Support de cours

Cours:

Initiation à la programmation (en C++)

Vidéo:

Typedef - alias de types (partie 1)

Concepts (extraits des sous-titres générés automatiquement) :

Tableaux dynamiques. Nombre de types. Types de base int. Tableau dynamique de double. Certaine taille. Type vecteur. Variable vecteur. Type matrice. Certaine valeur. Nom du type. Nouveau nom. Tableau dynamique de vecteur. Stade du cours. Écriture de ses types. Schéma général suivant.

vers la recherche de séquences vidéo
(dans Initiation à la programmation (en C++).)

vers la vidéo

EPFL

Typedef : alias de types
(Partie 1)

Initiation à la programmation (C++)

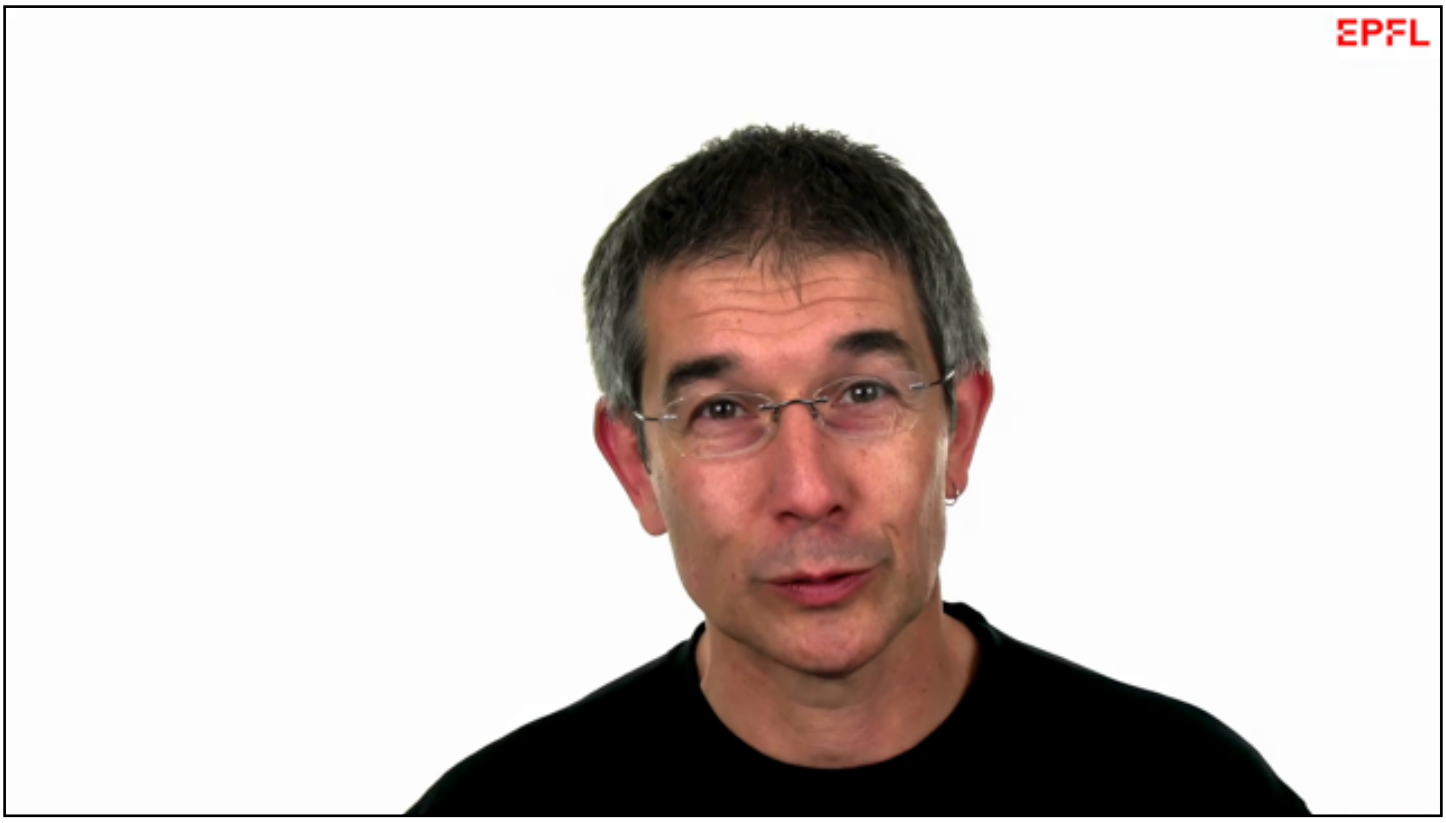
Vincent Lepetit, Jean-Cédric Chappelier et Jamila Sam

