Simon Schumacher

August 10, 2024

IT FDN 110 A Su 24: Foundations of Programming: Python

Assignment06

https://github.com/Simon-LaVassar/IntroToProg-Python-Mod06.git

Simplifying Code Utilizing Functions and Classes

Introduction

We continue to iterate on the program we have developed for this course. For this assignment, we utilized two classes, ‘FileProcessor’ and ‘IO’, and several functions to read and write to a JSON file. This represents the concept of *Separation of Concerns*, which allows us to break down complicated problems to their specific asks, analogous to a divide and conquer approach to problem-solving.

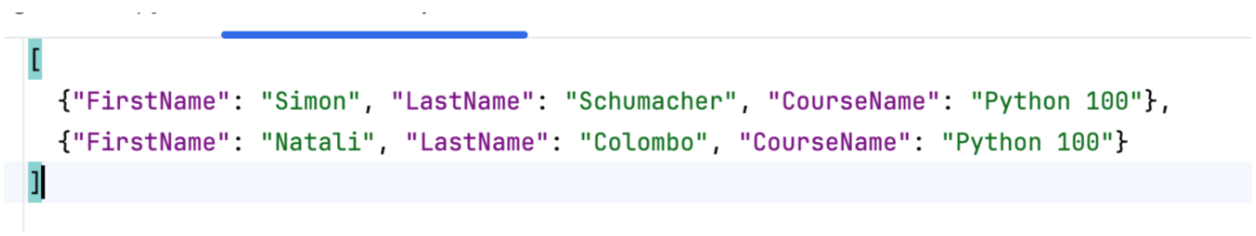
Figure 1 – The requisite Python code is as follows:

```
# ------------------------------------------------------------------------------------------ #
# Title: Assignment06
# Desc: This assignment demonstrates using functions
# with structured error handling
# Change Log: (Who, When, What)
#   RRoot,1/1/2030,Created Script
#   SSimon,08/07/2024,Finished Script
# ------------------------------------------------------------------------------------------ #
import json

# Define the MENU Data Constant
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 FILE_NAME Data Constant
FILE_NAME: str = "Enrollments.json"

# Define the Data Variables and constants
students: list = []  # a table of student data
menu_choice: str  # Hold the choice made by the user.
```

```
# Define the class 'IO'
class IO:

    @staticmethod
    def output_error_messages(message: str, error: Exception = None):
        """
        Prints an error message followed by technical details of the error if provided.

        :param message: A user-friendly error message.
        :param error: The caught exception object (default is None).
        """
        print(message)
        if error:
            print("-- Technical Error Message --")
            print(error.__doc__)
            print(error.__str__())

    @staticmethod
    def output_menu(menu: str):
        """
        Displays the program menu to the user.

        :param menu: The menu string to be displayed.
        """
        print(menu)

    @staticmethod
    def input_menu_choice() -> str:
        """
        Captures the user's menu choice.

        :return: The menu choice entered by the user as a string.
        """
        menu_choice = input("What would you like to do: ")
        return menu_choice

    @staticmethod
    def output_student_courses(students: list):
        """
        Displays the current list of students and their enrolled courses.

        :param students: A list of dictionaries, where each dictionary represents a student and
their course.
        """
        print("-" * 50)
        for student in students:
            print(f'Student {student["FirstName"]} '
                  f'{student["LastName"]} is enrolled in '
                  f'{student["CourseName"]}')
        print("-" * 50)

    @staticmethod
    def input_student_data(students: list) -> list:
        """
        Captures and validates student information, then adds it to the list of students.
        :param students: A list of dictionaries representing the current students.
        :return: The updated list of students with the new entry added.
        """
        try:
            student_first_name = input("Enter the student's first name: ")
            if not student_first_name.isalpha():
                raise ValueError("The first 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_data = {"FirstName": student_first_name,
                            "LastName": student_last_name,
                            "CourseName": course_name}
            students.append(student_data)
            print(f"You have registered {student_first_name} "
                  f"{student_last_name} for {course_name}.")

        except ValueError as e:
            IO.output_error_messages(e.__str__(), e)

        except Exception as e:
            IO.output_error_messages("Error: There was a problem "
                                     "with your entered data.", e)

        return students


# Define the class 'FileProcessor'
class FileProcessor:

    @staticmethod
    def read_data_from_file(file_name: str, students: list) -> list:
        """
        Reads student data from a JSON file and loads it into the students list.

        :param file_name: The name of the file to read from.
        :param students: The current list of students (will be overwritten by file data).
        :return: The updated list of students read from the file.
        """
        file = None
        try:
            file = open(file_name, "r")
            students = json.load(file)
            file.close()

        except Exception as e:
            IO.output_error_messages("Error: There was a problem "
                                     "with reading the file.\n"
                                     "Please check that the file exists and "
                                     "that it is in a json format.", e)

        finally:
            if file is not None and not file.closed:
                file.close()

        return students
```

```python
    @staticmethod
    def write_data_to_file(file_name: str, students: list):
        """
        Writes the current list of students to a JSON file.

        :param file_name: The name of the file to write to.
        :param students: The list of students to be saved to the file.
        """
        file = None
        try:
            file = open(file_name, "w")
            json.dump(students, file)

            file.close()
            print("The following data was saved to file!")
            for student in students:
                print(f'Student {student["FirstName"]} '
                      f'{student["LastName"]} is enrolled in '
                      f'{student["CourseName"]}')

        except Exception as e:
            if file is not None and not file.closed:
                file.close()
            IO.output_error_messages("Error: There was a "
                                     "problem with writing to the file."
                                     "Please check that the file is not "
                                     "open by another program.", e)


# When the program starts, read the file data into a list of lists (table)
# Extract the data from the file
students = FileProcessor.read_data_from_file(FILE_NAME, students)

# Present and Process the data
while True:

    # Present the menu of choices
    IO.output_menu(MENU)
    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(students)
        continue

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

        # Process the data to create and display a custom message
        IO.output_student_courses(students)
        continue

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

        FileProcessor.write_data_to_file(FILE_NAME, students)
        continue

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

print("Program Ended")
```

Figure 2 – An example JSON file utilized is as follows:



```
[
  {"FirstName": "Simon", "LastName": "Schumacher", "CourseName": "Python 100"},
  {"FirstName": "Natali", "LastName": "Colombo", "CourseName": "Python 100"}
]
```

Notable Steps

1. Download and unzip _Module06.zip
2. Open Assignment06-Starter.py
3. Rename Assignment04-Starter.py to Assignment06.py
4. Save Assignment06.py
5. Created the two classes, 'FileProcessor' and 'IO'.
6. Created the following seven functions within their respective class:
    a. IO Functions
        i. output_error_messages(message: str, error: Exception = None)
        ii. output_menu(menu: str)
        iii. input_menu_choice()
        iv. output_student_courses(student_data: list)
        v. input_student_data(student_data: list)
    b. FileProcessor Functions
        i. read_data_from_file(file_name: str, student_data: list):
        ii. write_data_to_file(file_name: str, student_data: list):
7. Utilizing our code from previous versions, nest functionality within each function.
8. Remove functional code from script; replace it with its respective method.
9. Write documentation describing each block of code.
10. Utilize both PyCharm and Terminal to run Assignment06.py
11. Upload code to GitHub and compress components to .zip file.

Conclusion

We created a program that iterated on the program we have been developing for this course. We utilized two classes, ‘FileProcessor’ and ‘IO’, and several functions to read and write student enrollment data to a JSON file. Our file has largely the same functionality as the previous version but would be much more approachable to develop as the functionality is largely confined to classes and functions that are utilized throughout the program.