Introduction
IBM Web Content Manager (WCM) provides a rich content experience for producing, managing and displaying web content. Requirements can often be met using the core product capabilities and typically very little coding, however a full web API is available for addressing more repetitive tasks or complex situations.

This article and associated sample package (SampleWCMAPI.paa) has been put together to show and describe how IBM Web Content Manager can be used programmatically for fetching and displaying content using commonly available programming techniques.

Programmatic approaches to using web content can often be found in mobile based applications and single page websites/applications, these samples help you understand how to use WCM in these use cases.

The samples show how WCM can be used in single page applications (SPA) where WCM can generate the list of links and the content itself.

The first sample shows a standard WCM menu generating links which are then used to update the content programmatically and without a page refresh.
The second example shows using a WCM menu which generates JSON, which is then used to produce the list of links and again the content is updated in the same way. The content item in these first two samples is retrieved via the Connect servlet directly.

The final sample doesn’t use any WCM components, just the WCM REST API. This can be used to create a query for the content and then fetch the content itself. When retrieving content using the WCM REST API it just returns the data and doesn’t render the content. So additional calls may be needed to for example if the element is a component reference.
Prerequisites

In order to install and take advantage of the samples you will need to have the following software installed:

- IBM Web Content Manager v8.x. For installation steps please refer to the official documentation instructions, a trial version is also available from here: https://ibm.biz/BdEbZN

- IBM Script Portlet application, see Appendix

Where the explicitly stated, IBM WebSphere Portal v8.5 CF05 is required. This is predominantly for the new Content as a Service (CaaS) capability introduced in v8.5 CF05 during March 2015.

Optional

Developing with JSON can be made simpler through the use of developer debugger plugins. Several plugins are available and in the development of these samples the RESTClient for Firefox and Postman for Google Chrome have proven to be very useful.

- Postman - https://chrome.google.com/webstore/detail/postman-rest-client/dfmmgjlnipnjgjioojodkiooicmcm

AJAX with Web Content Manager Standard Menu

The first sample shows how to use a menu to generate a list of items and then using jQuery programmatically get and set the content without doing a page refresh (the same approach could be taken using the framework of your choice). A standard WCM menu is used to generate the links which then trigger an XHR to get the content as JSON. The content is then dynamically added to the page.

This sample is setup by calling the HTML component from the Web Content Viewer application, the container component includes:

- jQuery JavaScript library
- CSS styling for positioning the results
- javascript function
- WCM menu for the content results

![Diagram](image)

*Figure 1 – Sample 1, component relationship.*

WCM uses Menu components to return content based upon a user defined configuration. In this situation the menu component returns all nested content items that are using:

- Authoring Templates = /SPA/Event
- Location = /SPA/SPA Demo

The menu returns mark-up in the form of an unordered list <ul>. Each lineitem <li> formatted to return the path to the content item (sitepath) and the content items name (title). The ‘a href’ is wrapped with a call to the javascript function .

Example, SPA/SPA+Demo/Exceptional+Digital+Expereinece+Conference

The getContent function is called each time a content item is clicked, requesting through an $.ajax (jQuery) call GET and URL the content items. The URL consists of the path to the WCM connect servlet '/wps/wcm/myconnect/' the sitepath as described earlier and lastly using a WCM query parameter (?presentationtemplate=) the format for the results and the parameter (&subtype=json) to get the content as JSON.

Example, /wps/wcm/myconnect/SPA/SPA+Demo/Exceptional+Digital+Expereinece+Conference?presentationtemplate=WCM+SPA/JSON&subtype=json
The Presentation Template uses the standard WCM tag library to result the results in a JSON format, for example the name key value pair is 'name': followed by the WCM Property tag of the current content items title field. The summary, body and imageUri are similar but use different WCM tags and formatting.

```json
{
    "name": "[Property context="current" type="content" htmlencode="true" field="title"]",
    "summary": "[Plugin:CopyText count="1" format="trim" escape="json" text="[Element context='current' type='content' key='Summary']"]",
    "body": "[Plugin:CopyText count="1" format="trim" escape="json" text="[Element context='current' type='content' key='Body']"]",
    "imageUri": "[Plugin:CopyText count="1" format="trim" escape="json" text="[Element context='current' type='content' key='Library Index Image' format='url']"]"
}
```

The JSON payload is returned to the getContent function which parses the results updating the <div id=contentbody> and <div id=contentsummary> in the container HTML component.

**Performance benefits can be gained through using jQuery as it provides default caching of XHR requests.**
AJAX with explicit JSON payload management

The second sample shows how to generate JSON from a menu and then use this data to generate the list programmatically. It uses many of the same elements of the first sample but a Script Portlet has been used instead of the Web Content Viewer application and also just a single WCM Menu component.

![Sequence Diagram](image)

**Figure 2 – Sample 2, sequence diagram.**

The JavaScript code requests the menu component JSON Event Menu and returns the queried contents in JSON format as explicitly described, the ‘subtype=json’ WCM parameter sets the correct content type for the payload.

JSON has been used for the results in this step (the event menu) for flexibility and reuse. The WCM menu could have returned HTML however JSON is used for easier programmatic iteration and general post processing.

The menu will return a key value pair of and using the WCM placeholder tag="title" and tag="sitepath" tags. As in the previous sample the configuration of the menu query is the same as we want to show the same content.

```
{
  "entry": [{"name":"[Placeholder tag="title"]",
  "link":"[Placeholder tag="sitepath"]"}]
}
```

Each item returned in the JSON payload an unordered list is created, each line item being an ‘a href’ call to the function updateContent followed by the URL and content item we just retrieved

Example,

```
{"name":"Exceptional Digital Experience,
  "link":"/SPA/SPA+Demo/ Exceptional+Digital+Expereinece+Conference "}
```
Finally, the function call is identical to the previous samples `getContent` and updates the `<div id=contentbody>` and `<div id=contentsummary>` in the container HTML when an item is clicked.

