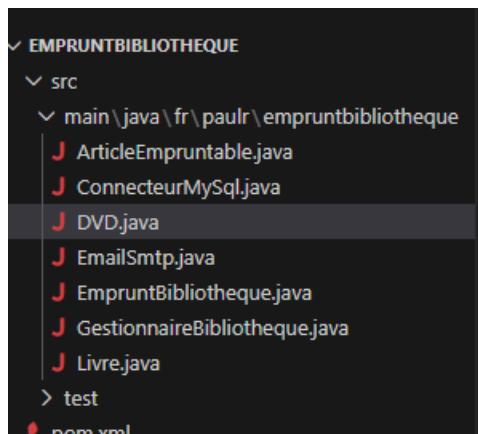


1. Importation du projet	1
2. Les Problème et les solutions	1
S – Single Responsibility Principle - SRP (Responsabilité unique)	2
O – Open/Closed Principle - OCP (Ouvert/Fermé)	3
L – Liskov Substitution Principle - LSP (Substitution de Liskov)	4
I – Interface Segregation Principle – ISP - (Séparation des interfaces)	4
D – Dependency Inversion Principle -DIP (Inversion des dépendances)	5

1. Importation du projet

Le projet a bien été Importé.



2. Les Problème et les solutions

S – Single Responsibility Principle - SRP (Responsabilité unique)

S.1

```
J DVD.java M X
src > main > java > fr > paulr > empruntbibliotheque > J DVD.java
11  public class DVD implements ArticleEmprunable{
12      ...
13
14      @Override
15      public void calculerPenaliteDeRetard(int jours) {
16          System.out.println("Forfait : 50 euros quel que soit le nombre de jours de retard.");
17      }
18
19      @Override
20      public void imprimeEtiquette() {
21          System.out.println("Etiquette DVD : " + titre);
22      }
23
24      @Override
25      public void connexionBDD() {
26          ConnecteurMySql db = new ConnecteurMySql();
27          db.connect("Serveur BDD", 3307, "admin", "admin");
28      }
29
30      @Override
31      public void envoyerNotifEmail(String message) {
32          System.out.println("Pas de notification par email pour les DVD.");
33      }
34
35 }
```

Problème :

Ces classes ne conviennent pas à la gestion des livres.

Solution :

De nouveaux fichiers avec des classes et des interfaces dédiés doivent être créer.

S.2

```
J Livre.java X
src > main > java > fr > paulr > empruntbibliotheque > J Livre.java
11  public class Livre implements ArticleEmprunable{
12      // LSP violé : comportement incohérent
13      @Override
14      public void calculerPenaliteDeRetard(int jours) {
15          if (jours < 0) {
16              throw new IllegalArgumentException("Le nombre de jours ne peut pas être négatif !");
17          }
18          System.out.println("Pénalités de retard : " + (jours * 10) + " euros");
19      }
20
21      @Override
22      public void imprimeEtiquette() {
23          System.out.println("Etiquette : " + titre);
24      }
25
26      // DIP violé : dépendance directe à une classe concrète
27      @Override
28      public void connexionBDD() {
29          ConnecteurMySql db = new ConnecteurMySql();
30          db.connect("localhost", 3306, "root", "root");
31      }
32
33      @Override
34      public void envoyerNotifEmail(String message) {
35          EmailSmtp sender = new EmailSmtp();
36          sender.send("user@example.com", "Message de votre bibliothèque", message);
37      }
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57 }
```

Problème :

