

Creating high-quality PDF/A documents using LaTeX

mathstat.dal.ca

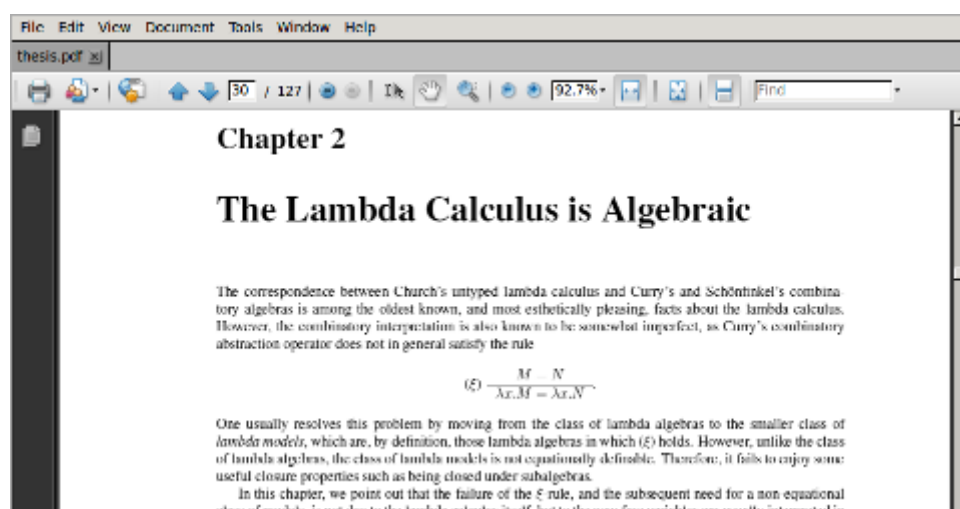
This document provides step-by-step instructions for generating valid PDF/A from LaTeX sources. (Written 2014/08/17. Revised 2016/11/13).

This page is still a draft. The files and instructions below are still subject to change, even in ways that aren't backward compatible. If you follow these instructions and find any mistakes, or run into problems you cannot solve, please feel free to email me at to selinger@mathstat.dal.ca. I will do my best to improve this document and add troubleshooting hints where necessary.

Here are my step-by-step instructions for converting an existing LaTeX document to PDF/A. These steps should work for most LaTeX documents.

To illustrate the steps, I will use my Ph.D. thesis as a running example. I wrote my thesis in 1997, long before I had ever heard of PDF, let alone PDF/A.

Initial version: [thesis.tex](#), [thesis.pdf](#)



1. Prerequisites:

you must have pdf_latex installed, as well as the hyperref and xmpincl packages. Most modern TeX distributions have these things preinstalled.

Option 1: If you want to generate PDF/A compliant documents "out of the box", you need at least pdf_lTeX 1.40.15, and preferably pdf_lTeX 1.40.16. As of this writing in August 2014, this version is not yet included in current software distributions such as Ubuntu, but it is available for installation from [Tex Live](http://tug.ctan.org/texlive). You can type `pdflatex --version` to determine your version.

Option 2: Even if you have an older version of pdf_lTeX (such as 1.40.14, which I am currently using), you can still generate PDF/A compliant documents if you also have access to Acrobat Pro. This requires two steps: first use `pdflatex` to generate an almost PDF/A compliant document, and then use Acrobat Pro to repair the few remaining errors.

2. Find your main source file.

I will assume, for the sake of argument, that your main LaTeX file is called `thesis.tex`. If it has a different name, adjust the following instructions accordingly.

