Subject: How to write a dll using Ultimate++?
Posted by Oblivion on Thu, 23 Aug 2007 11:56:17 GMT
View Forum Message <> Reply to Message

Hi,

I couldn't find any examples or clues in the documetaion as to writing a dll using upp classes.

My question is that are there any special steps to be taken at writing a dll which uses ultimate++?

thank you.

Subject: Re: How to write a dll using Ultimate++?
Posted by mirek on Thu, 23 Aug 2007 15:37:44 GMT
View Forum Message <> Reply to Message

Yes and no. Interestingly, it is relatively easy to create U++ .dll to be used from non-U++ application, to use non-U++ .dll in U++ application, but what is complicated is U++ app with U++ .dll

What is your focus?

Mirek

Subject: Re: How to write a dll using Ultimate++?
Posted by fudadmin on Thu, 23 Aug 2007 15:46:06 GMT
View Forum Message <> Reply to Message

Have you got DLL template dll.upt or similar? As I remember in the old 605 version there was one but not in 2007...
Maybe someone could upload one here?

Subject: Re: How to write a dll using Ultimate++?
Posted by Oblivion on Thu, 23 Aug 2007 19:51:52 GMT
View Forum Message <> Reply to Message

Well, currently I am converting my multi-protocol IM application (something like miranda, etc.) from MFC to UPP. I've already recoded the main IM framework into UPP. But my plugin system depends (as usual) on dll's and I want to use UPP instead of MFC in the plugin dlls. So that I want't to use UPP-dll in a UPP based application. Is there any way to use upp "in" the dll's? If yes, is there any barebone upp-dll example available?

(For example, in MFC you have to take some special steps (like using GETMANAGEDSTATE() macro, or in smartwin++ "cannibalizing" the main enty code with some smartwin++ routines) to get

it work.)

And, no I don't have any dll.upt template or similar. I'm using 2007.1

Thank you.

Subject: Re: How to write a dll using Ultimate++?
Posted by fudadmin on Thu, 23 Aug 2007 20:36:48 GMT
View Forum Message <> Reply to Message

Ok, I've found the template (but it might be not the original)

File Attachments
1) Win32dll.upt, downloaded 779 times

Subject: Re: How to write a dll using Ultimate++?
Posted by Zardos on Thu, 23 Aug 2007 20:49:09 GMT
View Forum Message <> Reply to Message

You can also check this thread:

http://www.ultimatepp.org/forum/index.php?t=msg&goto=103 67&&srch=dll#msg_10367

Subject: Re: How to write a dll using Ultimate++?
Posted by Oblivion on Thu, 23 Aug 2007 21:00:37 GMT
View Forum Message <> Reply to Message

Unfortunately, the template didn't help too much.
(thank you very much for your help, though)
But you are totally right fudadmin. I have to re-formulate my problem:

luzr wrote on Thu, 23 August 2007 17:37Yes and no. Interestingly, it is relatively easy to create U++ .dll to be used from non-U++ application, to use non-U++ .dll in U++ application, but what is complicated is U++ app with U++ .dll

What is your focus?

Mirek

I mean, how to write a U++ app with U++ dll?

regards.

Subject: Re: How to write a dll using Ultimate++?
Posted by fudadmin on Thu, 23 Aug 2007 22:45:23 GMT
View Forum Message <> Reply to Message

Quote:I mean, how to write a U++ app with U++ dll?
Is your life not enough hell? Would you like another one? I mean dll...
Ok, less serious questions:
1. Why do you want U++ packed into dll?
2. Have you got any experience writing and loading C++ dll?
3. Have you successfully compiled test dll project with the template I posted?

P.S. I had some very limited successful experiments in this area some time ago but I can't promise that I can find the correct files or the current version of U++ would work with them.

Subject: Re: How to write a dll using Ultimate++?
Posted by Oblivion on Fri, 24 Aug 2007 11:25:45 GMT
View Forum Message <> Reply to Message

fudadmin wrote on Fri, 24 August 2007 00:45
Ok, less serious questions:
1. Why do you want U++ packed into dll?
2. Have you got any experience writing and loading C++ dll?
3. Have you successfully compiled test dll project with the template I posted?

1. Actually, I have a bunch of dlls which contain GUI code in my project. If I link these dlls statically with U++, I get a "huge" dll file (1 to 3 megs depending on it's profile) Assume that I have 3 different dll's which contain U++ code, then the total file size of the dll's would get to almost 6-10 megs . Now, I call this hell! That's why I need U++ in a dll.

2. Yes, I am happy with C/C++ dll's

3. Yes, I have successfully compiled the template. But it is an ordinary win32 dll template and the problem is that whenever I load any U++ GUI class through the dll (eg. opening a TopWindow), The main or child windows layouts (eg, borders, colors) get corrupted or not drawn. In MFC, for example, AFX_MANAGE_STATE() macro is used to prevent such misfunctioning. Is there any macro, or method needed to be called before initalization of the dlls in u++?

4. Bonus: thank you for your patience

Subject: Re: How to write a dll using Ultimate++?
Posted by mr_ped on Fri, 24 Aug 2007 17:15:00 GMT
View Forum Message <> Reply to Message

AFAIK (I'm not really an expert for UPP, Mirek (luzr) will give you much more accurate information as he is creator of UPP):

The upp is not prepared to be included into DLLs and mixed up with main UPP app.
The problem is in clashing between different threads of the UPP core.

