Research Use Case

Use Case #1 - Downloading data files

The ONC researcher community is very diverse, with each researcher having a different set of questions that they want to be able to answer using ONC data. In order to answer those questions, they need to be able find out what data is available, what data products are available for it and be able to download it in a format they can use.

A typical workflow might be as follows

1. Identify the type of instrument(s) that observe the phenomenon of interest (ie, water temperature or pressure)
2. Narrow down the geographic area and time period
3. Identify the data products and file formats that meet their research needs
4. Identify the data product options (like QAQC or Resampling) that need to be applied to the data
5. Download the data products as files
6. Perform analysis on the data
7. Repeat the process for another area, time range or instrument

The following provides a sample python script with 2 classes with command line commands to support the following workflow

1. createXLS class - Used to create an Excel spreadsheet of data product download definitions for Locations and/or Devices, that meets the filter criteria.
    a. Each line represents a unique data product delivery request.
    b. Contains a tab for configurations, where the researcher can modify the specific parameters used to run the download process, such file output location, size of queue, number of download threads or how to break up the requests into smaller pieces.
    c. Researcher can manually modify the spreadsheet to further refine the download criteria
    d. Can be called from the command line using downloadONC.py createXLS.

2. download class - Used to generate and download all of the data product files defined in the Excel spreadsheet.
    a. The class methods will read configurations and data product definitions from the Excel spreadsheet
    b. A json file, containing the configuration and download progress information is created the first time it is run and is updated with the download progress results.
    c. The json file is updated when the Excel spreadsheet is updated.
    d. Multithreaded - uses a request queue, allowing multiple request and downloads to run at the same time.
    e. Script is rerunnable - It will pick up the process from the last time it was saved.
    f. Can be called from the command line using downloadONC.py download.

downloadONC.py classes and commands

Requirements

Requires the Python Client Library

Script

downloadONC.py

createXLS class & command

The purpose of this class & command is to discover ONC data for download and save the data product download input to a spreadsheet, which can be manually modified and then used with the download class or command to download the data files and output the download information and status into a download process log file.

CLI Usage

downloadONC.py createXLS -t <token> -o <outpath> -b <begin> -e <end> [options]

parameter	Description
Required	
-h, --help	Show Help
-t, --token <token>	Once logged in at https://data.oceannetworks.ca/login, your personal token can be retrieved or generated at https://data.oceannetworks.ca/Profile. Click on the "Web Services" tab, then click "Generate Token".
-o, --outpath <outpath>	The folder that file(s) will be saved to
-b, --begin <begin>	The beginning date/time of the data you would like to download
-e, --end <end>	The ending date/time of the data you would like to download
[options]	
-a, --append	Append configurations to an existing spreadsheet if it already exists, otherwise, create a new spreadsheet.
-f, --file <filename>	The destination Excel spreadsheet file name. If omitted, the default file name is <outpath>\download.xlxs
-m, --downloadMethod <byLocation|byDevice|all>	The method for downloading data. Data can be downloaded by location and/or device. If excluded, the 'byLocation' method is used.
-q, --qa	Use the QA Server. For Internal Use ONLY.
Filters	
-l, --locationCode <locationCode>	The code for the parent location you would like to use to search for instruments from. • You can obtain a locationCode from the Available Locations page. • or from the location filter in the Data Search URL when you navigate to a location of interest in https://data.oceannetworks.ca/DataSearch • or from the https://data.oceannetworks.ca/api/locations web service.
-c, --deviceCategoryCode <deviceCategoryCode>	The code for the device category of the instrument you would like to download data from. • You can obtain a deviceCategoryCode from the Available Device Categories page. • or from the deviceCategory filter in the Data Search URL when you navigate to a instrument of interest in https://data.oceannetworks.ca/DataSearch • or from the https://data.oceannetworks.ca/api/deviceCategories web service.
-d, --deviceCode <deviceCode>	The code for a specific device you would like to download data from • You can obtain a deviceCode from the Available Devices page. • or by clicking on the Details link for a list in the popup when you make a selection in https://data.oceannetworks.ca/DataSearch • or from the https://data.oceannetworks.ca/api/devices web service
-v, --propertyCode <propertyCode>	The code for the property you would like to download data for • You can obtain a property code from the Available Properties page. • or from the https://data.oceannetworks.ca/api/properties web service. ◦ Use filters to return only those properties that are supported on the devices or instruments of the device category, you are interested in.
-p, --dataProductCode <dataProductCode>	The code for the data product you would like to download • You can obtain a data product code from the Available Data Products page. • or from the https://data.oceannetworks.ca/api/dataProducts web service. ◦ Use filters to return only those data products that are available for the devices or instruments of a device category, you are interested in.

