H5O_GET_INFO3

Expand all Collapse all

- Jump to ...
- Summary
- Description
- Example
- Switch language ...
- C
- C++
- FORTRAN
- JAVA

Summary
Description
Example
JAVA
FORTRAN
C++
C

H5O_GET_INFO3

Retrieves the metadata for an object specified by an identifier

Procedure:

H5O_GET_INFO3 (loc_id, oinfo, fields)

Signature:

```
herr_t H5Oget_info3 ( hid_t loc_id, H5O_info2_t *oinfo, unsigned fields )

SUBROUTINE h5oget_info_f(object_id, object_info, hdferr, fields)

    USE, INTRINSIC :: ISO_C_BINDING
    IMPLICIT NONE
    INTEGER(HID_T)  , INTENT(IN)            :: object_id
    TYPE(h5o_info_t), INTENT(OUT), TARGET   :: object_info
    INTEGER         , INTENT(OUT)           :: hdferr
    INTEGER         , INTENT(IN), OPTIONAL  :: fields
```

Related Fortran2003 Derived Type: h5o_info_t

```
TYPE, BIND(C) :: h5o_info_t
   INTEGER(C_LONG)     :: fileno     ! File number that object is located in
   TYPE(H5O_TOKEN_T_F) :: token      ! Token for object in file
   INTEGER(C_INT)      :: type       ! Basic object type (group, dataset, etc.)
   INTEGER             :: rc         ! Reference count of object

   INTEGER, DIMENSION(8) :: atime ! Access time         !    -- NOTE --
   INTEGER, DIMENSION(8) :: mtime ! Modification time   ! Returns an integer array
   INTEGER, DIMENSION(8) :: ctime ! Change time         ! as specified in the Fortran
   INTEGER, DIMENSION(8) :: btime ! Birth time          ! intrinsic DATE_AND_TIME(VALUES)

   INTEGER(hsize_t) :: num_attrs  ! # of attributes attached to object
END TYPE h5o_info_t
```

Parameters:

hid_t `loc_id`	IN: Identifier for object of type specified by *H5O_type_t*; may be a file, group, dataset, named datatype or attribute identifier
H5O_info2_t * `oinfo`	OUT: Buffer in which to return object information
unsigned int `fields`	IN: Flags specifying the fields to include in `oinfo`

Description:

H5O_GET_INFO3 specifies an object by its identifier, `loc_id` , and retrieves the metadata describing that object in `oinfo` , an *H5O_info2_t* st ruct.

The *H5O_info2_t* struct is defined (in `H5Opublic.h`) as follows :

src / H5Opublic.h [129:141] hdf5_1_12 HDFFV/hdf5

```
} mesg;
} H5O_hdr_info_t;

/* Data model information struct for objects */
/* (For H5Oget_info / H5Oget_info_by_name / H5Oget_info_by_idx version 3) */
typedef struct H5O_info2_t {
    unsigned long fileno;    /* File number that object is located in */
    H5O_token_t   token;     /* Token representing the object        */
    H5O_type_t    type;      /* Basic object type (group, dataset, etc.) */
    unsigned      rc;        /* Reference count of object            */
    time_t        atime;     /* Access time                          */
    time_t        mtime;     /* Modification time                    */
    time_t        ctime;     /* Change time                          */
```

Note the following about *H50_info2_t*:

- Of the four time fields (`atime`, `mtime`, `ctime`, and `btime`) only `ctime` has been implemented.
- The `atime` value is the last time the object was read or written.
- The `mtime` value is the last time the raw data in the object was changed.
- The `ctime` value is the last time the metadata for the object was changed.
- The `btime` value is the time the object was created.

H5O_token_t is defined in `H5public.h` as follows:

src / H5public.h [337:341] hdf5_1_12 HDFFV/hdf5

```
/*
 * Allocation statistics info struct
 */
typedef struct H5_alloc_stats_t {
    unsigned long long total_alloc_bytes;        /* Running count of total # of bytes
allocated */
```

The *H5O_type_t* enum indicates the object type and is defined (in `H5Opublic.h`) as follows:

src / H5Opublic.h [101:109] hdf5_1_12 HDFFV/hdf5

```
/* Public Typedefs */
/*******************/

/* Types of objects in file */
typedef enum H5O_type_t {
    H5O_TYPE_UNKNOWN = -1,   /* Unknown object type  */
    H5O_TYPE_GROUP,          /* Object is a group  */
    H5O_TYPE_DATASET,        /* Object is a dataset  */
    H5O_TYPE_NAMED_DATATYPE, /* Object is a named data type */
```

Note that `object_id` refers only to the types specified by *H5O_type_t*.

The `fields` parameter contains flags to determine which fields will be filled in in the *H5O_info2_t* struct returned in `oinfo`. These flags are defined in the `H5Opublic.h` file:

Flag	Purpose
H5O_INFO_BASIC	Fill in the fileno, addr, type, and rc fields
H5O_INFO_TIME	Fill in the atime, mtime, ctime, and btime fields
H5O_INFO_NUM_ATTRS	Fill in the num_attrs field
H5O_INFO_HDR	Fill in the hdr field
H5O_INFO_META_SIZE	Fill in the meta_size field
H5O_INFO_ALL	H5O_INFO_BASIC | H5O_INFO_TIME | H5O_INFO_NUM_ATTRS | H5O_INFO_HDR | H5O_INFO_META_SIZE

Note:

If you are iterating through a lot of different objects to retrieve information via the H5O_GET_INFO family of routines, you may see memory building up. This can be due to memory allocation for metadata such as object headers and messages when the iterated objects are put into the metadata cache.

If the memory buildup is not desirable, you can configure a smaller cache via H5F_SET_MDC_CONFIG or set the file access property list via H5P_SET_MDC_CONFIG. A smaller sized cache will force metadata entries to be evicted from the cache, thus freeing the memory associated with the entries.

Returns:

Returns a non-negative value if successful; otherwise returns a negative value.

Example:

examples / h5_attribute.c [189:205] hdf5_1_12 HDFFV/hdf5

```
*/
    ret = H5Oget_info3(dataset, &oinfo, H5O_INFO_NUM_ATTRS);
    for (i = 0; i < (unsigned)oinfo.num_attrs; i++) {
        attr       = H5Aopen_by_idx(dataset, ".", H5_INDEX_CRT_ORDER, H5_ITER_INC,
(hsize_t)i, H5P_DEFAULT,
                                    H5P_DEFAULT);
        atype      = H5Aget_type(attr);
        type_class = H5Tget_class(atype);
        if (type_class == H5T_STRING) {
            atype_mem = H5Tget_native_type(atype, H5T_DIR_ASCEND);
            ret       = H5Aread(attr, atype_mem, string_out);
            printf("Found string attribute; its index is %d , value =   %s \n", i,
string_out);
            ret = H5Tclose(atype_mem);
        }
        ret = H5Aclose(attr);
        ret = H5Tclose(atype);
    }
```

fortran / test / tH5O_F03.F90 [347:355] 1.12/master HDFFV/hdf5

```
! Check h5oget_info
  CALL h5oget_info_f(grp, oinfo, error)
  CALL check("h5oget_info_f", error, total_error)
  IF(oinfo%rc.NE.1)THEN
     CALL check("h5oget_info_f", -1, total_error)
  ENDIF
  IF(oinfo%type.NE.H5O_TYPE_GROUP_F)THEN
     CALL check("h5oget_info_f", -1, total_error)
  ENDIF
```

History:

Release	Change
1.12.0	Function was introduced in this release.

--- Last Modified: March 13, 2020 | 11:06 AM