

YAML Support in Oxygen

Mircea Enachescu, Syncro Soft
mircea@oxygenxml.com

All rights reserved

SHELL
PHP
JS DOCKERFILE
CSS JSONL
JSON schema
LESS
YAML
JSON
POWER SHELL

Agenda

- **YAML** language
- **Editing** YAML documents
- **Validating** YAML documents
- **Tools** for YAML
- **OpenAPI/AsyncAPI** Support
- Future Plans



Oxygen YAML Support

- YAML support is available in:
 - Oxygen XML Editor
 - Oxygen XML Developer
 - Oxygen JSON Editor



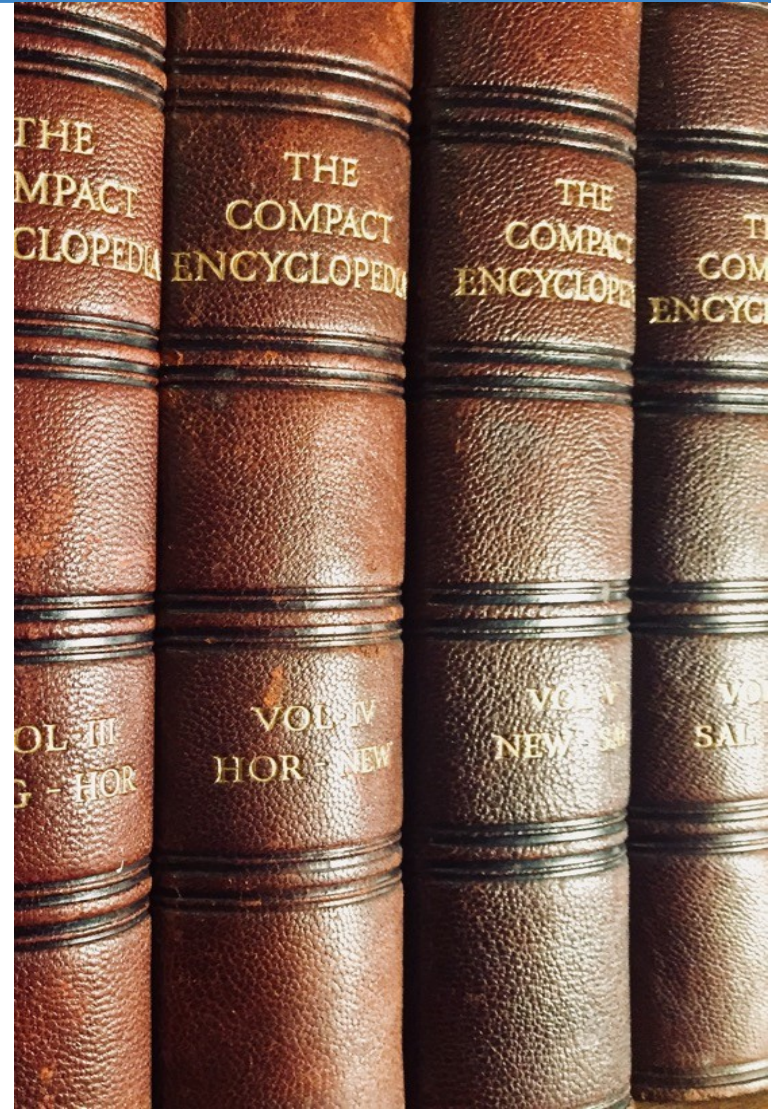
YAML

(Yet Another Markup Language)

(YAML Ain't Markup Language)

- First proposed by Clark Evans in 2001
- Human-friendly data serialization language
- Used for API and Configs
- Easy to read and understand

<https://yaml.org>



YAML Specification Versions

- YAML 1.0 - 29 January 2004
- YAML 1.1 - 18 January 2005
- YAML 1.2.0 - 21 July 2009
- YAML 1.2.1 - 1 October 2009
- YAML 1.2.2 - 1 October 2021



JSON files are valid YAML 1.2

The YAML specification is often seen as overly complicated for something which appears to be so simple.

