palani03-DPK-protokol

October 28, 2024

DPK Didaktika programování v kurikulu ZŠ - Protokol
Téma: SoundCloud Downloader
Jméno a příjmení studenta: Nicolas Palán
Datum vypracování: 28. října 2024
Vyučující: doc. RNDr. Petr Šaloun, Ph.D., KTIV, PdF UPOL

1. Témata k prostudování
Programování v jazyce Python

Česká stránka o programování v jazyce Python

Tvroba GUI s využitím knihovny Tkinter

IDE Jupyter Notebook

IDE Google Colab

Anaconda Cloud

2. Pokyny

Vypracování Protokolu 2 zahrnuje popis významných částí programu, ukázku jeho editace a případný výstup (myšleny jsou screenshoty programu ve vývojovém prostředí a okna zobrazujícího vstupní a výstupní hodnoty převodů). Vlastní převody mezi uvedenými číselnými soustavami realizujte výhradně vlastními funkcemi – není dovoleno využití přímé konverze, kterou python nativně zvládá. Při řešení vyjděte z podkladů v Moodle, případně konzultujte s vyučujícím (každopádně budete řešit GUI s pomocí vybraných balíčků/knihoven podporujících dialogy,).

Odevzdání prvního protokolu UDS2-ID.ZIP bude obsahovat (jména souborů jsou symbolická, soubory vhodně pojmenujte sami, bez diakritiky a mezer): - jmeno.py zdrojový text funkčního programu v pythonu, program je doporučeno vhodně komentovat, a určitě uvést identifikaci autora a případně účel vzniku programu, - text Protokolu 2 (tento dokument). Výsledné soubory zabalte do jediného ZIP archivu pojmenovaného UDS2-ID.ZIP, jde ID je Vaše školní ID, např. UDS2-novakp.zip. Výsledek konzultujte v případě potřeby ve cvičení s vyučujícím. Výsledek odevzdávejte výhradně do moodle pro daný předmět a Projekt 2. Jiný způsob odevzdání nebude akceptován.

Text Protokolu 2 můžete tvořit v Jupyter Notebook nebo v prostředí Anaconda Cloud, Google Colab, které je obdobné Jupyter Notebook.

3. Návrh řešení

Program je navržen tak, aby lidem usnadnil stahování hudby ze SoundCloudu pomocí jednoduchého grafického okna. Uživatel do okna zadá odkaz na skladbu, a program automaticky zjistí, o jakou skladbu jde, a zobrazí její název i obal alba. To funguje díky nástroji s názvem youtube-dl, který dokáže stahovat zvukové soubory z internetu. Kromě toho program používá knihovnu mutagen, což je nástroj, který pomáhá přidat k souborům další informace, jako je název, jméno interpreta, žánr, nebo právě obal alba, aby pak vypadaly profesionálněji – podobně jako skladby, které stáhneme z oficiálních obchodů.

Když je vše připraveno, uživatel klikne na tlačítko pro stažení, a program se postará o to, aby skladba byla uložena na vybrané místo. Současně program upozorní uživatele, pokud zadá neplatný odkaz, nebo pokud se během stahování objeví problém, čímž se zajistí, že uživatel vždy ví, co se děje.

4. Postup při zpracování

Nejdříve potřebujeme do Pythonu nainstalovat několik nástrojů (knihoven), které program využívá k tomu, aby mohl fungovat správně:

- PyQt5 – Tato knihovna je důležitá pro vytvoření grafického okna, kde se vše ovládá. Nainstalujeme ji příkazem:

[ ]:
```
pip install pyqt5
```

- Mutagen – Tento nástroj pomáhá upravovat a přidávat informace k MP3 souborům, jako jsou název, obrázek alba a další detaily. Instalace probíhá takto:

[ ]:
```
pip install mutagen
```

- youtube-dl – Toto je nástroj, který skutečně stáhne skladbu. Funguje nejen pro SoundCloud, ale i pro videa z YouTube. Nainstalujeme ho pomocí:

[ ]:
```
pip install youtube-dl
```

Každý z těchto nástrojů má v programu svůj konkrétní úkol, a všechny spolu dohromady umožní stahování skladeb i přidání jejich informací.

