OpcConditionNode Members

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

Constructors

OpcConditionNode(IOpcNode, OpcName)

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

C#

```
public OpcConditionNode(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 condition node can be accessed.

OpcConditionNode(IOpcNode, OpcName, OpcNodeId)

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

C#

```
public OpcConditionNode(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 condition node can be accessed.

`id` OpcNodeId

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

OpcConditionNode(OpcName)

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

C#

```
public OpcConditionNode(OpcName name)
```

Parameters

`name` OpcName

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

OpcConditionNode(OpcName, OpcNodeId)

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

C#

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

Parameters

`name` OpcName

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

`id` OpcNodeId

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

Properties

AddCommentCallback

Gets or sets a callback used to add a comment to the condition node.

C#

```
public OpcAddCommentCallback AddCommentCallback { get; set; }
```

Property Value

OpcAddCommentCallback

A OpcAddCommentCallback used to add a comment to the condition node. The value can also be a null reference (Nothing in Visual Basic).

AddCommentNode

Gets the OpcAddCommentMethodNode used to handle 'AddComment' method calls to add a comment to the condition node.

C#

```
public OpcAddCommentMethodNode AddCommentNode { get; }
```

Property Value

OpcAddCommentMethodNode

An instance of the OpcAddCommentMethodNode class. Which uses an OpcConditionNode defined callback to add a comment to the condition node.

AutoReportChanges

Gets or sets a value indicating whether the OpcConditionNode will automatically report an event when a change method call completes.

C#

```
public bool AutoReportChanges { get; set; }
```

Property Value

Boolean

The value true, if the event automatically reports any change performed using a method to modify the node; otherwise the value false (the default value is false).

BranchId

Gets or sets an identifier which identifies the branch to that the event does belong.

C#

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

Property Value

OpcNodeId

A null reference (Nothing in Visual Basic) if the event relates to the current state of the condition. Otherwise it identifies a previous state of this condition that still needs attention by an operator. If the current condition branch is transformed into a previous condition branch then the server needs to assign a non-null branch identifier. An initial event for the branch will generated with the values of the condition branch and the new branch identifier. The condition branch can be updated many times before it is no longer needed. When the condition branch no longer requires operator input the final event will have IsRetained set to false. The IsRetained property on the current event is true, as long as any condition

branches require operator input.

BranchIdNode

Gets the OpcNodeIdPropertyNode of the BranchId property.

C#

```
public OpcNodeIdPropertyNode BranchIdNode { get; }
```

Property Value

OpcNodeIdPropertyNode

An instance of the OpcNodeIdPropertyNode class.

ClientUserId

Gets or sets an identifier that is related to the Comment and contains the identity of the user who inserted the most recent Comment.

C#

```
public string ClientUserId { get; set; }
```

Property Value

String

A value which identifies the user who inserted the most recent Comment.

ClientUserIdNode

Gets the OpcPropertyNode´1 of the ClientUserId property.

C#

```
public OpcPropertyNode<string> ClientUserIdNode { get; }
```

Property Value

OpcPropertyNode<String>

An instance of the OpcPropertyNode´1 class.

Comment

Gets or sets the last comment provided for a certain state (condition branch).

C#

```
public OpcText Comment { get; set; }
```

Property Value

OpcText

The last comment provided for a certian state (condition branch).

Remarks

It may have been provided by an 'AddComment' method, some other method or in some other manner. The initial value of this variable is a null reference (Nothing in Visual Basic), unless it is provided in some other manner. If a method provides as an option the ability to set a comment, then the value of this variable is reset to a null reference (Nothing in Visual Basic) if an optional comment is not provided.

CommentNode

Gets the OpcTextConditionVariableNode of the Comment property.

C#

```
public OpcTextConditionVariableNode CommentNode { get; }
```

Property Value

OpcTextConditionVariableNode

An instance of the OpcTextConditionVariableNode class.

ConditionClassId

Gets or sets a value which specifies in which domain this condition is used. It is the OpcNodeId of the corresponding condition class type.

C#

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

Property Value

OpcNodeId

An OpcNodeId which identifies the corresponding condition class type.

ConditionClassIdNode

Gets the OpcNodeIdPropertyNode of the ConditionClassId property.

C#

```
public OpcNodeIdPropertyNode ConditionClassIdNode { get; }
```

Property Value

OpcNodeIdPropertyNode

An instance of the OpcNodeIdPropertyNode class.

ConditionClassName

Gets or sets a value that matches the display name of the condition class type.

C#

```
public OpcText ConditionClassName { get; set; }
```

Property Value

OpcText

The display name of the condition class referenced by the ConditionClassId.

ConditionClassNameNode

Gets the OpcTextPropertyNode of the ConditionClassName property.

C#

```
public OpcTextPropertyNode ConditionClassNameNode { get; }
```

Property Value

OpcTextPropertyNode

An instance of the OpcTextPropertyNode class.

ConditionName

Gets or sets a value which identifies the condition instance that the event originated from.

