See Also

Text Encodings on page III-459, **Text Encoding Names and Codes** on page III-490, **TextEncodingCode**

TextFile

`TextFile(`*`pathName`*`, `*`index`*` [, `*`creatorStr`*`])`

Note: TextFile is antiquated. Use **IndexedFile** instead.

The TextFile function returns a string containing the name of the *index*th TEXT file from the folder specified by *pathName*.

On Macintosh, TextFile returns only files whose file type property is TEXT, regardless of the file's extension.

On Windows, Igor considers files with ".txt" extensions to be of type TEXT.

Details

TextFile returns an empty string ("") if there is no such file.

pathName is the name of an Igor symbolic path; it is *not* a string.

index starts from zero.

creatorStr is an optional string argument containing four ASCII characters such as "IGR0". Only files of the specified Macintosh creator code are indexed. Set *creatorStr* to "`????`" to index all text files (or omit the argument altogether). This argument is ignored on Windows systems.

The order of files in a folder is determined by the operating system.

Examples

You can use TextFile in a procedure to sequence through each TEXT file in a folder, put the name of the text file into a string variable, and use this string variable as a parameter to the **LoadWave** or **Open** operations:

```
Function/S PrintFirstLineOfTextFiles(pathName)
    String pathName                   // Name of an Igor symbolic path.

    Variable refNum, index
    String str, fileName
    index = 0
    do
        fileName = TextFile($pathName, index)
        if (strlen(fileName) == 0)
            break                     // No more files
        endif
        Open/R/P=$pathName refNum as fileName
        FReadLine refNum, str         // Read first line including CR/LF
        Print fileName +":" + str     // Print file name and first line
        Close refNum
        index += 1                    // Next file
    while (1)
End
```

See Also

See the **IndexedFile** function, which is similar to TextFile but works on files of any type, and also **IndexedDir**. Also see the **LoadWave** and **Open** the operations.

TextHistogram

`TextHistogram [flags] `*`srcTextWave`*

The TextHistogram operation computes the histogram of a text wave where the output bins represent the count of occurrences of each unique string found in *srcTextWave*.

The TextHistogram operation was added in Igor Pro 9.00.

Flags

/CI	Performs case-insensitive string comparison. If you omit /CI TextHistogram performs case-sensitive comparison unless you include /LOC.
/DN=*binsCountsWave*	Specifies the numeric output wave that contains the count for each bin. If you omit /DN, the numeric output wave is created in the current data folder and named W_TextHistogram.
/DT=*binsTextWave*	Specifies the text output wave that contains the strings corresponding to each bin. If you omit /DT, the text output wave is created in the current data folder and named T_TextHistogram.
/FREE	Creates all output waves as a free waves. /FREE is permitted in user-defined functions only. If you use /FREE then all output wave parameters must be simple names, not paths or $ expressions. See **Free Waves** on page IV-91 for details on free waves.
/LOC	Performs case-insensitive string comparison following locale-aware rules. This option results in significantly slower performance.
/SORT=*mode*	Sets the order of the output bins. *mode*=0: The output waves are ordered from the bin with the largest count to the bin with the smallest count. This is the default if you omit /SORT. *mode*=1: The output waves are ordered from the bin with the longest string to the bin with the shortest string.
/Z	Suppress errors. You can use V_Flag to detect and handle errors yourself.

Details

TextHistogram does case-sensitive string comparisons unless you specify /CI or /LOC in which case it does case-insensitive string comparisons.

TextHistogram scans *srcTextWave* and counts matching string entries. It treats text waves of all dimensions as if they were 1D waves.

The output of the operation consists of two waves: a text wave containing the text corresponding to each bin and a numeric wave containing the bin counts. You can specify the destination waves using the /DT and /DN flags. If you omit /DT the text output wave is created in the current data folder and named T_TextHistogram. If you omit /DN the numeric output wave is created in the current data folder and named W_TextHistogram. The output waves overwrite any previously-existiing waves with the same names.

The operation creates automatic wave references for the output waves specified by /DT and /DN. See **Automatic Creation of WAVE References** on page IV-72 for details. It does not create wave references for the default output waves created if you omit /DT or /DN so you need explicit wave references.

When using case-insensitive mode the output text wave may use any one of the equivalent forms of the text represented by the bin.

Output Variables

TextHistogram sets the following output variable:

V_flag	0 if the operation succeeded or a non-zero error code.

See Also

Histogram, **FindDuplicates**, **Sort**