OpcAlarmConditionNode Members

Namespace: Opc.UaFx
Assemblies: Opc.UaFx.Advanced.dll, Opc.UaFx.Advanced.dll
The OpcAlarmConditionNode type exposes the following members.

Constructors

OpcAlarmConditionNode(IOpcNode, OpcName)

Initializes a new instance of the OpcAlarmConditionNode class accessible by the `name` specified as a child node of the `parent` node given.

C#

```
public OpcAlarmConditionNode(IOpcNode parent, OpcName name)
```

Parameters

`parent` IOpcNode

The IOpcNode used as the parent node or a null reference (Nothing in Visual Basic) in the case there is no parent node available.

`name` OpcName

The OpcName through that the new alarm condition node can be accessed.

OpcAlarmConditionNode(IOpcNode, OpcName, OpcNodeId)

Initializes a new instance of the OpcAlarmConditionNode class accessible by the `name` and `id` specified as a child node of the `parent` node given.

C#

```
public OpcAlarmConditionNode(IOpcNode parent, OpcName name, OpcNodeId id)
```

Parameters

`parent` IOpcNode

The IOpcNode used as the parent node or a null reference (Nothing in Visual Basic) in the case there is no parent node available.

`name` OpcName

The OpcName through that the new alarm condition node can be accessed.

`id` OpcNodeId

The OpcNodeId through that the new alarm condition node can be identified and accessed.

OpcAlarmConditionNode(OpcName)

Initializes a new instance of the OpcAlarmConditionNode class accessible by the `name` specified.

C#

```
public OpcAlarmConditionNode(OpcName name)
```

Parameters

`name` OpcName

The OpcName through that the new alarm condition node can be accessed.

OpcAlarmConditionNode(OpcName, OpcNodeId)

Initializes a new instance of the OpcAlarmConditionNode class accessible by the `name` and `id` specified.

C#

```
public OpcAlarmConditionNode(OpcName name, OpcNodeId id)
```

Parameters

`name` OpcName

The OpcName through that the new alarm condition node can be accessed.

`id` OpcNodeId

The OpcNodeId through that the new alarm condition node can be identified and accessed.

Properties

DefaultTypeDefinitionId

Gets the default identifier which identifies the node that defines the underlying node type from that this OpcInstanceNode has been created.

C#

```
protected override OpcNodeId DefaultTypeDefinitionId { get; }
```

Property Value

OpcNodeId

The OpcNodeId of the type node from that this OpcInstanceNode has been created from. These type node

defines the typical structure of an instance node of its type definition. If there exists no specific type definition node a null reference (Nothing in Visual Basic).

InputNodeId

Gets or sets an identifier which represents the node identifier of the variable the value of which is used as primary input in the calculation of the alarm state.

C#

```
public OpcNodeId InputNodeId { get; set; }
```

Property Value

OpcNodeId

If this variable is not in the address space, Null is provided. In some systems, an alarm may be calculated based on multiple variable values; it is up to the system to determine which variable's node identifier is used.

InputNodeIdNode

Gets the OpcNodeIdPropertyNode of the InputNodeId property.

C#

```
public OpcNodeIdPropertyNode InputNodeIdNode { get; }
```

Property Value

OpcNodeIdPropertyNode

An instance of the OpcNodeIdPropertyNode class.

IsActive

Gets a value indicating whether the alarm situation represented currently exists.

C#

```
public bool IsActive { get; }
```

Property Value

Boolean

A value indicating whether the alarm situation represented currently exists.

IsActiveNode

Gets the OpcTwoStateVariableNode of the IsActive property.

C#

```
public OpcTwoStateVariableNode IsActiveNode { get; }
```

Property Value

OpcTwoStateVariableNode

An instance of the OpcTwoStateVariableNode class.

IsSuppressed

Gets a value indicating whether the alarm is suppressed due to system specific reasons.

C#

```
public bool IsSuppressed { get; }
```

Property Value

Boolean

The value is used internally by the server to automatically suppress alarms due to system specific reasons. For example a system may be configured to suppress alarms that are associated with machinery that is shutdown, such as a low level alarm for a tank that is currently not in use.

IsSuppressedNode

Gets the OpcTwoStateVariableNode of the IsSuppressed property.

C#

```
public OpcTwoStateVariableNode IsSuppressedNode { get; }
```

Property Value

OpcTwoStateVariableNode

An instance of the OpcTwoStateVariableNode class.

IsSuppressedOrShelved

Gets a value indicating whether the alarm is either in the IsSuppressed or Shelving.

C#

```
public bool IsSuppressedOrShelved { get; }
```

Property Value

Boolean

The value true if the alarm is either in the IsSuppressed or Shelving; otherwise (if both are not present) the value false.

Remarks

In case there the value is false the client typically does not display the alarm. State transitions associated with the alarm do occur, but they are not typically displayed by the clients as long as the alarm remains in either the suppressed or shelved state.

IsSuppressedOrShelvedNode

Gets the OpcPropertyNode´1 of the IsSuppressedOrShelved property.

C#

```
public OpcPropertyNode<bool> IsSuppressedOrShelvedNode { get; }
```

Property Value

OpcPropertyNode<Boolean>

An instance of the OpcPropertyNode´1 class.

MaxTimeShelved

Gets or sets the maximum time that the alarm condition may be shelved.

