UNAVCO's Java Clients for GPS Data/Metadata Discovery

Contents

About the Java Clients
Recommended Java
Installing the Clients
Using the Clients – Introduction with Examples

About the Java Clients

The Java clients were created to give UNAVCO customers a convenient command line mechanism to do monument, metadata, and data discovery and file download from UNAVCO. We envision users experimenting with the configuration they need for doing regular discovery, and then most likely incorporating the client(s) into scripts for gathering the data desired. There are several hurdles for the user in using this type of client, starting with the Java installation on the users system. This document is not intended to fully identify/solve situations that the user may encounter that cause the clients to fail. A few common cases are covered, but please contact archive@unavco.org if you are having trouble.

In conjunction with the clients we are releasing a web GUI for interactive search and access. The web GUI and the command line clients make requests to the same web services on our server. The web services connect to UNAVCO's data sources. Not all data sources are updated continuously, some information sources feeding the clients are updated once per day.

WARNING! There are bugs in this software! Some are known to us and others are not. Please inform us of the bugs that you find by emailing archive@unavco.org.

WARNING! It is possible to submit requests via the clients, especially unavcoFiles.jar, that take a long time to return. In some cases the client will time out. The best strategy is to break up your queries into manageable chunks. For example, if you ask for a list of all hatanaka files for all of PBO for one day, you will most likely receive the result in a reasonable time. However, if you request a list of all hatanaka files for all of PBO the query will most certainly time out. Currently, requesting a list of all hatanaka files for a single PBO station, e.g. P041 installed in 2004, returns the list (around 3000 records in October 2012) in a few minutes.

Java Version

This application is works with Java Runtime Environments (versions 5 through 8).

Installing the Clients

Unzip the contents of unavcoClients.zip into a directory of your choosing. After unzipping, the directory will contain the following:

```
axiom-api-1.2.5.jar
axiom-impl-1.2.5.jar
axis2-kernel-1.3.jar
commons-codec-1.3.jar
commons-httpclient-3.0.1.jar
commons-logging-1.1.jar
stax-api-1.0.1.jar
unavcoFiles.jar
unavcoGrouping.jar
unavcoMetadata.jar
unavcoMonuments.jar
wsdl4j-1.6.2.jar
wstx-asl-3.2.1.jar
xbean-2.2.0.jar
xbean.jar
xmlpublic.jar
XmlSchema-1.3.2.jar
```

Most of these are the supporting library jar files. The actual "clients" are:

```
unavcoFiles.jar
unavcoGrouping.jar
unavcoMetadata.jar
unavcoMonuments.jar
```

To test the install and its compatibility with your Java installation execute the following command on your system (change directory to the one with the clients first):

```
java -jar unavcoMonuments.jar
```

The result should be the user help giving a summary of the command line options:

```
$ java -jar unavcoMonuments.jar
-groupingEqu     <grouping name> (grouping name is exact match)
-groupingStart   <grouping name> (grouping name starts with)
-groupingContains        <grouping name> (grouping name contains)
-boundingBox <min lat> <min lon> <max lat> <max lon>
-boundingRadius <lat> <lon> <radius>
-4charEqu       <list of station codes>      (station code is exact match)
-4charStart     <list of station codes>      (station code starts with)
-4charContains  <list of station codes>  (station code contains pattern)

-pending
-active
-inactive
-retired
```

```
-dataSpanIncludes <date>

-archiveStartBefore <date>
-archiveStartAfter <date>


*Valid Date Formats*
yyyy-ddd
yyyy-mmm-dd
yyyy-mm-dd

-hasMetPack

-xml            (outputs xml)

-permanent
-campaign
```

