Subject: Re: STRUCT_ASSIGN
Posted by Martin Schultz on Sun, 23 Aug 1998 07:00:00 GMT
View Forum Message <> Reply to Message

Robert S. Mallozzi wrote:
>
> Hi all,
>
> I have a structure defined as follows:
>
> a = { f1: 0, f2: { x: 0, y: 0}}
>
> [...]

> So, must I resort to field-by-field copy? The problem
> is that "a" is actually an object ("self"), which has
> some member variables that are large structures.
> A field-by-field copy would be tedious, as each
> structure has on the order of 30 fields, or so.
> I wanted to write a generic "set" method that will
> initialize the object's structures with some data.
> If I must do a field-by-field copy, I would then
> have to have several "set" methods, each of which is
> specialized for each of the different stuctures
> within the object.
>

As Mark points out, the problem here is that you are actually using two different anonymous structures. There has been a recent discussion in this newsgroup about a similar problem: If you type help,a.f2,data,/stru you will se something like :

```
** Structure <10304508>, 2 tags, length=4, refs=2:
   X               INT           0
   Y               INT           0
** Structure <10304708>, 2 tags, length=4, refs=1:
   X               INT          10
   Y               INT          20
```

The number in <> identifies the structure "type"...

> So, must I resort to field-by-field copy? The problem
> is that "a" is actually an object ("self"), which has
> some member variables that are large structures.
> A field-by-field copy would be tedious, as each
> structure has on the order of 30 fields, or so.
> I wanted to write a generic "set" method that will
> initialize the object's structures with some data.

> If I must do a field-by-field copy, I would then
> have to have several "set" methods, each of which is
> specialized for each of the different stuctures
> within the object.
>

What you describe here sounds like a typical application for pointers.
Nesting structures within structures within structures ... can usually
only be achieved with the help of these things. Here is an example:

```
  a = { f1:0, f2:ptr_new() }
  data = { x:10, y:20 }
  a.f2 = ptr_new(data,/NO_COPY)   ; careful: no_copy means
                                  ; "data" will be lost afterwards!

  help,*a.f2,/stru
```

You can test the contents of your sub-structure with tag_names():

```
  print,tag_names(*a.f2)
```

Hope, this gives a hint in the right direction,
Martin.

--
 -- -------
Dr. Martin Schultz
Department for Earth&Planetary Sciences, Harvard University
109 Pierce Hall, 29 Oxford St., Cambridge, MA-02138, USA

phone: (617)-496-8318
fax : (617)-495-4551

e-mail: mgs@io.harvard.edu
Internet-homepage: http://www-as.harvard.edu/people/staff/mgs/
 -- -------