Grille d'auto-évaluation sur des bonnes pratiques de programmation Java (Initial)

mailto:equipe-gl@telecom-bretagne.eu

v0.3

© 0 0 BY SA

Mon code est écrit en anglais (noms de variables, méthodes, fonctions, classes, packages, etc.) [P1, Po1, Pj1] Mes commentaires sont écrits dans une seule langue (préférentiellement l'anglais.) [P1, P12] Les commentaires (javadoc) des méthodes publiques en permettent la réutilisation. [P1] Il y a dans le code des commentaires à destination des mainteneurs (hors javadoc). [P1,P5] Style et conventions Les indentations sont homogènes. [NB : IDE eclipse peut gérer cela] [P18] La forme des noms respecte le style du langage (Par exemple, en Java : CamelCase pour les variables, CAPITALES pour les constantes) [P18] L'ordre des déclarations est toujours le même (Par exemple : attributs, constructeurs, méthodes publiques puis privées.) [P3] Les noms sont porteurs de sens. [P13] Modularité Le paquetage racine ne contient directement aucume classe. [Pj11] Le fichier package.info décrit le contenu et la responsabilité de chaque paquetage. [Pj2] Chaque classe a une responsabilité de calcul et ne se limite pas à stocker des données. [Po2, Po6] Il n'y a pas d'attribut public (encapsulation.) [Po3] Les méthodes ne sont pas trop longues (ex. 15 lignes de code.) [P6, P7] Sûreté Les méthodes se protègent (test des paramètres, levée d'exception.) [P20, Po13] Chaque méthode possède au moins un test. [P18] Les tests sont automatisés et s'enchainent pour vérifier la non-régression. [P18] Évolutivité/Maintenance Les code est factorisé. [P9] Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14] Le code est factorisé. [P9]		Commentaires
Mes commentaires sont écrits dans une seule langue (préférentiellement l'anglais.) [P1, P12]		Mon code est écrit en anglais (noms de variables, méthodes, fonctions, classes, pa-
P1, P12 Les commentaires (javadoc) des méthodes publiques en permettent la réutilisation. P1 Il y a dans le code des commentaires à destination des mainteneurs (hors javadoc). P1,P5 Style et conventions		ckages, etc.) [P1, Po1, Pj1]
□ Les commentaires (javadoc) des méthodes publiques en permettent la réutilisation. [P1] □ Il y a dans le code des commentaires à destination des mainteneurs (hors javadoc). [P1,P5] Style et conventions □ Les indentations sont homogènes. [NB : IDE eclipse peut gérer cela] [P18] □ La forme des noms respecte le style du langage (Par exemple, en Java : CamelCase pour les variables, CAPITALES pour les constantes) [P18] □ L'ordre des déclarations est toujours le même (Par exemple : attributs, constructeurs, méthodes publiques puis privées.) [P3] □ Les noms sont porteurs de sens. [P13] Modularité □ Le paquetage racine ne contient directement aucune classe. [Pj11] □ Le fichier package.info décrit le contenu et la responsabilité de chaque paquetage. [Pj2] □ Chaque classe a une responsabilité de calcul et ne se limite pas à stocker des données. [P02, P06] □ Il n'y a pas d'attribut public (encapsulation.) [P03] □ Les méthodes ne sont pas trop longues (ex. 15 lignes de code.) [P6, P7] Sûreté □ Les méthode possède au moins un test. [P18] □ Chaque méthode possède au moins un test. [P18] □ Les tests sont automatisés et s'enchainent pour vérifier la non-régression. [P18] □ Évolutivité/Maintenance □ Les classes importées sont nommées explicitement. [Pj8] □ Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14]		Mes commentaires sont écrits dans une seule langue (préférentiellement l'anglais.)
