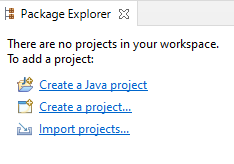
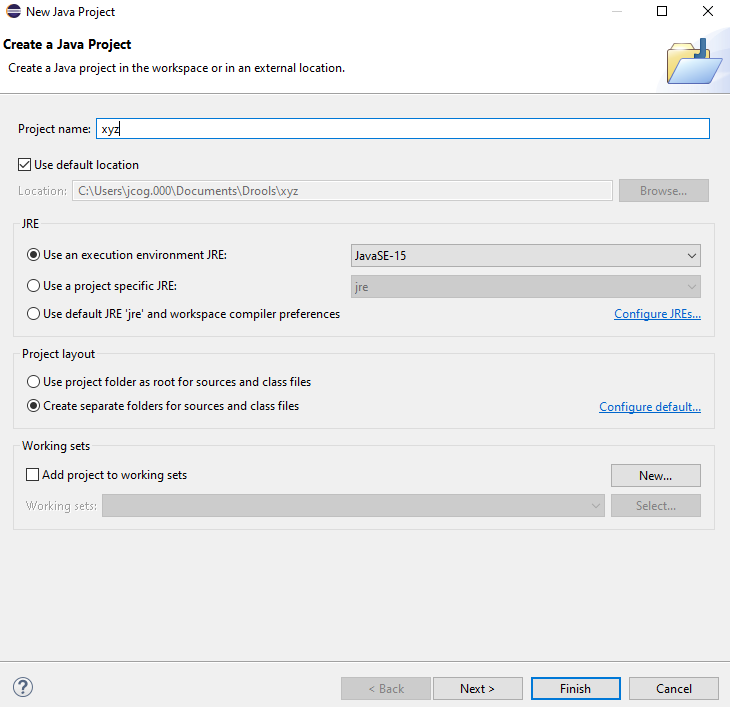
**Java and Java Spring application examples:**

