Variable

Variable [***flags***] ***varName***[**/N=*name***][**=*numExpr***][**, *varName***[**/N=*name***][**=*numExpr***]]…

The Variable operation creates real or complex variables and gives them the specified name.

Flags

/C	Declares a complex variable.
/D	Obsolete, included only for backward compatibility (see **Details**).
/G	Creates a variable with global scope and overwrites any existing variable.
/N=*name*	Specifies a local name for the global string variable. /N was added in Igor Pro 8.00 and is available in user-defined functions only. See **NVAR Creation** below for details.

Details

The variable is initialized when it is created if you supply the initial value. However, when Variable is used to declare a function parameter, it is an error to attempt to initialize it.

You can create more than one variable at a time by separating the names and optional initializers for multiple variables with a comma.

Numeric variables are double precision. In ancient times, variables could be single or double precision and the /D flag meant double precision. The /D flag is allowed for backward compatibility but is no longer needed and should not be used in new code.

If used in a macro or function the new variable is local to that macro or function unless the /G flag is used. If used on the command line, the new variable is global.

varName can include a data folder path.

NVAR Creation

In a user-defined function, you need a local NVAR reference to access a global string variable. If you use a simple name rather than a path, Igor automatically creates an NVAR:

```
Variable/G nVar1                    // Creates an NVAR named nVar1
```

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

```
Variable/G root:nVar2
NVAR nVar2 = root:nVar2             // Creates an NVAR named nVar2

String path = "root:nVar3"
Variable/G $path
NVAR nVar3 = $path                  // Creates an NVAR named nVar3
```

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

```
Variable/G nVar4/N=nVar4            // Creates an NVAR named nVar4
Variable/G root:nVar5/N=nVar5       // Creates an NVAR named nVar5

String path = "root:nVar6"
Variable/G $path/N=nVar6            // Creates an NVAR named nVar6
```

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

```
Variable/G nVar7/N=nv7              // Creates an NVAR named nv7
Variable/G root:nVar8/N=nv8         // Creates an NVAR named nv8

String path = "root:nVar9"
Variable/G $path/N=nv9              // Creates an NVAR named nv9
```

Examples

To initialize a complex variable, use the **cmplx** function. For example:

```
Variable/C cv1 = cmplx(1,2)
```

This sets the real part of cv1 to 1 and the imaginary part to 2.

See Also

Numeric Variables on page II-104, **Accessing Global Variables and Waves** on page IV-65