Subject: Structures and arrays of structures
Posted by rjp23 on Fri, 28 Oct 2016 10:12:47 GMT
View Forum Message <> Reply to Message

Hi,

This has me stumped at the minute so I was hoping someone could help as I thought it SHOULD be simple.

I have a anonymous structure in this form:

```
IDL> help, global_struct
** Structure <8341ea08>, 15 tags, length=138736, data length=138736, refs=1:
   DATE      FLOAT     Array[2392]
   ORIG      FLOAT     Array[2392]
   FUNC      FLOAT     Array[2392]
   POLY      FLOAT     Array[2392]
```

I have a second anonymous structure in this form:

```
IDL> help,aus_struct
** Structure <838a5ee8>, 15 tags, length=135780, data length=135778, refs=1:
   DATE      FLOAT     Array[2341]
   ORIG      FLOAT     Array[2341]
   FUNC      FLOAT     Array[2341]
   POLY      FLOAT     Array[2341]
```

I have many other structures in the same form and want them all to be contained in one item so they can be easily passed into other procedures, etc.

The tag names within the structure are the same so I can use code generically on any of them.

The structure name itself is generated from a string array ['global', 'aus', 'europe', 'us', etc]

I can't concatenate the structures into a single as they all have the same tag names. I thought I could create an array of structures but that doesn't seem to work. I wondered if I could create a structure of structures? main.global_struct.dat, main.aus_struct.date, etc but can't figure out how to.

Any suggestions would be really useful.

Cheers

Subject: Re: Structures and arrays of structures
Posted by rjp23 on Fri, 28 Oct 2016 11:58:13 GMT
View Forum Message <> Reply to Message

So I've never heard of hashes before but they seem to solve this.

I can even use hash.tostruct() to turn the hash into a nested structure like I wanted.

Far from the cleanest solution and I'd still like to know where I'm going wrong but it seems to work.

Subject: Re: Structures and arrays of structures
Posted by Markus Schmassmann on Fri, 28 Oct 2016 12:04:17 GMT
View Forum Message <> Reply to Message

On 10/28/2016 12:12 PM, rjp23@le.ac.uk wrote:
> This has me stumped at the minute so I was hoping someone could help as I thought it SHOULD be simple.
>
> I have a anonymous structure in this form:
>
> IDL> help, global_struct
> ** Structure <8341ea08>, 15 tags, length=138736, data length=138736, refs=1:
> DATE FLOAT Array[2392]
> ORIG FLOAT Array[2392]
> FUNC FLOAT Array[2392]
> POLY FLOAT Array[2392]
>
> I have a second anonymous structure in this form:
>
> IDL> help,aus_struct
> ** Structure <838a5ee8>, 15 tags, length=135780, data length=135778, refs=1:
> DATE FLOAT Array[2341]
> ORIG FLOAT Array[2341]
> FUNC FLOAT Array[2341]
> POLY FLOAT Array[2341]
>
> I have many other structures in the same form and want them all to
> be contained in one item so they can be easily passed into other
> procedures, etc.
>
> The tag names within the structure are the same so I can use code
> generically on any of them.
>
> The structure name itself is generated from a string array
> ['global','aus', 'europe', 'us', etc]
>
> I can't concatenate the structures into a single as they all have
> thesame tag names. I thought I could create an array of structures but that
> doesn't seem to work. I wondered if I could create a structure of
> structures? main.global_struct.dat, main.aus_struct.date, etc but can't
> figure out how to.

>
> Any suggestions would be really useful.
loc=['global', 'aus', 'europe', 'us']
cmd='main=create_struct('
for i=0,n_elements(loc)-1 do $
 cmd+="'"+loc[i]+"_struct',"+loc[i]+"_struct,"
cmd=strmid(cmd,0,strlen(cmd)-1)+")"
void=execute(cmd)

i hope that helps, Markus

Subject: Re: Structures and arrays of structures
Posted by rjp23 on Fri, 28 Oct 2016 12:15:16 GMT
View Forum Message <> Reply to Message

Hi Markus,

Thanks for the reply.

