WOW Extensions v12 Release Notes

Here are the release notes for each release of WOW Extensions.

- Changes in each revision of the WOW Extensions for RM/COBOL
- Troubleshooting Win32 and x64

WOW Extensions Version 12.18

DE10280 (645227) WIN	The WOW Designer's user interface is now scaled on high-DPI displays or any display where the system scaling is set to something other than 100%. Previously, the WOW Designer rendered as if the display was 96dpi regardless of the actual pixel pitch, which would result in very small fonts on very high-DPI displays, like the screens on 4K laptops or 4K desktop displays. There is no visible effect if the system scaling is set to 100% and is only possible on operating systems that implement this feature: Microsoft Windows 8.1, Windows 10, or later. This setting is controlled by a new preference that enables or disables scaling on high-DPI displays. This option is a checkbox on the Preferences->General page. The preference only affects the Designer's user interface and has no effect on the runtime system, which is currently always scaled. Note that leaving scaling enabled provides the closest match between WOW forms at design time and at runtime, but scaling may be disabled for compatibility with previous versions of the WOW Designer. The preference is stored using the **DpiAware** key in the **[Defaults]** section of the **CBLWOW.INI** file and any change to this setting requires that the WOW Designer be restarted to take effect.
DE7869 (639249) WIN	When changing monitor configurations like undocking a laptop connected to larger screens, it was previously possible for some dialogs like the properties or menu editor to remain off-screen. The work-around was to delete the saved window positions in the CBLWOW.INI file, resetting these dialogs to their default locations. The WOW Designer now ensures every window is at least partially visible at startup and moves dialogs to the nearest monitor. We still recommend that the WOW Designer be restarted when changing monitor configurations so it can adjust to a different scale factor.
DE8322 (645572) WIN	When the WOW Designer user interface is being scaled by Windows, tool window drag rectangles (the rubber bands) are not scaled with the rest of the user interface and will appear in the wrong locations. This is a problem in Windows and will be fixed in a future release of the operating system.
DE10915 (1096041) WIN	You cannot use "WOWSetProp" to check a radio button or check box on another form
621084 WIN	WOW Designer cannot write to .INI file in default Program Files location, moved into the user AppData folder

DE12566 (1121660) WIN	Font properties—those properties that allow a font to be specified—are among the unique properties of ActiveX controls and may be implemented by controls in different ways. An ActiveX control often accepts a text string that specifies the font in a control-specific way, and this is the form that is supported by all versions of WOW.
	
	However, starting with the 12.18 release, WOW also supports ActiveX controls that set FONT properties using the **IFontDisp** ActiveX interface. To support these controls, there is a new runtime option, described below, that when enabled causes the WOW runtime to check ActiveX properties with names that end in the letters "**FONT**" for this interface. If both are true, WOW will attempt to use the interface to set the font; if the interface is reported as not being present, WOW will fall back to the previous behavior, attempting to set the font as a string property.
	
	This new feature is optional because, while unlikely, it is possible that ActiveX controls in properly working WOW applications may not behave properly when queried for this interface. It is also possible that using the interface may result in a change in the appearance of the control. This feature must therefore be purposely enabled in the wowrt.ini file.
	
	To enable this new behavior, set the new "*Enable ActiveX Font Property Interface*" checkbox on the *Runtime Preferences* page in the Designer, or set the `EnableActiveXFontPropertyInterface` option to `True` in the `[Version Compatibility]` section of the **wowrt.ini** configuration file.
	
	If set to `True`, the format of the font string in *WowSetProp* string is:
	
	`fontface``, `*`fontsize`*`, BOLD|NOBOLD, ITALIC|NOITALIC, UNDERLINE|NOUNDERLINE, STRIKE|NOSTRIKE`
	
	The options must be comma-separated, any part may be omitted and the options canb be specified in any order. If *fontface* or *fontsize* is omitted, the WOW runtime will attempt to use the property's current font and font size, if available. If this is not possible, it will default to a *fontface* of "Arial" and a font size of 12 points. All other omitted options are set to FALSE (i.e. the NO variation).
	
	• *`Fontface`* may contain spaces.
	• *`Fontsize`* is the size of the font in points and may be specified as an integer like 8 or a decimal value like 8.5.
	• The other options are keywords and may be specified in uppercase or lowercase.
	
	For example, if the new functionality is enabled in the wowrt.ini file as described above, for a control that has TITLEFONT and HEADERFONT properties:
	
	`CALL WOWSETPROP USING WIN-RETURN my-activex-ctl-h`
	`'TITLEFONT' 'Tahoma, 8.5, BOLD',`
	`'HEADERFONT' 'Tahoma, 9'.`
	
	WOW will first check to see if `TITLEFONT` exposes the Font interface. If found, the interface will be used to set the font as specified above. If not found or the interface call fails, WOW will set the property to the string value as in all previous versions. In this case, we suggest that only the font face be specified unless the documentation for the control specifies something different.

1120480	A COBOL WOW design-time problem with the ComponentOne VSFlexGrid control displaying custom ActiveX properties has been corrected. This control requires an optional parameter that the WOW Designer was not providing. It is possible that other controls may also require this parameter, and those may also now work better in Design mode. In addition, this control provides an "About" method rather than the more common "AboutBox" method to display information about the control. Both methods are now supported by the Designer.
1120955	The COBOL WOW designer exits while generating code when project files are on a network. A problem in the WOW Designer that could cause a crash during "Build" code generation has been fixed: if the RMPATH environment variable was not set, an internal buffer overflow occurred when searching for COPY files to scan as part of the "Build" operation. This problem has been corrected and the RMPATH environment variable can be empty. A work-around is to set RMPATH to any directory, like "SET RMPATH=.".
DE59212 **DE80009**	The installed WOW sample projects previously would not run without being compiled, and the compiler, was not able to find the standard copy files (WINDOWS.CPY, WINDEFS.CPY, LOGFONT.CPY, etc.) unless RMPATH was set. The WOW Designer now always adds %RM_PROGRAM_DIR% from the environment, along with the directory that contains the running WOW Designer to the RMPATH to ensure the standard copy files can be located without further intervention.

