Top Dog Crypto Converter Documentation:

Source Code:

# Top Dog Crypto Converter  
# Python Project  
# Indroduction to Software Development  
# Ivy Tech Community College  
# 12.14.21  
# Dustin Hanshew  
# Professor Carver  
  
  
# My project will allow you to show the fiat rate conversion for USD, EUR, or XAU(Gold) to BTC, SHIB, DOGE, or ELON  
  
import json  
import tkinter as tk  
from functools import partial  
from tkinter import \*  
from tkinter import ttk  
  
# Real time data from currency conversion using coinmarketcap api  
class CurrencyConverter:  
  
 def validateLogin(username, password):  
 print("username entered :", username.get())  
 print("password entered :", password.get())  
 if username == '' or password == '':  
 print("fill the empty field!!!")  
 else:  
 if username == "" and password == "":  
 print("Login success")  
 else:  
 print("Wrong username or password!!!")  
  
 # window for username and password  
 tkWindow = Tk()  
 tkWindow.geometry('1000x300')  
 tkWindow.title('Top Dog Converter Log In Form')  
  
 #instructions label  
 instructionsLabel = Label(tkWindow, text="Welcome to Top Dog Converter! Enter in the username and password to continue. "  
 "Username is: username and Password is: password.").grid(row=16, column=2)  
 # username label and text entry box  
 usernameLabel = Label(tkWindow, text="User Name").grid(row=0, column=0)  
 username = StringVar()  
 usernameEntry = Entry(tkWindow, textvariable=username).grid(row=0, column=1)  
 # password label and password entry box  
 passwordLabel = Label(tkWindow, text="Password").grid(row=1, column=0)  
 password = StringVar()  
 passwordEntry = Entry(tkWindow, textvariable=password, show='\*').grid(row=1, column=1)  
  
 validateLogin = partial(validateLogin, username, password)  
  
 # login button  
 loginButton = Button(tkWindow, text="Login", command=validateLogin).grid(row=4, column=0)  
  
 # creating SHIB INU image  
 photo1 = PhotoImage(file=r"shiba.png")  
 photoimage1 = photo1.subsample(3, 3)  
 shib\_image = tk.Label(tkWindow, image=photoimage1)  
 shib\_image.place(x=30, y=150)  
  
 # creating DOGECOIN image  
 photo2 = PhotoImage(file=r"dogecoin.png")  
 photoimage2 = photo2.subsample(3, 3)  
 doge\_image = tk.Label(tkWindow, image=photoimage2)  
 doge\_image.place(x=410, y=150)  
  
 photo3 = PhotoImage(file=r"btc\_image.png")  
 photoimage3 = photo3.subsample(4, 4)  
 btc\_image = tk.Label(tkWindow, image=photoimage3)  
 btc\_image.place(x=900, y=150)  
 validateLogin()  
 tkWindow.mainloop()  
# imports from the api, tkinter, and json  
  
  
  
  
 # defining the fiat amount  
def convert(crypto, fiat, amount):  
  
 # url for coinmarket cap api  
 url = 'https://pro-api.coinmarketcap.com/v1/cryptocurrency/quotes/latest'  
 parameters = {  
 'symbol': crypto,  
 'convert': fiat  
 }  
 headers = {  
 'Accepts': 'application/json',  
 'X-CMC\_PRO\_API\_KEY': '4500cda3-9aea-44d5-8e16-c2bb712d74e4',  
  
 }  
 from requests.exceptions import ConnectionError, Timeout, TooManyRedirects  
 from requests import Session  
 session = Session()  
 session.headers.update(headers)  
 int\_amount = amount  
  
 try:  
 response = session.get(url, params=parameters)  
 data = json.loads(response.text)  
 amount = round(data['data'][crypto]['quote'][fiat]['price'] \* int\_amount, 2)  
 except (ConnectionError, Timeout, TooManyRedirects, KeyError) as e:  
 print(e)  
 return -1  
  
 return amount  
  
# Making the tkinter window  
class App(tk.Tk):  
  
 def \_\_init\_\_(self):  
 tk.Tk.\_\_init\_\_(self)  
 self.title = 'Top Dog Crypto Converter'  
 # setting window  
 self.configure(background='#856ff8')  
 self.geometry("520x400")  
  
 # Title  
 self.title = Label(self, text="Top Dog Crypto Converter", fg='white', bg='#2D496D', borderwidth=3)  
 self.title.config(font=('Helvetica', 15, 'bold'))  
 self.title.place(x=100, y=5)  
  
 # label  
 self.crypto\_label = Label(self, text='Crypto Currency', fg='white', bg='#2D496D', borderwidth=3)  
 self.crypto\_label.config(font=('Helvetica', 13, 'bold'))  
 self.fiat\_label = Label(self, text='Fiat Currency', fg='white', bg='#2D496D', borderwidth=3)  
 self.fiat\_label.config(font=('Helvetica', 13, 'bold'))  
  
 # retrieve list of currencies from cryptocurrency.txt  
 with open('cryptocurrency.txt') as f:  
 list = [line.split()[0] for line in f]  
  
 currList = []  
 temp = []  
 for i in list:  
 temp.append(i)  
 if i == 'Fiat:':  
 currList.append(temp)  
 temp = []  
 if temp:  
 currList.append(temp)  
  
 list\_crypto\_currency = currList[0]  
 list\_crypto\_currency = list\_crypto\_currency[1:-1]  
 fiat\_list\_currency = currList[1]  
  
