Software Changes from Release to Release for HDF5-1.10

(For a description of the major new features that were introduced, please see New Features in HDF5 Release 1.10.)

This page provides information on the changes that a maintenance developer needs to be aware of between successive releases of HDF5, such as:

- New or changed features or tools
- Syntax and behavioral changes in the existing application programming interface (the API)
- Certain types of changes in configuration or build processes

Note that bug fixes and performance enhancements in the C library are automatically picked up by the C++, Fortran, and Java libraries.

The following information is included below.

- Compatiblity and Performance Issues
- Release 1.10.7 versus 1.10.6
- Release 1.10.6 versus 1.10.5
- Release 1.10.5 versus 1.10.4, 1.10.3, and 1.10.2
- Release 1.10.4 versus Release 1.10.3
- Release 1.10.3 versus Release 1.10.2
- Release 1.10.2 versus Release 1.10.1
- Release 1.10.1 versus Release 1.10.0 (and 1.10.0-patch1)
- Release 1.10.0 of March 2016 versus Release 1.8.16

## Compatibility and Performance Issues

Not all HDF5-1.10 releases are compatible. **Users should NOT be using 1.10 releases prior to HDF5-1.10.3.** See the compatibility matrix below for details on compatibility between 1.10 releases:

Release	1.10.5+	1.10.4	1.10.3	1.10.2	1.10.1	1.10.0-patch1	1.10.0
1.10.5+		Yes	Yes	No	No	No	No
1.10.4	Yes		Yes	No	No	No	No
1.10.3	Yes	Yes		No	No	No	No
1.10.2	No	No	No		No	No	No

The following images show how performance has changed from release to release.
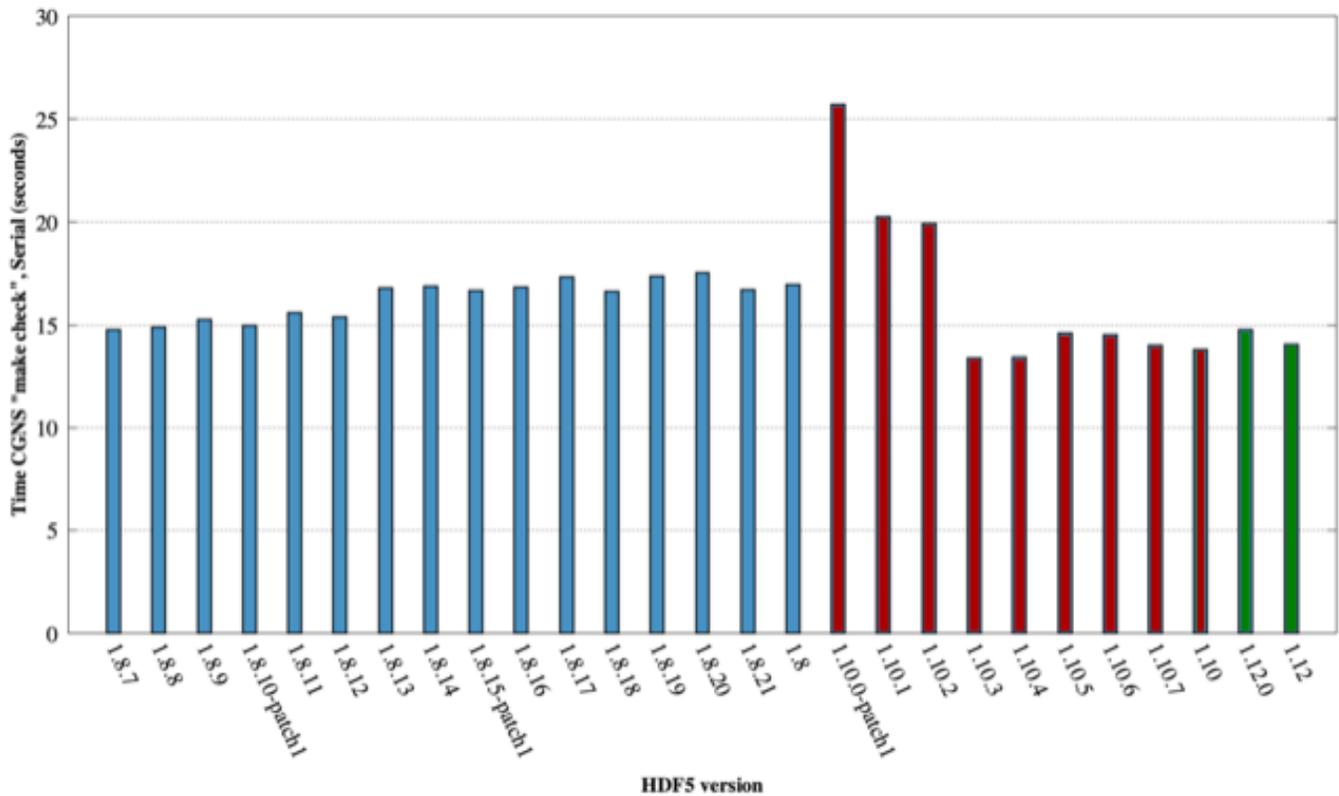
writeLargeNumDsets.cpp, 4 runs per HDF5 version, Jelly (CentOS 7)
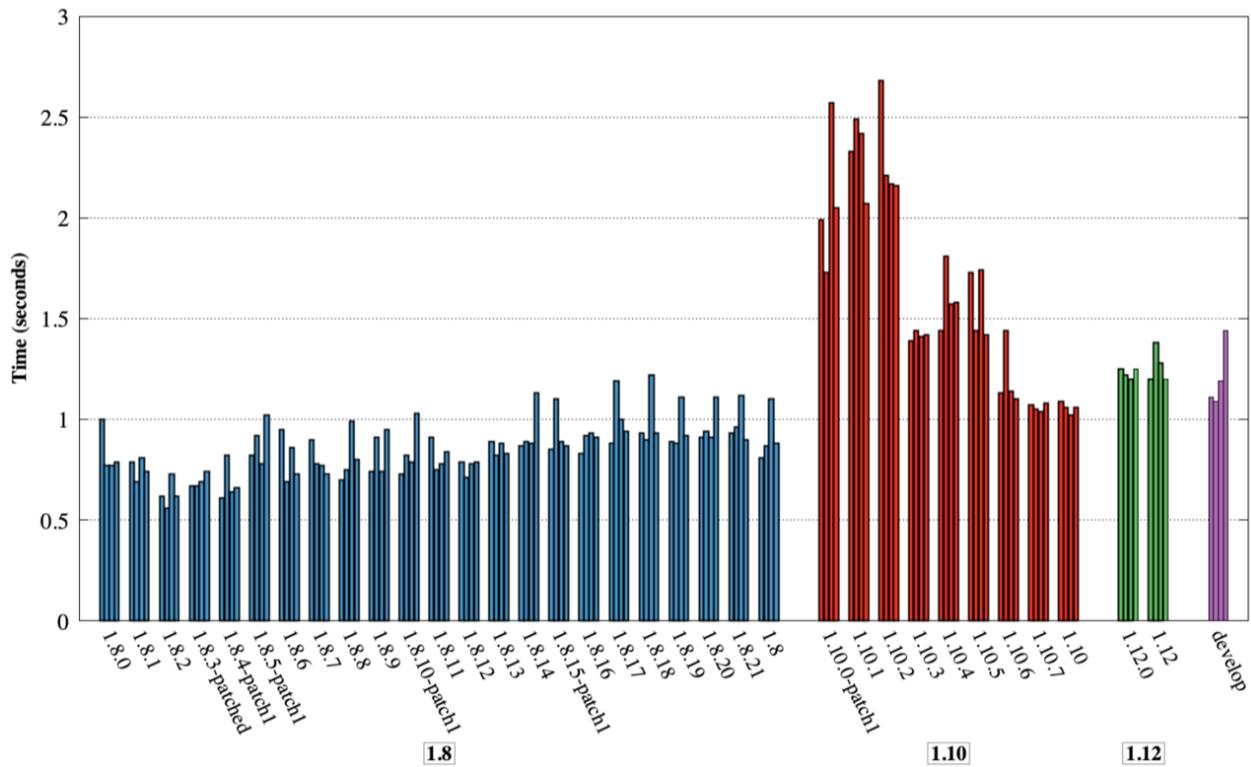
The release notes also list changes made to the library, but these notes tend to be more at a more detail-oriented level. The release notes may include new features, bugs fixed, supported configuration features, platforms on which the library has been tested, and known problems. The release note files are listed below and can be found at the top level of the HDF5 source code tree in the `release_docs` directory.