YAML Syntax

- File extension `.yaml` or `.yml`
- Whitespace indentation for block collections
- Mappings use a colon and space “: “
- Sequences starts with a dash and space “- “
- Comments begin with an hash (#)

```
personnel:  
  person:  
  - id: one.worker  
    name:  
      family: Worker  
      given: One  
    email: one@oxygenxml.com  
#    link:  
#      manager: Big.Boss  
  ..
```

YAML Data Types

- Scalars
 - String: *Hello*, *'Hello'*, *"Hello"*
 - Number (integer or floating): *23*, *1.56*
 - Boolean: *true*, *false*
- Sequence – list of values:
 - fruits:*
 - *apple*
 - *banana*
 - *orange*
- Map – key-value pair: *name: John*
- Null – represents the absence of value: *null*



YAML vs Other Languages

YAML

```
personnel:  
  person:  
    - id: one.worker  
      name:  
        family: Worker  
        given: One  
      email: one@oxygenxml.com  
      link:  
        manager: Big.Boss
```

JSON

```
{  
  "personnel": {  
    "person": [  
      {  
        "id": "one.worker",  
        "name": {  
          "family": "Worker.One",  
          "given": "One"  
        },  
        "email": "one@oxygenxmlcom",  
        "link": {"manager": "Big.Boss"}  
      }  
    ]  
  }  
}
```

XML

```
<personnel>  
  <person>  
    <array>  
      <id>one.worker</id>  
      <name>  
        <family>Worker.One</family>  
        <given>One</given>  
      </name>  
      <email>one@oxygenxmlcom</email>  
      <link>  
        <manager>Big.Boss</manager>  
      </link>  
    </array>  
  </person>  
</personnel>
```

