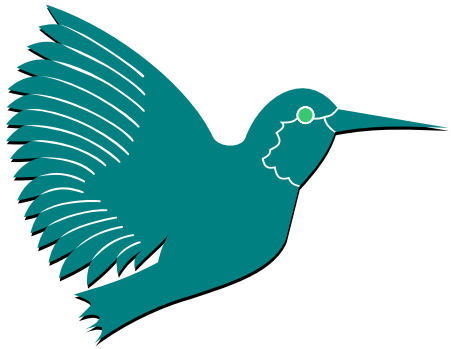
Accessibility in the LaTeX kernel — experiments in tagged PDF

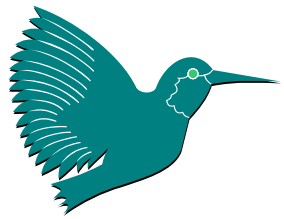
Chris Rowley Ulrike Fischer
LaTeX3 Team

August 2019, TUG Meeting — Palo Alto, USA



Outline of talk

- Accessibility of PDF
- "tagging" pdf: micro-introduction to the internals of a pdf file
- Experimental package: *tagpdf*
- Current status
- Future work



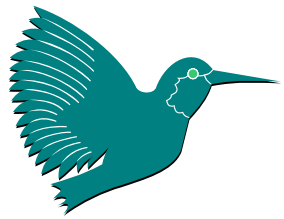
Accessible pdf

Standards

- PDF 1.7: equivalent to an ISO standard
- The PDF/UA: restrictions for Accessibility
- PDF 2.0: an emerging standard

PDF/UA mandates:

- "Tagged pdf", with two major components:
  - A Structure Hierarchy . . . like LaTeX
    (or Context, or HTML, or even MS-Word if used correctly)
  - The text/content streams (from classic PDF):
    "formatted content" plus, for tagged pdf,
    . . . "marked content" sections
  - The links:
    from the Structure Elements in the Structure Hierarchy
    to the relevant bits of "marked content"



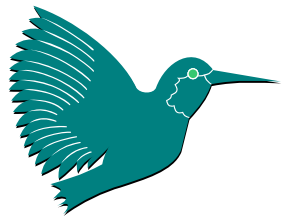
More important stuff: not in this talk

PDF/UA standard also requires, for example:

- Unicode mapping: Font slots $\mapsto$ Unicode slot
- document metadata in pdf file
- math, images, etc: "alt-text"
- specifies treatment of links (in *tagpdf* package)
- many design- and code-level "implementation details"
- detailed requirements of pdf,
  — such as an explicit char 32 (space) to delimit words

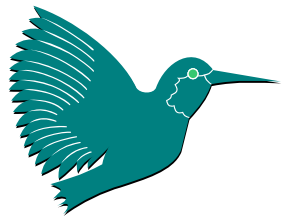
Many other issues, for example:

- reflowing (very important for small screens)
- Math . . . Nothing in this talk! << SMILEY >>



Structure of a **pdf file**

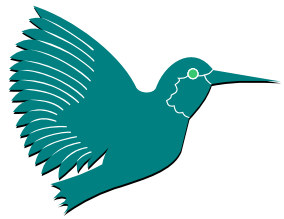
- Everything is . . . *PDF Objects* !
- Examples of important objects:
  - dictionaries: key-value (property) lists
  - text stream (within each page)
  - structure tree: nodes (NodeObj) are either
    - root node
    - structure element (node of tree):
      dictionary object containing refs to other nodes, parent
    - leaf nodes contain references, where each ref is to:
      — a page
      — a "marked part" of its text stream



Tagging code – add Marked Text within text stream of page

For Reference Only!

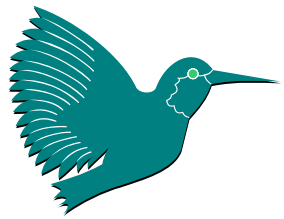
```
stream

BT
/F17 14.3462 Tf 124.802 706.129 Td [(0.1)-1100(Section)]TJ




/F15 9.9626 Tf 0 -21.819 Td [(hallo)]TJ




169.365 -593.872 Td [(1)]TJ
ET
```



Tagging code – add Marked Text within text stream of page

For Reference Only!

