OpcSubscription Members

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

Events

Changed

Occurs then a monitored is added, modified, deleted or changes on the OpcSubscription were applied.

C#

```
public event OpcSubscriptionChangedEventHandler Changed
```

DataChangeReceived

Occurs then a monitored item receives an event notification which is not handled by an OpcDataChangeReceivedEventHandler on its DataChangeReceived event.

C#

```
public event OpcDataChangeReceivedEventHandler DataChangeReceived
```

Remarks

This event only occurs if an event notification is not handled by the according DataChangeReceived event handler.

EventReceived

Occurs then a monitored item receives an event notification which is not handled by an OpcEventReceivedEventHandler on its EventReceived event.

C#

```
public event OpcEventReceivedEventHandler EventReceived
```

Remarks

This event only occurs if an event notification is not handled by the according EventReceived event handler.

IsPublishingChanged

Occurs then the IsPublishing property has changed.

C#

```
public event EventHandler IsPublishingChanged
```

NotificationReceived

Occurs then a OpcNotification for this OpcSubscription has been received by the session which owns this OpcSubscription.

C#

```
public event OpcNotificationReceivedEventHandler NotificationReceived
```

Properties

Client

Gets the OpcClient used to setup this OpcSubscription.

C#

```
public OpcClient Client { get; }
```

Property Value

OpcClient

An instance of the OpcClient class used to setup this OpcSubscription.

CurrentKeepAliveCount

Gets the actual keep-alive count used by the server for this OpcSubscription which has been negotiated by the server against the KeepAliveCount requested to meet its own constraints.

C#

```
public long CurrentKeepAliveCount { get; }
```

Property Value

Int64

The negoiated and actual KeepAliveCount used.

Remarks

The server negotiates the value up or down to meet its own constraints.

CurrentLifetimeCount

Gets the actual lifetime count used by the server for this OpcSubscription which has been negotiated by the server against the LifetimeCount requested to meet its own constraints.

C#

```
public long CurrentLifetimeCount { get; }
```

Property Value

Int64

The negoiated and actual LifetimeCount used.

Remarks

The server negotiates the value up or down to meet its own constraints.

CurrentPriority

Gets the actual relative priority of the subscription used to priorize notifications being send then more than one subscription is to be notified.

C#

```
public int CurrentPriority { get; }
```

Property Value

Int32

The actual Priority used.

Remarks

For more details see Priority.

CurrentPublishingInterval

Gets the actual publishing interval used by the server for this OpcSubscription which has been negotiated by the server against the PublishingInterval requested to meet its own constraints.

C#

```
public double CurrentPublishingInterval { get; }
```

Property Value

Double

The negoiated and actual PublishingInterval used.

Remarks

The server negotiates the value up or down to meet its own constraints.

CurrentPublishingIsEnabled

Gets the actual value indicating whether publishing is enabled by the server for this OpcSubscription which has been created and validated by the server against its own constraints.

C#

```
public bool CurrentPublishingIsEnabled { get; }
```

Property Value

Boolean

The value true if the server has enabled publishing for this OpcSubscription; otherwise the value false.

DisplayName

Gets or sets the name of the OpcSubscription used just for display purpose in the client application.

C#

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

Property Value

String

The name of the OpcSubscription used just for display purpose in the client application.

HasPendingChanges

Gets a value indicating whether there exists any pending change on the subscription which needs to be applied to take effect (ApplyChanges).

C#

```
public bool HasPendingChanges { get; }
```

Property Value

Boolean

The value true if there is at least one change not yet applied; otherwise the value false.

Id

Gets the server-assigned unique identifier for this OpcSubscription.

C#

```
public long Id { get; }
```

Property Value

Int64

The server-assigned unique identifier for this OpcSubscription.

Remarks

The identifier shall be unique for the entire server, not just for the Session, in order to allow the OpcSubscription to be transferred to another session.

IsCreated

Gets a value indicating whether this OpcSubscription has been created on the server.

C#

```
public bool IsCreated { get; }
```

Property Value

Boolean

The value true if the OpcSubscription has been created on the server; otherwise the value false.

IsPublishing

Gets a value indicating whether this OpcSubscription is receiving notifications.

C#

```
public bool IsPublishing { get; }
```

Property Value

Boolean

The value true if the OpcSubscription is receiving notifications; otherwise the value false.

KeepAliveCount

Gets or sets a value which defines the requested number of times the publishing timer have to expire without any notification being sent before the server have to send a keep-alive message to the client.

C#

```
public long KeepAliveCount { get; set; }
```

Property Value

Int64

A value of zero indicates that the server have to revise the smallest supported keep-alive count. The default value of this property is 10.

