The ltdocinit module

The LaTeX Project[*]

Version 0.95t, released 2022-11-03

1 ltdocinit documentation

This small module defined `\DocumentMetadata` but the definition has been moved to latex-lab. The documentation can be found in `documentmetadata-support-doc-.pdf`

`\DocumentMetadata` is also used to activate the new PDF management code and it loads a number of required files for the PDF management code. As this forces the loading of the backend files, a backend which can't be detected automatically like `dvipdfmx`, must be set in the first `\DocumentMetadata`.

Here only a few newer keys are defined and the older `\DeclareDocumentMetadata` is provided.

The module also defines commands to store document properties in a global container.

This module will slowly disappear.

1.1 `\DocumentMetadata`/`\DeclareDocumentMetadata`

`\DeclareDocumentMetadata`

`\DeclareDocumentMetadata{⟨key-value list⟩} (deprecated)`

This is an older alias for `\DocumentMetadata`

Additionally to the keys described in `documentmetadata-support-doc-.pdf` the following keys/values are implemented

`pdfstandard` Choice key to set the pdf standard.

Starting with version 0.95s it is also possible to use the values `X-4`, `X-4p`, `X-5g`, `X-5n`, `X-5pg`, `X-6`, `X-6n`, `X-6p`, `UA-1` for a PDF/X and PDF/UA standard. These keys set *only* the relevant XMP-metadata.

Beside this `A-1b`, `A-2a`, `A-2b`, `A-2u`, `A-3a`, `A-3b`, `A-3u` and `A-4` are accepted as values for A-standards. The casing is irrelevant, `a-1b` works too. Note that using these key doesn't mean that the document actually follows the standard. LaTeX can neither ensure nor check all requirements of a standard, and not everything it can do theoretically has already been implemented. For A-standard a color profile is included and the `/OutputIntent` is set and javascript action in hyperref are suppressed. The `u` variants do not force unicode, but they will pass the information to hyperref. The `a` variants do *not* enforce (or even test) a tagged pdf yet. More information can be found in the documentation of l3pdfmeta.

[*]E-mail: latex-team@latex-project.org

`pdfstandard` can be used more than once to set overlapping standards, e.g: `pdfstandard=A-2b,pdfstandard=X-4,pdfstandard=UA-1`

`xmp` A boolean, if set to false no XMP metadata are added to the PDF. The default is true. Details are described in the documentation of l3pdfmeta.

`testphase` This key is used to load testphase code. The values it accepts and their effect will change over time, when testphase packages are added or removed or when the code is moved into the kernel. New value here are

- **`new-or-1`** This patches a few commands related to the output routing. They are needed for the tagging of paragraphs, for the tagging of header and footer and to allow the PDF management to insert code which avoids that links happening at page breaks spills into the header and footer. This code is also loaded be the following values.
- **`new-or`** This loads more changes to the output routine required for the tagging. It is not compatible with every class! The code is also loaded by the `phase-II` value.

The `testphase` key can only be used in the first `\DocumentMetadata`.

`debug` This key activates some debug options. It takes a list of key-values as value. Currently the following keys are known:

- **`para`** with the default and only value `show`. It will activate the `paratagging-show` option of `tagpdf`,
- **`log`** with the values as described in the documentation `tagpdf`,
- **`uncompress`** which does the same as `uncompress` as main key
- **`pdfmanagement`** a boolean which allows to deactivate the pdfmanagement. This should only be done for debugging!
- **`firstaidoff`** This accepts a comma lists of keywords and disables the patches related to them. More information can be found in the documentation of pdfmanagement-firstaid.

1.2 Container for document properties

The module provides a container where classes, packages and users can store properties of the document which are perhaps of interest or use for other packages or the author.

The properties are stored with a key `label/property`. The values can be retrieved expandably.

`\AddToDocumentProperties` `\AddToDocumentProperties[⟨label⟩]{⟨property⟩}{⟨value⟩}`

This stores ⟨*value*⟩ under the key ⟨*label*⟩/⟨*property*⟩. By default ⟨*label*⟩ is the current package name `\@currname`. If another label is chosen, it should be one which avoids clashes with other packages using the container. The label `document` is reserved.

`\GetDocumentProperties`
`\pdfmanagement_get_documentproperties:n`

Updated: 2022-10-09

`\GetDocumentProperties{⟨label/property⟩}`
`\pdfmanagement_get_documentproperties:n{⟨label/property⟩}`

Expands to the ⟨*value*⟩ corresponding to ⟨*label/property*⟩ in the container. If ⟨*label/property*⟩ is missing, this has an empty expansion. The result is returned within `\exp_not:n`, which means that the ⟨*value*⟩ does not expand further when appearing in an x-type argument expansion.

`\pdfmanagement_get_documentproperties:nNTF`

New: 2022-10-09

```
\pdfmanagement_get_-
documentproperties:nNTF{⟨label/property⟩} ⟨token list
variable⟩
     {⟨true code⟩} {⟨false code⟩}
```

If the ⟨*label/property*⟩ is not present in the document properties container, leaves the ⟨*false code*⟩ in the input stream. The value of the ⟨*token list variable*⟩ is not defined in this case and should not be relied upon. If the ⟨*label/property*⟩ is present in the container, stores the corresponding ⟨*value*⟩ in the ⟨*token list variable*⟩ without removing it from the container, then leaves the ⟨*true code*⟩ in the input stream. The ⟨*token list variable*⟩ is assigned locally.

