Mutable And Immutable Class In C#

Mutable and immutable are English words that mean "can change" and "cannot change" respectively. The meaning of these words is the same in C# programming language; that means the mutable types are those whose data members can be changed after the instance is created but Immutable types are those whose data members can not be changed after the instance is created.

When we change the value of mutable objects, value is changed in same memory. But in immutable type, the new memory is created and the modified value is stored in new memory.

Let’s start with examples of predefined classes (String and StringBuilder).

String

Strings are immutable, which means we are creating new memory everytime instead of working on existing memory.

So, whenever we are modifying a value of the existing string, i.e., we are creating a new object which refers to that modified string and the old one becomes unreferenced. Hence, if we are modifying the existing string continuously, then numbers of the application performance will be decreased.

Example

```
            for (int i = 0; i < 1000; i++)
            {
                str += "Modified ";
            }
```

In the code given above, string str will update 1000 times inside the loop and every time it will create new instance so all old values will be collected by garbage collector after some time.

It is not a good approach for this solution so, it’s better to go for mutable type. So in C#, we have StringBuilder which is a mutable type. We have some advantages of immutable classes like immutable objects are simpler to construct, test, and use. immutable objects are always thread-safe and etc.

StringBuilder

StringBuilder is a mutable type, that means we are using the same memory location and keep on appending/modifying the stuff to one instance. It will not create any further instances hence it will not decrease the performance of the application.

Example

```
            StringBuilder strB = new StringBuilder();

            for (int i = 0; i < 10000; i++)
            {
                strB.Append("Modified ");
            }
```

In the code given above, It has the huge impact of allocated memory because it will not create instance each time.

We have many methods for modifying the StringBuilder String as below,

- Insert - Inserts a string or object into the specified index of the current StringBuilder.
- Remove - Removes a specified number of characters from the current StringBuilder.

Creating an Immutable Class

For creating an immutable class, we have to think about their properties or variables which will never change the value(s) after assigning the first time. So, we are going to follow some simple steps for creating the immutable class in C#.

Step 1

Make the variables read-only so we can not modify the variable after assigning the first time.

```
class MyClass
    {
        private readonly string myStr;
}
```

Step 2

Use parameterized constructor for assigning the myStr value for the class while creating the object of the class as below

```
class MyClass
    {
        private readonly string myStr;

        public MyClass(string str)
        {
            myStr = str;
        }
    }
```

Step 3

Use properties for getting the variables of the class and remove the setters of the property, use only getters as below.

```
        private readonly string myStr;

        public MyClass(string str)
        }

        public string GetStr
        {
            get { return myStr; }
        }
    }
```

In this easy way, we can create the immutable class in C#.

It is purely based on our project requirement which says it is better to use Mutable or Immutable class.

Next Recommended Article
String is Immutable in C#

Mutable And Immutable Class In C#

Vivek Kumar TOP 500
Microsoft Certified Professional | C# Corner MVP | Dzone MVB | Author | Speaker
https://vivekkumar.com

177 2.3m 2

19 12

C#Corner ecome a member Login

Type your comment here and press Enter Key (Minimum 10 characters)

Post ▼ Ask Question

Dharmendra Kumar
• 1282 • 636 • 3.4k

Feb 08, 2020
0 0 Reply

Then what is the difference between immutable class and a singleton class?

amit jain
• 1888 • 6 • 0

Jul 18, 2019
0 1 Reply

In Singleton we can still update properties if they are public

Ruchi Jaiswal
• 1884 • 10 • 0

Jan 31, 2020
0

It took me a while to figure advantage use immutable over mutable. Thread-safe is one of you noticed. Thank you!

Adam Schachabov
• 1817 • 77 • 0

Apr 25, 2019
0 0 Reply

Nice read... Thanks for sharing.

Rohanjoy Bhattacharya
• 1664 • 230 • 59.9k

Mar 31, 2017
2 1 Reply

Thank you Rohanjoy :)

Vivek Kumar
• 177 • 11.4k • 2.3m

Mar 31, 2017
0

Thanks for sharing sir ......

Manav Pandya
• 99 • 19.9k • 1.2m

Mar 30, 2017
2 1 Reply

Thank you Manav ...

Vivek Kumar
• 177 • 11.4k • 2.3m

Mar 30, 2017
0

• 1819 • 75 • 0

1 Reply

Thank you Rajesh for your valuable comment :)

• 177 • 11.4k • 2.3m

0

The below line is confusing me.."When we change the value of mutable objects, the new memory is created and the new value is stored in new memory. But in immutable type, value is changed in same memory."

Rajesh Kadam

Mar 30, 2017

• 1819 • 75 • 0

1 1 Reply

Thank you Rajesh kadam for your time. It was typo. I updated it.

Vivek Kumar

Mar 30, 2017

• 177 • 11.4k • 2.3m

0

C#Corner

ecome a member Login

Post ▼ Ask Question

FEATURED ARTICLES

Handling Errors in TypeScript

Recognizing Faces By Capturing them Using Python

Insert Data Into Azure Table Storage Using ASP.NET Core Application

TRENDING UP

01 Agile Or Scrum - Which One To Choose And Why?

02 Introduction to Network Virtualization

03 What is Docker?

04 Build A FAQ Chatbot With Power Virtual Agents

05 Build a Microsoft Bot Application With Power Virtual Agents

06 What is Software-Defined Networking and Virtual Networks in Physical Networks

07 Customize the Bot Conversation Message With Power Virtual Agent

09 Insert Data Into Azure Table Storage Using ASP.NET Core Application

10 Search Data Between Two Dates Using Web API And Angular 8