The IndependentModule Pragma

The IndependentModule pragma is a way for you to designate groups of one or more procedure files to be compiled and linked separately. Once compiled and linked, the code remains in place and is usable even though other procedures may fail to compile. This allows your control panels, menus, and procedures to continue to work regardless of user programming errors.

A file is designated as an independent module using

```
#pragma IndependentModule=imName
```

This is similar to `#pragma ModuleName=modName` (see **The ModuleName Pragma** on page IV-54) and, just as in the case of calling static functions in a procedure with #pragma ModuleName, calling nonstatic function in an IndependentModule from outside the module requires the use of *imName#functionName*() syntax.

To call any function, static or not, defined in an independent module from outside that module, you must qualify the call with the independent module name:

```
MyIndependentModule#Test()
```

For further discussion see **Independent Modules** on page IV-238.

The rtFunctionErrors Pragma

Normally, when an error occurs in a built-in function, the built-in function does not post an error but rather returns 0, NaN, or an empty string as the function result. For example, by default, the str2num built-in function in this user-defined function returns NaN but does not post an error:

```
Function Test()
   Variable val = str2num("abc")    // Returns NaN
   Print val
End
```

As a debugging aid, you can force Igor to post an error that occurs in a built-in function called from a user-defined function by adding this statement to the procedure file:

```
#pragma rtFunctionErrors = 1
```

Now, if you run the Test function, Igor displays an error dialog telling you that the str2num function expects a number.

The rtFunctionErrors pragma works for most errors in built-in functions called from user-defined functions, but not in all.

The rtFunctionErrors pragma was added in Igor Pro 7.00. Earlier versions of Igor ignore it.

The TextEncoding Pragma

The TextEncoding pragma provides a way for a programmer to mark an Igor procedure file as using a particular text encoding. This pragma was added in Igor Pro 7.00 and is ignored by earlier versions.

Here is an example:

```
#pragma TextEncoding = "UTF-8"
```

This pragma tells Igor7 or later that the procedure file is encoded using UTF-8, a form of Unicode. All new procedure files should use UTF-8.

Like all pragmas, the TextEncoding pragma must be flush against the left margin of the procedure file. It is an error to have more than one TextEncoding pragma in a given procedure file. The TextEncoding pragma can appear anywhere in the file and affects the whole file. By convention it is placed near the top of the file.

If your procedure file uses only ASCII characters then all versions of Igor on all platforms will interpret it correctly and there is no need for a TextEncoding pragma.