


# TD 09 – Premiers Principes de Conception Orientée Objet

## Cours 1 Single Responsibility Principle

### 1.1 Définition



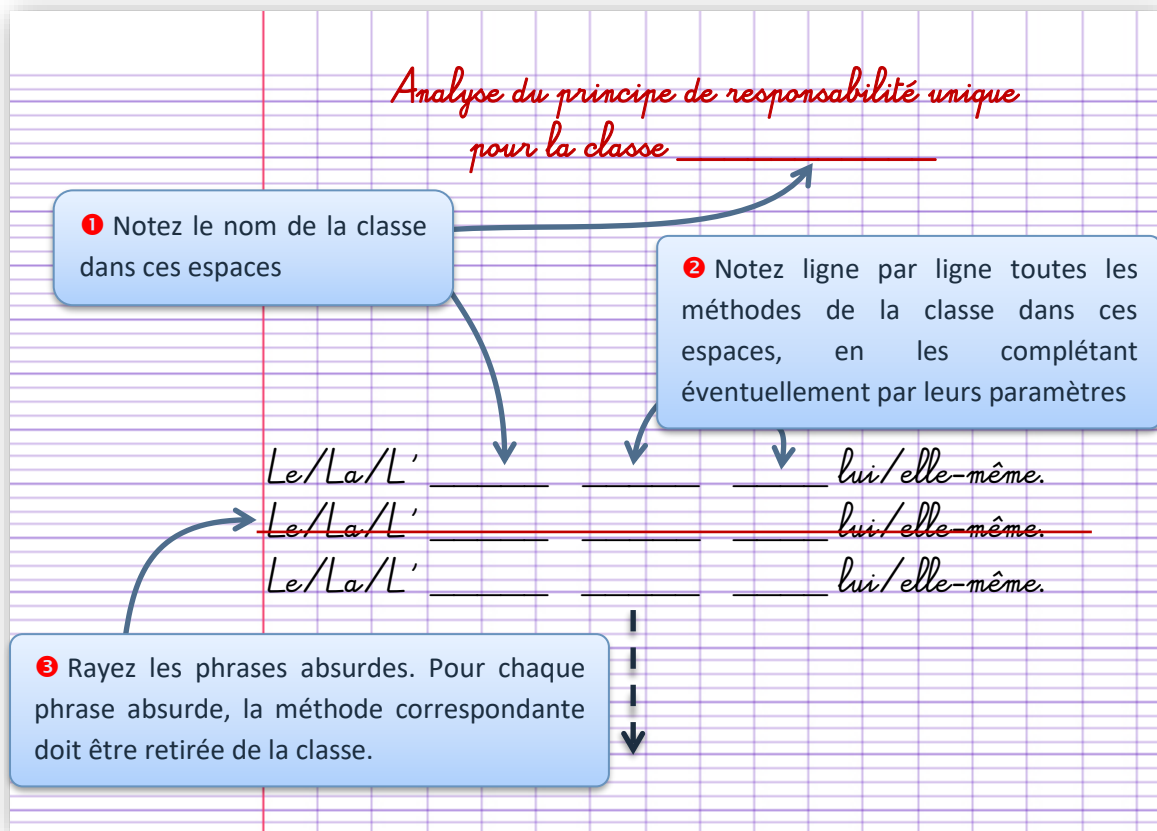
 Vous avez implémenté le principe de responsabilité unique si chacune de vos classes n'a qu'une *seule raison de changer*.

 Appliquer ce principe ne doit pas induire de duplication de code. Si cela est le cas, alors il vous faut probablement reprendre votre conception. Dans une bonne conception, une classe ne fait qu'une chose, la fait bien, et aucune autre classe ne la fait également.

 Ce principe est parfois appelé « cohésion »

