Infragistics Windows Forms 14.1 Service Release Notes – January 2015

Add complete usability and extreme functionality to your next desktop application with the depth and breadth our Windows Forms UI controls. Infragistics Windows Forms continues to make strides in user experience with the latest Windows® 7, Windows® 8, Office® 2010 and Office® 2013 styles.

Installation

Downloading	[Download Infragistics Windows Forms here.]()

What's Changed

Components	Product Impact	Description
WinComboEditor	Bug Fix	The typed text is cleared after exiting edit mode
WinDayView	Bug Fix	Appointments are not aligned correctly in certain scenarios
WinExplorerBar	Bug Fix	Mnemonics work even when UseMnemonic property is set to False
WinFormManager	Bug Fix	ArgumentOutOfRangeException is thrown when a mouse is over the close button and if ShowToolTips = true.
WinGrid	Bug Fix	Application stops responding on closing while the component is in edit mode on touchscreen
WinGrid	Bug Fix	ClearUnbound() does not clear unbound columns that are inserted before bound columns
WinGrid	Bug Fix	Cannot display image in a image column cell if datasource is old access database(*.mdb)

WinGrid	Bug Fix	Appearance object needs an InitializeFrom method to allow easier cloning Notes: We have added an InitializeFrom method to the Appearance class. However, due to certain technical considerations with deserialization, it is NOT recommended to override the Clone method and call InitializeFrom. Instead, the Clone method has now been updated to return an object of the derived class (instead of always returning the base Appearance class). As such, a better approach is to simply override InitializeFrom, call the base, cast the 'source' to the derived type, and to populate any extra properties of the derived class. This assumes that the derived type has a parameterless constructor (which it should). Protected Overrides Sub InitializeFrom(source As Infragistics.Win.Appearance) MyBase.InitializeFrom(source) Dim sourceAsMyAppearance As MyAppearance = DirectCast(source, MyAppearance) Me.SomeProperty = sourceAsMyAppearance.SomeProperty Me.SomeOtherProperty = sourceAsMyAppearance.SomeOtherProperty End Sub

WinGrid	Bug Fix	Scrolling when you have a lot of columns has degraded performance Notes: The grid was performing multiple scroll operations when you use the mouse wheel. Mouse drivers have a setting for the number of rows to scroll (typically 3). So when scrolling with the mouse wheel, the grid was performing 3 separate and distinct scroll operations: one line at a time. This was obviously inefficient and it also means that scrolling with the mouse wheel does not honor the ScrollStyle property. ScrollStyle is only honored when you scroll using the thumb or set the Position of the scrollbar directly. And since mouse wheel scrolling was being performed one row at a time, the property setting was not applied. To fix this, the grid now responds to the MouseWheel by explicitly setting the RowScrollRegions scroll position. This has the effect of drastically increasing both the actual performance and the responsiveness (since the grid updates immediately). This is technically a breaking change, since the event firing will now be different. The grid will fire the BeforeRowRegionScroll and AfterRowRegionScroll fewer times than it used to when you scroll with the mouse wheel. But since the event doesn't tell specify what operation took place, and gives you the OldState and the NewState in their entirely, this is very unlikely to cause breakage in any existing application.
WinGrid	Bug Fix	The touch support slows down the grid Notes: There was, indeed, a large performance hit on the painting of the grid caused by the Touch Metrics support. Many elements in the grid adjust their size based on whether touch metrics are enabled. The check to determine if touch metrics are enabled is not a terribly expensive check, but it is getting called quite often, when you have a large grid with many visible cells. To fix this, we have added a caching mechanism so that touch metrics availability is calculated

		only once each time the grid paints. The value is cached for the entire operation thus drastically increasing performance.
Office2010 Ribbon	Bug Fix	FileMenuButton and RibbonTab height differ
WinOptionSet	Bug Fix	UltraOptionSet does not apply its ErrorAppearance
WinOptionSet	Bug Fix	Images used for OptionSet glyphs (in an isl) are not honoring transparency Notes: This was not directly related to AppStylist, it was a problem with the custom glyph editor. If you open up the editor (by going to the GlyphInfo property, either in AppStylist on the UltraOptionSet or UltraCheckEditor component role, or the GlyphInfo property for either of those controls in the property grid) and select Custom and then assign some images, everything works fine. But as soon as you re-open the Glyph designer when there are already glyphs applied, the transparency on all of the glyph images was lost. This is now fixed and the designer will no longer corrupt the glyphs. However, any existing isl files or GlyphInfo properties where the transparency was lost cannot be corrected by this fix. It will be neccessary to reset the GlyphInfo and re-create it using corrected images.
WinTextEditor	Bug Fix	NotSupportedException is thrown when trying to get Text property when testing in VS 2013
WinTimeLineView	Bug Fix	AppointmentsDragDrop fires when AppointmentDragMode is WithinSameOwner and appointment is dropped on different owner
WinToolbarsManager	Bug Fix	UltraToolbarsManager's mnemonics remain active after enter in UltraGrid cell in edit mode
WinToolbarsManager	Bug Fix	System.ObjectDisposedException is thrown when press Alt with Ribbon set to AutoHide
WinToolbarsManager	Bug Fix	Memory leak with RibbonCustomizerProvider
WinToolbarsManager	Bug Fix	RibbonTool hotkeys do not work when the ribbon is dropped down.

WinTree	Bug Fix	Null reference exception on IsLastViewableSibling when calling VerifyChildElements(true)