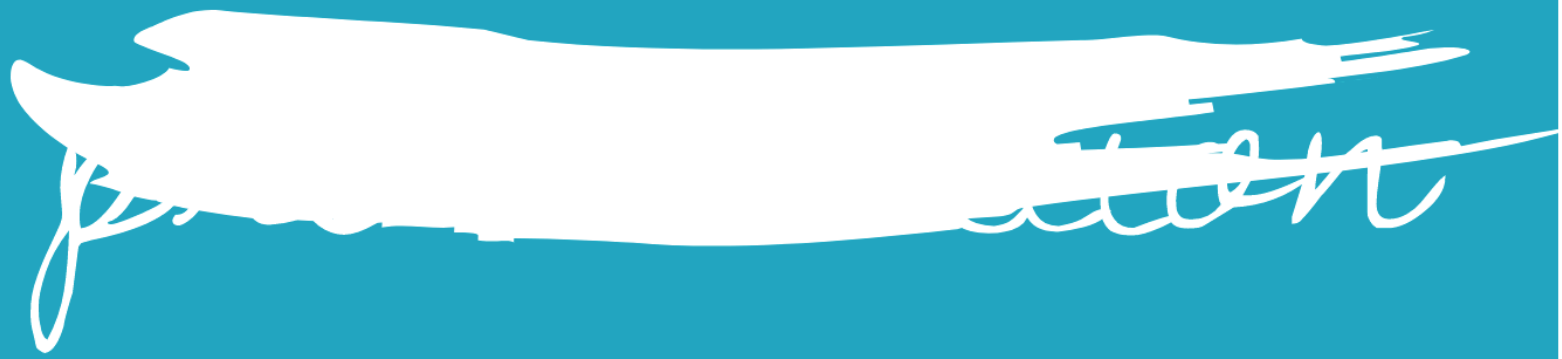
01
I
OPFCON - June 2020

Non-conforming PDF/A Conformance? How to Embed Customized PDF/A-Compliant XMP Metadata into PDFs



Elizabeth Kata - International Atomic Energy Agency
@SnarkivistWien

XMP in PDF/A

PDF/A has obtained wide acceptance as a suitable format for digital preservation. Metadata can be embedded in PDF/A using XMP schemas, several of which are defined (including Dublin Core and Adobe PDF). These schemas are somewhat limited, so when I had a very specific set of metadata I was interested in embedding, I felt like I needed something more. I wanted to write XMP metadata and ensure PDF/A compliance, so I turned to two tools: ExifTool and VeraPDF.

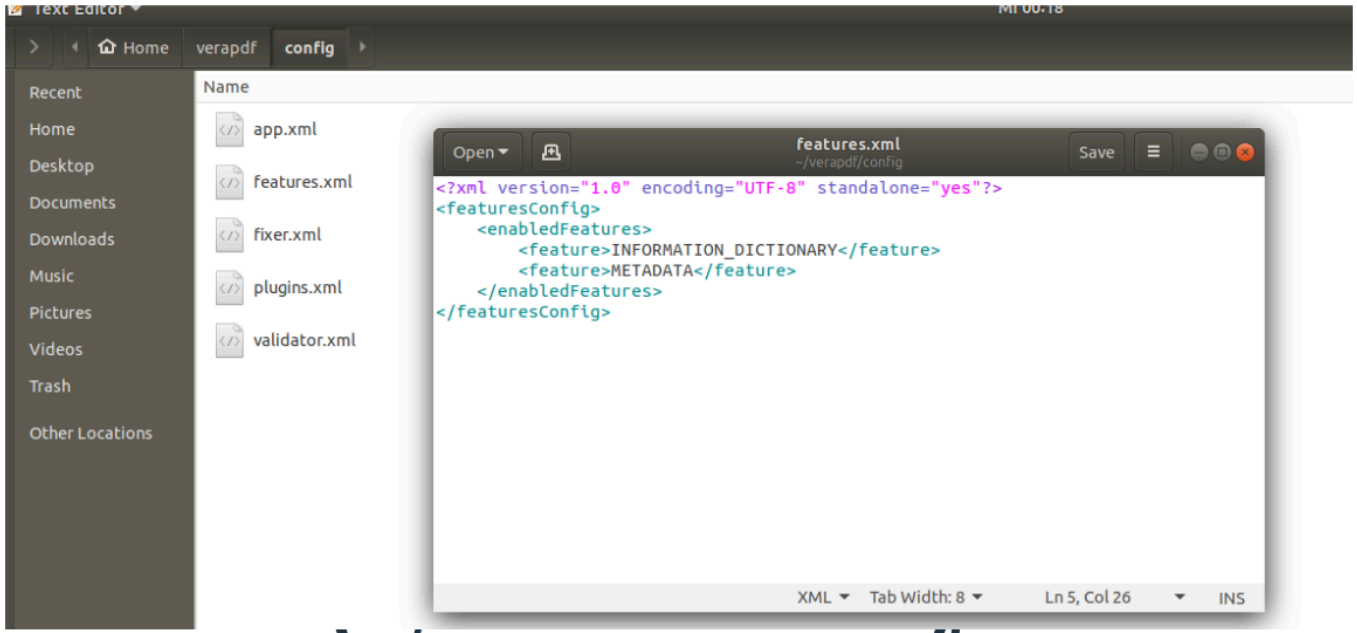
ExifTool
A powerful tool for reading and writing metadata. Chosen for the ability to create custom tags and bulk update metadata.