I'm already using exe to create the structures based on the strings but it looked like I was getting one key bit wrong.

When trying to create the main structure I was effectively doing:

main=create_struct(global_struct, aus_struct, etc)

but from your example I see it should be:

main=create_struct('global_struct', global_struct, 'aus_struct', aus_struct, etc)

Thanks for the help!

Rob

On Friday, October 28, 2016 at 1:04:20 PM UTC+1, Markus Schmassmann wrote:
> On 10/28/2016 12:12 PM, wrote:
>> This has me stumped at the minute so I was hoping someone could help as I thought it SHOULD be simple.
>>
>> I have a anonymous structure in this form:
>>
>> IDL> help, global_struct
>> ** Structure <8341ea08>, 15 tags, length=138736, data length=138736, refs=1:
>> DATE FLOAT Array[2392]
>> ORIG FLOAT Array[2392]
>> FUNC FLOAT Array[2392]
>> POLY FLOAT Array[2392]

>>
>> I have a second anonymous structure in this form:
>>
>> IDL> help,aus_struct
>> ** Structure <838a5ee8>, 15 tags, length=135780, data length=135778, refs=1:
>>   DATE   FLOAT   Array[2341]
>>   ORIG   FLOAT   Array[2341]
>>   FUNC   FLOAT   Array[2341]
>>   POLY   FLOAT   Array[2341]
>>
>> I have many other structures in the same form and want them all to
>> be contained in one item so they can be easily passed into other
>> procedures, etc.
>>
>> The tag names within the structure are the same so I can use code
>> generically on any of them.
>>
>> The structure name itself is generated from a string array
>> ['global','aus', 'europe', 'us', etc]
>>
>> I can't concatenate the structures into a single as they all have
>> thesame tag names. I thought I could create an array of structures but that
>> doesn't seem to work. I wondered if I could create a structure of
>> structures? main.global_struct.dat, main.aus_struct.date, etc but can't
>> figure out how to.
>>
>> Any suggestions would be really useful.
> loc=['global', 'aus', 'europe', 'us']
> cmd='main=create_struct('
> for i=0,n_elements(loc)-1 do $
> cmd+="'"+loc[i]+"_struct','"+loc[i]+"_struct,"
> cmd=strmid(cmd,0,strlen(cmd)-1)+")"
> void=execute(cmd)
>
> i hope that helps, Markus

Subject: Re: Structures and arrays of structures
Posted by Matthew Argall on Fri, 28 Oct 2016 12:59:32 GMT
View Forum Message <> Reply to Message

To create an array of structures, use the replicate function:

IDL> struct = {a: 0.0, b: 0.0}
IDL> struct_arr = replicate(struct, 100)
IDL>
IDL>
IDL> help, struct_arr

```
STRUCT_ARR      STRUCT    = -> <Anonymous> Array[100]
IDL>
IDL> help, struct_arr[0]
** Structure <11a1608>, 2 tags, length=8, data length=8, refs=3:
   A               FLOAT           0.00000
   B               FLOAT           0.00000
IDL> help, struct_arr.a
<Expression>    FLOAT     = Array[100]
```

Subject: Re: Structures and arrays of structures
Posted by rjp23 on Fri, 28 Oct 2016 13:54:40 GMT

View Forum Message <> Reply to Message

Thanks

That's what I tried initially but didn't seem to work. It came up with an error that structure_name__Define was not defined.

