

video-z-youtube-rozliseni

Návrh řešení: Aplikace pro stahování videí z YouTube s možností konvertování na jiný formát a s grafickým rozhraním pro uživatele. Pro tuto aplikaci budeme používat knihovny „tkinter“, „pytube“, „ffmpeg“ a „os“.

Nejprve si vytvoříme objekt okna a nastavíme parametry:

```
root = Tk()
root.title("Video downloader/converter")
root.geometry("850x420")
root.resizable(False, False)
root.configure(background=bg)
```

Vytvoříme objekty labelů:

```
# Labels
video_label = Label(root, text="Enter video link: ", bg=bg)
resolution_label = Label(root, text="Choose video resolution?", bg=bg)
download_format_label = Label(root, text="Choose format?", bg=bg)
bar_label = Label(root, text="Download bar: ", bg=bg)
convert_label = Label(root, text="Choose file for conversion: ", bg=bg)
format_label = Label(root, text="Choose file format?", bg=bg)
```

Deklarujeme proměnné:

```
# Vars
option_resolution = StringVar()
option_type = StringVar()
option_format = StringVar()
selected_file_path = StringVar()
download_path = StringVar()

# Entry
video_entry = Entry(root, bd=5, width=64)
```

Vytvoříme objekty tlačítek:

```
# Download Objects
BAR1 = ttk.Progressbar(root, orient="horizontal", mode="determinate", length=400, value=0, maximum=100)

R1 = Radiobutton(root, text="360p", value="360p", var=option_resolution, bg=bg)
R2 = Radiobutton(root, text="480p", value="480p", var=option_resolution, bg=bg)
R3 = Radiobutton(root, text="720p", value="720p", var=option_resolution, bg=bg)

R4 = Radiobutton(root, text="Audio", value="Audio", var=option_type, bg=bg)
R5 = Radiobutton(root, text="Video", value="Video", var=option_type, bg=bg)

B1 = Button(root, text="Select download folder", command=browse, bg=bg)
B2 = Button(root, text="Download", command=load, bg=bg)

# Convert Objects
R6 = Radiobutton(root, text="mp4", value=".mp4", var=option_format, bg=bg)
R7 = Radiobutton(root, text="mp3", value=".mp3", var=option_format, bg=bg)
R8 = Radiobutton(root, text="wav", value=".wav", var=option_format, bg=bg)

B3 = Button(root, text="Select file for conversion", command=select, bg=bg)
B4 = Button(root, text="Convert", command=convert, bg=bg)
```

Nastavíme proměnné:

```
# Set Variables
option_resolution.set(" ")
option_type.set(" ")
option_format.set(" ")
```

Vykreslíme všechny objekty do našeho okna:

```
# Download Grids
video_label.grid(row=1, column=0, padx=5, pady=5)
video_entry.grid(row=1, column=1, padx=5, pady=5)
resolution_label.grid(row=20, column=0, padx=5, pady=5)
download_format_label.grid(row=39, column=0, padx=5, pady=5)
R1.grid(row=30, column=0, padx=5, pady=5)
R2.grid(row=30, column=1, padx=5, pady=5)
R3.grid(row=30, column=2, padx=5, pady=5)
R4.grid(row=40, column=0, padx=5, pady=5)
R5.grid(row=40, column=1, padx=5, pady=5)
B1.grid(row=50, column=0, padx=5, pady=5)
B2.grid(row=60, column=0, padx=5, pady=5)
bar_label.grid(row=70, column=0, padx=5, pady=5)
BAR1.grid(row=70, column=1, columnspan=2, padx=5, pady=5)

# Convert Grids
convert_label.grid(row=80, column=0, padx=5, pady=5)
R6.grid(row=90, column=0, padx=5, pady=5)
R7.grid(row=90, column=1, padx=5, pady=5)
R8.grid(row=90, column=2, padx=5, pady=5)
B3.grid(row=80, column=1, padx=5, pady=5)
B4.grid(row=100, column=0, padx=5, pady=5)
```

Vytvoříme funkci pro vybrání uložistiště:

```
def browse():
    download_directory = filedialog.askdirectory(initialdir="", title="Save video")
    download_path.set(download_directory)
    path = download_path.get()
    download_directory_label = Label(root, text=path, bg=bg)
    download_directory_label.grid(row=50, column=1, padx=5, pady=5)
```

Dále vytvoříme funkci pro výběr souboru pro konverzi:

```
def select():
    global convert_directory_label
    global convert_directory_path_label
    selected_file = filedialog.askopenfilename(initialdir="", filetypes=[("mp3", "*.mp3"), ("mp4", "*.mp4"), ("wav", "*.wav")], title="Select a file")
    selected_file_path.set(selected_file)
    path = selected_file_path.get()
    convert_directory_path_label = Label(root, text=path, bg=bg)
    convert_directory_label = Label(root, text="File selected for conversion: ", bg=bg)
    convert_directory_label.grid(row=81, column=0, padx=5, pady=5)
    convert_directory_path_label.grid(row=81, column=1, padx=5, pady=5)
```

