Eric Hoyle

03-06-21

Foundations in Programming; Python

Assignment\_08

# Introduction

In week 8 I was more formally introduced to classes and took my first foray into software objects as we begin leaning toward Object Oriented Programming (OOP). I implemented, for the first time in my programming, objects created through a class instantiation. I then revised the ever-present CDInventory program to incorporate these newly acquired skills.

# Good Sir, I Object!

Ok, so class instantiated objects are pretty much the best thing since sliced bread. At first brush it was clear what a drastic improvement creating classes and instantiating objects was over the more tedious one-off builds we have been working with so far. Naturally, the nuances and intricacies of this approach took a little while to sink in but overall I felt I picked this up pretty intuitively. Perhaps more so than anything we have done so far. Maybe because this is the “link” to real world objects from which this approach gets its moniker.

Alright, so I learned something. But what did I do with it you ask? The first thing I thought when this topic was broached in the materials was “CDs in my program should be objects”. Now they are! I built my CD class to be a little CD object builder. Its lean for efficiency but gets the job done.

Text

Description automatically generatedFigure - The CD class

I built the \_\_init\_\_ constructor to spec to include the index, title, and artist attributes and defined the requisite property and setter functions as well with setters incorporating necessary exception handling. This made sense in the initial build of the class but after building out the rest of the program there isn’t much opportunity to redefine the attributes so the setter exception handling mostly falls by the wayside. Instead the exception handling is covered in the IO add\_cd() function.

Text

Description automatically generated

Figure 2- Add\_cd function with exception handling

I also made sure that the two methods in this object class were useful and pertinent for making exporting data simple. I defined the method file\_export to handle just such a task.



Figure 3 – This machine spits data as type str!

Speaking of which, let’s look at some file processing, shall we?

# Two Wrongs Don’t Make a Write

In fact, there is *no* *room* for error when writing strings to a text file in such a stringent manner. Good thing I cooked up that file\_export method for my CD class. Makes everything a breeze! But when we get into the file import things got a little dicey.

Graphical user interface, text

Description automatically generated

Figure 3 - The Read function

Because I sent the data out as a string, it comes back as a string and I needed to objectify it (or maybe I didn’t?) to keep in scope with this week’s assignment. Maybe I should have created another data processing class for this but I went ahead and stuck the object building in this class because I reasoned it was part of the import process. I also found a way to sort the data in order if CD index. I spent a little too long on that and still am not certain how it works, but it does. I also tried implementing the sort feature to the list of CDs prior to displaying the inventory but it kept raising exceptions for Type Errors which I couldn’t resolve.

# Summary

This week was a lot of fun and I felt like I was back on my feet and ready to charge ahead. This concept of creating objects makes things much clearer for me and made me feel recharged after a couple of grueling weeks past. Overall, I still have some lingering questions; namely the sort I was attempting and why it works on the data import but not in the inventory display. I also attempted to incorporate cd removal in my program and was successful at one point in removing some cd object elements from my list but then I made a few changes and couldn’t recover the functionality in the time I had. A quest for another day perhaps.

# In Spyder

Graphical user interface, text, application

Description automatically generated

Figure 4- Working in Spyder

# In Terminal

Text

Description automatically generated

Figure 5- Working in the Terminal

# Appendix

Using [Planetb’s](http://www.planetb.ca/syntax-highlight-word) [[1]](#footnote-1)(external reference) web page

Text

Description automatically generated with medium confidence

Text

Description automatically generated

A picture containing timeline

Description automatically generated

1. Retrieved 03-07-21 [↑](#footnote-ref-1)