

# Namespace Cobilas

## Classes

### [TypeUtilitarian](#)

Utility static class to obtain type or assembly.

## Structs

### [Interrupter](#)

Represents a list of switches.

# Struct Interrupter

Namespace: [Cobilas](#)

Assembly: Cobilas.Core.dll

Represents a list of switches.

```
[Serializable]
public struct Interrupter : IDisposable
```

Implements

[IDisposable](#)

Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetType\(\)](#)

Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#)

## Constructors

**Interrupter(int)**

Only one switch specifying the index will be used, the others will remain at false value.

```
public Interrupter(int Capacity)
```

Parameters

**Capacity int**

How many switches.

## Interrupter(int, bool)

Only one switch specifying the index will be used, the others will remain at false value.

```
public Interrupter(int Capacity, bool UseASwitch)
```

### Parameters

Capacity [int](#)

How many switches.

UseASwitch [bool](#)

Allows you to use one switch at a time.

## Properties

### CurrentIndex

Returns the current switch index.

```
public int CurrentIndex { get; }
```

### Property Value

[int](#)

### this[int]

Gets or sets a switch when specifying an index.

```
public bool this[int Index] { get; set; }
```

### Parameters

Index [int](#)

Property Value

[bool ↗](#)

## UseASwitch

This property allows the exchange of a single switch for multiple switches and vice versa.

```
public bool UseASwitch { get; set; }
```

Property Value

[bool ↗](#)

## Methods

### Dispose()

Resource disposal.

```
public void Dispose()
```

### ToString()

Returns a text representation of the object.

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

Returns

[string ↗](#)

# Class TypeUtilitarian

Namespace: [Cobilas](#)

Assembly: Cobilas.Core.dll

Utility static class to obtain type or assembly.

```
public static class TypeUtilitarian
```

## Inheritance

[object](#) ← TypeUtilitarian

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### GetAssemblies()

Get all assemblies for the current domain.

```
public static Assembly[] GetAssemblies()
```

Returns

[Assembly](#)[]

### GetType(string)

Get a specific type.

```
public static Type GetType(string fullName)
```

Parameters

**fullName** [string](#)

The full name of the type. (example:[System.String](#))

Returns

[Type](#)

## GetTypes()

Get all types of assembly.

```
public static Type[] GetTypes()
```

Returns

[Type](#)[]

## TypeExist(string)

Checks if the type exists.

```
public static bool TypeExist(string fullName)
```

Parameters

**fullName** [string](#)

The full name of the type. (example:[System.String](#))

Returns

[bool](#)

# Namespace Cobilas.Collections

## Classes

### [ArrayManipulation](#)

Array manipulation class.

### [ArrayToIEnumerator<T>](#)

Transforms an array into an enumerator.

### [ICollectionToIEnumerator<T>](#)

Transforms a collection into an enumerator.

## Interfaces

### [ILongCollection](#)

Defines size, enumerators, and synchronization methods for all long non-generic collections.

### [ILongList](#)

Represents a long, non-generic collection of objects that can be accessed individually by index.

# Class ArrayManipulation

Namespace: [Cobilas.Collections](#)

Assembly: Cobilas.Core.dll

Array manipulation class.

```
public static class ArrayManipulation
```

## Inheritance

[object](#) ← ArrayManipulation

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### AddNon\_Existing<T>(T, T[])

Adds a list of items to the target list.

The method does not add items that already exist in the target list.

```
public static T[] AddNon_Existing<T>(T item, T[] list)
```

#### Parameters

**item** T

The item that will be inserted into the list.

**list** T[]

The list that will receive the items.

#### Returns

T[]

This way, when adding the same object, the operation will not be performed and the list will be returned without being modified.

Type Parameters

T

Exceptions

[ArgumentNullException](#)

## AddNon\_Existing<T>(T, ref T[])

Adds a list of items to the target list.

The method does not add items that already exist in the target list.

```
public static void AddNon_Existing<T>(T item, ref T[] list)
```

Parameters

**item** T

The item that will be inserted into the list.

**list** T[]

The list that will receive the items.

Type Parameters

T

Exceptions

[ArgumentNullException](#)

## Add<T>(IEnumerable<T>, T[])

Adds a list of items to the target list.

```
public static T[] Add<T>(IEnumerable<T> collection, T[] list)
```

## Parameters

**collection** [IEnumerable](#)<T>

The items that will be inserted into the list.

**list** T[]

The list that will receive the items.

## Returns

T[]

## Type Parameters

T

## Exceptions

[ArgumentNullException](#)

## Add<T>(IEnumerable<T>, ref T[])

Adds a list of items to the target list.

```
public static void Add<T>(IEnumerable<T> collection, ref T[] list)
```

## Parameters

**collection** [IEnumerable](#)<T>

The items that will be inserted into the list.

**list** T[]

The list that will receive the items.

## Type Parameters

T

## Exceptions

[ArgumentNullException](#)

## Add<T>(IEnumerator<T>, T[])

Adds a list of items to the target list.

```
[Obsolete("Use the T[] Add<T>(IEnumerable<T>, T[]) method.")]
public static T[] Add<T>(IEnumerator<T> itens, T[] list)
```

## Parameters

itens [IEnumerator](#)<T>

The items that will be inserted into the list.

list T[]

The list that will receive the items.

## Returns

T[]

## Type Parameters

T

## Exceptions

[ArgumentNullException](#)

## Add<T>(IEnumerator<T>, ref T[])

Adds a list of items to the target list.

```
[Obsolete("Use the T[] Add<T>(IEnumerable<T>, ref T[]) method.")]
public static void Add<T>(IEnumerator<T> itens, ref T[] list)
```

## Parameters

**itens** [IEnumerator](#)<T>

The items that will be inserted into the list.

**list** T[]

The list that will receive the items.

## Type Parameters

T

## Exceptions

[ArgumentNullException](#)

## Add<T>(T, T[])

Adds a list of items to the target list.

```
public static T[] Add<T>(T item, T[] list)
```

## Parameters

**item** T

The item that will be inserted into the list.

**list** T[]

The list that will receive the items.

## Returns

T[]

## Type Parameters

T

## Exceptions

[ArgumentNullException](#)

## Add<T>(T, ref T[])

Adds a list of items to the target list.

```
public static void Add<T>(T item, ref T[] list)
```

## Parameters

**item** T

The item that will be inserted into the list.

**list** T[]

The list that will receive the items.

## Type Parameters

T

## Exceptions

[ArgumentNullException](#)

## Add<T>(T[], T[])

Adds a list of items to the target list.

```
public static T[] Add<T>(T[] itens, T[] list)
```

## Parameters

**itens** T[]

The items that will be inserted into the list.

**list** T[]

The list that will receive the items.

Returns

T[]

Type Parameters

T

Exceptions

[ArgumentNullException](#)

**Add<T>(T[], ref T[])**

Adds a list of items to the target list.

```
public static void Add<T>(T[] itens, ref T[] list)
```

Parameters

**itens** T[]

The items that will be inserted into the list.

**list** T[]

The list that will receive the items.

Type Parameters

T

Exceptions

## [ArgumentNullException](#)

### ArrayLength(ICollection)

Determines the length of a collection.

```
public static int ArrayLength(ICollection array)
```

Parameters

array [ICollection](#)

Returns

[int](#)

### ArrayLongLength(ILongCollection)

Determines the length of an Array.

```
public static long ArrayLongLength(ILongCollection array)
```

Parameters

array [ILongCollection](#)

Returns

[long](#)

### ArrayLongLength(Array)

Determines the length of an Array.

```
public static long ArrayLongLength(Array array)
```

Parameters

array [Array](#)

Returns

[long](#)

## ClearArray(Array)

Array cleaning.

```
public static void ClearArray(Array array)
```

Parameters

array [Array](#)

Exceptions

[ArgumentNullException](#)

## ClearArray(Array, int, int)

Array cleaning.

```
public static void ClearArray(Array array, int index, int length)
```

Parameters

array [Array](#)

index [int](#)

length [int](#)

Exceptions

[ArgumentNullException](#)

## [IndexOutOfRangeException](#)

### ClearArray(Array, long, long)

Array cleaning.

```
public static void ClearArray(Array array, long index, long length)
```

Parameters

array [Array](#)

index [long](#)

length [long](#)

Exceptions

[ArgumentNullException](#)

[RankException](#)

[ArgumentOutOfRangeException](#)

### ClearArraySafe(Array)

Array cleaning.

It will only perform cleaning if the array is not null.

```
public static void ClearArraySafe(Array array)
```

Parameters

array [Array](#)

### ClearArraySafe(Array, int, int)

Array cleaning.

It will only perform cleaning if the array is not null.

```
public static void ClearArraySafe(Array array, int index, int length)
```

Parameters

array [Array](#)

index [int](#)

length [int](#)

Exceptions

[IndexOutOfRangeException](#)

## ClearArraySafe(Array, long, long)

Array cleaning.

It will only perform cleaning if the array is not null.

```
public static void ClearArraySafe(Array array, long index, long length)
```

Parameters

array [Array](#)

index [long](#)

length [long](#)

Exceptions

[ArgumentNullException](#)

[RankException](#)

[IndexOutOfRangeException](#)

## ClearArraySafe<T>(int, int, ref T[])

Array cleaning.

In addition to clearing the array, it returns an empty array.

It will only perform cleaning if the array is not null.

```
public static void ClearArraySafe<T>(int index, int length, ref T[] array)
```

Parameters

index [int](#)

length [int](#)

array [T\[\]](#)

Type Parameters

T

Exceptions

[IndexOutOfRangeException](#)

## ClearArraySafe<T>(long, long, ref T[])

Array cleaning.

In addition to clearing the array, it returns an empty array.

It will only perform cleaning if the array is not null.

```
public static void ClearArraySafe<T>(long index, long length, ref T[] array)
```

Parameters

index [long](#)

`length long`

`array T[]`

Type Parameters

`T`

Exceptions

[ArgumentNullException](#)

[RankException](#)

[IndexOutOfRangeException](#)

**ClearArraySafe<T>(ref T[])**

Array cleaning.

In addition to clearing the array, it returns an empty array.

It will only perform cleaning if the array is not null.

```
public static void ClearArraySafe<T>(ref T[] array)
```

Parameters

`array T[]`

Type Parameters

`T`

**ClearArray<T>(int, int, ref T[])**

Array cleaning.

In addition to clearing the array, it returns an empty array.

```
public static void ClearArray<T>(int index, int length, ref T[] array)
```

## Parameters

index [int](#)

length [int](#)

array [T\[\]](#)

## Type Parameters

T

## Exceptions

[ArgumentNullException](#)

[IndexOutOfRangeException](#)

## ClearArray<T>(long, long, ref T[])

Array cleaning.

```
public static void ClearArray<T>(long index, long length, ref T[] array)
```

## Parameters

index [long](#)

length [long](#)

array [T\[\]](#)

## Type Parameters

T

## Exceptions

[ArgumentNullException](#)

[RankException](#)

[ArgumentOutOfRangeException](#)

## ClearArray<T>(ref T[])

Array cleaning.

In addition to clearing the array, it returns an empty array.

```
public static void ClearArray<T>(ref T[] array)
```

### Parameters

**array** T[]

### Type Parameters

T

### Exceptions

[ArgumentNullException](#)

## ConvertAll<TInput, TOutput>(TInput[], Converter<TInput, TOutput>)

Converts an array of one type to an array of another type.

```
public static TOutput[] ConvertAll<TInput, TOutput>(TInput[] array, Converter<TInput, TOutput> converter)
```

### Parameters

**array** TInput[]

The one-dimensional, zero-based [Array](#) to convert to a target type.

`converter` [Converter](#)<TInput, TOutput>

A [Converter](#)<TInput, TOutput> that converts each element from one type to another type.

Returns

TOutput[]

An array of the target type containing the converted elements from the source array.

Type Parameters

TInput

The type of the elements of the source array.

TOutput

The type of the elements of the target array.

Exceptions

[ArgumentNullException](#)

## CopyTo(Array, Array)

Copies all the elements of the current one-dimensional array to the specified one-dimensional array.

```
public static void CopyTo(Array sourceArray, Array destinationArray)
```

Parameters

sourceArray [Array](#)

destinationArray [Array](#)

## CopyTo(Array, Array, int)

Copies all the elements of the current one-dimensional array to the specified one-dimensional array.

```
public static void CopyTo(Array sourceArray, Array destinationArray, int length)
```

## Parameters

sourceArray [Array](#)

destinationArray [Array](#)

length [int](#)

## CopyTo(Array, Array, long)

Copies all the elements of the current one-dimensional array to the specified one-dimensional array.

```
public static void CopyTo(Array sourceArray, Array destinationArray, long length)
```

## Parameters

sourceArray [Array](#)

destinationArray [Array](#)

length [long](#)

## CopyTo(Array, int, Array, int, int)

Copies all the elements of the current one-dimensional array to the specified one-dimensional array.

```
public static void CopyTo(Array sourceArray, int sourceIndex, Array destinationArray, int destinationIndex, int length)
```

## Parameters

sourceArray [Array](#)

sourceIndex [int](#)

destinationArray [Array](#)

destinationIndex [int](#)

length [int](#)

## CopyTo(Array, long, Array, long, long)

Copies all the elements of the current one-dimensional array to the specified one-dimensional array.

```
public static void CopyTo(Array sourceArray, long sourceIndex, Array destinationArray, long destinationIndex, long length)
```

### Parameters

sourceArray [Array](#)

sourceIndex [long](#)

destinationArray [Array](#)

destinationIndex [long](#)

length [long](#)

## EmptyArray(ILongCollection)

Determines whether the collection is empty or null.

```
public static bool EmptyArray(ILongCollection array)
```

### Parameters

array [ILongCollection](#)

### Returns

[bool](#)

## EmptyArray(ICollection)

Determines whether the collection is empty or null.

```
public static bool EmptyArray(ICollection array)
```

Parameters

array [ICollection](#)

Returns

[bool](#)

## Exists(object, Array)

Determines whether the specified array contains elements that match the conditions defined by the specified predicate.

```
[Obsolete("Use bool:Exists<T>(T[], Predicate<T>)")]
public static bool Exists(object item, Array array)
```

Parameters

item [object](#)

array [Array](#)

Returns

[bool](#)

## Exists<T>(T, T[])

Determines whether the specified array contains elements that match the conditions defined by the specified predicate.

```
public static bool Exists<T>(T item, T[] array)
```

Parameters

item T

array T[]

Returns

[bool ↗](#)

Type Parameters

T

## Exists<T>(T[], Predicate<T>)

Determines whether the specified array contains elements that match the conditions defined by the specified predicate.

```
public static bool Exists<T>(T[] array, Predicate<T> match)
```

Parameters

array T[]

match [Predicate ↗ <T>](#)

Returns

[bool ↗](#)

Type Parameters

T

## FindAll<T>(T[], Predicate<T>)

Retrieves all elements that match the conditions defined by the specified predicate.

```
public static T[] FindAll<T>(T[] array, Predicate<T> match)
```

Parameters

array `T[]`

match [Predicate](#)<T>

Returns

`T[]`

Type Parameters

`T`

Exceptions

[ArgumentNullException](#)

[RankException](#)

## FindIndex<T>(T[], int, int, Predicate<T>)

Searches for an element that matches the conditions defined by the specified predicate and returns the zero-based index of the first occurrence within the range of elements in the Array that starts at the specified index and contains the specified number of elements.

```
public static int FindIndex<T>(T[] array, int index, int length, Predicate<T> match)
```

Parameters

array `T[]`

index `int`

length `int`

match [Predicate](#)<T>

Returns

[int](#)

Type Parameters

T

## FindIndex<T>(T[], int, Predicate<T>)

Searches for an element that matches the conditions defined by the specified predicate and returns the zero-based index of the first occurrence within the range of elements in the Array that starts at the specified index and contains the specified number of elements.

```
public static int FindIndex<T>(T[] array, int index, Predicate<T> match)
```

Parameters

array T[]

index [int](#)

match [Predicate](#)<T>

Returns

[int](#)

Type Parameters

T

## FindIndex<T>(T[], long, long, Predicate<T>)

Searches for an element that matches the conditions defined by the specified predicate and returns the zero-based index of the first occurrence within the range of elements in the Array that starts at the specified index and contains the specified number of elements.

```
public static long FindIndex<T>(T[] array, long index, long length, Predicate<T> match)
```

Parameters

array T[]

index long ↗

length long ↗

match Predicate<T>

Returns

long ↗

Type Parameters

T

Exceptions

[ArgumentNullException](#) ↗

[RankException](#) ↗

[ArgumentOutOfRangeException](#) ↗

## FindIndex<T>(T[], long, Predicate<T>)

Searches for an element that matches the conditions defined by the specified predicate and returns the zero-based index of the first occurrence within the range of elements in the Array that starts at the specified index and contains the specified number of elements.

```
public static long FindIndex<T>(T[] array, long index, Predicate<T> match)
```

Parameters

array T[]

index long ↗

match Predicate<T>

Returns

[long](#)

Type Parameters

T

Exceptions

[ArgumentNullException](#)

[RankException](#)

[ArgumentOutOfRangeException](#)

## FindIndex<T>(T[], Predicate<T>)

Searches for an element that matches the conditions defined by the specified predicate and returns the zero-based index of the first occurrence within the range of elements in the Array that starts at the specified index and contains the specified number of elements.

```
public static int FindIndex<T>(T[] array, Predicate<T> match)
```

Parameters

array T[]

match [Predicate](#)<T>

Returns

[int](#)

Type Parameters

T

## FindLastIndex<T>(T[], int, int, Predicate<T>)

Searches for an element that matches the conditions defined by a specified predicate and returns the zero-based index of the last occurrence in an Array or part of it.

```
public static int FindLastIndex<T>(T[] array, int index, int length, Predicate<T> match)
```

## Parameters

array T[]

index int ↗

length int ↗

match [Predicate ↗ <T>](#)

## Returns

[int ↗](#)

## Type Parameters

T

## Exceptions

[ArgumentNullException ↗](#)

[RankException ↗](#)

[ArgumentOutOfRangeException ↗](#)

## FindLastIndex<T>(T[], int, Predicate<T>)

Searches for an element that matches the conditions defined by a specified predicate and returns the zero-based index of the last occurrence in an Array or part of it.

```
public static int FindLastIndex<T>(T[] array, int index, Predicate<T> match)
```

## Parameters

array T[]

index [int](#)

match [Predicate](#) <T>

Returns

[int](#)

Type Parameters

T

Exceptions

[ArgumentNullException](#)

[RankException](#)

[ArgumentOutOfRangeException](#)

## FindLastIndex<T>(T[], long, long, Predicate<T>)

Searches for an element that matches the conditions defined by a specified predicate and returns the zero-based index of the last occurrence in an Array or part of it.

```
public static long FindLastIndex<T>(T[] array, long index, long length, Predicate<T> match)
```

Parameters

array T[]

index [long](#)

length [long](#)

match [Predicate](#) <T>

Returns

[long](#)

## Type Parameters

T

## Exceptions

[ArgumentNullException](#)

[RankException](#)

[ArgumentOutOfRangeException](#)

## FindLastIndex<T>(T[], long, Predicate<T>)

Searches for an element that matches the conditions defined by a specified predicate and returns the zero-based index of the last occurrence in an Array or part of it.

```
public static long FindLastIndex<T>(T[] array, long index, Predicate<T> match)
```

## Parameters

array T[]

index long

match [Predicate](#)<T>

## Returns

long

## Type Parameters

T

## Exceptions

[ArgumentNullException](#)

[RankException](#)

## [ArgumentOutOfRangeException](#)

### FindLastIndex<T>(T[], Predicate<T>)

Searches for an element that matches the conditions defined by a specified predicate and returns the zero-based index of the last occurrence in an Array or part of it.

```
public static int FindLastIndex<T>(T[] array, Predicate<T> match)
```

#### Parameters

array T[]

match [Predicate](#)<T>

#### Returns

[int](#)

#### Type Parameters

T

#### Exceptions

[ArgumentNullException](#)

[RankException](#)

[ArgumentOutOfRangeException](#)

### FindLast<T>(T[], Predicate<T>)

Searches for an element that matches the conditions defined by the specified predicate and returns the first occurrence in the entire Array.

```
public static T FindLast<T>(T[] array, Predicate<T> match)
```

#### Parameters

array T[]

match [Predicate](#)<T>

Returns

T

Type Parameters

T

Exceptions

[ArgumentNullException](#)

[RankException](#)

## Find<T>(T[], Predicate<T>)

Searches for an element that matches the conditions defined by the specified predicate, and returns the first occurrence within the entire Array.

```
public static T Find<T>(T[] array, Predicate<T> match)
```

Parameters

array T[]

match [Predicate](#)<T>

Returns

T

Type Parameters

T

Exceptions

[ArgumentNullException](#)

[RankException](#)

## ForSector(Array, in Action<object, long>)

The method traverses several parts of a list simultaneously.

```
public static void ForSector(Array array, in Action<object, long> action)
```

### Parameters

array [Array](#)

The array that will be read.

action [Action](#)<object, long>

Action that receives the object and the index of the list.

### Exceptions

[ArgumentNullException](#)

[ArgumentOutOfRangeException](#)

## ForSector(Array, in Action<object, long>, in long)

The method traverses several parts of a list simultaneously.

```
public static void ForSector(Array array, in Action<object, long> action, in  
long sectorCount)
```

### Parameters

array [Array](#)

The array that will be read.

action [Action](#)<object, long>

Action that receives the object and the index of the list.

#### **sectorCount** [long](#)

The number of sectors to read.

### Exceptions

#### [ArgumentNullException](#)

#### [ArgumentOutOfRangeException](#)

## ForSector(IList, in Action<object, int>)

The method traverses several parts of a list simultaneously.

```
public static void ForSector(IList list, in Action<object, int> action)
```

### Parameters

#### **list** [IList](#)

The array that will be read.

#### **action** [Action](#)<[object](#), [int](#)>

Action that receives the object and the index of the list.

### Exceptions

#### [ArgumentNullException](#)

#### [ArgumentOutOfRangeException](#)

## ForSector(IList, in Action<object, int>, in int)

The method traverses several parts of a list simultaneously.

```
public static void ForSector(IList list, in Action<object, int> action, in int sectorCount)
```

## Parameters

**list** [IList](#)

The array that will be read.

**action** [Action](#)<[Object](#), [int](#)>

Action that receives the object and the index of the list.

**sectorCount** [int](#)

The number of sectors to read.

## Exceptions

[ArgumentNullException](#)

[ArgumentOutOfRangeException](#)

## ForSector<T>(Array, in Action<T, long>)

The method traverses several parts of a list simultaneously.

```
public static void ForSector<T>(Array array, in Action<T, long> action)
```

## Parameters

**array** [Array](#)

The array that will be read.

**action** [Action](#)<T, [long](#)>

Action that receives the object and the index of the list.

## Type Parameters

**T**

## Exceptions

[ArgumentNullException](#)

[ArgumentOutOfRangeException](#)

## ForSector<T>(Array, in Action<T, long>, in long)

The method traverses several parts of a list simultaneously.

```
public static void ForSector<T>(Array array, in Action<T, long> action, in long sectorCount)
```

### Parameters

**array** [Array](#)

The array that will be read.

**action** [Action](#)<T, long>

Action that receives the object and the index of the list.

**sectorCount** [long](#)

The number of sectors to read.

### Type Parameters

T

### Exceptions

[ArgumentNullException](#)

[ArgumentOutOfRangeException](#)

## ForSector<T>(IList<T>, in Action<T, int>)

The method traverses several parts of a list simultaneously.

```
public static void ForSector<T>(IList<T> list, in Action<T, int> action)
```

## Parameters

**list** [IList](#)<T>

The array that will be read.

**action** [Action](#)<T, int>

Action that receives the object and the index of the list.

## Type Parameters

T

## Exceptions

[ArgumentNullException](#)

[ArgumentOutOfRangeException](#)

## ForSector<T>(IList<T>, in Action<T, int>, in int)

The method traverses several parts of a list simultaneously.

```
public static void ForSector<T>(IList<T> list, in Action<T, int> action, in int sectorCount)
```

## Parameters

**list** [IList](#)<T>

The array that will be read.

**action** [Action](#)<T, int>

Action that receives the object and the index of the list.

**sectorCount** [int](#)

The number of sectors to read.

## Type Parameters

T

## Exceptions

[ArgumentNullException](#)

[ArgumentOutOfRangeException](#)

## ForSector<T>(IList, in Action<T, int>)

The method traverses several parts of a list simultaneously.

```
public static void ForSector<T>(IList list, in Action<T, int> action)
```

## Parameters

**list** [IList](#)

The array that will be read.

**action** [Action](#)<T, int>

Action that receives the object and the index of the list.

## Type Parameters

T

## Exceptions

[ArgumentNullException](#)

[ArgumentOutOfRangeException](#)

## ForSector<T>(IList, in Action<T, int>, in int)

The method traverses several parts of a list simultaneously.

```
public static void ForSector<T>(IList list, in Action<T, int> action, in int sectorCount)
```

## Parameters

**list** [IList](#)

The array that will be read.

**action** [Action](#)<T, [int](#)>

Action that receives the object and the index of the list.

**sectorCount** [int](#)

The number of sectors to read.

## Type Parameters

T

## Exceptions

[ArgumentNullException](#)

[ArgumentOutOfRangeException](#)

## ForSector<T>(T[], in Action<T, long>)

The method traverses several parts of a list simultaneously.

```
public static void ForSector<T>(T[] array, in Action<T, long> action)
```

## Parameters

**array** T[]

The array that will be read.

**action** [Action](#)<T, [long](#)>

Action that receives the object and the index of the list.

## Type Parameters

T

## Exceptions

[ArgumentNullException](#)

[ArgumentOutOfRangeException](#)

## ForSector<T>(T[], in Action<T, long>, in long)

The method traverses several parts of a list simultaneously.

```
public static void ForSector<T>(T[] array, in Action<T, long> action, in long sectorCount)
```

## Parameters

**array** T[]

The array that will be read.

**action** [Action](#)<T, long>

Action that receives the object and the index of the list.

**sectorCount** long

The number of sectors to read.

## Type Parameters

T

## Exceptions

[ArgumentNullException](#)

[ArgumentOutOfRangeException](#)

## IndexOf(object, Array)

Searches for the specified object and returns the index of its first occurrence in a one-dimensional array or a range of elements in the array.

```
public static int IndexOf(object item, Array array)
```

Parameters

item [object](#)

array [Array](#)

Returns

[int](#)

## IndexOf(object, Array, int)

Searches for the specified object and returns the index of its first occurrence in a one-dimensional array or a range of elements in the array.

```
public static int IndexOf(object item, Array array, int index)
```

Parameters

item [object](#)

array [Array](#)

index [int](#)

Returns

[int](#)

## IndexOf(object, Array, int, int)

Searches for the specified object and returns the index of its first occurrence in a one-dimensional array or a range of elements in the array.

```
public static int IndexOf(object item, Array array, int index, int length)
```

Parameters

item [object](#)

array [Array](#)

index [int](#)

length [int](#)

Returns

[int](#)

## IndexOf(object, Array, long)

Searches for the specified object and returns the index of its first occurrence in a one-dimensional array or a range of elements in the array.

```
public static long IndexOf(object item, Array array, long index)
```

Parameters

item [object](#)

array [Array](#)

index [long](#)

Returns

[long](#)

## IndexOf(object, Array, long, long)

Searches for the specified object and returns the index of its first occurrence in a one-dimensional array or a range of elements in the array.

```
public static long IndexOf(object item, Array array, long index, long length)
```

Parameters

**item** [object](#)

**array** [Array](#)

**index** [long](#)

**length** [long](#)

Returns

[long](#)

## Insert<T>(IEnumerable<T>, long, T[])

Insert a list of items at a given index into a target array.

```
public static T[] Insert<T>(IEnumerable<T> collection, long index, T[] list)
```

Parameters

**collection** [IEnumerable](#)<T>

The items that will be inserted into the list.

**index** [long](#)

The index of the list where the items will be inserted.

**list** T[]

The list that will receive the items.

Returns

T[]

Type Parameters

T

## Exceptions

[ArgumentNullException](#)

## Insert<T>(IEnumerable<T>, long, ref T[])

Insert a list of items at a given index into a target array.

```
public static void Insert<T>(IEnumerable<T> collection, long index, ref T[] list)
```

## Parameters

**collection** [IEnumerable](#)<T>

The items that will be inserted into the list.

**index** [long](#)

The index of the list where the items will be inserted.

**list** T[]

The list that will receive the items.

## Type Parameters

T

## Exceptions

[ArgumentNullException](#)

## Insert<T>(IEnumerator<T>, long, T[])

Insert a list of items at a given index into a target array.

```
[Obsolete("Use the T[] Insert<T>(IEnumerable<T>, long, T[]) method")]
```

```
public static T[] Insert<T>(IEnumerator<T> itens, long index, T[] list)
```

## Parameters

**itens** [IEnumerator](#)<T>

The items that will be inserted into the list.

