Package ‘hdf5r’

March 25, 2020

Type  Package
Title  Interface to the 'HDF5' Binary Data Format
Version  1.3.2
Description  'HDF5' is a data model, library and file format for storing and managing large amounts of data. This package provides a nearly feature complete, object oriented wrapper for the 'HDF5' API <https://support.hdfgroup.org/HDF5/doc/RM/RM_H5Front.html> using R6 classes. Additionally, functionality is added so that 'HDF5' objects behave very similar to their corresponding R counterparts.

URL  https://hhoeflin.github.io/hdf5r,
https://github.com/hhoeflin/hdf5r

BugReports  https://github.com/hhoeflin/hdf5r/issues

License  Apache License 2.0 | file LICENSE

Copyright  For the hdf5r package: Novartis Institute for BioMedical Research Inc. For HDF5: see the HDF5_COPYRIGHTS file.

LazyLoad  true

Depends  R (>= 3.2.2), methods

Imports  R6, bit64, utils

Suggests  testthat, knitr, markdown, nycflights13, reshape2, formatR,

SystemRequirements  HDF5 (>= 1.8.13)

VignetteBuilder  knitr

NeedsCompilation  yes

RoxygenNote  6.1.1.9000

Collate  'Common_functions.R' 'Compound.R' 'H5constants.R'
         'Helper_functions.R' 'Misc.R' 'R6Classes.R' 'R6Classes_H5A.R'
         'R6Classes_H5D.R' 'R6Classes_H5File.R' 'R6Classes_H5Group.R'
         'R6Classes_H5P.R' 'R6Classes_H5R.R' 'R6Classes_H5S.R'
         'R6Classes_H5T.R' 'adapt_during_onLoad.R' 'convert.R'
         'factor_ext.R' 'globalVariables.R' 'h5errorHandling.R'
         'h5wrapper.R' 'hdf5r.R' 'high_level_UI.R' 'open_objs.R' 'zzz.R'
Author  Holger Hoefling [aut, cre],
       Mario Annau [aut],
       Novartis Institute for BioMedical Research (NIBR) [cph]
Maintainer  Holger Hoefling <hhoeflin@gmail.com>
Repository  CRAN
Date/Publication  2020-03-25 15:00:06 UTC

R topics documented:

hdf5r-package ............................. 3
as_hex ................................... 4
create_empty .............................. 5
factor_ext .................................. 5
factor_ext_functions ..................... 6
flatten_df .................................. 8
guess_chunks .............................. 9
guess_nelem ................................ 10
guess_space ................................ 11
h5-wraper .................................. 11
H5A-class .................................. 13
h5attributes ................................ 15
h5const .................................... 16
H5D-class .................................. 17
H5File-class ................................ 23
h5garbage_collect .......................... 30
H5Group-class ................................ 30
H5Group_access .............................. 35
H5P-class ................................... 37
H5P_ATTRIBUTE_CREATE-class ............. 38
H5P_CLASS-class ............................. 39
H5P_DATASET_ACCESS-class ................ 39
H5P_DATASET_CREATE-class ................ 40
H5P_DATASET_XFER-class ................... 42
H5P_FILE_ACCESS-class ..................... 43
H5P_FILE_CREATE-class ..................... 44
H5P_LINK_ACCESS-class ..................... 45
H5P_LINK_CREATE-class ..................... 46
H5P_OBJECT_COPY-class ..................... 47
H5P_OBJECT_CREATE-class ................... 47
H5R-class ................................... 48
H5RefClass-class ............................ 50
H5R_DATASET_REGION-class ................ 51
H5R_functions ................................ 52
H5R_OBJECT-class ............................ 54
H5S-class ................................... 55
H5S_H5D_subset_assign ..................... 58
H5T-class ................................... 60
hdf5r-package

hdf5r: A package to provide an interface to hdf5 from R

Description

A package that allows to interface with the HDF5 C-library. Provides access to most of its functionality from inside R using R6 classes. For more details please see the README at the github page https://github.com/hhoeflin/hdf5r.

Examples

```r
# create a new file
test_file <- tempfile(fileext=".h5")
file.h5 <- H5File$new(test_file, mode="w")
data(cars)
file.h5$create_group("test")
file.h5[["test/cars"]]<-cars
cars_ds <- file.h5[["test/cars"]]
h5attr(cars_ds,"rownames")<-rownames(cars)

# Close the file at the end
# the 'close' method closes only the file-id, but leaves object inside the file open
# This may prevent re-opening of the file. 'close_all' closes the file and all objects in it
file.h5$close_all()
# now re-open it
file.h5 <- H5File$new(test_file, mode="r+")

# lets look at the content
file.h5$ls(recursive=TRUE)
```
cars_ds <- file.h5["test/cars"]
# note that for now tables in HDF5 are 1-dimensional, not 2-dimensional
mycars <- cars_ds[]
h5attr_names(cars_ds)
h5attr(cars_ds, "rownames")

file.h5$close_all()

---

**as_hex**

*Convert a double or integer to hex*

**Description**

Convert a double or integer to hex

**Usage**

```r
as_hex(x)
```

**Arguments**

- `x` The integer or double vector to convert

**Details**

Converts a double or integer to hex. Contrary to the built-in `format`, this is done without any conversion of integers in double-format to integers in integer format.

**Value**

Character string with the hex value

**Author(s)**

Holger Hoefling
create_empty

Description
Create an empty R-object according to a given HDF5 datatype

Usage
create_empty(nelem, dtype)

Arguments
nelem The number of elements to use for the object
dtype The datatype based on which an empty R-object should be created

Details
With complex datatypes it can be useful to have a template that can be used so that the data input into a dataset conforms to expectations.

Given a datatype, this function creates an object of length nelem. Here, an empty datatype refers to objects with value 0 for numeric objects and empty strings.

Value
An empty R object corresponding to the datatype that was passed in

Author(s)
Holger Hoefling

factor_ext

Description
Create an extended factor

Usage
factor_ext(x, values, levels, drop = FALSE)
factor_ext_functions

Arguments

x The object to convert to an factor_ext
values The values used for the levels; This is were factor_ext is different from a factor, as values for levels do not have to be consecutive or start at 1.
levels The levels of the object; character string
drop Should non-occurring levels be dropped

Details

An extended version of a regular factor variable. Instead of the levels having values from 1 to n where n is the number of levels, any integer value can be used for any level (including 64bit integers). If all values are in the range of a regular 32-bit integer, it is coerced to int. Automatic coercion of extended factors to factors in H5ToR_Post for enums only works for 32-bit integer base types. In this page this is heavily used, as constants in HDF5 can be arbitrary integer values.

Value

An object of S3 class factor_ext

Author(s)

Holger Hoefling

Description

Various functions for factor_ext objects

Usage

values(x, ...)

## S3 method for class 'factor_ext'
values(x, ...)

## S3 method for class 'factor'
values(x, ...)

## Default S3 method:
values(x, ...)

## S3 method for class 'factor_ext'
as.character(x, ...)
## S3 method for class 'factor_ext'
x[[...]]

## S3 replacement method for class 'factor_ext'
x[[...]] <- value

## S3 method for class 'factor_ext'
x[... , drop = FALSE]

## S3 replacement method for class 'factor_ext'
x[...] <- value

is.factor_ext(x)

coercible_to_factor(x)

coerce_to_factor(x)

## S3 method for class 'factor_ext'
print(x, quote = FALSE, max.levels = NULL,
       width = getOption("width"), ...)

## S3 method for class 'factor_ext'
e1 == e2

## S3 method for class 'factor_ext'
e1 != e2

## S3 method for class 'factor_ext'
c(...)

### Arguments

- **x**
  - Object of type factor_ext
- **...**
  - Currently ignored
- **value**
  - The object to assign; here has be a level of factor_ext
- **drop**
  - Should dimensions of size 1 be dropped?
- **quote**
  - logical, indicating whether or not strings should be printed with surrounding quotes.
- **max.levels**
  - integer, indicating how many levels should be printed. if '0', no extra "Levels" line will be printed. The default, 'NULL', entails choosing 'max.levels' such that the levels print on one line of width 'width' (same for values).
- **width**
  - only used when max.levels is NULL (see above)
- **e1, e2**
  - The two objects in the equality or inequality comparison.

### Details

- **values** Extracts the underlying values of an object (the generic here)
values.factor_ext Extracts the underlying values of a factor_ext object
values.factor Extracts the underlying values of a factor
values.default Default of the values function; currently returns an error
as.character Coerces factor_ext to a character-representation using its levels, not values
[[.factor_ext Single-item subsetting of a factor_ext object
][<-.factor_ext Single-item subset assignment to a factor_ext object
.[.factor_ext Subsetting of a factor_ext object
[<-.factor_ext Subset assignment to a factor_ext object
is.factor_ext Check if it is a factor_ext object. Returns a logical
coercible_to_factor Checks if a factor_ext could be coerced to a factor. Return a logical.
coerce_to_factor Coerces to a factor, otherwise throws an error if not possible.
print.factor_ext Prints a factor_ext object.
==.factor_ext Compare two factor_ext for equality.
!=.factor_ext Compare two factor_ext for inequality.
c.factor_ext Concatenate objects of type factor_ext.

Value
Depending on the function

Author(s)
Holger Hoefling

flatten_df

Flatten a nested data.frame

Description
Flatten a nested data.frame

Usage
flatten_df(df, factor_ext_to_char = FALSE)

Arguments
df The data.frame to flatten
factor_ext_to_char Should extended factor variables be converted to characters (mainly for easy printing)
**Details**

HDF5 Compounds allow for nesting. Correspondingly, nested data.frames are being produced. This function flattens such a nested data.frame. For easier printing to the screen, it also allows for coercion of `factor_ext` to character variables.

**Value**

A flattened `data.frame`

**Author(s)**

Holger Hoefling

---

**guess_chunks**

*Guess the dimension of a chunk*

**Description**

Guess the dimension of a chunk

**Usage**

```r
guess_chunks(space_maxdims, dtype_size,
             chunk_size = getOption("hdf5r.chunk_size"))
```

**Arguments**

- `space_maxdims`: Maximal dimensions of the dataset
- `dtype_size`: Size of the datatype that is stored
- `chunk_size`: Size of each chunk in bytes

**Details**

The size of the chunk in bytes is first divided by the size of the datatype, giving the number of elements to be stored in each chunk. This is taken as a rough guideline. Then, the number of dimensions of the dataset is used. By default, the chunk is assumed to have the same size in each dimension, yielding an initial guess.

If the resulting chunk is larger than the entire dataset for a maximal dimension, this dimension of the chunk is reduced and redistributed to the other dimensions.

As a chunk is never allowed to be larger than the maximum dimension of the dataset itself,

**Value**

An integer vector giving the dimension of the chunk

**Author(s)**

Holger Hoefling
**Description**

Guess the HDF5 datatype of an R object

**Usage**

```
guess_nelem(x, dtype)
guess_dim(x)
guess_dtype(x, ds_dim = NULL, scalar = FALSE,
            string_len = getOption("hdf5r.default_string_len"))
```

**Arguments**

- `x` The object for which to guess the HDF5 datatype or the dimension or the number of elements
- `dtype` datatype; used in guessing the number of dataset elements of an R object
- `ds_dim` Can explicitly set the dimension of the dataset object. For scalar, this is one. Otherwise, this can be used so that a multi-dimensional object can be represented so that some of its dimensions are in the dataset, and some are inside an `H5T_ARRAY`
- `scalar` Should the datatype be created so that `x` can be represented as a scalar with that datatype? This is intended to know if a vector/array should be represented as an `H5T_ARRAY` or not.
- `string_len` If a string is in the R object, the length to which the corresponding HDF5 type should be set. If it is a positive integer, the string is of that length. If it is `Inf`, it is variable length. If it is set to estimate, it is set to the length of the longest string in the `x`.

**Details**

Given an object, it creates a datatype in HDF5 that would match this object. For simple datasets like arrays, this function is not so useful, but is very good for creating dataframes or hierarchical objects (like lists of dataframes) etc.

**Value**

An object of class `H5T` that represents the HDF5-type of the Robj that was passed in

**Author(s)**

Holger Hoefling
**guess_space**  
*Guess the dataspace of an object*

**Description**

Guess the dataspace of an object

**Usage**

```r
guess_space(x, dtype, chunked = TRUE)
```

**Arguments**

- `x`: The R object for which to guess the space
- `dtype`: Object of type `H5T`, that represents that datatype to use.
- `chunked`: Is the datatype chunked? If yes, `maxdims` of the space will be set to infinity, otherwise `maxdims` will be set to the original extent of the space.

**Details**

Creates a dataspace that fits an R object so that it can be written into a dataset. This is used for example in dataset creation based on an R-object, not a specifically defined dimensions.

**Value**

An object of type `H5S`

**Author(s)**

Holger Hoefling

---

**h5-wrapper**

*Wrapper functions to provide an h5 compatible interface.*

**Description**

The functions listed below provide a wrapper-interface compatible to functions specified in the `h5` package. The author(s) have decided to deprecate `h5` and join forces and still make the transition for `h5` users as smooth as possible. Additionally, almost all testcases could be transferred to `hdf5r` to improve test coverage even more.
Usage

h5file(...)
createGroup(object, name, ...)
openLocation(object, name)
openGroup(object, name)
createDataSet(object, name, ...)
readDataSet(object)
h5close(object)
h5flush(object)
existsGroup(object, name)
is.h5file(name)
extendDataSet(object, dims)

## S3 method for class 'H5D'
rbind(x, mat, ..., deparse.level = 1)

## S3 method for class 'H5D'
cbind(x, mat, ..., deparse.level = 1)

## S3 method for class 'H5D'
c(x, ...)

h5unlink(object, name)
list.attributes(object)

Arguments

... Additional parameters passed to create_group or h5file.
object CommonFG; Object implementing the CommonFG Interface (e.g. H5File, H5Group).
name Name of the group to create.
dims numeric; Dimension vector to which dataset should be extended.
x An object of class H5D; the dataset to add rows or columns to; Needs to be a matrix
mat The matrix to add to x
deparse.level Set to 1; ignored otherwise; only present as required by generic
H5A-class

Details
Below you can find a list of all h5 functions including hdf5r mappings.

h5file  Directly maps to H5File$new, see also H5File.
createGroup  Maps to object$create_group where object implements CommonFG.
opencLocation  Uses object$open where object implements CommonFG.
createDataSet  Maps to object$create_dataset where object implements CommonFG.
readDataSet  Maps to object$read, see also H5D.
h5close  Maps to object$close_all for H5File and object$close for other.
h5flush  Maps to object$flush where object implements CommonFGDTA.

The following interfaces are defined:

CommonFG  Implemented by objects of class H5File and H5Group.
CommonFGDTA  Implemented by objects of class H5File, H5Group, H5D, H5T and H5A.

References

H5A-class  Class for representing HDF5 attributes

Description
This class represents an HDF5 attribute. Usually it is easier to read and write attributes for groups, datasets and committed datatypes using the functions documented in h5attributes.

Details
Otherwise, the functionality for attributes is very similar to that of datasets (H5D), however with the notable exception that attributes always have to be read and written as a whole.

Value
Object of class H5A.
Methods

get_info()  This function implements the HDF5-API function H5Aget_info. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_GET_INFO for details.

attr_name()  This function implements the HDF5-API function H5Aget_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_GET_NAME for details.

get_space()  This function implements the HDF5-API function H5Aget_space. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_GET_SPACE for details.

get_type(native = TRUE)  This function implements the HDF5-API function H5Aget_type. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_GET_TYPE for details.

get_storage_size()  This function implements the HDF5-API function H5Aget_storage_size. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_GET_STORAGE_SIZE for details.

read_low_level(buffer, mem_type, duplicate_buffer = FALSE)  Only for advanced users. See documentation for read instead. This function implements the HDF5-API function H5Aread. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_READ for details.

read(flags = getOption("hdf5r.h5stor_default"), drop = TRUE)  Reads the data of the attribute and returns it as an R-object

Parameters

flags  Conversion rules for integer values. See also h5const

drop  Logical. Should dimensions of length 1 be dropped (R-default for arrays)

write_low_level(buffer, mem_type)  Only for advanced users. See documentation for write instead. This function implements the HDF5-API function H5Awrite. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_WRITE for details.

write(robj, mem_type = NULL, flush = getOption("hdf5r.flush_on_write"))  Writes the data of robj to the attribute

Parameters

robj  The object to write into the attribute

mem_type  The memory data type to use when transferring from HDF5 to intermediate storage. This is an advanced development feature and may be removed in the future.

print(...)  Prints information for the dataset

Parameters

...  ignored

flush(scope = h5const$H5F_SCOPE_GLOBAL)  This function implements the HDF5-API function H5Fflush. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5F.Flush for details.

get_filename()  This function implements the HDF5-API function H5Fget_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5F_GET_NAME for details.
**Author(s)**

Holger Hoefling

**Examples**

```r
fname <- tempfile(fileext = ".h5")
file <- H5File$new(fname, mode = "a")
h5attr(file, "attr_numeric") <- rnorm(10)
a <- file$attr_open("attr_numeric")
a$get_info()
a$attr_name()
a$get_space()
a$get_type()
a$get_storage_size()
a$read()
a$write(10:1)
a$print()
a$close()
file$close_all()
```

---

**h5attributes**  
*Interface for HDF5 attributes*

**Description**

Interface for HDF5 attributes

**Usage**

```r
h5attributes(x)

h5attr_names(x)

h5attr(x, which)

h5attr(x, which) <- value
```

**Arguments**

- **x**  
The object to which to attach the attribute to or retrieve it from. Can be one of `H5Group`, `H5D`, `H5T` or `H5File`
- **which**  
The name of the attribute to assign it to
- **value**  
The value to assign to the attribute.

