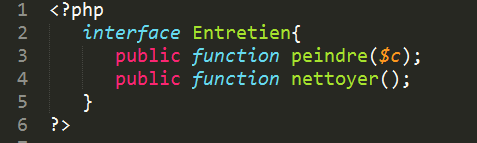
* **Une interface: une classe entièrement abstraite**

Une interface se comporte comme une classe abstraite dont toutes les méthodes sont abstraites. Cependant l'ultime différence entre une classe abstraite et une interface est que les classes qui vont hériter des classes abstraites constituent des sous familles de la classe mère, par contre une interface peut être utilisée par des classes dont les finalités ne sont pas forcément semblables.

Pour mieux comprendre le principe, supposons que nous avons une classe abstraite du nom de Vehicule. Les classes Filles qui peuvent en hériter sont par exemple Voiture, Camion et Fourgon. Ces classes sont en réalité des véhicules plus ou moins différents, donc ils constituent un sous ensemble de la classe Vehicule. Par contre les classes Vehicule et Maison ne sont pas de la même famille. Néanmoins chacune d'entre elles peut accueillir une méthode du nom de peindre(). C'est là où une interface qui renferme la méthode peindre() peut être utile.

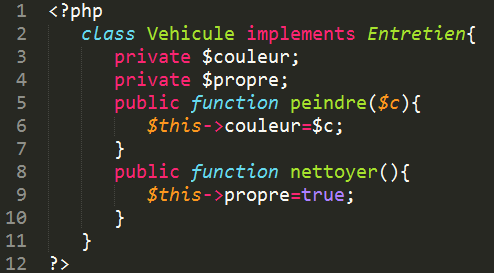
* **Définir une interface**

Pour définir une interface, on utilise le mot clé **interface** suivi du nom de celle-ci (avec une première lettre majuscule de préférence, tout comme pour une classe) suivi des accolades qui renfermeront les membres de l'interface comme ceci:



**Implémenter une interface**

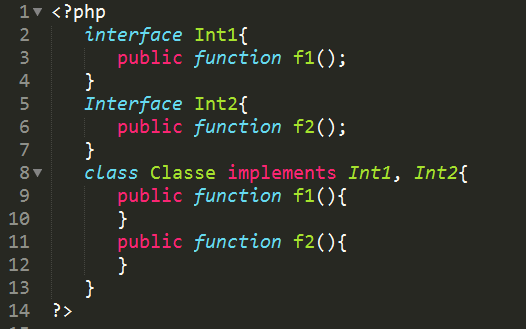
Pour se servir de l'interface créée, il faut l'implémenter par une classe à l'aide du mot clé **implements** comme ceci:



Dans la classe **Vehicule** on doit obligatoirement redéfinir les méthodes peindre() et nettoyer().

**Implémenter plusieurs interfaces**

A l'inverse de l'héritage en PHP, on peut implémenter plusieurs interface simultanément:



Dans ce cas, dans la classe **Classe** on doit redéfinir les méthodes f1() et f2() dont le prototype est défini respectivement dans les interfaces **Int1** et **Int2**.

**Exercice :**

Crée une interface nommée abscisse qui déclare une méthode setx(), dont le rôle est d’initialiser une abscisse . Il crée également une interface nommée ordonnee déclarant une méthode sety(), dont le rôle est similaire pour l’ordonnée d’un point . Ces interfaces sont implémentées par deux classes différentes. La classe pointaxe sente un point sur un axe, n’ayant donc qu’une seule coordonnée. Elle implémente l’interface abscisse et définit la méthode setx().

La classe pointplan représente un point du plan, avec deux coordonnées. Elle implémente donc les deux interfaces abscisse et ordonnee et définit les méthodes setx()et sety() comme l’imposent les interfaces.

L’implémentation des interfaces dans une classe impose certes la définition de leurs méthodes mais n’empêche pas d’enrichir la classe avec d’autres méthodes. La classe pointplan possède une méthode supplémentaire module(), qui retourne la distance du point M(x,y) à l’origine du repère . Elle pourrait, par exemple, s’enrichir d’une méthode retournant l’angle (Ox,OM) et permettre ainsi la gestion des nombres complexes.

