Katie Debetaz

Nov 26, 2024

Foundations of Programming: Python

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

Assignment 07: Classes and Objects

Introduction

In this assignment, I enhanced my program from Assignment 6 by adding a set of data classes. These new classes are a Person and Student class that will be discussed in detail throughout this assignment. The new classes will show how the methods they contain can be called without having to use the @staticmethod decorator.

Person Class

The first class I implemented was the Person class as shown in Figure 1. This class gives an outline to be used for each person. It includes variables to store the first and last names of each student. This just gives a template that can be used repeatedly for as many people as needed. The Student class, to be discussed later, references this class and then adds more variables on top of it.

```python
class Person:
    """
    This class represents person data.

    Properties:
        student_first_name: str
        student_last_name: str

    Katie Debetaz, 11/26/2024, Created Class
    """
    def __init__(self,student_first_name: str='',student_last_name: str=''):
        self.student_first_name = student_first_name
        self.student_last_name = student_last_name

    4 usages (2 dynamic)
    @property
    def student_first_name(self):
        return self.__student_first_name.title()

    3 usages (2 dynamic)
    @student_first_name.setter
    def student_first_name(self, value:str):
        if value.isalpha() or value == "":
            self.__student_first_name = value
        else:
            raise ValueError("The first name must be alphabetic")
```

Figure 1. Person Class

Student Class

The second class I created was the Student class which is shown in Figure 2. This class organizes all of the data for each particular student. The student class adds variables for the course name and assigns it to the students. The last function in the class returns a comma separated list of the students' first, last and course names.

```python
class Student(Person):
    """
    This class represents student data.

    Properties:
        student_first_name: str
        student_last_name: str
        course_name: str

    Katie Debetaz, 11/26/2024, Created Class
    """

    def __init__(self,student_first_name: str='',student_last_name: str='',course_name: str=''):
        super().__init__(student_first_name=student_first_name, student_last_name=student_last_name)
        self.course_name = course_name

    4 usages (2 dynamic)
    @property
    def course_name(self):
        return self.__course_name.title()

    4 usages (2 dynamic)
    @course_name.setter
    def course_name(self, value:str):
        self.__course_name = value


    def __str__(self):
        return f'{self.student_first_name},{self.student_last_name},{self.course_name}'
```

Figure 2. Class Student

Summary

This assignment demonstrates the addition of more classes and methods to the already existing program. These new classes show how their methods can be utilized without using the @staticmethod decorator. Overall, these additions improve the code by making it more organized and efficient.