

Support de cours

Cours:

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

Vidéo:

INIT-CPP-7_1-IntroPointeurs_references-pt1

Concepts (extraits des sous-titres générés automatiquement) :

Site internet. Technique d'utilisation des pointeurs. Durée de vie. Notion de référence. Numéro de téléphone d'un ami. Troisième navigateur. Séquence vidéo. Exemple du signet d'un site. Exemple de la vie courante. Pointeurs. Raisons d'utilisation des pointeurs. Terme générique. Copie du site. Objet b. Variable de votre programme.



[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/>

Pointeurs et références : introduction

(Partie 1)

Initiation à la programmation (C++)

Vincent Lepetit, Jean-Cédric Chappelier et Jamila Sam

...

notes

résumé

0m 0s





Dans cette séquence vidéo, nous allons nous intéresser

notes

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

résumé

0m 1s



.....

.....

.....

.....

.....

Comment gardez-vous un site intéressant sur Internet ?

- ☞ vous ne le copiez pas sur votre ordinateur mais vous gardez un « bookmark/signet » dans votre navigateur (browser)

C'est un **lien** vers l'information qui vous intéresse



à un nouveau type de données de type pointeurs. Dans cette première séquence, nous allons vous présenter les pointeurs en général et un concept très relié à la notion de référence, puis dans une seconde séquence vidéo, nous vous présenterons plus particulièrement la technique d'utilisation des pointeurs en détails. Les pointeurs ont la mauvaise réputation d'être considérés comme difficiles, mais en fait il n'en est rien, vous utilisez déjà des pointeurs dans la vie de tous les jours, peut-être sans le savoir. Par exemple, quand vous avez un site internet qui vous intéresse, qu'est-ce que vous faites ? Vous n'allez certainement pas dans votre navigateur

notes

résumé

0m 5s



recopier tout le site internet sur votre ordinateur ? Vous ne faites pas la copie du site, ce que vous faites c'est que depuis votre navigateur, vous avez un signet, une adresse qui va pointer sur le site internet, lequel site internet existe toujours de façon indépendante.

notes

résumé

0m 37s

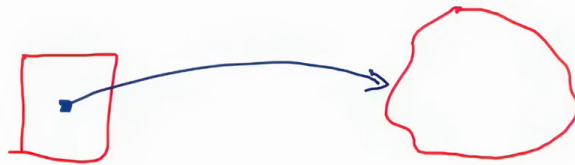


Comment gardez-vous un site intéressant sur Internet ?

- vous ne le copiez pas sur votre ordinateur mais vous gardez un « bookmark/signet » dans votre navigateur (browser)

C'est un **lien** vers l'information qui vous intéresse

Dans un programme, pour faire la même chose et garder un **lien** vers une donnée (c'est-à-dire une variable), en avoir une **référence universelle**, on utilise des **références** ou des **pointeurs**.



Et bien un pointeur c'est pas plus compliqué qu'un signet dans votre navigateur internet. Ainsi, dans un programme, chaque fois qu'on voudra faire un lien, une référence vers un objet qui existe par ailleurs, c'est pas un site internet cette fois-ci mais ça sera une autre variable par exemple dans votre programme, vous utiliserez un pointeur ou une référence chaque fois que vous voudrez comme ceci donc, référencer, indiquer,

notes

résumé

0m 57s





faire un lien vers une autre variable de votre programme.

notes

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

résumé

1m 25s



.....

.....

.....

.....

.....

Les « pointeurs », à quoi ça sert ?