If instead you get something like the following result:

```
$ java -jar client_libs/unavcoMonuments.jar
Exception in thread "main" java.lang.UnsupportedClassVersionError:
org/unavco/gpssearch/MetadataClient (Unsupported major.minor version 49.0)
        at java.lang.ClassLoader.defineClass0(Native Method)
        at java.lang.ClassLoader.defineClass(ClassLoader.java:539)
        at
java.security.SecureClassLoader.defineClass(SecureClassLoader.java:123)
        at java.net.URLClassLoader.defineClass(URLClassLoader.java:251)
        at java.net.URLClassLoader.access$100(URLClassLoader.java:55)
        at java.net.URLClassLoader$1.run(URLClassLoader.java:194)
        at java.security.AccessController.doPrivileged(Native Method)
        at java.net.URLClassLoader.findClass(URLClassLoader.java:187)
        at java.lang.ClassLoader.loadClass(ClassLoader.java:289)
        at sun.misc.Launcher$AppClassLoader.loadClass(Launcher.java:274)
        at java.lang.ClassLoader.loadClass(ClassLoader.java:235)
        at java.lang.ClassLoader.loadClassInternal(ClassLoader.java:302)
```

… you are most likely accessing an older version of Java. If you know you have the right version of Java on your system, you may have a path issue. To check, use an absolute path to your java. Otherwise, please upgrade to any versions 5 through 8. UNAVCO provides a link from our DAI version 2 GUI and Java clients page to the Java download page for your convenience:

https://www.unavco.org/data/gps-gnss/data-access-methods/dai2/dai2.html

If you still have trouble after installing the correct version of java, please inform us by emailing archive@unavco.org.

Using the Clients – Introduction with Examples

The four clients are:

```
unavcoMonuments.jar
unavcoGrouping.jar
unavcoMetadata.jar
unavcoFiles.jar
```

If you run each one with no command line options (e,g, java –jar <clientname.jar>[return]) you will get back the possible command line options for that client.

User Input Conventions:

> **Dates:** User-supplied dates use either YYYY-DOY or YYYY-MM-DD, all digits, or YYYY-MMM-DD, where MMM is the 3-letter month.
>
> **Latitude, Longitude:** User supplied geographic coordinates are decimal degrees. Longitudes should be in the range (-180., 180.], and latitudes [-90., 90.].
>
> **boundingRadius:** User supplied distance in km.
>
> **grouping*:** User supplied campaign or network name, enclose in single quotes if name contains spaces.

unavcoMonuments.jar:

This client is intended to show information about monuments within the UNAVCO database. Most monuments in the UNAVCO database correspond to data held at UNAVCO. (Note: A more comprehensive database of monuments in use by the UNAVCO community, but not restricted to monuments with data in the UNAVCO Archive is under construction.)

You can restrict the results returned by the client in a number of ways. Here are the command line options:

```
$ java -jar unavcoMonuments.jar
-groupingEqu     <grouping name> (grouping name is exact match)
-groupingStart   <grouping name> (grouping name starts with)
-groupingContains        <grouping name> (grouping name contains)
-boundingBox <min lat> <min lon> <max lat> <max lon>
-boundingRadius <lat> <lon> <radius>
-4charEqu       <list of station codes>      (station code is exact match)
-4charStart     <list of station codes>      (station code starts with)
-4charContains  <list of station codes>  (station code contains pattern)

-pending
-active
```

```
-inactive
-retired

-dataSpanIncludes <date>

-archiveStartBefore <date>
-archiveStartAfter <date>

*Valid Date Formats*

yyyy-ddd
yyyy-mmm-dd
yyyy-mm-dd

-hasMetPack

-xml            (outputs xml)

-permanent
-campaign
```

UNAVCO categorizes monuments as either “permanent” or “campaign”. There are over 2,700 permanent monuments and over 9,000 campaign monuments in the UNAVCO database. If you know you want strictly one or the other, you should be sure to use one of the `-permanent` or `-campaign` options since this will weed out a lot of results you are probably not interested in.

