Table of Contents

Namespace SampleDocApi

Classes

ClassTest

Here is a text class, to start off generation of some docs!

RecordTest

This is test Record type

Structs

StructTest

This is a test Struct

Interfaces

IDrives

This is an interface for something that can drive

IInterfaceTest

Here is an example of a basic interface

IRuns

This is an interface for something that can run

Enums

EnumTest

Here is a test enum to start be consumed in the ClassTest

Class ClassTest

Here is a text class, to start off generation of some docs!

Inheritance

↳ System.Object
  ↳ ClassTest

Implements

IInterfaceTest
IRuns
IDrives

Inherited Members

System.Object.ToString()
System.Object.Equals(System.Object)
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetHashCode()
System.Object.GetType()
System.Object.MemberwiseClone()

Namespace: SampleDocApi

Assembly: SampleDocApi.dll

Syntax

```
public class ClassTest : IInterfaceTest, IRuns, IDrives
```

Remarks

You can include all sorts of **markdown documentation** here

Constructors

ClassTest(String, DateTime, EnumTest)

This is a constructor for this TestClass. Use it to initiate an instance of this class!

Declaration

```
public ClassTest(string testString, DateTime testDateTime, EnumTest testEnum)
```

Parameters

Type	Name	Description
System.String	*testString*	This is a string, you can use it for alphanumeric content
System.DateTime	*testDateTime*	Here is a datetime, to tell you when something...
EnumTest	*testEnum*	Consumes the EnumTest enum

Examples

Here is the incredibly complicated way of using this constructor

```
var testClass = new TestClass("a string", new DateTime(), EnumTest.D);
```

Fields

TestTuple

A random tuple value type

Declaration

```
public (double Sum, int Count) TestTuple
```

Field Value

Type	Description
System.ValueTuple<System.Double, System.Int32>	

Properties

ParentString

Something inherited from IInterfaceTest

Declaration

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

Property Value

Type	Description
System.String	

TestEnum

Consumes the EnumTest enum. Can be changed at any time...

Declaration

```
public EnumTest TestEnum { get; set; }
```

Property Value

Type	Description
EnumTest	

Methods

DoSomethingWithInputs()

Here's a public method consuming some private properties

Declaration

```
public string DoSomethingWithInputs()
```

Returns

Type	Description
System.String	Returns a nice happy string

ReturnInteger(Int32, Int32)

Something else inherited from IInterfaceTest interface

Declaration

```
public int ReturnInteger(int firstInt, int secondInt)
```

Parameters

Type	Name	Description
System.Int32	*firstInt*	An int input
System.Int32	*secondInt*	Another int input

Returns

Type	Description
System.Int32	An integer, when it's implemented, which it isn't yet

Exceptions

Type	Condition
System.NotImplementedException	Throws exception at present because it isn't implement! :O

Implements

IInterfaceTest
IRuns
IDrives

Enum EnumTest

Here is a test enum to start be consumed in the ClassTest

Namespace: SampleDocApi

Assembly: SampleDocApi.dll

Syntax

```
public enum EnumTest
```

Remarks

Avoid usage of this enum. It likely will be deprecated as of v4.0

Fields

Name	Description
A	First Letter of the alphabet
B	Second Letter of the alphabet
C	Third Letter of the alphabet
D	Fourth Letter of the alphabet
E	Fifth Letter of the alphabet
F	Sixth Letter of the alphabet
G	Seventh Letter of the alphabet!

Interface IDrives

This is an interface for something that can drive

Namespace: SampleDocApi

Assembly: SampleDocApi.dll

Syntax

```
public interface IDrives
```

Interface IInterfaceTest

Here is an example of a basic interface

Namespace: SampleDocApi

Assembly: SampleDocApi.dll

Syntax

```
public interface IInterfaceTest
```

Properties

ParentString

Here is a string that must be implemented in inheriting classes

Declaration

```
string ParentString { get; set; }
```

Property Value

Type	Description
System.String	

Methods

ReturnInteger(Int32, Int32)

Do something which returns an integer

Declaration

```
int ReturnInteger(int firstInt, int secondInt)
```

Parameters

Type	Name	Description
System.Int32	*firstInt*	This is the first number to input
System.Int32	*secondInt*	This is the second number to input

