Ryan L. Blake

24 August 2021

IT FDN 100

Assignment 07

Link TBD

**Pickling and Error Handling**

# Introduction

# Topics

* Class objects
* Fundamental object methods
* GitHub desktop

# Summary

# Appendix: Python Script

*# ------------------------------------------------------------------------ #  
# Title: Assignment 08  
# Description: Working with classes  
  
# ChangeLog (Who,When,What):  
# RRoot,1.1.2030,Created started script  
# RRoot,1.1.2030,Added pseudo-code to start assignment 8  
# rblake50, 09.04.2021,Modified code to complete assignment 8  
# ------------------------------------------------------------------------ #  
# DESCRIPTION  
# This program allows a user to read list data from a pickled file and  
# read or add data to the list. Then, the user can exit the program and  
# save the data back to the pickled file. The data being stored is Product  
# information containing the name and price of the Product. The Product is  
# managed as a Python object through class properties.  
# ------------------------------------------------------------------------ #  
# ASSUMPTIONS  
# The user is reading from and writing to a pickled file (such as .dat)  
# that is properly organized with Product objects***import** pickle  
**from** datetime **import** datetime  
  
*# Data -------------------------------------------------------------------- #*strFileName = **'products.dat'**lstOfProductObjects = []  
  
**class** Product:  
 *"""Stores data about a product:  
  
 properties:  
 product\_name: (string) with the products's name  
 product\_price: (float) with the products's standard price  
 methods:  
 changelog: (When,Who,What)  
 RRoot,1.1.2030,Created Class  
 rblake50,09.04.2021,Modified code to complete assignment 8  
 """  
  
 # Fields* product\_name = **""** product\_price = 0.0  
  
 *# Constructors* **def** \_\_init\_\_(self):  
 print(**"A new product has been added!"**)  
  
 *# Attributes* self.product\_name = **""** self.product\_price = 0.0  
  
 **def** \_\_str\_\_(self):  
 tmp = **"name: "** + self.\_product\_name + **", price: $%.2f"** % self.\_product\_price  
 **return** tmp  
  
 *# Properties* @property *# Getter* **def** product\_name(self):  
 **return** self.product\_name  
  
 @product\_name.setter  
 **def** product\_name(self,name):  
 self.\_product\_name = name  
  
 @property *# Getter* **def** product\_price(self):  
 **return** self.product\_price  
  
 @product\_price.setter  
 **def** product\_price(self,price):  
 **try**:  
 self.\_product\_price = float(price)  
 **except** ValueError:  
 print(**"D'oh! Price must be a number. Try again."**)  
  
*# Processing ------------------------------------------------------------- #***class** FileProcessor:  
 *"""Processes data to and from a file and a list of product objects:  
  
 properties:  
 N/A  
 methods:  
 save\_data\_to\_file(file\_name, list\_of\_product\_objects):* **:param** *file\_name (string) name of file to save* **:param** *list\_of\_product\_objects (list) of product objects  
  
 read\_data\_from\_file(file\_name): -> (a list of product objects)* **:param** *file\_name (string) name of file to read* **:return** *list of product objects  
  
 changelog: (When,Who,What)  
 RRoot,1.1.2030,Created Class  
 rblake50,09.04.2021,Modified code to complete assignment 8  
 """  
  
 # Save data to a pickled file* @staticmethod  
 **def** save\_data\_to\_file(file\_name, list\_of\_product\_objects):  
  
 **with** open(file\_name, **'wb'**) **as** file:  
 pickle.dump(list\_of\_product\_objects, file)  
  
 print(**"File successfully pickled to "** + file\_name)  
  
 *# Save data to .txt file* @staticmethod  
 **def** save\_data\_to\_txt(file\_name, list\_of\_product\_objects):  
  
 file\_prefix = file\_name.split(**"."**)[0]  
 file\_to\_save = file\_prefix + **".txt"  
  
 with** open(file\_to\_save, **'w'**) **as** file:  
  
 file.write(datetime.now().strftime(**"%m/%d/%y, %H:%M:%S\n======\n"**))  
 **for** item **in** list\_of\_product\_objects:  
 file.write(item.\_\_str\_\_() + **"\n"**)  
  
 print(**"Text file successfully written to "** + file\_to\_save)  
  
 *# Read data from a pickled file* @staticmethod  
 **def** read\_data\_from\_file(file\_name):  
  
 lstData = [] *# Initiate as list* **try**:  
 **with** open(file\_name, **'rb'**) **as** fileIncoming:  
 lstData = pickle.load(fileIncoming)  
  
