**Gold Rate Calculation - Collections**

Description

**Objective:**

To work with a Spring Core application using annotations, autowiring, setter-based injection concepts, and configuring fields from property files.

**Concept Explanation:**

1. **Setter-based Injection:** Dependencies are injected into beans using setter methods, allowing for flexibility and easy reconfiguration of dependencies.
2. **Configuring Fields from Property Files:** Values are injected into bean fields from external property files, enhancing the flexibility and maintainability of the application configuration.

**Concept Implementation:**

1. Setter-based injection is employed to inject values into the **rateInfo** map of the **GoldRateInfo** class from the **goldRateDetails.properties** file.
2. This is achieved by providing a setter method for the **rateInfo** map and using the **@Value** annotation to inject values from the properties file.
3. These values are injected into the **rateInfo** map of the **GoldRateInfo** class using setter-based injection and the **@Value** annotation, enabling flexible configuration and easy maintenance of gold rate details.

**Gold Rate Calculation - Collections**

A jewellery shop needs an application to calculate the gold rate for the specific grams chosen by their customers. Create a spring core application using Maven to perform this task.

**GoldRateInfo**class with the below **private attribute**is provided as part of the code skeleton.

|  |  |
| --- | --- |
| rateInfo | Map<Integer,Double> |

**Getter and setter** methods for the above attribute are provided as part of the code skeleton. The **GoldRateInfo**class should be registered as a **bean** with appropriate annotations.

The values need to be injected into the **rateInfo** via **setter-based injection** in the **GoldRateInfo** class from the **goldRateDetails.properties**file**.**

A method**public double calculateGoldRate(int goldCarat, double grams)**will be provided in the **GoldRateInfo**class as a part of the code skeleton. Depending on the user input, fetch the appropriate gold rate from the **rateInfo** map, calculate the gold rate, and return the result back to the user.

The map values are given in the **goldRateDetails.properties** file for the gold rate per gram for various carats are as follows:

|  |  |
| --- | --- |
| **Carat (Integer)** | **Rate per gram (Double)** |
| 18 | 3800.0 |
| 22 | 4300.0 |
| 24 | 4500.0 |

**Driver**class with the below methods is provided as part of the code skeleton.

* **public static GoldRateInfo loadGoldRateDetails()**--> This method should fetch the **GoldRateInfo**object and return the same.
* **public static void main(String[] args)**-->  Inside the main method, invoke the **loadGoldRateDetails**method and obtain the **GoldRateInfo**object.

**Design Constraints:**

* **GoldRateInfo and Driver** class should be present in **com.spring.app** package
* Maintain the same **className/Attribute Name/PackageName** as specified in the problem statement. Do not create any new packages.
* All the configuration should be done using the annotations in the created Maven project.

**Sample Input:**  
Enter the carat:  
18  
Enter Total Grams:  
13  
  
**Sample Output:**Total Gold Rate is Rs:49400.0