VeraPDF
A tool for PDF/A validation and feature extraction, with both CLI and GUI. I chose to use the CLI. I am also using PDFs from the VeraPDF corpus in this example.

use case

VeraPDF: Enable Metadata Extraction

When I started using VeraPDF for PDF/A validation, I did a bit of a deep dive and discovered there was a plug-in for metadata extraction output as METS. In learning how to enable it, I realized it was also possible to extract more metadata by enabling "METADATA" under the features.xml file in the config folder. I recommend doing this if you are using the extraction feature of VeraPDF.



Verapdf

XMP in PDF/A: Defining the Schema

If you choose to use one of the pre-defined XMP schemas, you simply need to identify the tag name. But if you want to use your own custom schema, you have to first define it within the XMP-pdfaExtension schema. In order for the custom schema to be compliant, you will need to define:

Schema (Name)
Give your schema a name. For ease of reading, I found it useful if it is not identical to your namespace.

NamespaceURI
Give your schema a URI. It does not have to be resolvable.

Namespace
Assign a namespace, which is also your prefix. If I assign the namespace liz, my tags will look like "XMP-liz:"

Properties
For each property, you will need to assign it a name, valueType, category and description. It is possible to create custom value types, but I will use pre-defined valueTypes here.

The Schema

ExifTool: Setting it up

This presentation assumes a basic knowledge of how to install and run ExifTool. www.exiftool.org provides excellent documentation. I will focus on how to set up the config file to be able to write PDF/A-compliant custom XMP Tags. (An example with more possibilities can be found at exiftool.org/config.html.) To achieve PDF/A compliance, you will need to add the PDF/A Extension schema to ExifTool, so you can write to it. For ease of use, you can copy this from my GitHub page: https://github.com/archivist-liz/exifTool-configs/blob/master/.ExifTool_config. The impotant thing is to save the configuration file as .ExifTool_config in your ExifTool directory or home directory.

(On Mac and some Windows systems this must be done via the command line since the GUI's may not allow filenames to begin with a dot. Use the "rename" command in Windows or "mv" on the Mac.)

