```
   . . .

   SetDataFolder root:
   // The free data folder is deleted since there are no references to it.
   // jack is also deleted because there are no more references to it.

   . . .

End
```

Converting a Free Data Folder to a Global Data Folder

You can use **MoveDataFolder** to move a free data folder into the global hierarchy. The data folder and all of its contents then become global. The name of a free data folder created by NewFreeDataFolder is 'free-root'. You should rename it after moving to a global context. For example:

```
Function Test()
   DFREF saveDF = GetDataFolderDFR()
   DFREF dfr = NewFreeDataFolder()     // Create free data folder
   SetDataFolder dfr                   // Set as current data folder
   Make jack=sin(x/8)                  // Create some data in it
   SetDataFolder saveDF                // Restore original current data folder
   MoveDataFolder dfr, root:           // Free DF becomes root:freeroot
   RenameDataFolder root:freeroot,TestDF  // Rename with a proper name
   Display root:TestDF:jack
End
```

Note that MoveDataFolder requires that the data folder name, freeroot in this case, be unique within the destination data folder.

Structures in Functions

You can define structures in procedure files and use them in functions. Structures can be used only in user-defined functions as local variables and their behavior is defined almost entirely at compile time. Runtime or interactive definition and use of structures is not currently supported; for this purpose, use Data Folders (see Chapter II-8, **Data Folders**), the **StringByKey** function (see page V-997), or the **NumberByKey** function (see page V-714).

Use of structures is an advanced technique. If you are just starting with Igor programming, you may want to skip this section and come back to it later.

Simple Structure Example

Before we get into the details, here is a quick example showing how to define and use a structure.

```
Structure DemoStruct
   double dval
   int32 ival
   char str[100]
EndStructure

Function Subroutine(s)
   STRUCT DemoStruct &s       // Structure parameter

   Printf "dval=%g; ival=%d; str=\"%s\"\r", s.dval, s.ival, s.str
End

Function Routine()
   STRUCT DemoStruct s        // Local structure instance
   s.dval = 1.234
   s.ival = 4321
```