For permanent station monuments, there are several operational status categories that you may wish to use for narrowing your search. These are described by the options:

```
-pending
-active
-inactive
-retired
```

with definitions: active - data has been received within the past year; inactive - no data within the past year, but not retired; retired - disassembled or destroyed permanent station with no expectation of additional data that would extend the end time; pending - no data yet.

For permanent station monuments, you can restrict your search to stations with a met pack .using the -hasMetPack flag.

A simple example is to run the command with a known 4-character ID to find out if UNAVCO “knows about” the monument of interest. Bear in mind that UNAVCO’s 4-character ID for a given monument may vary from the ID you use, especially for campaign monuments.

```
$ java -jar unavcoMonuments.jar -4charEqu av04
Four char id, Monument Name, Lat, Lon,
AV04,AV04AUGST_AK2004,59.3626,-153.4447
```

This tells you that a monument with 4-character ID av04 exists in the UNAVCO database. Note that the latitude, longitude used for input and given as output are always decimal degrees. Also note that the input of the 4-character ID is case insensitive. In the UNAVCO database, IDs are in caps, and in the example above the input 'av04' requested yields the output result 'AV04'.

To illustrate that using `-4charEqu` may give unexpected results because the 4 character ID is not required to be unique in the UNAVCO database, run the same command for 'p100' instead of 'av04:

```
$ java -jar unavcoMonuments.jar -4charEqu p100
Four char id, Monument Name, Lat, Lon,
P100,P100,63.768,-145.7697
P100,ParkValleyUT2007,41.8568,-113.2942
```

The 4 character ID P100 exists as both the Park Valley Utah PBO station P100 and the Denali campaign site P100. Because UNAVCO does not require each monument to have a unique 4-character ID, you may see multiple records that use the same 4-character ID, and these will be from different monuments. Within the permanent station subset of UNAVCO monuments there will not be duplication of 4-character IDs for different monuments. Duplication may occur within the campaign subset and also a campaign monument 4-character ID may be a duplicate of a permanent station 4-character ID.

You can get more verbose output and additional information for each monument returned by using the `-xml` flag. For the av04 example:

```
$ java -jar unavcoMonuments.jar -4charEqu AV04 -xml
<xml-fragment xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <MaxDate>2007-12-23T23:59:00.000Z</MaxDate>
  <MinDate>2004-09-08T23:44:00.000Z</MinDate>
  <Monuments>
    <Monument>
      <MonumentID>18438</MonumentID>
      <MonumentName>AV04AUGST_AK2004</MonumentName>
      <MonumentType>Permanent</MonumentType>
      <FourCharCode>AV04</FourCharCode>
      <Grouping>PBO Magmatic, PBO, PBO Alaska Region</Grouping>
      <Latitude>59.3626</Latitude>
      <Longitude>-153.4447</Longitude>
      <StartTime>2004-09-08T23:44:00.000Z</StartTime>
      <EndTime>2007-12-23T23:59:00.000Z</EndTime>
      <Operational>Active</Operational>
    </Monument>
  </Monuments>
</xml-fragment>
```

In addition to the 4-character ID, monument name, latitude, and longitude, with the –xml flag you get lots of additional information formatted with xml tags including: the UNAVCO monument id, the monument type (either permanent or campaign, which represents its current or most recent utilization), grouping, (list of the networks or campaigns that have employed the monument), the start time (end time) of data in the

UNAVCO Archive associated with the monument, defined as the earliest (latest) data epoch in the UNAVCO Archive, and finally the operational status. In addition, the earliest start time and latest end time for data for all of the monuments returned is included ahead of the set of individual monument records. The information provided comes from a data source that is updated once per day.

Advanced searching on temporal information can be achieved using the options:

```
-archiveStartBefore <date> [YYYY-MM-DD]
-archiveStartAfter <date> [YYYY-MM-DD]
-dataSpanIncludes <date> [YYYY-MM-DD]
```

Looking at the xml output for the search above, you see the StartTime and EndTime fields. These give the span of data held at UNAVCO. Using the `-dataSpanIncludes` flag with a date will return monuments for which the date falls within the data span of the monument. It does not guarantee data for any particular day however. The best way to test for data for a particular day is to use the client unavcoFiles.jar.

