PyHexad Documentation

Release 0.1

The HDF Group

March 18, 2015

Contents

Contents:

CHAPTER 1

What is PyHexad?

PyHexad is an Excel add-in for accessing data stored in HDF5 files. It provides about a dozen functions (see *An Overview of PyHexad.*) for reading and writing data, and to create new HDF5 items from Excel. It aims to combine the ease of use, and convenience of Excel with the performance and efficiency of HDF5 smart data containers.

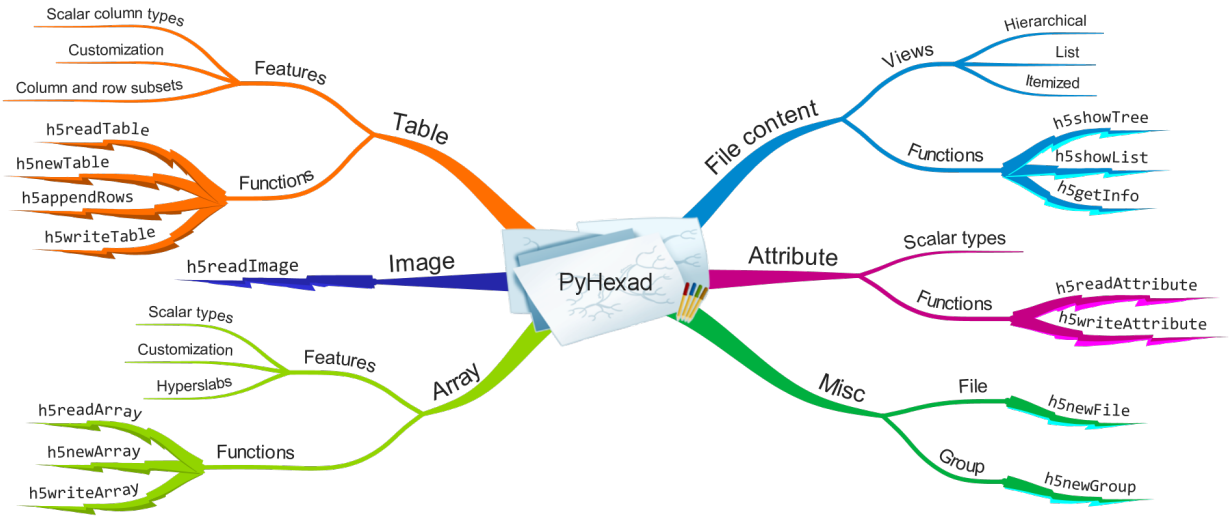


Figure 1.1: An Overview of PyHexad.

In its current form, the main audience might be intermediate Excel users who like to spice up their workbooks with Excel macros and functions, and who might have worked with external data sources such as relational databases. These users will have a natural aversion against "hardcoding" and try to automate their workbooks, they understand what a refresh dependence is, and they have some fluency in the symbology of referencing the content of cells on other worksheets, etc.

On the other end of the spectrum, for intermediate HDF5 users, a "backstairs leading to an old ideal" is open again: For a lot of data stored in HDF5 files, a spreadsheet software such as Excel is a very nice user interface. Check it out!

Wherever you might be coming from, welcome to PyHexad! Please help us improve this product by sharing user stories, reporting issues, requesting new features, and supporting the development.

1.1 Conventions

The syntax of the Excel functions provided by PyHexad is documented as follows:

```
h5function(A, B)
```

```
h5function(A, B [, C, D, E])
```

Parameters in brackets, such as `C`, `D`, `E` in the previous example, are **optional** parameters as opposed to mandatory parameters (`A` and `B`).

All parameters are **positional** parameters and must be used accordingly. An example might help to illustrate their use. The call

```
h5function(A, B, E)
```

passes `E` as the argument to the `C` position. This is quite different from

```
h5function(A, B, , , E)
```

which passes `E` as argument in the `E` position. (and no arguments in the `C` and `D` positions)

Unfortunately, Excel functions have no *keyword* parameters and this is a potential source of errors in using PyHexad functions.

CHAPTER 2

Installation

Currently, the installation is a two step process:

1. Verify that all the prerequisites are installed.
2. Install the PyHexad Python module.

The entire procedure is automated only for Enthought Canopy. If you are one of the lucky Canopy users you can skip the remainder of this chapter. For all other Python installations, take a deep breath and continue reading the remainder of this chapter!

2.1 Prerequisites

PyHexad depends on Microsoft Excel, Python 2.x, NumPy, h5py, PyXLL, and HDF5. Please download the prerequisites from the links provided below and follow the respective installation instructions.

- NumPy and h5py
- PyXLL
- HDF5

Our reference platform for PyHexad is configured as follows:

- Windows 8.1 Pro (64-bit)
- Excel 2013 (**32-bit**)
- Python 2.7.6.9 (**32-bit**)
- NumPy 1.9.1 (**32-bit**)
- h5py 2.3.1 (**32-bit**)
- PyXLL 2.2.2 (**32-bit**)
- [HDF5 1.8.14 (64-bit) (see *Finishing Touches*)]

Other versions/combination will most likely work, but I have not tested them.

Warning: Don't try to mix architectures! It won't work. With the exception of the operating system and the HDF5 tools, all components must be either 32-bit or 64-bit.

Note: There are 64-bit versions of all components and there is a good chance that they'll just work, but this has **not** been tested.

2.2 The pyhexad Python Module

There are at least two other options for installing the PyHexad module.

Use pip or setup.py and install PyHexad from PyPI

We have created a repository for PyHexad on PyPI. If you have pip (pip is a tool recommended for installing and managing Python packages.) installed run:

```
pip install pyhexad
```

Otherwise, download the package pyhexad-0.1.0.zip, unpack it, and run:

```
python setup.py install
```

Use a Windows Installer

Download and run the Windows installer pyhexad-0.1.0.win32.exe and follow the on-screen instructions.

Tell PyXLL about pyhexad

The PyXLL settings are controlled from a configuration file, pyxll.cfg, in the PyXLL installation directory. Your pyxll.cfg might look similar to the following:

```
[PYXLL]
developer_mode = 0
pythonpath =
modules =
        module1
        module2

[LOG]
verbosity = info
format = %(asctime)s - %(levelname)s : %(message)s
path = ./logs
file = pyxll.%(date)s.log
```

Please add pyhexad to the modules section of the file.:

```
[PYXLL]
developer_mode = 0
pythonpath =
modules =
        module1
        module2
        pyhead

[LOG]
verbosity = info
format = %(asctime)s - %(levelname)s : %(message)s
path = ./logs
file = pyxll.%(date)s.log
```

That's it! With the heavy lifting out of the way, it's time to verify that our effort wasn't in vain...

2.3 Sanity Check

After completing the installation, please verify that you have access to the PyHexad functions from Excel. Here's a simple test:

1. Open a blank workbook in Excel.
2. Place the cursor into a cell of a workbook, type `=h5py_version()`, and hit enter.

If the installation is "sane", while typing `h5py_version`, AutoComplete will already have suggested all kinds of completions starting with the `h5` prefix. The result should be the version of your `h5py` installation displayed in the cell where you placed that function call, e.g., `2.3.1`.

2.4 Finishing Touches

In *Prerequisites*, we listed HDF5 1.8.14 as one of the dependencies. There is only one function in PyHexad, `h5readImage`, which currently depends on the `h52gif` tool included in the standard Windows distribution of HDF5. If you are not interested in reading HDF5 images into Excel, you are all set and ready for the next chapter (*Displaying the Contents of an HDF5 File*).

Note: Good news: This dependence will most likely be gone in the release version, but it's there for now...

To ensure that PyHexad picks up a version of `h52gif`, please check that the configuration in PyHexad's `config.py` file matches your local installation. `config.py` is located in your Python packages directory, typically named `site-packages`. For example, on my machine the path is:

```
C:\\Python27\\Lib\\site-packages\\pyhexad
```

`config.py` stores the location and name of the `h52gif` tool in a class called `Places`:

```
class Places(object):

    HDF5_HOME = 'C:\\Progra~1\\HDF_Group\\HDF5\\1.8.14'
    H52GIF = 'h52gifdll.exe'
```

If `HDF5_HOME` or `H52GIF` don't match your local installation, please adjust them accordingly!

CHAPTER 3

Displaying the Contents of an HDF5 File

In figure *HDFView*, a typical display of the contents of an HDF5 file is shown.

