3_Codage

- Ne pas mettre de condition négative :
  Refusé :

```
if (!value.notOk())
```

  Accepté :

```
if (value.ok())
```

- Utiliser des constantes pour les nombres et les chaine de caractères :

```
public static final STRING VERSION="1.0"
```

- Lorsque l'on teste une chaine constante avec une variable, toujours utiliser la constante en premier (cela évite de tester si la variable est null)
  Ne jamais tester un objet avec l'opérateur '==', toujours utiliser la méthode 'equals'

```
if ("CHAINE".equals(maVar)
if (CHAINE.equals(maVar))
```

- Vérifier systématiquement si un objet est null
  Attention, l'écriture suivante fonctionne, mais aura un comportement différent en fonction de la JVM (Java Virtual Machine) Ne JAMAIS l'écrire

```
Object object = null;
if (object==0)
```

  Utiliser :

```
Object object = null;
if (object==null)
```

- Il faut toujours fermer les curseurs et les connexions à minima dans un finally pour ne pas laisser des curseurs/connexions ouverts en cas d'erreur:

```
DBUtil dbSys;
ResultSet rs = null;

try {
   dbSys = new DBUtil( ... );
   rs = dbSys.exec( ... );
   DBUtil.closeRS( rs );
   rs = null;
}
catch (Exception e) {
   xxx
}
finally {
   DBUtil.closeRS( rs );
   DBUtil.close( dbSys );
}
```

  Il n'est pas nécessaire de tester si rs ou dbSys soient null avant d'appeler les méthodes de DBUtil.closeRs et DBUtil.close (elles le vérifient).

  Attention dans le catch(Exception e) ne jamais mettre de return sinon on ne passe pas dans le finally.

  Si vous souhaitez retourner une valeur dans ce block catch alors il faut passer par une variable :

```
DBUtil dbSys;
ResultSet rs = null;

int retValue = 0; // initialiser la valeur
```

```
try {
    dbSys = new DBUtil( ... );
    rs = dbSys.exec( ... );

    retValue = ...; // mettre une autre valeur en fonction du métier
    DBUtil.closeRS( rs );
    rs = null;
}
catch (Exception e) {
    // faire du log d'erreur

 retValue = -1; // mettre la valeur à retourner en cas d'erreur
}
finally {
    DBUtil.closeRS( rs );
    DBUtil.close( dbSys );
}

return retValue; // un seul return par fonction pour renvoyer la valeur
```

- Il faut éviter absolument car cela masque les exceptions et on ne peut pas savoir ce qu'il se passe

catch (Exception e) {}

Mettre au moins un :

```
catch (Exception e) {
            e.printStackTrace();
        }
```