Tatsoft.FrameworX PowerShell Module Reference

Native PowerShell cmdlets for FrameworX — lifting IT deployment from raw `.exe + /silent` calls to one-liner cmdlets with `Get-Help`, parameter validation, pipeline composition, and structured exit-code handling.

Release Notes **IT Deployment RunBook** Tatsoft.FrameworX PowerShell Module Reference

Version 10.1.5+

Overview

`Tatsoft.FrameworX` is a single PowerShell module that wraps three existing FrameworX surfaces:

Surface	Wrapped CLI / endpoint	Audience
Product lifecycle	`FrameworX-Setup.exe`, `unins000.exe` (Inno Setup)	IT / DevOps deploying FrameworX
Service lifecycle	`TManageServices.exe` (per-service install / uninstall verbs)	IT / DevOps managing Windows Services
SolutionCenter API	`https://<host>:10108/api/v1/...` (TWebServices)	Fleet operators, OEM admin tooling, central monitoring

Compatible with **PowerShell 5.1** (Windows PowerShell, ships with Windows) and **PowerShell 7+** (cross-platform). The module is signed by Tatsoft and exports 33 cmdlets.

Install

```
# All-users (typical IT deployment, requires admin):
Copy-Item -Recurse '\\fileshare\Tatsoft.FrameworX' 'C:\Program Files\WindowsPowerShell\Modules\Tatsoft.
FrameworX'
Import-Module Tatsoft.FrameworX
```

The module is also dropped under `<InstallDir>\Tools\PowerShell\Tatsoft.FrameworX\` by future FrameworX installer revisions.

Quick start

Install on a fresh box

```
Install-FrameworX -InstallerPath C:\Setup\FrameworX-Setup.exe -InstallDir 'C:\Program Files\Tatsoft\FrameworX'
Install-FrameworXWebServices
Install-FrameworXRuntime -SolutionPath 'C:\Solutions\Plant1.tproj' -Profile Production
```

Manage solutions remotely

```
Connect-FrameworX -Endpoint https://plant1-srv07:10108 -ServiceAccountToken $env:FX_TOKEN
Get-FrameworXSolution
Start-FrameworXSolution -Name Plant1SCADA -Profile Production
Get-FrameworXSolution | Where-Object status -eq running | Stop-FrameworXSolution
Disconnect-FrameworX
```

Cmdlets

Product lifecycle

Cmdlet	Wraps	Notes

`Install-FrameworX -InstallerPath -InstallDir -LogPath`	`FrameworX-Setup.exe /VERYSILENT /SUPPRESSMSGBOXES /NORESTART /DIR= /LOG=`	Maps Inno exit codes to a structured result. Exit code 3010 = success + reboot required.
`Uninstall-FrameworX [-InstallDir]`	`unins000.exe /VERYSILENT`	Auto-detects install directory from HKLM.

Service lifecycle (machine-local, requires admin)

Cmdlet	Wraps
`Install-FrameworXRuntime -SolutionPath -Profile`	`/installtstartup /solution:"..." /profile:n` — installs `TStartup-<SolutionName>`
`Uninstall-FrameworXRuntime -SolutionName`	`/uninstalltstartup /solution:"..."`
`Install-FrameworXWebServices` / `Uninstall-FrameworXWebServices`	`/installtwebservices`, `/uninstalltwebservices`
`Install-FrameworXSecureGateway` / `Uninstall-FrameworXSecureGateway`	`/installtsecuregateway`, `/uninstalltsecuregateway`
`Install-FrameworXMqttBroker` / `Uninstall-FrameworXMqttBroker`	`/installtmqttbroker`, `/uninstalltmqttbroker`
`Install-FrameworXHardkey` / `Uninstall-FrameworXHardkey`	`/installthardkey`, `/uninstallthardkey`
`Install-FrameworXRuntimeMcpHttp` / `Uninstall-FrameworXRuntimeMcpHttp`	`/installruntimemcphttp`, `/uninstallruntimemcphttp`
`Remove-FrameworXAllServices`	`/removeallservices` — every FrameworX service in one call

Session

Cmdlet	Calls
`Connect-FrameworX -Endpoint -ServiceAccountToken`	(no HTTP — stores JWT and base URL in module state)
`Connect-FrameworX -Endpoint -OidcProvider`	Opens browser to `/api/v1/auth/oidc/login`; captures JWT via localhost listener
`Disconnect-FrameworX`	(no HTTP — clears module state, restores TLS validation)