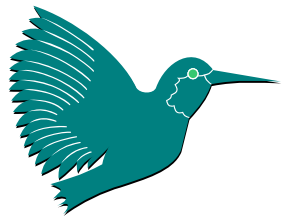
```
stream
/H <<MCID 0>> BDC
BT
/F17 14.3462 Tf 124.802 706.129 Td [(0.1)-1100(Section)]TJ
ET
EMC
/P <<MCID 1>> BDC
BT
/F15 9.9626 Tf 0 -21.819 Td [(hallo)]TJ
ET
EMC
/Artifact <</Type /Pagination>> BDC
BT
169.365 -593.872 Td [(1)]TJ
ET
EMC
```



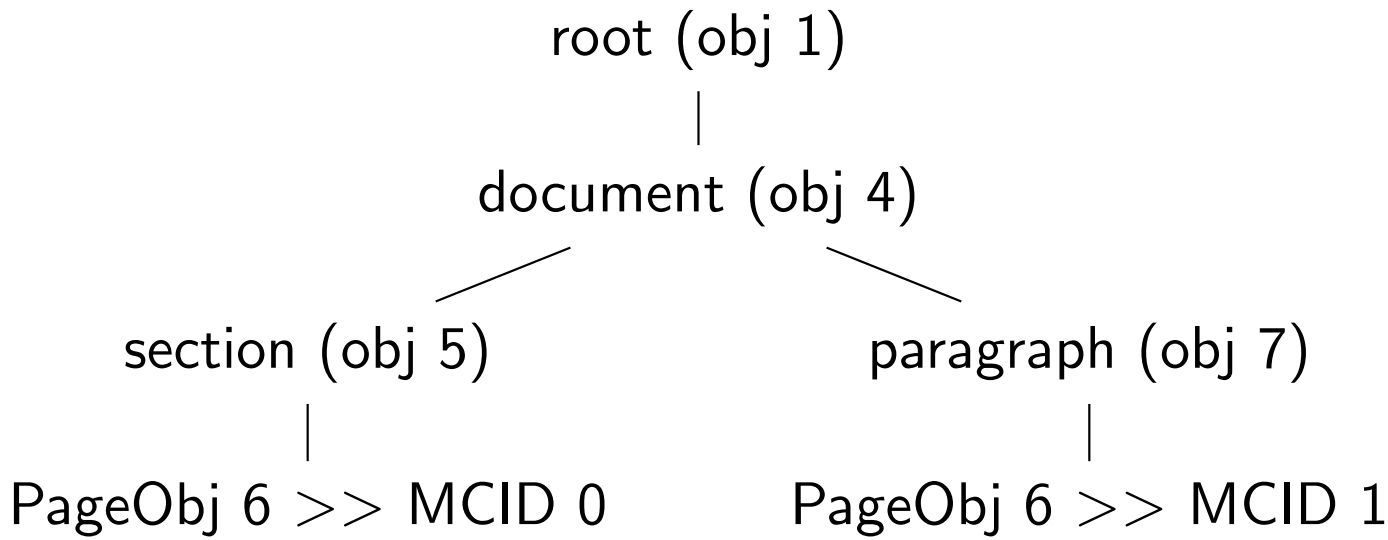
Tagging code – add structure objects with references to Page Objects and Marked Text

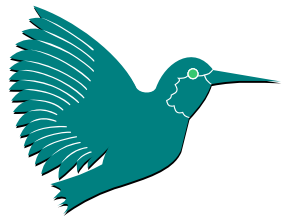
For Reference Only!

```
1 0 obj
<< /Type /StructTreeRoot /K 4 0 R
   /ParentTree 2 0 R /RoleMap 3 0 R>>
endobj
4 0 obj
<< /Type /StructElem /S /Document /P 1 0 R
   /K [5 0 R 7 0 R]>>
endobj
5 0 obj
<< /Type /StructElem /S /H1 /P 4 0 R
   /K <</Type /MCR /Pg 6 0 R /MCID 0>>>>
endobj
7 0 obj
<< /Type /StructElem /S /P /P 4 0 R
   /K <</Type /MCR /Pg 6 0 R /MCID 1>>>>
endobj
```