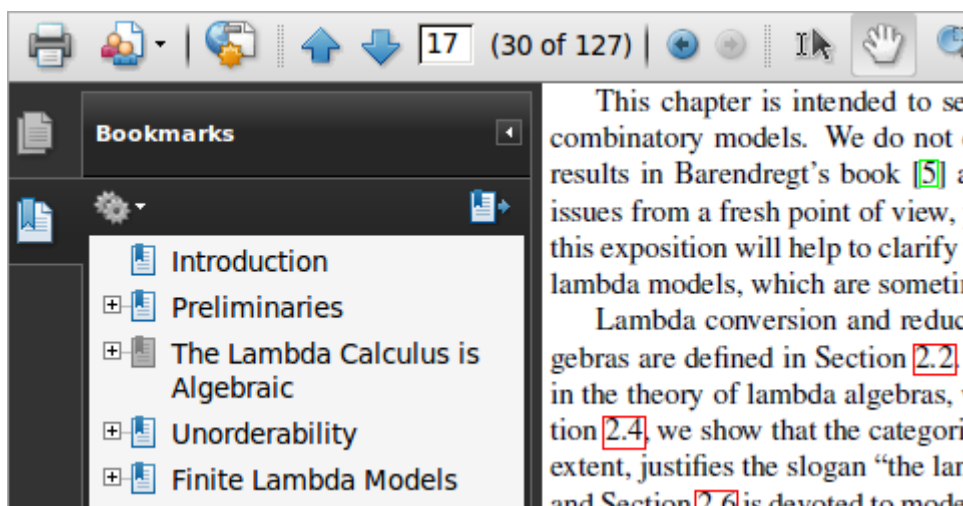
3. Activate hyperref.

Add `\usepackage{hyperref}` to the preamble of your LaTeX source. The preamble is the part of the file between `\documentclass` and `\begin{document}`. Do not specify any options to the hyperref package. Then try compiling it with

```
pdflatex thesis.tex
```

You may get some errors the first time around, but it should compile correctly the second time. If everything goes well, look at the resulting PDF file in Acrobat Reader. (You may have to select "File" → "Reload" if the file was already open before). You should notice a few changes: cross-references (theorems, equations, citations, page numbers, etc.) now show up as hyperlinks, and there should be a sidebar of PDF bookmarks. There are also other subtle changes; for example, Acrobat Reader now correctly understands that "page 17" is actually the 30th page of my thesis.

With hyperref: [thesis-step3.tex](#), [thesis-step3.pdf](#)



If anything goes wrong during this step, I can't offer much help. Try the documentation of the hyperref package, or try googling the error message.

4. Set up PDF/A files.

Make sure that you have at least version 1.5.8 of the pdfx package. This is probably preinstalled if your TeX distribution is from 2016 or newer. Please note that older versions of the pdfx package are buggy and do not work. If you don't have at least version 1.5.8, get it from CTAN:

- [The pdfx package at CTAN](#).

Copy the following file to the directory where your LaTeX sources are (your "source directory"). It may be helpful to right-click on the link and select "Save Link As".

- [sample.xmpdata](#)

5. Rename sample.xmpdata.

Rename the `sample.xmpdata` file as `thesis.xmpdata`, of course using the actual base name of your LaTeX source file instead of `thesis`.

6. Activate pdfx.

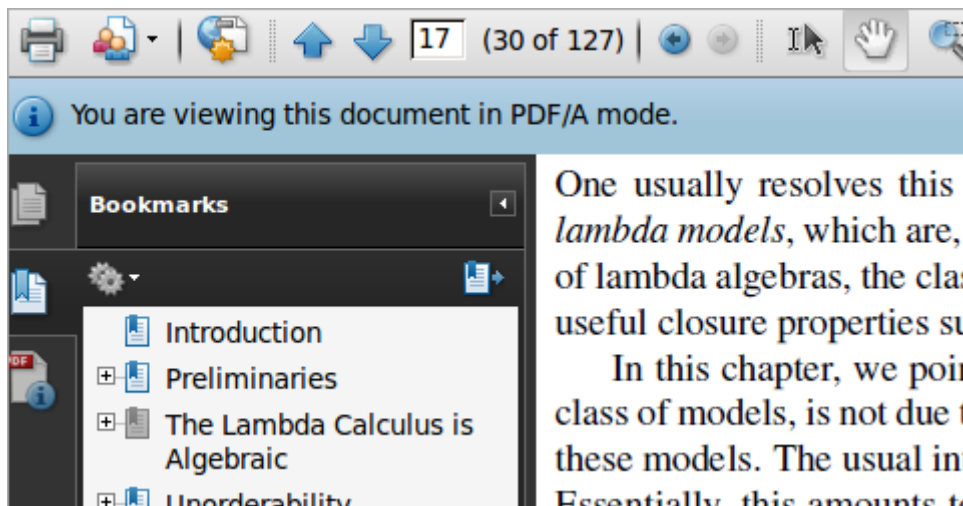
Add `\usepackage[a-1b]{pdfx}` to the preamble of your LaTeX source. Note that you must put this before the `\usepackage{hyperref}`. Then try compiling it with

```
pdflatex thesis.tex
```

In this step, I received one error, because my document defined the macro `\B`, which was internally used by the pdfx package. I changed `\newcommand{\B}{B}` to `\renewcommand{\B}{B}` and this fixed the problem.