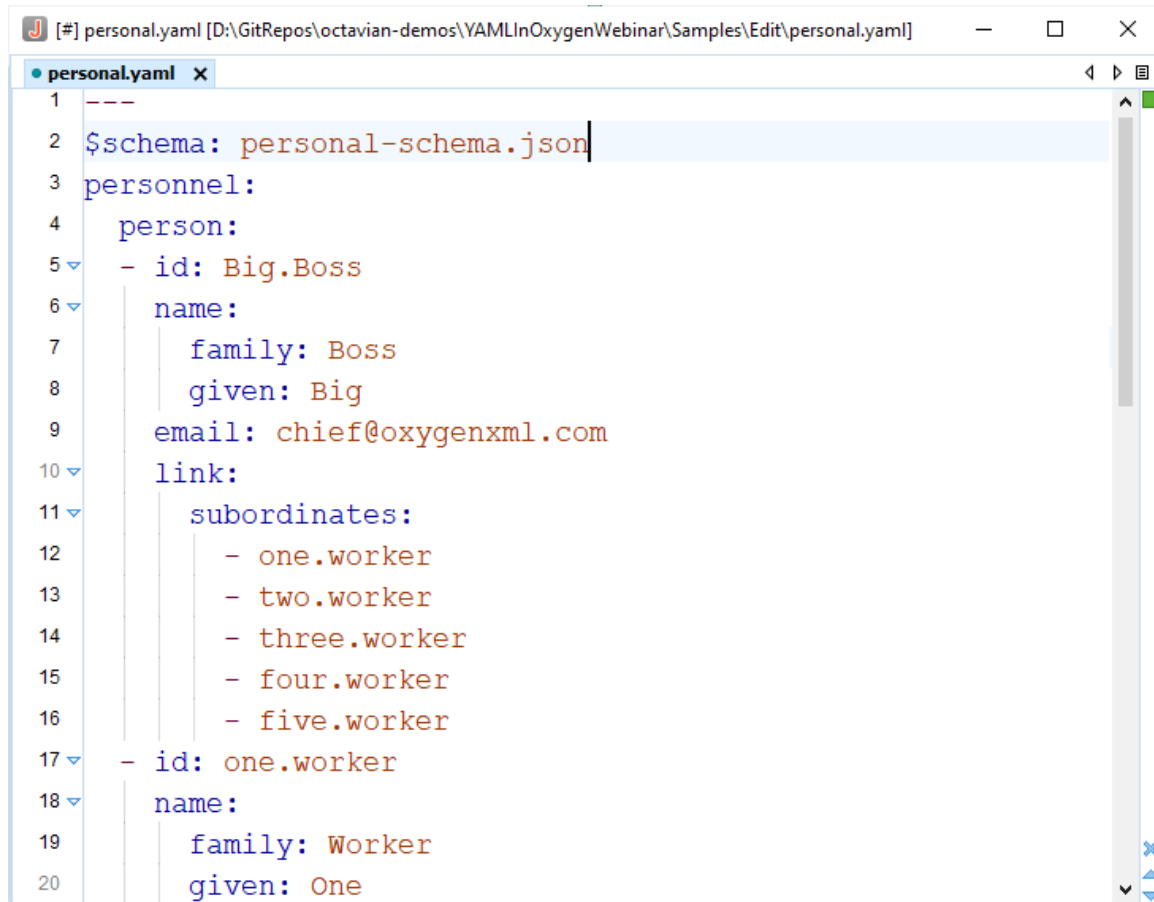

Editing YAML

- YAML Editor
- YAML Outline View
- Validation and Well-Formedness check
- Content Completion Assistant based on JSON Schema



YAML Editor

- Syntax Highlights
- Structure Folding
- Format and Indent
- New Document Templates

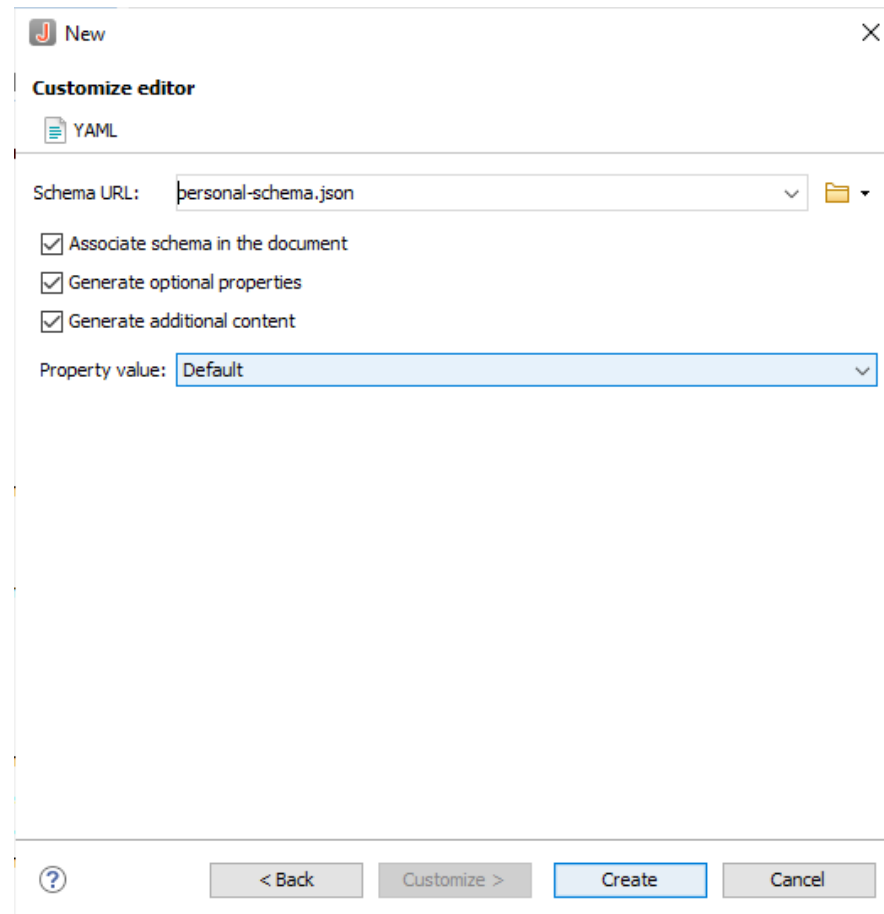
A screenshot of the Oxygen XML Editor interface. The window title is "[#] personal.yaml [D:\GitRepos\octavian-demos\YAMLInOxygenWebinar\Samples\Edit\personal.yaml]". The editor shows a YAML document with the following content:

```
1 ---
2 $schema: personal-schema.json
3 personnel:
4   person:
5     - id: Big.Boss
6       name:
7         family: Boss
8         given: Big
9       email: chief@oxygenxml.com
10      link:
11      subordinates:
12        - one.worker
13        - two.worker
14        - three.worker
15        - four.worker
16        - five.worker
17     - id: one.worker
18       name:
19         family: Worker
20         given: One
```

