Znaki wodne - widoczne i niewidoczne

Dokumentacja projektu

Steganografia

Adrian Sekuła

Adam Fedoruk

Znaki Wodne - widoczne i niewidoczne

Opis Aplikacji

Pierwszy moduł (main_app.py) stanowi główną część programu odpowiedzialną za uruchamianie podmodułów związanych z dodawaniem różnych rodzajów znaków wodnych. Program jest napisany w języku Python 3.

Struktura Kodu

Importowanie Bibliotek

```
import tkinter as tk
from visible_watermarks import VisibleWatermarks
from LSB_password_gui import LSB_Watermark_Password
from mp3_watermarker_tags import MP3_Watermarker
from web_watermark import WebWatermarksApp
```

Klasa `ZnakiWodneAplikacja`

Główna klasa programu, odpowiedzialna za interfejs graficzny i uruchamianie podmodułów.

Metoda `__init__(self)`

Konstruktor inicjujący główne okno programu.

```
def __init__(self):
    # Inicjalizacja głównego okna
    self.root = tk.Tk()
    self.root.geometry("500x200")
    self.root.title("Znaki wodne - Widoczne i Niewidoczne")

    # Tworzenie etykiety tytułowej
    self.title_label = tk.Label(
        self.root, text="Wybierz jedną z poniższych metod dodawania znaku
wodnego:")
    self.title_label.pack()

    # Tworzenie przycisków do uruchamiania różnych metod
    self.launch_button = tk.Button(
        self.root, text="Znak wodny widoczny (tekst/obraz)",
command=self.visible_watermarks)
    self.launch_button.pack()

    self.launch_button = tk.Button(
        self.root, text="Znak wodny ukryty - LSB z hasłem",
command=self.lsb_watermark_password)
```

```
    self.launch_button.pack()

    self.launch_button = tk.Button(
        self.root, text="Ukryty znak wodny w tagu pliku MP3",
command=self.mp3_watermarker)
    self.launch_button.pack()

    self.launch_button = tk.Button(
        self.root, text="Ukryty watermark w meta tagu HTML",
command=self.web_watermarks)
    self.launch_button.pack()

#### Metoda `launch_app(self)`
Metoda do uruchamiania głównego okna programu.

```python
def launch_app(self):
    self.root.mainloop()
```

Metody Uruchamiające Podmoduły

Metody te tworzą nowe okno dla każdego z podmodułów i uruchamiają je.

```
def visible_watermarks(self):
    newWindow = tk.Toplevel(self.root)
    newWindow.grab_set()
    x = self.root.winfo_x()
    y = self.root.winfo_y()
    newWindow.geometry("+%d+%d" % (x, y))
    app = VisibleWatermarks(newWindow)
    app.pack_elements()

def lsb_watermark_password(self):
    # Analogicznie do visible_watermarks

def mp3_watermarker(self):
    # Analogicznie do visible_watermarks

def web_watermarks(self):
    # Analogicznie do visible_watermarks
```

Blok Główny

Sprawdzanie, czy skrypt jest uruchamiany jako plik główny, a nie importowany jako moduł.

```
if __name__ == "__main__":
    app = ZnakiWodneAplikacja()
    app.launch_app()
```
```

Użycie

Uruchomienie tego skryptu skutkuje otwarciem głównego okna programu, umożliwiającego wybór różnych metod dodawania znaków wodnych.

Wymagania

Program wymaga zainstalowanych bibliotek: `tkinter`, oraz zaimportowanych kolejnych modułów programu `visible_watermarks`, `LSB_password_gui`, `mp3_watermarker_tags`, `web_watermark`.

Znak wodny widoczny (tekst/obraz)

Opis

Moduł `VisibleWatermarks` odpowiada za dodawanie widocznych znaków wodnych do obrazów. Moduł ten umożliwia użytkownikowi dodawanie zarówno tekstowych, jak i obrazowych znaków wodnych do wybranych obrazów.

Struktura Kodu

Importowanie Bibliotek

```
import tkinter as tk
import tkinter.ttk as ttk
from tkinter import filedialog, messagebox
from PIL import Image, ImageDraw, ImageFont, ImageTk
import numpy as np
```

Klasa `VisibleWatermarks`

Klasa ta zawiera metody i elementy interfejsu graficznego potrzebne do dodawania widocznych znaków wodnych.

