

# What's new in Visual Objects 2.8 SP3 (2834)

## Contents

Contents.....	1
Introduction.....	1
General changes.....	2
Changes in the Visual Objects Compiler .....	2
Source Code Control.....	2
Changes in the Visual Objects IDE.....	3
Source Code Editor .....	3
Debugger .....	4
IDE in general .....	4
Repository Browser.....	4
Menu Editor .....	4
DbServer Editor.....	4
Window Editor .....	4
Changes in the Class Libraries .....	4
Runtime DLL .....	4
DBFMDX RDD.....	5
_DBFCDX RDD.....	5
System Classes .....	5
SQL Classes.....	5
GUI Classes .....	5
OLE Classes.....	6
Internet Classes.....	6
Internet Server classes .....	6

## Introduction

This document summarizes the changes that have been made to Visual Objects 2.8 since the release of Service Pack 2 in September 2008.

**The changes in build 2834 have been marked in red.**

The document has been divided in several sections to make it easier to identify the area that the changes affect. These sections are:

- Changes in the Visual Objects language and compiler
- Changes in the Visual Objects repository, the Source Code Control API and Linker
- Changes in the IDE including its subsystems and tools
- Changes in the Class Libraries
- Changes in the Examples

Please note that this patch does NOT come with a new system repository. All 3rd party products for Visual Objects 2.8 SP1 and Visual Objects 2.8 SP2 should work without problems with 2.8SP3.

The source code for the system classes DOES have some changes, and these changes have been incorporated in the system DLLs. Changes to system code in libraries of the System Repository are not visible. If you want to use these changes you need to import the proper SDK AEF files from the SDK folder.

## General changes

- This build of Visual Objects has been compiled with Microsoft Visual Studio 2008 (the C9 Compiler). As a result of that the sizes of DLLs and EXEs may have changed even when no changes have been documented in this document.

## Changes in the Visual Objects Compiler

- Fixed (a very old) problem in the Compiler when calling un-typed functions or methods with a parameter of type DWORD.  
*Explanation:* When a DWORD parameter is larger than MAX\_LONG, then the parameter is converted to a USUAL of type FLOAT. This causes a dynamic memory allocation. If this dynamic memory allocation triggered the Garbage Collector, then dynamic parameters on the stack were not properly updated by the Garbage Collector.
- Alias expressions were not working for aliases of type string. This has been fixed.
- Fixed an alias related problem in statements such as `_FIELD->(<Expression>)`
- Fixed an issue where changing the default value for a parameter would not trigger recompilation of code that was calling this function/method.
- VO was not always setting the correct dependency relation between a class and classes that were used by that class as instance variables. This could lead to several problems.

## Adam

- The 'hidden' flag for modules was not properly set when importing AEF and MEF files.
- Fixed an issue in SysMakeDir() that was introduced in build 2831 which caused problems when Adam was creating directories (for example when creating a new project or when working with SCC)
- Changes to external modules were not always seen when compiling. This has been fixed.
- When the 'standard UDC' path in Adam Options was > 12 characters, the path was truncated. This has been fixed.
- Fixed a problem where reindexing would crash due to a set that contained a reference back to itself
- Fixed a problem that could occur when an atom table contained corrupted data
- VO was always verifying the contents of a Project Catalog when it was opened. Projects that could not be accessed were deleted from the Project Catalog. This behaviour can now be controlled with a flag in the project catalog (`cavodir\Projects\projects.vo`). Adding an entry `RemoveInvalidProjects=0` to the [Projects] section in this catalog will prevent VO from deleting inaccessible projects.

## Source Code Control

- 'Get Latest version' at the application level was always touching all entities in the application. That has been fixed.
- VO now remembers the name of the last database and last project in the Project File. When a 'Get app from SCC' operation is performed this information is passed back to the SCC provider.
- VO now uses a different Icon to indicate that a module or app has changed in SCC. Since requesting this info is an 'expensive' operation it only refreshes the status when you refresh the icons from the Application Context menu, or when you open a context menu on a module. The icon for changed modules/options looks like this:



- Fixed an import problem after retrieving old files from SCC
- Fixed several errors that could occur when working with a SCC database for a long time. VO now periodically calls a method of the SCC Provider to keep the current connection alive. This behaviour can be modified by changing the following entry in the project file and/or cavo28.cfg:  
[Sc]  
Timer=1  
The entry specifies the timer period in minutes. Setting the timer value to 0 will disable the timer.
- When an external module is placed under SCC VO now automatically marks the external file Readonly when the module is checked in and Read/Write when the module is checked out to the current user.
- When a SCC operation showed dialog, then the active window was sometimes changed. VO now always restores the Active Window after a SCC dialog.

