Structure

Structure *structureName*
    memType memName [***arraySize***] [**,** ***memName*** [***arraySize***]]
    …
EndStructure

The Structure keyword introduces a structure definition in a user function. Within the body of the structure you declare the member type (*memType*) and the corresponding member name(s) (*memName*). Each *memName* may be declared with an optional array size.

Details

Structure member types (*memType*) can be any of the following Igor objects: Variable, String, WAVE, NVAR, SVAR, DFREF, FUNCREF, or STRUCT.

Igor structures also support additional member types, as given in the next table, for compatibility with C programming structures and disk files.

Igor Member Type	C Equivalent	Size	Note
char	signed 8-bit int	1 byte	
uchar	unsigned 8-bit int	1 byte	
int16	signed 16-bit int	2 bytes	
uint16	unsigned 16-bit int	2 bytes	
int32	signed 32-bit int	4 bytes	
uint32	unsigned 32-bit int	4 bytes	
int64	signed 64-bit int	8 bytes	Requires Igor Pro 7.00 or later
uint64	unsigned 64-bit int	8 bytes	Requires Igor Pro 7.00 or later
float	float	4 bytes	
double	double	8 bytes	

The Variable and double types are identical although Variable can be also specified as complex (using the /C flag).

Each structure member may have an optional *arraySize* specification, which gives the number of elements contained by the structure member. The array size is an integer number from 1 to 400 except for members of type STRUCT for which the upper limit is 100.

See Also

Structures in Functions on page IV-99 for further information.

See the **STRUCT** declaration for creating a local reference to a Structure.

StrVarOrDefault

StrVarOrDefault(*pathStr*, *defStrVal*)

The StrVarOrDefault function checks to see if *pathStr* points to a string variable and if so, it returns its value. If the string variable does not exist, returns *defStrVal* instead.

Details

StrVarOrDefault initializes input values of macros so they can remember their state without needing global variables to be defined first. Numeric variables use the corresponding numeric function, **NumVarOrDefault**.

Examples

```
Macro foo(nval,sval)
   Variable nval=NumVarOrDefault("root:Packages:mypack:nvalSav",2)
   String sval=StrVarOrDefault("root:Packages:mypack:svalSav","Hi")

   DFREF dfSav= GetDataFolderDFR()
   NewDataFolder/O/S root:Packages
```