...

notes

résumé

0m 0s



A ce stade du cours, nous avons déjà rencontré un certain nombre de types.

notes

résumé

0m 1s

Alias de types

EPFL

Pour des types composés complexes, dont l'*utilisation directe est difficile*, on peut utiliser la commande `typedef` pour

donner un autre nom (alias) à ce type

Syntaxe : `typedef type alias;`

où *alias* est le nouveau nom de type et *type* un type élémentaire ou composé.

Exemples :

```
typedef vector<double> Vecteur;
typedef vector<Vecteur> Matrice;

Matrice rotation(3, Vecteur(3, 1.0));
```

```
Vecteur v;
Vecteur v2 (3, 1.0);
vector< vector < double >> m;
Matrice m;
```

On a vu les types de base int, double et puis, on a vu aussi des types plus avancés comme les tableaux dynamiques que l'on peut même composer, tableaux dynamiques de tableaux dynamiques etc... ce qui rend parfois l'écriture de ses types un petit peu fastidieuse. Pour ses raisons et pour d'autres que je vais développer juste dans un instant, il peut être intéressant de définir des alias de types, c'est-à-dire de donner un autre nom à des types qu'on a déjà défini. Cela se fait avec le schéma général suivant : Vous commencez votre ligne avec le mot clé réservé typedef puis ensuite, vous mettez le nom du type auquel vous voulez donner un autre nom et enfin, le nouveau nom que vous voulez donner à ce type et vous terminez par un point virgule. Par exemple, supposons que vous vouliez appeler Vecteur quelque chose qui est un tableau dynamique de double et bien, ce que vous allez faire c'est que vous allez commencer une ligne avec typedef. Ensuite, le type que vous voulez renommer Vector et puis enfin, le nouveau nom que vous choisissez pour ce type vecteur. A partir de cette ligne, vous pourrez utiliser soit l'ancien nom vector si vous voulez, soit votre nouveau nom. Vecteur avec un V majuscule ici est strictement équivalent, c'est un alias de vecteur de double. Vous pourrez par exemple déclarer une variable Vecteur v pour déclarer, v étant une variable tableau dynamique de double. Vous pouvez bien sûr garder toutes les syntaxes que vous aviez précédemment. Par exemple, initialiser un vecteur v2 avec une certaine taille 3 et puis, une certaine valeur par exemple, Vecteur avec un V majuscule comme vous l'avez spécifié là est strictement équivalent à vector. Vous pouvez même le réutilisez lui-même dans un autre type, par

notes

résumé

0m 5s

Alias de types

EPFL

Pour des types composés complexes, dont l'*utilisation directe est difficile*, on peut utiliser la commande `typedef` pour

donner un autre nom (alias) à ce type

Syntaxe : `typedef type alias;`

où *alias* est le nouveau nom de type et *type* un type élémentaire ou composé.

Exemples :