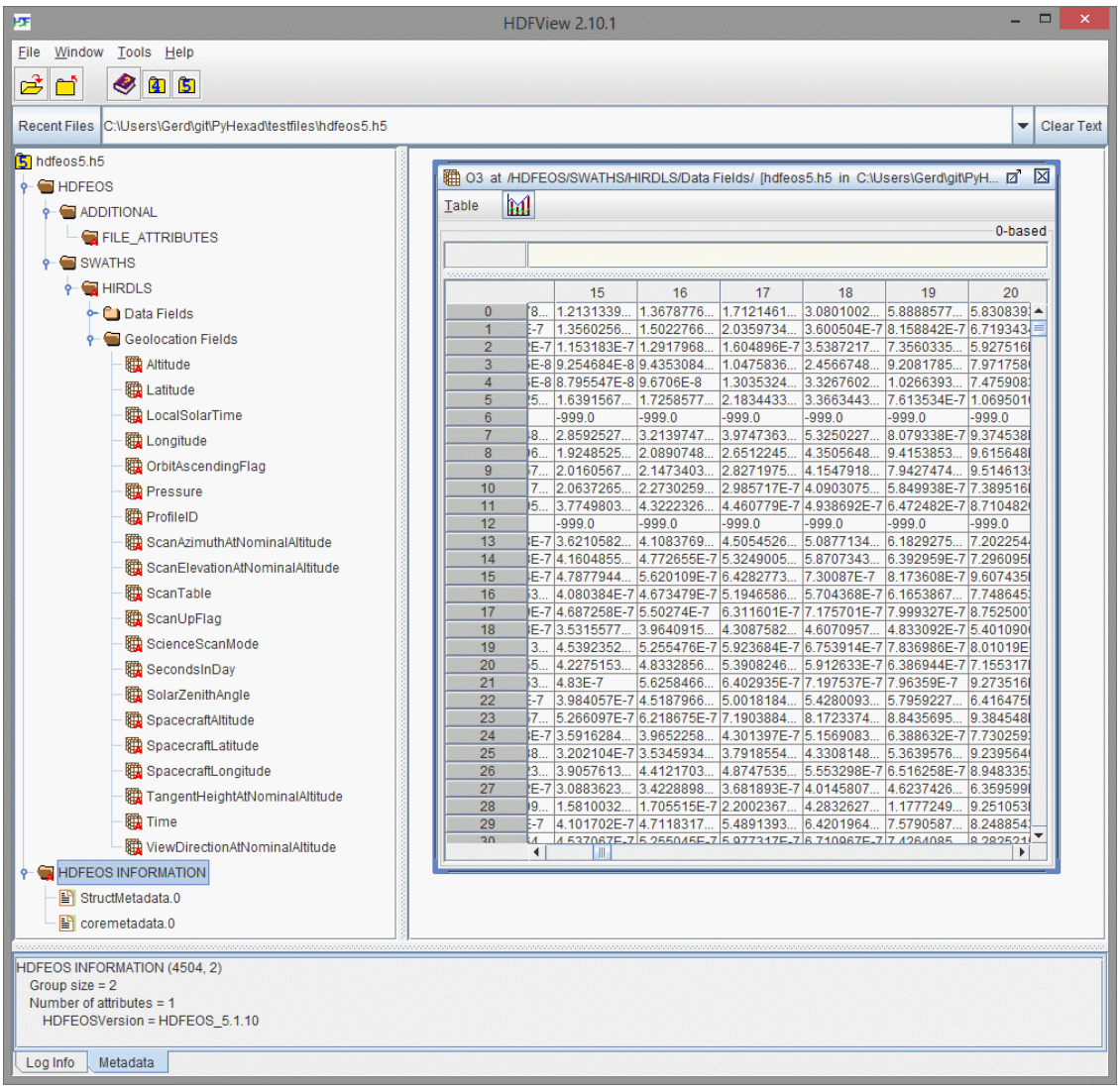


Figure 3.1: HDFView screenshot.

Tools with a graphical user interface, such as [HDFView], usually employ some form of tree-view control to render the imaginary hierarchy of items stored in an HDF5 file. Text-based command line tools mimic this appearance via indentation of text labels (plus ASCII "graphics").

Without the use of custom menus or controls, an Excel worksheet is not the ideal medium for rendering such structures and one has to make due with the limitations of a (coarse) rectangular grid of cells. PyHexad offers three functions to explore and display information about the contents of an HDF5 file:

The first function, *h5showTree*, populates a range of cells with HDF5 link names, creating an impression similar to that of command line tools. This is useful for a first orientation, but is of limited use since it offers no further event handling capabilities.

The second function, *h5showList*, offers a tabular representation with additional descriptions such as the HDF5 datatype and shape of HDF5 datasets and attributes.

The third function, *h5getInfo*, is intended to provide detailed information about individual HDF5 items and their properties.

3.1 A Hierarchical View: `h5showTree`

`h5showTree` displays the contents of an HDF5 file in *hierarchical* or tree-view form. Starting at the HDF5 root group or any other HDF5 group in the file, all reachable HDF5 objects are visited recursively. The HDF5 path names of HDF5 objects which are "deeper" in the HDF5 group hierarchy appear on a worksheet in cells farther to the right, which mimics a tree-view within the confines of a worksheet.

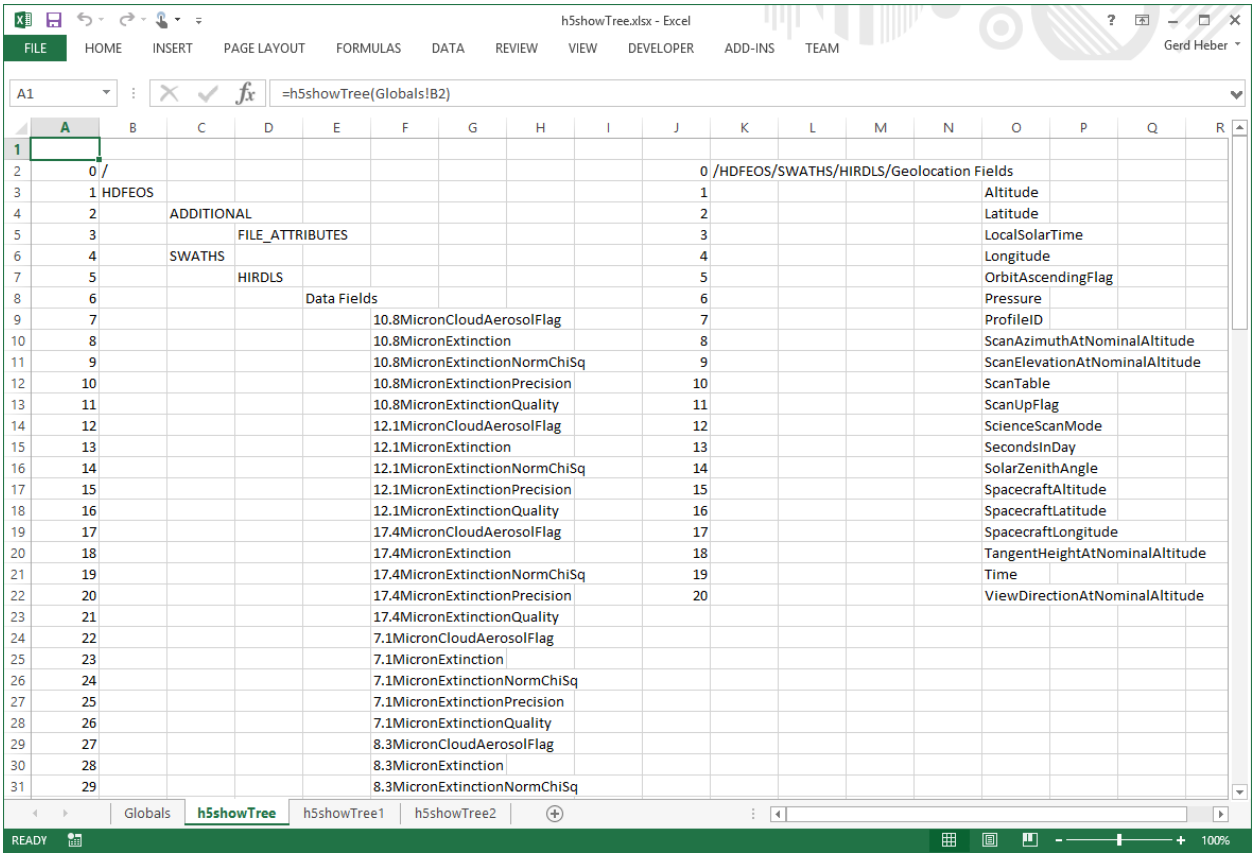


Figure 3.2: An Excel *h5showTree* screenshot.

Excel UDF Syntax

```
h5showTree(filename)
```

```
h5showTree(filename [, location])
```

Mandatory Arguments

Argument	Description
`filename`	A text string specifying the name of an HDF5 file.

Optional Arguments

Argument	Description
`location`	A text string (path) specifying where to begin the traversal

Return Value

On success, `h5showTree` populates a range of cells with the HDF5 path names of the objects visited.

On error, an error message (string) is returned.

