Proyecto de Programación Orientada a Objetos

Tecnología de la Programación

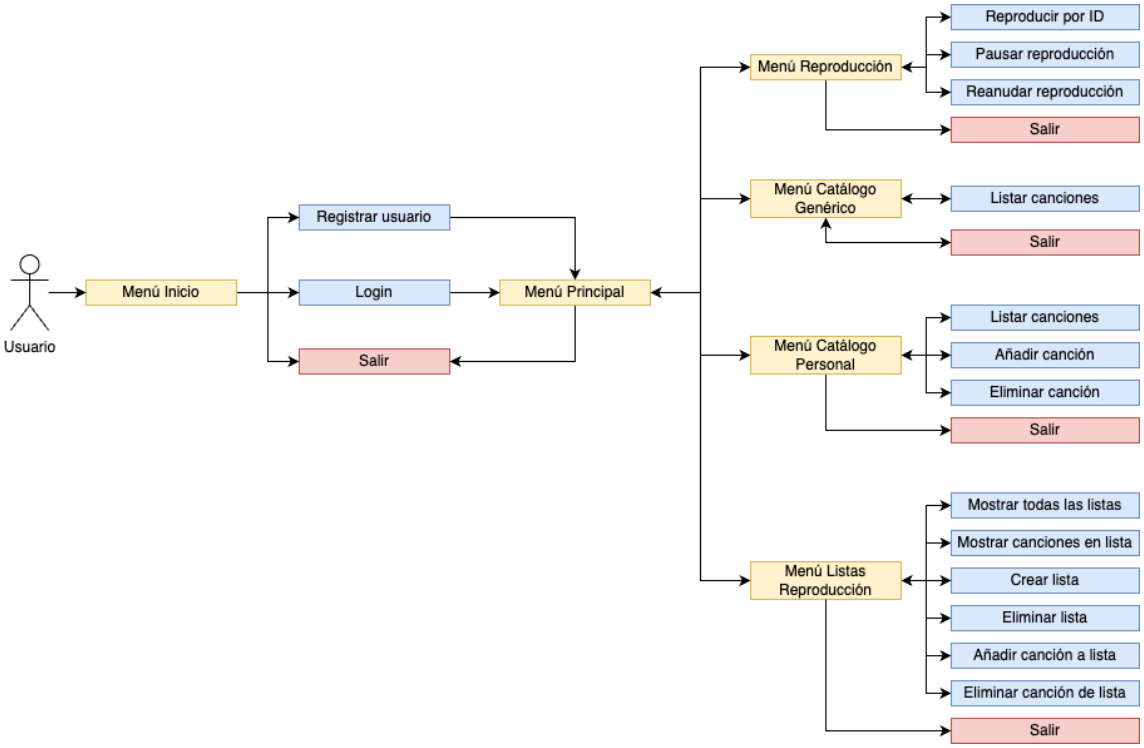
Curso 2025-2026

Convocatoria de Diciembre-Enero

1. Enunciado

El proyecto a realizar es una aplicación de reproducción de música llamada **UMusic**, de tipo Spotify, que permitirá buscar, reproducir o parar canciones, así como crear, editar y borrar listas de reproducción.

El desarrollo del proyecto será mediante interacciones con la terminal, a modo de menús. En cada momento, el usuario puede seleccionar una acción de entre un conjunto predefinido de acciones, pudiendo moverse entre las diferentes funcionalidades de la aplicación. A modo de ejemplo, se muestra un posible flujo de funcionamiento de la aplicación:



En general, en todo menú se puede salir de la aplicación o volver al menú anterior. Este funcionamiento puede ser modificado o adaptado por cada pareja, siempre que se incluyan las funcionalidades descritas a continuación.

Registro de usuarios

El usuario podrá registrarse en el servicio indicando su nombre de usuario (único en el sistema), fecha de nacimiento, correo electrónico, contraseña y la fecha de registro. Habrá dos perfiles de usuario registrado: usuario estándar y usuario premium. Si se trata de un usuario premium, además se deben indicar los datos de la tarjeta de débito (número de 16 dígitos, CVV, fecha de caducidad y nombre completo del titular). Este registro debe implementar persistencia de usuarios y contraseñas (para luego poder validar los inicios de sesión posteriores).