Choose **Create a Java project**

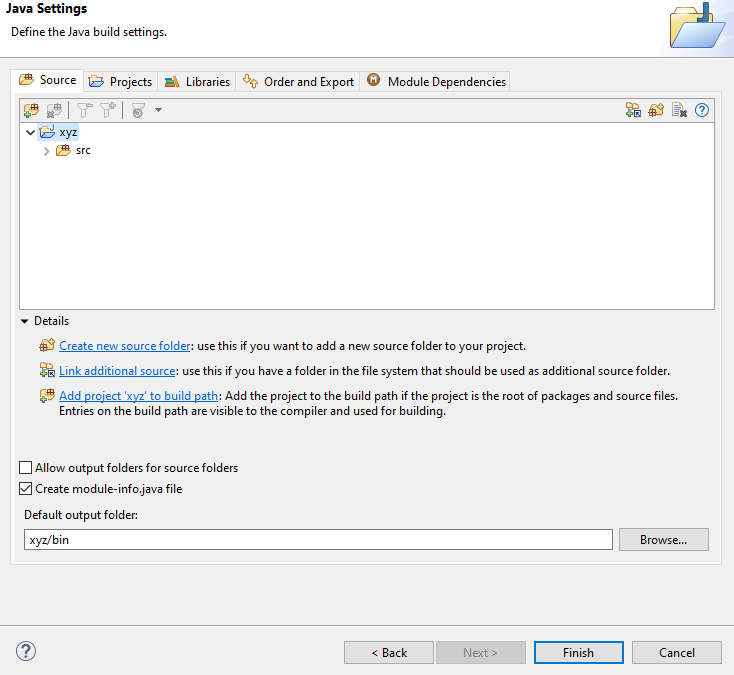


Name **xyz**

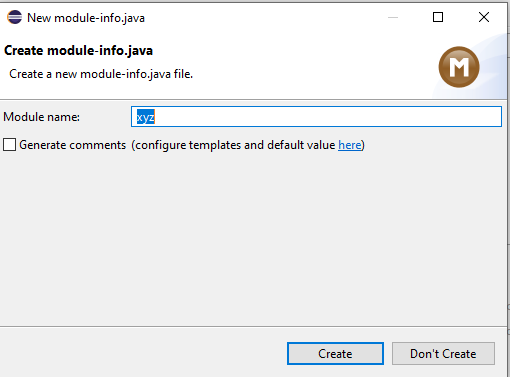
**Accept Defaults and Next**



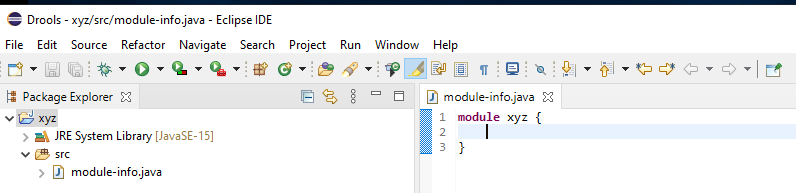
**Accept the Defaults and Finish**



**Accept Defaults and Create:**



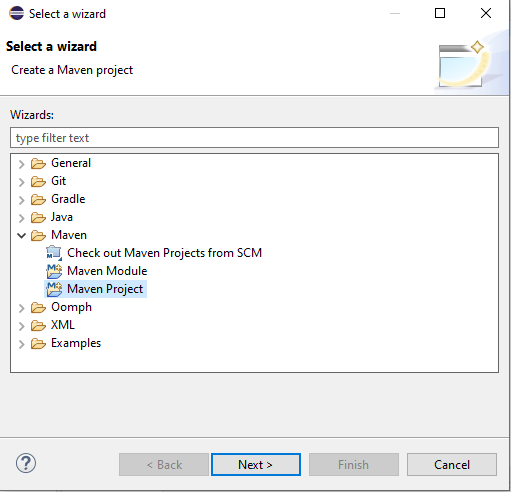
**Which results in this:**



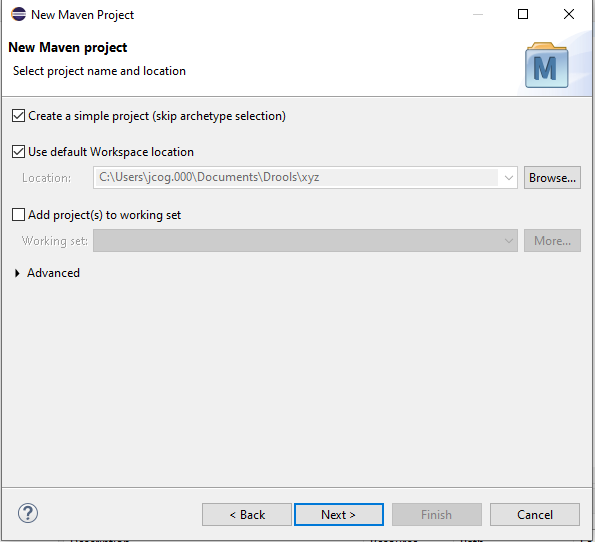
**Install Postman:**

Create your first Drools Project:

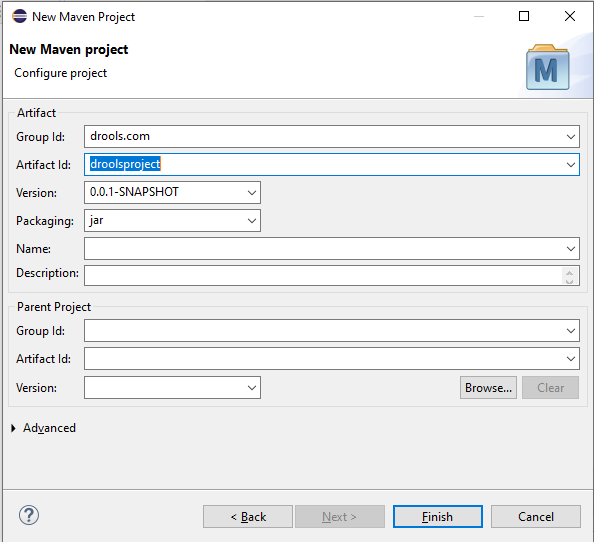
New >> Other >> Maven Project



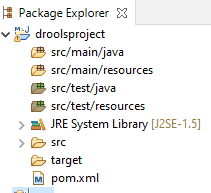
Next, check Create a simple project(skip archetype selection)



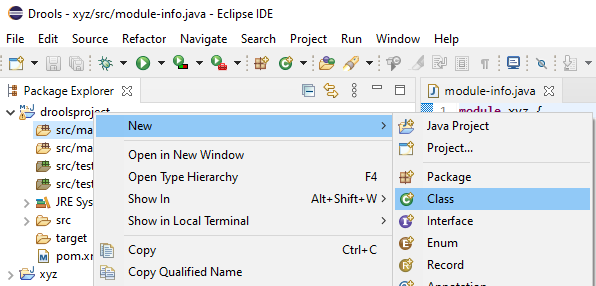
Next, Add Group Id: drools.com and Artifact Id: droolsproject Finish



