Runtime Installation Reference

Install and configure runtime components.

Reference **Solution** **Deployment** Runtime | Server | Client Types | Container | Redundancy | Security

---

Runtime Installation (Reference): Deploying FrameworX runtime components for production environments.

Deployment Scenarios
Silent Install and Exit Codes
Solution Startup Configuration
Windows Service Configuration
Startup Parameters Reference
Per-Process Log File Configuration
Linux/Docker Startup
Client Display Considerations

Deploying in production? This page covers runtime installation options and startup parameters in detail. For the end-to-end IT workflow (silent install switches, exit codes, service install, monitoring, update, uninstall, automation patterns), see IT Deployment RunBook.

Deployment Scenarios

Windows Runtime Deployment

For Windows-only solutions targeting .NET Framework 4.8:

- Install the complete FrameworX platform (same as Designer)
- No separate runtime-only installer exists
- Configure to use only runtime components in production
- Disable Designer access if needed for security

Multiplatform Runtime Deployment (.NET 10)

For solutions targeting Linux, Docker, or edge devices, or for web-based clients using WebAssembly:

Option	Components	Size	Use Case
Runtime-Only	TServer.exe, drivers, minimal files	~100 MB	Production edge devices
Full Platform	All files including Designer	~900 MB+	Enable remote ClickOnce Designer
TWebServices-Only	Communication server only	Minimal	Development/testing

---

Silent Install and Exit Codes

The Windows installer is an Inno Setup package. For unattended deployment (SCCM, Intune, Ansible, PowerShell DSC):

```
Start-Process -FilePath ".\FrameworX-Setup.exe" `
  -ArgumentList "/VERYSILENT", "/SUPPRESSMSGBOXES", "/NORESTART", `
                "/LOG=C:\Logs\frameworx-install.log", `
                "/DIR=C:\Program Files\Tatsoft\FrameworX" `
  -Wait -PassThru
```

Replace `FrameworX-Setup.exe` with the name of the installer file you downloaded.

Installer exit codes (Inno Setup standard):

Code	Meaning
0	Success
1	Setup failed to initialize or user aborted
2	User cancelled during preparation
5	Install aborted during copy
3010	Success, reboot required to complete

For full automation patterns including SCCM/Intune packaging and rollback, see IT Deployment RunBook.

Solution Startup Configuration

TStartup.exe Overview

TStartup.exe is the executable that runs FrameworX solutions. It supports various parameters for debugging, redundancy, and advanced configuration.

Basic Startup Methods

Manual Start:

- From SolutionCenter: Select solution Runtime Startup
- From Designer: Runtime Startup Run Startup
- From a Windows shortcut with parameters (see Desktop Shortcut)

Windows Service:

- Runs on computer startup
- No user login required
- Configure at Runtime Startup

Command Line/Batch File:

```
cd "C:\Program Files\Tatsoft\FrameworX\fx-10"
TStartup.exe /Solution:"C:\Solutions\MySolution.dbsln"
```

`C:\Program Files\Tatsoft\FrameworX\fx-10` is the default installation folder. On Windows, `TStartup.exe` in that folder starts both Windows (.NET 4.8) and Multiplatform (.NET 10) solutions. Without `/profile` the Production profile runs; add `/profile:1` (Development), `/profile:2` (Validation) or `/profile:3` (Custom) to run another profile. The value is a number: a name such as `/profile:Development` is read as Production.

Desktop Shortcut

To start a solution from a desktop icon, create a Windows shortcut to `TStartup.exe` and add the solution path to its target:

1. In File Explorer, open the `fx-10` folder of the FrameworX installation. The default is `C:\Program Files\Tatsoft\FrameworX\fx-10`.
2. Right-click `TStartup.exe` and select **Send to Desktop (create shortcut)**. On Windows 11, **Send to** is under **Show more options**.
3. On the desktop, right-click the new shortcut and select **Properties**.
4. On the **Shortcut** tab, replace the **Target** field with:

   ```
   "<InstallFolder>\fx-10\TStartup.exe" /Solution:"<SolutionFolder>\<SolutionName>.dbsln"
   ```

5. Click **OK**. Double-click the shortcut to start the runtime.

Replace `<InstallFolder>` with the folder where FrameworX is installed (default `C:\Program Files\Tatsoft\FrameworX`), and `<SolutionFolder>\<SolutionName>.dbsln` with the full path of the solution file. Keep both paths in quotes, since they usually contain spaces.