C#

```
public string ConditionName { get; set; }
```

Property Value

String

A value which identifies the condition instance that the event originated from.

Remarks

It can be used together with the SourceName in a user display to distinguish between different condition instances. If a condition source has only one instance of a condition type, and the server has no instance name, the server shall supply the condition type browse name.

ConditionNameNode

Gets the OpcPropertyNode´1 of the ConditionName property.

C#

```
public OpcPropertyNode<string> ConditionNameNode { get; }
```

Property Value

OpcPropertyNode<String>

An instance of the OpcPropertyNode´1 class.

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).

DisableCallback

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

C#

```
public OpcNodeFunc<OpcConditionNode> DisableCallback { get; set; }
```

Property Value

OpcNodeFunc<OpcConditionNode>

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

DisableNode

Gets the OpcActionMethodNode used to handle 'Disable' method calls to disable the condition.

C#

```
public OpcActionMethodNode DisableNode { get; }
```

Property Value

OpcActionMethodNode

An instance of the OpcActionMethodNode class. Which uses an OpcConditionNode defined callback to disable the condition.

EnableCallback

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

C#

```
public OpcNodeFunc<OpcConditionNode> EnableCallback { get; set; }
```

Property Value

OpcNodeFunc<OpcConditionNode>

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

EnableNode

Gets the OpcActionMethodNode used to handle 'Enable' method calls to enable the condition.

C#

```
public OpcActionMethodNode EnableNode { get; }
```

Property Value

OpcActionMethodNode

An instance of the OpcActionMethodNode class. Which uses an OpcConditionNode defined callback to enable the condition.

IsEnabled

Gets a value indicating whether the condition is enabled.

C#

```
public bool IsEnabled { get; }
```

Property Value

Boolean

A value indicating whether the condition is enabled.

Remarks

A condition's IsEnabled state effects the generation of event notifications and as such results in the following specific behaviour: * When the condition enters the disabled state, the IsRetained property of this condition is set to the false enabled state by the server so indicate to the client that the condition is currently not of interest to clients.

- When the condition enters the enabled state, the condition is to be evaluated and all of its properties updated to reflect the current values. If this evaluation causes the IsRetained property to transition to to true for any condition branch, then an event notification is generated for that condition branch.
- The server may choose to continue to test for a condition while it is disabled. However, no event notifications will be generated while the condition is disabled.
- For any condition that exists in the address space the attributes and the following variables will continue to have valid values even in the disabled state; EventId, EventTypeId, SourceNodeId, SourceName, Time, and IsEnabled. Other properties may no longer provide current valid values. All variables that are no longer provided will return a status of BadConditionDisabled. The event that reports the disabled state will report the properties as a null reference (Nothing in Visual Basic) or with a status of BadConditionDisabled.

IsEnabledNode

Gets the OpcTwoStateVariableNode of the IsEnabled property.

C#

```
public OpcTwoStateVariableNode IsEnabledNode { get; }
```

Property Value

OpcTwoStateVariableNode

An instance of the OpcTwoStateVariableNode class.

IsRetained

Gets or sets a value indicating whether the condition is in a state that is interesting for a client wishing to synchronize its state with the server's state.

C#

```
public bool IsRetained { get; set; }
```

Property Value

Boolean

The value true if the condition (or condition branch) is being in a state that is interesting for a client wishing to synchronize its state with the server's state. The logic to determine how this flag is set is server specific. Typically all active alarms would have the IsRetained flag set; however, it is also possible for inactive alarms to have their IsRetained flag set to true. In general events are only generated for conditions that have their IsRetained property set to the value true.

IsRetainedNode

Gets the OpcPropertyNode´1 of the IsRetained property.

C#

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

Property Value

OpcPropertyNode<Boolean>

An instance of the OpcPropertyNode´1 class.

LastSeverity

Gets or sets a value which provides the previous severity of the condition branch.

C#

```
public OpcEventSeverity LastSeverity { get; set; }
```

Property Value

OpcEventSeverity

The initial value of this property is Undefined; it will provide a more meaningful value only after a severity change. The new severity is supplied via the Severity property.

LastSeverityNode

Gets the OpcConditionVariableNode´1 of the LastSeverity property.

C#

```
public OpcConditionVariableNode<ushort> LastSeverityNode { get; }
```

Property Value

OpcConditionVariableNode<UInt16>

An instance of the OpcConditionVariableNode´1 class.

Quality

Gets or sets a value which reveals the status of process values or other resources that this condition is based upon.

C#

```
public OpcStatus Quality { get; set; }
```

Property Value

OpcStatus

The quality of data that this condition is based upon. If, for example, a process value is 'Uncertain', the associated 'LevelAlarm' condition is also questionable. Values for the Quality can be any of the OpcStatusCode enumeration members as well as Good, Uncertain and Bad. These status codes are similar to but slightly more generic than the description of data quality in the various field bus specifications. It is the responsibility of the server to map internal status information to these codes. A server which supports no quality information returns Good. This quality can also reflect the communication status associated with the system that this value or resource is based on and from which this alarm was received. For communication errors to the underlying system, especially those that result in some unavailable event fields, the quality is BadNoCommunication.

