Subject: Re: Importing nested structures?
Posted by skerr on Wed, 16 Jun 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Hello again IDL gurus,

While waiting for a reply to my previous post, I thought a bit more about the problem and I think I figured out what the correct procedure is: you need to call IDL_MakeStruct for each nested structure, and assign the returned pointer to the "type" field of the parent structure. The procedure is illustrated in the sample code below. This sample works with IDL 4 when the program is run both standalone and from IDL with CALL_EXTERNAL.

Could anyone verify that this is indeed the correct way to nest structures for both IDL 4 and IDL 5?

Regards,
Stephen Kerr

---------- start of new sample prog ----------

```
#include <stdio.h>
#include "/usr/local/rsi/idl/external/export.h"

int main(int argc, char **argv)
{
  static IDL_LONG one = 1;

  static IDL_STRUCT_TAG_DEF substruct_tags[] = {
    {"P", 0, (void *) IDL_TYP_INT},
    {"Q", 0, (void *) IDL_TYP_INT},
    {0}
  };

  static IDL_LONG matrix_dims[] = {2,2,2};
  static IDL_LONG substruct_dims[] = {1,1};
  static IDL_STRUCT_TAG_DEF s_tags[] = {
    { "MATRIX", matrix_dims, (void *) IDL_TYP_FLOAT},
    { "SUBSTRUCT", substruct_dims, NULL},  /* NULL is a placeholder */
    {0}
  };

  typedef struct substruct {
    short int p;
    short int q;
  } SUBSTRUCT;
```

```
  typedef struct data_struct {
    float mat [2] [2];
    SUBSTRUCT sub;
  } DATA_STRUCT;

  static DATA_STRUCT s_data, *s_new_data;
  void *s;
  IDL_VPTR v;
  int i,j;

  IDL_Init(0,&argc,argv);

  s_data.mat[0][0] = s_data.mat[1][1] = 1.0;
  s_data.mat[0][1] = s_data.mat[1][0] = 0.0;
  s_data.sub.p = -999;
  s_data.sub.q = +1001;
  s = IDL_MakeStruct("SUBSTRUCT", substruct_tags);  /* define SUBSTRUCT */
  s_tags[1].type = s;                               /* tell s_tags about it */
  s = IDL_MakeStruct("FOO", s_tags);

  printf("Almost all folks.\n");

  v = IDL_ImportNamedArray("FOO",1,&one,IDL_TYP_STRUCT,
      (UCHAR *) &s_data, 0, s);

  s_new_data = (DATA_STRUCT *) v->value.s.arr->data;
  for (i=0;i<2;i++)
    for (j=0;j<2;j++)
      printf("%f\n",s_new_data->mat[i][j]);
  printf("%i\n", s_new_data->sub.p);
  printf("%i\n", s_new_data->sub.q);

  printf("That's all folks.\n");
}

------------------------ end of new sample prog ------------------------
```