Rachel Hostetler

05/28/2024

Foundations of Programming: Python

Assignment06

https://github.com/rachelhostetler/IntroToProg-Python-Mod06

Assignment06

Introduction

This assignment introduces the usage of classes to execute functions and uses the previous examples of menu options and student registration to show how classes can be used.

Creating the Script

First I opened Visual Studio, then opened "Assignment06-Starter.py," where I added my information to the change log at the top (Figure 1).

```
# ------------------------------------------------------------------------
# Title: Assignment06
# Desc: This assignment demonstrates using functions
# with structured error handling
# Change Log: (Who, When, What)
#   RRoot,1/1/2030,Created Script
#   Rachel Hostetler,05/25/2024,Assignment06
# ------------------------------------------------------------------------
```

Figure 1 – Header.

The next step was to define my data constants and variables based on the assignment and the information from the starter file (Figure 2).

```
# Define the Data Constants
MENU: str = '''
---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
-----------------------------------------
'''
# Define the Data Constants
FILE_NAME: str = "Enrollments.json"

# Define the Data Variables and constants
menu_choice: str = ''  # Hold the choice made by the user.
students: list = []
```

Figure 2 – Constants and variables.

The next step was to define the class “FileProcessor,” which contains functions related to processing the JSON file (reading and writing). Within this class, some functions also have error handling; examples include if a file is not found, if it is the wrong format, or if there is an undefined error (Figures 3.1 and 3.2).

```python
# define FileProcessor class
class FileProcessor:
    """
    this class processes files, such as reading and writing data to files

    ChangeLog: (Who, When, What)
    RachelHostetler,5/25/2024,Created Class
    """

    # define functions
    @staticmethod
    def read_data_from_file(file_name: str, student_data: list):
        """
        this function reads data from the file

        ChangeLog: (Who, When, What)
        RachelHostetler,5/25/2024,Created function
        """
        # When the program starts, read the file data into a list of lists (table)
        # Extract the data from the file
        try:
            file = open(file_name, "r")

            # JSON Answer
            student_data = json.load(file)

            file.close()

        except FileNotFoundError as e:
            IO.output_error_messages("File not found!", e)

        except Exception as e:
            IO.output_error_messages("Undefined Error!", e)

        finally:
            if file.closed == False:
                file.close()
        return student_data
```

Figure 3.1 – defining “FileProcessor” class and the function for reading the JSON file.

```python
@staticmethod
def write_data_to_file(file_name: str, student_data: list):
    """
    this function writes data to the file

    ChangeLog: (Who, When, What)
    RachelHostetler,5/25/2024,Created function
    """
    try:
        file = open(file_name, "w")

        # # JSON answer
        json.dump(student_data, file)

        file.close()

        IO.output_student_courses(student_data=student_data)

    except TypeError as e:
        IO.output_error_messages("check that data file is JSON format", e)
    except Exception as e:
        IO.output_error_messages("Undefined error!", e)

    finally:
        if file.closed == False:
            file.close()
```

Figure 3.2 – defining "FileProcessor" class and the function for saving the inputted data to a JSON file.

The next step was to define the second class, "IO." This class contains functions for the inputs and outputs. Some of the functions have handling errors, such as if a menu option outside of 1-4 is chosen, or if numbers are entered for the student's first and last names. This class has functions for displaying the menu options, inputting menu choices from the user, and displaying the output of the user-defined inputs (Figures 4.1- 4.3).

```python
# define IO class
class IO:
    """
    this class does inputs and outputs

    ChangeLog: (Who, When, What)
    RachelHostetler,5/25/2024,Created Class
    """

    # define functions
    @staticmethod
    def output_error_messages(message: str, error: Exception = None):
        """
        shows output error message, has handing error

        ChangeLog: (Who, When, What)
        RachelHostetler,5/25/2024,Created function
        """
        print(message, end="\n")
        if error is not None:
                print("- Error - ")
                print(error, error.__doc__, type(error), sep='\n')

    @staticmethod
    def output_menu(menu: str):
        """
        displays menu options
        ChangeLog: (Who, When, What)
        RachelHostetler,5/25/2024,Created function

        """
        print(menu)
```

