Subject: Re: Q: Array of Structures
Posted by davidf on Fri, 12 Jun 1998 07:00:00 GMT
View Forum Message <> Reply to Message

Saeid Zoonematkermani (Saeid.Zoonematkermani@sunysb.edu) writes:

> The output of one of my applications is an anonymous structure. After a
> long session, I would like to create an array ofthese structures. I have
> tried the following unsuccessfully.
>
> IDL> tmp = [str1, str2]
> % Conflicting data structures: STR2,concatenation.
> % Execution halted at: $MAIN$
>
> I have also tried:
>
> IDL> tmp=replicate(str1,2)
> IDL> tmp(1)=str2
> % Conflicting data structures: STR2,TMP.
> % Execution halted at: $MAIN$
>
> I have string tags in the structure and my first thought was that the
> unequal string length from one structure to another is causing the
> problem. This does not seem to be the case since I picked two structures
> with identical lengths and the same error message popped up.
>
> Since these structures are created inside the same application, all the
> tags are the same and I had hoped there would be no problem with creating
> the arrays after the processing. My work around has been to write a simple
> procedure that assigns the tags individually for each array element.
>
> I am sure I am missing some thing very simple but with the world cup going
> on, my brain seems to be on vacation. Any help would be greatly
> appreciated. I am using IDL Version 5.1 (MacOS PowerMac). Thanks in
> advance,

It is odd how the universe works. I've never been asked this
question before and this week I encounter it twice! Weird. :-)

An array must be a collection of identical "things". For
example, if your anonymous structures were identical you
would have no problem. (By "identical" I mean that they
have the same fields and that those fields are defined
in exactly the same way.) For example:

```
a = { data:FltArr(10), name:'First Structure' }
b = { data:FltArr(10), name:'Second Structure' }
c = [a, b]
```

Notice that this works even though the name fields have strings in them of different lengths. The length of a string name is not important. It is the definition of the field as a string that is important.

Where you have problems, and where you will get the error you report above, is when the fields have different definitions. For example, this doesn't work:

```
a = { data:FltArr(10), name:'First Structure' }
b = { data:FltArr(15), name:'Second Structure' }
c = [a, b]
```

It doesn't work because even though the data field in B is defined as a float array, it has 15 elements in it as opposed to the 10 in the data field of A. Thus, these structures are not "identical".

A way around this, of course, is to create an array of pointers to structures that are defined differently. For example:

```
a = { data:FltArr(10), name:'First Structure' }
b = { data:FltArr(15), name:'Second Structure' }
c = [Ptr_New(a), Ptr_New(b)]
```

But now you have to do structure de-referencing when you refer to the structures:

```
Plot, (*c[1]).data
```

You also have to be sure to destroy your pointers when you are finished with them. Otherwise, they will persist in memory.

```
For j=0,N_Elements(c)-1 Do Ptr_Free, c[j]
```

Cheers,

David

David Fanning, Ph.D.
Fanning Software Consulting
E-Mail: davidf@dfanning.com
Phone: 970-221-0438
Coyote's Guide to IDL Programming: http://www.dfanning.com/