
Subject: Re: structure (reform_struct)
Posted by [R.Bauer](#) on Tue, 27 Apr 1999 07:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

Thanks for some ideas,

I was thinking more about a function like reform_struct.

The most reason why I am needing a function like this is that I have learned that's widget_table is very fast if I use a vector of structures instead of a string array.

To start the discussion again about an editor widget for structures this routine is usefull.

```
data={a:findgen(10),b:sin(findgen(10))}  
result=reform_struct(data,10,/struct_array)
```

```
xvaredit,result
```

R.Bauer

```
; Copyright (c) 1999, Forschungszentrum Juelich GmbH ICG-1  
; All rights reserved.  
; Unauthorized reproduction prohibited.  
; This software may be used, copied, or redistributed as long as it is not  
; sold and this copyright notice is reproduced on each copy made. This  
; routine is provided as is without any express or implied warranties  
; whatsoever.  
;  
;  
;+  
; NAME:  
;   reform_struct  
;  
; PURPOSE:  
;   This function is used to reform a structure  
;  
; CATEGORY:  
;   PROGTOOLS/STRUCTURE  
;  
; CALLING SEQUENCE:  
;   result=reform_struct(structure,dim1,[dim2],[dim3],[dim4],[di
```

```

m5],[dim6],[dim7],[dim8],[/struct_array],[/tag_array]
;
; INPUTS:
;   structure: a structures
;   dim1: the new dimension of the input structure (1D to 8D)
;
; OPTIONAL INPUTS:
;   dim2 to dim8: additional dimensions if not declared in dim1
;
; KEYWORD PARAMETERS:
;   struct_array: if is set output structure is an array
;   tag_array:   if is set output structure is 1D and tags are arrays
;   You have to set one of these keywords
;
; EXAMPLE:
;   data={a:findgen(10),b:sin(findgen(10))}
;   result=reform_struct(data,2,5,/struct_array)
;
; MODIFICATION HISTORY:
;   Written by:   R.Bauer (ICG-1), 1999-Apr-23
;-

```

```

FUNCTION reform_struct,struct,name=name,dim1,dim2,dim3,dim4,dim5,dim6
,dim7,dim8,struct_array=struct_Array,tag_array=tag_array

```

```

names=TAG_NAMES(struct)
n=N_ELEMENTS(names)

```

```

IF KEYWORD_SET(struct_Array) THEN BEGIN
  FOR i=0,n-1 DO build_structure,in_struct,names[i],struct.(i)[0]

```

```

CASE N_PARAMS() OF
  2: BEGIN

```

```

  CASE N_ELEMENTS(dim1) OF

```

```

    1: in_struct=REPLICATE(in_struct,dim1[0])

```

```

    2 : in_struct=REPLICATE(in_struct,dim1[0],dim1[1])

```

```

    3 : in_struct=REPLICATE(in_struct,dim1[0],dim1[1],dim1[2])

```

```

    4 : in_struct=REPLICATE(in_struct,dim1[0],dim1[1],dim1[2],dim1[3 ])

```

```

    5 : in_struct=REPLICATE(in_struct,dim1[0],dim1[1],dim1[2],dim1[3 ],dim1[4])

```

```

    6 : in_struct=REPLICATE(in_struct,dim1[0],dim1[1],dim1[2],dim1[3 ],dim1[4],dim1[5])

```

```

    7 : in_struct=REPLICATE(in_struct,dim1[0],dim1[1],dim1[2],dim1[3

```

```

],dim1[4],dim1[5],dim1[6])

```

```

    8 : in_struct=REPLICATE(in_struct,dim1[0],dim1[1],dim1[2],dim1[3

```

```

],dim1[4],dim1[5],dim1[6],dim1[7])

```

```

  ELSE:

```

```

  ENDCASE

```

```

END
3: in_struct=REPLICATE(in_struct,dim1,dim2)
4: in_struct=REPLICATE(in_struct,dim1,dim2,dim3)
5: in_struct=REPLICATE(in_struct,dim1,dim2,dim3,dim4)
6: in_struct=REPLICATE(in_struct,dim1,dim2,dim3,dim4,dim5)
7: in_struct=REPLICATE(in_struct,dim1,dim2,dim3,dim4,dim5,dim6)
8: in_struct=REPLICATE(in_struct,dim1,dim2,dim3,dim4,dim5,dim6, dim7)
9: in_struct=REPLICATE(in_struct,dim1,dim2,dim3,dim4,dim5,dim6, dim7,dim8)
ELSE:
ENDCASE

n=N_ELEMENTS(TAG_NAMES(struct))
FOR i=0,n-1 DO in_struct[*].(i)=struct.(i)

RETURN,in_struct
ENDIF

IF KEYWORD_SET(tag_array) THEN BEGIN
FOR i=0,n-1 DO BEGIN
CASE N_PARAMS() OF
2: build_Structure,in_struct,names[i],REFORM(struct.(i),dim1)
3: build_Structure,in_struct,names[i],REFORM(struct.(i),dim1,di m2)
4: build_Structure,in_struct,names[i],REFORM(struct.(i),dim1,di m2,dim3)
5: build_Structure,in_struct,names[i],REFORM(struct.(i),dim1,di m2,dim3,dim4)
6: build_Structure,in_struct,names[i],REFORM(struct.(i),dim1,di m2,dim3,dim4,dim5)
7: build_Structure,in_struct,names[i],REFORM(struct.(i),dim1,di m2,dim3,dim4,dim5,dim6)
8: build_Structure,in_struct,names[i],REFORM(struct.(i),dim1,di
m2,dim3,dim4,dim5,dim6,dim7)
9: build_Structure,in_struct,names[i],REFORM(struct.(i),dim1,di
m2,dim3,dim4,dim5,dim6,dim7,dim8)
ELSE:
ENDCASE

ENDFOR

RETURN,in_Struct
ENDIF

IF N_ELEMENTS(in_Struct) EQ 0 THEN BEGIN
MESSAGE,'Forgotten switch: /struct_array OR /tag_array',/info
RETURN,struct
ENDIF
END

;
; Copyright (c) 1997, Forschungszentrum Juelich GmbH ICG-1
; All rights reserved.