Release Notes	Technical notes regarding the current release of the HDF5 library (`RELEASE.txt` in the source code)

`HISTORY-1_10.txt`	Release information for all HDF5-1.10 releases
`HISTORY-1_8_0-1_10_0.txt`	Development history between the HDF5-1.8.0 and HDF5-1.10.0 releases
`HISTORY-1_8.txt`	Release information for HDF5-1.8.0 through HDF5-1.8.17 releases
`HISTORY-1_0-1_8_0_rc3.txt`	Technical notes starting with HDF5-1.0.0 and ending with HDF5-1.8.0-rc3 (the state of the code prior to the HDF5-1.8.0 release)

Release 1.10.7 versus 1.10.6

New and Changed Functions, Classes, Subroutines, Wrappers, and Macros

In the C Interface (main library)

The following are new C functions in this release:

Function	Description
H5P_GET_FAPL_SPLITTER	Retrieves information for a splitter file access property list
H5P_SET_FAPL_SPLITTER	Sets the file access property list to use the splitter driver
H5P_GET_FILE_LOCKING	Gets the file locking property values
H5P_SET_FILE_LOCKING	Sets the file locking property values
H5_GET_ALLOC_STATS	Gets the memory allocation statistics for the library
H5_GET_FREE_LIST_SIZES	Gets the current size of the free lists used to manage memory
H5S_COMBINE_HYPERSLAB	Performs an operation on a hyperslab and an existing selection and returns the resulting selection
H5S_COMBINE_SELECT	Combines two hyperslab selections with an operation, returning a dataspace with the resulting selection
H5S_MODIFY_SELECT	Refines a hyperslab selection with an operation using a second hyperslab to modify it
H5S_SELECT_ADJUST	Adjusts a selection by subtracting an offset
H5S_SELECT_COPY	Copies a selection from one dataspace to another
H5S_SELECT_INTERSECT_BLOCK	Checks if current selection intersects with a block
H5S_SELECT_PROJECT_INTERSECTION	Projects the intersection of two source selections to a destination selection
H5S_SELECT_SHAPE_SAME	Checks if two selections are the same shape

