

Runtime MCP Reference

Live-runtime MCP surface for FrameworkX. Read tags, alarms, historian, and UNS state from a running TServer.

[AI Integration](#) Runtime MCP Reference

New in 10.1.5. RuntimeMCP gains an HTTP transport on port 10120, a Windows Service wrapper, and target rebinding by (solution, profile) for automated testing workflows that switch between Execution Profiles without restarting the MCP process.

RuntimeMCP is the MCP server that speaks to a **running** FrameworkX runtime. Unlike DesignerMCP (design-time, live .dbs1n in an IDE) and ConsoleMCP (design-time, file-based JSON workspace), RuntimeMCP targets **runtime state** — live tag values, active alarms, historical queries, and the UNS namespace as the runtime sees it. Use RuntimeMCP when AI-driven workflows need to observe or test a running solution, not author one.

Setup steps are covered in [Claude Code MCP Setup](#) and [MCP and Claude Setup](#). Server-surface overview lives in [MCP SDK Reference](#). Execution Profile details are in [Runtime Execution Profiles Reference](#).

[Vocabulary vs. DesignerMCP and ConsoleMCP](#)
[Server Variants](#)
[Connection Identity — \(solution, profile\)](#)
[Startup Modes \(HTTP only\)](#)
[Tool Catalog](#)
[runtime_set_target](#)
[MCP Client Registration](#)
[Error Codes](#)
[Version History](#)

Vocabulary vs. DesignerMCP and ConsoleMCP

RuntimeMCP tools carry a runtime_ prefix so the mode is always obvious in a transcript. The design-time tools in DesignerMCP and ConsoleMCP work on configuration (the solution); the runtime tools work on state (the running process).

Design-time (Designer / Console)	Runtime (this server)	Object of action
search_uns	runtime_search_uns	Configured UserTypes / AssetTree vs. live UNS instances.
get_objects	runtime_get_object_context	Configuration row vs. live object with inheritance and cross-references.
browse_object_model	runtime_browse_uns	Namespace schema vs. live namespace walk.
(no equivalent — values are runtime-only)	runtime_get_value, runtime_get_tag_history	Live and historical tag values.
(no equivalent — alarms are runtime-only)	runtime_get_active_alarms, runtime_query_alarm_history	Live alarm state and history.

The runtime_ prefix is load-bearing — do not unify names with DesignerMCP. A tool name is the agent's strongest cue about which world it is talking to.

Server Variants

RuntimeMCP ships as three executables, each tailored to a deployment shape.

Server	Transport	Target framework	Use case
RuntimeMCP	stdio	.NET 10	Per-session subprocess, spawned by Claude Desktop or Claude Code. One runtime per client session.
RuntimeMCPHttp	HTTP (SSE) on port 10120	.NET 10	Long-running daemon. Supports target rebinding and multi-runtime setups. Required for any workflow that switches targets without restarting the MCP process.
RuntimeMCPHttpService	N/A (launches subprocess)	.NET Framework 4.8	Windows Service Control Manager wrapper. Installs, starts, and stops RuntimeMCPHttp as a system service. The service itself hosts no tools — it spawns dotnet RuntimeMCPHttp.dll /service as a child process and forwards lifecycle events.

All three share the same tool surface. The HTTP variants add bearer-token authentication through the configured API key in RuntimeMCPHttp.json. Use a reverse proxy if you expose the HTTP surface beyond a trusted subnet.

Connection Identity — (solution, profile)

A FrameworkX runtime is identified by **which solution** it is running and **which Execution Profile** it is running under. The same `.dbsIn` can run concurrently under multiple profiles (Production, Development, Validation, Custom), each bound to its own TCP port via the solution's `RuntimeExecutionProfiles` configuration. Port is a consequence of the active profile, not an identity; editing profile port offsets in the solution changes the port without changing what solution or profile is running.

RuntimeMCP therefore accepts `solution` and `profile` as the connection coordinates. Port is looked up at bind time from InfoServer's registry of running TServers and returned to the client in the bind response for observability.

Profile values:

Integer	Name	Aliases accepted
0	Production	prod, production
1	Development	dev, development
2	Validation	val, validation
3	Custom	custom

Aliases are case-insensitive. Responses always echo both the canonical integer and the full profile name.

Startup Modes (HTTP only)

RuntimeMCPHttp supports two startup modes. The stdio RuntimeMCP binary is always process-per-session and retargeting it means restarting the subprocess — modes do not apply there.

Retargetable (default)

Accepts `runtime_set_target(solution, profile)` at runtime to switch the bound TServer without restarting the process. Designed for automated tests that drive many targets from a single long-running MCP instance. The initial target can be supplied via `/solution + /profile` startup args or deferred until the first `runtime_set_target` call — tool calls before an initial target return `ERR_NO_TARGET`.

One Claude Code session per retargetable instance. Retarget state is held in the server process and shared across all connected clients — if two sessions hit the same retargetable instance, the second session's `runtime_set_target` silently retargets the first session's tools. For multi-session setups, run multiple RuntimeMCPHttp instances on distinct ports in pinned mode.

Pinned

Bound at startup via required `/solution + /profile` args, immutable for the process lifetime. `runtime_set_target` is hidden from the MCP manifest in this mode — the tool is not advertised to clients, so agents cannot attempt to retarget. Designed for A/B-style test setups: launch N RuntimeMCPHttp instances on N distinct listen ports, each pinned to its own `(solution, profile)`, and register each as a separate MCP client entry.

