Update 4b

This is the latest update for FrameworX version 10.1.

Published on April 16, 2026. Version 10.1.4b (build 1107)

Latest Update

Module	Description
AI Designer Tutor	**Learn FrameworX through guided, hands-on lessons directly in the Designer.** The AI Designer Tutor is a structured learning experience built into the FrameworX Designer. Instead of switching between documentation and the product, engineers can now learn by doing: the Tutor guides users through interactive lessons — from first steps to advanced configuration — entirely within live MCP sessions, using the actual project they are working on as the learning environment. Lessons are delivered progressively across three tiers (Essentials, Professional, and Expert), with individual progress tracked per user on the local machine. The Tutor picks up where the learner left off, across sessions and project restarts. A first-time nudge introduces the feature automatically when the AI Designer is opened for the first time. This release includes the full Tutor infrastructure: lesson delivery engine, progress tracking, and the first set of Essentials lessons — giving teams a practical path to onboarding new engineers faster and getting more value out of the AI Designer from day one.
Display	**TextBlock** • **Custom format patterns now display correctly when value is zero.** TextBlock components using format masks such as `##.#` were showing a blank field instead of `0` when the live tag value was zero. Standard .NET formats like `N2` were unaffected. Format masks now behave consistently across all values, preserving correct display behavior for projects migrated from earlier versions. • **Now correctly shows "?" for bad-quality tags in arithmetic expressions.** When a TextBlock rendered a tag expression involving arithmetic operations (e.g. `{Tag.Value/10}`), a tag with bad quality silently showed blank or `0.00` instead of the standard ? quality indicator. Operators can now rely on the ? indicator regardless of how the expression is structured — a critical reliability improvement for production environments where data quality visibility matters.
Script	**Script class read-only properties now accessible from Script Tasks.** Script classes exposing getter-only properties — a standard C# pattern — could not be consumed from Script Tasks: the compiler returned a type conversion error instead of reading the value. This is now resolved, and all standard property patterns work correctly across the scripting engine.
Securty	**Edit Security dialog now works for all built-in item types.** The Edit Security popup was silently non-functional for built-in entries in Reports (Forms and WebData) and Scripts (Classes and Tasks). Security permissions on these items can now be configured normally through the Designer interface.
Designer	**SmartClient and HTML5 WebClient links now work correctly over HTTPS.** With SSL/TLS enabled, the SmartClient and HTML5 WebClient quick-launch links generated by the Designer — which do not include a trailing slash — were returning a connection error in the browser. Both links now resolve correctly over HTTPS, ensuring that SSL-secured deployments are fully accessible directly from the Designer without manual URL adjustment.
Runtime / Platform	**FrameworX 10.1.4 is the long-term supported release on .NET 8.** Version 10.1.4 establishes the stable, production-hardened baseline of the FrameworX 10.1 platform running on .NET 8. This release will continue to receive targeted quality updates — and is fully supported alongside FrameworX 10.1.5 (.NET 10) in side-by-side installations on the same machine. Organizations that require .NET 8 compatibility — whether due to infrastructure constraints, certification requirements, or deployment standards — can remain on 10.1.4 with confidence while planning their own upgrade timeline.

Version 10.1.4 (build 1100)