Examples

Display the HDF5 hierarchy starting at the path `/HDFEOS/GRIDS` in file `file.he5`.

```
h5showTree("file.he5", "/HDFEOS/GRIDS")
```

Error Conditions

The following conditions will create an error:

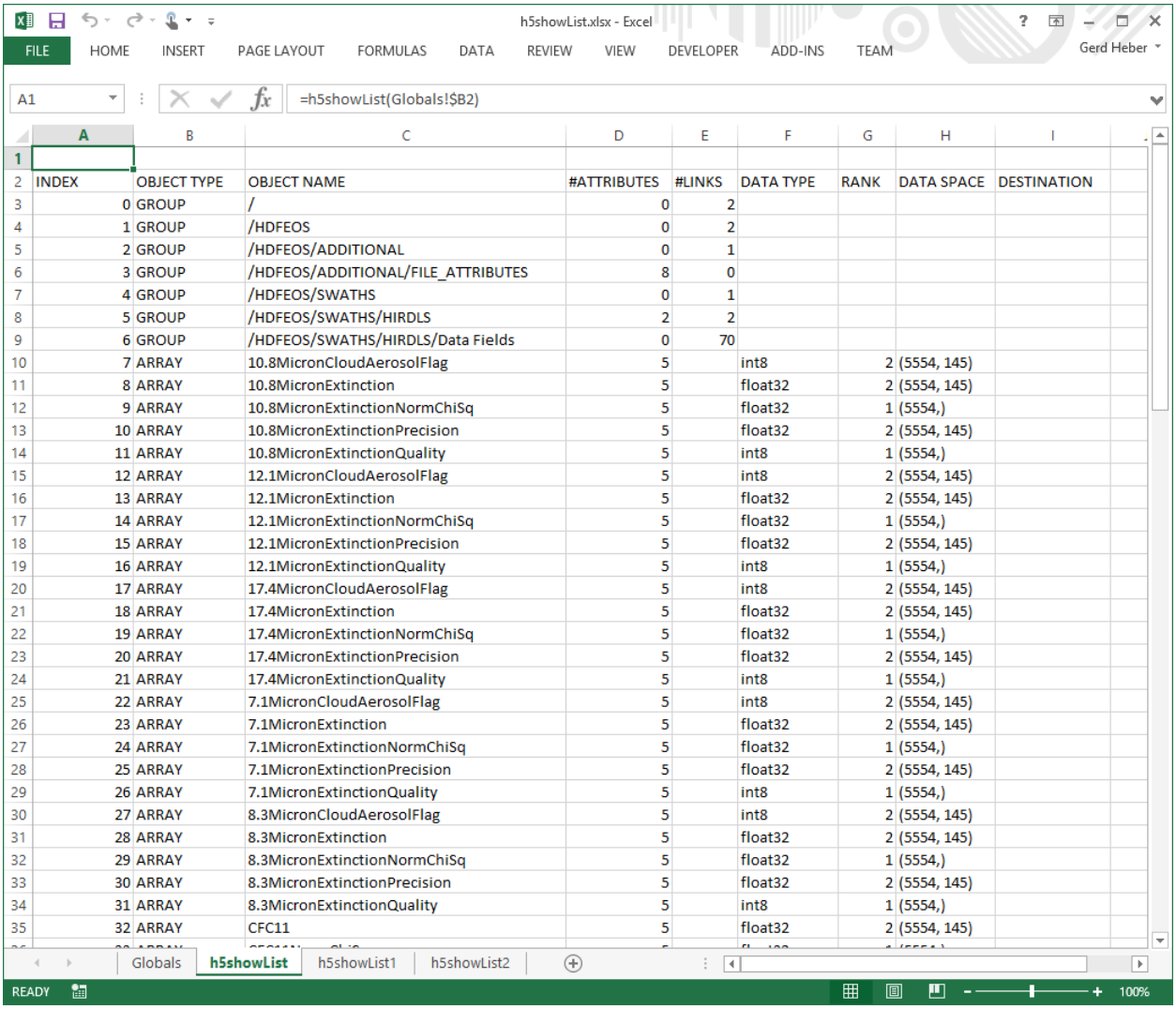
1. An invalid file name
    - An empty string or a string that contains characters not supported by the operating system
    - It refers to a file system location for which the user has insufficient access privileges
2. An invalid location
    - An empty string
    - No HDF5 object exists at the specified location

See Also

h5getInfo, *h5showList*

3.2 A Tabular View: `h5showList`

`h5showList` displays the contents of an HDF5 file in *tabular* form. Unless an alternative starting location is specified, all HDF5 objects in the HDF5 file are visited recursively. HDF5 objects are clustered by their "parent HDF5 group", and object properties, such as HDF5 attribute count, are shown in cells to the right. (The specifics displayed depend on the particular kind of HDF5 object or item.)



INDEX	OBJECT TYPE	OBJECT NAME	#ATTRIBUTES	#LINKS	DATA TYPE	RANK	DATA SPACE	DESTINATION
0	GROUP	/	0	2				
1	GROUP	/HDFEOS	0	2				
2	GROUP	/HDFEOS/ADDITIONAL	0	1				
3	GROUP	/HDFEOS/ADDITIONAL/FILE_ATTRIBUTES	8	0				
4	GROUP	/HDFEOS/SWATHS	0	1				
5	GROUP	/HDFEOS/SWATHS/HIRDLS	2	2				
6	GROUP	/HDFEOS/SWATHS/HIRDLS/Data Fields	0	70				
7	ARRAY	10.8MicronCloudAerosolFlag	5		int8	2	(5554, 145)	
8	ARRAY	10.8MicronExtinction	5		float32	2	(5554, 145)	
9	ARRAY	10.8MicronExtinctionNormChiSq	5		float32	1	(5554,)	
10	ARRAY	10.8MicronExtinctionPrecision	5		float32	2	(5554, 145)	
11	ARRAY	10.8MicronExtinctionQuality	5		int8	1	(5554,)	
12	ARRAY	12.1MicronCloudAerosolFlag	5		int8	2	(5554, 145)	
13	ARRAY	12.1MicronExtinction	5		float32	2	(5554, 145)	
14	ARRAY	12.1MicronExtinctionNormChiSq	5		float32	1	(5554,)	
15	ARRAY	12.1MicronExtinctionPrecision	5		float32	2	(5554, 145)	
16	ARRAY	12.1MicronExtinctionQuality	5		int8	1	(5554,)	
17	ARRAY	17.4MicronCloudAerosolFlag	5		int8	2	(5554, 145)	
18	ARRAY	17.4MicronExtinction	5		float32	2	(5554, 145)	
19	ARRAY	17.4MicronExtinctionNormChiSq	5		float32	1	(5554,)	
20	ARRAY	17.4MicronExtinctionPrecision	5		float32	2	(5554, 145)	
21	ARRAY	17.4MicronExtinctionQuality	5		int8	1	(5554,)	
22	ARRAY	7.1MicronCloudAerosolFlag	5		int8	2	(5554, 145)	
23	ARRAY	7.1MicronExtinction	5		float32	2	(5554, 145)	
24	ARRAY	7.1MicronExtinctionNormChiSq	5		float32	1	(5554,)	
25	ARRAY	7.1MicronExtinctionPrecision	5		float32	2	(5554, 145)	
26	ARRAY	7.1MicronExtinctionQuality	5		int8	1	(5554,)	
27	ARRAY	8.3MicronCloudAerosolFlag	5		int8	2	(5554, 145)	
28	ARRAY	8.3MicronExtinction	5		float32	2	(5554, 145)	
29	ARRAY	8.3MicronExtinctionNormChiSq	5		float32	1	(5554,)	
30	ARRAY	8.3MicronExtinctionPrecision	5		float32	2	(5554, 145)	
31	ARRAY	8.3MicronExtinctionQuality	5		int8	1	(5554,)	
32	ARRAY	CFC11	5		float32	2	(5554, 145)	

Figure 3.3: Excel *h5showList* screenshot.

Excel UDF Syntax

```
h5showList(filename)

h5showList(filename [, location])
```

Mandatory Arguments

Argument	Description
`filename`	A text string specifying the name of an HDF5 file.

Optional Arguments

Argument	Description
`location`	A text string (path) specifying where to begin the traversal

Return Value

On success, `h5showList` populates a range of cells with different HDF5 object properties in tabular form.

On error, an error message (string) is returned.

Examples

Display in table form information about the HDF5 objects in file `file.h5`, beginning traversal at location `/HDFEOS/GRIDS/SET2/Data Fields`.

```
h5showList("file.h5", "/HDFEOS/GRIDS/SET2/Data Fields")
```

Error Conditions

The following conditions will create an error:

1. An invalid file name
    - An empty string or a string that contains characters not supported by the operating system
    - It refers to a file system location for which the user has insufficient access privileges
2. An invalid location
    - An empty string
    - No HDF5 object exists at the specified location

See Also

h5getInfo, *h5showTree*

3.3 Item Details: `h5getInfo`

`h5getInfo` displays detailed information about an HDF5 item in an HDF5 file. The specifics depend on the kind of HDF5 item, such as HDF5 group, dataset, or external link.