WOW Extensions Version 12.17

638949 **WIN**	A problem that affected both CodeWatch and the COBOL WOW Designer has been fixed: dragging a toolbar or tool window on a multi-monitor Windows 10 system left ghost lines on the display. This fix is in Microsoft Windows 10 1809, 1903, and 1909, and is applied by the following updates: Version 1903, 1909: March 24, 2020—KB4541335 (OS Builds 18362.752 and 18363.752) https://support.microsoft.com/en-us/help/4541335/windows-10-update-kb4541335 Version 1809: March 17, 2020—KB4541331 (OS Build 17763.1131) https://support.microsoft.com/en-us/help/4541331/windows-10-update-kb4541331 Once these updates are applied, the problem will no longer occur.

WOW Extensions Version 12.16

The following reported product incidents (RPIs) have been fixed in this version:

1115357 **WIN**	A problem in the 64-bit version of the WOW runtime that prevented mouse button events from being propagated into the application has been fixed. This problem only affected applications that were generated by pre-2007 versions of the WOW Designer and the problem does not occur in the 32-bit version.

639387 WIN	A problem in the WOW Panels runtime where a multi-line Edit control in a WOW GUI application could lead to memory corruption has been fixed.

1115711 WIN	A problem in the 12.15 version of the WOW runtime has been fixed: if the following directory does not exist when the WOW runtime system is started:
	`"C:\Users\username\AppData\Roaming\Micro Focus"`
	A message box is displayed that reports this directory as missing. Note that WOW runs properly after the message box is dismissed.
	This situation occurs because the runtime is attempting to create the `WOWRT.INI` file in this directory:
	`"C:\Users\username\AppData\Roaming\Micro Focus\RM\CobolWOW"`
	And is expecting the "`Micro Focus`" directory to already exist.
	With 12.15 only, the work-around is to pre-create the "`Micro Focus`" directory above before launching the runtime system. If the work-around is used, be sure to create the above directory for each separate user that might run the WOW application on the current machine.

WOW Extensions Version 12.15

The following reported product incidents (RPIs) have been fixed in this version:

1098875 WIN	In the WOW extensions for RM/COBOL, a window activation problem has been corrected: a newly created modal form was not being activated if the window was created in a `SETFOCUS` event. A click in another application followed by a click in the modal form would work around the problem, but the root cause has now been fixed. Note that this problem was originally fixed in v12.14 Hotfix 1.
628144 WIN	For the WOW Extensions Designer, the `cblwow.ini` and `wowrt.ini` files have been moved from `%APPDATA%\Liant\CobolWOW` to the new standard user "application" roaming data directory: `%APPDATA%\Micro Focus\RM\CobolWOW` from. At startup, if these files do not exist, the WOW Designer will look in the old location. If the older `.ini` files exist, the WOW Designer will copy and upgrade the files, thereby migrating settings. The original file is not removed to allow for compatibility with older versions but note that settings changed in v12.15 or later will not be picked up by older versions.
1105267 WIN	The WOW Extensions Designer could unexpectedly terminate if a project was opened using a UNC pathname (i.e. `"\\server\share\project.wpj"`). This is because the WOW designer sets the current working directory to the network share, but also attempts to change the drive letter. This problem has now been fixed, but it is likely that some programs will not work unless the working directory has a drive letter, especially programs that use non-COBOL subprograms. There may also be ActiveX controls that will not run without a current drive letter. We strongly recommend that when projects reside on a network share, a drive letter be mapped to the share. For example, executing this command before launching the WOW Designer:

	`NET USE W: \\server\share` will allow the above project to be loaded as "`W:\Project.wpj`".

WOW Extensions Version 12.14

Due to a change in the RM/COBOL compiler, if a version 12 compiler is invoked from within the WOW Designer, it must be version 12.14 or later. Note that version 11, 10, 9, and 8 compilers may still be used, and pre-v12.14 compilers may be used to compile WOW programs from outside the IDE.

WOW Extensions Version 12.13

With this release, many WOW programs can now run with the 64-bit RM/COBOL runtime on 64-bit Microsoft Windows.

While RM/COBOL programs need not be recompiled when switching between 32 and 64 bits runtime systems, the correct 32-bit or 64-bit version of the WOW runtime system must be specified with the L= option on the runtime command line.

The 64-bit WOW runtime DLLs that are distributed with the RM/COBOL Runtime Plus system are located in the same directory as the 64-bit runtime:

- `C:\Program Files\Liant\RMCOBOLv12-64\WOWRT.DLL`
- `C:\Program Files\Liant\RMCOBOLv12-64\WOWMFCRT.DLL`

The WOW Designer is still a 32-bit program but can build and run either 32-bit or 64-bit programs. Be sure to read the section on ActiveX controls later in this document for some notes regarding 64-bit controls.

Changes in the WOW Designer for v12.13

The WOW Designer is a 32-bit program but can develop for and launch either the 32-bit or 64-bit RM/COBOL runtime system.

The CBLWOW.INI File

The `cblwow.ini` file stores configuration information for the WOW Designer.

When WOW is installed, a prototype version of `cblwow.ini` is installed into the same location as the WOW program. However, recent versions of Windows do not allow files in the installation directory to be changed after installation.

