# DO_LOOP, EXIT_LOOP, END_LOOP

## DO_LOOP, EXIT_LOOP, END_LOOP action

**Function**

The actions allows to implement a loop.

**Declaration**

```
[FOR _ctrlVar [RANGE struct | = lBoundExpr TO uBoundExpr] ] DO_LOOP

 ...; actions 1

 [EXIT_LOOP [expression]]

 ...; actions 2

 [EXIT_LOOP [expression]]

 ...; akctions 3

 END_LOOP
```

**Parameters**

| | | |
|---|---|---|
| expression | in | Expression of Boolean type. |

**Description**

The loop has two variants:

1. Loop with using a control variable. At the beginning of the loop, the control variable is set to the value 1 in case of the variant with using the **RANGE** keyword, or it is set to the value given by the expression **lBoundExpr**. Value of the expression must be valid. High limit for the value of the control variable is evaluated once, too. In the first case (**RANGE**), the high limit is the size of given structure (*struct\DIM*). In the second case, the high limit is acquired by evaluation of the expression **uBoundExpr**. Value of the expression must be valid. During individual iterations, the control variable of the loop will be increased in successive steps up to the high limit. The control variable must be defined as **INT** type. It can be changed in the loop body. Possible invalidation of its value will causes an error when executing the **END_LOOP** action. The loop can be aborted by the **EXIT_LOOP** action.
   After terminating the cycle, the value of the control variable is the value of the high limit increased by 1.

```
INT _i
 INT _uBound
 _uBound := 10

 FOR _i=2 TO _uBound DO_LOOP
  _uBound := _uBound + 1 ; value change has no effect on the number
of iterations

 END_LOOP
 ; value of the variable _i is 11
```

```
INT _i
 RECORD (SD.ArchVal) _struct

 REDIM _struct[10]

FOR _i RANGE _struct DO_LOOP
   REDIM _struct[2] ; value change has no effect on the number of
iterations
 END_LOOP
 ; value of the variable _i is 11
```

2. Loop with using no control variable. Actions enclosed between the actions **DO_LOOP** a **END_LOOP** will be cyclically executed. The action **EXIT_LOOP** may terminate a cycle. If the action is with a parameter, then the loop will be terminated when the expression will get the value

@TRUE.

```
INT _i
_i := 1
DO_LOOP
  EXIT_LOOP _i = 10
  _i := _i + 1
END_LOOP
```

Equivalent declaration with no conditional termination of the loop:

```
INT _i
_i := 1DO_LOOP

  IF _i = 10 THEN
    EXIT_LOOP
  ENDIF
  _i := _i + 1 END_LOOP
```

**Note**

Invalid notation of a loop occurs, if its beginning and end is "crossed" with the action IF THEN.

For example:

```
INT _i
 _i := 1

  IF Sec = 1 THEN
   DO_LOOP
 ENDIF
   EXIT_LOOP _i = 10
   _i := _i + 1
 END_LOOP
```

or

```
INT _i

 _i := 1
 DO_LOOP
   IF _i = 10 THEN
     EXIT_LOOP
   ENDIF
   _i := _i + 1
 IF Sec = 1 THEN
     _i := _i + 1
   END_LOOP
 ELSE
   END_LOOP
 ENDIF
```

**Related pages:**

Script actions