Assuming there are no further errors, open the resulting PDF file in Acrobat Reader. (You may have to select "File" → "Reload" if the file was already open before). Acrobat Reader should now show a blue header announcing that "You are viewing this document in PDF/A mode." This does not yet mean that your file has been validated to be valid PDF/A. It only means that the file claims to be PDF/A.

With pdfx: [thesis-step6.tex](#), [thesis-step6.xmpdata](#), [thesis-step6.pdf](#)



7. Check for Unicode mapping.

At this point, you should be able to copy and paste some text from your PDF/A file and get reasonable output. Try copying and pasting some math formulas, text containing ligatures, etc. Of course, one cannot expect every complex math formula to be converted to sensible Unicode; however, you should see a marked improvement in simple mathematical symbols, Greek letters, and so on.

For example, copying and pasting the beginning of the Proof of Proposition 2.22 from page 27

Proof. 1. \Rightarrow 3.: Let A be weakly extensional and $A \models A = B$. Assume $FV(A, B) \subseteq \bar{x}$. By weak extensionality, $A \models \lambda^* \bar{x}. A = \lambda^* \bar{x}. B$. This is a closed equation, hence $A \models^{abs} \lambda^* \bar{x}. A = \lambda^* \bar{x}. B$, and finally $A \models^{abs} A = B$ by Lemma 2.15.

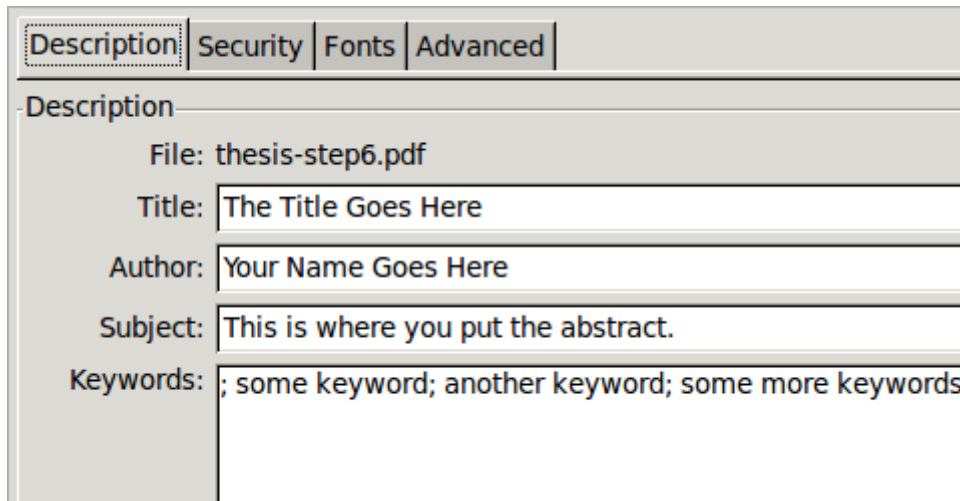
gives the following result:

Proof. 1. \Rightarrow 3.: Let A be weakly extensional and $A \models A = B$. Assume $FV(A, B) \subseteq \bar{x}$.

By weak extensionality, $A \models \lambda \bar{x}. A = \lambda \bar{x}. B$. This is a closed equation, hence $A \models_{\text{abs}} \lambda \bar{x}. A = \lambda \bar{x}. B$, and finally $A \models_{\text{abs}} A = B$ by Lemma 2.15.