**Details**

Implements high-level functions that allows interactions with HDF5-attributes in a way very similar to regular R-object attributes in R are handled.
Value

For h5attributes, a named list with the content of the attributes read out. For h5attr_names, a vector of names of the attributes attached to the object x. For h5attr, the content of the attribute and for h5attr<-, the assignment, the original object to which the attributes are attached (so that chaining is possible).

Author(s)

Holger Hoefling

---

h5const  All constants used in HDF5

Description

These are all constants used in HDF5. They are stored in an environment with locked bindings so that they cannot be changed. An overview of all constants can be seen with h5const$overview, listing all of them. Each constant can be accessed using $ and the name of the constant. See the examples below.

Details

There are also some flags that govern edge cases of conversion from HDF5 to R. This is related to how integers are being treated and the issue of R not being able to natively represent 64bit integers and not at all being able to represent unsigned 64bit integers (even using add-on packages). The constants all start with the term H5TOR. There are currently possible values:

- H5TOR_CONV_NONE Doesn’t do any conversion. Every integer datatype with more than 32 bit is returned as 64bit integers. For unsigned 64bit integers, the conversion to signed 64bit integers is done by truncation.
- H5TOR_CONV_INT64_INT_NOLOSS Under this setting, whenever a 64 bit integer would be returned, it is checked if it would also fit into a 32 bit integer without data loss and returned as such if possible.
- H5TOR_CONV_INT64_FLOAT_NOLOSS Under this setting, whenever a 64 bit integer would be returned, it is checked if it would also fit into a 64 bit floating point value without data loss and returned as such if possible.
- H5TOR_CONV_INT64_FNOLOSS Combines H5TOR_CONV_INT64_INT_NOLOSS and H5TOR_CONV_INT64_FLOAT_NOLOSS and is set as the default in the hdf5r.h5tor_default option.
- H5TOR_CONV_INT64_FLOAT_FORCE Under this setting, whenever a 64 bit integer would be returned, it is coerced to a double even if this results in a loss of precision. If a loss of precision occurs, a warning is issued. Please note that this also overrides the use of H5TOR_CONV_UINT64_NA. As loss of precision is already accepted, UINT64-values that are larger than LLONG_MAX will be represented as their next possible floating point value.
- H5TOR_CONV_UINT64_NA When converting an unsigned 64bit integer, any values that don’t fit into a signed 64bit integer are set to NA. If this flag is not set, then the values will be truncated to LLONG_MAX, the largest 64bit signed integer.
- H5TOR_CONV_DEFAULT Is both H5TOR_CONV_INT64_INT and H5TOR_CONV_UINT64_FLOAT
**H5D-class**

**Author(s)**
Holger Hoefling

**Examples**

```
h5const$overview
h5const$H5F_ACC_RDWR
h5const$H5F_ACC_DEFAULT
# Combining flags
bitwOr(h5const$H5TOR_CONV_UINT64_NA, h5const$H5TOR_CONV_INT64_INT_NOLOSS)
```

**H5D-class**

*Class for representing HDF5 datasets*

**Description**

In HDF5, datasets can be located in a group (see **H5Group**) or at the root of a file (see **H5File**). They can be created either with a pre-existing R-object (arrays as well as data.frames are supported, but not lists or other complex objects), or by specifying an explicit datatype (for available datatypes see **h5types$overview**) as well as the dimension. In addition, other features are supported such as transparent compression (for which a chunk-size can be selected).

**Details**

In order to create a dataset, the `create_dataset` methods of either **H5Group** or **H5File** should be used. Please see the documentation there for how to create them.

The most important parts of a dataset are the

- **Space** The space of the dataset. It describes the dimension of the dataset as well as the maximum dimensions. Can be obtained using the `get_space` of the **H5S** object.
- **Datatype** The datatypes that is being used in the dataset. Can be obtained using the `get_type` method. See **H5T** to get more information about using datatypes.

In order to read and write datasets, the `read` and `write` methods are available. In addition, the standard way of using `[` to access arrays is supported as well (see **H5S_H5D_subset_assign** for more help).

Other information/action of possible interest are

- **Storage size** The size of the dataset can be extracted using `get_storage_size`
- **Size change** The size of the dataset can be changed using the `set_extent` method

Please also note the active methods

- **dims** Dimension of the dataset
- **maxdims** Maximum dimensions of the dataset
- **chunk_dims** Dimension of the chunks
- **key_info** Returns the space, type, property-list and dimensions
**Value**

Object of class `H5D`.

**Methods**

`new(id = NULL)` Initializes a new dataset-object. Only for internal use. Use the `create_dataset` function for `H5Group` and `H5File` objects

**Parameters**

- `id` For internal use only

`get_space()` This function implements the HDF5-API function `H5Dget_space`. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5D_GET_SPACE](https://portal.hdfgroup.org/display/HDF5/H5D_GET_SPACE) for details.

`get_space_status()` This function implements the HDF5-API function `H5Dget_space_status`. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5D_GET_SPACE_STATUS](https://portal.hdfgroup.org/display/HDF5/H5D_GET_SPACE_STATUS) for details.

`get_type(native = TRUE)` This function implements the HDF5-API function `H5Dget_type`. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5D_GET_TYPE](https://portal.hdfgroup.org/display/HDF5/H5D_GET_TYPE) for details.

`get_create_plist()` This function implements the HDF5-API function `H5Dget_create_plist`. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5D_GET_CREATE_PLIST](https://portal.hdfgroup.org/display/HDF5/H5D_GET_CREATE_PLIST) for details.

`get_access_plist()` This function implements the HDF5-API function `H5Dget_access_plist`. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5D_GET_ACCESS_PLIST](https://portal.hdfgroup.org/display/HDF5/H5D_GET_ACCESS_PLIST) for details.

`get_offset()` This function implements the HDF5-API function `H5Dget_offset`. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5D_GET_OFFSET](https://portal.hdfgroup.org/display/HDF5/H5D_GET_OFFSET) for details.

`get_storage_size()` This function implements the HDF5-API function `H5Dget_storage_size`. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5D_GET_STORAGE_SIZE](https://portal.hdfgroup.org/display/HDF5/H5D_GET_STORAGE_SIZE) for details.

`vlen_get_buf_size(type, space)` This function implements the HDF5-API function `H5Dvlen_get_buf_size`. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5D_VLEN_GET_BUF_SIZE](https://portal.hdfgroup.org/display/HDF5/H5D_VLEN_GET_BUF_SIZE) for details.

`vlen_reclaim(buffer, type, space, dataset_xfer_pl = h5const$H5P_DEFAULT)` This function implements the HDF5-API function `H5Dvlen_reclaim`. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5D_VLEN_RECLAIM](https://portal.hdfgroup.org/display/HDF5/H5D_VLEN_RECLAIM) for details.

`read_low_level(file_space = h5const$H5S_ALL, mem_space = NULL, mem_type = NULL, dataset_xfer_pl = h5const$H5P_DEFAULT, flags = getOption("hdf5r.h5tor_default"), set_dim = FALSE, dim_to_set = NULL, drop = TRUE)` This function is for advanced users. It is recommended to use `read` instead or the `[` interface. This function implements the HDF5-API function `H5Dread`, with minor changes to the API to accommodate R. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5D_READ](https://portal.hdfgroup.org/display/HDF5/H5D_READ) for details. It reads the data in the dataset as specified by `mem_space` and return it as an R-obj

**Parameters**

- `file_space` An HDF5-space, represented as class `H5S` that determines which part of the dataset is being read. Can also be given as an id
**mem_space**  The space as it is represented in memory; advanced feature; may be removed in the future. Can also be given as an id.

**mem_type**  Memory type; extracted from the dataset if null (can be passed in for efficiency reasons Can also be given as an id.

**dataset_xfer_pl**  Dataset transfer property list. See `H5P_DATASET_XFER`

**flags**  Conversion rules for integer values. See also `h5const`

**set_dim**  If TRUE, the dimension attribute is set in the return value. How it is set is determined by `dim_to_set`.

**dim_to_set**  The dimension to set; Has to be numeric and needs to be specified if `set_dim` is TRUE. If the result is a data.frame, i.e. the data-type is a compound, then the dimension is ignored as the correct dimension is already set.

**drop**  Logical. Should dimensions of length 1 be dropped (R-default for arrays)

*`read`*

**args**  The indices for each dimension to subset given as a list. This makes this easier to use as a programmatic API. For interactive use we recommend the use of the `[` operator. If set to NULL, the entire dataset will be read.

**envir**  The environment in which to evaluate `args`

**dataset_xfer_pl**  An object of class `H5P_DATASET_XFER`.

**flags**  Some flags governing edge cases of conversion from HDF5 to R. This is related to how integers are being treated and the issue of R not being able to natively represent 64bit integers and not at all being able to represent unsigned 64bit integers (even using add-on packages). The constants governing this are part of `h5const`. The relevant ones start with the term `H5TOR` and are documented there. The default set here returns a regular 32bit integer if it doesn’t lead to an overflow and returns a 64bit integer from the `bit64` package otherwise. For 64bit unsigned int that are larger than 64bit signed int, it return a double. This looses precision, however.

**drop**  Logical. When reading data, should dimensions of size 1 be dropped.

*Return*

The data that was read as an R object

*`write_low_level`*

**robj**  The object to write into the dataset

**mem_space**  The space as it is represented in memory; advanced feature; may be removed in the future

**mem_type**  Memory type; extracted from the dataset if null (can be passed in for efficiency reasons

**file_space**  An HDF5-space, represented as class `H5S` that determines which part of the dataset is being written.
**dataset_xfer_pl** Dataset transfer property list. See `H5P_DATASET_XFER`.

**flush** Should a flush be done after the write.

```r
write(args, value, dataset_xfer_pl = h5const$H5P_DEFAULT, envir = parent.frame())
```

Main interface for writing data to the dataset. It is the function that is used by `[<-`, where all indices are being passed in the parameter `args`.

**Parameters**

- **args** The indices for each dimension to subset given as a list. This makes this easier to use as a programmatic API. For interactive use we recommend the use of the `[ operator. If set to `NULL`, the entire dataset will be read.
- **value** The data to write to the dataset.
- **envir** The environment in which to evaluate `args`.
- **dataset_xfer_pl** An object of class `H5P_DATASET_XFER`.

**Return**

The HDF5 dataset object, returned invisibly.

- **set_extent(dims)** This function implements the HDF5-API function H5Dset_extent. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5D_SET_EXTENT](https://portal.hdfgroup.org/display/HDF5/H5D_SET_EXTENT) for details.
- **get_fill_value()** This function implements the HDF5-API function H5Pget_fill_value, automatically supplying the datatype of the dataset for convenience. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5P_GET_FILL_VALUE](https://portal.hdfgroup.org/display/HDF5/H5P_GET_FILL_VALUE) for details.
- **create_reference(...)** This function implements the HDF5-API function H5Rcreate. The parameters are interpreted as in `[`. The function always creates `H5R_DATASET_REGION` references. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5R_CREATE](https://portal.hdfgroup.org/display/HDF5/H5R_CREATE) for details.
- **print(..., max.attributes = 10)** Prints information for the dataset.

**Parameters**

- **...** ignored.
- **max.attributes** Maximum number of attribute names to print.

- **obj_info(remove_internal_use_only = TRUE)** This function implements the HDF5-API function H5Oget_info. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5O_GET_INFO](https://portal.hdfgroup.org/display/HDF5/H5O_GET_INFO) for details.
- **get_obj_name()** This function implements the HDF5-API function H5Iget_name. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5I_GET_NAME](https://portal.hdfgroup.org/display/HDF5/H5I_GET_NAME) for details.
- **create_attr(attr_name, robj = NULL, dtype = NULL, space = NULL)** This function implements the HDF5-API function H5Acreate2. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5A_CREATE2](https://portal.hdfgroup.org/display/HDF5/H5A_CREATE2) for details.
- **attr_open(attr_name)** This function implements the HDF5-API function H5Aopen. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5A_OPEN](https://portal.hdfgroup.org/display/HDF5/H5A_OPEN) for details.
- **create_attr_by_name(attr_name, obj_name, robj = NULL, dtype = NULL, space = NULL, link_access_pl = h5const$H5P_DEFAULT)** This function implements the HDF5-API function H5Acreate_by_name. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5A_CREATE_BY_NAME](https://portal.hdfgroup.org/display/HDF5/H5A_CREATE_BY_NAME) for details.
attr_open_by_name(attr_name, obj_name, link_access_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5Aopen_by_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_OPEN_BY_NAME for details.

attr_open_by_idx(n, obj_name, idx_type = h5const$H5_INDEX_NAME, order = h5const$H5_ITER_NATIVE, link_access_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5Aopen_by_idx. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_OPEN_BY_IDX for details.

attr_exists_by_name(attr_name, obj_name, link_access_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5Aexists_by_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_EXISTS_BY_NAME for details.

attr_exists(attr_name) This function implements the HDF5-API function H5Aexists. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_EXISTS for details.

attr_rename_by_name(old_attr_name, new_attr_name, obj_name, link_access_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5Arename_by_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_RENAME_BY_NAME for details.

attr_rename(old_attr_name, new_attr_name) This function implements the HDF5-API function H5Arename. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_RENAME for details.

attr_delete(attr_name) This function implements the HDF5-API function H5Adelete. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_DELETE for details.

attr_delete_by_name(attr_name, obj_name, link_access_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5Adelete_by_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_DELETE_BY_NAME for details.

attr_delete_by_idx(n, obj_name, idx_type = h5const$H5_INDEX_NAME, order = h5const$H5_ITER_NATIVE, link_access_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5Adelete_by_idx. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_DELETE_BY_IDX for details.

attr_info_by_name(attr_name, obj_name, link_access_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5Aget_info_by_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_GET_INFO_BY_NAME for details.

attr_info_by_idx(n, obj_name, idx_type = h5const$H5_INDEX_NAME, order = h5const$H5_ITER_NATIVE, link_access_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5Aget_info_by_idx. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_GET_INFO_BY_IDX for details.

attr_name_by_idx(n, obj_name, idx_type = h5const$H5_INDEX_NAME, order = h5const$H5_ITER_NATIVE, link_access_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5Aget_name_by_idx. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_GET_NAME_BY_IDX for details.

attr_get_number() This function implements the HDF5-API function H5Aget_numAttrs. Please see the documentation at https://support.hdfgroup.org/HDF5/doc/RM/RM_H5A.html#Annot-NumAttrs for details.
flush(scope = h5const$H5F_SCOPE_GLOBAL) This function implements the HDF5-API function H5Fflush. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5F_FLUSH for details.

get_filename() This function implements the HDF5-API function H5Fget_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5F_GET_NAME for details.

dims() Get the dimension of the dataset
maxdims() Get the maximal dimension of the dataset
chunk_dims() Return the dimension of the chunks. NA if the dataset is not chunked
key_info() Returns the key types as a list, consisting of type, space, dataset_create_pl, type_size_raw, type_size_variable, dims and chunk_dims. type_size_raw versus variable differs for variable length types, which return Inf for type_size_variable and the underlying size for type_size_raw

Author(s)

Holger Hoefling

Examples

# First create a file to create datasets in it
fname <- tempfile(fileext = ".h5")
file <- H5File$new(fname, mode = "a")

# Show the 3 different ways how to create a dataset
file[["directly"]]
file$create_dataset("from robj", matrix(1:10, ncol=2))
dset <- file$create_dataset("basic", dtype=h5types$H5T_NATIVE_INT,
    space=H5S$new("simple", dims=c(5, 2), maxdims=c(10,2)), chunk_dims=c(5,2))

# Different ways of reading the dataset
dset$read(args=list(1:5, 1))
dset$read(args=list(1:5, quote(expr=)))
dset$read(args=list(1:5, NULL))
dset[1:5, 1]
dset[1:5, ]
dset[1:5, NULL]

# Writing to the dataset
dset$write(args=list(1:3, 1:2), value=11:16)
dset[4:5, 1:2] <- -(1:4)
dset[,]

# Extract key information
dset$dims
dset$maxdims
dset$chunk_dims
dset$key_info
dset

file$close_all()
file.remove(fname)
Description

`H5File` objects are the main entry point to access HDF5 data from binary files. This class represents an open HDF5 File-id. It inherits all functions of the `H5RefClass`.