-x, --extension <extension>	The file extension for the data product you would like to download • You can obtain a list of extensions available for a data product from the Available Data Products page. • or from the https://data.oceannetworks.ca/api/dataProducts web service. ◦ Use filters to return only those extensions that are available for the data product you are interested in.

Example

```
c:\onc>downloadONC.py createXLS --token=YOUR_TOKEN_HERE --outPath="c:/ONC/Data" --locationCode=USDDL --
deviceCategoryCode=CTD --begin="2017-05-01T00:00:00.000Z" --end="2017-06-01T00:00:00.000Z"
```

Python Example

```
from downloadONC import createXLS

x = createXLS('<YOUR_TOKEN_HERE>',"c:/ONC/Data")

x.locationCode='USDDL'
x.deviceCategoryCode='CTD'
x.begin='2017-05-01T00:00:00.000Z'
x.end='2017-06-01T00:00:00.000Z'

x.createWorkbook('C:/ONC/Data/download.xlsx')
```

Results

When the command or class is successfully run, a spreadsheet is generated which contains 2 or more tabs, depending on the value of the -m, --downloadMethod option. If the -a, --append option is used, the spreadsheet will be updated with any new locations or devices that meet the filter criteria.

byLocation

If the -m, --downloadMethod option is excluded, equals byLocation or all, the 'By Location' tab is added to the generated spreadsheet. The tab contains a row for each unique ***locationCode***, ***deviceCategoryCode***, ***propertyCode***, deployment ***begin*** and **end** datetime, ***dataProductCode***, ***extension***, ***data product options*** which matches the filter criteria. Each row represents a unique data product request, and each bolded column represents a dataProductDelivery service request filter. When the download command/class is run, each row with a *download* value = T, will generate one or more data product delivery request, depending on the splitDateRange value in the configuration tab. In addition to the columns used by the data product delivery request, there are a number of additional metadata columns, which provide additional information about the instrument deployment, to assist research and validation. These columns are not bolded and include locationName, deviceCategoryName, dataProductName, helpDocument link and Data Search link. Each deployment will have a deviceCode, latitude, longitude and depth along with a Device Listing link. The 'data product options' column my be blank when the row is created. It MUST be populated manually in order for the data product to be successfully requested. Please ensure that it is populated using information obtained from the helpDocument link. If is not populated correctly, the request will fail when the download command/class is run, and produce a message in the command output and save it to the json log file (see logFile value in the configuration tab). The values in all bolded columns can be modified prior to running the download command/class, to better suit your download needs. In addition, unwanted columns can be manually deleted and new columns can be created as long as all of the required columns (bolded) are populated.

