REPRODUCTOR DE MÚSICA MP3.

202105668 – Daved Abshalon Ejcalon Chonay

Resumen

El proyecto tiene como objetivo principal que los estudiantes desarrollen una solución integral utilizando programación orientada a objetos (POO) en Python, implementando tipos de datos abstractos (TDA). La aplicación IPC music es un reproductor de música de escritorio con una interfaz intuitiva que permite organizar la biblioteca musical, reproducir canciones, y generar estadísticas. Se enfoca en cargar bibliotecas desde archivos XML, crear listas de reproducción, y mostrar reportes en HTML y Graphviz.

Palabras clave

TDA
Graphviz
Lista enlazada
XML
HTML

Abstract

The main objective of the project is for students to develop a comprehensive solution using object-oriented programming (OOP) in Python, implementing abstract data types (ADT). The IPC music application is a desktop music player with an intuitive interface that allows organizing the music library, playing songs, and generating statistics. It focuses on loading libraries from XML files, creating playlists, and displaying reports in HTML and Graphviz.

Keywords

TDA
Graphviz
Linked List
XML
HTML

Introducción

Este proyecto tiene como objetivo principal capacitar a los estudiantes en el desarrollo de soluciones integrales mediante la implementación de Tipos de Datos Abstractos (TDA) en el contexto de la programación orientada a objetos. El propósito general es que los estudiantes adquieran habilidades para abordar problemas complejos mediante la aplicación de conceptos de programación orientada a objetos y estructuras de datos avanzadas. Los objetivos específicos incluyen la implementación de programación orientada a objetos en Python, el uso de estructuras de programación secuenciales, cíclicas y condicionales, así como la utilización de listas ordenadas para construir una matriz dispersa con asignación dinámica de memoria.

La aplicación de escritorio denominada IPCmusic se presenta como un reproductor de música. Su interfaz de usuario busca ser amigable e intuitiva, permitiendo al usuario organizar y visualizar su biblioteca musical. Entre sus funcionalidades destacan la reproducción de canciones con información detallada sobre artistas y álbumes, opciones de pausa, avance y retroceso.

Desarrollo del tema

En el ámbito de la programación y desarrollo de software, diversos conceptos y tecnologías desempeñan un papel crucial para lograr soluciones efectivas y eficientes. Algunos de estos conceptos que se consideraron en la realización de este proyecto fueron los siguientes: Graphviz, Linked List, XML y HTML. Cada uno de ellos desempeña un papel específico y contribuye de manera significativa a diferentes aspectos del desarrollo de software, desde la representación visual de datos hasta el intercambio de información estructurada y la presentación en entornos web.

a. Graphviz

Graphviz es una herramienta de visualización de gráficos que permite representar gráficamente relaciones entre elementos. Utiliza un lenguaje de descripción de gráficos llamado DOT para especificar la estructura del grafo. Esta herramienta es valiosa para representar visualmente la conectividad y la jerarquía en conjuntos de datos complejos, facilitando la comprensión de relaciones entre entidades.

b. Listas

La Linked List, o lista enlazada, es una estructura de datos fundamental en programación. Consiste en nodos enlazados, donde cada nodo contiene datos y un puntero al siguiente nodo. Este enfoque de almacenamiento dinámico permite una gestión eficiente de la memoria y una manipulación flexible de datos. La Linked List es esencial para situaciones en las que la inserción y eliminación frecuentes de elementos son comunes.

c. XML

XML es un lenguaje de marcado diseñado para almacenar y transportar datos de manera legible tanto para humanos como para máquinas. Su estructura jerárquica permite representar información de manera flexible y extensible. XML es ampliamente utilizado para el intercambio de datos entre sistemas heterogéneos y es esencial en el desarrollo de soluciones que requieren una estructura clara y organizada para la información.

d. HTML

HTML es el lenguaje de marcado estándar para la creación y presentación de contenido en la web. Define la estructura y el diseño de las páginas web mediante una serie de elementos y etiquetas. La combinación de HTML con otras tecnologías como CSS (Cascading Style Sheets) y JavaScript permite la creación de interfaces web dinámicas e interactivas.

Desarrollo técnico del programa

Clase "Menu.py". Esta clase del código es el apartado de interfaz gráfica de usuario (GUI) escrita en Python con el **uso de la biblioteca PyQt6.**

En esta clase se importó xml.etree.ElementTree para trabajar con archivos XML y os y webbrowser para operaciones relacionadas con el sistema operativo y navegadores web.