Starting with version 12.13, the WOW Designer looks for this file only in your **ApplicationData** directory here:

```
%APPDATA%\Liant\CobolWOW\cblwow.ini
```

For example,

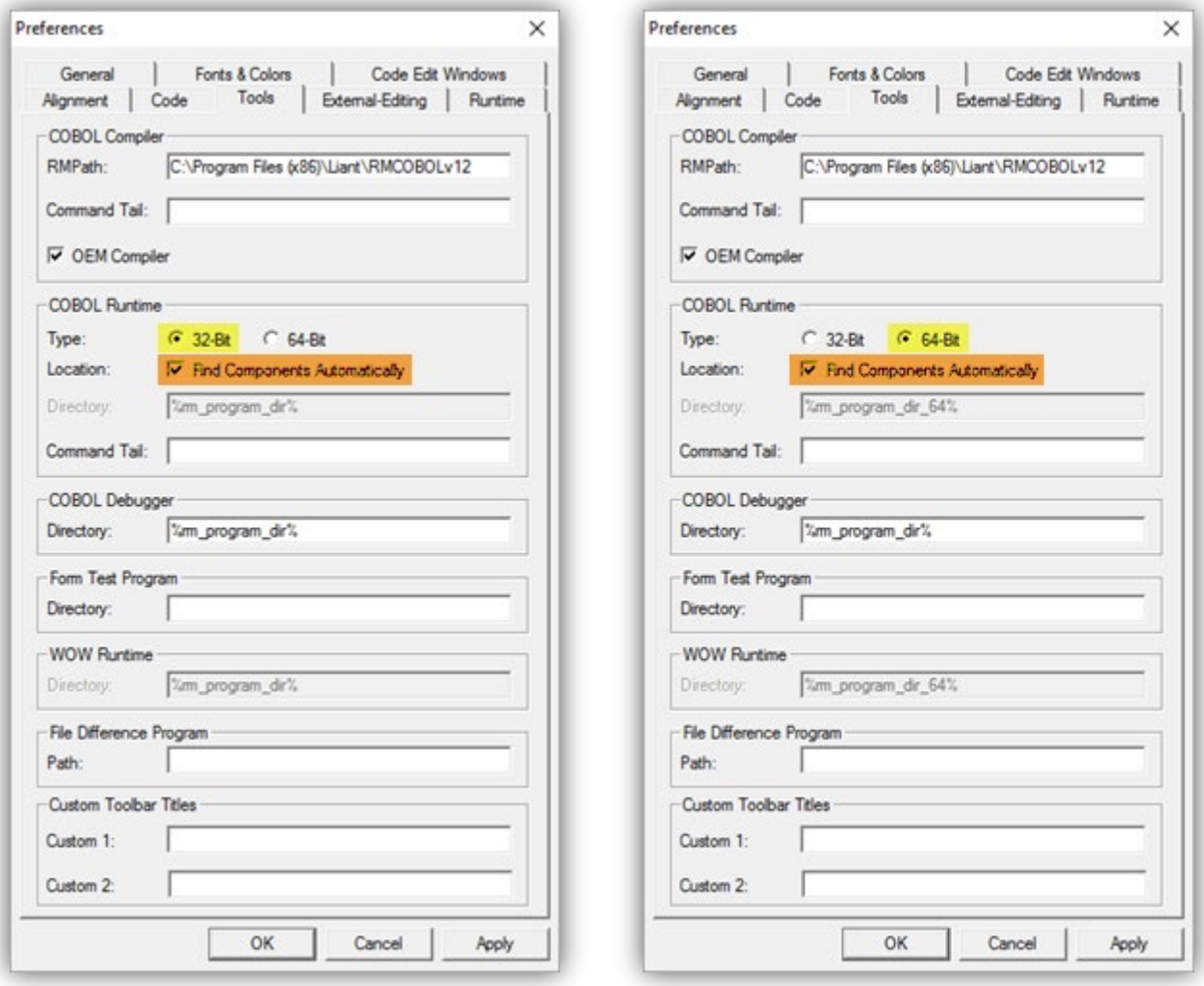
```
C:\Users\JohnSmith\AppData\Roaming\Liant\CobolWOW\cblwow.ini
```

Each user has a private application directory. The WOW Designer will copy the `cblwow.ini` file from the directory that contains the WOW program into the above **ApplicationData** subdirectory the first time the application is started.

Note that if you install the 12.13 WOW Designer over an earlier WOW Designer, your current `cblwow.ini` file will be copied and used, and the original file will not be altered or referenced by the 12.13 WOW Designer.

Preferences Dialog

The WOW Designer can start either the Win32 or Win64 version of the RM/COBOL runtime system. The system that will be launched by the Designer when **Project→Run** is selected is determined by a new option in the **Options→Preferences→Tools** property sheet:



These options apply to all projects and are not project-specific. The new options are:

32-Bit	Uses the 32-bit runtime system, if installed, when Project→Run is selected.
64-Bit	Uses the 64-bit runtime system, if installed, when Project→Run is selected.

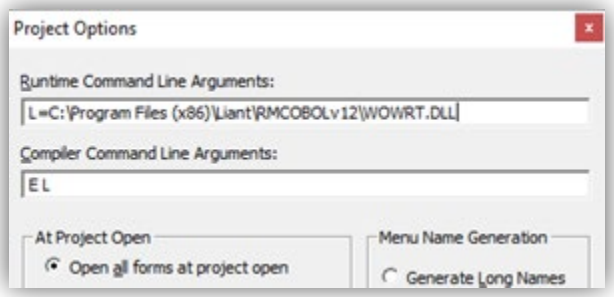
Find Components Automatically is another new option. If this checkbox is selected (recommended), the WOW Designer will locate the runtime system automatically and the entries for **COBOL Runtime Directory** and **WOW Runtime** below are not used.

If this option is not checked, you must enter the location of the runtime system and the WOW runtime in the edit fields as before, separately for 32-bit and 64-bit programs. Note that the WOW Designer tracks 32-bit and 64-bit setting separately.

If you choose to not find components automatically, a new environment variable set at installation, `%RM_PROGRAM_DIR_64%`, may be used to specify the 64-bit runtime location. Note that `%RM_PROGRAM_DIR%` continues to be available for 32-bit programs.