column	Description	Example
locationCode	The code for the location to download data for.	BACAX
locationName	The name of the Location.	Axis
deviceCategoryCode	The code for the device category of the instrument you would like to download data from.	HYDROPHONE
deviceCategoryName	The name of the Device Category.	Hydrophone
propertyCode	The code for the property you would like to download data for. This	
download	Indicates if the row is to be downloaded.	Y
begin	The beginning date/time of the deployment.	2011-07-13 0:00:00
end	The ending date/time of the deployment.	2011-07-14 0:00:00
deviceCode	The code for a specific deployed device.	NAXYS_HYD_007

latitude	The latitude of the deployment.	48.316758
longitude	The longitude of the deployment.	-126.050183
depth	The depth of the deployment.	981
dataProductCode	The code for the data product to be downloaded.	AD
dataProductName	The name of the data product to be downloaded.	Audio Data
extension	The file format of the data product to be downloaded.	wav
data product options	The data product options for the data product request. *This column may be blank by default. Please ensure that it is populated by the appropriate data product options obtained using the helpDocument link.	dpo_hydrophoneDataDiversionMode=All
helpDocument	A link to the Data Product help document.	https://wiki.oceannetworks.ca/display/DP/7
Data Search	A link to the Data Search page for the location.	http://qa.oceannetworks.ca/DataSearch?location=BACAX&deviceCategory=HYDROPHONE
Device Listing	A link to the Device Listing for the deployed device.	http://qa.oceannetworks.ca/DeviceListing?DeviceId=11207

byDevice

If the -m, --downloadMethod option equals byDevice or all, the 'By Device' tab is added to the generated spreadsheet. The tab contains a row for each unique ***deviceCode***, ***propertyCode***, deployment ***begin*** and **end** datetime, ***dataProductCode***, ***extension***, ***data product options*** which matches the filter criteria. Each row represents a unique data product request, and each bolded column represents a dataProductDelivery service request filter. When the download command/class is run, each row with a *download* value = T, will generate one or more data product delivery request, depending on the splitDateRange value in the configuration tab. In addition to the columns used by the data product delivery request, there are a number of additional metadata columns, which provide additional information about the instrument deployment, to assist research and validation. These columns are not bolded and include deviceName, dataProductName, helpDocument link and Device List link. Each deployment will have a locationCode, latitude, longitude and depth. The 'data product options' column my be blank when the row is created. It MUST be populated manually in order for the data product to be successfully requested. Please ensure that it is populated using information obtained from the helpDocument link. If is not populated correctly, the request will fail when the download command/class is run, and produce a message in the command output and it save to the json log file (see logFile value in the configuration tab). The values in all bolded columns can be modified prior to running the download command/class, to better suit your download needs. In addition, unwanted columns can be manually deleted and new columns can be created as long as all of the required columns (bolded) are populated.

column	Description	Example
deviceCode	The code for the device to download data for.	NAXYS_HYD_007
deviceName	The name of the device.	Naxys Hydrophone 02345 (S/N 007)
propertyCode	The code for the property you would like to download data for.	
download	Indicates if the row is to be downloaded.	Y
begin	The beginning date/time of the deployment.	2011-07-13 0:00:00
end	The ending date/time of the deployment.	2011-07-14 0:00:00
locationCode	The location code of the deployment.	BACAX
latitude	The latitude of the deployment.	48.316758
longitude	The longitude of the deployment.	-126.050183
depth	The depth of the deployment.	981
dataProductCode	The code for the data product to be downloaded.	AD
dataProductName	The name of the data product to be downloaded	Audio Data
extension	The file format of the data product to be downloaded.	wav
data product options	The data product options for the data product request.	dpo_hydrophoneDataDiversionMode=All
helpDocument	A link to the Data Product help document.	https://wiki.oceannetworks.ca/display/DP/7
Device Listing	A link to the Device Listing for the deployed device.	http://qa.oceannetworks.ca/DeviceListing?DeviceId=11207

Configuration

The Configuration tab stores all of the configurable settings used by the download command/class when it is run. The splitDateRange configuration allows you to split the data product requests into small chunks to keep processing time more managable for larger and more complex data products. When using the splitDateRange = week, set the start day using the dayOfWeek configuration. The runThreads configuration controls how many data product request the code can have in the task queue or running at any given time. Currently, 4 concurrent requests per user should not clog up the queue. Please be aware that increasing this number can and will have significant impact on the system health and processing wait times of other users and processes. The downloadThreads configuration controls how many concurrent file download requests the code will perform at any given time. The beginRun and endRun configurations control when, during the day, the code will make data product requests and make download data file rquests. If you want to only run the download between 10pm and 6am, set the values to beginRun=22:00:00 and endRun=06:00:00. By default the values are beginRun=00:00:00 and endRun=00:00:00 which allows the code to run immediately.

