

The most important attributes of a dimension scale are:

CLASS	Set to "DIMENSION_SCALE" to indicate that the dataset is a dimension scale.
NAME	Name of dimension such as "X".
REFERENCE_LIST	An array of structures used to keep track of the datasets and dimensions to which the dimension scale is attached. Each element of the structure includes a reference to a dataset and a dimension index.

The most important attributes of a dataset to which a dimension scale is attached are:

DIMENSION_LIST	A variable-length array of references used to keep track of the dimensions used by each dimension of the dataset. The array has one column for each dataset dimension. Each column has one row for each dimension scale attached to the corresponding dataset dimension. A given dataset dimension can have multiple attached dimension scales.
DIMENSION_LABELS	An 1D array containing labels for the dimensions of the dataset.

## HDF5 Dimension Scale Reference

For experts, some sources of additional information on HDF5 dimension scales and related netCDF-4 features are available in Igor's online help topic "HDF5 Dimension Scale Reference".

## Other HDF5 Issues

This section is mostly of interest to advanced HDF5 users.

### HDF5 String Formats

Strings can be written to HDF5 files as datasets or as attributes using several formats:

- Variable length
- Fixed length with null termination
- Fixed length with null padding
- Fixed length with space padding

In addition, strings can be marked as either ASCII or UTF-8.

Usually you do not need to know or care about the format used to write strings. However, some programs do not support all of the formats. If you attempt to load an HDF5 file written by Igor into one of those programs, you may get an error. In that event, you may be able to fix the problem by controlling the string format used to write the file. The rest of this section provides information that may help in that event.

The variable-length string format is most useful when writing a dataset or attribute containing multiple strings of different lengths. For example, when a dataset containing the two strings "ABC" and "DEFGHIJKLMNOP" is written as variable length, each string and its length is stored in the HDF5 file. That requires 3 bytes for "ABC" and 13 bytes for "DEFGHIJKLMNOP" plus the space required to store the length for each string. If these strings were written as fixed length, the fixed length for the dataset or attribute would have to be at least 13 bytes and at least 10 bytes would be wasted by padding when writing "ABC".

The fixed-length string format is most useful when writing a dataset or attribute consisting of a single string. For example, "ABC" can be written using a 3-byte fixed-length datatype with 0 padding bytes.

If more than one string is written as fixed length, the padding mode determines how extra bytes are filled. In null-terminated mode, the extra bytes are filled with null (0) and the first null marks the end of a string. In null-padded mode, the extra bytes are filled with null (0) all consecutive nulls at the end of the string

mark the end of the string. In space-padded mode, the extra bytes are filled with space all consecutive spaces at the end of the string mark the end of the string.

In addition to the variable-length versus fixed-length issue, there is a text encoding issue with HDF5. A string dataset or attribute is marked as either ASCII or UTF-8 depending on the software that wrote it. The marking does not guarantee that the text is valid ASCII or valid UTF-8 as the HDF5 library does not check to make sure that written text is valid as marked nor does it do any text encoding conversions. The marking is merely a statement of the intended text encoding. Some software may fail when reading datasets or attributes marked as ASCII or UTF-8 because the HDF5 library does require that the reading program use a compatible datatype.

With some exceptions explained below, you can control the string format used to save datasets and attributes using the `/STRF={fixedLength,paddingMode,charset}` flag with the **HDF5SaveData** and **HDF5SaveGroup** operations. The `/STRF` flag was added in Igor Pro 9.00.

If *fixedLength* is 0, **HDF5SaveData** writes strings using a variable-length HDF5 string datatype. If *fixedLength* is greater than 0, **HDF5SaveData** writes strings using a fixed-length HDF5 string datatype of the specified length with padding specified by padding mode. If *fixedLength* is -1, **HDF5SaveData** determines the length of the longest string to be written for a given dataset or attribute and writes strings using a fixed-length HDF5 string datatype of that length with padding specified by *paddingMode*.

If *paddingMode* is 0, **HDF5SaveData** writes fixed-length strings as null terminated strings. If *paddingMode* is 1, **HDF5SaveData** writes fixed-length strings as null-padded strings. If *paddingMode* is 2, **HDF5SaveData** writes fixed-length strings as space-padded strings. When writing strings as variable length (*fixedLength*=0), *paddingMode* is ignored.

If *charset* is 0, **HDF5SaveData** writes strings marked as ASCII. If *charset* is 1, **HDF5SaveData** writes strings marked as UTF-8.

An exception is zero-length datasets or attributes which are always written as variable-length UTF-8. Another exception is string variables written by **HDF5SaveGroup** which are always written as fixed-length null padded UTF-8.

This table shows the default string format used for various situations if you omit `/STRF` and whether or not **HDF5SaveData** and **HDF5SaveGroup** honor the `/STRF` flag for the corresponding situation:

	<b>HDF5SaveData Default</b>	<b>HDF5SaveData Behavior</b>
Text Wave Zero Element as Dataset	Variable,NULLPAD,UTF-8	Ignores <code>/STRF</code>
Text Wave Single Element as Dataset	Variable,NULLPAD,UTF-8	Honors <code>/STRF</code>
Text Wave Multiple Elements as Dataset	Variable,NULLPAD,UTF-8	Honors <code>/STRF</code>
String Variable	N/A	HDF5SaveData can not save string variables
Text Wave Zero Element as Attribute	Variable,NULLPAD,UTF-8	Ignores <code>/STRF</code>
Text Wave Single Element as Attribute	Fixed,NULLPAD,UTF-8	Honors <code>/STRF</code>
Text Wave Multiple Elements as Attribute	Variable,NULLPAD,UTF-8	Honors <code>/STRF</code>
	<b>HDF5SaveGroup Default</b>	<b>HDF5SaveGroup Behavior</b>
Text Wave Zero Element as Dataset	Variable,NULLPAD,UTF-8	Ignores <code>/STRF</code>
Text Wave Single Element as Dataset	Variable,NULLPAD,UTF-8	Honors <code>/STRF</code>
Text Wave Multiple Elements as Dataset	Variable,NULLPAD,UTF-8	Honors <code>/STRF</code>
Zero-Length String Variable	Variable,NULLPAD,UTF-8	Ignores <code>/STRF</code>
String Variable > Zero-Length	Fixed,NULLPAD,UTF-8	Ignores <code>/STRF</code>