I'm not sure about creating UPP dll with core, and than using this DLL in plugins DLLs.
But as far as I can remember you are exploring new grounds and you will very likely need some tweaks inside UPP to make this possible.

I think if your usage of UPP from plugins is limited enough, maybe it would be easier to create a wrapper interface class with those few functions used, than instantiate this wrapper class in UPP dll, which will run the only upp core in whole application, and make the wrapper instance public to plugins DLLs and call anything from UPP trough that instance.

I'm not comfortable enough with DLLs to understand what will happen with heap, memory allocations, etc... in such case. I have some vague idea that every DLL has it's own memory allocator and heap? This may get you on really thin ice, if you are not sure what you are doing.

Subject: Re: How to write a dll using Ultimate++?
Posted by Novo on Fri, 24 Aug 2007 21:12:23 GMT
View Forum Message <> Reply to Message

mr_ped wrote on Fri, 24 August 2007 13:15I'm not comfortable enough with DLLs to understand what will happen with heap, memory allocations, etc... in such case. I have some vague idea that every DLL has it's own memory allocator and heap? This may get you on really thin ice, if you are not sure what you are doing.

As long as you are linking against CRT *dynamically*, and using only one version of CRT (let say MT DLL) you will be fine.

Subject: Re: How to write a dll using Ultimate++?
Posted by Novo on Fri, 24 Aug 2007 21:23:21 GMT
View Forum Message <> Reply to Message

fudadmin wrote on Thu, 23 August 2007 18:45Quote:I mean, how to write a U++ app with U++ dll?
Is your life not enough hell? Would you like another one? I mean dll...

I've been using DLLs on Windows and Unix for long time, and I can help with making U++ more DLL-frendly ...

DLL is not hell, especially in case of Unix.

Subject: Re: How to write a dll using Ultimate++?
Posted by Novo on Fri, 24 Aug 2007 21:35:26 GMT
View Forum Message <> Reply to Message

Oblivion wrote on Thu, 23 August 2007 17:00
I mean, how to write a U++ app with U++ dll?

You won't find dllexport/dllimport in the source code. So, the answer is very simple: there is no way (on Windows).

Subject: Re: How to write a dll using Ultimate++?
Posted by mirek on Fri, 24 Aug 2007 22:09:07 GMT
View Forum Message <> Reply to Message

Actually, no dllimport/dllexport is not a problem: TheIDE builders take care about this stupid detail, making U++ dynamic linking equalt to the one in Linux in this regard.

It was even possible to build all packages as .dlls - that is what "SO" option in the Output mode... are for.

Some time ago this has worked in Windows too, but it is too annoying to maintain U++ sources in "dll" friendly state - there is too much you are not allowed to do in .dll in windows... (global variables to start the list...).

Mirek

Subject: Re: How to write a dll using Ultimate++?
Posted by fudadmin on Sat, 25 Aug 2007 23:49:36 GMT
View Forum Message <> Reply to Message

Something wrong here with question (project structure?) formulation...
Oblivion, you said you can't pack U++ code into dll and then you say:
Quote: Actually, I have a bunch of dlls which contain GUI code in my project.
That means, you have got some extra-terrestrial (not U++) GUI code like wxWidgets, Fox etc., does it?
(additional question: Why not to convert all the GUI into U++, then)?

Quote:If I link these dlls statically with U++, I get a "huge" dll file (1 to 3 megs depending on it's profile Shocked)
I wouldn't say 1-3 Mb is huge these days...

Quote:Assume that I have 3 different dll's which contain U++ code, then the total file size of the dll's would get to almost 6-10 megs Shocked . Now, I call this hell! Razz That's why I need U++ in a dll. (I asssume you've checked it in release mode) But... is 6-10mb a really such a big problem to load for nowadays computers?

OTOH, check and compare theide and let's say firefox memory usage... One year ago and a bit, when I was using widows, theide memory usage was comparable to that of notepad.exe...

Also, the assumption logic that, if you merge 3 U++ programs of 2 Mb into one, you'l get 6 Mb is very suspicious (mildly speaking, imo). Could someone (Mirek?) confirm/deny that?

So, the size of app is your main worry?

If that is the case maybe is worth to think about e.g compression of the executable, using better compiler on windows, playings with optim flags.

And lastly, why not to make U++ your main part and to load other parts as dlls?

Subject: Re: How to write a dll using Ultimate++?
Posted by Oblivion on Sun, 26 Aug 2007 01:15:09 GMT
View Forum Message <> Reply to Message

Quote:That means, you have got some extra-terrestrial (not U++) GUI code like wxWidgets, Fox etc., does it?
(additional question: Why not to convert all the GUI into U++, then)?

I was using MFC (as a dll) in my project(s). Actually, I was nearly a "diehard" MFC user. But after considering the disadvantages of the age-old MFC and the advantages or the possible benefits of the ultimate++, I decided to convert my MFC based multi-protocol IM project into U++. I've already finished converting the main framework into U++ without any problems. The main executable (which consists of a service handler and a GUI) is now working fine (And I have to admit that, with the u++ it looks and acts much more better).

