Default value for user defined class in C#

Asked 9 years, 8 months ago Active 3 years ago Viewed 57k times

52

3

I see some code will return the default value, so I am wondering for a user defined class, how will the compiler define its default value?

c# default-value

edited Dec 11 '10 at 23:58
Peter Mortensen
14.4k 19 88 117

asked Dec 17 '09 at 7:53
Benny
4,672 6 48 84

2 Just to clarify: all classes (including the ones you create) will default to `null` . Number value types will default to zero and structs are implementation defined (values are set in the constructor). – Isak Savo Dec 17 '09 at 7:57

3 Actually, structs have an implicit default constructor that can't be overriden which zero's out the memory used by the struct, so the default value is still compiler defined, not implementation defined. – Matthew Scharley Dec 17 '09 at 8:03

1 Important distinction: *Class members* are initialized with their default value. Uninitialised local function variables just give compiler errors ;) – Nyerguds Jun 28 '16 at 11:03

8 Answers

60

✓

To chime in with the rest, it will be `null` , but I should also add that you can get the default value of any type, using `default`

```
default(MyClass) // null
default(int) // 0
```

It can be especially useful when working with generics; you might want to return `default(T)` , if your return type is `T` and you don't want to assume that it's nullable.

110k 27 187 208

3 +1 There is a full explanation of the default keyword at msdn.microsoft.com/en-us/library/xwth0h0d%28VS.80%29.aspx – Joe Daley Dec 17 '09 at 7:59

14

The default value for `class` is a `null`

answered Dec 17 '09 at 7:55

Graviton

43.4k 125 376 553

11

Note: A DefaultValueAttribute will not cause a member to be automatically initialized with the attribute's value. You must set the initial value in your code.

You can decorate your properties with the DefaultValueAttribute.

```
private bool myVal=false;

[DefaultValue(false)]
 public bool MyProperty {
    get {
       return myVal;
    }
    set {
       myVal=value;
    }
 }
```

I know this doesn't answer your question, just wanted to add this as relevant information.

For more info see http://msdn.microsoft.com/en-us/library/system.componentmodel.defaultvalueattribute.aspx

edited Apr 27 '15 at 6:41

answered Dec 17 '09 at 7:58

Peter

9,353 14 62 103

of how it should be initialized. – Nyerguds Jun 28 '16 at 10:59

7

The default value for classes is `null` . For structures, the default value is the same as you get when you instantiate the default parameterless constructor of the structure (which can't be overriden by the way). The same rule is applied recursively to all the fields contained inside the class or structure.

answered Dec 17 '09 at 8:00

Konamiman
44k 15 101 128

3

If it is a reference type, the default value will be `null` , if it is a value type, then it depends.

answered Dec 17 '09 at 7:55

Darin Dimitrov
874k 233 3068 2786

3

I would make this "default" class instance a field rather than property, like how `System.String.Empty` looks:

```
public class Person
{
    public string Name { get; set; }
    public string Address { get; set; }

    public static readonly Person Default = new Person()
    {
        Name = "Some Name",
        Address = "Some Address"
    };
}

...

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

```
        Person person = Person.Default;

        //the rest of your code
    }
```

answered Aug 25 '16 at 7:25

Yom S.

2,971 2 11 22

2

```
Assert.IsTrue(default(MyClass) == null);
```

answered Dec 17 '09 at 7:58

Bryan Watts

36.3k 15 75 84

1 what is the purpose of this assert? – |--''''''---------'''''''''''' Mar 4 '16 at 16:45

1

If you are looking to somehow define a non-null default value for a reference type, realize that it wouldn't make sense for a reference type to have default value. At some point a reference would need to point to the allocated memory for the constructed object. To make it possible, I imagine two implementations:

1. Define a static instance as the default: this could work but would probably require a really clunky way to somehow identify to the compiler where to get the static instance. Not really a feature worth putting into the language.
2. Allow calling the default constructor every time the default value occurred in an expression: this would be terrible because now you have to worry about two default values possibly being unequal because you didn't override Equals to not compare references, and also terrible because of the memory management and memory allocation performance hit, and side affects from the constructor.

So in summary I think it is a language feature that we really don't want. And default(MyClass) will always == null.

answered Apr 3 '15 at 15:01

AndyClaw

600 5 11