Module	Description

Display	• **Linear Gauge — significant enhancements for alarm visualization.** The Linear Gauge component has been substantially extended, enabling engineers to configure HPG-compliant Alarm Linear Gauges directly from standard component properties without any custom development. New capabilities include: unlimited configurable range segments (removing the previous 5-segment limit); conditional range coloring — ranges can independently activate their highlight color only when the live value enters that range, fully supporting Lo, LoLo, Hi, HiHi alarm patterns; configurable setpoint markers (diamond or arrow style) with optional inline value label; a "Value Near Pointer" display option for live readout; per-setpoint enable/disable control; and improved Historian time label rendering. Fully supported in both WPF and HTML5 clients. • **TrendChart PenColor picker restored at runtime.** Clicking the PenColor column in the TrendChart's runtime DataGrid now correctly opens the color picker dialog, restoring full pen color customization during live sessions. • **TrendChart no longer plots quality=64 (Undefined) values during startup.** Trend charts previously plotted tag values during solution startup before drivers established communication, when tags still carried quality=64 (Undefined) and value=0. These spurious startup points are now suppressed for device-addressed and data-linked tags, keeping historical charts clean. • **LineColor dynamic preserves gradient configuration.** Gradient color settings defined in the LineColor dynamic — applicable to Buttons, Shapes, PolyLine, GridLine, Polygon, and similar components — are now correctly preserved when reopening the configuration dialog. Previously, all gradient values appeared to reset to black on reopen. • **PolyLine LineColor configuration consistency resolved.** Multiple configuration inconsistencies in the PolyLine component have been corrected: stroke thickness changes now correctly update the component's visual height; Fill and Line settings are now synchronized between the Settings and Appearance panels; and the LineColor configuration no longer resets to black when reopened. • **PolyLine FillColor now active on first insertion.** The FillColor dynamic on PolyLine components was non-functional when the component was first placed on a display, requiring the user to re-enter the configuration to activate it. This is now correctly initialized on insertion. • **Map component renders correctly on first HTML5 load.** The Map component now renders on the initial page load in the HTML5 client, eliminating the need for a manual refresh to display map content.
Alarms	• **Alarm Shelving (ISA-18.2 compliant).** Alarms can now be shelved — temporarily removed from the active alarm viewer while all historian logging and notification activity continues uninterrupted, fully complying with ISA-18.2 alarm management requirements. Three new properties are introduced: Alarm.Item.<Name>.IsShelved (bool) shelves or unshelves the alarm and automatically updates TotalCount and UnAckCount across the Group, Area, and root Alarm module; Alarm.Item.<Name>.ShelveTime (DateTimeOffset, read-only) records the timestamp when the alarm was shelved; Alarm.Group.<Name>.ShelveMaxDuration (int, milliseconds) defines the maximum shelve duration before automatic unshelving, following ISA-18.2 requirements. Properties can be set via scripts, displays, or any standard binding mechanism. • **Alarm Areas TreeView preserves navigation state.** Creating, renaming, deleting, or moving Alarm Areas in the Designer no longer collapses the entire TreeView. The tree now maintains its expansion state after each operation, consistent with the behavior of the Asset Tree.
HTML5 / Web Client	• **HTML5 client with Blazor elements now loads on mobile browsers.** Solutions using Blazor-based OpenSilver components now load correctly on mobile browsers, enabling full mobile access for modern HTML5 solutions. • **DataGrid columns with fixed/frozen layout now preserved after navigation.** DataGrid column configuration — including fixed and frozen column layouts backed by PostgreSQL tables — is now correctly retained when navigating away from and returning to a display in the HTML5 client. Column layout is fully restored without requiring any user interaction. • **New solution creation now available from browser.** Users can create new solutions directly from the browser-based Solution Center — no desktop Designer required. This completes the zero-install workflow by extending project creation to the browser environment. • **New TK.GetObjectValueAsync method for HTML5 compatibility.** A new async method TK.GetObjectValueAsync is now available for use in CodeBehind and scripts running in HTML5 pages. Unlike TK.GetObjectValue, which blocks execution on the first call in the browser environment (producing empty results), the async version correctly awaits the server response. Usage: await TK.GetObjectValueAsync("Tag.TagName").
Solution Center	• **Solution Center no longer shows the community edition link when a valid license is active.** The community edition option is now correctly hidden whenever a valid license is present, eliminating false licensing prompts that appeared before license information finished loading at startup.