El método "carga_biblioteca" lee y parase el archivo XML usado, también extrae la información sobre canciones, artistas y álbumes del archivo XML y la agrega en la estructura de datos "self.biblioteca". El método "reporudcir" y "reproducir_aleatoria" verifica si hay canciones en la biblioteca.

En resumen, la lógica principal del programa incluye cargar una biblioteca de música desde un archivo XML, reproducir música de manera normal o aleatoria, crear listas de reproducción, y generar informes sobre las canciones más reproducidas y la biblioteca en sí. También se incluyen funciones para cambiar el estilo de los botones y gestionar la interfaz gráfica.

Clase "crear_lista.py". Utiliza PyQt6 para crear las interfaces graficas del programa. Aquí se gestión las listas de reproducción de la música, en este aparatado se importaron los módulos necesarios para el "TDA".

Creación de Elementos de la Interfaz:

- Se crean etiquetas (QLabel) para mostrar títulos en la interfaz.
- Se crean botones (QPushButton) con iconos para realizar acciones como crear listas, guardar y cargar listas.
- Se crea un campo de entrada (QLineEdit) para que el usuario ingrese el nombre de una nueva lista de reproducción.
- Se crean dos tablas (QTableWidget) para mostrar canciones disponibles y listas de reproducción existentes.

Conexión de Señales y Slots:

Python

```
self.crear_btn.clicked.connect(self.crear_playlist)
self.regresar_btn.clicked.connect(self.regresar)
self.guardar_btn.clicked.connect(self.guardar_playlist)
self.cargar_btn.clicked.connect(self.cargar_playlist)
```

Se conectan funciones (métodos) a eventos de botones. Por ejemplo, cuando se hace clic en el botón "Crear lista", se llama a la función crear_playlist.

Métodos para Manipular Datos y Actualizar la Interfaz:

- llenar_tabla: Llena la tabla de canciones disponibles con datos de la biblioteca.
- llenar_tabla_listas: Llena la tabla de listas de reproducción con datos existentes.
- cambiar_estilo: Cambia el estilo de un botón cuando se pasa el ratón sobre él.
- guardar_playlist: Guarda las listas de reproducción en un archivo XML.
- crear_elemento_lista y crear_elemento_cancion: Métodos auxiliares para crear elementos XML.
- cargar_playlist: Carga listas de reproducción desde un archivo XML.
- crear_playlist: Crea una nueva lista de reproducción a partir de canciones seleccionadas.

Función Principal regresar: Cierra la ventana actual

cuando se hace clic en el botón "Regresar"

En resumen, en la clase “crear_lista.py”, interactúa con una biblioteca (TDA) que parece contener estructuras de datos para gestionar la información de la música.

Clase “reproductor_aleatorio.py”. Importa módulos personalizados (TDA y OS) relacionados con el tipo de estructura de datos de la aplicación.

```
from PyQt6.QtWidgets import QWidget, QPushButton, QVBoxLayout, QLabel, QHBoxLayout, QMessageBox
from PyQt6.QtGui import QIcon, QPixmap, QMovie
from PyQt6.QtCore import Qt
import os
import TDA as tda
import random
```

Clase “ventanaReproductorAleatorio” contiene un constructor de clase el cual recibe una instancia de la biblioteca de música “biblioteca” como parámetro.

Atributos Relacionados con la Biblioteca de Música:

- self.biblioteca: Almacena una instancia de la biblioteca de música (tda.Biblioteca) que se pasa como parámetro al constructor. Esta biblioteca parece contener información sobre las canciones disponibles.

Atributos Relacionados con la Reproducción:

- self.playlist_actual: Representa la playlist actual que se está reproduciendo.
- self.cancion_repruduciendo: Almacena la información de la canción que se está reproduciendo actualmente.
- self.reproduciendo_musica: Un indicador booleano que representa si la música está reproduciéndose o no.
- self.pila_reproduccion: Una pila que se utiliza para almacenar las canciones anteriores en caso de que se desee retroceder a la canción anterior.

Atributos para Configurar la Interfaz Gráfica:

- color_fondo, color_botones, color_texto, color_hover: Variables que almacenan códigos de color para la apariencia visual de la interfaz.
- Botones y etiquetas que forman parte de la interfaz gráfica, como self.regresar_btn, self.anterior_btn, self.play_btn, self.siguiente_btn, self.cancion_label, self.artista_label, self.album_label, self.label_portada, self.label_equalizer.

