

Trabalho prático de Programação
Orientada a Objetos
Grupo 37

Dinis Costa A106872

Rita Cunha A104261

Rui Faria A106899

Constituintes do Grupo



Dinis Costa



Rita Cunha



Rui Faria

Índice

Constituintes do Grupo	2
1. Arquitetura de Classes	4
1.1. Entidades da Aplicação	4
1.1.1. Delegate	4
1.1.2. Model	5
1.1.3. Util	6
1.1.4. Exceptions	7
1.1.5. Testes	7
2. Descrição da Aplicação	8
2.1. Como Executar	8
2.2. Uso da Aplicação	8
2.2.1. Uso como administrador	9
2.2.2. Uso Como Utilizador	9
3. Conclusão	12

1. Arquitetura de Classes

Nesta secção, descreve-se e justifica-se a arquitetura de classes da aplicação SpotifUM. O diagrama de classes encontra-se disponível no ficheiro “DiagramaDeClassesG37.pdf”.

1.1. Entidades da Aplicação

O nosso projeto divide-se em 3 *packages* principais:

- **delegate**, onde optamos por uma variação do MVC onde cada menu (View) inclui diretamente os handlers (Controller), formando classes ‘delegate’ que simplificam a navegação.
 - **model**, que representa a base de dados do projeto, onde os objetos são guardados e ondem estão os métodos para os tratar.
 - **util**, que contém outras necessidades, como o parser e adaptadores usados no mesmo.
- Temos, por fim, outros dois *packages*: **test**, para as classes relacionadas a testes unitários e **excpetions** para exceções.

1.1.1. Delegate

O *package* **delegate** contém as classes responsáveis pela interface em linha de comandos (CLI) e pelo controlo de fluxo da aplicação, combinando View e Controller num só módulo:

SpotifUMUI

Ponto de entrada da aplicação. Exibe o menu principal, permitindo ao utilizador escolher entre iniciar sessão como Admin ou como User, ou registar-se como novo utilizador.

AdminUI

Disponibiliza todas as operações de gestão: importação/exportação de dados (JSON e serialização Java), criação, listagem e remoção de álbuns, utilizadores e playlists, além das estatísticas do sistema.

UserUI

Garante a experiência do utilizador final autenticado: gestão da subscrição (Free, PremiumBase, PremiumTop), criação e edição de playlists, visualização de álbuns e playlists públicas, e acesso ao leitor de música.

PlayerUI

Implementa o simulador de reprodução: apresenta letra, informações extra sobre músicas multimédia e explícitas e aceita comandos em tempo real (n para next, p para previous, s para stop e r para shuffle).

1.1.2. Model

No *package* **model** definimos as entidades centrais do SpotifUM:

Álbuns

Cada Album mantém internamente um `ArrayList<Song>` — escolha que facilita iterar, adicionar e remover faixas. Para além da lista de músicas, armazenamos atributos como título, artista, ano de lançamento, género e a referência à “faixa atual”.

Músicas

As músicas, classe `Song` só existem no contexto de um Album: adotámos composição para que o ciclo de vida de cada música seja gerido pela classe Album. Cada *Song* contém informação essencial (nome, intérprete, editora, letra, notação musical, género e duração) e um contador de reproduções.

Para suportar tipos especiais de faixas (música explícita ou multimédia), criámos subclasses de `Song` que acrescentam campos e métodos específicos — mantendo toda a estrutura base de uma música “normal”.

Playlists

Ao contrário do que acontece nos álbuns, as playlists não gerem o ciclo de vida das músicas, mas também tem de as conter, por isso optámos por usar agregação.

As playlists contêm dados que nos ajudem a tratá-las, criador, nome, privacidade, música atual a ser tocada e também um `ArrayList` de músicas, mais uma vez, devido à sua fácil iteração.

Interface “Playable”

Como tanto a playlist como o album servem para tocar músicas, foi fácil notar que haveriam métodos comuns entre os dois, então criámos a interface `Playable`, a ser implementada por ambos, onde estariam os métodos referentes a tocar música, passar para o próximo ou retroceder na música a tocar.

Utilizadores e Subscrições

Os users são a entidade principal para o qual a aplicação foi criada, têm todos os dados necessários para a sua identificação e tratamento, nome, email, password, pontos e além disso contém uma subscrição e uma lista de históricos.