The value of this property configures the keep-alive counter that counts the number of consecutive publishing cycles in which there have been no notifications to report to the client. When the maximum keep-alive count is reached, a publish request is de-queued and used to return a keep-alive message. This keep-alive message informs the client that the subscription is still active. The maximum keep-alive count may be subsequently modified. If there are no publish requests queued, the server waits for the next one to be received and sends the keep-alive immediately without waiting for the next publishing cycle.

The value of the underlying property is an UInt32 which is covered using a Int64 to assure CLS compliance. In fact this restricts the value range of this property to MinValue and MaxValue.

Remarks

The server needs to be notified about changes to this property to take effect. To do so use the ApplyChanges method.

LastNotification

Gets the last notification received from the server.

C#

```
public OpcNotification LastNotification { get; }
```

Property Value

OpcNotification

An instance of the OpcNotification class which represents the last notfication received from the server or a null reference (Nothing in Visual Basic) if the OpcSubscription has not yet received a notification from the server or MaxMessageCount is less or equals zero.

LastNotificationTime

Gets the time the last notification which has been received from the server.

C#

```
public DateTime? LastNotificationTime { get; }
```

Property Value

Nullable<DateTime>

A DateTime identifying the instant of time the last notification which has been received from the server or a null reference (Nothing in Visual Basic) if the OpcSubscription has not yet received a notification from the server.

LifetimeCount

Gets or sets a value which defines the requested number of times the publishing timer have to expire without any publish request before the server have to delete this OpcSubscription.

C#

```
public long LifetimeCount { get; set; }
```

Property Value

Int64

The number of times the publishing timer have to expire without any publish request. A recommended value is at least a minimum of three times of the KeepAliveCount. The default value of this property is 100.

Remarks

The value of the underlying property is an UInt32 which is covered using a Int64 to assure CLS compliance. In fact this restricts the value range of this property to MinValue and MaxValue.

The server needs to be notified about changes to this property to take effect. To do so use the ApplyChanges method.

MaxMessageCount

Gets or sets the maximum number of notifications to keep in the cache.

C#

```
public int MaxMessageCount { get; set; }
```

Property Value

Int32

The maximum number of notification to keep in the cache. The default value of this property is 10.

MaxNotificationsPerPublish

Gets or sets a value which defines the maximum number of notifications that the client wants to receive in a single publish response.

C#

```
public long MaxNotificationsPerPublish { get; set; }
```

Property Value

Int64

The maximum number of notifications that the client wants to receive in a single publish response. The number of notifications per publish is the sum of MonitoredItems in a data change related notification and the number of events in an event notification. The default value of this property is 1000.

Remarks

A value of zero indicates that there is no limit.

The value of the underlying property is an UInt32 which is covered using a Int64 to assure CLS compliance. In fact this restricts the value range of this property to MinValue and MaxValue.

MinLifetimeInterval

Gets or sets a value which defines the minimum lifetime interval to apply in respect of the PublishingInterval and the LifetimeCount configured.

C#

```
public long MinLifetimeInterval { get; set; }
```

Property Value

Int64

The interval which should be a multiple of the PublishingInterval to assure that the LifetimeCount requested is sensible enough to get covered within the PublishingInterval.

MonitoredItems

Gets a read-only collection of OpcMonitoredItem instances associated with this OpcSubscription.

C#

```
public OpcMonitoredItemReadOnlyCollection MonitoredItems { get; }
```

Property Value

OpcMonitoredItemReadOnlyCollection

An instance of the OpcMonitoredItemReadOnlyCollection class.

Remarks

Use the monitored item related methods of this OpcSubscription to manipulate the items contained in the collection offered by this property.

Notifications

Gets a sequence of OpcNotification instances received from the server.

C#

```
public IEnumerable<OpcNotification> Notifications { get; }
```

Property Value

IEnumerable<OpcNotification>

A sequence of OpcNotification instances representing the notifications received from the server. The sequence is empty if the OpcSubscription has not yet received a notification from the server.

Remarks

The number of instances in the sequence is limited to the maximum number of messages in the cache (see MaxMessageCount).

Priority

Gets or sets a value which defines the relative priority of the subscription used to priorize notifications being send then more than one subscription is to be notified.

C#

```
public int Priority { get; set; }
```

Property Value

Int32

The relative priority of the subscription to notify.

Remarks

In case of subscriptions with equal priority the server shall send the notifications in a round-robin fashion. A client that does not require special priority settings should set this value to zero.

The value of the underlying property is a Byte which is covered using an Int32 to assure an even API level. In fact this restricts the value range of this property to MinValue and MaxValue.