Login de usuarios

Se puede iniciar la aplicación de manera anónima (sin usuario) o con una cuenta previamente registrada. En el segundo caso, se deberá introducir el usuario y la contraseña correcta para poder acceder al menú principal.

Menú principal

Contiene las funcionalidades principales de UMusic para gestionar canciones. Cada canción viene determinada por su identificador (ID) de YouTube, nombre de canción, artista/grupo, género musical y año. El ID es una cadena de texto (de 11 caracteres alfanuméricos sensibles a las mayúsculas y minúsculas) que aparece en una URL de YouTube detrás del "v=". *Por ejemplo, el ID de https://www.youtube.com/watch?v=**dQw4w9WgXcQ** es dQw4w9WgXcQ.*

En particular, el menú principal ofrecerá las siguientes funcionalidades:

1. ***Reproductor***

Cualquier tipo de usuario (anónimo, estándar o premium) puede reproducir cualquier canción proporcionando su ID de YouTube correspondiente. Para la implementación se hará uso del **módulo Reproductor** proporcionado por los profesores, explicado en el Anexo 1. Al iniciar la reproducción, se imprimirán los datos de la canción si existe en el catálogo genérico (para todos los usuarios) o en el catálogo personal (sólo usuarios premium). En caso de no existir en el sistema, se mostrará un mensaje de canción desconocida. Se puede pausar una canción en reproducción, o reanudar una canción pausada. En cualquier momento se podría reproducir una nueva canción. Todas las posibles excepciones lanzadas por el módulo Reproductor deben ser capturadas y tratadas correctamente para evitar que el programa falle.

2. ***Catálogo genérico***

Cualquier tipo de usuario (anónimo, estándar o premium) puede listar las canciones existentes en UMusic, pudiendo filtrar por artista y/o género. Estas canciones son comunes a todos los usuarios y serán proporcionadas en un fichero CSV por los profesores, que se deberá leer y cargar cada vez que se inicie la aplicación. Se mostrará toda la información relativa a cada canción en un formato amigable para el usuario.

3. ***Catálogo personal***

Cada usuario premium tendrá su propio catálogo personal de canciones, capaz de almacenar los datos de cualquier canción de YouTube. En este caso, se podrá listar las canciones existentes en dicho catálogo (pudiendo filtrar por artista y/o género), añadir canción (proporcionando todos los datos de la misma), eliminar canción o volver al menú principal. Los catálogos personales deben ser persistentes a lo largo del tiempo.

4. Listas de reproducción

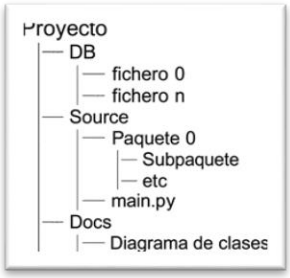
Sólo los usuarios estándar y premium podrán acceder a sus listas de reproducción individuales. Cada lista de reproducción viene dada por un nombre, descripción, lista de canciones, fecha de creación y usuario creador. Se debe permitir al usuario mostrar todas sus listas de reproducción, mostrar las canciones de una lista dada, crear o eliminar una lista, así como añadir o eliminar canciones de una lista. Sólo se pueden añadir canciones existentes en el catálogo genérico o personal (este último sólo para usuarios premium), indicando error en el caso de no ser así. No es necesario implementar la funcionalidad de reproducir una lista completa de reproducción.

Cualquier submenú tendrá siempre las opciones de salir de la aplicación y de volver al menú principal.

2. Normas de entrega del proyecto

La práctica deberá **realizarse obligatoriamente por parejas** y presentarse por el Aula Virtual, vía Tareas, en las fechas establecidas y que serán comunicadas por los profesores de la asignatura.