Details

HDF5 files can be opened or generated using the `H5File$new()` function and a specified file access mode. `H5File$new()` returns a `H5File` object which can be used to access `H5Groups` and Datasets (see `H5D`) using subsetting parameters or according class methods.

HDF5 files which have been created or opened through `H5File$new()` need to be closed afterwards using `$close_all()`. `$close_all()` not only closes the file itself, but also all objects that are still open inside it (such as groups or datasets). `$flush()` can be used to flush unwritten data to an HDF5 file.

HDF5 Files typically contain the following objects:

**Groups** Similar to a file system folder, used to organize HDF5 objects in a hierarchical way, see also `H5Group`

**Datasets** Objects to store actual data, see also `H5D`

**Attributes** Meta data objects to store extra information about Files, Groups and Datasets, see also `H5A`

Value

Object of class `H5File`.

Methods

- `new(filename = NULL, mode = c("a", "r", "r+", "w", "w-", "x"), file_create_pl = h5const$H5P_DEFAULT, file_access_pl = h5const$H5P_DEFAULT, id = NULL)`
  Opens or creates a new HDF5 File

  **Parameters**

  - `filename` Name of the file
  - `mode` How to open it. `a` creates a new file or opens an existing one for read/write. `r` opens an existing file for reading, `r+` opens an existing file for read/write. `w` creates a file, truncating any existing ones and `w-` or `x` are synonyms, creating a file and failing if it already exists.

  - `get_obj_count(types = h5const$H5F_OBJ_ALL)` This function implements the HDF5-API function `H5Fget_obj_count`. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5F_GET_OBJ_COUNT for details.

  - `get_obj_ids(types = h5const$H5F_OBJ_ALL)` This function implements the HDF5-API function `H5Fget_obj_ids`. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5F_GET_OBJ_IDS for details.
get_filesize() This function implements the HDF5-API function H5Fget_filesize. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5F_GET_FILESIZE for details.

file_info() This function implements the HDF5-API function H5Fget_info2. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5F_GET_INFO2 for details. Please note that the returned information differs if HDF5 Version 1.8.16 or HDF5 Version >= 1.10.0 is being used.

get_intent() This function implements the HDF5-API function H5Fget_intent. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5F_GET_INTENT for details.

close_all(close_self = TRUE) Closes the file, flushes it and also closes all open objects that are still open in it. This is the recommended way of closing any file. If not all objects in a file are closed, the file remains open and cannot be re-opened the regular way.

print(..., max.attributes = 10, max.listing = 10) Prints information for the file

Parameters

max.attributes Maximum number of attribute names to print
max.listing Maximum number of ls-items to print
... ignored

open(name, link_access_pl = h5const$H5P_DEFAULT, dataset_access_pl = h5const$H5P_DEFAULT, type_access_pl = h5const$H5P_DEFAULT) Opens groups, datasets or types using the appropriate HDF5-API functions. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5D_OPEN for datasets, https://portal.hdfgroup.org/display/HDF5/H5O_OPEN for types and https://portal.hdfgroup.org/display/HDF5/H5O_OPEN for general objects.

open_by_idx(n, group_name = ".", index_type = h5const$H5_INDEX_NAME, order = h5const$H5_ITER_NATIVE, link_access_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5Oopen_by_idx. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5O_OPEN_BY_IDX for details.

ls(recursive = FALSE, detailed = FALSE, index_type = h5const$H5_INDEX_NAME, order = h5const$H5_ITER_NATIVE, link_access_pl = h5const$H5P_DEFAULT, dataset_access_pl = h5const$H5P_DEFAULT, type_access_pl = h5const$H5P_DEFAULT) Returns the contents of a file or group as a data.frame.

exists(name, link_access_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5Lexists. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_EXISTS for details.

path_valid(path, check_object_valid = TRUE) This function implements the HDF5-API function H5LTpath_valid. Please see the documentation at https://support.hdfgroup.org/HDF5/doc/HL/RM_H5LT.html#H5LTpath_valid for details.

link(obj, new_link_name, link_create_pl = h5const$H5P_DEFAULT, link_access_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5Olink. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5O_LINK for details.

obj_copy_to(dst_loc, dst_name, src_name, object_copy_pl = h5const$H5P_DEFAULT, link_create_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5Ocopy. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5O_COPY for details.

obj_copy_from(src_loc, src_name, dst_name, object_copy_pl = h5const$H5P_DEFAULT, link_create_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5Ocopy. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5O_COPY for details.
**H5File-class**

H5File-class

25

obj_info_by_idx(n, group_name = ".", index_field = h5const$H5_INDEX_NAME, order = h5const$H5_ITER_NATIVE, remove_internal_use_only = TRUE)

This function implements the HDF5-API function H5Oget_info_by_idx. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5O_GET_INFO_BY_IDX for details.

obj_info_by_name(object_name, remove_internal_use_only = TRUE)

This function implements the HDF5-API function H5Oget_info_by_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5O_GET_INFO_BY_NAME for details.

group_info()

This function implements the HDF5-API function H5Gget_info. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5G_GET_INFO for details.

group_info_by_name(name, link_access_pl = h5const$H5P_DEFAULT)

This function implements the HDF5-API function H5Gget_info_by_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5G_GET_INFO_BY_NAME for details.

group_info_by_idx(n, group_name = ".", index_field = h5const$H5_INDEX_NAME, order = h5const$H5_ITER_NATIVE, link_access_pl = h5const$H5P_DEFAULT)

This function implements the HDF5-API function H5Gget_info_by_idx. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5G_GET_INFO_BY_IDX for details.

create_group(name, link_create_pl = h5const$H5P_DEFAULT, group_create_pl = h5const$H5P_DEFAULT, group_access_pl = h5const$H5P_DEFAULT)

This function implements the HDF5-API function H5Gcreate2 and H5Gcreate_anon (if name is NULL). Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5G_CREATE2 for regular groups and https://portal.hdfgroup.org/display/HDF5/H5G_CREATE_ANON for anonymous groups for details.

create_dataset(name, robj = NULL, dtype = NULL, space = NULL, dims = NULL, chunk_dims = "auto", gzip_level = 4, link_create_pl = h5const$H5P_DEFAULT, dataset_create_pl = h5const$H5P_DEFAULT, dataset_access_pl = h5const$H5P_DEFAULT)

This function is the main interface to create a new dataset. Its parameters allow for customization of the default behavior, i.e. in order to get a specific datatype, a certain chunk size or dataset dimensionality. Also note that this function implements the HDF5-API function H5Dcreate2 and H5Dcreate_anon (if name is NULL). Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5D_CREATE2 for regular groups and https://portal.hdfgroup.org/display/HDF5/H5D_CREATE_ANON for anonymous groups for details.

**Parameters**

ame The name of the new dataset. If missing, an anonymous dataset is created

robj An R-object to take as a template for creating the dataset. Either robj or both dtype and space have to be provided

dtype The datatype to use for the creation of the object. Can be null if robj is given.

space The space to use for the object creation. Can be null if robj is given. Otherwise an object of type H5S which specifies the dimensions of the dataset.

dims Dimension of the new dataset; used if space is NULL. overwrite the dimension guessed from robj if robj is given.

chunk_dims Size of the chunk. Has to have the same length as the dataset dimension. If "auto" then the size of each chunk is estimated so that each chunk is roughly as large in bytes as the value in the hdf5r.chunk_size option. See also guess_chunks for a more detailed explanation. If set to NULL, then no chunking is used, unless explicitly set in dataset_create_pl.

gzip_level Only if chunk_dims is not null. If given, then the dataset_create_pl is set to enable zipping at the level given here. If set to NULL, then gzip is not set (but could be set otherwise in dataset_create_pl).
**link_create_pl**  Link creation property list. See H5P_LINK_CREATE

**dataset_create_pl**  Dataset creation property list. See H5P_DATASET_CREATE

**dataset_access_pl**  Dataset access property list. See H5P_DATASET_ACCESS

commit(name, dtype, link_create_pl = h5const$H5P_DEFAULT, type_create_pl = h5const$H5P_DEFAULT, type_access_pl = h5const$H5P_DEFAULT)

This function implements the HDF5-API function H5Tcommit2. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5T_COMMIT2 for details.

link_create_hard(obj_loc, obj_name, link_name, link_create_pl = h5const$H5P_DEFAULT, link_access_pl = h5const$H5P_DEFAULT)

This function implements the HDF5-API function H5Lcreate_hard. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_CREATE_HARD for details.

link_create_soft(target_path, link_name, link_create_pl = h5const$H5P_DEFAULT, link_access_pl = h5const$H5P_DEFAULT)

This function implements the HDF5-API function H5Lcreate_soft. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_CREATE_SOFT for details.

link_create_external(target_filename, target_obj_name, link_name, link_create_pl = h5const$H5P_DEFAULT, link_access_pl = h5const$H5P_DEFAULT)

This function implements the HDF5-API function H5Lcreate_external. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_CREATE_EXTERNAL for details.

link_exists(name, link_access_pl = h5const$H5P_DEFAULT)

This function implements the HDF5-API function H5Lexists. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_EXISTS for details.

link_move_from(src_loc, src_name, dst_name, link_create_pl = h5const$H5P_DEFAULT, link_access_pl = h5const$H5P_DEFAULT)

This function implements the HDF5-API function H5Lmove. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_MOVE for details.

link_move_to(dst_loc, dst_name, src_name, link_create_pl = h5const$H5P_DEFAULT, link_access_pl = h5const$H5P_DEFAULT)

This function implements the HDF5-API function H5Lmove. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_MOVE for details.

link_copy_from(src_loc, src_name, dst_name, link_create_pl = h5const$H5P_DEFAULT, link_access_pl = h5const$H5P_DEFAULT)

This function implements the HDF5-API function H5Lcopy. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_COPY for details.

link_copy_to(dst_loc, dst_name, src_name, link_create_pl = h5const$H5P_DEFAULT, link_access_pl = h5const$H5P_DEFAULT)

This function implements the HDF5-API function H5Lcopy. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_COPY for details.

link_delete(name, link_access_pl = h5const$H5P_DEFAULT)

This function implements the HDF5-API function H5Ldelete. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_DELETE for details.

link_delete_by_idx(n, group_name = ".", index_field = h5const$H5_INDEX_NAME, order = h5const$H5_ITER_NATIVE, link_access_pl = h5const$H5P_DEFAULT)

This function implements the HDF5-API function H5Ldelete_by_idx. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_DELETE_BY_IDX for details.

link_info(name, link_access_pl = h5const$H5P_DEFAULT)

This function implements the HDF5-API function H5Lget_info. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_GET_INFO for details.

link_info_by_idx(n, group_name = ".", index_field = h5const$H5_INDEX_NAME, order = h5const$H5_ITER_NATIVE, link_access_pl = h5const$H5P_DEFAULT)

This function implements the HDF5-API function H5Lget_info_by_idx. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_GET_INFO_BY_IDX for details.
link_value(name, link_access_pl = h5const$H5P_DEFAULT)  This function implements the HDF5-API function H5Lget_val. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_GET_VAL for details.

link_value_by_idx(n, group_name = ".", index_field = h5const$H5_INDEX_NAME, order = h5const$H5_ITER_NATIVE, link_access_pl = h5const$H5P_DEFAULT)  This function implements the HDF5-API function H5Lget_val_by_idx. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_GET_VAL_BY_IDX for details.

link_name_by_idx(n, group_name, idx_type = h5const$H5_INDEX_NAME, order = h5const$H5_ITER_NATIVE, link_access_pl = h5const$H5P_DEFAULT)  This function implements the HDF5-API function H5Lget_name_by_idx. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_GET_NAME_BY_IDX for details.

mount(name, child)  This function implements the HDF5-API function H5Fmount. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5F_MOUNT for details.

unmount(name)  This function implements the HDF5-API function H5Funmount. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5F_UNMOUNT for details.

create_reference(name = ".", space = NULL)  This function implements the HDF5-API function H5Rcreate. If space=NULL then a H5R_OBJECT reference is created, otherwise a H5R_DATASET_REGION reference Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5R_CREATE for details.

obj_info(remove_internal_use_only = TRUE)  This function implements the HDF5-API function H5Oget_info. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5O_GET_INFO for details.

get_obj_name()  This function implements the HDF5-API function H5Iget_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5I_GET_NAME for details.

create_attr(attr_name, robj = NULL, dtype = NULL, space = NULL)  This function implements the HDF5-API function H5Acreate2. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_CREATE2 for details.

attr_open(attr_name)  This function implements the HDF5-API function H5Aopen. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_OPEN for details.

create_attr_by_name(attr_name, obj_name, robj = NULL, dtype = NULL, space = NULL, link_access_pl = h5const$H5P_DEFAULT)  This function implements the HDF5-API function H5Acreate_by_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_CREATE_BY_NAME for details.

attr_open_by_name(attr_name, obj_name, link_access_pl = h5const$H5P_DEFAULT)  This function implements the HDF5-API function H5Aopen_by_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_OPEN_BY_NAME for details.

attr_open_by_idx(n, obj_name, idx_type = h5const$H5_INDEX_NAME, order = h5const$H5_ITER_NATIVE, link_access_pl = h5const$H5P_DEFAULT)  This function implements the HDF5-API function H5Aopen_by_idx. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_OPEN_BY_IDX for details.

attr_exists_by_name(attr_name, obj_name, link_access_pl = h5const$H5P_DEFAULT)  This function implements the HDF5-API function H5Aexists_by_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_EXISTS_BY_NAME for details.
attr_exists(attr_name) This function implements the HDF5-API function H5A_exists. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_EXISTS for details.

attr_rename_by_name(old_attr_name, new_attr_name, obj_name, link_access_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5A_rename_by_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_RENAME_BY_NAME for details.

attr_rename(old_attr_name, new_attr_name) This function implements the HDF5-API function H5A_rename. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_RENAME for details.

attr_delete(attr_name) This function implements the HDF5-API function H5A_delete. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_DELETE for details.

attr_delete_by_name(attr_name, obj_name, link_access_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5A_delete_by_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_DELETE_BY_NAME for details.

attr_delete_by_idx(n, obj_name, idx_type = h5const$H5_INDEX_NAME, order = h5const$H5_ITER_NATIVE, link_access_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5A_delete_by_idx. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_DELETE_BY_IDX for details.

attr_info_by_name(attr_name, obj_name, link_access_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5A_get_info_by_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_GET_INFO_BY_NAME for details.

attr_info_by_idx(n, obj_name, idx_type = h5const$H5_INDEX_NAME, order = h5const$H5_ITER_NATIVE, link_access_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5A_get_info_by_idx. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_GET_INFO_BY_IDX for details.

attr_name_by_idx(n, obj_name, idx_type = h5const$H5_INDEX_NAME, order = h5const$H5_ITER_NATIVE, link_access_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5A_get_name_by_idx. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_GET_NAME_BY_IDX for details.

attr_get_number() This function implements the HDF5-API function H5A_get_num_attrs. Please see the documentation at https://support.hdfgroup.org/HDF5/doc/RM/RM_H5A.html#Annot-NumAttrs for details.

flush(scope = h5const$H5F_SCOPE_GLOBAL) This function implements the HDF5-API function H5F_flush. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5F_FLUSH for details.

get_filename() This function implements the HDF5-API function H5F_get_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5F_GET_NAME for details.

names(link_access_pl = h5const$H5P_DEFAULT) Returns the names of the items in the group or at the root of the file

Parameters

link_access_pl The link-access property list. See https://portal.hdfgroup.org/display/HDF5/Link+Access+Properties for more detail.
Extract/List File Contents

The following functions are defined to extract HDF5 file contents:

- `list.groups` List HDF5 groups in file.
- `list.datasets` List HDF5 datasets in file.
- `names` List all items in a file or group (applicable for `H5File` and `H5Group`)
- `list.attributes` List Attributes of HDF5 object (file, group or dataset).
- `h5attr_names` Attribute names of an HDF5 object; similar to `list.attributes`

Author(s)

Holger Hoefling, Mario Annau

See Also

`h5file`

Examples

```r
# The following examples generates a HDF5 file with the different HDF5
# Objects and shows its contents:
fname <- tempfile(fileext = ".h5")
file <- H5File$new(fname, mode = "a")
file["testdataset"] <- 1:10
h5attr(file, "testattrib") <- LETTERS[1:10]
file$create_group("testgroup")
file["testgroup/testdataset2"] <- 1:10
# Show contents of file
file
# Close file and delete
file$close_all()

# The following example shows hdf5 file contents and how to use them to iterate over HDF5 elements:
file <- h5file(fname, mode = "a")
sapply(c("testgroup1", "testgroup2", "testgroup3"), file$create_group)
file["testgroup1/testset1"] <- 1:10
file["testgroup2/testset2"] <- 11:20
file["testgroup3/testset3"] <- 21:30

# Extract first 3 elements from each dataset and combine result to matrix
sapply(list.datasets(file, recursive = TRUE), function(x) file[[x]][1:3])
# Close file
file$close_all()
file.remove(fname)
```
h5garbage_collect  
*Trigger the HDF5 garbage collection*

**Description**

Trigger the HDF5 garbage collection

**Usage**

`h5garbage_collect()`

**Details**

This function triggers the HDF5 internal garbage collection. It is independent of the R garbage collection and currently has to be triggered by hand.

