Table of Contents

BuildFromCSharpSourceCode ... 3
Classes
CSharp ... 4
BuildFromProject ... 5
Issue8540 ... 6
A ... 7
Classes
A ... 8
B ... 9
Classes
B ... 10
Classes
Class1 ... 11
Class1.Issue8665 ... 16
Class1.Issue8696Attribute ... 18
Class1.Issue8948 ... 20
Class1.Test<T> ... 21
Inheritdoc ... 22
Inheritdoc.Issue6366 ... 23
Inheritdoc.Issue6366.Class1<T> ... 24
Inheritdoc.Issue6366.Class2 ... 25
Inheritdoc.Issue7035 ... 26
Inheritdoc.Issue7484 ... 27
Inheritdoc.Issue8101 ... 29
Structs
Inheritdoc.Issue8129 ... 31
Interfaces
Class1.IIssue8948 ... 32
IInheritdoc ... 33
Enums
Class1.Issue9260 ... 34
BuildFromVBSourceCode ... 35
Classes
BaseClass1 ... 36
Class1 ... 37
CatLibrary ... 39
Core ... 41
Classes

ContainersRefType.ContainersRefTypeChild 42
ExplicitLayoutClass 43
Issue231 44
Structs
ContainersRefType 45
Interfaces
ContainersRefType.ContainersRefTypeChildInterface 47
Enums
ContainersRefType.ColorType 48
Delegates
ContainersRefType.ContainersRefTypeDelegate 49
Classes
Cat<T, K> 50
CatException<T> 57
Complex<T, J> 58
ICatExtension 59
Tom 61
TomFromBaseClass 63
Interfaces
IAnimal 64
ICat 66
Delegates
FakeDelegate<T> 67
MRefDelegate<K, T, L> 68
MRefNormalDelegate 69
MRef 70
Demo 71
Enumeration 72
Enums
ColorType 73

Namespace BuildFromCSharpSourceCode

Classes

CSharp

Class CSharp

Namespace: BuildFromCSharpSourceCode

```
public class CSharp
```

Inheritance

object ← CSharp

Inherited Members

object.Equals(object?), object.Equals(object?, object?), object.GetHashCode(), object.GetType(), object.MemberwiseClone(), object.ReferenceEquals(object?, object?), object.ToString()

Methods

Main(string[])

```
public static void Main(string[] args)
```

Parameters

`args` string[]

Namespace BuildFromProject

Namespaces

BuildFromProject.Issue8540

Classes

Inheritdoc.Issue6366.Class1<T>

Class1

Inheritdoc.Issue6366.Class2

Inheritdoc

Inheritdoc.Issue6366

Inheritdoc.Issue7035

Inheritdoc.Issue7484

This is a test class to have something for DocFX to document.

Inheritdoc.Issue8101

Class1.Issue8665

Class1.Issue8696Attribute

Class1.Issue8948

Class1.Test<T>

Structs

Inheritdoc.Issue8129

Interfaces

IInheritdoc

Class1.IIssue8948

Enums

Class1.Issue9260

Namespace BuildFromProject.Issue8540

Namespaces

BuildFromProject.Issue8540.A

BuildFromProject.Issue8540.B

Namespace BuildFromProject.Issue8540.A

Classes

A

Class A

Namespace: BuildFromProject.Issue8540.A
Assembly: BuildFromProject.dll

```
public class A
```

Inheritance

object ← A

Inherited Members

object.Equals(object?), object.Equals(object?, object?), object.GetHashCode(),
object.GetType(), object.MemberwiseClone(), object.ReferenceEquals(object?, object?),
object.ToString()

Namespace BuildFromProject.Issue8540.B

Classes

B

Class B

Namespace: BuildFromProject.Issue8540.B
Assembly: BuildFromProject.dll

```
public class B
```

Inheritance

object ← B

Inherited Members

object.Equals(object?), object.Equals(object?, object?), object.GetHashCode(),
object.GetType(), object.MemberwiseClone(), object.ReferenceEquals(object?, object?),
object.ToString()

Class Class1

Namespace: BuildFromProject
Assembly: BuildFromProject.dll

```
public class Class1 : IClass1
```

Inheritance

object ← Class1

Implements

IClass1

Inherited Members

object.Equals(object?), object.Equals(object?, object?), object.GetHashCode(), object.GetType(), object.MemberwiseClone(), object.ReferenceEquals(object?, object?), object.ToString()

Methods

Issue1651()

Pricing models are used to calculate theoretical option values

- 1Black Scholes
- 2Black76
- 3Black76Fut
- 4Equity Tree
- 5Variance Swap
- 6Dividend Forecast

```
public void Issue1651()
```

