

[Home](#) [x](#) [QueryDefs](#) [x](#) [Recordsets](#) [x](#) [TableDefs](#) [x](#) [TempVars](#) [x](#) [Database Functions](#) [x](#) [DAvg](#) [x](#)
[DCount](#) [x](#) [DLookup](#) [x](#) [DMin, DMax](#) [x](#) [DStDev, DStDevP](#) [x](#) [DSum](#) [x](#) [DVar, DVarP](#) [x](#) [Definitions](#) [x](#)
[Collection](#) [x](#) [Object Model](#) [x](#) [Error handling](#) [x](#) [DebugPrint](#) [x](#) [Error Handler](#) [x](#) [TraceConsole](#) [x](#)
[TraceError](#) [x](#) [TraceLevel](#) [x](#) [TraceLog](#) [x](#) [Events handling](#) [x](#) [Event](#) [x](#) [Events](#) [x](#) [Events Handler](#) [x](#)

Events Handler

tags:

[Events handling](#)

Events can automatically execute a macro when a specified software event occurs by assigning the desired macro to the event. The macros might be defined at database document, **form**, **subform**, **dialog**, **control**, menu item or toolbar button level. The triggered macro has one argument of type *Object*.

The *Events Handler* of *Access2Base* provides a mean to standardize the processing of events and to increase the reusability of macros: indeed the same macro could be used for several controls - if meaningful, of course - as there is an easy way to identify the control which triggered the macro.

Note that the described technique has nothing in common with the events processing in *MSAccess*.

Step by step

1. Assign the event to a macro: see the OpenOffice/LibreOffice Help or documentation.
2. In the code of the macro invoke the **Events** collection. Note that the macro may be a **Sub** or a **Function** depending on the necessity or not to return the value **False** to cancel the event.
3. Use the properties of the returned object to process the event. In particular the *Source* property.

```
Sub myEventMacro(poEvent As Object)
Dim oeEvent As Object, myControl As Object
    Set oeEvent = Events(poEvent)
    Set myControl = oeEvent.Source      '      Return the Control object
                                      '      that triggered the event

    REM ...
End Sub
```

Event types

The types of events supported by the *Access2Base* API are listed below. See the OpenOffice/LibreOffice documentation for more details.

The invocation of `Events()` returns a **Null** value when other event types occur.

Event type	Description
DOCUMENTEVENT	At document (database) level.
EVENTOBJECT	Generic for most <i>form</i> and <i>control</i> events.
ACTIONEVENT	Triggered by a button, ...
FOCUSEVENT	Triggered by a focus change.
INPUTEVENT	Use of special keys.
ITEMEVENT	Action in menu or listbox.
KEYEVENT	Use of normal keys.
MOUSEEVENT	Mouse move or click.
ROWCHANGEEVENT	Insert, update or delete action.
TEXTEVENT	Edition of <i>control</i> content.

See also

Events
Event
getObject

Example

Assign next macro to the *TextModified* event of a combo box (other controls could require that the *Changed* event should be used for the same purpose).

When the user changes his/her selection in the combo box the content of the form is requiered and refreshed on the screen.

```
Sub SelectChanged(poEvent As Object)

REM Combo value has changed
```

```
REM => requery form
```

```
Dim oeEvent As Object, ocCombo As Object, sSQL As String, oForm As Object
    Set oeEvent = Events(poEvent)
    Set ocCombo = oeEvent.Source
    sSQL = "SELECT [PRODUCT CODE],[DENOMINATION],[SUPPLIER ID] FROM PRODUCTS WHERE " _
        & "[SUPPLIER ID]=' " & ocCombo.Value & "' "
    Set oForm = Forms("myComboForm")
    oForm.RecordSource = sSQL

End Sub
```

Bookmark this page » » [Events Handler](#)