**Value**

Invisible NULL

**Author(s)**

Holger Hoefling

---

**H5Group-class**  
*Class for representing HDF5 groups*

**Description**

HDF5-Groups are essentially equivalent to directories in a file system. Inside the groups, other groups or datasets can be created. For the most parts, groups behave like files, so please also look at the documentation of `H5File`.

**Value**

Object of class `H5Group`.

**Methods**

`print(..., max.attributes = 10, max.listing = 10)` Prints information for the group

**Parameters**

- `max.attributes` Maximum number of attribute names to print
- `max.listing` Maximum number of ls-items to print
  
... ignored
open(name, link_access_pl = h5const$H5P_DEFAULT, dataset_access_pl = h5const$H5P_DEFAULT, type_access_pl = h5const$H5P_DEFAULT)


open_by_idx(n, group_name = ".\", index_type = h5const$H5_INDEX_NAME, order = h5const$H5_ITER_NATIVE, link_access_pl = h5const$H5P_DEFAULT)

This function implements the HDF5-API function H5Oopen_by_idx. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5O_OPEN_BY_IDX for details.

ls(recursive = FALSE, detailed = FALSE, index_type = h5const$H5_INDEX_NAME, order = h5const$H5_ITER_NATIVE, link_access_pl = h5const$H5P_DEFAULT, dataset_access_pl = h5const$H5P_DEFAULT, type_access_pl = h5const$H5P_DEFAULT)

Returns the contents of a file or group as a data.frame.

events(name, link_access_pl = h5const$H5P_DEFAULT)

This function implements the HDF5-API function H5Lexists. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_EXISTS for details.

path_valid(path, check_object_valid = TRUE)

This function implements the HDF5-API function H5LPvalid. Please see the documentation at https://support.hdfgroup.org/HDF5/doc/HL/RM_H5LT.html#H5LTpath_valid for details.

link(obj, new_link_name, link_create_pl = h5const$H5P_DEFAULT, link_access_pl = h5const$H5P_DEFAULT)

This function implements the HDF5-API function H5Olink. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5O_LINK for details.

obj_copy_to(dst_loc, dst_name, src_name, object_copy_pl = h5const$H5P_DEFAULT, link_create_pl = h5const$H5P_DEFAULT)

This function implements the HDF5-API function H5Ocopy. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5O_COPY for details.

obj_copy_from(src_loc, src_name, dst_name, object_copy_pl = h5const$H5P_DEFAULT, link_create_pl = h5const$H5P_DEFAULT)

This function implements the HDF5-API function H5Ocopy. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5O_COPY for details.

obj_info_by_idx(n, group_name = ".\", index_field = h5const$H5_INDEX_NAME, order = h5const$H5_ITER_NATIVE, link_access_pl = h5const$H5P_DEFAULT)

This function implements the HDF5-API function H5Oget_info_by_idx. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5O_GET_INFO_BY_IDX for details.

obj_info_by_name(object_name, remove_internal_use_only = TRUE)

This function implements the HDF5-API function H5Oget_info_by_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5O_GET_INFO_BY_NAME for details.

group_info() 

This function implements the HDF5-API function H5Gget_info. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5G_GET_INFO for details.

group_info_by_name(name, link_access_pl = h5const$H5P_DEFAULT)

This function implements the HDF5-API function H5Gget_info_by_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5G_GET_INFO_BY_NAME for details.

create_group(name, link_create_pl = h5const$H5P_DEFAULT, group_create_pl = h5const$H5P_DEFAULT, group_access_pl = h5const$H5P_DEFAULT)

This function implements the HDF5-API function H5Gcreate2 and H5Gcreate_anon (if name is NULL). Please see the documentation at https://portal.hdfgroup.org/display/HDF5/
H5G_CREATE2 for regular groups and https://portal.hdfgroup.org/display/HDF5/H5G_CREATE_ANON for anonymous groups for details.

`create_dataset(name, robj = NULL, dtype = NULL, space = NULL, dims = NULL, chunk_dims = "auto", gzip_level = NULL, link_create_pl = h5const$H5P_DEFAULT, dataset_create_pl = h5const$H5P_DEFAULT, dataset_access_pl = h5const$H5P_DEFAULT)`

This function is the main interface to create a new dataset. Its parameters allow for customization of the default behavior, i.e. in order to get a specific datatype, a certain chunk size or dataset dimensionality. Also note that this function implements the HDF5-API function H5Dcreate2 and H5Dcreate_anon (if name is NULL). Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5D_CREATE2 for regular groups and https://portal.hdfgroup.org/display/HDF5/H5D_CREATE_ANON for anonymous groups for details.

**Parameters**

- **name** The name of the new dataset. If missing, an anonymous dataset is created.
- **robj** An R-object to take as a template for creating the dataset. Either robj or both dtype and space have to be provided.
- **dtype** The datatype to use for the creation of the object. Can be null if robj is given.
- **space** The space to use for the object creation. Can be null if robj is given. Otherwise an object of type H5S which specifies the dimensions of the dataset.
- **dims** Dimension of the new dataset; used if space is NULL. overwrite the dimension guessed from robj if robj is given.
- **chunk_dims** Size of the chunk. Has to have the same length as the dataset dimension. If "auto" then the size of each chunk is estimated so that each chunk is roughly as large in bytes as the value in the hdf5r.chunk_size option. See also guess_chunks for a more detailed explanation. If set to NULL, then no chunking is used, unless explicitly set in dataset_create_pl.
- **gzip_level** Only if chunk_dims is not null. If given, then the dataset_create_pl is set to enable zipping at the level given here. If set to NULL, then gzip is not set (but could be set otherwise in dataset_create_pl)
- **link_create_pl** Link creation property list. See H5P_LINK_CREATE
- **dataset_create_pl** Dataset creation property list. See H5P_DATASET_CREATE
- **dataset_access_pl** Dataset access property list. See H5P_DATASET_ACCESS

`commit(name, dtype, link_create_pl = h5const$H5P_DEFAULT, type_create_pl = h5const$H5P_DEFAULT, type_access_pl = h5const$H5P_DEFAULT)`

This function implements the HDF5-API function H5Tcommit2. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5T_COMMIT2 for details.

`link_create_hard(obj_loc, obj_name, link_name, link_create_pl = h5const$H5P_DEFAULT, link_access_pl = h5const$H5P_DEFAULT)`

This function implements the HDF5-API function H5Lcreate_hard. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_CREATE_HARD for details.

`link_create_soft(target_path, link_name, link_create_pl = h5const$H5P_DEFAULT, link_access_pl = h5const$H5P_DEFAULT)`

This function implements the HDF5-API function H5Lcreate_soft. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_CREATE_SOFT for details.

`link_create_external(target_filename, target_obj_name, link_name, link_create_pl = h5const$H5P_DEFAULT)`

This function implements the HDF5-API function H5Lcreate_external. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_CREATE_EXTERNAL for details.

`link_exists(name, link_access_pl = h5const$H5P_DEFAULT)`

This function implements the HDF5-API function H5Lexists. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_EXISTS for details.
This function implements the HDF5-API function H5Lmove. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_MOVE for details.

This function implements the HDF5-API function H5Lmove. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_MOVE for details.

This function implements the HDF5-API function H5Lcopy. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_COPY for details.

This function implements the HDF5-API function H5Lcopy. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_COPY for details.

This function implements the HDF5-API function H5Ldelete. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_DELETE for details.

This function implements the HDF5-API function H5Ldelete. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_DELETE_BY_IDX for details.

This function implements the HDF5-API function H5Lget_info. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_GET_INFO for details.

This function implements the HDF5-API function H5Lget_info_by_idx. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_GET_INFO_BY_IDX for details.

This function implements the HDF5-API function H5Lget_val. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_GET_VAL for details.

This function implements the HDF5-API function H5Lget_val_by_idx. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_GET_VAL_BY_IDX for details.

This function implements the HDF5-API function H5Lget_name_by_idx. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5L_GET_NAME_BY_IDX for details.

This function implements the HDF5-API function H5Fmount. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5F_MOUNT for details.

This function implements the HDF5-API function H5Fumount. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5F_UNMOUNT for details.

This function implements the HDF5-API function H5Rcreate. If space=NULL then a H5R_OBJECT reference is created, otherwise a H5R_DATASET_REGION reference Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5R_CREATE for details.
obj_info(remove_internal_use_only = TRUE) This function implements the HDF5-API function H5Oget_info. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5O_GET_INFO for details.

get_obj_name() This function implements the HDF5-API function H5Iget_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5I_GET_NAME for details.

create_attr(attr_name, robj = NULL, dtype = NULL, space = NULL) This function implements the HDF5-API function H5Acreate2. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_CREATE2 for details.

attr_open(attr_name) This function implements the HDF5-API function H5Aopen. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_OPEN for details.

create_attr_by_name(attr_name, obj_name, robj = NULL, dtype = NULL, space = NULL, link_access_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5Acreate_by_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_CREATE_BY_NAME for details.

attr_open_by_name(attr_name, obj_name, link_access_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5Aopen_by_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_OPEN_BY_NAME for details.

attr_open_by_idx(n, obj_name, idx_type = h5const$H5_INDEX_NAME, order = h5const$H5_ITER_NATIVE, link_access_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5Aopen_by_idx. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_OPEN_BY_IDX for details.

attr_exists_by_name(attr_name, obj_name, link_access_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5Aexists_by_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_EXISTS_BY_NAME for details.

attr_exists(attr_name) This function implements the HDF5-API function H5Aexists. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_EXISTS for details.

attr_rename_by_name(old_attr_name, new_attr_name, obj_name, link_access_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5Arename_by_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_RENAME_BY_NAME for details.

attr_rename(old_attr_name, new_attr_name) This function implements the HDF5-API function H5Arename. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_RENAME for details.

attr_delete(attr_name) This function implements the HDF5-API function H5Adelete. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_DELETE for details.

attr_delete_by_name(attr_name, obj_name, link_access_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5Adelete_by_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_DELETE_BY_NAME for details.

attr_delete_by_idx(n, obj_name, idx_type = h5const$H5_INDEX_NAME, order = h5const$H5_ITER_NATIVE, link_access_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5Adelete_by_idx. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_DELETE_BY_IDX for details.
attr_info_by_name(attr_name, obj_name, link_access_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5Aget_info_by_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_GET_INFO_BY_NAME for details.

attr_info_by_idx(n, obj_name, idx_type = h5const$H5_INDEX_NAME, order = h5const$H5_ITER_NATIVE, link_access_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5Aget_info_by_idx. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_GET_INFO_BY_IDX for details.

attr_name_by_idx(n, obj_name, idx_type = h5const$H5_INDEX_NAME, order = h5const$H5_ITER_NATIVE, link_access_pl = h5const$H5P_DEFAULT) This function implements the HDF5-API function H5Aget_name_by_idx. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_GET_NAME_BY_IDX for details.

attr_get_number() This function implements the HDF5-API function H5Aget_num_attrs. Please see the documentation at https://support.hdfgroup.org/HDF5/doc/RM/RM_H5A.html#Annot-NumAttrs for details.

flush(scope = h5const$H5F_SCOPE_GLOBAL) This function implements the HDF5-API function H5Fflush. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5F_FLUSH for details.

get_filename() This function implements the HDF5-API function H5Fget_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5F_GET_NAME for details.

names(link_access_pl = h5const$H5P_DEFAULT) Returns the names of the items in the group or at the root of the file

Parameters

- link_access_pl The link-access property list. See https://portal.hdfgroup.org/display/HDF5/Link+Access+Properties for more detail.

Author(s)

Holger Hoefling

Examples

fname <- tempfile(fileext = ".h5")
file <- H5File$new(fname, mode = "a")
group <- file$create_group("testgroup")
group$print()
group$close()
file$close_all()
Usage

```r
## S3 method for class 'H5Group'
x[[name, ..., link_access_pl = h5const$H5P_DEFAULT, 
    dataset_access_pl = h5const$H5P_DEFAULT, 
    type_access_pl = h5const$H5P_DEFAULT]]

## S3 method for class 'H5File'
x[[name, ..., link_access_pl = h5const$H5P_DEFAULT, 
    dataset_access_pl = h5const$H5P_DEFAULT, 
    type_access_pl = h5const$H5P_DEFAULT]]

## S3 replacement method for class 'H5Group'
x[[name, ...]] <- value

## S3 replacement method for class 'H5File'
x[[name, ...]] <- value
```

Arguments

- `x` An object of class `H5File` or `H5Group`
- `name` Name of the object to retrieve. Has to be a character vector of length one. No integer values allowed.
- `...` Currently ignored
- `link_access_pl` An object of class `H5P_LINK_ACCESS`.
- `dataset_access_pl` An object of class `H5P_DATASET_ACCESS`.
- `type_access_pl` Currently always `h5const$H5P_DEFAULT`.
- `value` What to assign. Has to be one of `H5Group`, `H5D` or `H5T`.

Details

Works similar to retrieving objects in a list. `x[["my_name"]]]` retrieves object `my_name` from the HDF5-File or HDF5-Group `x`.

One can also assign objects under a not yet existing name. For either a `H5Group` or `H5D`, a hard link is created. If it is a datatype, `H5T`, this is committed under the chosen name `name`.

Value

A `H5Group`, `H5D` or `H5T`, depending on the object saved in the group under the requested name.

Author(s)

Holger Hoefling
**Description**

This is the base class for all property lists, but most have a specialized class. It inherits all functions of the `H5RefClass`. It is also the base class for many other classes, specifically

- **Dataset Creation** `H5P_DATASET_CREATE`
- **Dataset Access** `H5P_DATASET_ACCESS`
- **Dataset Transfer** `H5P_DATASET_XFER`
- **Link Creation** `H5P_LINK_CREATE`
- **Link Access** `H5P_LINK_ACCESS`
- **Object Creation** `H5P_OBJECT_CREATE`
- **Object Copy** `H5P_OBJECT_COPY`
- **Attribute Creation** `H5P_ATTRIBUTE_CREATE`

The base class is unlikely to be needed by users - they should use the appropriate subclass required.

**Value**

Object of class `H5P`.

**Methods**

- **new(id = NULL)** Create a new property list; this function itself is unlikely to be needed by users. Users should use the classes of the type they actually require

  **Parameters**

  - **id** Internal use only

- **get_class()** This function implements the HDF5-API function H5Pget_class. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5P_GET_CLASS](https://portal.hdfgroup.org/display/HDF5/H5P_GET_CLASS) for details.

- **get_class_name()** This function implements the HDF5-API function H5Pget_class_name. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5P_GET_CLASS_NAME](https://portal.hdfgroup.org/display/HDF5/H5P_GET_CLASS_NAME) for details.

