



API documentation

Cristian Talau

cristi_talau@oxygenxml.com

<https://github.com/ctalau/>

Types of APIs

	REST API	Code API
End-user product	GitHub, Google Drive	Plugin for Firefox
Software component	Spell checking service	React - a web framework

Types of docs - reference guide

Overview Package Class Tree Deprecated Index Help

Package **ro.sync.contentcompletion.xml**

Objects used to offer information about what elements or attributes are allowed at a certain offset in the XML.
See: Description

Interface Summary

Interface	Description
CiAttribute.DefaultValueProvider	Default value provider for an attribute.
CiElement	Interface for objects holding information about element proposals used in the content completion process.
NodeDescription	Node description is in fact a collection of properties for a node.
SchemaManagerFilter	Interface for objects used to filter the editor content completion schema manager proposals.

Class Summary

Class	Description
CiAttribute	Interface for objects holding information about attributes used in the content completion process.
CiElementAdapter	A CiElement adapter.
CiValue	Interface for objects holding information about element or attribute values used in the content completion process.
Context	The context for a node contains: elementStack - the stack with ContextElement up to the root.
ContextElement	Store information about an element inside a context, involved in the content completion process.
ExternalEntityNameValue	A pair class with name and value.
NameValue	A pair class with name and value.
SchemaManagerFilterBase	Base class for objects used to filter the editor content completion schema manager proposals.
StyleGuideSchemaManagerFilterBase	Style guide schema manager filter base.
WhatAttributesCanGoHereContext	Used by the schema manager to find out the attributes that can be inserted in a given context.
WhatElementsCanGoHereContext	It is used to determine the elements that can be inserted in the current context.
WhatPossibleValuesHasAttributeContext	It is used to determine the possible values of the current attribute.

Enum Summary

Enum	Description
CiAttribute.EditableState	The editable state of the attribute.

Package ro.sync.contentcompletion.xml Description

Objects used to offer information about what elements or attributes are allowed at a certain offset in the XML.

POST /api/review/tasks/{id}/comment Adds a comment on a particular review task

GET /api/review/tasks/{id}/download/modified/{filename} Downloads the current content of a particular review task

GET /api/review/tasks/{id}/download/original Download the original version of a particular review task

GET /api/review/tasks/{id}/download/original/zip Download original task

POST /api/review/tasks/{id}/read Marks the actively in particular review task as being read

POST /api/review/tasks/{id}/settings Changes the settings for a particular review task

Parameters

Parameter	Value	Description	Parameter Type	Data Type
body	<input type="text"/>	The settings to be applied	body	Model Example Value
id	(required) <input type="text"/>	id of the review task	path	string

Parameter content type:

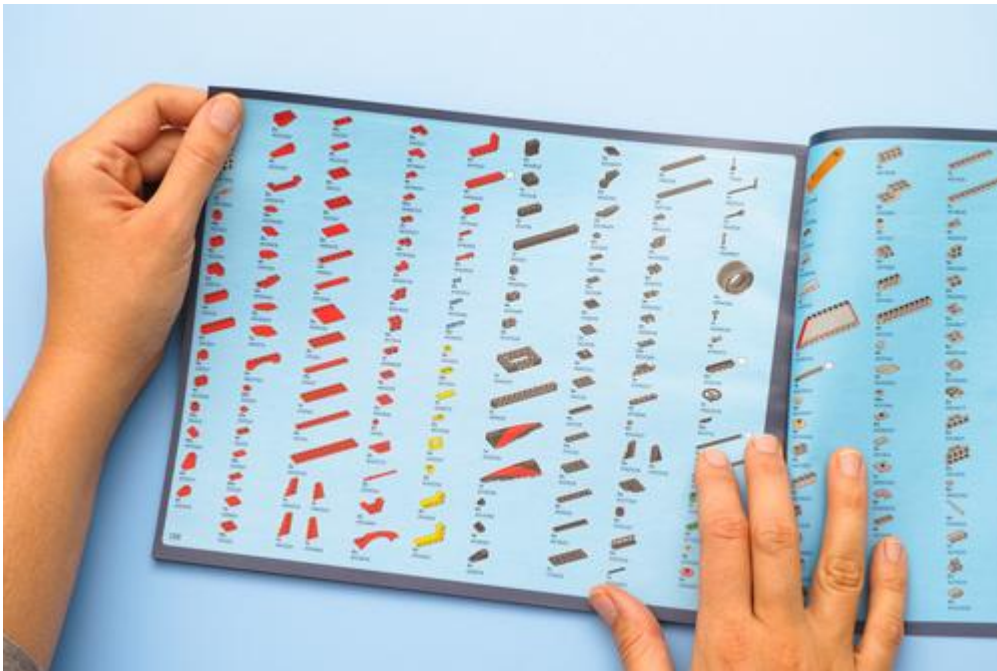
Response Messages

HTTP Status Code	Reason	Response Model	Headers
default	successful operation		

GET /api/review/tasks/{taskId}/files Returns a list of files attached a particular review task

Types of docs - topical guide

- Presents concepts
- Oriented on use-case and on user-profile



vs.