Solutions

Cmdlet	Calls
`Get-FrameworXSolution`	`GET /api/v1/solutions/`
`Get-FrameworXSolution -Name X`	`GET /api/v1/solutions/X/info` (enriched with `isRunning` + `runningProfiles`)
`Test-FrameworXSolutionRunning -Name X`	`GET /api/v1/solutions/X/isrunning` (returns `[bool]`)
`Start-FrameworXSolution -Name -Profile`	`POST /api/v1/solutions/X/run`
`Stop-FrameworXSolution -Name`	`POST /api/v1/solutions/X/stop`
`Set-FrameworXSolutionAutoStart -Name -AutoStart`	`PUT /api/v1/solutions/X/autostart`

License

Cmdlet	Calls
`Get-FrameworXLicenseInfo`	`GET /api/v1/license/info`
`Get-FrameworXSiteCode`	`GET /api/v1/license/site-code`

`Set-FrameworXLicenseKey -LicenseKey`	`PUT /api/v1/license/key`
`Invoke-FrameworXLicenseActivation -ActivationCode [-SerialNumber] [-SiteCode] [-Upgrade]`	`POST /api/v1/license/activate`
`Invoke-FrameworXLicenseDeactivation -ActivationCode`	`POST /api/v1/license/deactivate`

Machine settings

Cmdlet	Calls
`Get-FrameworXMachineSetting`	`GET /api/v1/machine/settings` (lists the allowlist)
`Get-FrameworXMachineSetting -Filename`	`GET /api/v1/machine/settings/{filename}`
`Set-FrameworXMachineSetting -Filename -Content`	`PUT /api/v1/machine/settings/{filename}` (requires `installation:admin`)

Server

Cmdlet	Calls
`Get-FrameworXServerInfo`	`GET /api/v1/server/info`
`Get-FrameworXConnection`	`GET /api/v1/server/connections` (requires `installation:admin`)
`Disconnect-FrameworXConnection -Guid`	`DELETE /api/v1/server/connections/{guid}` (requires `installation:admin`)

Discovery

Cmdlet	Calls
`Get-FrameworXOpenApi [-OutFile]`	`GET /api/v1/openapi` (anonymous, bypasses the activation gate)

Authentication

The SolutionCenter API requires a JWT Bearer token on every call (RFC 9068 `at+jwt`). Two issuance paths:

- **Service-account JWT** — pre-issued by your OEM admin tool, dropped on the box as a config file or held in an environment variable. Pass via `Connect-FrameworX -ServiceAccountToken <jwt>`.
- **OIDC interactive** — `Connect-FrameworX -OidcProvider <name>` opens the default browser to the configured IdP via `/api/v1/auth/oidc/login`, then captures the JWT redirect via a localhost `HttpListener`. The named provider must be configured in `MachineSettings/SolutionCenterApi-OidcProviders.json` on the target installation.

Scopes (assigned at issuance time): `installation:read`, `installation:control`, `installation:files`, `installation:license`, `installation:machine`, `installation:admin`. Scopes do not subsume each other.

Activation gate

The SolutionCenter API ships **default-OFF in 10.1.5 GA**. API cmdlets surface `503 service-unavailable` until the operator sets `SolutionCenterApi.Enabled = true` in TWebServices machine settings AND the running build is `>= MinSafeBuildForSolutionCenterApi`. Service-lifecycle cmdlets (`Install-FrameworX*`) are unaffected by the gate. `Get-FrameworXOpenApi` is anonymous and bypasses the gate.

TLS notes

- **PowerShell 7+**: `Connect-FrameworX -SkipCertificateCheck` is honored per-call (`Invoke-RestMethod -SkipCertificateCheck`). Production deployments should use a properly-signed certificate.
- **PowerShell 5.1**: `-SkipCertificateCheck` installs a process-wide `ServicePointManager.ServerCertificateValidationCallback`, restored when `Disconnect-FrameworX` runs. The module also force-enables TLS 1.2 (the PS 5.1 default rejects modern endpoints).

Error handling

API calls failing with non-2xx responses raise terminating errors of id `FrameworX.Api.HttpError`. The error message includes the RFC 7807 Problem Details payload from the server. Catch with `try { ... } catch { ... }`:

```
try {
    Get-FrameworXLicenseInfo
}
catch {
    Write-Warning "License query failed: $($_.Exception.Message)"
}
```

CLI cmdlets (`Install-*` / `Uninstall-*`) raise terminating errors when `TManageServices.exe` reports a non-zero exit code (= number of failed operations) or when the Inno Setup installer reports a failure exit code.

Out of scope (v1.0 of this module)

- File operations (`/api/v1/files/...`) — chunked upload / download, exists, delete-solution. Wrapped in v1.1.
- Visual dashboard UI — tracked in a separate Tatsoft v2 work item.
- MSI distribution package.
- Bulk fan-out (managing many machines from a single shell).

Related

- IT Deployment RunBook — the broader IT Ops guide.
- SolutionCenter API Reference — full API spec, OpenAPI doc, scope model.