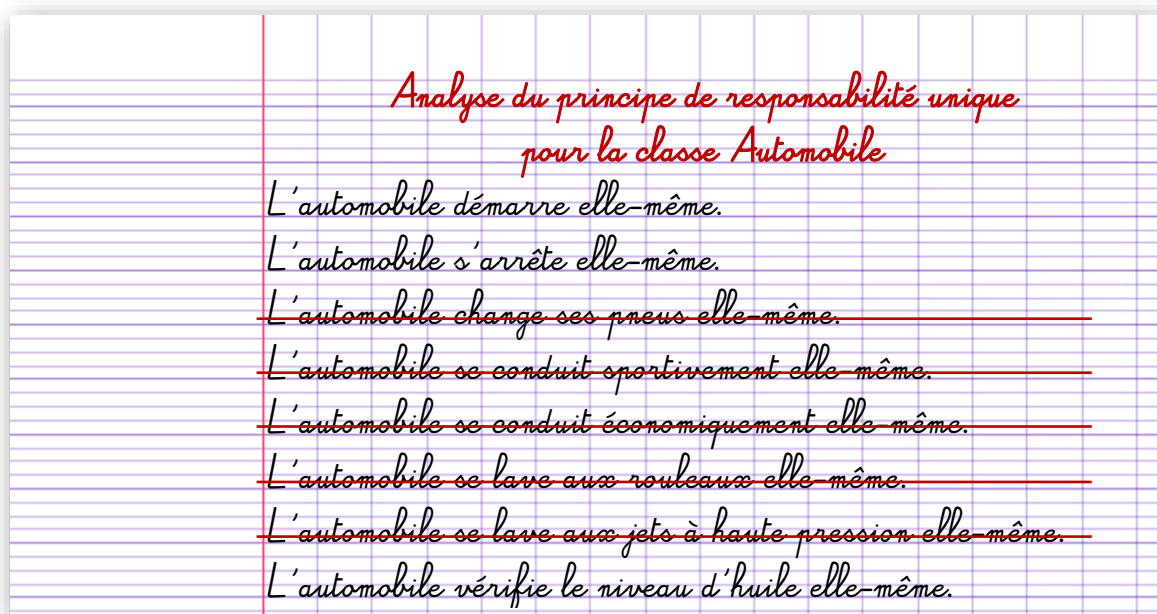
### 1.2 Comment détecter les responsabilités multiples d'une classe ?

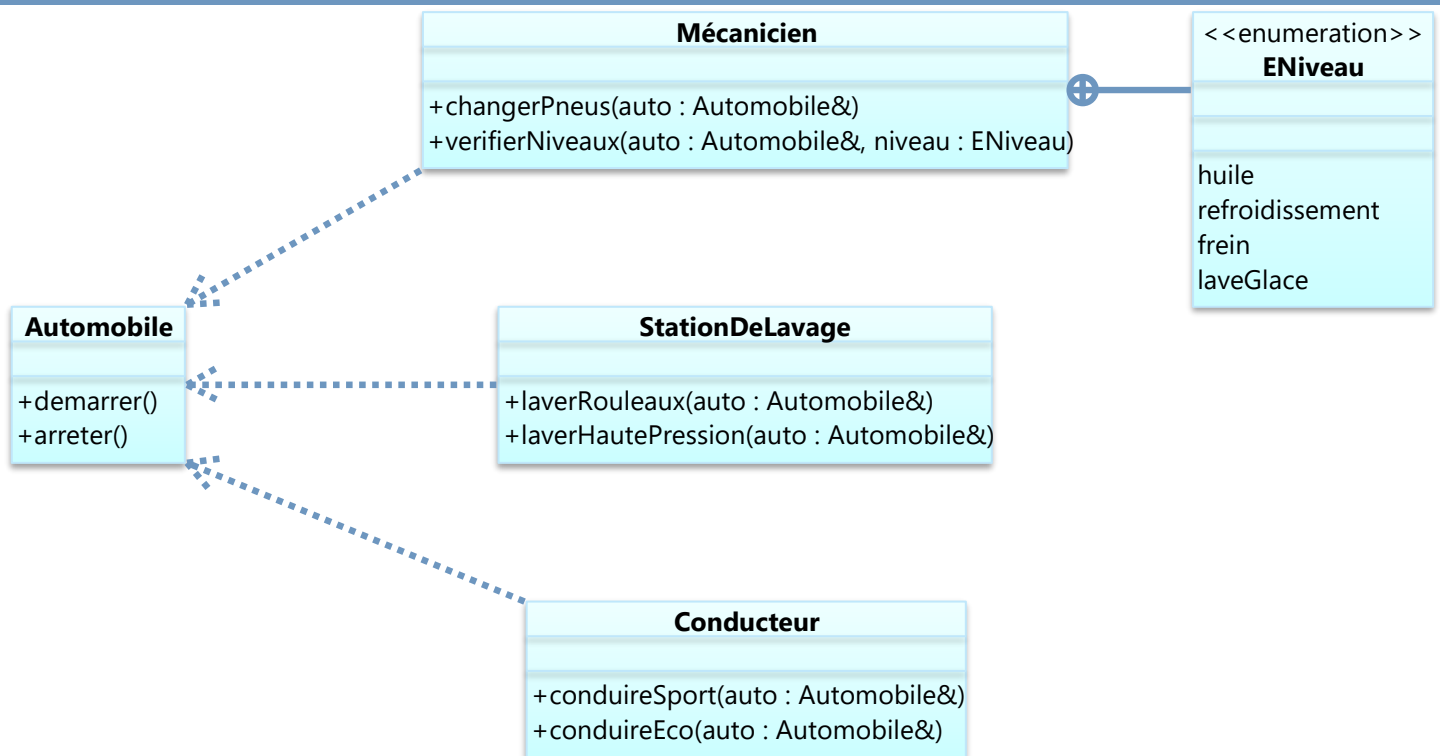
Il suffit de réaliser un petit test simple sur la classe. Pour chaque méthode de la classe, demandez-vous si un objet de cette classe réalise cette action lui-même. Ainsi, pour chaque classe, on peut établir une fiche comme le modèle suivant :



## Exercice 2 Un autre regard


- ☞ Appliquez le principe de responsabilité unique à la modélisation de l'automobile.
- ☞ Utilisez le processus décrit dans la section 1.2
- ☞ Redessinez le nouveau diagramme de classes.