## Changes in the Visual Objects IDE

### Source Code Editor

- Autotype was sometimes causing 'hiccups' in the editor, resulting in characters being inserted in the buffer in the incorrect order. This has been fixed. The workaround (disabling 'Transition Effects' on XP, or disabling 'Fade or Slide Tooltips' in Vista) is no longer necessary.
- Autotype was no longer working inside comments. This feature has been restored.
- Autotype shortcuts that were starting with a dot were no longer working. This has been fixed.
- Added menu option and shortcut key (Ctrl-Shift-Y) to add current word to the list of token tips. This can be used to quickly replace an identifier in the list with incorrect casing with the current selected word.
- Line numbers are now shown relative to the start of the current entity and no longer as line numbers in the file (the width of the line # column however is determined by the actual # of lines in the current editor window)
- Parameter tips for functions/methods with a very long prototype are now displayed over multiple lines.
- Autotext would also appear after the Backspace key was pressed. This is now disabled.
- Fixed a problem where the cursor would be displayed on an incorrect line after opening multiple entities in the editor.
- Added a hotkey (Ctrl-M) and menu option (File - Open Current Module) that loads all entities of the current module in the editor.
- Added a hotkey (Alt-F2) to the Clear Bookmarks menu option
- Added a hotkey (Ctrl-Alt-F2) and a menu option to clear the bookmarks in all open editor windows
- Fixed an issue with token tips for symbols
- Updated the code that determines if a module is under SCC and sets the readonly property of the editor accordingly
- Fixed an issue with the Find dialog that could cause an exception when closing that dialog
- The 'resize font' behaviour in the editor (with Ctrl-Mousewheel) was not consistent with the Repository browser. The new behaviour for the editor is now:
  - Ctrl-Scroll towards = Zoom-In = Large Font
  - Ctrl-Scroll away = Zoom Out = Smaller Font

## ***Debugger***

- Fixed tooltip problem for parameters
- The debugger was not always deleting the hidden module that it uses to evaluate expressions. That has been fixed.
- Fixed a speed problem during debugging
- Improved debugger support for Multi Threaded DLLs
- Fixed an issue where breakpoints were accidentally deleted

## ***IDE in general***

- The dialogs in the IDE are now centered on the active monitor for multiple monitor machines.
- Closing the 'Select Project' dialog will now obey the 'Confirm on Exit' settings of the IDE
- The 'Select Project' dialog is now visible on the taskbar

## ***Repository Browser***

- Fixed a problem with restoring the IDE Desktop
- When opening a tool from the menu for the first time, the IDE would sometimes open a second instance of the Repository Explorer. This has been fixed.
- Clicking 'Edit' on an item of type class in Classview would cause an error message. This has been fixed.
- Fixed a problem where selecting a new treeview item would not refresh the listview
- Selecting/Deselecting the path name column for modules was not working properly from the View/Options dialog

## ***Menu Editor***

- Fixed a problem when saving menu entities that use the syntax to dynamically load captions from a resource DLL.

## ***DbServer Editor***

- Updated Field/Fieldspec code generator to properly handle fields with different names/captions etc than their fieldspecs

## ***Window Editor***

- The Window editor would try to load the bitmaps for the tool palette from the template directory but not the cursors. This inconsistency has been fixed

## **Changes in the Class Libraries**

### ***Runtime DLL***

- The performance of some of the functions inside the Runtime DLL has been enhanced. Especially in functions like Crypt() you may see the difference.
- The runtime was not properly initializing the SetPath() contents with the current path from the OS(). This has been fixed.
- The Crypt() function would crash when one of its parameters was a NULL\_STRING. This has been fixed.

