OpcApplicationInstance<TTransport, TSecurity> Members

Namespace: Opc.UaFx
Assemblies: Opc.UaFx.Advanced.dll, Opc.UaFx.Advanced.dll
The OpcApplicationInstance<t, y> type exposes the following members.

Constructors

OpcApplicationInstance(OpcApplicationType)

C#

```
protected OpcApplicationInstance(OpcApplicationType type)
```

Parameters

`type` OpcApplicationType

Exceptions

ArgumentNullException

Events

CertificateRequested

Occurs when the certificate of this application instance could not automatically determined using the information of this application instance. This event is when used to query the certificate to use for the application instance. Additionally use this event to determine the case that an appropriate instance certificate is missing and in case there AutoCreateCertificate of CertificateStores is equals true a new certificate is being created automatically after the event has been processed (and this may take some time).

C#

```
public event OpcCertificateRequestedEventHandler CertificateRequested
```

CertificateValidationFailed

Occurs when the validation of a opponent application instance certificate does not fulfill the requirements for valid and trusted certificates.

C#

```
public event OpcCertificateValidationFailedEventHandler CertificateValidationFailed
```

Remarks

This event can be used to complement the default validation of opponent application instance certificates. Using the event arguments it is possible to accept the certificate using different/custom circumstances.

Properties

ApplicationName

Gets or sets the name of the application.

C#

```
public virtual string ApplicationName { get; set; }
```

Property Value

String

The name of the application. The default value is “OPC Application” in case there is no entry assembly available; otherwise the value of the AssemblyTitleAttribute of the entry assembly.

Remarks

The name does not have to be unique. The value of this property is also used as the Cryptography.X509Certificates.X509Certificate2.SubjectName of the application certificate whenever an application certificate is to be created/looked-up (from the application certificates store).

ApplicationUri

Gets or sets the URI of the application.

C#

```
public Uri ApplicationUri { get; set; }
```

Property Value

Uri

The Uri or a null reference (Nothing in Visual Basic) which uniquely identifies the application instance.

Remarks

In case there the value of this property is a null reference (Nothing in Visual Basic) the URI of the application is determined by the subject alternate name of the application certificate used. In case there an application certificate is to be automatically created (if Certificate is unspecified and AutoCreateCertificate is equals true), this value is used to define the certificates subject alternate name.

If there exists already an application certificate the value of this property is replaced with the subject alternate name in the certificate. Nevertheless another URI has been manually configured.

Certificate

Gets or sets the application certificate representing the applications identity. Changing the application certificate using this property will change the PathType of the ApplicationStore to System.

C#

```
public X509Certificate2 Certificate { get; set; }
```

Property Value

Cryptography.X509Certificates.X509Certificate2

An instance of the Cryptography.X509Certificates.X509Certificate2 class used to identify the application instance.

CertificateStores

Gets the OpcCertificateStores instance used to maintain the different certificate stores used by the server application.

C#

```
public OpcCertificateStores CertificateStores { get; }
```

Property Value

OpcCertificateStores

An instance of the OpcCertificateStores class used by the IOpcApplicationInstance to setup and maintain the certificate stores and their certificates used by the application.

Configuration

Gets or sets an instance of the OpcApplicationConfiguration class which is used as a low-level representation of the code/file based application configuration.

C#

```
[CLSCompliant(false)]
public virtual OpcApplicationConfiguration Configuration { get; set; }
```

Property Value

OpcApplicationConfiguration

An instance of the OpcApplicationConfiguration class configured with the application specific setup to use. An assignment of a null reference (Nothing in Visual Basic) will lead to the creation of a new instance of the OpcApplicationConfiguration class with all defaults typically used by this type of applications.

Namespaces

Gets a read-only collection of namespaces used by the application instance.

C#

```
public abstract OpcReadOnlyNamespaceCollection Namespaces { get; }
```

Property Value

OpcReadOnlyNamespaceCollection

An instance of the OpcReadOnlyNamespaceCollection class with OpcNamespace items for each 'NamespaceUri' known/used.

ProductUri

Gets or sets the URI of the product.

C#

```
public Uri ProductUri { get; set; }
```

Property Value

Uri

The Uri or a null reference (Nothing in Visual Basic) which uniquely identifies the product.

Remarks

In case there the value of this property is a null reference (Nothing in Visual Basic) the URI of the product is determined by the AssemblyCompanyAttribute and AssemblyProductAttribute used.

