

Re: "data hiding" prototype code

Source: http://coding.derkeiler.com/Archive/C_CPP/comp.lang.c/2007-04/msg01442.html

- *From:* "sofeng" <sofengboe@xxxxxxxx>
 - *Date:* 12 Apr 2007 09:41:53 -0700
-

On Apr 11, 4:39 pm, "Jonas" <spamhereple...@xxxxxxxx> wrote:

"sofeng" <sofeng...@xxxxxxxx> wrote in message

news:1176328415.501183.20910@xx

Thank you for your reply. See my comments below.

On Apr 11, 1:18 pm, "bluejack" <bluu...@xxxxxxxx> wrote:

First of all, making the variables static at file scope doesn't really achieve the goal of encapsulation, particularly when you publish the struct definition in the header. Secondly, static variables at file scope are not much better than globals: they're certainly not thread safe, and you open yourself to other sorts of errors as well.

I'm not sure what you mean by "publish the struct definition in the header". I only put the typedef in the header— that doesn't publish the struct definition, right?

In your header file, you have

```
typedef struct {  
double data1;  
double data2;  
double data3;  
} ALGORITHM_DATA;
```

Instead, separate the typedef and the struct definition into header and

Re: "data hiding" prototype code

implementation files:

```
/* foo.h */
typedef struct foo_st foo_t;

/* foo.c */
#include "foo.h"

struct foo_st {
    int bar;
};
```

Now, struct foo_st will be undefined outside of foo.c, which is what you want to achieve. Pointers to foo_t can be defined by users of foo, but foo_t objects cannot, leaving your code in charge of this and anything that has to do with foo_t internals.

--
Jonas

But struct foo_st is synonymous with foo_t so why does it matter that it is available outside of foo.c? Don't we really care about the actual variable definition (where the memory is allocated) of type "struct foo_st" or "foo_t"? For example, for the definition of foo_var, "foo_t foo_var", foo_var should not be made available outside of foo.c.

.