Name : Prathik Balaji N
Date : 08-08-24

Java and Microservices

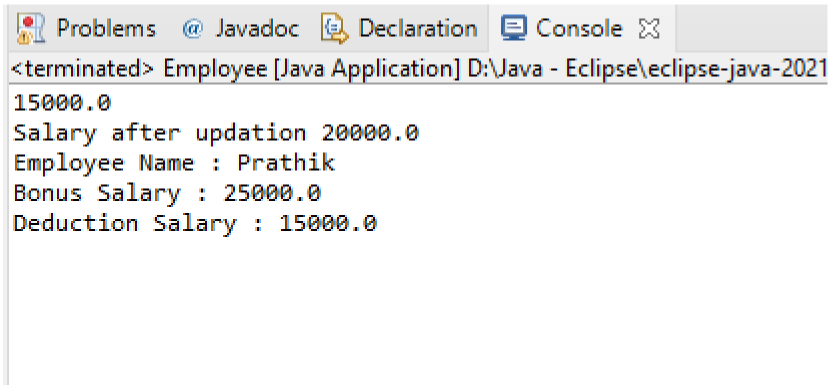
1. Create Class named Employee program with class variables as companyName, instance variables with employeeName, employeeID , employeeSalary.
2. Use Data Encapsulation and use getters and setters for updating the employeeSalary
3. Show function overloading to calculate salary of employee with bonus and salary of employee with deduction.

Code :

```
package Samp;

public class Employee {

	static String CompanyName = "Jah's Company" ;
	private String EmployeeName ;
	private int EmployeeID;
	private double EmployeeSalary;

	Employee(String name , int id , double salary){
		this.EmployeeName = name;
		this.EmployeeID = id;
		this.EmployeeSalary = salary;
	}

	public double getEmployeeSalary(){
```

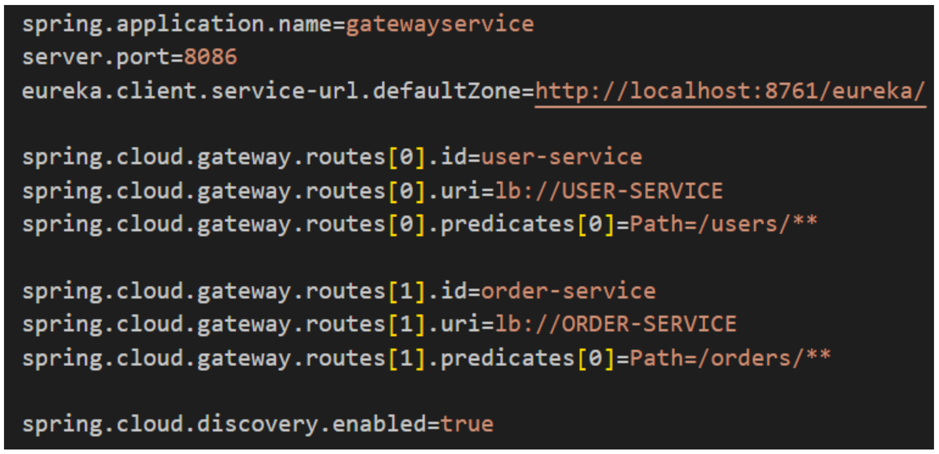
```
                return EmployeeSalary;

        }

        public void setEmployeeSalary(double salary) {
                this.EmployeeSalary = salary ;
        }

        private double calSalary(double DeductSal, boolean d) {
                return EmployeeSalary - DeductSal ;
        }

        private double calSalary(double BonusSal) {
                return EmployeeSalary + BonusSal ;
        }

        public static void main(String[] args) {

                Employee emp = new Employee("Prathik", 11290, 15000.00);
                System.out.println(emp.getEmployeeSalary());
                emp.setEmployeeSalary(20000.00);
                System.out.println("Salary after updation " +
emp.getEmployeeSalary());
                System.out.println("Employee Name : " + emp.EmployeeName);
                System.out.println("Bonus Salary : "+emp.calSalary(5000.00));
                System.out.print("Deduction Salary : "+emp.calSalary(5000.00,false));
        }

}
```

Output :

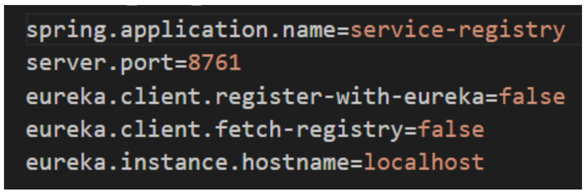

Problems @ Javadoc Declaration Console

```
<terminated> Employee [Java Application] D:\Java - Eclipse\eclipse-java-2021
15000.0
Salary after updation 20000.0
Employee Name : Prathik
Bonus Salary : 25000.0
Deduction Salary : 15000.0
```

4. What are the Microservices – that use this Gateway and Service Discovery methods using below screen shot:



```
spring.application.name=gatewayservice
server.port=8086
eureka.client.service-url.defaultZone=http://localhost:8761/eureka/

spring.cloud.gateway.routes[0].id=user-service
spring.cloud.gateway.routes[0].uri=lb://USER-SERVICE
spring.cloud.gateway.routes[0].predicates[0]=Path=/users/**

spring.cloud.gateway.routes[1].id=order-service
spring.cloud.gateway.routes[1].uri=lb://ORDER-SERVICE
spring.cloud.gateway.routes[1].predicates[0]=Path=/orders/**

spring.cloud.discovery.enabled=true
```



```
spring.application.name=service-registry
server.port=8761
eureka.client.register-with-eureka=false
eureka.client.fetch-registry=false
eureka.instance.hostname=localhost
```

Gateway Service :

Port: 8086

Eureka Client Configuration: It uses Eureka for service discovery as it's pointing to a Eureka server at http://localhost:8761/eureka/.

Routes:

user-service: Maps /users/** to the USER-SERVICE.

order-service: Maps /orders/** to the ORDER-SERVICE.

Service Registry :

Port: 8761

Eureka Server Configuration: Acts as the Eureka Server for service registration and discovery.