Re: C++ in ternms of C

Source: http://www.tech−archive.net/Archive/VC/microsoft.public.vc.language/2005−04/msg01164.html

- *From*: comeau@xxxxxxxxx (Greg Comeau)
- *Date*: 22 Apr 2005 10:00:54 −0400

In article <DeltaOne.1ntz5w@xxxxxxxxxxxxxxxxxxxxx>,
DeltaOne <DeltaOne.1ntz5w@xxxxxxxxxxxxxxxxxxxxx> wrote:
>Hi, I am planning to study how we can implement things that we implement
>in c++ in c and using a bit asm also. First thing i want to know is
>about class and struct difference. How is the private and public part
>implemented in the c++ exactly or how is the internal representation
>done. I tried to analise the ASM file produ ced my the VC++ compil er.
>But there i wrote the following code class t{ private: __int8 i;
>public: void valF(); }; int main(){ t dat; t dat1; t p; dat.valF();
>p.valF(); dat1. valF(); return 1; } void t::v alF(){ i=3; return ; } in
>the above class "t" i changed the specif ier of "i" from private to
>public. but in both cases th e ASM code generated was same. Even the
>exe file was same.Then how in the ru n time it knows which data is
>private and which data is public. Is is implem ented by the runtime
>libraries? or how? is there any site or book from which i can ge t more
>info on it? Please let me know this first case.Hos can i im plement this
>private and public thing in C and if required some ASM inline a lso. I
>want to use a pure C compiler for this not a C++ compiler.How is it r
>epresented by the c++ compiler in translation to intermediate stage.

Check out Lippman's Inside the C++ Object Model.
Details on it can be found at http://www.comeaucomputing.com/booklist

Generally speaking, the issue of C++ to C translation is not
trivial. In fact it is grossly complicated. Even if you are
trying to set up a higher level coding stylesheet or something
for this, mimicing things too deeply may not be the best approach.

That said, on the issue above, note that access specifiers
such as private are protection against coding accidents,
and not against fraud or security per se. Therefore,
at runtime, a C like struct in C++ is going to usually
look exactly that same as the same struct in C. Of course,
things complicate because C++ also have extensions and
differences over C, initialization and ordering issues, etc.
But if you talking about POD structs and only at the simplest
levels, the similarity is there.

Lastly, you ask about how it is representing by the C++ compiler

in the intermediate stage. It is of course at that point entered
into the respective tables with their respective characterstics.
The emitted output code, of whatever form, then needs to pull
from that info to determine what is what. In a normal POD struct,
there normally isn't anything else additional going on that the
compiler needs to consider over what a C compiler needs to consider.
As one move from POD struct to non−POD structs, they sky begins
to become the limit given all the possibilities, complexities, etc.
−−
Greg Comeau / Comeau for the Mac? Stay tuned.
Comeau C/C++ ONLINE ==> http://www.comeaucomputing.com/tryitout
World Class Compilers: Breathtaking C++, Amazing C99, Fabulous C90.
Comeau C/C++ with Dinkumware's Libraries... Have you tried it?
.

- ***References***:
  - ♦ ***C++ in ternms of C***
    - ◊ *From:* DeltaOne

- Prev by Date: ***Re: nested functions in VC++***
- Next by Date: ***Re: C++ in ternms of C***
- Previous by thread: ***Re: C++ in ternms of C***
- Next by thread: ***C++ in ternms of C***
- Index(es):
  - ♦ ***Date***
  - ♦ ***Thread***