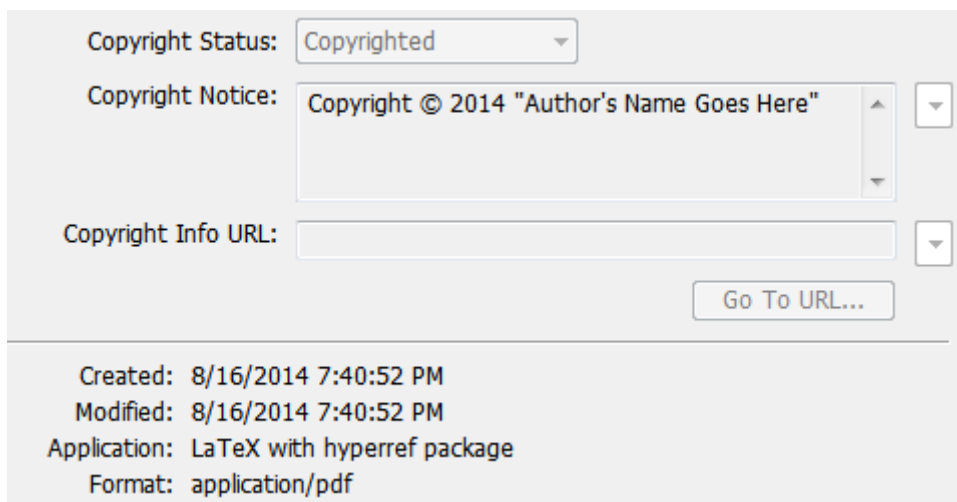
8. Check for metadata.

In Acrobat Reader, go to "File" \rightarrow "Properties". You should see a sample title, sample author, sample subject, and sample keywords.



The screenshot shows the 'Description' tab of the Acrobat Properties dialog box. The 'File' field is 'thesis-step6.pdf'. The 'Title' field contains 'The Title Goes Here'. The 'Author' field contains 'Your Name Goes Here'. The 'Subject' field contains 'This is where you put the abstract.'. The 'Keywords' field contains '; some keyword; another keyword; some more keywords'.

If you have Acrobat Pro, you can also click on "Additional Metadata", and you should also see a sample copyright notice.



The screenshot shows the 'Additional Metadata' dialog box. The 'Copyright Status' dropdown is set to 'Copyrighted'. The 'Copyright Notice' text area contains 'Copyright © 2014 "Author's Name Goes Here"'. The 'Copyright Info URL' text field is empty. There is a 'Go To URL...' button. At the bottom, the following information is displayed: Created: 8/16/2014 7:40:52 PM, Modified: 8/16/2014 7:40:52 PM, Application: LaTeX with hyperref package, Format: application/pdf.

9. Edit the metadata.

Now you should edit the file `thesis.xmpdata` to create the metadata that is appropriate for your document. You can delete any entry (such as `\Keywords` or `\Subject`) that you do not need, and edit the remaining ones to suit your purpose. You can use UTF-8 encoded Unicode in this file, in case you would like to include any foreign letters, math formulas, etc. See [Section 5](#), "Metadata format", for more details on what you can put here.

Then recompile your document, reopen the file in Acrobat, and check that your metadata shows up correctly. (Don't forget "File" \rightarrow "Reload" in case the file is still open from before).

With metadata: [thesis-step9.tex](#), [thesis-step9.xmpdata](#), [thesis-step9.pdf](#)

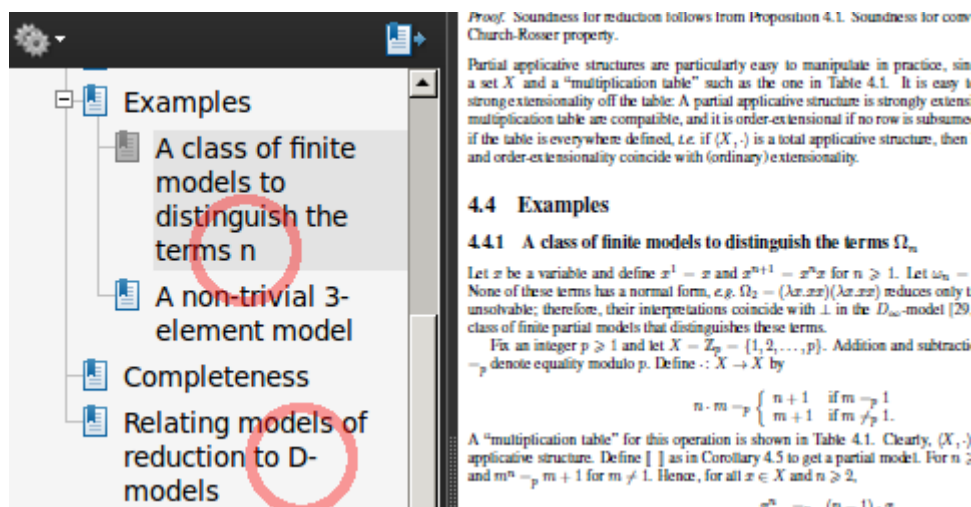
Description	Security	Fonts	Advanced
Description			
File:	thesis-step9.pdf		
Title:	Functionality, polymorphism, and concurrency: a mathem		
Author:	Peter Selinger		
Subject:	The search for mathematical models of computational ph		
Keywords:	; lambda calculus; unorderability; polymorphism; PL-categ models; communication processes; asynchronicity		