En programmation, les « pointeurs » servent essentiellement à trois choses :

- ① à permettre à plusieurs portions de code de *partager* des objets (données, fonctions,..)
sans les dupliquer

☞ **référence**

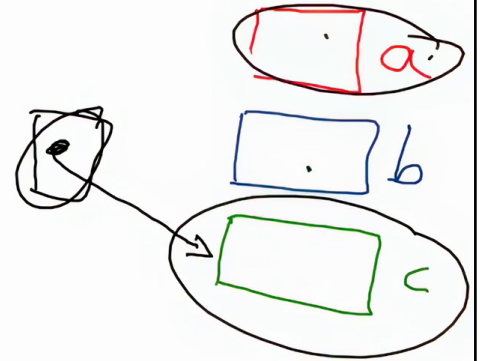
- ② à pouvoir *choisir des éléments* non connus *a priori* (au moment de la programmation)

☞ **généricité**

- ③ à pouvoir manipuler des objets dont la *durée de vie* dépasse la portée

☞ **allocation dynamique**

(moins important depuis **C++11** en raison de la *move semantic*)



Les pointeurs entre guillemets, j'inclue pour l'instant sous ce terme générique les vrais pointeurs et les références, nous ferons la distinction plus tard. Les pointeurs donc, dans un programme, servent fondamentalement à trois choses. Tout d'abord, ils peuvent servir de référence, exactement comme l'exemple du signet d'un site internet, vous avez un objet dessiné en rouge ici, par exemple que j'appelle a, et vous voulez le référencer depuis plusieurs endroits de votre programme. Par exemple, si c'était l'exemple du site internet, a représente un site internet dans le navigateur, dans votre navigateur vous avez un lien vers ce site, dans le navigateur d'un de vos amis vous avez encore un autre lien vers ce site, dans un troisième navigateur de quelqu'un qu'on ne connaît pas vous avez un lien vers ce site aussi. Toutes ces utilisations ici, ces pointeurs référencent donc l'objet que l'on veut référencer. Ça c'est le premier cas d'utilisation des pointeurs. Le deuxième cas d'utilisation des pointeurs, c'est ce que j'appelle l'utilisation pour la générique. Le dessin est un petit peu symétrique du dessin pour les références. Ici, vous avez disons plusieurs objets que vous ne connaissez pas forcément au départ, donc par exemple un objet a, un objet b, un objet c, et vous voulez décrire disons de façon générique ces différents objets par, à un moment donné, vous allez donc à ce moment-là utiliser un pointeur. Ce pointeur à un moment donné pointera par exemple sur l'objet a, puis à un autre moment à ce moment-là pointera vers par exemple l'objet b, et enfin, à peut-être un autre moment dans votre programme le même pointeur pourra pointer, indiquer un objet c. Un exemple de la vie courante, c'est par exemple si vous avez le numéro de téléphone d'un ami, vous le stockez dans votre carnet d'adresses et puis si votre ami donc a

notes

résumé

1m 31s



Les « pointeurs », à quoi ça sert ?

En programmation, les « pointeurs » servent essentiellement à trois choses :

- ① à permettre à plusieurs portions de code de *partager* des objets (données, fonctions,..) *sans les dupliquer*

☞ **référence**

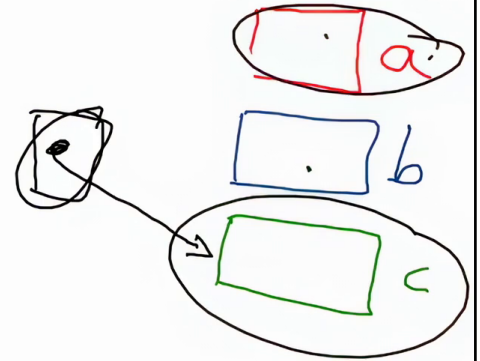
- ② à pouvoir *choisir des éléments* non connus *a priori* (au moment de la programmation)

☞ **généricité**

- ③ à pouvoir manipuler des objets dont la *durée de vie* dépasse la portée

☞ **allocation dynamique**

(moins important depuis `C++11` en raison de la *move semantic*)



à un moment un certain téléphone, puis à un autre moment il change d'opérateur, il a changé de numéro, dans votre carnet d'adresses vous allez changer le numéro de téléphone de votre ami pour le nouveau numéro. Voilà ce que j'appelle donc l'utilisation pour la générique. On a de façon générique un pointeur qui décrit l'un ou l'autre

notes

résumé



ou un troisième objet possible au fur et à mesure que le programme va se dérouler. Le troisième cas d'utilisation des pointeurs est lui un petit peu technique. Il a à voir avec ce qu'on appelle techniquement la portée et la durée de vie.

notes

résumé

3m 49s



Les « pointeurs », à quoi ça sert ?