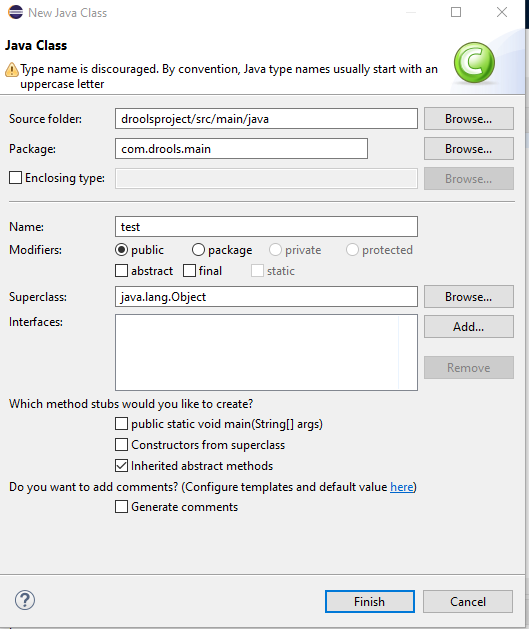
Drools Project is created

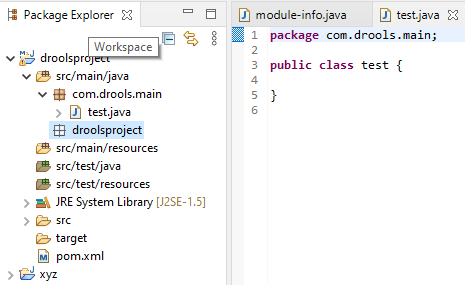


Create a Java Class called test.java in src/main/java



Name the class “test” package ‘com.drools.main’ Finish

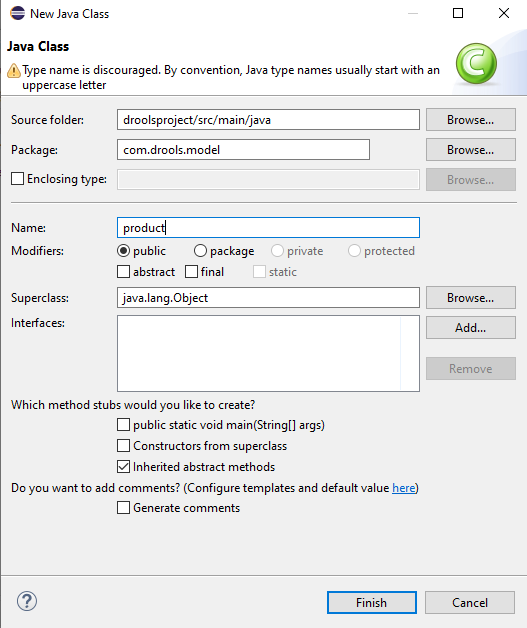


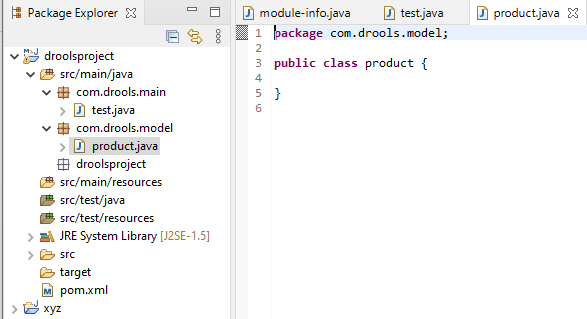


Now, create a model package with a project class:

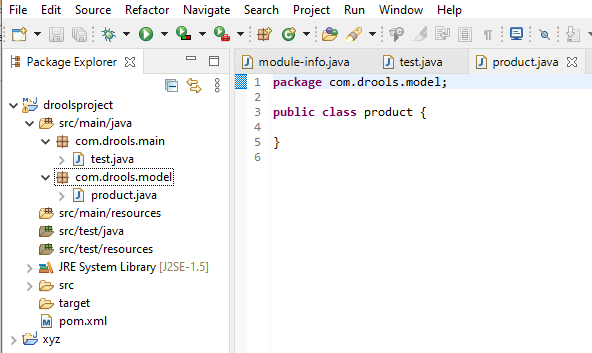
Click on the droolsproject New >> class

Name the class ’product’ Name the package ‘com.drools.model’ Finish





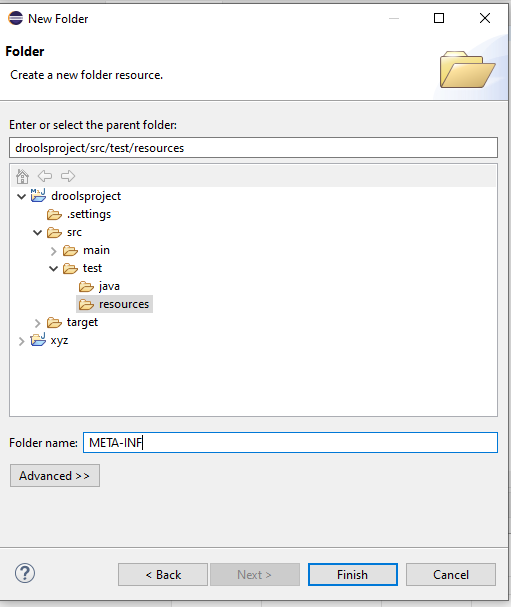
Select the empty package droolsproject and delete it!!

