Inhaltsverzeichnis

Modul:Format TemplateData/Doku

Dies ist die Dokumentationsseite für Modul:Format TemplateData

Vorlage:Nutshell Vorlage:Lua

`TemplateData` – Module with auxilary functions for template documentation, especially by TemplateData.

Core functionality is improved presentation on documentation pages.

Inhaltsverzeichnis

1 Vorlage:Anchor Improve template documentation page – MediaWiki disappointing 2
1.1 Vorlage:Anchor Improved presentation 3
1.2 Eliminate disadvantages 4
2 General workflow 4
3 Vorlage:Anchor Functions for templates 5
3.1 Details 5
3.2 Examples (test page) 6
4 Vorlage:Anchor Functions for Lua modules (API) 6
5 Usage 6
6 Dependencies 6

Vorlage:Anchor Improve template documentation page – MediaWiki disappointing

For presentation of template depiction in VisualEditor agreement was made to abandon all markup and clickable links, permitting all tooltips in all environments. Basically this is reasonable, albeit tooltips with markup and clickable links are supported as HTML application for decades now and JavaScript is present anyway when VisualEditor is used.

- In consequence it was decided, that also presentation on template documentation views never ever is permitted to contain effective links or markup.
- That involved, that on many template documentation pages two separated parameter documentation tables are needed and required to be maintained simultaneously: One plain text version for VisualEditor, and a useful one for complex circumstances, with links and markup and lists and tables. – BTW, VisualEditor has not only tooltips, but also a static GUI view, where the lack of deepening links in parameter description is painful.

This state is indefensible.

Vorlage:Anchor Improved presentation

In addition to the simple syntax supported by MediaWiki and presented in the VisualEditor, the following features can be added to the JSON code for the template documentation page. They affect the elements classified as *InterfaceText*, but are only useful for description fields.

Wikilinks (internal format)

- Using double square brackets pages can be linked as common.
- In VisualEditor, only link title is visible, as it is displayed otherwise.

External links (URL format)

- Open URL are linked as usual by themselves. In VisualEditor they appear as normal text.
- External links enclosed in simple square brackets are displayed normally on the template documentation page. In VisualEditor the title is omitted and the URL is displayed so that the users can c&p it and transfer it to the address field of the browser. There is no other way.

Apostrophs `'` for italic and bold

- They can be used to emphasize on the documentation page and are missing in VisualEditor (regular script).

HTML entities

- The following entities can be used: `<` `>` `&` `&quot;` ` ` and all numeric formats.

HTML tags

- HTML tags (and the MediaWiki elements that are not replaced in advance) are removed for the VisualEditor. Otherwise, they remain effective.
- Attributes are often included in `"`, which conflicts with the JSON syntax. It is important to make sure that `'` is used, which can be a problem with template transclusions.

`<noexport>` ... `</noexport>`

- The enclosed areas are not exported to the VisualEditor.
- More complex wiki syntax and extensive explanations can be restricted to the documentation page.
- Within a *noexport* area, the line structure of the source text is considered. Otherwise everything is running in a single line, as it would also be represented in the VisualEditor.

Templates

- In particular when the template parameter `JSON=` is used, templates can be distributed anywhere in the JSON code. However, the expanded syntax might collide with the JSON syntax.

More effects

- According to the state (required, suggested, optional, deprecated) the table rows are highlighted in light blue, white, gray and pale red.
- When sorting by state, this importance is taken into account and not the alphabetical sequence of the keywords.
- Each parameter can be addressed as a jump destination. The fragment is `#templatedata:` *parameter-name*.
- Missing labels are highlighted as errors.
- A maintenance category is triggered if errors occur.
- If there are no parameters, the element `params:{}` is not required.

Eliminate disadvantages

Two aspects were found to be particularly disturbing in 2013–2017:

1. Even if no parameters at all were defined, a table head is always displayed for a table without content. Even more, this is sortable.
   - A reduction was rejected with Vorlage:Phab. A sortable table of the parameters would be always necessary, even if the table has no rows at all and consists only of the header row.
   - This ridiculous statement led to the development of this module in 2016.
2. Even if the context does not permit that default values or even AutoValue specifications will be defined ever, a content-free six-line definition list is output for each individual parameter value.
   - Vorlage:Phab / Vorlage:Phab / Vorlage:Phab / Vorlage:Phab
   - MediaWiki did not even deign to answer the disastrous documentation page situation.

The general comments show that MediaWiki only regards the presentation of TemplateData specifications in the VisualEditor as important. However, someone has to program and maintain the templates and someone needs to generate the template description and make it manageable beside the functionality in the VisualEditor form, but that is beyond ken.

General workflow

- An attempt is made to read the JSON object (string) from passed template parameters.
- If this failed, the source code of the current and the documentation page is searched for `<templatedata>` elements.
- Two representations are obtained from the JSON object input:
  1. A localized version, markup etc. stripped off, in JSON format.
  2. An HTML structure, basically similar to the MediaWiki representation, possibly with table of the parameters, with enhanced features.

- The result of the template is a visible documentation with markup, followed by a hidden `<templatedata>` element. This is done for the export and corresponds to the MediaWiki guidelines.
  - If current page has been identified as documentation page the hidden `<templatedata>` is suppressed, and those pages do not appear separately in Special:PagesWithProp/templatedata.

Vorlage:Anchor Functions for templates

Details

f Vorlage:Anchor

Improve TemplateData-presentation; used in Template:Format TemplateData

Parameters of template transclusion environment (all optional):

1
: JSON string or `<templatedata>` object

JSON
: JSON string
(precedes **1**)
Transition from `<templatedata>` objects with pipe symbols needs special attention: Pipes are to be represented as `{{!}}`, on double curly brackets one should be encoded by HTML entity.

TOC
: `1` – Insert table of contents after general purpose descriptions; but before parameter list, if present
Example

lazy
: `1` – Presentation only, do not generate an effective data block
For general method descriptions.

debug
: `1` – developer mode

Parameters of `#invoke` for particular project adaption (all optional):

cat
: Title of a maintenance category on invalid parameter value etc.

debug
: Development mode, if provided and not equal `0`

docpageCreate
: Pattern for creation of subpage names; `%s/Doku`

docpageDetect
: Pattern for recognition of subpage names; `/Doku$`

msgDescMiss
: Localisation: complaint text on missing `description`

Returns: HTML code; and/or error message, probably with `class="error"`

failsafe Vorlage:Anchor

Version identification: `2017-11-06`

Optional additional parameter `1` – requested minimal version identification

Returns: (empty), if minimal version condition not matched

Examples (test page)

A test page illustrates practical use.

Vorlage:Anchor Functions for Lua modules (API)

Some functions described above can be used by other modules:

```
local lucky, TemplateData = pcall( require, "Module:Format TemplateData" )
if type( TemplateData ) == "table" then
    TemplateData = TemplateData.TemplateData()
else
    -- failure; TemplateData is the error message
    return "<span class='error'>" .. TemplateData .. "</span>"
end
```

TemplateData.failsafe(atleast)

1. atleast
   optional
   nil or minimal version request

Returns: *string* or *false*

TemplateData.getPlainJSON(adapt)

Reduce enhanced JSON information to MediaWiki JSON

1. adapt
   string, with JSON (enhanced)

Returns: *string*, with JSON (MediaWiki)

TemplateData.test(adapt, arglist)

Simulation of template functionality

1. adapt
   table, `#invoke` parameters
2. arglist
   table, template parameters

Returns: *string*

Usage

Currently focusing on one template only:

- Template:Format TemplateData

Dependencies

- Module:Plain text

- Module:TNT (for the Vorlage:Para parameter)