ExifTool

ExifTool: Defining your schema

ExifTool is written in Perl. I don't know Perl, but it is pretty easy to edit based on the sample config file mentioned above. Here are a couple tips based on issues I encountered:

- Be consistent with your namespace/prefix. This needs to be added to the Main XMP table as well as be defined in a separate section.
- Unfortunately the valueTypes defined in XMP do not always have the same names as the values needed for the ExifTool config file. What is valueType "text" in XMP is "string." Double check the value you types you need and any constraints (such as date formating).
- Don't worry about making mistakes. You can restore the original files with the command: exiftool -restore_original /filepath/

The ExifTool Config File

```
# The %Image::ExifTool::UserDefined hash defines new tags to be added
# to existing tables.
%Image::ExifTool::UserDefined = (
    # new XMP namespaces (eg. xxx) must be added to the Main XMP table:
    'Image::ExifTool::XMP::Main' => {
            pdfaExtension => { # <-- must be the same as the NAMESPACE prefix
        SubDirectory => {
            TagTable => 'Image::ExifTool::UserDefined::pdfaExtension',
            # (see the definition of this table below)
        },
    },
    # add more user-defined XMP namespaces here...
            premis => { # <-- must be the same as the NAMESPACE prefix
        SubDirectory => {
            TagTable => 'Image::ExifTool::UserDefined::premis',
            # (see the definition of this table below)
        },
    },
    # add more user-defined XMP namespaces here...
            liz => { # <-- must be the same as the NAMESPACE prefix
        SubDirectory => {
            TagTable => 'Image::ExifTool::UserDefined::liz',
            # (see the definition of this table below)
        },
    },
);

# This is a basic example of the definition for a new XMP namespace.
# This table is referenced through a SubDirectory tag definition
# in the %Image::ExifTool::UserDefined definition above.
# The namespace prefix for these tags is 'xxx', which corresponds to
# an ExifTool family 1 group name of 'XMP-xxx'.
 %Image::ExifTool::UserDefined::pdfaExtension = (
    GROUPS => { 0 => 'XMP', 1 => 'XMP-pdfaExtension' },
    NAMESPACE => { 'pdfaExtension' => 'http://www.aiim.org/pdfa/ns/extension/' },
    WRITABLE => 'string', # (default to string-type tags)
    schemas => {
                List => 'Bag',
                Struct => {
                        NAMESPACE => {'pdfaSchema' => 'http://www.aiim.org/pdfa/ns/schema#'},
                        schema => {},
                        namespaceURI => {},
                        prefix => {},
                        property => {
                                List => 'Seq',
                                Struct => {
                                        NAMESPACE => {'pdfaProperty' => 'http://www.aiim.org/pdfa/ns/property#'},
                                                name => {},
                                                valueType => {},
                                                category => {},
                                                description => {},
                                        }
                                }
```

