Update 5f - Detailed Release Notes

Complete feature and improvement details for FrameworX 10.1 Update 5f, organized by product area.

Update 5f replaces Update 5e. This page lists the complete content of Update 5f, which includes everything delivered in Update 5e (August 31, 2026). If Update 5e is installed, replace it with Update 5f.

FrameworX 10.1 Update 5f, September 9, 2026

Build: 2113

Alarms, Historian and redundancy

- **Historian data keeps replicating to the standby on a redundant pair whose history database is very large and uses TimescaleDB compression.** Rows still waiting to be replicated when their portion of the database was compressed could no longer be read, so each cycle read the whole table, exceeded the connection time limit, and restarted without draining the backlog. Alarm replication was unaffected, because alarm rows are replicated before compression takes place.
- **A batch carrying a few unresolvable rows no longer stops Tag Historian replication in normalized configuration.** The standby imports the rows it can resolve and reports the ones it cannot, and the active server records which tag index it could not map.
- **The active server's runtime process stays bounded under alarm-history load** on a redundant pair. A `TServer.json` key, `Alarm.HistoryQueryMaxRows` (default 100000), sets a server-side ceiling on how many alarm-history rows a single query may load. A query above the ceiling succeeds and returns the newest rows up to it, with a warning naming the requested count, the applied ceiling, and the configuration key.
- **Redundancy settings are applied when the runtime auto-starts as a Windows service after a computer restart.** The runtime reads the machine's network addresses correctly during early Windows startup, and any remaining network-timing condition is reported in the Windows Event Log instead of being silently ignored. A new optional `/redundancywaittimeout:<seconds>` TStartup argument waits for the network stack on machines where the network comes up slowly (default: no wait).
- **Documented safe automatic-failback practices and `/TimeAutoSwitch` tuning.** During automatic failback, the returning primary receives the history the secondary recorded while it was down, while it waits as standby for `/TimeAutoSwitch`; rows not sent before the switch remain on the secondary until it is active again (`@Server.Redundancy.SwitchToStandby()`).
- **Alarm history queries complete without a database syntax error** on alarm historians reached through the native PostgreSQL, Oracle, and MySQL connectors. Where the query was rejected, the chart kept drawing the data it already held, so the failure was easy to miss, but the alarm markers were missing or out of date and the query failed again on every refresh. The correction covers the Trend Chart, the Drilling Chart, and the Alarm Viewer in both Alarm History and Audit Trail modes, in the rich client and the HTML5 client. *Known limitation on alarm historians reached through an ODBC or OleDb connection, described below.*
- See **Breaking Changes and Deprecations** below for the withdrawal of the handover replication drain barrier.

Alarm historians over ODBC or OleDb

On an alarm historian reached through an ODBC or OleDb connection, every query that separates alarm records from audit trail records is rejected by the database, so the Trend Chart and Drilling Chart show no alarm markers and the Alarm Viewer's Audit Trail view returns nothing. The Alarm Viewer's Alarm History view is affected only when Audit Trail is enabled. With Audit Trail off it works normally, and the combined Alarm History plus Audit Trail view is unaffected.

This covers Microsoft Access, Microsoft Excel and CSV files, and Oracle, MySQL, or Firebird when connected through ODBC or OleDb rather than their native connector. Alarm historians on SQL Server, SQLite, or on PostgreSQL, Oracle, and MySQL through their native connectors are not affected.

HTML5 and web client

- **Faster HTML5 client startup and lower latency for remote clients.** The connection sequence runs its steps in parallel, the browser opens its communication channel while the page loads, and repeated round trips were removed from the startup path.
- **Server responses reach remote clients in a single network round trip**, improving the responsiveness of displays, trends, and reports over remote connections.
- **Upgrades stay compatible with servers and clients from earlier 10.1.5 updates.**
- **Combobox is populated on first load in HTML5.**
- **Icons and images stay in place on the HTML5 client after a reload.**
- **Symbol Rotate animations no longer accumulate memory in the client.**

Scripts

- **Script profiler durations are measured with a high resolution timer.** Short running scripts previously reported a duration of zero on Windows, because the clock in use had a resolution of roughly 16 milliseconds. A method taking a few microseconds is now timed accurately.
- **The reported time covers only the script execution**, and no longer includes the parameter conversion and the writing of the log entry that precede it.
- **Every started entry is followed by exactly one matching terminator**, including when the script raises an error. A Script.Class method that raises an error is reported as failed rather than finished, alongside its SCRIPT EXCEPTION entry, and the started entry for a Script Task no longer shows a meaningless execution time of zero.
- **Method parameter values written into the log entry are limited in length**, and a parameter that cannot be converted to text no longer truncates the rest of the entry.

- *Two notes for anyone enabling DebugMessages.* Parameter values passed to Script.Class methods are written to the trace log as they are, so avoid passing passwords, tokens, or connection strings to a script method while this log level is on. Each entry is written according to the log level in effect at the moment that entry is due, so turning the level off while a script is still running can leave its started entry without a matching terminator. Entries for Script.Class methods are recorded when the method is invoked from a client session; a class method called directly from within another script runs in the same compiled unit and does not produce its own entry.
- **Saving a Script Class no longer freezes the Designer** in solutions where script classes reference each other.

