Subject: Re: HASH question
Posted by Gray on Tue, 08 Mar 2011 12:23:26 GMT
View Forum Message <> Reply to Message

On Mar 7, 4:38 pm, Chris Torrence <gorth...@gmail.com> wrote:
> On Mar 7, 2:05 pm, Paulo Penteado <pp.pente...@gmail.com> wrote:
>
>
>
>
>
>
>
>
>
>
>> On Mar 7, 6:00 pm, Chris Torrence <gorth...@gmail.com> wrote:
>
>>> Hi all,
>
>>> You could also use a nested hash of hashes. For example:
>
>>> h = HASH('Sirius', HASH('Color': 'blue', 'Size': 'big'), 'Betelgeuse',
>>> HASH(...), ...)
>
>>> In IDL 8.1 you will be able to index into array/list/hash elements
>>> within a Hash (or List) using simple array indexing. So in the above
>>> case, you would be able to do:
>>> print, h['Sirius', 'Color']
>>> And it will print out "blue".
>
>>> This also works for assignment as well.
>>> h['Sirius', 'Color'] = 'white'
>
>> Nice. Just what I was doing with an inherited class. Any predictions
>> on when 8.1 will be out? Maybe I will not have to finish the half-
>> written classes I have.
>
> Soon enough that you should not have to finish your classes... :-)
>
> -Chris

So, here's an update. I did end up going with a HASH of structures,
and just dealt with the difficulty. Note that a HASH indexed by an
array of keys returns a LIST, not an array (since the values which
have been indexed are not necessarily homogeneous). However, I was
able to make good use of the HasKey() method, so my code snippet looks
something like this:

```
collect_my_info, id, type, mag1, mag2
tmp = replicate({mystruct},n_elements(id))
old = where(myHash.haskey(id),nold)
if (nold gt 1) then tmp[old] = (myHash[id[old]]).toArray() $
  else if (nold eq 1) then tmp[old] = myHash[id[old]]
tmp.id = id & tmp.type = type & tmp.mag1 = mag1 & tmp.mag2 = mag2
myHash[id] = temporary(tmp)
```

A little unwieldy, but the HasKey() method is very useful so I don't
have to search on my keys individually.
