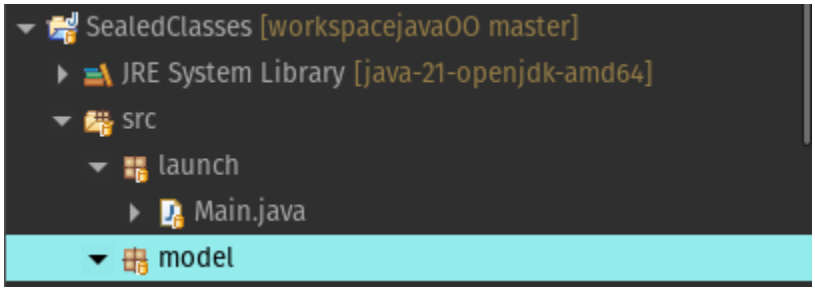
Sealed Classes

Objectif

Savoir créer des classes sealed
Savoir les utiliser
Comprendre l'intérêt de ce mécanisme

Action

Créer un projet avec un package model et launch
Dans launch créer une classe Main avec une méthode main comme suit:



Créer une classe scellée PaymentMethod comme suit:

```java
package model;

public sealed class PaymentMethod permits CreditCard, DebitCard, Paypal{
    String welcomeMessage() {
        return "general payment method";
    }

}
```

Puis creer les classes necessaires suivantes

```java
package model;

public final class CreditCard extends PaymentMethod {
    @Override
    String welcomeMessage() {
        return "CreditCard payment method";
    }

    void creditMethod() {
        System.out.println("creditMethod");
    }
}
```

```java
package model;

public non-sealed class DebitCard extends PaymentMethod
    @Override
    String welcomeMessage() {
        return "DebitCard payment method";
    }

    void debitMethod() {
        System.out.println("debitMethod");
    }
}
```

```java
package model;

public final class Paypal extends PaymentMethod {
    @Override
    String welcomeMessage() {
        return "Paypal payment method";
    }

    void paypalMethod() {
        System.out.println("paypalMethod");
    }
}
```

Créer la classe PaymentMethod qui grace a un switch case appelle les méthodes générales et spécifiques de chacune

```java
package model;

public class PaymentProcessor {
    public void processPayment(PaymentMethod paymentMethod, double amount) {
        switch (paymentMethod) {
        case CreditCard cc -> {
            // Process credit card payment
            cc.welcomeMessage();
            cc.creditMethod();
        }
        case DebitCard dc -> {
            // Process debit card payment
            dc.welcomeMessage();
            dc.debitMethod();
        }
        case Paypal pp -> {
            // Process PayPal payment
            pp.welcomeMessage();
            pp.paypalMethod();
        }
        default -> System.out.println("Unrecognized payment method");

        }
    }
}
```

Instancier tout cela dans le main pour execution

```
 9 public class Main {
10
11     public static void main(String[] args) {
12
13         var cc = new CreditCard();
14         var dc = new DebitCard();
15         var pp = new Paypal();
16
17
18         var processor = new PaymentProcessor();
19
20         processor.processPayment(cc, 0);
21         processor.processPayment(dc, 0);
22         processor.processPayment(pp, 0);
23
24
25
26     }
```

Bravo!

En plus

Créer une classe DifferedDebitCard qui hérite de DebitCard et l'ajouter au main

Que constate-t-on?
Peut-on l'ajouter au switch