H5L_GET_VAL

H5L_GET_VAL

Returns the value of a symbolic link

Procedure:

H5L_GET_VAL(link_loc_id, link_name, linkval_buff, size, lapl_id)

Signature:

```
herr_t H5Lget_val( hid_t link_loc_id, const char *link_name, void *linkval_buff, size_t size, hid_t lapl_id
)
```

Parameters:

hid_t `link_loc_id`	IN: Location identifier; may be a file, group, dataset, named datatype or attribute identifier
const char `*link_name`	IN: Link whose value is to be returned
void `*linkval_buff`	OUT: The buffer to hold the returned link value
size_t `size`	IN: Maximum number of characters of link value to be returned
hid_t `lapl_id`	IN: List access property list identifier

Description:

H5L_GET_VAL returns the link value of the link `link_name`.

The parameter `link_loc_id` is a location identifier.

`link_name` identifies a symbolic link and is defined relative to `link_loc_id`. Symbolic links include soft and external links and some user-defined links. This function is not for use with hard links.

The link value is returned in the buffer `linkval_buff`. For soft links, this is the path to which the link points, including the null terminator; for external and user-defined links, it is the link buffer.

`size` is the size of `linkval_buff` and should be the size of the link value being returned. This size value can be determined through a call to H5L_GET_INFO; it is returned in the `val_size` field of the `H5L_info_t` struct.

If `size` is smaller than the size of the returned value, then the string stored in `linkval_buff` will be truncated to `size` bytes. For soft links, this means that the value will not be null terminated.

In the case of external links, the target file and object names are extracted from `linkval_buff` by calling H5L_UNPACK_ELINK_VAL.

The link class of `link_name` can be determined with a call to H5L_GET_INFO.

`lapl_id` specifies the link access property list associated with the link `link_name`. In the general case, when default link access properties are acceptable, this can be passed in as `H5P_DEFAULT`. An example of a situation that requires a non-default link access property list is when the link is an external link; an external link may require that a link prefix be set in a link access property list (see H5P_SET_ELINK_PREFIX).

This function should be used only after H5L_GET_INFO has been called to verify that `link_name` is a symbolic link. This can be deteremined from the `link_type` field of the `H5L_info_t` struct.

Returns:

Returns a non-negative value, with the link value in `linkval_buff`, 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 | 12:47 PM