The tagpdf-roles module
Tags, roles and namesspace code
part of the tagpdf package

Ulrike Fischer[*]

Version 0.82, released 2021-06-14

```
⟨@@=tag⟩
⟨*roles⟩
\ProvidesExplPackage {tagpdf-roles-code} {2021-06-14} {0.82}
 {part of tagpdf - code related to roles and structure names}
```

1 Code related to roles and structure names

1.1 Variables

Tags have both a name (a string) and a number (for the lua attribute). Testing a name is easier with a prop, while accessing with a number is better done with a seq. So both are used and must be kept in sync if a new tag is added. The number is only relevant for the MC type, tags with the same name from different names spaces can have the same number.

\g__tag_role_tags_seq
\g__tag_role_tags_prop

```
\__tag_seq_new:N  \g__tag_role_tags_seq  %to get names (type/NS) from numbers
\__tag_prop_new:N \g__tag_role_tags_prop %to get numbers  from names (type/NS)
```

(*End definition for* `\g__tag_role_tags_seq` *and* `\g__tag_role_tags_prop`*.*)

\g__tag_role_tags_NS_prop in pdf 2.0 tags belong to a name space. For every tag we store a default name space. The keys are the tags, the value shorthands like pdf2, or mathml. There is no need to access this from lua, so we use the standard prop commands.

```
\prop_new:N    \g__tag_role_tags_NS_prop %to namespace info
```

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

\g__tag_role_NS_prop The standard names spaces are the following. The keys are the name tagpdf will use, the urls are the identifier in the namespace object.

mathml http://www.w3.org/1998/Math/MathML

pdf2 http://iso.org/pdf2/ssn

pdf http://iso.org/pdf/ssn (default)

[*]E-mail: fischer@troubleshooting-tex.de

user `\c__tag_role_userNS_id_str` (random id, for user tags)

More namespaces are possible and their objects references and the ones of the namespaces must be collected so that an array can be written to the StructTreeRoot at the end (see tagpdf-tree). We use a prop to store also the object reference as it will be needed rather often.

```
\prop_new:N \g__tag_role_NS_prop % collect namespaces
```

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

We need also a bunch of temporary variables:

`\l__tag_role_tag_tmpa_tl`
`\l__tag_role_tag_namespace_tmpa_tl`
`\l__tag_role_role_tmpa_tl`
`\l__tag_role_role_namespace_tmpa_tl`

```
\tl_new:N \l__tag_role_tag_tmpa_tl
\tl_new:N \l__tag_role_tag_namespace_tmpa_tl
\tl_new:N \l__tag_role_role_tmpa_tl
\tl_new:N \l__tag_role_role_namespace_tmpa_tl
```

(*End definition for* `\l__tag_role_tag_tmpa_tl` *and others.*)

1.2 Namesspaces

The following commands setups a names space. Namespace dictionaries can contain an optional `/Schema` and `/RoleMapNS` entry. We only reserve the objects but delay the writing to the finish code, where we can test if the keys and the name spaces are actually needed This commands setups objects for the name space and its rolemap. It also initialize a prop to collect the rolemaps if needed.

`\__tag_role_NS_new:nnn` `\__tag_role_NS_new:nnn{⟨shorthand⟩}{⟨URI-ID⟩}Schema`

`\__tag_role_NS_new:nnn`

```
\cs_new_protected:Npn \__tag_role_NS_new:nnn #1 #2 #3
  {
    \msg_redirect_name:nnn { pdfdict } { empty-value } { none }
    \pdf_object_new:nn {tag/NS/#1}{dict}
    \pdfdict_new:n     {g__tag_role/Namespace_#1_dict}
    \pdf_object_new:nn {__tag/RoleMapNS/#1}{dict}
    \pdfdict_new:n     {g__tag_role/RoleMapNS_#1_dict}
    \pdfdict_gput:nnn
      {g__tag_role/Namespace_#1_dict}
      {Type}
      {/Namespace}
    \pdf_string_from_unicode:nnN{utf8/string}{#2}\l_tmpa_str
    \pdfdict_gput:nnx
      {g__tag_role/Namespace_#1_dict}
      {NS}
      {\l_tmpa_str}
    %RoleMapNS is added in tree
    \pdfdict_gput:nnx{g__tag_role/Namespace_#1_dict}
      {Schema}{#3}
    \prop_gput:Nnx \g__tag_role_NS_prop {#1}{\pdf_object_ref:n{tag/NS/#1}~}
    \msg_redirect_name:nnn { pdfdict } { empty-value } { warning }
  }
```