Issue1887()

IConfiguration related helper and extension routines.

```
public void Issue1887()
```

Issue2623()

```
public void Issue2623()
```

Examples

```
MyClass myClass = new MyClass();

void Update()
{
    myClass.Execute();
}
```

Remarks

For example:

```
MyClass myClass = new MyClass();

void Update()
{
    myClass.Execute();
}
```

Issue2723()

```
public void Issue2723()
```

Remarks

> **NOTE**
> This is a <note>. & " '

Inline `<angle brackets>`.

link

```
for (var i = 0; i > 10; i++) // & " '
var range = new Range<int> { Min = 0, Max = 10 };
```

```
  var range = new Range<int> { Min = 0, Max = 10 };
```

Issue4017()

```
  public void Issue4017()
```

Examples

```
  public void HookMessageDeleted(BaseSocketClient client)
  {
      client.MessageDeleted += HandleMessageDelete;
  }

  public Task HandleMessageDelete(Cacheable<IMessage, ulong> cachedMessage, ISocketMessag
  {
      // check if the message exists in cache; if not, we cannot report what was removed
      if (!cachedMessage.HasValue) return;
      var message = cachedMessage.Value;
      Console.WriteLine($"A message ({message.Id}) from {message.Author} was removed fron
          + Environment.NewLine
          + message.Content);
      return Task.CompletedTask;
  }
```

Remarks

```
  void Update()
  {
      myClass.Execute();
  }
```

Issue4392()

```
  public void Issue4392()
```

Remarks

```
@"\\?\" @"\\?\"
```

Issue7484()

```
public void Issue7484()
```

Remarks

There's really no reason to not believe that this class can test things.

Term	Description
A Term	A Description
Bee Term	Bee Description

Issue8764<T>()

```
public void Issue8764<T>() where T : unmanaged
```

Type Parameters

`T`

Issue896()

Test

```
public void Issue896()
```

See Also

Class1.Test<T>, Class1

Issue9216()

Calculates the determinant of a 3-dimensional matrix:

$$A = \begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix}$$

Returns the smallest value:

$$\begin{cases} a, a < b \\ b, b > a \end{cases}$$

```
public static double Issue9216()
```

Returns

double

XmlCommentIncludeTag()

This method should do something...

```
public void XmlCommentIncludeTag()
```

Remarks

This is remarks.

Class Class1.Issue8665

Namespace: BuildFromProject
Assembly: BuildFromProject.dll

```
public class Class1.Issue8665
```

Inheritance

object ← Class1.Issue8665

Inherited Members

object.Equals(object?), object.Equals(object?, object?), object.GetHashCode(), object.GetType(), object.MemberwiseClone(), object.ReferenceEquals(object?, object?), object.ToString()

Constructors

Issue8665()

```
public Issue8665()
```

Issue8665(int)

```
public Issue8665(int foo)
```

Parameters

`foo` int

Issue8665(int, char)

```
public Issue8665(int foo, char bar)
```

Parameters

`foo` int

`bar` char

Issue8665(int, char, string)

```
public Issue8665(int foo, char bar, string baz)
```

Parameters

`foo` int

`bar` char

`baz` string

Properties

Bar

```
public char Bar { get; }
```

Property Value

char

Baz

```
public string Baz { get; }
```

Property Value

string

Foo

```
public int Foo { get; }
```

Property Value

int

Class Class1.Issue8696Attribute

Namespace: BuildFromProject
Assembly: BuildFromProject.dll

```
public class Class1.Issue8696Attribute : Attribute
```

