@define@key{equationsbox}{bottom,b}[]{\let\eql@box@box\ vbox}
3732 \eql@define@key{equationsbox}{colsep}{\def\eql@box@colsep{#1}}

```

Modes for Equations Environment. **TODO:** describe

```

3733 \eql@define@key{equations}{equation,eq,single,1}[]{\eql@mode@equation}
3734 \eql@define@key{equations}{gathered,gather,ga,lines,lined,ln,\string~}[]{}%
3735 \eql@mode@lines}
3736 \eql@define@key{equations}{aligned,align,al,columns,col,@}[]{}%
3737 \eql@mode@align}
3738 \eql@define@key{equations}{native}[true]{}%
3739 \eql@decide@bool{#3}{#2}{#1}\eql@single@native%
3740 \ifdefined\eql@single@native\let\eql@flushleft\eql@false\fi}
3741 \eql@define@key{setup}{native}[true]{}%
3742 \eql@decide@bool{#3}{#2}{#1}\eql@single@native}
3743 \eql@define@key{setup}{scanequation}[true]{}%
3744 \eql@decide@bool{#3}{#2}{#1}\eql@single@doscan}
3745 \eql@define@key{setup}{sqropt}[]{}%
3746 \def\eql@equations@sqr@opt{equation,#1}}
3747 \eql@define@key{setup}{angopt}[]{}%
3748 \def\eql@equations@ang@opt{align,#1}}

```

Vertical Spacing. **TODO:** set at end of env only! **TODO:** describe

```

3749 \def\eql@keycat{equations,equationsbox,setup}
3750 \eql@define@key\eql@keycat{spread}[\jot]{\def\eql@spread{#1}}
3751 \eql@define@key\eql@keycat{strut}[true]{\eql@decide@if{#3}{#2}{#1}%
3752 {\let\eql@strut@field\eql@strut}{\let\eql@strut@field\relax}}
3753 \eql@define@key\eql@keycat{strutttag}[true]{\eql@decide@if{#3}{#2}{#1}%
3754 {\let\eql@strut@tag\eql@strut}{\let\eql@strut@tag\relax}}

```

TODO: describe **TODO:** maybe also add pre and post variants? for general setup?

```

3755 \eql@define@key{equations}{displaybreak}[4]{\eql@displaybreak@pre{#1}}
3756 \def\eql@keycat{equations,setup}
3757 \eql@define@key\eql@keycat{allowbreaks,allowdisplaybreaks}[4]{}%
3758 \interdisplaylinepenalty\eql@getdsp@pen{#1}\relax}
3759 \eql@define@key\eql@keycat{displayheight}[\ht\eql@strutbox@]{}%

```



```

3760 \def\eqldisplay@height{#1}}
3761 \eqld@define@key\eql@keycat{displaydepth}[\dp\eql@strutbox@]{%
3762 \def\eqldisplay@depth{#1}}
3763 \eqld@define@key\eql@keycat{displayheight*}[]{%
3764 \let\eqldisplay@height\undefined}
3765 \eqld@define@key\eql@keycat{displaydepth*}[]{%
3766 \let\eqldisplay@depth\undefined}

```

TODO: describe **TODO:** short should just apply to above?! or as far as short would apply...

```

3767 \eqld@define@key{equations}{noskip}[]{%
3768 \eqld@decide@abovebelow{#3}{#2}{#1}%
3769 {\def\eql@skip@force@above{5}}%
3770 {\def\eql@skip@force@below{5}}%
3771 \eqld@define@key{equations}{short}[above]{%
3772 \eqld@decide@abovebelow{#3}{#2}{#1}%
3773 {\def\eql@skip@force@above{1}}%
3774 {\def\eql@skip@force@below{1}}%
3775 \eqld@define@key{equations}{long}[]{%
3776 \eqld@decide@abovebelow{#3}{#2}{#1}%
3777 {\def\eql@skip@force@above{0}}%
3778 {\def\eql@skip@force@below{0}}%
3779 \eqld@define@key{equations}{medskip}[]{%
3780 \eqld@decide@abovebelow{#3}{#2}{#1}%
3781 {\def\eql@skip@force@above{6}}%
3782 {\def\eql@skip@force@below{6}}%
3783 \eqld@define@key{equations}{par}[par]{%
3784 \eqld@decide@select{#3}{#2}{#1}{%
3785 {{default},}\let\eql@skip@force@leave\undefined},%
3786 {{cont,hmode}{\let\eql@skip@force@leave\z}},%
3787 {{par,vmode}{\let\eql@skip@force@leave\one
3788 \ifdefined\eql@skip@force@below\else
3789 \def\eql@skip@force@below{3}%
3790 \fi}},%
3791 {{top}{\let\eql@skip@force@leave\tw@
3792 \ifdefined\eql@skip@force@below\else
3793 \def\eql@skip@force@below{4}
3794 \fi}}}}

```

TODO: describe

