is not the same thing. In this case the note of wave_joe would contain the expression that myStringVar would depend on! Also, wave_joe would have to exist for Igor to understand the statement.

See Also

Chapter IV-9, **Dependencies**, and the **GetFormula** function.

SetIdlePeriod

```
SetIdlePeriod period
```

The SetIdlePeriod operation changes and reports the period of Igor's main idle loop. The units of *period* are milliseconds. Setting *period* to zero does not change the period.

SetIdlePeriod was added in Igor Pro 8.00.

The default idle period is 20 milliseconds. Setting the period higher may make a slight improvement in the performance of some computation-heavy tasks.

Setting the period lower can make some parts of Igor more responsive. In particular, background tasks can be made to run more often, as the minimum period between runs of a background task is determined by Igor's idle period. Reducing the period too far increases the likelihood that your background task will run too long. Setting the period to very low values can make Igor very sluggish.

Output Variables

SetIdlePeriod creates the output variable V_value and sets it to the idle period before the call was made. Yu can use this value to restore the idle period after temporarily changing it. If period is zero, the idle period is not changed, but the current value is returned in V_value.

See Also

Background Tasks on page IV-319

SetIgorHook

```
SetIgorHook [/K/L] [hookType = [procName]]
```

The SetIgorHook operation tells Igor to call a user-defined "hook" function at the following times:

- After procedures have been successfully compiled (**AfterCompiledHook**)
- After a file is opened (**AfterFileOpenHook**)
- After the MDI frame window is resized on Windows (**AfterMDIFrameSizedHook**)
- After a window is created (**AfterWindowCreatedHook**)
- Before the debugger is opened (**BeforeDebuggerOpensHook**)
- Before an experiment is saved (**BeforeExperimentSaveHook**)
- Before a file is opened (**BeforeFileOpenHook**)
- Before a new experiment is opened (**IgorBeforeNewHook**)
- Before Igor quits (**IgorBeforeQuitHook**)
- When a menu item is selected (**IgorMenuHook**)
- During Igor's quit processing (**IgorQuitHook**)
- When Igor starts or a new experiment is created (**IgorStartOrNewHook**)

The term “hook” is used as in the phrase “to hook into”, meaning to intercept or to attach.

Hook functions are typically used by a sophisticated procedure package to make sure that the package's private data is consistent.

In addition to using SetIgorHook, you can designate hook functions using fixed function names (see **User-Defined Hook Functions** on page IV-280). The advantage of using SetIgorHook over fixed hook names is that you don't have to worry about name conflicts.

You can designate hook functions for specific windows using window hooks (see **SetWindow** on page V-865).

Flags

/K Removes *procName* from the list of functions called for the *hookType* events.

If *procName* is not specified all *hookType* functions are removed.

If *hookType* is not specified all functions are removed for all *hookType* events, returning Igor to the pre-SetIgorHook state.

/L Executes *procName* last. Without /L, a newly added hook function runs before previously registered hook functions.

A function that has been previously registered with SetIgorHook can be moved from being called first to being called last by calling SetIgorHook again with /L.

To move a function from being called last to being called first requires removing the hook function with /K and then calling SetIgorHook without /L.

Parameters

hookType Specifies one of the fixed-name hook function names:

AfterCompiledHook

AfterFileOpenHook

AfterMDIFrameSizedHook

AfterWindowCreatedHook

BeforeDebuggerOpensHook

BeforeExperimentSaveHook

BeforeFileOpenHook

IgorBeforeNewHook

IgorBeforeQuitHook

IgorMenuHook

IgorQuitHook

IgorStartOrNewHook

See the note below about these *hookType* names.

hookType is required except with /K.

procName Names the user-defined hook function that is called for the *hookType* event.

Details

The parameters and return type of the user-defined function *procName* varies depending on the *hookType* it is registered for.

For example, a function registered for the AfterFileOpenHook type must have the same parameters and return type as the shown for the **AfterFileOpenHook** on page IV-282.

The *procName* function is called *after* any window-specific hook for these *hookType*s, and the *procName* function is called *before* any other hook functions previously registered by calling SetIgorHook *unless the /L flag is given*, in which case it still runs after window-specific hook functions, but also *after* all other previously registered hook functions.

The *procName* function should return a nonzero value (1 is typical) to prevent later functions from being called. Returning 0 allows successive functions to be called.

SetIgorHook does not work at Igor start or new experiment time, so SetIgorHook IgorStartOrNewHook is disallowed. Define a global or static fixed-name **IgorStartOrNewHook** function (see page IV-292).

The saved Igor experiment file remembers the SetIgorHooks that are in effect when the experiment is saved:

Hook Function Interactions

After all the SetIgorHook functions registered for *hookType* have run (and all have returned 0), any static fixed-name hook functions are called and then the (only) fixed-name user-defined hook function, if any, is called. As an example, when a menu event occurs, Igor handles the event by calling routines in this order:

1. The top window's hook function as set by **SetWindow**
2. Any SetIgorHook-registered hook functions
3. Any static fixed-named IgorMenuHook functions (in any independent module)
4. The one-and-only non-static fixed-named IgorMenuHook function (in only the ProcGlobal independent module)

1. SetWindow event	**2. SetIgorHook *hookType***	**3. User-defined Hook Function(s)**
(called first)	**(called second)**	**(called last)**
enableMenu	IgorMenuHook	IgorMenuHook
menu	IgorMenuHook	IgorMenuHook

Note: Although you can technically use one of the fixed-name functions, as described in **User-Defined Hook Functions** on page IV-280, for *procName*, the result would be that the function will be called twice: once as a registered named hook function and once as the fixed-named hook function. That is, don’t use SetIgorHook this way:

```
SetIgorHook AfterFileOpenHook=AfterFileOpenHook // NO
```

Variables

SetIgorHook returns information in the following variables:

`S_info` Semicolon-separated list of all current hook functions associated with *hookType*, listed in the order in which they are called. S_info includes the full independent module paths (e.g.,"ProcGlobal#MyMenuHook;MyIM#MyModule#MyMenuHook;").

Examples

This hook function invokes the Export Graphics menu item when Command-C (*Macintosh*) or Ctrl+C (*Windows*) is selected for a graph, preventing the usual Copy.

```
SetIgorHook IgorMenuHook=CopyIsExportHook
```

```
Function CopyIsExportHook(isSelection,menuName,itemName,itemNo,win,wType)
   Variable isSelection
   String menuName,itemName
   Variable itemNo
   String win
   Variable wType

   Variable handledIt= 0
   if( isSelection && wType==1 ) // menu was selected, window is graph
      if( Cmpstr(menuName,"Edit")==0 && CmpStr(itemName,"Copy")==0 )
         DoIgorMenu "Edit", "Export Graphics"          // dialog instead
         handledIt= 1             // don't call other IgorMenuHook functions.
      endif
   endif
   return handledIt
End
```

To unregister CopyIsExportHook as a hook procedure:

```
SetIgorHook/K IgorMenuHook=CopyIsExportHook // unregister CopyIsExportHook
```

To discover which functions are associated with a *hookType*, use a command such as:

```
SetIgorHook IgorMenuHook   // inquire about names registered for IgorMenuHook
Print S_info               // list of functions
```

To remove (or “unregister”) named hooks:

```
SetIgorHook/K                 // removes all hook functions for all hookTypes
SetIgorHook/K IgorMenuHook    // removes all IgorMenuHook functions
SetIgorHook/K IgorMenuHook=CopyIsExportHook// removes only this hook function
```