Designer	• **Remote server connection indicator added to Designer.** When editing a solution via ProjectServer connected to a non-localhost remote server, the Designer now displays the server address in two locations: the status bar (bottom right, showing "Server: {IP}") and the title bar (showing "SolutionName Designer — Editing at server {IP}"). No indicator is shown for local connections. • **CrossReference navigation now lands on the correct object.** CrossReference now navigates correctly across all supported object types, including symbol labels, Historian Tags, Historian Tables, Report triggers, Script Task triggers, and CodeBehind references — each navigating to the precise location in the Designer rather than a generic page. • **CategoryType (EnableMCP) checkbox now saves correctly.** The EnableMCP checkbox in Solutions/Categories now correctly persists its value after saving. An additional related fix ensures that importing a SolutionCategories JSON file no longer forces EnableMCP to true on all rows regardless of the file contents. • **Category column is now editable on built-in items.** The Category column is now editable for built-in entries in Displays, Script Tasks, and Script Classes — both in single-row edit and via Edit Combined Rows. All other columns remain locked for built-in items. This allows teams to control MCP access to built-in components using the same workflow as custom items. • **Maps and FileExplorer controls can now be placed on WPF Page Designer screens.** The following controls can now be correctly dragged and dropped onto WPF Page Designer screens: FileExplorer, MapsESRI, MapsGMap, and MapsOSM. Previously, these controls appeared in the Toolbox but could not be inserted onto the designer surface. • **Column visibility preferences saved per screen.** Column enable/disable preferences are now correctly persisted per screen across multiple Designer pages — including Historian Tags, Device Points, Display Images, Script Tasks, Script Classes, Expressions, Security Users, and Secrets — preserving user layout choices across sessions and restarts. • **Plugin import performance significantly improved.** Plugin import operations that were taking several hours for large solutions (thousands of tags, alarms, and displays) have been optimized to complete in minutes.
Security	• **AutoLogOff now correctly enforces the configured session duration.** The Session > AutoLogOff setting now honors the DurationHours value as configured for both the Duration and Both modes. Previously, sessions were terminated after approximately 10 seconds regardless of the defined duration.
OPC UA	• **OPC UA Server now correctly enforces UDT member visibility.** Tags marked as Private within UDTs are now correctly excluded from the OPC UA Server address space. Tags marked as Protected are correctly exposed as read-only. Previously, all UDT members were published as read/write regardless of their configured visibility. • **OPC UA Importer stability improvements.** The OPC UA Importer has been stabilized to prevent Designer unresponsiveness when connecting to OPC servers with large or complex address spaces. Import operations now complete reliably across all tested configurations.
Tags / Asset Tree	• **Tags under AssetFolders with reserved names now appear in PropertyWatch and TrendChart.** Tags organized under AssetFolders named with reserved keywords — such as Historian, Alarm, Device, Display, Script, Dataset, or Security — are now correctly visible and browsable in both the PropertyWatch tree and the TrendChart tag browser.
Historian / Timebase	• **TrendChart now retrieves historical data from the Timebase connector.** Tags connected via the traditional Device > Channel > Node > Point configuration are now correctly routed through the Historian when using the Timebase (Timeflow) connector, enabling TrendChart to display historical data from Timebase alongside other historian sources.
Runtime / Devices	• **AccessType Disable property is now correctly scoped per channel.** The AccessType Disable property is now applied independently per channel. Previously it was shared globally, causing unintended access suppression across all channels when only a single channel was targeted. • **System Monitor now exposes .NET 8 Windows performance metrics.** System Monitor provides Windows CPU and memory performance data using the modern .NET 8 Process API, removing the dependency on the legacy PerformanceCounter infrastructure — which is not supported on .NET 8 — and broadening deployment compatibility across Windows and Linux environments. • **Automatic log file compression and rotation.** Log files are now automatically managed to prevent unbounded disk consumption in long-running production environments. Previous days log files are compressed to ZIP format and removed from disk, while the current day's log file remains active. Cleanup runs automatically once per hour.
Build / Scripts	• **Full Linux build no longer breaks Windows-only script classes.** Running a full build on the Linux build server no longer disrupts script classes that depend on Windows-only assemblies. The root cause was in the generation of public method signatures from script class tables — not in the Roslyn compilation itself — and has been corrected. Mixed-platform CI/CD workflows now operate reliably.
Import / Export	• **JSON export log now shows actual filenames.** The console output during JSON export now displays the correct filenames instead of temporary internal paths, making export logs actionable and easy to audit. • **Export UNS Objects to JSON now completes successfully.** The Export UNS Objects to JSON function (File > Export > UNS Objects to JSON) now correctly generates the output files in the Exchange folder. Previously the operation created an empty folder without producing any JSON output.

ControlLogix	• **Tag import now handles large tag sets reliably.** Imports via "Import Tags" or "TagProvider" now complete successfully for any tag set size. Previously, imports were failing with "Function failed to do described task" when the tag count approached or exceeded 1,000 tags. • **L5K import now correctly identifies REAL tag types.** Tags defined as REAL (Float) in L5K files were being incorrectly imported as Digital. Tag types are now mapped correctly, preserving the data type defined in the PLC project. • **ControlLogix import race condition resolved.** A race condition in the ControlLogix driver's tag scanning sequence was causing import failures on specific PLC models. The fix has been validated with affected hardware configurations. • **Byte Swap behavior corrected for ControlLogix.** The Byte Swap modifier now operates correctly on ControlLogix points. A new ForceCIPLittleEndian option has been introduced per Point, giving users explicit per-tag control over whether the driver handles CIP byte ordering automatically or defers to the manually configured Byte Swap modifier.
SmartClient	• **Custom DLLs in WPFControls folder are now distributed via SmartClient.** DLLs placed in the WPFControls folder are now correctly included in SmartClient distribution packages and downloaded to the client machine's WPFControlsCache folder, ensuring custom controls are available without manual intervention. • **Property Watch and Trace Window available via RuntimeStartup.** Property Watch and Trace Window are now accessible through a new "Run Tools" checkbox in RuntimeStartup. The option is disabled by default across all runtime profiles and can be enabled on demand for debugging and troubleshooting sessions.
Online Configuration	• **Online configuration changes no longer disrupt other active modules.** Applying online configuration changes no longer causes interference with other running modules — including device communication channels. This resolves a regression where saving a display change could cause a critical section timeout that stopped command execution across all drivers. • **Tag type changes no longer generate Invalid Token log floods.** Changing a tag type through online configuration no longer triggers a flood of Invalid Token log entries caused by stale binding tokens in connected clients. The client now correctly re-subscribes with updated tokens after a configuration change, eliminating the log noise entirely.