- **copy()** This function implements the HDF5-API function H5Pcopy. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5P_COPY](https://portal.hdfgroup.org/display/HDF5/H5P_COPY) for details.

- **equal(cmp)** This function implements the HDF5-API function H5Pequal. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5P_EQUAL](https://portal.hdfgroup.org/display/HDF5/H5P_EQUAL) for details.

**Author(s)**

Holger Hoefling
Examples

```r	nfile <- tempfile(fileext = ".h5")
file <- H5File$new(fname, mode = "a")
file["testdataset"] <- 1:10
p <- file["testdataset"]$get_create_plist()
p$get_class()
p$get_class_name()
p$copy()
p$equal(p)
file$close_all()
```

H5P_ATTRIBUTE_CREATE-class

`Class for HDF5 property list for attribute creation`

Description

It inherits all functions of the `H5P`.

Value

Object of class `H5P_ATTRIBUTE_CREATE`.

Methods

`new(id = NULL)` Create a new class of type `H5P_ATTRIBUTE_CREATE`

Parameters

- `id` Internal use only

`set_char_encoding(encoding = h5const$H5T_CSET_UTF8)` This function implements the HDF5-API function H5Pset_char_encoding. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5P_SET_CHAR_ENCODING](https://portal.hdfgroup.org/display/HDF5/H5P_SET_CHAR_ENCODING) for details.

`get_char_encoding()` This function implements the HDF5-API function H5Pget_char_encoding. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5P_GET_CHAR_ENCODING](https://portal.hdfgroup.org/display/HDF5/H5P_GET_CHAR_ENCODING) for details.

Author(s)

Holger Hoefling

See Also

H5P
**H5P_CLASS-class**

Class for HDF5 property list classes (not HDF5 property lists)

**Description**

It inherits all functions of the **H5RefClass**. The intent of this class is to provide a mechanism to compare the class of HDF5 property classes. This is mainly intended for internal use to get the class type of an HDF5 identifier that is known to be a property list, but not of which type.

**Value**

Object of class **H5P_CLASS**.

**Methods**

equal(cmp) This function implements the HDF5-API function H5Pequal. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_EQUAL for details.

**Author(s)**

Holger Hoefling

**See Also**

**H5P**

---

**H5P_DATASET_ACCESS-class**

Class for HDF5 property list for dataset access

**Description**

It inherits all functions of the **H5P**.

**Value**

Object of class **H5P_DATASET_ACCESS**.

**Methods**

set_chunk_cache(rdcc_nslots = -1, rdcc_nbytes = -1, rdcc_w0 = -1) This function implements the HDF5-API function H5Pset_chunk_cache. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_SET_CHUNK_CACHE for details.

get_chunk_cache() This function implements the HDF5-API function H5P_get_chunk_cache. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_GET_CHUNK_CACHE for details.
**Author(s)**
Holger Hoefling

**See Also**
H5P

---

**H5P_DATASET_CREATE-class**

*Class for HDF5 property list for dataset creation*

**Description**

It inherits all functions of the **H5P**.

**Value**

Object of class **H5P_DATASET_CREATE**.

**Methods**

- **new(id = NULL)** Create a new class of type **H5P_DATASET_CREATE**
  - **Parameters**
  - **id** Internal use only

- **set_layout(layout = h5const$H5D_CHUNKED)** This function implements the HDF5-API function H5Pset_layout. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5P_SET_LAYOUT](https://portal.hdfgroup.org/display/HDF5/H5P_SET_LAYOUT) for details.

- **get_layout()** This function implements the HDF5-API function H5Pget_layout. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5P_GET_LAYOUT](https://portal.hdfgroup.org/display/HDF5/H5P_GET_LAYOUT) for details.

- **set_chunk(chunk)** This function implements the HDF5-API function H5Pset_chunk. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5P_SET_CHUNK](https://portal.hdfgroup.org/display/HDF5/H5P_SET_CHUNK) for details.

- **get_chunk(max_ndims)** This function implements the HDF5-API function H5Pget_chunk. If the layout is not chunked, returns NA. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5P_GET_CHUNK](https://portal.hdfgroup.org/display/HDF5/H5P_GET_CHUNK) for details.

- **set_deflate(level)** This function implements the HDF5-API function H5Pset_deflate. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5P_SET_DEFLATE](https://portal.hdfgroup.org/display/HDF5/H5P_SET_DEFLATE) for details.

- **set_fill_value(dtype, value)** This function implements the HDF5-API function H5Pset_fill_value. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5P_SET_FILL_VALUE](https://portal.hdfgroup.org/display/HDF5/H5P_SET_FILL_VALUE) for details.

- **get_fill_value(dtype)** This function implements the HDF5-API function H5Pget_fill_value. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5P_GET_FILL_VALUE](https://portal.hdfgroup.org/display/HDF5/H5P_GET_FILL_VALUE) for details.
set_fill_time(fill_time = h5const$H5D_FILL_TIME_IFSET) This function implements the HDF5-API function H5Pset_fill_time. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_SET_FILL_TIME for details.

get_fill_time() This function implements the HDF5-API function H5Pget_fill_time. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_GET_FILL_TIME for details.

set_alloc_time(alloc_time = h5const$H5D_ALLOC_TIME_DEFAULT) This function implements the HDF5-API function H5Pset_alloc_time. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_SET_ALLOC_TIME for details.

get_alloc_time() This function implements the HDF5-API function H5Pget_alloc_time. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_GET_ALLOC_TIME for details.

set_filter(filter = h5const$H5Z_FILTER_DEFLATE, flags = h5const$H5Z_FLAG_OPTIONAL, cd_values = integer(0)) This function implements the HDF5-API function H5Pset_filter. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_SET_FILTER for details.

all_filters_avail() This function implements the HDF5-API function H5Pall_filters_avail. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_ALL_FILTERS_AVAIL for details.

generate_filter(idx) This function implements the HDF5-API function H5Pgenerate_filter2. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_GET_FILTER2 for details.

modify_filter(filter = h5const$H5Z_FILTER_DEFLATE, flags = h5const$H5Z_FLAG_OPTIONAL, cd_values = integer(0)) This function implements the HDF5-API function H5Pmodify_filter. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_MODIFY_FILTER for details.

remove_filter(filter = h5const$H5Z_FILTER_ALL) This function implements the HDF5-API function H5Premove_filter. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_REMOVE_FILTER for details.

set_fletcher32() This function implements the HDF5-API function H5Pset_fletcher32. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_SET_FLETCHER32 for details.

set_nbit() This function implements the HDF5-API function H5Pset_nbit. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_SET_NBIT for details.

set_scaleoffset(scale_type = h5const$H5Z_SO_FLOAT_DSCALE, scale_factor = 0) This function implements the HDF5-API function H5Pset_scaleoffset. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_SET_SCALEOFFSET for details.

set_shuffle() This function implements the HDF5-API function H5Pset_shuffle. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_SET_SHUFFLE for details.

set_szip() This function implements the HDF5-API function H5Pset_szip. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_SET_SZIP for details.
set_external(filename, offset, size) This function implements the HDF5-API function H5Pset_external. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_SET_EXTERNAL for details.

get_external_count() This function implements the HDF5-API function H5Pget_external_count. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_GET_EXTERNAL_COUNT for details.

get_external(idx) This function implements the HDF5-API function H5Pget_external. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_GET_EXTERNAL for details.

Author(s)

Holger Hoefling

See Also

H5P

H5P_DATASET_XFER-class

Class for HDF5 property list for dataset transfer

Description

It inherits all functions of the H5P.

Value

Object of class H5P_DATASET_XFER.

Methods

new(id = NULL) Create a new class of type H5P_DATASET_XFER

Parameters

  id  Internal use only

set_buffer(size = 2^20) This function implements the HDF5-API function H5Pset_buffer. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_SET_BUFFER for details.

set_edc_check(check = h5const$H5Z_ENABLE_EDC) This function implements the HDF5-API function H5Pset_edc_check. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_SET_EDC_CHECK for details.

get_edc_check() This function implements the HDF5-API function H5Pget_edc_check. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_GET_EDC_CHECK for details.
set_hyper_vector_size(size = 2^10) This function implements the HDF5-API function H5Pset_hyper_vector_size. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_SET_HYPER_VECTOR_SIZE for details.

get_hyper_vector_size() This function implements the HDF5-API function H5Pget_hyper_vector_size. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_GET_HYPER_VECTOR_SIZE for details.

set_btree_ratios(left, middle, right) This function implements the HDF5-API function H5Pset_btree_ratios. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_SET_BTREE_RATIOS for details.

get_btree_ratios() This function implements the HDF5-API function H5Pget_btree_ratios. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_GET_BTREE_RATIOS for details.

Author(s)
Holger Hoefling

See Also
H5P

H5P_FILE_ACCESS-class  Class for HDF5 property list for file creation

Description
It inherits all functions of the H5P.

Value
Object of class H5P_FILE_ACCESS.

Methods
new(id = NULL) Create a new class of type H5P_FILE_ACCESS

Parameters
id  Internal use only

set_cache(rdcc_nslots = 521, rdcc_nbytes = 2^20, rdcc_w0 = 0.75) This function implements the HDF5-API function H5Pset_cache. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_SET_CACHE for details.

get_cache() This function implements the HDF5-API function H5Pget_cache. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_GET_CACHE for details.

Author(s)
Holger Hoefling
**H5P_FILE_CREATE-class**  
*Class for HDF5 property list for file creation*

**Description**

It inherits all functions of the H5P.

**Value**

Object of class H5P_FILE_CREATE.

**Methods**

- `new(id = NULL)` Create a new class of type H5P_FILE_CREATE
  
  **Parameters**
  
  id  Internal use only

- `set_userblock(size)` This function implements the HDF5-API function H5Pset_userblock. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_SET_USERBLOCK for details.

- `get_userblock()` This function implements the HDF5-API function H5Pget_userblock. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_GET_USERBLOCK for details.

- `set_sizes(sizeof_addr, sizeof_size)` This function implements the HDF5-API function H5Pset_sizes. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_SET_SIZES for details.

- `get_sizes()` This function implements the HDF5-API function H5Pget_sizes. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_GET_SIZES for details.

- `set_sym_k(ik, lk)` This function implements the HDF5-API function H5Pset_sym_k. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_SET_SYM_K for details.

- `get_sym_k()` This function implements the HDF5-API function H5Pget_sym_k. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_GET_SYM_K for details.

- `set_istore_k(ik)` This function implements the HDF5-API function H5Pset_istore_k. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_SET_ISTORE_K for details.

- `get_istore_k()` This function implements the HDF5-API function H5Pget_istore_k. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_GET_ISTORE_K for details.

**See Also**

H5P
set_file_space(strategy, threshold) This function implements the HDF5-API function H5Pset_file_space. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_SET_FILE_SPACE_STRATEGY for details.

get_file_space() This function implements the HDF5-API function H5Pget_file_space. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_GET_FILE_SPACE_STRATEGY for details.

Author(s)

Holger Hoefling

See Also

H5P

---

**H5P_LINK_ACCESS-class**  
*Class for HDF5 property list for link access*

---

**Description**

It inherits all functions of the H5P.

**Value**

Object of class H5P_LINK_ACCESS.

**Methods**

new(id = NULL) Create a new class of type H5P_LINK_ACCESS

**Parameters**

id Internal use only

set_nlinks(nlinks) This function implements the HDF5-API function H5Pset_nlinks. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_SET_NLINKS for details.

get_nlinks() This function implements the HDF5-API function H5Pget_nlinks. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_GET_NLINKS for details.

set_elink_prefix(elink_prefix) This function implements the HDF5-API function H5Pset_elink_prefix. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_SET_ELINK_PREFIX for details.

get_elink_prefix() This function implements the HDF5-API function H5Pget_elink_prefix. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_GET_ELINK_PREFIX for details.

set_elink_acc_flags(elink_acc_flags = h5const$H5F_ACC_RDWR) This function implements the HDF5-API function H5Pset_elink_acc_flags. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_SET_ELINK_ACC_FLAGS for details.
get_elink_acc_flags()  This function implements the HDF5-API function H5Pget_elink_acc_flags. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_GET_ELINK_ACC_FLAGS for details.

Author(s)
Holger Hoefling

See Also
H5P

---

H5P_LINK_CREATE-class  Class for HDF5 property list for link creation

Description
It inherits all functions of the H5P.

Value
Object of class H5P_LINK_CREATE.

Methods

new(id = NULL)  Create a new class of type H5P_LINK_CREATE

Parameters
  id  Internal use only

set_char_encoding(encoding = h5const$H5T_CSET_UTF8)  This function implements the HDF5-API function H5Pset_char_encoding. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_SET_CHAR_ENCODING for details.

get_char_encoding()  This function implements the HDF5-API function H5Pget_char_encoding. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_GET_CHAR_ENCODING for details.

set_create_intermediate_group(create = TRUE)  This function implements the HDF5-API function H5Pset_create_intermediate_group. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_SET_CREATE_INTERMEDIATE_GROUP for details.

get_create_intermediate_group()  This function implements the HDF5-API function H5Pget_create_intermediate_group. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_GET_CREATE_INTERMEDIATE_GROUP for details.

Author(s)
Holger Hoefling

See Also
H5P
**H5P_OBJECT_COPY-class**  
*Class for HDF5 property list for object copying*

**Description**

It inherits all functions of the H5P.

**Value**

Object of class H5P_OBJECT_COPY.

**Methods**

- `new(id = NULL)` Create a new class of type H5P_OBJECT_COPY
  
  **Parameters**
  
  id Internal use only

- `set_copy_obj(copy_options = 0)` This function implements the HDF5-API function H5Pset_copy_object. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5P_SET_COPY_OBJECT](https://portal.hdfgroup.org/display/HDF5/H5P_SET_COPY_OBJECT) for details.

- `get_copy_obj()` This function implements the HDF5-API function H5Pget_copy_object. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5P_GET_COPY_OBJECT](https://portal.hdfgroup.org/display/HDF5/H5P_GET_COPY_OBJECT) for details.

**Author(s)**

Holger Hoefling

**See Also**

H5P

---

**H5P_OBJECT_CREATE-class**  
*Class for HDF5 property list for object creation*

**Description**

It inherits all functions of the H5P.

**Value**

Object of class H5P_OBJECT_CREATE.
Methods

new(id = NULL) Create a new class of type H5P_OBJECT_CREATE

Parameters

id Internal use only

set_obj_track_times(track_times = TRUE) This function implements the HDF5-API function H5Pset_obj_track_times. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_SET_OBJ_TRACK_TIMES for details.

get_obj_track_times() This function implements the HDF5-API function H5Pget_obj_track_times. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_GET_OBJ_TRACK_TIMES for details.

set_attr_phase_change(max_compact, min_dense) This function implements the HDF5-API function H5Pset_attr_phase_change. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_SET_ATTR_PHASE_CHANGE for details.

get_attr_phase_change() This function implements the HDF5-API function H5Pget_attr_phase_change. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_GET_ATTR_PHASE_CHANGE for details.

set_attr_creation_order(crt_order_flags = 0) This function implements the HDF5-API function H5Pset_attr_creation_order. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_SET_ATTR_CREATION_ORDER for details.

get_attr_creation_order() This function implements the HDF5-API function H5Pget_attr_creation_order. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5P_GET_ATTR_CREATION_ORDER for details.

Author(s)

Holger Hoefling

See Also

H5P

---

H5R-class

Class for HDF5 Reference datatypes.

Description

H5R is only the common base class and is never used. User should not create objects of this class by themselves and instead use the create_reference methods of H5D, H5Group or H5File classes. Sub-classes are H5R_OBJECT and H5R_DATASET_REGION

Value

Object of class H5R.
Methods

subset_read(dim_index, drop = TRUE) Method that returns a subset of the data in the H5R-object

Parameters

  dim_index A list of dimension indices as usually pasted into [
  drop Logical. Should dimensions of size 1 be dropped.

subset2_read(i, exact = TRUE) Method to read a single item

Parameters

  i The single item to read
  exact Is the item name exact or should partial matching be allowed?

subset_assign(dim_index, value) Assign values into a subset of the H5R-vector

Parameters

  dim_index A list of dimension indices as usually passed into [
  value The value to assign

subset2_assign(i, exact = TRUE, value) Assign a value to a single value in the array

Parameters

  i the index where to assign the value
  value The value to assign

t() Transpose the object if it is a matrix (i.e. has rank 2

length() Get the length of the object

ref(ref) Get or assign the internal raw-vector representation of the data. Usually, user’s shouldn’t have to use this.

dim(x) Get or assign the dimensionality of the object

dimnames(x) Get or assign the dimnames of the object

names(x) Get or assign the names of the object

rank() Get the rank of the object

Author(s)

Holger Hoefling

Examples

fname <- tempfile(fileext = ".h5")
file <- H5File$new(fname, mode = "a")
file["testset"] <- matrix(rnorm(9), nrow = 3)
dset <- file["testset"]
r <- file$create_reference("testset")
file$close_all()
**H5RefClass-class**

*Base class that tracks the ids and allows for closing an id*

**Description**

This class is not intended for creating objects, but as a base class for all other H5-derived classes to provide common functionality for id tracking.