**index** [long](#)

The index of the list where the items will be inserted.

**list** T[]

The list that will receive the items.

## Returns

T[]

## Type Parameters

T

## Exceptions

[ArgumentNullException](#)

## Insert<T>(T, long, T[])

Insert a list of items at a given index into a target array.

```
public static T[] Insert<T>(T item, long index, T[] list)
```

## Parameters

**item** T

The item that will be inserted into the list.

**index** [long](#)

The index of the list where the items will be inserted.

**list** T[]

The list that will receive the items.

Returns

T[]

Type Parameters

T

Exceptions

[ArgumentNullException](#)

## Insert<T>(T, long, ref T[])

Insert a list of items at a given index into a target array.

```
public static void Insert<T>(T item, long index, ref T[] list)
```

Parameters

**item** T

The item that will be inserted into the list.

**index** long

The index of the list where the items will be inserted.

**list** T[]

The list that will receive the items.

Type Parameters

T

## Exceptions

[ArgumentNullException](#)

## Insert<T>(T[], long, T[])

Insert a list of items at a given index into a target array.

```
public static T[] Insert<T>(T[] itens, long index, T[] list)
```

### Parameters

**itens** T[]

The items that will be inserted into the list.

**index** long

The index of the list where the items will be inserted.

**list** T[]

The list that will receive the items.

### Returns

T[]

### Type Parameters

T

## Exceptions

[ArgumentNullException](#)

## Insert<T>(T[], long, ref T[])

Insert a list of items at a given index into a target array.

```
public static void Insert<T>(T[] itens, long index, ref T[] list)
```

## Parameters

**itens** `T[]`

The items that will be inserted into the list.

**index** `long` ↗

The index of the list where the items will be inserted.

**list** `T[]`

The list that will receive the items.

## Type Parameters

**T**

## Exceptions

[ArgumentNullException](#) ↗

## IsFixedSizeSafe(IList)

Determines whether the collection has a fixed size.

```
public static bool IsFixedSizeSafe(IList array)
```

## Parameters

**array** [IList](#)

## Returns

`bool` ↗

## IsFixedSizeSafe(IList)

Determines whether the collection has a fixed size.

```
public static bool IsFixedSizeSafe(IList array)
```

Parameters

array [IList](#)

Returns

[bool](#)

## IsReadOnlySafe(ILongList)

Determines whether the collection is read-only.

```
public static bool IsReadOnlySafe(ILongList array)
```

Parameters

array [ILongList](#)

Returns

[bool](#)

## IsReadOnlySafe(IList)

Determines whether the collection is read-only.

```
public static bool IsReadOnlySafe(IList array)
```

Parameters

array [IList](#)

Returns

[bool](#) ↗

## IsSynchronizedSafe(ILongCollection)

Determines whether the collection is synchronized.

```
public static bool IsSynchronizedSafe(ILongCollection collection)
```

Parameters

[collection](#) [ILongCollection](#)

Returns

[bool](#) ↗

## IsSynchronizedSafe(ICollection)

Determines whether the collection is synchronized.

```
public static bool IsSynchronizedSafe(ICollection collection)
```

Parameters

[collection](#) [ICollection](#) ↗

Returns

[bool](#) ↗

## LastIndexOf(object, Array)

Returns the index of the last occurrence of a value in a one-dimensional Array or part of the Array.

```
public static int LastIndexOf(object item, Array array)
```

Parameters

item [object](#)

array [Array](#)

Returns

[int](#)

## LastIndexOf(object, Array, int)

Returns the index of the last occurrence of a value in a one-dimensional Array or part of the Array.

```
public static int LastIndexOf(object item, Array array, int index)
```

Parameters

item [object](#)

array [Array](#)

index [int](#)

Returns

[int](#)

## LastIndexOf(object, Array, int, int)

Returns the index of the last occurrence of a value in a one-dimensional Array or part of the Array.

```
public static int LastIndexOf(object item, Array array, int index, int length)
```

Parameters

`item` [object](#)

`array` [Array](#)

`index` [int](#)

`length` [int](#)

Returns

[int](#)

## LastIndexOf(object, Array, long)

Returns the index of the last occurrence of a value in a one-dimensional Array or part of the Array.

```
public static long LastIndexOf(object item, Array array, long index)
```

Parameters

`item` [object](#)

`array` [Array](#)

`index` [long](#)

Returns

[long](#)

## LastIndexOf(object, Array, long, long)

Returns the index of the last occurrence of a value in a one-dimensional Array or part of the Array.

```
public static long LastIndexOf(object item, Array array, long index, long length)
```

Parameters

`item` [object](#)

array [Array](#)

index [long](#)

length [long](#)

Returns

[long](#)

## LongClearArray(Array)

Array cleaning.

```
public static void LongClearArray(Array array)
```

Parameters

array [Array](#)

Exceptions

[ArgumentNullException](#)

[RankException](#)

[ArgumentOutOfRangeException](#)

## LongClearArraySafe(Array)

Array cleaning.

It will only perform cleaning if the array is not null.

```
public static void LongClearArraySafe(Array array)
```

Parameters

array [Array](#)

## Exceptions

[ArgumentNullException](#)

[RankException](#)

[IndexOutOfRangeException](#)

## LongClearArraySafe<T>(ref T[])

Array cleaning.

In addition to clearing the array, it returns an empty array.

It will only perform cleaning if the array is not null.

```
public static void LongClearArraySafe<T>(ref T[] array)
```

## Parameters

array T[]

## Type Parameters

T

## Exceptions

[ArgumentNullException](#)

[RankException](#)

[IndexOutOfRangeException](#)

## LongClearArray<T>(ref T[])

Array cleaning.

```
public static void LongClearArray<T>(ref T[] array)
```

## Parameters

array T[]

## Type Parameters

T

## Exceptions

[ArgumentNullException](#)

[RankException](#)

[ArgumentOutOfRangeException](#)

# LongConvertAll<TInput, TOutput>(TInput[], Converter<TInput, TOutput>)

Converts an array of one type to an array of another type.

```
public static TOutput[] LongConvertAll<TInput, TOutput>(TInput[] array, Converter<TInput, TOutput> converter)
```

## Parameters

array TInput[]

The one-dimensional, zero-based [Array](#) to convert to a target type.

converter [Converter](#)<TInput, TOutput>

A [Converter<TInput, TOutput>](#) that converts each element from one type to another type.

## Returns

TOutput[]

An array of the target type containing the converted elements from the source array.

## Type Parameters

## TInput

The type of the elements of the source array.

## TOutput

The type of the elements of the target array.

## Exceptions

[ArgumentNullException](#)

[RankException](#)

## LongFindIndex<T>(T[], Predicate<T>)

Searches for an element that matches the conditions defined by the specified predicate and returns the zero-based index of the first occurrence within the range of elements in the Array that starts at the specified index and contains the specified number of elements.

```
public static long LongFindIndex<T>(T[] array, Predicate<T> match)
```

## Parameters

array T[]

match [Predicate](#)<T>

## Returns

[long](#)

## Type Parameters

T

## Exceptions

[ArgumentNullException](#)

[RankException](#)

## [ArgumentOutOfRangeException](#)

### LongFindLastIndex<T>(T[], Predicate<T>)

Searches for an element that matches the conditions defined by a specified predicate and returns the zero-based index of the last occurrence in an Array or part of it.

```
public static long LongFindLastIndex<T>(T[] array, Predicate<T> match)
```

#### Parameters

array T[]

match [Predicate](#)<T>

#### Returns

[long](#)

#### Type Parameters

T

#### Exceptions

[ArgumentNullException](#)

[RankException](#)

[ArgumentOutOfRangeException](#)

### LongIndexOf(object, Array)

Searches for the specified object and returns the index of its first occurrence in a one-dimensional array or a range of elements in the array.

```
public static long LongIndexOf(object item, Array array)
```

#### Parameters

[item](#) [object](#)

[array](#) [Array](#)

Returns

[long](#)

## LongLastIndexOf(object, Array)

Returns the index of the last occurrence of a value in a one-dimensional Array or part of the Array.

```
public static long LongLastIndexOf(object item, Array array)
```

Parameters

[item](#) [object](#)

[array](#) [Array](#)

Returns

[long](#)

## LongReverse(Array)

Reverses the order of the elements in a one-dimensional Array or in a portion of the Array.

```
public static void LongReverse(Array array)
```

Parameters

[array](#) [Array](#)

The one-dimensional Array to reverse.

## ReadOnlySafe<T>(T[])

Turn a list into a read-only list

```
public static ReadOnlyCollection<T> ReadOnlySafe<T>(T[] list)
```

Parameters

list T[]

Returns

[ReadOnlyCollection](#)<T>

If the list is null, it will return an empty read-only list.

Type Parameters

T

## ReadOnly<T>(T[])

Turn a list into a read-only list

```
public static ReadOnlyCollection<T> ReadOnly<T>(T[] list)
```

Parameters

list T[]

Returns

[ReadOnlyCollection](#)<T>

Type Parameters

T

## Remove<T>(long, long, T[])

Remove items from the target list.

```
public static T[] Remove<T>(long index, long length, T[] list)
```

## Parameters

**index** [long](#)

The target index to remove from the target list.

**length** [long](#)

The length and number of items to remove from the list from the index.

**list** [T\[\]](#)

The list from which items will be removed.

## Returns

[T\[\]](#)

## Type Parameters

**T**

## Exceptions

[ArgumentNullException](#)

[IndexOutOfRangeException](#)

## Remove<T>(long, long, ref T[])

Remove items from the target list.

```
public static void Remove<T>(long index, long length, ref T[] list)
```

## Parameters

**index** [long](#)

The target index to remove from the target list.

### **length** [long](#)

The length and number of items to remove from the list from the index.

### **list** [T\[\]](#)

The list from which items will be removed.

## Type Parameters

### **T**

## Exceptions

### [ArgumentNullException](#)

### [IndexOutOfRangeException](#)

## Remove<T>(long, T[])

Remove items from the target list.

```
public static T[] Remove<T>(long index, T[] list)
```

## Parameters

### **index** [long](#)

The target index to remove from the target list.

### **list** [T\[\]](#)

The list from which items will be removed.

## Returns

### [T\[\]](#)

## Type Parameters

T

## Exceptions

[ArgumentNullException](#)

[IndexOutOfRangeException](#)

## Remove<T>(long, ref T[])

Remove items from the target list.

```
public static void Remove<T>(long index, ref T[] list)
```

## Parameters

**index** [long](#)

The target index to remove from the target list.

**list** [T\[\]](#)

The list from which items will be removed.

## Type Parameters

T

## Exceptions

[ArgumentNullException](#)

[IndexOutOfRangeException](#)

## Remove<T>(T, T[])

Remove items from the target list.

```
public static T[] Remove<T>(T item, T[] list)
```

## Parameters

**item** T

The target item to remove from the target list.

**list** T[]

The list from which items will be removed.

## Returns

T[]

## Type Parameters

T

## Exceptions

[ArgumentNullException](#)

[IndexOutOfRangeException](#)

## Remove<T>(T, ref T[])

Remove items from the target list.

```
public static void Remove<T>(T item, ref T[] list)
```

## Parameters

**item** T

The target item to remove from the target list.

**list** T[]

The list from which items will be removed.

## Type Parameters

T

## Exceptions

[ArgumentNullException](#)

[IndexOutOfRangeException](#)

## Resize<T>(ref T[], int)

Changes the number of elements of a one-dimensional array to the specified new size.

```
public static void Resize<T>(ref T[] array, int newSize)
```

### Parameters

array T[]

newSize [int](#)

### Type Parameters

T

## Resize<T>(ref T[], long)

Changes the number of elements of a one-dimensional array to the specified new size.

```
public static void Resize<T>(ref T[] array, long newSize)
```

### Parameters

array T[]

newSize [long](#)

### Type Parameters

T

# Exceptions

## [ArgumentOutOfRangeException](#)

### Reverse(Array)

Reverses the order of the elements in a one-dimensional Array or in a portion of the Array.

```
public static void Reverse(Array array)
```

#### Parameters

##### [array](#) [Array](#)

The one-dimensional Array to reverse.

### Reverse(Array, int, int)

Reverses the order of the elements in a one-dimensional Array or in a portion of the Array.

```
public static void Reverse(Array array, int index, int length)
```

#### Parameters

##### [array](#) [Array](#)

The one-dimensional Array to reverse.

##### [index](#) [int](#)

The starting index of the section to reverse.

##### [length](#) [int](#)

The number of elements in the section to reverse.

### Reverse(Array, long, long)

Reverses the order of the elements in a one-dimensional Array or in a portion of the Array.

```
public static void Reverse(Array array, long index, long length)
```

## Parameters

**array** [Array](#)

The one-dimensional Array to reverse.

**index** [long](#)

The starting index of the section to reverse.

**length** [long](#)

The number of elements in the section to reverse.

## Exceptions

[ArgumentNullException](#)

[RankException](#)

[ArgumentOutOfRangeException](#)

## SeparateList<T>(T[], long, out T[], out T[])

Separate a list into two using an index.

```
public static void SeparateList<T>(T[] array, long separationIndex, out T[] part1, out  
T[] part2)
```

## Parameters

**array** [T\[\]](#)

The list that will be separated.

**separationIndex** [long](#)

The index where the list will be separated.

**part1** T[]

**part2** T[]

Type Parameters

T

Exceptions

[ArgumentNullException](#)

[RankException](#)

[ArgumentOutOfRangeException](#)

**TakeStretch<T>(long, long, T[])**

This function performs a cut in a list.

```
public static T[] TakeStretch<T>(long index, long length, T[] list)
```

Parameters

**index** [long](#)

The index where the clipping will begin.

**length** [long](#)

The index where the clipping will end.

**list** T[]

The list that will be cut.

Returns

T[]

The function will return a list of items that were cut from the original list. The original list will not be modified.

# Type Parameters

T

## Exceptions

[ArgumentNullException](#)

[ArgumentException](#)

[IndexOutOfRangeException](#)

# Class ArrayToIEnumerator<T>

Namespace: [Cobilas.Collections](#)

Assembly: Cobilas.Core.dll

Transforms an array into an enumerator.

```
public class ArrayToIEnumerator<T> : IEnumerator<T>, IDisposable, IEnumerator
```

## Type Parameters

T

### Inheritance

[object](#) ← ArrayToIEnumerator<T>

### Implements

[IEnumerator](#)<T>, [IDisposable](#), [IEnumerator](#)

### Derived

[ICollectionToIEnumerator](#)<T>

### Inherited Members

[object.ToString\(\)](#), [object.Equals\(object\)](#), [object.Equals\(object, object\)](#),  
[object.ReferenceEquals\(object, object\)](#), [object.GetHashCode\(\)](#), [object.GetType\(\)](#),  
[object.MemberwiseClone\(\)](#)

### Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#),  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#),  
[Object CB Extension.CompareType<T>\(object\)](#)

## Constructors

[ArrayToIEnumerator\(\)](#)

Internal constructor.

```
protected ArrayToIEnumerator()
```

## ArrayToIEnumerator(T[])

```
public ArrayToIEnumerator(T[] list)
```

Parameters

`list` T[]

## Fields

`current`

```
protected T current
```

Field Value

T

`index`

```
protected long index
```

Field Value

[long](#)

`list`

```
protected T[] list
```

Field Value

T[]

## Properties

### Current

Gets the element in the collection at the current position of the enumerator.

```
public virtual T Current { get; }
```

Property Value

T

## Methods

### Dispose()

Performs application-defined tasks associated with freeing, releasing, or resetting unmanaged resources.

```
public void Dispose()
```

### Dispose(bool)

Internal disposal of the object.

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

Parameters

disposing [bool](#)

### MoveNext()

Advances the enumerator to the next element of the collection.

```
public virtual bool MoveNext()
```

Returns

[bool](#)

## Reset()

Sets the enumerator to its initial position, which is before the first element in the collection.

```
public virtual void Reset()
```

# Class ICollectionToIEnumerator<T>

Namespace: [Cobilas.Collections](#)

Assembly: Cobilas.Core.dll

Transforms a collection into an enumerator.

```
public class ICollectionToIEnumerator<T> : ArrayToIEnumerator<T>, IEnumerator<T>,  
IDisposable, IEnumerator
```

## Type Parameters

T

## Inheritance

[object](#) ← [ArrayToIEnumerator](#)<T> ← ICollectionToIEnumerator<T>

## Implements

[IEnumerator](#)<T>, [IDisposable](#), [IEnumerator](#)

## Inherited Members

[ArrayToIEnumerator](#)<T>.list , [ArrayToIEnumerator](#)<T>.index , [ArrayToIEnumerator](#)<T>.current ,  
[ArrayToIEnumerator](#)<T>.Dispose() , [object.ToString\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#)

## Constructors

[ICollectionToIEnumerator\(ICollection<T>\)](#)

```
public ICollectionToIEnumerator<T> collection)
```

## Parameters

collection [ICollection](#)<T>

# Properties

## Current

Gets the element in the collection at the current position of the enumerator.

```
public override T Current { get; }
```

## Property Value

T

# Methods

## Dispose(bool)

Performs application-defined tasks associated with freeing, releasing, or resetting unmanaged resources.

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

## Parameters

disposing [bool](#)

## MoveNext()

Advances the enumerator to the next element of the collection.

```
public override bool MoveNext()
```

Returns

bool ↗

## Reset()

Sets the enumerator to its initial position, which is before the first element in the collection.

```
public override void Reset()
```

# Interface ILongCollection

Namespace: [Cobilas.Collections](#)

Assembly: Cobilas.Core.dll

Defines size, enumerators, and synchronization methods for all long non-generic collections.

```
public interface ILongCollection : IEnumerable
```

## Inherited Members

[IEnumerable.GetEnumerator\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#)

# Properties

## Count

Gets the number of elements contained in the [ILongCollection](#).

```
long Count { get; }
```

## Property Value

[long](#)

## IsSynchronized

Gets a value indicating whether access to the [ILongCollection](#) is synchronized (thread safe).

```
bool IsSynchronized { get; }
```

Property Value

[bool](#)

## SyncRoot

Gets an object that can be used to synchronize access to the [IICollection](#).

```
object SyncRoot { get; }
```

Property Value

[object](#)

## Methods

### CopyTo(Array, long)

Copies the elements of the [IICollection](#) to an Array, starting at a particular Array index.

```
void CopyTo(Array array, long index)
```

Parameters

**array** [Array](#)

The one-dimensional Array that is the destination of the elements copied from **ICollection**. The Array must have zero-based indexing.

**index** [long](#)

The zero-based **index** in **array** at which copying begins.

# Interface ILongList

Namespace: [Cobilas.Collections](#)

Assembly: Cobilas.Core.dll

Represents a long, non-generic collection of objects that can be accessed individually by index.

```
public interface ILongList : ILongCollection, IEnumerable
```

## Inherited Members

[ILongCollection.Count](#) , [ILongCollection.IsSynchronized](#) , [ILongCollection.SyncRoot](#) ,  
[ILongCollection.CopyTo\(Array, long\)](#) , [IEnumerable.GetEnumerator\(\)](#) ↗

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#)

## Properties

### IsFixedSize

Gets a value indicating whether the [ILongList](#) has a fixed size.

```
bool IsFixedSize { get; }
```

### Property Value

[bool](#) ↗

### IsReadOnly

Gets a value indicating whether the [ILongList](#) is read-only.

```
bool IsReadOnly { get; }
```

Property Value

[bool](#)

this[long]

Gets or sets the element at the specified index.

```
object this[long index] { get; set; }
```

Parameters

index [long](#)

The zero-based index of the element to get or set.

Property Value

[object](#)

## Methods

Add(object)

Adds an item to the [ILongList](#).

```
long Add(object value)
```

Parameters

value [object](#)

The object to add to the [ILongList](#).

Returns

## [long](#)

The position into which the new element was inserted, or -1 to indicate that the item was not inserted into the collection.

## Clear()

Removes all items from the [ILongList](#).

```
void Clear()
```

## Contains(object)

Determines whether the [ILongList](#) contains a specific value.

```
bool Contains(object value)
```

Parameters

**value** [object](#)

The object to locate in the [ILongList](#).

Returns

[bool](#)

**true** if the Object is found in the [ILongList](#); otherwise, **false**.

## IndexOf(object)

Determines the index of a specific item in the [ILongList](#).

```
long IndexOf(object value)
```

Parameters

**value** [object](#)

The object to locate in the [ILongList](#).

Returns

[long](#)

The index of **value** if found in the list; otherwise, -1.

## Insert(long, object)

Inserts an item to the [ILongList](#) at the specified index.

**void** [Insert](#)(**long** index, [object](#) value)

Parameters

**index** [long](#)

The zero-based index at which **value** should be inserted.

**value** [object](#)

The object to insert into the [ILongList](#).

## Remove(object)

Removes the first occurrence of a specific object from the [ILongList](#).

**void** [Remove](#)([object](#) value)

Parameters

**value** [object](#)

The object to remove from the [ILongList](#).

## RemoveAt(long)

Removes the [ILongList](#) item at the specified index.

```
void RemoveAt(long index)
```

### Parameters

**index** [long](#) ↗

The zero-based index of the item to remove.

# Namespace Cobilas.Collections.Generic

## Classes

### [LongList<T>](#)

Represents a long, strongly typed list of objects that can be accessed by index. Provides methods for searching, sorting, and manipulating lists.

### [ReadOnlyLongCollection<T>](#)

Provides the base class for a read-only generic Long collection.

## Interfaces

### [ILongCollection<T>](#)

Defines methods for manipulating Long generic collections.

### [ILongList<T>](#)

Represents a long collection of objects that can be accessed individually by index.

### [IReadOnlyLongCollection<T>](#)

Represents a long, read-only, strongly typed collection of elements.

### [IReadOnlyLongList<T>](#)

Represents a long read-only collection of elements that can be accessed by index.

# Interface ILongCollection<T>

Namespace: [Cobilas.Collections.Generic](#)

Assembly: Cobilas.Core.dll

Defines methods for manipulating Long generic collections.

```
public interface ILongCollection<T> : IEnumerable<T>, IEnumerable
```

## Type Parameters

T

The type of the elements in the collection.

## Inherited Members

[IEnumerable<T>.GetEnumerator\(\)](#)

## Extension Methods

[Object\\_CB\\_Extension.CompareType\(object, Type\)](#) ,  
[Object\\_CB\\_Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object\\_CB\\_Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object\\_CB\\_Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object\\_CB\\_Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object\\_CB\\_Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object\\_CB\\_Extension.CompareType<T>\(object\)](#)

# Properties

## Count

Gets the number of elements contained in the [ILongCollection<T>](#).

```
long Count { get; }
```

## Property Value

[long](#)

## IsReadOnly

Gets a value indicating whether the [ILongCollection<T>](#) is read-only.

```
bool IsReadOnly { get; }
```

## Property Value

[bool](#)

## Methods

### Add(T)

Adds an item to the [ILongCollection<T>](#).

```
void Add(T item)
```

#### Parameters

**item** T

The object to add to the [ILongCollection<T>](#).

### Clear()

Removes all items from the [ILongCollection<T>](#).

```
void Clear()
```

### Contains(T)

Determines whether the [ILongCollection<T>](#) contains a specific value.

```
bool Contains(T item)
```

## Parameters

**item** T

The object to locate in the [ILongCollection<T>](#).

## Returns

**bool** ↗

**true** if **item** is found in the [ILongCollection<T>](#); otherwise, **false**.

## CopyTo(T[], long)

Copies the elements of the [ILongCollection<T>](#) to an Array, starting at a particular Array index.

**void CopyTo(T[] array, long arrayIndex)**

## Parameters

**array** T[]

The one-dimensional Array that is the destination of the elements copied from [ILongCollection<T>](#).

The Array must have zero-based indexing.

**arrayIndex** long ↗

The zero-based index in **array** at which copying begins.

## Remove(T)

Removes the first occurrence of a specific object from the [ILongCollection<T>](#).

**bool Remove(T item)**

## Parameters

**item** T

The object to remove from the [ILongCollection<T>](#).

Returns

bool ↗

# Interface ILongList<T>

Namespace: [Cobilas.Collections.Generic](#)

Assembly: Cobilas.Core.dll

Represents a long collection of objects that can be accessed individually by index.

```
public interface ILongList<T> : ILongCollection<T>, IEnumerable<T>, IEnumerable
```

## Type Parameters

T

The type of elements in the list.

## Inherited Members

[ILongCollection<T>.Count](#) , [ILongCollection<T>.IsReadOnly](#) , [ILongCollection<T>.Add\(T\)](#) ,  
[ILongCollection<T>.Clear\(\)](#) , [ILongCollection<T>.Contains\(T\)](#) , [ILongCollection<T>.CopyTo\(T\[\], long\)](#) ,  
[ILongCollection<T>.Remove\(T\)](#) , [IEnumerable<T>.GetEnumerator\(\)](#) ↗

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#)

## Properties

### this[long]

Gets or sets the element at the specified index.

```
T this[long index] { get; set; }
```

## Parameters

`index` [long](#)

Property Value

T

## Methods

### IndexOf(T)

Determines the index of a specific item in the [ILongList<T>](#).

`long` `IndexOf(T item)`

Parameters

`item` T

The object to locate in the [ILongList<T>](#).

Returns

[long](#)

The index of `item` if found in the list; otherwise, -1.

### Insert(long, T)

Inserts an item to the [ILongList<T>](#) at the specified index.

`void` `Insert(long index, T item)`

Parameters

`index` [long](#)

The zero-based index at which `item` should be inserted.

`item` T

The object to insert into the [ILongList<T>](#).

## RemoveAt(long)

Removes the [ILongList<T>](#) item at the specified index.

```
void RemoveAt(long index)
```

### Parameters

**index** [long](#) ↗

The zero-based index of the item to remove.

# Interface IReadOnlyLongCollection<T>

Namespace: [Cobilas.Collections.Generic](#)

Assembly: Cobilas.Core.dll

Represents a long, read-only, strongly typed collection of elements.

```
public interface IReadOnlyLongCollection<out T> : IEnumerable<T>, IEnumerable
```

## Type Parameters

T

The type of the elements.

This type parameter is covariant. That is, you can use either the type you specified or any type that is more derived. For more information about covariance and contravariance, see Covariance and Contravariance in Generics.

## Inherited Members

[IEnumerable<T>.GetEnumerator\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#)

## Properties

### Count

Gets the number of elements in the collection.

```
long Count { get; }
```

## Property Value

[long↗](#)

# Interface IReadOnlyLongList<T>

Namespace: [Cobilas.Collections.Generic](#)

Assembly: Cobilas.Core.dll

Represents a long read-only collection of elements that can be accessed by index.

```
public interface IReadOnlyLongList<out T> : IReadOnlyLongCollection<T>,
IEnumerable<T>, IEnumerable
```

## Type Parameters

T

The type of elements in the read-only list.

This type parameter is covariant. That is, you can use either the type you specified or any type that is more derived. For more information about covariance and contravariance, see Covariance and Contravariance in Generics.

## Inherited Members

[IReadOnlyLongCollection<T>.Count](#) , [IEnumerable<T>.GetEnumerator\(\)](#)

## Extension Methods

```
Object CB Extension.CompareType(object, Type),
Object CB Extension.CompareType(object, params Type[]),
Object CB Extension.CompareTypeAndSubType(object, Type),
Object CB Extension.CompareTypeAndSubType(object, Type, bool),
Object CB Extension.CompareTypeAndSubType<T>(object),
Object CB Extension.CompareTypeAndSubType<T>(object, bool),
Object CB Extension.CompareType<T>(object)
```

## Properties

### this[long]

Gets the element at the specified index in the read-only list.

```
T this[long index] { get; }
```

## Parameters

index [long ↗](#)

Property Value

T

# Class LongList<T>

Namespace: [Cobilas.Collections.Generic](#)

Assembly: Cobilas.Core.dll

Represents a long, strongly typed list of objects that can be accessed by index. Provides methods for searching, sorting, and manipulating lists.

```
[Serializable]
public class LongList<T> : ILongList<T>, ILongCollection<T>, IReadOnlyLongList<T>,
IReadOnlyLongCollection<T>, IEnumerable<T>, ILongList, ILongCollection,
IEnumerable, ICloneable
```

## Type Parameters

T

The type of elements in the list.