Otros Métodos:

Método __init__ - Configuración de la Ventana:

- Establece el tamaño fijo de la ventana, el título, el icono y el color de fondo.

Método `init - Creación de la Interfaz Gráfica:

- Inicializa botones, etiquetas y otras partes de la interfaz con sus respectivos atributos y estilos.

Método `init - Conexión de Funciones a Botones:

- Conecta funciones específicas a los eventos de clic de los botones.

Método `init - Inicialización del Ecualizador y Obtención de la Playlist:

- Llama a los métodos self.actualizar_ecualizador() y self.obtener_toda_playlist() para configurar el estado inicial del ecualizador y obtener la playlist completa.
- Métodos como actualizar_datos_cancion_actual, actualizar_ecualizador, cambiar_estilo,

reproduccion_cancion, regresar, play, obtener_toda_playlist, anterior, siguiente implementan lógica específica relacionada con la actualización de la interfaz y el control de la reproducción.

En resumen, los atributos se utilizan para almacenar información sobre el estado de reproducción, la interfaz gráfica y la biblioteca de música. La lógica de la clase se centra en proporcionar funcionalidades para controlar la reproducción de música y mostrar información relevante en la interfaz gráfica.

Clase "reproductor_normal.py". Contiene la implementación para permitir que los usuarios puedan elegir entre listas de reproducción, reproducir, pausar y controlar la reproducción de canciones, así como realizar acciones como retroceder, avanzar y activar la reproducción aleatoria.

Clase VentanaReproductorNormal:

Atributos Principales:

- self.biblioteca: Un objeto que representa la biblioteca de música.
- self.listas_reproduccion: Una lista de listas de reproducción disponibles.
- self.playlist_actual: La lista de reproducción actual seleccionada.
- self.cancion_actual: La canción actual en reproducción.
- self.reproduciendo_musica: Un indicador booleano que representa si la música está reproduciéndose o pausada.

Métodos Principales:

- __init__(self, biblioteca, listas_reproduccion): Constructor de la clase.
- actualizar_datos_cancion_actual(self): Actualiza las etiquetas y la portada de la canción actual.
- actualizar_label_lista(self, nombre_lista): Actualiza el texto de la etiqueta que muestra la lista de reproducción actual.
- actualizar_ecualizador(self): Actualiza el visualizador de ecualizador según si la música está reproduciéndose o no.
- cambiar_estilo(self, boton, color, font_size, border_radius, hover=True): Cambia el estilo de los botones en función de eventos de entrada.
- escoger(self): Abre una ventana secundaria para que el usuario elija una lista de reproducción.
- procesar_dato(self, nombre_seleccionado): Procesa la lista de reproducción seleccionada por el usuario.
- reproduccion_cancion(self): Actualiza la reproducción de la canción actual.
- regresar(self): Cierra la ventana actual.
- play(self): Inicia o pausa la reproducción de la canción actual.
- anterior(self): Reproduce la canción anterior en la lista de reproducción actual.
- siguiente(self): Reproduce la siguiente canción en la lista de reproducción actual.
- aleatorio(self): Mezcla la lista de reproducción actual.

Clase VentanaEleccionLista:

Atributos Principales:

- listas_reproduccion: La lista de listas de reproducción disponibles.

Métodos Principales:

- __init__(self, listas_reproduccion): Constructor de la clase.

- actualizar_tabla(self): Actualiza la tabla con las listas de reproducción disponibles.
- celda_clicada(self, row, col): Maneja el evento de clic en la celda de la tabla y emite una señal con el nombre de la lista seleccionada.

En resumen, "reproductor_normal.py" implementa un reproductor de música interactivo con una interfaz gráfica intuitiva utilizando PyQt6, lo que permite a los usuarios gestionar y disfrutar de su biblioteca de música.

Clase "TDA.py". Este script de Python define varias clases y estructuras de datos relacionadas con la gestión de una biblioteca musical. Aquí hay una descripción general de las clases principales y sus funcionalidades:

Clase Artista:

- Atributos: nombre (nombre del artista) y lista_albumes (una instancia de la clase ListaAlbumes que contiene los álbumes del artista).
- Métodos: __str__ para imprimir información del artista.

Clase Album:

- Atributos: nombre (nombre del álbum) e imagen (ruta de la imagen del álbum) y lista_canciones (una instancia de la clase ListaCanciones que contiene las canciones del álbum).
- Métodos: __str__ para imprimir información del álbum.

