

Evaluating the Acrobat PDF accessibility checker

8 November, 2010

Contents

Summary	3
Problem 1: “false positives”	3
Problem 2: not a reliable measure of accessibility	3
Problem 3: it’s not the best automated checker	4
Method for evaluating an automated checker	4
Reading order	4
Alt text	6
Heading structure	6
Table of contents	7
Links	7
Lists	7
Tables	8
And so on	8
Assessing automated checkers’ performance	8
The relative benefits of manual and automated testing	9
Manual test method	9
Conclusion one	10
Conclusion two	10

Summary

Acrobat Professional comes with an inbuilt, automated PDF accessibility checking tool. However, for the most part, effective checking of a PDF for accessibility requires human knowledge, experience and judgment. Automated testing does have its place, but it should be used with caution. Over-reliance on an automated checker can even be counter-productive, for a number of reasons:

Problem 1: “false positives”

Some organisations stipulate that PDFs **must** pass the Acrobat “accessibility full check” before they can be published online. But automated checkers (the Acrobat one and others) tend to produce many “false positives” – things identified as problems that just aren’t. If your organisation’s policy states that your PDFs must pass the checker, you are liable to spend a great deal of time tracing and “fixing” non-existent problems – time that could have been spent fixing real problems. This is not a hypothetical situation but an everyday, real-world problem.

Problem 2: not a reliable measure of accessibility

Having made a document pass the checker, there is a danger that it might then be assumed to be accessible and “good to go”. **Stop, right there!**

Despite its name, the “accessibility full check” *cannot* tell you if a document is accessible or not. Nor can it give you any indication of *how* accessible it is. What it actually does is to measure a small subset of a document’s properties that may affect its accessibility. But, for the most part, these are not the most important things that you need to get right.

For this reason, an accessible document can fail the checker whilst an inaccessible one can pass. Much of this article will explain how and why this is so, and why manual checking is essential. And to reinforce the point, included below are links to a PDF version of the article which *fails* the checker despite its generally high standard of accessibility, and to another version that passes, despite its very poor standard of accessibility.

Problem 3: it's not the best automated checker

Having completed a manual inspection, it's fine to run an automated test to check that you haven't overlooked anything. But, if you do, there is now a better tool than the inbuilt Acrobat one. It's called [PAC](#) (PDF Accessibility Checker) and it's available free from *Access For All*. (More on PAC below.)

Method for evaluating an automated checker

In order to evaluate the performance of an automated checker it is necessary to understand what are the most important ingredients of an accessible PDF and then to examine the checker's ability to test against these. There will undoubtedly be variations from one document to another, but the following sets out what are probably the most important requirements for a typical (non-form) document.¹

Reading order

Certainly one of the most important requirements is a coherent reading order. One reading order problem that will go undetected by the checker is the position of footnotes. Without correction, all

¹ Forms are excluded from this analysis as they require a completely different approach to the way they are built, tested and used.

footnotes (unless the document originated in Word 2007 or later) will cause real problems for screen reader users. For example, in the inaccessible PDF version (originated in Word 2003), all three of the article's footnotes appear in sequence at the end of *this* paragraph (which is about reading order), despite the fact that the first footnote relates to the previous paragraph (on the method of evaluating the checker). The next two footnotes are on other topics altogether that come later in the article. Confused? You should be. The problem has, of course, been fixed in the accessible PDF version. (See [Making PDF footnotes accessible](#) for more on this.)

Another example, again in pre-Word 2007-originated PDFs, is that if text is wrapped around an image, the image and its alternate text (alt text) will appear for screen reader users at the bottom of the page, regardless of its actual position on the page. The image in the "Alt text" section below behaves like this in the inaccessible PDF version (it actually appears after the "Heading structure" section), but has been corrected in the accessible version. Again, this common problem will not be detected by the accessibility checker.

Similar problems are common in PDFs originated in InDesign because, without intervention, the reading order in the PDF's reflow view ² will be in the order that the content was *created* on the page, *not* the order it appears visually.

In sum, there are many common reading order problems that will not be picked up by an automated checker, but that will have a significant impact on the accessibility of a PDF.

² As used by those viewing on small screens or those who require screen magnification but without horizontal scrolling.

Alt text

Also a high priority will be the inclusion of *appropriate* alt text for non-decorative images. The image of a cat to the right (whoops! ³) has alt text of "Dog". This is obviously a complete failure from an accessibility point of view. More seriously though, it is common to see complex graphs and charts with completely inadequate alt text such as "Sales figures,



2009". In such a case, unless the data from which the chart is drawn is made available alongside it (probably in a table), its alt text should contain *all* of the information conveyed by the chart. But no automated checker can tell the difference between inadequate alt text such as "Sales figures, 2009" and the real thing. Only a human can do that. (Picture by Jaroslaw Poczarski, [some rights reserved](#).)

