The XMP inclusion package*

Maarten Sneep
sneep@nat.vu.nl

2005/02/15

1 Introduction

The `XMP` (eXtensible Metadata Platform) is a framework to add metadata to digital material to enhance the workflow in publication. References are given below, but the essence is that the metadata is stored in an `XML` file, and this XML stream is then embedded in the file to which it applies. How you create this `XML` file is up to you, but I started investigating this, because I wanted to embed some licensing information in the files I create. The license I chose is one of the Creative Commons licenses, and their web-site offers this information in a valid `XML` file, suitable for direct inclusion.

Note that this package is released under the CC-GNU GPL license. You can redistribute, but I kindly request that you update the version number, add a description of what you added or changed – if possible with an explanation as to why – and re-submit to CTAN, to keep it all in a single location. You can also submit changes to me, I regularly read `comp.text.tex` on usenet.

Many thanks to James Howison for pushing me to put in the `<?xpacket ?>` writing code, and suggesting to test whether they are already there.

1.1 Usage

This package defines a single command, `\includexmp{}`. The `xmp` file is specified as an argument to this command. Although there is no real specification as to where the xml-stream should be inserted into the document, I would advise to put it at the start of the file, so call the `\includexmp{}` command before `\begin{document}`. Note that the package will add the extension `.xmp` to the base filename. To include the file `metadata.xmp`, use the following:

```
. . .
\usepackage{xmpincl}
\includexmp{metadata}
. . .
```

---

[*] This document corresponds to xmpincl.dtx v2.1, dated 2005/02/15.

```
\begin{document}
. . .
```

The file `metadata.xmp` should exist in the same directory as the master document. At the end of this documentation a sample file is included that will yield a valid `XMP` enhanced pdf file.

Previous versions of this package required the inclusion of the `<?xpacket ?>` tags into the `XMP` file. This is against the standards, and several users requested that this functionality be added to the package. This new release (version 2.0) does add the `<?xpacket ?>` tags, if they are *not* present in the `xmp` file.

1.2 New in the current release (v2.1)

There used to be a clash between the Memoir document class and the `ifpdf` package. As of version 2005/03/23 v3.9 of `mempatch.sty`, this clash has been removed, and glue code that was present here, has been removed in this update. Note that this may mean that you'll need to update your distribution to include teh latest `mempatch.sty`.

1.3 References

- `http://creativecommons.org/`
- `http://creativecommons.org/technology/xmp-help`
- `http://www.adobe.com/products/xmp`

2 Implementation

First we determine if we run under pdfLaTeX, in pdf-production mode. This is best done with the `ifpdf` package. The Memoir documentclass also defines the `\ifpdf` boolean, but didn't actually load the `ifpdf` package. As of version "2005/03/23 v3.9 Patches for memoir class v1.61" of `mempatch.sty` teh loading of the ifpdf package is properly faked, and we no longer need to check for the existence of the `\ifpdf` boolean.

```
⟨*package⟩
\RequirePackage{ifpdf}
\ifpdf\else
```

Apparently we do not run under pdflatex, or we are producing DVI. Someone else may try to do this correcltly for PostScript output. Note that a metacomment has to be added to the start of the `.ps` document, and that is something for which I have no clue on how to accomplish that from within TeX.

Right now I just skip non-pdflatex support and issue a warning.

```
\PackageWarningNoLine{xmpincl}%
  {Only pdflatex is supported by the xmpincl package}
```

`includexmp`

This is the latex (`DVI`) edition: just issue a warning, and stop reading the rest of this file

```
\newcommand{\includexmp}[1]{%
  \PackageError{xmpincl}%
  {latex is not supported by the \protect\includexmp\space package}%
  {You tried to include XMP metadata in DVI production.\MessageBreak
   That doesn't work, and I friendly tried to warn you.\MessageBreak
   Just continue and pretend nothing is wrong,\MessageBreak
   but please remove the package or switch to pdflatex.}
}
```

Stop reading this file, as the rest only works when generating `pdf` directly.

```
\relax\expandafter\endinput
\fi
```

The `ifthen` package is loaded, for the string comparisons later on.

```
\RequirePackage{ifthen}
```

`mcs@xmpincl@patchFile`

Based on popular feedback, we now add the `<?xpacket . . .?>` parts ourselves. This can be a bit tricky, so bear with me. I basically create a new file `.xmpi` which starts off with the `<?xpacket ?>` tag, copy the whole `XMP` file to this new file, and add the `<?xpacket ?>` close tag. Of course, this is new functionality, so we still have to take care of our backward compatibility issues. So we check that what we've read is not an `<?xpacket ?>` tag. I'm aware of the odd combination of TeX and LaTeX coding here, but I didn't manage to get the string comparison working in palin TeX code, and this LaTeX code is maintained by others, something I really don't mind.

The macro starts off by opening a few files, one for reading the user supplied `XMP`, and another one for writing it back out again, but this time with the `<?xpacket ?>` bits included.

```
\newcommand*{\mcs@xmpincl@patchFile}[1]{
\begingroup
\newwrite\xmpinclWrite
\newread\xmpinclRead
\immediate\openin\xmpinclRead #1.xmp
\immediate\openout\xmpinclWrite #1.xmpi
```

`mcs@xmpinclStart`
`mcs@xmpinclStartAlt`
`mcs@xmpinclEnd`

The begin and en `<?xpacket ?>` strings are put in some macros to easier access. Yes, that start string was double checked against the documentation provided by Adobe. The alternate starting string is there because the `id` seems to be optional, if I understand the documentation correctly.

```
\newcommand{\mcs@xmpinclStart}%
  {<?xpacket begin='' id='W5MOMpCehiHzreSzNTczkc9d'?> }
\newcommand{\mcs@xmpinclStartAlt}%
  {<?xpacket begin='' id=''?> }
\newcommand{\mcs@xmpinclEnd}%
  {<?xpacket end='r'?> }
```