A nyní potřebujeme naimportovat samotné knihovny potřebné pro náš program.

[ ]:
```
import sys
import os
import subprocess
import requests
from io import BytesIO
from PIL import Image
from mutagen.mp3 import MP3
from mutagen.id3 import ID3NoHeaderError, APIC, TPE1, TCON, TDRC
from PyQt5 import QtWidgets, QtGui, QtCore
import mutagen.id3
```

Co to znamená? Import je jako nákup nástrojů v obchodě, které nejsou součástí Pythonu, ale umí dělat užitečné věci:
• sys, os, subprocess: Tyto knihovny pomáhají programu komunikovat s počítačem a operačním systémem (OS). Pomocí těchto nástrojů může program třeba spouštět příkazy nebo přistupovat k souborům uloženým na počítači.
• requests: Tato knihovna slouží ke stahování dat z internetu, konkrétně ke stahování informací a obrázků k hudebním skladbám pomocí URL odkazu.
• io (BytesIO): Tato část knihovny slouží k tomu, aby bylo možné načítat data z internetu jako proudy bytů. To je důležité pro práci s obrázky nebo dalšími soubory přímo z internetu.
• PIL (Pillow): Knihovna Pillow (PIL) umožňuje práci s obrázky. V tomto programu načítá obal alba skladby a mění velikost obrázku, aby se zobrazil správně.
• mutagen.mp3 a mutagen.id3: Mutagen se stará o úpravu informací u MP3 souborů, jako je jméno interpreta, název skladby, žánr nebo přidání obalu alba. To zajišťuje, že když se skladba uloží, vypadá jako běžné hudební soubory s informacemi.
• PyQt5 (QtWidgets, QtGui, QtCore): Tato knihovna se stará o vytvoření grafického okna a uživatelského rozhraní (UI), kde uživatel zadává odkaz na skladbu a kliká na tlačítka.

Nyní potřebujeme vytvořit globální proměnné.

[ ]:
```
global title, img_url, artist, genre
```

K čemu slouží?
Tyto proměnné (název skladby, URL obrázku, umělec, žánr) jsou definovány jako „globální“, což znamená, že budou přístupné ve všech částech programu. Díky tomu je můžeme používat i v jiných funkcích programu a nemusíme je vytvářet pokaždé znovu.

Také musíme nadefinovat třídu SoundCloudDownloader.

[ ]:
```
class SoundCloudDownloader(QtWidgets.QWidget):
```

Co to dělá?
Tato třída je jako návod, který popisuje, jak má aplikace fungovat a jak má vypadat. Obsahuje několik částí (nazývaných metody nebo funkce), které určují jednotlivé funkce aplikace.

Jedná z částí je konstruktor ___init___

[ ]:
```
def __init__(self):
    super().__init__()
    self.title = ""
    self.img_url = ""
    self.setWindowIcon(QtGui.QIcon("icon_1x1.png"))
    self.initUI()
```

Jak funguje?
• super().___init___(): Volá se funkce (QWidget), což umožňuje nastavit základní vlastnosti okna, jako je třeba ikona nebo nadpis.
• Proměnné self.title a self.img_url: Zatím jsou prázdné, ale budou sloužit pro ukládání názvu skladby a obrázku.
• self.setWindowIcon: Nastaví ikonu aplikace (pokud máme připravený soubor icon_1x1.png).
• self.initUI(): Spustí další metodu, která definuje vzhled a rozložení prvků v aplikaci.

Jako další část máme metodu initUi, která nastaví vzhled aplikace a umístí všechny potřebné prvky.

[ ]:
```
def initUI(self):
    self.setWindowTitle('SoundCloud Downloader')
    self.setGeometry(100, 100, 400, 400)
```

Vysvětlení jednotlivých prvků:
- Název a velikost okna: setWindowTitle nastaví název okna a setGeometry jeho velikost.
- Pole pro zadání URL: self.url_entry je pole, kam uživatel zadá odkaz na skladbu.
- Tlačítko pro stažení: self.download_button je tlačítko, které uživatel klikne, aby zahájil stahování.
- Obrázek skladby: Pokud je dostupný obrázek alba nebo skladby, načte se a zobrazí v okně.