The server needs to be notified about changes to this property to take effect. To do so use the ApplyChanges method.

PublishingInterval

Gets or sets the requested number of milliseconds of the interval within the server have to send cyclic notifications to the client.

C#

```
public int PublishingInterval { get; set; }
```

Property Value

Int32

The interval within the server have to send cyclic notifications to the client. A value of zero or negative indicates that the server have to revise the fastest supported publishing interval.

Remarks

The publishing interval of a subscription defines the cyclic rate at which the subscription executes. Each time it executes, it attempts to send a message to the client. Messages contain notifications that have not yet been reported to client.

The negotiated value of this property is used as the default sampling interval for the MonitoredItems assigned to this OpcSubscription.

The server needs to be notified about changes to this property to take effect. To do so use the ApplyChanges method.

PublishingIsEnabled

Gets or sets a value indicating whether publishing is enabled for this OpcSubscription.

C#

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

Property Value

Boolean

The value true if publishing is enabled for this OpcSubscription; otherwise the value false. The default value of this property is true.

Remarks

The value of this property does not affect the value of the monitoring mode property of the MonitoredItems.

The server needs to be notified about changes to this property to take effect. To do so use the ApplyChanges method.

PublishTime

Gets the time the last notification was published from the server.

C#

```
public DateTime? PublishTime { get; }
```

Property Value

Nullable<DateTime>

A DateTime identifying the instant of time the last notification has been published from the server or a null reference (Nothing in Visual Basic) if the OpcSubscription has not yet received a notification from the server or MaxMessageCount is less or equals zero.

ReceivedDataChangeCallback

Gets or sets the callback used to process data changes of the monitored items associated with this OpcSubscription.

C#

```
public OpcDataChangeReceivedCallback ReceivedDataChangeCallback { get; set; }
```

Property Value

OpcDataChangeReceivedCallback

A OpcDataChangeReceivedCallback used to process data change items. The value can also be a null reference (Nothing in Visual Basic).

Remarks

Use this property to directly handle all data change items without to rely on the chain of registered event handlers.

ReceivedEventCallback

Gets or sets the callback used to process events of the monitored items associated with this OpcSubscription.

C#

```
public OpcEventReceivedCallback ReceivedEventCallback { get; set; }
```

Property Value

OpcEventReceivedCallback

A OpcDataChangeReceivedCallback used to process event items. The value can also be a null reference (Nothing in Visual Basic).

Remarks

Use this property to directly handle all event items without to rely on the chain of registered event handlers.

SequenceNumber

Gets the sequence number of the last notification which has been received from the server.

C#

```
public long? SequenceNumber { get; }
```

Property Value

Nullable<Int64>

The SequenceNumber of the last notification which has been received from the server or a null reference (Nothing in Visual Basic) if the OpcSubscription has not yet received a notification from the server.

Tag

Gets or sets the object that contains additional user data about the subscription.

C#

```
public object Tag { get; set; }
```

Property Value

Object

An Object that contains additional user data about the subscription. The default is null (Nothing in Visual Basic).

TimestampsToReturn

Gets or sets the timestamps the server shall transmit for each monitored item.

C#

```
public OpcTimestampsToReturn TimestampsToReturn { get; set; }
```

Property Value

OpcTimestampsToReturn

One of the members defined by the OpcTimestampsToReturn enumeration. The default value of this property is Both.

Remarks

The “timestamps to return” criteria is only applied to OpcValue instances. This means that in case of monitoring events only event fields of the type OpcValue are effected by the value of this property.

The server needs to be notified about changes to this property to take effect. To do so use the ApplyChanges method.

UseMonitoredItemDataCache

Gets or sets a value indicating whether this OpcSubscription shall forward notifications received from the server to monitored item individual caches. This enables the monitored items to process the data changes and event fields on their own.

C#

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

Property Value

Boolean

The value true if the OpcMonitoredItem instances associated with this OpcSubscription receive their own item-related notification and can process the notification data on their own; otherwise the value false. The default value of this property is true.

Methods

AddMonitoredItem(IEnumerable<OpcMonitoredItem>)

Adds the monitored items specified by `items` to this OpcSubscription.

C#

```
public void AddMonitoredItem(IEnumerable<OpcMonitoredItem> items)
```

Parameters

`items` IEnumerable<OpcMonitoredItem>

A sequence of OpcMonitoredItem istances to add.

Exceptions

ArgumentNullException

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

Remarks

The server needs to be notified about the new monitored item to take effect. To do so use the ApplyChanges method.

AddMonitoredItem(OpcMonitoredItem)

Adds the monitored item specified by `item` to this OpcSubscription.