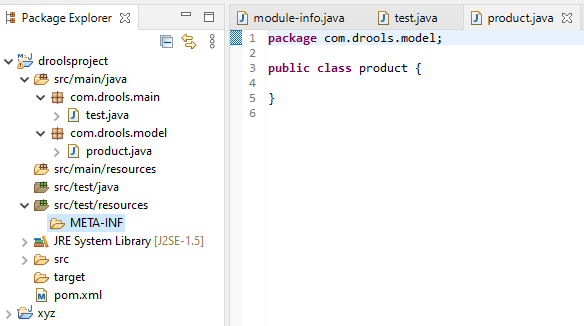


Create a **META-INF** Folder in the resources directory

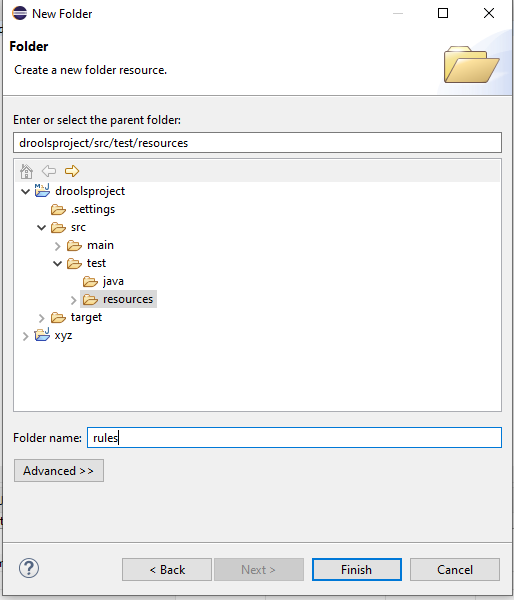
New >> Other >> General >> Folder

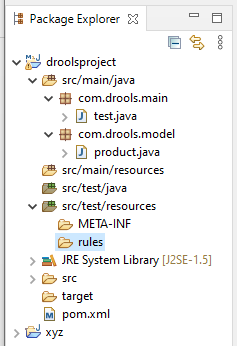
Next >> name META-INF Finish



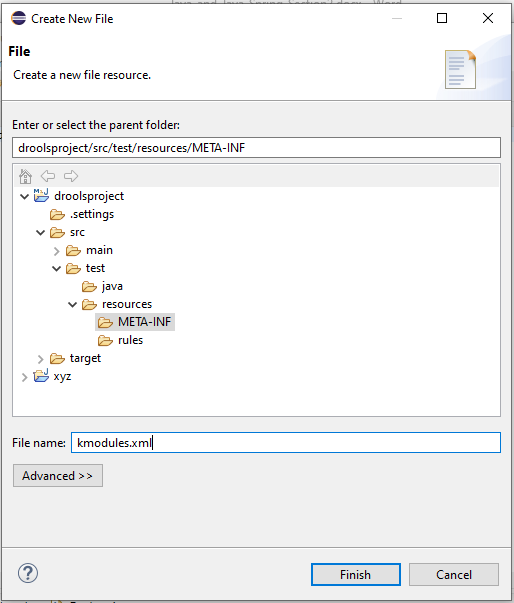


Create a rules folder in the resources:

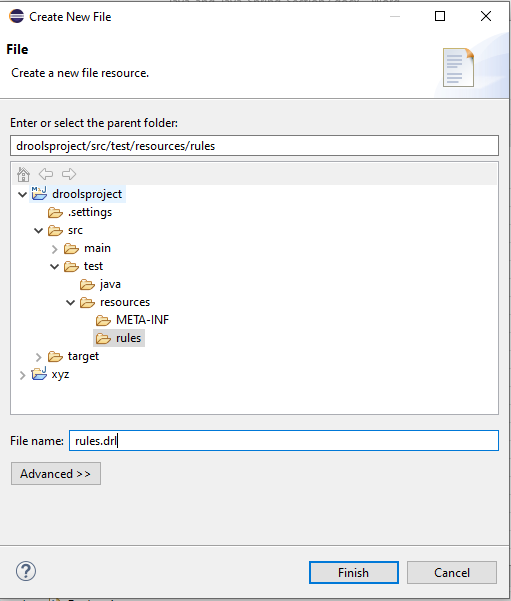


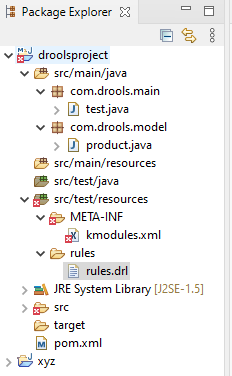


Create a kmodules.xml file in the META-INF folder



Create a rules.drl file in the rules folder



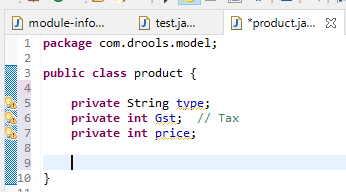


Populate the Maven POM file:

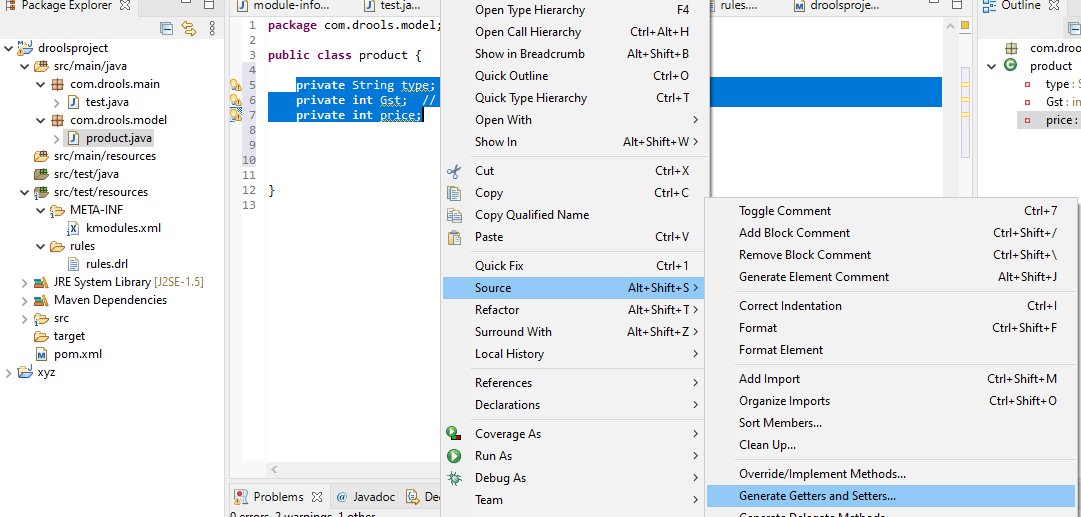
****

Now, define the products in the Grocery Store:

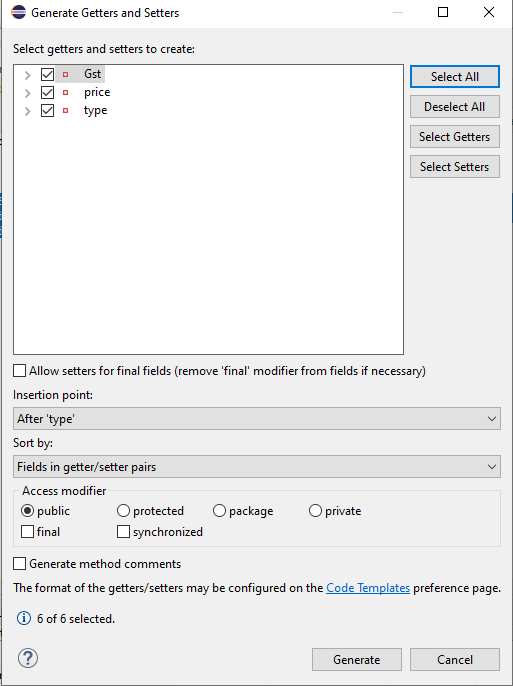
Create 3 Fields in your Model Class product:



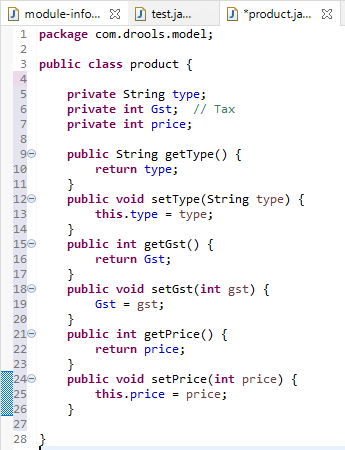
Generate Getters and Setters for the fields:



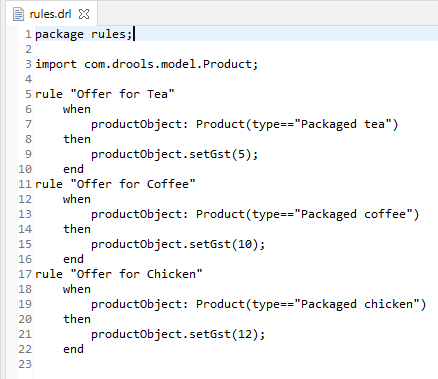
Select all and Generate



Now, we have our model:



Now, populate the Rules:



Now, create a Driver or test to run the program

**package** com.drools.main;

**import** java.util.Scanner;

**import** org.drools.compiler.compiler.DroolsParserException;

**import** org.drools.core.base.RuleNameEndsWithAgendaFilter;

**import** org.kie.api.KieServices;

**import** org.kie.api.runtime.KieContainer;

**import** org.kie.api.runtime.KieSession;

**import** com.drools.model.product;

**public** **class** test {

**public** **static** **void** main(String args[]) {

System.***err***.close();

System.*setErr*(System.***out***);

**try** {

KieServices ks = KieServices.Factory.*get*();

KieContainer kContainer = ks.getKieClasspathContainer();

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

product product = **new** product();

product.setType("Package coffee");

kSession.insert(product);

kSession.fireAllRules();

System.***out***.println("Enter our price for "+product.getType());

Scanner input = **new** Scanner(System.***in***);

**int** price = input.nextInt();

product.setPrice(price);

System.***out***.println("The Gst tax for the "+product.getType()+" is "+product.getGst()+"%");

System.***out***.println("You have to pay "+" : "+((product.getPrice()+(product.getPrice()\*product.getGst())/100)+" $"));

}

**catch**(Exception e) {

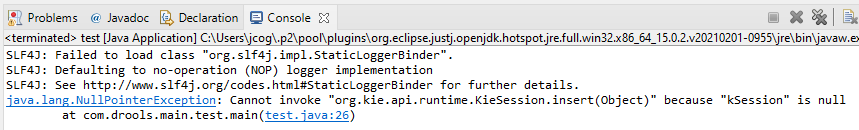
e.printStackTrace();

}

}

}

Run as an Application:



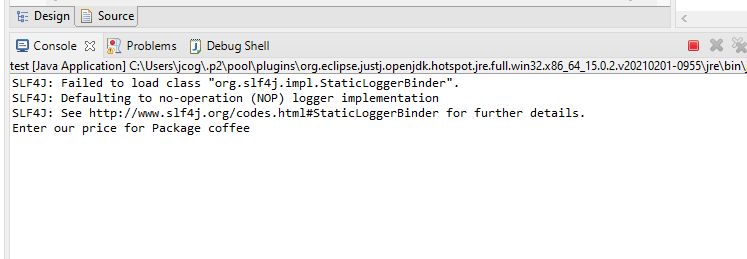
I had to refactor or rename the Kmodules to Kmodule.xml.