Tagging – structure tree nodes with refs

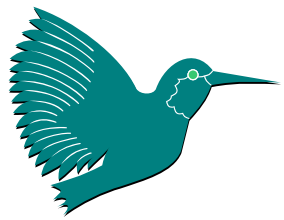




Current work: package

Experimental package: *tagpdf*

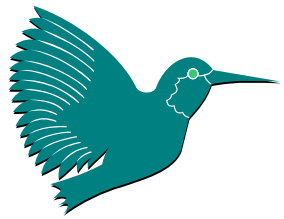
- uses existing "hooks" etc.
- contains code basis for some "kernel stuff"
- Experimental !!
  . . . So it needs experimenters:
  - author users:
    what is really needed in an accessible doc?
  - package maintainers/writers:
    what is needed to make conversion of a package as easy as possible?
- Purposes:
  - allow experiments to identify problems of tagging and the other accessibility requirements
  - experiment with code basis for expl3 / latex kernel



Current status: package

Features:

- Provides low-level mark-up commands to:
  - add structure element nodes to the structure tree
  - add "marked content" tags to the content stream
  - add to the structure tree nodes all the pointers to the marked content associated to that node
- Sets up links as required
- Supports the input of document metatdata

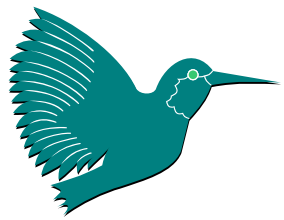


Package Features (Contd)

- Usage: very good documentation in `tagpdf.pdf`, Including:
  - lists of known problems and future work;
  - descriptions of how pdf works
  - How to ask questions, via Github!
  - Feedback, lots please!
- Engines status:

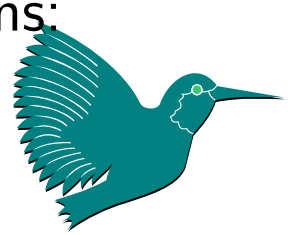
  lualatex Works okay:
  the documentation in `tagpdf.pdf` has been tagged with it and Ulrike writes that
  'the result is not perfect but it passes validation'

  pdflatex Currently, manual intervention is needed at page breaks



Into the kernel: automation support

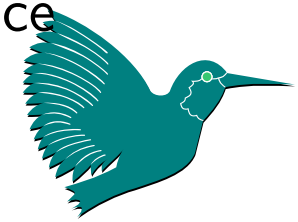
- In the "dev" branch only!!
- Current work needs "hooks", "configuration points":
  - all "mark-up" needs them: sections, section-heads, lists/items, toc entries, +++ !!
    — in kernel + in all packages
  - "internal processes", such as tabular or page-makeup, also need them
  - Use Author's "LaTeX document tags":
    needs rolemap, parenttree, etc ...
  - methods for adding tagging information when declaring and defining environments
    (and "structure commands")
- Automate insertion of "marked content" into text streams:
  - identification problems;
  - line- and page-break problems.



Future work

Still working to find more/better solutions, in areas such as:

- how to tag "paragraphs" and similar structures,
  and elements "within paragraphs"
- remove the need for two-passes when using current PDFtex engine (use TEX's marks?)
- compatibility with existing packages that extend the possibilities
  to produce pdf material
- document core data (language, pdf version, pdf standard, . . .)
  must be declared in a well defined place, and very early:
  — Before the "documentclass line" in a well defined place



The End

- Summary
- Call for ideas, experimenters for package
- Collaboration with Adobe
- Thanks to TUG and DANTE,
  . . . and to everyone for listening!