Universidad de los Andes > FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

ENTREGA 3 PROYECTO
Programación Orientada a Objetos

Equipo 13

Integrantes:

Carmen Benavente

Roberto Vergara

Luis Zeballos

Profesor:

Carlos Díaz

Andrés Howard

9 de junio de 2020

CONCEPTUALIZACIÓN

Nuestro proyecto Spotiflix está orientado principalmente al tiempo de ocio que el usuario tiene, especialmente en estos tiempos de cuarentena donde hay muchas actividades que se tienen que dejar de lado. Nuestro programa es capaz de reproducir música y películas (tráileres por un tema de peso del programa) así como realizar varias operaciones con estas como agregarlas a playlists, calificarlas y ponerles me gusta. Por otro lado, existe toda un área de seguimiento de personas, donde uno puede acceder a las playlists de ciertos usuarios o a las canciones de los artistas. En las entregas 1 y 2 tuvimos problemas con los métodos innovadores, por lo que para esta entrega implementamos un método innovador en donde uno puede escribir un texto o subir un archivo y hacer que el programa lo lea en voz alta. Como segundo método innovador que queremos implementar para la siguiente entrega estamos principalmente entre 3 ideas: un “Party Mode” que cambiaría la interfaz y tendría métodos para reproducir música como para una fiesta, una opción de poder ver la letra de las canciones y poder traducirla a distintos idiomas, y por último incluir en la interfaz uno o varios juegos del estilo “Flappy Bird” o “Snake”. Todas estas opciones están pensadas más que nada en el tiempo de ocio del usuario, donde uno tendría cubierto varias áreas del entretenimiento dentro de un mismo programa. Implementamos un buscador en nuestro programa para películas, canciones y personas, donde se pueden generar búsquedas de varios tipos de características y con dos datos de la misma característica al mismo tiempo, generando así el buscador con varias combinaciones de and y or. Todo el resto de los requerimientos para esta entrega están completos, tales como cambiar las imágenes, la opción de descargar las canciones y agregar canciones y películas por parte de un administrador. Nuestro programa acepta varios tipos de formato de video y de música, pero por un tema de simplicidad solo están descargadas películas en .mp4 y canciones en .mp3. Para efectos prácticos las canciones y películas que están cargadas en el programa están con datos genéricos como por ejemplo el largo de la canción o la memoria que utilizan. En el manual de usuario se detallan todas las especificaciones que se deben saber para utilizar nuestro programa, así como los requerimientos que tiene. A continuación, se enlistan las distintas clases utilizadas con sus respectivos atributos y métodos con una pequeña descripción de cada uno.

Movies:

Corresponde a la clase que tendrá de objetos a las películas, con todas las especificaciones necesarias para que se pueda desarrollar el programa. Esta clase es Serializable. Además, esta clase contiene un método AgregarPersona(Person p) que agrega personas a la base de datos si que se repitan.

Atributos:

Title: string → Título de la película.

Director: Person → director de la película, corresponde a un objeto de tipo Person.

Actors: List<Person> → Corresponde a una lista con todos los actores involucrados (lista de tipo Person).

Writer: Person → Escritor de la película, objeto de tipo Person.

Lenght: integer → Duración de la película en minutos.

Categories: List<string> → Lista que contiene todas las categorías a las que pertenece la película.

Studio: string → Estudio que realizó la película.

Description: string → Descripción de la película.

Year: string → Año publicación.

Resolution: string → Resolución disponible.

Memory: string → Corresponde al peso de la película (gb).

numReproductions: integer → Cantidad de reproducciones.

Rating: List<double> → Puntuaciones que ha recibido de la película.

RatingProm: double→ Promedio de puntuaciones de la película.

Trailer: string → Nombre del archivo del tráiler.

Video: string → Nombre del archivo con la película.

MovieSongs: List<Songs> → Canciones involucradas en la película.

Min: integer → Corresponde al minuto en que queda la película, por defecto será 0 si nunca ha sido reproducida.

MovieDirection: string → Nombre del archivo de la imagen de la película.