In the C++ Wrapper

The following C++ wrappers were added:

Wrapper	Description
FileAccPropList::getFileLocking	See H5P_GET_FILE_LOCKING for details
FileAccPropList::setFileLocking	See H5P_SET_FILE_LOCKING for details

Compatibility Notes and Reports

See the API compatibility report for the HDF5 library between 1.10.6 and 1.10.7 for information regarding compatibility with the previous release. The API Compatibility Report page includes all 1.10 compatibility reports.

Release 1.10.6 versus 1.10.5

New and Changed Functions, Classes, Subroutines, Wrappers, and Macros

In the C Interface (main library)

The following are new C functions in this release:

Function	Description
H5P_GET_FAPL_HDFS	Gets the information of the given Read-Only HDFS virtual file driver
H5P_GET_FAPL_ROS3	Gets the information of the given Read-Only S3 virtual file driver
H5P_SET_FAPL_HDFS	Sets up Read-Only HDFS virtual file driver
H5P_SET_FAPL_ROS3	Sets up Read-Only S3 virtual file driver

In the C++ Wrapper

The following C++ wrapper was added:

Wrapper	Description
LinkCreatPropList::getCreateIntermediateGroup ( ) const	See H5P_GET_CREATE_INTERMEDIATE_GROUP
LinkCreatPropList::setCreateIntermediateGroup (bool crt_intmd_group) const	See H5P_SET_CREATE_INTERMEDIATE_GROUP

Compatibility Notes and Reports

See the API Compatibility Report for information regarding compatibility with previous releases.

Release 1.10.5 versus 1.10.4, 1.10.3, and 1.10.2

New and Changed Functions, Classes, Subroutines, Wrappers, and Macros

In the C Interface (main library)

The following are new C functions in this release:

Function	Description
H5D_GET_CHUNK_INFO	Retrieves information about a chunk specified by the chunk index
H5D_GET_CHUNK_INFO_BY_COORD	Retrieves information about a chunk specified by its coordinates
H5D_GET_NUM_CHUNKS	Retrieves number of chunks that have nonempty intersection with a specified selection
H5F_GET_DSET_NO_ATTRS_HINT	Retrieves the setting for determining whether the specified file does or does not create minimized dataset object headers
H5F_SET_DSET_NO_ATTRS_HINT	Sets the flag to create minimized dataset object headers
H5P_GET_DSET_NO_ATTRS_HINT	Retrieves the setting for determining whether the specified DCPL does or does not create minimized dataset object headers
H5P_SET_DSET_NO_ATTRS_HINT	Sets the flag to create minimized dataset object headers

The following changed in this release:

Function	Description

| H5O_GET_INFO, H5O_GET_INFO_BY_NAME, H5O_GET_INFO_BY_IDX, H5O_VISIT, H5O_VISIT_BY_NAME | In 1.10.3 the original functions were versioned to H5Oget_info*1 and H5Ovisit*1 and the macros H5Oget_info* and H5Ovisit* were created. This broke the API compatibility for a maintenance release. In HDF5-1.10.5, the macros introduced in HDF5-1.10.3 were removed. The H5Oget_info*1 and H5Ovisit*1 APIs were copied to H5Oget_Info* and H5Ovisit*. As an example, H5Oget_info and H5Oget_info1 are identical in this release. |

In the C++ Wrapper

The following C++ wrapper was added:

C++ Wrapper	Description
H5Object::visit()	Wrapper for the C API H5O_VISIT2. Recursively visit elements reachable from an HDF5 object and perform a common set of operations across all of those elements. See H5O_VISIT2 for more information on this function.

In the Fortran Wrapper

The following Fortran wrappers were added or changed:

Fortran Wrapper	Description
h5fget_dset_no_attrs_hint_f h5fset_dset_no_attrs_hint_f h5pget_dset_no_attrs_hint_f h5pset_dset_no_attrs_hint_f	Wrappers for the dataset object header minimization calls. See H5F_GET_DSET_NO_ATTRS_HINT, H5F_SET_DSET_NO_ATTRS_HINT, H5P_GET_DSET_NO_ATTRS_HINT, and H5P_SET_DSET_NO_ATTRS_HINT.
h5ovisit_f h5oget_info_by_name_f h5oget_info h5oget_info_by_idx h5ovisit_by_name_f	Added new Fortran 'fields' optional parameter. See H5O_VISIT2, H5O_GET_INFO_BY_NAME2, H5O_GET_INFO2, H5O_GET_INFO_BY_IDX2, and H5O_VISIT_BY_NAME2.

The following Fortran utility function was added:

Function	Description
h5gmtime	converts (C) 'time_t' structure to Fortran DATE AND TIME storage format

A new Fortran derived type was added:

Derived Type	Description
c_h5o_info_t	This is interoperable with C's h5o_info_t. This is needed for callback functions which pass C's h5o_info_t data type definition. See the Fortran signature for H5O_GET_INFO2.

