DFANaddfds

```
intn DFANaddfds(int32 file_id, char *description, int32 desc_len)
```

file_id	IN:	The HDF file identifier returned by **Hopen** or **HDFopen**
description	IN:	Sequence of ASCII characters (may include NULL '\0')
desc_len	IN:	Length of the description

Purpose — Adds a file description to a file.

Return value — Returns SUCCEED(0) if successful and FAIL(-1) otherwise.

Description — These annotations are associated with the file, not with any particular object within the file. The parameter *description* can contain any sequence of ASCII characters. It does not have to be a string. Use the general purpose routines **Hopen** or **HDFopen** and **Hclose** or **HDFclose** to manage file access because the file annotation routines will not open and close HDF files for you.

Example — This example illustrates the use of **DFANaddfid** and **DFANaddfds** to write to an HDF file a label and description for the file itself.

```
#include "hdf.h"
#define MAXLABLEN 80
#define MAXDESCLEN 1000
...
int32 file_id;
char label[MAXLABLEN+1], description[MAXDESCLEN+1];

file_id = Hopen("myfile", DFACC_WRITE, 0);

/* store a file label in the file */
strcpy (label, "File #1");
DFANaddfid(file_id, label);

/* store description in file */
DFANaddfds(file_id, description, desc_len(description));

Hclose(file_id);
...
```

FORTRAN

```
integer function daafds(file_id, description, desc_len)
   integer*4      file_id
   character*(*)  description
   integer*4      desc_len
```

DFANaddfid

daafid

DFANaddfid

```
intn DFANaddfid(int32 file_id, char *label)
```

file_id	IN:	The HDF file identifier returned by **Hopen** or **HDFopen**
label	IN:	A single string of null-terminated text

Purpose
Writes a file label to a file.

Return value
Returns SUCCEED(0) if successful and FAIL(-1) otherwise.

Description
These annotations are associated with the file, not with any particular object within the file. The label must be a single string. Use the general purpose routines **Hopen** or **HDFopen** and **Hclose** or **HDFclose** to manage file access because the file annotation routines will not open and close HDF files for you.

In the FORTRAN version, the string length for the label should be close to the actual expected string length, because in FORTRAN string lengths generally are assumed to be the declared length of the array that holds the string.

FORTRAN

```
integer function daafid(file_id, label)
   integer*4      file_id
   character*(*)  label
```

dagdesc

DFANgetdesc

```
intn DFANgetdesc(char *filename, uint16 tag, uint16 ref, char *desc_buff,
          int32 buff_len)
```

filename	IN:	Name of the HDF file
tag	IN:	Tag of the data object assigned the description
ref	IN:	Reference number of the data object assigned the description
desc_buff	OUT:	The buffer allocated to hold the description
buff_len	IN:	Size of the buffer allocated to hold the description

Purpose Reads the description assigned to the data object with the given tag and reference number.

Return value Returns `SUCCEED(0)` if successful and `FAIL(-1)` otherwise.

Description The parameter *buff_len* gives the amount of space that is available for storing the description. The length of *buff_len* must be at least one greater than the anticipated length of the description, because a `'\0'` byte is appended to the annotation.

FORTRAN

```
integer function dagdesc(filename, tag, ref, desc_buff, buff_len)
    character*(*)  filename, desc_buff
    integer*2      tag, ref
    integer*4      buff_len
```

DFANgetdesclen

dagdlen

DFANgetdesclen

```
int32 DFANgetdesclen(char *filename, uint16 tag, uint16 ref)
```

filename	IN:	Name of the HDF file
tag	IN:	Tag of the data object assigned the description
ref	IN:	Reference number of the data object assigned the description

Purpose — To get the length of a description of the data object with the given tag and reference number.

Return value — Returns the length of a description if successful and `FAIL(-1)` otherwise.

Description — This routine should be used to insure that there is enough space allocated for a description before actually reading it.

FORTRAN

```
integer function dagdlen(filename, tag, ref)
   character*(*)  filename
   integer*2      tag, ref
```

dagfds

DFANgetfds

```
int32 DFANgetfds(int32 file_id, char *desc_buff, int32 buff_len, intn isfirst)
```

file_id	IN:	The HDF file identifier returned by **Hopen** or **HDFopen**
desc_buff	OUT:	The buffer allocated to hold the description
buff_len	IN:	Size of the buffer allocated to hold the description
isfirst	IN:	1 to read the first description; 0 to read the next description

Purpose — Reads the next file description from a file.

