Subject: Re: run-time function creation
Posted by PMan on Wed, 10 Apr 2013 14:47:52 GMT
View Forum Message <> Reply to Message

On Wednesday, April 10, 2013 10:33:31 AM UTC-4, Paul Mallas wrote:
> I know this is a bit off the wall, but I was wondering if there is a technique or method for doing run-time function creation.
>
>
>
> The reason I ask is I was looking at the IDL integration routines (e.g., qsimp) and this requires a singular argument function name as input. My problem is I have a function with several parameters I need to integrate, but don't have these parameters until I calculate them at run time. If I could somehow create the function dynamically, I could create a function that would satisfy the the qsimp requirement for a single argument function, but prior to run-time I can't.
>
>
>
> Anyway, I know this is a wacky and maybe even a silly question, but can't hurt to ask.
>
>
>
> Regards,
>
> Paul

Actually, I think I just answered my own question. I could use the "common" block to pass the other parameters, huh? I have thought about this problem for a day or two, and as soon as I post the question, I come up with a solution myself. I guess I can't think straight until I clutter up the mail boxes of others :)

If there are other suggestions, let me know.

Subject: Re: run-time function creation
Posted by Brian Daniel on Wed, 10 Apr 2013 14:55:45 GMT
View Forum Message <> Reply to Message

Another option would be to populate a structure full of the parameters needed and send that to the function. If the dimensions or data types change of the parameters you're specifying, then use a pointer to a structure. Lists or hashes may also be useful, but I have no experience with them.

-Brian

On Wednesday, April 10, 2013 10:33:31 AM UTC-4, Paul Mallas wrote:
> I know this is a bit off the wall, but I was wondering if there is a technique or method for doing run-time function creation.
>

>
>
> The reason I ask is I was looking at the IDL integration routines (e.g., qsimp) and this requires a singular argument function name as input. My problem is I have a function with several parameters I need to integrate, but don't have these parameters until I calculate them at run time. If I could somehow create the function dynamically, I could create a function that would satisfy the the qsimp requirement for a single argument function, but prior to run-time I can't.
>
>
>
> Anyway, I know this is a wacky and maybe even a silly question, but can't hurt to ask.
>
>
>
> Regards,
>
> Paul

Subject: Re: run-time function creation
Posted by David Fanning on Wed, 10 Apr 2013 14:56:51 GMT
View Forum Message <> Reply to Message

Paul Mallas writes:

> Actually, I think I just answered my own question. I could use the "common" block to pass the other parameters, huh? I have thought about this problem for a day or two, and as soon as I post the question, I come up with a solution myself. I guess I can't think straight until I clutter up the mail boxes of others :)

This is the Standard Operating Procedure in newsgroups, and one of their biggest benefits. In my case, I normally recognize what a fool I am a half second after I hit the Send button. :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
Sepore ma de ni thue. ("Perhaps thou speakest truth.")

Subject: Re: run-time function creation
Posted by Craig Markwardt on Wed, 10 Apr 2013 17:04:49 GMT
View Forum Message <> Reply to Message

On Wednesday, April 10, 2013 10:33:31 AM UTC-4, Paul Mallas wrote:
> I know this is a bit off the wall, but I was wondering if there is a technique or method for doing run-time function creation.
>
>
>
> The reason I ask is I was looking at the IDL integration routines (e.g., qsimp) and this requires a singular argument function name as input. My problem is I have a function with several parameters I need to integrate, but don't have these parameters until I calculate them at run time. If I could somehow create the function dynamically, I could create a function that would satisfy the the qsimp requirement for a single argument function, but prior to run-time I can't.

Well, I have two answers for you.

The first answer is that you can use my QPINT1D which is a better integrator than the IDL-standard quadrature functions. Plus, it allows you to enter in a private variable (a structure) for other parameters.

If you really want run-time functions, then FILE_COMPILE will do that for you. You need to write out a scratch file with the function you want, and then FILE_COMPILE will do the tricky part of compiling it. (not so tricky, but getting the paths right is more work than you might think)

Craig

QPINT1D
http://cow.physics.wisc.edu/~craigm/idl/math.html#QPINT1D

FILE_COMPILE
http://cow.physics.wisc.edu/~craigm/idl/introspect.html#FILE _COMPILE

Subject: Re: run-time function creation
Posted by PMan on Wed, 10 Apr 2013 18:07:06 GMT
View Forum Message <> Reply to Message

On Wednesday, April 10, 2013 10:33:31 AM UTC-4, Paul Mallas wrote:
> I know this is a bit off the wall, but I was wondering if there is a technique or method for doing run-time function creation.
>
>
>
> The reason I ask is I was looking at the IDL integration routines (e.g., qsimp) and this requires a singular argument function name as input. My problem is I have a function with several parameters I need to integrate, but don't have these parameters until I calculate them at run time.