Pro získání informací o skladbě nám slouží metoda get_song_info

[ ]:
```
def get_song_info(self):
    url = self.url_entry.text()
```

Jak funguje?
- Načítání URL: Nejprve se z pole načte URL, které uživatel zadal.
- Příkaz youtube-dl: Pomocí příkazu youtube-dl –get-filename program zjistí název skladby, adresu obrázku, a uloží je do proměnných.
- Zobrazení informací: Jakmile program získá informace, zobrazí je ve formě textu nebo obrázku v aplikaci.

A teď to nejdůležitější, samotné stahování skladby – metoda download_song. Tato metoda stáhne skladbu do počítače.

[ ]:
```
def download_song(self):
    url = self.url_entry.text()
```

Co přesně dělá?
- Kontrola URL: Nejprve se znovu zkontroluje, že uživatel zadal správný odkaz.
- Dialog pro uložení souboru: Otevře se okno, kde se uživatel zadá kam chce soubor uložit.
- Stahování pomocí youtube-dl: Program spustí příkaz, který skladbu stáhne na místo, které uživatel vybral.
- Přidání metadat k MP3: Program pomocí mutagen přidá k MP3 souboru obrázek alba a další informace o skladbě, což je užitečné při přehrávání ve většině přehrávačů.

Poslední část kódu nám spustí samotnou aplikaci.

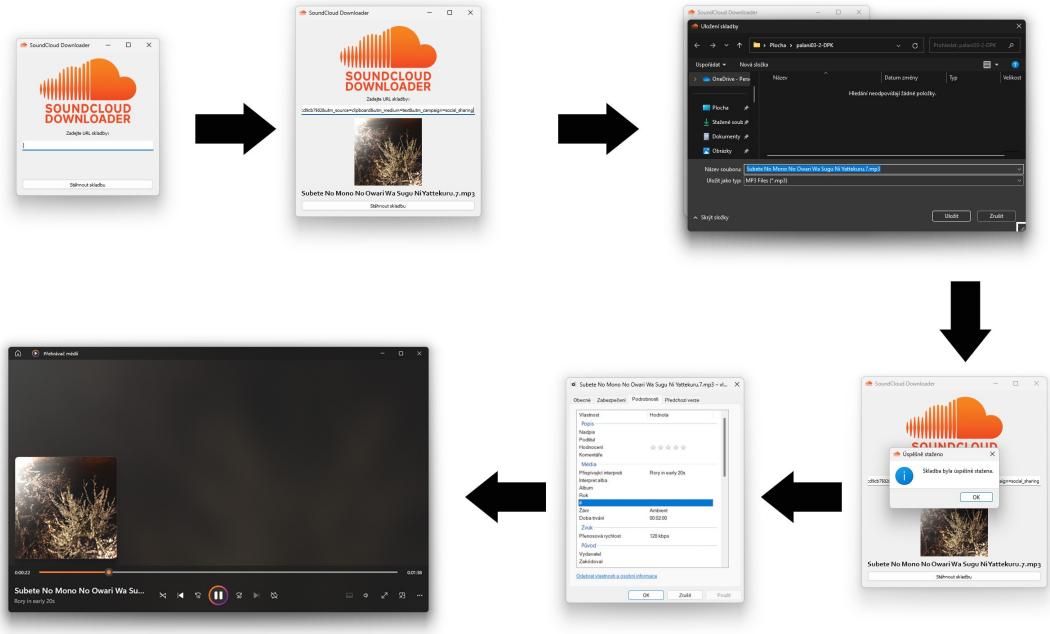
[ ]:
```
if __name__ == '__main__':
    app = QtWidgets.QApplication(sys.argv)
    downloader = SoundCloudDownloader()
    downloader.show()
    sys.exit(app.exec_())
```

Jak funguje?
- Vytvoření aplikace: Pomocí QApplication se spustí celá aplikace, což vytvoří okno na obrazovce.

- Zobrazení okna: downloader.show() zobrazí okno uživateli.
- Ukončení aplikace: Jakmile uživatel okno zavře, sys.exit() ukončí běh aplikace.

