You can test the function with this command:

```
ExecuteUnixShellCommand("ls", 1, 1)
```

Life is a bit more complicated if the command that you want to execute contains spaces or other nonstandard Unix command characters. For example, imagine that you want to execute this:

```
ls /System/Library/Image Capture
```

These commands will not work because of the space in the command's file path:

```
String unixCmd = "ls /System/Library/Image Capture"
ExecuteUnixShellCommand(unixCmd, 1, 1)
```

You need to quote the path part of the command so that the UNIX parses:

```
/System/Library/Image Capture
```

as one argument, not two (/System/Library/Image and ./Capture).

There are two ways to do this:

1. Use single-quote characters around the parts of the command where needed:

   ```
   String unixCmd = "ls '/System/Library/Image Capture'"
   ExecuteUnixShellCommand(unixCmd, 1, 1)
   ```

2. Use double-quote characters with the backslashes needed to make it through Igor's parser and AppleScript's parser:

   ```
   String unixCmd = "ls \\\"/System/Library/Image Capture\\\""
   ExecuteUnixShellCommand(unixCmd, 1, 1)
   ```

   Igor's parser converts \\ to \ and \" to ", so AppleScript sees this:

   ```
   "ls \"/System/Library/Image Capture\""
   ```

   AppleScript's parser converts \" to " so Unix sees this:

   ```
   ls "/System/Library/Image Capture"
   ```

ActiveX Automation

ActiveX Automation, often called just Automation, is Microsoft's technology for allowing one program to control another. The program that does the controlling is called the Automation client. The program that is controlled is called the Automation Server. The client initiates things by making calls to the server which carries out the requested actions and returns results.

Automation client programs are most often written in Visual Basic or C#. They can also be written in C++ and other programming languages, and in various scripting languages such as VBScript, JavaScript, Perl, Python and so on.

Igor can play the role of Automation Server. If you want to write an client program to drive Igor, see "Automation Server Overview" in the "Automation Server" help file in "\Igor Pro Folder\Miscellaneous\Windows Automation".

Igor Pro does not directly support playing the role of Automation client. However, it is possible to write an Igor program which generates a script file which can act like an Automation client. For an example, choose File→Example Experiments→Programming→CallMicrosoftWord.

Calling Igor from Scripts

You can call Igor from shell scripts, batch files, Apple Script, and the Macintosh terminal window using an operation-like syntax. You can also use this feature to register an Igor license.

The syntax for calling Igor is:

```
<IGOR> [/I /Q /X /N /Automation]  [pathToFileOrCommands] [pathToFile] ...

<IGOR> [/I /N /Automation]  [pathToFileOrCommands] [pathToFile] ...

<IGOR> [/I /Q /X /Automation] "commands"

<IGOR> /SN=num /KEY="key" /NAME="name" [/ORG="org" /QUIT]
```

where `<IGOR>` is the full path to the Igor executable file.

On Windows, the Igor executable file resides in a folder within the Igor Pro folder. The full path will be something like:

```
"C:\Program Files\WaveMetrics\IgorBinaries_x64\Igor Pro Folder\Igor64.exe"
```

On Macintosh, the Igor application is an "application bundle" and the actual executable file is inside the bundle. The full path will be something like:

```
'/Applications/Igor Pro Folder/Igor64.app/Contents/MacOS/Igor64'
```

In the following discussions, `<IGOR>` means "the full path to the Igor executable file".

Parameters

All parameters are optional. If you omit all parameters, including just the full path to the Igor executable, a new instance of Igor is launched.

The usual parameter is a file for Igor to open. It is recommended that both the path and the path to the file parameter be enclosed in quotes.

You can open multiple files by using a space between one quoted file path and the next.

With the /X flag, only one parameter is allowed and it is interpreted as an Igor command.

Flags

When you specify a flag, you can use a - instead of /. For example, you can write /Q or -Q.

/Automation	This flag is supported on Windows only. The Windows OS uses it when launching Igor Pro as an ActiveX Automation server. It is not intended for use in batch files.
/I	Launches a new instance of Igor if one would otherwise not be launched. See ***Details*** for a discussion of instances.
/KEY="*key*"	Specifies the license activation key when registering a license. For example: `/KEY="ABCD-EFGH-IJKL-MNOP-QRST-UVWX-Y"` Do not omit the quotes, or it will fail.
/N	Forces the current experiment to be closed without saving if any of the file parameters are an experiment file. To save a currently open experiment, use: `<IGOR> /X "SaveExperiment"`
/NAME="*name*"	Specifies the name of the licensed user when registering a license. A name is required when registering.
/ORG="*org*"	Specifies the optional name of the licensed organization when registering a license. /ORG is optional and defaults to "".
/Q	Prevents the command from being displayed in Igor's command line as it is executing.
/QUIT	Quits Igor Pro after entering license information when used with /SN, /KEY, and /NAME. Otherwise /QUIT is ignored. To quit Igor Pro, use: `<IGOR> /X "Quit/N"`
/SN=*num*	Specifies the license serial number when registering a license.