The shortcut runs the Production profile. To run another profile, add `/profile:1` (Development), `/profile:2` (Validation) or `/profile:3` (Custom) at the end of the target. Any other parameter from Startup Parameters Reference can be added the same way.

Startup Shortcuts in the User Utilities Folder

`StartRuntime` and `StartRichClient` are not executables. They are `.bat` shortcuts created in the user `Utilities` folder the first time the product runs, both for an installed copy and for a copy started directly from the binaries package:

```
<User>\Documents\FrameworX\Utilities\StartRuntime.bat
<User>\Documents\FrameworX\Utilities\StartRichClient.bat
```

Each shortcut calls the real executable in the product folder, with the full installation path already written into the file, so it runs from any working directory:

Shortcut	Starts	Purpose
`StartRuntime.bat`	`fx-10\TStartup.exe`	Starts the solution runtime on this machine.
`StartRichClient.bat`	`fx-10\TRichClient.exe`	Starts the Rich Client.

The same folder holds the other shortcuts generated for the installed components, for example `StartWebServices.bat`, `SolutionCreator.bat`, `TDatabaseBuilder.bat`, `UpgradeSolution.bat`, `RemoveAllServices.bat`, and the AI listener shortcuts `StartDesignerMCPHttp.bat`, `StartRuntimeMCPHttp.bat` and `StartConsoleMCPHttp.bat`. The exact set depends on the components installed.

> ⚠ There are two folders named `Utilities` and they hold different things. `<User>\Documents\FrameworX\Utilities\` holds the generated shortcuts described here. The product folder `fx-10\Utilities\` holds the stand-alone utilities themselves, `AutoStartClient.exe` and `CertificateConverter.exe`. Command-line tools such as `SolutionCreator.exe` and `TDatabaseBuilder.exe` are in the product folder root. See Installation Folders and Utilities Reference.

Required Parameters

The minimum parameter needed:

```
/Solution:"C:\Solutions\Solution.dbsln"
```

Solution Merging on Startup

The platform supports modular project design by automatically merging multiple project files at runtime based on naming conventions.

How It Works:

When launching a main solution (e.g., `MyProject`), FrameworX automatically merges any projects with names following the pattern:

- Base name + double underscore + suffix + double underscore
- Example: `MyProject__ModuleName__`

Example Structure:

```
MyProject.dbsln                    Main solution
MyProject__DeviceConfig__.dbsln    Device configurations
MyProject__DBTest__.dbsln          Test database settings
MyProject__CustomerA__.dbsln       Customer customizations
```

Runtime Behavior:

- Detects all matching `MyProject__*__.dbsln` files in the same folder
- Merges contents (tags, scripts, displays, connections)
- Creates unified runtime environment
- Maintains modular development structure

Windows Service Configuration

Install the service with **AutoStart** in Runtime Startup (see Runtime Startup Reference, Enabling AutoStart), or from a script with TManageServices CLI.

Service Security Settings

Log On Configuration:

- Default: Local System account
- Custom: Domain user for external resources
- Configure in Windows Services console

Start Type:

- **Automatic (Delayed Start)** is the installer default and the recommended start type. The delay gives the machine time to complete the network link, negotiation and address assignment before the runtime starts.
- Keep the Windows default delay of 120 seconds. It applies to the whole machine and is not a per-service setting.
- On a redundant node this is not optional: the node resolves its Primary or Secondary role from the local addresses of the machine at startup, so a service that starts before the network is ready starts as a standalone node. See Redundancy Reference.