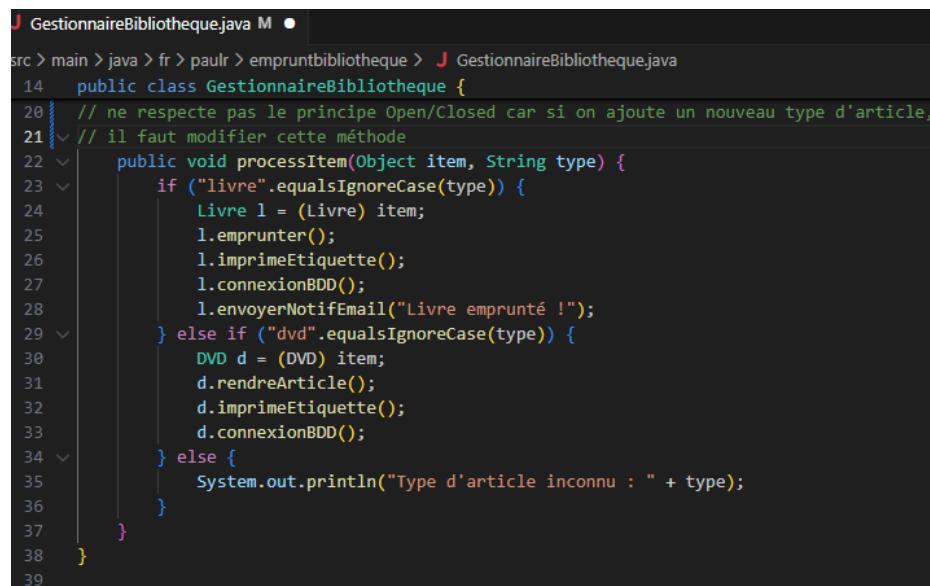
Ces classes ne conviennent pas à la gestion des DVD en plus de contreviennent à d'autres principes SOLID.

Solution :

De nouveaux fichiers avec des classes et des interfaces dédiés doivent être créés.

O – Open/Closed Principle - OCP (Ouvert/Fermé)

O.1



```
src > main > java > fr > paulr > empruntblibrotheque > GestionnaireBibliotheque.java
14  public class GestionnaireBibliotheque {
15      // ne respecte pas le principe Open/Closed car si on ajoute un nouveau type d'article,
16      // il faut modifier cette méthode
17      public void processItem(Object item, String type) {
18          if ("livre".equalsIgnoreCase(type)) {
19              Livre l = (Livre) item;
20              l.emprunter();
21              l.imprimeEtiquette();
22              l.connexionBDD();
23              l.envoyerNotifEmail("Livre emprunté !");
24          } else if ("dvd".equalsIgnoreCase(type)) {
25              DVD d = (DVD) item;
26              d.rendreArticle();
27              d.imprimeEtiquette();
28              d.connexionBDD();
29          } else {
30              System.out.println("Type d'article inconnu : " + type);
31          }
32      }
33  }
```

Problème :

Afin de rajouter un nouveau type d'article il faudrait modifier ce code

Solution :

Création d'un nouveau fichier avec une interface objEmprunté.

L – Liskov Substitution Principle - LSP (Substitution de Liskov)

L.1

```
J Livre.java X
src > main > java > fr > paulr > empruntbibliotheque > J Livre.java
11  public class Livre implements ArticleEmprurable{
12      // LSP violé : comportement incohérent
13      @Override
14      public void calculerPenaliteDeRetard(int jours) {
15          if (jours < 0) {
16              throw new IllegalArgumentException("Le nombre de jours ne peut pas être négatif !");
17          }
18          System.out.println("Pénalités de retard : " + (jours * 10) + " euros");
19      }
20
21      @Override
22      public void imprimeEtiquette() {
23          System.out.println("Etiquette : " + titre);
24      }
25 }
```

Problème :

La classe fille modifie le comportement de la classe mère en lançant une exception pour un cas que la classe mère acceptait.

Solution :

Rectifier le comportement en retirant l'exception ou en définir la règle dans l'abstraction pour que toutes les classes dérivées respectent les règles.

I – Interface Segregation Principle – ISP - (Séparation des interfaces)

I.1

```
J ArticleEmprunable.java M X
src > main > java > fr > paulr > empruntbibliotheque > J ArticleEmprunable.java
 7  /**
11 | // ne respecte pas le ISP car cette interface n'est pas spécialisée.
12 | public interface ArticleEmprunable {
13 |
14 |     void emprunter();
15 |
16 |     void rendreArticle();
17 |
18 |     void calculerPenaliteDeRetard(int jours);
19 |
20 |     void imprimeEtiquette();
21 |
22 |     void connexionBDD();
23 |
24 |     void envoyerNotifEmail(String message);
25 }
```

Problème :

L'interface n'est pas spécifique, elle est bien trop générale.