If I could somehow create the function dynamically, I could create a function that would satisfy the the qsimp requirement for a single argument function, but prior to run-time I can't.
>
>
>
> Anyway, I know this is a wacky and maybe even a silly question, but can't hurt to ask.
>
>
>
> Regards,
>
> Paul

Thanks for the tips

Subject: Re: run-time function creation
Posted by PMan on Wed, 10 Apr 2013 19:57:17 GMT
View Forum Message <> Reply to Message

On Wednesday, April 10, 2013 1:04:49 PM UTC-4, Craig Markwardt wrote:
> On Wednesday, April 10, 2013 10:33:31 AM UTC-4, Paul Mallas wrote:
>
>> I know this is a bit off the wall, but I was wondering if there is a technique or method for doing run-time function creation.
>
>>
>
>>
>
>>
>
>> The reason I ask is I was looking at the IDL integration routines (e.g., qsimp) and this requires a singular argument function name as input. My problem is I have a function with several parameters I need to integrate, but don't have these parameters until I calculate them at run time. If I could somehow create the function dynamically, I could create a function that would satisfy the the qsimp requirement for a single argument function, but prior to run-time I can't.
>
>
>
> Well, I have two answers for you.
>
>
>
> The first answer is that you can use my QPINT1D which is a better integrator than the IDL-standard quadrature functions. Plus, it allows you to enter in a private variable (a structure) for other parameters.
>

>
>
> If you really want run-time functions, then FILE_COMPILE will do that for you. You need to write out a scratch file with the function you want, and then FILE_COMPILE will do the tricky part of compiling it. (not so tricky, but getting the paths right is more work than you might think)
>
>
>
> Craig
>
>
>
> QPINT1D
>
> http://cow.physics.wisc.edu/~craigm/idl/math.html#QPINT1D
>
>
>
> FILE_COMPILE
>
> http://cow.physics.wisc.edu/~craigm/idl/introspect.html#FILE _COMPILE

Craig - I used your qpint1. Excellent! It was just what I needed and it worked great. I even have the imsl lib and qpint1 worked much better than the imsl_intfcn. imsl_intfcn is fast and gives the right answer, but gives tons of floating point warnings and - for some reason - it kills the workbench when I run it. qpint1 appears slightly faster and does not deleterious side effects. Thank you!

Subject: Re: run-time function creation
Posted by rtk on Thu, 11 Apr 2013 15:35:04 GMT
View Forum Message <> Reply to Message

With IDL 8.x's new list and hash routines the idea of dynamically defined functions makes sense (consider implementing higher-order functions, like Python's map and reduce).

I have a lambda.pro routine, and a make_function routine, that pull off the trick Craig mentioned. What would be best is to define a function from a string and to be able to remove that function from memory. Still, dumping to a temporary file and compiling it on the fly is really quite fast with modern OSes. A couple of milliseconds typically.

Send me an email (rkneusel at exelisvis com) and I'll send them out to whoever is interested along with some higher-order function routines.

Subject: Re: run-time function creation
Posted by Craig Markwardt on Thu, 11 Apr 2013 16:28:24 GMT

On Wednesday, April 10, 2013 3:57:17 PM UTC-4, Paul Mallas wrote:
> On Wednesday, April 10, 2013 1:04:49 PM UTC-4, Craig Markwardt wrote:
>
>> On Wednesday, April 10, 2013 10:33:31 AM UTC-4, Paul Mallas wrote:
>
>>
>
>>> I know this is a bit off the wall, but I was wondering if there is a technique or method for doing run-time function creation.
>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>>
>
>>
>
>>> The reason I ask is I was looking at the IDL integration routines (e.g., qsimp) and this requires a singular argument function name as input. My problem is I have a function with several parameters I need to integrate, but don't have these parameters until I calculate them at run time. If I could somehow create the function dynamically, I could create a function that would satisfy the the qsimp requirement for a single argument function, but prior to run-time I can't.
>
>>
>
>>
>
>>
>
>>
>
>> Well, I have two answers for you.
>
>>
>
>>
>
>>
>
>>
>
>>
>
>> The first answer is that you can use my QPINT1D which is a better integrator than the IDL-standard quadrature functions. Plus, it allows you to enter in a private variable (a structure) for other parameters.

>
>>
>
>>
>
>>
>
>> If you really want run-time functions, then FILE_COMPILE will do that for you. You need to write out a scratch file with the function you want, and then FILE_COMPILE will do the tricky part of compiling it. (not so tricky, but getting the paths right is more work than you might think)
>
>>
>
>>
>
>>
>
>> Craig
>
>>
>
>>
>
>>
>
>> QPINT1D
>
>>
>
>> http://cow.physics.wisc.edu/~craigm/idl/math.html#QPINT1D
>
>>
>
>>
>
>>
>
>> FILE_COMPILE
>
>>
>
>> http://cow.physics.wisc.edu/~craigm/idl/introspect.html#FILE _COMPILE
>
>
>
> Craig - I used your qpint1. Excellent! It was just what I needed and it worked great. I even have the imsl lib and qpint1 worked much better than the imsl_intfcn. imsl_intfcn is fast and gives the right answer, but gives tons of floating point warnings and - for some reason - it kills the

workbench when I run it. qpint1 appears slightly faster and does not deleterious side effects.
Thank you!

You're welcome. It's kind of funny because both QPINT1D and IMSL_INTFCN have the same heritage (the QUADPACK package).

Craig