10. Fix the PDF bookmarks.

When you open your document in Acrobat Reader or Acrobat Pro, you should see a bookmarks sidebar. It should roughly correspond to the table of contents of your document. Check all of the entries. If any of the entries contain math formulas, they will probably look messed-up. For example, my thesis contains a section called "

A class of finite models to distinguish the terms Ω_n ", but it may show up in the PDF bookmarks as "

A class of finite models to distinguish the terms n ". Also, instead of "D ∞ -models" we may see "D-models".



More complicated math formulas, particularly those involving `\boldmath` or similar commands, can lead to even-stranger-looking garbage characters in the bookmarks. There are two options for fixing this:

- **Option 1: Using plain ASCII.**

We will specify an alternate plain ASCII version of each math formula in a section title, using the command `\texorpdfstring`. For example, replace

```
\section{A class of finite models to distinguish the terms  $\Omega_n$ }
```

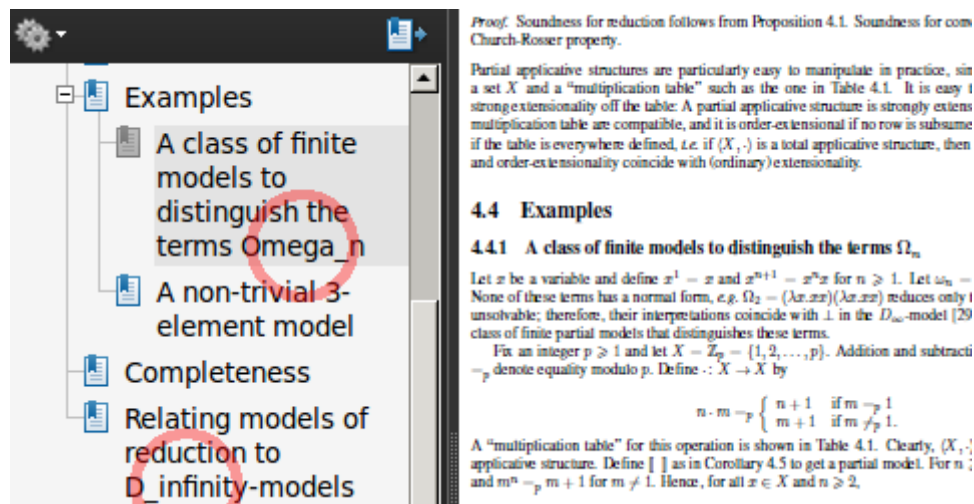
by

```
\section{A class of finite models to distinguish the terms
\texorpdfstring{$\Omega_n$}{Omega\_n}}
```

The `\texorpdfstring` command is very simple. It takes two arguments. The first one is what will be typeset in the document, and the second one is the ASCII equivalent that will go into the bookmarks section. Note that the ASCII you can use here is very basic: for example, you can't specify superscripts or subscripts, and if you want to use an underscore "_", you have to escape it as "_". Some other symbols, such as "^", can't be used at all.

Note that you have to run `pdflatex` at least twice for any changes to take effect. Here is what the output looks like:

ASCII bookmarks: [thesis-step10a.tex](#), [thesis-step10a.xmpdata](#), [thesis-step10a.pdf](#)



- **Option 2: Using Unicode.**

To use Unicode characters in the PDF bookmarks, place the following four commands in your LaTeX preamble:

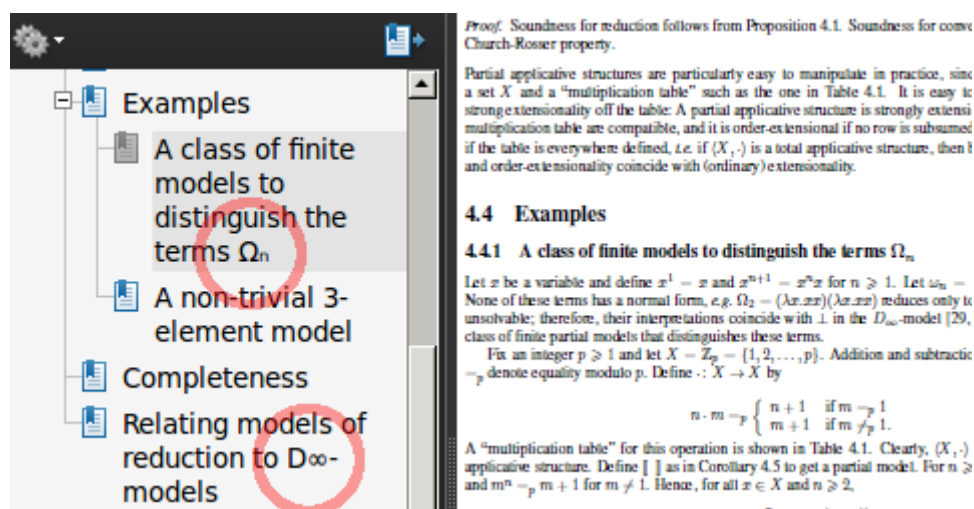
```
\usepackage{inputenc}
\hypersetup{pdfencoding=utf8}
\inputencoding{utf8}
\makeatother
```

Note that this will change the input encoding of your LaTeX source to UTF-8. It will also allow you to use Unicode characters in the second argument of `\texorpdfstring`. For example:

```
\section{A class of finite models to distinguish the terms
\texorpdfstring{$\Omega_n$}{Ωn}}
```

The result looks rather nice:

Unicode bookmarks: [thesis-step10b.tex](#), [thesis-step10b.xmpdata](#), [thesis-step10b.pdf](#)



Of course, here I have presupposed that you know how to enter Unicode characters in the editor that you use to write your LaTeX file. I usually do this by copying and pasting the character from some website. For example, if you google "[Unicode subscript n](#)", you'll find a [browser test page](#) for that character, from which you can then copy and paste.

If you don't want to enter actual Unicode characters in your LaTeX file, TeX also offers a way to escape them. This requires you to look up the UTF-8 encoding of each character, which you can [also find via Google](#) (scroll down to "UTF-8 (hex)"). For example, the UTF-8 encoding of "Omega" is `0xCE 0xA9`, and the UTF-8 encoding of "subscript n" is `0xE2 0x82 0x99`. You can then enter the characters in your LaTeX file like this:

```
\section{A class of finite models to distinguish the terms}
\texorpdfstring{$\Omega_n$}{^ce^a9^e2^82^99}}
```

An example file is thesis-step10c.tex; the output is identical to the above.

11. Verify PDF/A-1b compliance.

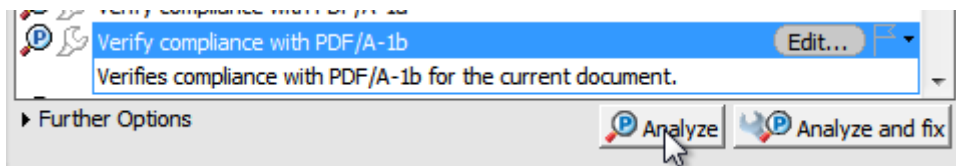
If your pdfTeX version is at least 1.40.15 (see Step 1: Prerequisites), then your generated document

should already comply with the PDF/A-1b standard at this point.

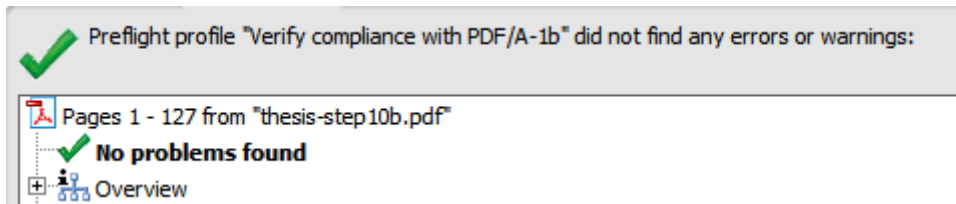
However, it is possible, for a variety of reasons, that you did everything correctly, and your document still doesn't comply. For example, PDF/A-1b does not permit transparency in images, so if you included any images that use transparency, your document may not be compliant. Or you may have included an image that uses the wrong color space. Some font that you used may have been slightly buggy. Or you may have used an older version of pdfTeX.

Don't worry. In Steps 11 and 12, we will use Acrobat Pro to verify that our document complies with the PDF/A-1b standard, or to fix it if necessary. If you don't have access to Acrobat Pro, you may also try one of the other tools mentioned in [Section 2](#) above.