## Inheritance

[object](#) ↗ ← LongList<T>

## Implements

[ILongList](#)<T>, [ILongCollection](#)<T>, [IReadOnlyLongList](#)<T>, [IReadOnlyLongCollection](#)<T>,  
[IEnumerable](#)<T>, [ILongList](#), [ILongCollection](#), [IEnumerable](#) ↗, [ICloneable](#) ↗

## Inherited Members

[object.ToString\(\)](#) ↗, [object.Equals\(object\)](#) ↗, [object.Equals\(object, object\)](#) ↗ ,  
[object.ReferenceEquals\(object, object\)](#) ↗, [object.GetHashCode\(\)](#) ↗, [object.GetType\(\)](#) ↗ ,  
[object.MemberwiseClone\(\)](#) ↗

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#)

# Constructors

## LongList()

Creates a new instance of the object.

```
public LongList()
```

## LongList(IEnumerable<T>)

Creates a new instance of the object.

```
public LongList(IEnumerable<T> collection)
```

Parameters

`collection` [IEnumerable](#)<T>

## LongList(long)

Creates a new instance of the object.

```
public LongList(long capacity)
```

Parameters

`capacity` [long](#)

## LongList(params T[])

Creates a new instance of the object.

```
public LongList(params T[] collection)
```

Parameters

`collection T[]`

## Properties

### Capacity

Gets or sets the total number of elements the internal data structure can hold without resizing.

```
public long Capacity { get; set; }
```

Property Value

[long](#) ↗

### Count

Gets the number of elements contained in the [ILongCollection<T>](#).

```
public long Count { get; }
```

Property Value

[long](#) ↗

### IsReadOnly

Gets a value indicating whether the [ILongCollection<T>](#) is read-only.

```
public bool IsReadOnly { get; }
```

Property Value

[bool](#) ↗

`this[long]`

Gets or sets the element at the specified index.

```
public T this[long index] { get; set; }
```

## Parameters

**index** [long](#)

## Property Value

T

# Methods

## Add(T)

Adds an item to the [ILongCollection<T>](#).

```
public void Add(T item)
```

## Parameters

**item** T

The object to add to the [ILongCollection<T>](#).

## AddRange(IEnumerable<T>)

Adds the elements of the specified collection to the end of the [LongList<T>](#).

```
public void AddRange(IEnumerable<T> collection)
```

## Parameters

**collection** [IEnumerable](#)<T>

The collection whose elements should be added to the end of the [LongList<T>](#). The collection itself cannot be [null](#), but it can contain elements that are [null](#), if type T is a reference type.

## AsReadOnly()

Returns a read-only ReadOnlyLongCollection< T > wrapper for the current collection.

```
public ReadOnlyLongCollection<T> AsReadOnly()
```

Returns

[ReadOnlyLongCollection<T>](#)

## Clear()

Removes all items from the [ILongCollection<T>](#).

```
public void Clear()
```

## Clone()

Creates a new object that is a copy of the current instance.

```
public object Clone()
```

Returns

[object](#)

A new object that is a copy of this instance.

## Contains(Predicate<T>)

Determines whether the [ILongCollection<T>](#) contains a specific value.

```
public bool Contains(Predicate<T> match)
```

Parameters

`match` [Predicate](#) <T>

The match parameter allows you to create custom comparison logic.

Returns

[bool](#)

`true` if `item` is found in the [ILongCollection](#)<T>; otherwise, `false`.

## Contains(T)

Determines whether the [ILongCollection](#)<T> contains a specific value.

`public bool Contains(T item)`

Parameters

`item` T

The object to locate in the [ILongCollection](#)<T>.

Returns

[bool](#)

`true` if `item` is found in the [ILongCollection](#)<T>; otherwise, `false`.

## ConvertAll<TOoutput>(Converter<T, TOoutput>)

Converts an array of one type to an array of another type.

`public TOoutput[] ConvertAll<TOoutput>(Converter<T, TOoutput> converter)`

Parameters

`converter` [Converter](#)<T, TOoutput>

A [Converter](#)<TInput, TOoutput> that converts each element from one type to another type.

Returns

TOutput[]

An array of the target type containing the converted elements from the source array.

Type Parameters

TOutput

The type of the elements of the target array.

Exceptions

[ArgumentNullException](#)

[RankException](#)

## CopyTo(T[], long)

Copies the elements of the [ILongCollection<T>](#) to an Array, starting at a particular Array index.

```
public void CopyTo(T[] array, long arrayIndex)
```

Parameters

array T[]

The one-dimensional Array that is the destination of the elements copied from [ILongCollection<T>](#).

The Array must have zero-based indexing.

arrayIndex long

The zero-based index in **array** at which copying begins.

## Find(Predicate<T>)

Searches for an element that matches the conditions defined by the specified predicate, and returns the first occurrence within the entire Array.

```
public T Find(Predicate<T> match)
```

Parameters

match [Predicate](#) <T>

Returns

T

Exceptions

[ArgumentNullException](#)

[RankException](#)

## FindAll(Predicate<T>)

Retrieves all elements that match the conditions defined by the specified predicate.

```
public T[] FindAll(Predicate<T> match)
```

Parameters

match [Predicate](#) <T>

Returns

T[]

Exceptions

[ArgumentNullException](#)

[RankException](#)

## FindIndex(long, long, Predicate<T>)

Searches for an element that matches the conditions defined by the specified predicate and returns the zero-based index of the first occurrence within the range of elements in the Array that starts at the specified index and contains the specified number of elements.

```
public long FindIndex(long startIndex, long count, Predicate<T> match)
```

Parameters

**startIndex** [long](#)

**count** [long](#)

**match** [Predicate](#) <T>

Returns

[long](#)

Exceptions

[ArgumentNullException](#)

[RankException](#)

[ArgumentOutOfRangeException](#)

## FindIndex(long, Predicate<T>)

Searches for an element that matches the conditions defined by the specified predicate and returns the zero-based index of the first occurrence within the range of elements in the Array that starts at the specified index and contains the specified number of elements.

```
public long FindIndex(long startIndex, Predicate<T> match)
```

Parameters

**startIndex** [long](#)

**match** [Predicate](#) <T>

Returns

[long](#)

Exceptions

[ArgumentNullException](#)

[RankException](#)

[ArgumentOutOfRangeException](#)

## FindIndex(Predicate<T>)

Searches for an element that matches the conditions defined by the specified predicate and returns the zero-based index of the first occurrence within the range of elements in the Array that starts at the specified index and contains the specified number of elements.

```
public long FindIndex(Predicate<T> match)
```

Parameters

match [Predicate](#)<T>

Returns

[long](#)

## FindLast(Predicate<T>)

Searches for an element that matches the conditions defined by the specified predicate and returns the first occurrence in the entire Array.

```
public T FindLast(Predicate<T> match)
```

Parameters

match [Predicate](#)<T>

Returns

T

Exceptions

[ArgumentNullException](#)

[RankException](#)

## FindLastIndex(long, long, Predicate<T>)

Searches for an element that matches the conditions defined by a specified predicate and returns the zero-based index of the last occurrence in an Array or part of it.

```
public long FindLastIndex(long startIndex, long count, Predicate<T> match)
```

Parameters

startIndex [long](#)

count [long](#)

match [Predicate](#)<T>

Returns

[long](#)

Exceptions

[ArgumentNullException](#)

[RankException](#)

[ArgumentOutOfRangeException](#)

## FindLastIndex(long, Predicate<T>)

Searches for an element that matches the conditions defined by a specified predicate and returns the zero-based index of the last occurrence in an Array or part of it.

```
public long FindLastIndex(long startIndex, Predicate<T> match)
```

Parameters

startIndex [long](#)

match [Predicate](#)<T>

Returns

[long](#)

Exceptions

[ArgumentNullException](#)

[RankException](#)

[ArgumentOutOfRangeException](#)

## FindLastIndex(Predicate<T>)

Searches for an element that matches the conditions defined by a specified predicate and returns the zero-based index of the last occurrence in an Array or part of it.

```
public long FindLastIndex(Predicate<T> match)
```

Parameters

match [Predicate](#)<T>

Returns

[long](#)

Exceptions

[ArgumentNullException](#)

[RankException](#)

[ArgumentOutOfRangeException](#)

## ForEach(Action<T>)

Performs the specified action on each element of the [LongList<T>](#).

```
public void ForEach(Action<T> action)
```

### Parameters

**action** [Action](#)<T>

The [Action<T>](#) delegate to perform on each element of the [LongList<T>](#).

### Exceptions

[ArgumentNullException](#)

[InvalidOperationException](#)

## GetEnumerator()

Returns an enumerator that iterates through the collection.

```
public IEnumrator<T> GetEnumerator()
```

### Returns

[IEnumerator](#)<T>

An enumerator that can be used to iterate through the collection.

## GetRange(long, long)

Creates a shallow copy of a range of elements in the source [LongList<T>](#).

```
public LongList<T> GetRange(long index, long count)
```

## Parameters

**index** [long](#)

The zero-based [LongList<T>](#) index at which the range starts.

**count** [long](#)

The number of elements in the range.

## Returns

[LongList](#)<T>

A shallow copy of a range of elements in the source [LongList<T>](#).

## Exceptions

[ArgumentOutOfRangeException](#)

## IndexOf(T)

Determines the index of a specific item in the [ILongList<T>](#).

```
public long IndexOf(T item)
```

## Parameters

**item** T

The object to locate in the [ILongList<T>](#).

## Returns

[long](#)

The index of **item** if found in the list; otherwise, -1.

## Insert(long, T)

Inserts an item to the [ILongList<T>](#) at the specified index.

```
public void Insert(long index, T item)
```

### Parameters

**index** [long](#)

The zero-based index at which **item** should be inserted.

**item** [T](#)

The object to insert into the [ILongList<T>](#).

## InsertRange(long, IEnumerable<T>)

Inserts the elements of a collection into the [LongList<T>](#) at the specified index.

```
public void InsertRange(long index, IEnumerable<T> collection)
```

### Parameters

**index** [long](#)

The zero-based index at which the new elements should be inserted.

**collection** [IEnumerable](#)<[T](#)>

The collection whose elements should be inserted into the [LongList<T>](#). The collection itself cannot be [null](#), but it can contain elements that are [null](#), if type [T](#) is a reference type.

## Remove(T)

Removes the first occurrence of a specific object from the [ILongCollection<T>](#).

```
public bool Remove(T item)
```

## Parameters

**item** `T`

The object to remove from the [ILongCollection<T>](#).

## Returns

[bool](#)

## RemoveAll(Predicate<T>)

Removes all the elements that match the conditions defined by the specified predicate.

```
public long RemoveAll(Predicate<T> match)
```

## Parameters

**match** [Predicate](#) <T>

The [Predicate<T>](#) delegate that defines the conditions of the elements to remove.

## Returns

[long](#)

The number of elements removed from the [LongList<T>](#).

## Exceptions

[ArgumentNullException](#)

## RemoveAt(long)

Removes the [ILongList<T>](#) item at the specified index.

```
public void RemoveAt(long index)
```

## Parameters

**index** [long](#)

The zero-based index of the item to remove.

## RemoveRange(long, long)

Removes a range of elements from the [LongList<T>](#)

```
public void RemoveRange(long index, long count)
```

### Parameters

**index** [long](#)

The zero-based starting index of the range of elements to remove.

**count** [long](#)

The number of elements to remove.

## Reverse()

Reverses the order of the elements in a one-dimensional Array or in a portion of the Array.

```
public void Reverse()
```

## Reverse(long, long)

Reverses the order of the elements in a one-dimensional Array or in a portion of the Array.

```
public void Reverse(long index, long length)
```

### Parameters

**index** [long](#)

The starting index of the section to reverse.

`length` `long`

The number of elements in the section to reverse.

## Exceptions

[ArgumentNullException](#)

[RankException](#)

[ArgumentOutOfRangeException](#)

## ToArrayList()

Copies the elements of the [LongList<T>](#) to a new array.

```
public T[] ToArrayList()
```

## Returns

`T[]`

# Class ReadOnlyLongCollection<T>

Namespace: [Cobilas.Collections.Generic](#)

Assembly: Cobilas.Core.dll

Provides the base class for a read-only generic Long collection.

```
[Serializable]
public class ReadOnlyLongCollection<T> : ILongList, ILongCollection, ILongList<T>,
ILongCollection<T>, IReadOnlyLongList<T>, IReadOnlyLongCollection<T>, IEnumerable<T>,
IEnumerable, ICloneable
```

## Type Parameters

T

The type of elements in the collection.

## Inheritance

[object](#) ← `ReadOnlyLongCollection<T>`

## Implements

[ILongList](#), [ILongCollection](#), [ILongList<T>](#), [ILongCollection<T>](#), [IReadOnlyLongList](#)<T>, [IReadOnlyLongCollection](#)<T>, [IEnumerable](#)<T>, [IEnumerable](#), [ICloneable](#)

## Inherited Members

[object.ToString\(\)](#), [object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.ReferenceEquals\(object, object\)](#), [object.GetHashCode\(\)](#), [object.GetType\(\)](#), [object.MemberwiseClone\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#),  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#),  
[Object CB Extension.CompareType<T>\(object\)](#)

## Constructors

### ReadOnlyLongCollection(IEnumerable<T>)

Creates a new instance of the object.

```
public ReadOnlyLongCollection(IEnumerable<T> enumerable)
```

Parameters

enumerable [IEnumerable](#)<T>

## Properties

### Count

Gets the number of elements contained in the [ILongCollection](#).

```
public long Count { get; }
```

Property Value

[long](#)

### this[long]

Gets the element at the specified index in the read-only list.

```
public T this[long index] { get; }
```

Parameters

index [long](#)

Property Value

T

# Methods

## Clone()

Creates a new object that is a copy of the current instance.

```
public object Clone()
```

Returns

[object](#)

A new object that is a copy of this instance.

## GetEnumerator()

Returns an enumerator that iterates through the collection.

```
public IEnumerator<T> GetEnumerator()
```

Returns

[IEnumerator](#) <T>

An enumerator that can be used to iterate through the collection.

# Namespace Coblas.IO.ATLF

## Classes

### [ATLFBase](#)

Base class for all ATLF classes.

### [ATLFEException](#)

### [ATLFReader](#)

Base class for ATLF read classes.

### [ATLFSBReader](#)

### [ATLFSBWriter](#)

### [ATLFStreamReader](#)

### [ATLFStreamWriter](#)

### [ATLFTBReader](#)

### [ATLFTBWriter](#)

### [ATLFTextReader](#)

### [ATLFTextWriter](#)

### [ATLFWriter](#)

Base class for ATLF writing classes.

## Structs

### [ATLFNode](#)

Represents an ATLF node.

## Enums

### [ATLFNodeType](#)

Represents the ATLF node type.

# Class ATLFBase

Namespace: [Cobilas.IO.Atlf](#)

Assembly: Cobilas.Core.dll

Base class for all ATLF classes.

```
public abstract class ATLFBase : IDisposable
```

Inheritance

[object](#) ← ATLFBase

Implements

[IDisposable](#)

Derived

[ATLFReader](#), [ATLFWriter](#)

Inherited Members

[object.ToString\(\)](#), [object.Equals\(object\)](#), [object.Equals\(object, object\)](#),  
[object.ReferenceEquals\(object, object\)](#), [object.GetHashCode\(\)](#), [object.GetType\(\)](#),  
[object.MemberwiseClone\(\)](#)

Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#),  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#),  
[Object CB Extension.CompareType<T>\(object\)](#).

## Properties

### CloseFlow

This property is used to close the workflow automatically.

This property should be used in cases where you directly access a stream. Example: when a flow is called using the [File](#).Open(string) method.

```
protected abstract bool CloseFlow { get; set; }
```

Property Value

[bool](#)

## Closed

Indicate whether the flow has been closed.

```
public abstract bool Closed { get; protected set; }
```

Property Value

[bool](#)

## Encoding

Sets or returns the encoding used.

```
public abstract Encoding Encoding { get; set; }
```

Property Value

[Encoding](#)

## Indent

Determines whether text should be indented.

```
public abstract bool Indent { get; set; }
```

Property Value

[bool](#)

## NodeCount

The number of ATLF nodes stored.

```
public abstract long NodeCount { get; }
```

Property Value

[long](#) ↗

## Nodes

Where ATLF nodes are stored.

```
protected abstract ATLFNode[] Nodes { get; set; }
```

Property Value

[ATLFNode\[\]](#)

## RefObject

Sets or returns the current stream.

```
protected abstract MarshalByRefObject RefObject { get; set; }
```

Property Value

[MarshalByRefObject](#) ↗

## TargetVersion

Represents the version that the ATLF object is using.

```
public abstract string TargetVersion { get; set; }
```

## Property Value

[string ↗](#)

## Methods

### Close()

Allows you to close the current flow.

```
public abstract void Close()
```

### Dispose()

Allows you to discard the current stream.

```
public abstract void Dispose()
```

# Class ATLException

Namespace: [Cobilas.IO.Atlf](#)

Assembly: Cobilas.Core.dll

```
[Serializable]
public class ATLException : Exception, ISerializable, _Exception
```

## Inheritance

[object](#) ← [Exception](#) ← ATLException

## Implements

[ISerializable](#), [Exception](#)

## Inherited Members

[Exception.GetBaseException\(\)](#), [Exception.ToString\(\)](#),  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#), [Exception.GetType\(\)](#),  
[Exception.Message](#), [Exception.Data](#), [Exception.InnerException](#), [Exception.TargetSite](#),  
[Exception.StackTrace](#), [Exception.HelpLink](#), [Exception.Source](#), [Exception.HResult](#),  
[Exception.SerializeObjectState](#), [object.Equals\(object\)](#), [object.Equals\(object, object\)](#),  
[object.ReferenceEquals\(object, object\)](#), [object.GetHashCode\(\)](#), [object.MemberwiseClone\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#),  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#),  
[Object CB Extension.CompareType<T>\(object\)](#)

# Constructors

## ATLException()

```
public ATLException()
```

## ATLException(SerializationInfo, StreamingContext)

```
protected ATLException(SerializationInfo info, StreamingContext context)
```

### Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## ATLException(string)

```
public ATLException(string message)
```

### Parameters

message [string](#)

## ATLException(string, Exception)

```
public ATLException(string message, Exception inner)
```

### Parameters

message [string](#)

inner [Exception](#)

## Methods

### ATLFClosed()

msg:The ATLFWriter object has already been closed.

```
public static InvalidOperationException ATLFClosed()
```

Returns

[InvalidOperationException](#)

## ATLFFlowAfterClosing()

msg:It is not possible to release resources to the flow after closing the ATL object.

```
public static InvalidOperationException ATLFFlowAfterClosing()
```

Returns

[InvalidOperationException](#)

## ATLFReaderAfterClosing()

```
public static InvalidOperationException ATLFReaderAfterClosing()
```

Returns

[InvalidOperationException](#)

## ATLFReaderTagAfterClosing()

```
public static InvalidOperationException ATLFReaderTagAfterClosing()
```

Returns

[InvalidOperationException](#)

## ATLFTagsAfterClosing()

```
public static InvalidOperationException ATLFTagsAfterClosing()
```

Returns

[InvalidOperationException](#)

## GetATLException(string, params object[])

```
public static ATLException GetATLException(string format, params object[] args)
```

Parameters

format [string](#)

args [object](#)[]

Returns

[ATLException](#)

# Struct ATLFNode

Namespace: [Cobilas.IO.Atlf](#)

Assembly: Cobilas.Core.dll

Represents an ATLF node.

```
public struct ATLFNode : IDisposable
```

Implements

[IDisposable](#)

Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetType\(\)](#)

Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#)

## Properties

Name

```
public string Name { get; }
```

Property Value

[string](#)

NodeType

```
public ATLNodeType NodeType { get; }
```

Property Value

[ATLNodeType](#)

Value

```
public string Value { get; }
```

Property Value

[string](#)

## Methods

**Dispose()**

Performs application-defined tasks associated with freeing, releasing, or resetting unmanaged resources.

```
public void Dispose()
```

**ToString()**

Returns the fully qualified type name of this instance.

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

Returns

[string](#)

The fully qualified type name.

# Enum ATLFNodeType

Namespace: [Cobilas.IO.Atlf](#)

Assembly: Cobilas.Core.dll

Represents the ATLF node type.

```
public enum ATLFNodeType : byte
```

## Extension Methods

[Enum CB Extension.Format\(Enum, object, string\)](#) , [Enum CB Extension.GetEnumPair\(Enum\)](#) ,  
[Enum CB Extension.GetEnumPairs\(Enum\)](#) , [Enum CB Extension.GetName\(Enum\)](#) ,  
[Enum CB Extension.GetName\(Enum, object\)](#) , [Enum CB Extension.GetNames\(Enum\)](#) ,  
[Enum CB Extension.HasFlag\(Enum, params Enum\[\]\)](#) , [Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#).

## Fields

**Comment** = 0

ATLF Comment.

**Spacing** = 2

The spacing used in the ATLF file.

**Tag** = 1

ATLF element.

# Class ATLFReader

Namespace: [Cobilas.IO.Atlf](#)

Assembly: Cobilas.Core.dll

Base class for ATLF read classes.

```
public abstract class ATLFReader : ATLFBase, IDisposable, IEnumerable<ATLFNode>, IEnumerable
```

Inheritance

[object](#) ← [ATLFBase](#) ← ATLFReader

Implements

[IDisposable](#), [IEnumerable](#)<[ATLFNode](#)>, [IEnumerable](#)

Derived

[ATLFSBReader](#), [ATLFTBReader](#)

Inherited Members

[ATLFBase.NodeCount](#), [ATLFBase.Indent](#), [ATLFBase.Encoding](#), [ATLFBase.TargetVersion](#), [ATLFBase.Closed](#),  
[ATLFBase.CloseFlow](#), [ATLFBase.Nodes](#), [ATLFBase.RefObject](#), [ATLFBase.Close\(\)](#), [ATLFBase.Dispose\(\)](#),  
[object.ToString\(\)](#), [object.Equals\(object\)](#), [object.Equals\(object, object\)](#),  
[object.ReferenceEquals\(object, object\)](#), [object.GetHashCode\(\)](#), [object.GetType\(\)](#),  
[object.MemberwiseClone\(\)](#)

Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#),  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#),  
[Object CB Extension.CompareType<T>\(object\)](#).

## Methods

Create(Stream)

```
public static ATLReader Create(Stream stream)
```

Parameters

stream [Stream](#)

Returns

[ATLReader](#)

## Create(TextReader)

```
public static ATLReader Create(TextReader text)
```

Parameters

text [TextReader](#)

Returns

[ATLReader](#)

## Create(string)

```
public static ATLReader Create(string filePath)
```

Parameters

filePath [string](#)

Returns

[ATLReader](#)

## Create(StringBuilder)

```
public static ATLReader Create(StringBuilder builder)
```

Parameters

builder [StringBuilder](#)

Returns

[ATLReader](#)

## Create<T>(Stream)

```
public static T Create<T>(Stream stream) where T : ATLFSBReader
```

Parameters

stream [Stream](#)

Returns

T

Type Parameters

T

## Create<T>(TextReader)

```
public static T Create<T>(TextReader text) where T : ATLFTBReader
```

Parameters

text [TextReader](#)

Returns

T

## Type Parameters

T

### Create<T>(string)

```
public static T Create<T>(string filePath) where T : ATLFSBReader
```

#### Parameters

filePath [String](#)

#### Returns

T

## Type Parameters

T

### Create<T>(StringBuilder)

```
public static T Create<T>(StringBuilder builder) where T : ATLFTBReader
```

#### Parameters

builder [StringBuilder](#)

#### Returns

T

## Type Parameters

T

## GetATLFDecoding(string)

Gets the ATLF encoding.

```
protected abstract ATLFDecoding GetATLFDecoding(string targetVersion)
```

Parameters

`targetVersion` [string](#)

Returns

[ATLFDecoding](#)

The method returns the encoder corresponding to the version passed in the `targetVersion` parameter. If the previous version does not exist, the default version will be returned.

## GetAllComments()

Gets all comments from the ATLF file.

```
public abstract ATLFNode[] GetAllComments()
```

Returns

[ATLFNode\[\]](#)

## GetEnumerator()

Gets all nodes within the buffer.

```
public abstract IEnumerator<ATLFNode> GetEnumerator()
```

Returns

[IEnumerator](#) <[ATLFNode](#)>

## GetHeader()

Gets the ATLF header tags.

```
public abstract ATLFNode[] GetHeader()
```

Returns

[ATLFNode\[\]](#)

## GetTag(string)

Gets the value of the tag.

```
public abstract string GetTag(string name)
```

Parameters

**name** [string](#)

The name of the target tag.

Returns

[string](#)

## GetTagGroup(string)

Gets a group of ATLF tags within a path.

```
public abstract ATLFNode[] GetTagGroup(string path)
```

Parameters

**path** [string](#)

Returns

## ATLFNode[]

Returns a list of ATLF tags according to a path. Example: there are three nodes with the name `com.cob.lib.tag1`, `com.cob.lib.tag2` and `com.cob.cli.tag-1` and passing in the `path` parameter `com.cob.lib` the tags `tag1` and `tag2` will be obtained.

## Reader()

Starts the process of reading the ATLF file.

```
public abstract void Reader()
```

# Class ATLFSBReader

Namespace: [Cobilas.IO.Atlf](#)

Assembly: Cobilas.Core.dll

```
public abstract class ATLFSBReader : ATLReader, IDisposable, IEnumerable<ATLFNode>,  
IEnumerable
```

## Inheritance

[object](#) ← [ATLFBase](#) ← [ATLFReader](#) ← [ATLFSBReader](#)

## Implements

[IDisposable](#), [IEnumerable](#)<[ATLFNode](#)>, [IEnumerable](#)

## Derived

[ATLFStreamReader](#)

## Inherited Members

[ATLFReader.Reader\(\)](#), [ATLFReader.GetHeader\(\)](#), [ATLFReader.GetTag\(string\)](#),  
[ATLFReader.GetAllComments\(\)](#), [ATLFReader.GetTagGroup\(string\)](#), [ATLFReader.GetEnumerator\(\)](#),  
[ATLFReader.GetATLFDecoding\(string\)](#), [ATLFReader.Create<T>\(Stream\)](#), [ATLFReader.Create<T>\(string\)](#),  
[ATLFReader.Create<T>\(TextReader\)](#), [ATLFReader.Create<T>\(StringBuilder\)](#), [ATLFReader.Create\(Stream\)](#),  
[ATLFReader.Create\(string\)](#), [ATLFReader.Create\(TextReader\)](#), [ATLFReader.Create\(StringBuilder\)](#),  
[ATLFBase.NodeCount](#), [ATLFBase.Indent](#), [ATLFBase.Encoding](#), [ATLFBase.TargetVersion](#), [ATLFBase.Closed](#),  
[ATLFBase.CloseFlow](#), [ATLFBase.Nodes](#), [ATLFBase.RefObject](#), [ATLFBase.Close\(\)](#), [ATLFBase.Dispose\(\)](#),  
[object.ToString\(\)](#), [object.Equals\(object\)](#), [object.Equals\(object, object\)](#),  
[object.ReferenceEquals\(object, object\)](#), [object.GetHashCode\(\)](#), [object.GetType\(\)](#),  
[object.MemberwiseClone\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#),  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#),  
[Object CB Extension.CompareType<T>\(object\)](#)

# Properties

## Stream

Represents the stream converted to [Stream](#).

```
protected abstract Stream Stream { get; set; }
```

Property Value

[Stream](#)

# Class ATLFSBWriter

Namespace: [Cobilas.IO.Atlf](#)

Assembly: Cobilas.Core.dll

```
public abstract class ATLFSBWriter : ATLFWriter, IDisposable
```

## Inheritance

[object](#) ← [ATLFBase](#) ← [ATLFWriter](#) ← [ATLFSBWriter](#)

## Implements

[IDisposable](#)

## Derived

[ATLFStreamWriter](#)

## Inherited Members

[ATLFWriter.IndentChars](#) , [ATLFWriter.Flush\(\)](#) , [ATLFWriter.WriteHeader\(\)](#) ,  
[ATLFWriter.WriteComment\(string\)](#) , [ATLFWriter.WriteWhitespace\(string\)](#) ,  
[ATLFWriter.WriteLine\(string, string\)](#) , [ATLFWriter.WriteWhitespace\(int, string\)](#) ,  
[ATLFWriter.GetATLFEncoding\(string\)](#) , [ATLFWriter.AddNode\(string, string, ATLFNodeType\)](#) ,  
[ATLFWriter.Create<T>\(Stream\)](#) , [ATLFWriter.Create<T>\(string\)](#) , [ATLFWriter.Create<T>\(TextWriter\)](#) ,  
[ATLFWriter.Create<T>\(StringBuilder\)](#) , [ATLFWriter.Create<T>\(StringBuilder, IFormatProvider\)](#) ,  
[ATLFWriter.Create\(Stream\)](#) , [ATLFWriter.Create\(string\)](#) , [ATLFWriter.Create\(TextWriter\)](#) ,  
[ATLFWriter.Create\(StringBuilder\)](#) , [ATLFWriter.Create\(StringBuilder, IFormatProvider\)](#) ,  
[ATLFBase.NodeCount](#) , [ATLFBase.Indent](#) , [ATLFBase.Encoding](#) , [ATLFBase.TargetVersion](#) , [ATLFBase.Closed](#) ,  
[ATLFBase.CloseFlow](#) , [ATLFBase.Nodes](#) , [ATLFBase.RefObject](#) , [ATLFBase.Close\(\)](#) , [ATLFBase.Dispose\(\)](#) ,  
[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#)

# Properties

## Stream

Represents the stream converted to [Stream](#).

```
protected abstract Stream Stream { get; set; }
```

Property Value

[Stream](#)

# Class ATLFStreamReader

Namespace: [Cobilas.IO.Atlf](#)

Assembly: Cobilas.Core.dll

```
public class ATLFStreamReader : ATLFSBReader, IDisposable, IEnumerable<ATLFNode>,  
IEnumerable
```

## Inheritance

[object](#) ← [ATLFBase](#) ← [ATLFReader](#) ← [ATLFSBReader](#) ← [ATLFStreamReader](#)

## Implements

[IDisposable](#), [IEnumerable](#)<[ATLFNode](#)>, [IEnumerable](#)

## Inherited Members

[ATLFReader.Create<T>\(Stream\)](#), [ATLFReader.Create<T>\(string\)](#), [ATLFReader.Create<T>\(TextReader\)](#),  
[ATLFReader.Create<T>\(StringBuilder\)](#), [ATLFReader.Create\(Stream\)](#), [ATLFReader.Create\(string\)](#),  
[ATLFReader.Create\(TextReader\)](#), [ATLFReader.Create\(StringBuilder\)](#), [object.ToString\(\)](#),  
[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.GetHashCode\(\)](#), [object.GetType\(\)](#), [object.MemberwiseClone\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#),  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#),  
[Object CB Extension.CompareType<T>\(object\)](#).