C#

```
public void AddMonitoredItem(OpcMonitoredItem item)
```

Parameters

`item` OpcMonitoredItem

The OpcMonitoredItem to add.

Exceptions

ArgumentNullException

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

AddMonitoredItem(OpcMonitoredItem[])

Adds the monitored items specified by `items` to this OpcSubscription.

C#

```
public void AddMonitoredItem(params OpcMonitoredItem[] items)
```

Parameters

`items` OpcMonitoredItem[]

An array of OpcMonitoredItem istances to add.

Exceptions

ArgumentNullException

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

Remarks

The server needs to be notified about the new monitored item to take effect. To do so use the ApplyChanges method.

AddMonitoredItem(OpcNodeId)

Adds a new monitored item which subscribes to changes on the Value attribute of the node specified by `nodeId`.

C#

```
public OpcMonitoredItem AddMonitoredItem(OpcNodeId nodeId)
```

Parameters

`nodeId` OpcNodeId

The OpcNodeId of the node its Value attribute is to be monitored.

Returns

OpcMonitoredItem

A new instance of the OpcMonitoredItem class setup with the information specified. The monitored item

will then monitor the node identified by the values passed to this method.

Exceptions

ArgumentException

The `nodeId` is empty.

ArgumentNullException

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

Remarks

The server needs to be notified about the new monitored item to take effect. To do so use the ApplyChanges method.

AddMonitoredItem(OpcNodeId, OpcAttribute)

Adds a new monitored item which subscribes to the `attribute` of the node specified by `nodeId`.

C#

```
public OpcMonitoredItem AddMonitoredItem(OpcNodeId nodeId, OpcAttribute attribute)
```

Parameters

`nodeId` OpcNodeId

The OpcNodeId of the node its `attribute` is to be monitored.

`attribute` OpcAttribute

The OpcAttribute to monitor.

Returns

OpcMonitoredItem

A new instance of the OpcMonitoredItem class setup with the information specified. The monitored item will then monitor the node identified by the values passed to this method.

Exceptions

ArgumentException

The `nodeId` is empty.

ArgumentNullException

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

Remarks

The server needs to be notified about the new monitored item to take effect. To do so use the ApplyChanges method.

AddMonitoredItem(OpcNodeId, OpcAttribute, OpcDataChangeFilter, OpcDataChangeReceivedEventHandler)

Adds a new monitored item which subscribes to the `attribute` of the node specified by `nodeId` using the further configured `filter`.

C#

```
public OpcMonitoredItem AddMonitoredItem(OpcNodeId nodeId, OpcAttribute attribute,
OpcDataChangeFilter filter, OpcDataChangeReceivedEventHandler received)
```

Parameters

`nodeId` OpcNodeId

The OpcNodeId of the node its `attribute` is to be monitored.

`attribute` OpcAttribute

The OpcAttribute to monitor.

`filter` OpcDataChangeFilter

The OpcDataChangeFilter to use to apply different conditions while monitoring the attribute.

`received` OpcDataChangeReceivedEventHandler

The OpcDataChangeReceivedEventHandler method to assign as the event handler of the DataChangeReceived event or a null reference (Nothing in Visual Basic) if there no event handler is to be assigned first.

Returns

OpcMonitoredItem

A new instance of the OpcMonitoredItem class setup with the information specified. The monitored item will then monitor the node identified by the values passed to this method.

Exceptions

ArgumentException

The `nodeId` is empty.

ArgumentNullException

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

Remarks

The server needs to be notified about the new monitored item to take effect. To do so use the ApplyChanges method.

AddMonitoredItem(OpcNodeId, OpcAttribute, OpcDataChangeReceivedEventHandler)

Adds a new monitored item which subscribes to the `attribute` of the node specified by `nodeId`.

C#

```
public OpcMonitoredItem AddMonitoredItem(OpcNodeId nodeId, OpcAttribute attribute,
OpcDataChangeReceivedEventHandler received)
```

Parameters

`nodeId` OpcNodeId

The OpcNodeId of the node its `attribute` is to be monitored.

`attribute` OpcAttribute

The OpcAttribute to monitor.

`received` OpcDataChangeReceivedEventHandler

The OpcDataChangeReceivedEventHandler method to assign as the event handler of the DataChangeReceived event or a null reference (Nothing in Visual Basic) if there no event handler is to be assigned first.

Returns

OpcMonitoredItem

A new instance of the OpcMonitoredItem class setup with the information specified. The monitored item will then monitor the node identified by the values passed to this method.

Exceptions

ArgumentException

The `nodeId` is empty.