```

3795 \eqld@define@key{equations}{skip}{%
3796 \def\eql@skip@force@above{7}%
3797 \def\eql@skip@custom@above{#1}%
3798 \let\eql@skip@force@below\eql@skip@force@above
3799 \let\eql@skip@custom@below\eql@skip@custom@above}
3800 \eqld@define@key{equations}{aboveskip}{%
3801 \def\eql@skip@force@above{7}%
3802 \def\eql@skip@custom@above{#1}}
3803 \eqld@define@key{equations}{belowskip}{%
3804 \def\eql@skip@force@below{7}%
3805 \def\eql@skip@custom@below{#1}}
3806 \eqld@define@key{equations}{abovespace}{%
3807 \advance\eql@abovespace@glueexpr#1\relax}
3808 \eqld@define@key{equations}{belowspace}{%
3809 \advance\eql@belowspace@glueexpr#1\relax}

```

TODO: describe

```

3810 \eql@define@key{intertext}{skip}{%
3811   \def\eql@skip@force@above{7}%
3812   \def\eql@skip@custom@above{#1}%
3813   \let\eql@skip@force@below\eql@skip@force@above
3814   \let\eql@skip@custom@below\eql@skip@custom@above}
3815 \eql@define@key{intertext}{aboveskip}{%
3816   \def\eql@skip@force@below{7}%
3817   \def\eql@skip@custom@below{#1}}
3818 \eql@define@key{intertext}{belowskip}{%
3819   \def\eql@skip@force@above{7}%
3820   \def\eql@skip@custom@above{#1}}
3821 \eql@define@key{intertext}{noskip}[]{%
3822   \eql@decide@abovebelow{#3}{#2}{#1}%
3823   {\def\eql@skip@force@below{5}}%
3824   {\def\eql@skip@force@above{5}}}
3825 \eql@define@key{intertext}{short}[]{%
3826   \eql@decide@abovebelow{#3}{#2}{#1}%
3827   {\def\eql@skip@force@below{1}}%
3828   {\def\eql@skip@force@above{1}}}
3829 \eql@define@key{intertext}{long}[]{%
3830   \eql@decide@abovebelow{#3}{#2}{#1}%
3831   {\def\eql@skip@force@below{0}}%
3832   {\def\eql@skip@force@above{0}}}
3833 \eql@define@key{intertext}{medskip}[]{%
3834   \eql@decide@abovebelow{#3}{#2}{#1}%
3835   {\def\eql@skip@force@below{6}}%
3836   {\def\eql@skip@force@above{6}}}

```

TODO: describe

```

3837 \eql@define@key{setup}{skip, longskip}{%
3838   \abovedisplayskip\glueexpr#1\relax
3839   \belowdisplayskip\abovedisplayskip
3840   \def\eql@skip@long@above{#1}%
3841   \let\eql@skip@long@below\eql@skip@long@above}
3842 \eql@define@key{setup}{aboveskip, abovelongskip}{%
3843   \abovedisplayskip\glueexpr#1\relax
3844   \def\eql@skip@long@above{#1}}
3845 \eql@define@key{setup}{belowskip, belowlongskip}{%
3846   \belowdisplayskip\glueexpr#1\relax
3847   \def\eql@skip@long@below{#1}}
3848 \eql@define@key{setup}{aboveshortskip}{%
3849   \abovedisplayshortskip\glueexpr#1\relax
3850   \def\eql@skip@short@above{#1}}
3851 \eql@define@key{setup}{belowshortskip}{%
3852   \belowdisplayshortskip\glueexpr#1\relax
3853   \def\eql@skip@short@below{#1}}
3854 \eql@define@key{setup}{tagskip}{%
3855   \def\eql@skip@tag@above{#1}%
3856   \let\eql@skip@tag@below\eql@skip@tag@above}
3857 \eql@define@key{setup}{abovetagskip}{%
3858   \def\eql@skip@tag@above{#1}}
3859 \eql@define@key{setup}{belowtagskip}{%
3860   \def\eql@skip@tag@below{#1}}
3861 \eql@define@key{setup}{medskip}{%
3862   \def\eql@skip@med@above{#1}%
3863   \let\eql@skip@med@below\eql@skip@med@above}
3864 \eql@define@key{setup}{abovemedskip}{%
3865   \def\eql@skip@med@above{#1}}

```

