Bit Members

Namespace: Opc.UaFx
Assemblies: Opc.UaFx.Advanced.dll
The Bit struct defines the following members.

Fields

High

Represents the Bit value high.

C#

```
public static readonly int High
```

Field Value
Int32

HighString

Represents the Bit value high as a string.

C#

```
public static readonly string HighString
```

Field Value
String

Low

Represents the Bit value low.

C#

```
public static readonly int Low
```

Field Value
Int32

LowString

Represents the Bit value low as a string.

C#

```
public static readonly string LowString
```

Field Value
String

Methods

CompareTo(Bit)

Compares the current Bit with another Bit.

C#

```
public int CompareTo(Bit other)
```

Parameters

`other` Bit

The Bit to compare with this Bit.

Returns

Int32

A 32-bit signed integer that indicates the relative order of the objects being compared (CompareTo()).

CompareTo(Object)

Compares the current Bit with the `other`.

C#

```
public int CompareTo(object other)
```

Parameters

`other` Object

The Bit to compare with this Bit.

Returns

Int32

A 32-bit signed integer that indicates the relative order of the objects being compared (CompareTo(Object)).

Equals(Bit)

Determines whether the specified `other` is equal to this Bit.

C#

```
public bool Equals(Bit other)
```

Parameters

`other` Bit

The Bit to compare to the current Bit.

Returns

Boolean

The value true if the specified Bit is equal to the current Bit; otherwise the value false.

Equals(Object)

Determines whether the specified `other` is equal to this Bit.

C#

```
public override bool Equals(object other)
```

Parameters

`other` Object

The Bit to compare to the current Bit.

Returns

Boolean

The value true if the specified Bit is equal to the current Bit; otherwise the value false.

GetHashCode()

Retrieves a hash code for this Bit.

C#

```
public override int GetHashCode()
```

Returns

Int32

An Int32 that contains the hash code for the Bit.

GetTypeCode()

Returns the type code for the Bit value type.

C#

```
public TypeCode GetTypeCode()
```

Returns

TypeCode

The enumerated constant Boolean.

Parse(String)

Converts the specified string representation of a logical value to its Bit equivalent.

C#

```
public static Bit Parse(string value)
```

Parameters

`value` String

A string containing the value to convert.

Returns

Bit

The High if value is equivalent to HighString; Low if value is equivalent to LowString.

Exceptions

ArgumentNullException

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

FormatException

The `value` is not equivalent to HighString or LowString.

ToString()

Converts the value of this instance to its equivalent string representation (either “High” or “Low”).

C#

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

Returns

String

High“ (the value of the HighString field) if the value of this instance is true, or “Low” (the value of the LowString field) if the value of this instance is false.

ToString(IFormatProvider)

Converts the value of this instance to its equivalent string representation (either “High” or “Low”).

C#

```
public string ToString(IFormatProvider provider)
```

Parameters

`provider` IFormatProvider

(Reserved) An IFormatProvider object.

Returns

String

HighString if the value of this instance is true, or LowString if the value of this instance is false.

TryParse(String, out Bit)

Tries to convert the specified string representation of a logical value to its Bit equivalent. A return value indicates whether the conversion succeeded or failed.

C#

```
public static bool TryParse(string value, out Bit result)
```

Parameters

`value` String

A String containing the value to convert.

`result` Bit

When this method returns, if the conversion succeeded, contains High if `value` is equal to HighString or Low if `value` is equal to LowString. If the conversion failed, contains false. The conversion fails if `value` is a null reference (Nothing in Visual Basic) or is not equal to the value of either the HighString or LowString field.

Returns

Boolean

The value true if `value` was converted successfully; otherwise the value false.

Operators

Equality(Bit, Bit)

Returns a value indicating whether two values of Bit are equal.

C#

```
public static bool operator ==(Bit left, Bit right)
```

GreaterThan(Bit, Bit)

Determines whether the first specified Bit value is greater than the second specified Bit value.

C#

```
public static bool operator>(Bit left, Bit right)
```

GreaterThanOrEqual(Bit, Bit)

Determines whether the first specified Bit value is greater than or equal to the second specified Bit value.

C#

```
public static bool operator >=(Bit left, Bit right)
```

Implicit(Bit to Boolean)

Converts a Bit to a Boolean value.

C#

```
public static implicit operator bool (Bit value)
```

Implicit(Bit to Int32)

Converts a Bit to a Int32 value.

C#

```
public static implicit operator int (Bit value)
```

Implicit(Boolean to Bit)

Converts a Boolean to a Bit value.

C#

```
public static implicit operator Bit(bool value)
```

Implicit(Byte to Bit)

Converts a Byte to a Bit value.

C#

```
public static implicit operator Bit(byte value)
```

Implicit(Int16 to Bit)

Converts a Int16 to a Bit value.

C#

```
public static implicit operator Bit(short value)
```

Implicit(Int32 to Bit)

Converts a Int32 to a Bit value.

C#

```
public static implicit operator Bit(int value)
```

Implicit(Int64 to Bit)

Converts a Int64 to a Bit value.

C#

```
public static implicit operator Bit(long value)
```

Implicit(SByte to Bit)

Converts a SByte to a Bit value.

C#

```
public static implicit operator Bit(sbyte value)
```

Implicit(UInt16 to Bit)

Converts a UInt16 to a Bit value.

C#

```
public static implicit operator Bit(ushort value)
```

Implicit(UInt32 to Bit)

Converts a UInt32 to a Bit value.

C#

```
public static implicit operator Bit(uint value)
```

Implicit(UInt64 to Bit)

Converts a UInt64 to a Bit value.

C#

```
public static implicit operator Bit(ulong value)
```

Inequality(Bit, Bit)

Returns a value indicating whether two values of Bit are not equal.

C#

```
public static bool operator !=(Bit left, Bit right)
```

LessThan(Bit, Bit)

Determines whether the first specified Bit value is less than the second specified Bit value.

C#

```
public static bool operator <(Bit left, Bit right)
```

LessThanOrEqual(Bit, Bit)

Determines whether the first specified Bit value is less than or equal to the second Bit value.

C#

```
public static bool operator <=(Bit left, Bit right)
```

Table of Contents

Fields 1
High 1
HighString 1
Low 1
LowString 1
Methods 2
CompareTo(Bit) 2
CompareTo(Object) 2
Equals(Bit) 3
Equals(Object) 3
GetHashCode() 3
GetTypeCode() 4
Parse(String) 4
ToString() 4
ToString(IFormatProvider) 5
TryParse(String, out Bit) 5
Operators 6
Equality(Bit, Bit) 6
GreaterThan(Bit, Bit) 6
GreaterThanOrEqual(Bit, Bit) 6
Implicit(Bit to Boolean) 6
Implicit(Bit to Int32) 6
Implicit(Boolean to Bit) 7
Implicit(Byte to Bit) 7
Implicit(Int16 to Bit) 7
Implicit(Int32 to Bit) 7
Implicit(Int64 to Bit) 7
Implicit(SByte to Bit) 7
Implicit(UInt16 to Bit) 8
Implicit(UInt32 to Bit) 8
Implicit(UInt64 to Bit) 8
Inequality(Bit, Bit) 8
LessThan(Bit, Bit) 8
LessThanOrEqual(Bit, Bit) 8