Rocket software

Rocket Uniface Library 10.4

Variables

Variables are placeholders for data whose value can change during the runtime duration of the ProcScript module, component, or application for which they have scope. The scope is determined by where they are declared.

- Local variables are available only in the ProcScript module in which they are defined, which may be an Operation, Trigger, Function, or Entry.
- Component variables are available to all ProcScript in the component. They are defined in the Declarations container of a component. For component variables, it is also possible to specify a display format. For more information, see Component Variables.
- Global variables are accessible to all ProcScript in an application's components, and in global ProcScripts that use the same library or the system library. They are defined and deployed in a runtime library. For more information, see Working with Global Variables.
- General variables—variables $1 through $99 do not need to be declared and are available throughout the application. It is best to agree on a consistent use of general variables, to prevent them being accidentally overwritten.

CAUTION: In a large complex application, the risk of several components overwriting the value of global and general variables increases. When using these variables it is important to clear them when they are no longer required. It is usually preferable to use component and local variables, rather than global variables.

Using Variables

Local and component variables are declared in a variables block delimited by the `variables` and `endvariables` ProcScript statements. For example:

```
variables
 string vName
 date vBirthDate
 numeric vAge
endvariables
```

To refer to variables in ProcScript:

- For local variables, use the name of the variable, for example: `vName`.
- For component variables, use the name of the variable preceded and followed with a $-character, for example: `$vName$`.
- For global variables, use the name of the variable preceded with 2 $-characters, for example: `$$SalesTax`.
- For general variables, use a number from 1 to 99 preceded with a $-character, for example: `$1`.

Tip: References to variables are not case sensitive, but it is good practice to refer to variables in lower or mixed case if you use uppercase for references to fields (or visa versa).

General Variables ($1 through $99)

A general variable does not have a predefined type; values of any type can be assigned to them. When a value of a

© 2025 Rocket Software, Inc. All Rights Reserved | Rocket Uniface Library 10.4

specific type is assigned to a general variable, the general variable takes on this new type. If the assigned value came from a variable or field, the defined display format (DIS format) for the field or variable is inherited by the general variable.

The following example shows a ProcScript function with a declaration of and references to local variables:

```
function myProcFunction
; Parameter declaration first
params
 numeric myFirstParameter : IN
 string mySecondParameter : IN
 string myThirdParameter : OUT
endparams
; Variables declaration second
variables
 numeric myNumber1, myNumber2
 string myString1, myString2, myString3
endvariables
myNumber1 = $length(mySecondParameter)
myNumber2 = FIELD1 + myNumber1
; More ProcScript...
end
```

Related concepts

- variables...endvariables

Related tasks

- Working with Global Variables

© 2025 Rocket Software, Inc. All Rights Reserved | Rocket Uniface Library 10.4