H5G_ITERATE

Expand all Collapse all

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

Summary
Description
Example
JAVA
FORTRAN
C++
C

H5G_ITERATE

Iterates an operation over the entries of a group

> *This function is deprecated in favor of the function H5L_ITERATE.*

Procedure:

H5G_ITERATE(loc_id, name, idx, operator, operator_data)

Signature:

```
int H5Giterate(hid_t loc_id, const char *name, int *idx, H5G_iterate_t operator, void *operator_data )
```

There is no direct FORTRAN counterpart for the C function `H5Giterate`. Instead, that functionality is provided by two FORTRAN functions:

```
SUBROUTINE h5gn_members_f(loc_id, name, nmembers, hdferr)
  IMPLICIT NONE
  INTEGER(HID_T), INTENT(IN) :: loc_id        ! File or group identifier
  CHARACTER(LEN=*), INTENT(IN) :: name        ! Name of the group
  INTEGER, INTENT(OUT) :: nmembers            ! Number of members in the group
  INTEGER, INTENT(OUT) :: hdferr              ! Error code
                                              ! 0 on success and -1 on failure
END SUBROUTINE h5gn_members_f
```

```
SUBROUTINE h5gget_obj_info_idx_f(loc_id, name, idx, &
                                 obj_name, obj_type, hdferr)
  IMPLICIT NONE
  INTEGER(HID_T), INTENT(IN) :: loc_id        ! File or group identifier
  CHARACTER(LEN=*), INTENT(IN) :: name        ! Name of the group
  INTEGER, INTENT(IN) :: idx                  ! Index of member object
  CHARACTER(LEN=*), INTENT(OUT) :: obj_name   ! Name of the object
  INTEGER, INTENT(OUT) :: obj_type            ! Object type :
                                              !     H5G_LINK_F
                                              !     H5G_GROUP_F
                                              !     H5G_DATASET_F
                                              !     H5G_TYPE_F
  INTEGER, INTENT(OUT) :: hdferr              ! Error code
                                              ! 0 on success and -1 on failure
END SUBROUTINE h5gget_obj_info_idx_f
```

Parameters:

hid_t `loc_id`	IN: File or group identifier
const char `*name`	IN: Group over which the iteration is performed
int `*idx`	IN/OUT: Location at which to begin the iteration
H5G_iterate_t `operator`	IN: Operation to be performed on an object at each step of the iteration
void `*operator_data`	IN/OUT: Data associated with the operation

Description:

H5G_ITERATE iterates over the members of `name` in the file or group specified with `loc_id`. For each object in the group, the `operator_data` and some additional information, specified below, are passed to the `operator` function. The iteration begins with the `idx` object in the group and the next element to be processed by the operator is returned in `idx`. If `idx` is NULL, then the iterator starts at the first group member; since no stopping point is returned in this case, the iterator cannot be restarted if one of the calls to its operator returns non-zero. H5G_ITERATE does not recursively follow links into subgroups of the specified group.

The prototype for `H5G_iterate_t` is:

	`typedef` *herr_t* (`*H5G_iterate_t`) (*hid_t* `group_id`, *const char* `*member_name`, *void* `*operator_data`);

The operation receives the group identifier for the group being iterated over, `group_id`, the name of the current object within the group, `member_name`, and the pointer to the operator data passed in to H5G_ITERATE, `operator_data`.

The return values from an operator are:

- Zero causes the iterator to continue, returning zero when all group members have been processed.
- Positive causes the iterator to immediately return that positive value, indicating short-circuit success. The iterator can be restarted at the next group member.
- Negative causes the iterator to immediately return that value, indicating failure. The iterator can be restarted at the next group member.

H5G_ITERATE assumes that the membership of the group identified by `name` remains unchanged through the iteration. If the membership changes during the iteration, the function's behavior is undefined.

H5G_ITERATE is not recursive. In particular, if a member of `name` is found to be a group, call it `subgroup_a`, H5G_ITERATE does not examine the members of `subgroup_a`. When recursive iteration is required, the application must handle the recursion, explicitly calling H5G_ITERATE on discovered subgroups.

Returns:

Returns the return value of the last operator if it was non-zero, or zero if all group members were processed. Otherwise returns a negative value.

Example:

Coming soon!

History:

Release	Change
1.8.0	Function deprecated in this release.

--- Last Modified: April 25, 2019 | 11:32 AM