```
Menu "Macros"
   "Smooth Wave In Graph...",SmoothWaveInGraphDialog()
End

Function SmoothWaveInGraphDialog()
   String traceName
   Prompt traceName,"Wave",popup,TraceNameList("",";",1)
   DoPrompt "Smooth Wave In Graph",traceName

   WAVE w = TraceNameToWaveRef("", traceName)
   Smooth 5, w
End
```

The traceName parameter alone is not sufficient to specify which wave we want to smooth because it does not identify in which data folder the wave resides. The TraceNameToWaveRef function returns a wave reference which solves this problem.

Saving Parameters for Reuse

It is possible to write a procedure that presents a simple input dialog with default values for the parameters saved from the last time it was invoked. To accomplish this, we use global variables to store the values between calls to the procedure. Here is an example that saves one numeric and one string variable.

```
Function TestDialog()
   String saveDF = GetDataFolder(1)
   NewDataFolder/O/S root:Packages
   NewDataFolder/O/S :TestDialog

   Variable num = NumVarOrDefault("gNum", 42)
   Prompt num, "Enter a number"
   String str = StrVarOrDefault("gStr", "Hello")
   Prompt str, "Enter a string"
   DoPrompt "test",num,str

   Variable/G gNum = num      // Save for next time
   String/G gStr = str

   // Put function body here
   Print num,str

   SetDataFolder saveDF
End
```

This example illustrates the NumVarOrDefault and StrVarOrDefault functions. These functions return the value of a global variable or a default value if the global variable does not exist. 42 is the default value for gNum. NumVarOrDefault returns 42 if gNum does not exist. If gNum does exist, it returns the value of gNum. Similarly, “Hello” is the default value for gStr. StrVarOrDefault returns “Hello” if gStr does not exist. If gStr does exist, it returns the value of gStr.

Multiple Simple Input Dialogs

Prompt statements can be located anywhere within the body of a function and they do not need to be grouped together, although it will aid code readability if associated Prompt and DoPrompt code is kept together. Functions may contain multiple DoPrompt statements, and Prompt statements can be reused or redefined.

The following example illustrates multiple simple input dialogs and prompt reuse:

```
Function Example()
   Variable a= 123
   Variable/C ca= cmplx(3,4)
   String s

   Prompt a,"Enter a value"
   Prompt ca,"Enter complex value"
```