Vytvoříme funkci pro vizualizaci stavu stahování:

```
def perc_function(x,y):
    total = ((x-y)/x)*100
    return total

def pbar(stream, chunk, bytes_remaining):
    file_size = stream.filesize
    remaining = bytes_remaining
    perc = perc_function(file_size, remaining)
    BAR1["value"] = int(perc)
    # add label
    root.update()
```

Po vykreslení a spuštění aplikace by měla vypadat přibližně takto:



Vytvoříme funkci pro ošetření errorů:

```
def error(x):  
    if x == 1:  
        return messagebox.showerror("LINK ERROR", "MISSING OR BROKEN LINK")  
    if x == 2:  
        return messagebox.showerror("VIDEO ERROR", "VIDEO IS RESTRICTED OR NOT IN SELECTED QUALITY\nTRY LOWERING QUALITY")  
    if x == 3:  
        return messagebox.showerror("PATH ERROR", "CHOOSE DOWNLOAD FOLDER")  
    if x == 4:  
        return messagebox.showerror("FILE ERROR", "CHOOSE FILE FOR CONVERSION")  
    if x == 5:  
        return messagebox.showerror("FORMAT ERROR", "FILE IS ALREADY IN SELECTED FORMAT")  
    if x == 6:  
        return messagebox.showerror("FORMAT ERROR", "CHOOSE FORMAT")
```

Vytvoříme hlavní funkci pro stahování:

```

# Download
def load():

    video = video_entry.get()
    resolution = option_resolution.get()
    type = option_type.get()
    download_folder = download_path.get()

    try:
        yt = YouTube(video, on_progress_callback=pbar)
        try:
            if type=="Audio":
                stream = yt.streams.get_audio_only(subtype="mp4")

            if type=="Video":
                stream = yt.streams.filter(res=resolution).first()

            try:
                if download_folder != "":
                    stream.download(download_folder)
                    messagebox.showinfo("DONE", "Download complete")
                    BAR1["value"] = 0

                    return

            except:
                error(3)
        except:
            error(2)
    except:
        error(1)

```

A hlavní funkci pro konverzi:

```

# Convert
def convert():
    try:
        file = selected_file_path.get()
        # Renaming and splitting file
        old_file_name = file
        new_file_name = file.replace(" ", "_")
        file_basename = os.path.basename(file)
        os.rename(old_file_name, new_file_name)
        file_name, file_format = os.path.splitext(file_basename)
        #
        format = option_format.get()
    except:
        try:
            # Same format error
            if file_format == format:
                return error(5)

            # MP4 into MP3
            if format == ".mp3" and file_format == ".mp4":
                os.system(f"FFmpeg -i {new_file_name} {new_file_name[:-4]}.mp3")
                os.rename(new_file_name, old_file_name)

            if format == ".wav" and file_format == ".mp4":
                os.system(f"FFmpeg -i {new_file_name} {new_file_name[:-4]}.wav")
                os.rename(new_file_name, old_file_name)

            # MP3 into WAV
            if format == ".wav" and file_format == ".mp3":
                os.system(f"FFmpeg -i {new_file_name} -acodec pcm_u8 -ar 22050 {new_file_name[:-4]}.wav")
                os.rename(new_file_name, old_file_name)

            if format == ".mp4" and file_format == ".mp3":
                os.system(f"FFmpeg -i {new_file_name} {new_file_name[:-4]}.mp4")
                os.rename(new_file_name, old_file_name)

            if format == ".mp4" and file_format == ".wav":
                os.system(f"FFmpeg -i {new_file_name} {new_file_name[:-4]}.mp4")
                os.rename(new_file_name, old_file_name)

            if format == ".mp3" and file_format == ".wav":
                os.system(f"FFmpeg -i {new_file_name} -vn -ar 44100 -ac 2 -b:a 192k {new_file_name[:-4]}.mp3")
                os.rename(new_file_name, old_file_name)
            convert_directory_label.destroy()
            convert_directory_path_label.destroy()
            return messagebox.showinfo("DONE", "Conversion complete")
        except:
            error(6)
    except:
        error(4)

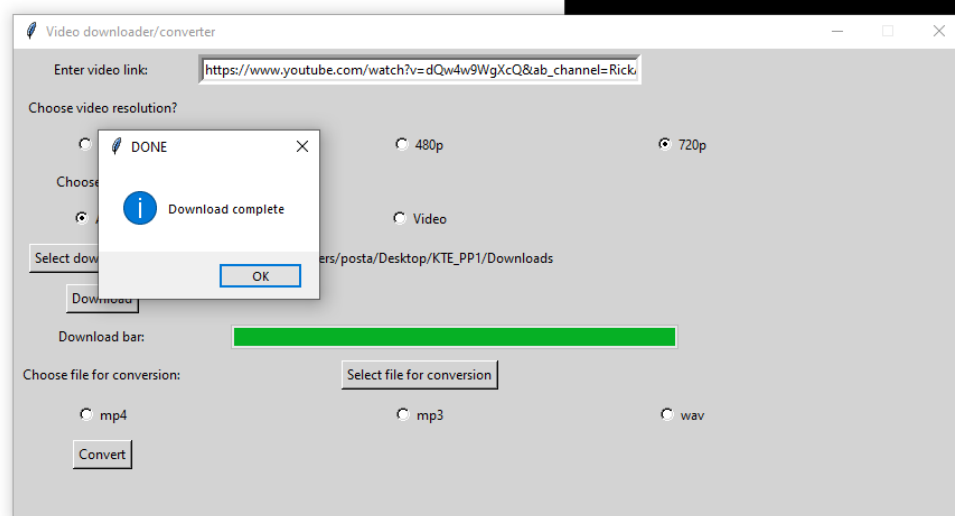
```

