The xgalley package
Galley

The LaTeX Project*

Released 2021-11-12

1 Introduction

In LaTeX3 terminology a galley is a rectangular area which receives text and other material filling it from top. The vertically extend of a galley is normally not restricted: instead certain chunks are taken off the top of an already partially filled galley to form columns or similar areas on a page. This process is typically asynchronous but there are ways to control or change its behaviour.

Examples for galleys are "the main galley", where the continuous document data gets formatted into and from which columns and pages are constructed, and "vertical box galleys", such as the body of a minipage environment. The latter galleys are typically not split after formatting, though there can be exceptions.

2 Formatting layers

The present module is mainly concerned with the formatting of text in galleys. The mechanism by which this is achieved uses four (somewhat) distinct layers, some of which can be addressed using the templates provided here.

2.1 Layer one: external dimensions

The bottom layer of the system is the external dimensions of the galley. Normally only the horizontal dimension is fixed externally, while the vertical (filling) dimension is unspecified. The external dimensions are fixed when starting a new galley, and are therefore not modifiable within the galley.

There are no templates for setting this layer directly, although the external values are influenced by other parts of the system (for example when creating minipage environments).

*E-mail: latex-team@latex-project.org

2.2 Layer two: internal dimensions

The second layer is the internal dimensions of the galley: the *measure* used for paragraph text and the position of the paragraph relative to the edges of the galley.

This layer is normally accessed by higher-level templates *via* the object type `measure`. Changes made using level two templates will often extend for large parts of a document (up to and including the entire document).

2.3 Layer three: paragraph shape

The third layer defines the paragraph shape within the measure as provided by the second layer. In the absence of any specification for that layer the paragraph shape used will be that of a rectangular area of the width of the current measure.

There are some restrictions imposed on the shape of a paragraph by the underlying TEX mechanisms. For example, cut out sections in paragraphs can be specified from the top of the paragraph but not from the bottom.

2.4 Layer four: formatting inside the paragraph

The forth layer deals with the paragraph formatting aspects such as hyphenation and justification within the paragraph (this is sometimes referred to as "`h&j`" or "`hj`").

3 Templates

3.1 Layer two: internal dimensions

3.2 The object type 'measure'

Arg:

Semantics:

Sets the width available to typeset material within the galley. The ⟨*left margin*⟩ and ⟨*right margin*⟩ values are used in the adjustment to over-ride any given in the template. Depending upon the template in use, the margins may be absolute (relative only to the edges of the galley) or relative (taking account of `measure` adjustments already made). The template applies to the galley from the point of us forward, unless over-ridden by another use of the `measure` object type.

3.3 The template 'absolute' (object type measure)

Attributes:

left-margin (length) The distance from the left edge of the galley to the left edge of the area for typeset material. A negative value will cause the typeset material to extend beyond the edge of the galley. Default: 0pt

right-margin (length) The distance from the right edge of the galley to the right edge of the area for typeset material. A negative value will cause the typeset material to extend beyond the edge of the galley. Default: 0pt

Semantics & Comments:

This template sets up the typesetting area such that typeset material runs from `left-margin` away from the left edge of the galley to `right-margin` away from the right edge of the galley. Both of these distances are absolute, *i.e.* no account is taken of previous `measure` settings. Either on or both values may be negative, in which case the typeset material will protrude outside of the edges of the galley.

3.4 The template 'relative' (object type measure)

Attributes:

left-margin (length) The distance from the previous left margin of the typeset material within the galley to the new position of the left margin. A negative value will cause the new margin to be "outside" of the previous one, and *may* cause the typeset material to protrude outside of the edge of the galley. Default: 0pt

right-margin (length) The distance from the previous right margin of the typeset material within the galley to the new position of the right margin. A negative value will cause the new margin to be "outside" of the previous one, and *may* cause the typeset material to protrude outside of the edge of the galley. Default: 0pt

Semantics & Comments:

This template sets up the typesetting area such that it has margins `left-margin` and `right-margin` within those previously set. For a galley within no previous margins, this will result in margins relative to the edges of the galley. Within a galley in which the `measure` has already been set, using the `relative` template will indent the typeset material relative to the existing margins. Either on or both values may be negative, in which case the typeset material may protrude outside of the edges of the galley.

3.5 Layer three: paragraph shape

3.6 The object type 'parshape'

Arg:

Semantics:

Template of this type define any shaping of the paragraph within the current measure of the galley. Thus they are used to generate "special" paragraph shapes, for example placing a cutout in one side of the paragraph. Typically, `parshape` templates will apply in a limited sense (to a single paragraph or a defined number of lines). However, `parshape` templates may also apply in an "ongoing" manner.