The two options `-archiveStartBefore` and `–archiveStartAfter` are intended to allow the user to discover when UNAVCO has added to its collection of monuments. For example if you run this command each day,

```
$ java -jar unavcoMonuments.jar –archiveStartAfter <yesterdays date>
```

advancing the date by one each day, you can discover any new monuments on a daily basis. Keep in mind that the `dataSpanIncludes` dates are generally unrelated to the `archiveStartBefore(After)` date. For example, UNAVCO may receive and archive historical data for a retired site. In this case the all of the data span will precede the date that UNAVCO introduced the monument.

If you are interested in a particular spatial area and wish to find new monuments by checking each day, you can try (for example):

```
$ java -jar unavcoMonuments.jar –archiveStartAfter <yesterdays date> -
boundingBox 40 -112 40.25 -110
```

unavcoGrouping.jar:

This client is intended to show the full set of groupings (permanent station network names, campaign names) in the UNAVCO database. This can be useful for use with the other clients in specifying matches to the `-grouping*` flags.

unavcoMetadata.jar:

This client is intended to show processing metadata within the UNAVCO database for a particular monument. The entire history of the metadata for each matching monument will be shown. Alternatively, the client can give the metadata for a single date. Most monuments in the UNAVCO database correspond to data held at UNAVCO, and for all of these there will be metadata. You can restrict the results returned by the client in a number of ways. Here are the command line options:

```
$ java -jar unavcoMetadata.jar
-groupingEqu     <grouping name> (grouping name is exact match)
-groupingStart   <grouping name> (grouping name starts with)
-groupingContains        <grouping name> (grouping name contains
-4charEqu       <list of station codes>      (station code is exact match)
-4charStart     <list of station codes>      (station code starts with)
-4charContains  <list of station codes>  (station code contains pattern)

-monID  <UNAVCO monument id>
-visitID <UNAVCO visit id>
-stationID  <UNAVCO station id>

-metadataDate <date>
-newerThan <date>
-olderThan <date>
-newestOnly

-recordModifiedBefore <date>
-recordModifiedAfter <date>
-recordModifiedOn <date>

-permanent
-campaign

-recordCountonly
-xml           (outputs xml)
```

The options -permanent; -campaign are as for unavcoMonuments.jar, to restrict the search to one type or the other.

-metadataDate restricts the metadata returned to a single day. There may be more than one record that day if there were one or more changes on the day of interest. The output gives the date-time for the start and end of the time span over which the metadata is applicable.

Here is an example returning all of the metadata records for PBO station AV04:

```
$ java -jar unavcoMetadata.jar -4charEqu av04
4_char_id,station_name,elev,lat,lon,start_time,end_time,antenna_model,antenna_s
erial,antenna_height,radome_model,radome_serial,receiver_model,receiver_serial,
receiver_firmware
AV04,AV04AUGST_AK2004,915.9493,59.3626,-153.4447,2004-09-08T23:44:00.000Z,2005-
08-25T19:25:00.000Z,TRM29659.00,0220335650,0.0083,SCIT,not provided,TRIMBLE
NETRS,4420233978,0.7-0 07 May 2004
AV04,AV04AUGST_AK2004,915.9493,59.3626,-153.4447,2005-08-25T19:41:00.000Z,2005-
09-07T23:59:00.000Z,TRM29659.00,0220335650,0.0083,SCIT,not provided,TRIMBLE
NETRS,4522251506,1.1-2 19 Apr 2005
```

```
AV04,AV04AUGST_AK2004,915.9493,59.3626,-153.4447,2005-09-08T20:48:00.000Z,2006-
01-17T17:00:00.000Z,TRM29659.00,0220335650,0.0083,SCIT,not provided,TRIMBLE
NETRS,4450241622,1.1-2 19 Apr 2005
AV04,AV04AUGST_AK2004,915.9493,59.3626,-153.4447,2006-09-02T22:27:00.000Z,2006-
09-04T00:28:00.000Z,TRM29659.00,0220369851,0.0083,SCIT,not provided,TRIMBLE
NETRS,4450241622,1.1-2 19 Apr 2005
AV04,AV04AUGST_AK2004,915.9493,59.3626,-153.4447,2006-09-04T02:49:00.000Z,2008-
07-21T23:59:00.000Z,TRM29659.00,0220369851,0.0083,SCIT,0604,TRIMBLE
NETRS,4552261622,1.1-2 19 Apr 2005
```

