

Support de cours

Cours:

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

Vidéo:

### W15-05-polymcoll1-CPP-pt3

Concepts (extraits des sous-titres générés automatiquement) :

**Exemple. Éléments pointés. Bonne conception. Bêtise de ce genre. Jeu. Gros soucis. Programmeur. Collection. Objets. Fonction. Pointeur. Personnages. Moment. Référence. Paramètres.**



[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/>

# Collections hétérogènes

## (Partie 3)

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

Jean-Cédric Chappelier, Jamila Sam et Vincent Lepetit

...

notes

résumé

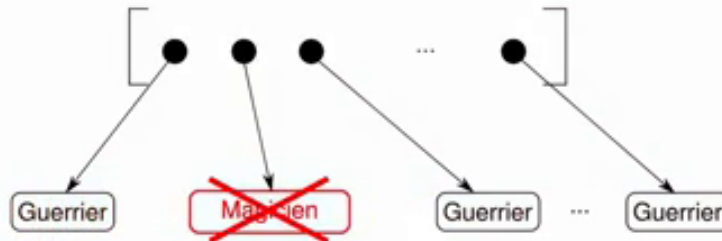
0m 0s



## Pointeurs et intégrité des données

Cette classe `Jeu` comporte cependant *un danger potentiel* :

Pour que tout fonctionne bien, il est nécessaire que les éléments pointés *existent aussi longtemps* que leurs pointeurs.



**Attention !** La co-existence des pointeurs et des éléments pointés n'est cependant pas du tout garantie !

Au programmeur de ne pas faire de bêtises.

Exemple...

Qu'en est-il si jamais l'un des objets que l'on a mis dans la collection, plus exactement dont on a mis une adresse dans la collection, si l'un de ces objets venait à disparaître ? Alors à ce moment là, on aurait toujours dans la collection le pointeur sur l'objet qui aurait par ailleurs été détruit, et cela pourrait causer de gros soucis. Pour que tout fonctionne bien, il est vraiment nécessaire que tous les éléments pointés existent au moins aussi longtemps que leurs pointeurs. Donc dans l'exemple ici, il faut que les personnages qui sont dans le jeu existent

notes

résumé

0m 1s



```
void creer_magicien(Jeu& jeu) {  
    Magicien mago(...);  
    jeu.ajouter_personnage(&mago);  
}  
// ...  
int main() {  
    Jeu mon_jeu;  
    creer_magicien(mon_jeu);  
    mon_jeu.afficher(); // ouille !  
    return 0;  
}
```

au moins aussi longtemps que le jeu existe. Et ça, c'est au programmeur de le garantir par une bonne conception. Donc, au programmeur de ne pas faire de bêtise de ce genre là. Par exemple, imaginons un programmeur un peu naïf qui a voulu créer une fonction pour créer et ajouter un magicien dans un jeu. Donc cette fonction modifiant le jeu prend un « Jeu » par référence et elle crée un « Magicien » ici, en passant des paramètres à son constructeur ; puis elle appelle la méthode « ajouter\_personnage » et comme on doit ajouter un « Personnage » en passant un pointeur, elle passe donc ici l'adresse de ce « Magicien ». Et puis donc dans le « main() », on aurait le jeu. On appelle cette fonction « creer\_magicien ». On passe le jeu par référence pour lui ajouter un magicien. Et puis on affiche le jeu. Et puis on affiche le jeu.

notes

résumé

0m 37s

