GETARCHVAL

GETARCHVAL action

Function

Reading a value of specified historical value in given time.

Declaration

```
 GETARCHVAL valueIdent_Real, archIdent, timeExpression_TmA [STATUS
[isDataIdent_Bool], [archFlagsIdent_Int], [archivInstance_Int]]
```

Parameters

valueIdent_Real	out	Reference to one value of historical value, reference to object or item identifier of Structured variable type object (*Note:* values of object or item must be archived). **Warning:** If the parameter is the reference to an object archived several times, there is not specified which one of the historical objects is to be used.
archIdent	in	Reference to one value of historical value, reference to object or item identifier of Structured variable type object.
timeExpression_TmA	in	Expression of *Absolute* time type.
isDataIdent_Bool	out	Identifier of *Boolean* type - attribute of successful reading from the archive, optional parameter.
archFlagsIdent_Int	out	Identifier of *Int* type: archive flags, optional parameter.
archivInstance_Int	in	Optional identifier of *Int* type - identification of archive instance. If the parameter is not defined, the value 0 will replace it.

Description

The action reads values from the archive for the given time. It writes the result (value) to the variable *ValueIdent_Real*. The read value matches with a time interval reading. This time interval has both begin time and end time the same and a time step is 0.

If the parameter *archIdent* is the reference to an object of Historical value type, the action performance is described above. If the parameter is the reference to an object (not of Historical value type) or a structured variable item that is not of ***Object*** type, the system is attempting to find an object of Historical value type that archives values of the object (item).
If the parameter *archIdent* is the reference to a structured variable item that is of ***Object*** type, the item "points" to an object in the system. If the object is of Historical value type, the action will read data from it. If it is not, the system is attempting to find an object of Historical value type that archives values of the object.

The reading result (value) is to be assigned to the variable `valueIdent_Real`. If this identifier is stated, the action will assign to it the value:

- **True** - data are loaded,
- **False** - no data.

If the identifier *isDataIdent_Bool* is not used and there are no data, the action doesn't modify a value of the identifier *valueIdent_Real*.

I the identifier `archFlagsIdent_Int` is stated, archive flags for a value, that was read, are saved into the identifier (archive flags are defined as the sum of the following constants):

- **1** - (ArcStart) - value stored into the database at the moment of D2000 Archiv start
- **2** - (ArcStop) - value stored into the database at the moment of D2000 Archiv stop
- **4** - (ArcBlock) - value stored into the database at the beginning of blocking the archiving (by the stop condition of archiving configured in D2000 CNF)
- **8** - (ArcUnBlock) - value stored into the database at the end of blocking the archiving (by the start condition of archiving configured in D2000 CNF)
- **16** - (ArcDeleted) - value from the archive database that was deleted by an user. The action ignores the deleted value (as if it were not in the archive)
- **32** - (ArcUsermodify) - value in the archive that was modified by an user
- **64** - (ArcOldVal) - old value that was read from a communication station
- **128** - (ArcProcessModify) - value that was modified by a process (not by an user: ESL Script, D2000 VBApi, ...)
- **256** (ArcLoadData) - obsolete: value was obtained from OS/2 database SQL Gupta via On-line archive database import
- **512** (ArcMonoTime) - value is stored with monotonous time
- **1024** (ArcK) - value of periodic archive is generated during reading as a copy of previous value

The logical AND (&), that may be also applied into integer operands, is used to test archive flags.
For example: If the condition **`aflags & 4 = 4`** holds, then the flag of a value that was read from the archive database by means of the action **GETARCHVAL** is **ArcBlock** (blocking of the archiving).

Value of parameter *archivInstance_Int* defines the instance of archive which executes the request. If the parameter is not defined (or the value is 0), the active instance of archive will execute the request.

Note

- If the particular archive process is not running, the action generates the error _ERR_ARCHIV_NOT_RUNNING.
- If an an object of Historical value is archived:
    - periodically, reading a value out of the period is unsuccessful
    - according to a filter, value for the required time is to be derived from the last stored value in the archive.

Example

Reading of archived value:

```
REAL _value
TIME _bt
BOOL _bIsArchData

; calculation of current minute begin
_bt := SysTime
_bt := _bt - %ModTime(_bt, 60)
 ; reading of value and active instance of archive
 GETARCHVAL _value, H.Sec, _bt STATUS _bIsArchData,,0
; has the value been read?
IF _bIsArchData THEN
; processing of value which has been read
 ENDIF
```

Related pages:

Script actions