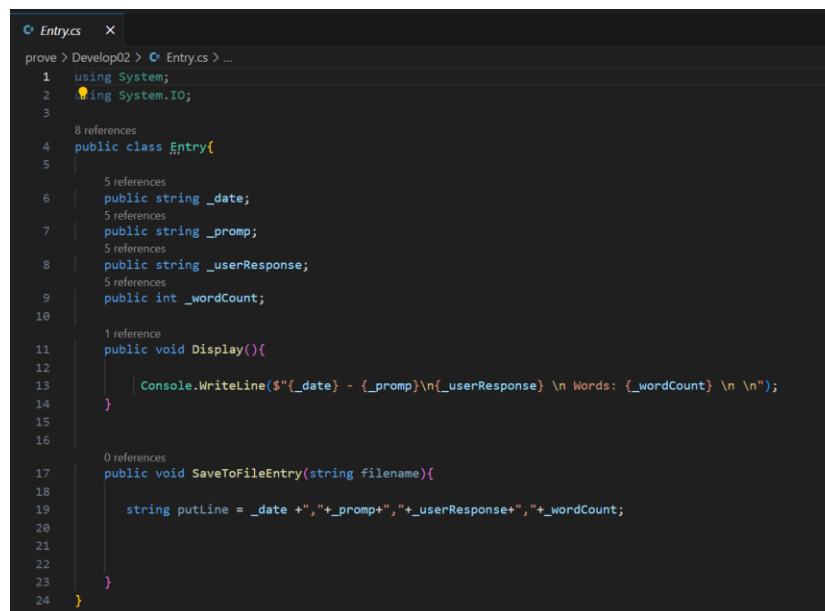


Abstraction - Articulate

What is abstraction and why is it important?

- Explain the meaning of Abstraction
 - The principle of abstraction is to convert something complex into something simple, its beginning may contain a lot of content but at the time of use it may be very simple, in other words for a program to print a text on the screen it would require many lines of code from the base but thanks to the abstraction as I have learned, user only `Console.WriteLine()` would be enough Highlight a benefit of Abstraction
- Highlight a benefit of Abstraction
 - Personally, I consider that one of the highlights of Abstraction is the reuniting of code, because if abstraction did not exist, we would have to write the entire code every time you needed to perform the same action, which would make the code difficult to maintain and understand, for this reason obstruction is very important in our case, the creation of classes and behaviors make this topic possible
- Provide an application of Abstraction
 - `Console.WriteLine()`:
 - This is a great example because behind this method there are many lines of code, and the abstraction means that we only have to use that line to perform the action.
- Code example of Abstraction from the program you wrote
 - In my program the Entry class is an example of abstraction because it allows adding multiple data and at the same time has a behavior that allows printing the attributes,



```
1  using System;
2  using System.IO;
3
4  8 references
5  public class Entry{
6      5 references
7      public string _date;
8      5 references
9      public string _prompt;
10     5 references
11     public string _userResponse;
12     5 references
13     public int _wordCount;
14
15     1 reference
16     public void Display(){
17         Console.WriteLine($"{_date} - {_prompt}\n{_userResponse} \n Words: {_wordCount} \n \n");
18     }
19
20     0 references
21     public void SaveToFileEntry(string filename){
22         string putline = _date + ","+_prompt+","+_userResponse+","+_wordCount;
23     }
24 }
```

Esta es la clase que permite la abstracción

```
Program.cs X
prove > Develop02 > Program.cs > ...
45
46     string[] words = entryTxt.Split(' ');
47
48     //creating entry
49     Entry entry1 = new Entry();
50     //geting date
51     DateTime theCurrentTime = DateTime.Now;
52     string dateText = theCurrentTime.ToShortDateString();
53
54     //adding to list new entry
55     entry1._date =dateText;
56     entry1._prompt =txtPrompt;
57     entry1._userResponse = entryTxt;
58     entry1._wordCount =words.Length;
59     myJurnal._entry.Add(entry1);
60     break;
61
62     case 2:
63         myJurnal.Display();
64         break;
65
66     case 3:
67         Console.WriteLine("What is the filename to load?");
68         filename = Console.ReadLine();
69         Console.WriteLine("Loading File .....");
70         myJurnal._filename = filename;
71         myJurnal.LoadFile();
72         Console.WriteLine("The file was loaded .....");
73         break;
74
75     case 4:
76         Console.WriteLine("What is the filename?");
77         filename = Console.ReadLine();
78         Console.WriteLine("Saving File .....");
79         myJurnal._filename = filename;
80         myJurnal.SaveToFile();
81         Console.WriteLine($"The entries were saved in the file {filename}!!!");
```

In the following code you can see the principle of abstraction because it allows us to act on the Entity class with a few lines of code and we do not need to write all the content of the Entity class every time we need

```
Program.cs X
prove > Develop02 > Program.cs > ...
45
46     string[] words = entryTxt.Split(' ');
47
48     //creating entry
49     Entry entry1 = new Entry();
50     //geting date
51     DateTime theCurrentTime = DateTime.Now;
52     string dateText = theCurrentTime.ToShortDateString();
53
54     //adding to list new entry
55     entry1._date =dateText;
56     entry1._prompt =txtPrompt;
57     entry1._userResponse = entryTxt;
58     entry1._wordCount =words.Length;
59     myJurnal._entry.Add(entry1);
60     break;
61
62     case 2:
63         myJurnal.Display();
64         break;
65
66     case 3:
67         Console.WriteLine("What is the filename to load?");
68         filename = Console.ReadLine();
69         Console.WriteLine("Loading File .....");
70         myJurnal._filename = filename;
71         myJurnal.LoadFile();
72         Console.WriteLine("The file was loaded .....");
73         break;
74
75     case 4:
76         Console.WriteLine("What is the filename?");
77         filename = Console.ReadLine();
78         Console.WriteLine("Saving File .....");
79         myJurnal._filename = filename;
80         myJurnal.SaveToFile();
81         Console.WriteLine($"The entries were saved in the file {filename}!!!");
```

to use it