OpcTypeSystem<T> Members

Namespace: Opc.UaFx
Assemblies: Opc.UaFx.Advanced.dll, Opc.UaFx.Advanced.dll
The OpcTypeSystem<T> type exposes the following members.

Constructors

OpcTypeSystem(OpcTypeSystemScopes)

Initializes a new instance of the OpcTypeSystem´1 class using the `scope` specified without any fallback type system.

C#

```
protected OpcTypeSystem(OpcTypeSystemScopes scope)
```

Parameters

`scope` OpcTypeSystemScopes

The OpcTypeSystemScopes the new type system covers.

OpcTypeSystem(OpcTypeSystemScopes, OpcTypeSystem<T>)

Initializes a new instance of the OpcTypeSystem´1 class using the `scope` and `fallbackTypeSystem` specified.

C#

```
protected OpcTypeSystem(OpcTypeSystemScopes scope, OpcTypeSystem<T> fallbackTypeSystem)
```

Parameters

`scope` OpcTypeSystemScopes

The OpcTypeSystemScopes the new type system covers.

`fallbackTypeSystem` OpcTypeSystem`1

The type system used as the fallback system to query type information which is referenced in the new OpcTypeSystem´1 but is not declared.

Properties

FallbackTypeSystem

Gets the type system used as the fallback system to query type information which is referenced in the current OpcTypeSystem´1 but is not declared.

C#

```
public virtual OpcTypeSystem<T> FallbackTypeSystem { get; }
```

Property Value

OpcTypeSystem`1

An instance of the OpcTypeSystem´1 class used to query type information which is referenced but not declared within the current OpcTypeSystem´1 or a null reference (Nothing in Visual Basic) if the current OpcTypeSystem´1 is independent from a different type system. Such type systems typically declare types which only reference others declared in the same system as well.

IsEmpty

Gets a value indicating whether the current OpcTypeSystem´1 represents a type system to use if there is no specific type system described.

C#

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

Property Value

Boolean

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

Scope

Gets the scope of types covered by the current OpcTypeSystem´1.

C#

```
public OpcTypeSystemScopes Scope { get; }
```

Property Value

OpcTypeSystemScopes

One of the members defined by the OpcTypeSystemScopes enumeration.

Methods

GetType(OpcName)

Retrieves the `T` object which is known under the `name` specified.

C#

```
public T GetType(OpcName name)
```

Parameters

`name` OpcName

The OpcName of the `T` object to retrieve.

Returns

T

The `T` object its Name is equals to the `name` specified; otherwise a null reference (Nothing in Visual Basic).

Exceptions

ArgumentNullException

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

GetType(OpcNodeId)

Retrieves the `T` object which declares the type which is identified by the `typeId` specified.

C#

```
public T GetType(OpcNodeId typeId)
```

Parameters

`typeId` OpcNodeId

The OpcNodeId which identifies the `T` to retrieve.

Returns

T

The `T` object which is known under the `typeId` specified or a null reference (Nothing in Visual Basic) if there doesn't exist a known `T` object which is associated with the `typeId` specified.

Exceptions

ArgumentNullException

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

GetType(String)

Retrieves the `T` object which is known under the `name` specified.

C#

```
public T GetType(string name)
```

Parameters

`name` String

The String to use to identify the `T` object to retrieve.

Returns

T

The `T` object its Name is equals (regarding its Value) to the `name` specified; otherwise a null reference (Nothing in Visual Basic).

Exceptions

ArgumentException

The `name` is equals Empty.

ArgumentNullException

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

GetType(Type)

Retrieves the `T` object which declares the type implemented by the `underlyingType` specified.

C#

```
public T GetType(Type underlyingType)
```

Parameters

`underlyingType` Type

The Type which implements the `T` to retrieve.

Returns

T

The `T` object which declares the `underlyingType` specified or a null reference (Nothing in Visual Basic) if there isn't a `T` object associated with the `underlyingType` specified.

Exceptions

ArgumentNullException

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

GetTypeCore(OpcName)

Retrieves the `T` object which is known under the `name` specified.

C#

```
protected abstract T GetTypeCore(OpcName name)
```

Parameters

`name` OpcName

The OpcName of the `T` object to retrieve.

Returns

T

The `T` object its Name is equals to the `name` specified; otherwise a null reference (Nothing in Visual Basic).

Remarks

It is already assured that the passed `name` is not a null reference (Nothing in Visual Basic).

GetTypeCore(OpcNodeId)

Retrieves the `T` object which declares the type which is identified by the `typeId` specified.

C#

```
protected abstract T GetTypeCore(OpcNodeId typeId)
```

Parameters

`typeId` OpcNodeId

The OpcNodeId which identifies the `T` to retrieve.

Returns

T

The `T` object which is known under the `typeId` specified or a null reference (Nothing in Visual Basic) if there doesn't exist a known `T` object which is associated with the `typeId` specified.

Remarks

It is already assured that the passed `typeId` is not a null reference (Nothing in Visual Basic).

GetTypeCore(String)

Retrieves the `T` object which is known under the `name` specified.

C#

```
protected abstract T GetTypeCore(string name)
```

Parameters

`name` String

The String to use to identify the `T` object to retrieve.

Returns

T

The `T` object its Name is equals (regarding its Value) to the `name` specified; otherwise a null reference (Nothing in Visual Basic).

Remarks

It is already assured that the passed `name` is not a null reference (Nothing in Visual Basic) nor equals Empty.

GetTypeCore(Type)

Retrieves the `T` object which declares the type implemented by the `underlyingType` specified.

C#

```
protected abstract T GetTypeCore(Type underlyingType)
```

Parameters

`underlyingType` Type

The Type which implements the `T` to retrieve.

Returns

T

The `T` object which declares the `underlyingType` specified or a null reference (Nothing in Visual Basic) if there isn't a `T` object associated with the `underlyingType` specified.

Remarks

It is already assured that the passed `underlyingType` is not a null reference (Nothing in Visual Basic).

GetTypes()

Retrieves all `T` objects offered by the OpcTypeSystem´1.

C#

```
public T[] GetTypes()
```

Returns

T

An array that contains all `T` objects that are offered by the OpcTypeSystem´1.

GetTypesCore()

Retrieves all `T` objects offered by the current OpcTypeSystem´1 .

C#

```
protected abstract T[] GetTypesCore()
```

Returns

T

An array that contains all `T` objects that are offered by the OpcTypeSystem´1.

Table of Contents