column	Description	Example
token	The user authentication token that the web services will be run under. Once logged in at https://data.oceannetworks.ca/login, your token can be retrieved or generated at https://data.oceannetworks.ca/Profile. Click on the "Web Services" tab, then click "Generate Token".	<YOUR_TOKEN>
downloadFolder	The folder that the data files will be saved to.	C:/ONC/Data/hydrophones
logFile	The configuration file used to create download definitions and track progress.	C:/ONC/Data/hydrophones/download.json
splitDateRange	Used to determine how the requests are split up by date range. Valid: all, year, month, week, day	all
dayOfWeek	MO, TU, WE, TH, FR, SA, SU - Only used if requestTimeSpan=week	SU
runThreads	The size of the queue and number of threads that can be used for running data product requests.	4
downloadThreads	The number of threads that can be used for downloading data on.	10
downloadQueueSize	The size of the queue for downloading files. unlimited can grow to any size, however, the number of concurrent requests is determined by the number of download threads.	unlimited
beginRun	Time to begin adding run tasks to the queue.	00:00:00
endRun	Time to end adding run tasks to the queue	00:00:00

download class / command

The purpose of this class / command is to read data product requests from an Excel spreadsheet, create a json log file, run them, download the files, and save the process back to the file. The process runs on multiple threads for running and polling the request status while simultaneously downloading files on a pool of different threads. The process log file can be manually updated to include new locations, device categories and date ranges, and then the runDownloadProcessLogUpdate.py can be run to create new download processes, which will be picked up, next time this script is run.

CLI Usage:

downloadONC.py download -f <file>

parameter	Description
Required	
-f, --file <file>	The data product definition json file name
[Options]	
-h, --help	Show Help
-q, --qa	Use the QA Server. For External Use ONLY.

Example:

```
$ downloadONC.py download --file="C:/temp/download/download.xlsx"
```

Python Example:

```
from downloadONC import download
d = download()
d.execute('C:/temp/download/download.xlsx')
```

Examples

A typical whale researcher wants to be able to use the audio files from all of the ONC hydrophones in the North East Pacific to generate spectra for a complete migration cycle.

Example

1. Root ONC Tree node location code: NEP - North East Pacific
2. Device Category: HYDROPHONE
3. Data Product: AD - Audio Data
4. File format: mp3
5. Date Range: June 1st, 2016 to May 31st, 2017

```
$downloadONC.py createXLS --token=YOUR_TOKEN_HERE --outPath="c:/temp" --locationCode=NEP --
deviceCategoryCode=HYDROPHONE --dataProductCode=AD --extension=mp3 --begin="2016-06-01T00:00:00.000Z" --end="
2017-05-31T23:59:59.999Z"
```

A typical Marine Geoscience Researcher needs to be able to download all of the data, for all of the instruments at a specific location in MatLab format.

1. Root ONC Tree node location code: LSDDL- Lower Slope Delta Dynamics Laboratory
2. Device Category: all
3. Data Product: all
4. File format: mat
5. Date Range: Oct 23, 2013 - May 31st, 2017

```
$ downloadONC.py createXLS --token=YOUR_TOKEN_HERE --outPath="c:/temp" --locationCode=LSDDL --extension=mat --
begin="2013-20-23T00:00:00.000Z" --end="2017-05-31T23:59:59.999Z"
```

A typical Ocean and Atmospheric Sciences Researcher wants to be able to download all available scalar data in json, for the Strait of Georgia in json format for the past year.