Startup Parameters Reference

Main Parameters

Parameter	Description	Example
`/Solution`	Solution path (required)	`/Solution:"C:\Solutions\Solution.dbsln"`

`/username`	Runtime user	`/username: Administrator`
`/password`	User password	`/password: SecurePass123`
`/wa`	Windows Authentication	`/wa`
`/ip1`	Primary server IP	`/ip1: 192.168.1.100`
`/port1`	Primary server port	`/port1:3101`
`/CloseSplash`	Closes the TStartup splash dialog once the solution has finished starting, instead of leaving it on screen. Set it from the **CloseSplash** checkbox in **Runtime Startup**, next to the Startup Command field, which adds it to the generated command line. Distinct from `/nosplash` on the client, which suppresses the client splash altogether.	`/CloseSplash`

Redundancy Parameters

Parameter	Description	Default
`/redundancy`	Enable redundancy	-
`/autoswitch`	Auto-switch to primary	-
`/timeautoswitch`	Switch delay (seconds)	60
`/ip2`	Secondary server IP	-
`/port2`	Secondary server port	3101
`/connectiontimeout`	Redundancy watchdog timeout, in seconds. Command-line form of the Timeout field of the redundancy dialog.	3
`/connectionretry`	Retry attempts	1
`/redundancywaittimeout`	Seconds to wait for the network stack when the runtime starts as a Windows Service with `/redundancy` and neither configured address is local yet. The startup re-checks every 5 seconds and continues with redundancy enabled as soon as one of the addresses becomes local. With the default of 0 it does not wait. Available from FrameworX 10.1.5e.	0
`/SolutionIPPathName`	Remote solution path	-

Redundancy Example:

```
TStartup.exe /Solution:"C:\Solutions\Solution.dbsln" ^
  /redundancy ^
  /ip1:192.168.1.1 /port1:3101 ^
  /ip2:192.168.1.2 /port2:3101 ^
  /autoswitch /timeautoswitch:60
```

Diagnostic Parameters

Parameter	Description	Output Location
`/SaveLog:3`	Debug logging	`C:\Users\Public\Documents\FrameworX\TraceLogs`
`/AutoRunDiagnostics:30`	Auto diagnostics (minutes)	`C:\ProgramData\FrameworX`
`/LogDeviceWriteStartup`	Device write logging	`C:\ProgramData\FrameworX\Device`

Advanced Parameters

Parameter	Description	Default
`/NumberOfDevicesAtSameTime`	Concurrent device startup	3
`/TimeBetweenModules`	Module start delay (seconds)	0
`/AutoCheckWatchDog`	Auto-restart frozen channels	Disabled

`/viewonly` and `/nocache` are Rich Client parameters, and `/port1WA` is read by TServer. TStartup does not accept them and reports each one as unrecognized.

Unrecognized Parameters

From FrameworX 10.1.6, TStartup reports each launch parameter it does not recognize, instead of ignoring it silently, and the startup continues. A near miss of a valid parameter gets a suggested spelling. A parameter with no resemblance to a valid one is reported without a suggestion. Parameter values do not appear in the message. For example, `/profle:1` produces:

```
TStartup: unrecognized launch parameter '/profle' - ignored. Did you mean '/profile'?
```

TStartup writes these warnings to the console and the TStartup trace window, and on the Multiplatform runtime also to the trace log when you start it with `/SaveLog`. It lists up to 20 unrecognized parameters. SolutionCreator reports unrecognized parameters the same way, in the file given by its `/logfile` parameter, and still creates the solution.

> **Changed in FrameworX 10.1.6**
>
> In 10.1.5, TStartup and SolutionCreator ignore unrecognized launch parameters without a message. After moving to 10.1.6, check the startup output of your shortcuts, batch files and service command lines, and correct or remove each parameter reported as unrecognized. The startup itself is not affected.

Per-Process Log File Configuration

Each runtime process (`TServer`, `TStartup`, `TRunModule`, `TRichClient`) can size-limit its active log file and control retention of daily compressed archives, directly from that process's JSON configuration file (e.g. `TServer.json`).

