If you use a path or a $ expression, Igor does not create an automatic SVAR reference. You can explicitly create SVARs like this:

```
String/G root:sVar2
SVAR sVar2 = root:sVar2          // Creates an SVAR named sVar2

String path = "root:sVar3"
String/G $path
SVAR sVar3 = $path               // Creates an SVAR named sVar3
```

In Igor Pro 8.00 and later, you can explicitly create an SVAR reference in a user-defined function using the /N flag, like this:

```
String/G sVar4/N=sVar4           // Creates an SVAR named sVar4

String/G root:sVar5/N=sVar5      // Creates an SVAR named sVar5

String path = "root:sVar6"
String/G $path/N=sVar6           // Creates an SVAR named sVar6
```

The name used for the SVAR does not need to be the same as the name of the global variable:

```
String/G sVar7/N=sv7             // Creates an SVAR named sv7

String/G root:sVar8/N=sv8        // Creates an SVAR named sv8
String path = "root:sVar9"

String/G $path/N=sv9             // Creates an SVAR named sv9
```

See Also

String Variables on page II-105, **Working With Strings** on page IV-13, **Accessing Global Variables and Waves** on page IV-65

StringByKey

`StringByKey(`*`keyStr`*`, `*`kwListStr`*` [, `*`keySepStr`*` [, `*`listSepStr`*` [, `*`matchCase`*`]]])`

The StringByKey function returns a substring extracted from *kwListStr* based on the specified key contained in *keyStr*. *kwListStr* should contain keyword-value pairs such as `"KEY=value1,KEY2=value2"` or `"Key:value1;KEY2:value2"`, depending on the values for *keySepStr* and *listSepStr*.

Use StringByKey to extract a string value from a string containing a `"key1:value1;key2: value2;"` style list such as those returned by functions like **AxisInfo** or **TraceInfo**.

If the key is not found or if any of the arguments is `""` then a zero-length string is returned.

keySepStr, *listSepStr,* and *matchCase* are optional; their defaults are ":", ";", and 0 respectively.

Details

kwListStr is searched for an instance of the key string bound by *listSepStr* on the left and a *keySepStr* on the right. The text up to the next *listSepStr* is returned.

kwListStr is treated as if it ends with a *listSepStr* even if it doesn't.

Searches for *keySepStr* and *listSepStr* are always case-sensitive. Searches for *keyStr* in *kwListStr* are usually case-insensitive. Setting the optional *matchCase* parameter to 1 makes the comparisons case sensitive.

In Igor6, only the first byte of *keySepStr* and *listSepStr* was used. In Igor7 and later, all bytes are used.

If *listSepStr* is specified, then *keySepStr* must also be specified. If *matchCase* is specified, *keySepStr* and *listSepStr* must be specified.

Examples

```
Print StringByKey("BKEY", "AKEY:hello;BKEY:nok-nok")  // prints "nok-nok"
Print StringByKey("KY", "KX=1;ky=hello", "=")         // prints "hello"
Print StringByKey("KY", "KX:1,KY:joey,", ":", ",")    // prints "joey"
Print StringByKey("kz", "KZ:1st,kz:2nd,", ":", ",")   // prints "1st"
Print StringByKey("kz", "KZ:1st,kz:2nd,", ":", ",", 1)// prints "2nd"
```

See Also

The **NumberByKey**, **RemoveByKey**, **ReplaceNumberByKey**, **ReplaceStringByKey**, **ItemsInList**, **AxisInfo**, **IgorInfo**, **SetWindow**, and **TraceInfo** functions.