The Project Options Dialog

When a new project is created, the **Project→Options** dialog specifies a command line parameter that loads the WOW runtime DLL (`wowrt.dll`). Previous projects specified a complete path to the WOW runtime as a library `L=` option:



For projects created with version 12.13 or later, only the name of the WOW runtime DLL itself is placed here:

And the **WOW Runtime Path** in the **Options→Preferences→Tools** is then used to locate the runtime system.

Migrated projects are not updated automatically. Because there are different `WOWRT.DLL` libraries for 32-bit and 64-bit runtime systems, you may have to update this field when switching between 32-bit and 64-bit programs. The best practice is to set this field to `L=WOWRT.DLL`, then check "**Find Components Automatically"** or set the correct paths on the **Options→Preferences→Tools** page and let the WOW Designer locate this DLL.

ActiveX Controls

While Windows common controls are available both in 32-bit and 64-bit flavors, this is not necessarily true for ActiveX controls.

The ActiveX control must match the runtime system being used: 32-bit WOW programs can only use 32-bit ActiveX controls, and 64-bit WOW programs can only use 64-bit ActiveX controls.

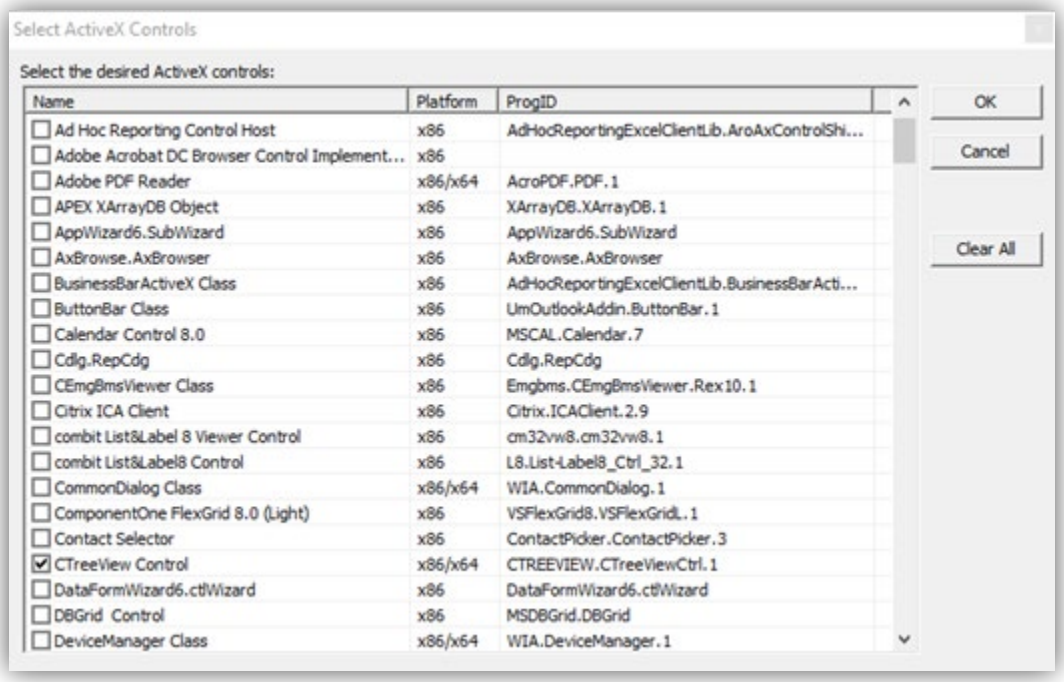
While many vendors have chosen to implement 64-bit versions of their controls, it may not be possible to obtain a 64-bit version of specific ActiveX controls.

At design-time, the WOW Designer only uses 32-bit ActiveX controls.

Selecting ActiveX Controls

The **Options→Select ActiveX Controls** menu pick displays a dialog that allows ActiveX controls to be added to the WOW Designer's toolbox.

In v12.13, this dialog has been enhanced to use checkboxes to select ActiveX controls, and there is an indication regarding whether the control is available for x86 (Win32-only) or x86/x64 (Win32 and Win64).



Only the controls checked in this dialog will appear in the toolbox.

Changes in the WOW Runtime for v12.13

The WOW Runtime is a non-COBOL library DLL loaded by the RM/COBOL runtime system using the **L** command line option – for example, **L=WOWRT.DLL**. There are two versions: a 32-bit version and a 64-bit version that must be used with the matching runtime system.

The WOWRT.INI File

The `wowrt.ini` file stores configuration information for the WOW runtime system.

This file is initially created by the WOW Designer; it contains settings and preferences that are set in the runtime project settings. Up through v12.12, the `wowrt.ini` was created in the same directory as the `wowrt.dll` program. However, on recent versions of Windows, this location in `C:\Program Files` is not writable by default.

Starting with version 12.13, the WOW Designer now creates this file, and the WOW runtime looks for this file in your **ApplicationData** directory here:

```
%APPDATA%\Liant\CobolWOW\wowrt.ini
```

For example,

```
C:\Users\JohnSmith\AppData\Roaming\Liant\CobolWOW\wowrt.ini
```

Each user has a private application directory. If a file exists in the WOW program directory, the WOW Designer and WOW Runtime system will copy the `wowrt.ini` file from the directory that contains the WOW program into the above **ApplicationData** subdirectory the first time the application is started.

Note that if you install the 12.13 runtime system over an earlier runtime system, your current `wowrt.ini` file will be copied and used, and the original file will not be altered or referenced in the future. If the file does not already exist, it will be created in the **ApplicationData** directory.

RPI 609620 Modal form is disabled when called from an ActiveX Control

A problem that occurred when an ActiveX button control created a modal dialog has been fixed. This problem first appeared in version 12.03, and prevented mouse clicks from being sent to the new modal dialog until the keyboard was used, or the user clicked on another application.

Troubleshooting Win32 and x64

Here are some errors you may encounter, and suggested solutions.

