News from the LaTeX Tagged PDF project: 2025

Ulrike Fischer, Frank Mittelbach

Abstract

The LaTeX Tagged PDF project was started in spring 2020 and announced to the TeX community by the LaTeX Team at the (online) 2020 TUG conference. This short report describes some news of this multi-year project.

Contents

1 Introduction

Over the last few years we have written a number of papers describing the motivation, the goals and the progress of the Tagged PDF project, e.g., [2, 3, 6, 7, 9]. Please refer to these papers for background. Here we note only some current news.

More material can be found on GitHub: `github.com/latex3/tagging-project`

2 Consolidating the interface

The tagging code started in 2018 as the standalone package `tagpdf`.

In 2022 we added the `\DocumentMetadata` command to be used as the very first declaration in a document as a means to load new code needed for the tagging support, and so also to act as a trigger command to identify documents that *want* to load new code.[1]

Initially the command loaded automatically only the PDF management. Other new code, organized in modules, had to be loaded manually with the help of the (temporary) `testphase` key. The values for this key have changed over time with the growing number of available modules that added tagging support to sectioning, graphics, lists and more. Over the last years the recommended values were for example:

```
\DocumentMetadata{testphase=phase-I}
\DocumentMetadata{testphase=phase-II}
\DocumentMetadata{testphase=phase-III}
\DocumentMetadata{testphase=
   {phase-III,
    math,
    title,
    firstaid,
    tikz}}
```

The last one was so long that we decided to introduce a shortcut to load "all that is sensible": `\DocumentMetadata{testphase=latest}`.

The tagging support now covers all standard document elements and we decided that it is time to retire the `testphase` key and to offer a more stable interface: the `tagging` key. Thus, in a current LaTeX 2025-06-01 release

- `\DocumentMetadata{}` loads the PDF management, sets the font encoding to T1 and the PDF version to 2.0,
- `\DocumentMetadata{tagging=off}` also loads all the modules for tagging support that were loaded with `testphase=latest`, but doesn't activate tagging.
- `\DocumentMetadata{tagging=on}` loads all the modules and activates tagging.

However, this will further change with the upcoming release and `\DocumentMetadata` will then always load the new code, i.e., will be equivalent to specifying `tagging=off` as explained in the next section.

2.1 Future plans

Loading the various modules individually was useful for testing and development, but it is a problem for the growing number of packages and classes that want to add support for tagging: they cannot easily test which parts of the new code are loaded in a document. While the use of `\DocumentMetadata` can be tested with `\IfDocumentMetadataTF`, this test doesn't allow deciding if, for example, the new math code or the new list code was active, or not.

So, starting with LaTeX 2025-11-01 we will *always* load all modules directly if `\DocumentMetadata` is used. `\DocumentMetadata{}` will do the same as `\DocumentMetadata{tagging=off}`. Key settings like `testphase={phase-III,math}` or `testphase=phase-I` will be meaningless as they will no longer load only a restricted set of modules.

For documents that want to load only the PDF management, we provide a package. Such documents should replace

[1] Thus `\DocumentMetadata` identifies a new generation of LaTeX documents, just like the use of `\documentclass` in 1994 identified new LaTeX 2ε documents with functionality different from LaTeX 2.09 documents that used `\documentstyle`.

doi.org/10.47397/tb/46-2/tb143fischer-tagging25

```
\DocumentMetadata{pdfstandard=A-2b}
```

by

```
\RequirePackage{pdfmanagement}
\SetKeys[document/metadata]{pdfstandard=A-2b}
```

These setup changes can already be tested by using the newest `latex-dev`.

3 `show-pdf-tags`: A tool to check and validate tagged PDF

Marcel Krüger and David Carlisle have developed a tool, `show-pdf-tags` [4], which extracts the structure tree from a PDF and either outputs it as a tree or interprets it as XML. The tool is available in TeX Live and MiKTeX.

The resulting XML may be validated using standard XML tools using RelaxNG schema that are provided.

The tool can be run from the command line, e.g.:

```
show-pdf-tags --xml file.pdf
```

shows an output like this (modulo line breaks):

```
<PDF>
 <StructTreeRoot>
  <Document xmlns="http://iso.org/pdf2/ssn"
     id="ID.0002"
    >
   <Sect xmlns="http://iso.org/pdf2/ssn"
      id="ID.0005"
      xmlns:orig-ns="https://www.latex-project
    .org/ns/local/ltx-talk"
      rolemapped-from="orig-ns:frame"
     >
    <Part xmlns="http://iso.org/pdf2/ssn"
       id="ID.0006"
       xmlns:orig-ns="https://www.latex-
    project.org/ns/dflt"
       rolemapped-from="orig-ns:text-unit"
      >
     <Title xmlns="http://iso.org/pdf2/ssn"
        id="ID.0007"
       >
      <P xmlns="http://iso.org/pdf2/ssn"
         id="ID.0008"
         xmlns:Layout="http://iso.org/pdf/ssn/
    Layout"
         Layout:TextAlign="Center"
         xmlns:orig-ns="https://www.latex-
    project.org/ns/dflt"
         rolemapped-from="orig-ns:text"
        >
       <?MarkedContent page="1" ?>Status of
    the LaTeX Tagged PDF Project
      </P>
    ...
```

There is also an online version at `texlive.net/showtags` where you can upload a file and then validate it with the RelaxNG schema.

