

Support de cours

Cours:

Introduction à la programmation orientée objet (en Java)

Vidéo:

W11-02-progclasses-JAVA-pt3

Concepts (extraits des sous-titres générés automatiquement) :

Déclaration des méthodes. Sein de la classe. Cas de notre classe rectangle. Cas de notre exemple de la classe. Méthode surface. Hauteur fois. Classe rectangle. Attributs de la classe. Seule différence. Nom de la méthode. Entête de la méthode. Liste des éventuels paramètres. Type de retour. Méthode de la classe rectangle. Valeur de retour double.



[vers la recherche de séquences vidéo](#)

(dans Introduction à la programmation orientée objet (en Java).)



[vers la vidéo](#)

Center for Digital Education. Plus de matériel de soutien pédagogique ici :

<https://www.epfl.ch/education/educational-initiatives/cede/educational-technologies-gallery/boocs-en/>

Classes, objets, attributs et méthodes en Java

(Partie 3)

Introduction à la programmation orientée objet (en Java)

Jamila Sam, Jean-Cédric Chappelier et Vincent Lepetit

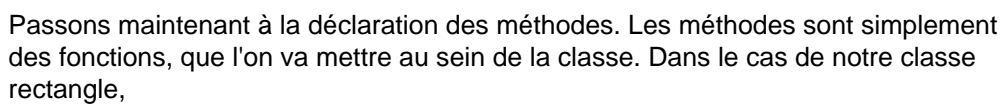
...

notes

résumé

0m 0s





Les attributs d'une classe constituent des variables *directement accessibles* dans toutes les méthodes de la classe (*i.e.* des « variables *globales à la classe* »).

On parle de « portée de classe ».

Il n'est donc **pas nécessaire de les passer comme arguments des méthodes**.

Par exemple, dans toutes les méthodes de la classe `Rectangle`, l'identificateur `hauteur` (resp. `largeur`) fait *a priori* référence à la valeur de l'attribut `hauteur` (resp. `largeur`) de l'instance concernée (par l'appel de la méthode en question)

on va s'intéresser ici à lui ajouter la méthode surface. Donc une méthode, ça se déclare simplement comme d'habitude, avec le type de retour, le nom de la méthode, puis ensuite entre parenthèses rondes, la liste des éventuels paramètres. Tout ceci qui est l'entête de la méthode, étant suivi par la définition de la méthode, le corps de cette méthode, entre accolades. La seule différence est que les méthodes sont mises dans la classe elle-même. Dans le cas de notre exemple de la classe `Rectangle`, la méthode surface s'ajouterait donc dans la classe `Rectangle`, ici avec donc sa valeur de retour double, une surface return, un double, son nom. Ici elle n'a pas besoin de prendre de paramètres, et elle retournerait donc la hauteur fois la largeur. La question qu'on pourrait se poser c'est, mais où sont donc passés les paramètres ? En effet, si l'on avait à écrire une méthode, comme nous le faisons auparavant, en dehors de toute classe, alors on aurait eu besoin de passer la hauteur et passer la largeur, comme argument à la fonction, de sorte que hauteur et largeur soient connues pour effectuer ce calcul. Comment se fait-il qu'ici dans la méthode de la classe `Rectangle` on ait rien eu besoin de passer ? Parce que tout simplement, cette hauteur et cette largeur sont des attributs de la classe. La méthode surface faisant partie de la classe rectangle, elle a accès aux attributs hauteur et largeur, de la classe rectangle.

notes

résumé

0m 13s



Les méthodes sont donc :

- ▶ des fonctions propres à la classe
- ▶ qui ont donc **accès** aux attributs de la classe

👉 Il ne faut donc **pas** passer les attributs comme arguments aux méthodes de la classe !

Exemple :

```
class Rectangle {  
    //...  
    double surface()  
    {  
        return hauteur * largeur;  
    }  
}
```

C'est ce qu'on appelle techniquement une portée de classe. C'est-à-dire que tous les attributs d'une classe sont connus de l'entièreté de cette classe, en particulier de chacune de ces méthodes. Par exemple, toutes les méthodes de la classe rectangle ont accès à la hauteur et à la largeur. Il n'est pas nécessaire de passer les attributs comme arguments aux méthodes. Pour résumer, une méthode, c'est une fonction propre à une classe, on va la mettre dans les classes, mais qui en plus de ce fait, a accès aux différents attributs de la classe. Il est donc très important, et c'est une erreur de débutant, des premières écritures de méthodes. Il est très important de ne pas passer les attributs à une méthode de classe. Pour revenir encore une fois sur l'exemple, l'exemple est tout à fait correct ici, on a bien une méthode surface qui ne prend pas du tout de paramètres, et qui peut accéder à la hauteur et à la largeur de Rectangle, qui ont été déclarées comme précédemment dans la classe. comme précédemment dans la classe.

notes

résumé

1m 37s

