A document that says what it is

Everything on this page is in a structure tree, and nothing here asked for one. PdfDocumentRenderer.TagContent defaults to true, so PinataLayout records what it is drawing as it draws it: this paragraph is a /P, the line above it is an /H1, and the table further down is a /Table whose first row is headers.

The tree is what a screen reader, a reflowing viewer and a text extractor all read instead of guessing from coordinates. Without one, a page is a bag of glyphs at positions, and the order they were painted in is the only order there is - which is drawing order, not reading order, and on a two-column page those are different.

What the renderer maps

A section becomes a /Sect and a paragraph a /P, with /H1 to /H6 taken from Format.OutlineLevel. A run of list paragraphs becomes one /L of /LI, each with its bullet as the /Lbl. A hyperlink becomes a /Link that also carries a description on the annotation. Headers, footers, borders and shading become artifacts, which is the tree's way of saying "this is decoration, skip it".

A table that can be navigated

A sighted reader finds the meaning of a cell by looking up its column. A reader hearing the document has to be told, which is what /TH and its /Scope are for: a header cell announced before the value under it turns a grid of numbers back into sentences.

Region	Revenue (GBP thousand)	Headcount
North	1,240	38
Midlands	980	31
South West	1,505	44

Row.HeadingFormat is the whole of it. It was already there to repeat the row over a page break, and it turns out to be exactly the header-versus-data distinction the tree needs - which is lucky, because that association is otherwise the hardest part of tagging a table.

A picture, described or dismissed

There are two honest things to do with an image and no third. Describe it, and it is a /Figure with /Alt. Say it is decoration, and it is an artifact a reader skips. What conforms to nothing is a /Figure with nothing to say, and that is the one thing a document produces by accident.

The image above carries AlternativeText and is therefore a described /Figure. Set it to nothing and the same call draws the same picture as an artifact - still visible, and honestly marked as saying nothing.

Claiming PDF/UA-1

Tagging a document and conforming to PDF/UA are not the same thing, and the gap is mostly rules that are cheap to check and easy to break. Setting Options.UAConformance to PdfUA1 makes the claim, and the writer then walks the document before writing a byte and throws on the first rule it breaks.

The rules are listed on PdfUaValidator, and this link is itself one of them: a /Link with no description leaves a reader able to say only "link", so the annotation gets /Contents from the hyperlink's own text and is joined to the tree with a /StructParent.

Two things the writer settles rather than demands, because there is exactly one right answer and refusing over it would teach nobody anything: DisplayDocTitle is set, so a reader announces the title rather than the file name, and /Tabs is set to /S on every page, so the tab key walks the structure rather than the order the annotations happen to sit in.

What a successful save is not

It is not a validator's verdict. What can be settled by looking at the document is checked; what cannot is not. That no content sits outside the tree, that the reading order makes sense, that headings do not skip a level - none of those are here, and a page imported from an untagged document passes every rule and conforms to nothing. veraPDF has the last word.

The refusals, in their own words

Each line below is a real exception message, caught from a document built to break one rule and then asked to save. Nothing here is quoted from documentation - the text is whatever the library said when this demo was run.

Not tagged at all

```
A PDF/UA document has to be tagged, and nothing in this one is. Build a structure tree through
PdfDocument.Structure and XGraphics.BeginMarkedContent, or let PinataLayout do it — its renderer
tags what it draws unless told not to.
```

No title

```
A PDF/UA document has to have a title. Set Info.Title — it is what a reader announces the
document as, so the file name standing in for it is the failure this rule exists to stop.
```

No declared language

```
A PDF/UA document has to declare its natural language, or a reader cannot choose a voice to read
it in. Set PdfDocument.Language to an RFC 3066 tag such as "en-GB".
```

A figure with nothing to say

```
A figure in the structure tree has no alternate text, which leaves it saying nothing at all to
the reader it was tagged for. Give it one, or make it an artifact — a decorative image honestly
marked as decoration conforms, and an undescribed figure does not. From PinataLayout, set
Image.AlternativeText.
```