XSL API for Tag Management

Overview

With the release of Tag Management in OU Campus version 10.4, the CMS has the ability to store tag data for any file on the staging server. While other portions of the CMS are able to access these tags from within the interface, other steps are needed in order to output any tag data to a published page on the production server.

Using XSL, users can perform `doc()` calls to several API calls in order to pull this data onto a published page. This page will detail the API calls and provide a sample output for each call. A `doc()` call allows XSL to retrieve data from an external source. To learn more about this function, visit the Transforming External Data Sources into XSL page.

Inside the `doc()` call, the path of the API call is entered, along with specific URL query strings that allow values to be passed (such as the name of a tag or file). For example:

```
doc('ou:/Tag/GetTags?site=[site-name]&amp;path=[encoded-path-to-file]')
```

The available calls are as follows:

Tag-Related Calls

These calls return data for a tag or tags that satisfy the parameters specified by the call. The output of these doc calls will be an XML file that has the following structure:

```
<tags>
<tag>
<name>tagName</name>
<disabled>true|false</disabled>
<children>[integer value]</children>
</tag>
<tag>
...
</tag>
</tags>
```

The `<name>` node lists the name of the tag, `<disabled>` notes whether the tag has been disabled from the list view in **Setup > Tags**, and `<children>` denotes whether the tag has any children, which makes it a collection. If the value inside `<children>` is `0`, the tag is not a collection. If the value is an integer `n` greater than `0`, the tag is a collection and contains `n` tags.

- GetAllTags
- GetTags
- GetFixedTags
- GetCombinedTags
- GetParents
- GetChildren

GetAllTags

This call returns all tags that have been created in the site.

Example

```
<xsl:copy-of select="doc('ou:/Tag/GetAllTags')/tags" />
```

Output

```
<tags>
<tag>
<name>academics</name>
<disabled>false</disabled>
<children>0</children>
</tag>
<tag>
<name>athletics</name>
<disabled>false</disabled>
<children>3</children>
</tag>
<tag>
...
</tag>
</tags>
```

GetTags

This call returns the tags that have been manually applied to a particular page. This does not include any fixed tags that are enforced at the directory level.

The URL requires the site name and the root-relative path to the file on staging to be passed to it. Any `/` in the path of the file must be properly encoded as `%2F`.
Use

`&`
to separare parameters (e.g. between

`site=`
and `path=`

).

Example

```
<xsl:copy-of select="doc('ou:/Tag/GetTags?site=gallena&amp;path=%2Ftag-
management%2Ftagdemo.pcf')/tags" />
```

Output

```
<tags>
<tag>
<name>testTag</name>
<disabled>false</disabled>
<children>0</children>
</tag>
<tag>
...
</tag>
</tags>
```

GetFixedTags

This call returns the tags that have been applied to a particular directory through Tag Access Settings.

The URL requires the site name and the root-relative path to the directory on staging to be passed to it. Any `/` in the path must be properly encoded as `%2F`.
Use

`&`
to separare parameters (e.g. between

`site=`
and `path=`

).

Example

```
<xsl:copy-of select="doc('ou:/Tag/GetFixedTags?site=gallena&amp;path=
%2Ftag-management')/tags" />
```

Output

```
<tags>
<tag>
<name>fixedTag</name>
<disabled>false</disabled>
<children>0</children>
</tag>
<tag>
...
</tag>
</tags>
```

GetCombinedTags

This call returns the combination of the manually-applied tags and fixed tags for a given file (essentially combining GetTags and GetFixedTags for a given file).

The URL requires the site name and the root-relative path to the file on staging to be passed to it. Any `/` in the path must be properly encoded as `%2F`.
Use

`&`
to separare parameters (e.g. between

`site=`
and `path=`

).

Example

```
<xsl:copy-of select="doc('ou:/Tag/GetCombinedTags?site=gallena&amp;path=
%2Ftag-management%2Ftagdemo.pcf')/tags" />
```

Output

```
<tags>
<tag>
<name>fixedTag</name>
<disabled>false</disabled>
<children>0</children>
</tag>
<tag>
<name>testTag</name>
<disabled>false</disabled>
<children>0</children>
<tag>
...
</tag>
</tags>
```

GetParents

This call returns any collections that a given tag belongs to.

The URL requires the name of the tag to be passed to it.

Example

```
<xsl:copy-of select="doc('ou:/Tag/GetParents?tag=basketball')/tags" />
```

Output

```
<tags>
<tag>
<name>athletics</name>
<disabled>false</disabled>
<children>3</children>
</tag>
</tags>
```

GetChildren

This call returns any children of a given collection.

The URL requires the name of the collection to be passed to it.