Quote:And lastly, why not to make U++ your main part and to load other parts as dlls?

Firstly; U++ is the main part of my project and I use some other parts of GUI code (gui-extension plugins) as dlls. But as I stated before,

Quote:3. Yes, I have successfully compiled the template. But it is an ordinary win32 dll template and the problem is that whenever I load any U++ GUI class through the dll (eg. opening a TopWindow), The main or child windows layouts (eg, borders, colors) get corrupted or not drawn. In MFC, for example, AFX_MANAGE_STATE() macro is used to prevent such misfunctioning. Is

there any macro, or method needed to be called before initalization of the dlls in u++?

Is there any solution to this odd behaviour? Upp based GUI-extension dlls compile without any errors or warnings. But the gui components looks corrupted every time I load them through the dlls. I didn't come by any special dll entry point (dllmain()) function or macro in the documentation or in the upp source code, which I should invoke. Am I missing something?

It's no problem for me to code these dll's in any other framework. But U++ has many benefits as far as I can see. So that's why I'm insisting on the dll issue.

Secondly; as to a U++.dll, ok I give up this idea for some time

Regards.

Subject: Re: How to write a dll using Ultimate++?
Posted by mirek on Sun, 26 Aug 2007 07:29:48 GMT
View Forum Message <> Reply to Message

Well, if you are going to produce production quality code, make sure that you

- use MSC compiler (exe size drops 50%, is 2x faster) - the best results are with MSC7.1 if you can get it, IME MSC8.0 is worse compiler. Obviously, you need commandline tools only...

- compile in release mode

And yes, the minimal application size is 1.2MB (e.g. UWord).

Mirek

Subject: Re: How to write a dll using Ultimate++?
Posted by Novo on Mon, 27 Aug 2007 18:19:08 GMT
View Forum Message <> Reply to Message

luzr wrote on Fri, 24 August 2007 18:09Actually, no dllimport/dllexport is not a problem: TheIDE builders take care about this stupid detail, making U++ dynamic linking equalt to the one in Linux in this regard.

That is very interesting. I know only two ways to export functions from DLL:
1) dllexport;
2) def-file;

There is probably a possibility to export everything (like -E with gcc).

How did you manage to do that? In order to build a def-file you have to know mangled names of functions/methods.

Quote:
It was even possible to build all packages as .dlls - that is what "SO" option in the Output mode... are for.

I couldn't find the "SO" option, I only see "Shared libs" and "All shared", but they don't seem to have any effect.

Quote:
Some time ago this has worked in Windows too, but it is too annoying to maintain U++ sources in "dll" friendly state - there is too much you are not allowed to do in .dll in windows... (global variables to start the list...).

Mirek

IMHO, global variables are allowed in DLLs on Windows. Could you please describe that problem in more details or point me out to an appropriate discussion?

P.S. IMHO, this discussion should be continued in "technology" group.

Subject: Re: How to write a dll using Ultimate++?
Posted by mirek on Mon, 27 Aug 2007 18:56:31 GMT
View Forum Message <> Reply to Message

Quote:
That is very interesting. I know only two ways to export functions from DLL:
1) dllexport;
2) def-file;

There is probably a possibility to export everything (like -E with gcc).

How did you manage to do that? In order to build a def-file you have to know mangled names of functions/methods.

Look into .obj file and collect the mangled names, then autogenerate the appropriate .def file

Quote:
I couldn't find the "SO" option, I only see "Shared libs" and "All shared", but they don't seem to have any effect.

"All shared". Reflected as "SO" compile flag.

Quote:
IMHO, global variables are allowed in DLLs on Windows. Could you please describe that problem in more details or point me out to an appropriate discussion?

Unfortunately they are not. GCC has some crazy hack that makes it possible (something sort of calling a function and returning the pointer to variable), but in raw .dll, technically it is impossible, there are entries for functions only.

Mirek

Subject: Re: How to write a dll using Ultimate++?
Posted by Novo on Mon, 27 Aug 2007 21:27:58 GMT
View Forum Message <> Reply to Message

luzr wrote on Mon, 27 August 2007 14:56Quote:
That is very interesting. I know only two ways to export functions from DLL:
1) dllexport;
2) def-file;

There is probably a possibility to export everything (like -E with gcc).

How did you manage to do that? In order to build a def-file you have to know mangled names of functions/methods.

Look into .obj file and collect the mangled names, then autogenerate the appropriate .def file

It's a cool hacker's trick. And TheIDE is a a good C++ hacking tool. But that doesn't make UPP portable, if it is a plan.

Quote:
Quote:
IMHO, global variables are allowed in DLLs on Windows. Could you please describe that problem in more details or point me out to an appropriate discussion?

Unfortunately they are not. GCC has some crazy hack that makes it possible (something sort of calling a function and returning the pointer to variable), but in raw .dll, technically it is impossible, there are entries for functions only.

Mirek

I still do not understand you. A few years back I wrote a simple dll-testing application (mostly to test static/global object destruction order and DLL unloading order). I couldn't find any problems

with DLL + MinGW (I was able to crash gcc from Cygwin). Could you please post a code snippet that demonstrates the problem? I can upload my dll-testing application.

Thanks.