In the Java wrapper

The following Java wrappers were added or changed:

Java Wrapper	Description
H5Fset_libver_bounds	See the C API H5F_SET_LIBVER_BOUNDS for information on this function

H5Fget_dset_no_attrs_hint H5Fset_dset_no_attrs_hint H5Pget_dset_no_attrs_hint H5Pset_dset_no_attrs_hint	Wrappers for the dataset object header minimization calls See H5F_GET_DSET_NO_ATTRS_HINT, H5F_SET_DSET_NO_ATTRS_HINT, H5P_GET_DSET_NO_ATTRS_HINT, and H5P_SET_DSET_NO_ATTRS_HINT for more information on these APIs.

Compatibility Notes and Reports

See these API Compatibility Reports for 1.10 for information regarding compatibility with previous releases. Reports are available comparing HDF5-1.10.5 vs 1.10.2, HDF5-1.10.5 vs 1.10.3, and HDF5-1.10.5 vs 1.10.4.

Release 1.10.4 versus Release 1.10.3

See the API compatibility Report for information regarding compatibility with previous releases

Release 1.10.3 versus Release 1.10.2

New and Changed Functions, Classes, Subroutines, Wrappers, and Macros

In the C Interface (main library)

The following are new C functions in this release:

Function	Description
H5D_READ_CHUNK	Moved from HDF5 High Level Optimizations library to core library
H5D_WRITE_CHUNK	Moved from HDF5 High Level Optimizations library to core library
H5O_GET_INFO H5O_GET_INFO1 H5O_GET_INFO2	The function H5O_GET_INFO was moved to H5O_GET_INFO1, and the macro H5O_GET_INFO was created that can be mapped to either H5O_GET_INFO1 or H5O_GET_INFO2. *For HDF5-1.10 and earlier releases, H5O_GET_INFO is mapped to H5O_GET_INFO1 by default.*
H5O_GET_INFO_BY_IDX H5O_GET_INFO_BY_IDX1 H5O_GET_INFO_BY_IDX2	The function H5O_GET_INFO_BY_IDX was moved to H5O_GET_INFO_BY_IDX1, and the macro H5O_GET_INFO_BY_IDX was created that can be mapped to either H5O_GET_INFO_BY_IDX1 or H5O_GET_INFO_BY_IDX2. *For HDF5-1.10 and earlier releases, H5O_GET_INFO_BY_IDX is mapped to H5O_GET_INFO_BY_IDX1 by default.*
H5O_GET_INFO_BY_NAME H5O_GET_INFO_BY_NAME1 H5O_GET_INFO_BY_NAME2	The function H5O_GET_INFO_BY_NAME was moved to H5O_GET_INFO_BY_NAME1, and the macro H5O_GET_INFO_BY_NAME was created that can be mapped to either H5O_GET_INFO_BY_NAME1 or H5O_GET_INFO_BY_NAME2. *For HDF5-1.10 and earlier releases, H5O_GET_INFO_BY_NAME is mapped to H5O_GET_INFO_BY_NAME1 by default.*
H5O_VISIT H5O_VISIT1 H5O_VISIT2	The function H5O_VISIT was moved to H5O_VISIT1, and the macro H5O_VISIT was created that can be mapped to either H5O_VISIT1 or H5O_VISIT2. *For HDF5-1.10 and earlier releases, H5O_VISIT is mapped to H5O_VISIT1 by default.*
H5O_VISIT_BY_NAME H5O_VISIT_BY_NAME1 H5O_VISIT_BY_NAME2	The function H5O_VISIT_BY_NAME was moved to H5O_VISIT_BY_NAME1, and the macro H5O_VISIT_BY_NAME was created that can be mapped to either H5O_VISIT_BY_NAME1 or H5O_VISIT_BY_NAME2. *For HDF5-1.10 and earlier releases, H5O_VISIT_BY_NAME is mapped to H5O_VISIT_BY_NAME1 by default.*

In the C High Level Interface

The following C functions were deprecated in this release:

Function	Description
H5DO_READ_CHUNK	Deprecated, moved to H5D_READ_CHUNK
H5DO_WRITE_CHUNK	Deprecated, moved to H5D_WRITE_CHUNK

In the C++ Wrapper

Several C++ wrappers were added or modified to provide additional support. See the API Compatibility Report for details.

Compatibility Notes and Report

See the API Compatibility Report for information regarding compatibility with previous releases.

Release 1.10.2 versus Release 1.10.1

This section lists interface-level changes and other user-visible changes in behavior in the transition from HDF5 Release 1.10.1 to Release 1.10.2.

