**Lab Exercise 11- Drool Attribute (Controlling Rule Execution with salience)**

**Objective:**

To learn how the salience attribute affects the order in which Drools rules are executed, and how to prioritize rules based on business logic.

**Exercise Steps:**

**1. Setup the Drools Project**

* Create a new Drools project.
* Set up your project with the necessary Drools dependencies if you're using a Java IDE.

**2. Create the Fact Class**

* Create a simple Java class named Order with the following properties:

public class Order {

private String customerType;

private double totalAmount;

public Order(String customerType, double totalAmount) {

this.customerType = customerType;

this.totalAmount = totalAmount;

}

public String getCustomerType() {

return customerType;

}

public void setCustomerType(String customerType) {

this.customerType = customerType;

}

public double getTotalAmount() {

return totalAmount;

}

public void setTotalAmount(double totalAmount) {

this.totalAmount = totalAmount;

}

@Override

public String toString() {

return "Order [customerType=" + customerType + ", totalAmount=" + totalAmount + "]";

}

}

**3. Write the DRL (Drools Rule Language) File**

* Create a new .drl file in your project, e.g., salienceExample.drl.
* Define three rules with different salience values to apply discounts based on the customer type. The higher the salience value, the higher the priority of the rule.

package com.example.rules;

import com.example.Order;

rule "VIP Customer Discount"

salience 100

when

$order : Order(customerType == "VIP", totalAmount > 0)

then

System.out.println("Applying 20% discount for VIP customer");

$order.setTotalAmount($order.getTotalAmount() \* 0.8);

end

rule "Regular Customer Discount"

salience 50

when

$order : Order(customerType == "Regular", totalAmount > 0)

then

System.out.println("Applying 10% discount for Regular customer");

$order.setTotalAmount($order.getTotalAmount() \* 0.9);

end

rule "New Customer Discount"

salience 10

when

$order : Order(customerType == "New", totalAmount > 0)

then

System.out.println("Applying 5% discount for New customer");

$order.setTotalAmount($order.getTotalAmount() \* 0.95);

end

**4. Create the Main Class**

* Write a Java class to create an instance of the Order class and fire the rules.
* Example DroolsTest class:

package com.example;

import org.kie.api.runtime.KieContainer;

import org.kie.api.runtime.KieSession;

import org.kie.api.KieServices;

public class DroolsTest {

public static void main(String[] args) {

try {

KieServices ks = KieServices.Factory.get();

KieContainer kContainer = ks.getKieClasspathContainer();

KieSession kSession = kContainer.newKieSession("ksession-rules");

// Create an Order instance

Order order1 = new Order("VIP", 1000);

Order order2 = new Order("Regular", 1000);

Order order3 = new Order("New", 1000);

// Insert facts into the session

kSession.insert(order1);

kSession.insert(order2);

kSession.insert(order3);

// Fire all rules

kSession.fireAllRules();

// Print the final order amounts

System.out.println("Final Order Amounts:");

System.out.println(order1);

System.out.println(order2);

System.out.println(order3);

kSession.dispose();

} catch (Exception e) {

e.printStackTrace();

}

}

}

**5. Run the Application**

* Run the DroolsTest class.
* Observe the output in the console. You should see the discount being applied in the order of VIP, Regular, and then New customers, based on the salience values.

**6. Experiment with Different salience Values**

* Modify the salience values in the rules and re-run the application to see how the execution order changes.
* Try setting the same salience for multiple rules and observe the behavior.

**7. Reflection**

* Consider scenarios where controlling the order of rule execution is crucial, such as applying business rules with dependencies or handling conflicting conditions.
* Reflect on how the salience attribute can help prioritize important rules and prevent unintended outcomes.

**Expected Output:**

After running the application, the console should show:

Applying 20% discount for VIP customer

Applying 10% discount for Regular customer

Applying 5% discount for New customer

Final Order Amounts:

Order [customerType=VIP, totalAmount=800.0]

Order [customerType=Regular, totalAmount=900.0]

Order [customerType=New, totalAmount=950.0]