Subject: Re: HASH question
Posted by Michael Galloy on Mon, 07 Mar 2011 17:43:41 GMT
View Forum Message <> Reply to Message

On 3/7/11 8:34 AM, Paul van Delst wrote:
> Jeremy Bailin wrote:
>> On Saturday, March 5, 2011 8:41:53 AM UTC-5, Gray wrote:
>>> Hi all,
>>>
>>> I have a bunch of information which I'd like to store in an organized
>>> fashion:
>>> ~IDs of some stars
>>> ~Stellar types
>>> ~Magnitudes and fluxes in different images
>>>
>>> One way I could store the information would be as an array of
>>> structures, with each element being a single star, but I don't a
>>> priori know how many stars I have, and to find a particular star I'd
>>> have to search on the ID element. So, I could use a HASH of
>>> structures so I could index by ID, which would be ideal, but then
>>> assigning values to the individual tags of the structures is much more
>>> complicated. I could instead have a bunch of hashes, one for each type
>>> of information, but that would get pretty unwieldy.
>>>
>>> So, IDL gurus, anyone have a suggestion for how to organize this most
>>> efficiently and elegantly? Thanks!
>>>
>>> --Gray
>>
>> I haven't used hashes in idl, but I think that a hash of structures makes the most sense. What makes that too complicated?
>
> 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: $MAIN$
```

You would need parentheses to get to the correct place, but that's still not OK:

```
IDL> (z[123]).type = 6
% Attempt to store into an expression: Structure reference.
% 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.

You could do a hash of hashes:

```
[501]> star_table = hash()
[502]> star_table[123] = hash('id', 123, 'name', 'blue', 'type', 5,
'flux', 3.14e7)
[504]> print, star_table[123]
name: blue
id:      123
type:        5
flux:   3.14000e+07
[506]> print, (star_table[123])['type']
       5
[507]> (star_table[123])['type'] = 6
[508]> print, (star_table[123])['type']
       6
```

Mike
--
www.michaelgalloy.com
Research Mathematician
Tech-X Corporation