Metoda `__init__(self, root)`

Konstruktor inicjujący obiekt `VisibleWatermarks`. Przyjmuje jako argument główne okno aplikacji.

```
def __init__(self, root):
    # Inicjalizacja głównego okna
    self.root = root

    # Inicjalizacja zakładek (Notebook)
    self.notebook = ttk.Notebook(root)
    self.text_tab = tk.Frame(self.notebook)
    self.image_tab = tk.Frame(self.notebook)
    self.notebook.add(self.text_tab, text='Watermark - Text')
    self.notebook.add(self.image_tab, text='Watermark - Image')
```

```
    # Elementy interfejsu graficznego dla zakładki tekstowej
    # ...

    # Elementy interfejsu graficznego dla zakładki obrazowej
    # ...

    # Dodatkowy przycisk "Pomoc"
    self.info_button = tk.Button(
        root, text="Pomoc", command=self.show_help_box)
```

Metoda `pack_elements(self)`

Metoda ta umieszcza elementy interfejsu graficznego na odpowiednich zakładkach.

```
def pack_elements(self):
    # Pakowanie zakładek
    self.notebook.pack()

    # Pakowanie elementów interfejsu graficznego dla zakładki tekstowej
    # ...

    # Pakowanie elementów interfejsu graficznego dla zakładki obrazowej
    # ...

    # Pakowanie dodatkowego przycisku "Pomoc"
    self.info_button.pack()
```

Metoda `add_watermark_and_save(self)`

Metoda ta obsługuje dodawanie znaków wodnych do obrazu i zapisywanie go. Wywoływana jest po naciśnięciu przycisku "Generuj i zapisz obraz z znakiem wodnym". Metoda ta obsługuje zarówno znaki wodne obrazowe jak i tekstowe. Znajduje się w niej logika odpowiadająca za umieszczenie znaku wodnego w odpowiednim miejscu i w odpowiednim rozmiarze.

```
def add_watermark_and_save(self):
    # ...
```

Metoda `show_help_box(self)`

Metoda ta wyświetla okno pomocy z opisem funkcji i sposobu użycia modułu.

```
def show_help_box(self):
    # ...
```

Użycie

Aby skorzystać z funkcji modułu `VisibleWatermarks`, należy utworzyć obiekt tej klasy i użyć metody `pack_elements()`. Następnie użytkownik może wybrać zakładkę i dostosować ustawienia znaku wodnego, a po kliknięciu przycisku "Generuj i zapisz obraz z znakiem wodnym" wybrać obraz do oznaczenia oraz miejsce zapisu.

Pomoc

Watermark - Text

Pierwsza zakładka służy dodawaniu tekstu jako znaku wodnego do obrazu.

Opcje:

- Tekst znaku wodnego: wpisany przez użytkownika znak wodny tekstowy.
- Kolor znaku wodnego: wybór koloru tekstu znaku wodnego.
- Położenie znaku wodnego: określa położenie tekstu.
- Rozmiar znaku wodnego: określa rozmiar tekstu znaku wodnego.

Aby wygenerować obraz ze znakiem wodnym:

1. Wpisz tekst znaku wodnego.
2. Ustaw opcje znaku wodnego.
3. Kliknij przycisk "Generuj...".
4. W nowym oknie wybierz obraz do oznakowania.
5. W kolejnym oknie wybierz miejsce zapisu obrazu oraz jego nazwę.

Watermark - Obraz

Druga zakładka służy dodawaniu obrazu PNG jako znaku wodnego do obrazu.

Opcje:

- Położenie znaku wodnego: określa położenie obrazu znaku wodnego.

Aby wygenerować obraz ze znakiem wodnym:

1. Ustaw położenie znaku wodnego.
2. Kliknij przycisk "Generuj...".
3. W nowym oknie wybierz obraz, który chcesz oznakować.
4. W kolejnym oknie wybierz znak wodny, który chcesz dodać do obrazu.
5. W kolejnym oknie wybierz nazwę oraz miejsce zapisu oznakowanego obrazu.

