Encapsulation Articulate Activity

Encapsulation means putting all of one class of code into a sort of “capsule” with a shell around it. Whatever is inside the encapsulated code cannot be accessed from other files or parts of the program unless “public” is specified. Specifically attributes inside should be private, or inaccessible from anything outside the class. This is to protect their values.

A benefit of encapsulation is that certain functions never need to be accessed outside the class, so they stay hidden from other parts of the program. Attributes of a class also stay hidden from other parts of the program so there are no name complications or mis-calls.

An example of encapsulation is the Console class. I can use the console class and its functions but there is much going on behind the scenes that I never have to worry about because I can only call its functions that are useful to me. Many if not all of the methods and attributes in Console are private.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

class Word

{

    private string \_text;

    private bool \_isHidden;

    public Word(string text) // Constructor to initialize attributes

    {

        \_text = text;

        \_isHidden = false;

    }

…

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

This code from my program is clearly partial to what is there, but demonstrates the existence of private attributes \_text, and \_isHidden, which are defined as private and therefore cannot be accessed from anywhere outside the class.