```

3866 \eql@define@key{setup}{belowmedskip}{%
3867   \def\eql@skip@med@below{#1}}
3868 \eql@define@key{setup}{medtagskip}{%
3869   \def\eql@skip@medtag@above{#1}%
3870   \let\eql@skip@medtag@below\eql@skip@medtag@above}
3871 \eql@define@key{setup}{abovemedtagskip}{%
3872   \def\eql@skip@medtag@above{#1}}
3873 \eql@define@key{setup}{belowmedtagskip}{%
3874   \def\eql@skip@medtag@below{#1}}
3875 \eql@define@key{setup}{abovetopskip}{%
3876   \def\eql@skip@top@above{#1}}
3877 \eql@define@key{setup}{belowtopskip}{%
3878   \def\eql@skip@top@below{#1}}
3879 \eql@define@key{setup}{aboveparskip}{%
3880   \def\eql@skip@par@above{#1}}
3881 \eql@define@key{setup}{belowparskip}{%
3882   \def\eql@skip@par@below{#1}}
3883 \eql@define@key{setup}{abovepartagskip}{%
3884   \def\eql@skip@partag@above{#1}}
3885 \eql@define@key{setup}{belowpartagskip}{%
3886   \def\eql@skip@partag@below{#1}}
3887 \eql@define@key{setup}{abovecontskip}{%
3888   \def\eql@skip@cont@above{#1}}
3889 \eql@define@key{setup}{abovecontskip*}[]{%
3890   \def\eql@skip@cont@above{\eql@spread-\eql@skip@long@below}}
3891 \eql@define@key{setup}{belowcontskip}{%
3892   \def\eql@skip@cont@below{#1}}
3893 \eql@define@key{setup}{shortmode}{%
3894   \eql@decide@select{#3}{#2}{#1}{%
3895     {{off,never,no}{\eql@skip@mode@short@z}},%
3896     {{above,neverbelow,notbelow,belowoff}{\eql@skip@mode@short@one}},%
3897     {{belowone,belowsingle}{\eql@skip@mode@short@tw}},%
3898     {{belowall,always,on}{\eql@skip@mode@short@thre}}}}
3899 \eql@define@key{setup}{belowcontmode}{%
3900   \eql@decide@select{#3}{#2}{#1}{%
3901     {{long}{\eql@skip@mode@cont@z}},%
3902     {{short}{\eql@skip@mode@cont@one}},%
3903     {{cont}{\eql@skip@mode@cont@tw}}}}
3904 \eql@define@key{setup}{belowparmode}{%
3905   \eql@decide@select{#3}{#2}{#1}{%
3906     {{long}{\eql@skip@mode@par@z}},%
3907     {{short}{\eql@skip@mode@par@one}},%
3908     {{cont}{\eql@skip@mode@par@tw}},%
3909     {{par}{\eql@skip@mode@par@thre}}}}
3910 \eql@define@key{setup}{belowtopmode}{%
3911   \eql@decide@select{#3}{#2}{#1}{%
3912     {{long}{\eql@skip@mode@top@z}},%
3913     {{short}{\eql@skip@mode@top@one}},%
3914     {{cont}{\eql@skip@mode@top@tw}},%
3915     {{par}{\eql@skip@mode@top@thre}},%
3916     {{top}{\eql@skip@mode@top@4relax}}}}

```

Labels and Tag Declaration. **TODO:** describe

```

3917 \def\eql@keycat{equations,subequations}
3918 \eql@define@key\eql@keycat{label}{\eql@blocklabel@set{#1}}
3919 \eql@define@key\eql@keycat{tag}{\eql@blocktag@set{#1}}
3920 \eql@define@key\eql@keycat{tag*}{\eql@blocktag@setstar{#1}}

```

Tag Spacing. **TODO:** describe

```
3921 \def\eq@keycat{equations,setup}
3922 \eq@define@key\eq@keycat{tagmargin}{\def\eq@tagmargin@val{#1}}
3923 \eq@define@key\eq@keycat{tagmargin*}{%
3924   \settowidth\dimen@{#1}\edef\eq@tagmargin@val{\the\dimen@}}
3925 \eq@define@key\eq@keycat{tagmargin@val}[\let\eq@tagmargin@val\undefined}
3926 \eq@define@key\eq@keycat{tagmargin@val}[\let\eq@tagmargin@val\undefined}
3927 \eq@define@key\eq@keycat{tagmargin@val}[\let\eq@tagmargin@val\undefined}
3928 \eq@define@key\eq@keycat{tagmargin@val}[\let\eq@tagmargin@val\undefined}
3929 \def\eq@tagmargin@threshold{#1}}
3930 \eq@define@key\eq@keycat{tagmargin@val}[\let\eq@tagmargin@val\undefined}
3931 \eq@define@key\eq@keycat{tagmargin@val}[\let\eq@tagmargin@val\undefined}
3932 \settowidth\dimen@{#1}\edef\eq@tagmargin@val{\the\dimen@}}
3933 \eq@define@key\eq@keycat{tagmargin@val}[\let\eq@tagmargin@val\undefined}
```

Tag Layout. **TODO:** describe

```
3934 \eq@define@key{setup}{taglayout}{\eq@tag@setbox{#1}}
3935 \eq@define@key{setup}{taglayout*}{\eq@tag@setbox@{#1}}
3936 \eq@define@key{setup}{tagform}{\eq@tag@setform{#1}}
3937 \eq@define@key{setup}{tagform*}{\eq@tag@setform@{#1}}
3938 \eq@define@key{setup}{subeqtemplate}{\def\eq@subequations@template{#1}}
3939 \eq@define@key{setup}{autolabel}[true]{%
3940   \eq@decide@bool{#3}{#2}{#1}\eq@autolabel}
3941 \eq@define@key{setup}{autotag}[true]{%
3942   \eq@decide@bool{#3}{#2}{#1}\eq@autotag}
```

Equation Numbering. **TODO:** describe

```
3943 \def\eq@keycat{equations,setup}
3944 \eq@define@key\eq@keycat{numberline,numline,n}[all]{%
3945   \eq@numbering@set{##1}}
3946 \eq@define@key\eq@keycat{nonumber,nn,*}[\let\eq@numbering@active\eq@false}
3947 \eq@define@key\eq@keycat{donumber,dn,!}[\let\eq@numbering@active\eq@true}
3948 \eq@define@key\eq@keycat{number,num}[true]{%
3949   \eq@decide@bool{#3}{#2}{#1}\eq@numbering@active}
3950 \eq@define@key\eq@keycat{tagsleft,leqno}[\let\eq@tagsleft\eq@true}
3951 \eq@define@key\eq@keycat{tagsright,reqno}[\let\eq@tagsleft\eq@false}
3952 \eq@define@key\eq@keycat{tags,eqno}{%
3953   \eq@decide@select{#3}{#2}{#1}{%
3954     {\right,r}{\let\eq@tagsleft\eq@false}},%
3955     {\left,l}{\let\eq@tagsleft\eq@true}}}
```

Horizontal Layout. **TODO:** describe