Wymagania

Moduł ten wymaga zainstalowanych bibliotek: `tkinter`, `PIL`, `numpy`. Można je zainstalować używając narzędzia do zarządzania pakietami w Pythonie, na przykład pip:

```
pip install tkinter
pip install pillow
pip install numpy
```

Znak wodny ukryty - LSB z hasłem

Opis

Moduł `LSB_Watermark_Password` umożliwia ukrywanie i odczytywanie tekstowych znaków wodnych w obrazach PNG za pomocą metody najmniej znaczącego bitu (LSB). Do zakodowania znaku wodnego wykorzystywane jest hasło, które jest również niezbędne do późniejszego odczytania znaku.

Struktura Kodu

Importowanie Bibliotek

```
import sys
import numpy as np
from PIL import Image
import tkinter as tk
from tkinter import filedialog, messagebox
from tkinter import ttk
```

Klasa `LSB_Watermark_Password`

Klasa ta zawiera metody i elementy interfejsu graficznego potrzebne do zakodowania i odczytania tekstowego znaku wodnego przy użyciu metody LSB.

Metoda `__init__(self, root)`

Konstruktor inicjujący obiekt `LSB_Watermark_Password`. Przyjmuje jako argument główne okno aplikacji.

```
def __init__(self, root):
    # Inicjalizacja głównego okna
    self.root = root

    # Inicjalizacja zakładek (Notebook)
    self.notebook = ttk.Notebook(root)
    self.root.title("Znak Wodny LSB z hasłem")
```

```
        # Inicjalizacja zakładek dla zakodowania i odczytania znaku wodnego
        self.encode_tab = ttk.Frame(self.notebook)
        self.decode_tab = ttk.Frame(self.notebook)
        self.src_filename = None
        self.src_filename_decode = None

        # Dodatkowy przycisk "Pomoc"
        self.info_button = tk.Button(
            root, text="Pomoc", command=self.show_help_box)
```

Metoda `Encode(self, src, message, dest, password)`

Metoda kodująca tekstowy znak wodny w obrazie. Wymaga podania pliku źródłowego (`src`), treści znaku wodnego (`message`), ścieżki docelowej (`dest`) i hasła (`password`).

```
def Encode(self, src, message, dest, password):
    # ...
```

Metoda `Decode(self, src, password)`

Metoda odczytująca tekstowy znak wodny z obrazu. Wymaga podania pliku źródłowego (`src`) i hasła (`password`).

```
def Decode(self, src, password):
    # ...
```

Metoda `pack_elements(self)`

Metoda umieszczająca elementy interfejsu graficznego na odpowiednich zakładkach.

```
def pack_elements(self):
    # ...
```

Metoda `show_help_box(self)`

Metoda wyświetlająca okno pomocy z opisem funkcji i sposobu użycia modułu.

```
def show_help_box(self):
    # ...
```

Użycie

Aby skorzystać z funkcji modułu `LSB_Watermark_Password`, należy utworzyć obiekt tej klasy i użyć metody `pack_elements()`. Następnie użytkownik może wybrać zakładkę "Dodaj znak wodny" lub "Odczytaj znak wodny" i dostosować ustawienia znaku wodnego, a po kliknięciu odpowiedniego przycisku przeprowadzić proces kodowania lub odczytywania.

Pomoc

Podprogram pozwala na ukrycie tekstowego znaku wodnego w obrazie PNG. <!> Program nie wspiera polskich znaków (ą, ś, ...).

Dodaj znak wodny

Opcje:

- Obraz źródłowy (PNG) - wybranie obrazu bazowego
- Znak wodny do ukrycia - tekstowy znak wodny do ukrycia w obrazie
- Hasło - hasło potrzebne do odkodowania znaku wodnego

Aby dodać niewidoczny tekstowy znak wodny:

1. Wybierz obraz źródłowy PNG
2. Wpisz znak wodny tekstowy do pola tekstowego
3. Wpisz hasło wymagane do odkodowania znaku wodnego
4. Kliknij "Zakoduj"
5. W nowym oknie wybierz nazwę i miejsce zapisu podpisanego obrazu

Odczytaj znak wodny

Opcje:

- Obraz źródłowy (PNG) - wybranie obrazu bazowego z ukrytym tekstem
- Hasło - hasło podane przy zakodowywaniu znaku wodnego

Aby odczytać ukryty tekstowy znak wodny:

1. Wybierz obraz źródłowy PNG
2. Wpisz hasło wymagane do odkodowania znaku wodnego
3. Kliknij przycisk "Zdekoduj"
4. Odczytaj znak wodny w nowym oknie

Wymagania

Moduł ten wymaga zainstalowanych bibliotek: `tkinter`, `PIL`, `numpy`. Można je zainstalować używając narzędzia do zarządzania pakietami w Pythonie, na przykład pip:

```
pip install tkinter
pip install pillow
pip install numpy
```

Ukryty znak wodny w tagu pliku MP3

Opis

Moduł `MP3_Watermarker` umożliwia dodawanie i odczytywanie ukrytych znaków wodnych w plikach MP3 poprzez edycję meta tagów.