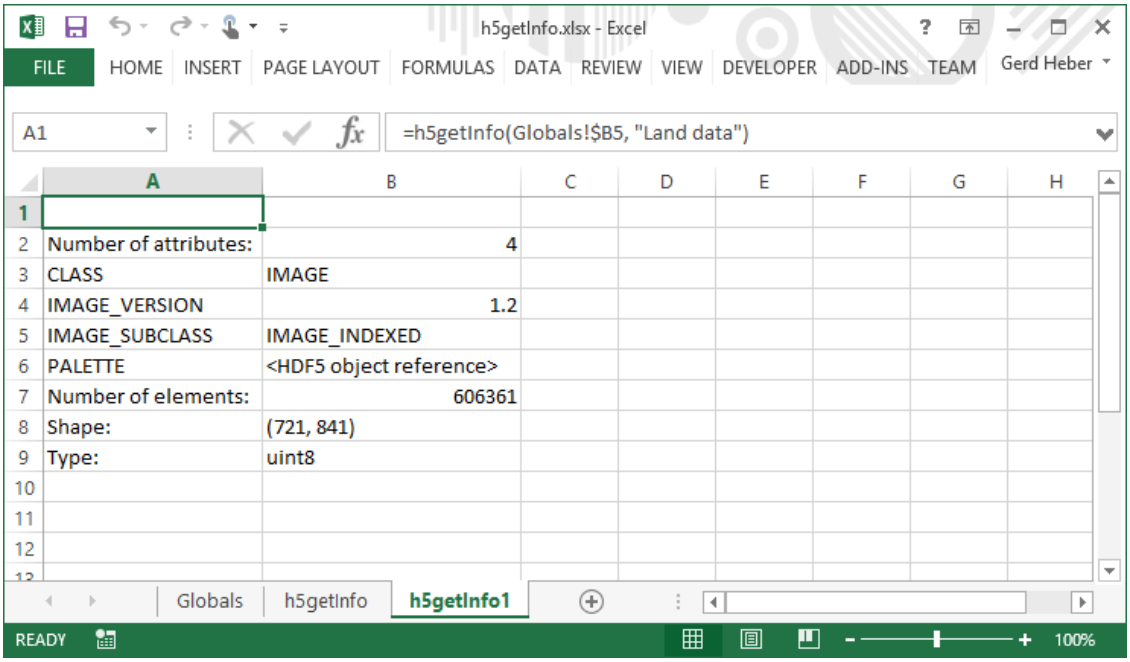


Figure 3.4: Excel *h5getInfo* screenshot.

Excel UDF Syntax

```
h5getInfo(filename, location)
```

Mandatory Arguments

Argument	Description
`filename`	A text string specifying the name of an HDF5 file.
`location`	A text string (path) specifying the location of an HDF5 object

Return Value

On success, `h5getInfo` populates a range of cells with detailed information about an HDF5 object.

On error, an error message (string) is returned.

Examples

Display detailed information about the HDF5 object at location `/HDFEOS/SWATHS/HIRDLS/Geolocation Fields/Pressure` in file `file.he5`.

```
h5showTree("file.he5", "/HDFEOS/SWATHS/HIRDLS/Geolocation Fields/Pressure")
```

Error Conditions

The following conditions will create an error:

1. An invalid file name
    - An empty string or a string that contains characters not supported by the operating system
    - It refers to a file system location for which the user has insufficient access privileges
2. An invalid location
    - An empty string
    - No HDF5 object exists at the specified location

See Also

h5showList, *h5showTree*

CHAPTER 4

Working with HDF5 Arrays

HDF5 supports the storage of up to 32-dimensional arrays of an arbitrary pre-defined or user-defined HDF5 element datatype. This level of generality goes far beyond of what can be rendered on an Excel worksheet and what might be of interest to most Excel users. PyHexad supports standard tasks on one- and two-dimensional HDF5 datasets of pre-defined scalar element types (integers, floating-point numbers, strings). Standard array tasks include the reading of arrays from an HDF5 file, the creation of new arrays in an HDF5 file, and the writing of cell ranges to HDF5 arrays. This is the purpose of the three PyHexad functions in the HDF5 array category: *h5readArray*, *h5newArray*, and *h5writeArray*. All three functions come with several options, for example, to read or write only a sub-array, or to enable compression of the HDF5 arrays in the file.

4.1 Reading Arrays: `h5readArray`

`h5readArray` reads elements of one- and two-dimensional HDF5 arrays. There are variants for reading all elements, a contiguous rectilinear subset (hyperslab), or a strided rectilinear subset of an *HDF5 array*.

Excel UDF Syntax

```
h5readArray(filename, arrayname)

h5readArray(filename, arrayname [, first, last, step])
```

Mandatory Arguments

Argument	Description
`filename`	A text string specifying the name of an HDF5 file
`arrayname`	A text string (path) specifying the location of an HDF5 array

Optional Arguments

Argument	Description
`first`	An integer array specifying the position of the first element to be read
`last`	An integer array specifying the position of the last element to be read
`step`	An integer array specifying the number of positions to skip in each dimension for each element read

Note: The optional arguments are integer arrays whose length must be equal the rank (number of dimensions) of the

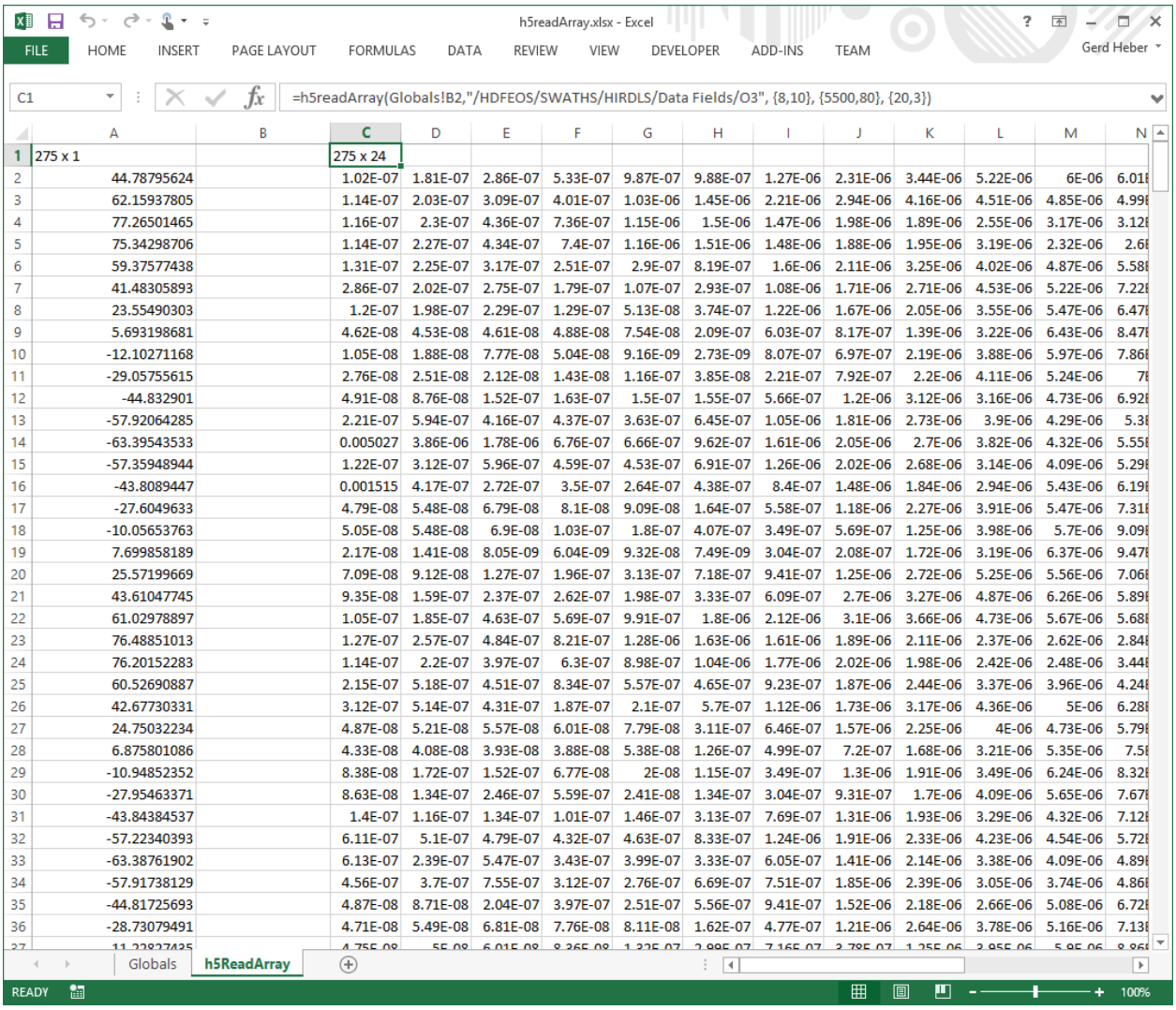


Figure 4.1: An Excel *h5readArray* screenshot.

