SYVO

PYTHON PROJECT

A thesis submitted in partial fulfilment of the requirements

for the degree of B.Tech. Computer Science and Engineering

NOV 2021

By

Vinod Kumar

(Roll No: 120CS0008)

Himanshu Mahaur

(Roll No: 120CS0007)

Abstract

SYVO (save youtube videos offline) is a software made using python, which allows you to download youtube videos to your devices for offline consumption.

Reasons for making SYVO

- There are some situations when we are unable to access internet connections,in such cases watching online youtube videos is not possible.
- Youtube videos generally have ads,which is undesirable.
- Sharing online Youtube videos is can be hassle sometimes

*Our software (SYVO) allows us to solve all these problems easily.

Why is SYVO better?

- Other Youtube video downloaders are paid while ours(SYVO) is free and open source.
- No virus and Spywares
- No limitation on number of downloads.

Functions of SYVO

- Download Youtube videos using their URL.
- Select the quality of downloading the video.
- Download the files in audio only format.
- Browse button, for changing save file directory.
- Pop up window on download completion.

Acknowledgement

We would like to express our special thanks of gratitude to our professor Dr.K.Nagaraju Sir as well as our Director sir Dr.DVLN Somayajulu sir, who gave us the golden opportunity to do this wonderful python project on the topic YouTube video downloader, which also helped us in doing a lot of research and we came to know about so many new things we are really thankful to them.

Secondly we would also like to thank our friends who helped us a lot in finalizing this project within the limited time frame.

Contents

Introduction

We(Himanshu and Vinod) are assigned to do a python project by Dr.K.Nagaraju Sir,we were aware of what we can do,and what our capabilities are, so we decided to do a youtube video downloader using python.
Youtube is a free video sharing website that makes it easy to watch online videos. You can even create and upload your own videos to share with others. Originally created in 2005, YouTube is now one of the most popular sites on the Web, with visitors watching around 6 billion hours of video every month.

Save Youtube Videos Offline (SYVO) is an application to download videos from YouTube. This provides users to download videos they need on their devices and watch them offline.
The SYVO is a python project. The object of this project is to download any type of video in a fast and easy way from youtube on your device.

In this python project, users have to copy the youtube video URL that they want to download and simply paste that URL in SYVO. click on the download button, it will start downloading the video.when the video is downloading it shows progress bar(how much it is downloaded). When video downloading finishes, it shows a message ‘downloaded’ popup on the window.

WHY DID WE BUILD SYVO?

PROBLEM

Streaming videos has become a way of life these days. From YouTube to Facebook, Hulu and Netflix, streaming videos are everywhere. But the thing with streaming videos is the fact that you need to be online to do so.

You could argue that everybody is always connected via their smartphones and tablets or the fact that Wi-Fi is now offered practically anywhere, but the truth is, there are times when people don’t have Internet access.

SOLUTION

We made SYVO so that you can easily enjoy your favorite videos from YouTube, even without a working internet connection.

Since YT videos are not in your possession there is always a possibility that it might get deleted, by saving it offline you can ensure that you can access it anytime you want.
You might have come across YouTube ads while watching videos online, SYVO eliminates this issue.

SYVO allows you to save audio only files, which YouTube doesn't provide by default.

Our Project started with the vision to solve most of these problems.

Description

Project Prerequisites

- To implement this project we used the basic concepts of python, tkinter, threading, pillow and pytube library.
- Tkinter is a standard GUI library and it is one of the easiest ways to build a GUI application.
- Pytube used for downloading videos from youtube
- To install the required modules run pip installer command on the command line:

  pip install pillow
  pip install tk
  pip install pytube

Roadmap for SYVO development

1. Installing required libraries on workbench
2. Writing terminal code, and check its working
3. Enabling user input in the software
4. Adding GUI elements to the software
5. Adding quality of life improvements, like dark mode etc.

Libraries

Tkinter:

Tkinter is the de facto way in Python to create Graphical User interfaces (GUIs) and is included in all standard Python Distributions. In fact, it's the only framework built into the Python standard library.
This Python framework provides an interface to the Tk toolkit and works as a thin object-oriented layer on top of Tk. The Tk toolkit is a cross-platform collection of graphical control elements, aka widgets, for building application interfaces.

