



Aplicación PlayList en JAVA

Examen 4 de junio de 2019

Estructura

En este documento incluyo la estructura de la aplicación, con las clases y métodos utilizados

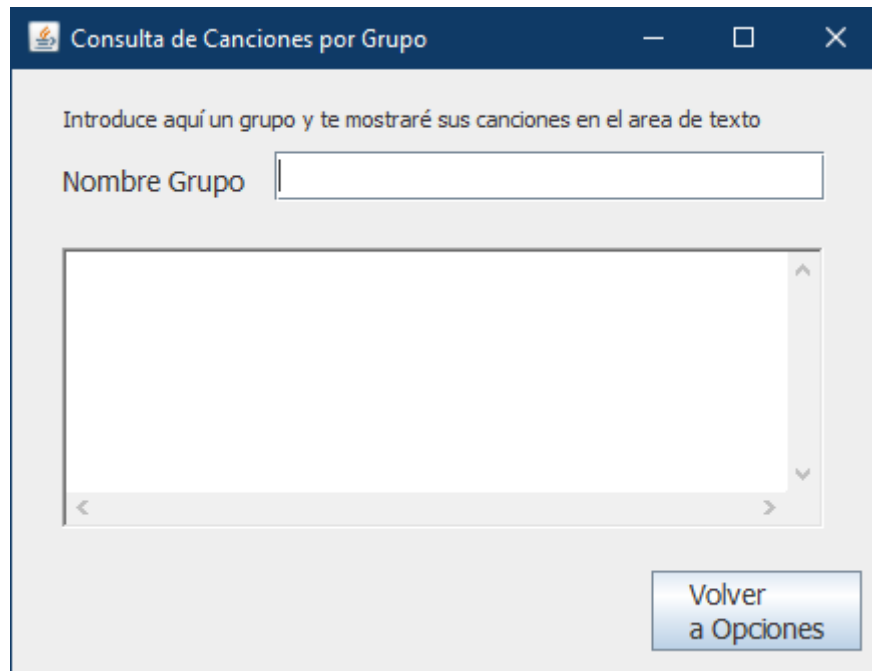
Robert Gutiérrez
jrgutinez@gmail.com

Estructura de la aplicación

Clases utilizadas:

- **AccesoDatos**
 - ❖ Métodos
 1. **ConectarBD()** – Para conectar con la BD
 2. **ConsultaBD()** – Para hacer consultas a la BD
 3. **Insert()** – Para añadir datos a la BD
 4. **Delete()** – Para borrar datos de la BD
 5. **cerrarConexion()** – Para desconectar de la BD
- **Administrador - Hija de Usuarios**
 1. Sin métodos, solo variables, getters y setters.
- **Canciones**
 - ❖ Métodos
 1. **bajaCancion()** – Para borrar canciones de la BD
 2. **altaCancion()** – Para añadir canciones a la BD
 3. **checkExisteCancion()** – Para comprobar si la canción existe en la BD
 4. **CrearFicheroCanciones()** – Para crear un fichero canciones.csv desde la BD
 5. **leerCsv()** – Para leer los ficheros csv
Además, dentro relleno los ArrayList de listagrupos y listacanciones para usarlos después.
 6. **crearListaReproduccion()** – Para crear lista de reproducción – Lo uso en Ventana ListaRepro.

- ConsultaGrupo



1. Ventana JFrame que llama al método comprobarGrupo de la clase Grupos y uso un ArrayList<String> al que he llamado canAL → (canciones Array List).

