Using Multiple Classes, Class Attributes, Accessing and Modifying Attributes in Java

Project Abstract

The purpose of this project is to demonstrate how multiple classes interact with each other, how to define and access class attributes, and how to modify these attributes in Java. This project focuses on:

1. Use multiple classes in a Java program.
2. Define attributes in classes.
3. Access and modify class attributes.
4. Use methods to interact with attributes of objects.

Tasks Overview

Task 1: Using Multiple Classes

Objective: Create two classes: Person and Address. The Person class will have attributes like name, age and address. Whereas the Address class will hold information about the person's address like street and city. We will use both classes together in the main program.

Detailed Description: In this task, you are required to define two classes: Person and Address.

The Address class will contain properties like street and city.
The Person class will have properties like name, age, and a reference to the Address class to store the address of the person.

- Steps:

1. Define a class named Address with properties street and city, and a constructor to initialize them.
2. Define a class named Person with properties name, age, and a reference to an Address object, and include a constructor to initialize these properties.
3. In the Person class, define a method displayDetails() to print the person's name, age, and address using the displayAddress() method from the Address class.

Note:

- ❖ In the `displayAddress()` method inside Address class, you need to print in the below format:

```
Street: <street>
City: <city>
```

- ❖ In the `displayDetails()` method of `Person`, you need to print in the below format:

```
Name: <name>
Age: <age>
invoke displayAddress() method from the Address class
```

Task 2: Class Attributes

Objective: Define and initialize the attributes of the Person and Address classes.

Detailed Description: In this task, you will define attributes within the Person and Address classes and initialize them using the constructor.

- Steps:

4. Define properties for the Address class: street (String) and city (String).
5. Define properties for the Person class: name (String), age (int), and a reference to the Address class.
6. Initialize these properties using constructors in both classes.

Task 3: Accessing Attributes

Objective: Access and display the attributes of the Person and Address classes.

Detailed Description: In this task, you will create an object for both Person and Address classes, and access their attributes to display the details in main().

- Steps:

7. Instantiate an object address1 of the Address class with specific values for street and city.
8. Instantiate an object person1 of the Person class and pass the address1 object to the constructor.
9. Call the displayDetails() method of person1 to display the person's name, age, and address.

Expected Output Format:

Name: John

Age: 25

Street: Main St

City: New York

Task 4: Modifying Attributes

Objective: Modify the attributes of the Person and Address objects.

Detailed Description: In this task, you will modify the attributes of the Person and Address objects.

- Steps:

10. Create a method with name modifyDetails() inside Person class which should accept name and age in its arguments and assign these to class variables. Use the modifyDetails() method of the Person class to change the name and age attributes in main() ..
11. Create a method with name modifyAddress() inside Person class which should accept street and city in its arguments and assign these to Address class variables using address reference. Use the modifyAddress() method of the Person class to change the street and city of the Address object.
12. After modifying the attributes, call the displayDetails() method again to display the updated details.

Expected Output Format After Modification:

Name: John Doe

Age: 30

Street: Broadway St

City: Los Angeles

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.AttributesManagementAssignment"
   ```

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.