COBOL Error 223 and/or COBOL Error 214

If an error like this appears when you start a COBOL program from within the Designer:

```
COBOL procedure error 223 (nonCOBOL library load failure). Error processing library C:\Program Files
(x86)\Liant\RMCOBOLv12\WOWRT.DLL.
COBOL procedure error 214 (library not valid). Error starting application…
```

You may also see a dialog box that reports **COBOL error code 214.** To troubleshoot, use **Project→Options** and make sure the **L=** option refers to a **WOWRT.DLL** that is compatible with the runtime you are using – either 32-bit or 64-bit. If you use the **"Find Components Automatically"** option in **Options→Preferences→Tools**, it is normally sufficient to specify **L=WOWRT.DLL** (now the default for newly created projects) and allow the WOW Designer to set the appropriate path.

If you specify additional non-COBOL libraries, they must also match the runtime system – 32-bit or 64-bit. In this case, to switch between these systems, you can place the 32-bit control in the 32-bit runtime directory, and the 64-bit DLL in the 64-bit runtime directory and then use **L=** to simply specify the name and not the full path.

WOW Extensions v11 (and earlier) Release Notes

This document contains the following sections:

- Changes and Enhancements
- Documentation Changes
- Technical Notes
- Compatibility with Other Products
- Problems in This Release
- Media Contents

Changes and Enhancements

Changes in Version 11.01:

See the Designer help file for changes and enhancements in this version.

Changes in Version 10.01:

Numerous defects reported against version 9.01 have been resolved. The following enhancements have been made:

- A new **Tab Order Editor** dialog box displays a list of controls on the active form and allows you to change their tab order easily. Similarly, a new **Z-Order Editor** dialog box displays a list of controls on the active form and allows you to change their control stacking order (or z-order) easily (that is, the order in which controls are created).

- Up to 16 user-defined custom colors are now saved in the WOW Designer's configuration file (`cblwow.ini`). This means that the custom colors are always available to all projects.

- A `[VERSION COMPATIBILITY]` section was added to the WOW runtime initialization file (`wowrt.ini`) in order to provide the capability to restore pre-version 9 runtime behavior. Note that it is also possible to define these settings within the WOW Designer by using the Runtime Preferences page.

- The `[WOWRT]` section of the wowrt.ini file contains a new option, `ToolTipTimeOutSeconds`, which allows you to set the length of time, in seconds, that tooltips are visible at runtime.

- It is now possible to prevent ActiveX event code from being executed while another event is being processed by setting the **AllowMultipleEvents** property to `FALSE` for each ActiveX control, and by either using the **Enable ActiveX Multiple Event Blocking** option on the Runtime Preferences page or setting the `EnableActiveXMultipleEventBlocking` option in the [WOWRT] section of the wowrt.ini file.

- The location of the WOW Extensions runtime system, `wowrt.dll`, can now be specified on the Tools Preferences page.

- The External Edit Comparison dialog box contains a new command button, **Run Diff**, which causes the WOW Designer to write the information in the `Original Code` and `Externally Edited Code` areas to a file and invokes a file comparison tool to report the results of the comparison. If you wish to use a file comparison tool other than the default, Microsoft Windows ***windiff***, the path of that tool can be specified on the Tools Preferences page.

- A new option, **Generate Old-Style Working Storage Guard Names**, has been added to the **Code Preferences** page, it causes the WOW Designer to generate `Working-Storage` guard names that are limited to 30 characters in length, which was the case in versions prior to 10.01. The version 10.01 Designer will generate names up to 240 characters long. Checking this option prevents pre-10.01 projects from having compilation errors due to references to the old 30-character names.

- The **View menu** now includes a **Lock Icons** command to show or hide the lock icons on locked controls.

- The **Form menu** now includes a **Test** command. The location of the form test program, ***wowtestform.exe***, can be specified on the **Tools Preferences** page. This test form program is executed with the ***formname*** as a command line option, and adds a `.wfd` or `.wws` extension to the ***formname*** depending on the value of the **WSDefinition** property of the form.

- Default font settings (font name, style, special effects, if any, and size) for controls can now be established at design time using the **Control Font** option on the **Default Colors and Font Preferences** page.

- The **Enable Mouse Move Events** option was added to the **Runtime Preferences** page to allow developers to control whether the WOW runtime will recognize mouse move events. Mouse move events consume an inordinate amount of CPU resources even when no event code is attached. The ability to de-select this option is useful in preventing such resource consumption unless the developer deems it necessary.

- **MouseDown**, **MouseMove**, and **MouseUp** events were added to many controls as well as to forms.

- The date time picker and month calendar controls feature new events. Support is now available for **GotFocus**, **KeyDown**, **KeyPress**, **KeyUp**, and **LostFocus** events.

- The WOW Designer online help file now documents whether properties can be manipulated at design time, runtime, or both.

- Support for the bitmap control is now available for panels.

- A new form property, **KeyPreview**, has been added that indicates whether the form will receive key events before the event is passed to the control that has focus.

- The **Version** (runtime) property has also been added for forms.

- The **WSDefinition** property has been added to forms which enables definition data to be written to a file having a `.wfd` (WOW form definition).

- In addition to setting tool tips at design time, it is now possible to set them at runtime using the **ToolTipText** property.

- The font properties (**FontBold**, **FontItalic**, **FontName**, **FontSize**, **FontStrikethru** and **FontUnderline**) of a status bar control can now be set at design time as well as at runtime.

- Several new ActiveX properties have been added, including the following:

    - **AllowMultipleEvents**
    - **EnableKeyPressforTabMode**
    - **Group**
    - **NewDoMethodArgumentPassing**

- Two new properties have been added to the **Edit Box Control**. **SelectAllOnGotFocus** is a design-time-only property that determines whether the text in an edit box control will be selected when the control has focus. The **WantPopupMenu** property determines whether a standard Windows popup menu will display when the user right-clicks while in an edit box.