```
3956 \def\eq@keycat{equations,setup}
3957 \eq@define@key\eq@keycat{layout}{\eq@decide@select{#3}{#2}{#1}{%
3958   {\center,c}{\let\eq@flushleft\eq@false}},%
3959   {\left,l}{\let\eq@flushleft\eq@true}}}}
3960 \eq@define@key\eq@keycat{center}[\let\eq@flushleft\eq@false}
3961 \eq@define@key\eq@keycat{flushleft,left}[\let\eq@flushleft\eq@true}
3962 \eq@define@key\eq@keycat{leftmargin}{\def\eq@flushleftmargin@val{#1}}
3963 \eq@define@key\eq@keycat{leftmargin*}{%
3964   \settowidth\dimen@{#1}\edef\eq@flushleftmargin@val{\the\dimen@}}
3965 \eq@define@key\eq@keycat{minleftmargin}{%
3966   \eq@flushleftmarginmin@glueexpr#1\relax}
3967 \eq@define@key\eq@keycat{maxleftmargin}{%
```

```

3968 \eq@flushleftmarginmax@glueexpr#1\relax}
3969 \eq@define@key\eq@keycat{maxleftmargin*}[]{%
3970 \eq@flushleftmarginmax@.5\maxdimen}

```

Horizontal Spacing and Columns. **TODO:** describe

```

3971 \def\eq@keycat{equations,setup}
3972 \eq@define@key\eq@keycat{marginbadness}{\eq@marginbadness@#1\relax}
3973 \eq@define@key\eq@keycat{maxbadness}{\eq@maxbadness@#1\relax}
3974 \eq@define@key\eq@keycat{mincolsep}{\def\eq@colsepmin@val{#1}}
3975 \eq@define@key\eq@keycat{maxcolsep}{\def\eq@colsepmax@val{#1}}
3976 \eq@define@key\eq@keycat{maxcolsep*}[]{\def\eq@colsepmax@val{.5\maxdimen}}
3977 \eq@define@key\eq@keycat{margins}[true]{%
3978 \eq@decide@bool{#3}{#2}{#1}\eq@align@margins}
3979 \def\eq@keycat{equationsbox,setup}
3980 \eq@define@key\eq@keycat{margin}{%
3981 \def\eq@box@marginleft{#1}\def\eq@box@marginright{#1}}
3982 \eq@define@key\eq@keycat{marginleft}{\def\eq@box@marginleft{#1}}
3983 \eq@define@key\eq@keycat{marginright}{\def\eq@box@marginright{#1}}

```

Horizontal Shape. **TODO:** describe

```

3984 \def\eq@keycat{equations,equationsbox,setup}
3985 \eq@define@key\eq@keycat{shape}[default]{\eq@shape@set{#1}}
3986 \eq@define@key\eq@keycat{padding,pad}[\eq@indent@val]{%
3987 \let\eq@paddingmax\eq@false
3988 \def\eq@paddingleft@val{#1}\def\eq@paddingright@val{#1}}
3989 \eq@define@key\eq@keycat{padleft}[\eq@indent@val]{%
3990 \let\eq@paddingmax\eq@false\def\eq@paddingleft@val{#1}}
3991 \eq@define@key\eq@keycat{padright}[\eq@indent@val]{%
3992 \let\eq@paddingmax\eq@false\def\eq@paddingright@val{#1}}
3993 \eq@define@key\eq@keycat{padmax}[true]{%
3994 \eq@decide@bool{#3}{#2}{#1}\eq@paddingmax}
3995 \eq@define@key\eq@keycat{indent}[2em]{%
3996 \def\eq@indent@val{#1}}
3997 \eq@define@key\eq@keycat{indent*}[2em]{%
3998 \def\eq@indent@val{#1}\def\eq@paddingleft@val{#1}}

```

Math Classes at Alignment. **TODO:** describe

```

3999 \def\eq@keycat{equations,equationsbox,setup}
4000 \eq@define@key\eq@keycat{classout}{\eq@class@innerleft@set{#1}}
4001 \eq@define@key\eq@keycat{classin}{\eq@class@innerright@set{#1}}
4002 \eq@define@key\eq@keycat{classin*}{\eq@class@innerlead@set{#1}}
4003 \eq@define@key\eq@keycat{ampeq}[]{\eq@class@ampeq}
4004 \eq@define@key\eq@keycat{eqamp}[]{\eq@class@eqamp}
4005 \eq@define@key\eq@keycat{class}{\eq@decide@select{#3}{#2}{#1}{%
4006 {{ampeq,amprel,eqafter,beforerel}\eq@class@ampeq},%
4007 {{eqamp,relamp,eqbefore,afterrel}\eq@class@eqamp}}}}

```

Punctuation. **TODO:** describe

```

4008 \let\eq@punct@main\relax
4009 \def\eq@keycat{equations,equationsbox,setup}
4010 \eq@define@key\eq@keycat{punctsep}[\,]{\def\eq@punct@sep{#1}}
4011 \eq@define@key\eq@keycat{punct}[.]{\def\eq@punct@main{#1}}
4012 \eq@define@key\eq@keycat{punctline}[,]{\def\eq@punct@line{#1}}

```

