Displays Units Conversion Reference

Add automatic units conversion.

Displays Units Conversion (Reference) enables automatic conversion between measurement systems, allowing multiple client displays to show values in their preferred units (Metric/Imperial) while connected to the same server.

Units Conversion provides:

- Metric to Imperial conversion
- Imperial to Metric conversion
- Custom conversion tables
- Client-specific display units
- Runtime unit switching
- Extensible conversion definitions

Each client station can display values in its preferred measurement system without affecting other connected clients or server values.

Configuration Steps
Conversion Table Properties
Standard Conversions
Runtime Usage
Creating Custom Tables
Multi-Client Scenarios
Best Practices
Measurement Types
Troubleshooting
Examples

Configuration Steps

1. **Set Tag Units**
   - Navigate to **Unified Namespace Tags**
   - Configure Units property for each tag
   - Or set via script: `@Tag.Temperature.Units = "°C"`
2. **Apply Conversion Table**
   - Assign conversion table to client
   - Set `@Client.Units = "MetricToImperial"`
   - Values automatically convert
3. **Display Converted Units**
   - Use `@Tag.Temperature.DisplayUnits`
   - Shows converted unit string
   - Updates with conversion changes

Conversion Table Properties

Property	Description	Example
BaseUnit	Original unit before conversion	°C
NewUnit	Unit after conversion	°F
Div	Division factor for conversion	1.8
Add	Addition factor for conversion	32
BaseName	Base unit display name	Celsius
NewName	New unit display name	Fahrenheit
MeasurementType	Category of measurement	Temperature

Conversion Formula

```
NewValue = (BaseValue / Div) + Add
```

Standard Conversions

Temperature

Base	New	Div	Add
°C	°F	0.556	32

°F	°C	1.8	-32

Length

Base	New	Div	Add
m	ft	0.3048	0
ft	m	3.281	0
km	mi	1.609	0
mi	km	0.621	0

Pressure

Base	New	Div	Add
bar	psi	0.0689	0
psi	bar	14.504	0
kPa	psi	6.895	0

Runtime Usage

Client-Side Configuration

csharp

```csharp
// Set conversion table for client
@Client.Units = "MetricToImperial";

// Check current conversion
string currentTable = @Client.Units;

// Remove conversion
@Client.Units = "";
```

Tag Properties

csharp

```csharp
// Original unit
string baseUnit = @Tag.Pressure.Units;  // "bar"

// Converted unit (after conversion applied)
string displayUnit = @Tag.Pressure.DisplayUnits;  // "psi"

// Original value
double baseValue = @Tag.Pressure.BaseValue;  // 10 bar

// Converted value
double displayValue = @Tag.Pressure.Value;  // 145.04 psi
```

Creating Custom Tables

1. Navigate to **Displays Units Conversion**
2. Create new conversion set
3. Define conversions:
    - BaseUnit and NewUnit
    - Div and Add factors
    - MeasurementType category

Example: Custom Speed Conversion

```
BaseUnit: m/s
NewUnit: km/h
Div: 0.278
Add: 0
MeasurementType: Speed
```

Multi-Client Scenarios

Different Units Per Client

csharp

```
// Client 1 - US Operator
@Client.Units = "ImperialUnits";
// Sees: 72°F, 14.7 psi, 100 ft

// Client 2 - EU Operator
@Client.Units = "MetricUnits";
// Sees: 22.2°C, 1.01 bar, 30.5 m

// Server stores single value set
// Each client sees converted display
```

Best Practices

1. **Define Units Early** - Set during tag creation
2. **Group by Type** - Organize conversions by measurement
3. **Test Conversions** - Verify accuracy
4. **Document Custom** - Explain custom conversions
5. **Consider Precision** - Account for rounding
6. **Default Tables** - Provide standard sets
7. **User Preferences** - Store per operator

Measurement Types

Organize conversions by category:

- **Temperature** - °C, °F, K
- **Pressure** - bar, psi, kPa, MPa
- **Flow** - m³/h, gpm, l/s
- **Length** - m, ft, km, mi
- **Mass** - kg, lb, ton
- **Volume** - l, gal, m³
- **Speed** - m/s, km/h, mph
- **Energy** - kWh, BTU, J

Troubleshooting

Values not converting:

- Check Units property set on tag
- Verify conversion table assigned
- Confirm table contains unit mapping
- Review Div/Add factors

Wrong conversion:

- Check formula factors
- Verify unit strings match exactly
- Test with known values
- Review calculation order

Client differences:

- Confirm each client's Units setting
- Check local vs server values
- Verify table assignments
- Test client isolation

Examples

Temperature Gauge

xml

```
<TextBlock Text="{Tag.Temperature.Value}" />
<TextBlock Text="{Tag.Temperature.DisplayUnits}" />
<!-- Shows: "72" and "°F" for Imperial client -->
```

Dynamic Switching

csharp

```
public void ToggleUnits()
{
    if (@Client.Units == "MetricToImperial")
        @Client.Units = "";
    else
        @Client.Units = "MetricToImperial";
}
```

In this section...
