

OpcDataTypeInfo Members

Namespace: Opc.UaFx

Assemblies: Opc.UaFx.Advanced.dll, Opc.UaFx.Advanced.dll

The [OpcDataTypeInfo](#) type exposes the following members.

Properties

BaseType

Gets the type from which the current [OpcDataTypeInfo](#) directly inherits.

C#

```
public virtual OpcDataTypeInfo BaseType { get; }
```

Property Value

[OpcDataTypeInfo](#)

An instance of the [OpcDataTypeInfo](#) which represents the type from which the current [OpcDataTypeInfo](#) directly inherits or a null reference (Nothing in Visual Basic) if the type does not inherit from a different [OpcDataTypeInfo](#).

ByteOrder

Gets the byte order of the data represented by the type.

C#

```
public virtual OpcByteOrder? ByteOrder { get; }
```

Property Value

[Nullable<OpcByteOrder>](#)

One of the members defined by the [OpcByteOrder](#) enumeration which overrides the [ByteOrder](#) of its [TypeDictionary](#) or a null reference (Nothing in Visual Basic) if the [OpcByteOrder](#) is to be used from the [TypeDictionary](#) of this type.

Category

Gets the category the [IOpcType](#) falls within.

C#

```
public OpcTypeCategory Category { get; }
```

Property Value

OpcTypeCategory

One of the members defined by the [OpcTypeCategory](#) enumeration.

Remarks

This implementation always returns [DataType](#).

CustomAttributes

Gets an array of custom attributes associated with this type.

C#

```
public virtual string[] CustomAttributes { get; }
```

Property Value

String[]

An array of [String](#) values used by the author of this type.

Remarks

Applications are not be required to understand these attributes in order to interpret an encoded instance of the type.

Documentation

Gets any semantic information that would help a human to understand what the type represents.

C#

```
public virtual string Documentation { get; }
```

Property Value

String

A human readable text that describes the type and that would help a human to understand what the type represents.

Empty

Gets the type to use if their is no specific type data available.

C#

```
public static OpcDataTypeInfo Empty { get; }
```

Property Value

[OpcDataTypeInfo](#)

An instance of the [OpcDataTypeInfo](#) class which refers to the [Empty](#).

Encoding

Gets the information used to identify the type of encoding used for the data of the type represented.

C#

```
public OpcEncoding Encoding { get; }
```

Property Value

[OpcEncoding](#)

An instance of the [OpcEncoding](#) class which represents the information used to encode data of the type or a null reference (Nothing in Visual Basic) if is no encoding information associated with the type.

EncodingMask

Gets the characteristics used to encode the existence of optional fields.

C#

```
public virtual OpcEncodingMask EncodingMask { get; }
```

Property Value

[OpcEncodingMask](#)

An instance of the [OpcEncodingMask](#) class.

Encodings

Gets information of all encodings supported by the type represented.

C#

```
public virtual OpcEncoding[] Encodings { get; }
```

Property Value

[OpcEncoding\[\]](#)

An array of [OpcEncoding](#) instances supported by the type.

IsArray

Gets a value indicating whether the type is an array.

C#

```
public virtual bool IsArray { get; }
```

Property Value

Boolean

The value true if the current type is an array; otherwise the value false.

IsEmpty

Gets a value indicating whether the current [OpcDataTypeInfo](#) represents a type to use if there is no specific type described.

C#

```
public virtual bool IsEmpty { get; }
```

Property Value

Boolean

The value true if the type does not declare a specific type; otherwise the value false.

IsEnum

Gets a value indicating whether the current [OpcDataTypeInfo](#) represents an enumeration.

C#

```
public virtual bool IsEnum { get; }
```

Property Value

Boolean

The value true if the current [OpcDataTypeInfo](#) represents an enumeration; otherwise the value false.

Remarks

The OPC UA handles enumerations as a numeric value that has a fixed set of valid values. The encoded value described by an enumeration is always an unsigned integer (see [UInt32](#)) with a fixed size.

IsOpaque

Gets a value indicating whether the current [OpcDataTypeInfo](#) describes a primitive fixed size type.