En programmation, les « pointeurs » servent essentiellement à trois choses :

- ① à permettre à plusieurs portions de code de *partager* des objets (données, fonctions,..)
sans les dupliquer

☞ **référence**

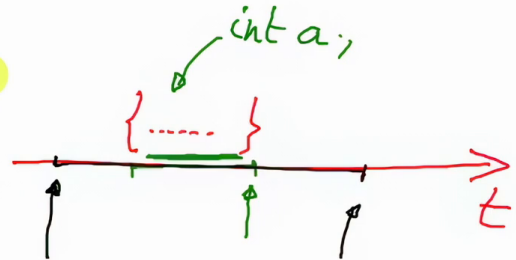
- ② à pouvoir *choisir des éléments* non connus *a priori* (au moment de la programmation)

☞ **généricité**

- ③ à pouvoir manipuler des objets dont la *durée de vie* dépasse la *portée*

☞ **allocation dynamique**

(moins important depuis **C++11** en raison de la *move semantic*)



Donc, la portée, je vous rappelle, c'est l'ensemble des lignes de code où un objet est défini. La durée de vie ça va être le temps pendant lequel un objet existe en mémoire pendant que le programme se déroule. Illustrons à nouveau avec un petit dessin, si je représente comme ça sur un axe horizontal le temps où le programme se déroule, évidemment le programme va exécuter à un moment donné une certaine portion de code qui était comprise entre accolades. Si dans cette portion de code j'avais par exemple une variable de type entier que j'avais déclarée, donc par exemple a, alors la durée de vie de cette variable va être égale au temps d'exécution de sa portée. Sa portée vous vous souvenez, c'est le bloc dans lequel elle a été définie et si je regarde donc le déroulement du programme au cours du temps, la durée de vie de cette variable va donc être ici égale au temps d'exécution de sa portée. Le compilateur va terminer, l'existence va supprimer la variable du moment qu'il en a plus besoin, c'est-à-dire du moment qu'on a terminé sa portée. Et bien si techniquement vous voulez augmenter la durée de vie d'un objet, alors on verra techniquement plus tard comment, mais vous voulez décider qu'un objet commence à être disponible dans votre mémoire à partir de cet instant-là, et puis a une durée de vie donc, qui est plus grande que sa portée,

notes

résumé

4m 6s



Les « pointeurs », à quoi ça sert ?

En programmation, les « pointeurs » servent essentiellement à trois choses :

- ① **référence**
 - ② **généricité**
 - ③ **allocation dynamique**
- 👉 **Important :** Il faut toujours avoir clairement à l'esprit pour lequel de ces trois objectifs on utilise un pointeur dans un programme !

à ce moment-là vous allez être dans ce qu'on appelle l'allocation dynamique, c'est le troisième cas d'utilisation des pointeurs.

notes

résumé

5m 25s



Donc, pour résumer, les pointeurs servent fondamentalement à trois choses. Soit un objet peut être référencé depuis plusieurs endroits avec des pointeurs, soit on va avoir un même pointeur qui lui-même va pointer à différents moments vers différents objets, soit on veut vouloir gérer nous-même la mémoire, le temps de vie des objets en mémoire et utiliser ce que l'on appelle l'allocation dynamique. Ces trois raisons d'utilisation des pointeurs ne sont pas exhaustives et on peut vouloir en combiner plusieurs, mais ce qui compte surtout c'est d'avoir bien présent à l'esprit la raison, ou les raisons, pour lesquelles vous voulez utiliser un pointeur dans votre programme. dans votre programme.

résumé

5m 31s

