ANannlen/afannlen

int32 ANannlen(int32 *ann_id*)

ann_id IN: Annotation identifier returned by **ANcreate**, **ANcreatef**, or **ANselect**

Purpose Returns the length, in bytes, of the annotation specified by the given annotation identifier.

Return value Returns the length of the target annotation or `FAIL` (or `-1`) otherwise.

Description A common use of **ANannlen** is to determine if a buffer is large enough to contain the entire target annotation.

Example This example illustrates the use of **ANannlen**:

```
#include "hdf.h"

int32 an_id, ann_id, file_id, ann_len, stat;

file_id = Hopen("myfile", DFACC_READ, 0);
an_id = ANstart(file_id);
ann_id = ANselect(an_id, index, annot_type);
ann_len = ANannlen(ann_id);
  ...
stat = ANendaccess(ann_id);
stat = ANend(an_id);
Hclose(file_id);
```

FORTRAN

```
integer function afannlen(ann_id)

integer ann_id
```

ANannlist/afannlist

intn ANannlist(int32 *an_id*, ann_type *annot_type,* uint16 *obj_tag,* uint16 *obj_ref,* int32 **ann_list*)

an_id	IN:	Multifile annotation interface identifier returned by **ANstart**
annot_type	IN:	Target annotation type
obj_tag	IN:	Tag of the object the target annotation is attached to
obj_ref	IN:	Reference number of the object the target annotation is attached to
ann_list	OUT:	Buffer for the returned annotation identifiers

Purpose

Returns a list of annotation identifiers for the annotations in the file that correspond to the annotation type specified by *annot_type*, the tag specified by *obj_tag* and the reference number specified by *obj_ref*.

Return value

Returns the number of qualifying annotations or `FAIL` (or `-1`) otherwise.

Description

The identifiers in the list returned by **ANannlist** can refer to either data labels or a data descriptions. The returned identifier list is used in conjunction with **ANnumann** and **HDmalloc** (or **malloc)** to specify the size of a buffer that will be used to store the list of identifiers returned by **ANannlist** - see the example below.

Valid values for *annot_type* are: `AN_DATA_LABEL` for data labels and `AN_DATA_DESC` for data descriptions.

Note that only data labels and data descriptions (identified by the `AN_DATA_LABEL` and `AN_DATA_DESC` definitions) are supported by **ANannlist**. To return file labels or file descriptions use **ANfileinfo** to determine the number of file labels and descriptions, then use **ANselect** with **ANreadann** to read them.

Example

This example illustrates the use of **ANannlist** in returning the list of annoations attached to the second numeric data group object in the file:

```
#include "hdf.h"

int32 an_id, ann_id, file_id, *ann_list, stat;
ann_type annot_type = AN_DATA_LABEL;
uint16 obj_tag = DFTAG_NDG;
uint16 obj_ref = 2;
int num_ann;

file_id = Hopen("myfile", DFACC_READ, 0);
an_id = ANstart(file_id);
num_ann = ANnumann(an_id, annot_type, obj_tag, obj_ref);
ann_list = HDmalloc(num_ann * sizeof(int32));
num_ann = ANannlist(an_id, annot_type, obj_tag, obj_ref,
```

```
                    ann_list);
  ...
stat = ANend(an_id);
Hclose(file_id);
```

FORTRAN

```
integer function afannlist(an_id, annot_type, obj_tag, obj_ref
    ann_list)

integer ann_list(*)
integer an_id, obj_tag, obj_ref, annot_type
```

ANatype2tag/afatypetag

uint16 ANatype2tag(int32 *ann_type*)

ann_type	IN:	Annotation type

Purpose Returns the tag (prefaced by `DFTAG_`) corresponding to the specified annotation type (prefaced by `AN_`).

Return value A tag: `DFTAG_FID`, `DFTAG_FD`, `DFTAG_DIL`, `DFTAG_DIA`.

FORTRAN

```
integer function afatypetag(ann_type)

integer ann_type
```

ANcreate/afcreate

int32 ANcreate(int32 *an_id,* uint16 *tag,* uint16 *ref,* int32 *annot_type*)

an_id	IN:	Multifile data annotation interface identifier returned by **ANstart**
tag	IN:	Tag of the element the created data annotation will be applied to
ref	IN:	Reference number of the object the created data annotation will be applied to
annot_type	IN:	Type of data annotation

Purpose Creates an data annotation for the object identified by the specified tag and reference number.

Return value Returns a data annotation identifier if successful and `FAIL` (or `-1`) otherwise.

Description The data annotation identifier returned can either point to a data label or a data description.

Valid values for *annot_type* are: `AN_DATA_LABEL` for data labels and `AN_DATA_DESC` for data descriptions.

