Subject: Re: Can one close only one of STDIN and STDOUT when using SPAWN with the UNIT keyword?
Posted by bstecklu on Wed, 27 Mar 2013 13:27:28 GMT
View Forum Message <> Reply to Message

Tom Grydeland wrote:
> Hi all,
>
> Some commands only produce output after all of the input has been consumed.
> For such a command, the usual approach when given input and output streams is
> to open both, feed the process all its input, close the input stream and then
> read the result on the output stream.
>
> In IDL, when using SPAWN with the UNIT keyword, one file unit is supposed to
> be used for both directions. Is it possible to do the above from IDL?
>
> For example, if the external command is a checksum program, like 'md5sum':
>
> IDL> SPAWN, 'md5sum', UNIT=unit
> IDL> WRITEU, unit, 'foo'
> > IDL> {close unit in the input-to-md5sum direction}
> IDL> READU, unit, checksum
> IDL> {proceed with checksum}
>
> Can the above be done from IDL? How?
>
> --T (tom.grydeland@norut.no)

May be I am missing your point but

IDL> WRITEU, unit, 'foo' & SPAWN, 'md5sum foo', checksum

will give you the result in the string (array) checksum.

Bringfried

Subject: Re: Can one close only one of STDIN and STDOUT when using SPAWN with the UNIT keyword?
Posted by Craig Markwardt on Wed, 27 Mar 2013 19:02:08 GMT
View Forum Message <> Reply to Message

On Tuesday, March 26, 2013 6:30:04 PM UTC-4, Tom Grydeland wrote:
> Hi all,
>
> Some commands only produce output after all of the input has been consumed. For such a command, the usual approach when given input and output streams is to open both, feed the process all its input, close the input stream and then read the result on the output stream.

>
> In IDL, when using SPAWN with the UNIT keyword, one file unit is supposed to be used for both directions. Is it possible to do the above from IDL?

This is the standard race condition that exists for bidirectional pipes. I don't think you can work around it with IDL. You might be better off redirecting the output of your program to a file, and then reading the file when the SPAWN'd process is complete.

Craig

Subject: Re: Can one close only one of STDIN and STDOUT when using SPAWN with the UNIT keyword?
Posted by tom.grydeland on Sun, 31 Mar 2013 09:35:08 GMT
View Forum Message <> Reply to Message

> On Tuesday, March 26, 2013 6:30:04 PM UTC-4, Tom Grydeland wrote:
>> In IDL, when using SPAWN with the UNIT keyword, one file unit is supposed to be used for both directions. Is it possible to do the above from IDL?

On Wednesday, March 27, 2013 8:02:08 PM UTC+1, Craig Markwardt wrote:
> This is the standard race condition that exists for bidirectional pipes. I don't think you can work around it with IDL.

The race condition I believe you refer to occurs when both processes wait for input from the other side. In the case of a reducing utility (such as 'wc' or 'md5sum'), there is no race if one is able to signal the end of input to the other side (i.e. CLOSE its input, as I asked about originally).

This simply begs the question: Why use one bidirectional pipe instead of separate pipes for stdin and stdout (and stderr, if so desired)? This approach creates problems, and appears not to solve any.

> You might be better off redirecting the output of your program to a file, and then reading the file when the SPAWN'd process is complete.

I understand workarounds. I program in IDL, so I eat, drink, dream and breathe workarounds. It is just that IDL would be so much nicer to program in if one didn't have to expend so much of the creative energy on workarounds.

There are legitimate resons why one might not want to use temporary files for all kinds of tasks, and the UNIT keyword to the SPAWN command seems to have been provided to cater for this demand. Only it doesn't.

> Craig

Possible language changes that would solve the issue:

1) Two or three new keywords to SPAWN that are used to return separate UNITs for the three

standard streams
2) A way to close one direction only of a bidirectional pipe.
3) New routine(s), along the lines of those in the popen2 module (Python) or Open3 module (Ruby).

Cheers,

--T