HDF5 array. This includes the case of a one-dimensional array, e.g., a *first* argument of 5 must be specified as the Excel array literal *{5}*.

Return Value

On success, `h5readArray` populates a cell range with the requested elements.

On error, an error message (string) is returned.

Examples

Read all elements of the `Tot_Precip_Water` array.

```
h5readArray("GSSTF.2b.2008.01.01.he5", \
            "/HDFEOS/GRIDS/SET2/Data Fields/Tot_Precip_Water")
```

Read only every other element of the two-dimensional `Tot_Precip_Water` array.

```
h5readArray("GSSTF.2b.2008.01.01.he5", \
            "/HDFEOS/GRIDS/SET2/Data Fields/Tot_Precip_Water", , , {2,2})
```

Read a contiguous rectangular region of the `Tot_Precip_Water` array.

```
h5readArray("GSSTF.2b.2008.01.01.he5", \
            "/HDFEOS/GRIDS/SET2/Data Fields/Tot_Precip_Water", \
            {25,10}, {356, 89})
```

Error Conditions

The following conditions will create an error:

1. An invalid file name
    - An empty string or a string that contains characters not supported by the operating system
    - It refers to a file system location for which the user has insufficient access privileges
2. An invalid array name
    - An empty string
    - No HDF5 object exists at the specified location
    - The HDF5 object at the specified location is not an HDF5 array
3. The number of elements requested exceeds the maximum Excel row or column count
4. An invalid first position
    - The position is not empty and not an array of non-negative integers
5. An invalid last position
    - The position is not empty and not an array of non-negative integers
6. An invalid step
    - The position is not empty and not an array of positive integers

See Also

h5readTable, *h5readAttribute*, *h5readImage*

4.2 Creating New Arrays: `h5newArray`

`h5newArray` creates a new *HDF5 array*. Array creation can be customized via a creation property list.

Excel UDF Syntax

```
h5newArray(filename, arrayname, size)

h5newArray(filename, arrayname, size [, properties])
```

Mandatory Arguments

Argument	Description
`filename`	A text string specifying the name of an HDF5 file. If the file doesn't exist, it will be created.
`arrayname`	A text string (path) specifying the location of the HDF5 array. Missing intermediate groups will be created automatically. Existing arrays will not be overwritten.
`size`	An Excel array specifying the dimensions of the HDF5 array. Up to 32 dimensions are supported. Dimensions must be non-zero. A negative dimension is treated as unlimited and its absolute value will be used as the initial size. `size` can be specified as a row or column cell range.

Caution: Although the creation of HDF5 arrays of more than two dimensions is supported, there is currently very little supporting functionality in PyHexad for accessing such HDF5 arrays.

Optional Arguments

Argument	Description
`properties`	A text string that is formatted as a list of comma-separated pairs of `Name,Value` arguments. See the **Properties** section for the supported keywords and value ranges. Unrecognized names (and their values) will be ignored.

Properties

Name	Description	Values	Default
`Datatype`	Defines the datatype of the dataset.	A scalar type. See *Supported Types*.	`double`
`Chunksize`	Defines the chunk size for the dataset.	An array of positive integers of the same rank as the dataset.	Not chunked.
`Deflate`	Enables GZIP compression sets level.	(0-9)	0 (no compression)
`FillValue`	Defines the fill value for numeric array.	A literal that can converted to a value of the array element type.	0
`Fletcher32`	Enable Fletcher32 checksum generation for the array.	Boolean	`false`
`Shuffle`	Enable the Shuffle filter.	Boolean	`false`

Return Value

On success, `h5newArray` returns `arrayname` (string).

On error, an error message (string) is returned.

Examples

Create a two-dimensional 12x13 dataset of unsigned 1-byte integers with contiguous layout:

```
h5newArray("file.h5", "/A", {12, 13}, "Datatype,uint8")
```

Create a two-dimensional, extendible (in the second dimension) dataset of single-precision floating-point numbers with chunked layout. The initial extent of the dataset is 12x16 and a chunk size will be chosen automatically.

```
h5newArray("file.h5", "/C/D", {12, -16}, "Datatype,single")
```

Create a three-dimensional, extendible (in the first dimension), gzip compressed (level 6) dataset of double-precision floating-point numbers with chunked (4x4x64 chunks) layout and Fletcher32 checksums generated.

```
h5newArray("file.h5", "/E/F (gzipped + checksums)", {-12, 1024, 768}, \
           "ChunkSize,[4 4 64],Deflate,6,Fletcher32,true")
```

Error Conditions

The following conditions will create an error:

1. An invalid file name
   - An empty string or a string that contains characters not supported by the operating system
   - It refers to a file system location for which the user has insufficient access privileges.
   - It refers to a read-only file.
2. An invalid array name
   - An empty string
   - An HDF5 object exists at the specified location
   - Missing intermediate groups cannot be created.

3. An invalid array size
    - An empty array or an array which contains more than 32 elements
    - A zero dimension
4. Invalid properties
    - A string which is not formatted as a comma-separated list
    - A comma separated list with an odd number of elements
    - A value which is outside the admissible range for the corresponding key

See Also

h5newTable, *h5newGroup*

4.3 Writing Arrays: `h5writeArray`

`h5writeArray` writes data from an Excel range to one- or two-dimensional HDF5 arrays. There are variants for writing all elements, a contiguous rectilinear subset (hyperslab), or a strided rectilinear subset of an *HDF5 array*.

If the HDF5 array does not already exist it will be created, however, optional arguments will be ignored.

Shape mismatches are handled within the extensibility limits of the destination array. That is, an extensible array will be reshaped automatically to accomodate the data within its specified bounds. For a fixed-shape array, a shape mismatch will generate an error.

Excel UDF Syntax

```
h5writeArray(filename, arrayname, data)

h5writeArray(filename, arrayname, data, [, first, last, step])
```

Mandatory Arguments

Argument	Description
`filename`	A text string specifying the name of an HDF5 file
`arrayname`	A text string (path) specifying the location of an HDF5 array
`data`	An Excel range of data to be written to the HDF5 array

Optional Arguments

Argument	Description
`first`	An integer array specifying the position of the first element to be written
`last`	An integer array specifying the position of the last element to be written
`step`	An integer array specifying the number of positions to skip in each dimension for each element written

Note: The optional arguments are integer arrays whose length must be equal to the rank (number of dimensions) of the HDF5 array.

Return Value

On success, `h5writeArray` echos `arrayname`.

On error, an error message (string) is returned.

Examples

Write the content of cell range *$D3:I11* to HDF5 array at */A/B/9 x 6* in file *file.h5*. The array will be created if it doesn't exist already.

```
h5writeArray("file.h5", "/A/B/9 x 6", $D3:I11)
```

Overwrite the fith row with values from range *$D24:I24*.

```
h5writeArray("file.h5", "/A/B/9 x 6", $D24:I24, {5,1}, {5,6}, {1,1})
```

Error Conditions

The following conditions will create an error:

1. An invalid file name
   - An empty string or a string that contains characters not supported by the operating system
   - It refers to a file system location for which the user has insufficient access privileges
2. An invalid array name
   - An empty string
   - No HDF5 object exists at the specified location
   - The HDF5 object at the specified location is not an HDF5 array
3. An invalid data range
   - The cell content cannot be cast to a supported HDF5 array element type.
4. An invalid first position
   - The position is not empty and not an array of non-negative integers
5. An invalid last position
   - The position is not empty and not an array of non-negative integers
6. An invalid step
   - The position is not empty and not an array of positive integers

See Also

h5writeTable, *h5writeAttribute*

CHAPTER 5

Working with HDF5 Tables

HDF5 tables are one-dimensional, extensible HDF5 datasets whose elements are of an HDF5 compound datatype. PyHexad supports compound types whose fields ("columns") are scalar, pre-defined HDF5 datatypes (integers, floating-point numbers, strings). There are currently four Excel functions in PyHexad to support standard oerpations on HDF5 tables.

The first function, *h5readTable*, can be used to read an entire HDF5 table into Excel or just a subset of columns or rows.

The second function, *h5newTable*, lets one create a new HDF5 table and customize its properties.

The third and fourth functions, *h5appendRows* and *h5writeTable*, allow one to append rows to an existing HDF5 table or overwrite existing rows.

Caution: Beyond the name *table*, HDF5 tables don't share very much with their relational cousins. At the file level, they are stored as HDF5 datasets and that largely dictates their "semantics", which means that reading from and writing to an HDF5 table is much more akin to accessing an array than querying or updating a relational table. See [PyTables] and [pandas] for better table abstractions on top of HDF5.

5.1 Reading Tables: `h5readTable`

`h5readTable` reads rows from an *HDF5 table*. There are variants for reading a subset of columns or reading a contiguous or strided range of rows.

