sphinxcontrib-opendataservices

Open Data Services

Jul 07, 2021
## CONTENTS

1 Installation 3

2 JSON Include Directives 5
   2.1 Standard JSON Include ................................. 5
   2.2 Flat JSON Include .................................... 8

3 JSON Schema Directives 11
   3.1 CSS ................................................. 14

4 Misc Directives 15
   4.1 Directives in other repositories ......................... 17

Index 19
Open Data Services’ Sphinx directives.
Add this line to your requirements.txt:

```bash
-e git+https://github.com/OpenDataServices/sphinxcontrib-opendataservices.
   ...git@23ce17656feaa237584af8822bd57ac39b498f93#egg=sphinxcontrib-opendataservices
```

Then run `pip install -r requirements.txt`.

Edit your doc's `conf.py` and add `sphinxcontrib.opendataservices` to the `extensions` array. e.g.

```python
extensions = ['sphinxcontrib.opendataservices']
```
JSON INCLUDE DIRECTIVES

2.1 Standard JSON Include

.. jsoninclude::
   :jsonpointer: /a/0/b
   :expand: e

Output:

```json
{
    "c": 1,
    "d": 2,
    "e": [
        {
            "f": 3
        },
        {
            "f": 4
        }
    ],
    "g": 5
}
```

The `expand` option is needed to expand a list when the json is folded by javascript. For more info see Javascript below.

exclude option:

Source:

```eval-rst
.. jsoninclude:: _static/example.json
   :jsonpointer: /a/0/b

(continues on next page)```
... :exclude: e,g

Output:
```
{
  "c": 1,
  "d": 2
}
```

include_only option:

Source:
```
```{eval-rst}
.. jsoninclude:: _static/example.json
  :jsonpointer: /a/0/b
  :include_only: e,g
  :expand: e
```...

Output:
```
{
  "e": [
    {
      "f": 3
    },
    {
      "f": 4
    }
  ],
  "g": 5
}
```

### 2.1.1 Javascript

To fold the JSON with javascript, you need to include the following files:

- renderjson.css
- renderjson.js
- json-example-format.js

You need to add the files to a _static folder within your docs, and then add the following to _templates/layout.html.

{% extends "!layout.html" %}
{% set css_files = css_files + ["_static/renderjson.css"] %}
{% set script_files = script_files + ["_static/renderjson.js", "_static/json-example-format.js"] %}

The option expand can be used to control which lists are expanded initially. The option title can be used to give the json include a title; only one of consecutive includes will be shown, with a select box to switch:
Source:

```{eval-rst}
.. jsoninclude:: _static/example.json
   :jsonpointer:
   :title: collapsed

.. jsoninclude:: _static/example.json
   :jsonpointer:
   :expand: a,b,e,h
   :title: expanded
```

Output:

```json
{
    "a": [
        {
            "b": {
                "c": 1,
                "d": 2,
                "e": [
                    {
                        "f": 3
                    },
                    {
                        "f": 4
                    }
                ],
                "g": 5
            }
        }
    ],
    "h": [
        {
            "i": 11,
            "j": 12
        },
        {
            "k": 13,
            "l": 14
        }
    ]
}
```

```json
{
    "a": [
        {
            "b": {
                "c": 1,
                "d": 2,
                "e": [
                    {
                        "f": 3
                    },
                    {
                        "f": 4
                    }
                ],
                "g": 5
            }
        }
    ]
}
```

(continues on next page)
2.2 Flat JSON Include

.. jsoninclude-flat::
   Include a section of a JSON file, flattened into a table representation, given a jsonpointer.

Examples, using this json file:

Source:

```{eval-rst}
.. jsoninclude-flat:: _static/example.json
   :jsonpointer: /a/0/b
```

Output:

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>5</td>
</tr>
</tbody>
</table>

recursive (include nested dicts and lists):

Source:

```{eval-rst}
.. jsoninclude-flat:: _static/example.json
   :jsonpointer: /a/0/b
   :recursive:   
```

Chapter 2. JSON Include Directives
Output:

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>/a/0/b/c</td>
<td>/a/0/b/d</td>
<td>/a/0/b/e/0/f</td>
<td>/a/0/b/e/1/f</td>
<td>/a/0/b/g</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

List of items directly under the json pointer:

Source:
```
```{eval-rst}
.. jsoninclude-flat:: _static/example.json
   :jsonpointer: /h
```

Output:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>/h/k</td>
<td>/h/l</td>
</tr>
<tr>
<td>11</td>
<td>12</td>
</tr>
<tr>
<td>13</td>
<td>14</td>
</tr>
</tbody>
</table>

Remove part of the path from the headings using `ignore_path`:

Source:
```
```{eval-rst}
.. jsoninclude-flat:: _static/example.json
   :jsonpointer: /a/0/b
   :ignore_path: /a/0/b/
```

Output:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>d</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

| g | 5 |

`jsoninclude-flat` also has the options `exclude` and `include_only`, the same as for `jsoninclude` (see above).
.. jsonschema::

    The core jsonschema directive, mainaated in the sphinxcontrib-jsonschema repository.

    The exact output of this is somewhat specific to the OCDS usecases, so we subclass the directive to create
alternatives for other data standards, see below.