`\ShowDocumentProperties`

`\ShowDocumentProperties`

This show the current content of the container.

References

[1] Frank Mittelbach and Chris Rowley: *LaTeX Tagged PDF — A blueprint for a large project*. https://latex-project.org/publications/indexbyyear/2020/

2 ltdocinit implementation

```
⟨@@=pdfmanagement⟩
⟨*header⟩
\ProvidesExplPackage{ltdocinit}{2022-11-03}{0.95t}
  {Initialize document metadata}
⟨/header⟩
```

`\DocumentMetadata` will be defined by the kernel in short time. So we define it and the keys here only if it is not already defined.

2.1 The keys for `\DocumentMetadata`

We define the keys first so that we can test if `\DocumentMetadata` exist (testing for the format date would be nice but would fail for the current latex-dev).

`\g__pdfmanagement_firstaidoff_clist`

A list to store the firstaid code which should be disabled

```
⟨*package⟩
\clist_new:N \g__pdfmanagement_firstaidoff_clist
```

(*End definition for* `\g__pdfmanagement_firstaidoff_clist`.)

\g__pdfmanagement_testphase_tl a tl to store the testphase loading code so that we can load them at the end of the command.

```
\tl_new:N \g__pdfmanagement_testphase_tl
```

(*End definition for* `\g__pdfmanagement_testphase_tl`*.*)

```
\keys_define:nn { document / metadata }
  {
    ,testphase / new-or-1 .code:n =
      {
        \tl_gput_right:Nn\g__pdfmanagement_testphase_tl
           {
             \file_if_exist_input:nF {new-or-1-latex-lab-testphase.ltx}
               {
                \RequirePackage{output-patches-tmp-ltx}
               }
           }
      }
  }
```

2.2 \DeclareDocumentMetadata

We define the older alias \DeclareDocumentMetadata

\DeclareDocumentMetadata

```
\NewCommandCopy\DeclareDocumentMetadata\DocumentMetadata
```

(*End definition for* `\DeclareDocumentMetadata`*. This function is documented on page 1.*)

2.3 Container for document Properties

The container for the document properties is a prop

\g__pdfmanagement_documentproperties_prop

```
\prop_new:N \g__pdfmanagement_documentproperties_prop %
```

(*End definition for* `\g__pdfmanagement_documentproperties_prop`*.*)

\AddToDocumentProperties

```
\NewDocumentCommand\AddToDocumentProperties{O{\@currname}mm}
  {
    \exp_args:NNx
      \prop_gput:Nnn \g__pdfmanagement_documentproperties_prop
        {
          \tl_if_blank:eTF {#1}{top-level/}{#1/} #2
        }
        { #3}
  }
```

(*End definition for* `\AddToDocumentProperties`*. This function is documented on page 2.*)

\GetDocumentProperties

```
\NewExpandableDocumentCommand\GetDocumentProperties{m}
  {
    \prop_item:Nn \g__pdfmanagement_documentproperties_prop {#1}
  }
```

(*End definition for* `\GetDocumentProperties`*. This function is documented on page 3.*)

For uses in modules (e.g. l3pdfmeta) we provide an expl3 version without the xparse overhead:

`\pdfmanagement_get_documentproperties:n`

```
\cs_new:Npn\pdfmanagement_get_documentproperties:n #1
  {
    \prop_item:Nn \g__pdfmanagement_documentproperties_prop {#1}
  }
```

(*End definition for* `\pdfmanagement_get_documentproperties:n`*. This function is documented on page 3.*)

The following allows to retrieve the values with branching.

`\pdfmanagement_get_documentproperties:nNTF`

```
\prg_new_protected_conditional:Npnn
  \pdfmanagement_get_documentproperties:nN #1#2 { T , F , TF }
    {
     \prop_get:NnNTF \g__pdfmanagement_documentproperties_prop  {#1} #2
       {
        \prg_return_true:
       }
       {
        \prg_return_false:
       }
    }
```

(*End definition for* `\pdfmanagement_get_documentproperties:nNTF`*. This function is documented on page 3.*)

`\ShowDocumentProperties`

```
\msg_new:nnn  { pdfmanagement } { show-properties }
  {
    The~following~document~properties~have~been~stored:
    #1
  }
\NewDocumentCommand\ShowDocumentProperties {}
  {
    \msg_show:nnx {pdfmanagement}{show-properties}
      {
        \prop_map_function:NN \g__pdfmanagement_documentproperties_prop \msg_show_item:nn
      }
  }
```

(*End definition for* `\ShowDocumentProperties`*. This function is documented on page 3.*)

```
⟨/package⟩
```

Index

The italic numbers denote the pages where the corresponding entry is described, numbers underlined point to the definition, all others indicate the places where it is used.