Subject: Re: How to write a dll using Ultimate++?
Posted by mirek on Sat, 08 Sep 2007 14:24:19 GMT
View Forum Message <> Reply to Message

Novo wrote on Mon, 27 August 2007 17:27luzr wrote on Mon, 27 August 2007 14:56
Look into .obj file and collect the mangled names, then autogenerate the appropriate .def file

It's a cool hacker's trick. And TheIDE is a a good C++ hacking tool. But that doesn't make UPP portable, if it is a plan.

Well, that is the question. It certainly makes it more portable w.r.t. to moving applications to Linux - this is how Linux works.

Quote:
I still do not understand you. A few years back I wrote a simple dll-testing application (mostly to test static/global object destruction order and DLL unloading order). I couldn't find any problems with DLL + MinGW (I was able to crash gcc from Cygwin). Could you please post a code snippet that demonstrates the problem? I can upload my dll-testing application.

Have you tested with MSC?

Mirek

Subject: Re: How to write a dll using Ultimate++?
Posted by Novo on Mon, 10 Sep 2007 17:08:21 GMT
View Forum Message <> Reply to Message

luzr wrote on Sat, 08 September 2007 10:24
Have you tested with MSC?

Mirek

Checked at home (I ran these tests two or three years back) ...

Windows MSVC v7.1
Windows cygwin gcc v3.4.4
Windows MinGW gcc v3.4.5
Linux GCC 3.4.0 (Slackware)
Solaris GCC v4.0.1

I'll also try to find testing results at work. I had a lot of them.

I'm really interested in discussion about troubles with DLLs on COFF and ELF platforms because this is a part of my job.

Subject: Re: How to write a dll using Ultimate++?
Posted by Novo on Mon, 10 Sep 2007 17:34:56 GMT
View Forum Message <> Reply to Message

An example of test output.

Windows cygwin gcc v3.4.4 ...

```
Run test_dll_app_01
+ 0 Created  : test_dll_01 global
+ 1 Registered atexit_funct: test_dll_01 global
+ 2 Created  : test_dll_app_01 global
+ 3 Registered atexit_funct: test_dll_app_01 global
+ 4 Created  : test_dll_app_01 local static
+ 5 Registered atexit_funct: test_dll_app_01 local
+ 6 Created  : test_dll_01 local static do_it
+ 7 Registered atexit_funct: test_dll_01 local do_it
- 0 Destroyed: test_dll_01 global
- 7 Called atexit_funct: test_dll_01 local do_it
- 6 Destroyed: test_dll_01 local static do_it
- 5 Called atexit_funct: test_dll_app_01 local
- 4 Destroyed: test_dll_app_01 local static
- 2 Destroyed: test_dll_app_01 global
- 3 Called atexit_funct: test_dll_app_01 global
- 1 Called atexit_funct: test_dll_01 global


Run test_dll_app_02
+ 0 Created  : test_dll_app_02 global
+ 1 Registered atexit_funct: test_dll_app_02 global
+ 2 Created  : test_dll_app_02 local static
+ 3 Registered atexit_funct: test_dll_app_02 local
Load test_dll_01 manually ...
+ 4 Created  : test_dll_01 global
```

```
+ 5 Registered atexit_funct: test_dll_01 global
+ 6 Created  : test_dll_01 local static do_it
+ 7 Registered atexit_funct: test_dll_01 local do_it
Unoad test_dll_01 manually ...
- 4 Destroyed: test_dll_01 global
- 7 Called atexit_funct: test_dll_01 local do_it
- 6 Destroyed: test_dll_01 local static do_it
- 5 Called atexit_funct: test_dll_01 global
- 3 Called atexit_funct: test_dll_app_02 local
- 2 Destroyed: test_dll_app_02 local static
- 0 Destroyed: test_dll_app_02 global
- 1 Called atexit_funct: test_dll_app_02 global


Run test_dll_app_02 -n (Do not unload DLLs)
+ 0 Created  : test_dll_app_02 global
+ 1 Registered atexit_funct: test_dll_app_02 global
+ 2 Created  : test_dll_app_02 local static
+ 3 Registered atexit_funct: test_dll_app_02 local
Load test_dll_01 manually ...
+ 4 Created  : test_dll_01 global
+ 5 Registered atexit_funct: test_dll_01 global
+ 6 Created  : test_dll_01 local static do_it
+ 7 Registered atexit_funct: test_dll_01 local do_it
- 4 Destroyed: test_dll_01 global
- 7 Called atexit_funct: test_dll_01 local do_it
- 6 Destroyed: test_dll_01 local static do_it
- 5 Called atexit_funct: test_dll_01 global
- 3 Called atexit_funct: test_dll_app_02 local
- 2 Destroyed: test_dll_app_02 local static
- 0 Destroyed: test_dll_app_02 global
- 1 Called atexit_funct: test_dll_app_02 global


Run test_dll_app_03
+ 0 Created  : test_dll_01 global
+ 1 Registered atexit_funct: test_dll_01 global
+ 2 Created  : test_dll_02 global
+ 3 Registered atexit_funct: test_dll_02 global
+ 4 Created  : test_dll_app_03 global
+ 5 Registered atexit_funct: test_dll_app_03 global
+ 6 Created  : test_dll_app_03 local static
+ 7 Registered atexit_funct: test_dll_app_03 local
+ 8 Created  : test_dll_02 local static do_it
+ 9 Registered atexit_funct: test_dll_02 local do_it
+ 10 Created  : test_dll_01 local static do_it
+ 11 Registered atexit_funct: test_dll_01 local do_it
- 0 Destroyed: test_dll_01 global
```