**Value**

Object of reference class `H5RefClass`.

**Fields**

- **id** Returns the id of the object as an integer

**Methods**

- **new(id = NULL)** Constructor for the basic class for hdf5 objects. Takes an id and stores it appropriately, including the necessary counting of object references that the package implements. This reference counting is included in addition to R’s internal method in order to allow for the invalidation of objects in R itself when all open handles in an R-file are being closed.
- **close()** Closes an object and calls the appropriate HDF5 function for the type of object
- **print(...)** Prints the class of the object and the id
- **methods()** Prints available methods on the screen
- **get_file_id()** This function implements the HDF5-API function H5Iget_file_id. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5I_GET_FILE_ID](https://portal.hdfgroup.org/display/HDF5/H5I_GET_FILE_ID) for details.
- **get_obj_type()** This function implements the HDF5-API function H5Iget_type. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5I_GET_TYPE](https://portal.hdfgroup.org/display/HDF5/H5I_GET_TYPE) for details.
- **get_ref()** This function implements the HDF5-API function H5Iget_ref. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5I_GET_REF](https://portal.hdfgroup.org/display/HDF5/H5I_GET_REF) for details.
- **inc_ref()** This function implements the HDF5-API function H5Iinc_ref. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5I_INC_REF](https://portal.hdfgroup.org/display/HDF5/H5I_INC_REF) for details.
- **dec_ref()** This function implements the HDF5-API function H5Idec_ref. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5I_DEC_REF](https://portal.hdfgroup.org/display/HDF5/H5I_DEC_REF) for details.
- **id()** Returns the id of the object
- **is_valid()** This function implements the HDF5-API function H5Iis_valid. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5I_IS_VALID](https://portal.hdfgroup.org/display/HDF5/H5I_IS_VALID) for details. Additionally, the R-object representing the HDF5-id can be invalidated as well. In this case, the class id is set to NA and *is_valid* returns FALSE.
- **message()** Legacy function; currently not used; may be removed
Author(s)

Holger Hoefling

H5R_DATASET_REGION-class

Class for HDF5 dataset-region references.

Description

H5R_DATASET_REGION is the reference class for dataset regions. Users should not create this class by themselves, but use the appropriate and instead use the create_reference methods of H5D, H5Group or H5File classes.

Value

Object of class H5R_DATASET_REGION.

Methods

new(num = 0, id = NULL) Create a new reference for dataset regions; Usually, users shouldn’t have to call this, but use the create_reference method of a dataset.

dereference(object_access_pl = h5const$H5P_DEFAULT, obj = NULL, get_value = FALSE) Dereference an H5R reference for a dataset region. The file the reference is pointing to is assigned automatically. It returns a list where each item is a list with components dataset, being an H5D object and space being a H5S object. When setting get_value=TRUE, then instead of these objects the data itself is returned. This function implements the HDF5-API function H5Rdereference. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5R_DEREFERENCE for details.

Parameters

obj Overriding the default file the reference is referring to

object_access_pl The object-access property list. Currently always the default

Author(s)

Holger Hoefling
H5R_functions

Various functions for H5R objects

Description

Various functions for H5R objects

Usage

is.H5R(x)

is.H5R_OBJECT(x)

is.H5R_DATASET_REGION(x)

## S3 method for class 'H5R'
names(x)

## S3 method for class 'H5R'
length(x)

## S3 method for class 'H5R'
x[i, j, ..., drop = TRUE]

## S3 replacement method for class 'H5R'
x[i, ...] <- value

## S3 method for class 'H5R'
c(..., recursive = FALSE)

## S3 method for class 'H5R'
dim(x)

## S3 replacement method for class 'H5R'
dim(x) <- value

## S3 method for class 'H5R'
t(x)

## S3 method for class 'H5R'
dimnames(x)

## S3 replacement method for class 'H5R'
dimnames(x) <- value

## S3 method for class 'H5R'
cbind(..., deparse.level = 1)
## S3 method for class 'H5R'
rbind(..., deparse.level = 1)

## S3 method for class 'H5R'
print(x, ...)

## S3 method for class 'H5R'
format(x, ...)

## S3 method for class 'H5R'
as.data.frame(x, row.names = NULL, optional = FALSE, ...

## S3 method for class 'H5R'
as.vector(x, mode = "any")

### Arguments

- **x**: Object of type H5R
- **i**: First dimension
- **j**: Second dimension
- **...**: Any other dimensions (for subsetting), or objects to concatenate (for c) or combine by row/col (for cbind or rbind) or ignored (for print and format)
- **drop**: Should dimensions of size 1 be dropped; LOGICAL
- **value**: The value in an assignment
- **recursive**: Ignored here
- **deparse.level**: integer controlling the construction of labels in the case of non-matrix-like arguments (for the default method): 'deparse.level = 0' constructs no labels; the default, 'deparse.level = 1' constructs labels from the argument names
- **row.names**: NULL or a character vector giving the row names for the data frame. Missing values are not allowed.
- **optional**: logical. If TRUE, setting row names and converting column names (to syntactic names: see make.names) is optional.
- **nm**: The column names to use
- **mode**: Only 'any' supported
- **width.cutoff**: ignored
- **collapse**: ignored

### Details

- **is.H5R**: Check if object inherits from H5R
- **is.H5R_OBJECT**: Check if object inherits from H5R_OBJECT
is.H5R_DATASET_REGION  Check if object inherits from H5R_DATASET_REGION
names.H5R     Returns the names of the elements of the vector
length.H5R    Returns the length of the vector
[.H5R    Array subsetting function
[<-.H5R  Array subset assignment
c.H5R     Concatenation of H5R vectors
dim.H5R    Dimensionality of the object
dim<-.H5R  Assign dimension of the object
t.H5R     Transpose a matrix of H5R objects
dimnames.H5R Get the dimnames of the object
dimnames<-.H5R Set the dimnames of the object
cbind.H5R  cbind functionality for H5R objects
rbind.H5R  rbind functionality for H5R objects
print.H5R  Printing of an object of class h5R
format.H5R Formatting of an H5R object
as.data.frame.H5R Coerce an H5R object to a data.frame
as.vector.H5R Coerce to a vector
as.data.frame.H5R Coerces the object to a data.frame
as.vector.H5R Coerces to a vector

Value
   Depending on the function

Author(s)
   Holger Hoefling

---

**H5R_OBJECT-class**  
*Class for HDF5 Object-references.*

**Description**

H5R_OBJECT is the reference class for objects. Users should not create this class by themselves, but use the appropriate and instead use the create_reference methods of H5D, H5Group or H5File classes.

**Value**

Object of class **H5R_OBJECT**.
Methods

new(num = 0, id = NULL) Create a new reference for object; Usually, users shouldn't have to call this, but use the create_reference method of a dataset, group of committed datatype
dereference(object_access_pl = h5const$H5P_DEFAULT, obj = NULL) Dereference an H5R reference. The file the reference is pointing to is assigned automatically This function implements the HDF5-API function H5Rdereference. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5R_DEREFERENCE for details.

Parameters

obj Overriding the default file the reference is referring to
object_access_pl The object-access property list. Currently always the default

Author(s)

Holger Hoefling

H5S-class

Class for representing HDF5 spaces

Description

This class represents Spaces in HDF5. These are mostly useful to define the dimensions of a dataset as well as the maximum dimensions to which it can grow. By default, the maximum dimension is equal to the initial dimension. If you want the array to be able to grow arbitrarily large in one dimension, set the maximum dimension for this index to Inf. See the examples below for code how to do this.

Value

Object of class H5S.

Methods

new(type = c("simple", "scalar", "null"), dims = NULL, maxdims = dims, decode_buf = NULL, id = NULL) Create a new HDF5-space. This can be done by either specifying a space with appropriate dimensions or by decoding a character string that represents an encoded space

Parameters

type Either a simple space, for which dims and maxdims have to be given or a scalar or null space. See the HDF5 user guide on spaces to explain the differences.
dims The dimension of the space in case it is of type simple
maxdims The maximal dimensions of the space
decode_buf The character string that holds the encoded representation of a space
id An existing HDF5 id; internal use only

copy() This function implements the HDF5-API function H5Scopy. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5S_COPY for details.
encode() This function implements the HDF5-API function H5Sencode. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5S_ENCODE for details.

is_simple() This function implements the HDF5-API function H5Sis_simple. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5S_IS_SIMPLE for details.

get_simple_extent_ndims() This function implements the HDF5-API function H5Sget_simple_extent_ndims. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5S_GET_SIMPLE_EXTENT_NDIMS for details.

offset_simple(offset) This function implements the HDF5-API function H5Soffset_simple. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5S_OFFSET_SIMPLE for details.

get_simple_extent_dims() This function implements the HDF5-API function H5Sget_simple_extent_dims. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5S_GET_SIMPLE_EXTENT_DIMS for details.

get_simple_extent_npoints() This function implements the HDF5-API function H5Sget_simple_extent_npoints. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5S_GET_SIMPLE_EXTENT_NPOINTS for details.

get_simple_extent_type() This function implements the HDF5-API function H5Sget_simple_extent_type. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5S_GET_SIMPLEExtent_TYPE for details.

extent_copy(h5s_source) This function implements the HDF5-API function H5Sextent_copy. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5S_EXTENT_COPY for details.

extent_equal(h5s_cmp) This function implements the HDF5-API function H5Sextent_equal. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5S_EXTENT_EQUAL for details.

set_extent_simple(dims, maxdims) This function implements the HDF5-API function H5Sset_extent_simple. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5S_SET_EXTENT_SIMPLE for details.

set_extent_none() This function implements the HDF5-API function H5Sset_extent_none. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5S_SET_EXTENT_NONE for details.

get_select_type() This function implements the HDF5-API function H5Sget_select_type. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5S_GET_SELECT_TYPE for details.

get_select_npoints() This function implements the HDF5-API function H5Sget_select_npoints. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5S_GET_SELECT_NPOINTS for details.

get_select_hyper_nblocks() This function implements the HDF5-API function H5Sget_select_hyper_nbblocks. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5S_GET_SELECT_HYPER_NBLOCKS for details.

get_select_hyper_blocklist(startblock = 0, numblocks = (self$get_select_hyper_nbblocks() - startblock)) This function implements the HDF5-API function H5Sget_select_hyper_blocklist. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5S_GET_SELECT_HYPER_BLOCKLIST for details.
get_select_elem_npoints() This function implements the HDF5-API function H5Sget_select_elem_npoints. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5S_GET_SELECT_ELEM_NPOINTS for details.

get_select_elem_pointlist(startpoint = 0, numpoints = (self$get_select_elem_npoints() - startpoint)) This function implements the HDF5-API function H5Sget_select_elem_pointlist. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5S_GET_SELECT_ELEM_POINTLIST for details.

get_select_bounds() This function implements the HDF5-API function H5Sget_select_bounds. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5S_SELECT_BOUNDS for details.

select_all() This function implements the HDF5-API function H5Sselect_all. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5S_SELECT_ALL for details.

select_none() This function implements the HDF5-API function H5Sselect_none. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5S_SELECT_NONE for details.

select_valid() This function implements the HDF5-API function H5Sselect_valid. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5S_SELECT_VALID for details.

select_elements(coord, op = h5const$H5S_SELECT_SET, byrow = TRUE) This function implements the HDF5-API function H5Sselect_elements. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5S_SELECT_ELEMENTS for details.

select_hyperslab(start, count, stride = NULL, block = NULL, op = h5const$H5S_SELECT_SET) This function implements the HDF5-API function H5Sselect_hyperslab. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5S_SELECT_HYPERSLAB for details.

subset(args, op = h5const$H5S_SELECT_SET, envir = parent.frame()) Subsetting the space. This is mainly intended as a helper function for the '[' function, but can also be used on its own.

Parameters

- **args** The indices for each dimension to subset given as a list. This makes this easier to use as a programmatic API. For interactive use we recommend the use of the '[' operator.
- **op** The operator to use. Same as for the other HDF5 space selection functions. One of the elements shown in h5const$H5S_seloper_t
- **envir** The environment in which to evaluate args

print(...) Prints information for the group

Parameters

- **...** ignored

dims() Get the dimensions of the space. Return NULL if the space is not simple (i.e. NULL-space) or a length-0 integer if it is a scalar

maxdims() Get the maximal dimensions of the space. Return NULL if the space is not simple (i.e. NULL-space) or a length-0 integer if it is a scalar

rank() This function implements the HDF5-API function H5Sget_simple_extent_ndims. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5S_GET_SIMPLE_EXTENT_NDIMS for details.
Author(s)
Holger Hoefling

Examples

```r
h5s_fixed <- H5S$new("simple", dims=c(5, 2))
h5s_fixed

h5s_variable <- H5S$new("simple", dims=c(5, 2), maxdims=c(Inf,2))
h5s_variable
h5s_variable$set_extent_simple(c(10,2), c(Inf, 2))
h5s_variable

# now select a subset of points
# argument evaluation has a heuristic; here it chooses point selection
h5s_variable[c(1, 3, 8), 1]
h5s_variable$get_select_type()
h5s_variable$get_select_elem_pointlist()

# and a hyperslab (chosen by the argument heuristic)
h5s_variable[2:7, 1:2]
h5s_variable$get_select_type()
h5s_variable$get_select_hyper_blocklist()
```

Description
Selecting and assigning subsets of HDF5-Spaces and HDF5-Datasets

Usage

```r
subset_h5.H5S(x, d1, ..., op = h5const$H5S_SELECT_SET, 
envir = parent.frame())
```

```r
## S3 method for class 'H5S'
x[d1, ..., op = h5const$H5S_SELECT_SET, 
envir = parent.frame()]
```

```r
subset_h5.H5D(x, d1, ..., dataset_xfer_pl = h5const$H5P_DEFAULT, 
flags = getOption("hdf5r.h5tor_default"), drop = TRUE, 
envir = parent.frame())
```

```r
## S3 method for class 'H5D'
x[d1, ..., dataset_xfer_pl = h5const$H5P_DEFAULT, 
flags = getOption("hdf5r.h5tor_default"), drop = TRUE, 
envir = parent.frame()]
```
subset_assign_h5.H5D(x, d1, ..., dataset_xfer_pl = h5const$H5P_DEFAULT,
envir = parent.frame(), value)

## S3 replacement method for class 'H5D'
x[d1, ..., dataset_xfer_pl = h5const$H5P_DEFAULT,
envir = parent.frame()] <- value

Arguments

- **x**: The **H5S** or **H5D** to subset or assign values to.
- **d1**: First dimension of the object.
- **...**: Used for other dimension of the object.
- **envir**: The environment in which the dimension indices d1,... are to be evaluated. Usually the environment from where the function is called.
- **dataset_xfer_pl**: An object of class **H5P_DATASET_XFER**.
- **flags**: Some flags governing edge cases of conversion from HDF5 to R. This is related to how integers are being treated and the issue of R not being able to natively represent 64bit integers and not at all being able to represent unsigned 64bit integers (even using add-on packages). The constants governing this are part of **h5const**. The relevant ones start with the term **H5TOR** and are documented there. The default set here returns a regular 32bit integer if it doesn’t lead to an overflow and returns a 64bit integer from the **bit64** package otherwise. For 64bit unsigned int that are larger than 64bit signed int, it return a double. This looses precision, however. See also documentation or **h5const**.
- **drop**: Logical. When reading data, should dimensions of size 1 be dropped.
- **value**: The value to assign to the dataset.

Details

Used for subsetting HDF5-Datasets or HDF5-Spaces or for assigning data into HDF5-Datasets. There are some differences to consider with R itself.