The checker *will* find images that have no alt text at all. However, such images are also easily detected in a manual inspection, and, in documents that have had *some* accessibility work done, it is far more common to find images with inadequate or inappropriate alt text rather than with no alt text at all. (Of course, there would be little point running the checker on a document that had had no accessibility work done.)

Heading structure

Also vital for accessibility in almost any document is an appropriate heading structure. The accessible PDF version of this article (which

³ There goes another failure (against WCAG 2.0, checkpoint 1.3.3) – again one that would evade the checker.

fails the checker) does have an appropriate heading structure. The inaccessible version (which passes) doesn't.

Table of contents

The inclusion of an active table of contents, that is, one containing links to the various sections of the document, is a high priority for most PDFs. Not having an active table of contents is a bit like building a website with no navigation – it's OK for a two-pager or so, but otherwise ... There is no automated check for the presence or absence of a table of contents.

Links

It is of course vital that links in any document are accessible and usable by all readers. The links in the accessible version of the PDF work with both a mouse and from the keyboard using a screen reader. In the inaccessible version the links are inaccessible to screen reader users. Such links are rare in Word originated documents (unless they span more than one line) but are the norm in InDesign-originated documents. The checker fails to identify any problem here too.

Lists

Also important in many documents will be the inclusion of well structured lists. Although incorrectly marked up lists will be identified by the checker, (visually) simulated "lists" such as:

- Item one
- Item two
- Item three

... are not picked up as errors. Such "lists" are, again, far more common than incorrectly marked up ones.

Tables

Similarly “tables” created using tabs, such as the one below, get past the checker too. Unfortunately, these are still commonly found, especially in typeset documents, but are, of course, accessibility disasters.

Day	Apples	Oranges	Pears..
Mon	20	23	25
Tues	14	15	21
Wed	16	19	22
Thu	12	11	27
Fri	22	22	18

And so on ...

It would perhaps be labouring the point to do so, but it would be easy to go on citing more and more common problems, none of which would get picked up. Examples? OK: the use of (poorly) “justified” text and double spaces between sentences (as in this paragraph – a real problem for some dyslexic people); the inclusion of a **serious colour contrast** problem; headings located too far away from the text that they relate to (such as the heading to this paragraph); incomprehensible URLs, acronyms and abbreviations, unmarked changes of language, and so on. (Note: the problems in this paragraph have not been fixed in any version of the article as it would make little sense to do so.)

Assessing automated checkers’ performance

The Acrobat checker fails to pick up a wide range of problems that will have a significant impact on the accessibility of a PDF. In addition, Acrobat doesn’t check for colour contrast problems, although, importantly, PAC does. PAC’s ability to assess colour contrast makes it

potentially a very useful tool, although at the moment it's still a little buggy and won't work with every document.

What the Acrobat checker does that can be useful is to check if a language has been specified for the document, if there are any character encoding problems, and, as mentioned above, if there are images with no alt text at all.

PAC does everything that the Acrobat checker does, and, in addition to measuring colour contrast, it checks for a document title, for bookmarks, for reading order problems (but of tags only) and for the presence of a heading structure (issues with this last category will be addressed in a separate forthcoming article).

The relative benefits of manual and automated testing

On the face of it, the benefits offered by PAC would seem to make it quite a useful tool. So why is automated testing of such limited use?

Manual test method

A manual test will identify virtually any problem that you are likely to encounter. Such an inspection will include checking a document's properties for title and language specification as well as checking for the presence of bookmarks. A typical test session will also include several passes of a document to check document structure and reading order in the TouchUp Reading Order panel, the order panel and the tag tree (in that order). It is also highly advisable to check with a screen reader to verify structure, reading order, tab order and general usability. Any character encoding problems will also show up straight away when you listen with a screen reader, as will poorly drafted alt text. Colour contrast should also be checked with one or

more of the many online or downloadable tools available. This, of course, is not an exhaustive list but it does cover the main points.

By comparison, as detailed throughout this article, automated checking is unable to identify many serious problems – problems that could render a document virtually unusable. With the possible exception of the PAC colour contrast test, which is potentially a great time saver, manual testing will always produce significantly better results.

Conclusion one

There is a case to be made for automated PDF accessibility testing, but **only** as a backup to manual checking, not as a substitute for it.

Conclusion two

Requiring as a matter of policy that your PDFs pass the Acrobat “accessibility full check” can actually be counter-productive, for three reasons:

- In most organisations there is usually limited time available for testing and editing PDFs for accessibility. Therefore, time spent “fixing” non-existent “problems” almost certainly means less time available for fixing *real* problems.
- There is a real danger that people might assume that because a document passes the automated checker it is accessible. This is not a safe assumption, not by any manner of means.
- The Acrobat tool is no longer the best available tool for automated PDF accessibility testing.

Of course, the situation might change with the imminent (at the time of writing) release Acrobat X.