Inheritance

object ← Attribute ← Class1.Issue8696Attribute

Inherited Members

Attribute.Equals(object?), Attribute.GetCustomAttribute(Assembly, Type), Attribute.GetCustomAttribute(Assembly, Type, bool), Attribute.GetCustomAttribute(MemberInfo, Type), Attribute.GetCustomAttribute(MemberInfo, Type, bool), Attribute.GetCustomAttribute(Module, Type), Attribute.GetCustomAttribute(Module, Type, bool), Attribute.GetCustomAttribute(ParameterInfo, Type), Attribute.GetCustomAttribute(ParameterInfo, Type, bool), Attribute.GetCustomAttributes(Assembly), Attribute.GetCustomAttributes(Assembly, bool), Attribute.GetCustomAttributes(Assembly, Type), Attribute.GetCustomAttributes(Assembly, Type, bool), Attribute.GetCustomAttributes(MemberInfo), Attribute.GetCustomAttributes(MemberInfo, bool), Attribute.GetCustomAttributes(MemberInfo, Type), Attribute.GetCustomAttributes(MemberInfo, Type, bool), Attribute.GetCustomAttributes(Module), Attribute.GetCustomAttributes(Module, bool), Attribute.GetCustomAttributes(Module, Type), Attribute.GetCustomAttributes(Module, Type, bool), Attribute.GetCustomAttributes(ParameterInfo), Attribute.GetCustomAttributes(ParameterInfo, bool), Attribute.GetCustomAttributes(ParameterInfo, Type), Attribute.GetCustomAttributes(ParameterInfo, Type, bool), Attribute.GetHashCode(), Attribute.IsDefaultAttribute(), Attribute.IsDefined(Assembly, Type), Attribute.IsDefined(Assembly, Type, bool), Attribute.IsDefined(MemberInfo, Type), Attribute.IsDefined(MemberInfo, Type, bool), Attribute.IsDefined(Module, Type), Attribute.IsDefined(Module, Type, bool), Attribute.IsDefined(ParameterInfo, Type), Attribute.IsDefined(ParameterInfo, Type, bool), Attribute.Match(object?), Attribute.TypeId, object.Equals(object?), object.Equals(object?, object?), object.GetHashCode(), object.GetType(), object.MemberwiseClone(), object.ReferenceEquals(object?, object?), object.ToString()

Constructors

Issue8696Attribute(string?, int, int, string[]?, bool, Type?)

```
[Class1.Issue8696("Changes the name of the server in the server list", 0, 0, null, fals
public Issue8696Attribute(string? description = null, int boundsMin = 0, int boundsMax
```

Parameters

`description` string?

`boundsMin` int

`boundsMax` int

`validGameModes` string[]?

`hasMultipleSelections` bool

`enumType` Type?

Class Class1.Issue8948

Namespace: BuildFromProject
Assembly: BuildFromProject.dll

```
public class Class1.Issue8948 : Class1.IIssue8948
```

Inheritance

object ← Class1.Issue8948

Implements

Class1.IIssue8948

Inherited Members

object.Equals(object?), object.Equals(object?, object?), object.GetHashCode(), object.GetType(), object.MemberwiseClone(), object.ReferenceEquals(object?, object?), object.ToString()

Methods

DoNothing<T>()

Does nothing with generic type T.

```
public void DoNothing<T>()
```

Type Parameters

T

A generic type.

Class Class1.Test<T>

Namespace: BuildFromProject
Assembly: BuildFromProject.dll

```
public class Class1.Test<T>
```

Type Parameters

T

Inheritance

object ← Class1.Test<T>

Inherited Members

object.Equals(object?), object.Equals(object?, object?), object.GetHashCode(), object.GetType(), object.MemberwiseClone(), object.ReferenceEquals(object?, object?), object.ToString()

Class Inheritdoc

Namespace: BuildFromProject
Assembly: BuildFromProject.dll

```
public class Inheritdoc : IInheritdoc, IDisposable
```

Inheritance

object ← Inheritdoc

Implements

IInheritdoc, IDisposable

Inherited Members

object.Equals(object?), object.Equals(object?, object?), object.GetHashCode(), object.GetType(), object.MemberwiseClone(), object.ReferenceEquals(object?, object?), object.ToString()

Methods

Dispose()

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

```
public void Dispose()
```

Issue7628()

This method should do something...

```
public void Issue7628()
```

Issue7629()

This method should do something...

```
public void Issue7629()
```

Class Inheritdoc.Issue6366

Namespace: BuildFromProject
Assembly: BuildFromProject.dll

```
public class Inheritdoc.Issue6366
```

Inheritance

object ← Inheritdoc.Issue6366

Inherited Members

object.Equals(object?), object.Equals(object?, object?), object.GetHashCode(), object.GetType(), object.MemberwiseClone(), object.ReferenceEquals(object?, object?), object.ToString()

Class Inheritdoc.Issue6366.Class1<T>

Namespace: BuildFromProject
Assembly: BuildFromProject.dll

```
public abstract class Inheritdoc.Issue6366.Class1<T>
```

Type Parameters

`T`

Inheritance

object ← Inheritdoc.Issue6366.Class1<T>

Inherited Members

object.Equals(object?), object.Equals(object?, object?), object.GetHashCode(), object.GetType(), object.MemberwiseClone(), object.ReferenceEquals(object?, object?), object.ToString()

Methods

TestMethod1(T, int)

This text inherited.

```
public abstract T TestMethod1(T parm1, int parm2)
```

Parameters

`parm1` T

This text NOT inherited.