The editor features syntax highlighting (keywords in blue, strings in orange) and structure folding (indicated by small triangles on the left margin). The current cursor is on line 2.

New Document Templates

- Create new YAML based on JSON Schema
- Generate properties and values
- Create your own custom document templates (save in [OXYGEN_INSTALL_DIR]/templates)

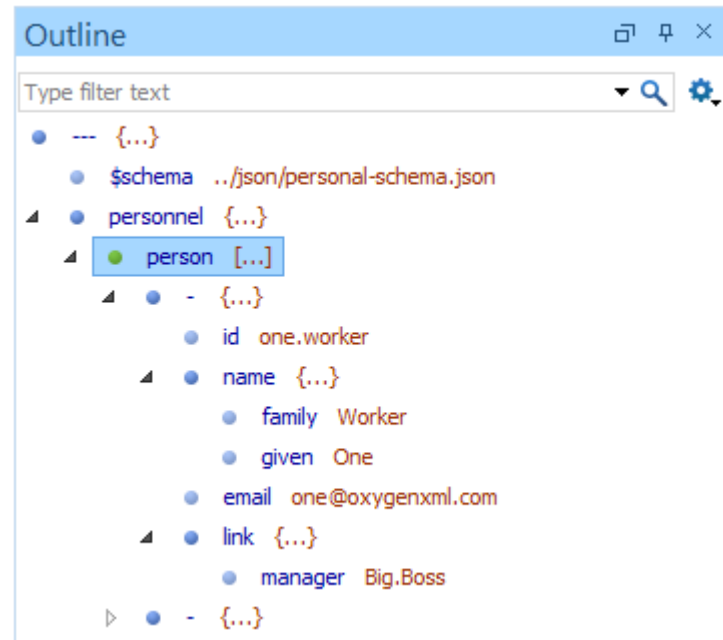


Structure Folding

- Collapse/expand data enclosed between brackets
- Actions:
 - Collapse Other Folds
 - Collapse Child Folds
 - Expand Child Folds
 - Expand All

YAML Outline View

- Document Overview
- Synchronization with Editor
- Filtering
- Error Markers



Associate JSON Schema

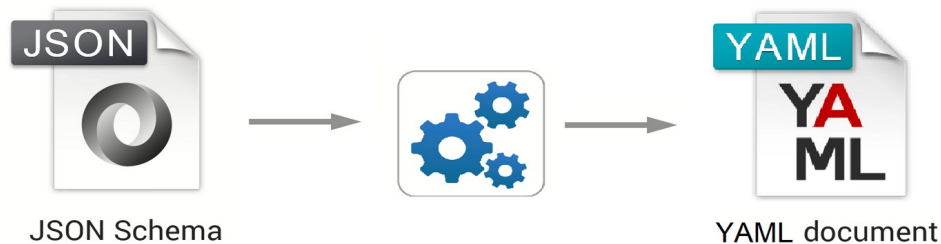
- Associating a Schema to YAML Documents