Pytube:

Pytube is a lightweight library written in Python. It has no third party dependencies and aims to be highly reliable.
pytube also makes pipelining easy, allowing you to specify callback functions for different download events, such as on progress or on complete.

Finally pytube also includes a command-line utility, allowing you to quickly download videos right from the terminal.

Pillow:

Pillow is a Python Imaging Library (PIL), which adds support for opening, manipulating, and saving images. The current version identifies and reads a large number of formats. Write support is intentionally restricted to the most commonly used interchange and presentation formats.

Threading:

Threading in python is used to run multiple threads (tasks, function calls) at the same time. Note that this does not mean that they are executed on different CPUs. Python threads will NOT make your program faster if it already uses 100 % CPU time.

Python threads are used in cases where the execution of a task involves some waiting. One example would be interaction with a service hosted on another computer, such as a web server. Threading allows python to execute other code while waiting; this is easily simulated with the sleep function.

Evaluation

Problems while making SYVO and solutions

- ## GUI becomes unresponsive while video is being downloaded.

    - **Since tkinter works on a single thread, when we call our download function the GUI becomes unresponsive and progressbar fails to work.**

        We used python multithreading to solve the problem, when the download function is called a new thread is assigned to execute it, meanwhile the GUI remains responsive, hence our problem is solved.

- ## Issues while implementing dark mode.

    - **In order to implement darkmode, we need to have mostly darker colours in the background. But, radio buttons don't support (.config) to change its background color.**

        To solve this problem we declared a separate set of buttons with required styling, and packed out the old set of buttons and replaced them with the new ones when required.

- ## Interference in progressbar.

    - **When the progress bar reaches over 50% completion, the numerical label from progress function interferes with the graphical indication.**

        In the progress function we made a trigger such that when progress of the download bar reaches above 50% we change the background colour of the label to the same hex colour of the windows default green. Which eliminates the problem

Conclusion and Future work

Even though we have reached our primary goals for this project, we are not done yet. There is still a lot more work to be done, and many more features to add.

Ideas

- We will try to add support for more resolutions to download
- We will work on custom theme option
- Try to add support to some more websites like daily motion etc.
- We will try to add search a video or URL in SYVO itself
- We will work on multi platform support like in Android etc.

and much more...

Source Code

- ## Importing libraries

```
#GUI libraries
from tkinter import *
from tkinter import ttk
from tkinter import messagebox
from tkinter import filedialog

#supporting libraries for tkinter
from PIL import ImageTk, Image

#library for multithreading operation
from threading import *

#library for youtube handelings
from pytube import YouTube
```

- ## Design of root window (GUI)

```
#root window GUI design
root = Tk()
root.configure(bg='teal')
root.geometry('758x600')
root.minsize(758, 600)
root.maxsize(758, 600)

root.title('SYVO - DOWNLOADER')
root.iconbitmap("C:/Users/himan/Desktop/CODE/python/SYVO/icon.ico")

logoImage =
ImageTk.PhotoImage(Image.open("C:/Users/himan/Desktop/CODE/python/SYVO/logo.png"))
logoLabel = Label(image=logoImage, bg='teal')
logoLabel.grid(row=1, column=1, padx=10, pady=0, columnspan=2)

frameLeft = LabelFrame(root, text='Stream', padx=50, pady=56, bg='teal', fg='white',
font='Helvetica 9 bold')
frameLeft.grid(row=2, column=1, padx=10, pady=0)

frameRight = LabelFrame(root, text='Quality', padx=50, pady=50, bg='teal', fg='white',
font='Helvetica 9 bold')
frameRight.grid(row=2, column=2, padx=0, pady=0)
```

```
frameRight1 = LabelFrame(root, text='Quality', padx=50, pady=50, bg='black', fg='white',
font='Helvetica 9 bold')

framebutton = LabelFrame(root, bg='teal', borderwidth=0)
framebutton.grid(row=0, column=2, columnspan=1, padx=50)
```

- **Entry widget (GUI)**

```
#entry, GUI - URL
linkLabel = Label(frameLeft, text='Link', bg='teal', fg='#ffffff', font='Helvetica 9 bold')
linkLabel.grid(row=1, column=1, padx=5, pady=10)

urlEntry = Entry(frameLeft, width=50, borderwidth=0, highlightthickness=1)
urlEntry.grid(row=1, column=2, padx=10, pady=10)


#entery, GUI - path
savePath = StringVar()

pathLabel = Label(frameLeft, text='Path', bg='teal', fg='#ffffff', font='Helvetica 9 bold')
pathLabel.grid(row=2, column=1, padx=5, pady=10)

pathEntry = Entry(frameLeft, text=savePath, width=50, borderwidth=0, highlightthickness=1)
pathEntry.grid(row=2, column=2, padx=10, pady=10)
```

- **Selection of video quality**

```
#stream quality function
quastm = IntVar()
quastm.set(22)
itag = 22

def func_quality(value):
    global itag
    itag = str(value)

    return 0

button720p = Radiobutton(frameRight, text='720p', variable=quastm, value=22,
command=lambda: func_quality(quastm.get()), bg='teal', fg='black', font='Helvetica 9
bold').pack()
button360p = Radiobutton(frameRight, text='360p', variable=quastm, value=18,
command=lambda: func_quality(quastm.get()), bg='teal', fg='black', font='Helvetica 9
bold').pack()
button1440p = Radiobutton(frameRight, text='144p', variable=quastm, value=17,
```

```
command=lambda: func_quality(quastm.get()), bg='teal', fg='black', font='Helvetica 9
bold').pack()
buttonAudio = Radiobutton(frameRight, text='Audio', variable=quastm, value=251,
command=lambda: func_quality(quastm.get()), bg='teal', fg='black', font='Helvetica 9
bold').pack()

button720p = Radiobutton(frameRight1, text='720p', variable=quastm, value=22,
command=lambda: func_quality(quastm.get()), bg='black', fg='teal', font='Helvetica 9
bold').pack()
button360p = Radiobutton(frameRight1, text='360p', variable=quastm, value=18,
command=lambda: func_quality(quastm.get()), bg='black', fg='teal', font='Helvetica 9
bold').pack()
button1440p = Radiobutton(frameRight1, text='144p', variable=quastm, value=17,
command=lambda: func_quality(quastm.get()), bg='black', fg='teal', font='Helvetica 9
bold').pack()
buttonAudio = Radiobutton(frameRight1, text='Audio', variable=quastm, value=251,
command=lambda: func_quality(quastm.get()), bg='black', fg='teal', font='Helvetica 9
bold').pack()
```

- **Select path to save file**

```
#browse directory function
def func_browse():
    global savePath
    PATH = filedialog.askdirectory()
    savePath.set(PATH)

    return 0
browseButton = Button(frameLeft, text='Browse', command=func_browse, bg='#73cbab',
fg='black', font='Helvetica 9 bold')
browseButton.grid(row=2, column=3, padx=10, pady=10)
```

- **Registering the progress of stream**

```
#progress register function
def func_progress(chunk, fh, bytes_remaining):
    global dProgress

    streamSize = stream.filesize
    dProgress = int(((streamSize - bytes_remaining) / streamSize) * 100)

    if(int(dProgress)>50):
        progresslabel.config(background="#06b025")

    progressbar['value']=dProgress
    gProgress = (str(dProgress)+"%")
    progresslabel.config(text=gProgress)
```

- **Pop-up after completion**

```
#completion register function
def func_complete(self, file_path):
    messagebox.showinfo('Status', 'Download Complete!')
```

- **Download function**

```
#pytube download function >>[itag - from func_quality(), URL, dLocation]<<
def func_download():
    global stream

    progresslabel.config(background="white")

    messagebox.showinfo('Status', 'Downloading..')
    progressbar['value']=0
    progresslabel.config(text="0%")

    URL = urlEntry.get()
    dLocation = pathEntry.get()

    yt = YouTube(str(URL), on_progress_callback=func_progress,
on_complete_callback=func_complete)
    stream = yt.streams.get_by_itag(int(itag))
    stream.download(str(dLocation))

    return 0
```

- **Threading**

```
#to make the program multi-threaded
def threading():
    # Call work function
    t1=Thread(target=func_download)
    t1.start()
```

- **Themes**

```
#(LIGHT THEME) changing config of different widgets depending upon theme
def light_theme():
    color = 'teal'
    color2 = 'white'

    root.config(bg=color)
    logoLabel.config(bg=color)

    frameLeft.config(bg=color, fg=color2)

    linkLabel.config(bg=color, fg=color2)
    urlEntry.config(highlightbackground=color2)

    pathLabel.config(bg=color, fg=color2)
    pathEntry.config(highlightbackground=color2)

    downloadButton.config(bg='#73cbab', fg='black')
    browseButton.config(bg='#73cbab', fg='black')

    #removing frameright of darktheme
    frameRight1.grid_forget()
    #adding frameright of lighttheme
    frameRight.grid(row=2, column=2, padx=0, pady=0)

    Def_Btns.config(bg='teal')
    Def_Btnm.config(bg='teal')


#(DARK THEME) changing config of different widgets depending upon theme
def dark_theme():
    color = 'black'
    color2 = 'white'

    root.config(bg=color)
    logoLabel.config(bg=color)

    frameLeft.config(bg=color, fg=color2)

    linkLabel.config(bg=color, fg=color2)
    urlEntry.config(highlightbackground=color2)

    pathLabel.config(bg=color, fg=color2)
    pathEntry.config(highlightbackground=color2)

    downloadButton.config(bg='#73cbab', fg='black')
    browseButton.config(bg='#73cbab', fg='black')

    #removing frameright of lighttheme
```

```
frameRight.grid_forget()
#adding frameright of darktheme
frameRight1.grid(row=2, column=2, padx=0, pady=0)

Def_Btns.config(bg='black')
Def_Btnm.config(bg='black')
```

- **Progress bar and %**

```
#download status (GUI) (depends upon callback of pytube)
progressbar = ttk.Progressbar(root, orient='horizontal', mode='determinate', length=550)
progressbar.grid(column=1, row=3, columnspan=4, padx=10, pady=0)


#download status (percentage)
progresslabel = ttk.Label(root, text="")
progresslabel.grid(column=1, row=3, columnspan=4, padx=10, pady=30)
```

- **More GUI elements for calling functions**

```
#download button (calling threading func)
downloadButton = Button(root, text='Download', command=threading, bg='#73cbab', fg='black',
font='Helvetica 13 bold', height=1, width=25)
downloadButton.grid(row=5, column=1, columnspan=2, pady=0)


#GUI elements for theme selection
lightimage = PhotoImage(file='C:/Users/himan/Desktop/CODE/python/SYVO/sun.png')
darkimage = PhotoImage(file='C:/Users/himan/Desktop/CODE/python/SYVO/moon.png')


#buttons for switching betweeen light and dark theme (calling respective functions) (theme
selection)
Def_Btns = Button(framebutton, bg='teal', image=lightimage, command=light_theme,
borderwidth=0)
Def_Btns.grid(row=0, column=0)

Def_Btnm = Button(framebutton, bg='teal',image=darkimage, command=dark_theme,
borderwidth=0)
Def_Btnm.grid(row=0, column=1)

root.mainloop()
```

Bibliography

1. **Tkinter tutorial by Freecodecamp**

   https://www.youtube.com/watch?v=YXPyB4XeYLA

2. **Official Tkinter documentation**

   https://docs.python.org/3/library/tkinter.html

3. **Pytube official website**

   https://pytube.io/en/latest/

4. **Pillow tutorial by PyMoondra**

   https://www.youtube.com/watch?v=dkp4wUhCwR4

5. **Geeks for geeks**

   https://www.geeksforgeeks.org/python-programming-language/

6. **Stackoverflow.com**

   https://stackoverflow.com/questions/tagged/python