[P1] □ Il y a dans le code des commentaires à destination des mainteneurs (hors javadoc). □ Es indentations sont homogènes. [NB : IDE eclipse peut gérer cela] [P18] □ La forme des noms respecte le style du langage (Par exemple, en Java : CamelCase pour les variables, CAPITALES pour les constantes) [P18] □ L'ordre des déclarations est toujours le même (Par exemple : attributs, constructeurs, méthodes publiques puis privées.) [P3] □ Les noms sont porteurs de sens. [P13] ■ Modularité □ Le paquetage racine ne contient directement aucune classe. [Pj11] □ Le fichier package.info décrit le contenu et la responsabilité de chaque paquetage. [Pj2] □ Chaque classe a une responsabilité de calcul et ne se limite pas à stocker des données. [Po2, Po6] □ Il n'y a pas d'attribut public (encapsulation.) [Po3] □ Les méthodes ne sont pas trop longues (ex. 15 lignes de code.) [P6, P7] ■ Sûreté □ Les méthode possède au moins un test. [P18] □ Les tests sont automatisés et s'enchainent pour vérifier la non-régression. [P18] ■ Évolutivité/Maintenance □ Les classes importées sont nommées explicitement. [Pj8] □ Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14]		[P1, P12]
Style et conventions Les indentations sont homogènes. [NB : IDE eclipse peut gérer cela] [P18] La forme des noms respecte le style du langage (Par exemple, en Java : CamelCase pour les variables, CAPITALES pour les constantes) [P18] L'ordre des déclarations est toujours le même (Par exemple : attributs, constructeurs, méthodes publiques puis privées.) [P3] Les noms sont porteurs de sens. [P13] Modularité Le paquetage racine ne contient directement aucune classe. [Pj11] Le fichier package.info décrit le contenu et la responsabilité de chaque paquetage. [Pj2] Chaque classe a une responsabilité de calcul et ne se limite pas à stocker des données. [Po2, Po6] Il n'y a pas d'attribut public (encapsulation.) [Po3] Les méthodes ne sont pas trop longues (ex. 15 lignes de code.) [P6, P7] Sûreté Les méthode possède au moins un test. [P18] Les tests sont automatisés et s'enchainent pour vérifier la non-régression. [P18] Évolutivité/Maintenance Les classes importées sont nommées explicitement. [Pj8] Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14]		Les commentaires (javadoc) des méthodes publiques en permettent la réutilisation.
Style et conventions Les indentations sont homogènes. [NB : IDE eclipse peut gérer cela] [P18] La forme des noms respecte le style du langage (Par exemple, en Java : CamelCase pour les variables, CAPITALES pour les constantes) [P18] L'ordre des déclarations est toujours le même (Par exemple : attributs, constructeurs, méthodes publiques puis privées.) [P3] Les noms sont porteurs de sens. [P13] Modularité Le paquetage racine ne contient directement aucune classe. [Pj11] Le fichier package.info décrit le contenu et la responsabilité de chaque paquetage. [Pj2] Chaque classe a une responsabilité de calcul et ne se limite pas à stocker des données. [P02, P06] Il n'y a pas d'attribut public (encapsulation.) [P03] Les méthodes ne sont pas trop longues (ex. 15 lignes de code.) [P6, P7] Sûreté Les méthodes se protègent (test des paramètres, levée d'exception.) [P20, P013] Chaque méthode possède au moins un test. [P18] Les tests sont automatisés et s'enchainent pour vérifier la non-régression. [P18] Évolutivité/Maintenance Les classes importées sont nommées explicitement. [Pj8] Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14]		[P1]
Style et conventions Les indentations sont homogènes. [NB : IDE eclipse peut gérer cela] [P18] La forme des noms respecte le style du langage (Par exemple, en Java : CamelCase pour les variables, CAPITALES pour les constantes) [P18] L'ordre des déclarations est toujours le même (Par exemple : attributs, constructeurs, méthodes publiques puis privées.) [P3] Les noms sont porteurs de sens. [P13] Modularité Le paquetage racine ne contient directement aucune classe. [Pj11] Le fichier package.info décrit le contenu et la responsabilité de chaque paquetage. [Pj2] Chaque classe a une responsabilité de calcul et ne se limite pas à stocker des données. [Po2, Po6] Il n'y a pas d'attribut public (encapsulation.) [Po3] Les méthodes ne sont pas trop longues (ex. 15 lignes de code.) [P6, P7] Sûreté Les méthodes se protègent (test des paramètres, levée d'exception.) [P20, Po13] Chaque méthode possède au moins un test. [P18] Les tests sont automatisés et s'enchainent pour vérifier la non-régression. [P18] Évolutivité/Maintenance Les classes importées sont nommées explicitement. [Pj8] Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14]		Il y a dans le code des <i>commentaires</i> à destination des mainteneurs (hors javadoc).