Note that `parshape` templates do not alter any first-line indent for paragraphs (or any other "in paragraph" setting). Instead, they define a shape inside which the paragraph material will be placed.

3.7 The template 'hang' (object type parshape)

Attributes:

indent (length) The hanging indent from either the left- or right-hand margin (as determined by `on-left-side`). Default: 0pt

on-left-side (boolean) If `true`, causes the hanging indent to be on the left-hand side of the paragraph. Default: true

lines (integer) The number of lines of full width before hanging begins. Default: 1

Semantics & Comments:

Sets the paragraph shape such that the after a number of full-width lines, specified by `lines`, the paragraph is indented by the `indent` from a margin. If `on-left-side` is `true` this indent will be from the left-hand margin, otherwise it will be from the right. In either case, the indent is relative to the edge of the current `measure` and may be negative (in which case an outdent will result). This template type applies only to a single paragraph.

3.8 The template 'initial' (object type parshape)

Attributes:

indent (length) The indent for the initial lines from either the left- or right-hand margin (as determined by `on-left-side`). Default: 0pt

on-left-side (boolean) If `true`, causes the indent to be on the left-hand side of the paragraph. Default: true

lines (integer) The number of lines of indented lines before full-width line begins. Default: 2

Semantics & Comments:

Sets the paragraph shape such that the first `lines` lines are indented by the `indent` given, before lines of full width begin. If `on-left-side` is `true` this indent will be from the left-hand margin, otherwise it will be from the right. In either case, the indent is relative to the edge of the current `measure` and may be negative (in which case an outdent will result). This template type applies only to a single paragraph.

3.9 The template 'std' (object type parshape)

Attributes:

()

Semantics & Comments:

Sets a rectangular paragraph shape which occupies the full width specified by the `measure`. It is therefore intended as a "do nothing" template for use where a paragraph shape is required but where no special formatting is needed. This template type applies only to a single paragraph.

3.10 Layer four: formatting inside the paragraph

3.11 The object type 'hyphenation'

Arg:

Semantics:

Controls whether hyphenation is attempted within the current galley. This object type may also alter the degree to which hyphenation is encouraged by manipulating the underlying TEX parameters. This object type applies to the galley from the point of use forward.

3.12 The template 'std' (object type hyphenation)

Attributes:

enable (boolean) Switches all hyphenation on or off. Default: true

enable-upper-case (boolean) Switches hyphenation on or off for words beginning with upper case letters. Default: true

penalty (choice) Sets the degree to which TEX is discouraged from undertaking hyphenation, from the choices `low`, `medium` and `high`. Default: low

Semantics & Comments:

Determines both whether hyphenation is allowed at all, and if so to what degree it is discouraged. Setting `penalty` to `high` does not prevent hyphenation: this is only done if `enable` is set `false`.

3.13 The object type 'justification'

Arg:

Semantics:

Controls the nature of justification undertaken within the galley. The template applies from the point of use forward.

3.14 The template 'std' (object type justification)

Attributes:

end-skip (skip) The skip inserted to fill the last line of a paragraph. Default: 0pt plus 1fil

fixed-word-spacing (boolean) Determines whether inter-word spacing has a stretch component (for non-monospaced fonts. Default: false

indent-width (length) The length of the indent inserted at the start of the first line of a new paragraph.

left-skip (skip) The skip between the left margin of the galley and the left edge of a paragraph. Default: 0pt

right-skip (skip) The skip between the right margin of the galley and the right edge of a paragraph. Default: 0pt

start-skip (skip) The skip inserted in addition to `indent-width` at the start of a paragraph. Default: 0pt

Semantics & Comments:

The `std` template for justification provides rubber lengths at the start and end of the paragraph and at each side of the paragraph. It also allows for both flexible and fixed inter-word spacing. The interaction between the settings is demonstrated in the selection of standard instances provided.

3.14.1 The instance 'justified' (template justification/std)

Attribute values:

indent-width 15pt

Layout description & Comments:

Sets paragraphs fully-justified with the first line indented by `15pt`.

3.14.2 The instance 'noindent' (template justification/std)

Attribute values:

end-skip 15pt plus 1fil
indent-width 0pt

Layout description & Comments:

Sets paragraphs fully-justified with no indent for the first line. To ensure that paragraphs have some visual distinction, the `end-skip` is set to insert some space in all cases.

