Automated tagging of LaTeX documents — what is possible today, in 2023?

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 TUG Conference 2020* [9]. This short report describes the progress and status of this multi-year project achieved with the LaTeX summer release 2023.

Contents

1 Introduction 262

2 Goals of the Tagged PDF project 263

3 Some problems we faced 263
- 3.1 Tags are not visible . . . 263
- 3.2 Missing PDF 2.0 support . . . 263
- 3.3 Parent-Child rules . . . 263

4 What is possible today: Tagging of "Leslie Lamport Documents" 263
- 4.1 Links . . . 264
- 4.2 Paragraphs . . . 264
- 4.3 Footnotes . . . 264
- 4.4 Sectioning . . . 264
- 4.5 Tables of contents and similar lists . 264
- 4.6 Display environments and lists . . . 264
- 4.7 Citations and bibliographies . . . 265
- 4.8 Graphics . . . 265
- 4.9 Floats . . . 265
- 4.10 `minipage` and `\parbox` . . . 265
- 4.11 Text commands . . . 265
- 4.12 Math . . . 265
- 4.13 Firstaid . . . 265

5 What is missing 265

6 Summary 266

1 Introduction

A tagged PDF is a PDF with an additional semantic structure. The purpose of tagging and the technical background have been described in earlier articles [3, 4] and in the documentation of the `tagpdf` package [2] and won't be repeated here.

As a short motivation for how tagging improves the accessibility and the reuse of data, let's look at bank statements as an example. For a few months Ulrike gets them as tagged PDF[1] and can compare them with earlier untagged versions. The tagging of the PDF is quite basic: apart from a few tags around paragraphs it contains only a large table with three columns (the date, a multiline description and the amount). As shown in figure 1, the tagging adds `TR` (**T**able **R**ow) tags around rows and `TD` (**T**able **D**ata) around the cells.

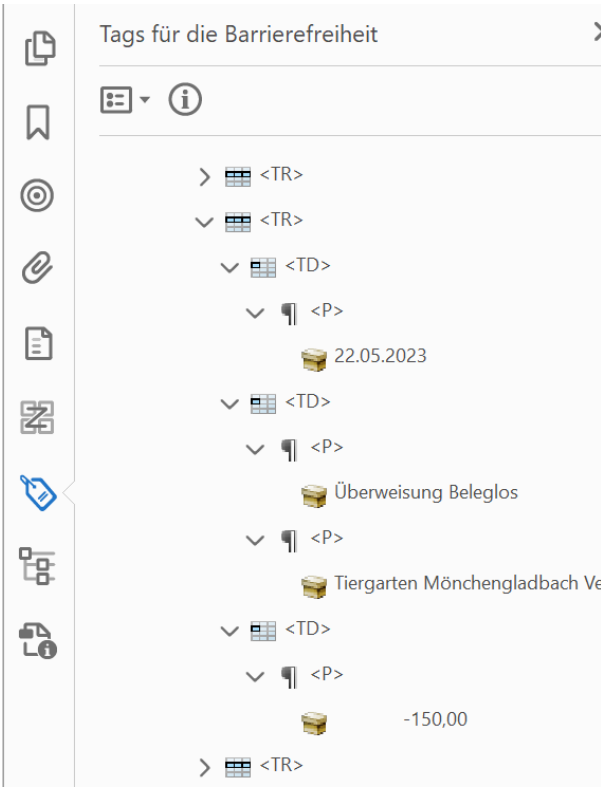

Figure 1: A tagged bank statement

Figure 2: Copy & paste of a tagged table into a spreadsheet

If such a bank statement is read with a screen reader, then with an *untagged* PDF the reading order is messy: the amount is read somewhere in the middle of the multiline description, text surrounding the table can leak into the reading, and there are no options to navigate in the table. With a *tagged* PDF the experience is quite different: the reading order is correct, the screen reader announces row and column numbers and it is possible to navigate by rows and cells. Copy & paste from a tagging aware PDF reader improves too: while with an *untagged* PDF the table content is dumped into one column of a spreadsheet, the content of a *tagged* PDF is correctly split into single cells, as can be seen in figure 2.

[1] Their tagging is not done with LaTeX.

doi.org/10.47397/tb/44-2/tb137fischer-tagging23

2 Goals of the Tagged PDF project

The example hopefully shows why tagging is generally important and the LaTeX team embarked on the task to support tagging. To address a common misunderstanding: The main goal of the Tagged PDF project is not to make tagging *possible* — this is already the case today: if you want to produce a tagged PDF with LaTeX you can do it. The main goal is to make tagging *automated* and *easy* to use. It should be possible for the average user to create a tagged PDF with only minimal changes to their sources.

