

Re: formal parameter 1 different from declaration?

Re: formal parameter 1 different from declaration?

Source: <http://www.tech-archive.net/Archive/VC/microsoft.public.vc.language/2008-12/msg00522.html>

- *From:* Robby <Robby@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Fri, 19 Dec 2008 09:05:01 -0800
-

Hi guys!

Thanks for getting back.

I mean you shouldn't define the same thing in two different ways. Note the declaration of callerFunction:

```
LR callerFunction(LR(*WNDPROC)(HWND, long, int, int), ...);
```

Okay, now the caller function and the typedefed are declared with the same parameters. Here is the typedef one:

```
typedef LR (*WNDPROC)(HWND, long, int, int);
```

It's not rocket science. Consider:
struct tagWnd; // forward declaration

I have done the forwards declaration just as you showed it. Unfortunately, I got something like 142 errors. I am sure that the forwards declaration is right, I must be doing something else that is incorrect.

It's also not clear to me why you want HWND to be a struct. Why not just
typedef tagWnd* HWND;

I didn't think about that one!!! :-) Thanks Igor!

Nonetheless for now I will continue debugging this with the original code!

Re: formal parameter 1 different from declaration?

Did you do this before or after you received the diagnostic in question?

Yes, but I have corrected this... see above !

Almost 200 lines of code and not a single indent. You really need to give us a fighting chance to help you.

Trust me Barry, I am already so embarased for posting such an attrocity! But I have no choice I am confused about the *whole* thing !

```
//CALLBACK FUNCTION
LR callerFunction(LR(*WNDPROC)(HWND, long, int, int) ,
HWND h, long m, int w, int l)
{return WNDPROC(h, m, w, l);
}
```

You actually define the function (not declare its prototype) in a header file?!? If acm.h ever gets included into more than one translation unit, you are going to have numerous link errors. While not illegal, object and function definitions should never appear in header files.

Understood. I don't know why I put that there? Okay, I have relocated it at the end of krn.c

```
===== krn.c
#include <stdio.h>
#include "krn.h"
#include "acm.h"
```

By virtue of this include, callerFunction is defined here.

Okay! for now I have defined it at the top of krn.c like this:

```
=====
#include <stdio.h>
// #include <stdlib.h>

#include "krn.h"
```

Re: formal parameter 1 different from declaration?

Re: formal parameter 1 different from declaration?

```
#include "acm.h"

//CALLBACK FUNCTION
LR callerFunction(LR(*WNDPROC)(HWND, long, int, int) ,
HWND h, long m, int w, int l);
=====
```

If you look carefully at this function signature and compare it to the one provided in the definition of callerFunction, you will see they are not the same. In particular, the first argument is different. One takes a struct tagHwnd and the other takes a long.

This has been changed. See above!

I noticed that in some of the functions I have a parameter called *pKM_MSG. This is also a global pointer to a tagMsg structure. Could that confuse the compiler? I retyped it like this:

```
zzz = callerFunction(((pKM_MSG->hwnd).handle)->lPFNWndProc,
pKM_MSG->hwnd, pKM_MSG->msg, pKM_MSG->wParam , pKM_MSG->lParam );
```

so I retyped the caller function like this:

```
zzz = callerFunction(((K->hwnd).handle)->lPFNWndProc,
K->hwnd, K->msg, K->wParam , K->lParam );
```

In spite of all the changes, I still get 135 errors.... discouraged!

To avoid confusion (since several changes took place since the original sample program) I will start a new post called "formal parameter 1 different from declaration_2" which will contain the whole program with the new changes.

—
Best regards
Roberto

"Barry Schwarz" wrote:

On Thu, 18 Dec 2008 14:20:04 -0800, Robby
<Robby@xxxxxxxxxxxxxxxxxxxxxxxxxxxx> wrote:

Hello,

Okay, I have to admit that my program sample below is rather large and I do

Re: formal parameter 1 different from declaration?

Re: formal parameter 1 different from declaration?

deeply apologize for this for it really is a post of last resort because I unfortunately find myself stuck like I have never been stuck before. Even though the program compiles successfully and has 1 warning, its been all day

This is self-contradictory. Unless you know the exact cause and significance of the diagnostic (in which case you wouldn't be asking us), any diagnostic means the program did not compile successfully.

The fact that the compiler writer chose to call this diagnostic a warning instead of an error is unfortunate. But that decision in no reflects the seriousness of the problem as it relates to your code.

snip

heres the warning!

```
c:\_dts_programming\c_programming\c\proczzz\proczzz\krn.c(96) : warning  
C4028: formal parameter 1 different from declaration
```

The warning points to the following line:

```
zzz = callerFunction(((pKM_MSG->hwnd).handle)->lPFNWndProc,  
pKM_MSG->hwnd,  
pKM_MSG->msg, pKM_MSG->wParam , pKM_MSG->lParam );
```

which is the call that calls the caller function.

What I am trying to do, is pretty similar to the way programming is done in native VC++ code where we register the windows and create them and show them

and then calling the correct WinProc callback function to process some messages. I am testing the code in VC++ but with .c files before actually code it in the PIC compiler. I am obviously doing this the best I can and it seems to be more difficult than I thought.

Before you view the code, there is something unethical about some of the declarations I have done in the krn.h file.

The line:

```
typedef LR (*WNDPROC)(long, long, int, int);
```

should be:

```
typedef LR (*WNDPROC)(HWND, long, int, int);
```

Did you do this before or after you received the diagnostic in question?

Re: formal parameter 1 different from declaration?

But if I do this, then the tagHwnd structure is out of scope because it is declared later on. If I decide to declare the tagHwnd structure before the typedef statement, then the tagWwnd structure will be out of scope and so forth. So, I really don't know what I should do to correct the problem. I guess you will come across this once the code is reviewed.