3.15 The template 'single' (object type justification)

Attributes:

end-skip (skip) The skip inserted to fill the last line of a paragraph. Default: 0pt plus 1fil

fixed-word-spacing (boolean) Determines whether inter-word spacing has a stretch component (for non-monospaced fonts. Default: false

indent-width (length) The length of the indent inserted at the start of the first line of a new paragraph.

left-skip (skip) The skip between the left margin of the galley and the left edge of a paragraph. Default: 0pt

right-skip (skip) The skip between the right margin of the galley and the right edge of a paragraph. Default: 0pt

start-skip (skip) The skip inserted in addition to `indent-width` at the start of a paragraph. Default: 0pt

stretch-last-line (boolean) Determines whether inter-word spacing in the last line is stretched. If `true`, the spacing in the last line is stretched in the by the same factor as that in the penultimate line. Default: false

Semantics & Comments:

The `single` template for justification provides rubber lengths at the start and end of the paragraph and at each side of the paragraph. It also allows for both flexible and fixed inter-word spacing. The interaction between the settings is demonstrated in the selection of standard instances provided. The template applies only to a single paragraph.

3.15.1 The instance 'ragged-left' (template justification/std)

Attribute values:

end-skip	0pt
fixed-word-spacing	true
indent-width	0pt
left-skip	0pt plus 2em
right-skip	0pt

Layout description & Comments:

Typesets material with a ragged left margin such that hyphenation will still occur and such that very short lines are discouraged. This is similar to the LaTeX 2_ε `ragged2e` `RaggedLeft` environment.

3.15.2 The instance 'ragged-right' (template justification/std)

Attribute values:

end-skip	0pt
fixed-word-spacing	true
indent-width	0pt
left-skip	0pt
right-skip	0pt plus 2em

Layout description & Comments:

Typesets material with a ragged right margin such that hyphenation will still occur and such that very short lines are discouraged. This is similar to the LaTeX 2_ε `ragged2e` `RaggedLeft` environment.

3.15.3 The instance 'center' (template justification/std)

Attribute values:

end-skip	0pt
fixed-word-spacing	true
indent-width	0pt
left-skip	0pt plus 1fil
right-skip	0pt plus 1fil

Layout description & Comments:

Centres typeset material such that hyphenation is discouraged and short lines are allowed.

3.16 The template 'compound' (object type justification)

Attributes:

first-paragraph (instance) Justification for the first paragraph.

other-paragraphs (instance) Justification for the remaining paragraphs.

Semantics & Comments:

Here, both keys should themselves be instances of the `justification` template. The `compound` template is used to set up a single "non-standard" paragraph followed by "standard" ones. For example, it can be used to ensure that one `noindent` paragraph is then followed by `std` justification.

3.17 The object type 'line-breaking'

Arg:

Semantics:

Controls the line breaking attempted by TEX when typesetting material for the galley. This does not include whether words are hyphenated, which is handled separately.

3.18 The template 'std' (object type line-breaking)

Attributes:

badness (integer) Boundary that if exceeded will cause TEX to report an underfull line. Default: 1000

binop-penalty (integer) Penalty charged if an inline math formula is broken at a binary operator. Default: 700

double-hyphen-demerits (integer) Extra demerit charge of two (or more) lines in succession end in a hyphen. Default: 10000

emergency-stretch (skip) Additional stretch assumed for each line if no better line breaking can be found without it. This stretch is not actually added to lines, so its use may result in underfull box warnings. Default: 0pt

final-hyphen-demerits (integer) Extra demerit charge if the second last line is hyphenated. Default: 5000

fuzz (length) Boundary below overfull lines are not reported. Default: 0.1pt

mismatch-demerits (integer) Extra demerit charge if two visually incompatible lines follow each other. Default: 10000

line-penalty (integer) Extra penalty charged per line in the paragraph. By making this penalty higher TEX will try harder to produce compact paragraphs.Default: 10

pretolerance (integer) Maximum tolerance allowed for individual lines to break the paragraph without attempting hyphenation. Default: 100

relation-penalty (integer) Penalty charged if an inline math formula is broken at a relational symbol. Default: 500

tolerance (integer) Maximum tolerance allowed for individual lines when breaking a paragraph while attempting hyphenation (if this limit can't be met `emergency-stretch` comes into play). Default: 200

Semantics & Comments:

This is an interface to the underlying TEX system for determining line breaking.

