# AIDA Command-line tools

This page describes how to set and get control system values via rudimentary AIDA command line tools. These are available on LCLS production nodes without setup, and on SLAC public machines after going through the basic setup described in Basic User Guide to Aida (see also below for a summary).

## Fetching Values: *aidaget*

The utility program named "**aidaget**" takes one mandatory argument (the name of the target). Here's a very simple example, as it happens getting an EPICS veriable's value:

```
aidaget HR81:STN:VOLT//VAL
  10-Sep-2008 12:09:40  0  0  0.18099030997326265
```

Several optional parameters control how it works.  You can specify "-help" as a parameter to see simple options. Here's that same EPICS process variable, using options to display labels, and in rows format.

```
aidaget HR81:STN:VOLT//VAL -labels -format=rows
time        06-Nov-2006 10:30:05
status                        0
severity                      0
value        0.22782705806669729
```

Some of the more interesting options are specific to each data-provider, which may support optional or require mandatory paramters.

The following example retrieves history data.  It illustrates how you can specify data provider parameters (in this case STARTTIME and ENDTIME) and control output format (-format=columns -labels):

```
aidaget HR81:CAV2:VACM//HIST.pepii -format=columns -labels \
-DSTARTTIME="02/01/2003 12:20:00" -DENDTIME="02/01/2003 12:40:00"
           d                     s           repCnt_l   unixtime
             1.93979E-9  01-Feb-2003 12:20:00         1  1044130800
              1.9181E-9  01-Feb-2003 12:20:55         0  1044130855
              1.9181E-9  01-Feb-2003 12:21:00         1  1044130860
            1.928914E-9  01-Feb-2003 12:21:18         0  1044130878
            1.928914E-9  01-Feb-2003 12:24:00         3  1044131040
```

As you can see, data provider parameters are specified as -Dname1=value1 -Dname2=value2.  Note the need to enclose embedded whitespace within " marks to prevent the shell from treating it as a boundary between strings.  The output format may be specified as "columns" or "rows".  The only difference is that one is a transpose of the other.  The option "-labels" indicates that the columns (or rows) should be named.  Not all data providers supply these names, in which case you'll need to consult their documentation to interpret the column numbers.

Getting BPM data:

```
aidaget P2BPMHER//BPMS -DBPMD=38 -DREF=0
  BPMS:PR10:6012       0.0       0.0  3128.0562          0.0        17       520
  BPMS:PR10:6022       0.0       0.0  3135.6562          0.0        17       520
  BPMS:PR10:6032       0.0       0.0   3143.256          0.0        17       520
  ...
```

Getting Magnets:

```
aidaget LINAC//XCOR:BDES -DMICROS=LI10-LI10 -DUNITS=1-9999
  XCOR:LI10:200  0.0
  XCOR:LI10:201  0.0
  XCOR:LI10:202  0.0
  ...
```

### Getting data from a number of devices at a time.

Combining a file of device names, or aidalist with a unix pipe, you can get a number of values at a time. For instance, get the Twiss parameters from the online model, of all the devices in a file. Say your devices were listed in a file (eg checkDevices27Oct2008.txt_Aida_NO), then you can get their values like this:

```
cat checkDevices27Oct2008.txt_Aida_NO | xargs -I {} -t aidaget "{}//twiss" -DMODE=5
```

The -t says echo what you're about to execute before executing it.

You can use output of aidalist directly to get values of a number of devices.
Combine aidalist with caget:

```
aidalist BPMS:LI27:%:X1 | xargs -I{} caget {}
BPMS:LI27:201:X1               -0.210293
BPMS:LI27:301:X1               0.0417948
BPMS:LI27:401:X1               0.246912
BPMS:LI27:501:X1               -0.252207
BPMS:LI27:601:X1               -0.699057
BPMS:LI27:701:X1               0.00149554
BPMS:LI27:801:X1               0.161638
BPMS:LI27:901:X1               -0.086897
```

This example gets the twiss of all the XCORs in LI22. Note, we use the 2 argument form of aidalist to confine the the output to just the twiss attribute (rather than all the PVs of all the XCORs in LI22), then we select only the forst column of output (the device names), then call aidaget with is the device name and '//twiss' appended:

```
aidalist XCOR:LI22:% twiss | awk '{print $1}' | xargs -I {} -t aidaget "{}//twiss" -DMODE=5
```

## Setting Value: *aidaset*

The utility program named "**aidaset**" takes at least two mandatory arguments (the name of the target and the desired value), and several optional parameters control how it operates.

An AIDA data provider that supports setting values will accept either a single scalar or vector value, or a list of named scalar or vector values. The EPICS channel access data provider is of the former sort, and the SLC Magnet server is of the latter.

***NB:*** Presently, only the EPICS Channel Access and SLC Magnet data providers support write operations.

An example of setting a scalar EPICS process variable:

```
aidaset HR81:STN:VOLT//VAL 0.227
```

And an example of setting a corrector via the Magnet server:

```
aidaset MAGNETSET//BDES -DMAGFUNC=TRIM /names XCOR:LI31:41 /values 4.0
```

In the EPICS channel access example, we sent a single unnamed value. In the magnet example, we sent a named value. As you can see, the magent server expects two vectors, one listing magnet names and the other listing the corresponding desired values. Unless the data provider documentation indicates otherwise, the names of the vectors are not important. The following is equivalent to the above example:

```
aidaset MAGNETSET//BDES -DMAGFUNC=TRIM /sun XCOR:LI31:41 /moon 4.0
```

Vector elements my be listed sequentially with whitespace separators as in the following example that sets several magnets simultaneously:

```
aidaset MAGNETSET//BDES -DMAGFUNC=TRIM /names XCOR:LI10:200 XCOR:LI10:201 /values 0.0123 -0.0321
```

## Setting up on an AFS Node

If you are using these tools on lcls nodes (lcls-prod02, lcls-srv01 etc) then the following setup has been done for you. If you are using a SLAC AFS node, then do this setup to use the above tools:

```
source /afs/slac/g/cd/soft/dev/script/ENVS.csh
source /afs/slac/g/cd/soft/dev/script/aidaSetEnv.csh prod
```

You may wish to add these to your .cshrc.