CRDS Interface

# Introduction

CRDS need to query the archive database for CRDS parameters. They will use this information to reevaluate reference assignments as new reference files arrive in order to determine which filesets should be reprocessed. The Archive will provide a RESTful interface that can be used to query the database.

# Definitions

This project is based upon the functionality that currently exists for HST, but the nomenclature is different for JWST.

FileSetName: The “root” of the filename, without the detector. If a filename does not include a suffix, then the FileSetName is simply the filename without the detector and extension. If a filename does include a suffix, then the FileSetName is the filename without the detector, suffix and extension. For example:

S0000000155\_01001\_03599.ssr => S0000000155\_01001\_03599

jw00002001001\_01101\_00001\_NRCB1\_original.fits => jw00002001001\_01101\_00001

DatasetIds: A datasetId, as used by CRDS, is a combination of the Association (not yet defined in the db) and the FileSetName and detector: “Association:FileSetName.Detector”. If no Association exists then it’s based upon the FileSetName and detector only: “FileSetName.Detector:FileSetName.Detector”. For example

Association exits: Assoc1: jw00035001001\_01101\_00001.MIRIMAGE

No Association: jw00035001001\_01101\_00001.MIRIMAGE : jw00035001001\_01101\_00001.MIRIMAGE

# Functionality

## DatasetIds Count by Instrument

Given an instrument return a count of datasetIds. If a date is also specified, only count datasetIds where the file’s ingestStartDate is greater than the specified date.

Input:

* instrument (required) examples: instrument=miri, instrument=MIRI
* minDate (optional) example: minDate=2015-06-22T15:26:31

Output:

* {“DatasetsCount”:count} example: {"DatasetsCount":200001}

**Note:** Association is not yet defined in the database so association is ignored when calculating the count.

Examples:

* Count unique datasetIds for instrument MIRI:

wget "http://jwdmsdevvm4:8888/crds/datasetsCount?instrument=miri"

returns

{"DatasetsCount":200001}

* Count unique datasetIds for instrument MIRI after 2015-07-11T19:55:00:

wget " http://jwdmsdevvm4:8888/crds/datasetsCount?instrument=miri&minDate=2015-07-11T19:55:00"

returns

{"DatasetsCount":2}

## DatasetIds by Instrument

Given an instrument return a list of datasetIds. If a date is also specified, only return datasetIds where the file’s ingestStartDate is greater than the specified date. An optional batchNum can also be specified, which return rows in the specified batch. If batchNum is not specified then it will return rows from the first batch. If batchNum is larger then the number of batches available then it will return no rows.

**Batching:** In order to prevent a ridiculously large number of datasetIds being returned by this function, it will only return up to maxHeaderBlockSize (see section 3.3) datasetIds. If more than that number of datasetsIds would be returned then the user will need to page through the results. The logic for doing this is as follows:

batchSize = wget maxHeaderBlockSize

numResults = wget datasetsCount

counter = 0

while (batchSize\*counter < numResults) {

wget datasets instrument=X batchNum=counter <minDate=Y>

counter ++

}

Input:

* instrument (required) examples: instrument=miri, instrument=MIRI
* minDate (optional) example: minDate=2015-06-22T15:26:31
* batchNum (optional) example: batchNum=5

Output:

* [“datasetId1”,”datasetId2”,…] example: ["ASSOCIATION:jw00001001001\_01101\_00001.MIRIMAGE","ASSOCIATION:jw00035001001\_01101\_00001.MIRIMAGE"]

**Note:** Association is not yet defined in the database so the example above has association hardcoded to “ASSOCIATION”

Examples:

**Note:** The code is still under construction, so the way to call this function may change in the future.

* Get first batch of datasetIds for instrument MIRI:

wget "http://jwdmsdevvm4:8888/crds/datasets?instrument=miri"

returns

["ASSOCIATION:jw00001001001\_01101\_00001.MIRIMAGE","ASSOCIATION:jw00035001001\_01101\_00001.MIRIMAGE"]

* Get first batch of datasetIds for instrument MIRI after 2015-06-22T15:26:31:

wget "http://jwdmsdevvm4:8888/crds/datasets?instrument=miri&minDate=2015-06-22T15:26:31"

returns

["ASSOCIATION:jw00035001001\_01101\_00001.MIRIMAGE"]

* Get first batch of datasetIds for instrument BADINSTRUMENT:

wget "http://jwdmsdevvm4:8888/crds/datasets?instrument=BADINSTRUMENT"

returns

[]