# Properties

## CloseFlow

This property is used to close the workflow automatically.

This property should be used in cases where you directly access a stream. Example: when a flow is called using the [File](#).Open(string) method.

```
protected override bool CloseFlow { get; set; }
```

Property Value

[bool](#)

## Closed

Indicate whether the flow has been closed.

```
public override bool Closed { get; protected set; }
```

Property Value

[bool](#)

## Encoding

Sets or returns the encoding used.

```
public override Encoding Encoding { get; set; }
```

Property Value

[Encoding](#)

## Indent

Determines whether text should be indented.

```
public override bool Indent { get; set; }
```

Property Value

[bool](#)

## NodeCount

The number of ATLF nodes stored.

```
public override long NodeCount { get; }
```

### Property Value

[long](#)

## Nodes

Where ATLF nodes are stored.

```
protected override ATLFNode[] Nodes { get; set; }
```

### Property Value

[ATLFNode\[\]](#)

## RefObject

Sets or returns the current stream.

```
protected override MarshalByRefObject RefObject { get; set; }
```

### Property Value

[MarshalByRefObject](#)

## Stream

Represents the stream converted to [Stream](#).

```
protected override Stream Stream { get; set; }
```

Property Value

[Stream ↗](#)

## TargetVersion

Represents the version that the ATLF object is using.

```
public override string TargetVersion { get; set; }
```

Property Value

[string ↗](#)

## Methods

### Close()

Allows you to close the current flow.

```
public override void Close()
```

### Dispose()

Allows you to discard the current stream.

```
public override void Dispose()
```

### Dispose(bool)

Performs an internal disposal of the object.

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

## Parameters

**disposing** [bool](#)

## ~ATLFStreamReader()

**protected** ~ATLFStreamReader()

## GetATLFDecoding(string)

Gets the ATLF encoding.

**protected override** ATLFDecoding [GetATLFDecoding\(string targetVersion\)](#)

## Parameters

**targetVersion** [string](#)

## Returns

[ATLFDecoding](#)

The method returns the encoder corresponding to the version passed in the **targetVersion** parameter. If the previous version does not exist, the default version will be returned.

## GetAllComments()

Gets all comments from the ATLF file.

**public override** ATLFNode[]  [GetAllComments\(\)](#)

## Returns

[ATLFNode\[\]](#)

## GetEnumerator()

Gets all nodes within the buffer.

```
public override IEnumerator<ATLFNode> GetEnumerator()
```

Returns

[IEnumerator](#)<[ATLFNode](#)>

## GetHeader()

Gets the ATLF header tags.

```
public override ATLFNode[] GetHeader()
```

Returns

[ATLFNode](#)[]

## GetTag(string)

Gets the value of the tag.

```
public override string GetTag(string name)
```

Parameters

**name** [string](#)

The name of the target tag.

Returns

[string](#)

## GetTagGroup(string)

Gets a group of ATLF tags within a path.

```
public override ATLFNode[] GetTagGroup(string path)
```

Parameters

path [string](#)

Returns

[ATLFNode\[\]](#)

Returns a list of ATLF tags according to a path. Example: there are three nodes with the name `com.cob.lib.tag1`, `com.cob.lib.tag2` and `com.cob.cli.tag-1` and passing in the `path` parameter `com.cob.lib` the tags `tag1` and `tag2` will be obtained.

## Reader()

Starts the process of reading the ATLF file.

```
public override void Reader()
```

# Class ATLFStreamWriter

Namespace: [Cobilas.IO.Atlf](#)

Assembly: Cobilas.Core.dll

```
public class ATLFStreamWriter : ATLFSBWriter, IDisposable
```

## Inheritance

[object](#) ← [ATLFBase](#) ← [ATLFWriter](#) ← [ATLFSBWriter](#) ← [ATLFStreamWriter](#)

## Implements

[IDisposable](#)

## Inherited Members

[ATLFWriter.Create<T>\(Stream\)](#) , [ATLFWriter.Create<T>\(string\)](#) , [ATLFWriter.Create<T>\(TextWriter\)](#) ,  
[ATLFWriter.Create<T>\(StringBuilder\)](#) , [ATLFWriter.Create<T>\(StringBuilder, IFormatProvider\)](#) ,  
[ATLFWriter.Create\(Stream\)](#) , [ATLFWriter.Create\(string\)](#) , [ATLFWriter.Create\(TextWriter\)](#) ,  
[ATLFWriter.Create\(StringBuilder\)](#) , [ATLFWriter.Create\(StringBuilder, IFormatProvider\)](#) , [object.ToString\(\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#)

# Properties

## CloseFlow

This property is used to close the workflow automatically.

This property should be used in cases where you directly access a stream. Example: when a flow is called using the [File](#).Open(string) method.

```
protected override bool CloseFlow { get; set; }
```

Property Value

[bool](#) ↗

## Closed

Indicate whether the flow has been closed.

```
public override bool Closed { get; protected set; }
```

Property Value

[bool](#) ↗

## Encoding

Sets or returns the encoding used.

```
public override Encoding Encoding { get; set; }
```

Property Value

[Encoding](#) ↗

## Indent

Determines whether text should be indented.

```
public override bool Indent { get; set; }
```

Property Value

[bool](#) ↗

## IndentChars

Represents the character of indentation.

```
public override string IndentChars { get; set; }
```

Property Value

[string](#)

## NodeCount

The number of ATLF nodes stored.

```
public override long NodeCount { get; }
```

Property Value

[long](#)

## Nodes

Where ATLF nodes are stored.

```
protected override ATLFNode[] Nodes { get; set; }
```

Property Value

[ATLFNode\[\]](#)

## RefObject

Sets or returns the current stream.

```
protected override MarshalByRefObject RefObject { get; set; }
```

## Property Value

[MarshalByRefObject](#)

## Stream

Represents the stream converted to [Stream](#).

```
protected override Stream Stream { get; set; }
```

## Property Value

[Stream](#)

## TargetVersion

Represents the version that the ATLF object is using.

```
public override string TargetVersion { get; set; }
```

## Property Value

[string](#)

## Methods

### AddNode(string, string, ATLFNodeType)

Adds a new node to the buffer.

```
protected override void AddNode(string name, string value, ATLFNodeType nodeType)
```

## Parameters

name [string](#)

value [string](#)

`nodeType` [ATLFNodeType](#)

## Close()

Allows you to close the current flow.

```
public override void Close()
```

## Dispose()

Allows you to discard the current stream.

```
public override void Dispose()
```

## Dispose(bool)

Performs an internal disposal of the object.

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

### Parameters

`disposing` [bool](#)

## ~ATLFStreamWriter()

```
protected ~ATLFStreamWriter()
```

## Flush()

When overridden in a derived class, clears all buffers for this stream and causes any buffered data to be written to the underlying device.

```
public override void Flush()
```

## GetATLFEencoding(string)

Gets the ATLF encoding.

```
protected override ATLFEencoding GetATLFEencoding(string targetVersion)
```

Parameters

`targetVersion` [string](#)

Returns

[ATLFEencoding](#)

The method returns the encoder corresponding to the version passed in the `targetVersion` parameter. If the previous version does not exist, the default version will be returned.

## WriteComment(string)

Write a comment in the stream.

```
public override void WriteComment(string value)
```

Parameters

`value` [string](#)

Write the message.

## WriteHeader()

Writes the atlf header to the stream.

```
public override void WriteHeader()
```

## WriteIndentation()

Performs automatic indentation.

```
protected void WriteIndentation()
```

## WriteNode(string, string)

Writes an ATLF node to the stream.

```
public override void WriteNode(string name, string value)
```

Parameters

`name` [string](#)

`value` [string](#)

## WriteWhitespace(int, string)

Writes an escape character to the stream.

```
public override void WriteWhitespace(int count, string spacing)
```

Parameters

`count` [int](#)

`spacing` [string](#)

## WriteWhitespace(string)

Writes an escape character to the stream.

```
public override void WriteWhitespace(string spacing)
```

## Parameters

**spacing** [string](#) ↗

# Class ATLFTBReader

Namespace: [Cobilas.IO.Atlf](#)

Assembly: Cobilas.Core.dll

```
public abstract class ATLFTBReader : ATLReader, IDisposable, IEnumerable<ATLFNode>,  
IEnumerable
```

## Inheritance

[object](#) ← [ATLFBase](#) ← [ATLFReader](#) ← [ATLFTBReader](#)

## Implements

[IDisposable](#), [IEnumerable](#)<[ATLFNode](#)>, [IEnumerable](#)

## Derived

[ATLFTTextReader](#)

## Inherited Members

[ATLFReader.Reader\(\)](#), [ATLFReader.GetHeader\(\)](#), [ATLFReader.GetTag\(string\)](#),  
[ATLFReader.GetAllComments\(\)](#), [ATLFReader.GetTagGroup\(string\)](#), [ATLFReader.GetEnumerator\(\)](#),  
[ATLFReader.GetATLFDecoding\(string\)](#), [ATLFReader.Create<T>\(Stream\)](#), [ATLFReader.Create<T>\(string\)](#),  
[ATLFReader.Create<T>\(TextReader\)](#), [ATLFReader.Create<T>\(StringBuilder\)](#), [ATLFReader.Create\(Stream\)](#),  
[ATLFReader.Create\(string\)](#), [ATLFReader.Create\(TextReader\)](#), [ATLFReader.Create\(StringBuilder\)](#),  
[ATLFBase.NodeCount](#), [ATLFBase.Indent](#), [ATLFBase.Encoding](#), [ATLFBase.TargetVersion](#), [ATLFBase.Closed](#),  
[ATLFBase.CloseFlow](#), [ATLFBase.Nodes](#), [ATLFBase.RefObject](#), [ATLFBase.Close\(\)](#), [ATLFBase.Dispose\(\)](#),  
[object.ToString\(\)](#), [object.Equals\(object\)](#), [object.Equals\(object, object\)](#),  
[object.ReferenceEquals\(object, object\)](#), [object.GetHashCode\(\)](#), [object.GetType\(\)](#),  
[object.MemberwiseClone\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#),  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#),  
[Object CB Extension.CompareType<T>\(object\)](#)

# Properties

# Stream

Represents the stream converted to [TextWriter](#).

```
protected abstract TextReader Stream { get; set; }
```

Property Value

[TextReader](#)

# Class ATLFTBWriter

Namespace: [Cobilas.IO.Atlf](#)

Assembly: Cobilas.Core.dll

```
public abstract class ATLFTBWriter : ATLFWriter, IDisposable
```

## Inheritance

[object](#) ← [ATLFBase](#) ← [ATLFWriter](#) ← [ATLFTBWriter](#)

## Implements

[IDisposable](#)

## Derived

[ATLFTTextWriter](#)

## Inherited Members

[ATLFWriter.IndentChars](#) , [ATLFWriter.Flush\(\)](#) , [ATLFWriter.WriteHeader\(\)](#) ,  
[ATLFWriter.WriteComment\(string\)](#) , [ATLFWriter.WriteWhitespace\(string\)](#) ,  
[ATLFWriter.WriteLine\(string, string\)](#) , [ATLFWriter.WriteWhitespace\(int, string\)](#) ,  
[ATLFWriter.GetATLFEncoding\(string\)](#) , [ATLFWriter.AddNode\(string, string, ATLFNodeType\)](#) ,  
[ATLFWriter.Create<T>\(Stream\)](#) , [ATLFWriter.Create<T>\(string\)](#) , [ATLFWriter.Create<T>\(TextWriter\)](#) ,  
[ATLFWriter.Create<T>\(StringBuilder\)](#) , [ATLFWriter.Create<T>\(StringBuilder, IFormatProvider\)](#) ,  
[ATLFWriter.Create\(Stream\)](#) , [ATLFWriter.Create\(string\)](#) , [ATLFWriter.Create\(TextWriter\)](#) ,  
[ATLFWriter.Create\(StringBuilder\)](#) , [ATLFWriter.Create\(StringBuilder, IFormatProvider\)](#) ,  
[ATLFBase.NodeCount](#) , [ATLFBase.Indent](#) , [ATLFBase.Encoding](#) , [ATLFBase.TargetVersion](#) , [ATLFBase.Closed](#) ,  
[ATLFBase.CloseFlow](#) , [ATLFBase.Nodes](#) , [ATLFBase.RefObject](#) , [ATLFBase.Close\(\)](#) , [ATLFBase.Dispose\(\)](#) ,  
[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#)

# Properties

## Stream

Represents the stream converted to [TextWriter](#).

```
protected abstract TextWriter Stream { get; set; }
```

Property Value

[TextWriter](#)

# Class ATLFTTextReader

Namespace: [Cobilas.IO.Atlf](#)

Assembly: Cobilas.Core.dll

```
public class ATLFTTextReader : ATLFTBReader, IDisposable, IEnumerable<ATLFNode>, IEnumerable
```

## Inheritance

[object](#) ← [ATLFBase](#) ← [ATLFReader](#) ← [ATLFTBReader](#) ← [ATLFTTextReader](#)

## Implements

[IDisposable](#), [IEnumerable](#)<[ATLFNode](#)>, [IEnumerable](#)

## Inherited Members

[ATLFReader.Create<T>\(Stream\)](#), [ATLFReader.Create<T>\(string\)](#), [ATLFReader.Create<T>\(TextReader\)](#),  
[ATLFReader.Create<T>\(StringBuilder\)](#), [ATLFReader.Create\(Stream\)](#), [ATLFReader.Create\(string\)](#),  
[ATLFReader.Create\(TextReader\)](#), [ATLFReader.Create\(StringBuilder\)](#), [object.ToString\(\)](#),  
[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.GetHashCode\(\)](#), [object.GetType\(\)](#), [object.MemberwiseClone\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#),  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#),  
[Object CB Extension.CompareType<T>\(object\)](#).

# Properties

## CloseFlow

This property is used to close the workflow automatically.

This property should be used in cases where you directly access a stream. Example: when a flow is called using the [File](#).Open(string) method.

```
protected override bool CloseFlow { get; set; }
```

Property Value

[bool](#)

## Closed

Indicate whether the flow has been closed.

```
public override bool Closed { get; protected set; }
```

Property Value

[bool](#)

## Encoding

Sets or returns the encoding used.

```
public override Encoding Encoding { get; set; }
```

Property Value

[Encoding](#)

## Indent

Determines whether text should be indented.

```
public override bool Indent { get; set; }
```

Property Value

[bool](#)

## NodeCount

The number of ATLF nodes stored.

```
public override long NodeCount { get; }
```

Property Value

[long](#)

## Nodes

Where ATLF nodes are stored.

```
protected override ATLFNode[] Nodes { get; set; }
```

Property Value

[ATLFNode\[\]](#)

## RefObject

Sets or returns the current stream.

```
protected override MarshalByRefObject RefObject { get; set; }
```

Property Value

[MarshalByRefObject](#)

## Stream

Represents the stream converted to [TextWriter](#).

```
protected override TextReader Stream { get; set; }
```

## Property Value

[TextReader ↗](#)

## TargetVersion

Represents the version that the ATLF object is using.

```
public override string TargetVersion { get; set; }
```

## Property Value

[string ↗](#)

# Methods

## Close()

Allows you to close the current flow.

```
public override void Close()
```

## Dispose()

Allows you to discard the current stream.

```
public override void Dispose()
```

## Dispose(bool)

Performs an internal disposal of the object.

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

## Parameters

**disposing** [bool](#)

## ~ATLFTextReader()

**protected** ~ATLFTextReader()

## GetATLFDecoding(string)

Gets the ATLF encoding.

**protected override** ATLFDecoding [GetATLFDecoding\(string targetVersion\)](#)

## Parameters

**targetVersion** [string](#)

## Returns

[ATLFDecoding](#)

The method returns the encoder corresponding to the version passed in the **targetVersion** parameter. If the previous version does not exist, the default version will be returned.

## GetAllComments()

Gets all comments from the ATLF file.

**public override** ATLFNode[]  [GetAllComments\(\)](#)

## Returns

[ATLFNode\[\]](#)

## GetEnumerator()

Gets all nodes within the buffer.

```
public override IEnumerator<ATLFNode> GetEnumerator()
```

Returns

[IEnumerator](#)<[ATLFNode](#)>

## GetHeader()

Gets the ATLF header tags.

```
public override ATLFNode[] GetHeader()
```

Returns

[ATLFNode](#)[]

## GetTag(string)

Gets the value of the tag.

```
public override string GetTag(string name)
```

Parameters

**name** [string](#)

The name of the target tag.

Returns

[string](#)

## GetTagGroup(string)

Gets a group of ATLF tags within a path.

```
public override ATLFNode[] GetTagGroup(string path)
```

Parameters

path [string](#)

Returns

[ATLFNode\[\]](#)

Returns a list of ATLF tags according to a path. Example: there are three nodes with the name `com.cob.lib.tag1`, `com.cob.lib.tag2` and `com.cob.cli.tag-1` and passing in the `path` parameter `com.cob.lib` the tags `tag1` and `tag2` will be obtained.

## Reader()

Starts the process of reading the ATLF file.

```
public override void Reader()
```

# Class ATLTextWriter

Namespace: [Cobilas.IO.Atlf](#)

Assembly: Cobilas.Core.dll

```
public class ATLTextWriter : ATLFTBWriter, IDisposable
```

## Inheritance

[object](#) ← ATLFBBase ← ATLFWriter ← ATLFTBWriter ← ATLTextWriter

## Implements

[IDisposable](#)

## Inherited Members

[ATLFWriter.Create<T>\(Stream\)](#) , [ATLFWriter.Create<T>\(string\)](#) , [ATLFWriter.Create<T>\(TextWriter\)](#) ,  
[ATLFWriter.Create<T>\(StringBuilder\)](#) , [ATLFWriter.Create<T>\(StringBuilder, IFormatProvider\)](#) ,  
[ATLFWriter.Create\(Stream\)](#) , [ATLFWriter.Create\(string\)](#) , [ATLFWriter.Create\(TextWriter\)](#) ,  
[ATLFWriter.Create\(StringBuilder\)](#) , [ATLFWriter.Create\(StringBuilder, IFormatProvider\)](#) , [object.ToString\(\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#)

# Properties

## CloseFlow

This property is used to close the workflow automatically.

This property should be used in cases where you directly access a stream. Example: when a flow is called using the [File](#).Open(string) method.

```
protected override bool CloseFlow { get; set; }
```

Property Value

[bool](#)

## Closed

Indicate whether the flow has been closed.

```
public override bool Closed { get; protected set; }
```

Property Value

[bool](#)

## Encoding

Sets or returns the encoding used.

```
public override Encoding Encoding { get; set; }
```

Property Value

[Encoding](#)

## Indent

Determines whether text should be indented.

```
public override bool Indent { get; set; }
```

Property Value

[bool](#)

## IndentChars

Represents the character of indentation.

```
public override string IndentChars { get; set; }
```

Property Value

[string](#)

## NodeCount

The number of ATLF nodes stored.

```
public override long NodeCount { get; }
```

Property Value

[long](#)

Returns the number of nodes already written.

## Nodes

Where ATLF nodes are stored.

```
protected override ATLFNode[] Nodes { get; set; }
```

Property Value

[ATLFNode\[\]](#)

## RefObject

Sets or returns the current stream.

```
protected override MarshalByRefObject RefObject { get; set; }
```

## Property Value

[MarshalByRefObject](#)

## Stream

Represents the stream converted to [TextWriter](#).

```
protected override TextWriter Stream { get; set; }
```

## Property Value

[TextWriter](#)

## TargetVersion

Represents the version that the ATLF object is using.

```
public override string TargetVersion { get; set; }
```

## Property Value

[string](#)

By default the value is [string](#).Empty which represents the current version.

## Methods

### AddNode(string, string, ATLFNodeType)

Adds a new node to the buffer.

```
protected override void AddNode(string name, string value, ATLFNodeType nodeType)
```

## Parameters

name [string](#)

value [string](#)

nodeType [ATLFNodeType](#)

## Close()

Allows you to close the current flow.

```
public override void Close()
```

## Dispose()

Allows you to discard the current stream.

```
public override void Dispose()
```

## Dispose(bool)

Performs an internal disposal of the object.

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

## Parameters

disposing [bool](#)

## ~ATLFTextWriter()

```
protected ~ATLFTextWriter()
```

## Flush()

When overridden in a derived class, clears all buffers for this stream and causes any buffered data to be written to the underlying device.

```
public override void Flush()
```

## GetATLFEencoding(string)

Gets the ATLF encoding.

```
protected override ATLFEencoding GetATLFEencoding(string targetVersion)
```

Parameters

**targetVersion** [string](#)

Returns

[ATLFEencoding](#)

The method returns the encoder corresponding to the version passed in the **targetVersion** parameter. If the previous version does not exist, the default version will be returned.

## WriteComment(string)

Write a comment in the stream.

```
public override void WriteComment(string value)
```

Parameters

**value** [string](#)

Write the message.

## WriteHeader()

Writes the atlf header to the stream.

```
public override void WriteHeader()
```

## WriteIndentation()

Performs automatic indentation.

```
protected void WriteIndentation()
```

## WriteNode(string, string)

Writes an ATLF node to the stream.

```
public override void WriteNode(string name, string value)
```

Parameters

`name` [string](#)

`value` [string](#)

## WriteWhitespace(int, string)

Writes an escape character to the stream.

```
public override void WriteWhitespace(int count, string spacing)
```

Parameters

`count` [int](#)

`spacing` [string](#)

## WriteWhitespace(string)

Writes an escape character to the stream.

```
public override void WriteWhitespace(string spacing)
```

### Parameters

spacing [string](#)

# Class ATLFWriter

Namespace: [Cobilas.IO.Atlf](#)

Assembly: Cobilas.Core.dll

Base class for ATLF writing classes.

```
public abstract class ATLFWriter : ATLFBase, IDisposable
```

Inheritance

[object](#) ↗ ← [ATLFBase](#) ← ATLFWriter

Implements

[IDisposable](#) ↗

Derived

[ATLFSBWriter](#), [ATLFTBWriter](#)

Inherited Members

[ATLFBase.NodeCount](#), [ATLFBase.Indent](#), [ATLFBase.Encoding](#), [ATLFBase.TargetVersion](#), [ATLFBase.Closed](#),  
[ATLFBase.CloseFlow](#), [ATLFBase.Nodes](#), [ATLFBase.RefObject](#), [ATLFBase.Close\(\)](#), [ATLFBase.Dispose\(\)](#),  
[object.ToString\(\)](#) ↗, [object.Equals\(object\)](#) ↗, [object.Equals\(object, object\)](#) ↗,  
[object.ReferenceEquals\(object, object\)](#) ↗, [object.GetHashCode\(\)](#) ↗, [object.GetType\(\)](#) ↗,  
[object.MemberwiseClone\(\)](#) ↗

Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#).

## Properties

### IndentChars

Represents the character of indentation.

```
public abstract string IndentChars { get; set; }
```

## Property Value

[string](#)

## Methods

### AddNode(string, string, ATLFNodeType)

Adds a new node to the buffer.

```
protected abstract void AddNode(string name, string value, ATLFNodeType nodeType)
```

#### Parameters

name [string](#)

value [string](#)

nodeType [ATLFNodeType](#)

### Create(Stream)

```
public static ATLFWriter Create(Stream stream)
```

#### Parameters

stream [Stream](#)

#### Returns

[ATLFWriter](#)

### Create(TextWriter)

```
public static ATLFWriter Create(TextWriter text)
```

Parameters

text [TextWriter](#)

Returns

[ATLFWriter](#)

## Create(string)

```
public static ATLFWriter Create(string filePath)
```

Parameters

filePath [string](#)

Returns

[ATLFWriter](#)

## Create(StringBuilder)

```
public static ATLFWriter Create(StringBuilder text)
```

Parameters

text [StringBuilder](#)

Returns

[ATLFWriter](#)

## Create(StringBuilder, IFormatProvider)

```
public static ATLFWriter Create(StringBuilder text, IFormatProvider formatProvider)
```

Parameters

text [StringBuilder](#)

formatProvider [IFormatProvider](#)

Returns

[ATLFWriter](#)

## Create<T>(Stream)

```
public static T Create<T>(Stream stream) where T : ATLFSBWriter
```

Parameters

stream [Stream](#)

Returns

T

Type Parameters

T

## Create<T>(TextWriter)

```
public static T Create<T>(TextWriter text) where T : ATLFTBWriter
```

Parameters

text [TextWriter](#)

Returns

T

Type Parameters

T

## Create<T>(string)

```
public static T Create<T>(string filePath) where T : ATLFSBWriter
```

Parameters

filePath [String](#)

Returns

T

Type Parameters

T

## Create<T>(StringBuilder)

```
public static T Create<T>(StringBuilder text) where T : ATLFTBWriter
```

Parameters

text [StringBuilder](#)

Returns

T

Type Parameters

T

## Create<T>(StringBuilder, IFormatProvider)

```
public static T Create<T>(StringBuilder text, IFormatProvider formatProvider) where T : ATLFTBWriter
```

### Parameters

**text** [StringBuilder](#)

**formatProvider** [IFormatProvider](#)

### Returns

T

### Type Parameters

T

## Flush()

When overridden in a derived class, clears all buffers for this stream and causes any buffered data to be written to the underlying device.

```
public abstract void Flush()
```

## GetATLFEncoding(string)

Gets the ATLF encoding.

```
protected abstract ATLFEncoding GetATLFEncoding(string targetVersion)
```

### Parameters

`targetVersion` [string ↗](#)

Returns

[ATLFEencoding](#)

The method returns the encoder corresponding to the version passed in the `targetVersion` parameter. If the previous version does not exist, the default version will be returned.

## WriteComment(string)

Write a comment in the stream.

```
public abstract void WriteComment(string value)
```

Parameters

`value` [string ↗](#)

Write the message.

## WriteHeader()

Writes the atlf header to the stream.

```
public abstract void WriteHeader()
```

## WriteNode(string, string)

Writes an ATLF node to the stream.

```
public abstract void WriteNode(string name, string value)
```

Parameters

`name` [string ↗](#)

**value** [string](#)

## WriteWhitespace(int, string)

Writes an escape character to the stream.

```
public abstract void WriteWhitespace(int count, string value)
```

### Parameters

**count** [int](#)

**value** [string](#)

## WriteWhitespace(string)

Writes an escape character to the stream.

```
public abstract void WriteWhitespace(string value)
```

### Parameters

**value** [string](#)

# Namespace Cobilas.IO.Atlf.Components

## Classes

### [CharacterCursor](#)

Allows you to read and modify a string of characters.

## Structs

### [CharacterCursor.LineEndColumn](#)

Represents the current column, row and index of the reading cursor.