3 Some problems we faced

The Tagged PDF project has to redefine many LaTeX commands but this is in some sense an expected task. Before we describe the current state of the project we want to discuss some problems, outside of the Project Team's control, that we identified in the last year.

3.1 Tags are not visible

The screenshot in Figure 1 was made with Acrobat Pro, and that costs money. Free viewers and browser plug-ins normally don't show tags even if (as with Adobe Reader) they use them to improve, for example, the copy & paste heuristic. This missing visibility makes it difficult to convince users that tags are important. It also makes development in LaTeX difficult as package maintainers can't easily check the tagging. Fortunately, the situation is starting to change and some free PDF viewers like PDF-XChange and PDFix Desktop Lite now can be used to check the tags.

3.2 Missing PDF 2.0 support

The ISO spec for PDF 2.0 [5] is more than 6 years old. PDF 2.0 added various new features that are crucial for tagging. For example, it extends the tag set with tags like `Title`, `Aside` and `FEnote`. It also adds two new features, both needed for the tagging of math: 1) it declares the MathML namespace, and 2) it supports attaching embedded files to structure objects; then they are called "associated files". Such associated files can contain the source code or the MathML representation. With PDF 2.0 it is also possible to link to a structure instead of to a location on a page. LaTeX is able to produce PDF 2.0 and can make use of the new features.

But sadly PDF 2.0 also has a problem: it is not well supported by PDF viewers and consumers. Even Acrobat Pro is, at the time of writing, not able to handle tag namespaces and doesn't pass associated files attached to a structure through its accessibility API. PAC3 crashes if a PDF 2.0 files is loaded.

Structure Type	Children		Parents	
	Occ.	Structure Type	Occ.	Structure Type
P	0..n	NonStruct	0..n	Document
	0..n	Private	0..n	DocumentFragment
	0..n	Note	‡	Part
	0..n	Code	‡	Div
	0..n	Sub	0..n	Art
	0..n	Lbl	0..n	Sect
	0..n	Em	0..n	TOCI
	0..n	Strong	0..n	Aside

Figure 3: Part of the Parent-Child rules for the `P` tag

We have here a clear chicken-and-egg problem: As there are only few tagged PDF 2.0 files in the wild, no viewer correctly processes them, and because no viewer handles tagged PDF 2.0 files properly there is no incentive for applications to produce tagged PDF 2.0 files. However, there is some hope for changes: The PDF 2.0 reference is now available at no cost; ISO 32005, which handles the combined PDF 1.7 and PDF 2.0 tag set, has been released [6] and PDF/UA-2 will be released shortly too.

3.3 Parent-Child rules

Structure elements defined by the PDF spec can't be used freely: The PDF format defines containment rules and describes which standard tag is allowed as a child or parent of other elements. If you declare your own tag names you have to also declare a *role mapping* to the standard tags and your tags then have to obey the containment rules of the standard tags they map to. In PDF 1.7 the rules were only described in a rather vague way. PDF 2.0 introduced a large matrix for the PDF 2.0 tags and the new ISO 32005 [6] extends this matrix to combine the PDF 1.7 and 2.0 tags. Figure 3 shows an excerpt of the rules.

Until last year the `tagpdf` package didn't check such rules, so authors had to look up the lists manually to decide if a tagging structure is valid or not. With more automated tagging, this became problematic: it became too easy to create a structure that violates the rules, so a data structure to store and automatically check the parent-child relation has been implemented. This isn't trivial, as the checks have to take role mapping and slight differences between PDF 1.7 and PDF 2.0 into account, and handle special tags (`Part`, `Div` and `NonStruct`) that don't have their own rules but inherit the containment rule from their parent in the structures.

4 What is possible today: Tagging of "Leslie Lamport Documents"

The next milestone in the project is to enable the automated tagging of what we call "Leslie Lamport

Documents". These are documents which use environments and commands as described in the LaTeX manual [7]. They are based on the standard classes `article`, `report` and `book` and load only a restricted set of compatible packages.

The code is being developed in various modules in the `latex-lab` bundle, and can be loaded in the `\DocumentMetadata` command with the `testphase` key. The supported values for the key are described in `documentmetadata-support-doc.pdf`. In LaTeX 2023-06-01, `phase-III` is the most comprehensive value that loads almost all available modules, and also activates tagging.

```
\DocumentMetadata{testphase=phase-III}
\documentclass{article}
```

Please refer to the documentation for details of the different modules.

The following sections describe what is tagged in this phase. For engines, pdfLaTeX or LuaLaTeX are recommended; other engines may work but are not much tested and do not offer support for real space characters and so give less accessible PDFs.

4.1 Links

The `hyperref` package is fully supported and links (both internal and external) are automatically tagged.

