

Structures (D2000 OBJApi)

D2000 OBJApi - Interface structures

From the version D2000 V9.02.034, there is available both 32 and 64-bit version with modified header file for use in the projects with the interpretation of text strings in 8-bit form (ANSI) or 16-bit form (wide character UNICODE), according to the parameter of the project "Character Set" in MS Visual Studio.

UniVal structure

UniVal contains all information about the value and status of a given object of the D2000 system.

Declaration in C language:

```
typedef struct _UniVal
{
    GenValueType gvaltyp;
    unsigned int status;
    tLimitStatus limitStatus;
    tProcAlarmType procAlarmStatus;
    ValueType type;
    double valtime;
    double procAlarmTime;
    unsigned int Flags;
    VOBJ Indirect;
    union
    {
        tBVal Boval;
        int Intval;
        double Realval;
        StVal Stval;
        struct
        {
            AlVal Alval;
            double AlTimes[4];
        } a;
        PrVal Prval;
        double TmAval;
        double TmRval;
        tQValue Qval;
        char *TxtVal;
        ArrayRecPtr ArrayValPtr;
        struct
        {
            ArrayRecPtr recordValPtr;
            HOBJ StructTypId;
            int ColsNr;
        } r;
    } v;
} UniVal;
```

Description of individual structure parts:

- **gvaltyp**
It determines an object value type for the variable part of the UniVal structure.
- **status**
Value status, gets a combination of the values :

Constant definition	Description
#define VAL_Valid 1U	Object value is valid
#define VAL_ProcAlarm 2U	Object has an active process alarm
#define VAL_NoAckPAlarm 4U	Object had an active process alarm, that is not acknowledged
#define VAL_PrAISilent 8U	Process alarms of the object are not evaluated

#define VAL_Weak 16U	Object value is suspicious – “weak”
#define VAL_NoAckValue 32U	Object value is changed and is not acknowledged
#define VAL_Transient 64U	Object value is in the transient status when is writing
#define VAL_Default 128U	Value is in the default status
#define VAL_Manual 256U	Object value modified manually
#define VAL_PrAlCrit 512U	Object has an active critical process alarm

- **limitStatus**

Object value status in regard to configured value limits:

```
typedef enum {InLimit, VL_Limit, L_Limit, H_Limit, VH_Limit, LimitsProblem} tLimitStatus;
```

Value status	Status Description
InLimit	Object value is in limits
L_Limit	Object value is below the low limit
VL_Limit	Object value is below the lowest limit
H_Limit	Object value is above the high limit
VH_Limit	Object value is above the highest limit
LimitsProblem	A problem with the evaluation of the limits (unknown value of active limit, crossing the values of the active limits)

- **procAlarmStatus**

Active process alarm type.

```
typedef enum {NoAlarm, ToOn, ToOff, On, Off, Err, Oscillate, ErrCmdOn, ErrCmdOff, SwToTrans, SwToOff, SwToOn, SwToErr, SwTrans, SwOff, SwOn, SwErr, ErrZalCmdOff, HL, VHL, LL, VLL, ToHL, ToVHL, ToLL, ToVLL, ErrWriteCmd, A28, A29, A30, A31, A32} tProcAlarmType;
```

- **type**

D2000 system object type.

- **valtime**

Time of the last object value change.

- **procAlarmTime**

Time of the last object process alarm change.

- **Flags**

Bit array: Object value flags (A up to P).

- **Indirect**

If the value is a value of a [structure](#) of *Object* type, there is a value source (object *id* and *row*, *col*).

The variable part, holding its own object value:

- **Boval**
Value of an object of Boolean type (type : Bo, Di, Do, De, Li).
- **Intval**
Value of an object of *Integer* type (type : Int, Ci, Co, Ce).
- **Realval**
Value of an object of *Real* type (type : Re, Ai, Ao, Ae).
- **Stval**
Value of an object of Station type (type : St).
- **a.Alval**
Value of an object of Alarm type (type : Al).
- **a.AlTimes**
Indexed times if individual alarm [states](#):

Alarm status	Index
Alarm	0

NoKvit	1
Kvit	2
Norm	3

!!! WARNING !!!

Since the D2000 version 7.0, the item *a.AITimes* is not used and therefore has been renamed to *a.AITimes_Unused*. The items of the array are filled with the value of 0.0.

- **Prval**
Value of an object of *Process* type (type : Pr).
- **TmAval**
Value of an object of *Absolute time* types (type : TmA, TiA, ToA).
- **TmRval**
Value of an object of *Relative time* types (type : TmR, TiR, ToR).
- **Qval**
Value of an object of *Quadrat input* type (type : Qi).
- **TxtVal**
Value of an object of *Text* type (type : Txt, TxtI, TxtO).
- **ArrayValPtr**
Value of an object of *Array* type (type : Arr).
- **r.recordValPtr**
Reference to an structure value (type : Rec).
- **r.StructTypId**
Id of an object of *Structure definition* type, that defines the value type.
- **r.ColsNr**
Number of columns in the structure.

Warning for some value types:

UniVal contains only references to values for objects of **Text** and **Array** types. Values are created dynamically and it is necessary to release them calling the function [FreeData](#). Exceptions are asynchronous calling the function [NewValueProc](#) (ObjAPI releases the values after return from the callback) and the function [ListOfObjects](#) (values are released by calling the function [FreeData](#) that releases structures of *ListObjData* type).

Representation of arrays and structures

Arrays and structures are represented by a value array of *UniVal* type, before which there is low (*lowIndex*) and high (*hiIndex*) indexes. Array items are indexed from 1. Before *hiIndex*, there are the following individual values in rows in successive steps.

```
typedef struct _ArrayRec
{
    int lowIndex;
    int hiIndex;
} ArrayRec;
typedef ArrayRec * ArrayRecPtr;
```

Access to a structure item (row, column) is represented by the following function:

```
UniValPtr GetRecordItem(UniVal value, int row, int col)
{
    UniValPtr uni_Ptr;
    int valueIdx;
    valueIdx = (row-1)*value.v.r.ColsNr+col - 1;
    // first value address
    uni_Ptr = (UniValPtr)((char *)value.v.r.recordValPtr +
        sizeof(*value.v.r.recordValPtr));
    uni_Ptr = &(uni_Ptr[valueIdx]);
    return uni_Ptr;
}
```

When calling interface procedures, it is important to correctly enter the indexes !!!