UNS Enumerations Reference

Standardize tag values with enumerations.

Reference **Modules** **UNS** **UI** Asset | Tags | UserTypes | Enumerations | Services | Monitor

---

UNS Enumerations (Reference) associate numeric tag values with descriptive text strings, enabling meaningful data presentation by displaying text instead of numbers in HMI displays.

Enumerations provide:

- Value-to-text mapping
- Descriptive state display
- Color associations
- Multilingual support
- Reusable definitions
- Consistent terminology

Enumerations transform raw numeric values into human-readable states like "Running/Stopped" or "Open/Closed".

Creating Enumerations
Configuration Properties
Using Enumerations
Common Enumeration Sets
Electrical System Examples
Multi-State Enumerations
Color Configuration
Best Practices
Display Integration
Troubleshooting
Advanced Usage

Creating Enumerations

1. Navigate to **Unified Namespace Enumerations**
2. Click **Insert New** button on toolbar
3. Enter enumeration name
4. Define value mappings:
    - **Value**: Numeric value
    - **Display Text**: Associated text
    - **ActiveColor**: Optional color
    - **InactiveColor**: Optional color

Configuration Properties

Property	Description	Required
Name	Enumeration set identifier	Yes
Value	Numeric value to map	Yes
Display Text	Text to show for value	Yes
Match	Matching criteria/condition	No
ActiveColor	Color when value matches	No
InactiveColor	Color when value doesn't match	No
Category	Classification group	No
Description	Documentation text	No

Using Enumerations

Assign to Tag

1. Go to **Unified Namespace Tags**
2. Show **Enumeration** column (right-click headers)
3. Select enumeration for tag
4. Tag now displays text instead of number

The enumeration is assigned **per instance**: each tag carries its own, so one UserType (UDT) can be reused across screens with a different enumeration on each instance — including a different enumeration on an individual member of a UDT instance, set from the **Tag Properties** dialog via **Edit Properties of member**. See UNS Tags Reference Per-instance enumeration override.

Access in Runtime

csharp

```
// Get numeric value
int value = @Tag.Motor1.Value;  // Returns: 1

// Get enumeration text
string text = @Tag.Motor1.ValueAsString;  // Returns: "Running"
```

Common Enumeration Sets

Basic States

OnOff

Value	Text	Use Case
0	Off	Equipment stopped
1	On	Equipment running

OpenClosed

Value	Text	Use Case
0	Open	Valve/breaker open
1	Closed	Valve/breaker closed

ManualAuto

Value	Text	Use Case
0	Manual	Local control
1	Auto	Automatic control

Electrical System Examples

Circuit States

EnumOpenedClosed

- 0 "Opened" (Circuit open)
- 1 "Closed" (Circuit closed)

Control Modes

EnumLocalRemote

- 0 "Local" (Local control)
- 1 "Remote" (Remote control)

Power States

EnumPower

- 0 "Unpowered" (No power)
- 1 "Powered" (Power present)

Connection Types

EnumTypeConnector

- 0 "Delta" (Delta connection)
- 1 "Wye" (Wye connection)
- 2 "WyeGrounded" (Grounded wye)

Multi-State Enumerations

Transfer Switch

EnumNormIntermTransf

Value	Text	Description
0	Normal	Standard operation
1	Intermediary	Transition state
2	Transferred	Alternate position

Priority Levels

EnumTicketColor

Value	Text	Color	Priority
0	Red	`#FF0000`	High
1	Green	`#00FF00`	Low

Color Configuration

Associate colors with enumeration values:

```
Enumeration: AlarmLevel
Value: 0, Text: "Normal", ActiveColor: Green
Value: 1, Text: "Warning", ActiveColor: Yellow
Value: 2, Text: "Critical", ActiveColor: Red
```

Display automatically uses colors when configured.

Best Practices

1. **Use standard names** - Consistent naming convention
2. **Start with 0** - Follow programming standards
3. **Document values** - Clear descriptions
4. **Plan for expansion** - Leave gaps for future values
5. **Group related sets** - Use categories
6. **Consider localization** - Plan for multiple languages
7. **Test all values** - Verify mappings work

Display Integration

TextBox Display

xml

```xml
<TextBox Text="{Tag.Motor1.ValueAsString}" />
```

ComboBox Selection

xml

```xml
<ComboBox SelectedValue="{Tag.Motor1.Value}">
    <ComboBoxItem Value="0">Manual</ComboBoxItem>
    <ComboBoxItem Value="1">Auto</ComboBoxItem>
</ComboBox>
```

Color Indication

xml

```xml
<Rectangle Fill="{Tag.Status.ActiveColor}" />
```

Troubleshooting

Text not displaying:

- Verify enumeration assigned to tag
- Check value exists in enumeration
- Use ValueAsString property
- Confirm display binding

Wrong text shown:

- Review value mappings
- Check for duplicate values
- Verify enumeration selection
- Test with known values

Colors not working:

- Configure ActiveColor/InactiveColor
- Use color properties in display
- Check color format (#RRGGBB)
- Test color visibility

Advanced Usage

Dynamic Enumeration Selection

The enumeration set of a tag is part of its configuration. Scripts can read the **Enumeration** property of a tag but cannot assign it. Select the enumeration set in Designer instead:

- **Tag**: the **Enumeration** column in **Unified Namespace Tags**.
- **UserType member**: the **Enumeration** column of the member in **Unified Namespace UserTypes** (right-click the column headers to show it). Every instance of the UserType uses this enumeration unless the instance overrides it.
- **One member of one instance**: the **Enumeration** field of the **Tag Properties** dialog for that member.

To show different text for the same value depending on a runtime condition, build the text in a script, as in the example below.

Custom Value Formatting

csharp

```
// Override enumeration display
string GetStatusText(int value)
{
    switch (value)
    {
        case 0: return "System Off";
        case 1: return "System Running";
        case 2: return "System in Fault";
        default: return "Unknown";
    }
}
```

In this section...