 **except** FileNotFoundError: *# File does not exist in immediate directory* print(**"The file "** + file\_name + **" does not exist! Add data and save to create file."**)  
  
 **except** EOFError: *# File exists in immediate directory but has no information* input(**"The file has no content! Please add data."**)  
  
 **return** lstData  
  
*# Presentation (Input/Output) -------------------------------------------- #***class** IO:  
 *"""Manages user input and file output  
  
 properties:  
 methods:  
 show\_menu()  
 1. Show user current data in the list of product objects  
 2. Let user add data to the list of product objects  
 3. let user save current data to file and exit program  
  
 return\_to\_menu()  
  
 get\_choice()* **:return** *string of user choice  
  
 show\_data(file\_name)* **:param** *list\_of\_product\_objects (list) of product objects to show  
  
 get\_data()* **:return** *Product object with name and price of product  
  
 changelog: (When,Who,What)  
 rblake50,09.04.2021,Modified code to complete assignment 8  
 """  
 # Show menu to user* @staticmethod  
 **def** show\_menu():  
 strMenu = **"=== MENU ===\n1. Show current data\n2. Add product\n3. Save and exit"** print(strMenu)  
  
 *# Show prompt to return to menu* @staticmethod  
 **def** return\_to\_menu():  
 input(**"Press ENTER to return to menu."**)  
  
 *# Get user's choice* @staticmethod  
 **def** get\_choice():  
 strChoice = input(**"What is your choice? "**)  
 **return** strChoice  
  
 *# Show the current data from the file to user* @staticmethod  
 **def** show\_data(list\_of\_product\_objects):  
  
 **if** len(list\_of\_product\_objects) != 0:  
 **try**:  
 print(**"Here is your list of items:"**)  
 **for** item **in** list\_of\_product\_objects:  
 print(item)  
  
 **except** TypeError:  
 input(**"List is the wrong type! Please add data."**)  
  
 **else**:  
 print(**"The list is empty. Please add items."**)  
  
 *# Get product data from user* @staticmethod  
 **def** get\_data():  
  
 **try**:  
 name = input(**"What is the product name? "**)  
 price = float(input(**"What is the product price? "**))  
  
 prod = Product()  
 prod.product\_name = name  
 prod.product\_price = price  
  
 **return** prod  
  
 **except** ValueError:  
 print(**"D'oh! Price must be a number. Try again."**)  
  
*# Main Body of Script ---------------------------------------------------- #  
  
# Load data from file into a list of product objects when script starts*lstOfProductObjects = FileProcessor.read\_data\_from\_file(strFileName)  
  
*# Show data from loaded file*IO.show\_data(lstOfProductObjects)  
  
**while True**:  
  
 *# Show user a menu of options* IO.show\_menu()  
  
 *# Get user's menu option choice* choice = IO.get\_choice()  
  
 *# 1. Show user current data in the list of product objects* **if** choice == **"1"**:  
  
 IO.show\_data(lstOfProductObjects)  
  
 IO.return\_to\_menu()  
  
 *# 2. Let user add data to the list of product objects* **elif** choice == **"2"**:  
 print(**"You chose to add data to the list."**)  
  
 *# Get data for new Product from user* product = IO.get\_data()  
 **if** product **is not None**: *# get\_data() will return None for invalid entry* lstOfProductObjects.append(product)  
  
 *# Prompt return to menu* IO.return\_to\_menu()  
  
 *# 3. let user save current data to file and exit program* **elif** choice == **"3"**:  
  
 print(**"You chose to save and exit."**)  
  
 *# Prompt user for a text file copy of the data* print(**"Do you want to export a .txt file? [y]es or [n]o."**)  
 choice = IO.get\_choice().lower()  
  
 *# User wants to save text file* **if** choice == **"y" or** choice == **"yes"**:  
  
 *# Save data to pickled file and text file with same file name (less extension)* FileProcessor.save\_data\_to\_file(strFileName, lstOfProductObjects)  
 FileProcessor.save\_data\_to\_txt(strFileName, lstOfProductObjects)  
 **break** *# User does not want to save text file* **elif** choice == **"n" or** choice == **"no"**:  
  
 *# Save data to pickled file but \*NOT\* to text file* FileProcessor.save\_data\_to\_file(strFileName, lstOfProductObjects)  
 print(**"File \*NOT\* written to text file. Good-bye!"**)  
 **break** *# Invalid entry* **else**:  
  
 print(**"Invalid selection! Data will not be saved."**)  
 IO.return\_to\_menu()  
  
 *# Invalid choice* **else**:  
 print(**"D'oh! Your choice is invalid."**)  
 IO.return\_to\_menu()