ArgumentNullException

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

Remarks

The server needs to be notified about the new monitored item to take effect. To do so use the ApplyChanges method.

AddMonitoredItem(OpcNodeId, OpcAttribute, OpcEventFilter, OpcEventReceivedEventHandler)

Adds a new monitored item which subscribes to the `attribute` of the node specified by `nodeId` using the further configured `filter`.

C#

```
public OpcMonitoredItem AddMonitoredItem(OpcNodeId nodeId, OpcAttribute attribute,
OpcEventFilter filter, OpcEventReceivedEventHandler received)
```

Parameters

`nodeId` OpcNodeId

The OpcNodeId of the node its `attribute` is to be monitored.

`attribute` OpcAttribute

The OpcAttribute to monitor.

`filter` OpcEventFilter

The OpcEventFilter to use to apply different conditions while monitoring the attribute.

`received` OpcEventReceivedEventHandler

The OpcEventReceivedEventHandler method to assign as the event handler of the EventReceived event or a null reference (Nothing in Visual Basic) if there no event handler is to be assigned first.

Returns

OpcMonitoredItem

A new instance of the OpcMonitoredItem class setup with the information specified. The monitored item will then monitor the node identified by the values passed to this method.

Exceptions

ArgumentException

The `nodeId` is empty.

ArgumentNullException

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

Remarks

The server needs to be notified about the new monitored item to take effect. To do so use the ApplyChanges method.

AddMonitoredItem(OpcNodeId, OpcAttribute, OpcEventReceivedEventHandler)

Adds a new monitored item which subscribes to the `attribute` of the node specified by `nodeId`.

C#

```
public OpcMonitoredItem AddMonitoredItem(OpcNodeId nodeId, OpcAttribute attribute,
OpcEventReceivedEventHandler received)
```

Parameters

`nodeId` OpcNodeId

The OpcNodeId of the node its `attribute` is to be monitored.

`attribute` OpcAttribute

The OpcAttribute to monitor.

`received` OpcEventReceivedEventHandler

The OpcEventReceivedEventHandler method to assign as the event handler of the EventReceived event or a null reference (Nothing in Visual Basic) if there no event handler is to be assigned first.

Returns

OpcMonitoredItem

A new instance of the OpcMonitoredItem class setup with the information specified. The monitored item will then monitor the node identified by the values passed to this method.

Exceptions

ArgumentException

The `nodeId` is empty.

ArgumentNullException

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

Remarks

The server needs to be notified about the new monitored item to take effect. To do so use the ApplyChanges method.

AddMonitoredItem(OpcNodeId, OpcAttribute, OpcMonitoringFilter)

Adds a new monitored item which subscribes to the `attribute` of the node specified by `nodeId` using the further configured `filter`.

C#

```
public OpcMonitoredItem AddMonitoredItem(OpcNodeId nodeId, OpcAttribute attribute,
OpcMonitoringFilter filter)
```

Parameters

`nodeId` OpcNodeId

The OpcNodeId of the node its `attribute` is to be monitored.

`attribute` OpcAttribute

The OpcAttribute to monitor.

`filter` OpcMonitoringFilter

The OpcMonitoringFilter (either a OpcDataChangeFilter or a OpcEventFilter) to use to apply different conditions while monitoring the attribute.

Returns

OpcMonitoredItem

A new instance of the OpcMonitoredItem class setup with the information specified. The monitored item will then monitor the node identified by the values passed to this method.

Exceptions

ArgumentException

The `nodeId` is empty.

ArgumentNullException

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

Remarks

The server needs to be notified about the new monitored item to take effect. To do so use the ApplyChanges method.

AddMonitoredItem(OpcNodeId, OpcDataChangeFilter, OpcDataChangeReceivedEventHandler)

Adds a new monitored item which subscribes to changes on the Value attribute of the node specified by `nodeId` using the further configured `filter`.

C#

```
public OpcMonitoredItem AddMonitoredItem(OpcNodeId nodeId, OpcDataChangeFilter filter,
OpcDataChangeReceivedEventHandler received)
```

Parameters

`nodeId` OpcNodeId

The OpcNodeId of the node its Value attribute is to be monitored.

`filter` OpcDataChangeFilter

The OpcDataChangeFilter to use to apply different conditions while monitoring the attribute.

`received` OpcDataChangeReceivedEventHandler

The OpcDataChangeReceivedEventHandler method to assign as the event handler of the DataChangeReceived event or a null reference (Nothing in Visual Basic) if there no event handler is to be assigned first.

Returns

OpcMonitoredItem

A new instance of the OpcMonitoredItem class setup with the information specified. The monitored item will then monitor the node identified by the values passed to this method.