```

4013 \eql@define@key\eql@keycat{punctcol}[,]{\def\eql@punct@col{#1}}
4014 \eql@define@key\eql@keycat{punct*}[]{\let\eql@punct@main\relax}
4015 \eql@define@key\eql@keycat{punctline*}[]{\let\eql@punct@line\relax}
4016 \eql@define@key\eql@keycat{punctcol*}[]{\let\eql@punct@col\relax}

```

Global Switches. **TODO:** describe

```

4017 \let\eql@multi@linesfallback\eql@true
4018 \let\eql@single@crerror\eql@true
4019 \let\eql@ampproof@active\eql@false
4020 \eql@define@key{setup}{linesfallback}[true]{%
4021   \eql@decide@bool{#3}{#2}{#1}\eql@multi@linesfallback}
4022 \eql@define@key{setup}{ampproof}[true]{%
4023   \eql@decide@bool{#3}{#2}{#1}\eql@ampproof@active}
4024 \eql@define@key{setup}{crerror}[true]{%
4025   \eql@decide@bool{#3}{#2}{#1}\eql@single@crerror}
4026 \eql@define@key{equations,setup}{rescan}[true]{%
4027   \eql@decide@if{#3}{#2}{#1}%
4028   {\let\eql@scan@body\eql@scan@body@rescan}%
4029   {\let\eql@scan@body\eql@scan@body@dump}}
4030 \eql@define@key{setup}{defaults}{%
4031   \eql@decide@select{#3}{#2}{#1}{%
4032     {classic}{\eql@defaults@classic}},%
4033     {eqnlines}{\eql@defaults@eqnlines}}}}

```

Package Options. **TODO:** describe

```

4034 \let\eql@provide@opt@equation\eql@true
4035 \let\eql@provide@opt@amsmathends\eql@true
4036 \let\eql@provide@opt@amsmath\eql@true
4037 \let\eql@provide@opt@ang\eql@true
4038 \let\eql@provide@opt@eqref\eql@true
4039 \eql@define@key{setup}{equation}[true]{%
4040   \eql@error@packageoption{#2}%
4041   \eql@decide@bool{#3}{#2}{#1}\eql@provide@opt@equation}
4042 \eql@define@key{setup}{amsmathends}[true]{%
4043   \eql@error@packageoption{#2}%
4044   \eql@decide@bool{#3}{#2}{#1}\eql@provide@opt@amsmathends}
4045 \eql@define@key{setup}{amsmath}[true]{%
4046   \eql@error@packageoption{#2}%
4047   \eql@decide@bool{#3}{#2}{#1}\eql@provide@opt@amsmath}
4048 \eql@define@key{setup}{ang}[true]{%
4049   \eql@error@packageoption{#2}%
4050   \eql@decide@bool{#3}{#2}{#1}\eql@provide@opt@ang}
4051 \eql@define@key{setup}{eqref}[true]{%
4052   \eql@error@packageoption{#2}%
4053   \eql@decide@bool{#3}{#2}{#1}\eql@provide@opt@eqref}

```

P.5 Component Selection

TODO: describe

```

4054 \newenvironment{eql@gathered}
4055   {\eqnaddopt{lines}\equationsbox}{\endequationsbox}
4056 \newenvironment{eql@multlined}
4057   {\eqnaddopt{lines,padding,shape=steps}\equationsbox}{\endequationsbox}
4058 \newenvironment{eql@aligned}

```



```
4059 {\eqnaddopt{align}\equationsbox}{\endequationsbox}
```

TODO: describe

```
4060 \newenvironment{eql@equation}
4061   {\eqnaddopt{equation}\equations}{\endequations}
4062 \newenvironment{eql@gather}
4063   {\eqnaddopt{lines}\equations}{\endequations}
4064 \newenvironment{eql@multline}
4065   {\eqnaddopt{lines, padmax, shape=steps, numberline=out}\equations}
4066   {\endequations}
4067 \newenvironment{eql@align}
4068   {\eqnaddopt{align}\equations}{\endequations}
4069 \newenvironment{eql@flalign}
4070   {\eqnaddopt{align, margins=false}\equations}{\endequations}
4071 \newenvironment{eql@equation*}
4072   {\eqnaddopt{nonumber}\eql@equation}{\endequations}
4073 \newenvironment{eql@gather*}
4074   {\eqnaddopt{nonumber}\eql@gather}{\endequations}
4075 \newenvironment{eql@multline*}
4076   {\eqnaddopt{nonumber}\eql@multline}{\endequations}
4077 \newenvironment{eql@align*}
4078   {\eqnaddopt{nonumber}\eql@align}{\endequations}
4079 \newenvironment{eql@flalign*}
4080   {\eqnaddopt{nonumber}\eql@flalign}{\endequations}
```

TODO: describe

```
4081 \def\eql@provide@movecmd#1#2{%
4082   \expandafter\let\csname #1\expandafter\endcsname\csname #2\endcsname
4083 }
4084 \def\eql@provide@undefinecmd#1{%
4085   \expandafter\let\csname #1\endcsname\@undefined
4086 }
4087 \def\eql@provide@moveenv#1#2{%
4088   \expandafter\let\csname #1\expandafter\endcsname\csname #2\endcsname
4089   \expandafter\let\csname end#1\expandafter\endcsname\csname end#2\endcsname
4090 }
4091 \def\eql@provide@undefineenv#1{%
4092   \expandafter\let\csname #1\endcsname\@undefined
4093   \expandafter\let\csname end#1\endcsname\@undefined
4094 }
```

TODO: describe

