

# Analyse statique et métrique du logiciel TP3

Pour ce TP, vous utiliserez la machine virtuelle windows Informatique-Développement et Netbeans 8.2

### Evaluation des TPs - remise individuelle obligatoire

A la fin de la séance, vous déposerez sur moodle un compte rendu de TP avec les réponses aux questions posées et des captures écran des modélisations que vous avez réalisées. Le dépôt doit être fait en un seul fichier pdf et le nom du fichier doit être au format votre\_nom\_votre\_prenom.pdf.

Le code java écrit doit être intégré au rapport au format texte. Les comptes rendus déposés sous forme d'archives ne seront pas évaluées

#### Exercice 1 : Respect des conventions de codage - google java style

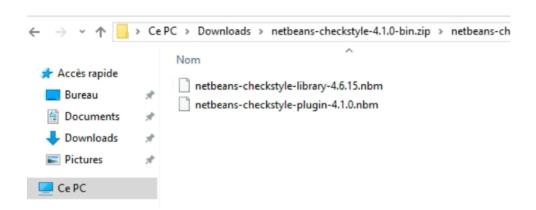
#### a) Préparation du projet

 Téléchargez le code Goldbach.java disponible sur l'ENT et créez un nouveau projet avec ce code.

#### b) Installation check style

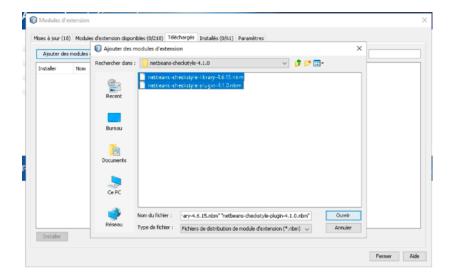
Téléchargez le plugin CheckStyle pour netBeans à l'adresse : <a href="http://plugins.netbeans.org/plugin/3413/checkstyle-beans">http://plugins.netbeans.org/plugin/3413/checkstyle-beans</a>

Décompressez cette archive sur le disque dur

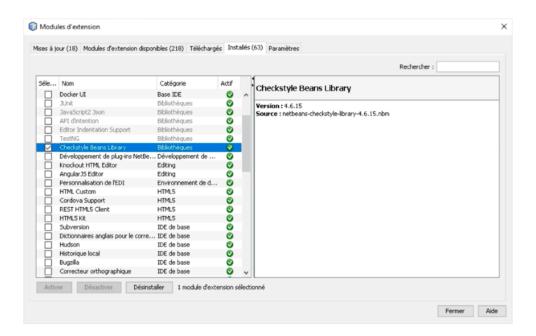


 Installez ce plugin par le menu Outils / Modules d'extension (onglets extensions téléchargées). Sélectionnez le répertoire local dans lequel se trouve le plugin, puis clichez sur « Installer ». A la fin de l'installation, l'IDE va redémarrer.





Après le redémarrage, vous devez activer l'extension :



Vous verrez alors apparaitre les erreurs check style dans la fenêtre de l'éditeur.

```
package javaapplication2;
7
      public class Goldbach {
              boolean[] isprime;
              int primes;
              int[] list ;
12
 ŧ
   口
              public int getPrimes(int n){
                       // count primes
                      int primes = 0;
                      for (int i = 2; i < n; i++)
                          if (isprime[i]) primes++;
19
                      return primes;
20
```



## c) Application des règles de codage

 Modifiez le code de manière à ce que checkStyle ne détecte plus de violations aux règles de codage.

#### Exercice 2: Metrics et refractoring

Calculez la complexité cyclomatique de la méthode GetComponent (vous donnerez dans votre compte rendu une photo du graphe de contrôle construit).

Cherchez à réduire cette complexité cyclomatique à 7 (ou moins) par refactoring.

Vous trouverez des informations sur le refactoring à l'adresse : https://sourcemaking.com/refactoring/refactorings

La classe Goldbach peut donner lieu à un certain nombre de refactoring et notamment :

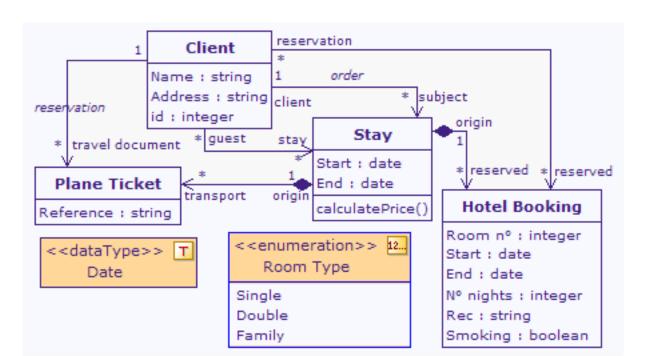
- la diminution de la complexité cyclomatique par extraction de méthodes : <a href="https://sourcemaking.com/refactoring/extract-method">https://sourcemaking.com/refactoring/extract-method</a>
- La simplification des conditions avec l'extraction de variable : <a href="https://sourcemaking.com/refactoring/extract-variable">https://sourcemaking.com/refactoring/extract-variable</a>

D'autres méthodes de refactoring tel que :

- Replace Temp with Query: https://sourcemaking.com/refactoring/replace-temp-withquery
- Replace Method with Object: https://sourcemaking.com/refactoring/replace-methodwith-method-object

#### Exercice 4: Métriques du TP précédent

Donnez les valeurs des métriques pour le code java écrit pour les classes correspondant à l'exercice :



Génie logiciel Licence informatique Université de la Rochelle Armelle Prigent

