H5P_REGISTER1

H5P_REGISTER1

Registers a permanent property with a property list class (DEPRECATED)

Procedure:

H5P_REGISTER1 (class, name, size, default, create, set, get, delete, copy, close)

Signature:

```
herr_t H5Pregister1(
                  hid_t class,
                  const char * name,
                  size_t size,
                  void * default,
                  H5P_prp_create_func_t create,
                  H5P_prp_set_func_t set,
                  H5P_prp_get_func_t get,
                  H5P_prp_delete_func_t delete,
                  H5P_prp_copy_func_t copy,
                  H5P_prp_close_func_t close
     )
```

```
Fortran90 Interface: h5pregister_f

SUBROUTINE h5pregister_f
  IMPLICIT NONE
  INTEGER(HID_T), INTENT(IN) :: class   ! Property list class identifier
  CHARACTER(LEN=*), INTENT(IN) :: name  ! Name of property to register
  INTEGER(SIZE_T), INTENT(IN) :: size   ! Size of the property value
  TYPE,   INTENT(IN) :: value          ! Property value
                                        ! Supported types are:
                                        !    INTEGER
                                        !    REAL
                                        !    DOUBLE PRECISION
                                        !    CHARACTER(LEN=*)
  INTEGER, INTENT(OUT) :: hdferr        ! Error code
                                        ! 0 on success and -1 on failure
END SUBROUTINE h5pregister_f
```

Parameters:

hid_t `class`	IN: Property list class to register permanent property within
const char * `name`	IN: Name of property to register
size_t `size`	IN: Size of property in bytes
void * `default`	IN: Default value for property in newly created property lists
H5P_prp_create_func_t `create`	IN: Callback routine called when a property list is being created and the property value will be initialized
H5P_prp_set_func_t `set`	IN: Callback routine called before a new value is copied into the property's value
H5P_prp_get_func_t `get`	IN: Callback routine called when a property value is retrieved from the property
H5P_prp_delete_func_t `delete`	IN: Callback routine called when a property is deleted from a property list
H5P_prp_copy_func_t `copy`	IN: Callback routine called when a property is copied from a property list
H5P_prp_close_func_t `close`	IN: Callback routine called when a property list is being closed and the property value will be disposed of

Description:

`H5P_REGISTER1` registers a new property with a property list class. The property will exist in all property list objects of `class` created after this routine finishes. The name of the property must not already exist, or this routine will fail. The default property value must be provided and all new property lists created with this property will have the property value set to the default value. Any of the callback routines may be set to NULL if they are not needed.

Zero-sized properties are allowed and do not store any data in the property list. These may be used as flags to indicate the presence or absence of a particular piece of information. The default pointer for a zero-sized property may be set to NULL. The property `create` and `close` callbacks are called for zero-sized properties, but the `set` and `get` callbacks are never called.

The `create` routine is called when a new property list with this property is being created. The `H5P_prp_create_func_t` callback function is defined as follows:

typedef herr_t(*`H5P_prp_create_func_t`)(*const char* *`name`,*size_t* `size`,*void* *`initial_value`);

The parameters to this callback function are defined as follows:

const char *`name`	IN: The name of the property being modified

size_t size	IN: The size of the property in bytes
void *initial_value	IN/OUT: The default value for the property being created, which will be passed to H5Pregister1

The create routine may modify the value to be set and those changes will be stored as the initial value of the property. If the create routine returns a negative value, the new property value is not copied into the property and the create routine returns an error value.

The set routine is called before a new value is copied into the property. The H5P_prp_set_func_t callback function is defined as follows:

typedef herr_t(*H5P_prp_set_func_t)(*hid_t* prop_id,*const char* *name,*size_t* size,*void* *new_value);

The parameters to this callback function are defined as follows:

hid_t prop_id	IN: The identifier of the property list being modified
const char *name	IN: The name of the property being modified
size_t size	IN: The size of the property in bytes
void **new_value	IN/OUT: Pointer to new value pointer for the property being modified

The set routine may modify the value pointer to be set and those changes will be used when setting the property's value. If the set routine returns a negative value, the new property value is not copied into the property and the set routine returns an error value. The set routine will not be called for the initial value, only the create routine will be called.