Example This example illustrates the use of **ANcreate** to create an data label for a numeric data group.

```
#include "hdf.h"

int32 an_id, ann_id, file_id, stat;
uint16 tag = DFTAG_NDG;
uint16 ref = 2;
ann_type type = AN_DATA_LABEL;

file_id = Hopen("myfile", DFACC_WRITE, 0);
an_id = ANstart(file_id);
ann_id = ANcreate(an_id, tag, ref, type);
   ...
stat = ANendaccess(ann_id);
stat = ANend(an_id);
Hclose(file_id);
```

FORTRAN

```
integer function afcreate(an_id, tag, ref, annot_type)

integer an_id, tag, ref, annot_type
```

ANcreatef/affcreate

int32 ANcreatef(int32 *an_id*, int32 *annot_type*)

an_id	IN:	Multifile annotation interface identifier returned by **ANstart**
annot_type	IN:	Type of file annotation

Purpose Creates an file label or file description annotation .

Return value Returns a file annotation identifier if successful and `FAIL` (or `-1`) otherwise.

Valid values for *annot_type* are: `AN_FILE_LABEL` for file labels and `AN_FILE_DESC` for file descriptions.

FORTRAN

```
integer function affcreate(an_id, type)

integer an_id, type
```

ANdestroy

intn ANdestroy(void)

Purpose Deallocates all internal data structures used by the multifile annotation interface.

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

ANend/afend

intn ANend(int32 *an_id*)

an_id	IN:	Multifile annotation interface identifier returned by **ANstart**

Purpose Terminates access to the multifile annotation interface.

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

Description This routine is used with the **ANstart** routine to define the extent of a multifile annotation interface session. As in other multifile interfaces, **ANend** disposes of the internal structures used by the remaining AN routines.

Example This example illustrates the use of **ANend**:

```
#include "hdf.h"

int32 an_id, file_id, stat;

file_id = Hopen("myfile", DFACC_WRITE, 0);
an_id = ANstart(file_id);
  ...
Hclose(file_id);
```

FORTRAN

```
integer function afend(an_id)

integer an_id
```

ANendaccess/afendaccess

intn ANendaccess(int32 *ann_id*)

ann_id	IN:	Annotation identifier returned by **ANcreate**, **ANcreatef** or **ANselect**

Purpose Terminates access to an annotation.

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

Description There should be one call to **ANendaccess** for every call to **ANselect**, **ANcreate** or **ANcreatef**.

Example This example illustrates the use of **ANendaccess**:

```
#include "hdf.h"

int32 an_id, ann_id, file_id, stat;
 uint16 tag = DFTAG_DIL;
 uint16 ref = 0;
 ann_type type = AN_DATA_LABEL;

file_id = Hopen("myfile", DFACC_WRITE, 0);
an_id = ANstart(file_id);
 ann_id = ANcreate(an_id, tag, ref, type);
   ...
stat = ANendaccess(ann_id);
stat = ANend(an_id);
Hclose(file_id);
```

FORTRAN

```
integer function afendaccess(ann_id)

integer ann_id
```

ANfileinfo/affileinfo

intn ANfileinfo(int32 *an_id*, int32 *n_file_label*, int32 **n_file_desc*, int32 **n_data_label*, int32 **n_data_desc*)

an_id	IN:	Multifile annotation interface identifier returned by **ANstart**
n_file_label	OUT:	Returned number of file labels
n_file_desc	OUT:	Returned number of file descriptions
n_data_label	OUT:	Returned total number of data labels
n_data_desc	OUT:	Returned total number of data descriptions

Purpose Returns the number of annotations of each type in the current file.

Return value Returns `SUCCEED` (or `0`) on successful completion or `FAIL` (or `-1`) otherwise.

Description This routine is generally used to find the range of acceptable indices for **ANselect** calls.

Note that the number of data labels and data descriptions (returned in the *n_data_label* and *n_data_desc* parameters) refer to the total number in the file, not the total number for a specific element. Use **ANnumann** to determine this number for a specific element.

Example This example illustrates the use of **ANfileinfo**:

```
#include "hdf.h"

int32 an_id, file_id, stat;
int32 n_data_label, n_data_desc, n_file_label, n_file_desc;

file_id = Hopen("myfile", DFACC_READ, 0);
an_id = ANstart(file_id);
stat = ANfileinfo(an_id, &n_file_label, &n_file_desc,
      &n_data_label, &n_data_desc);
   ...
stat = ANend(an_id);
Hclose(file_id);
```

FORTRAN

```
integer function affileinfo(an_id, n_file_label,
     n_file_desc, n_data_label, n_data_desc)

integer an_id, n_file_label, n_file_desc
integer n_data_label, n_data_desc
```

ANget_tagref/afgettagref

int32 ANget_tagref(int32 *an_id*, int32 **index*, int32 **type*, uint16 **ann_tag*, uint16 **ann_ref*)

an_id	IN:	Multifile annotation interface identifier returned by **ANstart**
index	IN:	Index of the annotation
type	IN:	Annotation type
ann_tag	OUT:	Tag of the annotation
ann_ref	OUT:	Reference number of the annotation

Purpose Returns the tag/reference number pair of the specified annotation.