Songs:

La clase Songs es bastante parecida a movies, básicamente sin un atributo video. Songs al igual que movies, crea objetos de tipo canción con sus respectivos atributos correspondientes. Esta clase es Serializable. Cabe destacar que esta clase, tiene dos métodos que no se mencionan, pero que básicamente agregan personas a la base de datos sin que estas se repitan, estos son AgregarPersona(Person p) y AgregarArtista(Artista a).

Atributos:

player: WindowsMediaPlayer → clase utilizada para reproducir la canción en consola.

Title: string → Título de la Canción.

Lenght: integer → Largo de la canción en minutos.

numReproduction: integer → número de reproducciones en la plataforma.

Rating: List<double> → Puntuaciones que recibe la Canción (de 1 a 5).

RatingProm: double → Promedio de puntuaciones.

Genre: List<string> → Lista que contiene los géneros de la canción.

Lyrics: string → Letra.

Composer: Person → Compositor de la Canción de tipo Person.

Resolution: string → Calidad de la canción, del sonido.

Memory: string → Cantidad de memoria que utiliza la canción en el programa.

Artist: Artist → artista de la canción de tipo Artists.

Album: Album → álbum de la canción de tipo álbum.

Music: string → Nombre del archivo de música.

Writer: Person → Escritor de la canción de tipo Person.

Type: string → tipo de archivo de música.

Min: integer → Corresponde al min en qué queda la canción, por defecto será 0 a no ser que se reproduzca y no se termine.

Files:

Files corresponderá a una clase estática que guarde listas de todas las películas, canciones, playlists, usuarios y personas, básicamente corresponderá a nuestra Base de Datos. Esta clase es Serializable por lo que es la encargada de guardar la información cuando el programa se cierre.

Atributos:

AllMovies: List<Movies> → Lista de Movies.

AllSongs: List<Songs> → Lista de Songs.

AllPlaylistsSongs: List<Playlists> → Playlists de canciones.

AllPlaylistsMovies: List<Playlists> → Playlists de películas.

AllUsers: Dictionary<int, List<string>> → Base de datos de User en el programa.

Users: List<User> → Lista de usuarios guardados.

AllPersons: List<Person> → Base de datos de las personas en el programa.

Métodos:

Get_Lista_Usuarios () → Método para mostrar los usuarios.

ChangePasword () → Método para cambiar la contraseña.

AddUser () → Método para agregar a un usuario.

GetData () → Método para obtener los datos de un usuario.

LogIn () → Método para realizar el LogIn.

Search:

Search corresponde a una clase estática que nos funcionará como un buscador, es decir, que desde palabras claves, personas, características, etc. nos encontrará las películas, canciones y personas relacionadas.

Métodos:

SearchingMovies () → Método llama a las otras funciones de search para las películas.

SearchingSongs () → Método llama a las otras funciones de search para las canciones.

SearchingPerson () → Método llama a las otras funciones de search para las personas.

QueryTitle () → Método que busca las películas con un título específico.

QueryPerson () → Método que busca las películas con una persona específica.

QueryCategories () → Método que busca las películas con una categoría específica.

QueryYear () → Método que busca las películas con un año específico.

QueryRating () → Método que busca las películas con un rating mayor o igual a uno específico.

QueryNumRep () → Método que busca las películas con un número de reproducciones mayor o igual a uno específico.

QueryTitleS () → Método que busca las canciones con un título específico.

QueryPersonS () → Método que busca las canciones con una persona específica.

QueryGenreS () → Método que busca las canciones con un género específico.

QueryNumRepS () → Método que busca las canciones con un número de reproducciones mayor o igual a uno específico.

QueryRatingS () → Método que busca las canciones con un rating mayor o igual a uno específico.

QueryAlbumS () → Método que busca las canciones con un álbum específico.

QueryName () → Método que busca las personas con un nombre en específico.

QuerySexo ()→ Método que busca las personas con un sexo en específico.

QueryEdad () → Método que busca las personas con una edad en específico.

GetAge ()→ Método que calcula la edad de una persona.