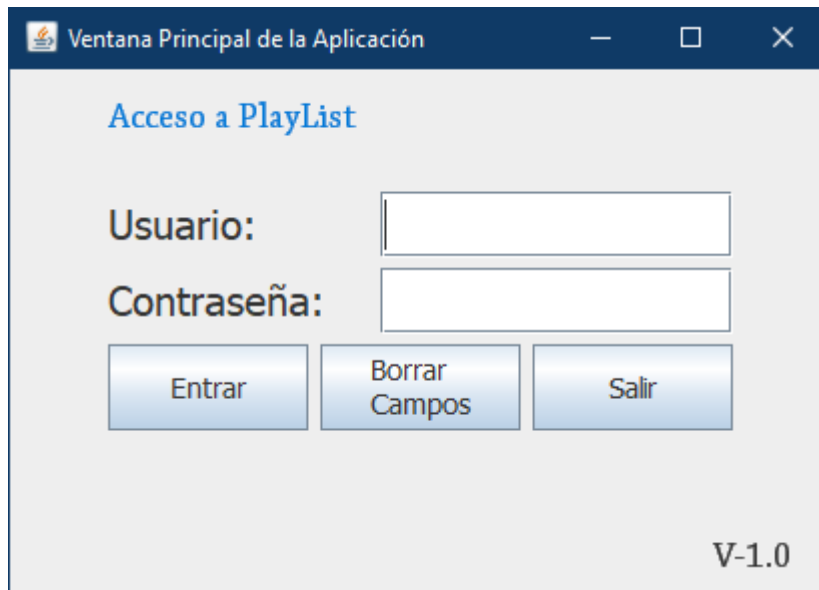
- Grupos

En esta clase implemento una interface que he llamado `valida` para usar métodos de comprobación en diferentes puntos.

❖ Métodos

1. `altaGrupo()` – Para añadir grupos a la BD
2. `bajaGrupo()` – Para borrar grupos de la BD
3. `crearFicheroGrupos()` – Para crear un fichero grupos.csv desde la BD
4. `valido2()` – Para impedir la entrada de datos String en int a través de interface valida
5. `checkCodigo()` – Para comprobar si el código introducido existe en la tabla grupos de la BD
6. `leerCsv()` – Para leer los ficheros csv
Además, dentro relleno los ArrayList de listagrupos y listacanciones para usarlos después.
7. `comprobarGrupo()` – Para comprobar si existe el grupo. Si existe, devuelve sus canciones. Lo consigo con for de los ArrayList → listacanciones y listagrupos llenando ArrayList<String> canAL y uso este método en la ventana ConsultaGrupo

- InicioLR → Ventana principal de la app Lista de reproducciones

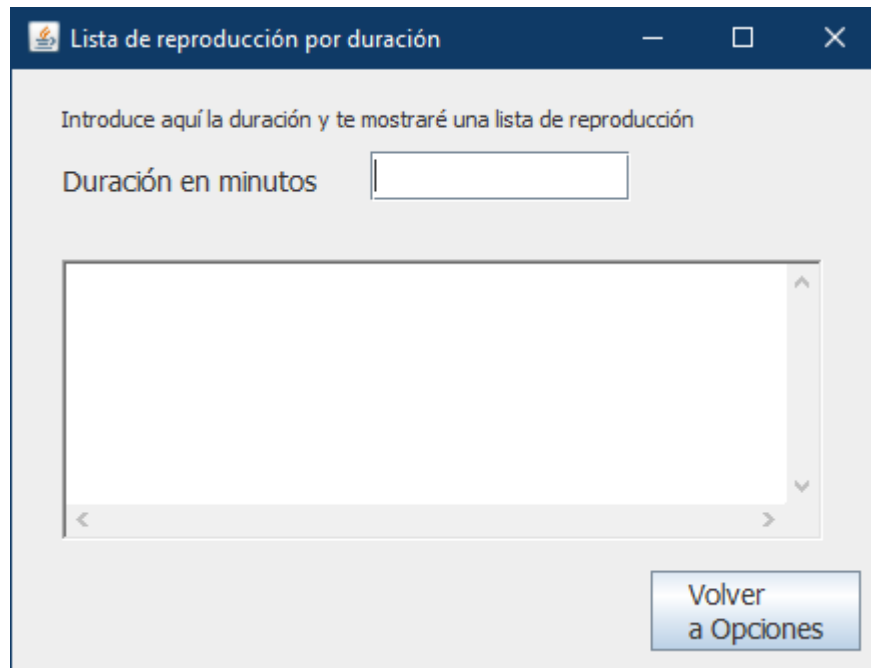


1. Ventana Principal de la aplicación donde se solicita al usuario que introduzca sus credenciales (Nombre de usuario y contraseña) y dependiendo de si es administrador o usuario final iremos a una zona privada o iremos a otra ventana de opciones de usuario final.