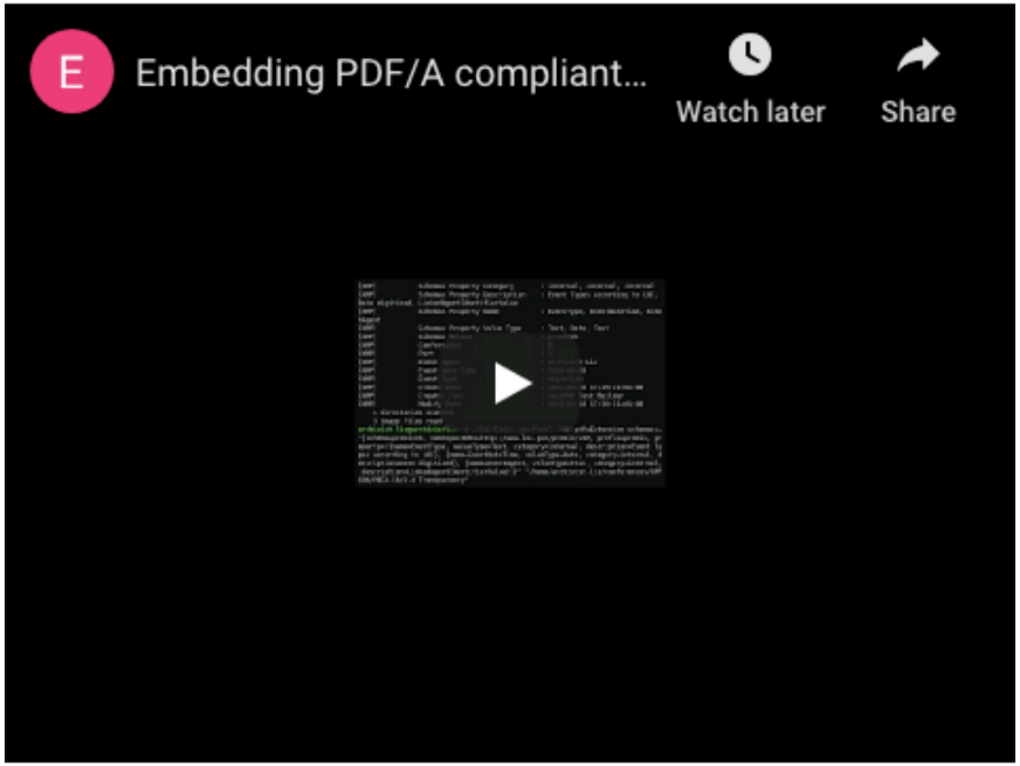
```
# This is a basic example of the definition for a new XMP namespace.
# This table is referenced through a SubDirectory tag definition
# in the %Image::ExifTool::UserDefined definition above.
# The namespace prefix for these tags is 'xxx', which corresponds to
# an ExifTool family 1 group name of 'XMP-xxx'.
 %Image::ExifTool::UserDefined::premis = (
     GROUPS => { 0 => 'XMP', 1 => 'XMP-premis' },
     NAMESPACE => { 'premis' => 'http://www.loc.gov/premis/v3#' },
     WRITABLE => 'string', # (default to string-type tags)
     EventType => { WRITABLE => 'string' },
     EventDateTime => { WRITABLE => 'date' },
     EventAgent => { WRITABLE => 'string' },
 );


 %Image::ExifTool::UserDefined::liz = (
     GROUPS => { 0 => 'XMP', 1 => 'XMP-liz' },
     NAMESPACE => { 'liz' => 'http://www.mywebsite.com/liz/v1#' },
     WRITABLE => 'string', # (default to string-type tags)
     GitHubRepo => { WRITABLE => 'url'},
     Message => { WRITABLE => 'string'},
 );
```

At the top, you will name your user-defined namespace(s).

The second block is part of the necessary addition of the pdfaExtension schema. Without this, you cannot embed the definition of your schema in a PDF/A compliant way.

Here I have defined two schemas, "premis" loosely based on a few elements of Premis v. 3. I also defined a schema named "liz."

Screencast of adding the custom XMP-premis schema to all the files in a directory.

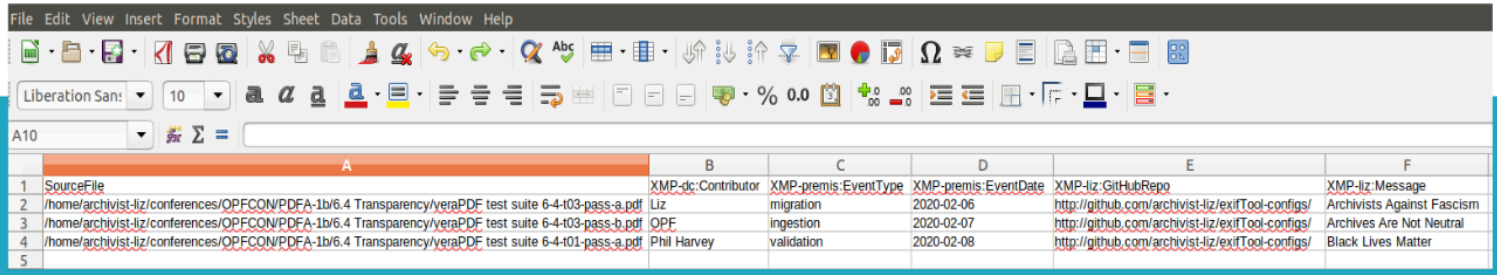


a test run

Bulk update via CSV

One of ExifTool's many excellent features is the ability to write metadata from CSV files. If you have a set of metadata you'd like embed, this is easily done. Just keep in mind:

- The first row must have your tags (written as ExifTool identifies them).
- The first column must be called SourceFile and identify which file the metadata applies to (including the file path).

