AI Runtime Developer Reference

Technical reference for building MCP tools that access FrameworX runtime data.
Reference MCP for Runtime Developer Reference

Overview

This document provides technical details for developers building MCP tools that interact with FrameworX runtime. It covers data access APIs, path syntax, available namespaces, and tool design guidelines.

Overview
Path Syntax
Tag Properties
Server Namespace Properties
Alarm Namespace Properties
Alarm History Database
Quality Codes
MCP Tool Design Guidelines
Built-in MCP Tools Reference
See Also

Core APIs

GetCurrentValue

Primary API for reading any value from the runtime.

```
bool success = TMCPServer.DataAccess.GetCurrentValue(
    string path,           // Full path to object
    out object value,      // Returned value
    out int quality,       // 0=Bad, 192=Good
    out DateTime timestamp // Last update time
);
```

This API accepts any namespace path:

Path Example	Returns
`Tag.Temperature1`	Tag current value
`Tag.Temperature1.Description`	Tag description string
`Tag.Temperature1.Quality`	Quality code
`Tag.Pump1.Speed`	UserType member value
`Server.SystemMonitor.CPUUsage`	System CPU percentage
`Alarm.TotalCount`	Active alarm count
`Device.Channel.Modbus1.Status`	Channel connection status

GetChildrenElements

Returns children of any path as a DataTable.

```
DataTable children = TK.GetChildrenElements(string path);
```

Behavior depends on path type:

Path Type	Example	Returns
Folder	`Tag.Plant1/Line2`	Tag names in that folder
Tag	`Tag.Temperature1`	Tag properties (Value, Quality, Description, etc.)
UserType Tag	`Tag.Pump1`	Member names (Speed, Status, Fault, etc.)
Namespace	`Server`	Namespace members
Namespace Object	`Alarm.Group`	All alarm group names

GetTagHistorian

Query time-series data from the Historian.

```
DataTable history = TK.GetTagHistorian(
    string tagName,
    DateTimeOffset startTime,
    TimeSpan duration
);
```

Returns columns: Timestamp, Value, Quality

Path Syntax

Separators

Separator	Purpose	Example
`/`	Hierarchy nodes (folders)	`Tag.Plant1/Line2/Tank1`
`.`	Member access	`Tag.Pump1.Speed`, `Server.DateTimeInfo.Day`

Path Structure

```
Tag.Plant1/Line2/Temperature1.Value
?   ?      ?     ?            ??? Property (member access with .)
?   ?      ?     ??? Tag name
?   ?      ??? Subfolder (hierarchy with /)
?   ??? Root folder (hierarchy with /)
??? Namespace
```

Path Examples

Type	Path	Description
Root tag	`Tag.Temperature1`	Tag at root level
Nested tag	`Tag.Plant1/Line2/Pressure`	Tag in folder hierarchy
Tag property	`Tag.Temperature1.Description`	Tag metadata
UserType member	`Tag.Motor1.Speed`	Member of UserType-based tag
UserType member property	`Tag.Motor1.Speed.Quality`	Quality of UserType member
Server property	`Server.ComputerName`	Server namespace
Nested server property	`Server.SystemMonitor.CPUUsage`	Nested namespace object
Alarm property	`Alarm.TotalCount`	Alarm module property
Alarm group	`Alarm.Group.Critical.TotalCount`	Specific alarm group
Device status	`Device.Channel.Modbus1.Status`	Device channel status

Tag Properties

Properties accessible via `Tag.{name}.{property}`:

Property	Type	Description
`Value`	varies	Current value
`Quality`	int	OPC quality (0=Bad, 192=Good)
`Timestamp`	DateTime	Last update time
`Description`	string	Human-readable description
`Units`	string	Engineering units (e.g., "°C", "PSI")
`EngUnits`	string	Alternative engineering units property
`Min`	varies	Minimum valid value

Max	varies	Maximum valid value
Type	int	Data type ID (see Tag Types in Object Model Quick Reference)
Category	string	User-defined classification

Server Namespace Properties

Server.SystemMonitor.*

Property	Type	Description
CPUUsage	double	System CPU percentage
CPUName	string	CPU model name
CPUClock	string	CPU clock speed in GHz (formatted text, e.g. "3.40")
MemoryUsage	double	System memory percentage
AvailableRAM	double	Available physical RAM (GB)
TotalRAM	double	Total physical RAM (GB)
DiskSpace	double	Total free disk space (GB), summed across local fixed drives
TServerCPUUsage	double	CPU percentage of the FrameworX runtime server (TServer.exe) process. Note: this measures TServer.exe, not the TStartup.exe launcher process.
TServerMemoryMB	double	Private working-set memory (MB) of the FrameworX runtime server (TServer.exe) process. On Windows the value matches Task Manager's Processes-tab "Memory (private working set)" column for TServer.exe — identical metric on both .NET 4.8 and .NET 10 Windows builds (perfcounter "Working Set - Private", with fallback to total working set if the perfcounter is unavailable). **Not** the TStartup.exe launcher process — TStartup spawns TServer at boot and remains lightweight afterward, so its Task Manager memory row will read significantly lower than this value.
Uptime	TimeSpan	Time since solution started
StartedDateTime	DateTime	When solution started
NumberOfTagsDefined	int	Total tags in solution
NumberOfTagsLoaded	int	Tags currently in memory
TagUpdatesPerSecond	double	Data throughput
TotalDevicePoints	int	Communication points