4.2 Paragraphs

Paragraphs are tagged with the help of the para hooks. Page breaks in paragraphs are handled automatically [8] through patches of the output routine.

This mechanism usually works well, but there can be problems if paragraphs are directly nested (the parent-child rules do not allow that), if the para hooks are not balanced (e.g., if a low-level `\vbox` is not properly ended with a `\par`), or if a package operates with low-level paragraph commands such as `\everypar` in an unexpected way.

4.3 Footnotes

The `latex-lab` module for footnotes is a full reimplementation of the footnote code and adds hooks and configuration points needed both for tagging and for the configuration of footnote layouts.

This code is not compatible with classes like `memoir` or `KOMA`, or with packages such as `bigfoot`, `manyfoot` or `footnote` that redefine internals related to footnotes. A replacement for the `footmisc` package has been written, so this package is already supported.

4.4 Sectioning

The changes to sectioning commands are currently rather lightweight; only a few internal commands like `\@startsection` and `\@sect` are redefined. The tagging not only surrounds the titles with a structure but also adds a `Sect` structure around heading and body of every section (see figure 4). Classes like `memoir` or `KOMA` and packages like `titlesec` are incompatible.

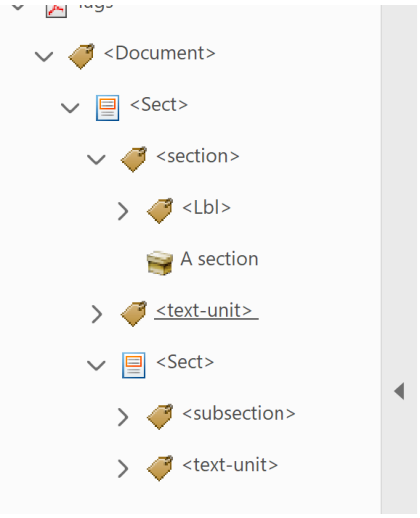

Figure 4: Tagging of sectioning commands

This implementation will not stay; the long-term plan is to reimplement the sectioning commands to offer more configuration options.

4.5 Tables of contents and similar lists

The implementation of the tagging support is rather lightweight and redefines, for example, `\@starttoc`, `\addcontentsline`, `\@dottedtocline`, `\l@part`, etc. Classes and packages like `memoir`, `KOMA`, `titletoc` and `etoc` are incompatible. Classes and documents that (re)define `\l@part`, `\l@chapter` or `\l@section` must adapt these definitions to add the hooks needed for tagging. An example of such a redefinition can be found in `ltugboat.cls`.

4.6 Display environments and lists

The list implementation in standard LaTeX serves a dual purpose: it implements true lists such as `itemize` and `enumerate`, and is also used as the basis for vertical blocks such as `center`, `quote`, `verbatim`, and for the theorem environments. These are all implemented as "trivial" lists with a single (hidden) item.

While this was convenient to get a consistent layout, it is not adequate when it comes to interpreting the structure of a document, because environments based on `trivlist` should not advertise themselves as being a "list" — after all, from a semantic point of view they aren't lists.

These environments have therefore been fully reimplemented and extended based on templates (from the `xtemplate` package). The code is currently incompatible with important packages like `enumitem`, `enumerate`, `fancyvrb`, `listings`, and also with theorem packages. Replacements or adaptations will be provided at a later stage.

4.7 Citations and bibliographies

Bibliographies are typically lists and so tagged by the list code of the previous section. Citations are tagged as `Reference` and point to the relevant bibliography entry. The `natbib` package is supported. `biblatex` is supported if `hyperref` is used.

4.8 Graphics

Graphics cannot be tagged automatically: For accessibility the author has to provide an alternative text. As an interface the new code adds[2] to `\includegraphics` and the `picture` environment an `alt` key that an author can use for this purpose. Ti*k*Z is not yet supported, but should follow soon.

```
\includegraphics
  [alt={This shows a duck}]{duck}

\begin{picture}
  [alt={This shows a duck too}](100,100)
   ...
\end{picture}
```

4.9 Floats

Floats (currently only `figure` and `table`) are tagged as "endfloats". That means the structures are moved to the end of the structure tree.[3] An example tree is shown in figure 5. The code also changes the placement of the `hyperref` anchor: it is now moved to the beginning of the float. If a float contains two captions, the author has to manually mark how the float should be split into two float structures.

Also missing, at the moment, is support for the `float` package, `[H]` ("here") floats (they will need a different structure), subcaptions and `\marginpar`.

4.10 `minipage` and `\parbox`

Initial support for `minipage` and `\parbox` has been added. But the content of such boxes can be arbitrarily complex, and it is unclear if the current code works as desired in all cases.

