Subject: Re: Difficulty with Class declaration
Posted by brokndodge on Tue, 06 Apr 2010 14:20:51 GMT
View Forum Message <> Reply to Message

Quote:I completely omit C/C++ references (ampersands in type definitions and functions calls), because they are simple to explain once you understand pointers. Also I did omit function pointers and other syntax sugar, which is not needed to understand the concept, because in the end, the concept is simple:
- values are numbers
- pointers are addresses to numbers (and address itself is implemented as a simple number too, so for CPU there's no difference between number/pointer, it's just 13 and 100, but compiler+syntax does allow you to use them in different context and warn you when you use it plainly wrong)

Your short introduction is actually a little more clear than the various tutorials and howto's I've read in the last week. The "why" still alludes me tho. I currently have two class's: UDMS and Popups. UDMS::non_static_members needs to call Popups::non_static_members, but doesn't need to know anything about them or get anything from them. I want Popups::non_static_members to hand some work off to MyAppSQL::non_static_members. Sometimes Popups::non_static_members might need to get some information back, but most of the time it will just pass information along.

I haven't implemented the new class yet because I still can't get UDMS to make a direct call to Popups. Right now I have a work around to call one Popups::non_static_member from UDMS, but it's ugly and will result in either a switch case or 30 additional functions in UDMS just to call various members of Popups. Then I have 8 more modules to write that will all need to be called from UDMS.

I tried to create just one class with each group of members in seperate .cpp files, but I didn't get the declaration right or something. I think my lack of understanding pointers is where the problem lies. I figured out the THISBACK macro is using a pointer "this". MingW doesn't want me to use "the address of a bound member function to form a pointer to a member function". The c++ shortcut << yields the same result, so it seems to mean the same thing as THISBACK.

So, it's the how and why of pointers that I'm missing. Or, maybe I shouldn't be using a pointer at all to call members of Popups. That is where my wrapper comes in. No pointer! void UDMS::CallToPopupsMember(){
Popups pop;
pop.Member();
};No pointer! But how do I do:void UDMS::ProspectDetail()
{detail.Button.WhenAction = THISBACK(CallToPopupsMember);
};without the wrapper function? THISBACK is a pointer. Right? If I just do Popups pop;
detail.Button.WhenAction = THISBACK(pop.Member());MingW throws a fit. The next set of windows I'm working on are on hold until I get this figured out. I'm writing the schema.sch when I get tired of looking for the solution. Then I come back to this problem for a lil while.

Best I can come up with right now is a switch case. Hand a String[Array] off to UDMS::CallToPopups that includes the Popups::Member I want to call plus the arguments for that

member. Then parse that String with a switch to figure out which Popups::Member needs called and pass the arguments to it. Seems ugly to me when I could just make a direct call.

I refuse to build my app with GTK-Perl. I don't want to hand code all those windows and I don't like Glade! So ugly will have to do for the first alpha release. I figure, if I can get the biggest part of the project coded, I will learn a lot along the way and can fix the ugliness before release candidate 1.

Thank you for your response Mr_Ped. Any further insight you could provide would be much appreciated. I know that I have tackled a pretty large project for my first c++ program. But, I believe it is the best language for this project and the project is the primary focus. I'll pick up the rest along the way.

PS. Ultimate++ is excellent and the community here is phenomenal!!! I understand more and more of the example code every day. I keep going back over them each day to gleam the next piece of insight.
