Subject: Re: Command line arguments
Posted by Craig Markwardt on Fri, 11 Feb 2005 20:46:49 GMT
View Forum Message <> Reply to Message

"Mr. No Address" <no_given_address@landofthelost.net> writes:
> Way back in the day I posted a very similar question. Someone gave me a
> solution using environment variables that has worked fine all this
> time, but now I'm updating things and I'm seeking a cleaner solution.
> Here we go...
>
> I have a Perl program that I compile and run an IDL program from. I'm
> now looping through several hundred times and it seems wasteful to
> compile the IDL program every loop.
>
> # The relevant Perl code...
> $ENV{"file"}="$filepath";
> open(IDL, "|/usr/local/bin/idl") || die "Can't open IDL: $!";
> print IDL ".Compile mentor \n";
> print IDL "mentor \n";
> close IDL;
>
> # The relevant IDL code...
> PRO mentor
> file=GETENV('file')
>
> I would like to compile the program once outside the loop and then pass
> the file using an argument instead grabbing the file from an environment
> variable. Can I do something like this?
>
> print IDL "mentor, $filepath\n";

You need to put quotation marks around it, as in:

```
print IDL "mentor, '$filepath'\n";
```

That has worked for me in the past.

Good luck!
Craig

--
---------- ----------
Craig B. Markwardt, Ph.D. EMAIL: craigmnet@REMOVEcow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
---------- ----------

Subject: Re: Command line arguments
Posted by Michael Wallace on Fri, 11 Feb 2005 21:17:59 GMT
View Forum Message <> Reply to Message

> I would like to compile the program once outside the loop and then pass
> the file using an argument instead grabbing the file from an environment
> variable. Can I do something like this?
>
> print IDL "mentor, $filepath\n";

Yes.

Maybe I don't understand what you're doing, but why are you 1.) explicitly compiling the command your going to run instead of letting IDL automatically compile it? and 2.) why don't you write a wrapper in IDL itself that handles the looping and you just call the wrapper the one time instead of calling the same IDL command multiple times?

My Perl is rusty, but a while back I wrote a Python wrapper to simulate command line arguments. I think I posted it in the newsgroup some time ago, but I couldn't find the old thread.

The first argument to the program below is the name of the IDL command to execute. The remainder of the arguments will be fed to the IDL command named in the first argument. I have defined my IDL programs that need to be run on the command line to include the keyword ARGS. The additional arguments are passed into the IDL command using this keyword. By doing this, I can use this one wrapper for all command line programs, but the argument list is not bound to something specific.

If I want to quickly do something and the program I want to run doesn't have an ARGS keyword, I can just put the entire IDL command I want to run in the first argument. Just quote the entire command so that it's interpreted as a single argument.

-Mike

```
#!/usr/bin/env python

import os
import sys

# Usage statement
usage = "usage: %s idlprog args" %os.path.basename(sys.argv[0])

# Check that the name of the IDL program was provided
if len(sys.argv) < 2:
```

```
    print usage
else:
    fd = os.popen('idl', 'w')

    # If extra arguments are given, pass them via the ARGS keyword
    if len(sys.argv) < 3:
        fd.write(sys.argv[1])
    else:
        fd.write(sys.argv[1] + ', ARGS = ' + `sys.argv[2:]`)

    fd.close()
```

Subject: Re: Command line arguments
Posted by Mr. No Address on Fri, 11 Feb 2005 21:49:22 GMT
View Forum Message <> Reply to Message

Craig Markwardt wrote:
> "Mr. No Address" <no_given_address@landofthelost.net> writes:
>
>> Way back in the day I posted a very similar question. Someone gave me a
>> solution using environment variables that has worked fine all this
>> time, but now I'm updating things and I'm seeking a cleaner solution.
>> Here we go...
>>
>> I have a Perl program that I compile and run an IDL program from. I'm
>> now looping through several hundred times and it seems wasteful to
>> compile the IDL program every loop.
>>
>> # The relevant Perl code...
>> $ENV{"file"}="$filepath";
>> open(IDL, "|/usr/local/bin/idl") || die "Can't open IDL: $!";
>> print IDL ".Compile mentor \n";
>> print IDL "mentor \n";
>> close IDL;
>>
>> # The relevant IDL code...
>> PRO mentor
>> file=GETENV('file')
>>
>> I would like to compile the program once outside the loop and then pass
>> the file using an argument instead grabbing the file from an environment
>> variable. Can I do something like this?
>>
>> print IDL "mentor, $filepath\n";
>
>
> You need to put quotation marks around it, as in:

>
> print IDL "mentor, '$filepath'\n";
>
> That has worked for me in the past.