New and Changed Functions, Classes, Subroutines, Wrappers, and Macros

In the C Interface (main library)

The following are new C functions in this release:

Function	Description
H5D_GET_CHUNK_STORAGE_SIZE	Returns storage amount allocated within a file for a raw data chunk in a dataset
H5F_GET_EOA	Retrieves the file's EOA
H5F_INCREMENT_FILESIZE	Sets the file's EOA to the maximum of (EOA, EOF) + increment
H5F_SET_LIBVER_BOUNDS	Enables the switch of version bounds setting for a file
H5FDdriver_query	Queries a VFL driver for its feature flags when a file is not available (not documented in Reference Manual)
H5P_GET_VIRTUAL_PREFIX	Retrieves prefix applied to VDS source file paths
H5P_SET_VIRTUAL_PREFIX	Sets prefix to be applied to VDS source file paths

The following C functions changed in this release:

Function	Change
H5P_SET_LIBVER_BOUNDS	HDF5-1.10 was added to the range of versions
H5P_SET_VIRTUAL	A change was made to the method of searching for VDS source files
H5PL*	The parameters for many of the H5PL APIs were renamed

In the C High Level Interface

The following new C function was added to this release:

Function	Description
H5DO_READ_CHUNK	Reads a raw data chunk directly from a dataset in a file

In the C++ Wrapper

The following C++ wrappers were added:

C++ Wrapper	Description
`H5Lcreate_soft`	Creates a soft link from link_name to target_name
`H5Lcreate_hard`	Creates a hard link from new_name to curr_name
`H5Lcopy`	Copy an object from a group of file
`H5Lmove`	Rename an object in a group or file
`H5Ldelete`	Removes the specified link from this location
`H5Tencode`	Creates a binary object description of this datatype
`H5Tdecode`	Returns the decoded type from the binary object description
`H5Lget_info`	Returns the information of the named link

These were also added:

- Class `LinkCreatPropList` for link create property list
- Overloaded functions `H5Location::createGroup` to take a link creation property list

See the API Compatibility report for complete details.

In the Java Wrapper

The following Java wrappers were added:

Java Wrapper	Description
`H5Pset_evict_on_close`	Controls the library's behavior of evicting metadata associated with a closed object
`H5Pget_evict_on_close`	Retrieves the file access property list setting that determines whether an HDF5 object will be evicted from the library's metadata cache when closed
`H5Pset_chunk_opts`	Sets the edge chunk option in a dataset creation property list
`H5Pget_chunk_opts`	Retrieves the edge chunk option setting from a dataset creation property list
`H5Pset_efile_prefix`	Sets the external dataset storage file prefix in the dataset access property list
`H5Pget_efile_prefix`	Retrieves the prefix for external raw data storage files as set in the dataset access property list
`H5Pset_virtual_prefix`	Sets prefix to be applied to VDS source file paths
`H5Pget_virtual_prefix`	Retrieves prefix applied to VDS source file paths

See the `Release.txt` file for details.

Tools

New options were added to the `h5clear` utility:

`--filesize`	Print the file's EOA and EOF
`--increment=C`	Set the file's EOA to the maximum of (EOA, EOF) + C for the file C is >= 0; C is optional and will default to 1M when not set

A new option was added to `h5diff`:

`--enable-error-stack`	Enable the error stack

Compatibility Notes and Report

See API Compatibility Reports for 1.10 for information regarding compatibility with previous releases.

Release 1.10.1 versus Release 1.10.0 (and 1.10.0-patch1)

This section lists interface-level changes and other user-visible changes in behavior in the transition from HDF5 Release 1.10.0 (and HDF5-1.10.0-patch1) to Release 1.10.1.

New Features

Several new features are introduced in HDF5 Release 1.10.1.

> Metadata Cache Image
>
> Metadata Cache Evict on Close
>
> Paged Aggregation
>
> Page Buffering

New Features, including associated C Functions

The following features are described and documented in New Features in HDF5 Release 1.10. Each new feature in 1.10.1 is listed below along with the associated C functions:

Metadata Cache Image:

`H5Pget_mdc_image_config`	Retrieves the metadata cache image configuration values for a file access property list.
`H5Pset_mdc_image_config`	Sets the metadata cache image option for a file access property list.
`H5Fget_mdc_image_info`	Gets information about a metadata cache image if it exists.

Metadata Cache Evict on Close:

`H5Pget_evict_on_close`	Retrieves the property list setting that determines whether an HDF5 object will be evicted from the library's metadata cache when it is closed.
`H5Pset_evict_on_close`	Controls the library's behavior of evicting metadata associated with a closed object.

Paged Aggregation:

`H5Pget_file_space_page_size`	Retrieves the file space page size for a file creation property list.
`H5Pset_file_space_page_size`	Sets the file space page size (used with paged aggregation) for a file creation property list.

| H5Pget_file_space_strategy | Retrieves the file space handling strategy for a file creation property list. |
| H5Pset_file_space_strategy | Sets the file space allocation strategy for a file creation property list. |

Page Buffering:

H5Pget_page_buffer_size	Retrieves the maximum size for the page buffer and the minimum percentage for metadata and raw data pages.
H5Pset_page_buffer_size	Sets the maximum size for the page buffer and the minimum percentage for metadata and raw data pages.
H5Fget_page_buffering_stats	Retrieves statistics about page access when it is enabled.
H5Freset_page_buffering_stats	Resets the page buffer statistics.