Rerun application:

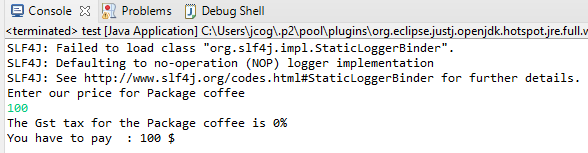
Failed

Moved the Folders for META\_INF and rules from the src/test/resources to src/main/resources

**Now, it works!!!**



Run application:



Code Failed to add the Tax:

Miss typed the Package type Fixed

Run application again:

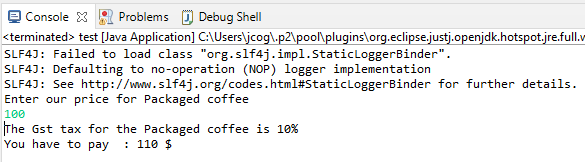
Add semicolons to the end of the packages in rules – no change still fails

Found Error: product.setType("Package coffee"); Add a d to Package coffee

product.setType("Packaged coffee");

Run Application Again:

It outputs correctly!!!!!!!!!!



**Let us alter the Program to enter each type of Offer!!**

Rewrote Program Code:

**try** {

KieServices ks = KieServices.Factory.*get*();

KieContainer kContainer = ks.getKieClasspathContainer();

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

product product = **new** product();

System.***out***.print("Enter the product type: ");

//enter coffee, tea or chicken

Scanner inputType = **new** Scanner(System.***in***);

product.setType("Packaged "+inputType.next());

kSession.insert(product);

kSession.fireAllRules();

System.***out***.print("Enter our price for "+product.getType()+":");

Scanner inputPrice = **new** Scanner(System.***in***);

**int** price = inputPrice.nextInt();

product.setPrice(price);

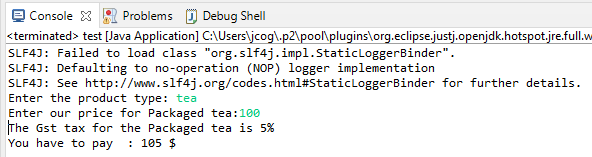
System.***out***.println("The Gst tax for the "+product.getType()+" is "+product.getGst()+"%");

System.***out***.print("You have to pay "+" : "+((product.getPrice()+(product.getPrice()\*product.getGst())/100)+" $"));

}**catch**(Exception e) {

e.printStackTrace();

}



**package** com.drools.main;

**Cleaned up code to accept 2 tokens, refactored java Class names(types) as first letter capital and removed 2 warnings**

**import** java.util.Scanner;

**import** org.drools.compiler.compiler.DroolsParserException;

import org.drools.core.base.RuleNameEndsWithAgendaFilter;

**import** org.kie.api.KieServices;

**import** org.kie.api.runtime.KieContainer;

**import** org.kie.api.runtime.KieSession;

**import** com.drools.model.Product;

**public** **class** Test {

**public** **static** **void** main(String args[]) {

System.***err***.close();

System.*setErr*(System.***out***);

**try** {

KieServices ks = KieServices.Factory.*get*();

KieContainer kContainer = ks.getKieClasspathContainer();

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

Product product = **new** Product();

System.***out***.print("Enter the product type: ");

//Packaged coffee, Packaged tea or Packaged chicken

// Press Enter is carriage return \\r\\n

Scanner inputType = **new** Scanner(System.***in***).useDelimiter("\\r\\n");

product.setType(inputType.next());

kSession.insert(product);

kSession.fireAllRules();

System.***out***.print("Enter our price for "+product.getType()+": ");

Scanner inputPrice = **new** Scanner(System.***in***);

**int** price = inputPrice.nextInt();

product.setPrice(price);

System.***out***.println("The Gst tax for the "+product.getType()+" is "+product.getGst()+"%");

System.***out***.print("You have to pay: "+((product.getPrice()+(product.getPrice()\*product.getGst())/100)+" $"));

inputType.close();

inputPrice.close();

}

**catch**(Exception e) {

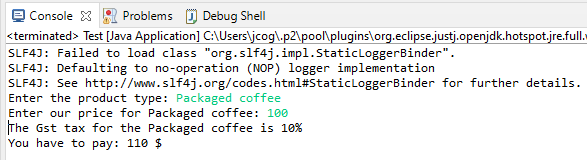
e.printStackTrace();

}

}

}

Output Improved:



Type is now Product not product in rules.drl

**You could put the code in a loop and test all 3 in one session:**

**public** **class** Test {

**public** **static** **void** main(String args[]) {

System.***err***.close();

System.*setErr*(System.***out***);

**try** {

KieServices ks = KieServices.Factory.*get*();

KieContainer kContainer = ks.getKieClasspathContainer();

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

// Initial outside the loop

Product product = **new** Product();

product.setType("Start");

Scanner inputType = **new** Scanner(System.***in***);

Scanner inputPrice = **new** Scanner(System.***in***);

// Exit instructions

System.***out***.println("Type \"Quit\" to exit the program");

**while**(!product.getType().equals("Quit")) {

System.***out***.print("Enter the product type: ");

//Packaged coffee, Packaged tea or Packaged chicken

//Press Enter is carriage return \\r\\n accept multiple //tokens

inputType = **new** Scanner(System.***in***).useDelimiter("\\r\\n");

product.setType(inputType.next());