Struktura Kodu

Importowanie Bibliotek

```
import tkinter as tk
import tkinter.ttk as ttk
from tkinter import filedialog, messagebox
from mutagen.id3 import ID3, TXXX
from shutil import copy2
import os
```

Klasa `MP3_Watermarker`

Klasa ta zawiera metody i elementy interfejsu graficznego potrzebne do dodawania i odczytywania tekstowych znaków wodnych w plikach MP3.

Metoda `__init__(self, root)`

Konstruktor inicjujący obiekt `MP3_Watermarker`. Przyjmuje jako argument główne okno aplikacji.

```
def __init__(self, root):
    # Inicjalizacja głównego okna
    self.root = root

    # Inicjalizacja zakładek (Notebook)
    self.notebook = tk.ttk.Notebook(self.root)
    self.root.title("MP3 Watermark")

    # Inicjalizacja zakładek dla dodawania i odczytywania znaku wodnego
    self.tab1 = tk.Frame(self.notebook)
    self.tab2 = tk.Frame(self.notebook)

    # Dodatkowy przycisk "Pomoc"
    self.info_button = tk.Button(
        root, text="Pomoc", command=self.show_help_box)
```

Metoda `pack_elements(self)`

Metoda umieszczająca elementy interfejsu graficznego na odpowiednich zakładkach.

```
def pack_elements(self):
    # ...
```

Metoda `open_file_tab1(self)`

Metoda otwierająca dialog wyboru pliku dla zakładki "Dodaj znak wodny".

```
def open_file_tab1(self):
    # ...
```

Metoda `open_file_tab2(self)`

Metoda otwierająca dialog wyboru pliku dla zakładki "Odczytaj znak wodny".

```
def open_file_tab2(self):
    # ...
```

Metoda `add_watermark_tab1(self)`

Metoda dodająca tekstowy znak wodny do pliku MP3. Znak wodny dodawany jest jako dodatkowy tag użytkownika z nazwą "Watermark".

```
def  add_watermark_tab1(self):
    filename  =  self.open_file_tab1()
    if  filename:
        original_file  =  filename
        original_file_copy  =  original_file  +  "_original"
        copy2(original_file, original_file_copy)

        audio  = ID3(original_file)
        if  "TXXX:Watermark"  in  audio:
            audio["TXXX:Watermark"].text = [self.watermark.get()]
        else:
            audio["TXXX:Watermark"] = TXXX(
                encoding=3, text=self.watermark.get())
        audio.save()

        watermarked_file  =  original_file[:-4] +  "_watermarked.mp3"
        os.rename(original_file, watermarked_file)
        os.rename(original_file_copy, original_file)
```

Metoda `read_watermark_tab2(self)`

Metoda odczytująca tekstowy znak wodny z pliku MP3.

```
def  read_watermark_tab2(self):
    filename  =  self.open_file_tab2()
    if  filename:
        audio  = ID3(filename)
        watermark  =  audio.getall("TXXX")
        if  watermark:
            messagebox.showinfo(
                "Znak wodny", "Odczytany znak wodny:\n"  +  watermark[0].text[0])
        else:
            messagebox.showinfo(
                "Znak wodny", "Nie znaleziono znaku wodnego!")
```

Metoda `show_help_box(self)`

Metoda wyświetlająca okno pomocy z opisem funkcji i sposobu użycia modułu.

```
def show_help_box(self):
    # ...
```

Użycie

Aby skorzystać z funkcji modułu `MP3_Watermarker`, należy utworzyć obiekt tej klasy i użyć metody `pack_elements()`. Następnie użytkownik może wybrać zakładkę "Dodaj znak wodny" lub "Odczytaj znak wodny", dostosować ustawienia znaku wodnego, a po kliknięciu odpowiedniego przycisku przeprowadzić proces dodawania lub odczytywania znaku wodnego.