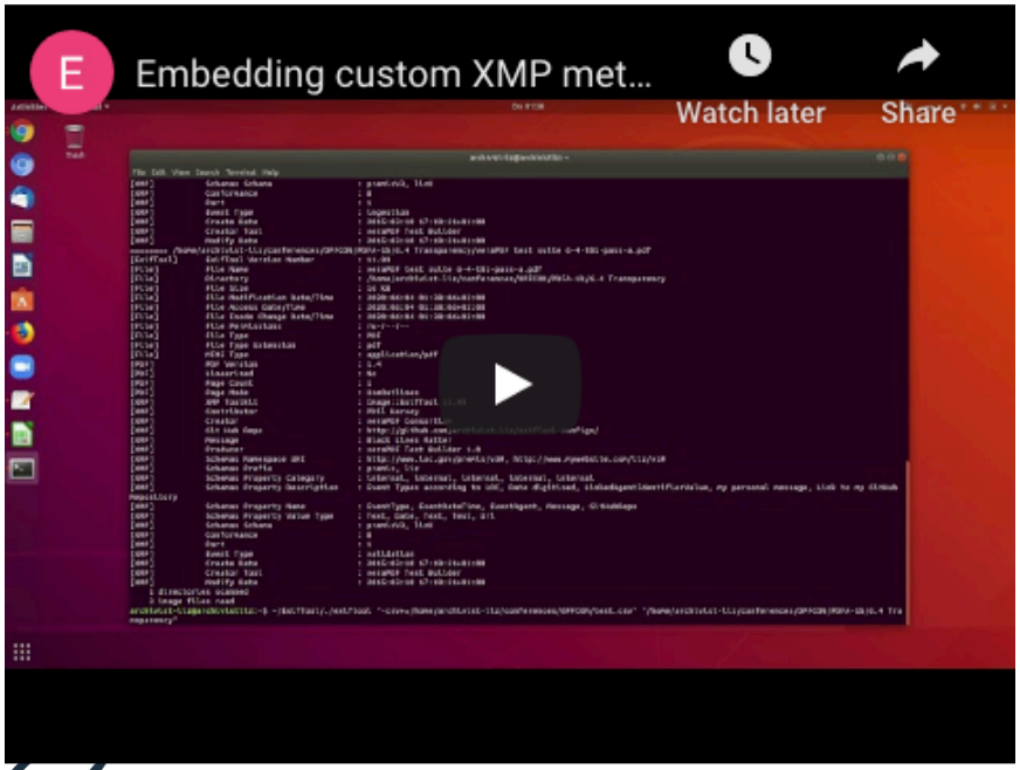


	A	B	C	D	E	F
1	SourceFile	XMP-dc:Contributor	XMP-premis:EventType	XMP-premis:EventDate	XMP-liz:GitHubRepo	XMP-liz:Message
2	/home/archivist-liz/conferences/OPFCON/PDFA-1b/6.4 Transparency/veraPDF test suite 6-4-t03-pass-a.pdf	Liz	migration	2020-02-06	http://github.com/archivist-liz/exifTool-configs/	Archivists Against Fascism
3	/home/archivist-liz/conferences/OPFCON/PDFA-1b/6.4 Transparency/veraPDF test suite 6-4-t03-pass-b.pdf	OPF	ingestion	2020-02-07	http://github.com/archivist-liz/exifTool-configs/	Archives Are Not Neutral
4	/home/archivist-liz/conferences/OPFCON/PDFA-1b/6.4 Transparency/veraPDF test suite 6-4-t01-pass-a.pdf	Phil Harvey	validation	2020-02-08	http://github.com/archivist-liz/exifTool-configs/	Black Lives Matter
5						