Clase Cancion:

- Atributos: nombre (nombre de la canción), ruta (ruta de la canción), imagen (ruta de la imagen asociada), artista (objeto de la clase Artista), album (objeto de la clase Album) y reproducciones (número de reproducciones de la canción).
- Métodos: aumentar_reproducciones y resetear_reproducciones para gestionar las reproducciones, y __str__ para imprimir información de la canción.

Clase Registro_Playlist:

- Atributos: nombre (nombre de la playlist) y contenido (una instancia de la clase Playlist que almacena las canciones de la playlist).
- Métodos: aumentar_reproducciones, resetear_reproducciones, set_contenido, get_contenido, y __str__ para imprimir información de la playlist.

Listas y Nodos:

- Nodo: Representa un nodo en las listas doblemente enlazadas.
- ListaDobleEnlazada, ListaArtistas, ListaAlbumes, ListaCanciones: Implementaciones de listas doblemente enlazadas que contienen nodos de artistas, álbumes y canciones, respectivamente.

Clase Playlist:

- Hereda de ListaDobleEnlazadaCircular.
- Métodos adicionales para gestionar playlists, como get_cancion_random, get_cancion_pos, copy, eliminar_cancion y mezclar.

Función graficar en ListaArtistas:

- Utiliza la biblioteca graphviz para generar un gráfico de la biblioteca musical, con nodos para artistas, álbumes y canciones, y enlaces entre ellos

El código también incluye el manejo de excepciones y algunos métodos para manipular las listas de reproducción y generar gráficos de la biblioteca musical.

Conclusiones

Relevancia de la Programación y POO: El proyecto IPCmusic resalta la relevancia de la programación y la programación orientada a objetos en el desarrollo de aplicaciones prácticas. La implementación de estructuras de datos y el manejo de archivos XML son habilidades clave en este contexto.

Colaboración y Participación Activa: La interacción entre el usuario y el modelo reflejó una forma de colaboración y participación activa. La capacidad de generar respuestas y contenidos específicos evidencia el potencial de la inteligencia artificial para facilitar la comunicación y el intercambio de conocimientos.

Referencias bibliográficas

Máximo 5 referencias en orden alfabético.

Tipo de Dato Abstracto. (s/f). Google.com. Recuperado el 19 de diciembre de 2023, de https://sites.google.com/site/programacioniiuno/temario/unidad-2---tipo-abstracto-de-dato/tipo-de-dato-abstracto

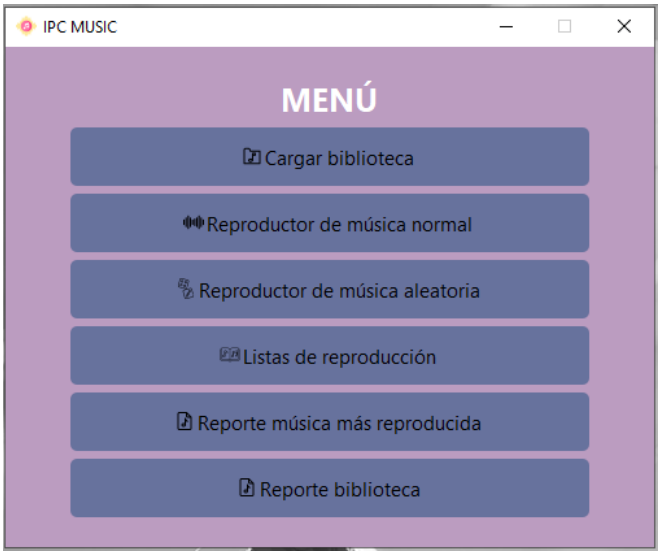
Graphviz — graphviz 0.20.1 documentation. (s/f). Readthedocs.Io. Recuperado el 19 de diciembre de 2023, de https://graphviz.readthedocs.io/en/stable/

Werner, D. A. R. (2018, mayo 4). Graphviz example: How to visualize structured content. *Contentful*. https://www.contentful.com/blog/using-graphviz-to-visualize-structured-content-from-contentful-spaces/

encapsulamiento. (s/f). El Libro De Python. Recuperado el 19 de diciembre de 2023, de https://ellibrodepython.com/encapsulamiento-poo

Manual de Usuario

Inicia el programa abriendo la carpeta en visual studio code, e inicial la clase “menú”