Return value Returns the tag/reference number pair if successful or `FAIL` (or `-1`) otherwise.

Description The *index* parameter is zero-based.

Valid values for the *type* parameter are:

```
AN_DATA_LABEL - for data labels

AN_DATA_DESC - for data descriptions

AN_FILE_LABEL - for file labels

AN_FILE_DESC - for file descriptions
```

FORTRAN

```
integer function afgettagref(an_id, index, type,
     ann_tag, ann_ref)

integer an_id, index, type
integer ann_tag, ann_ref
```

ANid2tagref/afidtagref

int32 ANid2tagref(int32 *ann_id*, uint16 **ann_tag,* uint16 **obj_ref*)

ann_id	IN:	Annotation identifier returned by **ANselect**, **ANcreate** or **ANcreatef**
ann_tag	OUT:	Tag of the annotation
obj_ref	OUT:	Reference number of the annotation

Purpose Returns the tag/reference number pair of the specified annotation.

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

FORTRAN

```
integer function afidtagref(ann_id, ann_tag, obj_ref)

integer ann_id, ann_tag, obj_ref
```

ANnumann/afnumann

intn ANnumann(int32 *an_id*, int32 *annot_type,* uint16 *obj_tag,* uint16 *obj_ref*)

an_id	IN:	Multifile annotation interface identifier returned by **ANstart**
annot_type	IN:	Target annotation type
obj_tag	IN:	Tag of the object the target annotation is attached to
obj_ref	IN:	Reference number of the object the target annotation is attached to

Purpose Returns the total number of annotations in the file that correspond to the annotation type specified by *annot_type*, the tag specified by *obj_tag* and the reference number specified by *obj_ref.*

Return value Returns the number of qualifying annotations or `FAIL` (or `-1`) otherwise.

Description The annotations referred to by the return value of **ANnumann** can be labels and/or descriptions. The return value is often used in conjunction with **HDmalloc** or **malloc** to specify the size of a buffer that will be used to store information about the target annotations, or in conjunction with **ANannlist**.

Valid values for the *annot_type* parameter are:

```
AN_DATA_LABEL - for data labels

AN_DATA_DESC - for data descriptions
```

Note that only data labels and data descriptions (identified by the `AN_DATA_LABEL` and `AN_DATA_DESC` definitions) are supported by **ANnumann**. Use **ANfileinfo** to set file labels and file descriptions.

Example This example illustrates the use of **ANnumann** in returning the number of annotations attached to the second number data group object in the file:

```
#include "hdf.h"

int32 an_id, ann_id, file_id, stat;
 ann_type annot_type = AN_DATA_LABEL;
 uint16 obj_tag = DFTAG_NDG;
 uint16 obj_ref = 2;
 int num_ann;

file_id = Hopen("myfile", DFACC_READ, 0);
an_id = ANstart(file_id);
num_ann = ANnumann(an_id, annot_type, obj_tag, obj_ref);
   ...
stat = ANend(an_id);
Hclose(file_id);
```

FORTRAN

```
integer function afnumann(an_id, annot_type, obj_tag, obj_ref)
```

```
integer an_id, obj_tag, obj_ref, annot_type
```

ANreadann/afreadann

intn ANreadann(int32 *ann_id*, char* *textbuf*, int32 *textbuf_length*)

ann_id	IN:	Annotation identifier returned by **ANcreate**, **ANcreatef** or **ANselect**
textbuf	OUT:	Buffer for the returned annotation text
textbuf_length	IN:	Length, in bytes, of *textbuf*

Purpose Reads the annotation identified by the annotation identifier *ann_id*.

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

Description **ANreadann** reads the annotation specified by *ann_id* and places it into *textbuf*. The *textbuf_length* parameter is the size of *textbuf*. A null termination is added to the buffer as necessary - therefore, the buffer should be sized to accomodate this null termination.

