Value (field)

The *Value* property specifies what is the current value of a **Field**.
The setting or return value is a `Variant` data type that evaluates to a value appropriate for the data type, as specified by the **type properties** of the field.

tags:
Properties

Applies to ...

Object	Type of field	Type of the *Value* property	Comment
Field	com.sun.star.sdbc.DataType.BIT com.sun.star.sdbc.DataType.BOOLEAN	vbBoolean	
	com.sun.star.sdbc.DataType.TINYINT	vbInteger	Ranging from −32768 to 32767, inclusive
	com.sun.star.sdbc.DataType.SMALLINT com.sun.star.sdbc.DataType.INTEGER	vbLong	Ranging from −2147483648 to 2147483647, inclusive
	com.sun.star.sdbc.DataType.BIGINT	vbBigInt	Ranging from −9223372036854775808 to 9223372036854775807, inclusive
	com.sun.star.sdbc.DataType.FLOAT	vbSingle	Absolute value ranging from 3.402823 x 10E38 to 1.401298 x 10E-45
	com.sun.star.sdbc.DataType.REAL com.sun.star.sdbc.DataType.DOUBLE	vbDouble	Absolute value ranging from 1.79769313486232 x 10E308 to 4.94065645841247 x10E-324
	com.sun.star.sdbc.DataType.NUMERIC com.sun.star.sdbc.DataType.DECIMAL	vbDouble vbLong	*vbDouble* if non-integer value, *vbLong* otherwise.

com.sun.star.sdbc.DataType.CHAR com.sun.star.sdbc.DataType.VARCHAR	vbString	
com.sun.star.sdbc.DataType.LONGVARCHAR	vbString	Error if length > 64K. To get a memo field exceeding the maximum value, use the **WriteAllText** method.
com.sun.star.sdbc.DataType.DATE com.sun.star.sdbc.DataType.TIME com.sun.star.sdbc.DataType.TIMESTAMP	vbDate	Use the Basic built-in date functions to - build the argument (f.i. *DateSerial* and *TimeSerial*) - process the returned value (f.i. *DatePart*)
com.sun.star.sdbc.DataType.BINARY com.sun.star.sdbc.DataType.VARBINARY com.sun.star.sdbc.DataType.LONGVARBINARY	vbLong	The <u>length</u> of the binary stream is returned, not the stream itself (equivalent to the **FieldSize** property). To get the stream use the **WriteAllBytes** method.
com.sun.star.sdbc.DataType.CLOB	NA	
com.sun.star.sdbc.DataType.BLOB	NA	

Syntax

field.`Value`

field.`Value` = *value*

Returned values / Arguments

`Variant` (see the table above)

Remarks

- If the database field contains `Null` the *Value* property will contain `Null` as well. To be tested with the `IsNull()` BASIC builtin function.
- To set the database field to `Null` set simply the *Value* property to `Null` as well.
- Setting the *Value* propery will be rejected if either the **AddNew** or the **Edit** method has not been applied on the concerned **Recordset** first. The modifications will be applied when the **Update** Method is applied on the same *recordset* object. If the *Update* method is never applied, or if the **CancelUpdate** method is invoked instead, the database will not be impacted.
- If the field to update is numeric (TINYINT, SMALLINT, BIGINT, FLOAT, REAL, DOUBLE, NUMERIC, DECIMAL) the *Value* property may be of any numeric type.
- Binary fields cannot be modified thru the *Value* property. Use the **ReadAllBytes** method instead.
- Trying to <u>set the *Value* property</u> in an inappropriate context (for example, the *Value* property of a *Field* object in the **Fields collection** of a **TableDef object**) will cause an error.
- When the application is <u>getting the *Value* property</u> in an erroneous context, the returned value will always be *empty* (to be tested in your code with the Basic `IsEmpty` builtin function).
- Next Basic type constants may be included in your own code, if relevant:

```
REM VarType constants
Const vbEmpty = 0
Const vbNull = 1
Const vbInteger = 2
Const vbLong = 3
Const vbSingle = 4
Const vbDouble = 5
Const vbCurrency = 6
Const vbDate = 7
Const vbString = 8
Const vbObject = 9
Const vbBoolean = 11
Const vbVariant = 12
Const vbByte = 17
Const vbUShort = 18
```

```
Const vbULong = 19
Const vbBigint = 35
Const vbDecimal = 37
```

Error messages

Property 'Value' not applicable in this context
Value '...' is invalid for property 'value'
Field length (...) exceeds maximum length. Use **WriteAllText** instead

See also

Value (controls)

Example

Add a new record to a table

```
Dim oRecordset As Object
        Set oRecordset = Application.CurrentDb().OpenRecordset("Expenses")
        With oRecordset
                .AddNew                          '          Fields initialised with the default va
                .Fields("AMOUNT").Value = 1234
                .Fields("DATE EXPENSE").Value = Now()
                .Fields("CATEGORY").Value = "Food"
                .Fields("DESCRIPTION").Value = "Lunch"
                .Fields("VAT CODE").Value = 3
                .Update
                .mClose
        End With
```

Modify the last record in a table

```
Dim oRecordset As Object
        Set oRecordset = Application.CurrentDb().OpenRecordset("Expenses")
        With oRecordset
                .MoveLast
                .Edit
                .Fields("AMOUNT").Value = 4321
                .Update
                .mClose
        End With
```

Bookmark this page » » Value (field)