Return value — Returns the length of the file description if successful and FAIL(-1) otherwise.

Description — If *isfirst* is 0, **DFANgetfds** gets the next file description from an HDF file. For example, if there are three file descriptions in a file, three successive calls to **DFANgetfds** will get all three descriptions. If *isfirst* is 1, **DFANgetfds** gets the first file description.

FORTRAN

```
integer function dagfds(file_id, desc_buff, buff_len, isfirst)
    integer*4       file_id
    character*(*)   desc_buff
    integer*4       buff_len
    integer         isfirst
```

DFANgetfdslen

dagfdsl

DFANgetfdslen

```
int32 DFANgetfdslen(int32 file_id, intn isfirst)
```

file_id	IN:	The HDF file identifier returned by **Hopen** or **HDFopen**
isfirst	IN:	`1` to read the first description; `0` to read the next description

Purpose — Returns the length of a file description.

Return value — Returns the length of the file description if successful and `FAIL(-1)` otherwise.

Description — When **DFANgetfdslen** is first called for a given file, it returns the length of the first file description. In order to get the lengths of successive file descriptions, you must call **DFANgetfds** between calls to **DFANgetfdslen**. Successive calls to **DFANgetfdslen** without calling **DFANgetfds** between them will return the length of the same file description.

FORTRAN

```
integer function dagfdsl(file_id, isfirst)
   integer*4      file_id
   integer        isfirst
```

DFANgetfid

```
int32 DFANgetfid(int32 file_id, char *desc_buff, int32 buff_len, intn isfirst)
```

file_id	IN:	The HDF file identifier returned by **Hopen** or **HDFopen**
label_buff	OUT:	The buffer allocated to hold the label
buff_len	IN:	Size of the buffer allocated to hold the label
isfirst	IN:	1 to read the first label; 0 to read the next label

Purpose — Reads a file label from a file.

Return value — Returns the length of the file description if successful and FAIL(-1) otherwise.

Description — If *isfirst* is 0, **DFANgetfid** gets the next file label from the file. If *isfirst* is 1, **DFANgetfid** gets the first file label in the file. If the *buff_len* is not large enough, the label is truncated to *buff_len*-1 characters in the buffer label_buff.

FORTRAN

```
integer function dagfid(file_id, desc_buff, buff_len, isfirst)
    integer*4       file_id
    character*(*)   desc_buff
    integer*4       buff_len
    integer         isfirst
```

DFANgetfidlen

dagfidl

DFANgetfidlen

```
int32 DFANgetfidlen(int32 file_id, intn isfirst)
```

file_id	IN:	The HDF file identifier returned by **Hopen** or **HDFopen**
isfirst	IN:	1 to read the first label; 0 to read the next label

Purpose — Returns the length of a file label.

Return value — Returns the length of the file label if successful and FAIL(-1) otherwise.

Description — When **DFANgetfidlen** is first called for a given file, it returns the length of the first file label. In order to get the lengths of successive file labels, you must call **DFANgetfid** between calls to **DFANgetfidlen**. Otherwise, successive calls to **DFANgetfidlen** will return the length of the same file label.

FORTRAN

```
integer function dagfidl(file_id, isfirst)
    integer*4       file_id
    integer         isfirst
```

DFANgetlabel

```
intn DFANgetlabel(char *filename, uint16 tag, uint16 ref, char *label_buff,
          int32 buff_len)
```

filename	IN:	Name of the HDF file
tag	IN:	Tag of the data object assigned the label
ref	IN:	Reference number of the data object assigned the label
label_buff	OUT:	The buffer allocated to hold the label
buff_len	IN:	Size of the buffer allocated to hold the label

Purpose Reads the label assigned to the data object with the given tag and reference number.

Return value Returns `SUCCEED(0)` if successful and `FAIL(-1)` otherwise.

Description The parameter *buff_len* gives the amount of space that is available for storing the label. The length of *buff_len* must be at least one greater than the anticipated length of the label, because a `'\0'` byte is appended to the annotation.