Thanks for the reply Craig. How do I assign $filepath to a variable in
the IDL code?

Cheers,
Gary

Subject: Re: Command line arguments
Posted by Mr. No Address on Fri, 11 Feb 2005 22:11:24 GMT
View Forum Message <> Reply to Message

Mr. No Address wrote:
> Craig Markwardt wrote:
>
>> "Mr. No Address" <no_given_address@landofthelost.net> writes:
>>
>>> Way back in the day I posted a very similar question. Someone gave me a
>>> solution using environment variables that has worked fine all this
>>> time, but now I'm updating things and I'm seeking a cleaner solution.
>>> Here we go...
>>>
>>> I have a Perl program that I compile and run an IDL program from. I'm
>>> now looping through several hundred times and it seems wasteful to
>>> compile the IDL program every loop.
>>>
>>> # The relevant Perl code...
>>> $ENV{"file"}="$filepath";
>>> open(IDL, "|/usr/local/bin/idl") || die "Can't open IDL: $!";
>>> print IDL ".Compile mentor \n";
>>> print IDL "mentor \n";
>>> close IDL;
>>>
>>> # The relevant IDL code...
>>> PRO mentor
>>> file=GETENV('file')
>>>
>>> I would like to compile the program once outside the loop and then pass
>>> the file using an argument instead grabbing the file from an environment
>>> variable. Can I do something like this?
>>>
>>> print IDL "mentor, $filepath\n";
>>
>>

>>
>> You need to put quotation marks around it, as in:
>>
>> print IDL "mentor, '$filepath'\n";
>>
>> That has worked for me in the past.
>
>
> Thanks for the reply Craig. How do I assign $filepath to a variable in
> the IDL code?

OK, I figured out this part of it. I changed the first line in the IDL
file to:
PRO mentor, file ; I added "file"

Now I seem to have a Perl problem in that I think Perl is executing the
lines after

print IDL "mentor, '$filepath'\n";

before it writes the idl.ps file. Seems I ran into this way back when too.

Thanks,
Gary

Subject: Re: Command line arguments
Posted by Mr. No Address on Fri, 11 Feb 2005 22:19:30 GMT
View Forum Message <> Reply to Message

Michael Wallace wrote:
>> I would like to compile the program once outside the loop and then pass
>> the file using an argument instead grabbing the file from an environment
>> variable. Can I do something like this?
>>
>> print IDL "mentor, $filepath\n";
>
>
> Yes.
>
> Maybe I don't understand what you're doing, but why are you 1.)
> explicitly compiling the command your going to run instead of letting
> IDL automatically compile it? and 2.) why don't you write a wrapper in
> IDL itself that handles the looping and you just call the wrapper the
> one time instead of calling the same IDL command multiple times?

Thanks for the reply. With your post and Craig's above I was able to
piece together what I was doing wrong. Still have problems, but I think

that scope is now Perl. In any case, to address your questions above...
If I understand your first question above correctly, I'm compiling first because "Way back in the day" I was never able to successfully run code without compiling it first. I'm sure I was doing something wrong, but I got things to work this way and went with it. On #2, This time around I set out to write everything in IDL, but knowing a bit of Perl and not really knowing IDL, it was too easy to fall back on my Perl skills. Most of that had to do with the way Perl handles time, e.g., timegm and gmtime.

Cheers,
Gary

> My Perl is rusty, but a while back I wrote a Python wrapper to simulate
> command line arguments. I think I posted it in the newsgroup some time
> ago, but I couldn't find the old thread.
>
> The first argument to the program below is the name of the IDL command
> to execute. The remainder of the arguments will be fed to the IDL
> command named in the first argument. I have defined my IDL programs
> that need to be run on the command line to include the keyword ARGS. The
> additional arguments are passed into the IDL command using this
> keyword. By doing this, I can use this one wrapper for all command line
> programs, but the argument list is not bound to something specific.
>
> If I want to quickly do something and the program I want to run doesn't
> have an ARGS keyword, I can just put the entire IDL command I want to
> run in the first argument. Just quote the entire command so that it's
> interpreted as a single argument.
>
> -Mike
>
>
> #!/usr/bin/env python
>
> import os
> import sys
>
> # Usage statement
> usage = "usage: %s idlprog args" %os.path.basename(sys.argv[0])
>
> # Check that the name of the IDL program was provided
> if len(sys.argv) < 2:
> print usage
> else:
> fd = os.popen('idl', 'w')
>

```
>     # If extra arguments are given, pass them via the ARGS keyword
>     if len(sys.argv) < 3:
>        fd.write(sys.argv[1])
>     else:
>        fd.write(sys.argv[1] + ', ARGS = ' + `sys.argv[2:]`)
>
>     fd.close()
```