So if we wished to get only the metadata in effect for the date 2005-08-24, one day before the receiver was replaced on 2005-08-25:

```
$ java -jar unavcoMetadata.jar -4charEqu av04 –metadataDate 2005-08-24
4_char_id,station_name,elev,lat,lon,start_time,end_time,antenna_model,antenna_s
erial,antenna_height,radome_model,radome_serial,receiver_model,receiver_serial,
receiver_firmware
AV04,AV04AUGST_AK2004,915.9493,59.3626,-153.4447,2004-09-08T23:44:00.000Z,2005-
08-25T19:25:00.000Z,TRM29659.00,0220335650,0.0083,SCIT,not provided,TRIMBLE
NETRS,4420233978,0.7-0 07 May 2004
```

On the date of the receiver replacement, the client returns both metadata records for that date:

```
$ java -jar unavcoMetadata.jar -4charEqu av04 –metadataDate 2005-08-25
4_char_id,station_name,elev,lat,lon,start_time,end_time,antenna_model,antenna_s
erial,antenna_height,radome_model,radome_serial,receiver_model,receiver_serial,
receiver_firmware
AV04,AV04AUGST_AK2004,915.9493,59.3626,-153.4447,2004-09-08T23:44:00.000Z,2005-
08-25T19:25:00.000Z,TRM29659.00,0220335650,0.0083,SCIT,not provided,TRIMBLE
NETRS,4420233978,0.7-0 07 May 2004
AV04,AV04AUGST_AK2004,915.9493,59.3626,-153.4447,2005-08-25T19:41:00.000Z,2005-
09-07T23:59:00.000Z,TRM29659.00,0220335650,0.0083,SCIT,not provided,TRIMBLE
NETRS,4522251506,1.1-2 19 Apr 2005
```

You can filter the information returned based on the date that the metadata record was changed in the UNAVCO database using these flags:

```
-recordModifiedBefore <date>
-recordModifiedAfter <date>
-recordModifiedOn <date>
```

Additional metadata fields are supplied with the `-xml` flag including the UNAVCO monument_id and monument style. The UNAVCO monument id flag `-monID` is very useful for narrowing the metadata records shown to the station of interest when the 4 character ID is non-unique in the UNAVCO database. The following example shows metadata records from both of the sites with 4 character id ‘P100’

```
$ java -jar unavcoMetadata.jar -4charEqu p100
4_char_id,station_name,elev,lat,lon,start_time,end_time,antenna_model,antenna_s
erial,antenna_height,radome_model,radome_serial,receiver_model,receiver_serial,
receiver_firmware
```

```
P100,P100,692,63.768,-145.7697,2003-06-18T18:31:00.000Z,2003-06-
18T18:31:00.000Z,TRM41249.00,12337743,1.1162,null,null,TRIMBLE
5700,0220280492,1.2
P100,P100,692,63.768,-145.7697,2003-10-17T17:36:00.000Z,2003-10-
17T23:45:00.000Z,TRM22020.00+GP,0220050509,0.9012,null,null,TRIMBLE
4000SSE,3337A03970,7.19
P100,P100,692,63.768,-145.7697,2005-06-17T02:14:00.000Z,2005-06-
19T05:24:00.000Z,TRM22020.00+GP,0220048770,0.843,null,null,TRIMBLE
4000SSI,3604A14201,7.19
P100,P100,692,63.768,-145.7697,2006-10-27T01:03:00.000Z,2006-11-
01T07:29:00.000Z,TRM22020.00+GP,0220050379,1.088,null,null,TRIMBLE
4000SSI,3628A16401,7.19
P100,ParkValleyUT2007,1884.12,41.8568,-113.2942,2007-03-29T01:55:00.000Z,2008-
07-22T23:59:00.000Z,TRM29659.00,0220375560,0.0083,SCIT,1939,TRIMBLE
NETRS,4549261333,1.1-2 19 Apr 2005
P100,P100,692,63.768,-145.7697,2007-10-05T02:36:00.000Z,2007-10-
07T21:13:00.000Z,TRM22020.00+GP,0220050481,1.0894,null,null,TRIMBLE
4000SSI,14493,7.19
```

