Rocket Uniface Library 10.4

Struct Dereference Operator (->)

Access the members of a Struct by name.

StructVariable `->` *MemberName* | *StructFunction*

Arguments

- *StructVariable*—variable or parameter of type `struct` which refers to one or more Structs
- *MemberName*—Struct member identifier; can be a literal name or a string; for example `mem` or `"mem"`. An empty string `""` returns Structs with no name.
- *Function*—Struct function. For more information, see ProcScript: Struct Functions.

Return Values

Returns an ordered collection of references to members of the Struct to which *StructVariable* refers.

If *StructVariable* refers to a collection of Structs rather than to a single Struct, the expression is evaluated for each Struct in the collection, and the resulting collections are combined into a single ordered collection of Structs.

If there are no members that satisfy the criteria, the result is NULL and an error is set in `$procerror`.

Table: Values of $procerror Commonly Returned Following ->

Error number	Error Constant	Meaning
`-84`	`ACTERR_NO_OBJECT`	Struct operator applied to non-Struct
`-1153`	`USTRUCTERR_INDEX_NOT_ALLOWED`	Struct index is not allowed
`-1155`	`USTRUCTERR_MEMBER_NOT_FOUND`	The Struct does not contain a member with the specified name.
`-1157`	`USTRUCTERR_ILLEGAL_MEMBER_TYPE`	*StructVariable* is not of type `struct` or `any`

Use

Allowed in all components.

Description

The arrow `->` is known as the dereference operator because the Struct being referred to changes (is de-referenced) from the one on the left to the one on the right. The dereference operator can be used multiple times in a statement to travers the nested levels of a Struct to get to the Struct member you need. For example, if you have a Struct referring to book that includes a Preface, you need to start from the top of the Struct nad use the dereference operator to get the Preface, and use it again to get the Acknowledgements section:

```
vStruct = Book->Preface->Acknowledgements
```

You can use the dereference operator to address Structs by name. By specifying an empty string (`""`), you can access nameless Structs. By using the * asterisk wildcard after the operator, you can get a collection of references to all members of a Struct.

> **Important:** The * wildcard can only be used after the dereference operator. It cannot be used in constructs such as `vStruct->c*` to get all Structs whose names begin with `c.->*` is actually a separate operator (the Struct collection operator).

Member Names as Strings

MemberName can be specified as a string, which means that it is possible to:

- Use string substitution to access a field name. For example, the following code returns the members with the name as specified in field F1.

  ```
  vStruct->"%%F1%%%"
  ```

- Specify member names that include any character, including spaces, `#`, and `$`.

  ```
  vStruct->"first name"
  ```

- Address Struct members that have an empty string as name:

  ```
  vStruct->""
  ```

If a Struct has several members with the same name, you can access one of these members specifically, by using the dereference operator in combination with the struct index operator {N}. For example:

```
vBookStruct->chapter{2}
```

Related concepts

[Substitution in String Values](#)