New and Changed Functions, Classes, Subroutines, Wrappers, and Macros

In the C Interface (main library)

The following new C functions were added:

| H5PLappend |
| H5PLget |
| H5PLinsert |
| H5PLprepend |
| H5PLremove |
| H5PLreplace |
| H5PLsize |

In the C++ Wrapper

New member functions were added to provide `const` versions. For example, these methods,

`ArrayType::getArrayDims ( hsize_t* dims ) const`

`ArrayType::getArrayNDims ( ) const`

replace these:

`ArrayType::getArrayDims ( hsize_t* dims )`

`ArrayType::getArrayNDims ( )`

Several functions were *moved* to other classes. For example, this method,

`H5Location::openDataSet ( char const* name ) const`

replaces:

`CommonFG::openDataSet ( char const* name ) const`

PLEASE review the Compatibility report below for complete information on the C++ changes in this release.

Compatibility Report

Compatibility report for Release 1.10.1 versus Release 1.10.0-patch1

See API Compatibility Reports for 1.10 for information regarding compatibility with previous releases.

Release 1.10.0 of March 2016 versus Release 1.8.16

This section lists interface-level changes and other user-visible changes in behavior in the transition from HDF5 Release 1.8.16 to Release 1.10.0.

Changed Type

`hid_t`

Changed from a 32-bit to a 64-bit value.

`hid_t` is the type is used for all HDF5 identifiers. This change, which is necessary to accomodate the capacities of modern computing systems, therefore affects all HDF5 applications. If an application has been using HDF5's `hid_t` the type, recompilation will normally be sufficient to take advantage of HDF5 Release 1.10.0. If an application uses an integer type instead of HDF5's `hid_t` type, those identifiers must be changed to a 64-bit type when the application is ported to the 1.10.x series.

New Features and Feature Sets

Several new features are introduced in HDF5 Release 1.10.0.

Single-Writer / Multiple-Reader or SWMR

Collective Metadata I/O

Fine-tuning the metadata cache

File Space Management

Virtual Datasets or VDS

Partial Edge Chunk Options

Relative Pathnames for External Links

Property List Encoding and Decoding

More substantial lists follow, including new and modified C functions and Fortran subroutines.

New Features, including associated C Functions and Fortran Wrappers

The following features are described and documented in New Features in HDF5 Release 1.10.0. On this page, we list each feature and its associated C functions and Fortran wrappers.

Single-writer / Multiple-reader, commonly called SWMR:

`H5Fstart_swmr_write`	Enables SWMR writing mode for a file.
`H5DOappend`	Appends data to a dataset along a specified dimension. *(This is a high-level API.)*
`H5Pget_append_flush`	Retrieves the values of the append property that is set up in the dataset access property list.
`H5Pset_append_flush`	Sets two actions to perform when the size of a dataset's dimension being appended reaches a specified boundary.
`H5Pget_object_flush_cb`	Retrieves the object flush property values from the file access property list.
`H5Pset_object_flush_cb`	Sets a callback function to invoke when an object flush occurs in the file.
`H5Odisable_mdc_flushes`	Prevents metadata entries for an HDF5 object from being flushed from the metadata cache to storage.
`H5Oenable_mdc_flushes`	Returns the cache entries associated with an HDF5 object to the default metadata flush and eviction algorithm.

H5Oare_mdc_flushes_disabled	
	Determines if an HDF5 object (dataset, group, committed datatype) has had flushes of metadata entries disabled.
H5Fdisable_mdc_flushes	Globally prevents dirty metadata entries from being flushed from the metadata cache to storage.
H5Fenable_mdc_flushes	Returns a file's metadata cache to the standard eviction and flushing algorithm.
H5Fare_mdc_flushes_disabled	
	Determines if flushes have been globally disabled for a file's metadata cache.
H5Fget_mdc_flush_disabled_obj_ids	
	Returns a list of all object identifiers for which flushes have been disabled in a file's metadata cache.

Command-line Tools:

h5watch	Allows users to output new records appended to a dataset under SWMR access as it grows. The functionality is similar to the Unix user command `tail` with the follow option, which outputs appended data as the file grows.
h5format_convert	This tool allows users to convert the indexing type of a chunked dataset made with a 1.10.x version of the HDF5 Library when the latest file format is used to the 1.8.x version 1 B-tree indexing type. For example, datasets created using SWMR access, can be converted to be accessed by the HDF5 1.18 library and tools. The tool does not rewrite raw data, but it does rewrite HDF5 metadata.

Collective Metadata I/O:

H5Pset_coll_metadata_write h5pset_coll_metadata_write_f	Establishes I/O mode property setting, collective or independent, for metadata writes.
H5Pget_coll_metadata_write h5pget_coll_metadata_write_f	Retrieves I/O mode property setting for metadata writes.
H5Pset_all_coll_metadata_ops h5pset_all_coll_metadata_ops_f	Establishes I/O mode, collective or independent, for metadata read operations.
H5Pget_all_coll_metadata_ops h5pget_all_coll_metadata_ops_f	Retrieves I/O mode for metadata read operations.

Fine-tuning the Metadata Cache:

H5Fget_metadata_read_retries_info	
	Retrieves the collection of read retries for metadata items with checksum.
H5Pget_metadata_read_attempts	
	Retrieves the number of read attempts from a file access property list.
H5Pset_metadata_read_attempts	
	Sets the number of read attempts in a file access property list.
H5Dflush	Causes all buffers associated with a dataset to be immediately written to disk without removing the data from the cache.
H5Drefresh	Causes all buffers associated with a dataset to be cleared and immediately re-loaded with updated contents from disk storage.
H5Gflush	Causes all buffers associated with a group to be immediately flushed to disk without removing the data from the cache.
H5Grefresh	Causes all buffers associated with a group to be cleared and immediately re-loaded with updated contents from disk storage.