On Friday, October 28, 2016 at 1:59:54 PM UTC+1, Matthew Argall wrote:
> To create an array of structures, use the replicate function:
>
> IDL> struct = {a: 0.0, b: 0.0}
> IDL> struct_arr = replicate(struct, 100)
> IDL>
> IDL>
> IDL> help, struct_arr
> STRUCT_ARR   STRUCT   = -> <Anonymous> Array[100]
> IDL>
> IDL> help, struct_arr[0]
> ** Structure <11a1608>, 2 tags, length=8, data length=8, refs=3:
>   A   FLOAT   0.00000
>   B   FLOAT   0.00000
> IDL> help, struct_arr.a
> <Expression>   FLOAT   = Array[100]

Subject: Re: Structures and arrays of structures
Posted by penteado on Fri, 28 Oct 2016 15:43:00 GMT

View Forum Message <> Reply to Message

On Friday, October 28, 2016 at 6:54:42 AM UTC-7, rj...@le.ac.uk wrote:
> That's what I tried initially but didn't seem to work. It came up with an error that structure_name__Define was not defined.

That sounds like you were trying to use a named structure, which had not been defined. Note that structures can be anonymous, as is the case of all the structures that showed up above, or

named. I am not talking about the names of the variables holding the structures, I am talking about the names of the structure types (think of it like a type name such as float, integer, complex, etc).

This is a way to obtain that error:

```
IDL> a=replicate({structure_name},10)
% Attempt to call undefined procedure: 'STRUCTURE_NAME__DEFINE'.
% Execution halted at: $MAIN$
```

This is because there was not a type structure_named already defined, so IDL looked for a procedure called struct_name__define.pro which would define it, but it also did not find such a procedure. Note that doing

```
IDL> structure_name={a:0,b:1.0}
IDL> a=replicate({structure_name},10)
% Attempt to call undefined procedure: 'STRUCTURE_NAME__DEFINE'.
% Execution halted at: $MAIN$
```

Results in the same error. That is because the first line is just creating an anonymous structure, and assigning it to a variable that happens to be called structure_name. The structure type is not named.

This works:

```
IDL> structure_name={a:0,b:1.0}
IDL> a=replicate(structure_name,10)
IDL> help,a
A               STRUCT    = -> <Anonymous> Array[10]
```

The call to replicate is asking for a structure array that has 10 copies of the structure in the variable structure_name. Note there were no {} around structure_name in the call to replicate.

If what you wanted really was a named structure, it would be defined as:

```
IDL> c={structure_name,a:0,b:1.0}
IDL> help,c
** Structure STRUCTURE_NAME, 2 tags, length=8, data length=6:
   A               INT             0
   B               FLOAT           1.00000
IDL> a=replicate({structure_name},10)
IDL> help,a
A               STRUCT    = -> STRUCTURE_NAME Array[10]
```

Note that both a and c are not anonymous, they are of the type structure_name. The same result would be obtained with

```
IDL> a=replicate(c,10)
```

```
IDL> help,a
A               STRUCT    = -> STRUCTURE_NAME Array[10]
```

That said, you cannot use replicate with your global_struct to create an array that can also hold your aus_struct, because the types global_struct and aus_struct are not compatible: in global_struct the fields have 2392 elements, in aus_struct they have 2341 elements:

```
IDL> s1={a:fltarr(2392),b:fltarr(2392)}
IDL> s2={a:fltarr(2341),b:fltarr(2341)}
IDL> s3=[s1,s2]
% Conflicting data structures: S1,S2.
% Execution halted at: $MAIN$
IDL> s3=replicate(s1,2)
IDL> s3[0]=s1
IDL> s3[1]=s2
% Conflicting data structures: S3,S2.
% Execution halted at: $MAIN$
```

But they can be nested structures, as discussed above. Or, since you have strings to identify them, I would find it more neat to use a hash (or maybe an orderedhash, if it is IDL 8.3 or newer):

```
IDL> ss=['s1','s2'] ;names of the variables to put in the hash
IDL> h=hash()
IDL> for i=0,n_elements(ss)-1 do h[ss[i]]=scope_varfetch(ss[i])
IDL> help,h
H               HASH  <ID=155135  NELEMENTS=2>
IDL> print,h.keys()
s1
s2
IDL> help,h['s2']
** Structure <1abaed30>, 2 tags, length=18728, data length=18728, refs=3:
   A            FLOAT     Array[2341]
   B            FLOAT     Array[2341]
```