# Class CharacterCursor

Namespace: [Cobilas.IO.Atlf.Components](#)

Assembly: Cobilas.Core.dll

Allows you to read and modify a string of characters.

```
public sealed class CharacterCursor : IDisposable
```

## Inheritance

[object](#) ← CharacterCursor

## Implements

[IDisposable](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#)

# Constructors

## CharacterCursor(byte[], Encoding)

```
public CharacterCursor(byte[] bytes, Encoding encoding)
```

## Parameters

bytes [byte](#)[]

encoding [Encoding](#)

## CharacterCursor(char[])

```
public CharacterCursor(char[] characters)
```

Parameters

characters [char](#)[]

## CharacterCursor(string)

```
public CharacterCursor(string text)
```

Parameters

text [string](#)

## CharacterCursor(StringBuilder)

```
public CharacterCursor(StringBuilder text)
```

Parameters

text [StringBuilder](#)

# Properties

## Column

Returns the current reading cursor column.

```
public long Column { get; }
```

## Property Value

[long ↗](#)

## Count

Returns the length of the string.

```
public long Count { get; }
```

## Property Value

[long ↗](#)

## CurrentCharacter

Returns the current character.

```
public char CurrentCharacter { get; }
```

## Property Value

[char ↗](#)

## Cursor

Returns a current representation of the row, column, and index of the string being read.

```
public CharacterCursor.LineEndColumn Cursor { get; }
```

## Property Value

[CharacterCursor.LineEndColumn](#)

## Index

Returns the current index of the string being read.

```
public long Index { get; }
```

Property Value

[long](#)

## this[long]

```
public char this[long index] { get; }
```

Parameters

index [long](#)

Property Value

[char](#)

## Line

Returns the current line of the reading cursor.

```
public long Line { get; }
```

Property Value

[long](#)

# Methods

## AddEscape(char)

Add escape character.

```
public void AddEscape(char escape)
```

Parameters

escape [char](#)

## CharIsEqualToIndex(char)

Compare a character with the current character.

```
public bool CharIsEqualToIndex(char character)
```

Parameters

character [char](#)

Returns

[bool](#)

## CharIsEqualToIndex(params char[])

Compare a character with the current character.

```
public bool CharIsEqualToIndex(params char[] characters)
```

Parameters

characters [char](#)[]

Returns

[bool](#)

## CharIsEqualToIndex(string)

Compare a character with the current character.

```
public bool CharIsEqualToIndex(string text)
```

Parameters

**text** [string](#)

Returns

[bool](#)

## CharIsEqualToIndex(params string[])

Compare a character with the current character.

```
public bool CharIsEqualToIndex(params string[] texts)
```

Parameters

**texts** [string](#)[]

Returns

[bool](#)

## Dispose()

Performs application-defined tasks associated with freeing, releasing, or resetting unmanaged resources.

```
public void Dispose()
```

## MoveToCharacter()

Move the reading cursor to the next character.

```
public bool MoveToCharacter()
```

Returns

[bool](#)

The method will return true whenever there are characters to read.

## MoveToCharacter(long)

Move the reading cursor to the next character.

```
public bool MoveToCharacter(long index)
```

Parameters

[index](#) [long](#)

How many jumps the cursor must make to move to the next character.

Returns

[bool](#)

The method will return true whenever there are characters to read.

## Reset()

Reset the position of the index, line and column of the reading course.

```
public void Reset()
```

## SliceText(long, long)

Allows you to make a cut in the character string.

```
public string SliceText(long index, long count)
```

## Parameters

**index** [long](#)

The index that will start the clipping.

**count** [long](#)

The number of characters that will be cut.

## Returns

[string](#)

The method will return a string containing the string of characters that were cut.

## ToString()

Returns a string that represents the current object.

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

## Returns

[string](#)

A string that represents the current object.

# Struct CharacterCursor.LineEndColumn

Namespace: [Cobilas.IO.Atlf.Components](#)

Assembly: Cobilas.Core.dll

Represents the current column, row and index of the reading cursor.

```
public readonly struct CharacterCursor.LineEndColumn
```

## Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetType\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#)

## Constructors

### LineEndColumn(long, long, long)

```
public LineEndColumn(long line, long column, long index)
```

#### Parameters

line [long](#)

column [long](#)

index [long](#)

## Properties

## Column

Returns the column where the reading cursor is.

```
public long Column { get; }
```

Property Value

[long](#)

## Default

Default value.(L:1, C:1, I:0)

```
public static CharacterCursor.LineEndColumn Default { get; }
```

Property Value

[CharacterCursor.LineEndColumn](#)

## Index

Returns the index of where the reading cursor is.

```
public long Index { get; }
```

Property Value

[long](#)

## Line

Returns the line where the reading cursor is.

```
public long Line { get; }
```

Property Value

[long](#) ↗

## Methods

### ToString()

Returns the fully qualified type name of this instance.

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

Returns

[string](#) ↗

The fully qualified type name.

# Namespace Coblas.IO.Atlf.Text

## Classes

### [ATLFDecoding](#)

Decoding base class.

### [ATLFEencoding](#)

Encoding base class.

### [ATLFVS10Decoding](#)

### [ATLFVS10Encoding](#)

### [EncodingsCollection](#)

Represents a collection of ATLF encoding and decoding.

### [NullATLFDecoding](#)

### [NullATLFEencoding](#)

# Class ATLFDecoding

Namespace: [Cobilas.IO.Atlf.Text](#)

Assembly: Cobilas.Core.dll

Decoding base class.

```
public abstract class ATLFDecoding
```

Inheritance

[object](#) ← ATLFDecoding

Derived

[ATLFVS10Decoding](#), [NullATLFDecoding](#)

Inherited Members

[object.ToString\(\)](#), [object.Equals\(object\)](#), [object.Equals\(object, object\)](#),  
[object.ReferenceEquals\(object, object\)](#), [object.GetHashCode\(\)](#), [object.GetType\(\)](#),  
[object.MemberwiseClone\(\)](#)

Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#),  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#),  
[Object CB Extension.CompareType<T>\(object\)](#).

## Fields

### Null

Represents a null value of type [NullATLFDecoding](#).

```
public static readonly ATLFDecoding Null
```

### Field Value

## Properties

### Version

Represents the decoder version.

```
public abstract string Version { get; }
```

### Property Value

[string](#)

## Methods

### Reader(params object[])

Read a list of objects.

```
public abstract ATLFNode[] Reader(params object[] args)
```

### Parameters

args [object](#)[]

### Returns

[ATLFNode](#)[]

### Reader4Byte(params object[])

Read a list of objects.

```
public abstract ATLFNode[] Reader4Byte(params object[] args)
```

Parameters

args [object](#)[]

Returns

[ATLFNode](#)[]

## ValidCharacter(char)

Checks whether the character is valid.

```
protected abstract bool ValidCharacter(char c)
```

Parameters

c [char](#)

Returns

[bool](#)

# Class ATLFEncoding

Namespace: [Cobilas.IO.Atlf.Text](#)

Assembly: Cobilas.Core.dll

Encoding base class.

```
public abstract class ATLFEncoding
```

## Inheritance

[object](#) ← ATLFEncoding

## Derived

[ATLFVS10Encoding](#), [NullATLFEncoding](#)

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#).

# Fields

## Null

Represents a null value of type [NullATLFEncoding](#).

```
public static readonly ATLFEncoding Null
```

## Field Value

## Properties

### Version

Represents the encoder version.

```
public abstract string Version { get; }
```

### Property Value

[string](#)

## Methods

### Writer(params object[])

Writes a list of objects.

```
public abstract string Writer(params object[] args)
```

### Parameters

args [object](#)[]

### Returns

[string](#)

### Writer4Byte(params object[])

Writes a list of objects.

```
public abstract byte[] Writer4Byte(params object[] args)
```

Parameters

`args` [object](#)[]

Returns

[byte](#)[]

# Class ATLFVS10Decoding

Namespace: [Cobilas.IO.Atlf.Text](#)

Assembly: Cobilas.Core.dll

```
public class ATLFVS10Decoding : ATLFDecoding
```

## Inheritance

[object](#) ← [ATLFDecoding](#) ← ATLFVS10Decoding

## Inherited Members

[ATLFDecoding.Null](#) , [object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#)

## Properties

### Version

Represents the decoder version.

```
public override string Version { get; }
```

### Property Value

[string](#)

# Methods

## GetComment(CharacterCursor)

```
protected virtual ATLFNode GetComment(CharacterCursor cursor)
```

### Parameters

cursor [CharacterCursor](#)

### Returns

[ATLFNode](#)

## GetTag(CharacterCursor)

```
protected virtual ATLFNode GetTag(CharacterCursor cursor)
```

### Parameters

cursor [CharacterCursor](#)

### Returns

[ATLFNode](#)

## GetTag(CharacterCursor, ATLFNodeType)

```
protected virtual ATLFNode GetTag(CharacterCursor cursor, ATLFNodeType nodeType)
```

### Parameters

cursor [CharacterCursor](#)

nodeType [ATLFNodeType](#)

Returns

[ATLFNode](#)

## Reader(CharacterCursor)

`protected virtual ATLFNode[] Reader(CharacterCursor cursor)`

Parameters

`cursor` [CharacterCursor](#)

Returns

[ATLFNode\[\]](#)

## Reader(params object[])

Read a list of objects.

`public override ATLFNode[] Reader(params object[] args)`

Parameters

`args` [object](#)[]

`args[0]` = [string](#)

Returns

[ATLFNode\[\]](#)

## Reader4Byte(params object[])

Read a list of objects.

```
public override ATLFNode[] Reader4Byte(params object[] args)
```

Parameters

args [object](#)[]

args[0] = [byte](#)[]

args[1] = [Encoding](#)

Returns

[ATLFNode](#)[]

## ValidCharacter(char)

Checks whether the character is valid.

```
protected override bool ValidCharacter(char c)
```

Parameters

c [char](#)

Returns

[bool](#)

# Class ATLFS10Encoding

Namespace: [Cobilas.IO.Atlf.Text](#)

Assembly: Cobilas.Core.dll

```
public class ATLFS10Encoding : ATLFFEncoding
```

## Inheritance

[object](#) ← [ATLFFEncoding](#) ← ATLFS10Encoding

## Inherited Members

[ATLFFEncoding.Null](#) , [object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#)

## Properties

### Version

Represents the encoder version.

```
public override string Version { get; }
```

### Property Value

[string](#)

# Methods

## Writer(ATLFNode[])

```
protected virtual string Writer(ATLFNode[] nodes)
```

### Parameters

nodes [ATLFNode\[\]](#)

### Returns

[string](#)

## Writer(params object[])

Writes a list of objects.

```
public override string Writer(params object[] args)
```

### Parameters

args [object\[\]](#)

args[0] = [ATLFNode\[\]](#)

### Returns

[string](#)

## Writer4Byte(params object[])

Writes a list of objects.

```
public override byte[] Writer4Byte(params object[] args)
```

### Parameters

args [object](#)[]

args[0] = [ATLFNode](#)[]

args[1] = [Encoding](#)[]

Returns

[byte](#)[]

# Class EncodingsCollection

Namespace: [Cobilas.IO.ATLF.Text](#)

Assembly: Cobilas.Core.dll

Represents a collection of ATLF encoding and decoding.

```
public static class EncodingsCollection
```

## Inheritance

[object](#) ← EncodingsCollection

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### ContainsDecoding(string)

Checks whether a certain version of the ATLF decoder exists.

```
public static bool ContainsDecoding(string version)
```

#### Parameters

version [string](#)

#### Returns

[bool](#)

### ContainsEncoding(string)

Checks whether a given version of the ATLF encoder exists.

```
public static bool ContainsEncoding(string version)
```

Parameters

version [string](#)

Returns

[bool](#)

## GetDecoding(string)

Gets a specific version of the ATLF decoder.

```
public static ATLFDecoding GetDecoding(string version)
```

Parameters

version [string](#)

Returns

[ATLFDecoding](#)

## GetDecodingVersionList()

Gets a list of ATLF decoder versions.

```
public static string[] GetDecodingVersionList()
```

Returns

[string](#)[]

## GetEncoding(string)

Gets a specific version of the ATLF encoder.

```
public static ATLFEncoding GetEncoding(string version)
```

Parameters

**version** [string](#)

Returns

[ATLFEncoding](#)

## GetEncodingVersionList()

Gets a list of ATLF encoder versions.

```
public static string[] GetEncodingVersionList()
```

Returns

[string](#)[]

# Class NullATLFDecoding

Namespace: [Cobilas.IO.Atlf.Text](#)

Assembly: Cobilas.Core.dll

```
public sealed class NullATLFDecoding : ATLFDecoding
```

## Inheritance

[object](#) ← [ATLFDecoding](#) ← NullATLFDecoding

## Inherited Members

[ATLFDecoding.Null](#) , [object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#)

# Properties

## Version

Represents the decoder version.

```
public override string Version { get; }
```

## Property Value

[string](#)

# Methods

## Reader(params object[])

Read a list of objects.

```
public override ATLFNode[] Reader(params object[] args)
```

Parameters

args [object](#)[]

Returns

[ATLFNode](#)[]

## Reader4Byte(params object[])

Read a list of objects.

```
public override ATLFNode[] Reader4Byte(params object[] args)
```

Parameters

args [object](#)[]

Returns

[ATLFNode](#)[]

## ValidCharacter(char)

Checks whether the character is valid.

```
protected override bool ValidCharacter(char c)
```

Parameters

c [char](#)

Returns

bool ↗

# Class NullATLFEencoding

Namespace: [Cobilas.IO.Atlf.Text](#)

Assembly: Cobilas.Core.dll

```
public sealed class NullATLFEencoding : ATLFEencoding
```

## Inheritance

[object](#) ← [ATLFEencoding](#) ← NullATLFEencoding

## Inherited Members

[ATLFEencoding.Null](#) , [object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#)

# Properties

## Version

Represents the encoder version.

```
public override string Version { get; }
```

## Property Value

[string](#)

# Methods

## Writer(params object[])

Writes a list of objects.

```
public override string Writer(params object[] args)
```

Parameters

args [object](#)[]

Returns

[string](#)[]

## Writer4Byte(params object[])

Writes a list of objects.

```
public override byte[] Writer4Byte(params object[] args)
```

Parameters

args [object](#)[]

Returns

[byte](#)[]

# Namespace Cobilas.IO.Serialization.Binary

## Classes

### [ScratchObject](#)

Base sketch object.

# Class ScratchObject

Namespace: [Cobilas.IO.Serialization.Binary](#)

Assembly: Cobilas.Core.dll

Base sketch object.

```
[Serializable]  
public abstract class ScratchObject
```

Inheritance

[object](#) ← ScratchObject

Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#).

## Properties

### Name

Draft name.

```
public abstract string Name { get; }
```

Property Value

[string](#)

# Methods

## LoadScratchObject(string)

Loads an object [ScratchObject](#) from a file.

```
public static ScratchObject LoadScratchObject(string filePath)
```

Parameters

`filePath` [string](#)

File path.

Returns

[ScratchObject](#)

## UnloadScratchObject(ScratchObject, string)

Downloads draft object to a file.

```
public static void UnloadScratchObject(ScratchObject scratch, string filePath)
```

Parameters

`scratch` [ScratchObject](#)

Scratch object.

`filePath` [string](#)

Path of the file where the object will be downloaded.

## UnloadScratchObject(ScratchObject, string, string)

Downloads draft object to a file.

```
public static void UnloadScratchObject(ScratchObject scratch, string folderPath, string extension = "sobj")
```

## Parameters

**scratch** [ScratchObject](#)

Scratch object.

**folderPath** [string](#)

Path of the directory where the file will be created.

**extension** [string](#)

File extension.

## UnloadScratchObject(ScratchObject, string, string, string)

Downloads draft object to a file.

```
public static void UnloadScratchObject(ScratchObject scratch, string folderPath, string name, string extension = "sobj")
```

## Parameters

**scratch** [ScratchObject](#)

Scratch object.

**folderPath** [string](#)

Path of the directory where the file will be created.

**name** [string](#)

File name.

**extension** [string](#)

File extension.

# Namespace Cobilas.IO.Serialization.Json

## Classes

### [Json](#)

#### [JsonContractResolver](#)

Used by Newtonsoft.Json.JsonSerializer to resolve a Newtonsoft.Json.Serialization.JsonContract for a given [Type](#).

# Class Json

Namespace: [Cobilas.IO.Serialization.Json](#)

Assembly: Cobilas.Core.dll

```
public static class Json
```

## Inheritance

[object](#) ← Json

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Methods

## Deserialize(string)

Deserializes the JSON to a .NET object using Newtonsoft.Json.JsonSerializerSettings.

```
public static object Deserialize(string value)
```

### Parameters

**value** [string](#)

The JSON to deserialize.

### Returns

[object](#)

## Deserialize(string, JsonSerializerSettings)

Deserializes the JSON to a .NET object using Newtonsoft.Json.JsonSerializerSettings.

```
public static object Deserialize(string value, JsonSerializerSettings settings)
```

## Parameters

**value** [string](#)

The JSON to deserialize.

**settings** [JsonSerializerSettings](#)

The Newtonsoft.Json.JsonSerializerSettings used to deserialize the object. If this is null, default serialization settings will be used.

## Returns

[object](#)

## Deserialize<T>(string)

Deserializes the JSON to a .NET object using Newtonsoft.Json.JsonSerializerSettings.

```
public static T Deserialize<T>(string value)
```

## Parameters

**value** [string](#)

The JSON to deserialize.

## Returns

T

## Type Parameters

T

## Deserialize<T>(string, JsonSerializerSettings)

Deserializes the JSON to a .NET object using Newtonsoft.Json.JsonSerializerSettings.

```
public static T Deserialize<T>(string value, JsonSerializerSettings settings)
```

## Parameters

**value** [string](#)

The JSON to deserialize.

**settings** [JsonSerializerSettings](#)

The Newtonsoft.Json.JsonSerializerSettings used to deserialize the object. If this is null, default serialization settings will be used.

## Returns

T

## Type Parameters

T

## Serialize(object)

Serializes the specified object to a JSON string using Newtonsoft.Json.JsonSerializerSettings.

```
public static string Serialize(object value)
```

## Parameters

**value** [object](#)

The object to serialize.

## Returns

[string](#)

## Serialize(object, JsonSerializerSettings)

Serializes the specified object to a JSON string using Newtonsoft.Json.JsonSerializerSettings.

```
public static string Serialize(object value, JsonSerializerSettings settings)
```

### Parameters

**value** [object](#)

The object to serialize.

**settings** JsonSerializerSettings

The Newtonsoft.Json.JsonSerializerSettings used to serialize the object. If this is null, default serialization settings will be used.

### Returns

[string](#)

## Serialize(object, bool)

Serializes the specified object to a JSON string using Newtonsoft.Json.JsonSerializerSettings.

```
public static string Serialize(object value, bool Indented)
```

### Parameters

**value** [object](#)

The object to serialize.

**Indented** [bool](#)

This parameter allows you to format the json file.

### Returns

[string](#)

# Class JsonContractResolver

Namespace: [Cobilas.IO.Serialization.Json](#)

Assembly: Cobilas.Core.dll

Used by Newtonsoft.Json.JsonSerializer to resolve a Newtonsoft.Json.Serialization.JsonContract for a given [Type](#).

```
public class JsonContractResolver : DefaultContractResolver, IContractResolver
```

## Inheritance

[object](#) ← DefaultContractResolver ← JsonContractResolver

## Implements

IContractResolver

## Inherited Members

[DefaultContractResolver.ResolveContract\(Type\)](#) ,  
[DefaultContractResolver.GetSerializableMembers\(Type\)](#) ,  
[DefaultContractResolver.CreateObjectContract\(Type\)](#) ,  
[DefaultContractResolver.CreateConstructorParameters\(ConstructorInfo, JsonPropertyCollection\)](#) ,  
[DefaultContractResolver.CreatePropertyFromConstructorParameter\(JsonProperty, ParameterInfo\)](#) ,  
[DefaultContractResolver.ResolveContractConverter\(Type\)](#) ,  
[DefaultContractResolver.CreateDictionaryContract\(Type\)](#) ,  
[DefaultContractResolver.CreateArrayContract\(Type\)](#) ,  
[DefaultContractResolver.CreatePrimitiveContract\(Type\)](#) ,  
[DefaultContractResolver.CreateLinqContract\(Type\)](#) ,  
[DefaultContractResolver.CreateSerializableContract\(Type\)](#) ,  
[DefaultContractResolver.CreateDynamicContract\(Type\)](#) ,  
[DefaultContractResolver.CreateStringContract\(Type\)](#) , [DefaultContractResolver.CreateContract\(Type\)](#) ,  
[DefaultContractResolver.CreateMemberValueProvider\(MemberInfo\)](#) ,  
[DefaultContractResolver.CreateProperty\(MemberInfo, MemberSerialization\)](#) ,  
[DefaultContractResolver.ResolvePropertyName\(string\)](#) ,  
[DefaultContractResolver.ResolveExtensionDataName\(string\)](#) ,  
[DefaultContractResolver.ResolveDictionaryKey\(string\)](#) ,  
[DefaultContractResolver.GetResolvedPropertyName\(string\)](#) ,  
DefaultContractResolver.DynamicCodeGeneration ,  
DefaultContractResolver.DefaultMembersSearchFlags ,  
DefaultContractResolver.SerializeCompilerGeneratedMembers ,

DefaultContractResolver.IgnoreSerializableInterface ,  
DefaultContractResolver.IgnoreSerializableAttribute ,  
DefaultContractResolver.IgnoreIsSpecifiedMembers ,  
DefaultContractResolver.IgnoreShouldSerializeMembers , DefaultContractResolver.NamingStrategy ,  
[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#)

# Methods

## CreateProperties(Type, MemberSerialization)

Creates properties for the given Newtonsoft.Json.Serialization.JsonContract.

```
protected override IList<JsonProperty> CreateProperties(Type type,  
MemberSerialization memberSerialization)
```

### Parameters

**type** [Type](#)

The type to create properties for.

**memberSerialization** [MemberSerialization](#)

The member serialization mode for the type.

### Returns

[IList](#)<JsonProperty>

Properties for the given Newtonsoft.Json.Serialization.JsonContract.

# Namespace Cobilas.Numeric

## Classes

### [BasicCalculation](#)

Basic functions of mathematical operations.

### [CalculationsCollection](#)

Base class used to store calculation and calculation overwriting functions.

### [ParseCalculation](#)

## Structs

### [MathOperator](#)

It represents the mathematical signal and performs its mathematical operation.

## Enums

### [SignalOrientation](#)

Indicates the direction where the operation numbers will be obtained.

# Class BasicCalculation

Namespace: [Cobilas.Numeric](#)

Assembly: Cobilas.Core.dll

Basic functions of mathematical operations.

```
public class BasicCalculation : CalculationsCollection
```

## Inheritance

[object](#) ← [CalculationsCollection](#) ← BasicCalculation

## Inherited Members

[CalculationsCollection.Merge\(CalculationsCollection, CalculationsCollection\)](#) , [object.ToString\(\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#).

# Properties

## Calculations

The list of calculation functions.

```
public override MathOperator[] Calculations { get; protected set; }
```

## Property Value

[MathOperator\[\]](#)

# OverwriteCalculations

The list of calculation functions that will replace other calculation functions.

```
public override MathOperator[] OverwriteCalculations { get; protected set; }
```

Property Value

[MathOperator\[\]](#)

## Methods

### Clac(double, string, double)

Function responsible for calling a calculation function specified the sign of the mathematical operation.

```
public override double Clac(double V1, string S, double V2)
```

Parameters

V1 [double](#)

S [string](#)

mathematical operator.

V2 [double](#)

Returns

[double](#)

### Initialization()

Function to initialize the calculation functions.

```
public override void Initialization()
```

# Class CalculationsCollection

Namespace: [Cobilas.Numeric](#)

Assembly: Cobilas.Core.dll

Base class used to store calculation and calculation overwriting functions.

```
public abstract class CalculationsCollection
```

## Inheritance

[object](#) ← CalculationsCollection

## Derived

[BasicCalculation](#)

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#).

# Properties

## Calculations

The list of calculation functions.

```
public abstract MathOperator[] Calculations { get; protected set; }
```

## Property Value

[MathOperator\[\]](#)

## OverwriteCalculations

The list of calculation functions that will replace other calculation functions.

```
public abstract MathOperator[] OverwriteCalculations { get; protected set; }
```

Property Value

[MathOperator\[\]](#)

## Methods

### Clac(double, string, double)

Function responsible for calling a calculation function specified the sign of the mathematical operation.

```
public abstract double Clac(double v1, string s, double v2)
```

Parameters

v1 [double](#)

s [string](#)

mathematical operator.

v2 [double](#)

Returns

[double](#)

### Initialization()

Function to initialize the calculation functions.

```
public abstract void Initialization()
```

## Merge(CalculationsCollection, CalculationsCollection)

Merge two collections of calculations.

```
public static CalculationsCollection Merge(CalculationsCollection A,  
CalculationsCollection B)
```

Parameters

A [CalculationsCollection](#)

B [CalculationsCollection](#)

Returns

[CalculationsCollection](#)

# Struct MathOperator

Namespace: [Cobilas.Numeric](#)

Assembly: Cobilas.Core.dll

It represents the mathematical signal and performs its mathematical operation.

```
public readonly struct MathOperator : IEquatable<string>, IEquatable<byte>,
IEquatable<SignalOrientation>
```

## Implements

[IEquatable<string>](#), [IEquatable<byte>](#), [IEquatable<SignalOrientation>](#)

## Inherited Members

[ValueType.ToString\(\)](#) , [object.Equals\(object, object\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.GetType\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#)

## Constructors

MathOperator(char, Func<double, double, double>, SignalOrientation)

```
public MathOperator(char signal, Func<double, double, double> function,
SignalOrientation orientation)
```

## Parameters

signal [char](#)

The sign of the mathematical operation.

**function** [Func<double, double, double>](#)

The function that performs the mathematical operation.

**orientation** [SignalOrientation](#)

Indicates the direction where the operation numbers will be obtained.

## MathOperator(char, Func<double, double, double>, SignalOrientation, byte)

```
public MathOperator(char signal, Func<double, double, double> function, SignalOrientation orientation, byte executionLevel)
```

### Parameters

**signal** [char](#)

The sign of the mathematical operation.

**function** [Func<double, double, double>](#)

The function that performs the mathematical operation.

**orientation** [SignalOrientation](#)

Indicates the direction where the operation numbers will be obtained.

**executionLevel** [byte](#)

The order of execution of the mathematical operation.

## MathOperator(string, Func<double, double, double>, SignalOrientation)

```
public MathOperator(string signal, Func<double, double, double> function, SignalOrientation orientation)
```

## Parameters

**signal** [string](#)

The sign of the mathematical operation.

**function** [Func](#)<[double](#), [double](#), [double](#)>

The function that performs the mathematical operation.

**orientation** [SignalOrientation](#)

Indicates the direction where the operation numbers will be obtained.

**MathOperator(string, Func<double, double, double>, SignalOrientation, byte)**

```
public MathOperator(string signal, Func<double, double, double> function, SignalOrientation orientation, byte executionLevel)
```

## Parameters

**signal** [string](#)

The sign of the mathematical operation.

**function** [Func](#)<[double](#), [double](#), [double](#)>

The function that performs the mathematical operation.

**orientation** [SignalOrientation](#)

Indicates the direction where the operation numbers will be obtained.

**executionLevel** [byte](#)

The order of execution of the mathematical operation.

## Properties

**ExecutionLevel**

The order of execution of the mathematical operation.

```
public byte ExecutionLevel { get; }
```

Property Value

[byte ↗](#)

## Function

The function that performs the mathematical operation.

```
public Delegate Function { get; }
```

Property Value

[Delegate ↗](#)

## Orientation

Indicates the direction where the operation numbers will be obtained.

```
public SignalOrientation Orientation { get; }
```

Property Value

[SignalOrientation](#)

## Signal

The sign of the mathematical operation.

```
public string Signal { get; }
```

Property Value

[string](#)

## Methods

### Equals(SignalOrientation)

Performs the comparison between [MathOperator](#) and [SignalOrientation](#).

```
public bool Equals(SignalOrientation other)
```

Parameters

other [SignalOrientation](#)

Returns

[bool](#)

### Equals(byte)

Performs the comparison between [MathOperator](#) and [byte](#).

```
public bool Equals(byte other)
```

Parameters

other [byte](#)

Returns

[bool](#)

### Equals(object)

Performs the comparison between [MathOperator](#) and [object](#).

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

Parameters

obj [object](#)

Returns

[bool](#)

## Equals(string)

Performs the comparison between [MathOperator](#) and [string](#).

```
public bool Equals(string other)
```

Parameters

other [string](#)

Returns

[bool](#)

## GetHashCode()

Return the hash code for this instance.

