

課題管理プログラム

チーム名:フル単

35714012 井田 礼慈
35714026 大橋 蒼一郎
35714128 松岡 遼

1. 自由課題の概要

本課題では、チームで協力して「課題管理プログラム」を開発しました。このプログラムは、以下の 5 項目のデータを管理します。

- 課題 ID
- 課題名
- 出題した授業名
- 締め切り
- 課題が完了したかどうか

2. 各ファイルの概要

ファイル名	概要
Task.java	課題データを表すデータモデル
DBUtil.java	データベース接続用のユーティリティクラス
TaskDAO.java	タスクに関するデータ操作の契約を定義
TaskDAOImpl.java	“TaskDAO.java”インターフェースの実装クラス。JDBC を用いて SQLite データベースとやり取りをする。
Main.java	アプリケーションのエントリーポイント。主な動作内容は項目参照。
Command.java	以下“AddTask.java”～“Exit.java”のコマンドパターンのみを実装したインターフェース
AddTask.java	Command インターフェースを実装する具体的なコマンドクラス。「タスクの追加」を表現。
DeleteTask.java	Command インターフェースを実装する具体的なコマンドクラス。「タスクの削除」を表現。
CompleteTask.java	Command インターフェースを実装する具体的なコマンドクラス。「タスク完了」を表現。
ShowAll.java	Command インターフェースを実装する具体的なコマンドク

	ラス.「全タスクの表示」を表現.
SortTasks.java	Command インターフェースを実装する具体的なコマンドクラス.「タスクのソート」を表現.
Exit.java	Command インターフェースを実装する具体的なコマンドクラス.「終了」を表現.
SortStrategy.java	以下”TitleSortStrategy.java”～”TaskSorter.java”ソート戦略のインターフェース. List<Task>を受け取り、ソートされたリストを返す.
TitleSortStrategy.java	課題名(タイトル)でソート. ひらがな, カタカナ→漢字の順にソート.
DeadlineSortStrategy.java	締切日でソート
CompletedSortStrategy.java	完了状態でソート
SubjectSortStrategy.java	授業名(科目)でソート. ひらがな, カタカナ→漢字の順にソート.
TaskSorter.java	ソート戦略を動的に切り替えてソートを実行. Strategy パターンの「コンテキスト」に相当.

3. 各ファイルの詳細説明

➤ Task.java

構成内容

- フィールド
 - ✧ id:タスクの識別子
 - ✧ title:タスクのタイトル
 - ✧ subject:タスクに関連する授業名(科目名)
 - ✧ deadline:タスクの締切日
 - ✧ completed:タスクが完了しているかどうか(boolean)

- その他各種ゲッター・セッター

➤ DBUtil.java

構成内容

- メソッド
 - ✧ setUrl: 接続先 URL を指定するためのメソッド. デフォルトは "jdbc:sqlite:tasks.db"で、テスト時に切り替える.
 - ✧ getConnection:SQLite データベースへの接続を取得するための静的メソッド

ド. SQLException をスローする. JDBC の DriverManager を使用.

➤ TaskDAO.java

構成内容

- メソッド
 - ✧ findAll:全タスクを取得するメソッド. 戻り値は Task のリスト.
 - ✧ findById:引数で指定した ID のタスクを取得するメソッド. 戻り値は Task.
 - ✧ addTask:新たなタスクを追加するメソッド. 引数に Task オブジェクトを取る.
 - ✧ markTaskCompleted:指定された ID のタスク(completed)を完了状態に更新するメソッド
 - ✧ deleteTask:指定された ID のタスクを削除するメソッド.
 - ✧ close: 接続をクローズするメソッド.

➤ TaskDAOImpl.java

構成内容

- コンストラクタ:データベースに接続し, Task テーブルが存在しない場合は新たに作成する.
 - ✧ findAll:tasks テーブルからすべてのレコードを取得し, Task オブジェクトのリストとして返す
 - ✧ findById: 指定した ID のタスクを取得する. 内部で executeQuery を呼び出し, findAll とクエリ実行と結果取得のロジックを共通化
 - ✧ addTask:新たなタスクを tasks テーブルに挿入.
 - ✧ markTaskCompleted:指定された ID のタスク(completed)を true に更新
 - ✧ deleteTask:指定された ID のタスクを tasks テーブルから削除
 - ✧ close: フィールドとして持っている Connection を閉じる.

➤ Main.java

処理内容

- “TaskDAO.java”の実装クラスを用いてタスクを追加
- “findAll”メソッドで全タスクを取得し, 各タスクの情報を標準出力に表示
- “markTaskCompleted”メソッドで指定された ID のタスクを完了状態に更新
- “deleteTask”で指定された ID のタスクを削除

4. コンパイル・実行方法

- ・ コンパイル: `javac -cp .:sqlite-jdbc-3.30.1.jar:junit-4.13.2.jar:hamcrest-core-1.3.jar [filename]`
- ・ テスト実行: `java -cp .:sqlite-jdbc-3.30.1.jar:junit-4.13.2.jar:hamcrest-core-1.3.jar org.junit.runner.JUnitCore [testname]`
- ・ Main 実行: `java -cp .:sqlite-jdbc-3.30.1.jar:junit-4.13.2.jar:hamcrest-core-1.3.jar Main`

5. 制限

- データベース
SQLite を用いて”tasks.db”に、課題データを永続的に保存・管理しています。
DBUtil.java でデータベース接続を行い、TaskDAOImpl.java で CRUD 操作を実装しました。
- Junit テストコード
 - ① TaskTest.java
Task.java のコンストラクタ、アクセッサの動作のテストを行う。
 - ② TaskDAOImplTest.java
SQLite のインメモリ DB を使用し、TaskDAOImpl.java の各メソッドの単体テストを行う。
 - ③ TaskSorterTest.java
各種ソートの切り替えのテストを行う。
 - ④ TitleSortStrategyTest.java
SubjectSortStrategyTest.java
DeadlineSortStrategyTest.java
CompletedSortStrategyTest.java
各種ソートの動作のテストを行う。
- GoF デザインパターン
使用パターン: Strategy パターン
課題のソート方法を柔軟に変更できるように、Strategy パターンを採用しました。
 - ・ SortStrategy インターフェースを定義以下の具体的な戦略クラスを実装
 - ✧ TitleSortStrategy
 - ✧ DeadlineSortStrategy
 - ✧ CompletedSortStrategy
 - ✧ SubjectSortStrategy

TaskSorter クラスで戦略を切り替え可能にし、ユーザーの選択に応じてソートを実行

- Java17

Sealed Classes(密封クラス)

- ・Command インターフェースを sealed として定義し、許可されたクラスのみが実装できるように制限。
- ・これにより、コマンドの種類を明確に制御し、安全性と可読性を向上。

Pattern Matching for switch(switch 文のパターンマッチ)

- ・TaskDAOImpl.java にて、Command の種類に応じた処理を switch 文で分岐。
- ・Java17 のパターンマッチング構文を活用し、より簡潔で安全なコードを実現。