StringMatch

`StringMatch(`*`string`*`, `*`matchStr`*`)`

The StringMatch function tests *string* for a match to *matchStr*. You may include asterisks in *matchStr* as a wildcard character.

StringMatch returns 1 to indicate a match, 0 for no match or NaN if it ran out of memory.

Details

matchStr is some combination of normal characters and the asterisk wildcard character that matches anything. For example:

"*"	Matches any string.
"xyz"	Matches the string “xyz” only.
"*xyz"	Matches strings ending with “xyz”, for instance “abcxyz”.
"xyz*"	Matches strings beginning with xyz, for instance “xyzpqr”.
"*xyz*"	Matches strings containing xyz, for instance “abcxyzpqr”.
"abc*xyz"	Matches strings beginning with abc and ending with xyz, for instance “abcpqrxyz”.

If *matchStr* begins with the ! character, a match is indicated if *string* does *not* match *matchStr*. For example:

"!*xyz"	Matches strings which *do not* end with xyz.

The ! character is considered to be a normal character if it appears anywhere else.

Note that matching is case-insensitive, so “xyz” also matches “XYZ” or “Xyz”.

Also note that it is impossible to match an asterisk in *string*: use **GrepString** instead.

Among other uses, the StringMatch function can be used to build your own versions of the **WaveList** function, using **NameOfWave** and stringmatch to qualify names of waves found by **WaveRefIndexedDFR**.

See Also

The **GrepString**, **CmpStr**, **strsearch**, **Demo**, **ListMatch**, and **ReplaceString** functions and the **sscanf** operation.

StringToUnsignedByteWave

`StringToUnsignedByteWave(`*`str`*`)`

The StringToUnsignedByteWave function returns a free unsigned byte wave containing the contents of *str*.

If *str* is an empty string, a zero point wave is returned. If *str* is null, a null wave reference is returned.

The StringToUnsignedByteWave function was added in Igor Pro 9.00.

Parameters

str is a string.

Details

The StringToUnsignedByteWave function returns a free wave so it can't be used on the command line or in a macro. If you need to convert the free wave to a global wave use **MoveWave**.

Using StringToUnsignedByteWave makes it possible to use commands that work on waves to manipulate the data originally stored in a string. This can be faster and more convenient than manipulating the data directly in the string. If necessary, you can create a string containing the manipulated data using WaveDataToString.

StringToUnsignedByteWave stores each byte of *str* in an element of the output wave. It does not do ASCII-to-binary conversion. For example, if *str* contains "123", it returns an unsigned binary wave containing three elements with values 49, 50, and 51 (0x31, 0x32, and 0x33). It does not return the numeric value 123.

Example

```
Function DemoStringToUnsignedByteWave()
    String theStr = "123"
```