Most importantly HDF5-COMPOUND objects only have a single dimension internally to HDF5 (a vector), but they correspond to R-data.frames, which are 2 dimensional. For an HDF5 COMPOUND object, it is currently not possible to only sub-select a specific column. All columns have to be extracted (using 1-dimensional access with [] and can then be subset in R itself. The same is true for writing a COMPOUND object (**H5T_COMPOUND**). A complete data-frame is needed, not just a subset of the columns.

Another important differences is for datasets of HDF5-ARRAY type **H5T_ARRAY** where the access to the object is only for the dimension of the object itself, not including the dimensions of the underlying array type.
Value

For \( x \) being a `H5S`, the same object is returned, but with the selection set as requested. For `H5D` it retrieves the subset of data requested or sets the subset of data assigned, as for any n-dimensional array in R.

Author(s)

Holger Hoefling

---

**H5T-class**

*Class for HDF5 datatypes.*

---

**Description**

This is the base class for all datatypes, but most have a specialised class. This class represents an HDF5 datatype. It inherits all functions of the `H5RefClass`. It is also the base class for many other classes well, specifically

- **Integer** `H5T_INTEGER`
- **Bitfield** `H5T_BITFIELD` (currently identical to the integer class)
- **Float** `H5T_FLOAT`
- **Enum** `H5T_ENUM`
- **Compound** `H5T_COMPOUND`
- **String** `H5T_STRING`
- **Complex** `H5T_COMPLEX`
- **Array** `H5T_ARRAY`
- **Variable Length** `H5T_VLEN`

**Value**

Object of class `H5T`.

**Methods**

- `new(id)` Internal use only
- `get_class()` This function implements the HDF5-API function H5Tget_class. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5T_GET_CLASS](https://portal.hdfgroup.org/display/HDF5/H5T_GET_CLASS) for details.
- `get_size(...)` This function implements the HDF5-API function H5Tget_size. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5T_GET_SIZE](https://portal.hdfgroup.org/display/HDF5/H5T_GET_SIZE) for details.

**Parameters**

- ... ignored
set_size(size) This function implements the HDF5-API function H5Tset_size. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5T_SET_SIZE for details.

set_precision(precision) This function implements the HDF5-API function H5Tset_precision. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5T_SET_PRECISION for details.

get_precision() This function implements the HDF5-API function H5Tget_precision. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5T_GET_PRECISION for details.

set_order(order) This function implements the HDF5-API function H5Tset_order. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5T_SET_ORDER for details.

get_order() This function implements the HDF5-API function H5Tget_order. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5T_GET_ORDER for details.

set_offset(offset) This function implements the HDF5-API function H5Tset_offset. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5T_SET_OFFSET for details.

get_offset() This function implements the HDF5-API function H5Tget_offset. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5T_GET_OFFSET for details.

set_pad(pad) This function implements the HDF5-API function H5Tset_pad. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5T_SET_PAD for details.

get_pad() This function implements the HDF5-API function H5Tget_pad. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5T_GET_PAD for details.

copy() This function implements the HDF5-API function H5Tcopy. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5T_COPY for details.

is_committed() This function implements the HDF5-API function H5Tcommitted. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5T_COMMITTED for details.

equal(dtype) This function implements the HDF5-API function H5Tequal. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5T_EQUAL for details.

is_vlen() This function detects if the underlying type is H5T_VLEN or a variable length string. This is used to know if after reading a dataset, memory has to be freed.

detect_class(dtype_class) This function implements the HDF5-API function H5Tdetect_class. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5T_DETECT_CLASS for details.

get_native_type(direction = h5const$H5T_DIR_ASCEND) This function implements the HDF5-API function H5Tget_native_type. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5T_GET_NATIVE_TYPE for details.

get_create_plist() This function implements the HDF5-API function H5Tget_create_plist. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5T_GET_CREATE_PLIST for details.

to_text(lang_type = h5const$H5LT_DDL) This function implements the HDF5-API function H5LTdtype_to_text. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5LT_DTYPE_TO_TEXT for details.
print(...) Prints information for the group

**Parameters**

... ignored

**obj_info**(`remove_internal_use_only` = `TRUE`) This function implements the HDF5-API function `H5Oget_info`. Please see the documentation at `https://portal.hdfgroup.org/display/HDF5/H5O_GET_INFO` for details.

**get_obj_name()** This function implements the HDF5-API function `H5Iget_name`. Please see the documentation at `https://portal.hdfgroup.org/display/HDF5/H5I_GET_NAME` for details.

**create_attr**(`attr_name`, `robj` = `NULL`, `dtype` = `NULL`, `space` = `NULL`) This function implements the HDF5-API function `H5Acreate2`. Please see the documentation at `https://portal.hdfgroup.org/display/HDF5/H5A_CREATE2` for details.

**attr_open**(`attr_name`) This function implements the HDF5-API function `H5Aopen`. Please see the documentation at `https://portal.hdfgroup.org/display/HDF5/H5A_OPEN` for details.

**create_attr_by_name**(`attr_name`, `obj_name`, `robj` = `NULL`, `dtype` = `NULL`, `space` = `NULL`, `link_access_pl` = `h5const$H5P_DEFAULT`) This function implements the HDF5-API function `H5Acreate_by_name`. Please see the documentation at `https://portal.hdfgroup.org/display/HDF5/H5A_CREATE_BY_NAME` for details.

**attr_open_by_name**(`attr_name`, `obj_name`, `link_access_pl` = `h5const$H5P_DEFAULT`) This function implements the HDF5-API function `H5Aopen_by_name`. Please see the documentation at `https://portal.hdfgroup.org/display/HDF5/H5A_OPEN_BY_NAME` for details.

**attr_open_by_idx**(`n`, `obj_name`, `idx_type` = `h5const$H5_INDEX_NAME`, `order` = `h5const$H5_ITER_NATIVE`, `link_access_pl` = `h5const$H5P_DEFAULT`) This function implements the HDF5-API function `H5Aopen_by_idx`. Please see the documentation at `https://portal.hdfgroup.org/display/HDF5/H5A_OPEN_BY_IDX` for details.

**attr_exists_by_name**(`attr_name`, `obj_name`, `link_access_pl` = `h5const$H5P_DEFAULT`) This function implements the HDF5-API function `H5Aexists_by_name`. Please see the documentation at `https://portal.hdfgroup.org/display/HDF5/H5A_EXISTS_BY_NAME` for details.

**attr_exists**(`attr_name`) This function implements the HDF5-API function `H5Aexists`. Please see the documentation at `https://portal.hdfgroup.org/display/HDF5/H5A_EXISTS` for details.

**attr_rename_by_name**(`old_attr_name`, `new_attr_name`, `obj_name`, `link_access_pl` = `h5const$H5P_DEFAULT`) This function implements the HDF5-API function `H5Arename_by_name`. Please see the documentation at `https://portal.hdfgroup.org/display/HDF5/H5A_RENAME_BY_NAME` for details.

**attr_rename**(`old_attr_name`, `new_attr_name`) This function implements the HDF5-API function `H5Arename`. Please see the documentation at `https://portal.hdfgroup.org/display/HDF5/H5A_RENAME` for details.

**attr_delete**(`attr_name`) This function implements the HDF5-API function `H5Adelete`. Please see the documentation at `https://portal.hdfgroup.org/display/HDF5/H5A_DELETE` for details.

**attr_delete_by_name**(`attr_name`, `obj_name`, `link_access_pl` = `h5const$H5P_DEFAULT`) This function implements the HDF5-API function `H5Adelete_by_name`. Please see the documentation at `https://portal.hdfgroup.org/display/HDF5/H5A_DELETE_BY_NAME` for details.
attr_delete_by_idx(n, obj_name, idx_type = h5const$H5_INDEX_NAME, order = h5const$H5_ITER_NATIVE, link_access_pl = h5const$H5P_DEFAULT)
This function implements the HDF5-API function H5Adelete_by_idx. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_DELETE_BY_IDX for details.

attr_info_by_name(attr_name, obj_name, link_access_pl = h5const$H5P_DEFAULT)
This function implements the HDF5-API function H5Aget_info_by_name. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_GET_INFO_BY_NAME for details.

attr_info_by_idx(n, obj_name, idx_type = h5const$H5_INDEX_NAME, order = h5const$H5_ITER_NATIVE, link_access_pl = h5const$H5P_DEFAULT)
This function implements the HDF5-API function H5Aget_info_by_idx. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_GET_INFO_BY_IDX for details.

attr_name_by_idx(n, obj_name, idx_type = h5const$H5_INDEX_NAME, order = h5const$H5_ITER_NATIVE, link_access_pl = h5const$H5P_DEFAULT)
This function implements the HDF5-API function H5Aget_name_by_idx. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5A_GET_NAME_BY_IDX for details.

attr_get_number()
This function implements the HDF5-API function H5Aget_num_attrs. Please see the documentation at https://support.hdfgroup.org/HDF5/doc/RM/RM_H5A.html#Annot-NumAttrs for details.

create_reference(name = ".", space = NULL)
This function implements the HDF5-API function H5Rcreate. If space=NULL then a H5R_OBJECT reference is created, otherwise a H5R_DATASET_REGION reference. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5R_CREATE for details.

Author(s)
Holger Hoefling

Examples

my_int <- h5types$H5T_NATIVE_INT
my_int$to_text()
my_int$get_size()

# Show how to commit a datatype
fname <- tempfile(fileext = ".h5")
file <- H5File$new(fname, mode = "a")
my_int$is_committed()
file$commit("my_int", my_int)
my_int$is_committed()

# can now also add attributes
h5attr(my_int, "test") <- "A string"
h5attributes(my_int)

file$close_all()
file.remove(fname)
h5types

These are all types that are used in HDF5

Description

HDF5 provides many native datatypes. These are all stored in the h5types environment. An overview of all available types can be seen using h5types$overview. Any specific type can be accessed using the $-operator. See also the examples below.

Author(s)

Holger Hoefling

Examples

h5types$overview
h5types$H5T_NATIVE_INT
h5types$H5T_NATIVE_DOUBLE

H5T_ARRAY-class

Class for HDF5 array datatypes.

Description

Inherits from class H5T. This class represents an array. As datasets in HDF5 are itself already arrays, this datatype is not needed there. It is mostly useful when a column in a H5T_COMPUND object is intended to be an array. This however makes it difficult to work with such objects in R - as a column of the corresponding data.frame has to be an array. So please use with care.

Value

Object of class H5T_ARRAY.

Methods

new(dims, dtype_base, id = NULL) Create an array datatype.

Parameters

dims The dimension of the datatype
dtype_base The datatype that makes up the elements of the arrayid internal use only

get_array_ndims() This function implements the HDF5-API function H5Tget_array_ndims. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5T_GET_ARRAY_NDIMS for details.
get_array_dims() This function implements the HDF5-API function H5Tget_array_dims2. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5T_GET_ARRAY_DIMS2 for details.

get_super() This function implements the HDF5-API function H5Tget_super. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5T_GET_SUPER for details.

describe() Print a detailed description of the datatype; this is experimental

Author(s)
Holger Hoefling

See Also
H5T

H5T_COMPLEX-class  Class for HDF5 complex datatypes

Description
In HDF5, complex numbers don’t actually exist. They are represented as H5T_COMPOUND with two columns named Real and Imaginary. Inherits from class H5T_COMPOUND.

Value
Object of class H5T_COMPLEX.

Methods
new(id = NULL) Create a new complex datatype

   Parameters
   id   Internal use only

Author(s)
Holger Hoefling

See Also
H5T, H5T_COMPOUND
**H5T_COMPOUND-class**  
Class for HDF5 compound datatypes.

**Description**

Inherits from class `H5T`.

**Value**

Object of class `H5T_COMPOUND`.

**Methods**

`new(labels, dtypes, size = NULL, offset = NULL, id = NULL)` Create a compound type that is the HDF5 equivalent of a table

- **Parameters**
  - `labels` The labels of the columns of the compound object
  - `dtypes` The datatypes of the columns of the object; this is usually a list of objects of class `H5T`
  - `size` The size of each datatype; if `NULL`, automatically inferred
  - `offset` The offset where each datatype starts; can be different from the sum of the individual sizes so that datatypes are aligned with memory addresses. If `NULL`, inferred automatically
  - `id` Internal use only

`pack()` This function implements the HDF5-API function H5Tpack. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5T_PACK for details.

`get_cpd_types()` Return `H5T` objects that represent the datatypes of the columns of the compound object. Returned as a list if more than 1

`get_cpd_labels()` Return the labels of the columns as a character vector

`get_cpd_classes()` Return the classes of the columns as an object of type `factor_ext`

`get_cpd_offsets()` Return the offsets of the datatypes

`describe()` Print a detailed description of the datatype; this is experimental

**Author(s)**

Holger Hoefling

**See Also**

`H5T`

**Examples**

```r
# create a H5T_COMPOUND corresponding to a data-frame
my_cpd <- H5T_COMPOUND$new(c("name", "age", "salary"),
  dtypes=list(H5T_STRING$new(size=200), h5types$H5T_NATIVE_INT, h5types$H5T_NATIVE_DOUBLE))
my_cpd
```
Class for HDF5 enumeration datatypes.

**Description**

Inherits from class `H5T`.

**Value**

Object of class `H5T_ENUM`.

**Methods**

new(labels, values = seq_along(labels), id = NULL) Create an enumeration datatype. This is either a factor-like object or a logical variable (that is internally represented as an ENUM-type.

**Parameters**

- `labels` The labels of the ENUM-type
- `values` The values corresponding to the labels
- `id` Internal use only

get_labels() Return all the labels of the enumeration type

get_values() Return the values of the enumeration type

set_size(size) Base type of every enum is H5T_INTEGER. This disables the set_size function

get_super() Returns H5T_INTEGER that is the base type of the enumeration

describe() Print a detailed description of the datatype; this is experimental

**Author(s)**

Holger Hoefling

**See Also**

`H5T`

**Examples**

nucleotide_enum <- H5T_ENUM$new(labels=c("A", "C", "G", "T"), values=0:3)
nucleotide_enum

# For HDF5 1.8.16 or higher, the size and precision are set optimally
nucleotide_enum$get_size()
nucleotide_enum$get_precision()
**H5T_FLOAT-class**  
*Class for HDF5 floating point datatypes.*

**Description**

Inherits from class `H5T`. Users should not create float types with this class, but instead use e.g. `h5types$H5T_NATIVE_DOUBLE`. Using the functions of this class, many aspects of the representation of the floating point number can then be manipulated.

**Value**

Object of class `H5T_FLOAT`.

**Methods**

- `set_fields(spos, epos, esize, mpos, msize)` This function implements the HDF5-API function `H5Tset_fields`. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5T_SET_FIELDS](https://portal.hdfgroup.org/display/HDF5/H5T_SET_FIELDS) for details.
- `get_fields()` This function implements the HDF5-API function `H5Tget_fields`. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5T_GET_FIELDS](https://portal.hdfgroup.org/display/HDF5/H5T_GET_FIELDS) for details.
- `set_ebias(ebias)` This function implements the HDF5-API function `H5Tset_ebias`. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5T_SET_EBIAS](https://portal.hdfgroup.org/display/HDF5/H5T_SET_EBIAS) for details.
- `get_ebias()` This function implements the HDF5-API function `H5Tget_ebias`. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5T_GET_EBIAS](https://portal.hdfgroup.org/display/HDF5/H5T_GET_EBIAS) for details.
- `set_norm(norm)` This function implements the HDF5-API function `H5Tset_norm`. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5T_SET_NORM](https://portal.hdfgroup.org/display/HDF5/H5T_SET_NORM) for details.
- `get_norm()` This function implements the HDF5-API function `H5Tget_norm`. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5T_GET_NORM](https://portal.hdfgroup.org/display/HDF5/H5T_GET_NORM) for details.
- `set_inpad(inpad)` This function implements the HDF5-API function `H5Tset_inpad`. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5T_SET_INPAD](https://portal.hdfgroup.org/display/HDF5/H5T_SET_INPAD) for details.
- `get_inpad()` This function implements the HDF5-API function `H5Tget_inpad`. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5T_GET_INPAD](https://portal.hdfgroup.org/display/HDF5/H5T_GET_INPAD) for details.
- `describe()` Print a detailed description of the datatype; this is experimental

**Author(s)**

Holger Hoefling

**See Also**

`H5T`
**H5T_INTEGER-class**

Class for HDF5 integer-datatypes.

---

**Description**

Inherits from class `H5T`. Users should not create integer datatypes themselves using this class. Instead, integer should be derived from one of the base-types such as `h5types$H5T_NATIVE_INT` (which internally automatically creates a copy of the type). For a complete list of types see `h5types$overview`.