// Quit typed as product type exit

**if** (product.getType().equals("Quit")) {

**break**;

}

kSession.insert(product);

kSession.fireAllRules();

System.***out***.print("Enter our price for "+product.getType()+": ");

**int** price = inputPrice.nextInt();

product.setPrice(price);

System.***out***.println("The Gst tax for the "+product.getType()+" is "+product.getGst()+"%");

System.***out***.println("You have to pay: "+((product.getPrice()+(product.getPrice()\*product.getGst())/100)+" $"));

}

System.***out***.println("Program has exited!!");

inputType.close();

inputPrice.close();

}

**catch**(Exception e) {

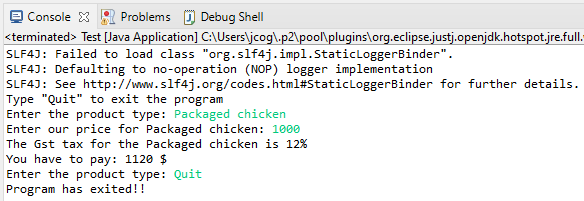
e.printStackTrace();

}

}

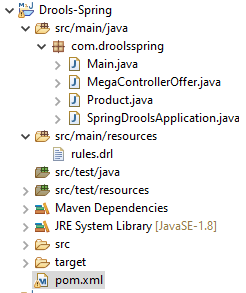
}

**Output:**



**Java Spring Example:**

**The Code:**



rules.drl