User:

Corresponde a la clase del Usuario el cual guarda su información personal como nombre, email y contraseña. Esta clase tiene tres constructores distintos, uno para implementar el crear cuenta, otro para un usuario free y uno para usuario premium. Se dejó "Admin" sin ser un objeto usuario.

Atributos:

Type: string → Tipo de usuario, puede ser Free o Premium.

UserName: string → ID del usuario.

Email: string → Email con el que se registró.

Password: string → Contraseña.

Login: boolean→ Si está abierta la cuenta en el programa o no.

Privacy: boolean → Si la cuenta es privada o no.

MyPlaylist: List<Playlist> → Playlist del usuario.

Follows: List<User> → Usuarios a los que sigue el usuario.

Follows2: List<Person> → Personas a las que sigue el usuario.

Métodos:

CheckCredentials () → Método que revisa que la contraseña y usuario ingresados sean correctos.

Person:

Clase del tipo persona, representa a una persona con sus atributos. También es padre de la clase Artist. Esta clase es Serializable.

Atributos:

Name: string → Nombre de la persona.

Birthday: date → Fecha de nacimiento de la persona.

Genre: char → Género de la persona.

Link: string → Enlace a wikipedia o imdb con la info de la persona.

Tipo: string → Tipo de una persona.

numReproduction: int → cantidad de reproducciones (pensado en heredar en artistas).

Artist:

Clase hija de Person, es la clase que representa al artista. El tipo es específico de “Artista”.

Álbum:

Clase que representa álbumes.

Atributos:

Name: string→Nombre del álbum.

Year: Date→Año de lanzamiento del álbum.

Image: string → Nombre de la imagen del álbum.

Artist: Artist→Artista del álbum.

Computer:

Quedó como la clase encargada de crear las playlist y smartplaylist.

Métodos:

CreatePlaylist () → Método encargado de crear un objeto Playlist.

CreateSmartPlaylist () → Método encargado de crear un objeto SmartPlaylist.

Playlist:

Esta clase es la encargada de generar listas con las canciones o películas que se le entregan. Es la clase padre de SmartPlaylist.

Atributos:

List<Songs> → lista de canciones que puede o no estar vacía.

List<Movies> → lista de películas que puede o no estar vacía.

Name: string → nombre de la lista.

Type: string → Ve si es una playlist de películas o de canciones.

SmartPlaylist

Esta clase es la que agrega a la SmartPlaylist todas las películas o canciones que cumplan con el criterio especificado. Clase que se hereda de la clase Playlist.

Atributos:

Criterio: string → Criterio de la smartplaylist.

NameCriterio: string → Valor del criterio de la smartplaylist.

Clases de Eventos:

ChangePasswordEventArgs (para el cambio de contraseña), LoginEventArgs (para el inicio de sesión) y RegisterEventArgs (para registrar una cuenta).

Clase de Controllers:

UserController → clase encargada de toda la interfaz de inicio de seción y creación de cuenta.

AppForm:

En esta clase se implementan todos los métodos que hacen funcionar el programa, pasó a ser la clase encargada de llamar a todos los métodos y recibir todos los eventos de clics de botones y cosas por el estilo. A continuación, se muestran los atributos de la clase y los métodos más importante por un tema de orden.

Atributos:

LoginEventHandler: delegate.

LoginButtonClicked: evento LoginEventHandler.

UserChecked: EventHandler<LoginEventArgs>

CreateAccountEventHandler: delegate.

CreateAccountClicked: evento CreateAccountEventHandler.

Ruta: string → ruta del elemento a reproducir.

Dest: string → ruta donde se debe guardar un elemento.

Name: string → nombre del elemento a descargar.

Imagen: string → ruta a la imagen a cambiar.

Queuesongs: Queue<Songs> → objeto Queue de Songs.

Queuemovies: Queue<Movies> → objeto Queue de Movies.

Qpeliculas: WMPLib.IWMPPlaylist → playlist de Windows Media player.

Qcanciones: WMPLib.IWMPPlaylist → playlist de Windows Media player.