```
4095 \def\eql@provide@onlyonce#1#2{%
4096   \def\eql@tmp{#2}%
4097   \def\@tempa{#1}%
4098   \ifx\eql@tmp\@tempa
4099     \let\eql@tmp\@undefined
4100   \fi
4101   \ifx\eql@tmp\@empty
4102     \let\eql@tmp\@undefined
4103   \fi
4104   \def\@tempa{*}%
4105   \ifx\eql@tmp\@tempa
4106     \def\eql@tmp{#1}%
4107   \fi
4108   \ifdefined\eql@tmp\else
4109     \ifcsname eql@provided@#1\endcsname
4110       \def\eql@tmp{#1}%

```

```

4111   \else
4112     \expandafter\let\csname eql@provided@#1\endcsname\eql@true
4113   \fi
4114 \fi
4115 }

```

TODO: describe

```

4116 \def\eql@provide@cmdonlyonce#1#2{%
4117   \def\eql@tmp{#2}%
4118   \edef\@tempb{\expandafter\noexpand\csname#1\endcsname}%
4119   \ifx\eql@tmp\@tempb
4120     \let\eql@tmp\undefined
4121   \fi
4122   \ifx\eql@tmp\@empty
4123     \let\eql@tmp\undefined
4124   \fi
4125   \def\@tempa{*}%
4126   \ifx\eql@tmp\@tempa
4127     \let\eql@tmp\@tempb
4128   \fi
4129   \ifdefined\eql@tmp
4130     \edef\eql@tmp{\expandafter\expandafter\expandafter\@gobble
4131       \expandafter\string\eql@tmp}%
4132   \else
4133     \ifcsname eql@provided@#1\endcsname
4134       \let\eql@tmp\@tempb
4135     \else
4136       \expandafter\let\csname eql@provided@#1\endcsname\eql@true
4137     \fi
4138   \fi
4139 }

```

TODO: describe

```

4140 \def\eql@provide@cmd#1#2{%
4141   \eql@provide@cmdonlyonce{#1}{#2}%
4142   \ifdefined\eql@tmp
4143     \expandafter\eql@provide@movecmd\expandafter{\eql@tmp}{eql@#1}%
4144   \else
4145     \eql@amsmath@after{%
4146       \eql@provide@movecmd{ams#1}{#1}%
4147       \eql@provide@movecmd{#1}{eql@#1}%
4148     }%
4149     \AddToHook{package/mathtools/after}{%
4150       \eql@provide@movecmd{#1}{eql@#1}%
4151     }%
4152     \eql@provide@movecmd{#1}{eql@#1}%
4153     \eql@amsmath@before{\eql@provide@undefinecmd{#1}}%
4154   \fi
4155 }

```

TODO: describe

```

4156 \def\eql@amsmath@endfix#1#2{%
4157   \long\edef\@tempa{\expandafter\noexpand\csname end#2\endcsname}%
4158   \long\edef\@tempb{\expandafter\noexpand\csname eql@amsmath@end#2\endcsname}%
4159   \expandafter\ifx\csname end#1\endcsname\@tempa
4160     \expandafter\let\csname end#1\endcsname\@tempb
4161   \fi
4162 }

```


TODO: describe

```
4163 \def\eql@amsmath@fixends{%
4164   \eql@amsmath@after{%
4165     \let\eql@amsmath@endmultline\endmultline
4166     \eql@amsmath@endfix{multline*}{multline}%
4167     \let\eql@amsmath@endgather\endgather
4168     \eql@amsmath@endfix{gather*}{gather}%
4169     \let\eql@amsmath@endalign\endalign
4170     \eql@amsmath@endfix{align*}{align}%
4171     \eql@amsmath@endfix{flalign}{align}%
4172     \eql@amsmath@endfix{flalign*}{align}%
4173     \eql@amsmath@endfix{alignat}{align}%
4174     \eql@amsmath@endfix{alignat*}{align}%
4175     \eql@amsmath@endfix{xalignat}{align}%
4176     \eql@amsmath@endfix{xalignat*}{align}%
4177     \eql@amsmath@endfix{xxalignat}{align}%
4178     \let\eql@amsmath@endaligned\endaligned
4179     \eql@amsmath@endfix{gathered}{aligned}%
4180     \eql@amsmath@endfix{alignedat}{aligned}%
4181   }
4182 }
```

TODO: describe

```
4183 \def\eql@provide@env#1#2{%
4184   \eql@provide@onlyonce{#1}{#2}%
4185   \ifdefined\eql@tmp
4186     \expandafter\eql@provide@moveenv\expandafter{\eql@tmp}{\eql@#1}%
4187   \else
4188     \eql@amsmath@after{%
4189       \eql@provide@moveenv{ams#1}{#1}%
4190       \eql@provide@moveenv{ams#1*}{#1*}%
4191       \eql@provide@moveenv{#1}{\eql@#1}%
4192       \eql@provide@moveenv{#1*}{\eql@#1*}%
4193     }%
4194     \AddToHook{package/mathtools/after}{%
4195       \eql@provide@moveenv{#1}{\eql@#1}%
4196       \eql@provide@moveenv{#1*}{\eql@#1*}%
4197     }%
4198     \eql@provide@moveenv{#1}{\eql@#1}%
4199     \eql@provide@moveenv{#1*}{\eql@#1*}%
4200     \eql@amsmath@before{\eql@provide@undefineenv{#1}}%
4201     \eql@amsmath@before{\eql@provide@undefineenv{#1*}}%
4202   \fi
4203 }
```

TODO: describe