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

Returns

[int](#)

# Operators

## operator ==(MathOperator, SignalOrientation)

```
public static bool operator ==(MathOperator A, SignalOrientation B)
```

Parameters

A [MathOperator](#)

B [SignalOrientation](#)

Returns

[bool](#)

## operator ==(MathOperator, byte)

```
public static bool operator ==(MathOperator A, byte B)
```

Parameters

A [MathOperator](#)

B [byte](#)

Returns

[bool](#)

## operator ==(MathOperator, string)

```
public static bool operator ==(MathOperator A, string B)
```

Parameters

A [MathOperator](#)

B [string](#)

Returns

[bool](#)

## operator ==(SignalOrientation, MathOperator)

```
public static bool operator ==(SignalOrientation A, MathOperator B)
```

Parameters

A [SignalOrientation](#)

B [MathOperator](#)

Returns

[bool](#)

## operator ==(byte, MathOperator)

```
public static bool operator ==(byte A, MathOperator B)
```

Parameters

A [byte](#)

B [MathOperator](#)

Returns

[bool](#)

## operator ==(string, MathOperator)

```
public static bool operator ==(string A, MathOperator B)
```

Parameters

A [string](#)

B [MathOperator](#)

Returns

[bool](#)

operator !=(MathOperator, SignalOrientation)

```
public static bool operator !=(MathOperator A, SignalOrientation B)
```

Parameters

A [MathOperator](#)

B [SignalOrientation](#)

Returns

[bool](#)

operator !=(MathOperator, byte)

```
public static bool operator !=(MathOperator A, byte B)
```

Parameters

A [MathOperator](#)

B [byte](#)

Returns

[bool](#)

## operator !=(MathOperator, string)

```
public static bool operator !=(MathOperator A, string B)
```

Parameters

A [MathOperator](#)

B [string](#)

Returns

[bool](#)

## operator !=(SignalOrientation, MathOperator)

```
public static bool operator !=(SignalOrientation A, MathOperator B)
```

Parameters

A [SignalOrientation](#)

B [MathOperator](#)

Returns

[bool](#)

## operator !=(byte, MathOperator)

```
public static bool operator !=(byte A, MathOperator B)
```

Parameters

A [byte](#)

B [MathOperator](#)

Returns

[bool](#)

## operator !=(string, MathOperator)

```
public static bool operator !=(string A, MathOperator B)
```

Parameters

A [string](#)

B [MathOperator](#)

Returns

[bool](#)

# Class ParseCalculation

Namespace: [Cobilas.Numeric](#)

Assembly: Cobilas.Core.dll

```
public static class ParseCalculation
```

## Inheritance

[object](#) ← ParseCalculation

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Properties

## Collections

All collections of mathematical operations created.

```
public static CalculationsCollection[] Collections { get; }
```

## Property Value

[CalculationsCollection\[\]](#)

# Methods

## DebugLogCalc(string)

Take a text containing a mathematical formula and perform the calculation.

Prints the mathematical formula and its result.

```
public static double DebugLogCalc(string text)
```

Parameters

**text** [string](#)

Returns

[double](#)

## Parse(string)

Take a text containing a mathematical formula and perform the calculation.

```
public static double Parse(string text)
```

Parameters

**text** [string](#)

Returns

[double](#)

## PrintConsoleCalc(string)

Take a text containing a mathematical formula and perform the calculation.

Prints the mathematical formula and its result.

```
public static double PrintConsoleCalc(string text)
```

Parameters

**text** [string](#)

Returns

[double](#)

# Enum SignalOrientation

Namespace: [Cobilas.Numeric](#)

Assembly: Cobilas.Core.dll

Indicates the direction where the operation numbers will be obtained.

```
public enum SignalOrientation : byte
```

## Extension Methods

[Enum CB Extension.Format\(Enum, object, string\)](#) , [Enum CB Extension.GetEnumPair\(Enum\)](#) ,  
[Enum CB Extension.GetEnumPairs\(Enum\)](#) , [Enum CB Extension.GetName\(Enum\)](#) ,  
[Enum CB Extension.GetName\(Enum, object\)](#) , [Enum CB Extension.GetNames\(Enum\)](#) ,  
[Enum CB Extension.HasFlag\(Enum, params Enum\[\]\)](#) , [Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#).

## Fields

**both = 1**

2<+>2=4 Need two numbers on the sign side.

**left = 2**

2<sqr=4 You need the numbers on the left side of the sign.

**none = 0**

None

**right = 3**

sqr>2=4 You need the numbers on the right side of the sign.

# Namespace Cobilas.Numeric.Convert

## Classes

[BitArray\\_Binary\\_Extension](#)

[String\\_Hexadecimal\\_Extension](#)

# Class BitArray\_Binary\_Extension

Namespace: [Cobilas.Numeric.Convert](#)

Assembly: Cobilas.Core.dll

```
public static class BitArray_Binary_Extension
```

## Inheritance

[object](#) ← BitArray\_Binary\_Extension

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Methods

## BitToDecimal(BitArray)

Convert [BitArray](#) to [decimal](#).

```
public static decimal BitToDecimal(this BitArray array)
```

## Parameters

array [BitArray](#)

## Returns

[decimal](#)

## GetBinaryArray(BitArray)

Transforms a [BitArray](#) into a [byte](#) list.

```
public static byte[] GetBinaryArray(this BitArray array)
```

Parameters

array [BitArray](#)

Returns

[byte](#)[]

## SetBinaryArray(BitArray, byte[])

Defines a list of bytes in the [BitArray](#) object.

```
public static void SetBinaryArray(this BitArray array, byte[] binaryArray)
```

Parameters

array [BitArray](#)

binaryArray [byte](#)[]

# Class String\_Hexadecimal\_Extension

Namespace: [Cobilas.Numeric.Convert](#)

Assembly: Cobilas.Core.dll

```
public static class String_Hexadecimal_Extension
```

## Inheritance

[object](#) ← String\_Hexadecimal\_Extension

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Methods

## HexToDecimal(string)

Converts hexadecimal value to decimal.

```
public static decimal HexToDecimal(this string str)
```

### Parameters

**str** [string](#)

### Returns

[decimal](#)

## IsHexadecimal(string)

Checks whether the text value is a hexadecimal value.

```
public static bool IsHexadecimal(this string str)
```

## Parameters

**str** [string](#) ↗

## Returns

[bool](#) ↗

# Namespace Cobilas.Threading.Tasks

## Classes

[TaskPool](#)

# Class TaskPool

Namespace: [Cobilas.Threading.Tasks](#)

Assembly: Cobilas.Core.dll

```
public static class TaskPool
```

## Inheritance

[object](#) ← TaskPool

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Properties

## Count

Returns the number of allocated tasks.

```
public static int Count { get; }
```

## Property Value

[int](#)

## NonVacantTaskCount

The number of tasks that are not vacant.

```
public static int NonVacantTaskCount { get; }
```

## Property Value

[int](#)

## VacantTaskCount

The number of tasks that are already vacant.

```
public static int VacantTaskCount { get; }
```

Property Value

[int](#)

## Methods

### InitTask(Action)

The number of tasks that are not vacant.

```
public static void InitTask(Action action)
```

Parameters

[action](#) [Action](#)

### InitTask(Action, out Task)

The number of tasks that are not vacant.

```
public static void InitTask(Action action, out Task task)
```

Parameters

[action](#) [Action](#)

[task](#) [Task](#)

## InitTask<TRes>(Func<TRes>)

The number of tasks that are not vacant.

```
public static void InitTask<TRes>(Func<TRes> func)
```

Parameters

func [Func](#)<TRes>

Type Parameters

TRes

## InitTask<TRes>(Func<TRes>, out Task<TRes>)

The number of tasks that are not vacant.

```
public static void InitTask<TRes>(Func<TRes> func, out Task<TRes> res)
```

Parameters

func [Func](#)<TRes>

res [Task](#)<TRes>

Type Parameters

TRes

# Namespace System

## Classes

[Char\\_CB\\_Extension](#)

[Enum\\_CB\\_Extension](#)

[Object\\_CB\\_Extension](#)

[String\\_CB\\_Extension](#)

[Type\\_CB\\_Extension](#)

## Enums

[EscapeSequence](#)

Represents an escape sequence.

# Class Char\_CB\_Extension

Namespace: [System](#)

Assembly: Cobilas.Core.dll

```
public static class Char_CB_Extension
```

## Inheritance

[object](#) ← Char\_CB\_Extension

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### EscapeSequenceToString(char)

Method that interprets a char value.

```
public static string EscapeSequenceToString(this char c)
```

#### Parameters

c [char](#)

#### Returns

[string](#)

The method returns a value of type [EscapeSequence](#) representing the type of escape sequence.

The return is converted to a [string](#).

### InterpretEscapeSequence(char)

Method that interprets a char value.

```
public static EscapeSequence InterpretEscapeSequence(this char c)
```

Parameters

c [char](#)

Returns

[EscapeSequence](#)

The method returns a value of type [EscapeSequence](#) representing the type of escape sequence.

ToSByte(char)

```
public static sbyte ToSByte(this char c)
```

Parameters

c [char](#)

Returns

[sbyte](#)

# Class Enum\_CB\_Extension

Namespace: [System](#)

Assembly: Cobilas.Core.dll

```
public static class Enum_CB_Extension
```

## Inheritance

[object](#) ← Enum\_CB\_Extension

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### Format(Enum, object, string)

Converts the specified value of a specified enumerated type to its equivalent [string](#) representation according to the specified format.

```
public static string Format(this Enum E, object value, string format)
```

#### Parameters

E [Enum](#)

The target enumerator.

value [object](#)

The value to convert.

format [string](#)

The output format to use.

#### Returns

## [string](#)

A string representation of value.

## Exceptions

### [ArgumentNullException](#)

The enumType, value, or format parameter is null.

### [ArgumentException](#)

The enumType parameter is not an [Enum](#) type. -or- The value is from an enumeration that differs in type from enumType. -or- The type of value is not an underlying type of enumType.

### [FormatException](#)

The format parameter contains an invalid value.

### [InvalidOperationException](#)

format equals "X", but the enumeration type is unknown.

## GetEnumPair(Enum)

Gets the tag with the current assigned value.

```
public static KeyValuePair<string, int> GetEnumPair(this Enum E)
```

## Parameters

### E [Enum](#)

The target enumerator.

## Returns

### [KeyValuePair](#) <string, int>

## GetEnumPairs(Enum)

Gets a list of all tags along with their assigned values.

```
public static KeyValuePair<string, int>[] GetEnumPairs(this Enum E)
```

## Parameters

E [Enum](#)

The target enumerator.

## Returns

[KeyValuePair](#)<string, int>[]

Returns a list of all tags along with their assigned values or an empty list.

## Exceptions

[ArgumentNullException](#)

enumType is null.

## GetName(Enum)

Retrieves the name of the constant in the specified enumeration that has the specified value.

```
public static string GetName(this Enum E)
```

## Parameters

E [Enum](#)

The target enumerator.

## Returns

[string](#)

A string containing the name of the enumerated constant in enumType whose value is value; or null if no such constant is found.

## Exceptions

### [ArgumentNullException](#)

enumType is null.

### [ArgumentException](#)

enumType is not an [Enum](#). -or- value is neither of type enumType nor does it have the same underlying type as enumType.

## GetName(Enum, object)

Retrieves the name of the constant in the specified enumeration that has the specified value.

```
public static string GetName(this Enum E, object value)
```

### Parameters

#### E [Enum](#)

The target enumerator.

#### value [object](#)

The value of a particular enumerated constant in terms of its underlying type.

### Returns

#### [string](#)

A [string](#) containing the name of the enumerated constant in enumType whose value is value; or null if no such constant is found.

## Exceptions

### [ArgumentNullException](#)

enumType is null.

### [ArgumentException](#)

enumType is not an [Enum](#). -or- value is neither of type enumType nor does it have the same underlying type as enumType.

## GetNames(Enum)

Retrieves an array of the names of the constants in a specified enumeration.

```
public static string[] GetNames(this Enum E)
```

### Parameters

E [Enum](#)

The target enumerator.

### Returns

[string](#)[]

A string array of the names of the constants in enumType.

### Exceptions

[ArgumentNullException](#)

enumType is null.

[ArgumentException](#)

enumType parameter is not an [Enum](#).

## HasFlag(Enum, params Enum[])

Determines whether one or more bit fields are set in the current instance.

```
public static bool HasFlag(this Enum E, params Enum[] flags)
```

### Parameters

E [Enum](#)

The target enumerator.

flags [Enum](#)[]

Comparison tags.

Returns

[bool](#)

# Enum EscapeSequence

Namespace: [System](#)

Assembly: Cobilas.Core.dll

Represents an escape sequence.

```
public enum EscapeSequence
```

## Extension Methods

[Enum CB Extension.Format\(Enum, object, string\)](#), [Enum CB Extension.GetEnumPair\(Enum\)](#),  
[Enum CB Extension.GetEnumPairs\(Enum\)](#), [Enum CB Extension.GetName\(Enum\)](#),  
[Enum CB Extension.GetName\(Enum, object\)](#), [Enum CB Extension.GetNames\(Enum\)](#),  
[Enum CB Extension.HasFlag\(Enum, params Enum\[\]\)](#), [Object CB Extension.CompareType\(object, Type\)](#),  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#),  
[Object CB Extension.CompareType<T>\(object\)](#).

## Fields

Alert = 5

Represents the escape sequence.(\a)

BackSlash = 3

Represents the escape sequence.(\b)

Backspace = 6

Represents the escape sequence.(\b)

CarriageReturn = 9

Represents the escape sequence.(\r)

DoubleQuote = 2

Represents the escape sequence.(")

**FormFeed** = 7

Represents the escape sequence.(\r)

**HorizontalTab** = 10

Represents the escape sequence.(\t)

**NewLine** = 8

Represents the escape sequence.(\n)

**None** = 0

Represents no escape sequence.

**Null** = 4

Represents the escape sequence.(\0)

**SingleQuote** = 1

Represents the escape sequence.(')

**VerticalTab** = 11

Represents the escape sequence.(\v)

# Class Object\_CB\_Extension

Namespace: [System](#)

Assembly: Cobilas.Core.dll

```
public static class Object_CB_Extension
```

## Inheritance

[object](#) ← Object\_CB\_Extension

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Methods

## CompareType(object, Type)

Compares the object's type to a specified type.

```
public static bool CompareType(this object o, Type type)
```

### Parameters

**o** [object](#)

The target object of comparison.

**type** [Type](#)

The specified comparison type.

### Returns

[bool](#)

## CompareType(object, params Type[])

Compares the object's type to a specified list of types.

```
public static bool CompareType(this object o, params Type[] types)
```

Parameters

**o** [object](#)

The target object of comparison.

**types** [Type](#)[]

The specified comparison types.

Returns

[bool](#)

## CompareTypeAndSubType(object, Type)

Compares class type and the class it inherits.

```
public static bool CompareTypeAndSubType(this object o, Type type)
```

Parameters

**o** [object](#)

**type** [Type](#)

Returns

[bool](#)

## CompareTypeAndSubType(object, Type, bool)

Compares class type and the class it inherits.

```
public static bool CompareTypeAndSubType(this object o, Type type, bool IncludeInterface)
```

Parameters

o [object](#)

type [Type](#)

IncludeInterface [bool](#)

Returns

[bool](#)

## CompareTypeAndSubType<T>(object)

Compares class type and the class it inherits.

```
public static bool CompareTypeAndSubType<T>(this object o)
```

Parameters

o [object](#)

Returns

[bool](#)

Type Parameters

T

## CompareTypeAndSubType<T>(object, bool)

Compares class type and the class it inherits.

```
public static bool CompareTypeAndSubType<T>(this object o, bool IncludeInterface)
```

Parameters

0 [object](#)

IncludeInterface [bool](#)

Returns

[bool](#)

Type Parameters

T

## CompareType<T>(object)

Compares the object's type with a specified generic type.

```
public static bool CompareType<T>(this object o)
```

Parameters

0 [object](#)

The target object of comparison.

Returns

[bool](#)

Type Parameters

T

The generic comparison type specified.

# Class String\_CB\_Extension

Namespace: [System](#)

Assembly: Cobilas.Core.dll

```
public static class String_CB_Extension
```

## Inheritance

[object](#) ← String\_CB\_Extension

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### Contains(string, params char[])

Returns a value indicating whether a specified substring occurs within this string.

```
public static bool Contains(this string s, params char[] value)
```

#### Parameters

s [string](#)

value [char\[\]](#)

#### Returns

[bool](#)

### ToByte(string)

```
public static byte ToByte(this string s)
```

Parameters

S [string](#)

Returns

[byte](#)

## ToByte(string, NumberStyles, IFormatProvider)

```
public static byte ToByte(this string S, NumberStyles style, IFormatProvider formatProvider)
```

Parameters

S [string](#)

style [NumberStyles](#)

formatProvider [IFormatProvider](#)

Returns

[byte](#)

## ToByte(string, IFormatProvider)

```
public static byte ToByte(this string S, IFormatProvider formatProvider)
```

Parameters

S [string](#)

formatProvider [IFormatProvider](#)

Returns

[byte](#)

## ToDecimal(string)

```
public static decimal ToDecimal(this string S)
```

### Parameters

S [string](#)

### Returns

[decimal](#)

## ToDecimal(string, NumberStyles, IFormatProvider)

```
public static decimal ToDecimal(this string S, NumberStyles style,  
IFormatProvider formatProvider)
```

### Parameters

S [string](#)

style [NumberStyles](#)

formatProvider [IFormatProvider](#)

### Returns

[decimal](#)

## ToDecimal(string, IFormatProvider)

```
public static decimal ToDecimal(this string S, IFormatProvider formatProvider)
```

### Parameters

S [string](#)

`formatProvider` [IFormatProvider](#)

Returns

[decimal](#)

## ToDouble(string)

`public static double ToDouble(this string s)`

Parameters

`s` [string](#)

Returns

[double](#)

## ToDouble(string, NumberStyles, IFormatProvider)

`public static double ToDouble(this string s, NumberStyles style,  
IFormatProvider formatProvider)`

Parameters

`s` [string](#)

`style` [NumberStyles](#)

`formatProvider` [IFormatProvider](#)

Returns

[double](#)

## ToDouble(string, IFormatProvider)

```
public static double ToDouble(this string S, IFormatProvider formatProvider)
```

Parameters

S [string](#)

formatProvider [IFormatProvider](#)

Returns

[double](#)

## ToFloat(string)

```
public static float ToFloat(this string S)
```

Parameters

S [string](#)

Returns

[float](#)

## ToFloat(string, NumberStyles, IFormatProvider)

```
public static float ToFloat(this string S, NumberStyles style,  
IFormatProvider formatProvider)
```

Parameters

S [string](#)

style [NumberStyles](#)

formatProvider [IFormatProvider](#)

Returns

[float](#)

## ToFloat(string, IFormatProvider)

```
public static float ToFloat(this string S, IFormatProvider formatProvider)
```

Parameters

S [string](#)

formatProvider [IFormatProvider](#)

Returns

[float](#)

## ToInt(string)

```
public static intToInt(this string S)
```

Parameters

S [string](#)

Returns

[int](#)

## ToInt(string, NumberStyles, IFormatProvider)

```
public static intToInt(this string S, NumberStyles style, IFormatProvider formatProvider)
```

Parameters

`S` [string](#)

`style` [NumberStyles](#)

`formatProvider` [IFormatProvider](#)

Returns

[int](#)

## ToInt(string, IFormatProvider)

```
public static intToInt(this string S, IFormatProvider formatProvider)
```

Parameters

`S` [string](#)

`formatProvider` [IFormatProvider](#)

Returns

[int](#)

## ToLong(string)

```
public static long ToLong(this string S)
```

Parameters

`S` [string](#)

Returns

[long](#)

## ToLong(string, NumberStyles, IFormatProvider)

```
public static long ToLong(this string S, NumberStyles style, IFormatProvider formatProvider)
```

Parameters

S [string](#)

style [NumberStyles](#)

formatProvider [IFormatProvider](#)

Returns

[long](#)

## ToLong(string, IFormatProvider)

```
public static long ToLong(this string S, IFormatProvider formatProvider)
```

Parameters

S [string](#)

formatProvider [IFormatProvider](#)

Returns

[long](#)

## ToSByte(string)

```
public static sbyte ToSByte(this string S)
```

Parameters

S [string](#)

Returns

## [sbyte](#)

### ToSByte(string, NumberStyles, IFormatProvider)

```
public static sbyte ToSByte(this string S, NumberStyles style,  
IFormatProvider formatProvider)
```

#### Parameters

S [string](#)

style [NumberStyles](#)

formatProvider [IFormatProvider](#)

#### Returns

[sbyte](#)

### ToSByte(string, IFormatProvider)

```
public static sbyte ToSByte(this string S, IFormatProvider formatProvider)
```

#### Parameters

S [string](#)

formatProvider [IFormatProvider](#)

#### Returns

[sbyte](#)

### ToShort(string)

```
public static short ToShort(this string S)
```

Parameters

S [string](#)

Returns

[short](#)

## ToShort(string, NumberStyles, IFormatProvider)

```
public static short ToShort(this string S, NumberStyles style,  
IFormatProvider formatProvider)
```

Parameters

S [string](#)

style [NumberStyles](#)

formatProvider [IFormatProvider](#)

Returns

[short](#)

## ToShort(string, IFormatProvider)

```
public static short ToShort(this string S, IFormatProvider formatProvider)
```

Parameters

S [string](#)

formatProvider [IFormatProvider](#)

Returns

[short](#)

## ToUInt(string)

```
public static uint ToUInt(this string s)
```

Parameters

**S** [string](#)

Returns

[uint](#)

## ToUInt(string, NumberStyles, IFormatProvider)

```
public static uint ToUInt(this string s, NumberStyles style, IFormatProvider formatProvider)
```

Parameters

**S** [string](#)

**style** [NumberStyles](#)

**formatProvider** [IFormatProvider](#)

Returns

[uint](#)

## ToUInt(string, IFormatProvider)

```
public static uint ToUInt(this string s, IFormatProvider formatProvider)
```

Parameters

**S** [string](#)

**formatProvider** [IFormatProvider](#)

Returns

[uint](#)

## ToULong(string)

```
public static ulong ToULong(this string s)
```

Parameters

[S](#) [string](#)

Returns

[ulong](#)

## ToULong(string, NumberStyles, IFormatProvider)

```
public static ulong ToULong(this string s, NumberStyles style,  
IFormatProvider formatProvider)
```

Parameters

[S](#) [string](#)

[style](#) [NumberStyles](#)

[formatProvider](#) [IFormatProvider](#)

Returns

[ulong](#)

## ToULong(string, IFormatProvider)

```
public static ulong ToULong(this string s, IFormatProvider formatProvider)
```

Parameters

[S string](#)

[formatProvider IFormatProvider](#)

Returns

[ulong](#)

## ToUShort(string)

```
public static ushort ToUShort(this string S)
```

Parameters

[S string](#)

Returns

[ushort](#)

## ToUShort(string, NumberStyles, IFormatProvider)

```
public static ushort ToUShort(this string S, NumberStyles style,  
IFormatProvider formatProvider)
```

Parameters

[S string](#)

[style NumberStyles](#)

[formatProvider IFormatProvider](#)

Returns

[ushort](#)

## ToUShort(string, IFormatProvider)

```
public static ushort ToUShort(this string s, IFormatProvider formatProvider)
```

### Parameters

s [string](#)

formatProvider [IFormatProvider](#)

### Returns

[ushort](#)

# Class Type\_CB\_Extension

Namespace: [System](#)

Assembly: Cobilas.Core.dll

```
public static class Type_CB_Extension
```

## Inheritance

[object](#) ← Type\_CB\_Extension

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Methods

## Activator(Type)

Creates an instance of an object from a type.

```
public static object Activator(this Type type)
```

### Parameters

type [Type](#)

### Returns

[object](#)

## Activator<T>(Type)

Creates an instance of an object from a type.

```
public static T Activator<T>(this Type type)
```

Parameters

type [Type](#)

Returns

T

Type Parameters

T

## GetAttribute<T>(Type)

Gets the type attribute.

```
public static T GetAttribute<T>(this Type type) where T : Attribute
```

Parameters

type [Type](#)

Returns

T

Type Parameters

T

The generic type used to obtain the target attribute.

## GetAttribute<T>(Type, bool)

Gets the type attribute.

```
public static T GetAttribute<T>(this Type type, bool inherit) where T : Attribute
```

Parameters

**type** [Type](#)

**inherit** [bool](#)

true to get attributes marked as inherited.

Returns

[T](#)

Type Parameters

[T](#)

The generic type used to obtain the target attribute.

## GetAttributes<T>(Type)

Gets all attributes of the type.

```
public static T[] GetAttributes<T>(this Type type) where T : Attribute
```

Parameters

**type** [Type](#)

Returns

[T\[\]](#)

Type Parameters

[T](#)

The generic type used to obtain the target attribute.

## GetAttributes<T>(Type, bool)

Gets all attributes of the type.

```
public static T[] GetAttributes<T>(this Type type, bool inherit) where T : Attribute
```

Parameters

**type** [Type](#)

**inherit** [bool](#)

true to get attributes marked as inherited.

Returns

T[]

Type Parameters

T

The generic type used to obtain the target attribute.

# Namespace System.IO

## Classes

[Stream\\_CB\\_Extension](#)

# Class Stream\_CB\_Extension

Namespace: [System.IO](#)

Assembly: Cobilas.Core.dll

```
public static class Stream_CB_Extension
```

## Inheritance

[object](#) ← Stream\_CB\_Extension

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Methods

## GenerateGuid(Stream)

Generates a [Guid](#) object from the current flow.

```
public static Guid GenerateGuid(this Stream F)
```

### Parameters

F [Stream](#)

Target stream.

### Returns

[Guid](#)

The method returns a Guid value by reading a copy of the current stream.

## GetChars(Stream)

Gets an array of characters from the current stream.

[Encoding](#).UTF8 is used by default in the method.

```
public static char[] GetChars(this Stream F)
```

Parameters

F [Stream](#)

Target stream.

Returns

[char](#)[]

## GetChars(Stream, Encoding)

Gets an array of characters from the current stream.

```
public static char[] GetChars(this Stream F, Encoding encoding)
```

Parameters

F [Stream](#)

Target stream.

encoding [Encoding](#)

The Encoding used to write to the current stream.

Returns

[char](#)[]

## GetString(Stream)

Gets a string from the current stream.

[Encoding](#).UTF8 is used by default in the method.

```
public static string GetString(this Stream F)
```

## Parameters

F [Stream](#)

Target stream.

## Returns

[string](#)

## GetString(Stream, Encoding)

Gets a string from the current stream.

```
public static string GetString(this Stream F, Encoding encoding)
```

## Parameters

F [Stream](#)

Target stream.

encoding [Encoding](#)

The Encoding used to write to the current stream.

## Returns

[string](#)

## Read(Stream)

When overridden in a derived class, reads a sequence of bytes from the current stream and advances the position within the stream by the number of bytes read.

```
public static byte[] Read(this Stream F)
```

## Parameters

F [Stream](#)

Target stream.

## Returns

[byte](#)[]

## Exceptions

[ArgumentException](#)

[ArgumentNullException](#)

[ArgumentOutOfRangeException](#)

[IOException](#)

[NotSupportedException](#)

[ObjectDisposedException](#)

## Write(Stream, byte[])

When overridden in a derived class, writes a sequence of bytes to the current stream and advances the current position within this stream by the number of bytes written.

```
public static void Write(this Stream F, byte[] bytes)
```

## Parameters

F [Stream](#)

Target stream.

bytes [byte](#)[]

An array of bytes. This method copies count bytes from buffer to the current stream.

## Exceptions

[ArgumentException](#)

[ArgumentNullException](#)

[ArgumentOutOfRangeException](#)

[IOException](#)

[NotSupportedException](#)

[ObjectDisposedException](#)

## Write(Stream, char[])

When overridden in a derived class, writes a sequence of bytes to the current stream and advances the current position within this stream by the number of bytes written.

[Encoding](#).UTF8 is used by default in the method.

```
public static void Write(this Stream F, char[] chars)
```

## Parameters

F [Stream](#)

Target stream.

chars [char](#)[]

The list of characters that will be written to the current stream.

## Exceptions

[ArgumentException](#)

[ArgumentNullException](#)

[ArgumentOutOfRangeException](#)

[IOException](#)

[NotSupportedException](#)

[ObjectDisposedException](#)

## Write(Stream, char[], Encoding)

When overridden in a derived class, writes a sequence of bytes to the current stream and advances the current position within this stream by the number of bytes written.

```
public static void Write(this Stream F, char[] chars, Encoding encoding)
```

### Parameters

F [Stream](#)

Target stream.

chars [char](#)[]

The list of characters that will be written to the current stream.

encoding [Encoding](#)

The Encoding used to write to the current stream.

### Exceptions

[ArgumentException](#)

[ArgumentNullException](#)

[ArgumentOutOfRangeException](#)

[IOException](#)

[NotSupportedException](#)

[ObjectDisposedException](#)

## Write(Stream, string)

When overridden in a derived class, writes a sequence of bytes to the current stream and advances the current position within this stream by the number of bytes written.

[Encoding](#).UTF8 is used by default in the method.

```
public static void Write(this Stream F, string text)
```

### Parameters

F [Stream](#)

Target stream.

text [string](#)

The text that will be written in the current stream.

## Write(Stream, string, Encoding)

When overridden in a derived class, writes a sequence of bytes to the current stream and advances the current position within this stream by the number of bytes written.

```
public static void Write(this Stream F, string text, Encoding encoding)
```

### Parameters

F [Stream](#)

Target stream.

text [string](#)

The text that will be written in the current stream.

encoding [Encoding](#)

The Encoding used to write to the current stream.

### Exceptions

[ArgumentException](#)

[ArgumentNullException](#)

[IOException](#)

[NotSupportedException](#)

[ObjectDisposedException](#)

# Namespace System.Reflection

## Classes

[MethodInfo\\_CB\\_Extension](#)

# Class FieldInfo\_CB\_Extension

Namespace: [System](#).[Reflection](#)

Assembly: Cobilas.Core.dll

```
public static class FieldInfo_CB_Extension
```

## Inheritance

[object](#) ← FieldInfo\_CB\_Extension

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

# Methods

## IsBackingField(FieldInfo)

Determines whether the field is a background field used by automatic properties.

```
public static bool IsBackingField(this FieldInfo F)
```

### Parameters

F [FieldInfo](#)

Target field.

