

L^AT_EX News

Issue 44, November 2026 — DRAFT version for upcoming release (L^AT_EX release 2026-11-01)

Contents

Introduction	1
News from the Tagged PDF project	1
Tags for paragraphs	1
Provide an <code>order</code> key mechanism for templates	1
Revisiting the para-handling logic for block environments	1
Replaced <code>legacy-code</code> key in blockenv template	2
Support <code>force-unnumbered</code> key in heading templates	2
Support <code>placement=ragged</code> in heading templates	2
Heading templates now support column spanning	2
Heading template instances for AMS classes . .	2
Code improvements	2
Support for <code>\UseName</code> in template key bindings	2
Further removal of “fake math” from the kernel	2
Documentation changes	2
US Spelling	2

Introduction

to write

News from the Tagged PDF project

Point to talks about accessibility given at Calgary

Tags for paragraphs

Most of the time a *paragraph* just consists of a number of sentences without any further structure. However, it is also possible that it has a more complicated structure, e.g., it may contain a quote or a list with some text before or after, all belonging to the same paragraph. For example

```
Don Knuth sends a warm welcome from afar
and writes:
\begin{quote}
  \sffamily Happy TeX'ing in Calgary
\end{quote}
This is what we did!
```

To model this in the PDF structure, L^AT_EX surrounds the whole paragraph with a `<semantic-para>` tag and inside individual text blocks with their own tags, i.e., in the above example the text `Don ... writes:` and the

text `This is ...!` each with a `<text-block>` tag and the quote with a `<quote>` tag¹, so you end up with

```
<semantic-para>
  <text-block>
    Don Knuth sends a warm welcome from afar
    and writes:
  </text-block>
  <quote>
    <text-block>
      Happy TeX'ing in Calgary
    </text-block>
  </quote>
  <text-block>
    This is what we did!
  </text-block>
</semantic-para>
```

Initially, we called these paragraph-related tags `<text-unit>` (now `<semantic-para>`) and `<text>` (now `<text-block>`) but the original names were difficult to understand, so we changed them now.

We also introduced the `<text-fragment>` tag for use inside `<text-block>`s if there is a need for a sub-structure.

Provide an order key mechanism for templates
to write

For now only available when `\DocumentMetadata` is used!

Revisiting the para-handling logic for block environments
In L^AT_EX, block environments like `itemize` or `quote` typically scan for a following empty line and if there is none then text following the environment is expected to belong to the current semantic paragraph and continues it, e.g., gets no paragraph indentation or anything else that signals the start of a new semantic paragraph.

In some situations or for some types of block environments (such as theorem-like environments) this behavior is not wanted. We have therefore added a new boolean template key `standalone` which if set to `true` causes the environment to surround itself with implicit `\par` commands mimicking the effect of explicit empty lines in the source.

¹Note that HTML does not have the context of a semantic paragraph and would mark the text before and after the quote each with a `<P>` and thus effectively claiming that they are individual paragraphs while in reality they form together a single paragraph.

We took the opportunity to also fix a couple of subtle problems with the mechanism so that it hopefully works better in a few unusual situations.
(tagging-project issues 1402 and 1415 among others)

Replaced legacy-code key in blockenv template

The blockenv template had a `legacy-code` key, used to support some legacy setup for the generic `list` environment and for the `verbatim` extension offered by the `ltugboat` class where you can write `\begin{verbatim}[\small]` to get a verbatim display in a smaller font size.

However, if the `verbatim` was directly preceded by text the content of the key was evaluated while still being in horizontal mode. In that case the `\small` changed not only the verbatim display but, as a side effect, also altered the `\baselineskip` of the preceding text. This has been corrected by providing two keys: `early-legacy-code` which is evaluated near the start of the template, possibly still in horizontal mode (used by `list`) and `late-legacy-code` which is evaluated after returning to vertical mode (used by `verbatim`).

(tagging-project issue 1503)

Support force-unnumbered key in heading templates to write
(tagging-project issue 1473)

Support placement=ragged in heading templates

The `placement` key was augmented to support the value `ragged` in addition to `page`, `top`, and `normal`. This works like `normal` (i.e., the heading can appear anywhere on the page) but if it happens to trigger a page break then the previous page will be set “raggedbottom” and not spread out to fill the page in the `book` or `report` classes. This is useful on pages that do not have a lot of stretchability in the first place and thus the space between paragraphs is used for stretching.

It can either be specified on the instance level by setting the `placement` key to `ragged` or for individual headings in the document by setting it in the optional argument to the heading.
(github issue 1844)

Heading templates now support column spanning

In twocolumn documents top-level headings usually start on a new page and span both columns. In $\text{\LaTeX}$ 2 ϵ this was hardwired and headings such as `\chapter` showed this behavior. In the new template-based implementation this is now more flexible and one sets this independently from forcing a new page by setting the key `column-spanning` to `true` (default for `\chapter`) or `false` (default for other headings). Thus

```
\section[column-spanning=true]{title}
```

generates a one-off `\section` heading that spans both columns. Due to the way the $\text{\LaTeX}$ output routine is currently implemented that automatically also starts a

new page, i.e., one can’t generate spanning headings in the middle of a page (that can only be done with the `multicol` package at the moment). That restriction may be lifted when the output routine is redefined.

(tagging-project issue 1436)

Heading template instances for AMS classes to write including \AddPunctuation

(tagging-project issue 1477)

Code improvements

Support for \UseName in template key bindings

We have extended the logic in the key-binding argument for `\DeclareTemplateCode`, such that you can use the keyword `name` here (in addition to `global`) without any performance impact when using templates. This allows automatic variable creation with “awkward” characters, most notably – and numerals. Of course, such variables then also need to be referred to via `\UseName` or some equivalent method in the code, so this is only useful if there is a pressing need for such special names.

Further removal of “fake math” from the kernel

In the previous release, we removed most use of “fake math” (math mode not denoting a formula but used for positioning or manipulating text material) from the kernel [2]. The use of `\underline` in text mode was missed in that update: this has now been addressed.

Documentation changes

US Spelling

We have (finally) decided that standardizing the spelling in the $\text{\LaTeX}$ sources to use US-English is the realistic position.² This means that the documentation better aligns with the code. Note that idiom may still be more varied: this is harder to adjust than spelling.

References

- [1] Leslie Lamport. *$\text{\LaTeX}$: A Document Preparation System: User’s Guide and Reference Manual*. Addison-Wesley, Reading, MA, USA, 2nd edition, 1994. ISBN 0-201-52983-1. Reprinted with corrections in 1996.
- [2] $\text{\LaTeX}$ Project Team. *$\text{\LaTeX}$ 2 ϵ news 1–44*. November, 2026. <https://latex-project.org/news/latex2e-news/ltnews.pdf>

²None of the native English speakers on the team are from the US!