(*End definition for* `\__tag_role_NS_new:nnn`.)

We need an id for the user space. For the tests it should be possible to set it to a fix value. So we use random numbers which can be fixed by setting a seed. We fake a sort of GUID but not try to be really exact as it doesn't matter ...

`\c__tag_role_userNS_id_str`

```
\str_const:Nx \c__tag_role_userNS_id_str
  { data:,
    \int_to_Hex:n{\int_rand:n {65535}}
    \int_to_Hex:n{\int_rand:n {65535}}
    -
    \int_to_Hex:n{\int_rand:n {65535}}
    -
    \int_to_Hex:n{\int_rand:n {65535}}
    -
    \int_to_Hex:n{\int_rand:n {65535}}
    -
    \int_to_Hex:n{\int_rand:n {16777215}}
    \int_to_Hex:n{\int_rand:n {16777215}}
  }
```

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

Now we setup the standard names spaces. Currently only if we detect pdf2.0 but this will perhaps have to change if the structure code gets to messy.

```
\pdf_version_compare:NnT > {1.9}
  {
    \__tag_role_NS_new:nnn {pdf}   {http://iso.org/pdf/ssn}{}
    \__tag_role_NS_new:nnn {pdf2}  {http://iso.org/pdf2/ssn}{}
    \__tag_role_NS_new:nnn {mathml}{http://www.w3.org/1998/Math/MathML}{}
    \exp_args:Nnx
    \__tag_role_NS_new:nnn {user}{\c__tag_role_userNS_id_str}{}
  }
```

1.3 Data

In this section we setup the standard data. At first the list of structure types. We split them in three lists, the tags with which are both in the pdf and pdf2 namespace, the one only in pdf and the one with the tags only in pdf2. We also define a rolemap for the pdfII only type to pdf so that they can always be used.