### Returns

[bool](#)

Returns [true](#) when it is a background field and [false](#) when it is not a background field.

# Namespace System.Security.Cryptography

## Classes

[HashAlgorithm\\_CB\\_Extension](#)

## Structs

[HashString](#)

The HashString structure can be used to transform a list of bytes with a minimum length of 16 into a [string](#) or [Guid](#).

# Class HashAlgorithm\_CB\_Extension

Namespace: [System](#) [.Security](#) [.Cryptography](#)

Assembly: Cobilas.Core.dll

```
public static class HashAlgorithm_CB_Extension
```

## Inheritance

[object](#) ← HashAlgorithm\_CB\_Extension

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### ComputeHash(HashAlgorithm, string)

Computes the hash value for the specified byte array.

```
public static byte[] ComputeHash(this HashAlgorithm H, string FilePath)
```

#### Parameters

H [HashAlgorithm](#)

Target object.

FilePath [string](#)

The full path of the file.

#### Returns

[byte](#)[]

The computed hash code.

## Exceptions

[ArgumentNullException](#)

[ObjectDisposedException](#)

[ArgumentException](#)

[PathTooLongException](#)

[DirectoryNotFoundException](#)

[IOException](#)

[UnauthorizedAccessException](#)

[FileNotFoundException](#)

[NotSupportedException](#)

[SecurityException](#)

## ComputeHashToString(HashAlgorithm, byte[])

Computes the hash value for the specified byte array.

```
public static HashString ComputeHashToString(this HashAlgorithm H, byte[] buffer)
```

### Parameters

H [HashAlgorithm](#)

Target object.

buffer [byte](#)[]

The input to compute the hash code for.

### Returns

[HashString](#)

The computed hash code for string.

## Exceptions

[ArgumentNullException](#)

[ObjectDisposedException](#)

## ComputeHashToString(HashAlgorithm, byte[], int, int)

Computes the hash value for the specified region of the specified byte array.

```
public static HashString ComputeHashToString(this HashAlgorithm H, byte[] buffer, int offset, int count)
```

### Parameters

**H** [HashAlgorithm](#)

Target object.

**buffer** [byte](#)[]

The input to compute the hash code for.

**offset** [int](#)

The offset into the byte array from which to begin using data.

**count** [int](#)

The number of bytes in the array to use as data.

### Returns

[HashString](#)

The computed hash code for string.

## Exceptions

[ArgumentException](#)

[ArgumentNullException](#)

[ArgumentOutOfRangeException](#)

[ObjectDisposedException](#)

## ComputeHashToString(HashAlgorithm, Stream)

Computes the hash value for the specified [Stream](#) object.

```
public static HashString ComputeHashToString(this HashAlgorithm H, Stream inputStream)
```

### Parameters

H [HashAlgorithm](#)

Target object.

inputStream [Stream](#)

The input to compute the hash code for.

### Returns

[HashString](#)

The computed hash code for string.

### Exceptions

[ObjectDisposedException](#)

## ComputeHashToString(HashAlgorithm, string)

Computes the hash value for the specified byte array.

```
public static HashString ComputeHashToString(this HashAlgorithm H, string FilePath)
```

### Parameters

H [HashAlgorithm](#)

Target object.

### [FilePath](#) `string` ↗

The full path of the file.

Returns

### [HashString](#)

The computed hash code for string.

Exceptions

### [ArgumentNullException](#) ↗

### [ObjectDisposedException](#) ↗

### [ArgumentException](#) ↗

### [PathTooLongException](#) ↗

### [DirectoryNotFoundException](#) ↗

### [IOException](#) ↗

### [UnauthorizedAccessException](#) ↗

### [FileNotFoundException](#) ↗

### [NotSupportedException](#) ↗

### [SecurityException](#) ↗

# Struct HashString

Namespace: [System](#).[Security](#).[Cryptography](#)

Assembly: Cobilas.Core.dll

The HashString structure can be used to transform a list of bytes with a minimum length of 16 into a [string](#) or [Guid](#).

```
public readonly struct HashString : IEquatable<HashString>, IComparable<HashString>,  
IEquatable<string>, IFormattable
```

## Implements

[IEquatable](#)<[HashString](#)>, [IComparable](#)<[HashString](#)>, [IEquatable](#)<[string](#)>, [IFormattable](#)

## Inherited Members

[object.Equals\(object, object\)](#), [object.ReferenceEquals\(object, object\)](#), [object.GetType\(\)](#)

## Extension Methods

```
Object CB Extension.CompareType\(object, Type\),  
Object CB Extension.CompareType\(object, params Type\[\]\),  
Object CB Extension.CompareTypeAndSubType\(object, Type\),  
Object CB Extension.CompareTypeAndSubType\(object, Type, bool\),  
Object CB Extension.CompareTypeAndSubType<T>\(object\),  
Object CB Extension.CompareTypeAndSubType<T>\(object, bool\),  
Object CB Extension.CompareType<T>\(object\)
```

# Constructors

## HashString(byte[])

Initializes a [HashString](#) object using a list of bytes with a minimum length of 16.

```
public HashString(byte[] hash)
```

## Parameters

hash [byte](#)[]

## Exceptions

[ArgumentNullException](#)

[ArgumentException](#)

## Fields

### Empty

A read-only instance of type [HashString](#) whose value is all zero.

```
public static readonly HashString Empty
```

### Field Value

[HashString](#)

## Methods

### CompareTo(HashString)

Compares this instance to a specified [HashString](#) object and returns an indication of their relative values.

```
public int CompareTo(HashString other)
```

### Parameters

**other** [HashString](#)

An object to compare to this instance.

### Returns

[int](#)

A signed number indicating the relative values of this instance and value.  
Return value Description  
A negative integer This instance is less than value.  
Zero This instance is equal to value.  
A positive integer This instance is greater than value.

## Equals(object)

Indicates whether this instance and a specified object are equal.

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

Parameters

**obj** [object](#)

The [object](#) to compare with the current instance.

Returns

[bool](#)

true if obj and this instance are the same type and represent the same value; otherwise, false.

## Equals(HashString)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(HashString other)
```

Parameters

**other** [HashString](#)

An object to compare with this object.

Returns

[bool](#)

true if the current object is equal to the other parameter; otherwise, false.

## Equals(string)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(string other)
```

## Parameters

**other** [string](#)

An object to compare with this object.

Use [Guid](#).ToString() or convert a list of bytes to a [string](#) using a [StringBuilder](#) or similar.

## Returns

[bool](#)

true if the current object is equal to the other parameter; otherwise, false.

## GetHashCode()

Returns the hash code for this instance.

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

## Returns

[int](#)

A 32-bit signed integer that is the hash code for this instance.

## ToString()

Returns a string representation of the value of this instance in registry format.

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

## Returns

[string](#)

The value of this [Guid](#), formatted by using the "D" format specifier as follows: `xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx` where the value of the GUID is represented as a series of lowercase hexadecimal digits in groups of 8, 4, 4, 4, and 12 digits and separated by hyphens. An example of a return value is "382c74c3-721d-4f34-80e5-57657b6cbc27". To convert the hexadecimal digits from a through f to uppercase, call the `System.String.ToUpper` method on the returned string.

## ToString(string)

Returns a string representation of the value of this [Guid](#) instance, according to the provided format specifier.

```
public string ToString(string format)
```

### Parameters

**format** [string](#)

A single format specifier that indicates how to format the value of this [Guid](#). The format parameter can be "N", "D", "B", "P", or "X". If format is null or an empty string (""), "D" is used.

### Returns

[string](#)

The value of this [Guid](#), represented as a series of lowercase hexadecimal digits in the specified format.

### Exceptions

[FormatException](#)

The value of format is not null, an empty string (""), "N", "D", "B", "P", or "X".

## ToString(string, IFormatProvider)

Returns a string representation of the value of this instance of the [Guid](#) class, according to the provided format specifier and culture-specific format information.

```
public string ToString(string format, IFormatProvider formatProvider)
```

## Parameters

### format [string](#)

A single format specifier that indicates how to format the value of this [Guid](#). The format parameter can be "N", "D", "B", "P", or "X". If format is null or an empty string (""), "D" is used.

### formatProvider [IFormatProvider](#)

(Reserved) An object that supplies culture-specific formatting information.

## Returns

### [string](#)

The value of this [Guid](#), represented as a series of lowercase hexadecimal digits in the specified format.

## Operators

### explicit operator Guid(HashString)

Explicit conversion from [HashString](#) object to [Guid](#).

```
public static explicit operator Guid(HashString hash)
```

## Parameters

### hash [HashString](#)

## Returns

### [Guid](#)

### implicit operator string(HashString)

Implicit conversion from [HashString](#) to [string](#) type.

```
public static implicit operator string(HashString hash)
```

Parameters

**hash** [HashString](#)

Returns

[string](#) ↗

# Namespace System.Xml

## Classes

### [CB XML Extension](#)

Extension that adds reading and writing functions for XML.

### [XMLIRW](#)

Base class for IRW class.

### [XMLIRWAttribute](#)

Represents an XML element of type Attribute.

### [XMLIRWCData](#)

Represents an XML element of type CDATA.

### [XMLIRWComment](#)

Represents an XML element of type Comment.

### [XMLIRWDeclaration](#)

Represents an xml declaration.

### [XMLIRWDocType](#)

Represents an XML element of type DocType.

### [XMLIRWElem](#)

XML improved reader and writer element.

### [XMLIRWProcessingInstruction](#)

Represents an XML element of type ProcessingInstruction.

### [XMLIRWText](#)

Represents XML text.

## Structs

### [XMLIRWValue](#)

Represents the value of a tag.

## Interfaces

### [ITextValue](#)

Interface for text XMLIRW elements.

### [IXMLIRWCollection](#)

Represents a collection of XMLIRW.



# Class CB\_XML\_Extension

Namespace: [System.Xml](#)

Assembly: Cobilas.Core.dll

Extension that adds reading and writing functions for XML.

```
public static class CB_XML_Extension
```

## Inheritance

[object](#) ← CB\_XML\_Extension

## Inherited Members

[object.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Methods

### ReadXMLIRW(XmlReader)

Reads an XML document and returns an [XMLIRWElement](#).

```
public static XMLIRWElement ReadXMLIRW(this XmlReader reader)
```

#### Parameters

reader [XmlReader](#)

#### Returns

[XMLIRWElement](#)

### WriterXMLIRW(XmlWriter, XMLIRWElement)

Uses an [XMLIRWElement](#) to write to the xml document.

```
public static void WriterXMLIRW(this XmlWriter writer, XMLIRWElem ent element)
```

## Parameters

writer [XmlWriter](#)

element [XMLIRWElem ent](#)

# Interface ITextValue

Namespace: [System.Xml](#)

Assembly: Cobilas.Core.dll

Interface for text XMLIRW elements.

```
public interface ITextValue
```

## Extension Methods

[Object\\_CB\\_Extension.CompareType\(object, Type\)](#) ,  
[Object\\_CB\\_Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object\\_CB\\_Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object\\_CB\\_Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object\\_CB\\_Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object\\_CB\\_Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object\\_CB\\_Extension.CompareType<T>\(object\)](#)

# Properties

## Text

Returns or sets the text of the XMLIRW element.

```
XMLIRWText Text { get; set; }
```

## Property Value

[XMLIRWText](#)

# Interface IXMLIRWCollection

Namespace: [System.Xml](#)

Assembly: Cobilas.Core.dll

Represents a collection of XMLIRW.

```
public interface IXMLIRWCollection : IEnumerable<XMLIRW>, IEnumerable, IDisposable
```

## Inherited Members

[IEnumerable<XMLIRW>.GetEnumerator\(\)](#) , [IDisposable.Dispose\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#)

## Properties

### AttributeCount

Gets the count of attributes on the element.

```
int AttributeCount { get; }
```

### Property Value

[int](#)

### Attributes

Gets the attributes on the element.

```
IEnumerable<XMLIRW> Attributes { get; }
```

## Property Value

[IEnumerable](#) <XMLIRW>

## IsEmpty

Checks whether the element has sub-elements or attributes.

```
bool IsEmpty { get; }
```

## Property Value

[bool](#)

## NoAttributes

Checks whether the element has attributes.

```
bool NoAttributes { get; }
```

## Property Value

[bool](#)

## NoElements

Checks whether the element has sub-elements.

```
bool NoElements { get; }
```

## Property Value

[bool](#)

## ValueIsEmpty

Checks whether the element has a text value.

```
bool ValueIsEmpty { get; }
```

Property Value

[bool](#)

## Methods

### Add(XMLIRW)

Adds a new XMLIRW element.

```
bool Add(XMLIRW element)
```

Parameters

[element](#) [XMLIRW](#)

Returns

[bool](#)

Returns [true](#) when the element is added XMLIRW.

# Class XMLIRW

Namespace: [System.Xml](#)

Assembly: Cobilas.Core.dll

Base class for IRW class.

```
public abstract class XMLIRW : IDisposable
```

Inheritance

[object](#) ← XMLIRW

Implements

[IDisposable](#)

Derived

[XMLIRWAttribute](#), [XMLIRWCData](#), [XMLIRWComment](#), [XMLIRWDeclaration](#), [XMLIRWDocType](#),  
[XMLIRWElement](#), [XMLIRWProcessingInstruction](#), [XMLIRWText](#)

Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.GetHashCode\(\)](#), [object.GetType\(\)](#), [object.MemberwiseClone\(\)](#)

Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#),  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#),  
[Object CB Extension.CompareType<T>\(object\)](#).

## Constructors

### XMLIRW()

Creates a new instance of the XMLIRW element.

```
protected XMLIRW()
```

## XMLIRW(string)

Creates a new instance of the XMLIRW element.

```
protected XMLIRW(string name)
```

Parameters

name [string](#)

## XMLIRW(string, XmlNodeType)

Creates a new instance of the XMLIRW element.

```
protected XMLIRW(string name, XmlNodeType type)
```

Parameters

name [string](#)

type [XmlNodeType](#)

## XMLIRW(XMLIRW, string)

Creates a new instance of the XMLIRW element.

```
protected XMLIRW(XMLIRW parent, string name)
```

Parameters

parent [XMLIRW](#)

name [string](#)

## XMLIRW(XMLIRW, string, XmlNodeType)

Creates a new instance of the XMLIRW element.

```
protected XMLIRW(XMLIRW parent, string name, XmlNodeType type)
```

## Parameters

parent [XMLIRW](#)

name [string](#)

type [XmlNodeType](#)

# Properties

## Name

Returns or sets the name of the XMLIRW object.

```
public abstract string Name { get; set; }
```

## Property Value

[string](#)

## Parent

Returns or sets the name of the parent object of the XMLIRW element.

```
public abstract XMLIRW Parent { get; set; }
```

## Property Value

[XMLIRW](#)

## Type

Returns or sets the type of the XMLIRW element.

```
public abstract XmlNodeType Type { get; set; }
```

## Property Value

[XmlNodeType](#)

## Methods

### Dispose()

Discard the resources of the XMLIRW element.

```
public abstract void Dispose()
```

### ToString()

Creates a [string](#) representation of the [object](#).

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

### Returns

[string](#)

# Class XMLIRWAttribute

Namespace: [System.Xml](#)

Assembly: Cobilas.Core.dll

Represents an XML element of type Attribute.

```
public class XMLIRWAttribute : XMLIRW, IDisposable, ITextValue
```

## Inheritance

[object](#) ← [XMLIRW](#) ← XMLIRWAttribute

## Implements

[IDisposable](#), [ITextValue](#)

## Inherited Members

[XMLIRW.ToString\(\)](#), [object.Equals\(object\)](#), [object.Equals\(object, object\)](#),  
[object.ReferenceEquals\(object, object\)](#), [object.GetHashCode\(\)](#), [object.GetType\(\)](#),  
[object.MemberwiseClone\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#),  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#),  
[Object CB Extension.CompareType<T>\(object\)](#).

## Constructors

### XMLIRWAttribute(string, object)

Creates a new instance of the XMLIRW element.

```
public XMLIRWAttribute(string name, object value)
```

## Parameters

`name` [string](#)

`value` [object](#)

## XMLIRWAttribute(string, XMLIRWValue)

Creates a new instance of the XMLIRW element.

```
[Obsolete("Use the XMLIRWAttribute(string, object) constructor.")]
public XMLIRWAttribute(string name, XMLIRWValue value)
```

### Parameters

`name` [string](#)

`value` [XMLIRWValue](#)

## XMLIRWAttribute(XMLIRWElement, string, object)

Creates a new instance of the XMLIRW element.

```
public XMLIRWAttribute(XMLIRWElement parent, string name, object value)
```

### Parameters

`parent` [XMLIRWElement](#)

`name` [string](#)

`value` [object](#)

## XMLIRWAttribute(XMLIRWElement, string, XMLIRWValue)

Creates a new instance of the XMLIRW element.

```
[Obsolete("Use the XMLIRWAttribute(XMLIRWElement, string, object) constructor.")]
public XMLIRWAttribute(XMLIRWElement parent, string name, XMLIRWValue value)
```

## Parameters

parent [XMLIRWElement](#)

name [string](#)

value [XMLIRWValue](#)

## Properties

### Name

Returns or sets the name of the XMLIRW object.

```
public override string Name { get; set; }
```

### Property Value

[string](#)

### Parent

Returns or sets the name of the parent object of the XMLIRW element.

```
public override XMLIRW Parent { get; set; }
```

### Property Value

[XMLIRW](#)

### Text

Returns or sets the text of the XMLIRW element.

```
public XMLIRWText Text { get; set; }
```

### Property Value

## [XMLIRWText](#)

### Type

Returns or sets the type of the XMLIRW element.

```
public override XmlNodeType Type { get; set; }
```

### Property Value

[XmlNodeType](#)

### Value

Returns or sets the text of the XMLIRW element.

```
[Obsolete("Use the Text property.")]  
public XMLIRWValue Value { get; set; }
```

### Property Value

[XMLIRWValue](#)

## Methods

### Dispose()

Discard the resources of the XMLIRW element.

```
public override void Dispose()
```

### Dispose(bool)

Discard the resources of the XMLIRW element.

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

Parameters

**disposing** [bool](#)

**~XMLIRWAttribute()**

Called when the object is finished.

```
protected ~XMLIRWAttribute()
```

# Class XMLIRWCDATA

Namespace: [System.Xml](#)

Assembly: Cobilas.Core.dll

Represents an XML element of type CDATA.

```
public class XMLIRWCDATA : XMLIRW, IDisposable, ITextValue
```

## Inheritance

[object](#) ← [XMLIRW](#) ← XMLIRWCDATA

## Implements

[IDisposable](#), [ITextValue](#)

## Inherited Members

[XMLIRW.ToString\(\)](#), [object.Equals\(object\)](#), [object.Equals\(object, object\)](#),  
[object.ReferenceEquals\(object, object\)](#), [object.GetHashCode\(\)](#), [object.GetType\(\)](#),  
[object.MemberwiseClone\(\)](#)

## Extension Methods

[Object\\_CB\\_Extension.CompareType\(object, Type\)](#),  
[Object\\_CB\\_Extension.CompareType\(object, params Type\[\]\)](#),  
[Object\\_CB\\_Extension.CompareTypeAndSubType\(object, Type\)](#),  
[Object\\_CB\\_Extension.CompareTypeAndSubType\(object, Type, bool\)](#),  
[Object\\_CB\\_Extension.CompareTypeAndSubType<T>\(object\)](#),  
[Object\\_CB\\_Extension.CompareTypeAndSubType<T>\(object, bool\)](#),  
[Object\\_CB\\_Extension.CompareType<T>\(object\)](#).

## Constructors

### XMLIRWCDATA(object)

Creates a new instance of the XMLIRW element.

```
public XMLIRWCDATA(object value)
```

## Parameters

`value` [object](#)

## XMLIRWCDATA(string, object)

Creates a new instance of the XMLIRW element.

```
[Obsolete("Use the XMLIRWCDATA(object) constructor.")]
public XMLIRWCDATA(string name, object value)
```

### Parameters

`name` [string](#)

`value` [object](#)

## XMLIRWCDATA(string, XMLIRWValue)

Creates a new instance of the XMLIRW element.

```
[Obsolete("Use the XMLIRWCDATA(object) constructor.")]
public XMLIRWCDATA(string name, XMLIRWValue value)
```

### Parameters

`name` [string](#)

`value` [XMLIRWValue](#)

## XMLIRWCDATA(XMLIRW, object)

Creates a new instance of the XMLIRW element.

```
public XMLIRWCDATA(XMLIRW parent, object value)
```

### Parameters

`parent` [XMLIRW](#)

`value object`

## XMLIRWCDATA(XMLIRW, string, object)

Creates a new instance of the XMLIRW element.

```
[Obsolete("Use the XMLIRWCDATA(XMLIRW, object) constructor.")]
public XMLIRWCDATA(XMLIRW parent, string name, object value)
```

### Parameters

`parent XMLIRW`

`name string`

`value object`

## XMLIRWCDATA(XMLIRW, string, XMLIRWValue)

Creates a new instance of the XMLIRW element.

```
[Obsolete("Use the XMLIRWCDATA(XMLIRW, object) constructor.")]
public XMLIRWCDATA(XMLIRW parent, string name, XMLIRWValue value)
```

### Parameters

`parent XMLIRW`

`name string`

`value XMLIRWValue`

## XMLIRWCDATA(XMLIRW, XMLIRWValue)

Creates a new instance of the XMLIRW element.

```
[Obsolete("Use the XMLIRWCDATA(XMLIRW, object) constructor.")]
```

```
public XMLIRWCDATA(XMLIRW parent, XMLIRWValue value)
```

## Parameters

parent [XMLIRW](#)

value [XMLIRWValue](#)

## XMLIRWCDATA(XMLIRWValue)

Creates a new instance of the XMLIRW element.

```
[Obsolete("Use the XMLIRWCDATA(object) constructor.")]
public XMLIRWCDATA(XMLIRWValue value)
```

## Parameters

value [XMLIRWValue](#)

# Properties

## Name

Returns or sets the name of the XMLIRW object.

```
public override string Name { get; set; }
```

## Property Value

[string](#)

## Parent

Returns or sets the name of the parent object of the XMLIRW element.

```
public override XMLIRW Parent { get; set; }
```

## Property Value

### [XMLIRW](#)

## Text

Returns or sets the text of the XMLIRW element.

```
public XMLIRWText Text { get; set; }
```

## Property Value

### [XMLIRWText](#)

## Type

Returns or sets the type of the XMLIRW element.

```
public override XmlNodeType Type { get; set; }
```

## Property Value

### [XmlNodeType](#)

## Value

Returns or sets the text of the XMLIRW element.

```
[Obsolete("Use the Text property.")]  
public XMLIRWValue Value { get; }
```

## Property Value

### [XMLIRWValue](#)

# Methods

## Dispose()

Discard the resources of the XMLIRW element.

```
public override void Dispose()
```

## Dispose(bool)

Discard the resources of the XMLIRW element.

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

Parameters

`disposing` [bool](#)

## ~XMLIRWCDATA()

Called when the object is finished.

```
protected ~XMLIRWCDATA()
```

# Class XMLIRWComment

Namespace: [System.Xml](#)

Assembly: Cobilas.Core.dll

Represents an XML element of type Comment.

```
public class XMLIRWComment : XMLIRW, ITextValue, IDisposable
```

## Inheritance

[object](#) ← [XMLIRW](#) ← XMLIRWComment

## Implements

[ITextValue](#), [IDisposable](#)

## Inherited Members

[XMLIRW.ToString\(\)](#), [object.Equals\(object\)](#), [object.Equals\(object, object\)](#),  
[object.ReferenceEquals\(object, object\)](#), [object.GetHashCode\(\)](#), [object.GetType\(\)](#),  
[object.MemberwiseClone\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#),  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#),  
[Object CB Extension.CompareType<T>\(object\)](#).

## Constructors

### XMLIRWComment(object)

Creates a new instance of the XMLIRW element.

```
public XMLIRWComment(object value)
```

## Parameters

[value](#) [object](#)

## XMLIRWComment(XMLIRW, object)

Creates a new instance of the XMLIRW element.

```
public XMLIRWComment(XMLIRW parent, object value)
```

### Parameters

[parent](#) [XMLIRW](#)

[value](#) [object](#)

## XMLIRWComment(XMLIRW, XMLIRWValue)

Creates a new instance of the XMLIRW element.

```
[Obsolete("Use the XMLIRWComment(XMLIRW, object) constructor.")]  
public XMLIRWComment(XMLIRW parent, XMLIRWValue value)
```

### Parameters

[parent](#) [XMLIRW](#)

[value](#) [XMLIRWValue](#)

## XMLIRWComment(XMLIRWValue)

Creates a new instance of the XMLIRW element.

```
[Obsolete("Use the XMLIRWComment(object) constructor.")]  
public XMLIRWComment(XMLIRWValue value)
```

### Parameters

[value](#) [XMLIRWValue](#)

# Properties

## Name

Returns or sets the name of the XMLIRW object.

```
public override string Name { get; set; }
```

## Property Value

[string](#)

## Parent

Returns or sets the name of the parent object of the XMLIRW element.

```
public override XMLIRW Parent { get; set; }
```

## Property Value

[XMLIRW](#)

## Text

Returns or sets the text of the XMLIRW element.

```
public XMLIRWText Text { get; set; }
```

## Property Value

[XMLIRWText](#)

## Type

Returns or sets the type of the XMLIRW element.

```
public override XmlNodeType Type { get; set; }
```

Property Value

[XmlNodeType](#)

## Value

Returns or sets the text of the XMLIRW element.

```
[Obsolete("Use the Text property.")]  
public XMLIRWValue Value { get; }
```

Property Value

[XMLIRWValue](#)

## Methods

### Dispose()

Discard the resources of the XMLIRW element.

```
public override void Dispose()
```

### Dispose(bool)

Discard the resources of the XMLIRW element.

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

Parameters

**disposing** [bool](#)

## `~XMLIRWComment()`

Called when the object is finished.

```
protected ~XMLIRWComment()
```

# Class XMLIRWDeclaration

Namespace: [System.Xml](#)

Assembly: Cobilas.Core.dll

Represents an xml declaration.

```
public class XMLIRWDeclaration : XMLIRW, IDisposable
```

## Inheritance

[object](#) ← [XMLIRW](#) ← XMLIRWDeclaration

## Implements

[IDisposable](#)

## Inherited Members

[XMLIRW.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#).

## Constructors

### XMLIRWDeclaration()

Creates a new instance of the XMLIRW element.

```
public XMLIRWDeclaration()
```

## XMLIRWDeclaration(string)

Creates a new instance of the XMLIRW element.

```
public XMLIRWDeclaration(string version)
```

Parameters

version [string](#)

## XMLIRWDeclaration(string, string)

Creates a new instance of the XMLIRW element.

```
public XMLIRWDeclaration(string version, string encoding)
```

Parameters

version [string](#)

encoding [string](#)

## XMLIRWDeclaration(string, string, string)

Creates a new instance of the XMLIRW element.

```
public XMLIRWDeclaration(string version, string encoding, string standalone)
```

Parameters

version [string](#)

encoding [string](#)

standalone [string](#)

## XMLIRWDeclaration(XMLIRW)

Creates a new instance of the XMLIRW element.

```
public XMLIRWDeclaration(XMLIRW parent)
```

Parameters

parent [XMLIRW](#)

## XMLIRWDeclaration(XMLIRW, string)

Creates a new instance of the XMLIRW element.

```
public XMLIRWDeclaration(XMLIRW parent, string version)
```

Parameters

parent [XMLIRW](#)

version [string](#)

## XMLIRWDeclaration(XMLIRW, string, string)

Creates a new instance of the XMLIRW element.

```
public XMLIRWDeclaration(XMLIRW parent, string version, string encoding)
```

Parameters

parent [XMLIRW](#)

version [string](#)

encoding [string](#)

## XMLIRWDeclaration(XMLIRW, string, string, string)

Creates a new instance of the XMLIRW element.

```
public XMLIRWDeclaration(XMLIRW parent, string version, string encoding, string standalone)
```

### Parameters

parent [XMLIRW](#)

version [string](#)

encoding [string](#)

standalone [string](#)

## Properties

### Encoding

Gets or sets the encoding level of the XML document.

```
public string Encoding { get; protected set; }
```

### Property Value

[string](#)

### Name

Returns or sets the name of the XMLIRW object.

```
public override string Name { get; set; }
```

### Property Value

[string](#)

## Parent

Returns or sets the name of the parent object of the XMLIRW element.

```
public override XMLIRW Parent { get; set; }
```

## Property Value

[XMLIRW](#)

## Standalone

Gets or sets the value of the standalone attribute.

```
public string Standalone { get; protected set; }
```

## Property Value

[string](#)

## Type

Returns or sets the type of the XMLIRW element.

```
public override XmlNodeType Type { get; set; }
```

## Property Value

[XmlNodeType](#)

## Version

Gets the XML version of the document.

```
public string Version { get; protected set; }
```

Property Value

[string](#) ↗

## Methods

### Dispose()

Discard the resources of the XMLIRW element.

```
public override void Dispose()
```

# Class XMLIRWDocType

Namespace: [System.Xml](#)

Assembly: Cobilas.Core.dll

Represents an XML element of type DocType.

```
public class XMLIRWDocType : XMLIRW, IDisposable
```

## Inheritance

[object](#) ← [XMLIRW](#) ← XMLIRWDocType

## Implements

[IDisposable](#)

## Inherited Members

[XMLIRW.ToString\(\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#) ,  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#) ,  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#) ,  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#) ,  
[Object CB Extension.CompareType<T>\(object\)](#).