This sample uses the Script Portlet application. To take advantage of WCM syndication the page where the script portlet is placed should be linked with a site area in your library. For example the `/SPA library`.

By default the script portlet artefacts are created in the Portal home library.
Web Content Manager REST API

The third sample shows using the WCM REST API to programmatically retrieve the content items and details rather than using a standard menu and the Connect servlet.

It uses a defined query to fetch all of the content created with a specific authoring template. The REST API is geared towards content creation and this example is provided to show how you could achieve the same results as in samples 1 and 2 using just the API.

As with the previous sample this third samples uses some of the same elements, including the HTML, CSS and some of the JavaScript functions.

Figure 3 - Sample 3, sequence diagram.

The code initially requests the id of the Event authoring template so that in the next step all content created with this template's id can be returned in a JSON formatted payload.

```json
{
    "feed": {
        "id": "wcmrest:query?name=Event&pragma=no-cache&type=ContentTemplate",
        "title": "wcmrest:query?name=Event&pragma=no-cache&type=ContentTemplate",
        "updated": "Mon, 02 Mar 2015 15:20:56.786Z",
        "entry": [
            {
                "id": "wcmrest:69daf411-9d84-48f1-97e8-c22ff2ac6b68",
                "title": 
```
A WCM REST query is used to return all the results for content items that have been created with the Event content template using the id= wcmrest:69daf411-9d84-48f1-97e8-c22ff2ac6b68. The results will be in JSON format.

```json
{
    "feed": {
        "id": "wcmrest:query?authoringtemplateid=wcmrest%3A69daf411-9d84-48f1-97e8-c22ff2ac6b68&pagesize=30",
        "title": "wcmrest:query?authoringtemplateid=wcmrest%3A69daf411-9d84-48f1-97e8-c22ff2ac6b68&pagesize=30",
        "updated": "Mon, 02 Mar 2015 15:20:57.119Z",
        "link": [
            {
                "rel": "next-page",
                "href": "/wps/mycontenthandler/!ut/p/digest!g08WqJSVlyOLOHmznFfnaA/wcmrest/query?pagesize=30&authoringtemplateid=wcmrest%3A69daf411-9d84-48f1-97e8-c22ff2ac6b68&page=2",
                "lang": "en",
                "label": "Next Page"
            }
        ],
        "entry": [
            {
                "id": "wcmrest:6059d298-8c2a-4974-8302-4b3a219589d8",
                "title": {
                    "lang": "en",
                    "value": "SpeCTCular"
                },
                "summary": {
                    "lang": "en"
                },
                "name": "SpeCTCular 2013",
                "type": "Content",
                "updated": "Wed, 17 Dec 2014 07:59:44.204Z",
                "author": [
                    {
                        "distinguishedName": "uid=virtuser,o=defaultWIMFileBasedRealm",
                        "uri": "/wps/mycontenthandler/!ut/p/digest!g08WqJSVlyOLOHmznFfnaA/um/users/profiles/29eAe3R003QOCNPC8MM4C09P4JMG643E4JM07L9C6MM8C6JDCMG17H9D2MR86G1",
                        "name": "virtuser"
                    }
                ]
            }
        ]
    }
}
```

IBM Web Content Manager, programmatically using content as a service.
As with Sample 2 for each item returned in the JSON payload an unordered list is created, each line item being an ‘a href’ call to the function updateContent followed by the URL and content item name we just retrieved.

Example, 
{"name":"Exceptional Digital Experience, "link":"/SPA/SPA+Demo/ Exceptional+Digital+Expereinece+Conference "} 

Finally, the function updateContent is called to retrieve the content item’s contents when the link title is click. This is different to the previous sample as the JSON payload contains more value pairs than before, also the previous samples JSON was explicitly controlled with WCM tags.

The individual content item is requested using the ajax GET request /wps/mycontenthandler/wcmrest/content/’ + id[1], the id array containing ‘wcmrest: ’, once again the result returned in JSON.

Iterating through the JSON returned for the content item the Body, Summary and Library Index Image are returned.

A secondary request for an image is required if a Library Index Image is found.

💡 When using the REST API if the content item contains a component reference, for example a file or image an additional REST call is required to return the final component URL.

💡 A useful feature of the REST API is that it provides for content paging by default. Using other mechanisms would require the use of WCM paging components.

The <div id=contentbody> and <div id=contentsummary> in the container HTML is updated when an item link is clicked.
Content as a Service (Caas)

This feature capability was introduced into the continuous deliver stream CF05 for IBM WebSphere Portal v8.5, in order to use this your system needs to be at this level. If you are on a lower level of v8.5 then you can upgrade simply using the Installation Manager (automated download and install) or via FixCentral (manual).
Appendix

IBM Script Portlet Application


JQuery Learn Centre - http://learn.jquery.com/

REST Service for Web Content Manager

http://www-01.ibm.com/support/knowledgecenter/SSDK36_8.5.0/wcm/wcm_rest.dita

WCM REST Component Formats

http://www-01.ibm.com/support/knowledgecenter/SSDK36_8.5.0/wcm/wcm_rest_content_formats.html

Authors

– IBM Digital Experience Strategist

IBM Certified L2 Senior IT Specialist, IBM Collaboration Solutions