- The **Tab Control** now has **Accelerator** and **TabEnabled** properties that can be set for individual tabs on the control.

- The **Toolbar Control** now has **BtnToolTipEnabled** and **BtnToolTipText** properties that can be set for individual buttons on the toolbar.

- Several new WOW functions have been added in version 10. (For complete details on the following, see the ***Functions and Messages*** online help file.)

    - **WOWGETINDEXPROP** and **WOWSETINDEXPROP** get and set, respectively, the value of one index property for certain standard controls.
    - **WOWFORMTOWOWVERSION** retrieves a string containing the version of WOW Extension used to create the form.
    - **WOWGETWINDOWTYPE** function retrieves the type of a window.

- A Windows' services version of ***RPCPlusServer***, ***RPCPlusService***, is now available with the RM/COBOL Development and Runtime Plus systems.

Changes in Version 9.01:

A number of defects reported in version 9.0 have been resolved. The following enhancements have been made:

- A new dialog box has been added to make changing the tab order and z-order of controls easier. The order editor can be accessed from **Tab Order Editor** and **Z-Order Editor** commands on the **Control menu**. Click on the controls in the order that you want to set the tab order or z-order. A number before control name indicates the current order and will change to its new order when clicked. The next number to be assigned can be changed by an edit box at the top of the dialog box.

- The **Project Tree** has a new look. It displays the complete tree structure of the form and indicates nesting for the controls embedded in a container and controls embedded on each tab of a **Tab Control**.
- The **Font** properties of a **Status Bar** can now be set at design time.
- The old `wowrt.ini` file option "**EnableOldStyleDoMethodArgumentPassing**" has been replaced by a new ActiveX property, "**NewDoMethodArgumentPassing**", which allows this behavior to be set differently for each ActiveX. The old `.ini` option will be used when processing `DoMethod` calls for a form built with a pre-9.01 WOW Designer.
- A new wowrt.ini file option "**EnableActiveXmultipleEventBLocking**" has been added to prevent ActiveX event code from being executed while another event is being processed. In order to prevent multiple events from occurring, this option must be set to TRUE and each ActiveX that the developer wishes to have multiple events blocked for, must have the property "**AllowMultipleEvents**" also set **FALSE**.
- A new ActiveX property "**EnableKeyPressForTabMode**" has been added to allow the version 9 runtime to use **KeyPress** events for tabbing (Dialog Motion) between controls (this was the pre 9.0 behavior). The version 9 behavior intercepts keyboard messages before they become ActiveX events and causes tabbing to occur at that point in time (the **KeyPress** event is not generated in this case).

Changes in Version 9.0 Service Pack 1:

This release supports running forms produced by versions of the WOW Designer prior to Version 9.

Several `.ini` file options have been added to enable runtime behaviors that were changed in Version 9. See the *"Technical Notes"* below for a description of these options.

Changes in Version 9.0

A number of defects have been resolved. The following enhancements have been made:

- Arguments are now supported on ActiveX events.
- A new dialog allows viewing all of the methods for a selected ActiveX.
- The designer will now optionally generate nested COBOL programs.
- Several new runtime functions are now available including:
    - `WowInitControl`
    - `WowInitAllControls`
    - `WowInputBox`
    - `HtmlHelp`
    - `ActiveXGetAxHandle`
    - `GetTempPath`
    - `WowInputBox`
    - `WowAddMultipleStrings`
    - `WowMultiControlGetProp`
    - `WowMultiControlSetProp`
    - `GetCursorPos`

    - SetCursorPos
    - WowStripTrailing
    - GetVersionEx

- Several new Thin Client functions are now available including:

    - GetClientArgs
    - GetClientAddr
    - RemoteCopy
    - RemoteCmd
    - RemoteShellExecute
    - WowThinClientSetNoWait

- Using the Spacing and Alignment commands, controls within containers will now align/space within that container.
- A new property for forms, "**ToolWindow**", will cause that form to have a shorter title bar and not appear in the Window's task bar.
- The code templates now appear in their own control bar.
- Events can now have their own private working storage (supported in the **Code Edit** windows).
- Controls have a new property, "**Tag**", which can store application private data.
- A new control property "**ToolTip**" allows developers to add tooltips to their controls.
- The Project **"Save As..."** command has several problems fixed.
- Menus can be saved and reloaded into other projects.
- The Designer's toolbars are now customizable.
- Several new standard controls are now available: **DateTimePicker**, **MonthCalendar**.
- COBOL compiler output now appears in a window within the Designer.
- The WOW Thin Client now supports `ACCEPT` and `DISPLAY` from a Windows server.
- A **Tab Control** editor dialog has been added.
- Accelerators can now be assigned to the tabs of a tab control.
- The WOW runtime [`WOWRT`] and `[Panels Function Keys]` section of the `cblwow.ini` file have been moved to a new WOW runtime file named `wowrt.ini`.
- New menu picks have been added:

    - **Project/Regenerate w/Forms** - regenerate COBOL code for the project and all of its forms
    - **Form/Open All**
    - **Form/Save All**

- **Form/Close All**
- **Form/Regenerate All**
- **Panel/Open All**
- **Panel/Save All**
- **Panel/Close All**
- **Panel/Export All**
- **Panel/Regenerate GUI All**
- **Panel/Import Preferences**

Changes in Version 4.0

Resolved defects and enhancements:

- Fixed a problem when double-clicking on `.wpj` files.
- Allow the Designer to resize combo boxes when changing the font size.
- The WOW Thin Client now supports Windows printing from a Windows or UNIX server.
- The WOW Thin Client now supports `ACCEPT` and `DISPLAY` from a UNIX server.
- Fixed several problems with ActiveX controls.

Changes in Version 3.12.00

Fixed a problem with ActiveX key events.

