Subject: Re: Copying a hash
Posted by Matt[3] on Thu, 09 Aug 2012 18:25:51 GMT

On Tuesday, August 7, 2012 11:44:45 AM UTC-4, Matt wrote:
> On Monday, August 6, 2012 7:41:12 PM UTC-4, David Fanning wrote:
>
>> Paul van Delst writes:
>
>>
>
>>
>
>>
>
>>> Bummer. To be honest, I'm not sure what the correct behaviour should be. Recursively copy all the components? I guess if
>
>>
>
>>> we think of the numbers and strings as objects also, then the answer should probably be yes.... ? Why duplicate one type
>
>>
>
>>> of object (int, float, or string) but not another (hash or list)? Still... it just doesn't seem right.
>
>>
>
>>
>
>>
>
>>
>
>> I think this takes us back to the need for a "deep copy" in objects.
>
>>
>
>>
>
>>
>
>>   http://www.idlcoyote.com/tips/copy_objects.html
>
>>
>
>>
>
>>

>
>> But, we have only been requesting it for 9 years, I see by
>
>>
>
>> the date on the article. I think the standard is 12 years
>
>>
>
>> before they either fix the problem or consign the requester
>
>>
>
>> to the loony bin. :-)
>
>>
>
>>
>
>>
>
>> Cheers,
>
>>
>
>>
>
>>
>
>> David
>
>>
>
>>
>
>>
>
>> --
>
>>
>
>> David Fanning, Ph.D.
>
>>
>
>> Fanning Software Consulting, Inc.
>
>>

>
>> Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
>
>>
>
>> Sepore ma de ni thui. ("Perhaps thou speakest truth.")
>
>
>
> Hi Guys,
>
>
>
> Thanks for the help, and sorry for somehow missing that rather clear bit of documentation.
>
>
>
> Yes, I'm not sure what the behavior should be either when there is a hash within a hash. The
default behavior seems likely to cause trouble! Anyway, to copy down through one or two hash
levels, the following lines seem to work:
>
>
>
>   copy=hash()
>
>   foreach variable, original, key do copy[key]=original[key, *]
>
>
>
> I'm sure there's a smart way of doing this recursively for an indefinite number of levels, but this
works for me, for now.
>
>
>
> Cheers,
>
>
>
> Matt

Probably no one cares about this but me, but the code I posted above doesn't copy arrays stored
in a hash correctly (it turns multi-dimensional arrays into 1-D arrays). This works though:

```
  copy=hash()
  foreach variable, original, key do begin
    if typeName(original[key]) eq 'HASH' then begin
      copy[key]=original[key, *]
    endif else begin
```

```
    copy[key]=original[key]
  endelse
endforeach
```