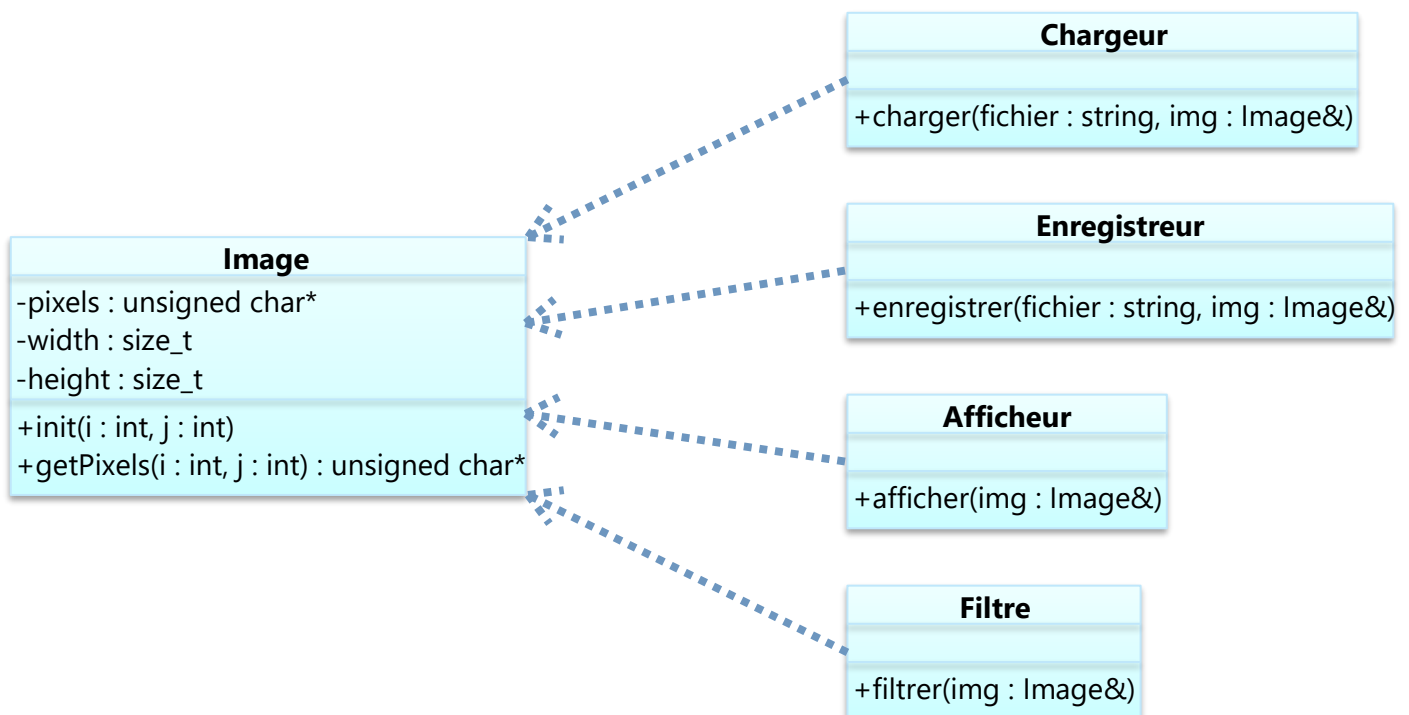




### Exercice 3 Etes-vous sûr d'avoir compris ?

Dans cet exercice, il s'agit d'analyser un nouveau problème. Nous souhaitons réaliser un logiciel de filtrage d'image. En plus de l'application de filtres, les images peuvent bien sûr se charger et s'enregistrer sur le système de fichiers et s'afficher à l'écran.

-  En appliquant le principe de responsabilité unique, analysez et concevez un logiciel répondant aux exigences.



*Analyse du principe de responsabilité unique  
pour la classe Image*

*L'image s'initialise elle-même.*

*L'image renvoie les valeurs de ses pixels (get) elle-même.*

*Analyse du principe de responsabilité unique  
pour la classe Chargeur*

*Le chargeur charge une image lui-même.*

*Analyse du principe de responsabilité unique  
pour la classe Enregistreur*

*L'enregistreur enregistre une image lui-même.*

*Analyse du principe de responsabilité unique  
pour la classe Afficheur*

*L'afficheur affiche une image lui-même.*

*Analyse du principe de responsabilité unique  
pour la classe Filtre*

*Le filtre filtre une image lui-même.*