Types of docs - topical guide

- API overview
- API getting started
- API general technical details
 - authentication and authorization
 - status and error codes
 - rate limiting and thresholds
- API quick reference
- API glossary
- API best practices

Types of docs - user manual

- Not applicable for software components
- Targeted to end-users, but developers should also know the concepts

Unified content experience

- Devs need to read
 - User Guide – to know what the product is doing
 - Topical Guide - to find solutions to their problems
 - Reference Guide – to understand the technical details

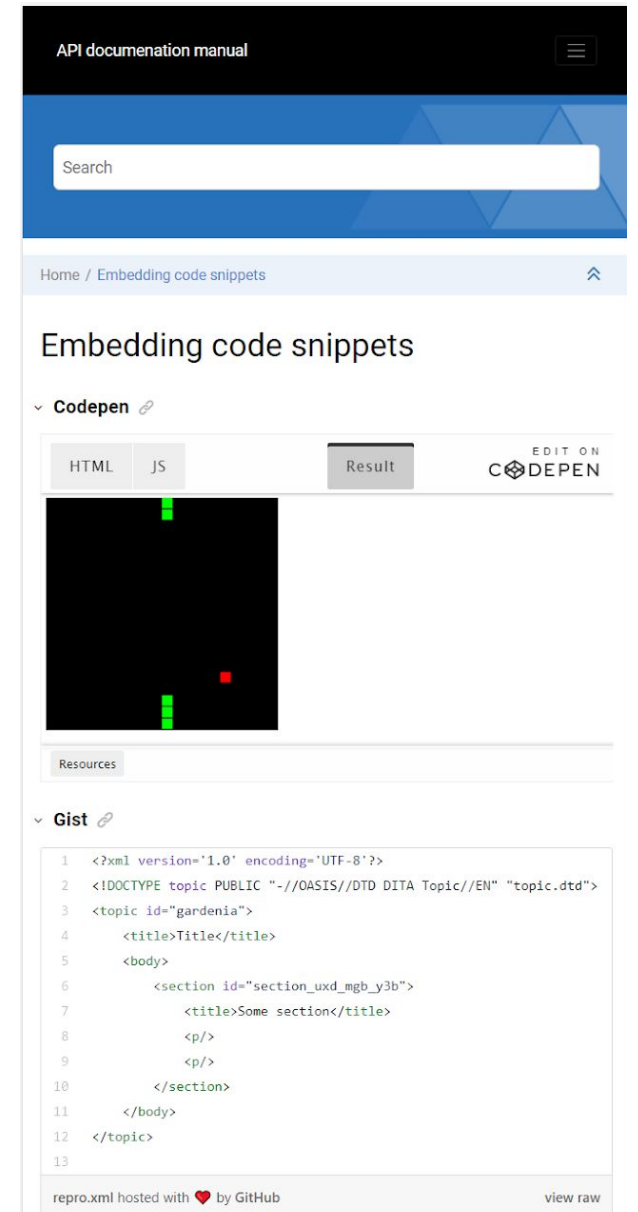
Topical guides

- Markdown
 - Can be used for simple topics
 - Can be converted to DITA to use advanced structure
 - Validation according to styleguide
- DITA
 - Advanced support for tables, definition lists, etc.
 - Special elements for describing programming constructs
 - Can re-use content from user-guide
 - Support for review & change tracking

Topical guides - code

- Multi-language code block
- Interactive code blocks, e.g. CodePen
- Test your code samples!

```
JavaScript Bash
http.request('https://oxygenxml.com/api/login', {
  method: 'POST',
  headers: {'Authorization', 'Bearer ' + personal_access_token}});
```



API documentation manual

Search

Home / Embedding code snippets

Embedding code snippets

Codepen

HTML JS Result EDIT ON CODEPEN

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE topic PUBLIC "-//OASIS//DTD DITA Topic//EN" "topic.dtd">
<topic id="gardenia">
  <title>Title</title>
  <body>
    <section id="section_uxd_mgb_y3b">
      <title>Some section</title>
      <p/>
      <p/>
    </section>
  </body>
</topic>
```

Resources

Gist

```
1 <?xml version='1.0' encoding='UTF-8'?>
2 <!DOCTYPE topic PUBLIC "-//OASIS//DTD DITA Topic//EN" "topic.dtd">
3 <topic id="gardenia">
4   <title>Title</title>
5   <body>
6     <section id="section_uxd_mgb_y3b">
7       <title>Some section</title>
8       <p/>
9       <p/>
10    </section>
11  </body>
12 </topic>
13
```

repro.xml hosted with ❤ by GitHub [view raw](#)

Topical Guides – xrefs to code API

- Code API not deployed yet to its final URL
- Use “apiname” and the link is added automatically
 - At editing time, it points to a staging server
 - When published it points to the production server