Pomoc

Ukryty znak wodny w tagu pliku MP3

Dodaj znak wodny

Zakładka pozwala dodać ukryty tekstowy znak wodny do pliku MP3.

Opcje:

- Tekst znaku wodnego: znak wodny tekstowy do ukrycia w pliku MP3.

Aby dodać ukryty znak wodny tekstowy do pliku MP3:

1. Wpisz tekst znaku wodnego.
2. Kliknij "Wybierz plik..."
3. W nowym oknie wybierz plik MP3.
4. Plik MP3 ze znakiem wodnym tekstowym zostanie zapisany w tym samym folderze co plik źródłowy.

Odczytaj znak wodny

Aby odczytać ukryty znak wodny z pliku MP3:

1. Kliknij "Wybierz plik..."
2. Odczytaj znak wodny w nowym oknie.

Wymagania

Moduł ten wymaga zainstalowanych bibliotek: `tkinter`, `mutagen`. Można je zainstalować używając narzędzia do zarządzania pakietami w Pythonie, na przykład pip:

```
pip install tkinter
pip install mutagen
```

Ukryty watermark w meta tagu HTML

Opis

Moduł `WebWatermarksApp` pozwala na dodawanie ukrytych znaków wodnych w plikach HTML poprzez modyfikację meta tagów.

Struktura Kodu

Importowanie Bibliotek

```
import tkinter as tk
import requests
from tkinter import messagebox, filedialog
from bs4 import BeautifulSoup
import hashlib
```

Klasa `WebWatermarksApp`

Klasa ta zawiera metody i elementy interfejsu graficznego potrzebne do dodawania ukrytych znaków wodnych do plików HTML.

Metoda `__init__(self, root)`

Konstruktor inicjujący obiekt `WebWatermarksApp`. Przyjmuje jako argument główne okno aplikacji.

```
def __init__(self, root):
    # Inicjalizacja głównego okna
    self.root = root

    # Inicjalizacja etykiety i pola tekstowego dla adresu URL
    self.url_label = tk.Label(root, text="URL:")
```

```
        self.url_entry = tk.Entry(root, width=30)

        # Inicjalizacja przycisku do generowania i zapisywania
        self.generate_button = tk.Button(
            root, text="Wygeneruj i zapisz", command=self.add_watermark_and_save)

        # Inicjalizacja przycisku pomocy
        self.info_button = tk.Button(
            root, text="Pomoc", command=self.show_help_box)
```

Metoda `pack_elements(self)`

Metoda umieszczająca elementy interfejsu graficznego w głównym oknie.

```
def pack_elements(self):
    # ...
```

Metoda `add_watermark_and_save(self)`

Metoda dodająca ukryty znak wodny do pliku HTML.

```
def add_watermark_and_save(self):
    # ...
```

Metoda `calculate_hash(self, data)`

Metoda obliczająca funkcję skrótu SHA-256 dla danych.

```
def calculate_hash(self, data):
    # ...
```

Metoda `show_help_box(self)`

Metoda wyświetlająca okno pomocy z opisem funkcji i sposobu użycia modułu.

```
def show_help_box(self):
    # ...
```

Użycie

Aby skorzystać z funkcji modułu `WebWatermarksApp`, należy utworzyć obiekt tej klasy i użyć metody `pack_elements()`. Następnie użytkownik może wprowadzić adres strony internetowej do pola "URL:", a po

kliknięciu przycisku "Wygeneruj i zapisz" zostanie dodany ukryty znak wodny do pliku HTML. Użytkownik musi również wybrać miejsce i nazwę dla wygenerowanego pliku HTML.

Pomoc

Ukryty watermark w meta tagu HTML

Podprogram służy do dodania ukrytego znaku wodnego w tagu pliku HTML. <!> W adresie strony należy zawrzeć przedrostek http:// lub https://

Aby dodać ukryty znak wodny:

1. W polu "URL:" podaj adres strony internetowej, do której chcesz dodać znak wodny.
2. Kliknij w przycisk "Wygeneruj i zapisz".
3. Wybierz miejsce oraz nazwę dla wygenerowanego pliku HTML.

Wymagania

Moduł ten wymaga zainstalowanych bibliotek: `tkinter`, `requests`, `bs4`. Można je zainstalować używając narzędzia do zarządzania pakietami w Pythonie, na przykład pip:

```
pip install tkinter
pip install requests
pip install beautifulsoup4
```