□ Les indentations sont homogènes. [NB : IDE eclipse peut gérer cela] [P18] □ La forme des noms respecte le style du langage (Par exemple, en Java : CamelCase pour les variables, CAPITALES pour les constantes) [P18] □ L'ordre des déclarations est toujours le même (Par exemple : attributs, constructeurs, méthodes publiques puis privées.) [P3] □ Les noms sont porteurs de sens. [P13] □ Modularité □ Le paquetage racine ne contient directement aucune classe. [Pj11] □ Le fichier package.info décrit le contenu et la responsabilité de chaque paquetage. [Pj2] □ Chaque classe a une responsabilité de calcul et ne se limite pas à stocker des données. [Po2, Po6] □ Il n'y a pas d'attribut public (encapsulation.) [Po3] □ Les méthodes ne sont pas trop longues (ex. 15 lignes de code.) [P6, P7] □ Sûreté □ Les méthodes se protègent (test des paramètres, levée d'exception.) [P20, Po13] □ Chaque méthode possède au moins un test. [P18] □ Les tests sont automatisés et s'enchainent pour vérifier la non-régression. [P18] □ Évolutivité/Maintenance □ Les classes importées sont nommées explicitement. [Pj8] □ Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14]		[P1,P5]
□ Les indentations sont homogènes. [NB : IDE eclipse peut gérer cela] [P18] □ La forme des noms respecte le style du langage (Par exemple, en Java : CamelCase pour les variables, CAPITALES pour les constantes) [P18] □ L'ordre des déclarations est toujours le même (Par exemple : attributs, constructeurs, méthodes publiques puis privées.) [P3] □ Les noms sont porteurs de sens. [P13] □ Modularité □ Le paquetage racine ne contient directement aucune classe. [Pj11] □ Le fichier package.info décrit le contenu et la responsabilité de chaque paquetage. [Pj2] □ Chaque classe a une responsabilité de calcul et ne se limite pas à stocker des données. [Po2, Po6] □ Il n'y a pas d'attribut public (encapsulation.) [Po3] □ Les méthodes ne sont pas trop longues (ex. 15 lignes de code.) [P6, P7] □ Sûreté □ Les méthodes se protègent (test des paramètres, levée d'exception.) [P20, Po13] □ Chaque méthode possède au moins un test. [P18] □ Les tests sont automatisés et s'enchainent pour vérifier la non-régression. [P18] □ Évolutivité/Maintenance □ Les classes importées sont nommées explicitement. [Pj8] □ Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14]		
□ La forme des noms respecte le style du langage (Par exemple, en Java : CamelCase pour les variables, CAPITALES pour les constantes) [P18] □ L'ordre des déclarations est toujours le même (Par exemple : attributs, constructeurs, méthodes publiques puis privées.) [P3] □ Les noms sont porteurs de sens. [P13] □ Modularité □ Le paquetage racine ne contient directement aucune classe. [Pj11] □ Le fichier package.info décrit le contenu et la responsabilité de chaque paquetage. [Pj2] □ Chaque classe a une responsabilité de calcul et ne se limite pas à stocker des données. [Po2, Po6] □ Il n'y a pas d'attribut public (encapsulation.) [Po3] □ Les méthodes ne sont pas trop longues (ex. 15 lignes de code.) [P6, P7] □ Sûreté □ Les méthodes se protègent (test des paramètres, levée d'exception.) [P20, Po13] □ Chaque méthode possède au moins un test. [P18] □ Les tests sont automatisés et s'enchainent pour vérifier la non-régression. [P18] □ Évolutivité/Maintenance □ Les classes importées sont nommées explicitement. [Pj8] □ Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14]		· · · · · · · · · · · · · · · · · · ·
