Subject: Re: HASH question
Posted by Jeremy Bailin on Mon, 07 Mar 2011 15:57:17 GMT
View Forum Message <> Reply to Message

> Well, you wouldn't be able to access the individual elements of the structure values in the hash without first pulling
> it out. E.g.
>
> IDL> z=hash()
> IDL> x={id:123,name:'blue',type:5,flux:3.14e+07}
> IDL> z[x.id]=x
> IDL> x={id:75,name:'red',type:5,flux:2.7e+07}
> IDL> z[x.id]=x
> IDL> help, z
> Z HASH <ID=8 NELEMENTS=2>
> IDL> print, z.keys()
> 123
> 75
>
> Let's say I want to change the "type" of the added star with id 75 from 5 to 4, i.e. it is in error
>
> IDL> help, z[75]
> ** Structure <95dd194>, 4 tags, length=24, data length=24, refs=4:
> ID LONG 75
> NAME STRING 'red'
> TYPE LONG 5
> FLUX FLOAT 2.70000e+07
>
> I can't just do:
>
> IDL> z[75].type = 4
> % Illegal subscript range: Z.
> % Error occurred at: $MAIN$
> % Execution halted at: $MAIN$
>
> I would have to extract it, change it, and then put it back:
>
> IDL> a = z[75]
> IDL> help, a
> ** Structure <95dd194>, 4 tags, length=24, data length=24, refs=3:
> ID LONG 75
> NAME STRING 'red'
> TYPE LONG 5
> FLUX FLOAT 2.70000e+07
> IDL> a.type = 4
> IDL> z[a.id] = a
> IDL> print, z
> 123: { 123 blue 5 3.14000e+07}

> 75: { 75 red 4 2.70000e+07}
>
> Now, while I don't think that's a particularly onerous thing to do, the OP might.
>
> Not being an OOP expert I may be blowing smoke out of my proverbial, but I think the way IDL does this is The Better Way
> - encapsulation and information hiding are the two OOP concepts that I find most frequently influence the way I write
> code (OO and regular old procedural) such that it is reusable, extendable, and easily maintained.

Really? That seems like a horrible design to me. It's fine if you're dealing with one element, but what if you want to change the type of many of them at once? Then there's no way to do it without a for loop?

-Jeremy.
