
Subject: recursive search in a dictionary

Posted by [Helder Marchetto](#) on Mon, 16 Feb 2015 21:45:25 GMT

[View Forum Message](#) <> [Reply to Message](#)

Hi,

I heavily rely on dictionaries in my work and I've come across the need to find a the value for a key of a dictionary of a dictionary of a dictionary... n-times.

I thought that this should be best done recursively and I've come up with a solution that works. I would however appreciate comments on how to improve this because it seems a bit clumsy (if not embarrassing).

Usage:

```
value = getDictVal(inputDict, key, found=found)
if found then print, key+' = '+strtrim(value,2) $
    else print, key+' not found'
```

Below are the two functions that do the work and a pro that to test it.

Thanks for any suggestions.

Regards,
Helder

```
function dictRecursiveCrawler, varIn, varName, str
Compile_Opt idl2
;get available keys
keys = varIn->keys()
;scan through the keys
foreach key, keys do begin
    ;if a match is found, add the key to the list in the str structure and return it
    if key eq varName then begin
        str.found = 1b
        str.sub->add,key
        return, str
    endif
    ;if a match is *not* found, check if it is a dictionary. If so, call this same function
    (sub-scan/recursively)
    if isa(varIn[key],'dictionary') then begin
        tmp = {found:0b, sub:list([str.sub->toArray(),key],/extract)}
        res = dictRecursiveCrawler(varIn[key], varName, tmp)
        if res.found then return, res
    endif
endforeach
;if the search did not give any results until now, then the key wasn't present at any level
return, {found:0b, sub:list()}
end
```

```

function getDictVal, varIn, searchFor, found=found
Compile_Opt idl2
;initialize search structure
str = {found:0b, sub:list()}
;search for structure
res = dictRecursiveCrawler(varIn, searchFor, str)
;check results and return output
found = res.found
if found then begin
    if n_elements(res.sub) gt 1 then begin
        last = varIn
        for i=0,n_elements(res.sub)-1 do last = last[res.sub[i]]
        return, last
    endif else return, varIn[res.sub[0]]
endif
return, 0b
end

pro testCrawler
varIn = dictionary('first_layer_a', 1,$
    'first_layer_b', 2,$
    'first_layer_c', dictionary('second_layer_a', 3,$
        'second_layer_b', 4,$
        'second_layer_c', dictionary('third_layer_a',5,$
            'third_layer_b',6)))

result_1 = getDictVal(varIn, 'first_layer_b', found=found)
if found then print, 'first_layer_b = ',strtrim(result_1,2) else print, 'first_layer_b not found'
result_2 = getDictVal(varIn, 'second_layer_b', found=found)
if found then print, 'second_layer_b = ',strtrim(result_2,2) else print, 'second_layer_b not found'
result_3 = getDictVal(varIn, 'third_layer_b', found=found)
if found then print, 'third_layer_b = ',strtrim(result_3,2) else print, 'third_layer_b not found'
result_3 = getDictVal(varIn, 'qwertyuiop', found=found)
if found then print, 'third_layer_b = ',strtrim(result_3,2) else print, 'third_layer_b not found'
end

```