5. Výsledky a fakta

Na obrázku můžeme vidět jak nám program funguje:
- Spustíme program.
- Zadáme URL adresu skladby, kterou chceme stáhnout. Pro demonstrační účely jsem vybral skladbu Subete No Mono No Owari Wa Sugu Ni Yattekuru.7 od interpreta Rory in early 20s. (https://soundcloud.com/rorynearly20s/subete-no-mono-no-owari-wa)
- Vybereme místo kam chceme skladbu uložit.
- Pravým kliknutím na staženou skladbu a otevřením vlastností si můžeme ověřit, že se nám opravdu stáhla skladba i s metadaty jako je žánr a interpret.
- Otevřeme skladbu pro ověření, že obsahuje obrázek a že nám opravdu hraje.



6. Závěr

Závěrem lze říci, že jsem docílil programu, kterého jsem docílit chtěl. Prozkoumal jsem práci s knihovnou PyQt5, a musím říct, že výsledek je mi vizuálně sympatičtější než moje předchozí programy využívající tkinter, ale přesto se mi s knihovnou tkinter pracovalo lépe. Chtěl jsem do programu zahrnout i výběr formátu pro stažení, nicméně k tomu by bylo potřeba extérního programu ffmpeg a nenašel jsem jiný způsob, jak toho docílit. Při přidávání interpreta a žánru jsem zjistil malý nedostatek, a to ten, že můj program neumí pracovat se speciálními znaky, takže když např. jméno interpreta nebo název skladby obsahuje speciální znak jako např. hvězdičku, tak se znak nezobrazí, ale místo něj se zobrazí pouze kód znaku. Zároveň jsem si vědom, že takové stahování může vyvolat problém s autorskými právy apod. to samé lze aplikovat na grafiku vytvořenou pro tento program, kombinací loga SoundCloud a textu Downloader, takže je důležité zmínit, že celý tento program slouží pouze pro účel zdokonalit se v programování.