`parm2` int

This text inherited.

Returns

T

This text inherited.

Class Inheritdoc.Issue6366.Class2

Namespace: BuildFromProject
Assembly: BuildFromProject.dll

```
public class Inheritdoc.Issue6366.Class2 : Inheritdoc.Issue6366.Class1<bool>
```

Inheritance

object ← Inheritdoc.Issue6366.Class1<bool> ← Inheritdoc.Issue6366.Class2

Inherited Members

Inheritdoc.Issue6366.Class1<bool>.TestMethod1(bool, int), object.Equals(object?), object.Equals(object?, object?), object.GetHashCode(), object.GetType(), object.MemberwiseClone(), object.ReferenceEquals(object?, object?), object.ToString()

Methods

TestMethod1(bool, int)

This text inherited.

```
public override bool TestMethod1(bool parm1, int parm2)
```

Parameters

`parm1` bool

This text NOT inherited.

`parm2` int

This text inherited.

Returns

bool

This text inherited.

Class Inheritdoc.Issue7035

Namespace: BuildFromProject
Assembly: BuildFromProject.dll

```
public class Inheritdoc.Issue7035
```

Inheritance

object ← Inheritdoc.Issue7035

Inherited Members

object.Equals(object?), object.Equals(object?, object?), object.GetHashCode(), object.GetType(), object.MemberwiseClone(), object.ReferenceEquals(object?, object?), object.ToString()

Methods

A()

```
public void A()
```

B()

```
public void B()
```

Class Inheritdoc.Issue7484

Namespace: BuildFromProject
Assembly: BuildFromProject.dll

This is a test class to have something for DocFX to document.

```
public class Inheritdoc.Issue7484
```

Inheritance

object ← Inheritdoc.Issue7484

Inherited Members

object.Equals(object?), object.Equals(object?, object?), object.GetHashCode(), object.GetType(), object.MemberwiseClone(), object.ReferenceEquals(object?, object?), object.ToString()

Remarks

We're going to talk about things now.

BoolReturningMethod(bool)	• Simple method to generate docs for.
DoDad	• A string that could have something.

Constructors

Issue7484()

This is a constructor to document.

```
public Issue7484()
```

Properties

DoDad

A string that could have something.

```
public string DoDad { get; }
```

Property Value

string

Methods

BoolReturningMethod(bool)

Simple method to generate docs for.

```
public bool BoolReturningMethod(bool source)
```

Parameters

`source` bool

A meaningless boolean value, much like most questions in the world.

Returns

bool

An exactly equivalently meaningless boolean value, much like most answers in the world.

Remarks

I'd like to take a moment to thank all of those who helped me get to a place where I can write documentation like this.

Class Inheritdoc.Issue8101

Namespace: BuildFromProject
Assembly: BuildFromProject.dll

```
public class Inheritdoc.Issue8101
```

Inheritance

object ← Inheritdoc.Issue8101

Inherited Members

object.Equals(object?), object.Equals(object?, object?), object.GetHashCode(), object.GetType(), object.MemberwiseClone(), object.ReferenceEquals(object?, object?), object.ToString()

Methods

Tween(float, float, float, Action<float>)

Create a new tween.

```
public static object Tween(float from, float to, float duration, Action<float> onChange
```

Parameters

`from` float

The starting value.

`to` float

The end value.

`duration` float

Total tween duration in seconds.

`onChange` Action<float>

A callback that will be invoked every time the tween value changes.

Returns

object

The newly created tween instance.

Tween(int, int, float, Action<int>)

Create a new tween.

```
public static object Tween(int from, int to, float duration, Action<int> onChange)
```

Parameters

`from` int

The starting value.

`to` int

The end value.

`duration` float

Total tween duration in seconds.

`onChange` Action<int>

A callback that will be invoked every time the tween value changes.

Returns

object

The newly created tween instance.

Struct Inheritdoc.Issue8129

Namespace: BuildFromProject
Assembly: BuildFromProject.dll

```
public struct Inheritdoc.Issue8129
```

Inherited Members

object.Equals(object?), object.Equals(object?, object?), object.GetHashCode(), object.GetType(), object.ReferenceEquals(object?, object?), object.ToString()

Constructors

Issue8129(string)

```
public Issue8129(string foo)
```

Parameters

`foo` string

Interface Class1.IIssue8948

Namespace: BuildFromProject
Assembly: BuildFromProject.dll

```
public interface Class1.IIssue8948
```

Methods

DoNothing<T>()

Does nothing with generic type `T`.

```
void DoNothing<T>()
```

Type Parameters

`T`

A generic type.