Returns

Type	Description
System.Int32	Returns a number which is the function of the two inputs

Interface IRuns

This is an interface for something that can run

Namespace: SampleDocApi

Assembly: SampleDocApi.dll

Syntax

```
public interface IRuns
```

Class RecordTest

This is test Record type

Inheritance

↳ System.Object
  ↳ RecordTest

Implements

System.IEquatable<RecordTest>

Inherited Members

System.Object.ToString()
System.Object.Equals(System.Object)
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetHashCode()
System.Object.GetType()
System.Object.MemberwiseClone()

Namespace: SampleDocApi

Assembly: SampleDocApi.dll

Syntax

```
public class RecordTest : IEquatable
```

Properties

FirstName

What is your first name?

Declaration

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

Property Value

Type	Description
System.String	

LastName

What is your last name (surname, family name)

Declaration

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

Property Value

Type	Description
System.String	

Implements

System.IEquatable<SampleDocApi.RecordTest>

Struct StructTest

This is a test Struct

Inherited Members

System.ValueType.Equals(System.Object)
System.ValueType.GetHashCode()
System.ValueType.ToString()
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetType()

Namespace: SampleDocApi

Assembly: SampleDocApi.dll

Syntax

```
public readonly struct StructTest
```

Constructors

StructTest(Boolean, String)

Constructor for the Struct to set the two values

Declaration

```
public StructTest(bool boolValue, string stringValue)
```

Parameters

Type	Name	Description
System.Boolean	*boolValue*	Either a true or a false
System.String	*stringValue*	Something with letters and numbers, maybe even some symbols

Properties

BoolValue

Either a true or a false

Declaration

```
public readonly bool BoolValue { get; }
```

Property Value

Type	Description
System.Boolean	

StringValue

Something with letters and numbers, maybe even some symbols

Declaration

```
public readonly string StringValue { get; }
```

Property Value

Type	Description
System.String	

Namespace SampleDocApi.Models

Classes

Contact

Represents a Person

Class Contact

Represents a Person

Inheritance

↳ System.Object
  ↳ Contact

Inherited Members

System.Object.ToString()
System.Object.Equals(System.Object)
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetHashCode()
System.Object.GetType()
System.Object.MemberwiseClone()

Namespace: SampleDocApi.Models

Assembly: SampleDocApi.dll

Syntax

```
public class Contact
```

Properties

DateOfBirth

Date the person was born

Declaration

```
public DateTime DateOfBirth { get; set; }
```

Property Value

Type	Description
System.DateTime	

FirstName

Person's first name

Declaration

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

Property Value

Type	Description
System.String	

LastName

Person's last name (aka surname or family name)

Declaration

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

Property Value

Type	Description
System.String	

Namespace SampleDocApi.Tests

Classes

SomeBasicTests

These are some extremely basic tests

Class SomeBasicTests

These are some extremely basic tests

Inheritance

↳ System.Object
  ↳ SomeBasicTests

Inherited Members

System.Object.ToString()
System.Object.Equals(System.Object)
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetHashCode()
System.Object.GetType()
System.Object.MemberwiseClone()

Namespace: SampleDocApi.Tests

Assembly: SampleDocApi.Tests.dll

Syntax

```
[TestFixture]
public class SomeBasicTests
```

Methods

Input_Booleans_Should_Assert_Their_Truthiness(Boolean, Boolean)

Given that a boolean value is input
And an identical boolean value is expected
When tested for equality
They should be equal

Declaration

```
[Test]
[TestCase(true, true, Description = "Test for trues")]
[TestCase(false, false, Description = "Test for falses")]
[TestCase(null, null, Description = "Test for nulls")]
public void Input_Booleans_Should_Assert_Their_Truthiness(bool inputValue, bool expectedValue)
```

Parameters

Type	Name	Description
System.Boolean	*inputValue*	True or false
System.Boolean	*expectedValue*	The same value as the *inputValue*

True_Should__Equal_True()

Given that something is true
When it is asserted that it's true
It should be true

Declaration

```
[Test]
public void True_Should__Equal_True()
```