Importing Packages and Accessing Classes and Methods in Java

Project Abstract

The purpose of this project is to demonstrate the concept of Importing Packages and Accessing Classes and Methods in Java. In Java, packages are used to group related classes and interfaces. This project will show how to create and use packages, how to define classes within packages, and how to import and access those classes and methods from other parts of the application. By completing this project, you will understand how to structure Java applications into packages, organize code efficiently, and use imports to access code across different packages.

Tasks Overview

Task 1: Create a Package and Define Classes

Objective: Demonstrate how to define a package and place classes within it.

Detailed Description: In this task, you will create a package called com.yaksha.utility. Inside this package, you will define two classes: MathOperations and StringOperations. The MathOperations class will have methods like add() and multiply(), and the StringOperations class will have methods like concatenate() and getLength().

- Steps:

1. Create the Package:
   - Define a package called com.yaksha.utility where the utility classes will be stored.
2. Define the MathOperations Class:
   - Inside com.yaksha.utility, define a class MathOperations.
   - Add methods like add(int, int) and multiply(int, int) to perform arithmetic operations.
3. Define the StringOperations Class:
   - Inside com.yaksha.utility, define a class StringOperations.
   - Add methods like concatenate(String, String) to join two strings and getLength(String) to return the length of a string.

Task 2: Import Packages and Access Methods

Objective: Import a package and access the methods from the classes defined inside it.

Detailed Description: In this task, you will import the com.yaksha.utility package into the Main class and call methods from the MathOperations and StringOperations classes. You will create objects of these classes and invoke their methods.

- Steps:

4. Import the Package:
    - In the Main class, use the import keyword to import the com.yaksha.utility package and access the classes MathOperations and StringOperations.
5. Access Methods from Imported Classes:
    - Create instances of the MathOperations with name as mathOperations and StringOperations with name as stringOperations classes.
    - Call methods like add(), multiply(), concatenate(), and getLength() using the created instances.
6. Print Results:
    - Print the results of the methods to verify their correctness as:

      "Addition of 5 and 3: " + mathOperations.add(5, 3)

      "Multiplication of 5 and 3: " + mathOperations.multiply(5, 3)

      "Concatenation of 'Hello' and 'World': " + stringOperations.concatenate("Hello", "World")

      "Length of 'Hello': " + stringOperations.getLength("Hello")

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.Main"
   ```
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.**