

**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);
```

**FORTTRAN**           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*)

<i>an_id</i>	IN: Multifile annotation interface identifier returned by <b>ANstart</b>
<i>annot_type</i>	IN: Target annotation type
<i>obj_tag</i>	IN: Tag of the object the target annotation is attached to
<i>obj_ref</i>	IN: Reference number of the object the target annotation is attached to
<i>ann_list</i>	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 annotations 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

---

## 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)
```

<i>an_id</i>	IN: Multifile data annotation interface identifier returned by <b>ANstart</b>
<i>tag</i>	IN: Tag of the element the created data annotation will be applied to
<i>ref</i>	IN: Reference number of the object the created data annotation will be applied to
<i>annot_type</i>	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);
```

**FORTTRAN**

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

integer an_id, tag, ref, annot_type
```

# ANcreatef/affcreate

---

## 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 )

<b>Purpose</b>	Deallocates all internal data structures used by the multifile annotation interface.
<b>Return value</b>	Returns <code>SUCCEED</code> (or 0) if successful and <code>FAIL</code> (or -1) otherwise.

# ANend/afend

---

## 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 `SUCCESS` (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**, **ANcreatf** 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 **ANcreatf**.

**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 **SUCCESS** (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);
```

**FORTTRAN**

```
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*)

<i>an_id</i>	IN: Multifile annotation interface identifier returned by <b>ANstart</b>
<i>index</i>	IN: Index of the annotation
<i>type</i>	IN: Annotation type
<i>ann_tag</i>	OUT: Tag of the annotation
<i>ann_ref</i>	OUT: Reference number of the annotation

<b>Purpose</b>	Returns the tag/reference number pair of the specified annotation.
<b>Return value</b>	Returns the tag/reference number pair if successful or <b>FAIL</b> (or -1) otherwise.
<b>Description</b>	The <i>index</i> 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

<b>FORTTRAN</b>	integer function afgettagref( <i>an_id</i> , <i>index</i> , <i>type</i> , <i>ann_tag</i> , <i>ann_ref</i> )  integer <i>an_id</i> , <i>index</i> , <i>type</i> integer <i>ann_tag</i> , <i>ann_ref</i>
-----------------	--

# ANid2tagref/afidtagref

---

## ANid2tagref/afidtagref

int32 ANid2tagref(int32 *ann\_id*, uint16 \**ann\_tag*, uint16 \**obj\_ref*)

<i>ann_id</i>	IN: Annotation identifier returned by <b>ANselect</b> , <b>ANcreate</b> or <b>ANcreatef</b>
<i>ann_tag</i>	OUT:Tag of the annotation
<i>obj_ref</i>	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, ann_type annot_type, uint16 obj_tag, uint16 obj_ref)
```

<i>an_id</i>	IN: Multifile annotation interface identifier returned by <b>ANstart</b>
<i>annot_type</i>	IN: Target annotation type
<i>obj_tag</i>	IN: Tag of the object the target annotation is attached to
<i>obj_ref</i>	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);
```

**FORTTRAN** integer function afnumann(*an\_id*, *annot\_type*, *obj\_tag*, *obj\_ref*)

# **A**Nnumann/afnumann

---

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

**ANreadann/afreadann**

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

<i>ann_id</i>	IN: Annotation identifier returned by <b>ANcreate</b> , <b>ANcreatf</b> or <b>ANselect</b>
<i>textbuf</i>	OUT: Buffer for the returned annotation text
<i>textbuf_length</i>	IN: Length, in bytes, of <i>textbuf</i>

**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 character */
stat = ANreadann(ann_id, textbuf, 18);
...
stat = ANendaccess(ann_id);
stat = ANend(an_id);
Hclose(file_id);
```

<b>FORTRAN</b>	integer function <code>afreadann(ann_id, buf, buf_length)</code>
	integer <code>ann_id</code> , <code>buf_length</code>
	character* (*) <code>buf</code>

# ANselect/afselect

---

## ANselect/afselect

int32 ANselect(int32 *an\_id*, int32 *index*, ann\_type *annot\_type*)

<i>an_id</i>	IN: Multifile annotation interface identifier returned by <b>ANstart</b>
<i>index</i>	IN: Location of the annotation in the file
<i>annot_type</i>	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);
```

**FORTTRAN**

```
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

---

## 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*)

<i>an_id</i>	IN: Annotation interface identifier
<i>tag</i>	IN: Tag of the annotation
<i>ref</i>	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.

**FORTTRAN**

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

# ANwriteann/afwriteann

---

## 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);
```

**FORTTRAN**           integer function afwriteann(ann\_id, label, ann\_length)

                  integer ann\_id, ann\_length

                  character\* (\*) label