

Support de cours

Cours:

Introduction à la programmation orientée objet (en Java)

Vidéo:

W11-02-progclasses-JAVA-pt2

Concepts (extraits des sous-titres générés automatiquement) :

Classe rectangle. Classe dessin. Premier cas. Environnement de développement. Déclaration initialisation d'une instance. Nom de l'attribut. Utilisation des attributs d'une instance. Compilation javac dessin.java. Seule fois. Méthode main. Environnement de développement intégré. Cas précédent. Java dessin. Fichier rectangle.class. Classes.



[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 2)

Introduction à la programmation orientée objet (en Java)

Jamila Sam, Jean-Cédric Chappelier et Vincent Lepetit

...

notes

résumé

0m 0s





notes

0m 1s



Où déclarer les classes ?

1. Déclaration de plusieurs classes dans le même fichier : la classe `Rectangle` et la classe qui l'utilise `Dessin` déclarées dans `Dessin.java`

```
class Dessin
{
    public static void main(String[] args)
    {
        Rectangle r;
        // ....
    }
}
class Rectangle
{
    double largeur;
    double hauteur;
    //...
}
```

> `javac Dessin.java`

produit les fichiers :

`Dessin.class`

`Rectangle.class`

!

Le compilateur crée un fichier `.class` pour chaque classe

Vous avez pour cela deux solutions. Vous pouvez déclarer toutes vos classes dans un même fichier, ou mettre une classe par fichier. Regardons le premier cas, où on déclare toutes les classes dans le même fichier. Supposons, peu importe les détails, retenons les idées, que l'on veuille avoir une classe `Dessin`, qui utilise par exemple une classe `Rectangle`. Tout ceci est déclaré dans un seul et même fichier, qu'on a décidé d'appeler `Dessin.java`. On a donc ici l'intégralité du fichier `Dessin.java`. On peut comme ça déclarer ici une classe `rectangle`, une autre classe `Dessin`. Concrètement, pour le compiler, on utilisera la compilation `javac Dessin.java`, que vous tapiez directement cette commande sur le terminal, ou que ce soit votre environnement de développement, Eclipse ou autre, qui le fasse pour vous. Cette compilation va par contre produire deux fichiers. Un fichier `Dessin.class`, et un fichier `Rectangle.class`, et ensuite pour exécuter tout ceci, la méthode `main` étant dans la classe `Dessin`,

notes

résumé

0m 5s



2. Déclaration de chaque classe dans un fichier à part :

- ▶ Rectangle déclarée dans `Rectangle.java`
- ▶ Dessin déclarée dans `Dessin.java`
- ▶ Compilation de *chaque* fichier nécessaire

```
> javac Rectangle.java  
> javac Dessin.java
```

produit les fichiers:

```
Rectangle.class  
Dessin.class
```

Seule la classe contenant main est exécutable:

```
> java Rectangle  
Exception in thread "main" java.lang.NoSuchMethodError: main
```

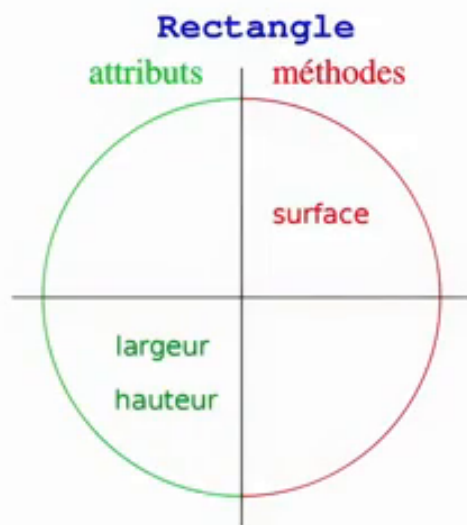
on écrirait simplement `java Dessin` sur le terminal. Ou on lancerait directement l'exécution de ce `dessin.class` depuis éclipse par exemple. Voilà donc la première solution pour déclarer les classes. Une autre solution consiste à déclarer une classe par fichier. Donc chaque fois que vous avez une classe, vous avez un fichier à part, donc vous déclareriez la classe `Rectangle` dans, par exemple, le fichier `Rectangle.java`, la classe `Dessin` dans le fichier `Dessin.java`. Et dans ce cas, si vous compilez depuis le terminal, il faudrait compiler séparément chacun des fichiers. Bien sûr dans un environnement de développement intégré, ces deux fichiers seraient intégrés dans le même projet, et on les compilerait séparément, en appelant une seule fois la commande `Build`. C'est éclipse qui se chargerait de compiler ces deux fichiers, et qui produirait comme dans le cas précédent, deux fichiers `.class`. Si on l'exécutait depuis le terminal, on taperait ici, `java Dessin`, puisque c'est la classe `Dessin` qui a la fonction `main`. C'est également ce que fera pour vous éclipse, si vous lancez l'exécution depuis là. Si par hasard, vous aviez envie de faire `java` sur `Rectangle.class`,

