Drawing Properties Panel Reference

Edit properties of display elements.

Drawing Properties Panel (Reference) displays context-sensitive properties for selected elements, organized in expandable sections within the Properties Sidebar.

Properties Panel provides:

- Element-specific settings
- Visual configuration
- Layout control
- Dynamic bindings
- Transform effects
- Replace operations

roperty Sections
Advanced Properties
Dynamic Operations
Draw Tool Options
Overview
Basic Tag Insertion
Auto-Generated Elements
Symbol Integration
Asset Usage in Displays
Best Practices Checklist
Troubleshooting

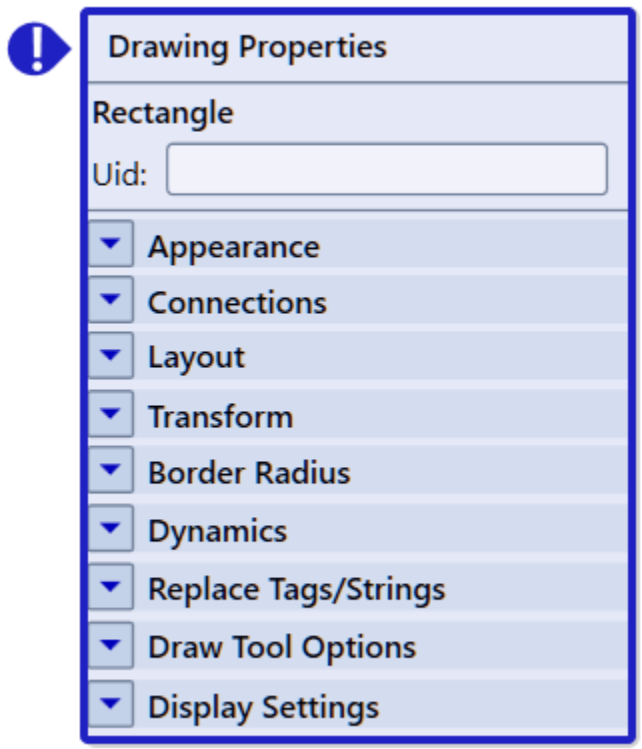
Located in **Properties Sidebar** (right side) when in Drawing workspace.

roperty Sections

Drawing Properties

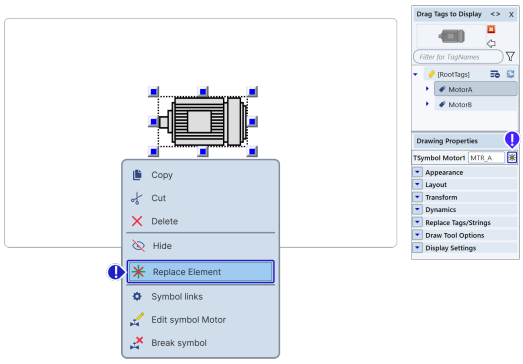
Core element identification:

- **Element** - Object type
- **Uid** - Custom identifier
- **Replace Element** - Swap symbols preserving bindings



Replace Graphical Component

The context menu, Replace Element, or the icon at the right side of the Drawing Properties element identified, allow to select another component, still preserving the tag binding.



Appearance

Visual styling controls:

- **Theme** - Preview theme (design-time only)
- **Fill** - Background color/pattern
- **Line** - Border properties
- **Opacity** - Transparency level

Connections

Enable shape connectors:

- Links shapes with polylines
- Auto-adjusts on movement
- Visual relationship indicators

Layout

Position and responsive settings:

Property	Function	Responsive
Left/Top	Position	? Enable
Width/Height	Size	? Enable

Checked properties scale proportionally when display resizes.

Advanced Properties

Transform

Visual effects:

- **DropShadow** - Shadow effects
- **Rotation** - Angular rotation
- **Skew** - Distortion effects

Border Radius

Rectangle corners (Rectangle only):

- **X** - Horizontal radius
- **Y** - Vertical radius

Text

Typography settings:

- Font family, size, style
- Alignment and wrapping
- Localization options

Contents Align

Internal alignment:

- Left, Center, Right
- Top, Middle, Bottom

Dynamic Operations

Dynamics Clipboard

- **GET** - Copy dynamics to clipboard
- **APPLY** - Paste dynamics to selection
- **EDIT** - Open dynamics editor

Right-click dynamic names to hide from apply.

Replace Tags/Strings

Batch replacement:

- Select source tag
- Enter replacement
- Updates all instances

Draw Tool Options

Visibility Helpers

- **Highlight Location** - Shows element boundaries
- **Apply Shadow on TextBlocks** - Improves text visibility (design-time only)

Insert Tags to Displays (Reference)

Insert Tags enables direct integration of solution tags into displays by dragging or pasting, automatically creating appropriate visual elements with data bindings.

Overview

Tag insertion provides:

- Automatic UI generation
- Symbol integration
- Array handling
- Custom type support
- Asset conversion
- Batch operations

Basic Tag Insertion

Copy and Paste Method

1. Navigate to **Unified Namespace Tags**
2. Right-click tag row

3. Select **Copy Selected Row**
4. In Draw workspace, right-click
5. Select **Paste**

Drag and Drop Method

1. Open Tags Panel (right side)
2. Locate desired tag
3. Drag to display canvas
4. Auto-creates appropriate control

Auto-Generated Elements

Simple Tags

Creates label and value display:

```
[Tag Name] [TextBox with Value]
```

Array Tags

Generates elements for all indices:

```
Tag.Array[0] [Value]
Tag.Array[1] [Value]
Tag.Array[2] [Value]
...
```

Custom Types

Creates hierarchical display:

```
Parent.Element1 [Value]
Parent.Element2 [Value]
```

Symbol Integration

When tag name matches symbol:

1. Symbol automatically selected
2. Tag binding pre-configured
3. Dynamic properties set

Configure at **Displays** → **Symbols** for tag-symbol mapping.

Asset Usage in Displays

Portable/HTML5 Displays

Assets require type conversion:

csharp

```csharp
// Convert asset to integer
@TK.ToInt(Asset.Path)

// Convert to string
@TK.ToString(Asset.Path)

// Use in visibility
Visible = @TK.ToInt(Asset.Status) == 1
```

WPF-Only Displays

Direct dynamic handling:

csharp

```csharp
// Direct asset usage
Value = @Asset.Path.Value
```

Best Practices Checklist

- [ ] **Organize Tags First** - Structure before inserting
- [ ] **Use Consistent Naming** - Enables symbol matching
- [ ] **Test Array Limits** - Verify all elements
- [ ] **Preview Layout** - Check spacing
- [ ] **Configure Symbols** - Set up matching names
- [ ] **Handle Assets Properly** - Convert for portability

Troubleshooting

Tags not pasting:

- Verify tag exists
- Check clipboard content
- Ensure draw area focus

Wrong element type:

- Check data type
- Review symbol mappings
- Verify tag configuration

Assets not working:

- Check display engine
- Apply type conversion
- Verify asset path

In this section...