C# Pass By Value with Examples

In c#, Passing a Value-Type (/tutorial/csharp/csharp-value-type-and-reference-type-with-examples) parameter to a method by value means passing a copy of the variable to the method. So the changes made to the parameter inside the called method will not affect the original data stored in the argument variable.

As discussed earlier, Value-Type (/tutorial/csharp/csharp-value-type-and-reference-type-with-examples) variables will contain the value directly on its memory, and Reference-Type (/tutorial/csharp/csharp-value-type-and-reference-type-with-examples) variables will contain a reference of its data.

C# Passing Parameters By Value Example

Following is the example of passing a value type (/tutorial/csharp/csharp-value-type-and-reference-type-with-examples) parameter to a method by value in the c# programming language.
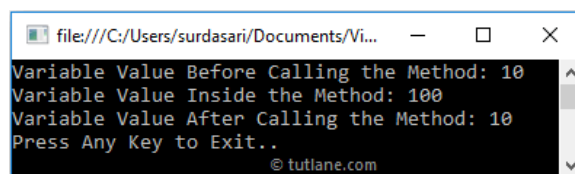
```
using System;

namespace Tutlane
{
    class Program
    {
        static void Main(string[] args)
        {
            int x = 10;
            Console.WriteLine("Variable Value Before Calling the Method: {0}", x);
            Multiplication(x);
            Console.WriteLine("Variable Value After Calling the Method: {0}", x);
            Console.WriteLine("Press Enter Key to Exit..");
            Console.ReadLine();
        }
        public static void Multiplication(int a)
        {
            a *= a;
            Console.WriteLine("Variable Value Inside the Method: {0}", a);
        }
    }
}
```

If you observe the above example, the variable **x** is a value type (/tutorial/csharp/csharp-value-type-and-reference-type-with-examples), and it is passed to the **Multiplication** method. The content of variable **x** copied to the parameter **a** and made required modifications in the **Multiplication** method. Still, the changes made inside the method do not affect the original value of the variable.

Output of C# Passing Parameters By Value Example

When we execute the above c# program, we will get the result below.

If you observe the above result, the variable value has not changed even after we made the modifications in our method.

This is how we can pass parameters to the method by value in the c# programming language based on our requirements.