#

OneDep Biocuration Content Web Service Interface

Introduction

The OneDep content web service provides selective access to content within the Biocuration system designed to support data exchange between wwPDB OneDep and federated partner sites.

This service is provided as a Python package. Once installed, access is provided using a command line client or pragmatically using a Python API. Access to these services requires an API key which can be requested by e-mail to site hosting the API web service.

Installation

Installation is via the program pip (https://pypi.python.org/pypi/pip) .

```
pip install onedep_biocuration_api
```

Command Line Usage

A binary script is installed called **`onedep_request`** which is a front end that invokes the packages APIs.

Start a new session

You must start a new session. The system keeps track of the current working session in the file **`~/.onedep_biocuration_current_session`** This can be changed with the *--session_file* argument.

```
onedep_request --new_session
```

Request entry reports

Initiate an entry report request with the command:

```
onedep_request --entry_id <data set id D_0000000000>  --entry_content_type  <report content type>

onedep_request --entry_id D_0000000000 --entry_content_type  'report-entry-example-test'
```

You can periodically check if the request is complete with the command:

```
onedep_request --status
```

or if you would like a value of 0 or 1:

```
onedep_request --test_complete
```

Retrieving results

When complete, you can retrieve the report results using:

```
onedep_request --output_file <json file>    --output_type <report_content_type>
onedep_request --output_file myreport.json  --output_type 'report-entry-example-test'
```

Python API

A Python API provides simple access to the content request web services in a programmatic manner. The ContentRequest class provides access to all services provided by the API including: session creation, entry and summary content requests, api completion status retrieval, and report download.

Entry content reports

The following example illustrates how to use the API to request content about individual OneDep entries. For convenience, two environmental variables are provided for setting the URL for the service provider, ONEDEP_BIOCURATION_API_URL, and the local path to the API key, ONEDEP_BIOCURATION_API_KEY_PATH.

```
from onedep_biocuration import __apiUrl__
from onedep_biocuration.api.ContentRequest import ContentRequest

def requestEntryContent(requestEntryId, requestContentType, requestFormatType, resultFilePath):
    """ Example of OneDep API content request.

        :param string requestEntryId :   the entry deposition data set identifier (D_0000000000)
        :param string requestContentType : the request content type
        :param string requestFormatType :  the request format type
        :param string resultFilePaht :  result file path for the requested content
```

```
    """
    #
    print_("Example content request service for: \n +entry   %s\n +content type %s\n" % (requestEntryId,
requestContentType))
    #
    # Check for alternative URL and KEY settings in the environment -
    #
    apiUrl = os.getenv("ONEDEP_BIOCURATION_API_URL") if os.getenv("ONEDEP_BIOCURATION_API_URL") else
__apiUrl__
    keyFilePath = os.getenv("ONEDEP_BIOCURATION_API_KEY_PATH") if
os.getenv("ONEDEP_BIOCURATION_API_KEY_PATH") else "~/.onedep_biocuration_apikey.jwt"
    apiKey = readApiKey(keyFilePath)
    cr = ContentRequest(apiKey=apiKey, apiUrl=apiUrl)

    #
    # Create a new service session -
    #
    print_("Creating new session for content request example\n")
    rD = cr.newSession()
    displayStatus(rD)
    #
    #
    print_("Request entry identifier %s content type %s\n" % (requestEntryId, requestContentType))
    #
    # Submit service request

    print_("Submitted content service request\n")
    rD = cr.requestEntryContent(requestEntryId, requestContentType, requestFormatType)
    displayStatus(rD)
    #
    #   Poll for service completion -
    #
    it = 0
    sl = 2
    while (True):
        #    Pause -
        it += 1
        pause = it * it * sl
```

```
        time.sleep(pause)
        rD = cr.getStatus()
        if rD['status'] in ['completed', 'failed']:
            break
        print_("[%4d] Pausing for %4d (seconds)\n" % (it, pause))
        #
    #
    print_("Storing content type %s  in result file %s\n" % (requestContentType, resultFilePath))
    rD = cr.getOutputByType(resultFilePath, requestContentType, formatType=requestFormatType)
    displayStatus(rD)
    print_("Completed\n")
```