pour les variables, CAPITALES pour les constantes) [P18] L'ordre des déclarations est toujours le même (Par exemple : attributs, constructeurs, méthodes publiques puis privées.) [P3] Les noms sont porteurs de sens. [P13] Modularité Le paquetage racine ne contient directement aucune classe. [Pj11] Le fichier package.info décrit le contenu et la responsabilité de chaque paquetage. [Pj2] Chaque classe a une responsabilité de calcul et ne se limite pas à stocker des données. [Po2, Po6] Il n'y a pas d'attribut public (encapsulation.) [Po3] Les méthodes ne sont pas trop longues (ex. 15 lignes de code.) [P6, P7] Sûreté Les méthodes se protègent (test des paramètres, levée d'exception.) [P20, Po13] Chaque méthode possède au moins un test. [P18] Les tests sont automatisés et s'enchainent pour vérifier la non-régression. [P18] Évolutivité/Maintenance Les classes importées sont nommées explicitement. [Pj8] Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14]		
□ L'ordre des déclarations est toujours le même (Par exemple : attributs, constructeurs, méthodes publiques puis privées.) [P3] □ Les noms sont porteurs de sens. [P13] □ Modularité □ Le paquetage racine ne contient directement aucune classe. [Pj11] □ Le fichier package.info décrit le contenu et la responsabilité de chaque paquetage. [Pj2] □ Chaque classe a une responsabilité de calcul et ne se limite pas à stocker des données. [Po2, Po6] □ Il n'y a pas d'attribut public (encapsulation.) [Po3] □ Les méthodes ne sont pas trop longues (ex. 15 lignes de code.) [P6, P7] □ Sûreté □ Les méthodes se protègent (test des paramètres, levée d'exception.) [P20, Po13] □ Chaque méthode possède au moins un test. [P18] □ Les tests sont automatisés et s'enchainent pour vérifier la non-régression. [P18] □ Évolutivité/Maintenance □ Les classes importées sont nommées explicitement. [Pj8] □ Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14]	Ш	
méthodes publiques puis privées.) [P3] Les noms sont porteurs de sens. [P13] Modularité Le paquetage racine ne contient directement aucune classe. [Pj11] Le fichier package.info décrit le contenu et la responsabilité de chaque paquetage. [Pj2] Chaque classe a une responsabilité de calcul et ne se limite pas à stocker des données. [Po2, Po6] Il n'y a pas d'attribut public (encapsulation.) [Po3] Les méthodes ne sont pas trop longues (ex. 15 lignes de code.) [P6, P7] Sûreté Les méthodes se protègent (test des paramètres, levée d'exception.) [P20, Po13] Chaque méthode possède au moins un test. [P18] Les tests sont automatisés et s'enchainent pour vérifier la non-régression. [P18] Évolutivité/Maintenance Les classes importées sont nommées explicitement. [Pj8] Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14]		- / L 1
Modularité Le paquetage racine ne contient directement aucune classe. [Pj11] Le fichier package.info décrit le contenu et la responsabilité de chaque paquetage. [Pj2] Chaque classe a une responsabilité de calcul et ne se limite pas à stocker des données. [Po2, Po6] Il n'y a pas d'attribut public (encapsulation.) [Po3] Les méthodes ne sont pas trop longues (ex. 15 lignes de code.) [P6, P7] Sûreté Les méthodes se protègent (test des paramètres, levée d'exception.) [P20, Po13] Chaque méthode possède au moins un test. [P18] Les tests sont automatisés et s'enchainent pour vérifier la non-régression. [P18] Évolutivité/Maintenance Les classes importées sont nommées explicitement. [Pj8] Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14]		y , and a second of the second
Modularité □ Le paquetage racine ne contient directement aucune classe. [Pj11] □ Le fichier package.info décrit le contenu et la responsabilité de chaque paquetage. [Pj2] □ Chaque classe a une responsabilité de calcul et ne se limite pas à stocker des données. [Po2, Po6] □ Il n'y a pas d'attribut public (encapsulation.) [Po3] □ Les méthodes ne sont pas trop longues (ex. 15 lignes de code.) [P6, P7] Sûreté □ Les méthodes se protègent (test des paramètres, levée d'exception.) [P20, Po13] □ Chaque méthode possède au moins un test. [P18] □ Les tests sont automatisés et s'enchainent pour vérifier la non-régression. [P18] □ Évolutivité/Maintenance □ Les classes importées sont nommées explicitement. [Pj8] □ Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14]		