- 2 Destroyed: test_dll_02 global
- 11 Called atexit_funct: test_dll_01 local do_it
- 10 Destroyed: test_dll_01 local static do_it
- 9 Called atexit_funct: test_dll_02 local do_it
- 8 Destroyed: test_dll_02 local static do_it
- 7 Called atexit_funct: test_dll_app_03 local
- 6 Destroyed: test_dll_app_03 local static
- 4 Destroyed: test_dll_app_03 global
- 5 Called atexit_funct: test_dll_app_03 global
- 3 Called atexit_funct: test_dll_02 global
- 1 Called atexit_funct: test_dll_01 global

Run test_dll_app_04
+ 0 Created : test_dll_app_04 global
+ 1 Registered atexit_funct: test_dll_app_04 global
+ 2 Created : test_dll_app_04 local static
+ 3 Registered atexit_funct: test_dll_app_04 local
Load test_dll_02 manually ...
+ 4 Created : test_dll_01 global
+ 5 Registered atexit_funct: test_dll_01 global
+ 6 Created : test_dll_02 global
+ 7 Registered atexit_funct: test_dll_02 global
+ 8 Created : test_dll_02 local static do_it
+ 9 Registered atexit_funct: test_dll_02 local do_it
+ 10 Created : test_dll_01 local static do_it
+ 11 Registered atexit_funct: test_dll_01 local do_it
Unoad test_dll_02 manually ...
- 6 Destroyed: test_dll_02 global
- 4 Destroyed: test_dll_01 global
     8 [main] test_dll_app_04 2504 _cygtls::handle_exceptions: Exception:
STATUS_ACCESS_VIOLATION
  1658 [main] test_dll_app_04 2504 open_stackdumpfile: Dumping stack trace to
test_dll_app_04.exe.stackdump

Run test_dll_app_04 -n (Do not unload DLLs)
+ 0 Created : test_dll_app_04 global
+ 1 Registered atexit_funct: test_dll_app_04 global
+ 2 Created : test_dll_app_04 local static
+ 3 Registered atexit_funct: test_dll_app_04 local
Load test_dll_02 manually ...
+ 4 Created : test_dll_01 global
+ 5 Registered atexit_funct: test_dll_01 global
+ 6 Created : test_dll_02 global
+ 7 Registered atexit_funct: test_dll_02 global
+ 8 Created : test_dll_02 local static do_it
+ 9 Registered atexit_funct: test_dll_02 local do_it

```
+ 10 Created  : test_dll_01 local static do_it
+ 11 Registered atexit_funct: test_dll_01 local do_it
- 6 Destroyed: test_dll_02 global
- 4 Destroyed: test_dll_01 global
      4 [main] test_dll_app_04 2748 _cygtls::handle_exceptions: Exception:
STATUS_ACCESS_VIOLATION
   1414 [main] test_dll_app_04 2748 open_stackdumpfile: Dumping stack trace to
test_dll_app_04.exe.stackdump


Run test_dll_app_05
+ 0 Created  : test_dll_03 global
+ 1 Registered atexit_funct: test_dll_03 global
+ 2 Created  : test_dll_app_05 global
+ 3 Registered atexit_funct: test_dll_app_05 global
+ 4 Created  : test_dll_app_05 local static
+ 5 Registered atexit_funct: test_dll_app_05 local
+ 6 Created  : test_dll_03 local static do_it
+ 7 Registered atexit_funct: test_dll_03 local do_it
Load test_dll_01 manually from test_dll_03 ...
+ 8 Created  : test_dll_01 global
+ 9 Registered atexit_funct: test_dll_01 global
+ 10 Created  : test_dll_01 local static do_it
+ 11 Registered atexit_funct: test_dll_01 local do_it
Unload test_dll_01 manually from test_dll_03 ...
- 8 Destroyed: test_dll_01 global
- 0 Destroyed: test_dll_03 global
      8 [main] test_dll_app_05 2428 _cygtls::handle_exceptions: Exception:
STATUS_ACCESS_VIOLATION
   1475 [main] test_dll_app_05 2428 open_stackdumpfile: Dumping stack trace to
test_dll_app_05.exe.stackdump


Run test_dll_app_06
+ 0 Created  : test_dll_app_06 global
+ 1 Registered atexit_funct: test_dll_app_06 global
+ 2 Created  : test_dll_app_06 local static
+ 3 Registered atexit_funct: test_dll_app_06 local
Load test_dll_03 manually ...
+ 4 Created  : test_dll_03 global
+ 5 Registered atexit_funct: test_dll_03 global
+ 6 Created  : test_dll_03 local static do_it
+ 7 Registered atexit_funct: test_dll_03 local do_it
Load test_dll_01 manually from test_dll_03 ...
+ 8 Created  : test_dll_01 global
+ 9 Registered atexit_funct: test_dll_01 global
+ 10 Created  : test_dll_01 local static do_it
+ 11 Registered atexit_funct: test_dll_01 local do_it
```

```
Unload test_dll_01 manually from test_dll_03 ...
- 8 Destroyed: test_dll_01 global
UnLoad test_dll_03 manually ...
- 4 Destroyed: test_dll_03 global
      8 [main] test_dll_app_06 1900 _cygtls::handle_exceptions: Exception:
STATUS_ACCESS_VIOLATION
   1792 [main] test_dll_app_06 1900 open_stackdumpfile: Dumping stack trace to
test_dll_app_06.exe.stackdump


Run test_dll_app_06 -n (Do not unload DLLs)
+ 0 Created  : test_dll_app_06 global
+ 1 Registered atexit_funct: test_dll_app_06 global
+ 2 Created  : test_dll_app_06 local static
+ 3 Registered atexit_funct: test_dll_app_06 local
Load test_dll_03 manually ...
+ 4 Created  : test_dll_03 global
+ 5 Registered atexit_funct: test_dll_03 global
+ 6 Created  : test_dll_03 local static do_it
+ 7 Registered atexit_funct: test_dll_03 local do_it
Load test_dll_01 manually from test_dll_03 ...
+ 8 Created  : test_dll_01 global
+ 9 Registered atexit_funct: test_dll_01 global
+ 10 Created  : test_dll_01 local static do_it
+ 11 Registered atexit_funct: test_dll_01 local do_it
Unload test_dll_01 manually from test_dll_03 ...
- 8 Destroyed: test_dll_01 global
- 4 Destroyed: test_dll_03 global
      8 [main] test_dll_app_06 2024 _cygtls::handle_exceptions: Exception:
STATUS_ACCESS_VIOLATION
   1434 [main] test_dll_app_06 2024 open_stackdumpfile: Dumping stack trace to
test_dll_app_06.exe.stackdump


Run test_dll_app_07
+ 0 Created  : test_dll_04 global
+ 1 Registered atexit_funct: test_dll_04 global
+ 2 Created  : test_dll_app_07 global
+ 3 Registered atexit_funct: test_dll_app_07 global
+ 4 Created  : test_dll_app_07 local static
+ 5 Registered atexit_funct: test_dll_app_07 local
+ 6 Created  : test_dll_04 local static do_it
+ 7 Registered atexit_funct: test_dll_04 local do_it
Load test_dll_01 manually from test_dll_04 ...
+ 8 Created  : test_dll_01 global
+ 9 Registered atexit_funct: test_dll_01 global
+ 10 Created  : test_dll_01 local static do_it
+ 11 Registered atexit_funct: test_dll_01 local do_it
```