C#

```
public virtual bool IsOpaque { get; }
```

Property Value

[Boolean](#)

The value true if the current [OpcDataTypeInfo](#) represents an opaque type; otherwise the value false.

IsPrimitive

Gets a value indicating whether the [OpcDataTypeInfo](#) is one of the primitive types.

C#

```
public virtual bool IsPrimitive { get; }
```

Property Value

[Boolean](#)

The value true if the [OpcDataTypeInfo](#) is one of the primitive types; otherwise the value false.

IsStruct

Gets a value indicating whether the [OpcDataTypeInfo](#) is a structured type; that is not a value type.

C#

```
public virtual bool IsStruct { get; }
```

Property Value

[Boolean](#)

The value true if the [OpcDataTypeInfo](#) is a structured type; otherwise the value false.

Remarks

The OPC UA handles structures as a sequence of values. Each value in the sequence is called a field. A field may specify that zero, one or multiple instances of the type appear within the sequence described by the structure. Some fields have lengths that are not multiples of 8 bits. Several of these fields may appear in a sequence in a structure, however, the total number of bits used in the sequence shall be aligned on a byte boundary. A sequence of fields which do not line up on byte boundaries are specified from the least significant bit to the most significant bit. Sequences which are longer than one byte overflow from the

most significant bit to the first byte into the least significant bit of the next byte.

IsSystemType

Gets a value indicating whether the current [OpcDataTypeInfo](#) represents one of the predefined built-in data types defined by the OPC UA.

C#

```
public virtual bool IsSystemType { get; }
```

Property Value

[Boolean](#)

The value true if the current data type represents a OPC UA is a built-in type; otherwise the value false.

IsUnknown

Gets a value indicating whether the [OpcDataTypeInfo](#) acts as a placeholder for a referenced type its type declaration could not resolved.

C#

```
public virtual bool IsUnknown { get; }
```

Property Value

[Boolean](#)

The value true if the [OpcDataTypeInfo](#) acts as a placeholder for an unresolved type declaration; otherwise the value false.

Name

Gets a value which defines the non-localizable human-readable name of the type represented. A node which represents this [OpcDataTypeInfo](#) uses the [Name](#) as its [BrowseName](#).

C#

```
public OpcName Name { get; }
```

Property Value

[OpcName](#)

An instance of the [OpcName](#) class with the [String](#) used as the name of the type which does not unambiguously identify the [OpcDataTypeInfo](#).

Size

Gets the size of the type represented as the number of bytes required.

C#

```
public virtual long? Size { get; }
```

Property Value

[Nullable<Int64>](#)

The size of the type as the number of bytes required; if this [OpcDataTypeInfo](#) represents a type with a fixed size (see [IsOpaque](#)).

SizeInBits

Gets the size of the type represented as the number of bits required.

C#

```
public virtual long? SizeInBits { get; }
```

Property Value

[Nullable<Int64>](#)

The size of the type as the number of bits required; if this [OpcDataTypeInfo](#) represents a type with a fixed size (see [IsOpaque](#)).

TypeDictionary

Gets the the [OpcDataTypedictionary](#) which contains the data type declaration represented.

C#

```
public OpcDataTypedictionary Typedictionary { get; }
```

Property Value

[OpcDataTypedictionary](#)

An instance of the [OpcDataTypedictionary](#) class which defines the current [OpcDataTypeInfo](#).

TypeId

Gets a value which identifies the type represented. A node which represents this [OpcDataTypeInfo](#) uses the [TypeId](#) as its [NodeId](#).

C#

```
public OpcNodeId TypeId { get; }
```

Property Value

OpcNodeId

An instance of the [OpcNodeId](#) class used as the identifier of the type which unambiguously identifies the [OpcDataTypeInfo](#).

UnderlyingType

Gets the type which provides the implementation of the [OpcDataTypeInfo](#).

C#

```
public Type UnderlyingType { get; }
```

Property Value

Type

The [Type](#) which implements the type declared or a null reference (Nothing in Visual Basic) if there does not exist a declaration of the type described by this [OpcDataTypeInfo](#).