7. Zdrojový kód

[3]:
```
# Palán Nicolas
# 28/10/2024
# KTE/DPK@
# SoundCloud Downloader
# Aplikace pro stahování skladeb ze SoundCloud
# využito open source knihovny youtube-dl: https://github.com/ytdl-org/
 ↪youtube-dl

import sys  # Importuje modul sys pro systémově specifické parametry a funkce
import os  # Importuje modul os pro interakci se systémem
import subprocess  # Importuje modul subprocess pro spouštění shellových příkazů
import requests  # Importuje modul requests pro HTTP požadavky
from io import BytesIO  # Importuje BytesIO pro zpracování byte streamů
from PIL import Image  # Importuje třídu Image z PIL pro zpracování obrázků
from mutagen.mp3 import MP3  # Importuje třídu MP3 z mutagen pro práci s MP3␣
 ↪soubory
from mutagen.id3 import ID3NoHeaderError, APIC, TPE1, TCON, TDRC  # Importuje␣
 ↪potřebné třídy pro ID3 tagy
from PyQt5 import QtWidgets, QtGui, QtCore  # Importuje potřebné třídy z PyQt5
import mutagen.id3  # Pro práci s ID3 tagy

# Globální proměnné pro uložení názvu skladby, URL obrázku a metadat
global title, img_url, artist, genre

class SoundCloudDownloader(QtWidgets.QWidget):
    def __init__(self):
        super().__init__()
        self.title = ""  # Inicializuje název
        self.img_url = ""  # Inicializuje URL obrázku

        # Nastaví ikonu okna
        self.setWindowIcon(QtGui.QIcon("icon_1x1.png"))  # Cesta k souboru s␣
 ↪ikonou

        # Inicializuje uživatelské rozhraní
        self.initUI()

    def initUI(self):
        # Nastaví vlastnosti okna
        self.setWindowTitle('SoundCloud Downloader')  # Nastaví název okna
        self.setGeometry(100, 100, 400, 400)  # Nastaví velikost okna

        # Vytvoření vertikálního rozložení
        layout = QtWidgets.QVBoxLayout()
```

```
        # Přidání loga do rozhraní
        try:
            self.logo = QtGui.QPixmap("soundcloud_downloader.png").scaled(300,␣
↪150, QtCore.Qt.KeepAspectRatio, QtCore.Qt.SmoothTransformation)  # Načte a␣
↪zmenší logo
            logo_label = QtWidgets.QLabel(self)  # Vytvoří label pro logo
            logo_label.setPixmap(self.logo)  # Nastaví pixmapu loga
            logo_label.setAlignment(QtCore.Qt.AlignCenter)  # Centrování loga
            layout.addWidget(logo_label)  # Přidá label loga do layoutu
        except Exception as e:
            print(f"Nepodařilo se načíst logo: {str(e)}")  # Chybové hlášení,␣
↪pokud se logo nenačte

        # Vytvoření labelu a vstupu pro URL
        self.url_label = QtWidgets.QLabel("Zadejte URL skladby:")  # Label pro␣
↪zadání URL
        self.url_label.setAlignment(QtCore.Qt.AlignCenter)  # Centrování labelu
        layout.addWidget(self.url_label)  # Přidá label do layoutu

        self.url_entry = QtWidgets.QLineEdit(self)  # Vytvoří textové pole pro␣
↪zadání URL
        layout.addWidget(self.url_entry)  # Přidá vstup pro URL do layoutu

        self.img_label = QtWidgets.QLabel(self)  # Vytvoří label pro zobrazení␣
↪obrázku skladby
        self.img_label.setAlignment(QtCore.Qt.AlignCenter)  # Centrování labelu␣
↪obrázku
        layout.addWidget(self.img_label)  # Přidá label obrázku do layoutu

        self.title_label = QtWidgets.QLabel("", self)  # Vytvoří label pro␣
↪zobrazení názvu skladby
        self.title_label.setFont(QtGui.QFont("Corbel", 12, QtGui.QFont.Bold)) ␣
↪# Nastaví font pro label názvu
        self.title_label.setAlignment(QtCore.Qt.AlignCenter)  # Centrování␣
↪labelu názvu
        layout.addWidget(self.title_label)  # Přidá label názvu do layoutu

        # Vytvoření tlačítka pro stažení skladby
        self.download_button = QtWidgets.QPushButton("Stáhnout skladbu", self) ␣
↪# Tlačítko pro stažení skladby
        self.download_button.clicked.connect(self.download_song)  # Připojí␣
↪kliknutí na tlačítko k metodě download_song
        layout.addWidget(self.download_button)  # Přidá tlačítko do layoutu

        # Nastaví layout pro hlavní widget
        self.setLayout(layout)
```

```
        # Připojení textového pole pro URL k automatickému zobrazení informací␣
↪o skladbě
        self.url_entry.textChanged.connect(self.get_song_info)  # Spustí␣
↪get_song_info při změně textu v URL

    def get_song_info(self):
        global title, img_url, artist, genre  # Přidáme nové globální proměnné
        url = self.url_entry.text()  # Získá text z URL vstupu

        if url:  # Pokud je URL poskytnuta
            try:
                # Kontrola, zda je URL platná pomocí HEAD požadavku
                response = requests.head(url)
                if response.status_code != 200:  # Pokud kód odpovědi není 200,␣
↪je URL neplatná
                    QtWidgets.QMessageBox.warning(self, "Upozornění", "Zadejte␣
↪prosím platnou URL skladby.")  # Zobrazí varování pro neplatnou URL
                    self.img_label.clear()  # Vymaže label obrázku
                    self.title_label.clear()  # Vymaže label názvu
                    return  # Ukončí metodu

                # Získání názvu skladby a URL obrázku pomocí youtube-dl
                command_info = f'youtube-dl --get-filename -o "%(title)s.
