Introduction

In Getting Started: Understanding the Response, we reviewed the basic structure of a JSON output and hypermedia link. In this article, we will share a few options to programmatically consume the JSON response from `/mdr/products/Terminology`.

Python

JSON is native to Python. Working with CDISC Library’s JSON outputs is fairly straightforward. This code show how to iterate through all the hypermedia links and simply dumping out the content:

```python
import requests
import json

baseURL = "http://library.cdisc.org/api"
queryEndpoint = "/mdr/products"
queryPath = "/Terminology"

r = requests.get(baseURL + queryEndpoint + queryPath, headers={'api-key': 'abcdef0123456789abcdef0123456789', 'Accept': 'application/json'})

if r.status_code == 200:
    content_dict = json.loads(r.text)
    for package in content_dict['_links']['packages']:
        print("%s is a %s link for %s" % (package['href'], package['type'], package['title']))
```

A sample output:

```text
/mdr/ct/packages/adamct-2014-09-26 is a Terminology link for ADaM Controlled Terminology Package 19 Effective 2014-09-26
/mdr/ct/packages/adamct-2015-12-18 is a Terminology link for ADaM Controlled Terminology Package 24 Effective 2015-12-18
...
/mdr/ct/packages/sendct-2018-12-21 is a Terminology link for SEND Controlled Terminology Package 36 Effective 2018-12-21
```

JSONPath

Another way is to use JSONPath. JSONPath is very similar to XPath (for XML) in many ways that allow you to query a JSON structure. The expression to obtain all the hrefs from the output:

```json
$_links.packages.[href]
```

This is an excerpt of the JSONPath result:

```json
[  
"/mdr/ct/packages/adamct-2014-09-26",
"/mdr/ct/packages/adamct-2015-12-18",
...
"/mdr/ct/packages/sendct-2018-12-21"
]
```

Further Readings

JSONPath Reference. Smartbear. [https://support.smartbear.com/readyapi/docs/testing/jsonpath-reference.html](https://support.smartbear.com/readyapi/docs/testing/jsonpath-reference.html)

We will be adding code snippets in other languages.