Simon Dunkley 1628623

Pass Task 2 - Counter

Related Learning Outcomes

ULO1 – Explain the OO Principles

This exercise demonstrated object encapsulation teaching me about how you can make features either public or private to only exist within the object. C# has a feature called properties to provide access to data within an object. Making an object static makes it a feature of the mainClass.

ULO2 – Use OO Language and Library

Demonstrated class and constructor declaration, the use of conditional statements (e.g. “if”), and assigning values to parameters. The task examines how fields can be used by an object to remember information. We used a Property to get the name and value from the counter and be able to access it outside the object.

ULO3 – Design, Develop and Test using an IDE

The code was developed using Xamarin Studio to build and run the program, as well as integrated debugging features to step and inspect values.

ULO4 – Communicate using UML Diagrams

I learned how to interpret a UML class diagram and write the related code.

ULO5 – Describe Elements of Good OO Design

The exercise demonstrated correct use of C# coding conventions.

Discussion

Encapsulation means to hide aspects within the object (capsule)
You can either make properties, methods and fields public or private so they could be hidden outside the Object but properties allows you to access the fields outside the class. Most Methods are public so you can call them outside the class. In C# you use properties that allow you access to the data from outside the Object (capsule) without actually giving them access to the fields themselves.
2 Counter objects were created in the task.
The relationship between the variables and objects is that the variables are inside of the object and “Counter 1” is referred as Name within the myCounter object.

When myCounter[2and0] was reset the object’s value called Count was set to 0 because they refer to the same object.

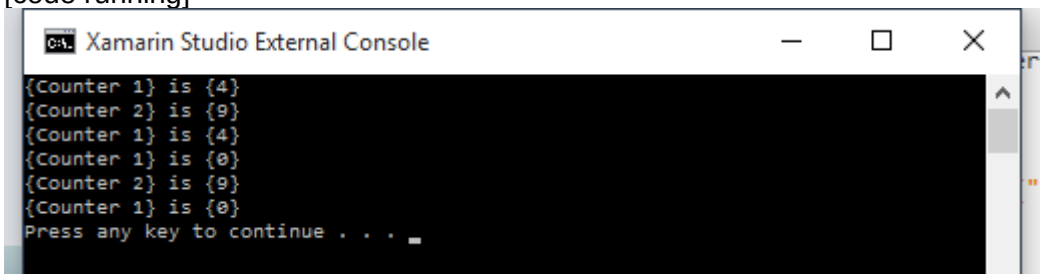
Example

```
public string Name
        {
            get
            {
```

```
                return _name;
            }
            set
            {
                _name = value;
            }
        }
```

Screenshots

[code running]



[use of IDE]