H5L_REGISTER

Expand all Collapse all

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

Summary
Description
Example
JAVA
FORTRAN
C++
C

H5L_REGISTER

Registers a user-defined link class or changes behavior of an existing class

Procedure:

H5L_REGISTER(link_class)

Signature:

```
herr_t H5Lregister( const H5L_class_t * link_class )
```

Parameters:

const H5L_class_t `*link_class`	IN: Pointer to a buffer containing the struct describing the user-defined link class

Description:

H5L_REGISTER registers a class of user-defined links, or changes the behavior of an existing class.

`link_class` is a pointer to a buffer containing a copy of the `H5L_class_t` struct. This struct is defined in `H5Lpublic.h` as follows:

```
typedef struct H5L_class_t {
      int version;                    /* Version number of this struct  */
      H5L_type_t class_id;            /* Link class identifier          */
      const char *comment;            /* Comment for debugging          */
      H5L_create_func_t create_func;  /* Callback during link creation  */
      H5L_move_func_t move_func;      /* Callback after moving link     */
      H5L_copy_func_t copy_func;      /* Callback after copying link    */
      H5L_traverse_func_t trav_func;  /* The main traversal function    */
      H5L_delete_func_t del_func;     /* Callback for link deletion     */
      H5L_query_func_t query_func;    /* Callback for queries           */
  } H5L_class_t;
```

The class definition passed with `link_class` must include at least the following:

- An `H5L_class_t` version (which should be `H5L_LINK_CLASS_T_VERS`)
- A link class identifier, `class_id`
- A traversal function, `trav_func`

Remaining struct members are optional and may be passed as `NULL`.

The link class passed in `class_id` must be in the user-definable range between `H5L_TYPE_UD_MIN` and `H5L_TYPE_UD_MAX` (see the “Link Class Identifiers...” table below) and will override any existing link class with that identifier.

As distributed, valid values of `class_id` used in HDF5 include the following (defined in `H5Lpublic.h`):

`H5L_TYPE_HARD`	Hard link
`H5L_TYPE_SOFT`	Soft link
`H5L_TYPE_EXTERNAL`	External link

The hard and soft link class identifiers cannot be modified or reassigned, but the external link class is implemented as an example in the user-definable link class identifier range. H5L_REGISTER is used to register additional link classes. It could also be used to modify the behavior of the external link class, though that is not recommended.

The following table summarizes existing link types and values and the reserved and user-definable link class identifier value ranges.

Link Class Identifiers or Value Ranges, Descriptions, and Class Names

Link class identifier or Value range	Description	Link class or other labl
`0 to 63`	Reserved range	
`64 to 255`	User-definable range	
`64`	Minimum user-defined value	`H5L_TYPE_UD_MIN`
`64`	External link	`H5L_TYPE_EXTERNAL`
`255`	Maximum user-defined value	`H5L_TYPE_UD_MAX`
`255`	Maximum value	`H5L_TYPE_MAX`
`-1`	Error	`H5L_TYPE_ERROR`

Note that HDF5 internally registers user-defined link classes only by the numeric value of the link class identifier. An application, on the other hand, will generally use a name for a user-defined class, if for no other purpose than as a variable name. Assume, for example, that a complex link type is registered with the link class identifier `73` and that the code includes the following assignment:

```
H5L_TYPE_COMPLEX_A = 73
```

The application can refer to the link class with a term, `H5L_TYPE_COMPLEX_A`, that conveys meaning to a human reviewing the code, while HDF5 recognizes it by the more cryptic numeric identifier, `73`.

Important details and considerations include the following:

- If you plan to distribute files or software with a user-defined link class, please contact the Help Desk at The HDF Group to help

prevent collisions between `class_id` values. See below.

- As distributed with HDF5, the external link class is implemented as an example of a user-defined link class with `H5L_LINK_EXTERNAL` equal to `H5L_LINK_UD_MIN`. `class_id` in the `H5L_class_t` struct must not equal `H5L_LINK_UD_MIN` unless you intend to overwrite or modify the behavior of external links.
- H5L_REGISTER can be used only with link class identifiers in the user-definable range (see table above).
- The hard and soft links defined by the HDF5 library, `H5L_TYPE_HARD` and `H5L_TYPE_SOFT`, reside in the reserved range below `H5L_TYPE_UD_MIN` and cannot be redefined or modified.
- H5L_IS_REGISTERED can be used to determine whether a desired link class identifier is available.
  Note that this function will tell you only whether the link class identifier has been registered with the installed copy of HDF5; it cannot tell you whether the link class has been registered with The HDF Group.

- `H5L_TYPE_MAX` is the maximum allowed value for a link type identifier.
- `H5L_TYPE_UD_MIN` equals `H5L_TYPE_EXTERNAL`.
- `H5L_TYPE_UD_MAX` equals `H5L_TYPE_MAX`.
- `H5L_TYPE_ERROR` indicates that an error has occurred.

Registration with The HDF Group:

There are sometimes reasons to take a broader approach to registering a user-defined link class than just invoking H5L_REGISTER. For example:

- A user-defined link class is intended for use across an organization, among collaborators, or across a community of users.
- An application or library overlying HDF5 invokes a user-defined link class that must be shipped with the software.
- Files are distributed that make use of a user-defined link class.
- Or simply, a specific user-defined link class is thought to be widely useful.

In such cases, you are encouraged to register that link class with The HDF Group's Helpdesk. The HDF Group maintains a registry of known user-defined link classes and tracks the selected link class identifiers. This registry is intended to reduce the risk of collisions between `class_id` values and to help coordinate the use of specialized link classes.

Returns:

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

Example:

Coming Soon!

History:

Release	Change
1.8.0	Function introduced in this release.

--- Last Modified: April 25, 2019 | 12:52 PM