Note: The set callback function may be useful to range check the value being set for the property or may perform some transformation or translation of the value set. The get callback would then reverse the transformation or translation. A single get or set callback could handle multiple properties by performing different actions based on the property name or other properties in the property list.

The get routine is called when a value is retrieved from a property value. The H5P_prp_get_func_t callback function is defined as follows:

typedef herr_t(*H5P_prp_get_func_t)(*hid_t* prop_id,*const char* *name,*size_t* size,*void* *value);

The parameters to the callback function are defined as follows:

hid_t prop_id	IN: The identifier of the property list being queried
const char * name	IN: The name of the property being queried
size_t size	IN: The size of the property in bytes
void * value	IN/OUT: The value of the property being returned

The get routine may modify the value to be returned from the query and those changes will be returned to the calling routine. If the set routine returns a negative value, the query routine returns an error value.

The delete routine is called when a property is being deleted from a property list. The H5P_prp_delete_func_t callback function is defined as follows:

typedef herr_t(*H5P_prp_delete_func_t)(*hid_t* prop_id,*const char* *name,*size_t* size,*void* *value);

The parameters to the callback function are defined as follows:

hid_t prop_id	IN: The identifier of the property list the property is being deleted from
const char * name	IN: The name of the property in the list
size_t size	IN: The size of the property in bytes
void * value	IN: The value for the property being deleted

The delete routine may modify the value passed in, but the value is not used by the library when the delete routine returns. If the delete routine returns a negative value, the property list delete routine returns an error value but the property is still deleted.

The copy routine is called when a new property list with this property is being created through a copy operation. The H5P_prp_copy_func_t callback function is defined as follows:

typedef herr_t(*H5P_prp_copy_func_t)(*const char* *name,*size_t* size,*void* *value);

The parameters to the callback function are defined as follows:

const char *name	IN: The name of the property being copied
size_t size	IN: The size of the property in bytes
void *value	IN/OUT: The value for the property being copied

The copy routine may modify the value to be set and those changes will be stored as the new value of the property. If the copy routine returns a negative value, the new property value is not copied into the property and the copy routine returns an error value.

The close routine is called when a property list with this property is being closed. The H5P_prp_close_func_t callback function is defined as follows:

typedef herr_t(*H5P_prp_close_func_t)(*hid_t* prop_id,*const char* *name,*size_t* size,*void* *value);

The parameters to the callback function are defined as follows:

hid_t prop_id	IN: The identifier of the property list being closed
const char *name	IN: The name of the property in the list
size_t size	IN: The size of the property in bytes
void *value	IN: The value for the property being closed

The close routine may modify the value passed in, but the value is not used by the library when the close routine returns. If the close routine returns a negative value, the property list close routine returns an error value but the property list is still closed.

Programming Note for C++ Developers Using C Functions:

If a C routine that takes a function pointer as an argument is called from within C++ code, the C routine should be returned from normally.

Examples of this kind of routine include callbacks such as H5P_SET_ELINK_CB and H5P_SET_TYPE_CONV_CB and functions such as H5T_CONVERT and H5E_WALK2.

Exiting the routine in its normal fashion allows the HDF5 C library to clean up its work properly. In other words, if the C++ application jumps out of the routine back to the C++ “catch” statement, the library is not given the opportunity to close any temporary data structures that were set up when the routine was called. The C++ application should save some state as the routine is started so that any problem that occurs might be diagnosed.

Returns:

Success: a non-negative value Failure: a negative value

Example:

Coming Soon!

History:

Release	Change
1.8.0	Function H5P_REGISTER renamed to H5P_REGISTER1 and deprecated in this release.

--- Last Modified: July 28, 2020 | 11:07 AM