1. Root ONC Tree node location code: SOG - Strait of Georgia
2. Device Category: all
3. Data Product: TSSD
4. File format: all
5. Date Range: June 1st, 2016 - May 31st, 2017

```
$ downloadONC.py createXLS --token=YOUR_TOKEN_HERE --outPath="c:/temp" --locationCode=SOG --
dataProductCode=TSSD --begin="2016-06-01T00:00:00.000Z" --end="2017-05-31T23:59:59.999Z"
```

A typical video researcher wants to be able to download 1 day's worth of videos and ancillary data from every deployed ONC video camera so that they can validate their algorithms.

1. Root ONC Tree node location code: ONC - Ocean Networks Canada
2. Device Category: VIDEOCAM
3. Data Product: MP4V,TSSD
4. File format: mp4,json
5. Date Range: May 31st, 2017 00:00:00 - 23:59:59

```
$ downloadONC.py createXLS --token=YOUR_TOKEN_HERE --outPath="c:/temp" --locationCode=ONC --
deviceCategoryCode=VIDEOCAM --dataProductCode=MP4V,TSSD --extension=mp4,json --begin="2017-05-31T00:00:00.000Z"
--end="2017-05-31T23:59:59.999Z"
```

Use Case Scenarios

- A researcher wants to discover ONC, finds out about ONC's APIs and starts playing. He wants to build his own web application but doesn't know what is where, so he would like to receive a list of locations and device types with data available in ONC's archive. After discovering hydrophone data in the Arctic he wants to know if there were data from 2012 available, and what other sensors were recording at the same locations. After finding both temperature and ice-thickness data to be of interest, he wants to see what data formats these data could be delivered in. Finally, he wants to download one year's worth of wav hydrophone files, and also temperature and ice-thickness as CSV files.
- A researcher has heard that many fin whale calls have been identified on ONC hydrophone data and would like to download the corresponding spectra, both as data and as PNG plots. Ideally his service can link the annotations to the download request, or else he would need to know the times and locations of the annotated data so his script can download the spectra automatically.
- A researcher operates a mobile platform and wants it to navigate autonomously by having the mobile platform access nearby sonar data on a scheduled request, ideally in real time, in order to check the sonar data for the latest position of the mobile platform. A python script cron job on the mobile platform's system determines when the last sonar scan took place; if it was subsequent to the latest sonar data download, a new sonar scan data download is initiated.
- A researcher has been conducting a long-running experiment in partnership with ONC. He has devised the instrument platform and gradually improved the design over the years. He maintains his own archive of data, which is populated daily by scripts that pull the latest data from his platform. He mashes up ONC data with other data pulled from other sources like Environment Canada and the Pacific Geosciences Centre. He has also created a sort of clunky dashboard, which shows a collection of latest pre-generated plots from ONC Data Preview. He wants his scripts to run without having to be updated after every expedition or configuration change – he doesn't really want to mess with his scripts once they are working. He also wants to know if data get reprocessed by ONC, so he can be sure he's not doing analyses on flawed data.
- A researcher checks daily if his data are showing anything interesting and he has a Matlab script that every midnight automatically downloads the last 24 hours of data so he has them on his local machine when he comes to work every morning. His script does some processing on his machine which he rather would be doing on an ONC server so he doesn't have to download the entire raw data but could download spectra as PNG files or down-sampled data or only those full data snippets associated with detected events. This will enable him to see and react to events such as a large turbidity event so he can mobilize a field survey to collect additional data.
- A researcher wants to see his data as near real-time as possible, both on the monitors in his office as well as on a public display he has running for outreach purposes. He has a python script running every 15 minutes to extract the raw scalar and complex data which then need to be processed and combined with some external data he accesses using ERDDAP, and then turned into to PNG files. This could all be saved if ONC would be able run his python code and ONC had a flexible dashboard that would show both these ONC data as well as non-ONC data. He usually is publishing new versions of his python code in GitHUB.

Please report all issues with the web services, documentation, samples and client libraries to the Oceans 3.0 Help Centre