Chandra X-ray Center

AHELP for CIAO 3.4   **dmhedit**   Context: tools

Jump to: Description Examples Parameters CHANGES IN CIAO 3.4 Bugs See Also

Synopsis

Edit datamodel file headers

Syntax

```
dmhedit infile filelist operation key value [datatype] [unit] [comment]
[verbose]
```

Description

Edits the header of a file in place, modifying the original file. There are two ways to use dmhedit: specifying parameters on the command line to change one keyword or with an ASCII edit list file to modify several keywords at once. The syntax for each method is illustrated in the EXAMPLES.

Format for the edit list file:

The edit list file can contain two kinds of lines, control lines and edit lines. The control lines, which must begin with a pound sign (#), specify what to do with the edit lines that follow it. The valid control lines are:

- "#add" : add a new keyword at the end of the block header
- "#delete" : remove an existing keyword
- "#add after KEYNAME" : add a new keyword after an existing one
- "#move after KEYNAME" : move one existing keyword to after another one

Anything other line beginning with a pound sign is treated as a comment. Note that there may not be any blank lines in the edit list file or the tool will complain; also, there may not be a space between the pound sign and the command in the control lines. A control line applies to all following commands until a new control line is encountered.

Each edit line has the format

```
KEYNAME = value  /  [unit] comment
```

The entries must be space delimited, tabs are not allowed. The unit and comment may be omitted; if both are omitted, the slash following the value should be omitted as well. If the value is a string value, it should be included in single quotes.

String values that contain slashes:

Any string value that contains a forward slash ("/") must be enclosed in single quotes within double quotes when setting the parameter, i.e. value="'/this/is/a/string/'".

Adding comment keywords:

The tool can also add comments to the header (i.e. COMMENT and HISTORY keywords). In this case, only the "key" and "value" parameters are used, with the latter containing the comment text. Since comments do not have unique key names, it is not possible to delete existing ones.

Users should be aware that dmhedit can also set structural FITS keywords which are normally hidden by the Data Model. However, this is not recommended and can easily corrupt files.

Example 1

```
dmhedit in.fits filelist=none operation=add key=LIVETIME value=123.3
datatype=indef
```

This is an example of using dmhedit to update a single keyword. The default block of the input file is modified to update the keyword LIVETIME with value 123.3 using the existing datatype.

Example 2

```
dmhedit in.fits none add NEWTIME "123.3 / [s] New time" datatype=double
```

Single key mode is again used to add the keyword NEWTIME with double value 123.3, units of seconds, and the comment "New time". This example shows how the value, unit, and comment may all be included in the "value" parameter.

Example 3

```
dmhedit in.fits none add LIVETIME 'some string value' datatype=string
```

The existing keyword LIVETIME (which has a double datatype) is updated with a string value.

Example 4

```
dmhedit in.fits none operation="add after OBJECT" key=AGNCLASS
value='Type 2 Seyfert' datatype=string
```

A new keyword, AGNCLASS, is added after the OBJECT key.

Example 5

```
dmhedit in.fits none operation="move after DEC_NOM" key=RA_NOM
```

The order of RA_NOM and DEC_NOM is switched via the "move after KEYNAME" operation. Since RA_NOM is an existing keyword, it is not necessary to supply a value, datatype, etc.

Example 6

```
dmhedit in.fits none add GAINFILE "'/path/name/gain.fits'" unit=''
comment='Gain file'
```

This shows the correct syntax when working with a string with forward slashes in it (in this case, a directory path).

Example 7

```
dmhedit in.fits edit.list
```

dmhedit is used in edit list mode to make multiple changes with one command. The file "edit.list" might look like:

```
#add
LIVETIME = 142.3 / [s]  Live time
INSTRUME = 'HRC-S'
#delete
PHAMAX
#add
GAUSS.POWER = -1.8  / Power law index
```

Parameters

name	type	ftype	def	min	max	reqd	stacks
infile	file	input				yes	yes
filelist	file					yes	no
operation	string					yes	
key	string					yes	
value	string					yes	
datatype	string		indef				
unit	string						
comment	string						
verbose	integer		0	0	5		

Example 5   3

Detailed Parameter Descriptions

Parameter=infile `(file required filetype=input stacks=yes)`

Input dataset

Input FITS file, or stack of files (see "ahelp stack" for more information), with an optional FITS extension in brackets, e.g. "filename.fits[GTI2]". The same header edits are applied to all files in the stack.

Parameter=filelist `(file required stacks=no)`

An ASCII edit list file

A file containing control and edit lines, see the full explanation in the DESCRIPTION. If set to "none" or empty, dmhedit uses the " operation", "key", and "value" parameter. If a filelist is given and the other parameters are specified, the filelist takes precedence.

Parameter=operation `(string required)`

Add/change or delete the key

Specifies what operation should be performed on the given keyword. Valid options are "add" and "delete". Use the "add" operation to update an existing key.

Parameter=key `(string required)`

Keyword to change

Name of the new keyword to add or an existing keyword to edit or delete. The syntax

```
value="value / [unit] comment"
```

may be used to provide the value, unit, and comment at the same time (e.g. value="45.89 / [ks] New time").

Parameter=value `(string required)`

Value of the keyword that is being modified

Remember that string values which contain forwarded slashes must be enclosed in single quotes within double quotes when setting the parameter, i.e. value="'/this/is/a/string/'".

Parameter=datatype `(string default=indef)`

Datatype of the keyword value

The datatype of the value when the "add" operation is being used. Possibilities are "string", "double", "float", "long", "short", "ulong", "ushort", "boolean", or "indef".

If "datatype=indef" and the keyword exists, the datatype will be set to that of the existing keyword. If "datatype" is set to anything else or a new key is being added, the tool will attempt to write the keyword using the specified datatype.

Any value can be written as a string, while only type–specific numeric values can be written using the numeric datatypes. If the value is a number and contains a decimal point, it will be written as a double; otherwise, numbers are written as short integers. If the value is not a number, it will be written as a string. Allowed values to update a boolean keyword are "T", "F", "TRUE", and "FALSE", all case–insensitive.

Parameter=unit `(string)`

Unit of the value

If there is a unit associated with the keyword value, it is input via this parameter.

Parameter=comment `(string)`

Keyword comment

A description of the keyword or other related comment.

Parameter=verbose `(integer default=0 min=0 max=5)`

Verbose level

Reports the input filenames and the key to be changed, as well as any warnings.

CHANGES IN CIAO 3.4

IRAF QPOE Files

Support for IRAF QPOE files has been removed from CIAO.

Bugs

See the bugs page for this tool on the CIAO website for an up–to–date listing of known bugs.

See Also

tools

dmdiff, dmhistory, dmkeypar, dmmakepar, dmreadpar

Parameter=unit (string)