`H5Oflush`	Causes all buffers associated with an object to be immediately flushed to disk without removing the data from the cache.
`H5Orefresh`	Causes all buffers associated with an object to be cleared and immediately re-loaded with updated contents from disk storage.
`H5Tflush`	Causes all buffers associated with a committed datatype to be immediately flushed to disk without removing the data from the cache.
`H5Trefresh`	Causes all buffers associated with a committed datatype to be cleared and immediately re-loaded with updated contents from disk storage.
`H5Fget_intent`	Determines the read/write or read-only status of a file.

Logging APIs:

`H5Pset_mdc_log_options`	Sets metadata cache logging options.
`H5Pget_mdc_log_options`	Gets metadata cache logging options.
`H5Fstart_mdc_logging`	Starts logging metadata cache events if logging was previously enabled.
`H5Fstop_mdc_logging`	Stops logging metadata cache events if logging was previously enabled and is currently ongoing.
`H5Pget_mdc_logging_status`	
	Gets the current metadata cache logging status.

File Space Management:

`H5Fget_free_sections`		Retrieves free-space section information for a file.
`H5Fget_freespace`		Returns the amount of free space in a file.
`H5Fget_info2`		Returns global information for a file.
`H5Pset_file_space`		Sets the file space management strategy and/or the free-space section threshold for an HDF5 file.
`H5Pget_file_space`		Retrieves the file space management strategy and/or the free-space section threshold for an HDF5 file.

The following tool has been modified to preserve or modify file freepace information appropriately when processing files employing the VDS feature:

`h5repack`	Repacks HDF5 files with various options, including the ability to change the applied filters. This version of `h5repack` understands the file free space feature and handles the file and metadata appropriately.

Virtual Dataset or VDS:

`H5Pset_virtual` `h5pset_virtual_f`	Sets the mapping between virtual and source datasets.
`H5Pget_virtual_count` `h5pget_virtual_count_f`	Retrieves the number of mappings for the virtual dataset.
`H5Pget_virtual_vspace` `h5pget_virtual_vspace_f`	Retrieves a dataspace identifier for the selection within the virtual dataset used in the mapping.
`H5Pget_virtual_srcspace` `h5pget_virtual_srcspace_f`	Retrieves a dataspace identifier for the selection within the source dataset used in the mapping.
`H5Pget_virtual_dsetname` `h5pget_virtual_dsetname_f`	Retrieves the name of a source dataset used in the mapping.
`H5Pget_virtual_filename` `h5pget_virtual_filename_f`	Retrieves the filename of a source dataset used in the mapping.

`H5Pset_virtual_printf_gap` `h5pset_virtual_printf_gap_f`	Sets maximum number of missing source files and/or datasets with printf-style names when getting the extent of an unlimited virtual dataset.
`H5Pget_virtual_printf_gap` `h5pget_virtual_printf_gap_f`	Returns maximum number of missing source files and/or datasets with printf-style names when getting the extent for an unlimited virtual dataset.
`H5Pset_virtual_view` `h5pset_virtual_view_f`	Sets the view of the virtual dataset to include or exclude missing mapped elements.
`H5Pget_virtual_view` `h5pget_virtual_view_f`	Retrieves the view of a virtual dataset.

Supporting Functions:

`H5Sis_regular_hyperslab` `h5sis_regular_hyperslab_f`	Determines whether a hyperslab selection is regular.

`H5Sget_regular_hyperslab`
 	
`h5sget_regular_hyperslab_f`
	Retrieves a regular hyperslab selection.

Modified Functions:
The following pre-exising functions have been modified to understand virtual datasets.

`H5Pset_layout` `h5pset_layout_f`	Specifies the layout to be used for a dataset. Virtual dataset, `H5D_VIRTUAL`, has been added to the list of layouts available through this function.
`H5Pget_layout` `h5pget_layout_f`	Retrieves the layout in use for a dataset. Virtual dataset, `H5D_VIRTUAL`, has been added to the list of layouts.

Partial Edge Chunks:

`H5Pset_chunk_opts`	Sets a partial edge chunk option in a dataset access property list.
`H5Pget_chunk_opts`	Retrieves partial edge chunk option setting from a dataset access property list.

Relative Pathnames for External Links:

`H5Pset_elink_prefix`	These functions enable the user to specify the locations of external files.
`H5Pget_elink_prefix`	*(These functions are not yet documented.)*

Property List Encoding and Decoding:

`H5Pencode`
`H5Pdecode`

Additional New Functions:

The following new functions appear in HDF5 Release 1.10.0 but are not yet documented:

```
H5Dformat_convert

H5Dget_chunk_index_type

H5FDlock

H5FDunlock

H5Fformat_convert

H5LDget_dset_dims
```

```
        H5LDget_dset_elmts

        H5LDget_dset_type_size
```

New and Changed Elements of the Packet Table (H5PT) High-level API

In the C Interface

Replacement functions:

`H5PTcreate`

Takes a property list identifier to provide flexibility on creation properties.

