Subject: Good practices with hashes
Posted by Fabzi on Wed, 22 May 2013 10:39:47 GMT
View Forum Message <> Reply to Message

Dear IDLers,

I'm trying to be "up to date" and use the new neat syntax offered by IDL 8. My apologies in advance if my question is trivial.

I was thinking about using hashes of hashes to store a kind of dynamic database. For example:

```
table = hash()
table['lake_1'] = hash('id', 1, 'depth', 100, 'name', 'lake1', $
'temperature', 14.5)
table['lake_2'] = hash('id', 2, 'depth', 56, 'name', 'lake2', $
'temperature', 11.8)
table['lake_3'] = hash('id', 3, 'depth', 12, 'name', 'lake3', $
'temperature', 18.1)
table['lake_4'] = hash('id', 4, 'depth', 167, 'name', 'lake4', $
'temperature', 7.2)
```

in this example, the attributes are scalars but temperature could be a vector or more complex things. I used hashes because I can dynamically fill the table as I read the data (lake per lake).

However, the resulting table is not as easy to use as I expected. Most of the time (not always), I would want to read the attribute for all lakes. So I would do:

```
depth = !NULL
foreach h, table, lake do depth = [depth, h['depth']]
```

But this is not "as easy" as if the table would be "swappable", for example if I was able to do:

```
depth = table['depth']
```

Second problem: since hashes are not sorted, I have to keep track of the lake ids to get the data arrays in the right order.

Long talk, two questions: is this the "right way" to use hashes? Is there a better solution for this quite simple problem?
(I am quite sure I am not the first one using tables...)

Thanks a lot for your advices!

Fab