Changes in Version 3.11.00

Resolved defects and enhancements:

- Fixed ActiveX selection dialog to exclude many ActiveX controls that were not usable with WOW.
- Fixed several problems that were causing lost Windows resources.
- Fixed parsing of `01` items when looking for linkage items.
- Added an `.ini` option to allow importing non-ANSI characters from older RM/Panels text fields.
- Fixed a problem when importing a Panel with `OCCURRING` fields (data was displayed in wrong field).
- Fixed storing of COBOL data in `GetClassInfo`.
- Several changes to make buttons (push/radio/check) behave more like Visual Basic 6.
- Check for numeric arguments correctly (**NSE** should be treated as alpha).
- Fixed setting of parent for ActiveX controls.
- Fix `WaitCursor` handling.

- Fix focus setting when returning to a WOW form from another form or application on Windows 2000 or WindowsNT.

Changes in Version 3.10.00

Resolved defects and enhancements:

- Add transparent background for most controls (defect 3066).
- Added the ability to print forms from the designer.
- Added a printable report of items to distribute with a WOW application.
- Added the ability to select multiple controls and set their properties in the designer.
- Control properties can now be organized by category (defect 3356).
- Position properties for shape controls can now be set at runtime (defect 3369).
- Several changes to the documentation addressing inaccuracies in the WOW tutorial have been corrected (defects 3343, 3345, 3348, and 3333).
- **TEXTMATRIX** now works with **VSGRID** (defect 3352).
- Corrected a problem with the cursor not changing when the mouse is over an edit box (defect 3355).
- Copy/Paste of multiple fields to tab now places the controls correctly (defect 3374).
- Changed the behavior of fields marked as "**Always Disabled**" in RM/Panels (defect 3376).
- Corrected a problem with files not showing when using "`*.*`" in the **Edit Project** window (defect 3380).
- Corrected a problem with the incorrect system color being highlighted (defect 3394).
- Corrected a problem with the Help menu options being disabled while editing a Panel (defect 3399).
- Corrected a problem that would not allow the border type of a form to be changed from "Dialog" or "Sizing" to "None" (defect 3404).
- Corrected a problem with ***AxGrid*** and setting the **ColAllowEdit** property that would causes the **BtnClick** event to activate (defect 3315).
- Corrected a problem with foreign characters that are not being displayed in RM/Panels help and error messages (defect 3335).
- Corrected a problem with **Scroll bars** on **Tab controls** not working at Runtime (defect 3349).
- Corrected a problem with `GETNEXTDLGTABITEM` not returning any control-ID (defect 3358).
- Corrected a problem where more than one option button could be to set to true (defect 3359).

- Corrected a problem where the **TabIndex** stop is being ignored with ActiveX controls (defect 3368).
- Corrected a problem where WOW forms are not properly added to WOW Project (defect 3379).
- Corrected a problem with the **Up/Down Control** not functioning in a group control (defect 3382).
- Corrected a problem with the "**CurTab**" property not working at runtime if the "Buttons" property is set to "True" (defect 3400).
- Added `AxGetHWnd` to Thin Client (defect 3401).
- Corrected a problem with the parent window "bleeding" into child window (defect 3402).
- Corrected a problem with the space bar not working in the multi-line edit box when entering text (RM/Panels) (defect 3413).

Changes in Version 3.00.01

Resolved defects and enhancements:

- The Designer now opens projects and forms with exclusive access (defect 3259).
- **FKEYTEST** in the original delivered Panels library causes a page fault under special circumstances (defect 3270).
- Removed directory path length restriction (defect 3271).
- Assure control icons are hidden at runtime (defect 3272).
- Do not lose font size when using "restore properties" or when creating a new control (defect 3273).
- Add ability to specify system colors (defect 3274).
- Correct saving of Greek characters (defect 3279).
- Correct dynamically created **EditBox** having wrong background color (defect 3283).
- Add ability to bring up **Menu Event** handling code from the **Menu Editor** (defect 3287).
- Restore `SelChange` and `SelChanging` events to **Tab control** (defect 3290).
- The `KeyAscii` value is now returned to the **KeyPress** event in `AXN-EVENT-RESULT` (defect 3210).
- Application Error on Windows 2000 (defect 3269).
- Tool tips are now available on controls at design time.
- Project "**Save As**..." dialog has been added.
- New view code toolbar buttons have been added.
- Finding code text has been enhanced with a "**Find in Files**" capability.

- The most conspicuous change is the addition of "**System Colors**" to various color dialogs. System colors are those colors returned from the `GetSysColor` Windows API function. The use of system colors, instead of explicit RGB colors, is encouraged so that an application's visual presentation will track the color settings made by the end user in the **Display Properties Appearance** tab. This can be especially important when an application must be made accessible for the visually impaired.

- The help files and other documentation will be modified in the upcoming version 3.10.00 release to reflect the minor changes caused by these enhancements and corrections.

Changes in Version 3.00.00

This section describes changes that have been made since WOW version 2.27 release. Where necessary, further information about these changes is included later in this document.

- The license file (***license.vlt***) must be present in the same directory where the Designer (***cblwow.exe***) is located.

- The WOW RM/Panels runtime (***wowpanrt.dll***) does not require a license file (***license.vlt***).

Documentation Changes

Micro Focus now distributes the documentation for this product on the software distribution media (CD-ROM). This electronic documentation is formatted in Adobe Portable Document Format (PDF). There is one PDF file per manual, each with the extension `.pdf`. *Adobe Acrobat Reader* is required to view and print the PDF documentation. If you need to install this software it is available on the product CD and is also freely available for download for most operating systems at http://www.adobe.com.

The PDF file for this product is the WOW Extensions User's Guide. On a Microsoft Windows system, this PDF file is located in the directory `x:\docs`, where `x:` is your CD-ROM drive. (Access to this documentation will also be provided by a shortcut icon entry in the **Programs** folder during installation of the WOW Extensions application.)

