CORONAL DIAGNOSTIC SPECTROMETER
SoHO

CDS SOFTWARE NOTE No. 22

Version 1.1 23 September 1994

CDS ON-LINE HELP UTILITIES

C. D. Pike
Rutherford Appleton Laboratories
Chilton Didcot
Oxon OX11 0QX

cdp@astro1.bnsc.rl.ac.uk

1 Introduction

There are two basic requirements for the CDS IDL on-line help system. The first is to be able to retrieve information from the general to the particular on specific routines whose name is known. The second is to be able to find out what routines are available to support a particular task when all the information the user has is either the task concept or a vague memory of the existence of some suitable program. The programs developed so far for use with the CDS IDL software complement and extend the intrinsic IDL utilities to satisy both these requirements.

2 Intrinsic utilities

It is as well not to forget the in-built IDL help functions and reference should be made to Chapter 13 of the IDL User's Guide.

The HELP procedure will given information on the current IDL session and takes numerous keywords to refine the information presented. For instance, a commonly used keyword (/STRUCTURE) details the contents of a structure variable.

Entering a question mark (?) at the IDL prompt will (on an X Windows device) present a graphical menu of topics and IDL-supplied procedure and function names. It is then possible to inspect the on-line documentation for any of those routines.

3 Header documentation

The ultimate point of contact in any request for on-line help is the documentation header contained in the source code file. Every CDS IDL source code file has a standard IDL header. The template used, with annotation, is given below and the 'empty' version is available for copying from file /cs/test/util/help/template.inc

```
;+
; Project     : SOHO - CDS
;
; Name        : Procedure/function name. Function names followed by ().
;
; Purpose     : Strictly a one-line description of the routine's purpose.
;
; Explanation : A fuller, usually multi-line, explanation of the program's
;               purpose and method of operation.
;
; Use         : An example call.
;
; Inputs      : An explanation of any input parameters.
;
; Opt. Inputs : An explanation of any optional input parameters.
```

```
;
; Outputs      : An explanation of any output parameters
;
; Opt. Outputs: An explanation of any optional output parameters.
;
; Keywords     : An explanation of any keywords recognised.
;
; Calls        : A list of any other CDS routines called by this routine.
;
; Common       : Details of any common blocks used.
;
; Restrictions: An explanation of any restrictions on the use of this routine.
;
; Side effects: Details of any (usually transparent) side effects.
;
; Category     : At least one software category name relevant to this routine.
;
; Prev. Hist. : Details of any previous history of the routine or idea.
;
; Written      : Author, institute, date.
;
; Modified     : Add and date a new entry each time the routine is modifed.
;
; Version      : Number and date of latest version.
;-
```

Not only does the required existence and completion of this header provide a discipline for programmers but it also makes the operation of programs attempting to read the documentation very much more easy and robust.

4 CDS help utilities

The following is a list of the routines which make use of the source code header in order to provide on-line help and documentation to the user.

XDOC:

This general documentation viewer is very similar in concept to the IDL '?' command. If a parameter (a specific routine name) is supplied, the documentation header and, optionally, the full code listing of that routine is displayed. The user may then proceed to inspect any routine in any of the CDS software directories by mouse selection of the directories and files. If **xdoc** is called with no parameters the available directories are still listed and may be selected by the mouse.

PURPOSE:

The procedure *PURPOSE* accesses the one-line explanation of programs held in the file header. The available keywords (*hard, path, quiet & list*) can be used to modify or preselect the information returned. The default mode is to list the one-liners from routines in the current directory. If the *path* keyword is used it can be made to extract information from all or selected parts of the CDS IDL path. The information, as well as being printed on the screen (if keyword *quiet* is not set) can also be returned in a variable specified by the *list* keyword.

As an example, if a user wishes to be reminded of the the CDS numerical utilities and is knowledgable enough to know they are likely to be in the CDS path in a directory containing the string */numerical* then the following command will produce the necessary result.

```
IDL>  purpose, path='/numerical'


Directory:  /disk2/cds/soft/test/util/numerical/

BIN2DEC          - Convert binary representation to decimal integer.
BIN2HEX          - Convert binary representation to hexadecimal.
BOX_AV()         - Produce a box average of an array.
BYTESWAP()       - Swaps the bytes in an integer (as a function).
CDS_GFUNCT       - Evaluate gaussian and its partial derivatives.
DEC2BIN          - Convert integer decimal number to binary representation.
DEC2HEX          - Convert a non-negative decimal integer to a hex string.
HEX2BIN          - Convert hexadecimal number to binary representation.
HEX2DEC          - Convert hexadecimal representation to decimal integer.
MODE_VAL()       - Returns the modal value of an array.
N_DIMENSIONS()   - Returns number of dimensions of a variable.
NINT()           - Returns the nearest integers to the input values.
NUM_CHK()        - Check if input is valid number.
ROUND_OFF()      - To round a number to a specified accuracy
SIZEOF()         - Calculates the size of an IDL variable
UNSIGN()         - Produces longword equivalent of an unsigned 16-bit integer.
VALID_NUM()      - Check if a string is a valid number representation.
```

Actually entering *purpose, path='num* would have achieved the same result.

TFTD:

The procedure TFTD can be used for a number of documentation tasks. In its simplest form it will list a specified number (default six) of one-liners with the intention of reminding users, on a random basis, of the existence of routines they may have forgotten or never knew existed.