□ Le paquetage racine ne contient directement aucune classe. [Pj11] □ Le fichier package.info décrit le contenu et la responsabilité de chaque paquetage. [Pj2] □ Chaque classe a une responsabilité de calcul et ne se limite pas à stocker des données. [Po2, Po6] □ Il n'y a pas d'attribut public (encapsulation.) [Po3] □ Les méthodes ne sont pas trop longues (ex. 15 lignes de code.) [P6, P7] □ Sûreté □ Les méthodes se protègent (test des paramètres, levée d'exception.) [P20, Po13] □ Chaque méthode possède au moins un test. [P18] □ Les tests sont automatisés et s'enchainent pour vérifier la non-régression. [P18] □ Évolutivité/Maintenance □ Les classes importées sont nommées explicitement. [Pj8] □ Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14]		Les noms sont porteurs de sens. [P13]
□ Le paquetage racine ne contient directement aucune classe. [Pj11] □ Le fichier package.info décrit le contenu et la responsabilité de chaque paquetage. [Pj2] □ Chaque classe a une responsabilité de calcul et ne se limite pas à stocker des données. [Po2, Po6] □ Il n'y a pas d'attribut public (encapsulation.) [Po3] □ Les méthodes ne sont pas trop longues (ex. 15 lignes de code.) [P6, P7] □ Sûreté □ Les méthodes se protègent (test des paramètres, levée d'exception.) [P20, Po13] □ Chaque méthode possède au moins un test. [P18] □ Les tests sont automatisés et s'enchainent pour vérifier la non-régression. [P18] □ Évolutivité/Maintenance □ Les classes importées sont nommées explicitement. [Pj8] □ Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14]		
Le fichier package.info décrit le contenu et la responsabilité de chaque paquetage. [Pj2] Chaque classe a une responsabilité de calcul et ne se limite pas à stocker des données. [Po2, Po6] Il n'y a pas d'attribut public (encapsulation.) [Po3] Les méthodes ne sont pas trop longues (ex. 15 lignes de code.) [P6, P7] Sûreté Les méthodes se protègent (test des paramètres, levée d'exception.) [P20, Po13] Chaque méthode possède au moins un test. [P18] Les tests sont automatisés et s'enchainent pour vérifier la non-régression. [P18] Évolutivité/Maintenance Les classes importées sont nommées explicitement. [Pj8] Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14]		
[Pj2] □ Chaque classe a une responsabilité de calcul et ne se limite pas à stocker des données. [Po2, Po6] □ Il n'y a pas d'attribut public (encapsulation.) [Po3] □ Les méthodes ne sont pas trop longues (ex. 15 lignes de code.) [P6, P7] Sûreté □ Les méthodes se protègent (test des paramètres, levée d'exception.) [P20, Po13] □ Chaque méthode possède au moins un test. [P18] □ Les tests sont automatisés et s'enchainent pour vérifier la non-régression. [P18] Évolutivité/Maintenance □ Les classes importées sont nommées explicitement. [Pj8] □ Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14]		
 □ Chaque classe a une responsabilité de calcul et ne se limite pas à stocker des données. [Po2, Po6] □ Il n'y a pas d'attribut public (encapsulation.) [Po3] □ Les méthodes ne sont pas trop longues (ex. 15 lignes de code.) [P6, P7] Sûreté □ Les méthodes se protègent (test des paramètres, levée d'exception.) [P20, Po13] □ Chaque méthode possède au moins un test. [P18] □ Les tests sont automatisés et s'enchainent pour vérifier la non-régression. [P18] □ Évolutivité/Maintenance □ Les classes importées sont nommées explicitement. [Pj8] □ Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14] 		
