PerfCounterR

%PerfCounter, %PerfCounterR functions

Function

The function returns the value of the given "Performance Counter".

Declaration

```
INT %PerfCounter(
   TEXT in text
 )
```

```
REAL %PerfCounterR(
   TEXT in text
 )
```

Parameters

text	Performance Counter definition.

Description

Performance Counter definition syntax is as follows:
\\ComputerName\Object(instance)\Counter.

If you want to monitor the counter on local computer, the first part giving the computer name may omitted. If an object has only one instance, the part "instance" is omitted.
If the function %PerfCounter (%PerfCounterR) is called frequently (more times than once per second), it returns wrong values (0).
If there is the need to call function each second for more system information, we recommend you to execute each calling on the different line in ESL script.

Example

```
%PerfCounter("\D2000 Server\MemUsed")    ; returns the memory size used by
process D2000 Server
```

Note 1

In some situation (for specific counters) %PerfCounter returns always zero at first calling, e.g. for load counter of processor: " \Processor(_Total)\% User Time". Then you must call this function twice in a loop with delay (the size must be set empirically) between the callings.
For other counters (e.g. "\Memory\Available MBytes", "\LogicalDisk(C:)\% Free Space" or "\Process (_Total)\Handle count"), this function works without any problems and returns the correct value immediately at first call.
Apparently, there is some problem with the counters that require a certain time between the registration of counter and loading its value.

This is an example of the code which calls %PerfCounter twice with delay 0.1 second if the value 0 has been loaded at first calling:

```
INT _i
REAL _result

FOR _i=1 TO 2 DO_LOOP
 _result := %PerfCounter("\Processor(_Total)\% User Time")
 IF _result\VLD THEN
  IF _result=0 & _i=1 THEN       ; workaround - at first calling (e.g. it
returns 0 on \Processor (_Total)\% User Time)
   DELAY 0.1 [s]
  ELSE
   EXIT_LOOP
  ENDIF
 ELSE
  EXIT_LOOP
 ENDIF
END_LOOP
```

This code does not work because the same row is not called twice, the same instance of PerfCounter function:

```
INT _i
REAL _result

_result := %PerfCounter(_counter)
IF _result\VLD THEN
 IF _result=0 THEN   ; NONFUNCTIONAL workaround - at first calling (e.g.
it returns 0 on \Processor(_Total)\% User Time)
  DELAY 0.05 [s]
  _result := %PerfCounter(_counter)
 ENDIF
ENDIF
```

Note 2

We recommend to use %PerfCounterR when it is assumed that the value will be higher than 2 147 483 647.

Related pages:

Implemented functions
Function arguments - types