Rocket software

Rocket Uniface Library 10.4

Example: Copying Structs

This function demonstrates what happens when copying references to Structs (copy by reference), and copying the Struct itself (copy by value).

- To copy by reference, the left side of the assignment must be a `struct` variable.
- To copy by value, the left side of the assignment must be a Struct member.

```
function COPY_STRUCT
variables
    struct vStruct1, vStruct2
endvariables

  call printHeader("COPY_STRUCT") ; display entry header in the message frame

; Copy by reference
  ; Build a Struct with two members
  vStruct1->a = "AAA"
  vStruct1->b = "BBB"

  ; Copy vStruct1 to vStruct2 (by reference)
  vStruct2 = vStruct1                [1]

  ; Update the Struct using vStruct2:
  vStruct2->b = "BBB-updated"        [2]
  putmess "Although vStruct2 changed the Struct, vStruct1->b returns the change:"
  putmess "%%(vStruct1->b)%%%"

; Copy by value
  ; Rebuild the Struct from scratch:
  vStruct1= $newstruct
  vStruct2= $newstruct
  vStruct1->a = "AAA"
  vStruct1->b = "BBB"

  ; Copy vStruct1 to vStruct2 (by value): [3]
  ;  the left side of the assignment is a Struct member
  vStruct2->subnode = vStruct1

  ; Update the copied struct
  vStruct2->subnode->b = "BBB-updated"  [4]

  ; Compare the resulting structs:      [5]
  putmess "Struct referred to by vStruct1:"
  putmess vStruct1->$dbgstring
  putmess "Struct referred to by vStruct2->subnode:"
  putmess vStruct2->subnode->$dbgstring

end ; function COPY_STRUCT
```

1. The left side of the assignment is a `struct` variable (`vStruct1`), so the reference to the Struct is copied to `vStruct2`. As a consequence, `vStruct1` and `vStruct2` refer to the same Struct.
2. When one of the variables is used to update the Struct, the change is reflected using the other variable.

```
Although vStruct2 changed the Struct, vStruct1->b returns the change:
BBB-updated
```

3. The left side of the assignment is a Struct node, so the Struct referred to by `vStruct->subnode` is a copy of the Struct referred to by `vStruct1`.
4. When the value of a member referred to by the `vStruct2` variable is changed, only the copy is changed, not the original Struct.
5. This is clearly visible in the output in the message frame:

```
Struct referred to by vStruct1:
[]
 [a] = "AAA"
 [b] = "BBB"
Struct referred to by vStruct2->subnode:
[subnode]
 [a] = "AAA"
 [b] = "BBB-updated"
```