Example

```
<xsl:copy-of select="doc('ou:/Tag/GetChildren?tag=athletics')/tags" />
```

Output

```
<tags>
<tag>
<name>baseball</name>
<disabled>false</disabled>
<children>0</children>
</tag>
<tag>
<name>basketball</name>
<disabled>false</disabled>
<children>0</children>
</tag>
<tag>
<name>soccer</name>
<disabled>false</disabled>
<children>0</children>
</tag>
</tags>
```

File-Related Calls

These calls return data for a file or files that satisfy the parameters specified by the call. The output of these doc calls will be an XML file that has the following structure:

```
<pages>
<page>
<path>/root/relative/path/to/staging-file.pcf</path>
<last-pub-date>[date and time stamp]</last-pub-date>
</page>
<page>
...
</page>
</pages>
```

- GetFilesWithAnyTags
- GetFilesWithAllTags

GetFilesWithAnyTags

This call returns any file that contains one or more of the listed tags.

The URL requires the site name and the list of tags to be passed to it. Each tag in the list must be declared separately with its own `tag=` preceding it

. Use

`&`
to separare parameters (e.g. between

`site=`
and the first

`tag=`
, and between

`tag=`
parameters).

Example

```
<xsl:copy-of select="doc('ou:/Tag/GetFilesWithAnyTags?
site=gallena&amp;tag=basketball&amp;tag=soccer')/pages" />
```

Output

```
<pages>
<page>
<path>/athletics/index.pcf</path>
<last-pub-date>2016-06-28T17:24:13Z</last-pub-date>
</page>
<page>
<path>/athletics/mens-basketball.pcf</path>
<last-pub-date>2016-06-28T17:24:13Z</last-pub-date>
</page>
<page>
<path>/athletics/womens-soccer.pcf</path>
<last-pub-date>2016-06-28T17:24:13Z</last-pub-date>
</page>
<page>
<path>/blog/inter-athletics-olympics.pcf</path>
<last-pub-date>2016-06-28T17:24:13Z</last-pub-date>
</page>
</pages>
```

GetFilesWithAllTags

This call returns any file that contains all of the listed tags.

The URL requires the site name and the list of tags to be passed to it. Each tag in the list must be declared separately with its own `tag=` preceding it

. Use `&` to separare parameters (e.g. between `site=` and the first `tag=`, and between `tag=` parameters).

Example

```
<xsl:copy-of select="doc('ou:/Tag/GetFilesWithAllTags?
site=gallena&amp;tag=basketball&amp;tag=soccer')/pages" />
```

Output

```
<pages>
<page>
<path>/athletics/index.pcf</path>
<last-pub-date>2016-06-28T17:24:13Z</last-pub-date>
</page>
<page>
<path>/blog/inter-athletics-olympics.pcf</path>
<last-pub-date>2016-06-28T17:24:13Z</last-pub-date>
</page>
</pages>
```

Using Functions to Call the APIs

While the above API calls can be called directly in the XSL using the `doc()` function, users may find that process longer and more difficult to configure in their site. An alternative method to do this is to make each API call a funciton, which allows for parameters to be passed into the query string of the API call, allowing for more dynamic use of the calls.

Take the following function as an example. The below code serves as a replacement for the direct GetPageTags API call:

```
<!-- PARAMETERS: None -->
<xsl:function name="ou:get-page-tags">
<xsl:sequence select="ou:get-page-tags($ou:stagingpath, $ou:site)" />
</xsl:function>
<!-- PARAMETERS: Staging Path -->
<xsl:function name="ou:get-page-tags">
<xsl:param name="path" />
<xsl:sequence select="ou:get-page-tags($path, $ou:site)" />
</xsl:function>
<!-- PARAMETERS: Staging Path, Site -->
<xsl:function name="ou:get-page-tags">
<xsl:param name="path" />
<xsl:param name="site" />
<xsl:variable name="tags-api" select="concat('ou:/Tag/GetTags?site=',
$site,'&amp;path=', encode-for-uri($path))" />
<xsl:copy-of select="ou:get-tags($tags-api)" />
</xsl:function>
<!-- TAGS API DOC CALL -->
<xsl:function name="ou:get-tags">
<xsl:param name="api" />
<xsl:try>
<xsl:sequence select="doc($api)/tags" />
<xsl:catch>
<tags></tags>
</xsl:catch>
</xsl:try>
</xsl:function>
```

Using this function, when a user wants to call GetPageTags, all they need to write is

```
<xsl:copy-of select="ou:get-page-tags()" />
```

as opposed to

```
<xsl:copy-of select="doc('ou:/Tag/GetTags?site=gallena&amp;path=%2Ftag-
management%2Ftagdemo.pcf')/tags" />
```

The function also has the added benefit of the ability to pass up to two parameters in order to get tags from a different page or site, which would have to be hard-coded if the API call was perfomed through a `doc()` call.

Download this ZIP file for a sample set of functions that correspond to each API call, as well as some other practice materials.