In addition, WOW Extensions also comes with extensive online Help files, which are designed to help you learn and use the product. You can access Help through the Help menu, or by pressing F1 or the Context Sensitive Help toolbar button to obtain context-sensitive help for particular parts of the Designer programming interface. The Help files include the Designer, a fundamental guide to the elements of the Designer interface, and *Functions and Messages*, a comprehensive reference documenting the ActiveX, WOW Extensions, and Windows API functions and messages used in WOW Extensions.

Note: The WOW Extensions documentation set assumes you know how to use a mouse, open a menu, and choose menu and dialog box options.

Technical Notes

The following sections contain notes that describe various WOW behaviors.

Restoring pre-Version 9 Behaviors

Several options have been added to restore pre Version 9 runtime behavior. These options are in the new `[VERSION COMPATIBILITY]` section of the WOW Extensions runtime **.ini** file, `wowrt.ini`.

The options and their default values:

`EnableDoubleClickOnNonStandardListBox` `(FALSE)`

Windows does not signal a double click on non-Standard List Boxes. This option turns on the simulation of this event that occurred in pre version 9 runtimes.

`EnableOldStyleFormSizeMode` `(FALSE)`

Previous version of WOW were not consistent in the handling of `SetProp` and `GetProp` when changing the size of a form. The values set by `SetProp` were not the same as those returned by subsequent `GetProp` operations. Version 9 fixed these inconsistencies. This option allows the old behavior to be turned on if desired.

`EnableOldStyleDoMethodArgumentPassing` `(TRUE)`

Certain ActiveX methods did not behave properly with the pre-version 9 method of passing arguments. Version 9 fixed these problems, but to be compatible with old WOW applications, this new method was not enabled by default. This option allows this new method to be turned on.

`EnableControlEventsOnTabKey` `(FALSE)`

Version 9 does not pass **Tab key** events to controls when they are being used for dialog motion. This option allows the control to receive the key events.

`EnableAppSwitchLostFocusEvents` `(FALSE)`

Version 9 suppresses **Lost Focus** events when the user switches to another application (this is also the Visual Basic behavior). This option allows the **Lost Focus** event to be received by the WOW application.

`EnableNonChildLostFocusEvents` `(FALSE)`

Version 9 suppresses **Lost Focus** events when the user switches to another non-child WOW form (in the same application). This option allows the Lost Focus event to be received by the WOW application.

ActiveX Notes

Calling ActiveX `METHODS` using literals is discouraged. It is recommended that a `WORKING STORAGE` variable or constant be used.

For example:

```
WORKING-STORAGE SECTION.
78  POPConnect          VALUE "Connect".
...
PROCEDURE DIVISION.
...
Call AXDoMethod Using Win-Return
                      POP3-H
                      POPConnect.
```

Some ActiveX controls contain methods that have optional parameters. If these parameters are not defined by the control as a `VARIANT` data type, they must be specified when called from WOW Extensions.

Form Notes

If you experience problems with a parent form bleeding into its child forms, set the parent's form property "**ClipSiblings**" to TRUE.

Using the Euro (€) with RM/COBOL and WOW Extensions

WOW Version 3.10 has a new initialization section: `[INTERNATIONALIZATION]`.

This section supports three keywords:

`EuroSupport=TRUE|FALSE`

Turns the ability to map the Euro currency symbol (€) on or off. The default is `TRUE` (on).

`EuroCodePointAnsi=<value>`

Specifies the Euro currency symbol (€) codepoint for ANSI character sets.
`<value>` is in the range 0 thru 255 (can be specified in decimal or hex, e.g., `128` or `0x80`).

`EuroCodePointOem=<value>`

Specifies the Euro currency symbol (€) codepoint for OEM character sets.
`<value>` is in the range 0 thru 255 (can be specified in decimal or hex, e.g., `213` or `0xD5`).

Some notes about using the Euro symbol (Alt 0128)

In order to use the Euro, the font used must contain the Euro (€) symbol. Use the Microsoft Windows **Character Map System Tools Utility** to determine if the font contains the Euro character. From the 'Character Map' you can display the maps of different fonts. The Euro symbol will usually be located in position 128. Some fonts that contain the Euro are Courier New, Times New Roman, Arial and Tahoma.

If the Euro symbol will display but not print, it is likely that the internal printer font was used instead of the display font. The printer font may not contain the Euro. To enable the printer font, select the printer properties and go to the Fonts tab. From there edit the **Font Substitution Table** to send the font as character outlines.

Compatibility with Other Products

WOW version 9.0 requires RM/COBOL v9 or later.

Problems in This Release

There were no known problems at the time of release.

Media Content

The following is a list of files included on the WOW Extensions development system media:

File	Description
SupplementV12-WOW.pdf	This README file.
Migrate.txt	Migrating to WOW Extensions
cblwow.exe	WOW Designer Program
scilexer.dll	WOW Designer Code editor support
cblwow.ini	WOW Designer INI file
cblwow.chm	WOW Designer help contents file
wowpanif.obj	RM/Panels COBOL object library
wowpntst.cob	RM/Panels COBOL test program (object)
rmrpc32s.dll	COBOL-RPC server DLL
rpcinit.cob	COBOL-RPC initialization file (object)
cobolrpc.ini	COBOL-RPC INI file
runpan2.cob	RM/Panels runtime library (object)
license.vlt	License vault
license.vlt.lck	License vault lock file
windows.cpy	WOW Extensions copy deck
logfont.cpy	Windows logical font copy deck
samples*.*	Various sample WOW Extensions programs
SupportTools*.*	Miscellaneous diagnostic utilities.

The following files are included with the RM/COBOL Runtime Plus system:

File	Description
wowrt.dll	The COBOL WOW runtime system, loaded as an RM/COBOL library
wowmfcrt.dll	Part of the COBOL WOW runtime system, loaded when **wowrt.dll** is loaded
cblwow.exe	WOW Designer Program
wowrt.ini	WOW Runtime INI file