Exceptions

ArgumentException

The `nodeId` is empty.

ArgumentNullException

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

Remarks

The server needs to be notified about the new monitored item to take effect. To do so use the ApplyChanges method.

AddMonitoredItem(OpcNodeId, OpcDataChangeReceivedEventHandler)

Adds a new monitored item which subscribes to changes on the Value attribute of the node specified by `nodeId`.

C#

```
public OpcMonitoredItem AddMonitoredItem(OpcNodeId nodeId, OpcDataChangeReceivedEventHandler
received)
```

Parameters

`nodeId` OpcNodeId

The OpcNodeId of the node its Value attribute is to be monitored.

`received` OpcDataChangeReceivedEventHandler

The OpcDataChangeReceivedEventHandler method to assign as the event handler of the DataChangeReceived event or a null reference (Nothing in Visual Basic) if there no event handler is to be assigned first.

Returns

OpcMonitoredItem

A new instance of the OpcMonitoredItem class setup with the information specified. The monitored item will then monitor the node identified by the values passed to this method.

Exceptions

ArgumentException

The `nodeId` is empty.

ArgumentNullException

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

Remarks

The server needs to be notified about the new monitored item to take effect. To do so use the ApplyChanges method.

AddMonitoredItem(OpcNodeId, OpcEventFilter, OpcEventReceivedEventHandler)

Adds a new monitored item which subscribes to changes on the EventNotifier attribute of the node specified by `nodeId` using the further configured `filter`.

C#

```
public OpcMonitoredItem AddMonitoredItem(OpcNodeId nodeId, OpcEventFilter filter,
OpcEventReceivedEventHandler received)
```

Parameters

`nodeId` OpcNodeId

The OpcNodeId of the node its EventNotifier attribute is to be monitored.

`filter` OpcEventFilter

The OpcEventFilter to use to apply different conditions while monitoring the attribute.

`received` OpcEventReceivedEventHandler

The OpcEventReceivedEventHandler method to assign as the event handler of the EventReceived event or a null reference (Nothing in Visual Basic) if there no event handler is to be assigned first.

Returns

OpcMonitoredItem

A new instance of the OpcMonitoredItem class setup with the information specified. The monitored item will then monitor the node identified by the values passed to this method.

Exceptions

ArgumentException

The `nodeId` is empty.

ArgumentNullException

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

Remarks

The server needs to be notified about the new monitored item to take effect. To do so use the ApplyChanges method.

AddMonitoredItem(OpcNodeId, OpcEventReceivedEventHandler)

Adds a new monitored item which subscribes to events on the EventNotifier attribute of the node specified by `nodeId`.

C#

```
public OpcMonitoredItem AddMonitoredItem(OpcNodeId nodeId, OpcEventReceivedEventHandler
received)
```

Parameters

`nodeId` OpcNodeId

The OpcNodeId of the node its EventNotifier attribute is to be monitored.

`received` OpcEventReceivedEventHandler

The OpcEventReceivedEventHandler method to assign as the event handler of the EventReceived event or a null reference (Nothing in Visual Basic) if there no event handler is to be assigned first.

Returns

OpcMonitoredItem

A new instance of the OpcMonitoredItem class setup with the information specified. The monitored item

will then monitor the node identified by the values passed to this method.

Exceptions

ArgumentException

The `nodeId` is empty.

ArgumentNullException

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

Remarks

The server needs to be notified about the new monitored item to take effect. To do so use the ApplyChanges method.

AddMonitoredItem(OpcNodeId, OpcMonitoringFilter)

Adds a new monitored item which subscribes to changes on the Value attribute of the node specified by `nodeId` using the further configured `filter`.

C#

```
public OpcMonitoredItem AddMonitoredItem(OpcNodeId nodeId, OpcMonitoringFilter filter)
```

Parameters

`nodeId` OpcNodeId

The OpcNodeId of the node its Value attribute is to be monitored.

`filter` OpcMonitoringFilter

The OpcMonitoringFilter (either a OpcDataChangeFilter or a OpcEventFilter) to use to apply different conditions while monitoring the attribute.

Returns

OpcMonitoredItem

A new instance of the OpcMonitoredItem class setup with the information specified. The monitored item will then monitor the node identified by the values passed to this method.

Exceptions

ArgumentException

The `nodeId` is empty.

ArgumentNullException

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

Remarks

The server needs to be notified about the new monitored item to take effect. To do so use the ApplyChanges method.

ApplyChanges()

Applies any changes made on the current OpcSubscription or on one of its MonitoredItems using the according services of the server connected to.

C#

```
public void ApplyChanges()
```

Exceptions

InvalidOperationException

The current OpcSubscription has not yet been created (see IsCreated).