4 `ltx-talk`: A new class for presentations

The `beamer` class is incompatible with the tagging code and due to the way it is implemented, it is not easily possible to change this. Joseph Wright therefore decided to write a new presentation class called `ltx-talk` with proper tagging support as a replacement for `beamer`.

The new class is already on CTAN and can be installed with the package manager of TeX Live and MiKTeX. It requires the LaTeX format 2025-11-01, so currently `pdflatex-dev` or (preferably) `lualatex-dev` must be used for the compilation.

Issues and feedback can be reported at the GitHub repository for the class [12].

5 Tagging of math with MathML

PDF 1.7 has a rather simplistic concept of the tagging of math: it views math equations as illustrative elements which should be made accessible similar to images by adding an alternative text to the `Formula` structure element. Such texts are problematic for many reasons; e.g., they are often much too verbose, and cannot be represented using braille math codes.

PDF 2.0 introduced the idea of using MathML instead — either as an embedded MathML file attached to the Formula structure element (an "associated file", AF), or by tagging the math with structure elements named after MathML elements.

Sadly, for a long time this was only a theoretical idea; there was no implementation. This has changed: we can now demonstrate the reading of math tagged with MathML. Frank Mittelbach gave some examples during his talk at the TUG 2025 conference in Trivandrum [5].

To make this possible various problems had to be solved.

5.1 Missing specification

PDF 2.0 introduced the idea of using MathML but basically all it says about it is in these three quotes:

> NOTE 1 MathML is the only domain-specific namespace defined in PDF 2.0.
>
> ISO 32000-2, 14.8.6.3

> When including mathematics structured as MathML 3.0, the math structure element type, as defined in MathML 3.0, shall be used to

> enclose the formula under the Formula structure element type. All MathML structure element types and their attributes shall have the MathML 3.0 namespace explicitly defined.
>
> ISO 32000-2, 14.8.6.3

> Example: MathML version of the equation embedded with an AFRelationship value Supplement, and associated using a structure element . . . .
>
> ISO 32000-2, 14.13.2

This doesn't give many details for either of the two methods. How to embed and associate files is known, but the format and content of the file is only vaguely described in an example as a "MathML version of the equation". PDF/UA-2 additionally requests "presentation" MathML.

For structure elements the gaps in the specification are even more pronounced: The "math structure element type" is *not* defined in MathML, and nowhere does it say how MathML, which is an XML language, should be used inside a PDF structure tree, which is explicitly not XML, and how consumers should pass the structure to a screen reader "as MathML".

As an example, consider the "less than" symbol <. In MathML it must be escaped as `&lt`, but in PDF it is used in the structure. Another problem is attributes. In MathML, attributes are in no namespace:

```
<mo fence="true" lspace="0" rspace="0"
    symmetric="true">(</mo>
```

In PDF 2.0, on the other hand, attributes *shall have the MathML 3.0 namespace defined.*

```
<<  /O/NSO/NS 13 0 R % namespace
   /fence(true)/lspace(0)
   /rspace(0)/symmetric(true) >>
<< /Type /StructElem
   /A  29 0 R
   /S /mo
   /NS 13 0 R  %namespace
   ...>>
```

There are no rules given to translate one version into the other.

Most prominently of all, the specification ignores, for both methods, the fact that equations can contain text, links and nested math, something that cannot be represented with pure MathML.

Not all these problems are resolved yet, but together with the tool `show-pdf-tags`, a document has been written by David Carlisle that describes how to interpret a PDF structure tree as XML [1]. A document specifying the rules for nested text and math is under development.

5.2 Various players

Tagging with MathML requires coordinated support from various players:

- The *producer* must add MathML.
- The *viewer* must pass the MathML to the Assistive technology (AT).
- The *Assistive technology* (the screen reader) must be able to handle MathML.

LuaLaTeX as producer can add MathML automatically (with the help of the `luamml` package) since last year. The Foxit PDF reader supports MathML in associated files (and soon as structure elements); the Adobe reader supports MathML structure elements. Finally, since the June 2025 release, the NVDA screen reader [8] handles the MathML with the help of the included MathCAT [10] library.

