Subject: Re: returning pointers and simple types from DLM functions
Posted by Nigel Wade on Mon, 14 Oct 2002 13:39:47 GMT
View Forum Message <> Reply to Message

Sidney Cadot wrote:

> Hi all,
>
> I am currently trying to program an IDL DLM that will provide a link to a
> previously-written C library. This is a first for me; while I have studied
> the IDL 5.4 external development guide quite extensively, I cannot find
> answers to the following two questions:
>
> (1) the C library yields opaque pointers as results of some functions that
> I need to pass back to IDL, in order for the IDL program to be able later
> on to pass these back into other library routines. Question: what type
> should I use to represent pointers (IDL_PTR or IDL_MEMINT) and how do I
> make sure my code works properly on both 32- and 64-bit platforms?

I use a byte array of length determined by sizeof() in C. That way you can
use the same code on 32 and 64 bit platforms.

This is a test function which returns a pointer as a return argument and as
a positional parameter. The only difference is that the temp variable is
copied into to the positional parameter to give it permanence, whereas the
return value is the temporary so it gets cleared up for you if you don't
use the return value (e.g. print,func(), where the return value is never
stored).

```
#include <stdio.h>

#include "export.h"


IDL_VPTR testfunc(int argc, IDL_VPTR argv[] ) {

   IDL_VPTR result;
   static int i = 32;
   static double d = 45.0;
   int *iptr = &i;
   double *dptr = &d;
   char *cptr;

   IDL_MEMINT dims[1];

   /*
    * pass back the value of the double pointer into
    * the first argument
    */
   /* set dims[0] to the number of "bytes" needed to hold the pointer */
   dims[0] = sizeof dptr;

   /* create a temporary byte array of the required length */
   cptr = IDL_MakeTempArray( IDL_TYP_BYTE, 1, dims, IDL_ARR_INI_NOP,
&result );

   /* copy the pointer into the byte array */
   memcpy( cptr, &dptr, dims[0] );

   /* copy the temp variable into the first parameter */
   IDL_VarCopy( result, argv[0] );

   /*
    * return the pointer to the integer as the function return value
    */

   dims[0] = sizeof iptr;
   cptr = IDL_MakeTempArray( IDL_TYP_BYTE, 1, dims, IDL_ARR_INI_NOP,
&result );


   memcpy( cptr, &iptr, dims[0] );

   return result;

}
```

>
> (2) How do I construct an IDL_VPTR when my function should simply return a
> basic type, (say, an IDL_INT)? Should I use IDL_Gettmp() for this, and if
> so: should I be setting the "type" and/or "flags" field myself? This may
> seem quite silly to ask, but I didn't find an example of this or explicit
> instructions in the external development guide.

If it's the return value, yes, use Gettmp(). Then set the type and the
value:

```
    IDL_VPTR result;

    result = IDL_Gettmp();

    result->type = IDL_TYP_LONG;
    result->value.l = 1234;

    return result;
```

If it's a parameter you can still use Gettmp, but you need to copy the variable into the parameter, just as with the array above.

Alternatively, you can do this:

```
   IDL_ALLTYPES result;

   result.l = 32;
   IDL_StoreScalar(argv[0], IDL_TYP_LONG, &result);
```

--
Nigel Wade, System Administrator, Space Plasma Physics Group,
University of Leicester, Leicester, LE1 7RH, UK
E-mail : nmw@ion.le.ac.uk
Phone : +44 (0)116 2523568, Fax : +44 (0)116 2523555