Alarms Areas Reference

Structure alarms by area or system zone.

Reference **Modules** **Alarms** **UI** Items | Groups | Areas | Global | Monitor

---

Alarm Areas (Reference) provide hierarchical organization for alarm management in your FrameworX solution, Alarm Areas enable:

- Hierarchical grouping of alarms
- Plant or process area organization
- Filtering and navigation in displays
- Area-based acknowledgment
- Roll-up alarm counts and area status

Areas create a tree structure for organizing alarms by location, system, or functional grouping.

Configuration Properties
Creating Alarm Areas
Hierarchical Structure
Area-Based Operations
Display Integration
Common Hierarchies
Runtime Namespace
Best Practices Checklist
Troubleshooting

Configuration Properties

Property	Description	Required
Name	Area identifier	Yes
Parent	Parent area for hierarchy	No (root level)
Description	Area purpose and details	No
Category	Classification group for organizing alarm areas	No

Creating Alarm Areas

1. Navigate to **Alarms** **Areas**
2. Click **Plus icon**
3. Enter **Name**
4. Select **Parent** for hierarchy (leave blank for root)
5. Add **Description**
6. Press **Enter** to save

Hierarchical Structure

Example plant hierarchy:

- Plant
  - Production
    - Line 1
    - Line 2
    - Line 3
  - Utilities
    - Power
    - Water
    - HVAC
  - Storage
    - RawMaterials
    - FinishedGoods

Configure AlarmItems:

```
AlarmItem.Area = "Plant.Production.Line1"
```

Area-Based Operations

Acknowledge by Area

`AckAll` and `AckAllLocal` are change-triggered command properties, not methods. Each change of value acknowledges, and the value is not reset afterwards, so writing the same value again does nothing (a constant 1 acknowledges only the first time). To acknowledge again, toggle the value: use the Toggle action (ActionType ToggleValue), or write the current value + 1. Change the value once per acknowledge: setting it to 0 and back to 1 in the same script may not acknowledge.

```
// Acknowledge every unacknowledged alarm in Production and all its child areas
@Alarm.Area.Production.AckAll = @Alarm.Area.Production.AckAll + 1;

// Acknowledge only the items assigned directly to this area (child areas untouched)
@Alarm.Area.Production.AckAllLocal = @Alarm.Area.Production.AckAllLocal + 1;
```

Filter Alarms

```
// Active alarms in an area and every area below it
int productionCount = @Alarm.Area.Production.TotalCount;

// Unacknowledged alarms in the same subtree
int productionUnAck = @Alarm.Area.Production.UnAckCount;

// Same counts restricted to items assigned directly to the area
int localCount = @Alarm.Area.Production.TotalCountLocal;

// Build a filter for this area plus its children, then apply it to the active alarm table
string filter = await @Alarm.Area.Production.GetFilterAsync();
var rows = @Alarm.QueryActive.Table.Select(filter);
```

Disable or Suspend Areas

```
// Disable alarm evaluation for the whole area subtree
@Alarm.Area.Storage.Disable = 1;

// Suspend new activations without clearing already-active alarms
@Alarm.Area.Storage.Suspend = 1;

// Non-recursive variants affect only items assigned directly to the area
@Alarm.Area.Storage.DisableLocal = 1;
```

Display Integration

AlarmAreas work with display controls for visualization:

AlarmViewer Filtering

Filter display to show only specific areas:

- Set AlarmViewer.AreaFilter = "Production.Line1"
- Users see only relevant alarms

Area Navigation Tree

Use AlarmArea control for hierarchical navigation:

- Tree view of all areas
- Click to filter associated alarms
- Visual indicators for active alarms per area

Common Hierarchies

By Location

- Site
  - Building_1
    - Floor_1
    - Floor_2
  - Building_B
    - Floor_1
    - Floor_2

By Function

- Operations

    - Process
    - Quality
    - Safety

By System

- `PlantSystems`
    - `FeedSystem`
    - `ReactorSystem`
    - `ProductSystem`

Runtime Namespace

Access areas programmatically:

```
// Look up an area by name
var area = @Alarm.Area["Production"];

// Identity
string areaName = area.Name;            // script-stable identifier
string label = area.DisplayName;        // localized label shown on operator screens
string parent = area.ParentArea;        // parent area name, empty at root level
string notes = area.Description;

// Counts and status
int total = area.TotalCount;            // active alarms in this area and below
int unAck = area.UnAckCount;            // unacknowledged alarms in this area and below
int totalLocal = area.TotalCountLocal;  // this area only
int unAckLocal = area.UnAckCountLocal;  // this area only
int state = area.AlarmState;            // rolled-up state of the area subtree
var worst = area.PriorityItem;          // highest-priority active item in the subtree

// Operations
area.AckAll = area.AckAll + 1;          // acknowledge the subtree (each change of value acknowledges)
area.Disable = 1;                       // disable the subtree
area.Suspend = 1;                       // suspend the subtree

// Child areas
string[] children = await area.GetChildrenAreasAsync(true);
```

Best Practices Checklist

- [ ] **Match plant structure** - Mirror physical or logical layout
- [ ] **Consistent naming** - Use standard naming conventions
- [ ] **Limit depth** - 3-4 levels maximum for usability
- [ ] **Group functionally** - Related equipment in same area
- [ ] **Document responsibility** - Use the Description column for team assignments
- [ ] **Plan hierarchy early** - Harder to restructure later
- [ ] **Test filtering** - Verify displays show correct areas

Troubleshooting

Alarms not appearing in area:

- Verify Area property in AlarmItem
- Check area name spelling/path
- Confirm the area is not Disabled or Suspended

Cannot acknowledge by area:

- Check user permissions
- Verify AlarmGroup.AckRequired
- Confirm alarms are active

Area tree not showing:

- Verify Parent relationships
- Check for circular references

- Confirm root areas have no Parent

In this section...