Interface IInheritdoc

Namespace: BuildFromProject
Assembly: BuildFromProject.dll

```
public interface IInheritdoc
```

Methods

Issue7629()

This method should do something...

```
void Issue7629()
```

Enum Class1.Issue9260

Namespace: BuildFromProject
Assembly: BuildFromProject.dll

```
public enum Class1.Issue9260
```

Fields

`Value = 0`

This is a regular enum value.

This is a remarks section. Very important remarks about Value go here.

`OldAndUnusedValue = 1`

This is old and unused. You shouldn't use it anymore.

Don't use this, seriously! Use Value instead.

`OldAndUnusedValue2 = 2`

This is old and unused. You shouldn't use it anymore.

Don't use this, seriously! Use Value instead.

Namespace BuildFromVBSourceCode

Classes

BaseClass1

This is the BaseClass

Class1

This is summary from vb class...

Class BaseClass1

Namespace: BuildFromVBSourceCode

This is the BaseClass

```
public abstract class BaseClass1
```

Inheritance

object ← BaseClass1

Derived

Class1

Inherited Members

object.Equals(object), object.Equals(object, object), object.Finalize(), object.GetHashCode(), object.GetType(), object.MemberwiseClone(), object.ReferenceEquals(object, object), object.ToString()

Methods

WithDeclarationKeyword(Class1)

```
public abstract DateTime WithDeclarationKeyword(Class1 keyword)
```

Parameters

`keyword` Class1

Returns

DateTime

Class Class1

Namespace: BuildFromVBSourceCode

This is summary from vb class...

```
public class Class1 : BaseClass1
```

Inheritance

object ← BaseClass1 ← Class1

Inherited Members

BaseClass1.WithDeclarationKeyword(Class1), object.Equals(object), object.Equals(object, object), object.Finalize(), object.GetHashCode(), object.GetType(), object.MemberwiseClone(), object.ReferenceEquals(object, object), object.ToString()

Fields

ValueClass

This is a *Value* type

```
public Class1 ValueClass
```

Field Value

Class1

Properties

Keyword

```
[Obsolete("This member is obsolete.", true)]
public Class1 Keyword { get; }
```

Property Value

Class1

Methods

Value(string)

This is a *Function*

```
public int Value(string name)
```

Parameters

`name` string

Name as the **String** value

Returns

int

Returns Ahooo

WithDeclarationKeyword(Class1)

What is **Sub**?

```
public override DateTime WithDeclarationKeyword(Class1 keyword)
```

Parameters

`keyword` Class1

Returns

DateTime

Namespace CatLibrary

Namespaces

CatLibrary.Core

Classes

Cat<T, K>

Here's main class of this *Demo*.

You can see mostly type of article within this class and you for more detail, please see the remarks.

this class is a template class. It has two Generic parameter. they are: T and K.

The extension method of this class can refer to ICatExtension class

CatException<T>

Complex<T, J>

ICatExtension

It's the class that contains ICat interface's extension method.

This class must be **public** and **static**.

Also it shouldn't be a geneic class

Tom

Tom class is only inherit from Object. Not any member inside itself.

TomFromBaseClass

TomFromBaseClass inherits from @

Interfaces

IAnimal

This is **basic** interface of all animal.

ICat

Cat's interface

Delegates

FakeDelegate<T>

Fake delegate

MRefDelegate<K, T, L>

Generic delegate with many constrains.

MRefNormalDelegate

Delegate in the namespace

Namespace CatLibrary.Core

Classes

ContainersRefType.ContainersRefTypeChild

ExplicitLayoutClass

Issue231

Issue231

Structs

ContainersRefType

Struct ContainersRefType

Interfaces

ContainersRefType.ContainersRefTypeChildInterface

Enums

ContainersRefType.ColorType

Enumeration ColorType

Delegates

ContainersRefType.ContainersRefTypeDelegate

Delegate ContainersRefTypeDelegate

Class ContainersRefType.ContainersRefType Child

Namespace: CatLibrary.Core
Assembly: CatLibrary.Core.dll

```
public class ContainersRefType.ContainersRefTypeChild
```

Inheritance

object ← ContainersRefType.ContainersRefTypeChild

Inherited Members

object.Equals(object?), object.Equals(object?, object?), object.GetHashCode(), object.GetType(), object.MemberwiseClone(), object.ReferenceEquals(object?, object?), object.ToString()

Class ExplicitLayoutClass

Namespace: CatLibrary.Core
Assembly: CatLibrary.Core.dll

```
public class ExplicitLayoutClass
```