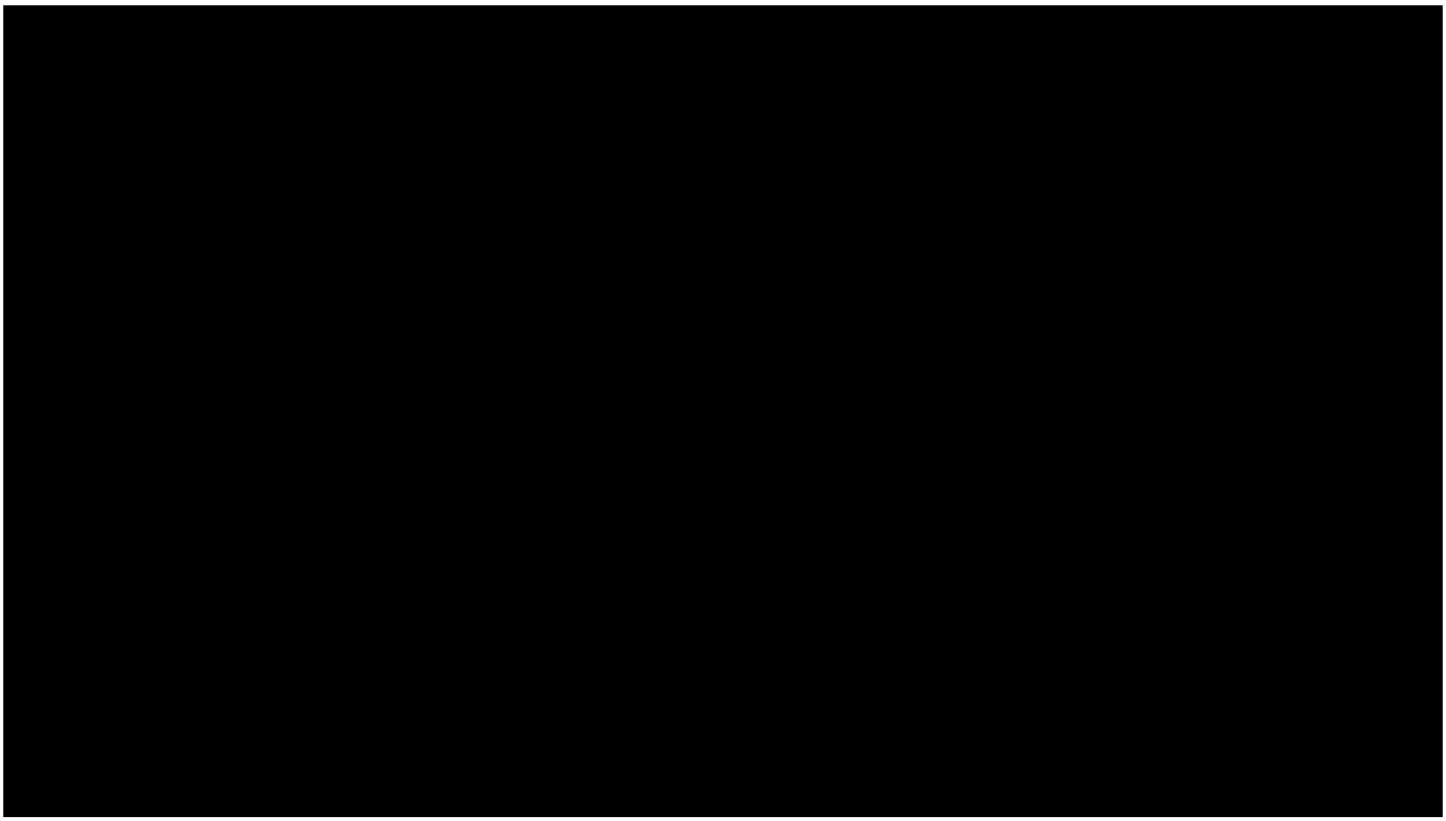
```
typedef vector<double> Vecteur;
typedef vector<Vecteur> Matrice;

Matrice rotation(3, Vecteur(3, 1.0));
```

Vecteur v;
Vecteur v2 (3, 1.0);
vector< vector < double >> m;
Matrice m;

exemple, vous pouvez définir le type Matrice comme étant un vector, tableau dynamique de Vecteur qui est ce vous avez défini juste au préalable, ce qui fait qu'avec cette ligne typedef vector que vous avez juste redéfini précédemment Matrice, le type Matrice est maintenant un tableau dynamique de tableau dynamique de double. Au lieu d'écrire vector> m, par exemple; vous pouvez maintenant tout à fait écrire après cette ligne Matrice m; ce qui est quand même beaucoup plus simple. Vous pouvez donc aussi ici utiliser l'initialisation, Matrice ici c'est le nouveau nom de type que vous avez donné, rotation, le nom de la variable et puis l'initialisation est ici entre paranthèses comme on initialise un tableau dynamique de tableau dynamique en disant que la variable rotation est initialisée à trois.

notes

résumé



Trois quoi ? Trois vecteurs ou tableaux dynamiques de double, mais puisque vous avez défini Vecteur, autant l'utiliser. On peut utiliser trois Vecteur, chacun étant initialisé ici, avec toutes les valeurs à 1 ce qui va avoir comme propriété de vous créer une matrice ici rotation qui sera un tableau dynamique de trois tableaux dynamiques de 3 double chacun a la valeur 1. chacun a la valeur 1.

notes

résumé

3m 1s