tutlane (/)

C# Encapsulation with Examples

In c#, **Encapsulation** is a process of binding the data members (/tutorial/csharp/csharp-variables-with-examples) and member functions (/tutorial/csharp/csharp-methods-functions-with-examples) into a single unit. In c#, the class is the real-time example for encapsulation because it will combine various types of data members (/tutorial/csharp/csharp-variables-with-examples) and member functions (/tutorial/csharp/csharp-methods-functions-with-examples) into a single unit.

Generally, in c# the encapsulation is used to prevent alteration of code (data) accidentally from the outside of functions (/tutorial/csharp/csharp-methods-functions-with-examples). In c#, by defining the class fields with properties (/tutorial/csharp/csharp-properties-get-set) we can protect the data from accidental corruption.

If we define class fields with properties (/tutorial/csharp/csharp-properties-get-set), then the encapsulated class (/tutorial/csharp/csharp-classes-and-objects-with-examples) won't allow us to access the fields directly instead, we need to use getter and setter functions (/tutorial/csharp/csharp-methods-functions-with-examples) to read or write data based on our requirements.

Following is the example of defining an **encapsulation** class (/tutorial/csharp/csharp-classes-and-objects-with-examples) using properties (/tutorial/csharp/csharp-properties-get-set) with **get** and **set** accessors.

```
class User
    {
        private string location;
        private string name;
        public string Location
        {
            get
            {
                return location;
            }
            set
            {
                location = value;
            }
        }
        public string Name
        {
            get
            {
                return name;
            }
            set
            {
                name = value;
            }
        }
    }
```

If you observe the above code, we defined variables (/tutorial/csharp/csharp-variables-with-examples) with private access modifiers (/tutorial/csharp/csharp-access-modifiers-public-private-protected-internal#divcspvtm) and exposing those variables in a public way by using properties (/tutorial/csharp/csharp-properties-get-set) **get** and **set** accessors. In case, if you want to make any modifications to the defined variables, then we can

make it by using properties (/tutorial/csharp/csharp-properties-get-set) with **get** and **set** accessors.

C# Encapsulation Example

Following is the example of defining an encapsulated class in c# programming language.

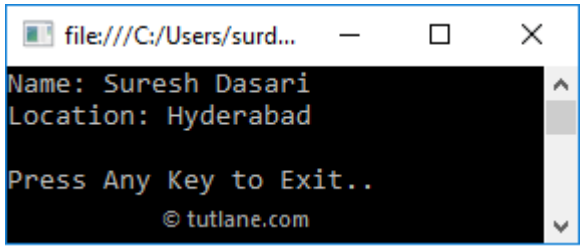
```
using System;
using System.Text;

namespace Tutlane
{
    class User
    {
        private string location;
        private string name;
        public string Location
        {
            get
            {
                return location;
            }
            set
            {
                location = value;
            }
        }
        public string Name
        {
            get
            {
                return name;
            }
            set
            {
```

```
                name = value;
            }
        }
    }
    class Program
    {
        static void Main(string[] args)
        {
            User u = new User();
            // set accessor will invoke
            u.Name = "Suresh Dasari";
            // set accessor will invoke
            u.Location = "Hyderabad";
            // get accessor will invoke
            Console.WriteLine("Name: " + u.Name);
            // get accessor will invoke
            Console.WriteLine("Location: " + u.Location);
            Console.WriteLine("\nPress Enter Key to Exit..");
            Console.ReadLine();
        }
    }
}
```

If you observe above example, we defined a fields (/tutorial/csharp/csharp-variables-with-examples) in encapsulated class (/tutorial/csharp/csharp-classes-and-objects-with-examples) using properties (/tutorial/csharp/csharp-properties-get-set) and we are able to manipulate field (/tutorial/csharp/csharp-variables-with-examples) values using get and set accessors of properties (/tutorial/csharp/csharp-properties-get-set).

When you execute the above c# program, you will get the result as shown below.



This is how you can use **encapsulation** in c# programming language to bind data members and member functions into a single unit by protecting the data from accidental corruption.

CONTACT US

Address: No.1-93, Pochamma Colony, Manikonda, Hyderabad, Telangana - 500089

Email: support@tutlane.com (mailto:support@tutlane.com)