Inheritance

object ← ExplicitLayoutClass

Inherited Members

object.Equals(object?), object.Equals(object?, object?), object.GetHashCode(), object.GetType(), object.MemberwiseClone(), object.ReferenceEquals(object?, object?), object.ToString()

Class Issue231

Namespace: CatLibrary.Core
Assembly: CatLibrary.dll, CatLibrary.Core.dll

```
public static class Issue231
```

Inheritance

object ← Issue231

Inherited Members

object.Equals(object?), object.Equals(object?, object?), object.GetHashCode(), object.GetType(), object.MemberwiseClone(), object.ReferenceEquals(object?, object?), object.ToString()

Methods

Bar(ContainersRefType)

```
public static void Bar(this ContainersRefType c)
```

Parameters

c ContainersRefType

Foo(ContainersRefType)

```
public static void Foo(this ContainersRefType c)
```

Parameters

c ContainersRefType

Struct ContainersRefType

Namespace: CatLibrary.Core
Assembly: CatLibrary.Core.dll

Struct ContainersRefType

```
public struct ContainersRefType
```

Inherited Members

object.Equals(object?), object.Equals(object?, object?), object.GetHashCode(), object.GetType(), object.ReferenceEquals(object?, object?), object.ToString()

Extension Methods

Issue231.Bar(ContainersRefType), Issue231.Foo(ContainersRefType)

Fields

ColorCount

ColorCount

```
public long ColorCount
```

Field Value

long

Properties

GetColorCount

GetColorCount

```
public long GetColorCount { get; }
```

Property Value

long

Methods

ContainersRefTypeNonRefMethod(params object[])

ContainersRefTypeNonRefMethod

array

```
public static int ContainersRefTypeNonRefMethod(params object[] parmsArray)
```

Parameters

`parmsArray` object[]

Returns

int

ContainersRefTypeEventHandler

```
public event EventHandler ContainersRefTypeEventHandler
```

Event Type

EventHandler

Interface ContainersRefType.ContainersRefTypeChildInterface

Namespace: CatLibrary.Core
Assembly: CatLibrary.Core.dll

```
public interface ContainersRefType.ContainersRefTypeChildInterface
```

Enum ContainersRefType.ColorType

Namespace: CatLibrary.Core
Assembly: CatLibrary.Core.dll

Enumeration ColorType

```
public enum ContainersRefType.ColorType
```

Fields

`Red = 0`

red

`Blue = 1`

blue

`Yellow = 2`

yellow

Delegate ContainersRefType.ContainersRefTypeDelegate

Namespace: CatLibrary.Core
Assembly: CatLibrary.Core.dll

Delegate ContainersRefTypeDelegate

```
public delegate void ContainersRefType.ContainersRefTypeDelegate()
```

Class Cat<T, K>

Namespace: CatLibrary
Assembly: CatLibrary.dll

Here's main class of this *Demo*.

You can see mostly type of article within this class and you for more detail, please see the remarks.

this class is a template class. It has two Generic parameter. they are: `T` and `K`.

The extension method of this class can refer to ICatExtension class

```
[Serializable]
[Obsolete]
public class Cat<T, K> : ICat, IAnimal where T : class, new() where K : struct
```

Type Parameters

`T`

This type should be class and can new instance.

`K`

This type is a struct type, class type can't be used for this parameter.

Inheritance

object ← Cat<T, K>

Implements

ICat, IAnimal

Inherited Members

object.Equals(object?), object.Equals(object?, object?), object.GetHashCode(), object.GetType(), object.MemberwiseClone(), object.ReferenceEquals(object?, object?), object.ToString()

Extension Methods

ICatExtension.Play(ICat, ContainersRefType.ColorType), ICatExtension.Sleep(ICat, long)

Examples

Here's example of how to create an instance of this class. As T is limited with `class` and K is limited with `struct`.

```
var a = new Cat(object, int)();
int catNumber = new int();
unsafe
{
    a.GetFeetLength(catNumber);
}
```

As you see, here we bring in **pointer** so we need to add `unsafe` keyword.

Remarks

Here's all the content you can see in this class.

Constructors

Cat()

Default constructor.

```
public Cat()
```

Cat(T)

Constructor with one generic parameter.

```
public Cat(T ownType)
```

Parameters

`ownType` T

This parameter type defined by class.

Cat(string, out int, string, bool)

It's a complex constructor. The parameter will have some attributes.

```
public Cat(string nickName, out int age, string realName, bool isHealthy)
```

Parameters

`nickName` string

it's string type.

`age` int

It's an out and ref parameter.