Security

Gets the `TSecurity` instance used to maintain the different security options used by the application instance.

C#

```
public TSecurity Security { get; }
```

Property Value

y

An instance of the `TSecurity` class used by the application instance to setup and maintain the different security options.

SystemContext

Gets the OpcContext which is used to store the context sensitive data and configuration used by the current application instance.

C#

```
public abstract OpcContext SystemContext { get; }
```

Property Value

OpcContext

An instance of the OpcContext class which might be at least Empty.

Transport

Gets the `TTransport` instance used to maintain the different transport options used by the application instance.

C#

```
public TTransport Transport { get; }
```

Property Value

t

An instance of the `TTransport` class used by the application instance to setup and maintain the different transport options.

Methods

CreateSecurity()

When implemented in a derived class; creates and initializes a new instance of the `TSecurity` which is used for the Security property.

C#

```
protected abstract TSecurity CreateSecurity()
```

Returns

y

A new instance of the type specified by `TSecurity`.

CreateTransport()

When implemented in a derived class; creates and initializes a new instance of the `TTransport` which is used for the Transport property.

C#

```
protected abstract TTransport CreateTransport()
```

Returns

t

A new instance of the type specified by `TTransport`.

DenyIfIsDisposed()

Verifies whether the current IOpcApplicationInstance instance has been already disposed of. In the case there isDisposed is equals true an ObjectDisposedException will be thrown.

C#

```
protected void DenyIfIsDisposed()
```

Exceptions

ObjectDisposedException

The object has been disposed of.

Dispose()

Releases all resources used by the IOpcApplicationInstance.

C#

```
public void Dispose()
```

Dispose(Boolean)

Releases the unmanaged resources used by the IOpcApplicationInstance and optionally releases the managed resources.

C#

```
protected virtual void Dispose(bool disposing)
```

Parameters

`disposing` Boolean

The value true to release both managed and unmanaged resources; otherwise the value false to release only unmanaged resources.

IsAddressSupported(String)

Determines if the `address` specified is supported.

C#

```
public static bool IsAddressSupported(string address)
```

Parameters

`address` String

A string representing the Uri to evaluate.

Returns

Boolean

The value true if the `address` is supported; otherwise the value false.

Remarks

The value of the `address` parameter can also be a null reference (Nothing in Visual Basic) or an empty string. In both cases the result will be false.

IsAddressSupported(Uri)

Determines if the `address` specified is supported.

C#

```
public static bool IsAddressSupported(Uri address)
```

Parameters

`address` Uri

The Uri to evalatuate.

Returns

Boolean

The value true if the `address` is supported; otherwise the value false.

Remarks

The value the `address` parameter can be a null reference (Nothing in Visual Basic). In this case the result of the will be false.

OnCertificateRequested(OpcCertificateRequestedEventArgs)

Raises the CertificateRequested event of the application instance.

C#

```
protected virtual void OnCertificateRequested(OpcCertificateRequestedEventArgs e)
```

Parameters

`e` OpcCertificateRequestedEventArgs

The event data.

OnCertificateValidationFailed(OpcCertificateValidationFailedEventArgs)

Raises the CertificateValidationFailed event of the application instance.

C#

```
protected virtual void OnCertificateValidationFailed(OpcCertificateValidationFailedEventArgs
e)
```

Parameters

`e` OpcCertificateValidationFailedEventArgs

The event data.

Setup(OpcApplicationConfiguration)

C#

```
[CLSCompliant(false)]
protected virtual void Setup(OpcApplicationConfiguration configuration)
```

Parameters

`configuration` OpcApplicationConfiguration

Table of Contents

Constructors 1
OpcApplicationInstance(OpcApplicationType) 1
Events 1
CertificateRequested 1
CertificateValidationFailed 2
Properties 2
ApplicationName 2
ApplicationUri 2
Certificate 3
CertificateStores 3
Configuration 3
Namespaces 4
ProductUri 4
Security 5
SystemContext 5
Transport 5
Methods 5
CreateSecurity() 6
CreateTransport() 6
DenyIfIsDisposed() 6
Dispose() 6
Dispose(Boolean) 7
IsAddressSupported(String) 7
IsAddressSupported(Uri) 7
OnCertificateRequested(OpcCertificateRequestedEventArgs) 8
OnCertificateValidationFailed(OpcCertificateValidationFailedEventArgs) 8
Setup(OpcApplicationConfiguration) 9