****

****

****

**MegaControllerOffer.java**

**Pr******

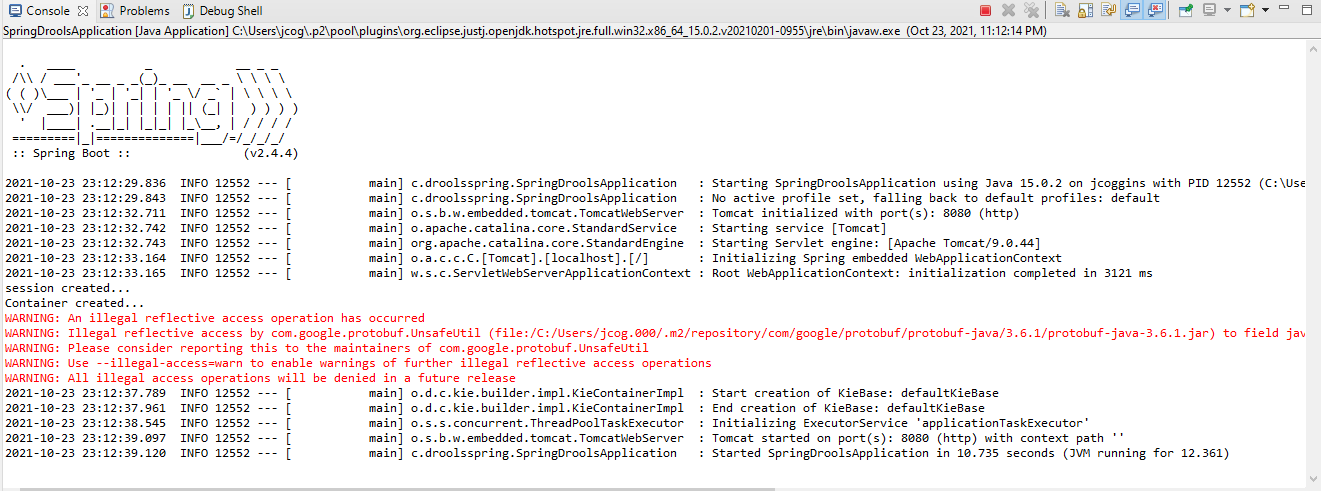
**Product.java**

****

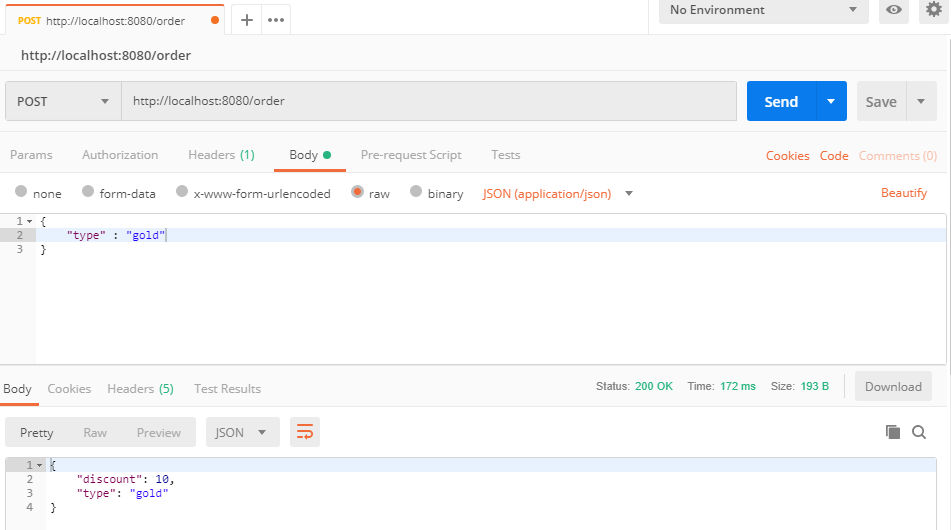
**SpringDroolsApplication.java**

****

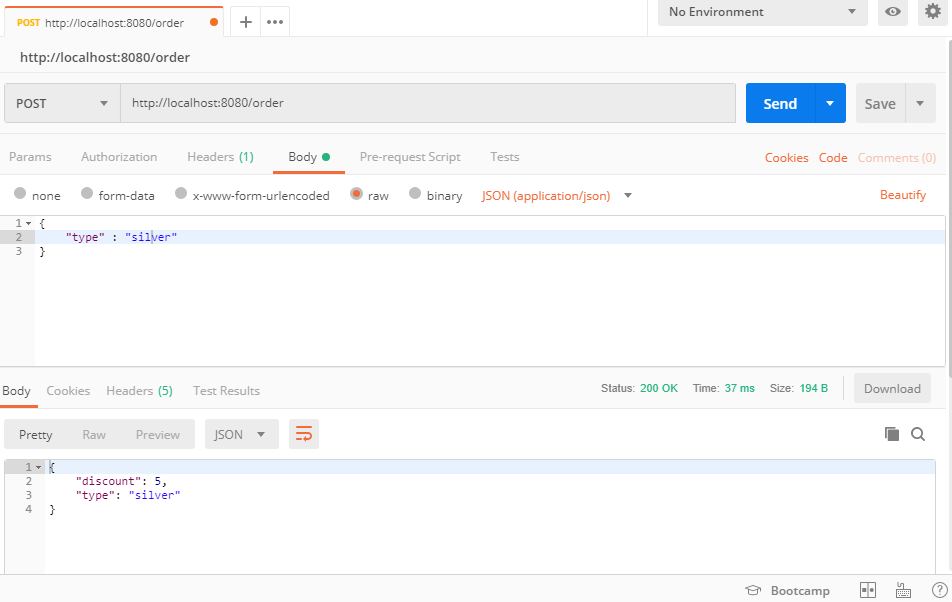
**Output successful:**



Try posting type is gold >> send >> 200 Ok >> discount is 10 for gold

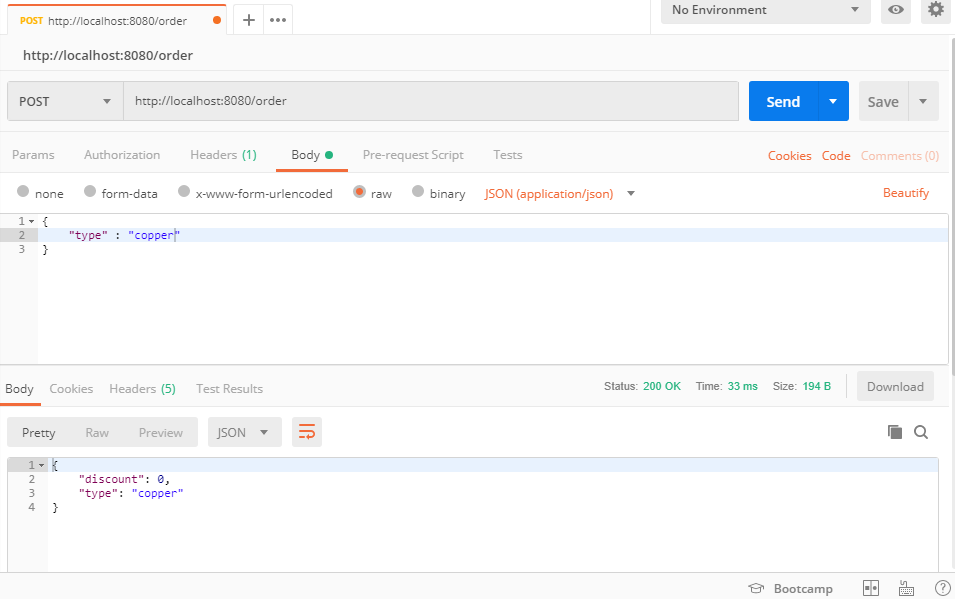


Try posting type is silver >> send >> 200 Ok >> discount is 5 for silver



If you try Copper, well the response is 200 Ok,

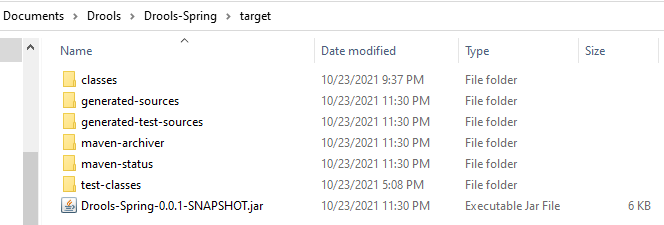
There is no Copper in the rules.drl file, so the discount is 0.

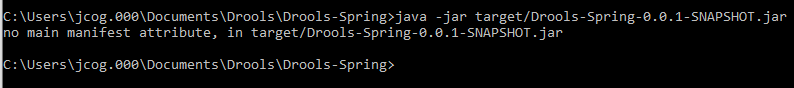


**Run the Spring app from the command line:**

**mvn install** will create a runnable jar file!!







Going have remake the jar with a manifest:

