## java linter

Le but de ce mini-projet est de créer un linter sous java.

Définition du linter : [Wikipedia](https://en.wikipedia.org/wiki/Lint\_%28software%29)

#### Travail à faire

Implémenter les règles suivantes :

| Identifiant |Niveau | Description |

|--|--|--|

| LINT\_REG\_000 |BLOCKER|Les erreurs de syntaxe ne sont pas autorisées|

| LINT\_REG\_001 |MEDIUM |Les imports non utilisés sont à supprimer |

| LINT\_REG\_002 |HIGHEST|Les types java `(class, interface, annotation, enum)` commence par une majuscule, elles ne doivent pas avoir des sous-tirets `\_` |

| LINT\_REG\_003 |HIGH|Les variables locales commencent par une minuscule |

| LINT\_REG\_004 |HIGH|Les attributs d'une classe commencent par une minuscule |

| LINT\_REG\_005 |MEDIUM|Les constantes d'une classe sont écrites en majuscule, avec des `\_` comme séparateur|

~~| LINT\_REG\_006 |HIGHEST|Les expressions booléennes ne doivent avoir plus que `2` opérandes |~~

| LINT\_REG\_007 |LOW|Les éléments d'une énumeration sont en majuscule, avec des `\_` comme séparateur|

| LINT\_REG\_008 |HIGHEST|Le corps d'une méthode ne doit pas dépasser `30` lignes|

~~| LINT\_REG\_009 |HIGH|Les instanciations anonymes sont à remplacer par des expressions `lambda`|~~

~~| LINT\_REG\_010 |MEDIUM|Les expressions `lambda` intuitives sont à remplacer par `method reference` |~~

| LINT\_REG\_011 |HIGHEST|Le nombre de méthodes ne doit pas dépasser `20` méthodes déclarées par classe|

| LINT\_REG\_012 |HIGHEST|Le nombre de paramètres d'une méthode/constructeur ne doit dépasser `2`|

| LINT\_REG\_013 |HIGHEST|Les attributs de classe doivent avoir une visibilité déclarée|

~~| LINT\_REG\_014 |LOW|Préférer l'utilisation d'une seule instruction de sortie `(return, throw)` dans les méthodes|~~

~~| LINT\_REG\_015|LOW|Ne pas catcher les exceptions sans les logger|~~

~~| LINT\_REG\_016|MEDIUM|Les variables non utilisées sont à supprimer|~~

~~| LINT\_REG\_017|MEDIUM|Les méthodes privées non utilisées sont à supprimer|~~

~~| LINT\_REG\_018 |LOW|les clauses `if` , `else` doivent avoir des accolades |~~

Si une règle est bloquante, alors remonter l'erreur est mettre fin à l'analyse du fichier en cours.

### INPUT :

Le format d'entrée est une liste de fichiers `.java` et/ou des dossiers contenant des fichiers `.java`

### OUTPUT :

Le résultat de l'analyse est écrit sur un fichier selon le type choisi par l'utilisateur : `CSV, JSON, HTML et Markdown`.

Exemple de résultat brut :

```

LEVEL RULE DESCRIPTION

MEDIUM [LINT\_REG\_001] Unused import 'java.util.List' at 'D:\0001\_PERSO\0003\_CODE\ensao-lint\testFiles\normalExecution\Example.java:1'

MEDIUM [LINT\_REG\_001] Unused import 'com.github.javaparser.ast.ImportDeclaration' at 'D:\0001\_PERSO\0003\_CODE\ensao-lint\testFiles\normalExecution\Example.java:2'

MEDIUM [LINT\_REG\_001] Unused import 'FFFF' at 'D:\0001\_PERSO\0003\_CODE\ensao-lint\testFiles\normalExecution\Example2.java:4'

MEDIUM [LINT\_REG\_001] Unused import 'org.apache.commons.lang3.StringUtils' at 'D:\0001\_PERSO\0003\_CODE\ensao-lint\testFiles\staticCall\StaticCallExample.java:1'

MEDIUM [LINT\_REG\_001] Unused import 'example.Access' at 'D:\0001\_PERSO\0003\_CODE\ensao-lint\testFiles\staticCall\StaticCallExample.java:2'

MEDIUM [LINT\_REG\_001] Unused import 'org.apache.commons.lang3.StringUtils' at 'D:\0001\_PERSO\0003\_CODE\ensao-lint\testFiles\staticCall\StaticCallExample.java:1'

MEDIUM [LINT\_REG\_001] Unused import 'example.Access' at 'D:\0001\_PERSO\0003\_CODE\ensao-lint\testFiles\staticCall\StaticCallExample.java:2'

```

### Soumission de votre travail

Procéder par un fork de ce répo, ensuite préparer un pull request vers le même répo.

### Grille de notation

| Elément| Pondération|

|--|--|

|Implémentation des règles | 25%|

|Utilisation des instructions java ≥ 8| 20%|

|Tests et couvertures|20%|

|Qualité du code |20%|

|Application des design patterns|10%|

|Documentation des règles |5%|

#### NB

`LINT\_REG\_000` et `LINT\_REG\_001` sont partiellement implémentées.

### Annexes

1. Utiliser maven pour exécuter les tests unitaires :

```

mvn clean test

```

2. le plugin `jacoco` est intégré et est configuré pour être exécuté pendant la phase de test, il permet d'établir le taux de la couverture des tests.

Une fois le rapport est établi, vous pouvez le consulter sur `target/site/jacoco/index.html`