CALL - Public procedure call

CALL action - call of Public procedures

Declaration

```
CALL [_unitIdent] ProcName [(paramIdent1  [,paramIdent2]...)]
```

Parameters

ProcName	in	Name of the procedure (it must conform to the rules for object name).
_unitIdent	in	Name of the local variable that represents UNIT.
paramIdent1, paramIdent2, ..., paramIdentN	in	Identifier of value for the first (second, third, ..., N-th) parameter. The number of parameters must correspond with the number of parameters of the called procedure.

Description

CALL action is used for calling the *ProcName* procedure. In front of the procedure name, there must be a recipient (UNIT, from which the particular procedure is going to be executed). The parameters, in the square brackets and separated by a comma, follow after the name of the procedure. The number of parameters must be equal to the number of parameters of the called procedure. If some parameter is an input/output in the declaration of procedure, the equivalent parameter must not be a constant when calling the procedure.

Example

```
;*******************************************************
; DESCRIPT: Unit Caller
;
;
; AUTHOR: Programmer
; LAST CHANGE:
;*******************************************************

UNIT (E.Unit1) _unit1
UNIT (E.Unit1) _unit12
UNIT (E.Unit2) _unit2

RPC PROCEDURE CheckValue(BOOL _bOk)
INT _iValue
INT _iValue2

CALL [_unit1] GetValue(_iValue)
CALL [_unit12] GetValue(_iValue2)

_bOk := _iValue # _iValue2

END CheckValue

BEGIN

 CALL [_unit1] Make
 CALL [_unit12] Make
 CALL [_unit12] Make

END
```

Related pages:

Script actions
ESL Unit Event