Example This example illustrates the use of **ANreadann** using a statically-sized buffer:

```
#include "hdf.h"

int32 an_id, ann_id, file_id, stat;
int32 index = 0;
char textbuf[20];
ann_type annot_type = AN_FILE_DESC;

file_id = Hopen("myfile", DFACC_READ, 0);
an_id = ANstart(file_id);
ann_id = ANselect(an_id, index, annot_type);
   ...
/* Read a 17 character annotation text string, */
/*  plus a null termination cha                */
stat = ANreadann(ann_id, textbuf, 18);
   ...
stat = ANendaccess(ann_id);
stat = ANend(an_id);
Hclose(file_id);
```

FORTRAN

```
integer function afreadann(ann_id, buf, buf_length)

integer ann_id, buf_length
character* (*) buf
```

ANselect/afselect

int32 ANselect(int32 *an_id*, int32 *index*, int32 *annot_type*)

an_id	IN:	Multifile annotation interface identifier returned by **ANstart**
index	IN:	Location of the annotation in the file
annot_type	IN:	Annotation type

Purpose Selects and returns the identifier for the annotation identified by the index *index* and the annotation type *annot_type*.

Return value Returns the identifier of the selected annotation.

Description The identifier returned by **ANselect** can refer to either a label or a description. The index supplied by the parameter *index* is zero-based.

Valid values for the *annot_type* parameter are:

```
AN_DATA_LABEL - for data labels

AN_DATA_DESC - for data descriptions

AN_DATA_LABEL - for data labels

AN_DATA_DESC - for data descriptions
```

Example This example illustrates the use of **ANselect** in returning the fourth file label in a file:

```
#include "hdf.h"

int32 an_id, ann_id, file_id, stat;
ann_type annot_type = AN_FILE_LABEL;
int32 index = 4;

file_id = Hopen("myfile", DFACC_READ, 0);
an_id = ANstart(file_id);
ann_id = ANselect(an_id, index, annot_type);
  ...
stat = ANendaccess(ann_id);
stat = ANend(an_id);
Hclose(file_id);
```

FORTRAN

```
integer function afselect(an_id, index, annot_type)

integer an_id, index
integer annot_type
```

ANstart/afstart

int32 ANstart(int32 *file_id*)

file_id	IN:	File identifier returned by **Hopen**

Purpose Initializes the multifile annotation interface.

Return value Returns an interface identifier if successful and `FAIL` (or `-1`) otherwise.

Description This routine is used with the **ANend** routine to define the extent of a multifile annotation interface session. **ANstart** initializes the internal interface structures needed for the remaining AN routines. Use the general purpose routines **Hopen** and **Hclose** to manage file access as the AN routines will not open and close HDF files.

Example This example illustrates the use of **ANstart** and **ANend** in initializing and terminating an AN interface session.

```
#include "hdf.h"
int32 an_id, file_id, stat;

file_id = Hopen("myfile", DFACC_WRITE, 0);
an_id = ANstart(file_id);
   ...
stat = ANend(an_id);
Hclose(file_id);
```

FORTRAN

```
integer function afstart(file_id)

integer file_id
```

ANtag2atype/aftagatype

int32 ANtag2atype(uint16 *tag*)

tag	IN:	Object tag

Purpose Returns the annotation type (prefaced by `AN_`) corresponding to the specified tag (prefaced by `DFTAG_`).

Return value Annotation type

Description Valid values for the *annot_type* parameter are:

```
AN_DATA_LABEL - for data labels
AN_DATA_DESC - for data descriptions
AN_DATA_LABEL - for data labels
AN_DATA_DESC - for data descriptions
```

FORTRAN

```
integer function aftagatype(tag)
integer tag
```

ANtagref2id/aftagrefid

int32 ANtagref2id(int32 *an_id*, uint16 **tag*, uint16 *ref*)

an_id	IN:	Annotation interface identifier
tag	IN:	Tag of the annotation
ref	IN:	Reference number of the annotation

Purpose Retrieves an annotation for a specified tag/reference number pair.

Return value Returns the identifier of the annotation if successful and `FAIL` (or `-1`) otherwise.

FORTRAN

```
integer function aftagrefid(an_id, tag, ref)

integer an_id, tag, ref
```

ANwriteann/afwriteann

intn ANwriteann(int32 *ann_id*, char* *label*, int32 *ann_length*)

ann_id	IN:	Annotation identifier returned by **ANcreate**, **ANcreatef** or **ANselect**
label	IN:	Text to be written to the annotation
ann_length	IN:	Length, in bytes, of the annotation text pointed to by *label*

Purpose Writes an annotation to the current HDF file.

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

Description **ANwriteann** will overwrite any preexisting annotation containing the text identified by *label*.

Example This example illustrates the use of **ANwriteann**:

```
#include "hdf.h"

int32 an_id, stat;
 char *label = "This is annotation text.";

file_id = Hopen("myfile", DFACC_WRITE, 0);
an_id = ANstart(file_id);
 ann_id = ANcreate(an_id, tag, ref, type);
   ...
stat = ANwriteann(ann_id, label, sizeof(label));
   ...
stat = ANendaccess(ann_id);
stat = ANend(an_id);
Hclose(file_id);
```

FORTRAN

```
integer function afwriteann(ann_id, label, ann_length)

integer ann_id, ann_length
character* (*) label
```