- Directly in YAML Documents

```
$schema: person-schema.json  
persons: ...
```

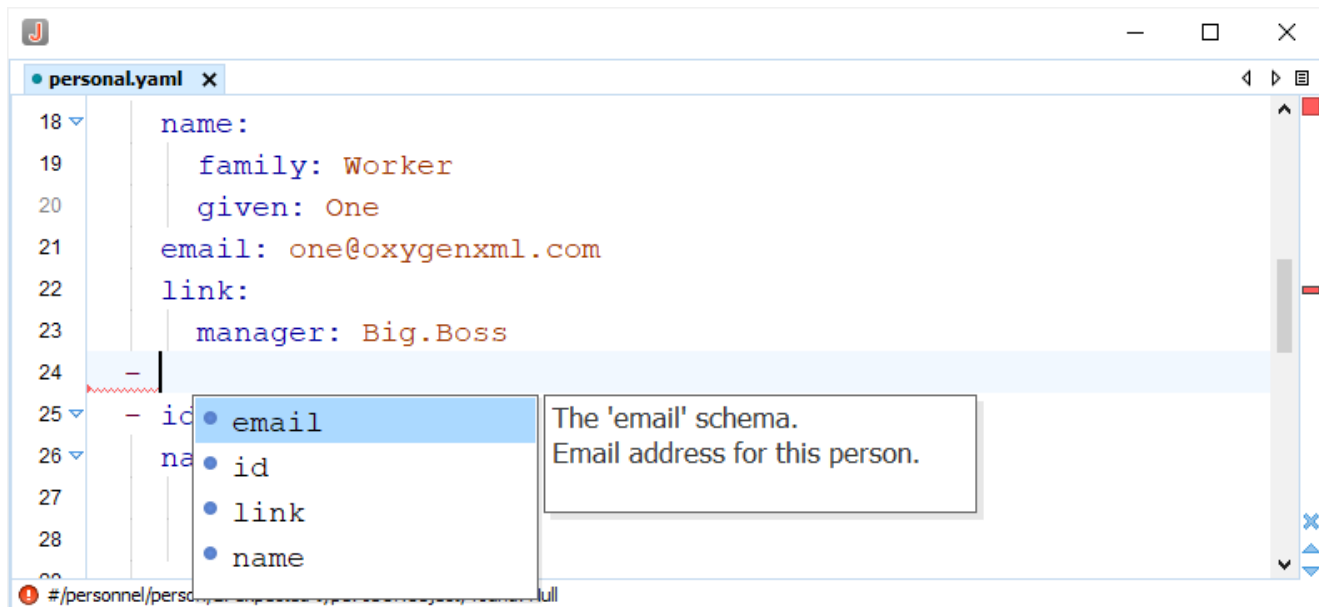
→ Absolute or relative URI

- In application options, through a Validation Scenario



Content Completion

- Context-Sensitive
- Valid Proposals
- Documentation from Schema
- Code Templates



The screenshot shows the Oxygen XML Editor interface with a file named `personal.yaml` open. The editor displays the following YAML content:

```
18 name:
19   family: Worker
20   given: One
21 email: one@oxygenxml.com
22 link:
23   manager: Big.Boss
24 -
25 - id
26 na
27
28
```

A content completion popup is visible over the `- id` line, listing the following options:

- email
- id
- link
- name

A tooltip is displayed next to the `email` option, containing the text: "The 'email' schema. Email address for this person."

The status bar at the bottom left shows the path `#/personnel/persc`.