```
<apiname outputclass="jsdoc">sync.view.ResourceDragHandler</apiname>
```

Here is a link to the `ro.sync.ecss.extensions.api.AuthorExternalObjectInsertionHandler` API to control the drag and drop behavior. You can also use the `sync.view.ResourceDragHandler` API if you control the source of the drag and drop operation.

Click to open:
<https://www.oxygenxml.com/maven/com/oxygenxml/oxygen-webapp/21.1.1.0/jsdoc/sync.view.ResourceDragHandler.html>

Reference guide

- Generated: OpenAPI, JavaDoc
 - API playground
 - Always up-to-date with code
- Hand-written: DITA, Markdown
 - Richer content model: e.g. notes
 - Consistent with other guides
 - API playground
 - with a DITA specialization

Javadoc links to topical guide

- Don't hardcode links to generated HTML
 - The topical guide is not in its final location
 - The topical guide might be moved at some point
 - The publishing format might be changed
- Use a proxy redirect gate
 - [https://oxygenxml.com/help.php?
pageId=webauthor-accessibility&
product=webauthor&
versions=21.1.1](https://oxygenxml.com/help.php?pageId=webauthor-accessibility&product=webauthor&versions=21.1.1)

OpenAPI editing

- OpenAPI is the standard for REST APIs
- What to touch?
 - “description”, “summary”
 - External documentation link

```
1 {
2   "paths": {
3     "/api/author/files": {
4       "get": {
5         "tags": ["task"],
6         "summary": "Get the content of a file.",
7         "description": "Get the content of a file.",
8         "operationId": "getContent",
9         "parameters": [
10          {
11            "name": "url",
12            "in": "query",
13            "required": false,
14            "type": "string"
15          }
16        ],
17        "responses": {
18          "default": {"description": "successful operation"}
19        }
20      },
21      "put": {
22        "tags": ["task"],
23        "summary": "Get the content of a file.",
24        "description": "Returns the content of the file",
25        "operationId": "setContent",
26        "produces": ["application/json"],
27        "parameters": [
28          {
29            "name": "url",
30            "in": "query",
31            "required": false,
32            "type": "string"
33          }
34        ],
35        "responses": {
36          "default": {"description": "successful operation"}
37        }
38      }
39    },
40    "/api/author/files/makeReviewable": {
41      "post": {
42        "tags": ["task"],
43        "operationId": "makeFileReviewable",
44        "consumes": ["application/json"],
45        "produces": ["application/json"],
46        "parameters": [
47          {
48            "in": "body",
49            "name": "body",
50            "required": false,
51            "schema": {"$ref": "#/definitions/FileLocationParams"}
52          }
53        ]
54      }
55    }
56  }
57 }
```

```
//*[local-name() = 'description' or local-name() = 'summary']
```

OpenAPI editing

- Configurable linter based on Schematron

```
"put": {  
  "tags": ["task"],  
  "summary": "Get the content of a file.",  
  "description": "returns the content of the file",  
  "operation": {  
    "parameters": {  
      "name": "url",  
      "in": "query",  
      "required": false,  
      "type": "string"  
    }  
  },  
  "responses": {  
    "default": {"description": "successful operation"}  
  }  
}
```

Validation:

❗ REST endpoint description should start with capital letter
Press F2 for focus

```
<sch:assert test="./description">  
  REST endpoint description is missing  
</sch:assert>  
<sch:assert test="matches(./text(), '^[A-Z].*')">  
  REST endpoint description should start with capital letter  
</sch:assert>  
<sch:assert test="./summary">  
  REST endpoint summary is missing  
</sch:assert>
```

Document REST API in DITA

- Use a DITA specialization to
 - Preserve semantics
 - Generate nice-looking output

GET /api/users/promote

curl -H "Authorization: Bearer <token>" https://fusion.oxygenxml.com/api/users/

Request Body

- userId *String*
The ID of the user to promote

Home / Fetch details about the current user

GET /api/users/me

Fetch details about the current user such as name, email and avatar.

Request

```
curl -H "Authorization: Bearer <ACCESS_TOKEN>" https://fusion.oxygenxml.com/api/users/me
```

Response

200

Sample response

```
{
  username: 'ctalau',
  displayName: 'Cristian Talau',
  user_id: 'cfbec0067fcfbec0067f',
  email: 'cristian_talau@oxygenxml.com',
  admin: true
}
```

Fields

Name	Type	Description
username	String	The user's handle
displayName	String	The user's name in a form suitable for display in the UI
user_id	String	An unique ID associated with the user Note: This ID is different for each OAuth application.
email	String	The primary email address of an user
admin	Boolean	Flag that indicates whether the user is an administrator

Include OpenAPI in a DITA project

- The source of truth is still Open API
 - Do not edit generated files
 - Use cross references and stable IDs
- Links
 - Links and content references from DITA topics to generated topics
 - Relationship tables
 - Conref push

Import OpenAPI in DITA

- Convert to DITA
- Need to update each time the API is extended
 - 3 way comparison: your version, v1 generated, v2 generated

THANK YOU!

Any questions?

Cristian Talau

cristi_talau@oxygenxml.com

<https://github.com/ctalau>