Excel UDF Syntax

```
h5readTable(filename, tablename)

h5readTable(filename, tablename [, columns, first, last, step])
```

Mandatory Arguments

Argument	Description
`filename`	A text string specifying the name of an HDF5 file
`tablename`	A text string (path) specifying the location of an HDF5 table

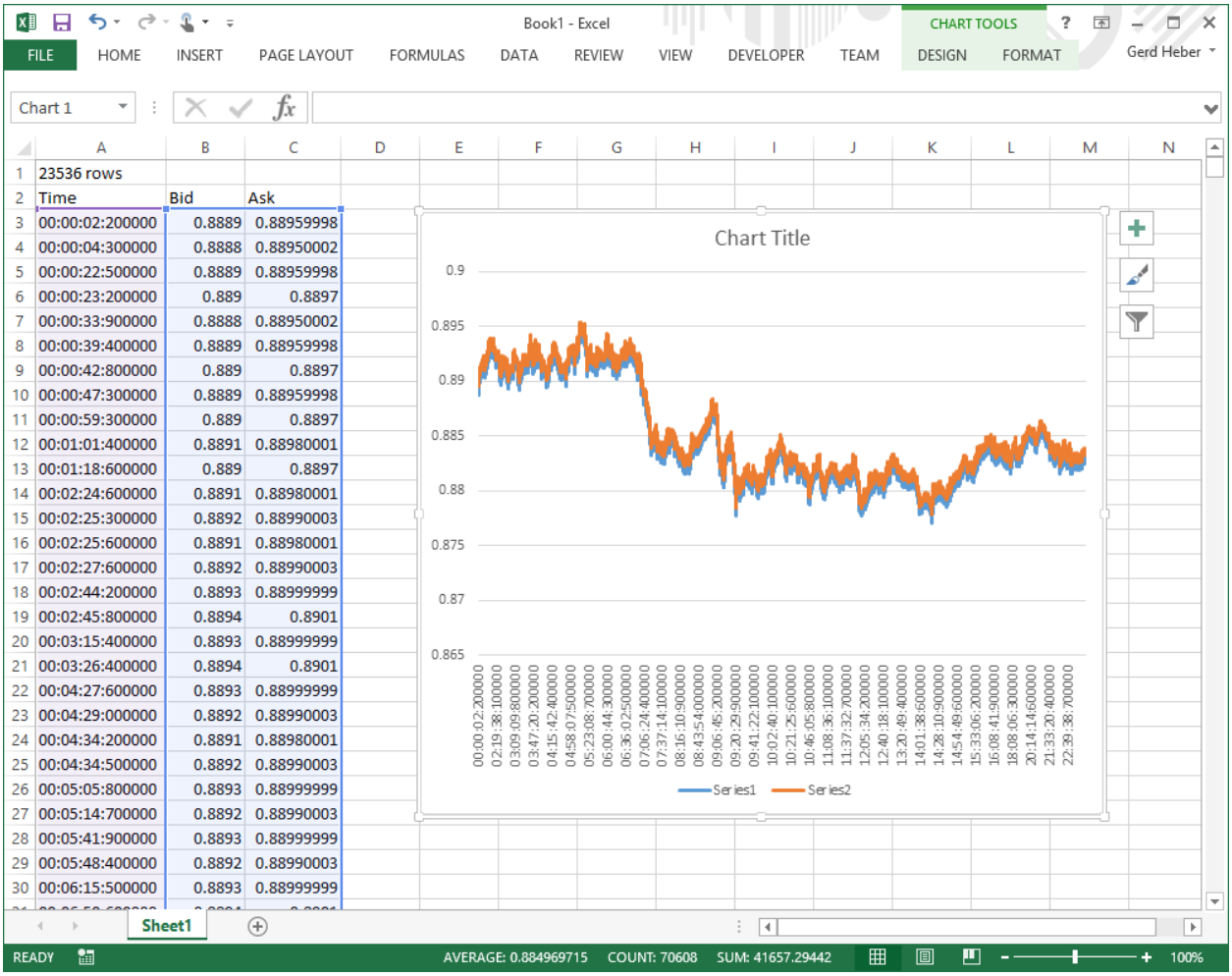


Figure 5.1: An Excel *h5readTable* screenshot.

Optional Arguments

Argument	Description
`columns`	An array of text strings specifying the columns to be read
`first`	An integer specifying the first row to be read
`last`	An integer specifying the last row to be read
`step`	An integer specifying the number of rows to skip for each read row.

Return Value

On success, `h5readTable` populates a cell range with a the requested table rows. The first row of the range contains the table heading (column names).

On error, an error message (string) is returned.

Examples

Read tick data from September 22, 2011.

```
h5readTable("tickdata.h5", "/22-09-2011")
```

Sample the tickdata and read only every 10th value.

```
h5readTable("tickdata.h5", "/22-09-2011", , , , 10)
```

Read the timestamp and ask only.

```
h5readTable("tickdata.h5", "/22-09-2011", {"Time", "Ask"})
```

Read only ticks between rows 1,000 and 15,000.

```
h5readTable("tickdata.h5", "/22-09-2011", , 1000, 15000)
```

Error Conditions

The following conditions will create an error:

1. An invalid file name
    - An empty string or a string that contains characters not supported by the operating system
    - It refers to a file system location for which the user has insufficient access privileges
2. An invalid table name
    - An empty string
    - No HDF5 object exists at the specified location
    - The HDF5 object at the specified location is not an HDF5 table
3. The number of rows requested exceeds the maximum Excel row count
4. An invalid column selection
    - An empty array
    - A column name that is not defined in the HDF5 table

5. An invalid first row index
    - The argument is not empty and not a non-negative integer
6. An invalid last row index
    - The argument is not empty and not a non-negative integer
7. An invalid step
    - The argument is not empty and not a positive integer

See Also

h5readArray, *h5readAttribute*, *h5readImage*

5.2 Creating New Tables: `h5newTable`

`h5newTable` creates a new *HDF5 table*. Table creation can be customized via a creation property list.

Excel UDF Syntax

```
h5newTable(filename, tablename, heading)

h5newTable(filename, tablename, heading [, properties])
```

Mandatory Arguments

Argument	Description
`filename`	A text string specifying the name of an HDF5 file. If the file doesn't exist, it will be created.
`tablename`	A text string (path) specifying the location of the HDF5 table. Missing intermediate groups will be created automatically. Existing table will **not** be overwritten.
`heading`	A text string specifying the column names and types as a list of comma-separated `Name,Type` pairs. Comma characters in field names must be scaped by a backslash `\` character. Restrictions on the field types apply. See *Supported Types*.

Optional Arguments

Argument	Description
`properties`	A text string that is formatted as a list of comma-separated pairs of `Name,Value` arguments. See the **Properties** section for the supported keywords and value ranges. Unrecognized names (and their values) will be ignored.

Properties

Name	Description	Values	Default
`Chunksize`	Defines the chunk size for the table.	A positive integer. positive integers	Auto.
`Deflate`	Enables GZIP compression sets level.	(0-9)	0 (no compression)
`Fletcher32`	Enable Fletcher32 checksum generation for the array.	Boolean	`false`

Return Value

On success, `h5newTable` returns `tablename` (string).

On error, an error message (string) is returned.

Examples

Create a table with a single column of unsigned integers. Note that the column name contains a comma character and needs to be escaped by `\`.

```
h5newTable("sample.h5", "/My/new/HDF5 table", "City\, State,uint8")
```

Create a table with 5 columns of different types.

```
h5newTable("sample.h5", "/table2", \
           "City\, State,uint8,x,double,y,double,A\, B,int16,v,single[2]")
```

Create a table with four columns and control the chunk size (= 128 rows), the compression level (= 6), and enable Fletcher32 checksum generation.

```
h5newTable("sample.h5", "/table3", \
           "Howdy,string,x,double,y,double,v,single[3]", \
           "Chunksize,128,Deflate,6,Fletcher32,true")
```

Error Conditions

The following conditions will create an error:

1. An invalid file name
   - An empty string or a string that contains characters not supported by the operating system
   - It refers to a file system location for which the user has insufficient access privileges.
   - It refers to a read-only file.
2. An invalid table name
   - An empty string
   - An HDF5 object exists at the specified location
   - Missing intermediate groups cannot be created.
3. An invalid heading
   - An empty string or a string which is not formatted as a comma-separated list

- A (non-escaped) comma separated list with an odd number of elements
- An invalid or unsupported datatype or fill value specification

4. Invalid properties
   - A string which is not formatted as a comma-separated list
   - A comma separated list with an odd number of elements
   - A value which is outside the admissible range for the corresponding key

See Also

h5newArray, *h5newGroup*

5.3 Appending Rows to a Table: `h5appendRows`

`h5appendRows` appends rows to an existing HDF5 table.

Excel UDF Syntax

```
h5appendRows(filename, tablename, rows)
```

Mandatory Arguments

Argument	Description
`filename`	A text string specifying the name of an HDF5 file
`tablename`	A text string (path) specifying the location of an HDF5 table
`rows`	An Excel range of rows to be appended to the HDF5 table

Note: The order of the columns in the row range must match the order of columns in the HDF5 table in the file.