Figure 4.1 – Defining functions within the “IO” class, specifically displaying the output menu and error handing messages.

```python
@staticmethod
def input_student_data(student_data: list):
    """
    this function enters data about the student

    ChangeLog: (Who, When, What)
    RachelHostetler,5/25/2024,Created function
    """
    try:
        student_first_name = input("Enter the student's first name: ")
        if not student_first_name.isalpha():
            raise ValueError("The last name should not contain numbers.")
        student_last_name = input("Enter the student's last name: ")
        if not student_last_name.isalpha():
            raise ValueError("The last name should not contain numbers.")
        course_name = input("Please enter the name of the course: ")
        student = {"FirstName": student_first_name,
                   "LastName": student_last_name,
                   "CourseName": course_name}
        student_data.append(student)
        print(f"You have registered {student_first_name} {student_last_name} for {course_name}.")
    except ValueError as e:
        IO.output_error_messages("Make sure input data is correct", e)
    except Exception as e:
        IO.output_error_messages("Undefined error!", e)
    return student_data
```

Figure 4.2 – Defining more functions of the “IO” class, specifically functions for inputting student data about first/last name and course title, while providing error handling when inputted data is not correct.

```
@staticmethod
def input_menu_choice():
    """
    input menu choices from user
    ChangeLog: (Who, When, What)
    RachelHostetler,5/25/2024,Created function
    """

    menu_choice = "0"
    try:
        menu_choice = input("Enter your menu choice number: ")
        if menu_choice not in ("1","2","3","4"):
            raise Exception("choose only 1, 2, 3, or 4")
    except Exception as e:
        IO.output_error_messages("undefined error!", e)
    return menu_choice

@staticmethod
def output_student_courses(student_data: list):
    """
    this function shows the students and their enrolled courses

    ChangeLog: (Who, When, What)
    RachelHostetler,5/25/2024,Created function
    """
    # Process the data to create and display a custom message
    print("-" * 50)
    for student in student_data:
        print(f'Student {student["FirstName"]} '
              f'{student["LastName"]} is enrolled in {student["CourseName"]}')
    print("-" * 50)
```

Figure 4.3 – defining the class functions of the ”IO” class, which include the input of menu choices and error handling when 1-4 is not chosen, and an output function that confirms that a student is enrolled for the course with the user-defined first and last names and course title.

The final step was to write the program that displays the menu options, takes inputs from a user, displays the inputted data back to a user, and saves the inputted data using the classes defined above. Menu option 1 inputs user-defined data using the “IO” class, option 2 shows outputs also using “IO,” while option 3 saves the data to the JSON file using the “FileProcessor” class. (Figure 5).

```
students = FileProcessor.read_data_from_file(file_name = FILE_NAME, student_data = students)

# Present and Process the data
while (True):

    IO.output_menu(menu = MENU)

    # Present the menu of choices
    menu_choice = IO.input_menu_choice()

    # Input user data
    if menu_choice == "1":  # This will not work if it is an integer!

        students = IO.input_student_data(student_data = students)
        continue

    # Present the current data
    elif menu_choice == "2":

        IO.output_student_courses(student_data = students)
        continue

    # Save the data to a file
    elif menu_choice == "3":

        FileProcessor.write_data_to_file(file_name = FILE_NAME, student_data = students)
        continue

    # Stop the loop
    elif menu_choice == "4":
        break  # out of the loop
    else:
        print("Please only choose option 1, 2, or 3")

print("Program Ended")
```

Figure 5. Program that uses the “IO” and “FileProcessor” classes to input data, display it back to the user, and save it to the a JSON file.

Summary

This assignment builds on the previous assignments of displaying Menu options and using these options to collect inputted data from a user, display that data back to the user as an output, and save the data to a file. This time the program saves to a JSON instead of a CSV, and uses classes to execute the functions.