OpcException

The deletion, modification, creation of monitored items or the modification of the subscription has been failed. For more information see exception details. The following issues can lead to that exception: BadNothingToDo, BadTooManyOperations, BadTimestampsToReturnInvalid, BadSubscriptionIdInvalid and BadMonitoredItemIdInvalid.

ChangeMonitoringMode(OpcMonitoringMode)

Changes the monitoring mode of all MonitoredItems.

C#

```
public void ChangeMonitoringMode(OpcMonitoringMode monitoringMode)
```

Parameters

`monitoringMode` OpcMonitoringMode

The new OpcMonitoringMode to apply on all MonitoredItems of this OpcSubscription.

Exceptions

InvalidOperationException

The current OpcSubscription has not yet been created (see IsCreated).

OpcException

Changing the monitoring mode has failed. For more information see exception details. The following issues can lead to that exception: BadNothingToDo, BadTooManyOperations, BadSubscriptionIdInvalid and BadMonitoringModeInvalid.

Remarks

The server does not need to be notified about the change to take effect, because of this method directly operates on the server.

ChangeMonitoringMode(OpcMonitoringMode, IEnumerable<OpcMonitoredItem>)

Changes the monitoring mode of one or more monitored items.

C#

```
public void ChangeMonitoringMode(OpcMonitoringMode monitoringMode,
IEnumerable<OpcMonitoredItem> items)
```

Parameters

`monitoringMode` OpcMonitoringMode

The new OpcMonitoringMode to apply on the `items` specified.

`items` IEnumerable<OpcMonitoredItem>

A sequence of OpcMonitoredItem instances, assigned to this OpcSubscription, its monitoring mode is to be changed.

Exceptions

ArgumentNullException

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

InvalidOperationException

The current OpcSubscription has not yet been created (see IsCreated).

OpcException

Changing the monitoring mode has failed. For more information see exception details. The following issues can lead to that exception: BadNothingToDo, BadTooManyOperations, BadSubscriptionIdInvalid and BadMonitoringModeInvalid.

Remarks

The server does not need to be notified about the change to take effect, because of this method directly operates on the server.

ChangeMonitoringMode(OpcMonitoringMode, OpcMonitoredItem[])

Changes the monitoring mode of one or more monitored items.

C#

```
public void ChangeMonitoringMode(OpcMonitoringMode monitoringMode, params OpcMonitoredItem[]
items)
```

Parameters

`monitoringMode` OpcMonitoringMode

The new OpcMonitoringMode to apply on the `items` specified.

`items` OpcMonitoredItem[]

An array of OpcMonitoredItem instances, assigned to this OpcSubscription, its monitoring mode is to be changed.

Exceptions

ArgumentNullException

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

InvalidOperationException

The current OpcSubscription has not yet been created (see IsCreated).

OpcException

Changing the monitoring mode has failed. For more information see exception details. The following issues can lead to that exception: BadNothingToDo, BadTooManyOperations, BadSubscriptionIdInvalid and BadMonitoringModeInvalid.

Remarks

The server does not need to be notified about the change to take effect, because of this method directly operates on the server.

GetMonitoredItem(OpcDataChangeDataSetItem)

Retrieves the OpcMonitoredItem the `dataChange` specified belongs to.

C#

```
public OpcMonitoredItem GetMonitoredItem(OpcDataChangeDataSetItem dataChange)
```

Parameters

`dataChange` OpcDataChangeDataSetItem

The OpcDataChangeDataSetItem the according OpcMonitoredItem is to be determined from the MonitoredItems of this OpcSubscription.

Returns

OpcMonitoredItem

The OpcMonitoredItem instance which belongs to the `dataChange` specified or a null reference (Nothing in Visual Basic) if the monitored item referenced by the OpcDataChangeDataSetItem instance could not be determined.

Exceptions

ArgumentNullException

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

OnChanged(OpcSubscriptionChangedEventArgs)

Raises the Changed event of the OpcSubscription.

C#

```
protected virtual void OnChanged(OpcSubscriptionChangedEventArgs e)
```

Parameters

`e` OpcSubscriptionChangedEventArgs

The event data.

OnDataChangeReceived(OpcDataChangeReceivedEventArgs)

Raises the DataChangeReceived event of the OpcSubscription.

C#

```
protected virtual void OnDataChangeReceived(OpcDataChangeReceivedEventArgs e)
```

Parameters

`e` OpcDataChangeReceivedEventArgs

The event data.

OnEventReceived(OpcEventReceivedEventArgs)

Raises the EventReceived event of the OpcSubscription.