    Source:

    ```{eval-rst}
    .. jsonschema:: _static/example_schema.json
    ```

    Output:
### jsonschema-titles:
Display titles, but not field names.
Currently only used for 360Giving.

Source:
```
```
Title | Description | Type | Required
--- | --- | --- | ---
Identifier | An identifier | string | True
Foo bar | Foo bar baz bar baz bar baz bar baz. Bar foo baz foo bar. Baz bar foo bar foo bar foo bar foo. | integer | False
When | When did this thing happen | date-time | True

Source:
```
```{eval-rst}
.. jsonschema-titles:: _static/example_schema.json
   :child: subthings
```

Output:
Description of SubThing

<table>
<thead>
<tr>
<th>Title</th>
<th>Description</th>
<th>Type</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>SubThing:Identifier</td>
<td>An identifier for this subthing</td>
<td>string</td>
<td>True</td>
</tr>
<tr>
<td>SubThing:Example</td>
<td>An example field</td>
<td>string</td>
<td>False</td>
</tr>
</tbody>
</table>

.. jsonschema-title-fieldname-map::
Display the mapping between titles and field names.
Currently only used for 360Giving.

Source:
```
```{eval-rst}
.. jsonschema-title-fieldname-map:: _static/example_schema.json
```

Output:

<table>
<thead>
<tr>
<th>Title</th>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identifier</td>
<td>id</td>
<td>string</td>
</tr>
<tr>
<td>Foo bar</td>
<td>foo</td>
<td>integer</td>
</tr>
<tr>
<td>When</td>
<td>when</td>
<td>date-time</td>
</tr>
<tr>
<td>Lorem ipsum</td>
<td>lorem</td>
<td>string</td>
</tr>
<tr>
<td>SubThing:Identifier</td>
<td>subthings/0/id</td>
<td>string</td>
</tr>
<tr>
<td>SubThing:Example</td>
<td>subthings/0/example</td>
<td>string</td>
</tr>
</tbody>
</table>

.. jsonschema-array::
Handle a jsonschema where the top element is an array. Don’t display titles in the table.
Currently only used for OpenReferral.

Source:

```{eval-rst}
.. jsonschema-array:: _static/example_schema_array.json
```

Output:

<table>
<thead>
<tr>
<th>Description</th>
<th>Type</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>string</td>
<td>True</td>
</tr>
<tr>
<td>foo</td>
<td>integer</td>
<td>False</td>
</tr>
<tr>
<td>when</td>
<td>date-time</td>
<td>True</td>
</tr>
<tr>
<td>lorem</td>
<td>string</td>
<td>False</td>
</tr>
<tr>
<td>subthings/0/id</td>
<td>string</td>
<td>True</td>
</tr>
<tr>
<td>subthings/0/example</td>
<td>string</td>
<td>False</td>
</tr>
</tbody>
</table>

3.1 CSS

To display the tables better, add this css to your site: `jsonschema.css`. You need to add the file to a `_static` folder within your docs, and then add the following to `_templates/layout.html`.

```{% extends "!layout.html" %}
{% set css_files = css_files + ["_static/jsonschema.css"] %}
```
.. csv-table-no-translate::
   :like csv-table, but the text inside the table is not translated. Useful when translation is handled by an external process, e.g. codelists.

   included_cols is the optional list of indices of columns to include.

.. markdown::
   :like markdown, using Recommonmark.

   Source:

   ```
   .. markdown::
      :class: hint

      .. admonition:: Worked example
         :class: hint

      .. markdown::

         Some markdown [a URL](http://example.org), `single backtick literals`.
   ```

   Output:

   Worked example
Some markdown a URL, single backtick literals.

.. directory_list::
   Return a bullet list for files in a directory.

   path is the path of the directory.
   url is a url prefix to form the links

Example:

Source:
```
```
.. directory_list::
   :path: exampledir
   :url: https://github.com/OpenDataServices/sphinxcontrib-opendataservices/blob/master/docs/exampledir/
```

Output:
```
• file1.txt
• file2.txt
```

.. localization-note::
   Create a note admonition that only will appear in languages that have “translated” it. This will not appear in the base language (normally English). If a translator wants to mark they have seen the message but do not want to add a note then they can leave a single hyphen ‘-‘. The contents of the translation will be treated as markdown. The text within the directive should contain information useful for the translator and instruct what to do when they encounter this. For example:
```
```
.. localization-note::

   DO NOT TRANSLATE THIS MESSAGE DIRECTLY

   Instead put some language specific context as to how to interpret this page.

   Put a '-' if you do not want this note to appear in this language.
```
```
```
4.1 Directives in other repositories

- `ocds_sphinx_directives` contains extensions that are specific to OCDS docs sites. Currently they all relate to extensions.
- OpenReferral’s JSON Table Schema include, because this is the only docs site we maintain that uses JSON Table Schema.
INDEX

C
csv-table-no-translate (directive), 15

D
directory_list (directive), 16

J
jsoninclude (directive), 5
jsoninclude-flat (directive), 8
jsonschema (directive), 11
jsonschema-array (directive), 13
jsonschema-title-fieldname-map (directive), 13
jsonschema-titles (directive), 12

L
localization-note (directive), 16

M
markdown (directive), 15