`\c__tag_role_sttags_pdf_pdfII_clist`
`\c__tag_role_sttags_only_pdf_clist`
`\c__tag_role_sttags_only_pdfII_clist`
`\c__tag_role_sttags_mathml_clist`
`\c__tag_role_sttags_pdfII_to_pdf_prop`

```
%
\clist_const:Nn \c__tag_role_sttags_pdf_pdfII_clist
  {
    Document,   %A complete document. This is the root element
                %of any structure tree containing
                %multiple parts or multiple articles.
    Part,       %A large-scale division of a document.
    Sect,       %A container for grouping related content elements.
    Div,        %A generic block-level element or group of elements
    Caption,    %A brief portion of text describing a table or figure.
    Index,
    NonStruct,  %probably not needed
    H,
    H1,
    H2,
    H3,
    H4,
    H5,
    H6,
    P,
    L,            %list
    LI,           %list item (around label and list item body)
    Lbl,          %list label
    LBody,        %list item body
    Table,
    TR,           %table row
    TH,           %table header cell
    TD,           %table data cell
    THead,        %table header (n rows)
    TBody,        %table rows
    TFoot,        %table footer
    Span,         %generic inline marker
    Link,         %
    Annot,
    Figure,
    Formula,
    Form,
    % ruby warichu etc ..
    Ruby,
    RB,
    RT,
    Warichu,
    WT,
    WP,
    Artifact % only MC-tag ?...
  }

\clist_const:Nn \c__tag_role_sttags_only_pdf_clist
 {
   Art,        %A relatively self-contained body of text
               %constituting a single narrative or exposition
   BlockQuote, %A portion of text consisting of one or more paragraphs
               %attributed to someone other than the author of the
               %surrounding text.
   TOC,        %A list made up of table of contents item entries
               %(structure tag TOCI; see below) and/or other
               %nested table of contents entries
   TOCI,       %An individual member of a table of contents.
               %This entry's children can be any of the following structure  tags:
               %Lbl,Reference,NonStruct,P,TOC
   Index,
   Private,
   Quote,       %inline quote
   Note,        %footnote, endnote. Lbl can be child
   Reference,   %A citation to content elsewhere in the document.
   BibEntry,    %bibentry
   Code
 }

\clist_const:Nn \c__tag_role_sttags_only_pdfII_clist
 {
   DocumentFragment
   ,Aside
   ,H7
   ,H8
   ,H9
   ,H10
   ,Title
   ,FENote
   ,Sub
   ,Em
   ,Strong
   ,Artifact
 }

\clist_const:Nn \c__tag_role_sttags_mathml_clist
 {
   abs
   ,and
   ,annotation
   ,apply
   ,approx
   ,arccos
   ,arccosh
   ,arccot
   ,arccoth
   ,arccsc
   ,arccsch
   ,arcsec
   ,arcsech
   ,arcsin
   ,arcsinh
   ,arctan
   ,arctanh
   ,arg
   ,bind
   ,bvar
   ,card
   ,cartesianproduct
   ,cbytes
   ,ceiling
   ,cerror
   ,ci
   ,cn
   ,codomain
   ,complexes
   ,compose
   ,condition
   ,conjugate
   ,cos
```

```
,cosh
,cot
,coth
,cs
,csc
,csch
,csymbol
,curl
,declare
,degree
,determinant
,diff
,divergence
,divide
,domain
,domainofapplication
,emptyset
,eq
,equivalent
,eulergamma
,exists
,exp
,exponentiale
,factorial
,factorof
,false
,floor
,fn
,forall
,gcd
,geq
,grad
,gt
,ident
,image
,imaginary
,imaginaryi
,implies
,in
,infinity
,int
,integers
,intersect
,interval
,inverse
,lambda
,laplacian
,lcm
,leq
,limit
,ln
,log
,logbase
,lowlimit
```

```
,lt
,maction
,maligngroup
,malignmark
,math
,matrix
,matrixrow
,max
,mean
,median
,menclose
,merror
,mfenced
,mfrac
,mglyph
,mi
,min
,minus
,mlabeledtr
,mlongdiv
,mmultiscripts
,mn
,mo
,mode
,moment
,momentabout
,mover
,mpadded
,mphantom
,mprescripts
,mroot
,mrow
,ms
,mscarries
,mscarry
,msgroup
,msline
,mspace
,msqrt
,msrow
,mstack
,mstyle
,msub
,msubsup
,msup
,mtable
,mtd
,mtext
,mtr
,munder
,munderover
,naturalnumbers
,neq
,none
```

```
  ,not
  ,notanumber
  ,notin
  ,notprsubset
  ,notsubset
  ,or
  ,otherwise
  ,outerproduct
  ,partialdiff
  ,pi
  ,piece
  ,piecewise
  ,plus
  ,power
  ,primes
  ,product
  ,prsubset
  ,quotient
  ,rationals
  ,real
  ,reals
  ,reln
  ,rem
  ,root
  ,scalarproduct
  ,sdev
  ,sec
  ,sech
  ,selector
  ,semantics
  ,sep
  ,set
  ,setdiff
  ,share
  ,sin
  ,sinh
  ,subset
  ,sum
  ,tan
  ,tanh
  ,tendsto
  ,times
  ,transpose
  ,true
  ,union
  ,uplimit
  ,variance
  ,vector
  ,vectorproduct
  ,xor
 }

\prop_const_from_keyval:Nn \c__tag_role_sttags_pdfII_to_pdf_prop
  {
    DocumentFragment = Art,
    Aside = Note,
    Title = H1,
    Sub   = Span,
    H7    = H6 ,
    H8    = H6 ,
    H9    = H6 ,
    H10   = H6,
    FENote= Note,
    Em    = Span,
    Strong= Span,
  }
```

