

# Re: Managed code, .H and .CPP

---

*Source:*

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.vc/2005-11/msg00182.html>

---

- *From:* "Peter Oliphant" <[poliphant@xxxxxxxxxxxxxxxxxxxxx](mailto:poliphant@xxxxxxxxxxxxxxxxxxxxx)>
  - *Date:* Thu, 10 Nov 2005 13:16:41 -0800
- 

Hi Ignazio,

Personally, I try to put everything in the .H file. The reason is it makes the generated code in-line (performance) and all in one place (source organization). With so much RAM in computers nowadays it usually makes sense to optimize for speed rather than size, and in-line usually contributes in this regard.

However, there are times you MUST put stuff in a .CPP file. This is when you have two classes which both must reference each other in a 'detailed' way (i.e., each calls a method in the other). If just the pointer is needed to an instance of another class, one can still stay in the .H by declaring the other class before it is used without defining it.

That is, this is ok to stay in two different .H files:

-----  
-----

header file A:

```
#include "B.h"
class A
{
void Method_A()
{
m_B_Ptr->Method_B();
};
:
B* m_B_Ptr;
};
```

-----  
header file B:

```
class A;
class B
{
:
A* m_A_Ptr;
```

```
};
```

-----  
-----  
But this is NOT ok:  
-----  
-----

header file A:

```
#include "B.h"  
class A  
{  
void Method_A()  
{  
m_B_Ptr->Method_B();  
};  
:  
B* m_B_Ptr;  
};
```

-----  
header file B:

```
class A; // replacing this with #include "A.h" causes circular definition  
errors  
class B  
{  
void Method_B()  
{  
m_A_Ptr->Method_A(); // error since doesn't know about Method_A  
}  
:  
A* m_A_Ptr;  
};
```

-----  
-----  
This can be corrected by changing the B.H file and adding a B.CPP like so:  
-----  
-----

header file B:

```
#include "A.h" ; // change declaration to included A.h  
  
class B  
{  
void Method_B() ; // declaration only  
:  
A* m_A_Ptr;  
};
```

-----  
CPP file B:

```
#include "B.h" ;
```

## Re: Managed code, .H and .CPP

```
ClassB::Method_B()
{
m_A_Ptr->Method_A() ; // no error
}
```

---

---

Note that B.CPP gets its definition of class A from the include of A.h in B.h.

Also, it is **STRONGLY** recommended you place a "#pragma once" at the top of each .H file (but never in a .CPP file) which means if it occurs more than once in a definition the compiler will ignore all but one include of this header file. That way you can have more than one .H file included in the same definition, all of which have a common .H included (e.g., program equates/defines header), file without problems...

[==P==]

"Ignazio" <Ignazio@xxxxxxxxxxxxxxxxxxxxxxxxxxxx> wrote in message [news:EE7E26EB-CA43-43A7-B748-50B103D53DCA@xxxxxxxxxxxxxxxxxxxx](mailto:news:EE7E26EB-CA43-43A7-B748-50B103D53DCA@xxxxxxxxxxxxxxxxxxxx)

```
> Thank you. If I understood well, I can write all the code of the forms in
> their .H without any problem.
> The only problem is with cross-dependance of two classes. For example:
>
> // A.h
> #include "B.h"
>
> ref class A {
> void f () {
> B::g();
> }
> void g () {
> Console::WriteLine("test");
> }
> };
>
> // B.h
> #include "A.h"
> ref class B {
> void g() {
> A::g();
> }
> };
>
> The compiler encounters an error, as it is right to do in C++. So, it is
> necessary to write a header and a cpp file for one of the two classes...
> Don't you find it a little tricky that I need just the .H for some files
> and
> both for other files?
```

Re: Managed code, .H and .CPP

>  
> Ignazio  
>  
>  
> "Carl Daniel [VC++ MVP]" wrote:  
>  
>> Ignazio wrote:  
>>> When creating forms with Visual C++ 2005, all the code for building  
>>> the interface (the InitializeComponent method) and event handlers are  
>>> set in the .H file, as they were inline methods. So I ask, for  
>>> managed code they are effectively inline methods or they are handled  
>>> indifferently if they are in the class declaration or outside of it?  
>>> Do I still have to write declarations and definitions in different  
>>> files for avoiding multiple compiling when including the headers or  
>>> it is handled more like C# for managed code?  
>>  
>> Methods defined in the class body are (I believe) NOT implicitly inline  
>> in a  
>> managed class – they have to be represented in the IL as methods, and  
>> then  
>> it's up to the JIT to inline them if it decides to.  
>>  
>> That said, methods defined in the class definition are linked "as if"  
>> they  
>> were inline, just as with standard C++, and fairly analogously to C#.  
>>  
>> -cd  
>>  
>>  
>>

---

• **References:**

◆ **Re: Managed code, .H and .CPP**

◇ From: Carl Daniel [VC++ MVP]

- Prev by Date: **Simple Mixed ATL DLL Project Crash At Exit with VS.NET 2005**
- Next by Date: **Re: Managed C++ will not be supported in the future...**
- Previous by thread: **Re: Managed code, .H and .CPP**
- Next by thread: **MFC or Windows Forms for Dialogue-based DirectX Audio App?**
- Index(es):
  - ◆ **Date**
  - ◆ **Thread**