❖ Métodos

1. `checkUser()` – Para comprobar si el usuario existe en la BD
2. `checkAuth()` – Para comprobar si el tipo de usuario en la BD
3. `borrarCampos()` – Para borrar campos de la ventana principal

- ListaRepro – Ventana



1. Sin métodos, solo variables y llama a otros métodos

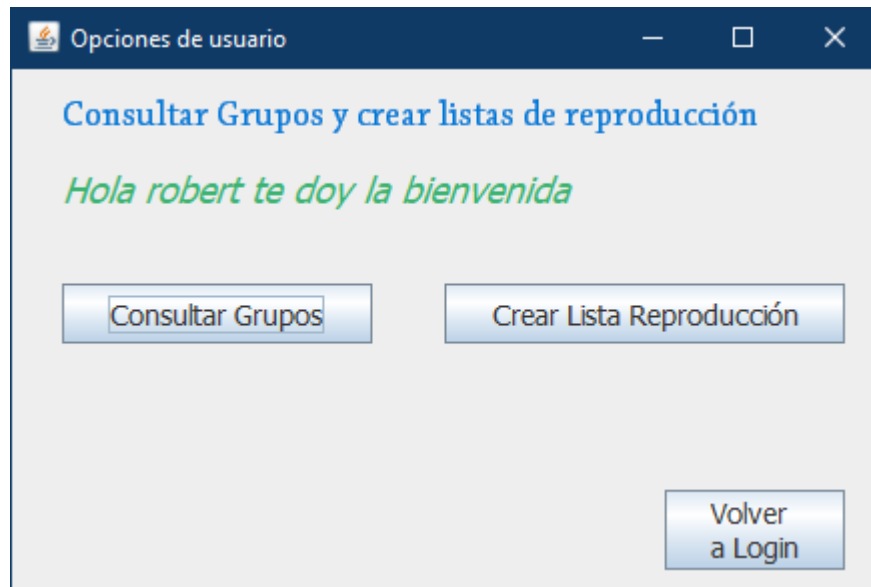
- MenuAdmin

Zona de administración

1. Dar de alta nuevo grupo en la BD
2. Dar de baja un grupo borrando todas sus canciones
3. Dar de baja una canción de un grupo
4. Dar de alta una canción de un grupo
5. Generar usuarios.txt - Consulta de usuarios
6. Generar grupos.csv - Consulta de grupos
7. Generar canciones.csv - Consulta de canciones
8. Volver a la ventana de login
9. Salir del programa

1. Sin métodos, solo variables, ArrayList de canciones y grupos y llama a otros métodos, esta zona solo es accesible a los administradores.

- Opciones



1. Sin métodos, solo variables y llama a otros métodos, esta zona es accesible a usuarios finales.

- Personas – Hija de Usuarios

- ❖ Métodos

1. `CrearFicheroUsuario()` – Para crear un fichero txt de los usuarios finales desde la BD.

- Usuarios – Clase padre (Abstracta)

- ❖ Métodos

1. `Valido()` – Es un método que he dejado en esta clase por si en algún momento añado una opción más al menú de administración para dar de alta usuarios nuevos y así comprobar si su dni es correcto o no. Este método depende de la interface valida.
2. `Valido2()` – Es un método para impedir que se introduzcan datos String en un campo de números enteros. Este método depende de la interface valida.

- Valida

- ❖ Métodos

1. Contiene los métodos booleanos `valido` y `valido2`, aunque se pueden añadir más si queremos.