C#

```
protected virtual void OnEventReceived(OpcEventReceivedEventArgs e)
```

Parameters

e OpcEventReceivedEventArgs

The event data.

OnIsPublishingChanged(EventArgs)

Raises the IsPublishingChanged event of the OpcSubscription.

C#

```
protected virtual void OnIsPublishingChanged(EventArgs e)
```

Parameters

e EventArgs

The event data.

OnNotificationReceived(OpcNotificationReceivedEventArgs)

Raises the NotificationReceived event of the OpcSubscription.

C#

```
protected virtual void OnNotificationReceived(OpcNotificationReceivedEventArgs e)
```

Parameters

e OpcNotificationReceivedEventArgs

The event data.

RefreshConditions()

Forces the server to refresh all current conditions to synchronize the client with the server.

C#

```
public void RefreshConditions()
```

Exceptions

InvalidOperationException

The current OpcSubscription has not yet been created (see IsCreated).

OpcException

Refreshing conditions has failed. For more information see exception details. The following issues can lead to that exception: BadSubscriptionIdInvalid, BadRefreshInProgress and BadUserAccessDenied.

Remarks

The server does not need to be notified about the call to take effect, because of this method directly operates on the server.

The client will request a refresh of all condition instances that currenlty are in an interesting state (they have the retain property set to the value true). This includes previous states of a condition instance for which the server maintains branches as well.

A client would typically invoke this method when it initially connects to a server and following any situations, such as communication disruptions, in which it would require resynchronization with the server.

RemoveMonitoredItem(IEnumerable<OpcMonitoredItem>)

Removes the monitored items specified by `items` from the MonitoredItems of this OpcSubscription.

C#

```
public void RemoveMonitoredItem(IEnumerable<OpcMonitoredItem> items)
```

Parameters

`items` IEnumerable<OpcMonitoredItem>

A sequence of OpcMonitoredItem instances to remove.

Exceptions

ArgumentNullException

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

Remarks

The server needs to be notified about the orphaned monitored items to take effect. To do so use the ApplyChanges method.

RemoveMonitoredItem(OpcMonitoredItem)

Removes the monitored item specified by `item` from the MonitoredItems of this OpcSubscription.

C#

```
public void RemoveMonitoredItem(OpcMonitoredItem item)
```

Parameters

`item` OpcMonitoredItem

The OpcMonitoredItem to remove.

Exceptions

ArgumentNullException

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

Remarks

The server needs to be notified about the orphaned monitored item to take effect. To do so use the ApplyChanges method.

RemoveMonitoredItem(OpcMonitoredItem[])

Removes the monitored items specified by `items` from the MonitoredItems of this OpcSubscription.

C#

```
public void RemoveMonitoredItem(params OpcMonitoredItem[] items)
```

Parameters

`items` OpcMonitoredItem[]

An array of OpcMonitoredItem instances to remove.

Exceptions

ArgumentNullException

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

Remarks

The server needs to be notified about the orphaned monitored items to take effect. To do so use the ApplyChanges method.

StartPublishing()

Enables notifications being send from the server to the client for this OpcSubscription.

C#

```
public void StartPublishing()
```

Exceptions

InvalidOperationException

The current OpcSubscription has not yet been created (see IsCreated).

OpcException

Enabling notifications has failed. For more information see exception details. The following issues can lead to that exception: BadNothingToDo and BadTooManyOperations.

Remarks

The server does not need to be notified about the call to take effect, because of this method directly operates on the server.

StopPublishing()

Disables notifications being send from the server to the client for this OpcSubscription.

C#

```
public void StopPublishing()
```

Exceptions

InvalidOperationException

The current OpcSubscription has not yet been created (see IsCreated).

OpcException

Disabling notifications has failed. For more information see exception details. The following issues can lead to that exception: BadNothingToDo and BadTooManyOperations.

Remarks

The server does not need to be notified about the call to take effect, because of this method directly operates on the server.

ToString()

Returns a string that represents the current OpcSubscription.

C#

```
public override string ToString()
```

Returns

String

A string that represents the current OpcSubscription including the used DisplayName, the CurrentPublishingIsEnabled and the CurrentPublishingInterval.

Unsubscribe()

Deletes this OpcSubscription and all its MonitoredItems.

C#

```
public void Unsubscribe()
```

Exceptions

InvalidOperationException

The current OpcSubscription has not yet been created (see IsCreated).

OpcException

Changing the monitoring mode has failed. For more information see exception details. The following issues can lead to that exception: BadNothingToDo and BadTooManyOperations.

Remarks

The server does not need to be notified about the call to take effect, because of this method directly operates on the server.

Table of Contents