In order to request just the campaign site you can use –campaign, or specify the UNAVCO monument id. The latter would be necessary if there were more than one campaign monument with the same 4 character ID. You can use the `-xml` flag with unavcoMonuments.jar or unavcoMetadata.jar to see the monument id:

```
$ java -jar unavcoMonuments.jar -4charEqu p100 -xml
<xml-fragment xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <MaxDate>2008-07-22T23:59:00.000Z</MaxDate>
  <MinDate>2003-06-18T18:31:00.000Z</MinDate>
  <Monuments>
    <Monument>
      <MonumentID>18126</MonumentID>
      <MonumentName>P100</MonumentName>
      <MonumentType>Campaign</MonumentType>
      <FourCharCode>P100</FourCharCode>
      <Grouping>Denali 2003-January-July, Denali 2003-July-November, Denali
2005, Denali 2006, Denali 2007</Grouping>
      <Latitude>63.768</Latitude>
      <Longitude>-145.7697</Longitude>
      <StartTime>2003-06-18T18:31:00.000Z</StartTime>
      <EndTime>2007-10-07T21:13:00.000Z</EndTime>
      <Operational>Episodic</Operational>
      <ArchiveStart>2004-02-26T00:00:00.000Z</ArchiveStart>
    </Monument>
    <Monument>
      <MonumentID>21157</MonumentID>
      <MonumentName>ParkValleyUT2007</MonumentName>
      <MonumentType>Permanent</MonumentType>
      <FourCharCode>P100</FourCharCode>
      <Grouping>PBO Extension, PBO, PBO Basin and Range Region</Grouping>
      <Latitude>41.8568</Latitude>
      <Longitude>-113.2942</Longitude>
      <StartTime>2007-03-29T01:55:00.000Z</StartTime>
      <EndTime>2008-07-22T23:59:00.000Z</EndTime>
      <Operational>Active</Operational>
      <ArchiveStart>2007-03-30T00:00:00.000Z</ArchiveStart>
    </Monument>
  </Monuments>
</xml-fragment>
```

The monument id for the permanent station is 21157, so use this to get the metadata for PBO's P100 only:

```
$ java -jar unavcoMetadata.jar -monID 21157
4_char_id,station_name,elev,lat,lon,start_time,end_time,antenna_model,antenna_s
erial,antenna_height,radome_model,radome_serial,receiver_model,receiver_serial,
receiver_firmware
P100,ParkValleyUT2007,1884.12,41.8568,-113.2942,2007-03-29T01:55:00.000Z,2008-
07-22T23:59:00.000Z,TRM29659.00,0220375560,0.0083,SCIT,1939,TRIMBLE
NETRS,4549261333,1.1-2 19 Apr 2005
```

unavcoFiles.jar:

This client is intended to allow the user to obtain the URLs to files available for download from the UNAVCO Archive ftp area. Because the Archive holds several million files on the ftp area, and to avoid the risk that this client may overload the database, the user must narrow the set of records requested. In other words, a request with no constraints will fail with a message that the user must restrict the request. The client packages the results in "pages", with a maximum of 2000 records in a page. If the request would return over 2000 records, the user will need to repeat the request to receive higher page numbers up to the full set of results (using the `-pagenumber <page number>` flag. The user must also specify the type(s) of files desired. [**Note:** the client currently does not return campaign data files in the UNAVCO Archive. This capability will be added in a future release.]