- 0 Destroyed: test_dll_04 global
- 8 Destroyed: test_dll_01 global
- 11 Called atexit_funct: test_dll_01 local do_it
- 10 Destroyed: test_dll_01 local static do_it
- 9 Called atexit_funct: test_dll_01 global
- 7 Called atexit_funct: test_dll_04 local do_it
- 6 Destroyed: test_dll_04 local static do_it
- 5 Called atexit_funct: test_dll_app_07 local
- 4 Destroyed: test_dll_app_07 local static
- 2 Destroyed: test_dll_app_07 global
- 3 Called atexit_funct: test_dll_app_07 global
- 1 Called atexit_funct: test_dll_04 global

Run test_dll_app_08
+ 0 Created : test_dll_app_08 global
+ 1 Registered atexit_funct: test_dll_app_08 global
+ 2 Created : test_dll_app_08 local static
+ 3 Registered atexit_funct: test_dll_app_08 local
Load test_dll_04 manually ...
+ 4 Created : test_dll_04 global
+ 5 Registered atexit_funct: test_dll_04 global
+ 6 Created : test_dll_04 local static do_it
+ 7 Registered atexit_funct: test_dll_04 local do_it
Load test_dll_01 manually from test_dll_04 ...
+ 8 Created : test_dll_01 global
+ 9 Registered atexit_funct: test_dll_01 global
+ 10 Created : test_dll_01 local static do_it
+ 11 Registered atexit_funct: test_dll_01 local do_it
UnLoad test_dll_04 manually ...
- 4 Destroyed: test_dll_04 global
- 8 Destroyed: test_dll_01 global
- 11 Called atexit_funct: test_dll_01 local do_it
- 10 Destroyed: test_dll_01 local static do_it
- 9 Called atexit_funct: test_dll_01 global
      4 [main] test_dll_app_08 2024 _cygtls::handle_exceptions: Exception:
STATUS_ACCESS_VIOLATION
   1477 [main] test_dll_app_08 2024 open_stackdumpfile: Dumping stack trace to
test_dll_app_08.exe.stackdump

Run test_dll_app_08 -n (Do not unload DLLs)
+ 0 Created : test_dll_app_08 global
+ 1 Registered atexit_funct: test_dll_app_08 global
+ 2 Created : test_dll_app_08 local static
+ 3 Registered atexit_funct: test_dll_app_08 local
Load test_dll_04 manually ...
+ 4 Created : test_dll_04 global