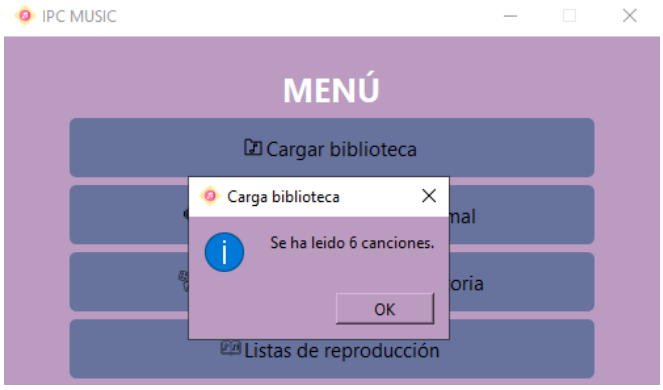


Se desplegará el cuadro de dialogo en el que podrás elegir diversas opciones.

Como usuario tradicional podrás cargar tus canciones en el primer botón. Cabe resaltar que el archivo a ingresar será un XML con la siguiente estructura.

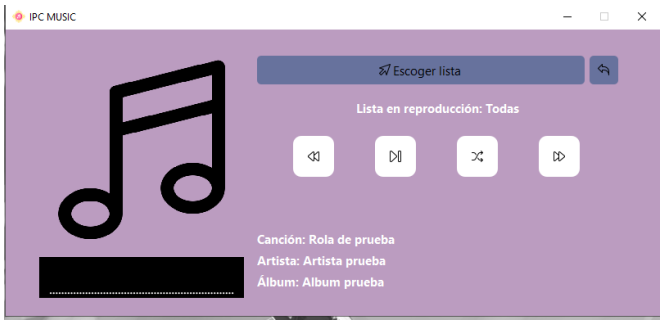
entrada_biblioteca.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<bilbioteca>
    <cancion nombre="Rola de prueba">
        <artista>Artista prueba</artista>
        <album>Album prueba</album>
        <imagen>C:\Users\alanp\Desktop\Proyectos pa\cover_prueba.png</imagen>
        <ruta>"C:\Users\Monica Calderon\Diciembre 2021\IPC2\AOB_BTS.mp3"</ruta>
    </cancion>
    <cancion nombre="Atack of bangtan">
        <artista>BTS</artista>
        <album>O!RUL8,2?</album>
        <imagen>C:\Users\Monica Calderon\Diciembre 2021\IPC2\AOB_BTS.jpg</imagen>
        <ruta>"C:\Users\Monica Calderon\Diciembre 2021\IPC2\AOB_BTS.mp3"</ruta>
    </cancion>
    <cancion nombre="the middle 2">
        <artista>Jimmy Eat World</artista>
        <album>Bleed American</album>
        <imagen>C:\Users\Monica Calderon\Diciembre 2021\IPC2\theMiddleJEW2.jpg</imagen>
        <ruta>C:\Users\Monica Calderon\Diciembre 2021\IPC2\theMiddleJEW2.mp3</ruta>
    </cancion>
    <cancion nombre="the middle">
        <artista>Jimmy Eat World</artista>
        <album>Bleed American</album>
        <imagen>C:\Users\Monica Calderon\Diciembre 2021\IPC2\theMiddleJEW.jpg</imagen>
        <ruta>"C:\Users\Monica Calderon\Diciembre 2021\IPC2\theMiddleJEW.mp3"</ruta>
    </cancion>
    <cancion nombre="Perfectly Perfect">
        <artista>Simple plan</artista>
        <album>Taking One for the Team</album>
        <imagen>C:\Users\Monica Calderon\Diciembre 2021\IPC2\SimplePlan_PP.jpg</imagen>
        <ruta>"C:\Users\Monica Calderon\Diciembre 2021\IPC2\SimplePlan_PP.mp3"</ruta>
    </cancion>
    <cancion nombre="Perfectly Perfect 2">