(*End definition for* `\c__tag_role_sttags_pdf_pdfII_clist` *and others.*)

We fill the structure tags in to the seq. We allow all pdf1.7 and pdf2.0, and role map if needed the 2.0 tags.

```
% get tag name from number: \seq_item:Nn \g__tag_role_tags_seq { n }
% get tag number from name: \prop_item:Nn \g__tag_role_tags_prop { name }

\clist_map_inline:Nn \c__tag_role_sttags_pdf_pdfII_clist
  {
    \__tag_seq_gput_right:Nn \g__tag_role_tags_seq { #1 }
    \prop_gput:Nnn \g__tag_role_tags_NS_prop    { #1 }{ pdf2 }
  }
\clist_map_inline:Nn \c__tag_role_sttags_only_pdf_clist
  {
    \__tag_seq_gput_right:Nn \g__tag_role_tags_seq { #1 }
    \prop_gput:Nnn \g__tag_role_tags_NS_prop    { #1 }{ pdf }
  }
\clist_map_inline:Nn \c__tag_role_sttags_only_pdfII_clist
  {
    \__tag_seq_gput_right:Nn \g__tag_role_tags_seq { #1 }
    \prop_gput:Nnn \g__tag_role_tags_NS_prop    { #1 }{ pdf2 }
  }
\pdf_version_compare:NnT > {1.9}
  {
     \clist_map_inline:Nn \c__tag_role_sttags_mathml_clist
       {
         \__tag_seq_gput_right:Nn \g__tag_role_tags_seq { #1 }
         \prop_gput:Nnn \g__tag_role_tags_NS_prop    { #1 }{ mathml }
       }
  }
```

For luatex and the MC we need a name/number relation. The name space is not relevant.

```
\int_step_inline:nnnn { 1 }{ 1 }{ \seq_count:N \g__tag_role_tags_seq }
  {
    \__tag_prop_gput:Nxn \g__tag_role_tags_prop
      {
        \seq_item:Nn \g__tag_role_tags_seq  { #1 }
      }
      { #1 }
  }
```

1.4 Adding new tags and rolemapping

1.4.1 pdf 1.7 and earlier

With this versions only RoleMap is filled. At first the dictionary:

g__tag_role/RoleMap_dict

```
\pdfdict_new:n {g__tag_role/RoleMap_dict}
```

(*End definition for* g__tag_role/RoleMap_dict*.*)

__tag_role_add_tag:nn

The pdf 1.7 version has only two arguments: new and rolemap name. To make pdf 2.0 types usable we directly define a rolemapping for them.

```
\cs_new_protected:Nn \__tag_role_add_tag:nn %(new) name, reference to old
  {
    \prop_if_in:NnF \g__tag_role_tags_prop {#1}
      {
         \msg_info:nnn { tag }{new-tag}{#1}
         \__tag_seq_gput_right:Nn \g__tag_role_tags_seq { #1 }
         \__tag_prop_gput:Nnx \g__tag_role_tags_prop    { #1 }
           {
             \seq_count:N \g__tag_role_tags_seq
           }
         \prop_gput:Nnn \g__tag_role_tags_NS_prop    { #1 }{ user }
      }
    \__tag_check_add_tag_role:nn {#1}{#2}
    \tl_if_empty:nF { #2 }
      {
        \pdfdict_gput:nnx {g__tag_role/RoleMap_dict}
          {#1}
          {\pdf_name_from_unicode_e:n{#2}}
      }
  }
\cs_generate_variant:Nn \__tag_role_add_tag:nn {VV}

\pdf_version_compare:NnT < {2.0}
  {
     \prop_map_inline:Nn \c__tag_role_sttags_pdfII_to_pdf_prop
       {
         \__tag_role_add_tag:nn {#1}{#2}
       }
  }