Výsledky a fakta:

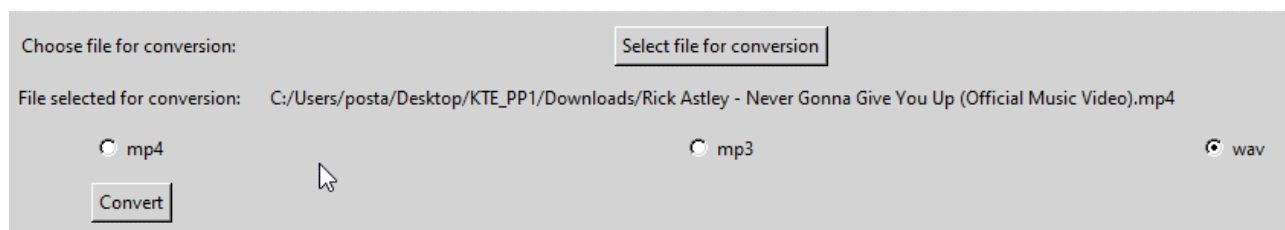
Nyní po spuštění sestavené aplikace, vyplnění všech potřebných kolonek a spuštění stahování nám aplikace stáhne dané video/audio do vybrané složky:



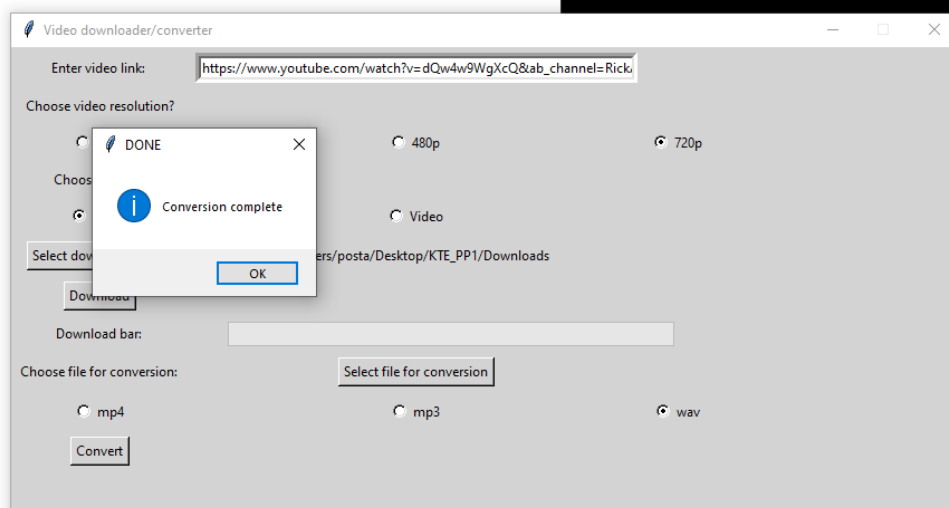
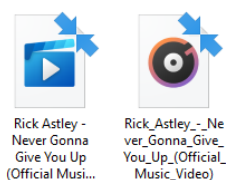
Rick Astley -
Never Gonna
Give You Up
(Official Musi...



Poté můžeme ihned využít konvertor, vybrat stažený soubor a převést jej například z formátu .mp4 na formát .wav:



Po kliknutí na „convert“ tlačítko, nám aplikace konvertuje vybraný .mp4 soubor na .wav soubor a uloží jej do stejné složky:



Závěr: Ukázali jsme si, že aplikace funguje bez žádných větších problémů. U pomalejších PC/Laptopů může dojít na chvíli k „zmrazení“ aplikace po kliknutí na tlačítko „download“, ale po krátké chvíli by měla jet zase plynule. Knihovna Pytube je velice dobře využitelná a díky ní, spolu s knihovnou Tkinter, se podařilo vytvořit i grafický ukazatel postupu stahování. Knihovna ffmpeg je výbornou volně dostupnou knihovnou pro konverzi souborů, avšak pro někoho, kdo nemá znalosti audio/video, je konverze poměrně náročná. Je totiž třeba nastavit správné kodeky, bitrate a další parametry.