- The algorithm for the QueryRTRegString(), QueryRTRegInt(), SetRTRegString(), SetRTRegInt() and DeleteRTRegKey() functions has changed a little bit, to make it behave better in a 'controlled' environment, such as Windows Vista with UAC turned on:
  - The QueryRT..() functions try to read from HKEY\_CURRENT\_USER first. When that fails, they try to read from HKEY\_LOCAL\_MACHINE
  - The SetRTReg..() functions will always update the value in HKEY\_CURRENT\_USER and will also try to update the value in HKEY\_LOCAL\_MACHINE. The return value indicates the result of the write operation to HKEY\_CURRENT\_USER.
  - The DeleteRTRegKey() function tries to delete both the key in HKEY\_CURRENT\_USER and HKEY\_LOCAL\_MACHINE.
- Several functions in the runtime were not setting the UsualType tag properly, which could cause problem when they were called from macros.
- The OS() function has been updated to return more recent OS versions when called with a parameter.
- The File() function was returning TRUE if a directory existed with the name of the file. That has been fixed.
- Fixed an issue in the Garbage Collector when collecting USUAL properties of objects that have an AXIT method.

## ***DBFMDX RDD***

- Fixed problem where an undefined function in an index condition caused a gpf

## ***\_DBFCDX RDD***

- Fixed a small memory leak

## ***System Classes***

- Fixed issue in the FileSpec class where copying to a UNC path (\\Machine\Path) was not working properly

## ***SQL Classes***

- Some people reported runtime errors 'Undefined method SqlSelect:Destroy()'. This has been fixed
- When opening a cursor for a driver that does not properly return the number of returned rows, VO was skipping to EOF to determine the number of rows. It does not do this anymore to improve the runtime speed.
- When reading long text fields, an extra Zero character was added after each block of 32k bytes. This has been fixed.
- Fixed issue for SqlConnection:Info() with values of WORD size

## ***GUI Classes***

- Fixed a problem inside Window: \_\_ProcessToolTip that could cause a runtime error "No Exported Variable: ToolTipText"
- ScrollEvent:Position was returning incorrect values for Slider Controls. This was causing problems when dragging the thumb in a slider control. This has been fixed.
- Optimized some code by removing sends on Array Elements and adding local typed Variables.
- The PASCAL calling convention was missing from some of the function declarations in the XP Theme support module. This has been fixed.

- Optimized DataBrowser: \_\_AutoLayout()
- Added three new cases to the Windows:Dispatch method. It now handles the following events too:
  - WM\_SysCommand (for values where wParam < 0x0000F000). This allows you to add your own menu events to the system menu, and have them send to the 'regular' menu handler
  - WM\_TIMER  
The standard windows WM\_TIMER events are now also sent to the Timer method of a window. The Event object is sent in as a parameter, so you can inspect the timer ID
  - WM\_APP  
All message ids >= WM\_APP are now sent to an AppMessage method of the window (if it exists). This method receives one parameter (the Event object)
- Fixed an issue with the Spinner class, and the SpinnerEvent class, where negative positions were not handled properly. The code now also uses the 32 bit version of the windows messages to handle larger values
- Added option to Control:LinkDF() and DataColumn:LinkDF() to control the method that is used to Read/Write field values from a DataServer. In the past VO was only calling FieldPut()/FieldGet() for classes of type DbServer, SqlSelect and SqlTable. For other classes VO/Vulcan was calling IvarPut()/IvarGet() to read/write. With this change you can now define a method IsBaseField() on your server class. This method receives one parameter (symDataField, the name of the field that is checked) and it should return a TRUE, indicating that you want to use FieldGet()/FieldPut or a FALSE, indicating that you have an Access and Assign and want to use IvarGet/IvarPut to read/write the field
- The Control:Status assign now also accepts a NULL\_OBJECT, which clears the current status label.
- The \_\_UnLink methods in DataBrowser and DataListView now have the proper parameter to match the method in the Control class
- Calling DialogWindow:ChangeFont after the window was destroyed would make all hidden windows on the desktop visible. This has been fixed.
- Fixed an issue in the ListView:ColumnOrder access
- When assigning a date range to a DateTimePicker now the time for the max date is set to 23:59:59.999 to allow all time values for the maximum date.
- Window:\_\_ProcessTooltip would crash for controls without tooltip **AND** without Hyperlabel.

## ***OLE Classes***

- Fixed a problem in OleAutoObject that would affect applications that use ocx controls, where people were changing properties from the OCX from inside event handlers.
- Added an OleBinary class. This class is **\_NOT\_** in the System Repo, but can be used to send Binary data to a COM component.

## ***Internet Classes***

- Added better error handing to CFtp:GetFile() and CFtp:PutFile()

## ***Internet Server classes***

- Fixed a problem in HTTPCGIContext:Init where reading the content in Post mode would fail.