`realName` string

It's an out paramter.

`isHealthy` bool

It's an in parameter.

Fields

isHealthy

Field with attribute.

```
[ContextStatic]
[NonSerialized]
[Obsolete]
public bool isHealthy
```

Field Value

bool

Properties

Age

Hint cat's age.

```
[Obsolete]
protected int Age { get; set; }
```

Property Value

int

Name

Ell property.

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

Property Value

string

this[string]

This is index property of Cat. You can see that the visibility is different between `get` and `set` method.

```
public int this[string a] { protected get; set; }
```

Property Value

int

Methods

CalculateFood(DateTime)

It's a method with complex return type.

```
public Dictionary<string, List<int>> CalculateFood(DateTime date)
```

Parameters

`date` DateTime

Date time to now.

Returns

Dictionary<string, List<int>>

It's a relationship map of different kind food.

Equals(object)

Override the method of `Object.Equals(object obj).`

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

Parameters

`obj` object

Can pass any class type.

Returns

bool

The return value tell you whehter the compare operation is successful.

GetTailLength(int*, params object[])

It's an `unsafe` method. As you see, `catName` is a **pointer**, so we need to add `unsafe` keyword.

```
public long GetTailLength(int* catName, params object[] parameters)
```

Parameters

`catName` int*

Thie represent for cat name length.

`parameters` object[]

Optional parameters.

Returns

long

Return cat tail's length.

Jump(T, K, ref bool)

This method have attribute above it.

```
[Conditional("Debug")]
public void Jump(T ownType, K anotherOwnType, ref bool cheat)
```

Parameters

`ownType` T

Type come from class define.

`anotherOwnType` K

Type come from class define.

`cheat` bool

Hint whether this cat has cheat mode.

Exceptions

ArgumentException

This is an argument exception

ownEat

Eat event of this cat

```
[Obsolete("This _event handler_ is deprecated.")]
public event EventHandler ownEat
```

Event Type

EventHandler

Operators

operator +(Cat<T, K>, int)

Addition operator of this class.

```
public static int operator +(Cat<T, K> lsr, int rsr)
```

Parameters

`lsr` Cat<T, K>

..

`rsr` int

~~

Returns

int

Result with *int* type.

explicit operator Tom(Cat<T, K>)

Expilicit operator of this class.

It means this cat can evolve to change to Tom. Tom and Jerry.

```
public static explicit operator Tom(Cat<T, K> src)
```

Parameters

`src` Cat<T, K>

Instance of this class.

Returns

Tom

Advanced class type of cat.

operator -(Cat<T, K>, int)

Similar with operaotr +, refer to that topic.

```
public static int operator -(Cat<T, K> lsr, int rsr)
```

Parameters

`lsr` Cat<T, K>

`rsr` int

Returns

int

Class CatException<T>

Namespace: CatLibrary
Assembly: CatLibrary.dll

```
public class CatException<T> : Exception, ISerializable
```

Type Parameters

T

Inheritance

object ← Exception ← CatException<T>

Implements

ISerializable

Inherited Members

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

Class Complex<T, J>

Namespace: CatLibrary
Assembly: CatLibrary.dll

```
public class Complex<T, J>
```

Type Parameters

`T`

`J`

Inheritance

object ← Complex<T, J>

Inherited Members

object.Equals(object?), object.Equals(object?, object?), object.GetHashCode(), object.GetType(), object.MemberwiseClone(), object.ReferenceEquals(object?, object?), object.ToString()

Class ICatExtension

Namespace: CatLibrary
Assembly: CatLibrary.dll

It's the class that contains ICat interface's extension method.

This class must be **public** and **static**.

Also it shouldn't be a geneic class

```
public static class ICatExtension
```

Inheritance

object ← ICatExtension

Inherited Members

object.Equals(object?), object.Equals(object?, object?), object.GetHashCode(), object.GetType(), object.MemberwiseClone(), object.ReferenceEquals(object?, object?), object.ToString()

Methods

Play(ICat, ColorType)

Extension method to let cat play

```
public static void Play(this ICat icat, ContainersRefType.ColorType toy)
```

Parameters

`icat` ICat

Cat

`toy` ContainersRefType.ColorType

Something to play

Sleep(ICat, long)

Extension method hint that how long the cat can sleep.

```
public static void Sleep(this ICat icat, long hours)
```

Parameters

`icat` ICat

The type will be extended.

`hours` long

The length of sleep.