Displays and Symbols

- **Map a Symbol opens the picker for basic type tags**, Digital, Integer, and Text, and not only for tags of a User Type.
- **A tag that already carries a symbol is shown as mapped in the Displays tag panel**, with its preview rendered, and dragging it onto a Display inserts the mapped symbol.
- **A symbol mapped on a User Type is reported as mapped in the Tags list and on the instances of that User Type**, so the Tags list and the Displays tag panel agree.
- **Clearing a mapping from the tag panel takes effect for basic type tags**, and dragging a tag whose mapped symbol can no longer be resolved inserts a default component instead of failing silently.
- **Display parameters used as part of a larger value track live values again.** A parameter combined with a comparison, a second condition, or any surrounding expression previously compiled without complaint and then stayed frozen at runtime. A button configured to stay disabled while a condition holds now disables as configured. Applies to displays opened as pages and as popups. *Known limitation: a dynamic inside a Symbol on a parameterized display is handled by separate rules and still reproduces the original behavior.*
- **Display geometry survives the import and export round trip** across four separate paths.
- **Custom symbols are preserved when migrating a solution from FrameworX 10.1.4 to 10.1.5.**
- **Popup displays imported with a fixed custom position keep that position.**
- **Dynamics Get and Apply keeps the theme binding on transferred colors** (LineColor, FillColor, and others).
- **The Asset Tree control no longer raises an exception in the Rich Client** when Localize is enabled.
- **Colors driven by an Enumeration resolve correctly.**

Unified Namespace, ontology and knowledge graph

- **Ontology import reports what it actually created**, not what it planned to create.
- **The same ontology property produces the same member name across UserTypes.**
- **Ontology import keeps a class as a class** instead of converting it into an Enumeration when its objects carry only a type and a name. Relationships pointing at those objects are preserved.
- **Ontology import resolves a relationship to an object that is not declared in the file** instead of silently storing it as plain text.
- **Knowledge Graph export produces HTML that renders in Firefox.**
- **The exported Markdown knowledge-graph diagram renders instance-of relationships.**
- **The exported knowledge-graph visual refers to UserTypes by their product name**, not “UDTs”.

AI, MCP and automation

- **Re-importing a file or solution over existing content updates the existing objects** instead of silently skipping them.
- **The LinearGauge label, tick, and scale-bar colors are exposed on the JSON / MCP write path**, so gauges authored from automation can match a legacy HMI palette.
- **`TK.PreloadObject` preserves the object name’s letter case.**

Designer and data

- **Import Solution brings Templates and Tags in correctly.** No unprefixed extra Templates, and the tags are not dropped.
- **Dataset Table inserts respect SQL Server DEFAULT values on unmapped columns** instead of overwriting them with NULL.
- **Alphabetical sort is available as soon as the Enumerations, UserTypes, Layouts, Localization, and Units Conversion pages open.**
- **Import Tags Wizard loads the Solution list** on the Solution selection row.

Devices and protocols

- **Modbus master reads device identification data using function code 43** (Read Device Identification).

Runtime and platform

- **Clients that connect while the runtime is still starting come up licensed** on a correctly licensed machine.
- **AutoStartClient detects a Rich Client that is already running** instead of opening another window.
- **A runtime running as a Windows Service no longer starts the Run Tools** (TraceWindow, PropertyWatch) as hidden background processes.
- **Large array and matrix tags populate immediately when a Display opens.**

Mobile client

- **The Settings screen connection fields are readable when the device is in Light mode.**

Security and access

- **OpenID Connect sign in works in solutions running on .NET Framework 4.8.** Sign in previously worked only in multi-platform solutions.

- **A logon is refused when the server's user check fails internally.** A failure part-way through the check was previously reported as a successful check and the client was allowed to connect. The refusal is reported to the client and recorded in the audit trail when remote-connection auditing is enabled. A normal logon is unaffected, and a wrong user name or password still returns the same specific reason as before.
- **Stored credential values can no longer be read or overwritten by the AI tools through the Designer grid cell edit action**, meaning passwords, API keys, tokens, and database logon passwords. Reading a protected column is refused instead of returning the stored value, and writing one is refused outright. Credentials continue to be set through the Designer security screens as before, and ordinary non-credential grid cells are unaffected.
- **A security guard built on a parameterized display disables as configured.** See **Displays and Symbols** above.

Breaking Changes and Deprecations

The handover replication drain barrier has been withdrawn. Alarm and Historian replication behavior around a redundant-pair handover is unchanged from 10.1.5c. The change introduced in Update 5d, which held a handover open until Alarm and Tag Historian replication finished draining, has been withdrawn together with its `SwitchDrainTimeoutSeconds` configuration key. A configuration file that still contains that key is accepted and the key is ignored. Rows recorded on the active node in the moments immediately before a handover may therefore not reach the other node.

On redundant pairs started with `/autoswitch`, use the existing `/TimeAutoSwitch:<seconds>` startup argument (default 60) to give replication time to finish before the restarting Primary reclaims the Active role. Increase it beyond the default for large Solutions, slow networks, or high alarm and data-logging rates.

Lifecycle and Support

Update 5f ships on **September 9, 2026** as a **free upgrade for every FrameworX 10.x customer**, with the platform's standard within-major-version 100% compatibility commitment. **Update 5f replaces Update 5e**, and installations running Update 5e should move to Update 5f. See **Breaking Changes and Deprecations** above for a redundancy behavior change that may require a startup-argument adjustment on affected installations.

Security note

This release contains security-relevant corrections. Upgrading is recommended for all installations.

One item requires action beyond upgrading: if you used the AI or MCP tooling against solutions that hold stored passwords, API keys, or database logon credentials on 10.1.5d or earlier, rotate those credentials.

Customers who need per-item detail, meaning affected versions, severity, and required action, for patch management or supplier-assurance purposes can request it from **support@tatsoft.com**, along with the component manifest (CycloneDX) for the version they are running.