+ 5 Registered atexit_funct: test_dll_04 global
+ 6 Created : test_dll_04 local static do_it
+ 7 Registered atexit_funct: test_dll_04 local do_it
Load test_dll_01 manually from test_dll_04 ...
+ 8 Created : test_dll_01 global
+ 9 Registered atexit_funct: test_dll_01 global
+ 10 Created : test_dll_01 local static do_it
+ 11 Registered atexit_funct: test_dll_01 local do_it
- 4 Destroyed: test_dll_04 global
- 8 Destroyed: test_dll_01 global
- 11 Called atexit_funct: test_dll_01 local do_it
- 10 Destroyed: test_dll_01 local static do_it
- 9 Called atexit_funct: test_dll_01 global
      7 [main] test_dll_app_08 1900 _cygtls::handle_exceptions: Exception:
STATUS_ACCESS_VIOLATION
   1720 [main] test_dll_app_08 1900 open_stackdumpfile: Dumping stack trace to
test_dll_app_08.exe.stackdump

Subject: Re: How to write a dll using Ultimate++?
Posted by Novo on Tue, 11 Sep 2007 15:54:23 GMT
View Forum Message <> Reply to Message

Novo wrote on Mon, 10 September 2007 13:08
Checked at home (I ran these tests two or three years back) ...

Windows MSVC v7.1
Windows cygwin gcc v3.4.4
Windows MinGW gcc v3.4.5
Linux GCC 3.4.0 (Slackware)
Solaris GCC v4.0.1

Checked at work (newer version of the test) ...

FreeBSD (4.x) GCC 3.4.2 ...
IRIX GCC 3.3.0 ...
IRIX MIPSpro73 ...
Linux (Slackware) GCC v3.4.0 ...
Linux (Slackware) ICC v9.0.0 ...
OSF1 GCC 3.3.2 ...
Solaris GCC v4.0.1 ...
Solaris 10 WorkShop v5.5 ...
Windows MSVC v7.1 ...
Windows MinGW gcc v3.4.5 ...
Windows cygwin gcc v3.4.4 ...

Subject: Re: How to write a dll using Ultimate++?
Posted by Novo on Tue, 11 Sep 2007 21:04:44 GMT
View Forum Message <> Reply to Message

luzr wrote on Sat, 08 September 2007 10:24Have you tested with MSC?

Mirek

BTW, I also have performance test with embedded databases:
Sqlite;
MSJET (ODBC and ADO);
Firebird;

This test is quite simple and designed after a similar test for SQLITE.

If somebody is interested in benchmarks and comparing different implementations of database access layer I can upload it. I currently have only version for Windows.

Subject: Re: How to write a dll using Ultimate++?
Posted by Novo on Wed, 12 Sep 2007 18:15:19 GMT
View Forum Message <> Reply to Message

Novo wrote on Tue, 11 September 2007 17:04

This test is quite simple and designed after similar test for SQLITE.

This test is also completely DLL-based
So, it is not off-topic.

Subject: Re: How to write a dll using Ultimate++?
Posted by mirek on Sat, 15 Sep 2007 22:53:47 GMT
View Forum Message <> Reply to Message

As for global vars and DLL, perhaps I was not specific.

The problem is not global var per see, but what you cannot do is to reference the global variable of other .dll.

In other words, "extern" does not work for data from another dll.

(It does with some trickery, witch involves converting the access to data to call to a function that

returns a pointer to data).

Mirek

Subject: Re: How to write a dll using Ultimate++?
Posted by mirek on Sat, 15 Sep 2007 22:54:18 GMT
View Forum Message <> Reply to Message

Novo wrote on Tue, 11 September 2007 17:04luzr wrote on Sat, 08 September 2007 10:24Have you tested with MSC?

Mirek

BTW, I also have performance test with embedded databases:
Sqlite;
MSJET (ODBC and ADO);
Firebird;

This test is quite simple and designed after a similar test for SQLITE.

If somebody is interested in benchmarks and comparing different implementations of database access layer I can upload it. I currently have only version for Windows.

I am always interested in benchmarks!

Mirek

Subject: Re: How to write a dll using Ultimate++?
Posted by Novo on Sun, 16 Sep 2007 01:41:36 GMT
View Forum Message <> Reply to Message

luzr wrote on Sat, 15 September 2007 18:54Novo wrote on Tue, 11 September 2007 17:04luzr wrote on Sat, 08 September 2007 10:24Have you tested with MSC?

Mirek

BTW, I also have performance test with embedded databases:
Sqlite;
MSJET (ODBC and ADO);
Firebird;

This test is quite simple and designed after a similar test for SQLITE.

If somebody is interested in benchmarks and comparing different implementations of database access layer I can upload it. I currently have only version for Windows.

I am always interested in benchmarks!

Mirek

Done.

http://www.ultimatepp.org/forum/index.php?t=msg&goto=115 05&#msg_11505

Subject: Re: How to write a dll using Ultimate++?
Posted by Novo on Sun, 16 Sep 2007 02:01:27 GMT
View Forum Message <> Reply to Message

luzr wrote on Sat, 15 September 2007 18:53As for global vars and DLL, perhaps I was not specific.

The problem is not global var per see, but what you cannot do is to reference the global variable of other .dll.

In other words, "extern" does not work for data from another dll.

(It does with some trickery, witch involves converting the access to data to call to a function that returns a pointer to data).

Mirek

That is not a problem at all comparing to real problems.

That is some sort of small inconvenience. I believe you can export data from DLL using __declspec(dllexport). Using of a function, which returns a pointer to data, is another solution.