CSV

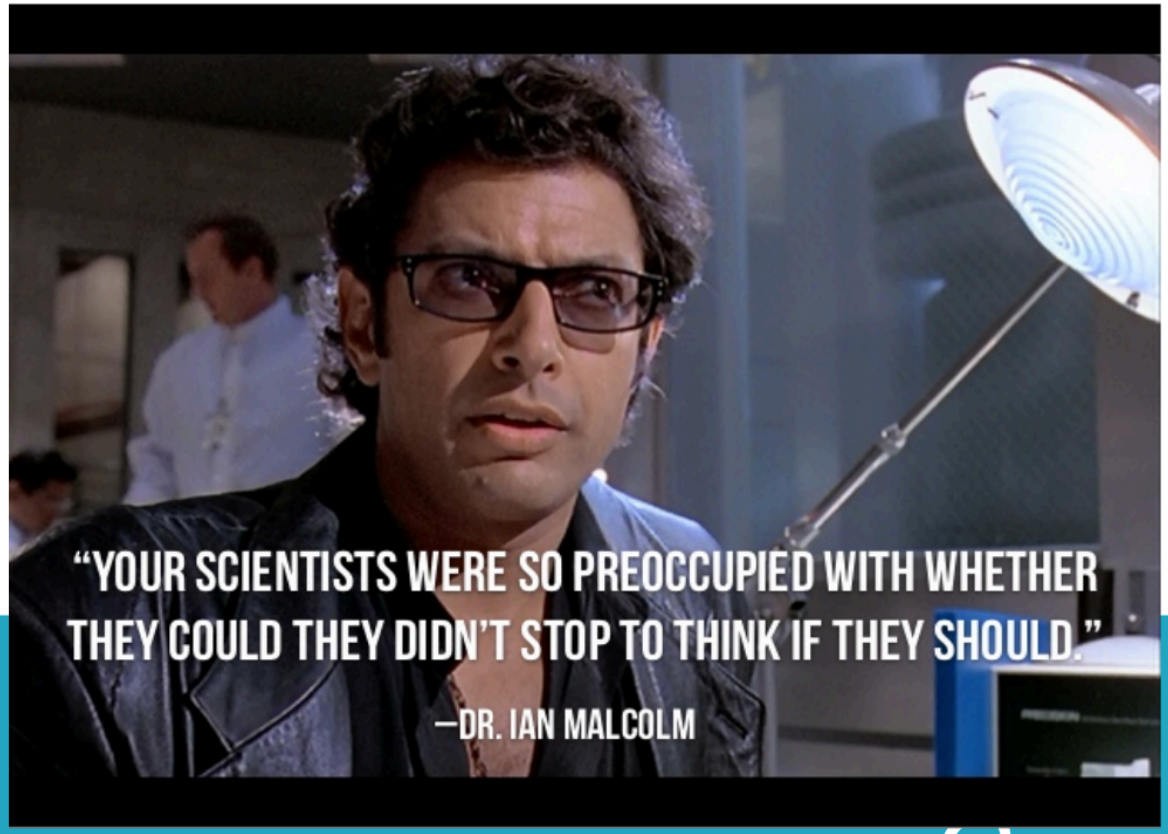
Screencast of Bulk Upload via CSV

After defining the two custom schemas according to the pdfaExtension schema, I bulk uploaded metadata from a CSV file. Voila: It's still PDF/A compliant!



Bulk upload

So it's possible, but should we do this?



wtf pdf?

Additional Resources

Here are several resources for diving further into this topic:

- **TechNote 0008: Predefined XMP Properties in PDF/A-1**: https://www.pdfa.org/wp-content/uploads/2011/08/tn0008_predefined_xmp_properties_in_pdfa-1_2008-03-20.pdf
- **TechNote 0009: XMP Extension Schemas in PDF/A-1:** https://www.pdfa.org/wp-content/uploads/2011/09/tn0009_xmp_extension_schemas_in_pdfa-1_2008-03-20.pdf
- **ExifTool Homepage** (lots of documentation and link to the forum can all be found here): https://exiftool.org
- **My repository for this work:** github.com/archivist-liz

Resources

Thank you!

You can find me at:

 github.com/archivist-liz  @SnarkivistWien

Thank You