Solution :

Il faut diviser l'interface en plusieurs petites interfaces spécifique.

```
J ArticleEmprunable.java M X
src > main > java > fr > paulr > empruntbibliotheque > J ArticleEmprunable.java
27 // Respecte le ISP en divisant l'interface en plusieurs interfaces spécifiques.
28 public interface EmprunterArticle {
29     void emprunter();
30 }
31 public interface RendreArticle {
32     void rendreArticle();
33 }
34 public interface CalculerPenalite {
35     void calculerPenaliteDeRetard(int jours);
36 }
37 public interface ImprimerEtiquette {
38     void imprimeEtiquette();
39 }
40 public interface ConnexionBDD {
41     void connexionBDD();
42 }
43 public interface EnvoyerNotifEmail {
44     void envoyerNotifEmail(String message);
45 }
```

D – Dependency Inversion Principle -DIP (Inversion des dépendances)

D.1

```

J DVD.java X
src > main > java > fr > paulr > empruntbibliotheque > J DVD.java

50 // Classes concrètes (DIP violé)
51 class MySqlDatabase {
52     public void connect(String host, int port, String user, String pass) {
53         System.out.println("Connexion MySQL à " + host + ":" + port);
54     }
55 }
56
57 class SmtpEmailSender {
58     public void send(String to, String subject, String body) {
59         System.out.println("Email envoyé à " + to + ": " + subject);
60     }
61 }

```

Problème :

Ces classes non pas leurs propres fichiers, elles sont écrites sur un fichier de bas niveau qui peut disparaître (si on propose plus de DVD).

Solution :

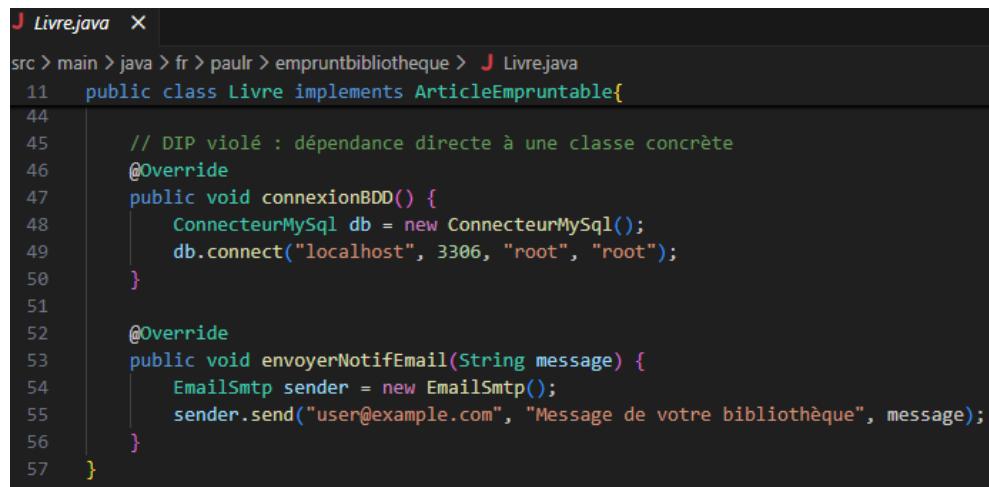
Il faut que ces classes est leur propre fichier.

```

J MySqlDatabase U X J SmtpEmailSender U
src > main > java > fr > paulr > empruntbibliotheque > J MySqlDatabase
1 public class MySqlDatabase {
2     public void connect(String host, int port, String user, String pass) {
3         System.out.println("Connexion MySQL à " + host + ":" + port);
4     }
5 }

J MySqlDatabase U J SmtpEmailSender U X
src > main > java > fr > paulr > empruntbibliotheque > J SmtpEmailSender
1 public class SmtpEmailSender {
2     public void send(String to, String subject, String body) {
3         System.out.println("Email envoyé à " + to + ": " + subject);
4     }
5 }

```



```
J Livre.java X
src > main > java > fr > paulr > empruntbibliotheque > J Livre.java
11  public class Livre implements ArticleEmprunable{
44
45      // DIP violé : dépendance directe à une classe concrète
46      @Override
47      public void connexionBDD() {
48          ConnecteurMySql db = new ConnecteurMySql();
49          db.connect("localhost", 3306, "root", "root");
50      }
51
52      @Override
53      public void envoyerNotifEmail(String message) {
54          EmailSmtp sender = new EmailSmtp();
55          sender.send("user@example.com", "Message de votre bibliothèque", message);
56      }
57 }
```

Problème :

Solution :