FORTRAN

```
integer function daglab(filename, tag, ref, label_buff, buff_len)
    character*(*)  filename, label_buff
    integer*2      tag, ref
    integer*4      buff_len
```

DFANgetlablen

dagllen

DFANgetlablen

```
int32 DFANgetlablen(char *filename, uint16 tag, uint16 ref)
```

filename	IN:	Name of the HDF file
tag	IN:	Tag of the data object assigned the label
ref	IN:	Reference number the data object assigned the label

Purpose — Returns the length of a label assigned to the object with a given tag and reference number.

Return value — Returns the length of the label if successful and `FAIL(-1)` otherwise.

Description — This routine should be used to insure that there is enough space allocated for a label before actually reading it.

FORTRAN

```
integer function dagllen(filename, tag, ref)
    character*(*)  filename
    integer*2      tag, ref
```

DFANlablist

```
int DFANlablist(char *filename, uint16 tag, unit16 ref_list[], char *label_list,
          int list_len, intn label_len, intn start_pos)
```

filename	IN:	Name of the HDF file
tag	IN:	Tag to use when searching for reference numbers and labels
ref_list	OUT:	The buffer allocated to hold the reference numbers
label_list	OUT:	The buffer (array of strings) allocated to hold the labels
list_len	IN:	Size of reference number list and the label list
label_len	IN:	Maximum length allowed for a label
start_pos	IN:	Entries are returned from the *start_pos* entry up to the *list_len* entry

Purpose Returns a list of all reference numbers and labels (if labels exist) for a given tag.

Return value Returns the number of reference numbers found if successful and FAIL(-1) otherwise.

Description The *list_len* determines the number of available entries in the reference number and label lists, *label_len* is the maximum length allowed for a label, and *start_pos* tells which label to start reading for the given tag. (If *start_pos* is 1, for instance, all labels will be read; if *start_pos* is 4, all but the first 3 labels will be read.) The *ref_list* contains a list of reference numbers for all objects with a given tag. The *label_list* contains a corresponding list of labels, where they exist. If there is no label stored for a given object, the corresponding entry in *label_list* is an empty string.

Taken together, the *ref_list* and *label_list* constitute a directory of all objects and their labels (where they exist) for a given tag. The list, *label_list*, can be displayed to show all of the labels for a given tag. Or, it can be searched to find the reference number of a data object with a certain label. Once the reference number for a given label is found, the corresponding data object can be accessed by invoking other HDF routines. Hence, this routine provides a mechanism for direct access to data objects in HDF files.

FORTRAN

```
integer function dallist(filename, tag, ref_list, label_list,
list_len, label_len, start_pos)
    character*(*)  filename, label_list
    integer*2      tag, ref_list(*)
    integer        list_len, label_len, start_pos
```

dalref

DFANlastref

```
uint16 DFANlastref(void)
```

Purpose	Returns the reference number for the annotation last written or read.
Return value	Returns the reference number if successful and `FAIL(-1)` otherwise.
Description	**`DFANlastref`** returns the reference number for the last annotation read or written.
FORTRAN	`integer function dalref()`

DFANputdesc

```
int DFANputdesc(char *filename, uint16 tag, uint16 ref, char *description,
          int32 desc_len)
```

filename	IN:	Name of the HDF file
tag	IN:	Tag of the data object to be assigned the description
ref	IN:	Reference number the data object to be assigned the description
description	IN:	Sequence of ASCII characters (may include NULL '\0')
desc_len	IN:	Length of the description

Purpose Writes a description for the data object with the given tag and reference number.

Return value Returns SUCCEED(0) if successful and FAIL(-1) otherwise.

Description The parameter *description* can contain any sequence of ASCII characters; it does not have to be a string.

FORTRAN

```
integer function dapdesc(filename, tag, ref, description, desc_len)
    character*(*)  filename, description
    integer*2      tag, ref
    integer*4      desc_len
```

DFANputlabel

daplab

DFANputlabel

```
intn DFANputlabel(char *filename, uint16 tag, uint16 ref, char *label)
```

filename	IN:	Name of the HDF file
tag	IN:	Tag of the data object to be assigned the label
ref	IN:	Reference number the data object to be assigned the label
label	IN:	A single string of null-terminated text

Purpose — Assigns a label to the data object with the given tag and reference number.

Return value — Returns `SUCCEED(0)` if successful and `FAIL(-1)` otherwise.

FORTRAN

```
integer function daplab(filename, tag, ref, label)
   character*(*)  filename, label
   integer*4      tag, ref
```