Subject: Re: Pointers to a variable...
Posted by H. Evans on Fri, 26 Aug 2011 13:22:09 GMT
View Forum Message <> Reply to Message

On Aug 26, 3:17 pm, "H. Evans" <bloggs...@googlemail.com> wrote:
> In other less friendly languages, e.g. C, the pointer points to an
> area of memory, which can coincide with a variable. This gives two
> methods to access the contents of the variable:
>
> #include <stdio.h>
> main() {
> int a=5;
> int *p;
>
> p = &a;
> printf("a=%i, *p=%i\n", a, *p);
> a= 10;
> printf("a=%i, *p=%i\n", a, *p);
>
> }
>
> outputs:
> a=5, *p=5
> a=10, *p=10
>
> So, now that IDL has pointers...can a pointer be set to point to a
> variable in the same way, i.e. to reference exactly the same memory
> space as the variable?
>
> From the examples, I am under the impression that these pointers don't
> quite work in the same way, i.e. the pointers don't point to the same
> memory space as the variables.
>
> The reason I ask is that there are some very large variables that I'd
> rather not duplicate (waste of memory), but would like to group
> serially via a pointer array.
>
> As a trivial example:
> a = FINDGEN(10000000L)
> b = DINDGEN(200000L)
> c = REPLICATE(!P, 10000L)
> p = PTRARR(3, /ALLOC)
> *p[0] = a
> *p[1] = b
> *p[2] = c
>
> for i=0,n_ELEMENTS(p)-1 DO print,N_ELEMENTS(*p[i])
>

> Is the only solution to create a,b, and c as heap variables in the
> first instance and then point p[i] to the heap variable?
>
> Ta.
> Hugh

Oh, and I am aware that this example could be done using a string array of the variable names and the scope_varfetch function. But this really provides for non-intuitive code, and a pain for future mainenance.

ta.
Hugh