Running the client with no flags gives the options:

```
$ java -jar unavcoFiles.jar
*constraints - you must supply either date or station or network constraints*

-4charEqu       <station code list>
-4charStart     <station code start character(s)>
-4charContains  <station code contains character(s)>

-monID  <UNAVCO monument id>

-groupingEqu <grouping pattern>
-groupingStart <grouping pattern>
-groupingContains <grouping pattern>

*Date format for the following arguments = 'YYYY-MM-DD' or 'YYYY-DOY'*
-dataExactDate <date>
-dataStartDate <date>
-dataEndDate <date>
-newerThan <date>
-olderThan <date>


*exported types to return - you must supply at least one*

-igsLog
-rawNotBinex
-binex
-rinexObs
-rinexHatanaka
-rinexNav
```

```
-rinexMet
-rinexQC
-sinex
-rms
-positionTarball
-position
-positionCSV
-position+CSVTarball
-velocity

*special flags*

-pageNumber <page number> <pages contain 2000 records. By default you get page
1; you may continue getting pages by resubmitting the query and incrementing
the page number.>
-recordCountOnly         <returns only the size and count of the files>
-xml            <outputs xml instead of csv, with extra fields>

*NOTE: It may take several minutes to receive results from your query*
*NOTE: This client will timeout with a Java error if the query takes more than
10 minutes to return*
*      Hint: queries that are not sufficiently narrow are more likely to
timeout*
*      If a timeout occurs, breaking up your query through station, grouping or
date constraints should be attempted*
```

The 4 character ID and network options behave as for the monument and metadata clients. The data file type(s) option is required and allow you to specify the type of files you wish to identify for download.

One way to limit results is to request only datafiles of one type, for example `-rinexHatanaka`, and one station for its entire history:

```
$ java -jar unavcoFiles.jar -4charEqu p456 -rinexHatanaka
Page: 1 of 1, total files = 31, total size(in bytes) = 19508428
Path,   Export Time,    MD5,    Size in bytes,  File Type
ftp://data-out.unavco.org/pub/rinex/obs/2008/205/p4562050.08d.Z, 2008-07-
24T00:32:39.000Z, 525319a2544de06e6be487c01ce04037, 635815, hatanaka-
compression of RINEX obs
ftp://data-out.unavco.org/pub/rinex/obs/2008/204/p4562040.08d.Z, 2008-07-
23T08:32:30.000Z, 66e0a782a2f6d8b97bcb496be14b9d51, 648697, hatanaka-
compression of RINEX obs
ftp://data-out.unavco.org/pub/rinex/obs/2008/203/p4562030.08d.Z, 2008-07-
23T08:32:11.000Z, 41f2fa7849ff03666b7fff7cd4d3062f, 643327, hatanaka-
compression of RINEX obs
ftp://data-out.unavco.org/pub/rinex/obs/2008/202/p4562020.08d.Z, 2008-07-
23T08:31:51.000Z, aa7c3fb5dec3164e9731109d96de8e4c, 646005, hatanaka-
compression of RINEX obs


<snip>
```

The results of the query, in CSV format, include the URL to the file, the date-time the file was posted on the ftp server. The URLs are returned in time order of the data contents, most recent first. If several stations are requested, the URLs are alphabetical for each date of data:

The date options allow a great deal of flexibility in identifying files that may be of interest. In some cases the user wants simply to specify an exact date (using `-dataExactDate`) or a date range using `-dataStartDate` and `-dataEndDate`) for the **contents** of the file.

```
$ java -jar unavcoFiles.jar -dev -4charEqu p123 p124 -rinexHatanaka -
dataStartDate 2008-01-01 -dataEndDate 2008-01-04
Page: 1 of 1, total files = 6, total size(in bytes) = 3716305
Path,   Export Time,    MD5,    Size in bytes,  File Type
ftp://data-out.unavco.org/pub/rinex/obs/2008/003/p1230030.08d.Z, 2008-01-
04T01:08:16.000Z, 2e4057c3e8bc0d34dcf824be72c7e2ed, 662447, hatanaka-
compression of RINEX obs
ftp://data-out.unavco.org/pub/rinex/obs/2008/003/p1240030.08d.Z, 2008-01-
04T01:09:01.000Z, 47decb2c51e0a04367c6b16adcf2fea9, 592481, hatanaka-
compression of RINEX obs
ftp://data-out.unavco.org/pub/rinex/obs/2008/002/p1230020.08d.Z, 2008-01-
03T01:07:04.000Z, aeb77344e582d9163a0d2039535737cd, 655031, hatanaka-
compression of RINEX obs
ftp://data-out.unavco.org/pub/rinex/obs/2008/002/p1240020.08d.Z, 2008-01-
03T01:07:25.000Z, 7f3ce9fc25084f5bbee705334dfadd22, 579342, hatanaka-
compression of RINEX obs
ftp://data-out.unavco.org/pub/rinex/obs/2008/001/p1230010.08d.Z, 2008-01-
02T01:00:36.000Z, b7a3ee2be3f18431e1483bed560457b7, 652085, hatanaka-
compression of RINEX obs
ftp://data-out.unavco.org/pub/rinex/obs/2008/001/p1240010.08d.Z, 2008-01-
02T03:26:37.000Z, 0f9d298e07341170894d021368f651c1, 574919, hatanaka-
compression of RINEX obs
```

Another variation, useful for finding if files have been replaced since they were previously picked up (for a possible scenario), involves using the `-newerThan` flag:

```
$ java -jar unavcoFiles.jar -4charEqu p346 –rinexHatanaka –newerThan 2008-07-07
–dataEndDate 2008-12-07


Page: 1 of 1, total files = 5, total size(in bytes) = 3172887
Path,   Export Time,    MD5,    Size in bytes,  File Type
ftp://data-out.unavco.org/pub/rinex/obs/2007/334/p3463340.07d.Z, 2008-07-
10T22:15:28.000Z, ce5122e9758d4d64c72a53b57f6d82c0, 634983, hatanaka-
compression of RINEX obs
ftp://data-out.unavco.org/pub/rinex/obs/2007/333/p3463330.07d.Z, 2008-07-
10T22:17:35.000Z, c1cee7d5808a864af3323f78729af110, 633979, hatanaka-
compression of RINEX obs
ftp://data-out.unavco.org/pub/rinex/obs/2007/332/p3463320.07d.Z, 2008-07-
10T22:17:03.000Z, 856e755e03c5b48ae18993169c7e1e0d, 636279, hatanaka-
compression of RINEX obs
ftp://data-out.unavco.org/pub/rinex/obs/2007/331/p3463310.07d.Z, 2008-07-
10T22:16:24.000Z, 25176ad04963853a32ef56de573d296c, 634529, hatanaka-
compression of RINEX obs
ftp://data-out.unavco.org/pub/rinex/obs/2007/330/p3463300.07d.Z, 2008-07-
10T22:15:48.000Z, 222c431b700986acbb178709426a7a14, 633117, hatanaka-
compression of RINEX obs
```

This shows files with contents having end times before December 12, 2007, but posted after 2008-07-07.

If you have questions that are not addressed by this document, please e-mail archive@unavco.org.