Subject: Re: Command line arguments
Posted by Craig Markwardt on Fri, 11 Feb 2005 22:52:14 GMT
View Forum Message <> Reply to Message

"Mr. No Address" <no_given_address@landofthelost.net> writes:
...
> PRO mentor, file ; I added "file"
>
> Now I seem to have a Perl problem in that I think Perl is executing
> the lines after
>
> print IDL "mentor, '$filepath'\n";
>
> before it writes the idl.ps file. Seems I ran into this way back when too.
...

Yes, that's true. Perl will not wait for IDL, it just stuffs
characters into a pipe. You will have to make some other kind of
locking mechanism. Like having perl wait until a certain file is
created, etc.

Craig

--
---------- ----------
Craig B. Markwardt, Ph.D. EMAIL: craigmnet@REMOVEcow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
---------- ----------

Subject: Re: Command line arguments
Posted by David Fanning on Sat, 12 Feb 2005 02:42:40 GMT
View Forum Message <> Reply to Message

Mr. No Address writes:

> If I understand your first question above correctly, I'm compiling
> first because "Way back in the day" I was never able to successfully run

> code without compiling it first. I'm sure I was doing something wrong,

No doubt. :-)

This article might be useful to you:

http://www.dfanning.com/tips/namefiles.html

Cheers,

David
--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: Command line arguments
Posted by Mr. No Address on Thu, 17 Feb 2005 16:23:48 GMT
View Forum Message <> Reply to Message

David Fanning wrote:
> Mr. No Address writes:
>
>
>> If I understand your first question above correctly, I'm compiling
>> first because "Way back in the day" I was never able to successfully run
>> code without compiling it first. I'm sure I was doing something wrong,
>
>
> No doubt. :-)
>
> This article might be useful to you:
>
> http://www.dfanning.com/tips/namefiles.html

It was. Thanks. In fact, self compiling worked so easily that I was
left wondering what I was doing wrong way back when. With some local
help I also have solved the Perl/IDL passing args issue that I'll
describe below for the next person.

Key words for google searches:
passing command line arguments args Perl IDL script code execute
executing system command commands

There were two problems I had. One was simply passing an argument from
within a Perl script to my IDL code. The second was getting Perl to
wait for the IDL process to finish before executing subsequent lines of

code.

Perl Code:
open(IDL, "|/usr/local/bin/idl") || die "Can't open IDL: $!";
print IDL "idlprog, \'$arg_to_pass\' \n";
print IDL "\$mv idl.ps renamed_file.ps\n";
close IDL;

The middle two print lines can be within a loop so you don't have to
open an IDL process for every iteration. I had to have the IDL process
rename the postscript file instead of doing:
`mv idl.ps renamed_file.ps`
or else Perl would zip by the first print statement and then fail to
find the idl.ps file. See
<http://archive.stsci.edu/iue/manual/dacguide/node8.html> for executing
system commands from IDL.

IDL Code:
PRO idlprog, passed_arg

Subject: Re: Command line arguments
Posted by JD Smith on Mon, 21 Feb 2005 17:56:15 GMT
View Forum Message <> Reply to Message

On Fri, 11 Feb 2005 19:42:40 -0700, David Fanning wrote:

> Mr. No Address writes:
>
>> If I understand your first question above correctly, I'm compiling
>> first because "Way back in the day" I was never able to successfully run
>> code without compiling it first. I'm sure I was doing something wrong,
>
> No doubt. :-)
>
> This article might be useful to you:
>
> http://www.dfanning.com/tips/namefiles.html

You'd be surprised how many people I know compile their routines by
hand, not once, but *twice*, just for good measure. This is what a
lifetime of IRAF can do to a person.

JD