↪%(ext)s" "{url}"'  # Příkaz pro získání názvu souboru
                title = subprocess.check_output(command_info, shell=True).
↪decode().strip()  # Provede příkaz a dekóduje výstup

                command_metadata = f'youtube-dl -j "{url}"'  # Příkaz pro␣
↪získání metadat
                metadata = subprocess.check_output(command_metadata,␣
↪shell=True).decode().strip()  # Provede příkaz a dekóduje výstup

                # Najde URL obrázku a další metadata v metadatech
                img_url = None  # Inicializuje URL obrázku
                artist = None
                genre = None

                for line in metadata.splitlines():
                    if '"thumbnail":' in line:  # Zkontroluje URL náhledu
                        img_url = line.split('"thumbnail": "')[1].split('"')[0]␣
↪ # Extrakce URL obrázku
                    if '"uploader":' in line:  # Hledání jména interpreta
                        artist = line.split('"uploader": "')[1].split('"')[0]
                    if '"genre":' in line:  # Hledání žánru
                        genre = line.split('"genre": "')[1].split('"')[0]
```

```
                # Zobrazení názvu skladby
                self.title_label.setText(title)  # Nastaví text pro label názvu

                # Zobrazení obrázku skladby
                if img_url:  # Pokud URL obrázku existuje
                    response = requests.get(img_url)  # Získá data obrázku
                    img_data = Image.open(BytesIO(response.content))  # Otevře␣
↪data obrázku
                    img_data = img_data.resize((150, 150), Image.LANCZOS)  #␣
↪Změní velikost obrázku
                    img_data = img_data.convert("RGB")  # Převod na RGB
                    qt_image = QtGui.QImage(img_data.tobytes(), img_data.width,␣
↪img_data.height, img_data.width * 3, QtGui.QImage.Format_RGB888)  # Převede␣
↪obrázek na Qt formát
                    pixmap = QtGui.QPixmap.fromImage(qt_image)  # Vytvoří␣
↪pixmapu z obrázku
                    self.img_label.setPixmap(pixmap)  # Zobrazí obrázek ve␣
↪labelu
                else:
                    self.img_label.clear()  # Pokud není URL obrázku, vymaže␣
↪label obrázku

            except Exception as e:
                QtWidgets.QMessageBox.critical(self, "Chyba", f"Nastala chyba:␣
↪{str(e)}")  # Zobrazí chybové hlášení
        else:
            QtWidgets.QMessageBox.warning(self, "Upozornění", "Zadejte prosím␣
↪URL skladby.")  # Zobrazí varování pro prázdné URL

    def download_song(self):
        global title, img_url, artist, genre
        url = self.url_entry.text()

        if not url:
            QtWidgets.QMessageBox.warning(self, "Upozornění", "Nejprve zadejte␣
↪URL skladby.")
            return

        if title:
            try:
                audio_file_path, _ = QtWidgets.QFileDialog.
↪getSaveFileName(self, "Uložení skladby", f"{title}", "MP3 Files (*.mp3);;All␣
↪Files (*)")
                if audio_file_path:
```

```
                    command_download = f'youtube-dl -o "{audio_file_path}" "{url}"'
                    subprocess.run(command_download, shell=True)

                    if img_url:
                        img_response = requests.get(img_url)
                        if img_response.status_code == 200:
                            audio = MP3(audio_file_path)
                            try:
                                audio.add_tags()
                            except ID3NoHeaderError:
                                pass

                            # Přidání obalu alba
                            audio.tags.add(APIC(mime='image/jpeg', type=3, desc='Cover', data=img_response.content))

                            # Přidání dalších tagů: interpret, žánr, rok vydání
                            if artist:
                                audio.tags.add(TPE1(encoding=3, text=artist))  # Přidání interpreta
                            if genre:
                                audio.tags.add(TCON(encoding=3, text=genre))  # Přidání žánru

                            audio.save()

                            QtWidgets.QMessageBox.information(self, "Úspěšně staženo", "Skladba byla úspěšně stažena.")
                        else:
                            QtWidgets.QMessageBox.warning(self, "Upozornění", "Nepodařilo se stáhnout obrázek.")
                    else:
                        QtWidgets.QMessageBox.warning(self, "Upozornění", "Nepodařilo se získat URL obrázku.")
                else:
                    QtWidgets.QMessageBox.warning(self, "Upozornění", "Soubor nebyl uložen.")
            except Exception as e:
                QtWidgets.QMessageBox.critical(self, "Chyba", f"Nastala chyba: {str(e)}")
        else:
            QtWidgets.QMessageBox.warning(self, "Upozornění", "Nejprve zadejte URL skladby.")

if __name__ == '__main__':
```

```
    app = QtWidgets.QApplication(sys.argv)  # Vytvoří aplikaci
    downloader = SoundCloudDownloader()  # Vytvoří instanci aplikace
    downloader.show()  # Zobrazí okno aplikace
    sys.exit(app.exec_())  # Ukončí aplikaci
```