```

```

; Unauthorized reproduction prohibited.
; This software may be used, copied, or redistributed as long as it is not
; sold and this copyright notice is reproduced on each copy made. This
; routine is provided as is without any express or implied warranties
; whatsoever.
;
;+
; NAME:
; is_tag
;
; PURPOSE:
; Test if tagname is defined
;
; CATEGORY:
; PROG_TOOLS/STRUCTURES
;
; CALLING SEQUENCE:
; Result=is_tag(structure,tagname)
;
; INPUTS:
; structure: the structure
; tagname: the tagname as string which should be searched in structure
;
; OUTPUTS:
; Result will be 1 or 0
;
; KEYWORD PARAMETERS:
; part: if is set strpos is used to find a part of a string
;
; EXAMPLE:
; print,is_tag(inhalt,'param')
; 1
;
; MODIFICATION HISTORY:
; Written by: R.Bauer (ICG-1) , Sep. 2 1996
; F.Rohrer (ICG-3), Mai 15 1997 downgrade to idl 3.6.1
; R.Bauer 1998-Jul-05 previously named as find_tag now renamed for better consistency
; R.Bauer 1998-Jul-05 upgraded to idl 5.1
; R.Bauer 1998-Nov-19 goto removed additional test of tag_name is string added
; R.Bauer 1999-Mar-26 keyword part is added
;
;-

```

FUNCTION is_tag,struct,tag_name,part=part

```

tagname=STRUPCASE(tag_name)

```

```

tags=TAG_NAMES(struct)
if keyword_set(part) then begin
a=WHERE(strpos(tags, tagname) eq 0,count)
if count gt 1 then message,'WARNING: count: '+strtrim(string(count),2),/info
endif else a=WHERE(tags EQ tagname,count)

RETURN, count
END

```

```

;
; Copyright (c) 1998, Forschungszentrum Juelich GmbH ICG-1
; All rights reserved.
; Unauthorized reproduction prohibited.
; This software may be used, copied, or redistributed as long as it is not
; sold and this copyright notice is reproduced on each copy made. This
; routine is provided as is without any express or implied warranties
; whatsoever.
;+
; NAME:
; build_structure
;
; PURPOSE:
; This routine accumulates values into a structure or concatenates two structures
;
; CATEGORY:
; PROG_TOOLS
;
; CALLING SEQUENCE:
; build_structure,structure,declaration,value,[name=name],[ra
ndom_structure_name],[/counted_structure_name]
;
; INPUTS:
; structure : if it's present the structure which will be accumulated to
; declaration: the tag which will be added to the structure
; value : the value which will be added to the structure
;
;
; OUTPUTS:
; structure: A structure holding value
;
; EXAMPLE:
; build_structure,def,'test',5
; help,test,/str
;** Structure <117e2f8>, 1 tags, length=2, refs=1:
; TEST INT 5

```

```

; build_structure,def,'AN',5
; help,def,/str
;** Structure <117ec88>, 2 tags, length=4, refs=1:
; TEST      INT      5
; AN        INT      5
;
;
;
; MODIFICATION HISTORY:
; Written by: R.Bauer (ICG-1), 1998-Jun-27
; 1999-02-23 Problem if tag is already defined in structure by is_tag resolved
; 1999-03-16 random_structure_name added
; 1999-03-20 counted_structure_name added
;-

```

PRO build_structure,structure,declaration,value

```

errvar=0
CATCH,errvar

IF errvar NE 0 THEN BEGIN
    RETURN
ENDIF

IF N_PARAMS() EQ 3 THEN BEGIN
    IF N_ELEMENTS(structure) EQ 0 THEN structure=CREATE_STRUCT(declaration,value)
ELSE BEGIN
    IF NOT is_tag(structure,declaration) THEN
structure=CREATE_STRUCT(structure[0],declaration,value) ELSE MESSAGE,declaration+'
already in structure',/cont
    ENDELSE
    ENDIF ELSE BEGIN
    IF size(declaration,/type) eq 8 THEN structure=CREATE_STRUCT(structure,declaration)

    ENDELSE
END

```

File Attachments

- 1) [reform_struct.pro](#), downloaded 98 times
 - 2) [is_tag.pro](#), downloaded 95 times
 - 3) [build_structure.pro](#), downloaded 92 times
-