This means that, since June, an end-to-end toolchain exists to *demonstrate* how PDFs containing math tagged with MathML can be produced and read (or turned into braille). While this is a huge step forward, support is still rather in its infancy: for one, NVDA is a Windows-only application; for another, it is not yet possible to create PDFs that can be read properly with both readers — you have to choose between associated files (Foxit), or structure elements (Adobe). (This problem will hopefully be resolved soon.)

Also problematic are the accessibility checkers: Many tools only know PDF/UA-1, do not see the MathML, complain if there is no alternative text and incorrectly flag the PDF as invalid. This makes it difficult for users to adopt the new, improved math tagging.

5.3 LaTeX source example

To create a file with MathML embedded as an associated file, it is enough to use LuaLaTeX with the package `unicode-math`, and activate tagging with `\DocumentMetadata`:

```
\DocumentMetadata{tagging=on}
\documentclass{article}
\usepackage{unicode-math}
\begin{document}
 \begin{align}
 ...
 \end{align}
\end{document}
```

To get a document with MathML structure elements instead, a setup key can be used:

```
\DocumentMetadata{
    tagging=on,
    tagging-setup={math/setup=mathml-SE}}
```

5.4 Future plans: Intents

Mathematical notation can be ambiguous. For example, (x, y) could be the coordinate of a point or it could be the open interval from x to y. $|x|$ could be an absolute value, or a norm.

The MathML attribute `intent` is meant to allow authors to express how they would like their notations read in such cases [11]. For example, to denote $|x|$ the following intent could be used:

```
<mrow intent="absolute-value($x)">
 <mo></mo>
 <mi arg="x">x</mi>
 <mo></mo>
</mrow>
```

Such intents can also be added as attributes in the PDF. LaTeX already inserts intents in some places automatically, e.g., in alignments to enhance the reading systems of equations and their labels:

```
<mtable class="align" intent=":system-of-
    equations">
...
<mtd intent=":equation-label">
```

The number of such intents will increase and commands to allow package authors to enhance definitions with such intents will be available in the next LaTeX release.

References

[1] D. Carlisle. Interpreting a PDF Structure Tree as XML, 2025. github.com/latex3/tagging-project/discussions/789

[2] U. Fischer. On the road to Tagged PDF: About StructElem, marked content, PDF/A and squeezed Bärs. *TUGboat* 42(2):170–173, 2021. doi.org/10.47397/tb/42-2/tb131fischer-tagpdf

[3] U. Fischer, F. Mittelbach. Automated tagging of LaTeX documents — what is possible today, in 2023? *TUGboat* 44(2):262–266, 2023. doi.org/10.47397/tb/44-2/tb137fischer-tagging23

[4] LaTeX Project Team. `show-pdf-tags` — Extract PDF tags from tagged PDF files, 2025. ctan.org/pkg/show-pdf-tags

[5] F. Mittelbach. How accessible are PDFs? — and how accessible can they be?, July 2025. Video of the talk given at the TUG 2025 conference, Trivandrum, India. youtube.com/watch?v=Eu_qM53tInw&t=5900s

[6] F. Mittelbach, U. Fischer. The LaTeX Tagged PDF project — a status and progress report. *TUGboat* 43(3):268–272, 2022. doi.org/10.47397/tb/43-3/tb135mitt-tagged

[7] F. Mittelbach, U. Fischer. Enhancing LaTeX to automatically produce tagged and accessible PDF. *TUGboat* 45(1):52–59, 2024. doi.org/10.47397/tb/45-1/tb139mitt-deims24

[8] NV Access. nvaccess.org

[9] C. Rowley, U. Fischer, F. Mittelbach. Accessibility in the LaTeX kernel — experiments in Tagged PDF. *TUGboat* 40(2):157–158, 2019. tug.org/TUGboat/tb40-2/tb125rowley-tagpdf.pdf

[10] N. Soiffer. MathCAT: Math Capable Assistive Technology. nsoiffer.github.io/MathCAT

[11] W3C. Mathematical Markup Language (MathML) version 4.0 — annotating MathML: intent, 2025. w3c.github.io/mathml/#mixing_intent

[12] J. Wright. *The ltx-talk package*. A class for typesetting presentations. github.com/josephwright/ltx-talk

◇ Ulrike Fischer
LaTeX Project, Bonn, Germany
fischer (at) troubleshooting-tex dot de
https://www.latex-project.org
ORCID 0009-0009-1456-9592

◇ Frank Mittelbach
LaTeX Project, Mainz, Germany
frank.mittelbach (at) latex-project dot org
https://www.latex-project.org
ORCID 0000-0001-6318-1230