3.19 Between paragraphs

3.20 The object type 'paragraph-breaking'

Arg:

Semantics:

This object type determines how TEX determines the behaviour when the paragraph-breaking algorithm is calculating whether to break up a paragraph. Thus for example an instance of this object type may prevent breaks within a paragraph, forbid widows or orphans, *etc.*

3.21 The template 'std' (object type paragraph-breaking)

Attributes:

badness (integer) Boundary that if exceeded will cause TEX to report an underfull vertical box. Default: 1000

broken-penalty (integer) Penalty for page breaking after a hyphenated line. Default: 100

club-penalty (integer) Penalty for generating a club line when page breaking. Default: 150

display-club-penalty (integer) Penalty for breaking between to leave a club line after display math. Default: 150

display-widow-penalty (integer) Penalty for breaking between to leave a widow line before display math. Default: 150

fuzz (length) Boundary below which overfull vertical boxes are not reported. Default: 0.1pt

interline-penalty (integer) Penalty for breaking between lines in a paragraph. Default: 0

pre-display-penalty (integer) Penalty for breaking between immediately before display math material. Default: 10000

post-display-penalty (integer) Penalty for breaking between immediately after display math material. Default: 0

widow-penalty (integer) Penalty for generating a widow line when page breaking. Default: 150

Semantics & Comments:

This template provides an interface to the underlying TEX mechanism for controlling page breaking. The template applies on an ongoing basis to all paragraphs after the template is used.

3.21.1 The instance 'std' (template paragraph-breaking/std)

Attribute values:

Layout description & Comments:

Sets paragraphs such that they can break with widows and orphans discouraged but not prevented. Breaks are possible after display math material but no immediately before it.

3.21.2 The instance 'nobreak' (template paragraph-breaking/std)

Attribute values:

interline-penalty	10000
post-display-penalty	10000

Layout description & Comments:

Sets paragraphs such that they cannot be broken at all (as far as is possible in TEX).

3.21.3 The instance 'nolone' (template paragraph-breaking/std)

Attribute values:

club-penalty	10000
display-widow-penalty	10000
widow-penalty	10000

Layout description & Comments:

Sets paragraphs such that they cannot be broken to leave a club or widow line (as far as is possible in TEX).

3.22 The template 'single' (object type paragraph-breaking)

Attributes:

badness (integer) Boundary that if exceeded will cause TEX to report an underfull vertical box. Default: ⟨*none*⟩

broken-penalty (integer) Penalty for page breaking after a hyphenated line. Default: ⟨*none*⟩

club-penalty (integer) Penalty for generating a club line when page breaking. Default: ⟨*none*⟩

display-club-penalty (integer) Penalty for breaking between to leave a club line after display math. Default: ⟨*none*⟩

display-widow-penalty (integer) Penalty for breaking between to leave a widow line before display math. Default: ⟨*none*⟩

fuzz (length) Boundary below which overfull vertical boxes are not reported. Default: ⟨*none*⟩

interline-penalty (integer) Penalty for breaking between lines in a paragraph. Default: ⟨*none*⟩

pre-display-penalty (integer) Penalty for breaking between immediately before display math material. Default: ⟨*none*⟩

post-display-penalty (integer) Penalty for breaking between immediately after display math material. Default: ⟨*none*⟩

widow-penalty (integer) Penalty for generating a widow line when page breaking. Default: ⟨*none*⟩

Semantics & Comments:

This template provides an interface to the underlying TEX mechanism for controlling page breaking. The template applies only to the next paragraph, and can thus be used to achieve effects such as non-breaking paragraphs.

3.22.1 The instance 'single-std' (template paragraph-breaking/single)

Attribute values:

Layout description & Comments:

Sets the next paragraph such that it can break with widows and orphans discouraged but not prevented. Breaks are possible after display math material but no immediately before it.

3.22.2 The instance 'single-nobreak' (template paragraph-breaking/single)

Attribute values:

interline-penalty	10000
post-display-penalty	10000

Layout description & Comments:

Sets the next paragraph such that it cannot be broken at all (as far as is possible in TEX).

3.22.3 The instance 'single-noclub' (template paragraph-breaking/single)

Attribute values:

club-penalty	10000
display-club-penalty	10000

Layout description & Comments:

Sets the next paragraph such that it cannot be broken to leave a club line (as far as is possible in TEX).

3.22.4 The instance 'single-nolone' (template paragraph-breaking/single)

Attribute values:

club-penalty	10000
display-club-penalty	10000
display-widow-penalty	10000
widow-penalty	10000

Layout description & Comments:

Sets the next paragraph such that it cannot be broken to leave a club or widow line (as far as is possible in TEX).

3.22.5 The instance 'single-nowidow' (template paragraph-breaking/single)

Attribute values:

display-widow-penalty	10000
widow-penalty	10000

Layout description & Comments:

Sets the next paragraph such that it cannot be broken to leave a widow line (as far as is possible in TEX).