Validating YAML Documents

- Checking Well-Formedness in YAML documents
- Validating YAML documents against JSON Schema
- Batch validate YAML documents

The screenshot shows the Oxygen XML Editor interface. The main editor window displays a YAML document named 'personal.yaml'. The content is as follows:

```
8     ... given: Big
9     ... email: chief#oxygenxml.com
10
11
12
13
```

A validation error is highlighted in a tooltip and a status bar message:

Validation:
! #/personnel/person/0/email: [chief#oxygenxml.com] is not a valid email address

Press F2 for focus

! #/personnel/person/0/email: [chief#oxygenxml.com] is not a valid email address

The 'Results' panel at the bottom shows the following error details:

Info	Description - 2 items
! -	E [YAML Validator] #/personnel/person/0/email: [chief#oxygenxml.com] is not a valid email address
! -	E [YAML Validator] #/personnel/person/2/name: required key [given] not found

The 'Problems' panel at the bottom shows the error list.

Validating Against JSON Schema

- Associating a JSON Schema
- Automatic Validation
- Manual Validation Actions
- Presenting Validation Errors in YAML Documents

The screenshot shows the Oxygen XML Editor interface with a file named 'personal.yaml'. The editor displays a list of personnel records in a tree structure. The third record, with ID 'three.worker', has a validation error highlighted in red. A tooltip window titled 'Validation:' is open over the error, displaying the message: '#/personnel/person/3: required key [link] not found'. The editor also shows a status bar at the bottom with the same error message. The code in the editor is as follows:

```

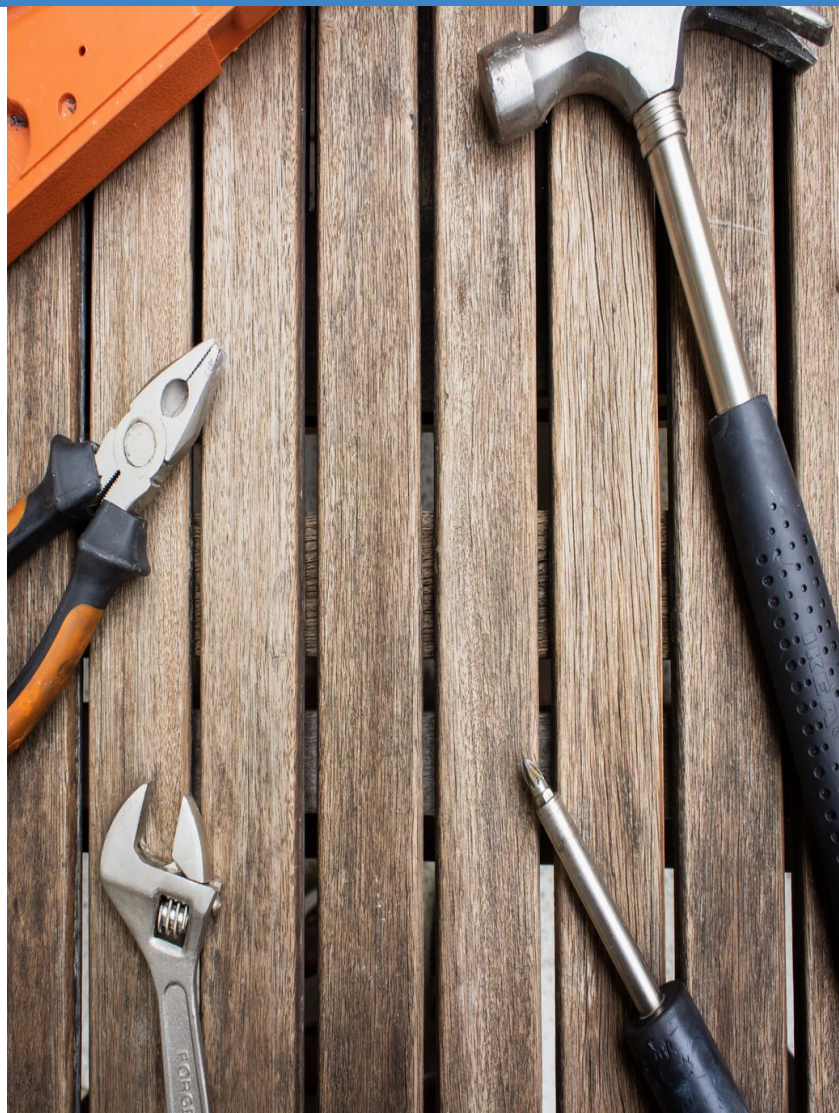
21   ... email: 'one@oxygenxml.com'
22   ... link:
23     ... manager: 'Big.Boss'
24   -- id: 'two.worker'
25   ... name:
26     ... family: 'Worker'
27     ... given: 'Two'
28   ... email: 'two@oxygenxml.com'
29   ... link:
30     ... manager: 'Big.Boss'
31   -- id: 'three.worker'
32   ... name:
33     ... family: 'Worker'
34     ... given: 'Three'
35     ... email: 'three@oxygenxml.com'
36   -- id: 'four.worker'
  