Server.*

Property	Type	Description
ComputerName	string	Server hostname
ComputerIP	string	Server IP address
HttpAddress	string	Web server address
OSVersion	string	Operating system
IsRunningAsService	bool	Running as Windows service
IsRunningOnDocker	bool	Running in container

Server.Redundancy.*

Property	Type	Description
`IsRedundancyEnabled`	bool	HA configuration active
`IsPrimary`	bool	This server is primary
`IsSecondary`	bool	This server is secondary
`PrimaryIP`	string	Primary server IP
`SecondaryIP`	string	Secondary server IP

Alarm Namespace Properties

Alarm.*

Property	Type	Description
`TotalCount`	int	Total active alarms
`UnAckCount`	int	Unacknowledged alarm count
`ActiveUnAckCount`	int	Active and unacknowledged
`QueryActive.Table`	DataTable	Full active alarm table

Alarm.Group.{name}.*

Property	Type	Description
`TotalCount`	int	Active alarms in group
`UnAckCount`	int	Unacknowledged in group
`Disable`	bool	Group disabled flag
`Description`	string	Group description

Alarm History Database

Historical alarm records are stored in `Dataset.DB.AlarmHistorian`.

Property	Value
Database	`Dataset.DB.AlarmHistorian`
Table Name	`AlarmHistory`
Access	SQL queries via Dataset module

Common columns: TagName, Message, ActiveTime, AckTime, NormTime, UserName, Priority, Area, GroupName

---

Quality Codes

Standard OPC quality values:

Value	Meaning	Description
0	Bad	Value is not reliable
192	Good	Value is valid and current

MCP Tool Design Guidelines

Tool Naming Conventions

Pattern	Example	Use Case
Get + Noun	`GetValue`, `GetChildren`	Simple retrieval
Get + Adjective + Noun	`GetTagHistory`, `GetActiveAlarms`	Qualified retrieval
Query + Source	`QueryAlarmDatabase`	SQL-based queries
Search + Target	`SearchTags`	Pattern-based discovery

Description Best Practices

Effective tool descriptions help AI assistants choose the right tool:

1. **First sentence:** State the purpose clearly
2. **Returns:** Describe what data is returned
3. **Use for:** List appropriate use cases
4. **Do NOT use:** Clarify when other tools are better (prevents misuse)
5. **Examples:** Provide sample input values

Template:

```
"[What it does in one sentence]. " +
"Returns [what is returned]. " +
"Use for: [use case 1], [use case 2], [use case 3]. " +
"[When NOT to use, if applicable]. " +
"Examples: [example1], [example2], [example3]."
```

Parameter Description Best Practices

1. State the expected format (e.g., "ISO 8601 format")
2. Provide concrete examples
3. Explain path syntax when relevant

Example:

```
[Description("Tag name (e.g., 'Tag.Temperature1', 'Tag.Plant1/Line2/Pressure')")]
string name
```

Built-in MCP Tools Reference

GetValue

Reads current value from any namespace path.

Property	Value
Parameters	`name` (string) — full path to tag or object
Returns	JSON with Name, Value, Quality, Timestamp
Use for	Process values, system properties, alarm counts, device status

GetTagHistory

Queries time-series data from Historian.

Property	Value
Parameters	`name`, `startDate`, `finishDate`
Returns	JSON table with Timestamp, Value, Quality columns
Use for	Trend analysis, historical comparisons

GetActiveAlarms

Returns currently active alarms.

Property	Value
Parameters	None
Returns	JSON table of active alarm records
Use for	Alarm status inquiries, problem investigation

GetChildren

Browses namespace structure.

Property	Value
Parameters	`path` (string) — path to browse
Returns	JSON table of children elements
Use for	Discovering available tags, understanding data structure

GetTagMetadata

Returns tag metadata without current value.

Property	Value
Parameters	`name` (string) — tag name
Returns	JSON with Description, Units, Min, Max, Type, Category
Use for	Understanding tag purpose before reading value

See Also

- Object Model Quick Reference — Complete object reference
- Built-in Namespaces Reference — All namespace properties
- AI Runtime Connector — Configuration and setup
- AI Runtime Tutorial — Step-by-step guide