## Constructors

### XMLIRWDocType(string, object, object, object)

Creates a new instance of the XMLIRW element.

```
public XMLIRWDocType(string name, object puid, object sysid, object subset)
```

## Parameters

`name` [string](#)

`puid` [object](#)

`sysid` [object](#)

`subset` [object](#)

## XMLIRWDocType(string, XMLIRWValue, XMLIRWValue, XMLIRWValue)

Creates a new instance of the XMLIRW element.

```
[Obsolete("Use the XMLIRWDocType(string, object, object, object) constructor.")]
public XMLIRWDocType(string name, XMLIRWValue puid, XMLIRWValue sysid, XMLIRWValue subset)
```

### Parameters

`name` [string](#)

`puid` [XMLIRWValue](#)

`sysid` [XMLIRWValue](#)

`subset` [XMLIRWValue](#)

## XMLIRWDocType(XMLIRW, string, object, object, object)

Creates a new instance of the XMLIRW element.

```
public XMLIRWDocType(XMLIRW parent, string name, object puid, object sysid, object subset)
```

### Parameters

`parent` [XMLIRW](#)

`name` [string](#)

`puid` [object](#)

[sysid](#) [object](#)

[subset](#) [object](#)

## XMLIRWDocType(XMLIRW, string, XMLIRWValue, XMLIRWValue, XMLIRWValue)

Creates a new instance of the XMLIRW element.

```
[Obsolete("Use the XMLIRWDocType(XMLIRW, string, object, object, object) constructor.")]
public XMLIRWDocType(XMLIRW parent, string name, XMLIRWValue puid, XMLIRWValue sysid,
XMLIRWValue subset)
```

### Parameters

[parent](#) [XMLIRW](#)

[name](#) [string](#)

[puid](#) [XMLIRWValue](#)

[sysid](#) [XMLIRWValue](#)

[subset](#) [XMLIRWValue](#)

## Properties

### Name

Returns or sets the name of the XMLIRW object.

```
public override string Name { get; set; }
```

### Property Value

[string](#)

## Parent

Returns or sets the name of the parent object of the XMLIRW element.

```
public override XMLIRW Parent { get; set; }
```

### Property Value

[XMLIRW](#)

## PudID

Gets the value of the public identifier on the DOCTYPE declaration.

```
public XMLIRWText PudID { get; }
```

### Property Value

[XMLIRWText](#)

## SubSet

Gets the value of the document type definition (DTD) internal subset on the DOCTYPE declaration.

```
public XMLIRWText SubSet { get; }
```

### Property Value

[XMLIRWText](#)

## SysID

Gets the value of the system identifier on the DOCTYPE declaration.

```
public XMLIRWText SysID { get; }
```

## Property Value

[XMLIRWText](#)

## Type

Returns or sets the type of the XMLIRW element.

```
public override XmlNodeType Type { get; set; }
```

## Property Value

[XmlNodeType](#)

## Methods

### Dispose()

Discard the resources of the XMLIRW element.

```
public override void Dispose()
```

### Dispose(bool)

Discard the resources of the XMLIRW element.

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

## Parameters

`disposing` [bool](#)

### ~XMLIRWDocType()

Called when the object is finished.

```
protected ~XMLIRWDocType()
```

# Class XMLIRWElement

Namespace: [System.Xml](#)

Assembly: Cobilas.Core.dll

XML improved reader and writer element.

```
public class XMLIRWElement : XMLIRW, ITextValue, IXMLIRWCollection, IEnumerable<XMLIRW>,  
IEnumerable, IDisposable
```

## Inheritance

[object](#) ← [XMLIRW](#) ← XMLIRWElement

## Implements

[ITextValue](#), [IXMLIRWCollection](#), [IEnumerable](#)<[XMLIRW](#)>, [IEnumerable](#), [IDisposable](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.GetHashCode\(\)](#), [object.GetType\(\)](#), [object.MemberwiseClone\(\)](#)

## Extension Methods

[Object\\_CB\\_Extension.CompareType\(object, Type\)](#),  
[Object\\_CB\\_Extension.CompareType\(object, params Type\[\]\)](#),  
[Object\\_CB\\_Extension.CompareTypeAndSubType\(object, Type\)](#),  
[Object\\_CB\\_Extension.CompareTypeAndSubType\(object, Type, bool\)](#),  
[Object\\_CB\\_Extension.CompareTypeAndSubType<T>\(object\)](#),  
[Object\\_CB\\_Extension.CompareTypeAndSubType<T>\(object, bool\)](#),  
[Object\\_CB\\_Extension.CompareType<T>\(object\)](#)

## Constructors

### XMLIRWElement(string)

Creates a new instance of the XMLIRW element.

```
public XMLIRWElement(string name)
```

## Parameters

`name` [string](#)

## XMLIRWElement(string, object)

Creates a new instance of the XMLIRW element.

```
public XMLIRWElement(string name, object value)
```

### Parameters

`name` [string](#)

`value` [object](#)

## XMLIRWElement(string, object, params XMLIRW[])

Creates a new instance of the XMLIRW element.

```
public XMLIRWElement(string name, object value, params XMLIRW[] itens)
```

### Parameters

`name` [string](#)

`value` [object](#)

`itens` [XMLIRW](#)[]

## XMLIRWElement(string, XMLIRWValue)

Creates a new instance of the XMLIRW element.

```
[Obsolete("Use the XMLIRWElement(string, object) constructor.")]  
public XMLIRWElement(string name, XMLIRWValue value)
```

### Parameters

`name` [string](#)

`value` [XMLIRWValue](#)

## XMLIRWElement(string, XMLIRWValue, params XMLIRW[])

Creates a new instance of the XMLIRW element.

```
[Obsolete("Use the XMLIRWElement(string, object, params XMLIRW[]) constructor.")]
public XMLIRWElement(string name, XMLIRWValue value, params XMLIRW[] itens)
```

### Parameters

`name` [string](#)

`value` [XMLIRWValue](#)

`itens` [XMLIRW\[\]](#)

## XMLIRWElement(string, params XMLIRW[])

Creates a new instance of the XMLIRW element.

```
public XMLIRWElement(string name, params XMLIRW[] itens)
```

### Parameters

`name` [string](#)

`itens` [XMLIRW\[\]](#)

## XMLIRWElement(XMLIRW, string, object)

Creates a new instance of the XMLIRW element.

```
public XMLIRWElement(XMLIRW parent, string name, object value)
```

## Parameters

parent [XMLIRW](#)

name [string](#)

value [object](#)

## XMLIRWElememt(XMLIRW, string, object, params XMLIRW[])

Creates a new instance of the XMLIRW element.

```
public XMLIRWElememt(XMLIRW parent, string name, object value, params XMLIRW[] itens)
```

## Parameters

parent [XMLIRW](#)

name [string](#)

value [object](#)

itens [XMLIRW\[\]](#)

## XMLIRWElememt(XMLIRW, string, XMLIRWValue)

Creates a new instance of the XMLIRW element.

```
[Obsolete("Use the XMLIRWElememt(XMLIRW, string, object) constructor.")]
public XMLIRWElememt(XMLIRW parent, string name, XMLIRWValue value)
```

## Parameters

parent [XMLIRW](#)

name [string](#)

value [XMLIRWValue](#)

## XMLIRWElement(XMLIRW, string, XMLIRWValue, params XMLIRW[])

Creates a new instance of the XMLIRW element.

```
[Obsolete("Use the XMLIRWElement(XMLIRW, string, object, params XMLIRW[]) constructor.")]
public XMLIRWElement(XMLIRW parent, string name, XMLIRWValue value, params XMLIRW[] itens)
```

### Parameters

parent [XMLIRW](#)

name [string](#) ↗

value [XMLIRWValue](#)

itens [XMLIRW\[\]](#)

## XMLIRWElement(XMLIRW, string, params XMLIRW[])

Creates a new instance of the XMLIRW element.

```
public XMLIRWElement(XMLIRW parent, string name, params XMLIRW[] itens)
```

### Parameters

parent [XMLIRW](#)

name [string](#) ↗

itens [XMLIRW\[\]](#)

## Properties

### AttributeCount

Gets the count of attributes on the element.

```
public int AttributeCount { get; }
```

Property Value

[int](#)

## Attributes

Gets the attributes on the element.

```
public IEnumerable<XMLIRW> Attributes { get; }
```

Property Value

[IEnumerable](#) <XMLIRW>

## IsEmpty

Checks whether the element has sub-elements or attributes.

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

Property Value

[bool](#)

## Name

Returns or sets the name of the XMLIRW object.

```
public override string Name { get; set; }
```

Property Value

[string](#)

## NoAttributes

Checks whether the element has attributes.

```
public bool NoAttributes { get; }
```

Property Value

[bool](#) ↗

## NoElements

Checks whether the element has sub-elements.

```
public bool NoElements { get; }
```

Property Value

[bool](#) ↗

## Parent

Returns or sets the name of the parent object of the XMLIRW element.

```
public override XMLIRW Parent { get; set; }
```

Property Value

[XMLIRW](#)

## Text

Returns or sets the text of the XMLIRW element.

```
public XMLIRWText Text { get; set; }
```

## Property Value

[XMLIRWText](#)

## Type

Returns or sets the type of the XMLIRW element.

```
public override XmlNodeType Type { get; set; }
```

## Property Value

[XmlNodeType](#)

## Value

Returns or sets the text of the XMLIRW element.

```
[Obsolete("Use the Text property.")]  
public XMLIRWValue Value { get; set; }
```

## Property Value

[XMLIRWValue](#)

## ValueIsEmpty

Checks whether the element has a text value.

```
public bool ValueIsEmpty { get; }
```

## Property Value

[bool](#)

# Methods

## Add(XMLIRW)

Adds a new XMLIRW element.

```
public bool Add(XMLIRW element)
```

Parameters

`element` [XMLIRW](#)

Returns

`bool` ↗

Returns `true` when the element is added XMLIRW.

## Dispose()

Discard the resources of the XMLIRW element.

```
public override void Dispose()
```

## Dispose(bool)

Discard the resources of the XMLIRW element.

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

Parameters

`disposing` `bool` ↗

## ~XMLIRWEElement()

Called when the object is finished.

```
protected ~XMLIRWElement()
```

## GetEnumerator()

Returns an enumerator that iterates through a collection.

```
public IEnumerator<XMLIRW> GetEnumerator()
```

Returns

[IEnumerator](#)<[XMLIRW](#)>

## ToString()

Creates a [string](#) representation of the [object](#).

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

Returns

[string](#)

# Class XMLIRWProcessingInstruction

Namespace: [System.Xml](#)

Assembly: Cobilas.Core.dll

Represents an XML element of type ProcessingInstruction.

```
public class XMLIRWProcessingInstruction : XMLIRW, ITextValue, IEnumerable<XMLIRWAttribute>,  
IEnumerable, IDisposable
```

## Inheritance

[object](#) ← [XMLIRW](#) ← XMLIRWProcessingInstruction

## Implements

[ITextValue](#), [IEnumerable](#)<[XMLIRWAttribute](#)>, [IEnumerable](#), [IDisposable](#)

## Inherited Members

[XMLIRW.ToString\(\)](#), [object.Equals\(object\)](#), [object.Equals\(object, object\)](#),  
[object.ReferenceEquals\(object, object\)](#), [object.GetHashCode\(\)](#), [object.GetType\(\)](#),  
[object.MemberwiseClone\(\)](#)

## Extension Methods

[Object\\_CB\\_Extension.CompareType\(object, Type\)](#),  
[Object\\_CB\\_Extension.CompareType\(object, params Type\[\]\)](#),  
[Object\\_CB\\_Extension.CompareTypeAndSubType\(object, Type\)](#),  
[Object\\_CB\\_Extension.CompareTypeAndSubType\(object, Type, bool\)](#),  
[Object\\_CB\\_Extension.CompareTypeAndSubType<T>\(object\)](#),  
[Object\\_CB\\_Extension.CompareTypeAndSubType<T>\(object, bool\)](#),  
[Object\\_CB\\_Extension.CompareType<T>\(object\)](#).

## Constructors

### XMLIRWProcessingInstruction(string, object)

Creates a new instance of the XMLIRW element.

```
public XMLIRWProcessingInstruction(string name, object value)
```

## Parameters

name [string](#)

value [object](#)

## XMLIRWProcessingInstruction(string, XMLIRWAttribute[])

Creates a new instance of the XMLIRW element.

```
public XMLIRWProcessingInstruction(string name, XMLIRWAttribute[] attributes)
```

## Parameters

name [string](#)

attributes [XMLIRWAttribute](#)[]

## XMLIRWProcessingInstruction(string, XMLIRWValue)

Creates a new instance of the XMLIRW element.

```
[Obsolete("Use the XMLIRWProcessingInstruction(string, object) constructor.")]  
public XMLIRWProcessingInstruction(string name, XMLIRWValue value)
```

## Parameters

name [string](#)

value [XMLIRWValue](#)

## XMLIRWProcessingInstruction(XMLIRW, string, object)

Creates a new instance of the XMLIRW element.

```
public XMLIRWProcessingInstruction(XMLIRW parent, string name, object value)
```

## Parameters

parent [XMLIRW](#)

name [string](#)

value [object](#)

## XMLIRWProcessingInstruction(XMLIRW, string, XMLIRWAttribute[])

Creates a new instance of the XMLIRW element.

```
public XMLIRWProcessingInstruction(XMLIRW parent, string name, XMLIRWAttribute[] attributes)
```

## Parameters

parent [XMLIRW](#)

name [string](#)

attributes [XMLIRWAttribute](#)[]

## XMLIRWProcessingInstruction(XMLIRW, string, XMLIRWValue)

Creates a new instance of the XMLIRW element.

```
[Obsolete("Use the XMLIRWProcessingInstruction(XMLIRW, string, object) constructor.")]  
public XMLIRWProcessingInstruction(XMLIRW parent, string name, XMLIRWValue value)
```

## Parameters

parent [XMLIRW](#)

name [string](#)

value [XMLIRWValue](#)

# Properties

## AttributeCount

Returns the number of attributes in the list.

```
public int AttributeCount { get; }
```

### Property Value

[int](#)

## IsAttributeList

Checks whether the value is a list of attributes.

```
public bool IsAttributeList { get; }
```

### Property Value

[bool](#)

## Name

Returns or sets the name of the XMLIRW object.

```
public override string Name { get; set; }
```

### Property Value

[string](#)

## Parent

Returns or sets the name of the parent object of the XMLIRW element.

```
public override XMLIRW Parent { get; set; }
```

## Property Value

[XMLIRW](#)

## Text

Returns or sets the text of the XMLIRW element.

```
public XMLIRWText Text { get; set; }
```

## Property Value

[XMLIRWText](#)

## Type

Returns or sets the type of the XMLIRW element.

```
public override XmlNodeType Type { get; set; }
```

## Property Value

[XmlNodeType](#)

## Value

Returns or sets the text of the XMLIRW element.

```
[Obsolete("Use the Text property.")]  
public XMLIRWValue Value { get; set; }
```

## Property Value

## Methods

### Dispose()

Discard the resources of the XMLIRW element.

```
public override void Dispose()
```

### Dispose(bool)

Discard the resources of the XMLIRW element.

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

Parameters

`disposing` [bool](#)

### ~XMLIRWProcessingInstruction()

Called when the object is finished.

```
protected ~XMLIRWProcessingInstruction()
```

### GetEnumerator()

Returns an enumerator that iterates through a collection.

```
public IEnumerator<XMLIRWAttribute> GetEnumerator()
```

Returns

[IEnumerator](#) <[XMLIRWAttribute](#)>

# Class XMLIRWText

Namespace: [System.Xml](#)

Assembly: Cobilas.Core.dll

Represents XML text.

```
public class XMLIRWText : XMLIRW, IDisposable, IConvertible
```

## Inheritance

[object](#) ← [XMLIRW](#) ← XMLIRWText

## Implements

[IDisposable](#), [IConvertible](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.GetHashCode\(\)](#), [object.GetType\(\)](#), [object.MemberwiseClone\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#),  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#),  
[Object CB Extension.CompareType<T>\(object\)](#)

# Constructors

## XMLIRWText(object)

Creates a new instance of the XMLIRW element.

```
public XMLIRWText(object textValue)
```

## Parameters

`textValue` [object](#)

## XMLIRWText(XMLIRW, object)

Creates a new instance of the XMLIRW element.

```
public XMLIRWText(XMLIRW parent, object textValue)
```

### Parameters

`parent` [XMLIRW](#)

`textValue` [object](#)

## Fields

### textValue

The value of the XML text.

```
protected object textValue
```

### Field Value

[object](#)

## Properties

### Empty

Creates an empty instance.

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

### Property Value

## XMLIRWText

### IsNull

Check if the value is null.

```
public bool IsNull { get; }
```

### Property Value

[bool](#)

### Name

Returns or sets the name of the XMLIRW object.

```
public override string Name { get; set; }
```

### Property Value

[string](#)

### Parent

Returns or sets the name of the parent object of the XMLIRW element.

```
public override XMLIRW Parent { get; set; }
```

### Property Value

[XMLIRW](#)

### Type

Returns or sets the type of the XMLIRW element.

```
public override XmlNodeType Type { get; set; }
```

## Property Value

[XmlNodeType](#)

## Value

The value of the XML text.

```
public object Value { get; }
```

## Property Value

[object](#)

# Methods

## Dispose()

Discard the resources of the XMLIRW element.

```
public override void Dispose()
```

## ToString()

Creates a [string](#) representation of the [object](#).

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

## Returns

[string](#)

## ToString(IFormatProvider)

Creates a [string](#) representation of the [object](#).

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

Parameters

[provider](#) [IFormatProvider](#)

Returns

[string](#)

## Operators

### explicit operator bool(XMLIRWText)

Provide a conversion from type [XMLIRWText](#) to [bool](#).

```
public static explicit operator bool(XMLIRWText text)
```

Parameters

[text](#) [XMLIRWText](#)

Returns

[bool](#)

### explicit operator byte(XMLIRWText)

Provide a conversion from type [XMLIRWText](#) to [byte](#).

```
public static explicit operator byte(XMLIRWText text)
```

Parameters

`text` [XMLIRWText](#)

Returns

[byte](#)

## explicit operator char[](XMLIRWText)

Provide a conversion from type [XMLIRWText](#) to [char](#)[].

```
public static explicit operator char[](XMLIRWText text)
```

Parameters

`text` [XMLIRWText](#)

Returns

[char](#)[]

## explicit operator DateTime(XMLIRWText)

Provide a conversion from type [XMLIRWText](#) to [DateTime](#).

```
public static explicit operator DateTime(XMLIRWText text)
```

Parameters

`text` [XMLIRWText](#)

Returns

[DateTime](#)

## explicit operator decimal(XMLIRWText)

Provide a conversion from type [XMLIRWText](#) to [decimal](#).

```
public static explicit operator decimal(XMLIRWText text)
```

Parameters

[text XMLIRWText](#)

Returns

[decimal](#)

## explicit operator double(XMLIRWText)

Provide a conversion from type [XMLIRWText](#) to [double](#).

```
public static explicit operator double(XMLIRWText text)
```

Parameters

[text XMLIRWText](#)

Returns

[double](#)

## explicit operator short(XMLIRWText)

Provide a conversion from type [XMLIRWText](#) to [short](#).

```
public static explicit operator short(XMLIRWText text)
```

Parameters

[text XMLIRWText](#)

Returns

[short](#)

## explicit operator int(XMLIRWText)

Provide a conversion from type [XMLIRWText](#) to [int](#).

```
public static explicit operator int(XMLIRWText text)
```

Parameters

**text** [XMLIRWText](#)

Returns

[int](#)

## explicit operator long(XMLIRWText)

Provide a conversion from type [XMLIRWText](#) to [long](#).

```
public static explicit operator long(XMLIRWText text)
```

Parameters

**text** [XMLIRWText](#)

Returns

[long](#)

## explicit operator sbyte(XMLIRWText)

Provide a conversion from type [XMLIRWText](#) to [sbyte](#).

```
public static explicit operator sbyte(XMLIRWText text)
```

Parameters

`text` [XMLIRWText](#)

Returns

[sbyte](#)

## explicit operator float(XMLIRWText)

Provide a conversion from type [XMLIRWText](#) to [float](#).

```
public static explicit operator float(XMLIRWText text)
```

Parameters

`text` [XMLIRWText](#)

Returns

[float](#)

## explicit operator string(XMLIRWText)

Provide a conversion from type [XMLIRWText](#) to [string](#).

```
public static explicit operator string(XMLIRWText text)
```

Parameters

`text` [XMLIRWText](#)

Returns

[string](#)

## explicit operator ushort(XMLIRWText)

Provide a conversion from type [XMLIRWText](#) to [ushort](#).

```
public static explicit operator ushort(XMLIRWText text)
```

Parameters

[text XMLIRWText](#)

Returns

[ushort](#)

## explicit operator uint(XMLIRWText)

Provide a conversion from type [XMLIRWText](#) to [uint](#).

```
public static explicit operator uint(XMLIRWText text)
```

Parameters

[text XMLIRWText](#)

Returns

[uint](#)

## explicit operator ulong(XMLIRWText)

Provide a conversion from type [XMLIRWText](#) to [ulong](#).

```
public static explicit operator ulong(XMLIRWText text)
```

Parameters

[text XMLIRWText](#)

Returns

[ulong](#)

## implicit operator XMLIRWText(bool)

Provide a conversion from type [bool](#) to [XMLIRWText](#).

```
public static implicit operator XMLIRWText(bool text)
```

Parameters

**text** [bool](#)

Returns

[XMLIRWText](#)

## implicit operator XMLIRWText(byte)

Provide a conversion from type [byte](#) to [XMLIRWText](#).

```
public static implicit operator XMLIRWText(byte text)
```

Parameters

**text** [byte](#)

Returns

[XMLIRWText](#)

## implicit operator XMLIRWText(char[])

Provide a conversion from type [char](#)[] to [XMLIRWText](#).

```
public static implicit operator XMLIRWText(char[] text)
```

Parameters

`text` [char\[\]](#)

Returns

[XMLIRWText](#)

## implicit operator XMLIRWText(DateTime)

Provide a conversion from type [DateTime](#) to [XMLIRWText](#).

```
public static implicit operator XMLIRWText(DateTime text)
```

Parameters

`text` [DateTime](#)

Returns

[XMLIRWText](#)

## implicit operator XMLIRWText(decimal)

Provide a conversion from type [decimal](#) to [XMLIRWText](#).

```
public static implicit operator XMLIRWText(decimal text)
```

Parameters

`text` [decimal](#)

Returns

[XMLIRWText](#)

## implicit operator XMLIRWText(double)

Provide a conversion from type [double](#) to [XMLIRWText](#).

```
public static implicit operator XMLIRWText(double text)
```

Parameters

text [double](#)

Returns

[XMLIRWText](#)

## implicit operator XMLIRWText(short)

Provide a conversion from type [short](#) to [XMLIRWText](#).

```
public static implicit operator XMLIRWText(short text)
```

Parameters

text [short](#)

Returns

[XMLIRWText](#)

## implicit operator XMLIRWText(int)

Provide a conversion from type [int](#) to [XMLIRWText](#).

```
public static implicit operator XMLIRWText(int text)
```

Parameters

text [int](#)

Returns

[XMLIRWText](#)

## implicit operator XMLIRWText(long)

Provide a conversion from type [long](#) to [XMLIRWText](#).

```
public static implicit operator XMLIRWText(long text)
```

Parameters

**text** [long](#)

Returns

[XMLIRWText](#)

## implicit operator XMLIRWText(sbyte)

Provide a conversion from type [sbyte](#) to [XMLIRWText](#).

```
public static implicit operator XMLIRWText(sbyte text)
```

Parameters

**text** [sbyte](#)

Returns

[XMLIRWText](#)

## implicit operator XMLIRWText(float)

Provide a conversion from type [float](#) to [XMLIRWText](#).

```
public static implicit operator XMLIRWText(float text)
```

Parameters

`text` [float](#)

Returns

[XMLIRWText](#)

## implicit operator XMLIRWText(string)

Provide a conversion from type [string](#) to [XMLIRWText](#).

```
public static implicit operator XMLIRWText(string text)
```

Parameters

`text` [string](#)

Returns

[XMLIRWText](#)

## implicit operator XMLIRWText(ushort)

Provide a conversion from type [ushort](#) to [XMLIRWText](#).

```
public static implicit operator XMLIRWText(ushort text)
```

Parameters

`text` [ushort](#)

Returns

[XMLIRWText](#)

## implicit operator XMLIRWText(uint)

Provide a conversion from type [uint](#) to [XMLIRWText](#).

```
public static implicit operator XMLIRWText(uint text)
```

Parameters

text [uint](#)

Returns

[XMLIRWText](#)

## implicit operator XMLIRWText(ulong)

Provide a conversion from type [ulong](#) to [XMLIRWText](#).

```
public static implicit operator XMLIRWText(ulong text)
```

Parameters

text [ulong](#)

Returns

[XMLIRWText](#)

# Struct XMLIRWValue

Namespace: [System.Xml](#)

Assembly: Cobilas.Core.dll

Represents the value of a tag.

```
[Obsolete("Use the XMLIRWText class to define values.")]
public struct XMLIRWValue : IDisposable, IEquatable<XMLIRWValue>, IConvertible
```

## Implements

[IDisposable](#), [IEquatable<XMLIRWValue>](#), [IConvertible](#)

## Inherited Members

[object.Equals\(object, object\)](#), [object.ReferenceEquals\(object, object\)](#), [object.GetType\(\)](#)

## Extension Methods

[Object CB Extension.CompareType\(object, Type\)](#),  
[Object CB Extension.CompareType\(object, params Type\[\]\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type\)](#),  
[Object CB Extension.CompareTypeAndSubType\(object, Type, bool\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object\)](#),  
[Object CB Extension.CompareTypeAndSubType<T>\(object, bool\)](#),  
[Object CB Extension.CompareType<T>\(object\)](#)

## Constructors

### XMLIRWValue(object)

```
public XMLIRWValue(object value)
```

## Parameters

**value** [object](#)

## Fields

# Empty

Represents an empty value.

```
public static readonly XMLIRWValue Empty
```

## Field Value

[XMLIRWValue](#)

# Methods

## Dispose()

Performs application-defined tasks associated with freeing, releasing, or resetting unmanaged resources.

```
public void Dispose()
```

## Equals(object)

Indicates whether this instance and a specified object are equal.

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

## Parameters

**obj** [object](#)

The object to compare with the current instance.

## Returns

[bool](#)

[true](#) if **obj** and this instance are the same type and represent the same value; otherwise, [false](#).

## Equals(XMLIRWValue)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(XMLIRWValue other)
```

Parameters

**other** [XMLIRWValue](#)

An object to compare with this object.

Returns

[bool](#)

[true](#) if the current object is equal to the **other** parameter; otherwise, [false](#).

## GetHashCode()

Returns the hash code for this instance.

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

Returns

[int](#)

A 32-bit signed integer that is the hash code for this instance.

## ToString()

Returns the fully qualified type name of this instance.

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

Returns

## [string](#)

The fully qualified type name.

## ToString(IFormatProvider)

Converts the value of this instance to an equivalent [string](#) using the specified culture-specific formatting information.

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

### Parameters

#### [provider](#) [IFormatProvider](#)

An [IFormatProvider](#) interface implementation that supplies culture-specific formatting information.

### Returns

## [string](#)

A [string](#) instance equivalent to the value of this instance.

## Operators

### operator ==(XMLIRWValue, XMLIRWValue)

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

### Parameters

#### [left](#) [XMLIRWValue](#)

#### [right](#) [XMLIRWValue](#)

### Returns

## [bool](#)

## explicit operator string(XMLIRWValue)

```
public static explicit operator string(XMLIRWValue v)
```

Parameters

v [XMLIRWValue](#)

Returns

[string](#)

## operator !=(XMLIRWValue, XMLIRWValue)

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

Parameters

left [XMLIRWValue](#)

right [XMLIRWValue](#)

Returns

[bool](#)