

Support de cours

Cours:

Introduction à la programmation orientée objet (en C++)

Vidéo:

W11-02-progclasses-CPP-pt2

Concepts (extraits des sous-titres générés automatiquement) :

Exemple de class rectangle. Nom de l'attribut. Champs d'une structure. Utilisation des attributs d'une instance. Attributs de la classe. Différents attributs. Point virgule. Double hauteur. Nom d'attribut. Nom de l'instance. Petit ingrédient. Attributs. Déclaration de notre classe. Structures telles. Double largeur.



[vers la recherche de séquences vidéo](#)

(dans Introduction à la programmation orientée objet (en C++).)



[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 C++ (Partie 2)

Introduction à la programmation orientée objet (en C++)

Jean-Cédric Chappelier, Jamila Sam et Vincent Lepetit

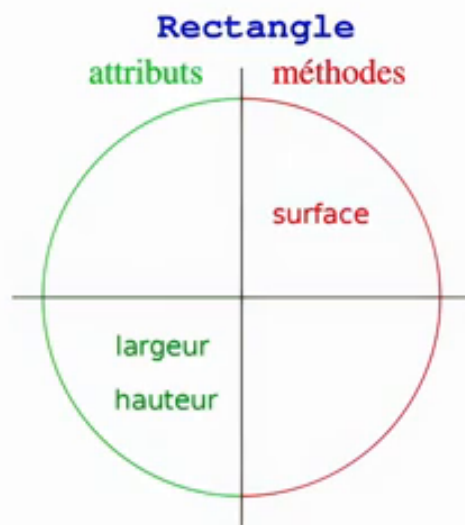
...

notes

résumé

0m 0s





Voyons maintenant comment remplir cette partie, c'est-à-dire comment mettre des attributs et des méthodes à notre classe.

notes

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

résumé

.....

.....

.....

.....

.....

0m 1s



La syntaxe de la déclaration des attributs est la même que celle des champs d'une `structure` :

```
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;  
};
```



Commençons par les attributs, les attributs de la classe « Rectangle » sont donc sa largeur et sa hauteur. Pour déclarer des attributs, il suffit simplement, comme les champs d'une structure, de déclarer les différents attributs en mettant le type et le nom de l'attribut

notes

résumé

0m 13s



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 class `Rectangle`, ceci donnerait simplement : « double hauteur » « double largeur » c'est aussi simple que ça, aussi simple que les champs d'une structure. De même, pour l'utilisation des attributs d'une instance, on va utiliser la même syntaxe que les champs d'une structure. Pour cela, on aura le nom de l'instance, suivi d'un point et du nom de l'attribut que l'on veut désigner. Par exemple, pour désigner la « hauteur » du rectangle « `rect1` », on écrirait : « `rect1.hauteur` ».

notes

résumé

0m 25s



Notre programme (2/4)

```
#include <iostream>
using namespace std;

class Rectangle {
    double hauteur;
    double largeur;
};

int main()
{
    Rectangle rect1;

    rect1.hauteur = 3.0;
    rect1.largeur = 4.0;

    cout << "hauteur : " << rect1.hauteur
         << endl;

    return 0;
}
```

Ce qui va donc donner le programme suivant complet. Ici, j'ai inclus la bibliothèque « iostream » parce que j'utilise les entrées et sorties, plus exactement l'affichage avec « cout » et « endl »

notes

résumé

1m 1s



Notre programme (2/4)

```
#include <iostream>
using namespace std;

class Rectangle {
    double hauteur;
    double largeur;
};

int main()
{
    Rectangle rect1;

    rect1.hauteur = 3.0;
    rect1.largeur = 4.0;

    cout << "hauteur : " << rect1.hauteur
         << endl;

    return 0;
}
```

« using namespace std » et puis ensuite, la déclaration de notre classe « Rectangle » qui contient, on n'oublie pas le point virgule à la fin, les deux attributs « hauteur » et « largeur ». Puis dans le « main », ce qu'on va faire, c'est que l'on va déclarer ici, une instance « rect1 » de la classe « Rectangle » et puis, on va pouvoir utiliser la « hauteur » de « rect1 »

notes

résumé

1m 13s





-- on va l'affecter ici à la valeur 3.0 -- la « largeur » toujours de « rect1 » à laquelle on va affecter la valeur 4.0 ; et puis, on peut par exemple afficher la « hauteur » de « rect1 ». Tout ceci est vraiment très similaire aux structures telles que présentées dans notre précédent cours d'introduction à la programmation. A noter cependant que la code donné ici ne compile pas en l'état. Il vous manque encore un petit ingrédient qui sera détaillé dans la séquence vidéo suivante. Mais, mis à part ça, l'esprit général et l'essentiel de la syntaxe est ici correct. et l'essentiel de la syntaxe est ici correct.

notes

résumé

1m 37s

