

|                                          |     |                                       |
|------------------------------------------|-----|---------------------------------------|
| Text Wave Zero Element as Attribute      | N/A | HDF5SaveGroup can not save attributes |
| Text Wave Single Element as Attribute    | N/A | HDF5SaveGroup can not save attributes |
| Text Wave Multiple Elements as Attribute | N/A | HDF5SaveGroup can not save attributes |

## HDF5 String Variable Text Encoding

Igor writes string variables as UTF-8 fixed-length string datasets.

String variables may contain null bytes and text that is invalid as UTF-8. This would occur, for example, if a variable were used to contain binary data. Such string variables are still written as UTF-8 fixed-length string datasets.

## HDF5 Wave Text Encoding

For background information on wave text encoding, see **Wave Text Encodings** on page III-472.

Igor text wave contents are written as variable-length string datasets using UTF-8 text encoding. Other wave elements (units, note, dimension labels) are written as UTF-8 fixed-length string attributes.

Text wave contents may contain null bytes and text that is invalid as UTF-8. This would occur, for example, if a text wave were used to contain binary data. Such contents are still written as UTF-8 variable-length string datasets.

Wave text elements can be marked in Igor as using any supported text encoding. No matter how a wave's text elements are marked, they are written to HDF5 as UTF-8 strings. Consequently, if you save a wave that uses non-UTF-8 text encodings to an HDF5 file and the load it back into Igor, its text encodings change but the characters represented by the text do not.

An exception applies to wave elements marked as binary (see **Text Waves Containing Binary Data** on page III-475 for background information). When you save a wave containing one or more binary elements as an HDF5 dataset, Igor adds the IGORWaveBinaryFlags attribute to the dataset. This attribute identifies the wave elements marked as binary using bits as defined for the **WaveTextEncoding** function. When you load the dataset from an HDF5 file, Igor restores the binary marking for the wave elements corresponding to the bits set in the attribute. The IGORWaveBinaryFlags attribute was added in Igor Pro 9.00.

In Igor Pro 9.00 and later, the HDF5LoadData and HDF5LoadGroup operations can check for binary data loaded into text waves from string datasets and mark such text waves as containing binary. If you load binary data from string datasets, see bit 1 of the /OPTS flag of those operations for details.

## HDF5 File Paths on Windows

The HDF5 library does not support Unicode file paths on Windows. This is explained under "Filenames" at <https://support.hdfgroup.org/HDF5/doc/Advanced/UsingUnicode/index.html>

Instead, on Windows the HDF5 library uses system text encoding (also known as "system locale" and "language for non-Unicode programs").

This worked fine with Igor6 because Igor6 also used system text encoding internally. However, Igor7 and later store all text internally as UTF-8. Consequently, on Windows Igor converts file paths from UTF-8 to system text encoding before passing those paths to HDF5 library routines. This allows you to use paths containing non-ASCII characters. However, you are limited to those characters that are supported by the current system text encoding on your system. For example, if your system is set to use Windows-1252 ("western" code page) as the system text encoding, then you will get an error if the path to your file contains, for example, Japanese, because Windows-1252 does not support Japanese characters.