Configuration

Add a `Logging` section under `appSettings`:

```
{
  "appSettings": {
    "Logging": {
      "MaxFileSizeMB": 10,
      "KeepAllZips": false
    }
  }
}
```

Omitting the `Logging` section entirely preserves prior (legacy) behavior: no size cap, and the daily archive compaction described below is unaffected.

MaxFileSizeMB

Caps the size of the process's active log file, in megabytes. When the limit is reached, the process stops writing to that log file and records a notice ("Log file size limit of <N> MB exceeded; logging to this file is stopped.").

- **Integer values only, minimum effective value 1.** Fractional or sub-1 values (e.g. `0.5`) are truncated to `0`, which disables the size limit, and no warning is logged when this happens. Sub-1 MB caps are not supported.
- A value of `0` or an omitted setting means no limit.

KeepAllZips: Daily Archive Retention

On each day change, the runtime compacts the previous day's per-process log: every non-current log file for that process is compressed into a same-named `.zip` archive, then the original text log is deleted. This daily compaction always runs, independent of `KeepAllZips`. The currently-active log file is never touched.

`KeepAllZips` controls only how many of those daily archives are retained afterward:

Value	Behavior
`true`	All daily `.zip` archives are kept. They accumulate day after day with no automatic pruning. Use when full log history retention is required.
`false`	Only the single most recent `.zip` archive is kept per process (determined by file last-write time); older archives are deleted at each rollover. Use to keep disk usage bounded.

Retention is evaluated independently per process log (`TServer`, `TStartup`, `TRunModule`, `TRichClient` are each pruned separately), and pruning only runs when more than one archive is present, so the newest archive is never at risk. `KeepAllZips` is independent of `MaxFileSizeMB`: the size cap governs the active log within a day, while `KeepAllZips` governs retention of the compressed archives across days.

Linux/Docker Startup

On Linux, TWebServices is the service that starts solutions at boot. Install FrameworX as described in Linux Deployment Reference (TWebServices in the FrameworX root folder, runtime binaries in `fx-10` below it), run TWebServices as a systemd service, and turn on AutoStart for the solution.

systemd Service

Create `/etc/systemd/system/frameworx.service`. Replace `/home/frameworx/FrameworX` with your FrameworX root folder, and check the `dotnet` path with `which dotnet`:

```
[Unit]
Description=FrameworX TWebServices
After=network.target

[Service]
Type=simple
WorkingDirectory=/home/frameworx/FrameworX
ExecStart=/usr/bin/dotnet /home/frameworx/FrameworX/TWebServices.dll
Restart=always

[Install]
WantedBy=multi-user.target
```

Enable and start it:

```
sudo systemctl daemon-reload
sudo systemctl enable --now frameworx.service
```

Then, in Solution Center connected to this computer, turn on **AutoStart** for the solution. TWebServices starts it every time the service starts (see TWebServices Auto-Start).

TWebServices Auto-Start

On Linux and macOS, turning on **AutoStart** for a solution in Solution Center adds a line for it to `AutoStartPrograms.txt` in the MachineSettings folder, and turning it off removes the line. When TWebServices starts, it starts every program listed in that file. Each line is `program;arguments`, for example:

```
dotnet;'/home/frameworx/FrameworX/fx-10/TStartup.dll' /solution:'/home/frameworx/Solutions/MySolution.dbsln'
```

Use AutoStart in Solution Center rather than editing the file by hand. On Windows, AutoStart registers a Windows service for the solution instead.

Docker

For containers, use the FrameworX images and start parameters described in Container Deployment Reference and Docker Deployment Guide.

> **Do not unzip or copy the Windows installation onto Linux**
>
> On Linux there is no `net10.0` subfolder: the runtime binaries sit directly in `fx-10`. A Windows Bin zip extracted on Linux, or a copied Windows install folder, has the wrong layout, and remote device launch and the TWebServices file services fail.

Client Display Considerations

A runtime running as a Windows Service has no user interface. To start clients at boot, see Installation Folders and Utilities Reference, AutoStartClient.exe.

In this section...