Cada utilizador tem um `SubscriptionPlan` — `Free`, `PremiumBase` ou `PremiumTop` — que implementa a interface `SubscriptionPlan`. Esta estratégia (padrão Strategy) isola as

regras de cada plano (por ex., “pode criar playlist?”, “pode retroceder faixas?”), simplificando a adição de novos planos no futuro: caso seja necessário implementar novos tipos de subscrição, nenhuma outra classe vai ser afetada pela mudança, pois os users apenas verificarão as permissões que tem. Através do tipo de subscrição é possível saber o que é que o user pode ou não fazer na aplicação.

Histórico de Reprodução

A classe History regista cada música reproduzida (com timestamp) e pertence por composição a User. Sempre que um utilizador ouve uma música, cria-se um novo History que fica guardado na lista interna do User.

Facade de Dados – SpotifUMData

A classe spotifumData representa o banco de dados da aplicação, contendo três HashMaps, um de Users, outro de Playlists e outro de Álbuns. Para a gestão dos mesmos (adição, remoção, edição). Decidimos usar Hashmaps devido a eficácia e velocidade dos métodos de procura, necessitando apenas de uma chave para encontrar o objeto que procurámos, em tempo constante, facilitando o tratamento de informação.

1.1.3. Util

O *package* util contém as classes de apoio encarregues de operações transversais de parsing e estatísticas:

JsonDataParser

Gere a importação e exportação de todo o estado de SpotifUMData, tanto em modo “ficheiro único” como em modo multi-ficheiro (users.json, albums.json, playlists.json). Utiliza a biblioteca Gson com adaptadores personalizados para suportar tipos complexos.

SongTypeAdapter

Permite o parsing polimórfico de faixas: distingue, no JSON, entre Song, MultimediaSong (com vídeo) e ExplicitSong. Usa flags ("multimedia", "explicit") e um delegado Gson para desserializar e serializar corretamente cada subclasse.

LocalDateTimeAdapter

Implementa JsonSerializer e JsonDeserializer para converter objetos LocalDateTime em strings ISO-8601 e vice-versa, garantindo compatibilidade e legibilidade nos ficheiros JSON.

SubscriptionPlanDeserializer

Regista um type adapter factory que, lendo o campo "type" no JSON, instancia a subclasse correta de SubscriptionPlan (FreePlan, PremiumBase ou PremiumTop), mantendo a lógica de estratégia de pontos e permissões de cada plano.

Stats

Conjunto de métodos estáticos que processam cópias dos mapas de dados e calculam métricas de utilização:

- música mais reproduzida
- artista mais ouvido
- utilizador que ouviu mais músicas
- utilizador que ouviu mais musicas desde uma determinada data
- utilizador com mais pontos
- género mais tocado
- contagem de playlists públicas
- utilizador que criou mais playlists

1.1.4. Exceptions

Foram implementadas três exceções:

- AlreadyExistsException, quando tentámos inserir alguma entidade no seu map mas ela já existe.
- DoesntExistException, quando procurámos uma entidade que não está guardada no map
- SubscriptionDoesNotAllowException, quando um user tenta tomar uma ação para a qual não tem permissão.

1.1.5. Testes

Foram implementados ainda testes unitários para grande parte das classes do programa. Estes são baseados em JUnit5 e encontra-se disponíveis no *package* test, podendo ser corridos com o comando `./gradlew test`

2. Descrição da Aplicação

2.1. Como Executar

1. Clone o repositório e entre na pasta do projeto
2. Execute o comando:

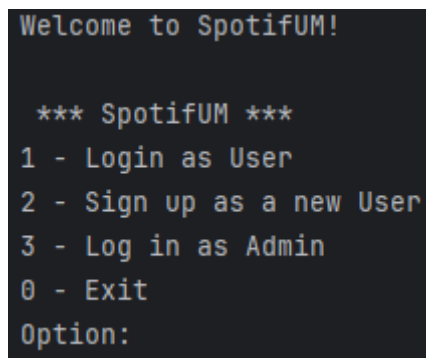
`./gradlew run`

Nota: Pode ser necessário executar previamente o comando `./gradlew build`.

Para correr os testes unitários basta executar: `./gradlew test`

2.2. Uso da Aplicação

Menu Principal:



```
Welcome to SpotifUM!

*** SpotifUM ***
1 - Login as User
2 - Sign up as a new User
3 - Log in as Admin
0 - Exit
Option:
```

Figura 1 - Menu Principal do SpotifUM

Opções:

- Login como Utilizador
- Registar Utilizador
- Login como Admin

Digite 3 e utilize o user admin123 e a password pass123 para aceder ao modo de administrador. Este passo é importante para carregar o estado e utilizar a aplicação.

