Using Enums and Enums Inside Classes in Java

Project Abstract

The purpose of this project is to demonstrate the use of Enums in Java, focusing on both simple Enums and Enums defined inside a class. Enums are special types that represent a fixed set of constants. This project will show how to create and use enums in Java, how to define an enum inside a class, and how to use methods to work with these enums.

Tasks Overview

Task 1: Implementing Enums

Objective: Define an enum to represent a set of constants, such as days of the week.

Detailed Description: In this task, you will create an enum called Day that represents the days of the week (Monday to Sunday). You will then demonstrate how to access and use the enum in your application.

- Steps:

1. Create the Day Enum:
    - Define an enum Day with values representing the days of the week (MONDAY, TUESDAY, WEDNESDAY, etc.).
    - Enums should be used when a fixed set of constants is required.

Task 2: Implementing Enums Inside a Class

Objective: Define an enum inside a class to represent different statuses.

Detailed Description: In this task, you will define an enum WeekStatus inside the WeekScheduler class. The enum will represent two values: WEEKDAY and WEEKEND. You will create a method getDayStatus that checks if a given day is a weekend or weekday.

- Steps:

2. Create the WeekStatus Enum Inside the WeekScheduler Class:
    - Define the enum WeekStatus with values WEEKDAY and WEEKEND.
3. Use the Enum in a Method:

- Implement a method getDayStatus() in the WeekScheduler class to return WEEKDAY or WEEKEND based on the input day.

4. Check the Enum Usage:
    - In the main() method, call getDayStatus() to check whether a day is a weekday or weekend.

Task 3: Implement main()

Objective: Demonstrate how to use enums inside methods and access the enum's values to control the flow of the program.

- Create an object with the name today of type Day and assign MONDAY to it.
- Print message as "Today is: " + today.
- Create an object with a name scheduler of type WeekScheduler.
- And print a message as "Is today a weekday or weekend? " + scheduler.getDayStatus(today).

Execution Steps to Follow:

1. **All actions like build, compile, running application, running test cases will be through Command Terminal.**
2. **To open the command terminal the test takers, need to go to Application menu (Three horizontal lines at left top) → Terminal →New Terminal.**
3. **This editor Auto Saves the code.**
4. **If you want to exit(logout) and continue the coding later anytime (using Save & Exit option on Assessment Landing Page) then you need to use CTRL+Shift+B-command compulsorily on code IDE. This will push or save the updated contents in the internal git/repository. Else the code will not be available in the next login.**
5. **These are time bound assessments the timer would stop if you logout and while logging in back using the same credentials the timer would resume from the same time it was stopped from the previous logout.**
6. **To run your project use command:**

    mvn compile exec:java

    -Dexec.mainClass="com.yaksha.assignment.EnumsAssignment"

7. **To test your project test cases, use the command**

   mvn test

8. **You need to use CTRL+Shift+B - command compulsorily on code IDE, before final submission as well. This will push or save the updated contents in the internal git/repository, and will be used to evaluate the code quality.**