

《用 Python 玩转数据》财经数据 GUI 项目

Dazhuang@NJU

尝试实现 7.8 中所述的项目

【参考代码见下一页】

PS：包含两个文件：my_finance.py 和 dji_wxPython.py

```
# Filename: my_finance.py
```

```
# -*- coding: utf-8 -*-
```

```
"""
```

```
get DJI data
```

```
@author: Dazhuang
```

```
"""
```

```
import json
```

```
import re
```

```
import requests
```

```
def retrieve_dji_list():
```

```
    try:
```

```
        r = requests.get('http://money.cnn.com/data/dow30/')
```

```
    except ConnectionError as err:
```

```
        print(err)
```

```
    search_pattern =
```

```
re.compile('class="wsod_symbol">(.*?)</a>.*<span.*">(.*?)</span>.*\n.*class="wsod_stream'>(.*?)</span>')>(.*?)</span>')>(.*?)</span>')
```

```
    dji_list_in_text = re.findall(search_pattern, r.text)
```

```
    dji_list = []
```

```
    for item in dji_list_in_text:
```

```
        dji_list.append({'code': item[0], 'name': item[1], 'price': float(item[2])})
```

```
    return dji_list
```

```
def retrieve_quotes_historical(stock_code, start = '', end = ''):
```

```
    quotes = []
```

```
    url = 'https://finance.yahoo.com/quote/%s/history?p=%s' % (stock_code, stock_code)
```

```
    try:
```

```
        r = requests.get(url)
```

```
    except ConnectionError as err:
```

```
        print(err)
```

```
    m = re.findall('\"HistoricalPriceStore\":{\"prices\":(.*?),\"isPending\"', r.text)
```

```
    if m:
```

```
        quotes = json.loads(m[0])
```

```
        quotes = quotes[::-1]
```

```
    return [item for item in quotes if not 'type' in item]
```

```
# Filename: dji_wxPython.py
```

```
# -*- coding: utf-8 -*-
```

```
"""
```

```
wxPython plot
```

```
@author: Dazhuang
```

```
"""
```

```
import datetime as dt
```

```
import my_finance as finance
```

```
import matplotlib.pyplot as plt
```

```
import pandas as pd
```

```
import _thread as thread
```

```
import wx
```

```
ID_EVENT_REFRESH = 9999
```

```
class StockFrame(wx.Frame):
```

```
    option_list = {'open': True, 'close': True, 'high': False, 'low': False, 'volume': False}
```

```
    def __init__(self, title):
```

```
        wx.Frame.__init__(self, None, title=title, size=(430,600))
```

```
        self.CreateStatusBar()
```

```
        menuBar = wx.MenuBar()
```

```
        filemenu= wx.Menu()
```

```
        menuBar.Append(filemenu,"&File")
```

```
        menuRefresh = filemenu.Append(ID_EVENT_REFRESH, "&Refresh", "Refresh the price")
```

```
        self.Bind(wx.EVT_MENU, self.OnRefresh, menuRefresh)
```

```
        menuQuit = filemenu.Append(wx.ID_EXIT, "Q&uit", "Terminate the program")
```

```
        self.Bind(wx.EVT_MENU, self.OnQuit, menuQuit)
```

```
        self.SetMenuBar(menuBar)
```

```
        panel = wx.Panel(self)
```

```
        codeSizer = wx.BoxSizer(wx.HORIZONTAL)
```

```
        labelText = wx.StaticText(panel, label="Stock Code:")
```

```
        codeSizer.Add(labelText, 0, wx.ALIGN_BOTTOM)
```

```
        # TODO: need a better way to put a spacer here than this:
```

```
        # codeSizer.Add((10, 10))
```

```
        codeText = wx.TextCtrl(panel, value='BA', style=wx.TE_PROCESS_ENTER)
```

```
        self.Bind(wx.EVT_TEXT_ENTER, self.OnTextSubmitted, codeText)
```

```
        codeSizer.Add(codeText)
```

```

optionSizer = wx.BoxSizer(wx.HORIZONTAL)
for key, value in self.option_list.items():
    checkBox = wx.CheckBox(panel, label = key.title())
    checkBox.SetValue(value)
    self.Bind(wx.EVT_CHECKBOX, self.OnChecked)
    optionSizer.Add(checkBox)

self.list = wx.ListCtrl(panel, wx.NewId(), style=wx.LC_REPORT)
self.createHeader()

pos = self.list.InsertItem(0, "--")
self.list.SetItem(pos, 1, "loading...")
self.list.SetItem(pos, 2, "--")
self.Bind(wx.EVT_LIST_ITEM_ACTIVATED, self.OnDoubleClick, self.list)

ctrlSizer = wx.BoxSizer(wx.HORIZONTAL)
ctrlSizer.Add((10, 10))

buttonQuit = wx.Button(panel, -1, "Quit")
self.Bind(wx.EVT_BUTTON, self.OnQuit, buttonQuit)
ctrlSizer.Add(buttonQuit, 1)
buttonRefresh = wx.Button(panel, -1, "Refresh")
self.Bind(wx.EVT_BUTTON, self.OnRefresh, buttonRefresh)
ctrlSizer.Add(buttonRefresh, 1, wx.LEFT | wx.BOTTOM)

sizer = wx.BoxSizer(wx.VERTICAL)
sizer.Add(codeSizer, 0, wx.ALL, 5)
sizer.Add(optionSizer, 0, wx.ALL, 5)
sizer.Add(self.list, -1, wx.ALL | wx.EXPAND, 5)
sizer.Add(ctrlSizer, 0, wx.ALIGN_BOTTOM)

panel.SetSizerAndFit(sizer)
self.Center()

# start loading data right after the window comes up
self.OnRefresh(None)

def createHeader(self):
    self.list.InsertColumn(0, "Symbol")
    self.list.InsertColumn(1, "Name")
    self.list.InsertColumn(2, "Last Trade")

def setData(self, data):

```

```

self.list.ClearAll()
self.createHeader()
pos = 0
for row in data:
    pos = self.list.InsertItem(pos + 1, row['code'])
    self.list.SetItem(pos, 1, row['name'])
    self.list.SetColumnWidth(1, -1)
    self.list.SetItem(pos, 2, str(row['price']))
    if pos % 2 == 0:
        # Set new look and feel for odd lines
        self.list.SetItemBackgroundColour(pos, (134, 225, 249))

def PlotData(self, code):
    quotes = finance.retrieve_quotes_historical(code)
    fields = ['date', 'open', 'close', 'high', 'low', 'volume']
    dates = []
    for i in range(0, len(quotes)):
        x = dt.datetime.utcfromtimestamp(int(quotes[i]['date']))
        y = dt.datetime.strftime(x, '%Y-%m-%d')
        dates.append(y)
    quotesdf = pd.DataFrame(quotes, index = dates, columns = fields)

    # remove unchecked fields
    fields_to_drop = ['date']
    for key, value in self.option_list.items():
        if not value:
            fields_to_drop.append(key)

    quotesdf = quotesdf.drop(fields_to_drop, axis = 1)
    quotesdf.plot()
    plt.show()

def OnDoubleClick(self, event):
    self.PlotData(event.GetText())

def OnTextSubmitted(self, event):
    self.PlotData(event.GetString())

def OnChecked(self, event):
    checkBox = event.GetEventObject()
    text = checkBox.GetLabel().lower()
    self.option_list[text] = checkBox.GetValue()

def OnQuit(self, event):

```

```
        self.Close()
        self.Destroy()

    def OnRefresh(self, event):
        thread.start_new_thread(self.retrieve_quotes, ())

    def retrieve_quotes(self):
        data = finance.retrieve_dji_list()
        if data:
            self.setData(data)
        else:
            wx.MessageBox('Download failed.', 'Message', wx.OK | wx.ICON_INFORMATION)

if __name__ == '__main__':
    app = wx.App(False)
    top = StockFrame("Dow Jones Industrial Average (^DJI)")
    top.Show(True)
    app.MainLoop()
```