* Get first batch of datasetIds for instrument MIRI after 2015-07-11T19:54:54 (Note: in this example maxHeaderBlockSize is 3 and there are a total of 7 datasetIds)

wget "http://jwdmsdevvm4:8888/crds/datasets? instrument=miri&minDate=2015-07-11T19:54:54&batchNum=0"

returns

["ASSOCIATION:jw00001001001\_01101\_00001.MIRIMAGE","ASSOCIATION:jw00025001001\_01107\_49995.MIRIMAGE","ASSOCIATION:jw00025001001\_01107\_49996.MIRIMAGE"]

* Get second batch of datasetIds for instrument MIRI after 2015-07-11T19:54:54

wget "http://jwdmsdevvm4:8888/crds/datasets? instrument=miri&minDate=2015-07-11T19:54:54&batchNum=1"

returns

["ASSOCIATION:jw00025001001\_01107\_49999.MIRIMAGE","ASSOCIATION:jw00025001001\_01107\_49998.MIRIMAGE","ASSOCIATION:jw00025001001\_01107\_49997.MIRIMAGE"]

* Get third batch of datasetIds for instrument MIRI after 2015-07-11T19:54:54

wget "http://jwdmsdevvm4:8888/crds/datasets? instrument=miri&minDate=2015-07-11T19:54:54&batchNum=2"

returns

["ASSOCIATION:jw00025001001\_01107\_50000.MIRIMAGE"]

* Get fourth batch of datasetIds for instrument MIRI after 2015-07-11T19:54:54 (no data)

wget "http://jwdmsdevvm4:8888/crds/datasets? instrument=miri&minDate=2015-07-11T19:54:54&batchNum=3"

returns

[]

## Header rows limit

In order to avoid large queries, the number of datasetIds that can be returned by the Header query (see below) is limited. This value is set in a DADS config file, and can be queried as follows:

Example:

wget http://jwdmsdevvm4:8888/crds/maxHeaderBlockSize

returns

{"MaxHeaderBlockSize":4000}

## Headers by datasetIds

Given a list of datasetIds and parameters, this function will return a list of datasetIds, parameters, and parameter values.

**NOTE:** If the number of datasetIds returned would be larger than the header rows limit (see section 3.2) then it is an error and no rows will be returned.

### Input:

* A file containing the datasetIds and parameters in the following format:

{

"CRDSHeaderQueryInput" : {

"datasetIds" : [

“<FileSetName>.<Detector>”,”<FileSetName>.<Detector?”,…

],

"parameters" : [

"<param1>",”<param2>”,…

]

}

}

There are 3 acceptable formats for datasetId:

1. Association:FileSetName.Detector will return the same Association:FileSetName.Dectector
2. Association will return all FileSetName.Detectors for that association: Association:FileSetName.Detector,…
3. FileSetName.Detector will return Association:FileSetName.Detector if an association exists for this fileSetName.Detector, otherwise it will return FileSetName.Detector:FileSetName.Detector

The parameters are in the format of the data model path, such as meta.instrument.name or meta.exposure.type

Output:

* {“datasetId1”, {“param1”:”value1”, “param2”:”value2”,…}, …} example: {"ASSOCIATION:jw00035001001\_01101\_00001.MIRIMAGE":{"meta.exposure.comprssd":"","foo.bad.field":"UNDEFINED","unit.section\_name.cdelt\_1":"0.11","META.INSTRUMENT.CCC\_STATE":"OPEN"}}
* If the requested parameter doesn’t exist then it will be returned with a value of “UNDEFINED”.
* If the requested parameter exists but the value is NULL it will be returned with a value of “”.
* If the requested datasetId doesn’t exist then it will be returned with an empty parameter list.
* If the requested datasetId is in a bad format then it will be ignored and will not appear in the output.

**NOTE:** We don’t have associations in the database yet, so if a FileSetName.Detector is passed in it will return ASSOCIATION:FileSetName.Detector. If an Association without a FileSetName.Detector is passed in it will be ignored and will not appear in the output. You can see both of these behaviors in the example below. These behaviors will change in a future version of the code once we have associations in the database.