Todo deberá entregarse en **un único fichero comprimido en formato ZIP**, con la siguiente estructura:



Las subcarpetas contendrán el siguiente contenido:

1. La carpeta **Docs** contiene la documentación de la práctica, que incluye:
    - Un **diagrama de clases UML en formato PDF,** generado con cualquier programa editor como *PowerPoint* o *draw.io*.
2. La carpeta **Source** contiene el proyecto PyCharm completo, que contiene solo paquetes y módulos (SIN incluir la carpeta **venv** del entorno virtual de Python).
3. La carpeta **DB** contiene los ficheros, en el formato elegido, para guardar usuarios y canciones. Debe incluirse como *base de datos* extensa para que el profesor pueda probar todas las funcionalidades pedidas. Las rutas de

lectura y escritura en el código deben ser relativas a esta ubicación (y no absolutas sobre vuestros ordenadores personales).

3. Evaluación del proyecto

Este proyecto constituye el 50% de la nota de prácticas de la asignatura.

El **plagio (total o parcial) en cualquiera de las partes presentadas para su evaluación conllevará una valoración de 0 puntos en la nota final de la convocatoria correspondiente para todos los estudiantes implicados**.

Los profesores usarán diversas herramientas de carácter informático para la detección automática de plagio sobre cualquier entrega realizada.

3.1. Requisitos mínimos

Para aprobar el proyecto, se deben cumplir los siguientes requisitos mínimos. De no cumplir alguno de ellos, **el proyecto se puede suspender automáticamente**:

- **Requisito 1**. Entregar en tiempo, a través de la tarea pertinente, un fichero .zip con todo el proyecto, incluyendo documentación y código, de acuerdo con las normas de entrega indicadas en el apartado 2.
- **Requisito 2**. El programa debe responder a las funcionalidades que se piden en el enunciado de este documento.
- **Requisito 3**. El programa principal se ejecutará sin errores y lo programado funciona correctamente.
- **Requisito 4**. Se usará única y exclusivamente la sintaxis y la funcionalidad de Python vistas en la asignatura, evitando aspectos avanzados. **No se pueden importar módulos externos.**

3.2. Criterios de evaluación y calificación

Los aspectos que se tendrán en cuenta durante la evaluación y calificación del trabajo presentado son los siguientes. Tanto el código como la documentación son de obligada entrega para poder evaluar la práctica.

- Diseño de la solución, distribuyendo la funcionalidad en clases de forma adecuada, presentando un diagrama de clases legible y bien estructurado.
- Funcionamiento correcto de la solución propuesta (no pueden tener errores en tiempo de ejecución).
- Originalidad de la solución al problema planteado.
- Usar correctamente la metodología de la POO en todos los aspectos que han sido mostrados en las sesiones teórico-prácticas, destacando:
    - Empleo de la declaración de tipos explícita y uso correcto de los tipos de datos.
    - Eficiencia y correctitud de las funciones, métodos y algoritmos.

    - Definir correctamente las clases (métodos y atributos) y la visibilidad de cada miembro.
    - Uso correcto de sobrecarga de métodos.
    - Definir las relaciones entre las clases (uso, asociación, agregación, composición y herencia). Uso correcto de delegación de responsabilidades entre clases.
    - Empleo adecuado de clases abstractas e interfaces.
    - Crear una buena estructuración en paquetes y módulos de la solución construida.
- Uso adecuado de excepciones para gestionar errores en el código.
- Uso correcto de entrada/salida en base a los requisitos del problema.
- Diseño y funcionamiento correcto del sistema de menús en la terminal.

Se valorarán positivamente aquellas funcionalidades extra propuestas por los alumnos, siempre que tengan sentido y estén bien implementadas.

4. Entrevista

Todas las parejas serán citadas para realizar una entrevista.

Durante la entrevista se preguntarán cómo están realizadas determinadas partes del proyecto de programación presentado, al tiempo que se podrá solicitar la modificación de alguna parte para que haga algo diferente.

El resultado de la entrevista puede cambiar por completo la calificación obtenida en el proyecto presentado.

Anexo I: Documentación del módulo Reproductor

El módulo "reproductor.py", subido como archivo adjunto a la tarea, permite reproducir el audio de videos alojados en YouTube, usando Python y VLC. **No se permite modificar el código fuente de este módulo**.