Thus a simple call to TFTD in an idle moment could produce the following surprises:

```
Thoughts for the day produced by TFTD q.v.

CONCAT3D()          - Concatenate two or more 3-d arrays.
GT_WINSIZE()        - Get the size of data windows from a file or data structure
INT_TABULATED       - This function integrates a tabulated set of data { x(i) ,
LABEL_CURVE         - Plots a label with a line from it to a curve.
HEX2DEC             - Convert hexadecimal representation to decimal integer.
GEAR                - Graphically display how the front and rear gears of a bike
```

Clearly if the idle moment lasts, the next command is **xdoc,'gear**!

Another, perhaps more valuable, feature of TFTD is its ability to search the complete list of program names and one-liners for specified strings. For instance, a user may want to remove duplicate values from a character array and then replace certain characters by another string. Are there any tools to help? A couple of calls to TFTD will give the answer (of course a null answer is often as useful, reinventing wheels is a frustrating occupation):

```
IDL> tftd,'duplic

Thoughts for the day produced by TFTD q.v.

DB_OR()             - Combine two vectors of entry numbers, removing duplicates.
FIND_DUP()          - Function to identify duplicate values in a vector.
REM_DUP()           - Function to remove duplicate values from a vector.


IDL> tftd,'repl

Thoughts for the day produced by TFTD q.v.

DBXPUT              - Routine to replace value of an item in a data base entry
LUDCMP              - Replaces an N by N matrix, A, with the LU decomposition.
REPCHAR()           - Replaces a character within a string by another.
REPLICATE           - Returns an array filled with the original scalar value.
REPSTR()            - Replaces a string within a string by another.
```

There is a third mode of operation for TFTD which accesses the *Category* entry of the documentation header. This can sometimes be useful when you want to know what routines are available under a certain vague category and you are not sure that the one-liners necessarily contain the appropriate string.

For example, suppose a listing is required of all routines that may pertain to on-line help. You could try **TFTD,'help'** but some routines may be missed if the word *help* didn't happen to figure in the one-liner. However, by using the keyword *cat* (for category) it is possible to select all routines for which the authors entered *help* under the Category heading.

Beware, the control over entries under Category has not been as tight as it should have been and so, for the moment, this mode could be a little flakey. Try getting a list of possible categories **tftd,cat='?'** *to see what I mean. Hopefully this can be rectified soon.*

So to find useful on-line help routines try:

```
IDL> tftd,cat='help

CDSLOC           - Locate any file name in /cs tree with specified string.
CHKARG           - Determine calling arguments of procedure or function.
DL_DOS           - Extract the documentation template of one or more procedures (DOS
DL_UNIX          - Extract the documentation template of one or more IDL modules
DL_VMS           - Extract the documentation template of one or more procedures.
DOC              - Obsolete-- use XDOC instead
DOC_LIBRARY      - Extract the documentation template of one or more IDL modules
DOC_MENU         - Extract documentation template of one or more procedures.
FILL_CATEGORY    - Load save file with current categories
FILL_TFTD        - Load save file with current one-liners
FLASH_MSG        - Flashes an information message in a text widget.
GET_DFONT()      - Get widget font with size compatible with current device
GET_LIB()        - Place elements of !PATH into a string array..
GET_MOD()        - Extract list of procedure modules.
GET_PROC()       - Extract procedure from a library or directory.
MAN_PROC         - Provide online documentation for IDL topics.  If the current
MK_LIBRARY_HELP  - Given a directory or a VMS text library containing IDL userlib
MP_BASIC         - Provide online documentation for IDL topics. The style
MP_WIDGETS       - Provide a graphical user interface to the online documentation.
PATH_EXPAND      - Expands VMS logical names in a search path.
QLHELP           - Widget to select help topics related to the QL Software.
SCANPATH         - Widget program for reading documentation in IDL procedures.
SP_COMMON        - Contains common blocks used by SCANPATH.
TFTD             - Search for a string in header documentation.
WHICH            - Search for file in IDL !path, print all occurrences
XDL              - Provide a graphical user interface to the DOC_LIBRARY user
XDOC             - Front end to online documentation software.
```

Hopefully the routines mentioned in this software note will form a subset of those above!

CHKARG:

If you know the name of a routine but can't remember whether it's a function or procedure or want a quick route to checking its parameters and keywords, try using **chkarg**. This is a memory-jog routine only since no explanation is given of the parameters and/or keywords. Thus:

```
IDL> chkarg,'break_file


---- Module: break_file.pro
---- From:   /disk2/cds/soft/test/util/os
---> Call: pro break_file, file, disk_log, dir, filnam, ext, fversion, node
```

WHICH:

For users who are intending to create new procedures or functions and wish to avoid creating program names which duplicate an existing one, **which** provides a quick and easy check for existing names. Thus:

```
IDL>
IDL> which,'find_data_runs
/disk2/cds/soft/test/plan/tech/find_data_runs.pro
/disk2/cds/soft/test/util/misc/find_data_runs.pro
IDL>
IDL> which,'daft_name
IDL>
```

The duplication of **find_data_runs** is an error, and there is no existing file with the name **daft_name**.

SHOW_STRUCT:

Although the intrinsic function **help,/str** shows detailed information on a structure, the procedure **show_struct** provides an easy and quick graphical way to peruse all levels of a structure variable.

5 Maintenance

For speed, **tftd** uses preprepared lists of one-liners and category classifications. If new programs are inserted into the CDS directories, these lists must be updated. To perform these tasks the routines **fill_tftd** & **fill_category** are provided.

These routines must be run by a user having write access to the CDS directories. A complete update can be performed by the following commands:

```
IDL> fill_tftd
IDL> fill_tftd, /prog
IDL> fill_category
IDL> fill_category, /prog
```

The selection of which directories to include in the lists is hardcoded into the routines **fill_tftd** & **fill_category** and so these routines must be edited to change or update that selection.