Example:

Input file testinput.json:

{

"CRDSHeaderQueryInput" : {

"datasetIds" : [

"MyAssoc:jw00002001001\_01101\_00001.NRCB1","badfileset.MIRIMAGE","jw00035001001\_01101\_00001.MIRIMAGE","jw00001001001\_01101\_00001.MIRIMAGE","OrphanAssoc","AnotherAssoc:jw00002001001\_01101\_00001.NRCB1"

],

"parameters" : [

"Meta.Exposure.Type","unit.section\_name.cdelt\_1","foo.bad.field","meta.instrument.channel","meta.instrument.detector","META.INSTRUMENT.CCC\_STATE","meta.exposure.comprssd"

]

}

}

wget --post-file=testinput.json --header="Content-type:application/json" http://jwdmsdevvm4:8888/crds/headers

returns

{"MyAssoc:jw00002001001\_01101\_00001.NRCB1":{"meta.exposure.comprssd":"F","meta.instrument.channel":"SHORT","foo.bad.field":"UNDEFINED","unit.section\_name.cdelt\_1":"0.032","META.INSTRUMENT.CCC\_STATE":"UNDEFINED","meta.instrument.detector":"NRCB1","Meta.Exposure.Type":"NRC\_IMAGEa"},"AnotherAssoc:jw00002001001\_01101\_00001.NRCB1":{"meta.exposure.comprssd":"F","meta.instrument.channel":"SHORT","foo.bad.field":"UNDEFINED","unit.section\_name.cdelt\_1":"0.032","META.INSTRUMENT.CCC\_STATE":"UNDEFINED","meta.instrument.detector":"NRCB1","Meta.Exposure.Type":"NRC\_IMAGEa"},"ASSOCIATION:jw00001001001\_01101\_00001.MIRIMAGE":{"meta.exposure.comprssd":"","meta.instrument.channel":"UNDEFINED","foo.bad.field":"UNDEFINED","unit.section\_name.cdelt\_1":"0.11","META.INSTRUMENT.CCC\_STATE":"OPEN","meta.instrument.detector":"MIRIMAGE","Meta.Exposure.Type":"MIR\_IMAGEb"},"ASSOCIATION:jw00035001001\_01101\_00001.MIRIMAGE":{"meta.exposure.comprssd":"","meta.instrument.channel":"UNDEFINED","foo.bad.field":"UNDEFINED","unit.section\_name.cdelt\_1":"0.11","META.INSTRUMENT.CCC\_STATE":"OPEN","meta.instrument.detector":"MIRIMAGE","Meta.Exposure.Type":"MIR\_LRS-FIXEDSLITd"},"ASSOCIATION:badfileset.MIRIMAGE":{}}

### Stale data

In the JWST database, when a level 1b file is processed it replaces the keyword values of the 1a file, when 2a is processed it replaces the 1b values, and when 2b is processed it replaces the 2a values. When data is reprocessed, the highest level data is marked as stale and the cycle begins again. When a reprocessed file with the same level as the stale data is ingested, the stale data will be removed. For example, if we have processed a file through 2b then the database should look like this:

FileX, level=2b, val1, val2, val3, … isStale=False

When we start reprocessing, this row will be marked stale and we’ll insert a new row:

FileX, level=2b, val1, val2, val3, … isStale=**True** (original)

FileX, level=**1a**, val1, val2, val3, … isStale=False (reprocessed)

Further along in reprocessing the database would look like this:

FileX, level=2b, val1, val2, val3, … isStale=True (original)

FileX, level=**1b**, val1, val2, val3, … isStale=False (reprocessed)

When reprocessing is complete the database would look like this:

FileX, level=2b, val1, val2, val3, … isStale=**False** (**reprocessed**)

Usually this happens quickly and the chance for encountering stale data is small. However, if there’s a problem during reprocessing it’s possible for there to be stale data for an extended period of time.

**This method will return the highest level data available (1a – 2b), even if it’s stale.**

### Multiple rows

In the case that the database query returns more than one unique row per FileSetName.Detector, the row with the highest level (up to 2b) will be returned.

If there is more than one unique row at the highest level (up to 2b) per FileSetName.Detector, only one row will be returned, and it is undefined which row it would be.

If there are no level 0.5-2b files available but a different level file is available it will be used. If there is more than one it is undefined which one will be returned.

# Miscellaneous

* In production these queries should probably run against a replicated database in order to avoid any impact on the pipeline.
* Associations aren’t defined in the database yet. Once they’re defined the code will need to be modified to account for them.
* A badly formatted input datasetid is not returned. An error is written to the log, but the datasetId is not returned to the user.
* This code is not as fast as it could be (returns parameters for 5000 ids in about 30 seconds when debugging is turned off). The Hibernate code could probably be tweaked to make it run faster.