notes

résumé

1m 13s





rectangle qui ne contient pas de main, va vous donner à ce moment-là une exception, qui vous dit qu'il n'existe pas de méthode main dans la classe rectangle. Voyons maintenant comment remplir cette partie, c'est-à-dire comment mettre des attributs et des méthodes à notre classe. Commençons par les attributs.

notes

résumé

2m 25s



La syntaxe de la déclaration des attributs est la suivante :

`type nom_attribut ;`

Exemple :

les attributs `hauteur` et `largeur`, de type `double`, de la classe `Rectangle` pourront être déclarés par :

```
class Rectangle {  
    double hauteur;  
    double largeur;  
}
```

Les attributs de la classe `Rectangle` sont sa largeur et sa hauteur. Pour déclarer des attributs, il suffit de déclarer les différents attributs en mettant le type et le nom de l'attribut.

notes

résumé

2m 48s



L'accès aux valeurs des attributs d'une instance de nom `nom_instance` se fait comme pour accéder aux champs d'une structure :

```
nom_instance.nom_attribut
```

Exemple :

la valeur de l'attribut `hauteur` d'une instance `rect1` de la classe `Rectangle` sera référencée par l'expression :

```
rect1.hauteur
```

On déclare ici, pour chacun des attributs, son type, son nom d'attribut, suivi d'un point virgule. Pour notre exemple de classe `Rectangle`, ceci donnerait simplement, double hauteur, double largeur, c'est aussi simple que ça.

notes

résumé

3m 1s



L'instruction :

```
nomClasse instance = new nomClasse();
```

crée une instance de type `nomClasse` et initialise tous ses attributs avec des valeurs par défaut :

<code>int</code>	<code>0</code>
<code>double</code>	<code>0.0</code>
<code>boolean</code>	<code>false</code>
<code>objets</code>	<code>null</code>

Exemple :

```
Rectangle rect = new Rectangle();
```

pour avoir la méthode main. Et qui utilise une classe Rectangle, dans laquelle on déclare une hauteur de type double, et une largeur de type double. La classe exemple ici va créer une instance de la classe Rectangle. Pour ceci, il faut utiliser la syntaxe suivante. La déclaration initialisation d'une instance se fait en utilisant le nom de classe, suivi du nom d'instance, et puis = new avec le nom de classe, suivi de parenthèses, comme ceci. Par exemple, on l'a vu dans l'exemple précédent,

notes

résumé

3m 37s



pour créer une instance rect de la classe Rectangle, on écrirait `rect = new Rectangle`. Ce qui va donc créer et initialiser tous les attributs avec les valeurs suivantes. Si on a des attributs de type int, ça sera la valeur zéro int. Si on a des attributs de type double, ça sera la valeur 0.0. Pour les boolean, ça sera false, et pour les objets, ça sera la valeur spéciale null, sur laquelle nous reviendrons par la suite. Dans notre exemple précédent, ici à ce stade, `rect1` a une largeur et une hauteur nulle, avant qu'on les modifie. avant qu'on les modifie.

résumé