 # Dropdown  
 self.crypto\_currency\_variable = StringVar(self)  
 self.crypto\_currency\_variable.set("SHIB")  
 self.fiat\_currency\_variable = StringVar(self)  
 self.fiat\_currency\_variable.set("USD")  
  
 font = ("Courier", 12, "bold")  
 self.option\_add('\*TCombobox\*Listbox.font', font)  
 self.cryto\_currency\_dropdown = ttk.Combobox(self, textvariable=self.crypto\_currency\_variable, values=list\_crypto\_currency, font=font, state='readonly', width=12, justify=tk.CENTER)  
 self.fiat\_currency\_dropdown = ttk.Combobox(self, textvariable=self.fiat\_currency\_variable, values=fiat\_list\_currency, font=font, state='readonly', width=12, justify=tk.CENTER)  
  
 # Entry box  
 valid = (self.register(self.restrictNumberOnly), '%d', '%P')  
 self.amount\_field = Entry(self, bd=3, relief=tk.RIDGE, justify=tk.CENTER, validate='key', validatecommand=valid)  
 self.converted\_amount\_field = Label(self, text='', fg='black', bg='white', relief=tk.RIDGE, justify=tk.CENTER, width=17, borderwidth=3)  
 # Convert button  
 self.convert\_button = Button(self, text="Convert", fg="black", command=self.function)  
 self.convert\_button.config(font=('Courier', 10, 'bold'))  
 self.convert\_button.place(x=225, y=135)  
  
 # Placement  
 self.crypto\_label.place(x=24, y=90)  
 self.fiat\_label.place(x=337, y=90)  
 self.cryto\_currency\_dropdown.place(x=30, y=120)  
 self.fiat\_currency\_dropdown.place(x=335, y=120)  
 self.amount\_field.place(x=37, y=150)  
 self.converted\_amount\_field.place(x=342, y=150)  
  
  
 # Exit button  
 self.exit\_button = Button(self, text="Exit", fg="black", command=self.destroy)  
 self.exit\_button.place(x=245, y=325)  
  
 # defining the exit button  
 def Close(self):  
 self.destroy()  
  
 # setting up protection in entry field (only numbers can be entered)  
 def restrictNumberOnly(action, string):  
 regex = re.compile(r"[0-9,]\*?(\.)?[0-9,]\*$")  
 result = regex.match(string)  
 return string == "" or (string.count('.') <= 1 and result is not None)  
  
 # defining the fiat and crypto conversion amount functions  
 def function(self):  
 amount = float(self.amount\_field.get())  
 crypto\_curr = self.crypto\_currency\_variable.get()  
 fiat\_curr = self.fiat\_currency\_variable.get()  
  
 if crypto\_curr == fiat\_curr:  
 converted\_amount = amount  
  
 else:  
  
 converted\_amount = CurrencyConverter.convert(crypto\_curr, fiat\_curr, amount)  
 if converted\_amount == -1:  
 self.converted\_amount\_field.config(text='invalid currency code')  
 else:  
 converted\_amount = round(converted\_amount, 2)  
  
 self.converted\_amount\_field.config(text=str(converted\_amount))  
  
  
# running the loop  
if \_\_name\_\_ == '\_\_main\_\_':  
 App()  
 mainloop()

Instructions:

I’ve included a README.txt file with instructions in the zip folder.

1. Unzip folder and extract it to a location
2. Open the .py file with IDLE Python 3.0 or higher
3. Run the .py file with IDLE
4. First Window is Log in Credentials
5. Enter in the following for the Username: username
6. Enter in the following for the Password: password
7. Message will appear in terminal if either password or username is incorrect
8. Message in terminal will also so the username and password entered in the window
9. With the mouse click the ‘X’ on the top right corner if log in credentials were correct
10. Next Window is the Crypto Converter to Fiat Currency
11. Drop down on the Crypto Currency to select from (SHIB,DOGE,BTC, or ELON)
12. Drop down on the Fiat Currency to select from (USD, EUR, or XAU)
13. Enter in the text box field for crypto currency the amount to convert
14. Click the ‘Convert’ button
15. Amount of Fiat currency will appear in the text box
16. Change the amount of Crypto Currency to convert or drop down to change currencies
17. Repeat steps until finished
18. Click ‘Exit’ to end the program

ZipFolder

1. Make sure to unzip the folder and extract with the txt file and images included
2. To change or add a different Fiat or Crypto Currency see below

Text File:

1. To add or delete a fiat currency or cryptocurrency
2. Open the txt file ‘cryptocurrency.txt’
3. Add different cryptos under CRYPTO
4. Add different fiat’s under FIAT
5. Must use symbols correct Currency Symbols for either
6. See [All Cryptocurrencies | CoinMarketCap](https://coinmarketcap.com/all/views/all/) for list of symbols

After the completion of the project, I’ve added the zip folder to GitHub repositories

Under [dhans86/TopDogCryptoConverter: Converts the price of USD for the top three meme coin/tokens (SHIBA INU, DOGECOIN, & DOGELONG MARS) (github.com)](https://github.com/dhans86/TopDogCryptoConverter)

Testing the code screenshots:

Graphical user interface, text, application, email

Description automatically generatedGraphical user interface, text, application

Description automatically generatedGraphical user interface, application, Teams

Description automatically generatedGraphical user interface, application

Description automatically generatedGraphical user interface, application, Teams

Description automatically generated