```

Batch Validation

- Check for well-formedness on a project
- Validate multiple YAML documents
- Use Oxygen Scripting for validation on a continuous integration server
https://www.oxygenxml.com/oxygen_scripting.html

YAML Tools

- Convert between [YAML and JSON](#)
- Convert between [YAML and XML](#)
- Batch conversion
 - YAML to JSON
 - JSON to YAML
 - YAML to XML
 - XML to YAML



OpenAPI/AsyncAPI Editor

- Syntax highlights
- Validation conforms with the schema
- Content completion based on the schema
- Outline view
- Search and replace

OpenAPI Documentation

OpenAPI Tester Tool

```
1 ---
2 openapi: 3.1.0
3 info:
4   version: 1.0.0
5   title: Example.com
6   description: This is an example API to demonstrate features of OpenAPI spe
7   termsOfService: https://example.com/terms/
8 contact:
9   email: contact@example.com
10  url: http://example.com/contact
11 license:
12   name: Apache 2.0
13   url: http://www.apache.org/licenses/LICENSE-2.0.html
14 tags:
15 - name: User
16   description: Operations about user
17
18 se
19 -
20 -
21 pa
22 webhooks
23 {}
24 - name: pretty_print
25   in: query
26   description: Pretty print response
27   schema:
28     type: boolean
29 get:
30   tags:
31   - User
```

The incoming webhooks that MAY be received as part of this API and that the API consumer MAY choose to implement. Closely related to the callbacks feature, this section describes requests initiated other than by an API call, for example by an out of band registration. The key name is a unique string to refer to each webhook, while the (optionally referenced) Path Item Object describes a request that may be initiated by the API provider and the expected responses. An example is available.

YAML Frameworks

- Define custom framework for your YAML documents
- Associate JSON Schema for validation and content completion
- Define custom templates

Conclusion

- Smart editing support for YAML
- Validation and content completion using JSON Schema
- Conversion tools
- OpenAPI/AsyncAPI in YAML



Future Plans

- Author page for YAML
- Query YAML documents using XPath
- Transform YAML documents using XSLT



Resources

- <https://yaml.org/spec>
- oxygenxml.com/json_editor/yaml.html
- oxygenxml.com/doc/ug-json/topics/editing-yaml.html
- oxygenxml.com/json_editor/videos.html



Video Demos

- oxygenxml.com/demo/yaml_editing_support_in_oxygen.html
- oxygenxml.com/demo/introducing_the_json_schema_design.html
- oxygenxml.com/demo/json_author.html
- oxygenxml.com/demo/json_tools.html
- oxygenxml.com/demo/json_validation.html
- oxygenxml.com/demo/json_editing.html
- oxygenxml.com/demo/json_query.html



Thank You!

Mircea Enachescu
Product Architect at Syncro Soft

mircea@oxygenxml.com