`H5PTcreate_fl` has been removed.

`H5PTfree_vlen_buff`

Replaces `H5PTfree_vlen_readbuff`.

New functions:

Two accessor functions have been added.

`H5PTget_dataset`

Returns the identifier of the dataset associated a packet table.

`H5PTget_type`

Returns the identifier of the datatype used by a packet table.

`H5PTis_varlen`

Determines whether a type is variable-length.

In the C++ Interface

Overloaded constructor

An overloaded constructor has been added.

`FL_PacketTable`

Takes a property list identifier to provide flexibility on creation properties.>/dd>

`H5PTfree_vlen_buff`

Replaces `H5PTfree_vlen_readbuff`.

Accessor wrappers

Two accessor wrappers are added to class `PacketTable`.

`PacketTable::GetDataset()`

Returns the identifier of the dataset associated a packet table.

`PacketTable::GetDatatype()`

Returns the identifier of the datatype used by a packet table.

Other wrappers

`PacketTable::FreeBuff()`

Replaces `VL_PacketTable::FreeReadBuff()`.

`PacketTable::IsVariableLength()`

Replaces `VL_PacketTable::IsVariableLength()`.

Overloaded functions:

Where a member functions has a `char*` as an argument, an overloaded functions has been added to provide the `const char*` argument.

The existing version will be deprecated in a future release.

Java Interface Changes

Integration into Main HDF5 Library

The Java HDF5 JNI library has been integrated into the HDF5 repository.

Configure option:

```
--enable-java
```

CMake option:

```
HDF5_BUILD_JAVA:BOOL=ON
```

Prior to the 1.10.x series, the HDF5 Java tools were built from an independent repository and were not as fully integrated with HDF5. were built from an independent repository and were not as fully integrated with HDF5.

Package Hierarchy Change

The package hierarchy has changed to `hdf.hdflib.hdf5`.

Prior to the 1.10.x series, the hierarchy was `ncsa.hdf.hdflib.hdf5`.

New Java APIs

A number of new APIs have been added in the Java interface, including APIs for the VDS and SWMR features.

Functions with Changed Behavior

H5Lexists

The behavior of this function has changed in this release. When testing the pathname `/` (a slash representing the root of an HDF5 file) `H5Lexists` now returns successfully with the value `1` (one). See the entry in the —*HDF5 Reference Manual* for `H5Lexists` for more information.

API Compatibility

See API Compatibility Macros in HDF5 for details on the following.

New API Compatibility Flag

A new `v18` flag was added enabling the building of HDF5 such that the default API is compatible with the HDF5 Release 1.8.x API:
```
--with-default-api-version=v18
```

New versioned functions and associated compatibility macros

Two functions and a struct have been converted to a versioned form in this release. Compatibility macros have been created for each.

`H5Fget_info`

The original function is renamed to `H5Fget_info1` and deprecated.

A new version of the function, `H5Fget_info2`, is introduced.

The compatiblity macro `H5Fget_info` is introduced.

`H5F_info_t`

This is the struct used by the `H5Fget_info` functions and macro.

The original struct is renamed to `H5F_info1_t` and deprecated.

A new version of the struct, `H5F_info2_t`, is introduced.

The compatiblity macro `H5F_info_t` is introduced.

`H5Rdereference`

The original function is renamed to `H5Rdereference1` and deprecated.

A new version of the function, `H5Rdereference2`, is introduced.

The compatiblity macro `H5Rdereference` is introduced.

Autotools Configuration and Large File Support

Autotools configuration has been extensively reworked and autotool's handling of large file support has been overhauled in this release.

See the following sections in `RELEASE.txt`:

- "Autotools Configuration Has Been Extensively Reworked"
- "LFS Changes"

`RELEASE.txt` is found in the `release_docs/` subdirectory at the root level of the HDF5 code distribution.

Compatibility Report and Comments

Compatibility report for Release 1.10.0 versus Release 1.8.16

See API Compatibility Reports for 1.10 for information regarding compatibility with previous releases.

Comments regarding the report

In the C interface, the `hid_t` change from 32-bit to 64-bit was made in order to address a performance problem that arose when the library "ran out" of valid object identifiers to issue and thus needed to employ an expensive algorithm to find previously issued identifiers that could be re-issued. This problem is avoided by switching the size of the `hid_t` type to 64-bit integers instead of 32-bit integers in order to make the pool of available integers significantly larger. (`H5E_major_t` and `H5E_minor_t` are aliased to `hid_t` which is why they changed size as well). (An alternate solution to this problem was applied in release HDF5 1.8.5 but this is the cleaner/preferred solution and had to wait until 1.10.0 to be included).

`hbool_t` will now be defined as a `_Bool` type when configure determines that it's available.

Public structs that have members of type `hid_t` or `hbool_t` are affected by the above changes accordingly.

The `H5Fget_info` function was renamed due to the introduction of a newer version of the function which returns additional information. The `H5R dereference` function was renamed due to the introduction of a newer version of the function which allows a data access property list to be passed in. Both changes are accompanied with compatibility macros, so while existing code will need to be recompiled in order to use the newer library version, these functions do not need to be changed in application code using them provided that the HDF5 API compatibility macros are configured appropriately.