XmlName

Gets the [Name](#) of this [OpcDataTypeInfo](#) qualified with the [XmlNamespace](#) of the [TypeDictionary](#) which contains the data type declaration represented.

C#

```
public XmlQualifiedName XmlName { get; protected set; }
```

Property Value

XmlQualifiedName

An instance of the [XmlQualifiedName](#) class which identifies this [OpcDataTypeInfo](#) using a full qualified name which includes the [XmlNamespace](#) and [Name](#) of this type using the [Value](#) of the [OpcName](#) used.

Remarks

The [XmlQualifiedName](#) is used in OPC UA to refer to existing types when for example declaring fields.

Methods

GetArrayRank()

Gets the number of dimensions in an array.

C#

```
public virtual int GetArrayRank()
```

Returns

[Int32](#)

An integer that contains the number of dimensions in the current type.

Exceptions

[ArgumentException](#)

The current type is not an array.

GetElementType()

Returns the [OpcDataTypeInfo](#) of the object encompassed or referred to by the current array.

C#

```
public virtual OpcDataTypeInfo GetElementType()
```

Returns

[OpcDataTypeInfo](#)

The [OpcDataTypeInfo](#) of the object encompassed or referred to by the current array or null if the current [OpcDataTypeInfo](#) is not an array.

GetField(String)

Searches for the field with the specified [name](#).

C#

```
public OpcDataFieldInfo GetField(string name)
```

Parameters

[name](#) [String](#)

The [String](#) containing the name of the data field to get.

Returns

OpcDataFieldInfo

An instance of the [OpcDataFieldInfo](#) representing the field with the specified name, if found; otherwise a null reference (Nothing in Visual Basic).

Exceptions

[ArgumentNullException](#)

The **name** is a null reference (Nothing in Visual Basic).

Remarks

The **name** specified is compared using [Ordinal](#).

GetFields()

Returns all the fields of the current [OpcDataTypeInfo](#).

C#

```
public OpcDataFieldInfo[] GetFields()
```

Returns

[OpcDataFieldInfo\[\]](#)

An array of [OpcDataFieldInfo](#) objects representing all the fields defined for the current [OpcDataTypeInfo](#) or an empty array of type [OpcDataFieldInfo](#) if no fields are defined for the current [OpcDataTypeInfo](#).

GetFieldsCore()

When overridden in a derived class, searches for the fields defined for the current [OpcDataTypeInfo](#).

C#

```
protected virtual IEnumerable<OpcDataFieldInfo> GetFieldsCore()
```

Returns

[IEnumerable<OpcDataFieldInfo>](#)

A sequence of [OpcDataFieldInfo](#) objects representing all fields defined for the current [OpcDataTypeInfo](#).

GetUnderlyingType()

Retrieves the [Type](#) which provides the implementation of the [OpcDataTypeInfo](#).

C#

```
protected virtual Type GetUnderlyingType()
```

Returns

Type

The Type which implements the type declared or a null reference (Nothing in Visual Basic) if there does not exist a declaration of the type described by this OpcDataTypeInfo.

Remarks

To determine the underlying Type this method queries the Type using the TypeResolve event. If there is no Type information provided the method queries the types defined by the foundation stack using the TypeId and Encoding of this OpcDataTypeInfo. If there is still no Type determined the method tries to resolve the Type using the Name to construct type names of types defined in the 'Opc.Ua' and 'System' namespace.

GetXmlName(String, OpcDataTypeIdDictionary)

Retrieves the XmlQualifiedName for the name and typeIdDictionary specified.

C#

```
public static XmlQualifiedName GetXmlName(string name, OpcDataTypeIdDictionary typeIdDictionary)
```

Parameters

name String

The name of the entity for which the XmlQualifiedName is to be created. The value is used for the XmlQualifiedName.Name.

typeIdDictionary OpcDataTypeIdDictionary

The OpcDataTypeIdDictionary its XmlNamespace is used to define the XmlQualifiedName.Namespace to retrieve. This can be a null reference (Nothing in Visual Basic) as well.