QualityNode

Gets the OpcStatusConditionVariableNode of the Quality property.

C#

```
public OpcStatusConditionVariableNode QualityNode { get; }
```

Property Value

OpcStatusConditionVariableNode

An instance of the OpcStatusConditionVariableNode class.

Methods

AddComment(OpcContext, Byte[], OpcText)

Applies a comment to the state reported by an event notification which can be identified by the `eventId` using the specified `context`.

C#

```
public void AddComment(OpcContext context, byte[] eventId, OpcText comment)
```

Parameters

`context` OpcContext

The OpcContext to use.

`eventId` Byte[]

The identifier which identifies the particular event notification its reported state for the condition node is to be commented.

`comment` OpcText

The text to apply on the condition state.

Exceptions

ArgumentNullException

The `context` or `eventId` is a null reference (Nothing in Visual Basic).

OpcException

The call failed (see exception details for more information).

AddComment(OpcContext, OpcText)

Applies a comment to the state reported by an event notification which can be identified by the EventId using the specified `context`.

C#

```
public void AddComment(OpcContext context, OpcText comment)
```

Parameters

`context` OpcContext

The OpcContext to use.

`comment` OpcText

The text to apply on the condition state.

Exceptions

ArgumentNullException

The `context` is a null reference (Nothing in Visual Basic).

OpcException

The call failed (see exception details for more information).

AddCommentCore(OpcNodeContext<OpcConditionNode>, Byte[], OpcText)

Applies the `comment` to the state of the condition node.

C#

```
protected virtual OpcStatusCode AddCommentCore(OpcNodeContext<OpcConditionNode> context,
byte[] eventId, OpcText comment)
```

Parameters

`context` OpcNodeContext<OpcConditionNode>

The OpcNodeContext´1 to use to apply the comment.

`eventId` Byte[]

The identifier identifying a particular event notification where a state was reported for a condition.

`comment` OpcText

A localized text to be applied to the condition.

Returns

OpcStatusCode

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

CreateBranch(OpcContext)

Creates a new instance of the OpcConditionNode class which can be used to maintain branched event information.

C#

```
public OpcConditionNode CreateBranch(OpcContext context)
```

Parameters

`context` OpcContext

The OpcContext to use.

Returns

OpcConditionNode

A new instance of the OpcConditionNode class there its BranchId is set to a Guid created using NewGuid.

CreateBranch(OpcContext, OpcNodeId)

Creates a new instance of the OpcConditionNode class using the `branchId` specified to maintain branched event information.

C#

```
public OpcConditionNode CreateBranch(OpcContext context, OpcNodeId branchId)
```

Parameters

`context` OpcContext

The OpcContext to use.

`branchId` OpcNodeId

The OpcNodeId to use to define the custom branch identifier to use for the BranchId of the new OpcConditionNode instance to create.

Returns

OpcConditionNode

A new instance of the OpcConditionNode class there its BranchId is set to the `branchId` specified.

CreateBranchCore()

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

C#

```
protected virtual OpcConditionNode CreateBranchCore()
```

Returns

OpcConditionNode

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

Disable(OpcContext)

Changes the condition state to 'Disabled' using the specified `context`.

C#

```
public void Disable(OpcContext context)
```

Parameters

`context` OpcContext

The OpcContext to use.

Exceptions

ArgumentNullException

The `context` is a null reference (Nothing in Visual Basic).

OpcException

The call failed (see exception details for more information).

DisableCore(OpcNodeContext<OpcConditionNode>)

Disables the condition node.

C#

```
protected virtual OpcStatusCode DisableCore(OpcNodeContext<OpcConditionNode> context)
```

Parameters

`context` OpcNodeContext<OpcConditionNode>

The OpcNodeContext´1 to use to disable the condition node.

Returns

OpcStatusCode

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

Enable(OpcContext)

Changes the condition state to 'Enabled' using the specified `context`.

C#

```
public void Enable(OpcContext context)
```

Parameters

`context` OpcContext

The OpcContext to use.

Exceptions

ArgumentNullException

The `context` is a null reference (Nothing in Visual Basic).

OpcException

The call failed (see exception details for more information).

EnableCore(OpcNodeContext<OpcConditionNode>)

Enables the condition node.

C#

```
protected virtual OpcStatusCode EnableCore(OpcNodeContext<OpcConditionNode> context)
```

Parameters

`context` OpcNodeContext<OpcConditionNode>

The OpcNodeContext´1 to use to enable the condition node.

Returns

OpcStatusCode

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

InitializeDefaults()

Initializes the default values used by the OpcConditionNode.

C#

```
protected override void InitializeDefaults()
```

Remarks

This method is used to ensure the availability of appropriate node specific default values. For more information like when this method is to be overwritten see InitializeDefaults.

Table of Contents