Startup argv

```
RuntimeMCPHttp.exe [/mode:retargetable|pinned]
                  [/solution:"<name>"] [/profile:0-3]
                  [/listen:<port>]      # default 10120
```

`/mode` defaults to `retargetable` if omitted. In pinned mode both `/solution` and `/profile` are required; missing either is a startup error.

Tool Catalog

Category	Tools
Tag read & browse	<code>runtime_get_value</code> , <code>runtime_browse_uns</code>
Semantic search	<code>runtime_search_uns</code> , <code>runtime_get_object_context</code> , <code>runtime_find_by_iri</code> , <code>runtime_list_by_type</code>
Alarms	<code>runtime_get_active_alarms</code> , <code>runtime_query_alarm_history</code>
Historian	<code>runtime_get_tag_history</code>
Runtime info	<code>runtime_get_info</code>

Target control (HTTP retargetable only)	runtime_set_target
---	--------------------

Note. RuntimeMCP does NOT expose any write to solution configuration — no `write_objects`, no `delete_objects`, no `rename_objects`. Configuration writes belong to DesignerMCP or ConsoleMCP. RuntimeMCP writes are limited to tag values, via tools that operate on runtime state only.

runtime_set_target

Rebinds a retargetable RuntimeMCPHttp instance to a different running TServer, identified by (solution, profile). Disconnects from the current target, resolves the new target through InfoServer, reconnects, and invalidates the in-process search cache. Atomic from the caller's perspective — on failure the previous target is left intact.

```
runtime_set_target(
  solution="SolarPanels MCP Demo",
  profile=1          # or "dev"
)
```

Parameter	Required	Notes
solution	Yes	Solution name (case-sensitive) as reported by InfoServer for a running TServer.
profile	Yes	Integer 0–3 or case-insensitive alias (see Connection Identity above).

Response includes the new bound target and, if the server was previously bound, the target it was on before:

Error rendering macro 'code': Invalid value specified for parameter 'com.atlassian.confluence.ext.code.render.InvalidValueException'

```
{
  "solution": "SolarPanels MCP Demo",
  "profile": 1,
  "profileName": "Development",
  "host": "localhost",
  "port": 3201,
  "connected": true,
  "previousTarget": {
    "solution": "Industrial Ontology Demo",
    "profile": 0,
    "profileName": "Production",
    "host": "localhost",
    "port": 3101
  }
}
```

In 10.1.5, all resolved targets use `host = "localhost"`; remote TServer binding is not supported. The field is returned for forward compatibility.

MCP Client Registration

For MCP client configuration (Claude Desktop, Claude Code, VS Code Copilot), register each RuntimeMCPHttp instance as its own HTTP MCP entry. Name the entries semantically — `runtime_{solution}_{profile}` for pinned instances; `runtime` or `runtime_retargetable` for a retargetable instance. Never encode the port in the entry name: ports can change when profile offsets are edited in the solution, and a port-named entry becomes a lie the moment that happens.

Example: side-by-side Dev vs. Production (pinned)

```
# Launch two RuntimeMCPHttp instances pinned to different profiles
RuntimeMCPHttp.exe /mode:pinned /solution:"SolarPanels" /profile:0 /listen:10120
RuntimeMCPHttp.exe /mode:pinned /solution:"SolarPanels" /profile:1 /listen:10121

# Register in the MCP client as two separate entries:
# runtime_solarpanels_prod -> http://localhost:10120
# runtime_solarpanels_dev  -> http://localhost:10121
```

Tools from each entry carry that entry's name as a prefix in the agent's transcript, so `runtime_solarpanels_dev.runtime_search_uns` is unambiguously the Development runtime, and side-by-side comparisons are legible.

Example: single retargetable instance for automated tests

```
# Launch one RuntimeMCPHttp in retargetable mode, no initial target
RuntimeMCPHttp.exe /mode:retargetable /listen:10120

# Register as a single entry: runtime_retargetable -> http://localhost:10120
# Scripted tests call runtime_set_target(solution, profile) before each scenario.
```

Error Codes

Code	Where	Meaning
ERR_NO_TARGET	Any runtime_* tool	Retargetable instance has not been bound yet. Call <code>runtime_set_target</code> first.
ERR_PINNED	<code>runtime_set_target</code>	Instance is in pinned mode and does not accept retargeting. The pinned target is reported in the error message. Run a separate instance for the desired target.
ERR_SOLUTION_NOT_RUNNING	<code>runtime_set_target</code>	InfoServer has no record of solution currently running under any profile.
ERR_PROFILE_NOT_RUNNING	<code>runtime_set_target</code>	Solution is running but not under the requested profile.
ERR_BIND_FAILED	<code>runtime_set_target</code>	Connection to the resolved (host, port) failed. The previous target (if any) is left intact.

Version History

FrameworkX	Notes
10.1.4	Initial ship of RuntimeMCP (stdio) with the runtime tool surface.
10.1.5	HTTP variant (RuntimeMCPHttp) on port 10120, Windows Service wrapper (RuntimeMCPHttpService), target rebinding via (<code>solution</code> , <code>profile</code>), <code>retargetable</code> and pinned startup modes, <code>runtime_set_target</code> tool, and the <code>runtime_get_object_context</code> / <code>runtime_find_by_iri</code> / <code>runtime_list_by_type</code> search tools.

In this section...