```

(*End definition for* __tag_role_add_tag:nn*.*)

1.4.2 The pdf 2.0 version

__tag_role_add_tag:nnnn

The pdf 2.0 version takes four arguments: tag/namespace/role/namespace

```
\cs_new_protected:Nn \__tag_role_add_tag:nnnn %tag/namespace/role/namespace
  {
    \msg_info:nnn { tag }{new-tag}{#1}
    \__tag_seq_gput_right:Nn \g__tag_role_tags_seq { #1 }
    \__tag_prop_gput:Nnx \g__tag_role_tags_prop    { #1 }
       {
        \seq_count:N \g__tag_role_tags_seq
      }
    \prop_gput:Nnn \g__tag_role_tags_NS_prop    { #1 }{ #2 }
    \__tag_check_add_tag_role:nn {#1}{#3}
    \pdfdict_gput:nnx {g__tag_role/RoleMapNS_#2_dict}{#1}
      {
        [
          \pdf_name_from_unicode_e:n{#3}
          \c_space_tl
          \pdf_object_ref:n {tag/NS/#4}
        ]
      }
  }
\cs_generate_variant:Nn \__tag_role_add_tag:nnnn {VVVV}
```

(*End definition for* `\__tag_role_add_tag:nnnn`.)

1.5 Key-val user interface

The user interface use the key `add-new-tag`, which takes either a keyval list as argument, or a tag/role.

`tag`
`tag-namespace`
`role`
`role-namespace`
`add-new-tag`

```
\keys_define:nn { __tag / tag-role }
  {
    ,tag .tl_set:N = \l__tag_role_tag_tmpa_tl
    ,tag-namespace  .tl_set:N = \l__tag_role_tag_namespace_tmpa_tl
    ,role .tl_set:N = \l__tag_role_role_tmpa_tl
    ,role-namespace .tl_set:N = \l__tag_role_role_namespace_tmpa_tl
  }

\keys_define:nn { __tag / setup }
  {
    add-new-tag .code:n =
     {
       \keys_set_known:nnnN
         {__tag/tag-role}
         {
           tag-namespace=user,
           role-namespace=, %so that we can test for it.
          #1
         }{__tag/tag-role}\l_tmpa_tl
       \tl_if_empty:NF \l_tmpa_tl
         {
           \exp_args:NNno \seq_set_split:Nnn \l_tmpa_seq { / } {\l_tmpa_tl/}
           \tl_set:Nx \l__tag_role_tag_tmpa_tl  { \seq_item:Nn \l_tmpa_seq {1} }
           \tl_set:Nx \l__tag_role_role_tmpa_tl { \seq_item:Nn \l_tmpa_seq {2} }
         }
      \tl_if_empty:NT \l__tag_role_role_namespace_tmpa_tl
         {
           \prop_get:NVNTF
             \g__tag_role_tags_NS_prop
             \l__tag_role_role_tmpa_tl
             \l__tag_role_role_namespace_tmpa_tl
```

```
              {
                 \prop_if_in:NVF\g__tag_role_NS_prop \l__tag_role_role_namespace_tmpa_tl
                  {
                    \tl_set:Nn \l__tag_role_role_namespace_tmpa_tl {user}
                  }
              }
              {
                \tl_set:Nn \l__tag_role_role_namespace_tmpa_tl {user}
              }
          }
       \pdf_version_compare:NnTF < {2.0}
        {
         %TODO add check for emptyness?
           \__tag_role_add_tag:VV
               \l__tag_role_tag_tmpa_tl
               \l__tag_role_role_tmpa_tl
        }
        {
          \__tag_role_add_tag:VVVV
            \l__tag_role_tag_tmpa_tl
            \l__tag_role_tag_namespace_tmpa_tl
            \l__tag_role_role_tmpa_tl
            \l__tag_role_role_namespace_tmpa_tl
        }
    }
  }
⟨/roles⟩
```

(*End definition for* tag *and others. These functions are documented on page* ??.)

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.