```
4204 \def\eql@provide@env@equation#1{%
4205   \eql@provide@onlyonce{equation}{#1}%
4206   \ifdefined\eql@tmp
4207     \expandafter\eql@provide@moveenv\expandafter{\eql@tmp}{\eql@equation}%
4208   \else
4209     \eql@amsmath@after{%
4210       \eql@provide@moveenv{amsequation}{equation}%
4211       \eql@provide@moveenv{amsequation*}{equation*}%
4212       \eql@provide@moveenv{equation}{\eql@equation}%
4213       \eql@provide@moveenv{equation*}{\eql@equation*}%
4214     }%
```

```

4215 \AddToHook{package/hyperref/after}{%
4216 \ifpackageloaded{amsmath}{}{%
4217 \let\latexequation\H@equation
4218 \let\endlatexequation\H@endequation
4219 \eql@provide@moveenv{hyperrefequation}{equation}%
4220 \eql@provide@moveenv{equation}{eql@equation}%
4221 }%
4222 }%
4223 \ifpackageloaded{amsmath}{}{\ifpackageloaded{hyperref}{}{%
4224 \eql@provide@moveenv{latexequation}{equation}%
4225 }}%
4226 \eql@provide@moveenv{equation}{eql@equation}%
4227 \eql@provide@moveenv{equation*}{eql@equation*}%
4228 \eql@amsmath@before{\eql@provide@undefineenv{equation*}}%
4229 \ifdefined\eql@tagging@on
4230 \AddToHook{begindocument/end}{%
4231 \eql@provide@moveenv{equation}{eql@equation}%
4232 \eql@provide@moveenv{equation*}{eql@equation*}%
4233 }%
4234 \fi
4235 \fi
4236 }

```

TODO: describe

```

4237 \def\eql@provide@env@multlined#1{%
4238 \eql@provide@onlyonce{multlined}{#1}%
4239 \ifdefined\eql@tmp
4240 \expandafter\eql@provide@moveenv\expandafter{\eql@tmp}{eql@multlined}%
4241 \else
4242 \AddToHook{package/mathtools/after}{%
4243 \eql@provide@moveenv{amsmultlined}{multlined}%
4244 \eql@provide@moveenv{multlined}{eql@multlined}%
4245 }%
4246 \eql@provide@moveenv{multlined}{eql@multlined}%
4247 \ifpackageloaded{mathtools}{}{\AddToHook{package/mathtools/before}{%
4248 \eql@provide@undefineenv{multlined}}}%
4249 \fi
4250 }

```

TODO: describe

```

4251 \def\eql@provide@env@subequations#1{%
4252 \eql@provide@onlyonce{subequations}{#1}%
4253 \ifdefined\eql@tmp
4254 \expandafter\eql@provide@moveenv
4255 \expandafter{\eql@tmp}{eql@subequations}%
4256 \else
4257 \eql@amsmath@after{%
4258 \eql@provide@moveenv{amssubequations}{subequations}%
4259 \eql@provide@moveenv{subequations}{eql@subequations}%
4260 }%
4261 \AddToHook{package/hyperref/after}{%
4262 \AddToHook{cmd/subequations/before}[hyperref]{}%
4263 \AddToHook{cmd/subequations/after}[hyperref]{}%
4264 \RemoveFromHook{cmd/subequations/before}[hyperref]%
4265 \RemoveFromHook{cmd/subequations/after}[hyperref]%
4266 \AddToHook{cmd/amssubequations/before}%
4267 {%
4268 \stepcounter{equation}%

```

```

4269     \protected@edef\theHparentequation{\theHequation}%
4270     \addtocounter{equation}{-1}%
4271   }%
4272   \AddToHook{cmd/amssubequations/after}%
4273   {%
4274     \def\theHequation{\theHparentequation\alph{equation}}%
4275     \ignorespaces
4276   }%
4277   \AddToHook{begindocument/end}{%
4278     \eql@provide@moveenv{subequations}{eql@subequations}%
4279   }%
4280 }%
4281 \eql@provide@moveenv{subequations}{eql@subequations}%
4282 \eql@amsmath@before{\eql@provide@undefineenv{subequations}}%
4283 \fi
4284 }

```

TODO: describe

```

4285 \def\eql@provide@sqr{%
4286   \let\[\eql@equations@sqr@open
4287   \let\]\eql@equations@sqr@close
4288   \eql@amsmath@after{%
4289     \let\[\eql@equations@sqr@open
4290     \let\]\eql@equations@sqr@close
4291   }%
4292   \ifdefined\eql@tagging@on
4293     \AddToHook{begindocument/end}{%
4294       \let\[\eql@equations@sqr@open
4295       \let\]\eql@equations@sqr@close
4296     }%
4297   \fi
4298 }

```

TODO: describe

```

4299 \def\eql@provide@ang{%
4300   \let\<\eql@equations@ang@open
4301   \let\>\eql@equations@ang@close
4302 }

```

TODO: describe