To verify compliance in Acrobat Pro, go to "View" → "Tools" → "Print Production" → "Preflight" → "Verify compliance with PDF/A-1b" → "Analyze".



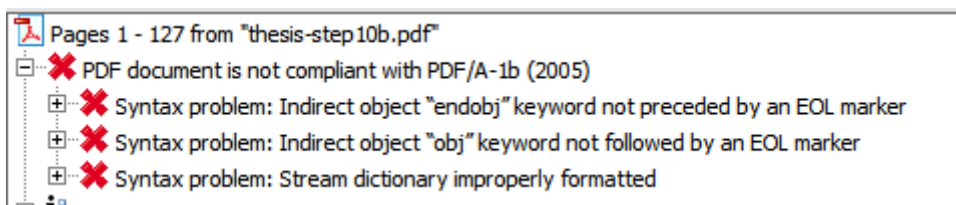
If your document passes the test, you get a report of "No problems found".



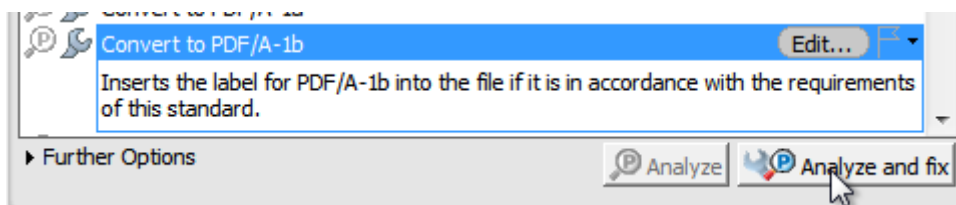
In this case, you are finished, and you can skip Step 12. However, if you get some errors here, go to Step 12 to fix them.

12. If necessary, convert to PDF/A-1b.

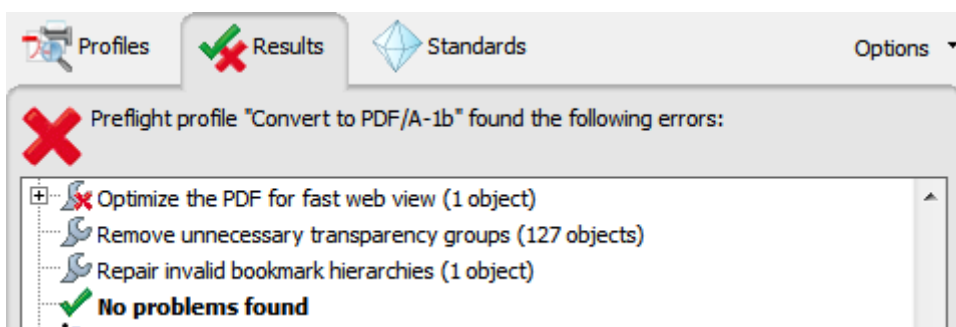
If your document fails to pass validation in Step 11 for some reason, then you will have gotten a Preflight report similar to this:



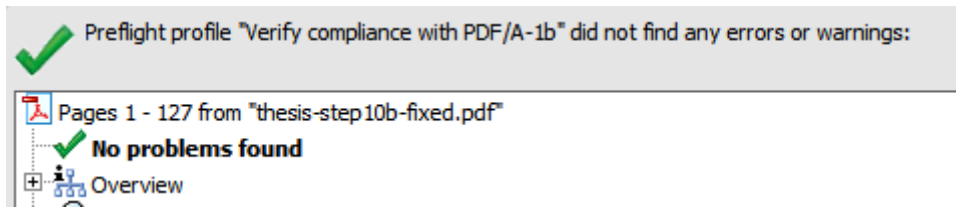
However, there is no cause for concern, as these errors are easily fixed. In Acrobat Pro, go to "View" → "Tools" → "Print Production" → "Preflight" → "Convert to PDF/A-1b" → "Analyze and fix".



You'll be asked for a filename for the fixed PDF/A file. I chose [thesis-step10b-fixed.pdf](#). If everything goes well, you'll receive a long report showing things that were fixed, and at the end, the message "No problems found".



At this point, you can optionally re-run PDF/A-1b validation on the new file, but there will be no further errors.



Congratulations. If all steps were successful, you should now have a high-quality PDF/A compliant document.