C#

```
public double MaxTimeShelved { get; set; }
```

Property Value

Double

The value is expressed as duration. Systems can use this property to prevent permanent shelving of an alarm. If this property is present it will be an upper limit on the duration passed into a TimedShelve method call. If a value that exceeds the value of this property is passed to the TimedShelve method, than a BadShelvingTimeOutOfRange error code is returned on the call. If this property is present it will also be enforced for the OneShotShelved state, in that the alarm condition will transition to the Unshelved state from the OneShotShelved state if the duration specified in this property expires following a OneShotShelve operation without a change of any of the other items associated with the condition.

MaxTimeShelvedNode

Gets the OpcPropertyNode´1 of the MaxTimeShelved property.

C#

```
public OpcPropertyNode<double> MaxTimeShelvedNode { get; }
```

Property Value

OpcPropertyNode<Double>

An instance of the OpcPropertyNode´1 class.

OneShotShelveCallback

Gets or sets a callback used to shelve the condition node once.

C#

```
public OpcNodeFunc<OpcAlarmConditionNode> OneShotShelveCallback { get; set; }
```

Property Value

OpcNodeFunc<OpcAlarmConditionNode>

A OpcNodeFunc´1 used to shelve the condition node once. The value can also be a null reference (Nothing in Visual Basic).

Shelving

Gets a node which is used to suggest whether an alarm shall (temporarily) be prevented from being displayed to the user. It is quite often used to block nuisance alarms.

C#

```
public OpcShelvedStateMachineNode Shelving { get; }
```

Property Value

OpcShelvedStateMachineNode

An instance of the OpcShelvedStateMachineNode to suggest whether an alarm shall (temporarily) be prevented.

TimedShelveCallback

Gets or sets a callback used to shelve the condition node for a specific amount of time.

C#

```
public OpcNodeFunc<OpcAlarmConditionNode, TimeSpan> TimedShelveCallback { get; set; }
```

Property Value

OpcNodeFunc<OpcAlarmConditionNode, TimeSpan>

A OpcNodeFunc´2 used to shelve the condition node for a specific amount of time. The value can also be a null reference (Nothing in Visual Basic).

TimedUnshelveCallback

Gets or sets a callback used to unshelve the condition node after a specific amount of time being shelved.

C#

```
public OpcNodeFunc<OpcAlarmConditionNode> TimedUnshelveCallback { get; set; }
```

Property Value

OpcNodeFunc<OpcAlarmConditionNode>

A OpcNodeAction´1 used to unshelve the condition node. The value can also be a null reference (Nothing in Visual Basic).

UnshelveCallback

Gets or sets a callback used to unshelve the condition node.

C#

```
public OpcNodeFunc<OpcAlarmConditionNode> UnshelveCallback { get; set; }
```

Property Value

OpcNodeFunc<OpcAlarmConditionNode>

A OpcNodeFunc´1 used to unshelve the condition node. The value can also be a null reference (Nothing in Visual Basic).

Methods

CreateBranchCore()

Creates a new instance of the OpcAlarmConditionNode using the same Id and Name as this node.

C#

```
protected override OpcConditionNode CreateBranchCore()
```

Returns

OpcConditionNode

A new instance of the OpcAlarmConditionNode identifiable and accessible througth the same Id and Name as this node.

OneShotShelveCore(OpcNodeContext<OpcAlarmConditionNode>)

Shelves a condition node once using the `context` specified.

C#

```
protected virtual OpcStatusCode OneShotShelveCore(OpcNodeContext<OpcAlarmConditionNode>
context)
```

Parameters

`context` OpcNodeContext<OpcAlarmConditionNode>

The OpcNodeContext´1 to use to shelve a condition node once.

Returns

OpcStatusCode

The OpcStatusCode specifying the outcome of the shelving using the OneShotShelveCallback or Good if there is no custom callback routine defined.

TimedShelveCore(OpcNodeContext<OpcAlarmConditionNode>, TimeSpan)

Shelves a condition node using the `context` and `duration` information specified.

C#

```
protected virtual OpcStatusCode TimedShelveCore(OpcNodeContext<OpcAlarmConditionNode>
context, TimeSpan duration)
```

Parameters

`context` OpcNodeContext<OpcAlarmConditionNode>

The OpcNodeContext´1 to use to shelve the condition node for specific amount of time.

`duration` TimeSpan

The amount of time a condition node is to be shelved.

Returns

OpcStatusCode

The OpcStatusCode specifying the outcome of the shelving using the TimedShelveCallback or Good if there is no custom callback routine defined.

TimedUnshelveCore(OpcNodeContext<OpcAlarmConditionNode>)

Unshelves a condition node using the `context` specified after being shelved for a specific amount of time.

C#

```
protected virtual OpcStatusCode TimedUnshelveCore(OpcNodeContext<OpcAlarmConditionNode>
context)
```

Parameters

`context` OpcNodeContext<OpcAlarmConditionNode>

The OpcNodeContext´1 to use to unshelve a condition node.

Returns

OpcStatusCode

The OpcStatusCode specifying the outcome of the unshelving using the TimedUnshelveCallback or Good if there is no custom callback routine defined.

UnshelveCore(OpcNodeContext<OpcAlarmConditionNode>)

Unshelves a condition node using the `context` specified.

C#

```
protected virtual OpcStatusCode UnshelveCore(OpcNodeContext<OpcAlarmConditionNode> context)
```

Parameters

`context` OpcNodeContext<OpcAlarmConditionNode>

The OpcNodeContext´1 to use to unshelve a condition node.

Returns

OpcStatusCode

The OpcStatusCode specifying the outcome of the unshelving using the UnshelveCallback or Good if there is no custom callback routine defined.

Table of Contents