Fikru Abasori

Nov30,2021

IT FDN 110A

Assignment 07

<https://github.com/fikru2021/IntroToProg-Python-Mod07>.

Intro

Working on this assignment, I have learned about about exception handling and Python’s pickling module, and how you use them.

1) Exception Handling in Python

Python Try Except: The try block lets you test a block of code for errors, where as the except block lets you handle the error. The finally block lets you execute code, regardless of the result of the try- and except blocks. When an error occurs, or exception as we call it , python will normally stop and generate an error message. Exceptions can be handled using the try statement:

For example: In this code below, the try block will generate an exception , because x is not defined:

try:  
  print(x)  
except:  
  print("An exception occurred")

Note: “ Since the try block raises an error, the except block will be executed. Without the try block, the program will crash and raise an error.” ( <https://www.w3schools.com/python/python_try_except.asp>. ) ( External site)

2) Pickling in Python

“ Pickling is the serializing and de-serializing of python objects to a byte stream. Unpicking is the opposite.” Pickling is used to store python objects. Objects likes lists, dictonaries, and class objects. Pickling is most useful with data analysis, where you are performing routine tasks on a data, such as pre-processing. In addition, it makes sense to use pickling if you are working with python-specific data types . such as dictonaries.

If you have a large dataset, for example, and you're loading that massive data set into memory every time you run the program, it may make a lot of sense to just pickle it, and then load that instead, because it will be far faster, again by 50 - 100x, sometimes far more depending on the size.

Example:

import pickle

example\_dict = {1:"6",2:"2",3:"f"}

pickle\_out = open("dict.pickle","wb")

pickle.dump(example\_dict, pickle\_out)

pickle\_out.close()

First, importpickle to use it, then define an exampke dictionary which is a pyhton object. Next open a file ( note that we open to write bytes in Python 3+), then we use pickle.dump() to put the dict into opened file, then close.

The above code will save the pickle file for us, now we need to cover how to access the pickled file:

pickle\_in = open("dict.pickle","rb")

example\_dict = pickle.load(pickle\_in)

open the pickle file, use pickle.load() to load it to a var.

print(example\_dict)

print(example\_dict[3])

This shows that we have retabined the dict data-type.

<https://pythonprogramming.net/python-pickle-module-save-objects-serialization/>. ( External site).

**Create** a new sub-folder called Assignment07 inside the \_PythonClass folder

**Graphical user interface, application

Description automatically generated**

Fig 1. Showing folder location for assignment 7

**Create** a new project in PyCharm that uses the \_PythonClass\Assignment07 folder as its location

Text

Description automatically generated

Fig 2. Showing a new project in Pychram in the right folder

**Create** a file called, "Assigment07.py," in your project.

Graphical user interface, text

Description automatically generated

Fig3. Showing Assignment07.py in the right folder

**Add** code to your script that performs the assignment’s task. Don't forget to include the script's header.

* Step 1: Import pickle from another code file

’

* Fig4.1 Showing code to import pickle
* Step 2: Pickling demo

Graphical user interface, text

Description automatically generated

* Fig 4.2. Showing pickling demo
* Step 3: Store the data using the pickle.dump method
* Text

  Description automatically generated
* Fig 4.3. Showing a code that is used to store data using pickle.dump method
* Step 4: Read the data back using the picke load method
* Text

  Description automatically generated
* Fig 4.4 showing code to read a data back using the pickle load method
* Step 5: Error handling
* Text

  Description automatically generated

Fig 5. Showing the error handling demo

**Run** the script in BOTH PyCharm and an OS command/shell window and capture images of it working on your computer

Text

Description automatically generated

Fig6. Showing the program running in PyChram

Text

Description automatically generated

Fig7. Showing the program using IDLE

**Verify** that it worked, by locating the binary file and opening it in a text editor. The file should be in the same folder as your script when you used the correct, **relative** file path!

Graphical user interface, application

Description automatically generated

Note: I see there Appdata.dat file in the folder but no data is loaded to it, could not figure out why.

**Summary**

In general, working on this assignment, , I have learned about about exception handling and Python’s pickling module, and how you use them.