Glen Kelley

10/25/2024

Prove: Articulate – Encapsulation

Encapsulation refers to containing the variables and behaviors of a class in such a way that they are not affected by any other code can see or affect them directly. The difference between encapsulation and abstraction is that in addition to hiding how a piece of code performs its logic, encapsulation ensures that the variables and logic used by that code can not be accessed from outside of the class or other piece of encapsulated code.

Some of the primary benefits of encapsulation during programming and code design is that the encapsulated code will not be altered unnecessarily by outside code. Additionally, so long as the code is created properly, it is easier to locate errors caused by interactions between difference pieces of code. An example can be found in the scripture and reference classes created this week.

public class Reference

{

    private string \_book;

    private int \_chapter;

    private int \_verse;

    private int \_endVerse;

    public string GetReference()

    {

        string reference;

        if (\_endVerse == 0)

        {

            reference = $"{\_book} {\_chapter}:{\_verse}";

        }

        else

        {

            reference = $"{\_book} {\_chapter}:{\_verse}-{\_endVerse}";

        }

        return reference;

    }

}

public class Scripture

{

    private Reference \_reference;

    private List<Word> \_verse = new List<Word>();

    public void Memorize()

    {

        while (true)

        {

            Console.Clear();

            Console.Write(\_reference.GetReference() + " ");

            foreach(Word word in \_verse)

            {

                if(!word.IsHidden())

                {

                    Console.Write(word.Display() + " ");

                }

                else

                {

                    Console.Write("\_\_\_" + " ");

                }

            }

            if (IsCompletelyHidden())

            {

                break;

            }

            HideWords();

            Console.Write("Press enter to continue, type 'quit' to finish or 'restart' to show verse. ->");

            String answer = Console.ReadLine();

            if (answer == "quit")

            {

                break;

            }

            if (answer == "restart")

            {

                ShowWords();

            }

        }

    }

}

While designing the memorize function, it needed to call upon a contained function from the reference class. When running the program, it could be seen that the format of the reference was not aligned correctly. Because that was completely encapsulated within the reference class, it was easy to locate and correct the error.