**Value**

Object of class `H5T_INTEGER`.

**Methods**

- `set_sign(sign)` This function implements the HDF5-API function `H5Tset_sign`. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5T_SET_SIGN](https://portal.hdfgroup.org/display/HDF5/H5T_SET_SIGN) for details.
- `get_sign()` This function implements the HDF5-API function `H5Tget_sign`. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5T_GET_SIGN](https://portal.hdfgroup.org/display/HDF5/H5T_GET_SIGN) for details.
- `describe()` Return a vector that describes the key features of the datatype

**Author(s)**

Holger Hoefling

**See Also**

`H5T`

**Examples**

```r
my_int <- h5types$H5T_NATIVE_INT
# make an int with 2 bit
my_int$set_sign(h5const$H5T_SGN_NONE)
my_int$set_size(1)
my_int$set_precision(2)
my_int$describe()
```
Description

Inherits from class `H5T`.

Value

Object of class `H5T_LOGICAL`.

Methods

`new(include_NA = TRUE, id = NULL)` Create a logical datatype. This is internally represented by an ENUM-type

Parameters

`id` Internal use only

Author(s)

Holger Hoefling

See Also

`H5T, H5T_ENUM`

---

Description

Inherits from class `H5T`.

Value

Object of class `H5T_STRING`.

---
**H5T_STRING-class**

**Methods**

new(type = c("c", "fortran"), size = 1, id = NULL) Create a string datatype

**Parameters**

- **type** C or fortran type string
- **size** Size of the string object. Set to Inf for variable size strings
- **id** internal use only

get_size(variable_as_inf = TRUE) Retrieves the length of the string, setting it to Inf if it is of variable length. This function implements the HDF5-API function H5Tis_variable_str. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5T_IS_VARIABLE_STR for details.

get_cset() This function implements the HDF5-API function H5Tget_cset. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5T_GET_CSET for details.

set_cset(cset = c("unknown", "UTF-8")) This function implements the HDF5-API function H5Tset_cset. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5T_SET_CSET for details.

set_strpad(strpad) This function implements the HDF5-API function H5Tset_strpad. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5T_SET_STRPAD for details.

get_strpad() This function implements the HDF5-API function H5Tget_strpad. Please see the documentation at https://portal.hdfgroup.org/display/HDF5/H5T_GET_STRPAD for details.

describe() Print a detailed description of the datatype; this is experimental

**Author(s)**

Holger Hoefling

**See Also**

H5T

**Examples**

```r
# fixed width string
str_flen <- H5T_STRING$new(size=100)
str_flen$is_vlen()
str_flen

# variable length string
str_vlen <- H5T_STRING$new(size=Inf)
str_vlen$is_vlen()
str_vlen
```
**H5T_VLEN-class**

*Class for HDF5 variable-length datatypes.*

**Description**

Inherits from class `H5T`. This can make any datatype a variable length datatype. This would mostly be intended for storing ragged arrays.

**Value**

Object of class `H5T_VLEN`.

**Methods**

- `new(dtype_base, id = NULL)` Create a variable length datatype
  - **Parameters**
    - `dtype_base` The basis-type of the variable length datatype
    - `id` Internal use only
  - `get_super()` This function implements the HDF5-API function `H5Tget_super`. Please see the documentation at [https://portal.hdfgroup.org/display/HDF5/H5T_GET_SUPER](https://portal.hdfgroup.org/display/HDF5/H5T_GET_SUPER) for details.
  - `describe()` Print a detailed description of the datatype; this is experimental

**Author(s)**

Holger Hoefling

**See Also**

`H5T`

**Examples**

```r
vlen_int <- H5T_VLEN$new(h5types$H5T_NATIVE_INT)
vlen_int
```
**h5version**

*Return the version of the HDF5-API*

**Description**

Return the version of the HDF5-API

**Usage**

```
h5version(verbose = TRUE)
```

**Arguments**

- **verbose**
  - Should the information be printed to the screen as well

**Details**

Return the version of the HDF5-API and print it to the screen if requested

**Value**

Version of the underlying HDF5 API as a string

**Author(s)**

Holger Hoefling

---

**is_hdf5**

*Check if a file is an HDF5 file*

**Description**

Check if a file is an HDF5 file

**Usage**

```
is_hdf5(name)
```

**Arguments**

- **name**
  - The name of the file to check

**Details**

Uses the HDF5 function `H5Fis_hdf5` to check if a file is of type HDF5.
list-groups-datasets

Value

Logical, TRUE if file is of type HDF5

Author(s)

Holger Hoefling

list-groups-datasets  List Groups and Datasets in object

Description

List all Group (H5Group) and Dataset (H5D) names in the current object. This function is part of the h5 wrapper classes and uses $ls() to retrieve group names.

Usage

list.groups(object, path = "/", full.names = FALSE, recursive = TRUE, ...)
list.datasets(object, path = "/", full.names = FALSE, recursive = TRUE, ...)
list.objects(object, obj_type = c("H5I_GROUP", "H5I_DATASET", "H5I_DATATYPE"), path = "/", full.names = FALSE, recursive = TRUE, ...)

Arguments

object  CommonFG; Object implementing the CommonFG Interface (e.g. H5File, H5Group).
path  character; Path named to be used for iteration.
full.names  character; Specify if absolute DataSet path names should be returned.
recursive  logical; Specify if object should be traversed recursively.
...  Additional Parameters passed to $ls()
obj_type  character; Object type to be returned.

Value

character
names.H5Group

Get the names of the items in the group or at the / root of the file

Description
Get the names of the items in the group or at the / root of the file

Usage
## S3 method for class 'H5Group'
names(x)

## S3 method for class 'H5File'
names(x)

Arguments
x An object of class H5File or H5Group

Details
Works similar to the regular names function for a list. The names of the items of either a H5File at the root or a H5Group are returned as a character vector. The items are then accessed, again similar to a list, using [[].

Value
A character vector with the names of the items in the group/file.

Author(s)
Holger Hoefling

print.data.frame_ext
Print a data frame with extended factor objects

Description
Print a data frame that includes extended factor objects

Usage
## S3 method for class 'data.frame_ext'
print(x, ...)
Arguments

x The data.frame_ext object to print; Is returned by ls from H5File and H5Group and this function allows for petter printing of factor_ext so that the label instead of the value is printed.

... Parameters to be passed on directly to print.data.frame

Details

The regular print function for data-frames has special methods built-in for factors so that the label is printed instead of the constant. This function is intended to provide the same functionality for data frames with extended factors, by adding the class data.frame_ext to the class vector.

Value

The object to print itself, invisibly

Author(s)

Holger Hoefling

text_to_dtype

Convert a text description to a datatype

Description

Convert a text description to a datatype

Usage

text_to_dtype(text, lang_type = h5const$H5LT_DDL)

Arguments

text The text to convert to the datatype

lang_type The type of language to use; currently only H5LT_DDL is supported.

Details

Converts a text to a datatype using the HDF5 function H5LT_text_to_dtype. Documentation can be found at https://portal.hdfgroup.org/display/HDF5/H5LT_TEXT_TO_DTYPE.

Value

A datatype corresponding to the text with the appropriate class inheriting from H5T.

Author(s)

Holger Hoefling
$.types_env

Retrieving a copy of a type

Description

Retrieving a copy of a type

Usage

```r
## S3 method for class 'types_env'
x$name

## S3 method for class 'types_env'
x[[name]]
```

Arguments

- `x` The environment to request it from
- `name` The name of the type that is requested

Details

The types are stored in the environment `h5types`. These types should not be accessed directly. Therefor, the `$`-operator is overloaded to ensure that every type that is accessed is a copy of the original type.

Value

Returns an object that is a copy of a type that was requested

Author(s)

Holger Hoefling
Index

!=.factor_ext (factor_ext_functions), 6
==.factor_ext (factor_ext_functions), 6
[.H5D (H5S_H5D_subset_assign), 58
[.H5R (H5R_functions), 52
[.H5S (H5S_H5D_subset_assign), 58
[.factor_ext (factor_ext_functions), 6
[<-.H5D (H5S_H5D_subset_assign), 58
[<-.H5R (H5R_functions), 52
[<-.factor_ext (factor_ext_functions), 6
[[.H5File (H5Group_access), 35
[[.H5Group (H5Group_access), 35
[[.factor_ext (factor_ext_functions), 6
[[<-.H5File (H5Group_access), 35
[[<-.H5Group (H5Group_access), 35
[[<-.factor_ext (factor_ext_functions), 6
$.types_env, 77
as.character.factor_ext
  (factor_ext_functions), 6
as.data.frame.H5R (H5R_functions), 52
as.vector.H5R (H5R_functions), 52
as.hex, 4
c.factor_ext (factor_ext_functions), 6
c.H5D (h5-wrapper), 11
c.H5R (H5R_functions), 52
cbind.H5D (h5-wrapper), 11
cbind.H5R (H5R_functions), 52
character, 74
c coerce_to_factor
  (factor_ext_functions), 6
c coercible_to_factor
  (factor_ext_functions), 6
create_empty, 5
createDataSet (h5-wrapper), 11
createGroup (h5-wrapper), 11
data.frame, 9
dim.H5R (H5R_functions), 52
dim<-.H5R (H5R_functions), 52
dimnames.H5R (H5R_functions), 52
dimnames<-.H5R (H5R_functions), 52
existsGroup (h5-wrapper), 11
extendDataSet (h5-wrapper), 11
factor_ext, 5, 9, 66, 76
factor_ext_functions, 6
flatten_df, 8
form, 4
format.H5R (H5R_functions), 52
guess_chunks, 9, 25, 32
guess_dim (guess_nelem), 10
guess_dtype (guess_nelem), 10
guess_nelem, 10
guess_space, 11
h5 (h5-wrapper), 11
h5-wrapper, 11
H5A, 13, 23
H5A (H5A-class), 13
H5A-class, 13
h5attr (h5attributes), 15
h5attr<-(h5attributes), 15
h5attr_names, 29
h5attr_names (h5attributes), 15
h5attributes, 13, 15
h5close (h5-wrapper), 11
h5const, 14, 16, 19, 59
H5D, 13, 15, 18, 23, 36, 59, 60, 74
H5D (H5D-class), 17
H5D-class, 17
H5File, 12, 13, 15, 17, 18, 23, 29, 36, 74–76
H5File (H5File-class), 23
h5file, 29
h5file (h5-wrapper), 11
H5File-class, 23
INDEX

h5flush (h5-wrap), 11
h5garbage_collect, 30
H5Group, 12, 13, 15, 17, 18, 23, 29, 30, 36, 74–76
H5Group (H5Group-class), 30
H5Group-class, 30
H5Group_access, 35
H5P, 37–40, 42–48
H5P (H5P-class), 37
H5P-class, 37
H5P_ATTRIBUTE_CREATE, 37, 38
H5P_ATTRIBUTE_CREATE
  (H5P_ATTRIBUTE_CREATE-class), 38
H5P_ATTRIBUTE_CREATE-class, 38
H5P_CLASS, 39
H5P_CLASS (H5P_CLASS-class), 39
H5P Class, 39
H5P_DATASET_ACCESS, 26, 32, 36, 37, 39
H5P_DATASET_ACCESS
  (H5P_DATASET_ACCESS-class), 39
H5P_DATASET_ACCESS-class, 39
H5P_DATASET_CREATE, 26, 32, 37, 40
H5P_DATASET_CREATE
  (H5P_DATASET_CREATE-class), 40
H5P_DATASET_CREATE-class, 40
H5P_DATASET_XFER, 19, 20, 37, 42, 59
H5P_DATASET_XFER
  (H5P_DATASET_XFER-class), 42
H5P_DATASET_XFER-class, 42
H5P_FILE_ACCESS, 43
H5P_FILE_ACCESS
  (H5P_FILE_ACCESS-class), 43
H5P_FILE_ACCESS-class, 43
H5P_FILE_CREATE, 44
H5P_FILE_CREATE
  (H5P_FILE_CREATE-class), 44
H5P_FILE_CREATE-class, 44
H5P_LINK_ACCESS, 36, 37, 43
H5P_LINK_ACCESS
  (H5P_LINK_ACCESS-class), 45
H5P_LINK_ACCESS-class, 45
H5P_LINK_CREATE, 26, 32, 37, 46
H5P_LINK_CREATE
  (H5P_LINK_CREATE-class), 46
H5P_LINK_CREATE-class, 46
H5P_OBJECT_COPY, 37, 47
H5P_OBJECT_COPY
  (H5P_OBJECT_COPY-class), 47
H5P_OBJECT_COPY-class, 47
H5P_OBJECT_CREATE, 37, 47, 48
H5P_OBJECT_CREATE
  (H5P_OBJECT_CREATE-class), 47
H5P_OBJECT_CREATE-class, 47
H5R, 48
H5R (H5R-class), 48
H5R-class, 48
H5R_DATASET_REGION, 48, 51
H5R_DATASET_REGION
  (H5R_DATASET_REGION-class), 51
H5R_DATASET_REGION-class, 51
H5R functions, 52
H5R_OBJECT, 48, 54
H5R_OBJECT (H5R_OBJECT-class), 54
H5R_OBJECT-class, 54
H5RefClass, 23, 37, 39, 50, 60
H5RefClass (H5RefClass-class), 50
H5RefClass-class, 50
H5S, 11, 17–19, 55, 59, 60
H5S (H5S-class), 55
H5S-class, 55
H5S_H5D_subset_assign, 17, 58
H5T, 10, 11, 13, 15, 17, 36, 60, 64–72, 76
H5T (H5T-class), 60
H5T-class, 60
H5T_ARRAY, 10, 59, 60, 64
H5T_ARRAY (H5T_ARRAY-class), 64
H5T_ARRAY-class, 64
H5T_BITFIELD, 60
H5T_BITFIELD (H5T_INTEGER-class), 69
H5T_COMPLEX, 60, 65
H5T_COMPLEX (H5T_COMPLEX-class), 65
H5T_COMPLEX-class, 65
H5T_COMPOUND, 59, 60, 65, 66
H5T_COMPOUND (H5T_COMPOUND-class), 66
H5T_COMPOUND-class, 66
H5T_ENUM, 60, 67, 70
H5T_ENUM (H5T_ENUM-class), 67
H5T_ENUM-class, 67
H5T_FLOAT, 60, 68
H5T_FLOAT (H5T_FLOAT-class), 68
H5T_FLOAT-class, 68
H5T_INTEGER, 60, 67, 69
H5T_INTEGER (H5T_INTEGER-class), 69
H5T_INTEGER-class, 69
H5T_LOGICAL, 70
H5T_LOGICAL (H5T_LOGICAL-class), 70
H5T_LOGICAL-class, 70
H5T_STRING, 60, 70
H5T_STRING (H5T_STRING-class), 70
H5T_STRING-class, 70
H5T_VLEN, 60, 72
H5T_VLEN (H5T_VLEN-class), 72
H5T_VLEN-class, 72
H5ToR_Post, 6
h5types, 64, 77
h5sunlink (h5-wrapper), 11
h5version, 73
hdf5r-package, 3

is.factor_ext (factor_ext_functions), 6
is.h5file (h5-wrapper), 11
is.H5R (H5R_functions), 52
is.H5R_DATASET_REGION (H5R_functions), 52
is.H5R_OBJECT (H5R_functions), 52
is_hdf5, 73

length.H5R (H5R_functions), 52
list-groups-datasets, 74
list.attributes, 29
list.attributes (h5-wrapper), 11
list.datasets, 29
list.datasets (list-groups-datasets), 74
list.groups, 29
list.groups (list-groups-datasets), 74
list.objects (list-groups-datasets), 74

names, 29
names.H5File (names.H5Group), 75
names.H5Group, 75
names.H5R (H5R_functions), 52

openGroup (h5-wrapper), 11
openLocation (h5-wrapper), 11

print.data.frame, 76
print.data.frame_ext, 75
print.factor_ext
(factor_ext_functions), 6
print.H5R (H5R_functions), 52

rbind.H5D (h5-wrapper), 11
rbind.H5R (H5R_functions), 52
readDataSet (h5-wrapper), 11

subset_assign_h5.H5D
(H5S_H5D_subset_assign), 58
subset.h5.H5D (H5S_H5D_subset_assign), 58
subset.h5.H5S (H5S_H5D_subset_assign), 58
t.H5R (H5R_functions), 52
text_to_dtype, 76
values (factor_ext_functions), 6