```

4303 \eql@define@key{provide}{equation} [] {\eql@provide@env@equation{#1}}
4304 \eql@define@key{provide}{gather} [] {\eql@provide@env{gather}{#1}}
4305 \eql@define@key{provide}{multline} [] {\eql@provide@env{multline}{#1}}
4306 \eql@define@key{provide}{align} [] {\eql@provide@env{align}{#1}}
4307 \eql@define@key{provide}{flalign} [] {\eql@provide@env{flalign}{#1}}
4308 \eql@define@key{provide}{aligned} [] {\eql@provide@env{aligned}{#1}}
4309 \eql@define@key{provide}{gathered} [] {\eql@provide@env{gathered}{#1}}
4310 \eql@define@key{provide}{multlined} [] {\eql@provide@env{multlined}{#1}}
4311 \eql@define@key{provide}{subequations} [] {\eql@provide@env@subequations{#1}}
4312 \eql@define@key{provide}{sqr} [] {\eql@provide@sqr}
4313 \eql@define@key{provide}{ang} [] {\eql@provide@ang}
4314 \eql@define@key{provide}{eqref} [] {\eql@provide@cmd{eqref}{#1}}
4315 \eql@define@key{provide}{notag} [] {\eql@provide@cmd{notag}{#1}}
4316 \eql@define@key{provide}{thetag} [] {\eql@provide@cmd{thetag}{#1}}
4317 \eql@define@key{provide}{allowdisplaybreaks} [] {%
4318   \eql@provide@cmd{allowdisplaybreaks}{#1}}
4319 \eql@define@key{provide}{numberwithin} [] {\eql@provide@cmd{numberwithin}{#1}}
4320 \eql@define@key{provide}{tagform} [] {%

```

```

4321 \def\tagform@##1{\maketag@@{\eql@tag@form{#1}}}}
4322 \eql@define@key{provide}{\maketag}[]{}%
4323 \def\maketag@@@##1{\eql@tag@box{##1}}}
```

TODO: describe

```

4324 \newcommand{\eqnlinesprovide}[1]{%
4325 (dev)\eql@dev@start\eqnlinesprovide
4326 \eql@setkeys{provide}{#1}}
```

P.6 Global and Package Options

Handle global and package options:

`\eqnlineset` The macro `\eqnlineset` processes global configuration options including the package options:

```

4327 \newcommand{\eqnlineset}[1]{%
4328 (dev)\eql@dev@start\eqnlineset
4329 \eql@setkeys{setup}{#1}}
```

Disable error message for exclusive package options:

```
4330 \let\eql@error@packageoption@gobble
```

Declare math layout options `leqno` and `fleqn` for common L^AT_EX classes:

```

4331 \DeclareOption{leqno}{\eqnlineset{tagsleft}}
4332 \DeclareOption{fleqn}{\eqnlineset{left}}
```

Pass undeclared options on to keyval processing:

```
4333 \DeclareOption*{\expandafter\eqnlineset\expandafter{\CurrentOption}}
```

Process package options:

```
4334 \ProcessOptions
```

`@error@packageoption` Enable error message for exclusive package options:

```

4335 \def\eql@error@packageoption#1{%
4336 \eql@error{may only use ‘#1’ as a package option}%
4337 }
```

Make sure that the `amsmath` conditionals `\iftagsleft@` and `\if@fleqn` are declared without spelling out their name which may upset the T_EX conditional parsing mechanism:

```

4338 \ifdefined\tagsleft@true\else
4339 \expandafter\newif\csname iftagsleft@\endcsname
4340 \fi
4341 \ifdefined\@fleqntrue\else
4342 \expandafter\newif\csname if@fleqn\endcsname
4343 \fi
```

Import `amsmath` switches `leqno` as `tagsleft` and `fleqn` as `left`:

```

4344 \ifdefined\eql@provide@opt@amsmath
4345 \let\eql@provide@opt@equation\eql@true
4346 \eql@amsmath@after{%
4347 \iftagsleft@
4348 \eqnlineset{tagsleft}
4349 \else
```

```

4350     \eqnlineset{tagsright}
4351     \fi
4352     \if@fleqn
4353     \eqnlineset{left}
4354     \else
4355     \eqnlineset{center}
4356     \fi
4357 }
4358 \fi

```

Provide native L^AT_EX environment `equation` and symbolic shortcut `\[...\]` if desired:

```
4359 \ifdefined\eql@provide@opt@equation\eqnlinesprovide{equation,sqr}\fi
```

Make the ending statements for `amsmath` environments independent if desired, so that they may be overwritten individually:

```
4360 \ifdefined\eql@provide@opt@amsmathends\eql@amsmath@fixends\fi
```

Provide `amsmath` equation environments if desired:

```

4361 \ifdefined\eql@provide@opt@amsmath
4362 \eqnlinesprovide{%
4363     multiline,gather,align,flalign,%
4364     multlined,gathered,aligned,%
4365     subequations}
4366 \fi

```

Provide symbolic shortcut `\<...\>` if desired:

```
4367 \ifdefined\eql@provide@opt@ang\eqnlinesprovide{ang}\fi
```

Provide equation reference `\eqref` if desired:

```
4368 \ifdefined\eql@provide@opt@eqref\eqnlinesprovide{eqref}\fi
```