

# Re: Forward references and recursion

---

*Source:* <http://www.tech-archive.net/Archive/VC/microsoft.public.vc.stl/2005-12/msg00024.html>

---

- *From:* muchan <[muchan@xxxxxxxxxxxx](mailto:muchan@xxxxxxxxxxxx)>
  - *Date:* Fri, 09 Dec 2005 11:29:28 +0100
- 

Corey Cooper wrote:

First let me say that someone will say that I should re-consider my solution and just not do this. If I don't find an easy solution I will, but it will mean cutting and pasting to duplicate a lot of code, and isn't that what the C++ and the STL are supposed to avoid? At any rate:

I have a two classes, each contains a list of the other, like so:

```
class A
{
    ...
    list<B> Blist;
}

class B
{
    ...
    list<A> Alist;
}
```

The declaration for class B is, of course, fine, but the list<B> in class A generates many errors. I've tried a simple forward declaration

(snipped)

The basic problem, IMO, is that a class need to know the size of the classes it "contains".

It needs not only the name, but it need to know the declaration, to calculate the size of that member, to calculate the size of itself...

## Re: Forward references and recursion

So, philosophically speaking ;) you need not only "forward declaration", but "forward definition" of class structure with all members of known size, which is not possible in recursive way, as you have tried.

A workaround, is to change the member as the pointer to the list, since the size of pointer for any object is known to compiler, it can decide the size of the class structure with only forward declaration of the name. That is,

```
class A;  
class B;
```

```
class A  
{  
...  
    list<B>* pBlist;  
};
```

```
class B  
{  
...  
    list<A>* Alist;  
};
```

and provide the constructor which initialises the pointer to either newly allocate or already existing list<>.

You may decide to change only class A or class B to contain a pointer, but in your place I'd probably decide for symmetry and similar interfaces. Or rather, I'm not very sure if they really should cross linked together,

(Can you draw down the relation of actual object A and B, and the contents of each A's and B's inside the lists inside A's and B's? Sorry, I can't imagine it well.

Probably you needs just a outside lists of plain data classes, and some ober classes to define the relations to each of them... In short, I think it's better take some hours to think about the designs

## Re: Forward references and recursion

of real objects your program should represent.

Think about some "atom" classes of data, and manager classes to manage them...  
without putting the data and mechanism inside just two classes.  
Well, just IMHO, or IMH Imagination.)

muchan