Return Value

On success, `h5appendRows` returns the number of rows appended.

On error, an error message (string) is returned.

Examples

Append the rows in range *A1:C23581* on worksheet *Sheet2* to the HDF5 table at */Ask & Bid/20140423* in the file *tickdata.h5*.

```
h5appendRows("tickdata.h5", "/Ask & Bid/20140423", Sheet2!$A1:C23581)
```

Error Conditions

The following conditions will create an error:

1. An invalid file name
   - An empty string or a string that contains characters not supported by the operating system
   - It refers to a file system location for which the user has insufficient access privileges
2. An invalid table name
   - An empty string
   - No HDF5 object exists at the specified location
   - The HDF5 object at the specified location is not an HDF5 table
3. An invalid row set
   - The number or type of columns in the rows set does not match the number or type of columns in the file

5.4 Writing to a Table: `h5writeTable`

`h5writeTable` writes rows to an existing HDF5 table. The write operation can be restricted to a subset of columns.

Warning: This is a destructive operation. Existing rows will be overwritten, or the table extended as necessary. Unless the table is empty, this is not an append operation. Use *h5appendRows* to append rows to an HDF5 table.

Excel UDF Syntax

```
h5writeTable(filename, tablename, rows)

h5writeTable(filename, tablename, rows [, columns])
```

Mandatory Arguments

Argument	Description
`filename`	A text string specifying the name of an HDF5 file
`tablename`	A text string (path) specifying the location of an HDF5 table
`rows`	An Excel range of rows to be written to the HDF5 table

Optional Arguments

Argument	Description
`columns`	A string array listing the columns to be written.

Return Value

On success, `h5writeTable` returns the number of rows written.

On error, an error message (string) is returned.

Examples

Overwrite the *Ask* column in the HDF5 table at */Ask & Bid/20140423* in the file *tickdata.h5* with data from the Excel range *B1:B23581* on worksheet *Sheet2*.

```
h5appendRows("tickdata.h5", "/Ask & Bid/20140423", Sheet2!$B1:B23581, "Ask")
```

Error Conditions

The following conditions will create an error:

1. An invalid file name
    - An empty string or a string that contains characters not supported by the operating system
    - It refers to a file system location for which the user has insufficient access privileges
2. An invalid table name
    - An empty string
    - No HDF5 object exists at the specified location
    - The HDF5 object at the specified location is not an HDF5 table
3. An invalid row set
    - The number or type of columns in the rows set does not match the number or type of columns in the file
4. An invalid set of columns.
    - One or more of the column names provided do not match the column names of the HDF5 table in the file.

See Also

h5writeArray, *h5writeAttribute*

CHAPTER 6

Working with HDF5 Attributes

HDF5 attributes are a convenient mechanism for attaching metadata to HDF5 objects (datasets, groups, datatype objects). PyHexad has two functions for reading and writing HDF5 attributes, *h5readAttribute* and *h5writeAttribute*. The latter does double duty for creating and updating such attributes.

6.1 Reading Attributes: `h5readAttribute`

`h5readAttribute` reads and renders the value of an HDF5 attribute as a string.

Excel UDF Syntax

```
h5readAttribute(filename, location, attr)
```

Mandatory Arguments

Argument	Description
`filename`	A text string specifying the name of an HDF5 file
`location`	A text string (path) specifying the location of an HDF5 object
`attr`	A text string, the attribute's name

Return Value

On success, `h5readAttribute` populates a cell with a string rendering of the attribute value.

On error, an error message (string) is returned.

Examples

Read the `Units` attribute of a dataset.

```
h5readAttribute("GSSTF.2b.2008.01.01.he5", \
                "/HDFEOS/GRIDS/SET2/Data Fields/Tot_Precip_Water", \
                "Units")
```

Read the `HDFEOSVersion` attribute of the object at `/HDFEOS INFORMATION`.

```
h5readAttribute(Sheet1!A1,"/HDFEOS INFORMATION", "HDFEOSVersion")
```

Note: In the last example, the file name is retrieved from cell `A1` on the worksheet `Sheet1`

Error Conditions

The following conditions will create an error:

1. An invalid file name
   - An empty string or a string that contains characters not supported by the operating system
   - It refers to a file system location for which the user has insufficient access privileges
2. An invalid location
   - An empty string
   - No HDF5 object exists at the specified location
3. An invalid attribute name
   - An empty string
   - The HDF5 object doesn't have an attribute of that name
4. The attribute size exceeds 32 KB.

See Also

h5readArray, *h5readAttribute*, *h5readImage*

6.2 Writing Attributes: `h5writeAttribute`

`h5writeAttribute` creates a new HDF5 attribute or updates (**overwrites**!) the value of an existing one.

Excel UDF Syntax

```
h5writeAttribute(filename, location, attname, attvalue)
```

Mandatory Arguments

Argument	Description
`filename`	A text string specifying the name of an HDF5 file
`location`	A text string (path) specifying the location of an HDF5 object
`attname`	A text string, the HDF5 attribute's name
`attvalue`	The attribute's value.

Return Value

On success, `h5writeAttribute` echoes the name of the HDF5 attribute written.

On error, an error message (string) is returned.

Examples

Write the `Units` attribute of a dataset.

```
h5writeAttribute("GSSTF.2b.2008.01.01.he5", \
                 "/HDFEOS/GRIDS/SET2/Data Fields/Tot_Precip_Water", \
                 "Units", "[mm]")
```

Write the `HDFEOSVersion` attribute of the object at `/HDFEOS INFORMATION`.

```
h5writeAttribute(Sheet1!A1,"/HDFEOS INFORMATION", "HDFEOSVersion", "5.1")
```

Note: In the last example, the file name is retrieved from cell `A1` on the worksheet `Sheet1`

Error Conditions

The following conditions will create an error:

1. An invalid file name
   - An empty string or a string that contains characters not supported by the operating system
   - It refers to a file system location for which the user has insufficient access privileges
2. An invalid location
   - An empty string
   - No HDF5 object exists at the specified location
3. An invalid attribute name
   - An empty string

See Also

h5writeArray, *h5writeTable*

CHAPTER 7

Working with HDF5 Images

An HDF5 file is a (smart) data container not just for numerical data. It is well equipped to store other multi-media content, including images, sounds, and video. Different communities have created several standards describing domain-specific conventions for storing their content in HDF5 files. The HDF5 Image and Palette Specification describes how to store a large class of raster images in HDF5. PyHexad supports the import and display of HDF5 images into a worksheet via the *h5readImage* function.

7.1 Reading Images: `h5readImage`

`h5readImage` reads an *HDF5 image* and renders it as Graphics Interchange Format (GIF) image on an Excel worksheet.

Excel UDF Syntax

```
h5readImage(filename, imagename)

h5readImage(filename, imagename [, palettename])
```

Mandatory Arguments

Argument	Description
`filename`	A text string specifying the name of an HDF5 file
`imagename`	A text string (path) specifying the location of an HDF5 image

Optional Arguments

Argument	Description
`palettename`	A text string (path) specifying the location of an HDF5 palette

Return Value

On success, `h5readImage` renders a GIF image on an Excel worksheet.

On error, an error message (string) is returned.