Datetime2: DateTime → Fecha actual.

vocesInfo: List<VoiceInfo> → Voces para el método innovador.

synthVoice: SpeechSynthesizer → sintetizador de voz.

isStopped: boolean → si la voz está parada.

Speaking: boolean → Si la voz está hablando.

stackPanels: List<Panels> → Lista de paneles que se van poniendo encima.

Panels: Dictionary<string, Panel> → Diccionario de todos los paneles existentes.

Métodos:

IniciarSerializacion() → Método que inicia la serialización de todas las clases serializables.

Serializacion() → Método que serializa todas las clases serializables.

ShowLastPanel() → Método que manda al frente el panel agregado al final de stackPanels.

setNameUser() → Método que muestra en una etiqueta el nombre del usuario.

OnUserChecked() → Método que chequea la información del usuario.

FillDataGriedViewSongs() → Método que rellena el gried con las distintas canciones pedidas.

FillDataGriedViewMovies() → Método que rellena el gried con las distintas películas pedidas.

RellenarInfoMovies() → Método que entrega la información de una película.

RellenarInfoSongs() → Método que entrega la información de una canción.

SacarRuta() → Método que saca la ruta completa de donde se encuentra un archivo.

GetUser() → Método que retorna el usuario que está activo.

FillDataGriedBuscadorSongs() → Método que rellena el gried con las canciones buscadas.

FillDataGriedBuscadorMovies() → Método que rellena el gried con las películas buscadas.

FillDataGriedMyPlaylist() → Método que rellena el gried con las playlist del usuario activo.

FillDataGriedAgregarPlaylist() → Método que recibe información para crear una playlist.

ChooseFolder() → Método que abre un cuadro de diálogo para elegir una carpeta donde descargar las canciones.

FillDataGriedFollowers() → Método que rellena el gried con las personas que se siguen.

FillDataGriedVerPlaylistUsuarioExterno() → Método que rellena el gried con las playlist de un usuario que no es el activo.

FillDataGriedBuscadorFollowers() → Método que rellena el gried con las personas buscadas.

FilldataGriedPanelPersona() → Método que rellena el gried con la información de una persona.

Copy() → Método que descarga la canción seleccionada.

AgregarCancion() → Método que agrega una canción.

FillDataGriedEliminarS() → Método que rellena el gried con las canciones a eliminar.

FillDataGriedEliminarM() → Método que rellena el gried con las películas a eliminar.

AgregarPelicula() → Método que agrega una película.

AgregarImagenCancion() → Método que agrega una imagen a un álbum.

AgregarImagenPelicula() → Método que agrega una imagen a una película.

QueueSongs() → Método que agrega una canción al queue.

QueueMovies() → Método que agrega una película al queue.

ReproducirPlaylist() → Método que reproduce una playlist.

SumNumRep() → Método que cambia el número de reproducciones de una canción, artista o película.

ReproducirQueueMovies() → Método que inicia la reproducción del queue de películas.

ReproducirQueueSongs() → Método que inicia la reproducción del queue de canción.

Reproducir() → Método que reproduce el elemento seleccionado.

Botones e Interacciones:

LoginButtonLoginView_Click() → Realiza todos los métodos para que el usuario inicie sesión.

OnLoginButtonClicked() → Envía el evento de inicio de sesión.

RegistrationButton_Click() → Crea al usuario con los datos ingresados.

OnCreateAccountClicked() → Envía el evento de crear cuenta.

CerrarSesiónButton_Click() → Cierra la sesión del usuario.

ModificarCuentaButton_Click() → Abre el panel donde se puede cambiar los datos del usuario.

BackToUserMenuButton_Click() → Retorna al panel inicial.

AceptarCambioNombreUsuarioButton_Click() → cambia el nombre del usuario.

CambiarUsuarioButton_Click() → Abre panel donde se puede cambiar al usuario.

CambiarContraseñaButton_Click() → Abre el panel donde se puede cambiar la contraseña.

AceptarNuevaContraseñaButton_Click() → Cambia la contraseña del usuario.