Subject: Re: How to write a dll using Ultimate++?
Posted by mirek on Sun, 16 Sep 2007 06:25:39 GMT
View Forum Message <> Reply to Message

Novo wrote on Sat, 15 September 2007 22:01luzr wrote on Sat, 15 September 2007 18:53As for

global vars and DLL, perhaps I was not specific.

The problem is not global var per see, but what you cannot do is to reference the global variable of other .dll.

In other words, "extern" does not work for data from another dll.

(It does with some trickery, witch involves converting the access to data to call to a function that returns a pointer to data).

Mirek

That is not a problem at all comparing to real problems.

That is some sort of small inconvenience. I believe you can export data from DLL using __declspec(dllexport).

Well, AFAIK, you cannot with MSC.

Quote:
Using of a function, which returns a pointer to data, is another solution.

Yes. Anyway, in some cases, this leads to much slower code. E.g. look into Core/Charset.h, to the set of definitions starting with IsLetter (there are two paths; for SO version you cannot use the inlines).

Mirek

Subject: Re: How to write a dll using Ultimate++?
Posted by Novo on Thu, 20 Sep 2007 14:57:25 GMT
View Forum Message <> Reply to Message

luzr wrote on Sun, 16 September 2007 02:25
Well, AFAIK, you cannot with MSC.

Check that with *Dependency Walker* yourself. I've attached an example.

Quote:
Quote:
Using of a function, which returns a pointer to data, is another solution.

Yes. Anyway, in some cases, this leads to much slower code. E.g. look into Core/Charset.h, to the set of definitions starting with IsLetter (there are two paths; for SO version you cannot use the inlines).

Mirek

Call to a function in DLL will always be a little bit slower. Same as with virtual functions. As usual, I have a test for this ...

Pointer to a function performance tests ...

Raw function: 0 sec.
Raw pointer to a function: 0.421 sec.
Boost.Function: 0.938 sec.
FastDelegate: 0.797 sec.

Pointer to a function in DLL performance tests ...

Raw function: 0.578 sec.
Raw pointer to a function: 0.594 sec.
Boost.Function: 0.937 sec.
FastDelegate: 0.75 sec.

Method call performance tests ...

Method call: 0 sec.
Raw pointer to a method: 0.391 sec.
Boost.Function: 0.719 sec.
FastDelegate: 0.406 sec.

Method call in DLL performance tests ...

Method call: 0.578 sec.
Raw pointer to a method: 0.609 sec.
Boost.Function: 0.735 sec.
FastDelegate: 0.578 sec.

Virtual method call performance tests ...

Virtual method call: 0.594 sec.
Raw pointer to a virtual method: 0.562 sec.
Boost.Function: 0.953 sec.
FastDelegate: 0.594 sec.

Virtual method call in DLL performance tests ...

Virtual method call: 0.609 sec.

Raw pointer to a virtual method: 0.563 sec.
Boost.Function: 0.922 sec.
FastDelegate: 0.593 sec.

File Attachments

1) test_dll_data.zip, downloaded 599 times

Subject: Re: How to write a dll using Ultimate++?
Posted by mirek on Thu, 20 Sep 2007 19:38:06 GMT
View Forum Message <> Reply to Message

Quote:
Call to a function in DLL will always be a little bit slower.

In this case it is not a normal call vs dll call, but inlining vs dll call. For very specific applications, the impact can be huge.

Subject: Re: How to write a dll using Ultimate++?
Posted by Novo on Thu, 20 Sep 2007 20:16:57 GMT
View Forum Message <> Reply to Message

luzr wrote on Thu, 20 September 2007 15:38Quote:
Call to a function in DLL will always be a little bit slower.

In this case it is not a normal call vs dll call, but inlining vs dll call. For very specific applications, the impact can be huge.

I won't argue with you. Words *huge*, *tremendous*, *incredible*, *very specific*, e.t.c. don't explain anything.

I've done my best. You have a DLL, which exports global data, and you have results of a test that clearly shows that there is no significant difference between call to a function in DLL and call via pointer to a function.

The rest is up to you.

Subject: Re: How to write a dll using Ultimate++?
Posted by Novo on Thu, 20 Sep 2007 20:36:39 GMT
View Forum Message <> Reply to Message

luzr wrote on Thu, 20 September 2007 15:38Quote:
Call to a function in DLL will always be a little bit slower.

In this case it is not a normal call vs dll call, but inlining vs dll call. For very specific applications, the impact can be huge.

BTW. Can somebody beat me in data insertion speed with SQLITE, JET, ODBC ? Those are DLL-based tests ...

Have fun.

Subject: Re: How to write a dll using Ultimate++?
Posted by mirek on Thu, 20 Sep 2007 23:40:33 GMT
View Forum Message <> Reply to Message

Novo wrote on Thu, 20 September 2007 16:16luzr wrote on Thu, 20 September 2007 15:38Quote:
Call to a function in DLL will always be a little bit slower.

In this case it is not a normal call vs dll call, but inlining vs dll call. For very specific applications, the impact can be huge.

I won't argue with you. Words *huge*, *tremendous*, *incredible*, *very specific*, e.t.c. don't explain anything.

Haha, a very good point I wish I had a spare hour for the benchmark. Anyway, you are absolutely right that guts feeling does not work in most case where the code performance is involved

Mirek