

Re: Compiling from native to managed VC++ /clr option dramatically enlarge static lib size.

## Re: Compiling from native to managed VC++ /clr option dramatically enlarge static lib size.

---

*Source:*

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.vc/2006-06/msg00147.html>

---

- *From:* "dovgani" <[dovgani@xxxxxxxxxxx](mailto:dovgani@xxxxxxxxxxx)>
  - *Date:* 13 Jun 2006 05:29:40 -0700
- 

Marcus Heege wrote:

"Bruno van Dooren" <[bruno\\_nos\\_pam\\_van\\_dooren@xxxxxxxxxxx](mailto:bruno_nos_pam_van_dooren@xxxxxxxxxxx)> wrote in message [news:O0%236qLmjGHA.1276@xxxxxxxxxxxxxxxxxxxxxxxxxxxx](mailto:news:O0%236qLmjGHA.1276@xxxxxxxxxxxxxxxxxxxxxxxxxxxx)

I have an unmanaged MFC project. The output is static lib. I would like to compile using /clr option. The native lib size is 64 megs and with /clr and /O1 options is 940 megs. Is it possibly Metadata enlarge size so dramaticly? And I would like to know any suitable solution of my problem.

I guess that you compile all your souce files with /clr. This is seldom necessary. Consider compiling only those filed with /clr that you really need to compile to managed code. In the best case, write managed code in a separate source file and call it from you native code.

But even so, this is a 15 times increase in size. is this normal?

--

Kind regards,  
Bruno van Dooren  
[bruno\\_nos\\_pam\\_van\\_dooren@xxxxxxxxxxx](mailto:bruno_nos_pam_van_dooren@xxxxxxxxxxx)  
Remove only "\_nos\_pam"

If you compile everything with /clr, this is possible. Roughly spoken,

Re: Compiling from native to managed VC++ /clr option dramatically enlarge static lib size.

Re: Compiling from native to managed VC++ /clr option dramatically enlarge static lib size.

without /clr, the only metadata is the decorated method names. With /clr, for every function, there is the decorated name (as before), an entry in the method table, the undecorated name, a custom attribute containing the decorated name, the method's signature, including C++/CLI specific signature modifiers ...

As Carl has mentioned, in addition to that there is a P/Invoke function definition for every native function that is called in the code. For each P/Invoke function definition, there is a similar overhead as described above. If the lib contains many functions that wrap another functions 1:1, the P/Invoke metadata can be a huge amount.

Marcus

Thank you guys.

A little more details.

I have large(very large) application. Static library I am talking about is presentation lib

I would like to use a new .NET controls (menu, toolbar).

Another words just to dress a new shell.

After our discussion. I think the best way is to rewrite that lib in managed code.

But this is not easy job.

.