Instalación de dependencias

Antes de usarlo, instala las librerías necesarias desde la consola de PyCharm (icono de prompt abajo a la izquierda, o ALT+F12), de la siguiente forma:

- python -m pip install –U yt-dlp
- python -m pip install –U python-vlc
- python -m pip install -U certifi

Esto hará que en el virtualenv (.venv) de tu proyecto, se instalen las dependencias necesarias para hacer funcionar el reproductor. También necesitas tener instalado:

- Programa VLC: el módulo utiliza su motor interno para reproducir audio.

Cómo importar y usar el módulo Reproductor

Una vez que el módulo "reproductor.py" esté incluido en tu proyecto, puedes importar la clase Reproductor y utilizarla de la siguiente manera para crear una instancia del reproductor:

```
from reproductor import Reproductor, EstadoReproductor, ...

    reproductor: Reproductor = Reproductor()
```

Sobre este objeto, se podrá usar la siguiente funcionalidad:

- **obtener_estado_reproductor()**: devuelve el estado actual del reproductor, usando los estados personalizados contenidos en el enumerado EstadoReproductor (REPRODUCIENDO, PAUSADO, SIN_REPRODUCCION)

  Valor de retorno
  - **EstadoReproductor**: estado del reproductor
  - **PlayError**: excepción en caso de que no haya un reproductor inicializado

- **reproducir_desde_youtube(video_id: str, timeout_sec: float)**: método encargado de establecer la comunicación con YouTube para reproducir el audio asociado a un video determinado.

  Parámetros

  - **video_id**: identificador del vídeo a reproducir
  - **tiempo_seg**: tiempo que esperará el método para establecer conexión. Por defecto, se establece en 10 segundos. Si fuese necesario, se podrán indicar tiempos superiores en caso de que la conexión sea más lenta de lo habitual.

  Valor de retorno

  - **True**: si todo ha ido correctamente y se inicia la reproducción
  - **EntradaInvalidaError**: si video_id es inválido
  - **ResolverStreamError**: si no se pudo obtener el recurso solicitado
  - **PlayError**: si no se pudo iniciar la reproducción
- **pausar()**: pausa la reproducción actual

  Valor de retorno

  - **True**: si se ha podido pausar la reproducción
  - **PlayError**: si ha habido un error al pausar

- **reanudar()**: continúa reproduciendo la canción actual

  Valor de retorno

  - **True**: si se ha podido continuar la reproducción
  - **PlayError**: si ha habido un error al reanudar

Cada usuario tendrá su instancia del reproductor, de tipo Reproductor, que usa cuando está interactuando con el sistema. A este objeto se le pide la funcionalidad de reproducir, reanudar o pausar canciones, en función del estado en el que esté. Así, antes de invocar la funcionalidad del reproductor, se deberá consultar su estado actual y, en base a él, realizar la acción deseada.

El módulo "test_reproductor.py" contiene un main de ejemplo, que sirve para poder usar el reproductor. Como se ve en dicho módulo, no es necesario comprender los detalles de implementación del mismo, sino que simplemente lo usamos según la especificación previamente comentada.

Gestión de excepciones

El módulo define una serie de excepciones personalizadas para identificar fácilmente los distintos tipos de errores que pueden ocurrir durante la reproducción de audio:

- **ReproduccionError**: Es la clase base de todas las excepciones del módulo. Todos los errores específicos del reproductor heredan de ella:
    - **EntradaInvalidaError**: Esta excepción se lanza cuando los parámetros de entrada son inválidos, por ejemplo, si se pasa un ID de vídeo vacío o con formato incorrecto.
    - **ResolverStreamError**: Se produce cuando el módulo no puede resolver la URL de streaming de YouTube. Esto suele deberse a errores en la conexión, restricciones regionales o cambios en la estructura del servicio.
    - **PlayError**: Indica que hubo un problema al iniciar o controlar la reproducción a través de VLC. Puede deberse a que el reproductor no se haya inicializado correctamente, que VLC no esté instalado o que haya conflictos con el dispositivo de audio.