Next we change the catcode of `#` to 'other'. This is just to prevent misinterpretation of this character. Of course there are more special characters, but as far as I can see, these aren't treated in any special way (`#` is doubled by TeX to `##`).

```
\catcode`\#=12
```

We deactivate `~` and `&` as well.

```
\catcode`\~=12
\catcode`\&=12
```

Read the first line of the input file, and compare it to the start tag, and the alternate start tag. If they match, write out the standard start tag (including the `id`). If they don't match, write out the start tag, followed by the line we've just read.

```
\immediate\read\xmpinclRead to\xmpinclReadln%
\ifthenelse{%
  \equal{\mcs@xmpinclStart}{\xmpinclReadln}%
  \or%
  \equal{\mcs@xmpinclStartAlt}{\xmpinclReadln}%
}%
{%
  \immediate\write\xmpinclWrite{\mcs@xmpinclStart}%
}%
{%
  \immediate\write\xmpinclWrite{\mcs@xmpinclStart}%
  \immediate\write\xmpinclWrite{\xmpinclReadln}%
}%
```

Start the `\loop`, and read a line. Check if it is equal to the end tag or to `\par`, and if it isn't, write it out to the `.xmpi` file. The check against `\par` ensures that empty lines are skipped, and not replaced by `\par`.

The `\ifeof` test checks whether we've reached the end of the original `.xmp` file, and `\repeat`s the `\loop` if we haven't.

```
\loop%
  \immediate\read\xmpinclRead to\xmpinclReadln%
  \ifthenelse{%
    \equal{\mcs@xmpinclEnd}{\xmpinclReadln}%
    }{% Note: no if.
    }{%
    \if\par\xmpinclReadln\else%
      \immediate\write\xmpinclWrite{\xmpinclReadln}%
    \fi%
  }%
  \ifeof\xmpinclRead\else%
\repeat
```

Since we skipped any end `<?xpacket ?>` tags, we write it here. After that we close both files and end the current group (restoring the meaning of `#`, `&`, and `~`).

```
\immediate\write\xmpinclWrite{\mcs@xmpinclEnd}
\immediate\closein\xmpinclRead
\immediate\closeout\xmpinclWrite
\endgroup
}
```

`includexmp` The meat of the business. Actually pretty trivial, once you know how…

```
\newcommand{\includexmp}[1]{%
```

First check that the file can be found, and if we use the new methods, convert it.

```
  \IfFileExists{#1.xmp}{
    \mcs@xmpincl@patchFile{#1}
```

Reset the `\pdfcompresslevel` to 0, do not compress the `XML` data. This is recommended by Adobe, so that file utilities can grep the `.pdf` file for metadata, without the full capability to actually parse the `pdf` file. Keep the change local.

```
    \begingroup
      \pdfcompresslevel=0
```

Write out the `pdf` object, with the specifications given in the reference manual found at `http://www.adobe.com/products/xmp`. The `file` attribute reads the specified file from disk, although it is not clear to me if it uses the full TeX search path. To be safe, specify a local path relative to the master document. Depending on the compatibility level, we use the original file, or the newly generated version.

```
      \immediate\pdfobj stream attr {/Type /Metadata /Subtype /XML}
      file{#1.xmpi}
```

Also add the newly created object to the catalog.

```
      \pdfcatalog{/Metadata \the\pdflastobj\space 0 R}
```

end the group, which resets the compression to whatever it was before.

```
    \endgroup
  }
```

The file does not exist, and we have to generate an error. Declare a placeholder for the missing file-name, to prevent double execution of the macro.

```
  {\newcommand{\mcs@xmpincl@filename}{#1.xmp}
    \PackageError{xmpincl}%
    {The file \mcs@xmpincl@filename\space was not found}
    {The file \mcs@xmpincl@filename\space The metadata file
     wasn't found.\MessageBreak Oops.}
  }
}
⟨/package⟩
```

3 A sample `.xmp` file

Note that this is the license of this package, CC-GNU GPL.

```
⟨*license⟩
<x:xmpmeta xmlns:x='adobe:ns:meta/'>
   <rdf:RDF xmlns="http://web.resource.org/cc/"
      xmlns:dc="http://purl.org/dc/elements/1.1/"
      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
      <Work rdf:about="">
         <dc:title>xmpincl</dc:title>
         <dc:date>2005</dc:date>
         <dc:description>
            A LaTeX package to include XMP metadata in
            files generated through pdfLaTeX
         </dc:description>
         <dc:creator>
            <Agent><dc:title>Maarten Sneep</dc:title></Agent>
         </dc:creator>
         <dc:rights>
            <Agent><dc:title>Maarten Sneep</dc:title></Agent>
         </dc:rights>
         <dc:source
            rdf:resource="ftp://ftp.tex.ac.uk/tex&#45;archive/macros/latex/contrib/xmpincl.tar.
         <license rdf:resource="http://creativecommons.org/licenses/GPL/2.0/" />
      </Work>
      <License rdf:about="http://creativecommons.org/licenses/GPL/2.0/">
         <permits rdf:resource="http://web.resource.org/cc/Reproduction" />
         <permits rdf:resource="http://web.resource.org/cc/Distribution" />
         <requires rdf:resource="http://web.resource.org/cc/Notice" />
         <permits rdf:resource="http://web.resource.org/cc/DerivativeWorks" />
         <requires rdf:resource="http://web.resource.org/cc/ShareAlike" />
         <requires rdf:resource="http://web.resource.org/cc/SourceCode" />
      </License>
   </rdf:RDF>
</x:xmpmeta>
⟨/license⟩
```