

Re: How to pass a struct to a function

Source: http://coding.derkeiler.com/Archive/C_CPP/comp.lang.c/2007-03/msg04893.html

- *From:* Keith Thompson <kst-u@xxxxxxx>
 - *Date:* Wed, 28 Mar 2007 14:49:54 -0700
-

"Bill" <bill.warner@xxxxxxxxxxxx> writes:

On Mar 28, 9:48 am, Chris Dollin <chris.dol...@xxxxxx> wrote:

Bill wrote:

I am trying to pass a struct to a function. How would that best be accomplished?

Assuming that the function has an argument of the right structure type, write an expression evaluating to your struct -- the name of a variable of that struct type is popular -- in the appropriate argument position in a call to that function.

What am I missing?

Please don't quote signatures. Trim quoted material to what's necessary for your followup to make sense to someone who hasn't see the parent article.

Here is what I have so far, won't even compile. And I am:

```
#include <stdio.h>
#include <string.h>
```

Your program doesn't use anything from <string.h>, so this #include directive is unnecessary. But, as we'll see, it *should**, so yes, you'll need this #include directive.

```
int AddEntry(struct entry);
```

At this point, you haven't declared a type "struct entry". Because

Re: How to pass a struct to a function

it's an incomplete struct type, there are some interesting details about how this declaration is handled, but that's not relevant; this is not what you want, and you'll need to fix it.

```
struct{
  char fName[51];
  char lName[51];
  char Phone[13];
}Entry;
```

Now you declare a struct object, but you **still** haven't declared a type "struct entry". In fact, you don't declare "struct entry" (other than as an incomplete type) anywhere in your program. What you've done here is (a) declared an anonymous struct type, and (b) declared a single object of that type, named "Entry". If all you want is a single object, that might be sensible thing to do (or you might as well declare the members as individual variables). But since you want to pass it a