


UNS Asset Tree Reference (10.1.5 draft)

 **New for 10.1.5 (draft preview).** This is a preview of how the parent page will look in FrameworkX 10.1.5 (planned April 2026). The parent page is the current 10.1.4 version. Content under review.

Organize your assets in a hierarchical tree.

[Reference](#) [Modules](#) [UNS UI](#) [Asset](#) | [Tags](#) | [UserTypes](#) | [Enumerations](#) | [Services](#) | [Monitor](#)

UNS Asset Tree (Reference): The Asset Tree provides:

- Hierarchical data organization
- Visual tag management
- TagProvider integration
- Runtime value monitoring
- Drag-and-drop organization
- Dynamic folder linking

The Asset Tree serves as the central hub for organizing the Unified Namespace, creating a logical structure that mirrors your physical assets or functional areas.

[Key Concepts](#)
[Ontology metadata on folders \(10.1.5 note\)](#)
[AI / MCP page action \(10.1.5+\)](#)
[User Interface](#)
[Configuration Workflow](#)
[Working with Elements](#)
[Tag Access Methods](#)
[TagProvider Integration](#)
[Runtime Features](#)
[Asset Folder Properties](#)
[Best Practices](#)
[Example Structures](#)
[Troubleshooting](#)

Key Concepts

- **Asset Tree:** Hierarchical representation of real-time data variables
- **Asset Folder:** Container nodes for organizing tags and sub-folders
- **RootTags Folder:** System folder containing unorganized tags
- **Linked Folder:** Folder dynamically connected to TagProvider data
- **Element:** Any item in the tree (folder or tag)

Ontology metadata on folders (10.1.5 note)

Asset Tree folders did NOT gain ontology columns in 10.1.5. All ontology metadata (`DisplayText`, `Labels`, `SourceIri`, `BaseUserType`, `Attributes`) lives on the typed Tag, not on the containing folder. When an OWL individual imports with typed members or a class IRI, it becomes a Tag named `<folderPath>/<entityName>/Attr`. The folder structure auto-creates from the slashes but carries no ontology properties of its own.

On export, the `/Attr` suffix is stripped so each OWL entity IRI reflects the entity's identity, not its FX storage leaf. Folders emit containment edges only (`hasChild` by default, or the policy's ordered `ContainmentPredicates` by depth).

See [Industrial Ontology Integration How-to](#) for the full model.

AI / MCP page action (10.1.5+)

When Designer is navigated to the Asset Tree page, the following action is exposed via the `designer_action` MCP tool:

- `designer_action("relationship_graph")`. Mirrors the **Open Visual Graph** toolbar button on the sibling `UserTypes` page. Generates an interactive HTML view (pan, zoom, filter via `cytoscape.js`) and opens it in the default browser. A Mermaid Markdown side-car lands next to the HTML in the Exchange folder under `Visualizations/` for diff and source-control use.

The action is listed in the `tabActions` array returned by `get_state()` only when the Asset Tree tab is active. See [Generate a visual report of your UNS](#) for the broader visual-report reference.

User Interface

Layout

- **Left Panel:** Tree structure with toolbar
- **Right Panel:** Details of selected node and children

Toolbar Actions

Icon	Action	Description
1	Reload Tree	Refresh structure
2	New AssetFolder	Create container
3	New Tag	Add tag to tree
4	New TagProvider	Create connection
5	Insert Provider Data	Link external data
6	Edit Element	Modify properties
7	Rename	Change element name
8	Create Template	Generate DataTemplate
9	Collapse All	Close all folders

Configuration Workflow

Creating Folders

1. Navigate to **Unified Namespace Asset Tree**
2. Options to create:
 - Click **New AssetFolder** toolbar icon
 - Right-click existing folder
 - Right-click root level

Adding Tags

1. Select target folder
2. Options to add:
 - Click **New Tag** toolbar icon
 - Right-click folder New Tag
 - Use main toolbar New Tag button
3. Configure tag properties
4. Tag appears in selected folder

Tags created without selecting a folder are placed in the RootTags folder. You cannot create sub-folders within RootTags.

Working with Elements

Common Operations

Moving Tags:

- Drag from source to destination folder
- Cut/paste using context menu
- Maintains all tag configurations

Copying Elements:

- Copy entire folders or individual tags
- Preserves alarm, historian, device settings
- Paste to any valid location

Renaming:

- Right-click Rename
- Updates all references automatically
- Maintains data connections

Deleting:

- Right-click Delete
 - Removes tag and configurations
 - Cannot be undone
-

Tag Access Methods

Direct Tag Syntax

```
// Full path including folders
@Tag.Areal/Line1/Temperature
@Tag.Plant/Boiler1/Pressure
```

Asset() Function

```
// Dynamic path resolution
Asset("Areal/Line1/Temperature")

// Relative path
Asset(@Client.Context.AssetPath + "/Temperature")

// TagProvider access
Asset("/OPCServer/Device1/Status")
```

TagProvider Integration

Linking External Data

1. Create TagProvider Connection
2. Folder auto-created in Asset Tree
3. Right-click folder **Insert from TagProvider**
4. Select data to import
5. Browse real-time values

Customizing Linked Folders

Right-click linked folder **Edit Asset Folder**:

- **Alias**: Custom display name
- **Visibility**: Control tag exposure
- **Access**: Read/Write permissions

Runtime Features

Value Monitoring

When solution running and Designer connected:

- Values display in right panel
- Real-time updates
- Quality indicators
- Timestamp information

Display Integration

Add AssetTree Control to displays:

1. Go to **Displays Draw**
2. Find **AssetTree** under Modules
3. Drop onto display
4. Configure properties

Asset Folder Properties

Property	Description	Options
----------	-------------	---------

Name	Folder identifier	Any valid name
Alias	Display name override	Custom text
Visibility	External access control	Private/Protected/Public
LinkedTo	TagProvider connection	Connection name
Path	Full tree path	Auto-generated

Best Practices

- 1. Plan hierarchy** - Mirror physical/logical structure
 - 2. Use meaningful names** - Clear identification
 - 3. Group related tags** - Logical organization
 - 4. Limit depth** - 3-5 levels maximum
 - 5. Document folders** - Add descriptions
 - 6. Regular cleanup** - Remove unused elements
 - 7. Consistent naming** - Follow conventions
-

Example Structures

By Location

```
/Plant
  /Area1
    /Line1
      /Motor1
      /Sensor1
    /Line2
      /Motor2
      /Sensor2
  /Area2
    /Tank1
    /Pump1
```

By Function

```
/Production
  /Mixing
    /Ingredients
    /Temperature
  /Packaging
    /Counter
    /Weight
/Utilities
  /Power
  /Water
```

By Equipment

```
/Motors
  /Motor001
  /Motor002
/Pumps
  /Pump001
  /Pump002
/Valves
  /Valve001
  /Valve002
```

Troubleshooting

Tree not updating:

- Click Refresh button
- Check runtime connection
- Verify tag creation
- Review folder permissions

Cannot move tags:

- Check source/destination validity
- Verify not moving to RootTags subfolder
- Confirm user permissions
- Check if tag is locked

TagProvider data missing:

- Verify connection active
- Refresh tree structure
- Check provider configuration
- Review access rights

Display not showing tree:

- Confirm AssetTree control added
- Check control bindings
- Verify runtime active
- Review display security