Class Tom

Namespace: CatLibrary
Assembly: CatLibrary.dll

Tom class is only inherit from Object. Not any member inside itself.

```
public class Tom
```

Inheritance

object ← Tom

Derived

TomFromBaseClass

Inherited Members

object.Equals(object?), object.Equals(object?, object?), object.GetHashCode(), object.GetType(), object.MemberwiseClone(), object.ReferenceEquals(object?, object?), object.ToString()

Methods

TomMethod(Complex<TomFromBaseClass, TomFromBaseClass>, Tuple<string, Tom>)

This is a Tom Method with complex type as return

```
public Complex<string, TomFromBaseClass> TomMethod(Complex<TomFromBaseClass, TomFromBas
```

Parameters

a Complex<TomFromBaseClass, TomFromBaseClass>

A complex input

b Tuple<string, Tom>

Another complex input

Returns

Complex<string, TomFromBaseClass>

Complex TomFromBaseClass

Exceptions

NotImplementedException

This is not implemented

ArgumentException

This is the exception to be thrown when implemented

CatException<T>

This is the exception in current documentation

Class TomFromBaseClass

Namespace: CatLibrary
Assembly: CatLibrary.dll

TomFromBaseClass inherits from @

```
public class TomFromBaseClass : Tom
```

Inheritance

object ← Tom ← TomFromBaseClass

Inherited Members

Tom.TomMethod(Complex<TomFromBaseClass, TomFromBaseClass>, Tuple<string, Tom>), object.Equals(object?), object.Equals(object?, object?), object.GetHashCode(), object.GetType(), object.MemberwiseClone(), object.ReferenceEquals(object?, object?), object.ToString()

Constructors

TomFromBaseClass(int)

This is a #ctor with parameter

```
public TomFromBaseClass(int k)
```

Parameters

`k` int

Interface IAnimal

Namespace: CatLibrary
Assembly: CatLibrary.dll

This is **basic** interface of all animal.

```
public interface IAnimal
```

Properties

Name

Name of Animal.

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

Property Value

string

this[int]

Return specific number animal's name.

```
string this[int index] { get; }
```

Property Value

string

Methods

Eat()

Animal's eat method.

```
void Eat()
```

Eat<Tool>(Tool)

Overload method of eat. This define the animal eat by which tool.

```
void Eat<Tool>(Tool tool) where Tool : class
```

Parameters

`tool` Tool

Tool name.

Type Parameters

`Tool`

It's a class type.

Eat(string)

Feed the animal with some food

```
void Eat(string food)
```

Parameters

`food` string

Food to eat

Interface ICat

Namespace: CatLibrary
Assembly: CatLibrary.dll

Cat's interface

```
public interface ICat : IAnimal
```

Implements

IAnimal

Extension Methods

ICatExtension.Play(ICat, ContainersRefType.ColorType), ICatExtension.Sleep(ICat, long)

eat

eat event of cat. Every cat must implement this event.

```
event EventHandler eat
```

Event Type

EventHandler

Delegate FakeDelegate<T>

Namespace: CatLibrary
Assembly: CatLibrary.dll

Fake delegate

```
public delegate int FakeDelegate<T>(long num, string name, params object[] scores)
```

Parameters

`num` long

Fake para

`name` string

Fake para

`scores` object[]

Optional Parameter.

Returns

int

Return a fake number to confuse you.

Type Parameters

`T`

Fake para

Delegate MRefDelegate<K, T, L>

Namespace: CatLibrary
Assembly: CatLibrary.dll

Generic delegate with many constrains.

```
public delegate void MRefDelegate<K, T, L>(K k, T t, L l) where K : class, IComparable
```

Parameters

`k` K

Type K.

`t` T

Type T.

`l` L

Type L.

Type Parameters

`K`

Generic K.

`T`

Generic T.

`L`

Generic L.

Delegate MRefNormalDelegate

Namespace: CatLibrary
Assembly: CatLibrary.dll

Delegate in the namespace

```
public delegate void MRefNormalDelegate(List<string> pics, out string name)
```

Parameters

`pics` List<string>

a name list of pictures.

`name` string

give out the needed name.

Namespace MRef

Namespaces

MRef.Demo

Namespace MRef.Demo

Namespaces

MRef.Demo.Enumeration

Namespace MRef.Demo.Enumeration

Enums

ColorType

Enumeration ColorType

Enum ColorType

Namespace: MRef.Demo.Enumeration
Assembly: CatLibrary.dll

Enumeration ColorType

```
public enum ColorType
```

Fields

`Red = 0`

this color is red

`Blue = 1`

blue like river

`Yellow = 2`

yellow comes from desert

Remarks

Red/Blue/Yellow can become all color you want.

See Also

object