AceptarCambioCuentaButton_Click() → Cambia el tipo de cuenta del usuario.

CambioTipoCuentaButton_Click() → Muestra el panel donde se puede cambiar el tipo de cuenta del usuario.

VerTodasLasCancionesButton_Click() → Muestra todas las canciones que hay en el programa.

VerTodasLasPeliculasButton_Click() → Muestra todas las películas que hay en el programa.

DataGriedMovieS_CellContentClick() → Muestra la información de la película seleccionada.

DataGriedSongS_CellContentClick() → Muestra la información de la canción seleccionada.

CrearPlaylistButton_Click() → Muestra el panel donde se ingresan los datos para crear una playlist.

AceptarPlaylist_Click() → Crea una playlist con la información ingresada.

NombrePlaylist_TextChanged() → Muestra información necesaria para crear una SmartPlaylist.

TipoPlaylist2_SelectedIndexChanged() → Define el criterio para la smartplaylist.

dataGridBuscadorMovies_CellContentClick() → Muestra la información de la película seleccionada en el buscador.

dataGridBuscadorSongs_CellContentClick() → Muestra la información de la canción seleccionada en el buscador.

SearchButton_Click() → Busca la información puesta en el panel.

CambiarNombrePlaylist_Click() → Muestra el panel donde se ingresa el nuevo nombre.

Aceptarcambiodenombre_Click()→ Cambia el nombre de una playlist seleccionada.

linklabelPerson_LinkClicked() → Se abre el enlace mostrado donde se muestra la página de Wikipedia o imdb.

dataGridFollowers_CellContentClick() → Muestra la información de la persona seleccionada.

dataGridVerPlaylistUsuarioExterno_CellContentClick() → Muestra la información de la playlist de un usuario externo.

dataGriedBuscadorFollowers_CellContentClick() → Muestra la información de una persona seleccionada en el buscador.

DescargarButtonShowSong_Click() → Inicia los métodos para descargar la canción seleccionada.

CambiarImagenShowSong_Click() → Inicia todos los métodos para cambiar la imagen de un álbum.

CalificarButtonShowSong_Click() → Agrega la calificación seleccionada a la información de la canción.

MeGustaButtonShowSong_Click() → Agrega canción a la playlist de me gustas del usuario.

CalificarShowMovie_Click() → Agrega calificación seleccionada a la información de la película.

MeGustaShowMovie_Click() → Agrega la película a la playlist de me gustas del usuario.

AddPlaylistShowSong_Click() → Muestra el panel para agregar la canción a una playlist.

AddPlaylistShowMovie_Click() → Muestra el panel para agregar la película a una playlist.

CambiarImagenShowMovie_Click() → Inicia todos los métodos para cambiar la imagen de la película.

dataGriedPanelPersona_CellContentClick() → Muestra la información de la persona seleccionada.

AgregarElemntoPlaylist_Click() → Muestra el panel para agregar elementos a la playlist seleccionada.

EliminardePlaylist_Click() → Muestra el panel para eliminar el elemento de una playlist.

AddToFollowersButton_Click() → Agrega un objeto Person a los seguidores del usuario.

FollowUserButton_Click() → Agrega un objeto User a los seguidores del usuario.

backbutton_Click() → Reproduce el elemento anterior de la lista que se está reproduciendo.

playpausebutton_Click() → Detiene o inicia la reproducción del elemento que se estaba reproduciendo.

nextbutton_Click() → Reproduce el siguiente elemento de la lista que se está reproduciendo.

dataGridAgregarAPlaylist_CellContentClick_1() → Agrega el elemento seleccionado a una playlist.

dataGridVerPlaylist_CellContentClick_1() → Muestra la información de la playlist seleccionada.

dataGridEliminar_CellContentClick() → Elimina el elemento de la playlist.

AbrirArchivo_Click() → Abre el archivo para reproducir en voz alta.

Hablar_Click() → Inicia la reproducción en voz alta del texto.

PauseButton_Click() → Detiene o inicia la reproducción en voz alta del texto.

StopButton_Click() → Cierra la reproducción en voz alta del texto.