CSE 210

Unit 02 Develop: Journal Program

Problem Overview

Many people see the value of keeping a journal to record important events, and many people even set this as a goal. And yet, very few people actually follow through and keep a journal consistently.

Think to yourself for a moment: What are some reasons people do not follow through with their goal to keep a journal? Could a program or app help with any of these?

Some of the reasons you thought of might include the following:

- We forget
- It's not convenient to get out our written journal or find the electronic document
- We don't feel like we have anything interesting to say
- We don't feel like we have time for it
- We aren't sure what to write
- We feel overwhelmed with writing every event of the day, so we just don't write anything.

While it will not solve all of people's problems, a great program or an app could help remove some of these barriers. For example, an app could give you a reminder at a certain time of day or give you a direct link to your document.

Consider the last challenge mentioned above, that of being overwhelmed because it feels like you must write every event during the day, this seems to be a big problem for many people. Could an app help with this?

Solution Idea

What if the Journal app gave people a simple prompt to respond to every day? It could also record the response somewhere for them and even add elements like the data automatically.

These features could help address some of the challenges that keep people from journaling, and could be included in a mobile app or on a web page. The actual interface is not that critical, but the ability for a program to help solve a real problem is important to recognize.

Program Specification

For this assignment you will write a program to help people record the events of their day by supplying prompts and then saving their responses along with the question and the date to a file.

Functional Requirements

This program must contain the following features:

1. Write a new entry - Show the user a random prompt (from a list that you create), and save their response, the prompt, and the date as an Entry.
2. Display the journal - Iterate through all entries in the journal and display them to the screen.
3. Save the journal to a file - Prompt the user for a filename and then save the current journal (the complete list of entries) to that file location.
4. Load the journal from a file - Prompt the user for a filename and then load the journal (a complete list of entries) from that file. This should replace any entries currently stored the journal.
5. Provide a menu that allows the user choose these options
6. Your list of prompts must contain at least five different prompts. Make sure to add your own prompts to the list, but the following are examples to help get you started:
    - Who was the most interesting person I interacted with today?
    - What was the best part of my day?
    - How did I see the hand of the Lord in my life today?
    - What was the strongest emotion I felt today?
    - If I had one thing I could do over today, what would it be?
7. Your interface should generally follow the pattern shown in the video demo below.

Design Requirements

In addition, your program must:

1. Contain classes for the major components in the program.
2. Contain at least two classes in addition to the **`Program`** class.
3. Demonstrate the principle of abstraction by using member variables and methods appropriately.

Simplifications

For the core requirements you do **not** need to worry about the following:

1. Saving your file as a .csv file requires you to handle commas and quotes in the content appropriately. At this point, you can ignore that and just choose a symbol for a separator that you think is unlikely to show up in the content (such as **`|`** or ~ or **`~|~`**).
2. You do not need to store the date as an actual C# **`DateTime`** object in your class or in the file. You can simply store it as a string.

Showing Creativity and Exceeding Requirements

Meeting the core requirements makes your program eligible to receive a 93%. To receive 100% on the assignment, you need to show creativity and exceed these requirements.

Here are some ideas you might consider:

- Think of other problems that keep people from writing in their journal and address one of those.
- Save other information in the journal entry.
- Improve the process of saving and loading to save as a .csv file that could be opened in Excel (make sure to account for quotation marks and commas correctly in your content.
- Save or load your document to a database or use a different library or format such as JSON for storage.

Report on what you have done to exceed requirements by adding a description of it in a comment in the Program.cs file.

Video Demo

The following video demonstrates the way this program should work:

Direct link: Journal Program Demo (3 minutes)

Code Helps

You might find the following code helps useful in this project:

- ▶ Reading Text Files in C#
- ▶ Writing Text Files in C#
- ▶ Working with Dates in C#

Still having difficulty reading and writing files?

The following video walks through through the development of a program that reads and writes a list of objects to a file in a similar way that you will need to for your program:

- CSV Reading and Writing Demo (20 Minutes)

Remember: Do not ever click "Debug Anyway".

If you have an error in your program and try to run it, VS Code will tell you that it cannot build a new version of your program to run. But, it may find an old version from the last time it successfully built the program. If you click "Debug Anyway" it will run the old version.

If you "Debug Anyway" you will be running a program that is different from your current code. This leads to lots of confusion, because you might make changes to your code, but the program does not show the changes.

To avoid these pitfalls, you should never click "Debug Anyway" instead, click "Abort" and track down your errors, or click "Show Errors" to go directly to them.

Design

You will work with your team to create a design for this program. Then, you will each write the code for the program individually.

Develop the Program

In the course repository, find the **`Develop02`** project in the **`Prove`** folder and write your program there.

Submission

1. Develop the program using the principle of Abstraction as described above.
2. Make sure to describe anything you have done to exceed the requirements in comments in the Program.cs file.
3. Commit your source code and push it to GitHub.
4. Verify that you can see your updated code at GitHub.

5. In I-Learn, submit a link to your GitHub repository. In the submission comment, describe anything you have done to show creativity and exceed the core requirements.

Working with Others

In this course you are encouraged to help and teach one another. But keep in mind that the goal is for you to **learn** how to write these programs yourself, not simply to turn in a working program.

Because the goal is for you to learn to write these programs on your own, you should *NOT* copy and paste any code from another student or from a repository you find online. Similarly, you should *NOT* look at their code and type it in directly from a picture or from what you see on your screen. This is a violation of the University Honor Code, will result in a 0 on the assignment or failing the class, and will not help you learn.

Instead, when you get help from another person, you should try to understand the principle of what they are doing, and then, use that to help you write your own code.

If you get help from a web page, include a link to that web page in your comments.

If you have any questions about this, please talk with your instructor.