

PlcNull Members

Namespace: IPS7Lnk.Advanced

Assemblies: IPS7LnkNet.Advanced.dll, IPS7LnkNet.Advanced.dll

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

Constructors

PlcNull(PlcAddress)

Initializes a new instance of the [PlcNull](#) class using the specified [address](#).

C#

```
public PlcNull(PlcAddress address)
```

Parameters

[address](#) [PlcAddress](#)

The address where the logical null value is to be located.

Exceptions

[ArgumentNullException](#)

The [address](#) is a null reference (Nothing in Visual Basic).

PlcNull(PlcAddress, Int32)

Initializes a new instance of the [PlcNull](#) class using the specified [address](#) and [length](#).

C#

```
public PlcNull(PlcAddress address, int length)
```

Parameters

[address](#) [PlcAddress](#)

The address where the logical null value is to be located.

[length](#) [Int32](#)

The reserved amount of address specific Bit's, Byte's, Word's or DWord's represented as null values.

Exceptions

[ArgumentNullException](#)

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

`ArgumentOutOfRangeException`

The `length` is lower than one.

Properties

Description

Gets or sets more meaningful information about the value and its usage than only using the `Name` property.

C#

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

Property Value

`String`

A `String` value containing additional information about the value and its usage.

Name

Gets the name of the value represented.

C#

```
public PlcName Name { get; }
```

Property Value

`PlcName`

An instance of the `PlcName` class containing the name of the value represented.

Tag

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

C#

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

Property Value

`Object`

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

Type

Gets the type of value represented by the PLC value.

C#

```
public PlcType Type { get; }
```

Property Value

PlcType

The [PlcType](#) of the value represented by the PLC value.

Methods

IsValidType(Object)

Determines whether a specified value is acceptable for this [PlcNull](#).

C#

```
public bool IsValidType(object value)
```

Parameters

value [Object](#)

The value to check.

Returns

[Boolean](#)

The value true, if the specified [value](#) is the [Type](#) or an acceptable derived type; otherwise the value false.

IsValidValue(Object)

Determines whether the provided value is accepted for the type of PLC value through basic type checking and also potentially if it is within the allowed range of value for that type.

C#

```
public bool IsValidValue(object value)
```

Parameters

value [Object](#)

The value to check.

Returns

Boolean

The value true, if the specified **value** is acceptable and is of the correct type or a derived type; otherwise the value false.

Relocate(Int32)

Relocates the **PlcNull** using the specified offset. The original **PlcNull** remains unmodified.

C#

```
public IPlcValue Relocate(int operandNumber)
```

Parameters

operandNumber **Int32**

The operand number offset used to adjust the address of the **Type**.

Returns

IPlcValue

A new instance of the **PlcNull** configured with the same metadata as this instance but relocated using the specified offset.

Exceptions

ArgumentOutOfRangeException

The offset specified by **operandNumber** result into a new value that would be out of the bounds defined by **MinOperandNumber** or **MaxOperandNumber**.

InvalidOperationException

It is not possible to relocate relative types.

Relocate(Int32, Int32)

Relocates the **PlcNull** using the specified offset. The original **PlcNull** remains unmodified.

C#

```
public IPlcValue Relocate(int operandNumber, int byteNumber)
```

Parameters

operandNumber **Int32**

The operand number offset used to adjust the address of the **Type**.

byteNumber Int32

The byte number offset used to adjust the address of the [Type](#).

Returns

IPlcValue

A new instance of the [PlcNull](#) configured with the same metadata as this instance but relocated using the specified offset.

Exceptions

ArgumentOutOfRangeException

One of the offsets specified by [operandNumber](#) or [byteNumber](#) result into a new value that would be out of the bounds defined by [MinOperandNumber](#), [MaxOperandNumber](#), [MinByteNumber](#) and [MaxByteNumber](#).

InvalidOperationException

It is not possible to relocate relative types.

Relocate(Int32, Int32, Int32)

Relocates the [PlcNull](#) using the specified offset. The original [PlcNull](#) remains unmodified.

C#

```
public IPlcValue Relocate(int operandNumber, int byteNumber, int bitNumber)
```

Parameters

operandNumber Int32

The operand number offset used to adjust the address of the [Type](#).

byteNumber Int32

The byte number offset used to adjust the address of the [Type](#).

bitNumber Int32

The bit number offset used to adjust the address of the [Type](#).

Returns

IPlcValue

A new instance of the [PlcNull](#) configured with the same metadata as this instance but relocated using the specified offset.

Exceptions

ArgumentOutOfRangeException

One of the offsets specified by `operandNumber`, `byteNumber` or `bitNumber` result into a new value that would be out of the bounds defined by `MinOperandNumber`, `MaxOperandNumber`, `MinByteNumber`, `MaxByteNumber`, `MinBitNumber` or `MaxBitNumber`.

InvalidOperationException

It is not possible to relocate relative types.

Relocate(PlcAddress)

Relocates the `PlcNull` using the specified `address`. The original `PlcNull` remains unmodified.

C#

```
public IPlcValue Relocate(PlcAddress address)
```

Parameters

`address` `PlcAddress`

The `PlcAddress` to that the type is to be relocated.

Returns

`IPlcValue`

A new instance of the `PlcNull` configured with the same metadata as this instance but relocated using the specified `address`.

Exceptions

ArgumentNullException

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

InvalidOperationException

It is not possible to relocate absolute object types without relative type information.

Remarks

The `RawType` of the `address` specified needs to be the same as defined by the address of the `Type` of the PLC value.

ToString()

Converts the value to its string representation.

C#

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

Returns

[String](#)

A string that contains the value.

ValidateValue(Object)

Validates the specified `value` whether it can be assigned to this [PlcValue](#)'1.

C#

```
public void ValidateValue(object value)
```

Parameters

`value` [Object](#)

The value to validate.

Exceptions

[ArgumentException](#)

The `value` or its type is not valid.

Table of Contents

Constructors	1
PlcNull(PlcAddress)	1
PlcNull(PlcAddress, Int32)	1
Properties	2
Description	2
Name	2
Tag	2
Type	3
Methods	3
IsValidType(Object)	3
IsValidValue(Object)	3
Relocate(Int32)	4
Relocate(Int32, Int32)	4
Relocate(Int32, Int32, Int32)	5
Relocate(PlcAddress)	6
ToString()	6
ValidateValue(Object)	7