2.2.1. Uso como administrador

Menu do Administrador:

```
*** SpotifUM ***
1 - Manage Albums
2 - Manage Users
3 - Manage Playlists
4 - Manage Data
5 - View System Stats / Execute Queries
0 - Exit
```

Figura 2 - Menu de administração do SpotifUM

Opções:

- Manage Albums: adicionar novo álbum (título, artista, ano, género, músicas), eliminar e listar.
- Manage Users: remover e listar utilizadores.
- Manage Playlists: criar playlist (genérica ou associada a um utilizador, descrição, estado), eliminar e listar.
- Manage Data: importar/exportar JSON, serializar/desserializar estado, limpar dados. Para utilizar o programa deve aceder a este menu e carregar o estado (instruções disponíveis no próprio programa).
- View System Stats: executar consultas/queries (música mais tocada, artista mais ouvido, top listener, género mais tocado, contagem de playlists públicas, etc.).

2.2.2. Uso Como Utilizador

Menu do Utilizador:

```
Login successful! Redirecting to User Menu...

*** SpotifUM ***
1 - Listen to Music
2 - Manage Playlists
3 - Update Subscription
4 - Edit Profile
0 - Exit
Option:
```

Figura 3 - Menu do Utilizador do SpotifUM

No menu inicial, registar uma nova conta com a opção 2 e fazer login com a opção 1. De seguida encontram-se disponíveis as opções:

- Ouvir Música, que permite de seguida:

```
*** SpotifUM ***
1 - Play from playlist
2 - Play from album
3 - Play free playlist
0 - Exit
Option: |
```

Figura 4 - submenu de ouvir música

- Reproduzir via Playlist, reproduzir via álbum ou, a única opção que os utilizadores Free podem seleccionar, reproduzir de playlist grátis. As opções 1 e 2 permitem também listar todas as playlists que o utilizador pode ouvir (públicas ou privadas mas cujo criador é o próprio), e listar todos os álbuns, respetivamente.

```
Now playing "Speak to Me" by Pink Floyd
This song has a music video!
You can watch it at https://www.youtube.com/watch?v=qkEn2Puh5xg
Lyrics:
Speak to me has no lyrics

Controls:
```

Figura 5 - Exemplo de Reprodução de faixa multimédia, no modo ouvir de um álbum (*The Dark Side of the Moon*)

```
Now playing "Bored" by Deftones
Attention! This is an explicit song!
Lyrics:
I get bored...

Controls:
n=next, p=prev, r=shuffle random song, s=stop
```

Figura 6 - Exemplo de reprodução de uma faixa explícita

- Gerir as Playlists, que permite ao utilizador:

```
*** SpotifUM ***
1 - Create User Playlist
2 - Add song to playlist
3 - Create Recommended Playlist
4 - View My Playlists
5 - Delete Playlist
0 - Exit
Option: |
```

Figura 7 - Submenu de gestão de playlists

- Criar uma nova Playlist
 - Adicionar uma música a uma playlist existente
 - Criar uma playlist com o recomendador do SpotifUM (permite filtrar tempo máximo ou exclusivamente músicas explícitas)
 - Ver as playlists que o utilizador criou
 - Apagar uma playlist que este criou
- Atualizar a subscrição, que permite ao utilizador alterar o seu plano:

```
*** SpotifUM ***
1 - FreePlan
2 - PremiumBase
3 - ---
0 - Exit
```

Figura 8 - submenu de atualização de plano

- E, por fim, editar o perfil, que permite ao utilizador alterar a sua password, email e morada.

```
*** SpotifUM ***
1 - Change Password
2 - Change Email
3 - Change Address
0 - Exit
```

Figura 9 - submenu de edição de perfil

3. Conclusão

O SpotifUM oferece uma plataforma modular de streaming em linha de comandos, com:

- Arquitectura limpa e extensível (interfaces, padrão Strategy, clonagem defensiva)
- Funcionalidades completas para administradores e utilizadores finais
- Vários planos de subscrição com permissões diferenciadas
- Estatísticas de uso detalhadas via classe Stats
- Persistência de estado em JSON e em formato serializado Java
- Cobertura de testes unitários para garantir robustez e regressão zero

Apesar disso, foram encontradas durante a realização do projeto algumas dificuldades, nomeadamente no desenvolvimento da interface baseada na linha de comandos, como o erro de por vezes um caractere “Enter” chegar ao input do próximo menu, e algumas incertezas do grupo quanto à definição de algumas funcionalidades, bem como algumas dúvidas em relação à arquitetura de classes e encapsulamento.

O diagrama de classes encontra-se disponível no ficheiro “DiagramaDeClassesG37.pdf”.