There is lots of code I had to remove since it was irrelevant to the problem! Again, I sincerely thank anyone who can give me some insight or recommendation on what should be relocated and how... since I don't know what else to try. If the sample is too long, let me know I will further try to reduce it.

Here's the code:

Almost 200 lines of code and not a single indent. You really need to give us a fighting chance to help you.

```
===== krn.h
typedef long *LR;
typedef long WPARAM;
typedef long LPARAM;

typedef LR (*WNDPROC)(long, long, int, int);

typedef struct tagWwnd{
WNDPROC lpfmWwndProc;
int style;
long backGround;
long titleMsg;
long extra;
} WND, *pWwnd;

typedef struct tagHwnd{
WND *handle;
} HWND;

typedef struct tagRwp{
HWND hwnd;
int showWinEnable;
} RWP;

typedef struct tagMsg{
HWND hwnd;
long msg;
WPARAM wParam;
LPARAM lParam;
}
```

Re: formal parameter 1 different from declaration?

```
long time;
} KM_MSG, *pKM_MSG;

short ULC_KERNEL_register_wp ( WND *wnd);
HWND CreateWindow ( int caption, int wType);
void ShowWindow ( HWND *hwnd, int nCmdShow);
long ULC_KERNEL_get_msg ( KM_MSG *pKM_MSG);
void ULC_KERNEL_update_all_inputs ( );
void ULC_KERNEL_dispatch_message ( KM_MSG *pKM_MSG);
```

===== acm.h

```
int ACM152_MAIN();
```

```
//CALLBACK FUNCTION
```

```
LR callerFunction(LR(*WNDPROC)(HWND, long, int, int) ,
HWND h, long m, int w, int l)
{return WNDPROC(h, m, w, l);
}
```

You actually define the function (not declare its prototype) in a header file?!? If acm.h ever gets included into more than one translation unit, you are going to have numerous link errors. While not illegal, object and function definitions should never appear in header files.

This definition says that callerFunction is a function that returns an LR (which is just a pointer to long) and takes the following arguments:

- a pointer to a function (described below)
- an HWND (which is a struct tagHwnd)
- a long
- an int
- and a second int

The function pointed to by the first argument returns an LR (still a long*) and takes the following arguments:

- an HWND (still a struct tagHwnd)
- a long
- an int
- and a second int

```
RWP rwp[3] = {0,0,0,0,0};
```

===== krn.c

```
#include <stdio.h>
#include "krn.h"
#include "acm.h"
```

Re: formal parameter 1 different from declaration?

By virtue of this include, callerFunction is defined here.

```
int main()
{
ACM152_MAIN();
return 0;
}

short ULC_KERNEL_register_wp ( WND *wnd)
{
int i;
short rVal = 0;

for(i=0; i < (sizeof(rwp)) / (sizeof(rwp[0])); i++)
{
if( (rwp[i].hwnd).handle == 0)
{
(rwp[i].hwnd).handle = wnd;
return (rVal = 1);
break;
}}
return rVal;
}

long ULC_KERNEL_get_msg ( KM_MSG *pKM_MSG)
{
ULC_KERNEL_update_all_inputs();

//other code not shown. For now pKM_MSG->msg is set to 1
pKM_MSG->msg = 1;

// RESET MESSAGE STRUCTURE
//pKM_MSG->hwnd = 0;
pKM_MSG->msg = 0;
pKM_MSG->lParam = 0;
pKM_MSG->wParam = 0;
pKM_MSG->time = 0;

return pKM_MSG->msg;
}

void ULC_KERNEL_update_all_inputs ( )
{
int i;

for(i=0; i < ((sizeof(rwp))/(sizeof(rwp[0]))); i++)
{
if(rwp[i].showWinEnable == 1 && // Is window enabled for show process?
(rwp[i].hwnd).handle != 0) // Is window registered?
```

Re: formal parameter 1 different from declaration?

```
{
//InsertMessage(KM_CREATE); // Send a KM_CREATE message
break;
}
}
//... other code ...
}

void ULC_KERNEL_dispatch_message ( KM_MSG *pKM_MSG)
{
int i;
LR zzz;
// Show only the right window respective to current Procedure!
for(i=0; i < ((sizeof(rwp)) / (sizeof(rwp[0]))); i++)
{
if((rwp[i].showWinEnable == 1 || // Is window enabled & sent KM_CREATE
msg.
rwp[i].showWinEnable == 2) // Window enabled for show.
&& (rwp[i].hwnd).handle != 0) // Is window registered?
{
rwp[i].showWinEnable = 2; // Set window enabled for show only.
pKM_MSG->hwnd = rwp[i].hwnd; // Read the Window handle!
break;
}
}

// Call the CALLBACK function!
zzz = callerFunction(((pKM_MSG->hwnd).handle)->lPFNWndProc,
```

Let's examine the type of this argument.

pKM_MSG is a pointer to struct tagMsg

struct tagMsg contains a member hwnd which is a struct tagHwnd

struct tagHwnd contains a member handle which is a pointer to struct tagWwnd

struct tagWwnd contains a member lPFNWndProc which is a WNDPROC

WNDPROC is a pointer to function that returns LR (still a long*) and takes the following arguments:

a long

a second long

an int

and a second int

If you look carefully at this function signature and compare it to the one provided in the definition of callerFunction, you will see they are not the same. In particular, the first argument is different. One takes a struct tagHwnd and the other takes a long.

Not only is the diagnostic correct but if you executed the code you would rapidly enter the realm of undefined behavior.

Re: formal parameter 1 different from declaration?

So now the question is: If callerFunction is designed to accept a pointer to a type A function, why are you sending it the pointer to a type B function?

—

Remove del for email