[2] The `alt` key was added to `\includegraphics` in LaTeX 2021-11-15 but does nothing if tagging is not used.

[3] Only the structures in the structure tree! This does not affect the typesetting!

Figure 5: Example structure tree with figure

4.11 Text commands

The LaTeX logo and `\emph` are tagged. To the logo an alternative text is added, and `\emph` is tagged as `EM`. `\textbf` is not tagged by default as it is too often used without a semantic meaning.

4.12 Math

There is also an early prototype for tagging formulas. It is not yet included in `phase-III` but can be loaded as an additional module:

```
\DocumentMetadata
  {testphase={phase-III,math}}
```

The code grabs the content of the math and reprocesses it. The code is compatible with `amsmath` (that package is explicitly loaded) but not compatible with packages defining their own math environments. Because of the missing PDF 2.0 support it is currently not possible to test if the provided tagging is in fact adequate.

The module can have side-effects on "fake math", for example urls from the `url` package, which internally use math mode for technical reasons, and are tagged as `Formula` at the time of this writing. Such places must be found and handled separately.

4.13 Firstaid

`latex-lab` contains also a small module `firstaid` which contains small temporary corrections to external packages to improve their compatibility with the tagging code. Like the math module it must be loaded explicitly:

```
\DocumentMetadata
  {testphase={phase-III,firstaid}}
```

5 What is missing

We are not yet at the point where we can state that all elements described in Leslie Lamport's manual on LaTeX are supported. One open point is `\savebox` and `\usebox` (and the TeX counterparts). Reusing

boxes still needs some research to decide how to handle boxes with and without internal structure.

Indexes, glossaries and similar lists are currently not properly handled either. They will not generate errors, but an index created with the standard `theindex` environment will tag every item and subitem and subsubitem as a paragraph of its own, so important semantic information about the nesting is lost.

And last, there is the structure with which this article started: tables. It is not very difficult to surround table rows and cells with `TR` and `TD` structures. But a properly tagged table should surround header cells with `TH` and complex tables need to reference the header from the data cells. The standard LaTeX input doesn't mark up the headers of the table and it is not yet clear how the syntax can be extended to detect headers reliably.

This was one of the three topics in the "tagging workshop" we ran a day before the 2023 TUG conference in Bonn, Germany. A report from this workshop is given elsewhere in this issue [1].

6 Summary

Starting with LaTeX 2023-06-01, quite a large set of standard documents can be tagged in an automated way — as an example this (*TUGboat*) article itself has been tagged with the code.

There is still much to do. We are grateful for everyone who tests the code and gives feedback. For this we have opened a dedicated Github repository for the tagging project where issues can be reported and discussions can be opened:

`https://github.com/latex3/tagging-project`

References

[1] D. Carlisle, U. Fischer, F. Mittelbach. Report on the LaTeX Tagged PDF Workshop, TUG 2023. *TUGboat* 44(2):267–269, 2023. `doi.org/10.47397/tb/44-2/carlisle-taggedpdfworkshop23`

[2] U. Fischer. *The `tagpdf` package*. `ctan.org/pkg/tagpdf`

[3] U. Fischer. Creating accessible pdfs with LaTeX. *TUGboat* 41(1):26–28, 2020. `tug.org/TUGboat/tb41-1/tb127fischer-accessible.pdf`

[4] 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`

[5] ISO. *ISO 32000-2:2020(en)*, 2nd ed., 2020. PDF 2.0. `iso.org/en/contents/data/standard/07/58/75839.html`

[6] ISO. *ISO/TS 32005:2023*, 1st ed., 2023. PDF 1.7 and 2.0 structure namespace inclusion. `iso.org/en/contents/data/standard/04/58/45878.html`

[7] L. Lamport. *LaTeX: A Document Preparation System: User's Guide and Reference Manual*. Addison Wesley, 2nd ed., 1994.

[8] F. Mittelbach. Taming the beast — advances in paragraph tagging with pdfTeX and XeTeX (2021): Automatic paragraph tagging with the pdfTeX and XeTeX engine[s] now possible. `latex-project.org/news/2022/09/06/TUG-online-talks-21-22/`

[9] F. Mittelbach, the LaTeX Project Team. Quo vadis LaTeX(3) Team — a look back and at the upcoming years. *TUGboat* 41(2):201–207, 2020. `tug.org/TUGboat/tb41-2/tb128mitt-quovadis.pdf`

◇ Ulrike Fischer
LaTeX project team
Bonn
Germany
`ulrike.fischer (at) latex-project.org`

◇ Frank Mittelbach
LaTeX project team
Mainz
Germany
`frank.mittelbach (at) latex-project.org`
ORCID 0000-0001-6318-1230