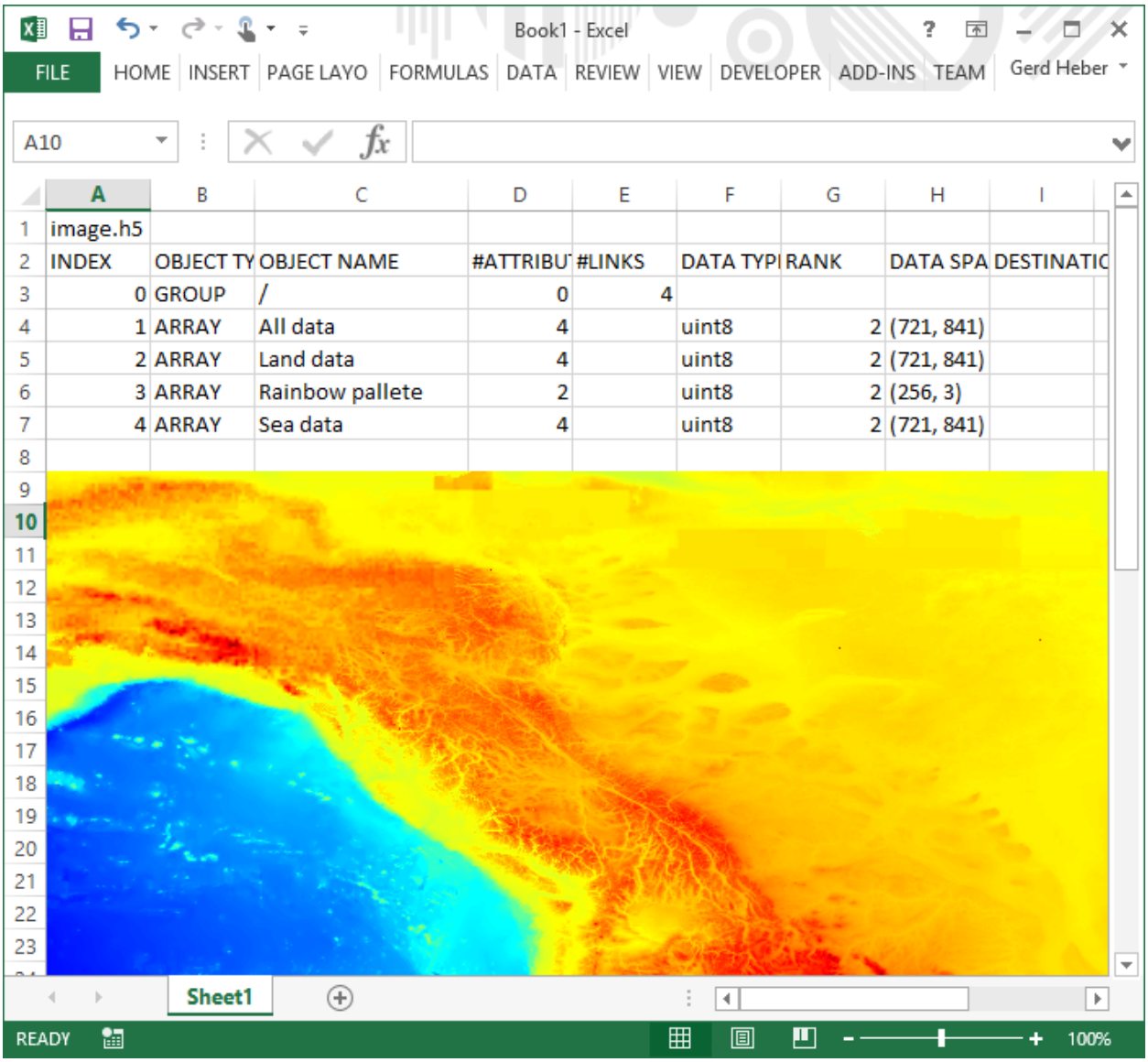


Figure 7.1: An Excel *h5readImage* screenshot.

Examples

Read the `hdflogo` image.

```
h5readImage("HDF5.h5", "/hdflogo")
```

Error Conditions

The following conditions will create an error:

1. An invalid file name
   - An empty string or a string that contains characters not supported by the operating system
   - It refers to a file system location for which the user has insufficient access privileges
2. An invalid image name
   - An empty string
   - No HDF5 object exists at the specified location
   - The HDF5 object at the specified location is not an HDF5 image
3. An invalid palette name
   - An empty string
   - No HDF5 object exists at the specified location
   - The HDF5 object at the specified location is not an HDF5 palette

See Also

h5readArray, *h5readAttribute*, *h5readTable*

CHAPTER 8

Miscellaneous

8.1 Creating Files: h5newFile

h5newFiles creates a new HDF5 file. If no file name is provided, a random file name will be generated and returned. Existing file will not be overwritten and an error will be generated instead.

Excel UDF Syntax

```
h5newFile()

h5newFile([filename])
```

Optional Arguments

Argument	Description
filename	A text string specifying the name of an HDF5 file.

Return Value

On success, h5newFile returns the file name of the newly created file.

On error, an error message (string) is returned.

Examples

Create an HDF5 file at location c:\tmp\sample.h5.

```
h5newGroup("c:\tmp\sample.h5")
```

Error Conditions

The following conditions will create an error:

1. An invalid file name
    - An empty string or a string that contains characters not supported by the operating system

- It refers to a file system location for which the user has insufficient access privileges.
- A file exists at that location.

See Also

h5newGroup

8.2 Creating Groups: `h5newGroup`

`h5newGroup` creates a new HDF5 group.

Excel UDF Syntax

```
h5newGroup(filename, groupname)
```

Mandatory Arguments

Argument	Description
`filename`	A text string specifying the name of an HDF5 file. If the file doesn't exist, it will be created.
`groupname`	A text string (path) specifying the location of the HDF5 group. Missing intermediate groups will be created automatically.

Return Value

On success, `h5newGroup` returns `groupname` (string).

On error, an error message (string) is returned.

Examples

Create an HDF5 group at location `/My/new/HDF5 group`.

```
h5newGroup("sample.h5", "/My/new/HDF5 group")
```

Error Conditions

The following conditions will create an error:

1. An invalid file name
   - An empty string or a string that contains characters not supported by the operating system
   - It refers to a file system location for which the user has insufficient access privileges.
   - It refers to a read-only file.
2. An invalid group name

- An empty string
- No HDF5 object that is not an HDF5 group exists at the specified location
- Missing intermediate groups cannot be created.

See Also

h5newFile

CHAPTER 9

Supported Types

9.1 Scalar Types

9.1.1 Integers

Signed

Types	Remarks	HDF5 File Type
`byte, int8`	C `char`	`H5T_STD_I8LE`
`short, int16`	C `short`	`H5T_STD_I16LE`
`int, int32`	C `int`	`H5T_STD_I32LE`
`long, int64`	C `long`	`H5T_STD_I64LE`

Unsigned

Types	Remarks	HDF5 File Type
`ubyte, uint8`	C `unsigned char`	`H5T_STD_U8LE`
`ushort, uint16`	C `unsigned short`	`H5T_STD_U16LE`
`uint, uint32`	C `unsigned int`	`H5T_STD_U32LE`
`ulong, uint64`	C `unsigned long`	`H5T_STD_U64LE`

9.1.2 Floating-Point Numbers

Types	Remarks	HDF5 File Type
`float, float32, single`	C `float`	`H5T_IEEE_F32LE`
`double, float64`	C `double`	`H5T_IEEE_F64LE`

9.1.3 Strings

Types	Remarks	HDF5 File Type
`stringN`	Fixed-length C ASCII string of length `N`	`H5T_C_S1`
`string`	Variable-length C ASCII string	HDF5 varaiable-length ASCII string

9.2 Non-Scalar Types

9.2.1 Arrays

An array type is specified as `T[a b c ... n]` where `T` is a scalar type and `a, b, c, ..., n` are positive integers (dimensions). Array types of up to 32 dimensions are supported.

9.2.2 Compounds

A compound type is specified as a comma separated list of field name and type pairs `Name1,Type1,Name2,Type2,...,NameN,TypeN` where `Name` is an ASCII string and `Type` is a scalar type name.

If `Name` contains a comma, it must be escaped with a backslash `\`, e.g., `City\, State`.

CHAPTER 10

Glossary

HDF5 Array A multi-dimensional HDF5 dataset whose elements are of a scalar HDF5 datatype.

HDF5 Image A two-dimensional HDF5 dataset whose elements are of an HDF5 integer datatype. Palettes...

HDF5 Link An explicit, single-source, single-destination, unidirectional association between an HDF5 group and another HDF5 item.

HDF5 Table A one-dimensional HDF5 dataset whose elements are of an HDF5 compound datatype.

CHAPTER 11

Copyright

```
Copyright Notice and License Terms for PyHexad
-------------------------------------------------------------------------------

PyHexad
Copyright 2014 by The HDF Group.

All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted for any purpose (including commercial purposes)
provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice,
   this list of conditions, and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice,
   this list of conditions, and the following disclaimer in the documentation
   and/or materials provided with the distribution.

3. In addition, redistributions of modified forms of the source or binary
   code must carry prominent notices stating that the original code was
   changed and the date of the change.

4. All publications or advertising materials mentioning features or use of
   this software are asked, but not required, to acknowledge that it was
   developed by The HDF Group and credit the contributors.

5. Neither the name of The HDF Group nor the name of any Contributor may
   be used to endorse or promote products derived from this software
   without specific prior written permission from The HDF Group or the
   Contributor, respectively.

DISCLAIMER:
THIS SOFTWARE IS PROVIDED BY THE HDF GROUP "AS IS" WITH NO WARRANTY OF ANY
KIND, EITHER EXPRESSED OR IMPLIED.  In no event shall The HDF Group or the
Contributors be liable for any damages suffered by the users arising out of
the use of this software, even if advised of the possibility of such damage.

-------------------------------------------------------------------------------
```

Bibliography

[Enthought] https://www.enthought.com/

[HDF5] http://www.hdfgroup.org/HDF5/

[HDFView] http://www.hdfgroup.org/products/java/hdfview/

[h5py] http://www.h5py.org/

[PyXLL] https://www.pyxll.com/

[Excel] http://office.microsoft.com/en-us/excel/

[PyTables] http://www.pytables.org/moin

[pandas] http://pandas.pydata.org/

[MathWorks] http://www.mathworks.com/help/matlab/hdf5-files.html

H