[Po2, Po6] □ Il n'y a pas d'attribut public (encapsulation.) [Po3] □ Les méthodes ne sont pas trop longues (ex. 15 lignes de code.) [P6, P7] Sûreté □ Les méthodes se protègent (test des paramètres, levée d'exception.) [P20, Po13] □ Chaque méthode possède au moins un test. [P18] □ Les tests sont automatisés et s'enchainent pour vérifier la non-régression. [P18] Évolutivité/Maintenance □ Les classes importées sont nommées explicitement. [Pj8] □ Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14]		
□ Il n'y a pas d'attribut public (encapsulation.) [Po3] □ Les méthodes ne sont pas trop longues (ex. 15 lignes de code.) [P6, P7] Sûreté □ Les méthodes se protègent (test des paramètres, levée d'exception.) [P20, Po13] □ Chaque méthode possède au moins un test. [P18] □ Les tests sont automatisés et s'enchainent pour vérifier la non-régression. [P18] □ Évolutivité/Maintenance □ Les classes importées sont nommées explicitement. [Pj8] □ Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14]		
□ Les méthodes ne sont pas trop longues (ex. 15 lignes de code.) [P6, P7] Sûreté □ Les méthodes se protègent (test des paramètres, levée d'exception.) [P20, Po13] □ Chaque méthode possède au moins un test. [P18] □ Les tests sont automatisés et s'enchainent pour vérifier la non-régression. [P18] Évolutivité/Maintenance □ Les classes importées sont nommées explicitement. [Pj8] □ Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14]		
Sûreté Les méthodes se protègent (test des paramètres, levée d'exception.) [P20, Po13] Chaque méthode possède au moins un test. [P18] Les tests sont automatisés et s'enchainent pour vérifier la non-régression. [P18] Évolutivité/Maintenance Les classes importées sont nommées explicitement. [Pj8] Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14]		/ - / - /
 □ Les méthodes se protègent (test des paramètres, levée d'exception.) [P20, Po13] □ Chaque méthode possède au moins un test. [P18] □ Les tests sont automatisés et s'enchainent pour vérifier la non-régression. [P18] □ Évolutivité/Maintenance □ Les classes importées sont nommées explicitement. [Pj8] □ Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14] 	Ш	Les méthodes ne sont pas trop longues (ex. 15 lignes de code.) [P6, P7]
 □ Les méthodes se protègent (test des paramètres, levée d'exception.) [P20, Po13] □ Chaque méthode possède au moins un test. [P18] □ Les tests sont automatisés et s'enchainent pour vérifier la non-régression. [P18] □ Évolutivité/Maintenance □ Les classes importées sont nommées explicitement. [Pj8] □ Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14] 		CA
 □ Chaque méthode possède au moins un test. [P18] □ Les tests sont automatisés et s'enchainent pour vérifier la non-régression. [P18] □ Évolutivité/Maintenance □ Les classes importées sont nommées explicitement. [Pj8] □ Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14] 		
 ☐ Les tests sont automatisés et s'enchainent pour vérifier la non-régression. [P18] Évolutivité/Maintenance ☐ Les classes importées sont nommées explicitement. [Pj8] ☐ Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14] 		
Évolutivité/Maintenance Les classes importées sont nommées explicitement. [Pj8] Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14]		
 □ Les classes importées sont nommées explicitement. [Pj8] □ Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14] 	Ш	Les tests sont automatises et s'enchainent pour verifier la non-regression. [P18]
 □ Les classes importées sont nommées explicitement. [Pj8] □ Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14] 		Évolutivité/ <i>Maintenance</i>
☐ Il n'y a pas de valeur utilisée sans être déclarée par une constante. [P14]		,
		Le code est factorisé. [P9]