        <artista>Simple plan</artista>
        <album>Taking One for the Team 2"</album>
        <imagen>C:\Users\Monica Calderon\Diciembre 2021\IPC2\SimplePlan_PP.jpg</imagen>
        <ruta>C:\Users\Monica Calderon\Diciembre 2021\IPC2\SimplePlan_PP2.mp3"</ruta>
    </cancion>
</bilbioteca>
```



Una vez ingresado el archivo XML, te podrás redirigir a las siguientes opciones en el aparatado de “Reproductor de música normal” reproducirás tu playlist conforme el orden del archivo XML ingresado.

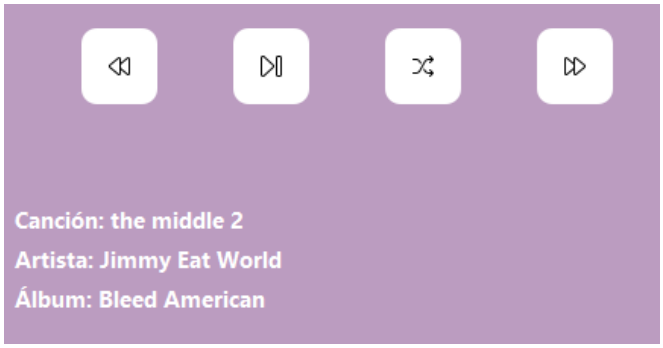
Para poder reproducirlas deveras escoger licha lista y posteriormente darle a play.



Para acceder a la siguiente canción de tu playlist darás click en el icono de siguiente canción. Así mismo puedes acceder a la anterior o bien pausar tus canciones.

(Ejemplo utilizando la opción “Siguiente canción”).

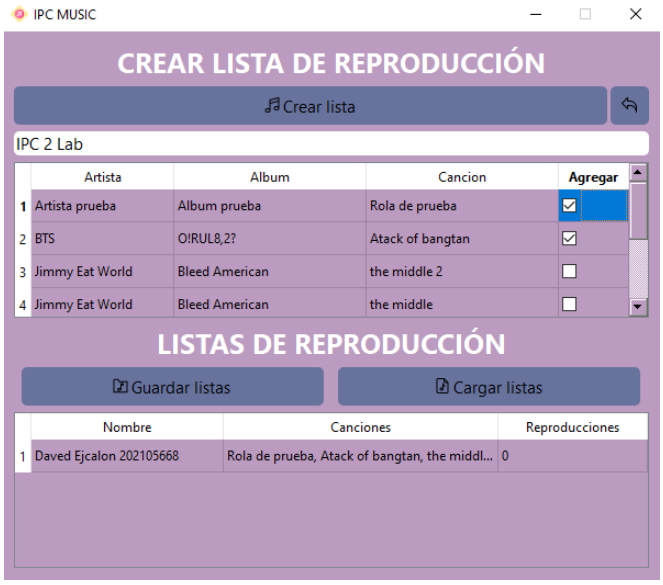
(Ejemplo utilizando la opción “Siguiente canción”).

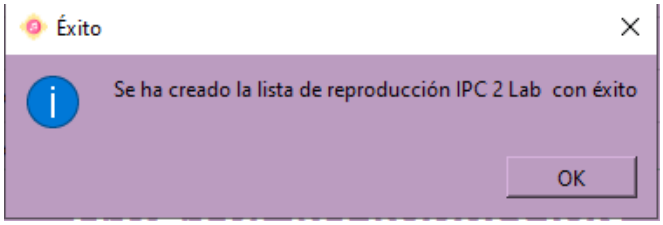


Para regresar al menú principal puedes darle click a el botón superior derecho.

Listas de reproducción

Para utilizar esta fucnion daras click en el botón “listas de reproducción” en el cual podrás acceder a un cuadro de diálogo el cual te permite accede a todas las canciones ingresadas en el XML, allí podrás seleccionar tus canciones favoritas y guardarlo como una lista personal. (Debes nombrar el nombre de tu lista).

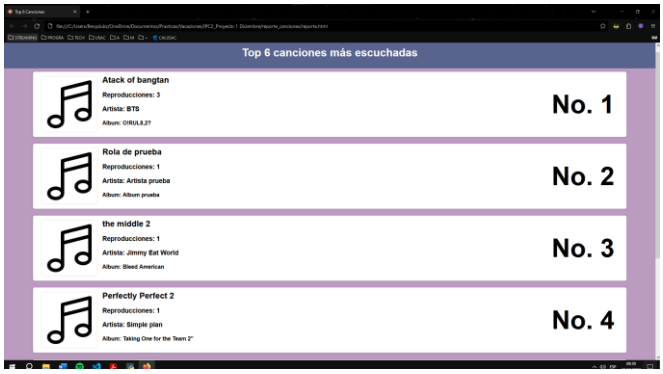




Una vez registrada tu lista, tendrás la opción de guardar tu lista de música como archivo XML, así como cargar otras listas que hayas guardado (Utilizando también la estructura XML antes vista.)

Reporte de música más reproducida

En este aparatado se abre un HTML con tu top de canciones mas reproducidas.



Reporte de biblioteca

Al darle click a este botón se graficará un png con la distribución musical por artista y álbum de cada canción.