Returns

XmlQualifiedName

A new instance of the XmlQualifiedName which uses the name for the XmlQualifiedName.Name and the typeIdDictionary (if specified) for the XmlQualifiedName.Namespace.

GetXmlName(String, OpcNamespace)

Retrieves the XmlQualifiedName for the name and namespace specified.

C#

```
public static XmlQualifiedName GetXmlName(string name, OpcNamespace namespace)
```

Parameters

name String

The name of the entity for which the [XmlQualifiedName](#) is to be created. The value is used for the [XmlQualifiedName.Name](#).

namespace [OpcNamespace](#)

The [OpcNamespace](#) its [Uri](#) or [Value](#) is used to define the [XmlQualifiedName.Namespace](#) to retrieve. This can be a null reference (Nothing in Visual Basic) as well.

Returns

[XmlQualifiedName](#)

A new instance of the [XmlQualifiedName](#) which uses the **name** for the [XmlQualifiedName.Name](#) and the **namespace** (if specified) for the [XmlQualifiedName.Namespace](#).

GetXmlName(String, String)

Retrieves the [XmlQualifiedName](#) for the **name** and **xmlNamespace** specified.

C#

```
public static XmlQualifiedName GetXmlName(string name, string xmlNamespace)
```

Parameters

name String

The name of the entity for which the [XmlQualifiedName](#) is to be created. The value is used for the [XmlQualifiedName.Name](#).

xmlNamespace String

The namespace within the entity has been declared and which is to be referenced by the [XmlQualifiedName.Namespace](#) to retrieve. This value can be a null reference (Nothing in Visual Basic) or equals to [Empty](#).

Returns

[XmlQualifiedName](#)

A new instance of the [XmlQualifiedName](#) which uses the **name** for the [XmlQualifiedName.Name](#) and the **xmlNamespace** (if specified) for the [XmlQualifiedName.Namespace](#).

MakeArrayType()

Returns a [OpcDataTypeInfo](#) object representing a one-dimensional array of the current type, with a lower bound of zero.

C#

```
public OpcDataTypeInfo MakeArrayType()
```

Returns

[OpcDataTypeInfo](#)

A [OpcDataTypeInfo](#) object representing a one-dimensional array of the current type, with a lower bound of zero.

MakeArrayType(Int32)

Returns a [OpcDataTypeInfo](#) object representing an array of the current type, with the specified number of dimensions.

C#

```
public OpcDataTypeInfo MakeArrayType(int rank)
```

Parameters

rank [Int32](#)

The number of dimensions for the array. This number must be less than or equal to 32.

Returns

[OpcDataTypeInfo](#)

An object representing an array of the current type, with the specified number of dimensions.

Exceptions

[ArgumentOutOfRangeException](#)

The **rank** is not between 1 and 32 (inclusive).

ToString()

Returns a [String](#) representing the [Name](#) or the [UnderlyingType](#) of the current [OpcDataTypeInfo](#).

C#

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

Returns

[String](#)

A [String](#) representing the [Name](#) or the [UnderlyingType](#) of the current [OpcDataTypeInfo](#).

Table of Contents

Properties	1
BaseType	1
ByteOrder	1
Category	1
CustomAttributes	2
Documentation	2
Empty	2
Encoding	3
EncodingMask	3
Encodings	3
IsArray	4
IsEmpty	4
IsEnum	4
IsOpaque	5
IsPrimitive	5
IsStruct	5
IsSystemType	6
IsUnknown	6
Name	6
Size	7
SizeInBits	7
TypeDictionary	7
TypeId	7
UnderlyingType	8
XmlName	8
Methods	8
GetArrayRank()	9
GetElementType()	9
GetField(String)	9
GetFields()	10
GetFieldsCore()	10
GetUnderlyingType()	10
GetXmlName(String, OpcDataTypeDictionary)	11
GetXmlName(String, OpcNamespace)	11
GetXmlName(String, String)	12
MakeArrayType()	12
MakeArrayType(Int32)	13
ToString()	13