

Programmation multifichiers — Cégep du Vieux Montréal — Godofroy Borduas



CÉGEP DU VIEUX MONTRÉAL

Programmation multifichiers *partie 2*

Basé sur les notes d'Alain Thiboulot — Hiver 2022

Programmation multifichiers — Cégep du Vieux Montréal — Godofroy Borduas

Les variables globales

Différence entre variables *locales et globales*

	Globale	Local
Emplacement de la déclaration	Extérieur d'une fonction	Intérieur d'une fonction
Moment de la création	Démarrage du programme (avant <code>main()</code>)	Démarrage de la fonction
Visibilité	Programme	Fonction
Durée de vie	Programme	Fonction

Programmation multifichiers — Cégep du Vieux Montréal — Godefroy Borduas

Pourquoi créer une variable globale ?

- Toujours défini dans un CPP
 - Doit suivre une logique d'organisation
- Contient une information utile à un sous-ensemble du programme
 - Dépasse le simple cadre d'une fonction particulière
- Toujours accessible pour le contenu des définitions du CPP
- Toujours publique, tous les modules y ont accès par défaut
 - Possible de la restringer à un module -> rendre privée
 - Aussi possible de rendre la variable privée à une fonction

Programmation multifichiers — Cégep du Vieux Montréal — Godefroy Borduas

Déclaration et définition

- Déclaration = Annonce au compilateur
 - Comme pour les fonctions
 - Annonce que la variable nommée `foo` de type `T` existe quelque part
- Les déclarations vont dans le fichier d'en-tête (`.h`)
- Déclaration de la variable + Définition du type `T` = Vérification des expressions qui l'utilisent
 - Faites lors de la première étape du processus de compilation
- Comme pour les fonctions, la définition doit exister quelque part
 - Pour utiliser un mot, on doit le connaître

Définition d'une variable

- La définition crée la variable
 - Possible de lui donner une valeur (initialiser)
 - Syntaxe -> Toujours la même qu'en C11
- Exemple :

```
double maVariable = 10;  
Info i = {};  
double notes [5];
```

Restreindre la variable à un module

- Une variable globale est toujours publique
 - Tout le monde peut la déclarer
 - Tout le monde peut l'appeler
- Le mot clé `static` limite la définition au fichier CPP
- Exemple :

```
static double maVariable;  
static Info i = {};  
static double notes [5] = {};
```

Programmation multifichiers — Cégep du Vieux Montréal — Godefroy Borduas

- Exemple :

- Est-ce que `reservoir = "15";` fait du sens ?
 - Sans savoir le type de `reservoir` -> impossible
- Est-ce que `Litre reservoir; /* ... */ reservoir = "15";` fait du sens ?
 - Sans savoir la signification de `Litre` -> impossible
- Est-ce que `using Litre = double; /* ... */ Litre reservoir; /* ... */ reservoir = "15";` fait du sens ?
 - Non, car :
 - a. 15 est un string
 - b. `reservoir` est un `Litre`
 - c. `Litre` est un `double`
 - d. Donc 15 n'est pas un `double`

Programmation multifichiers — Cégep du Vieux Montréal — Godefroy Borduas

Déclaration d'une variable globale

- La syntaxe d'une déclaration d'une variable globale est plus compliquée que pour les fonctions
- La déclaration se fait toujours avec le mot clé `extern`
 - Impossible d'initialiser la variable -> réservée à la définition
 - Le type doit toujours être disponible à la déclaration -> Pour vérifier
 - La déclaration permet de rendre la variable accessible à tous les modules
 - Toujours placé dans le fichier d'en-tête
- Exemple :

```
extern double maVariable;
extern Info i;
extern double notes [5];
```

Variable globale... à une fonction

- Toujours possible de limiter la variable globale à une fonction
 - Dans la déclaration locale -> ajouter le mot clé `static`
 - Existe pour toute la vie du programme
 - Accessible uniquement dans la fonction
- Exemple :

```
void foo() {  
    static int compteur = 0;  
    cout << "foo() -> compteur = " << compteur << "\n";  
    // ...  
    compteur++;  
}
```

- Peu fréquent, mais utile pour les gros calculs

Les constantes globales

- Même chose qu'une variable -> sauf que la valeur ne change pas
- Peut être publique ou privée à un module ou à une fonction
- Les déclarations et les définitions ne changent pas par rapport aux variables globales
- Par défaut, la constante est toujours privées à son modules.

Déclaration d'une constante globale

- Annonce l'existence de la constante
- La déclaration passe toujours par le mot clé `extern`.
- Jamais initialisé -> réservé à la définition
- Exemple :

```
extern const int max;
extern const double TAUX;
```

Définition d'une constante globale

- Toujours privé à son module
 - Contient explicitement le mot clé `static`
- Doit toujours avoir une valeur d'initialisation -> sinon mis à 0 ou équivalent
- Exemple :

```
const int MAX = 100;
const double TAUX = 0.15;
```

- Possible de la rendre publique -> doit ajouter le mot clé `extern`
 - Doit être dans le fichier d'en-tête
- Exemple :

```
extern const int MAX = 100;
```

Programmation multifichiers — Cégep du Vieux Montréal — Godefroy Borduas

Mauvais exemple

- Module A :

```
extern const TAILLE = 5;
```

- Module B :

```
int Tableau[TAILLE];
```

- Impossible -> Car TAILLE n'est pas connue à la compilation

Programmation multifichiers — Cégep du Vieux Montréal — Godefroy Borduas

Bon exemple

- Module A :

```
const TAILLE = 5;  
int Tableau[TAILLE];
```

- Possible -> Car TAILLE est connue à la compilation

Programmation multifichiers — Cégep du Vieux Montréal — Godefroy Borduas

- Possible de limiter à une fonction -> utiliser le mot clé `static`
- Exemple :

```
void foo()
{
    static const int BAR = 42;
    // ...
}
```

Équivalence entre définition sans `static` et avec

- Les deux définitions suivantes sont strictement équivalentes

```
const int MAX = 100;
static const int MAX = 100;
```

Programmation multifichiers — Cégep du Vieux Montréal — Godefroy Borduas

Cas particulier avec les tableaux

- La taille d'un tableau doit toujours être constante
 - Sinon, allocation dynamique
- Doit être connue à la compilation
- Peut être réalisé par une constante mais la constante doit être connue à la création (définition) du tableau
 - Donc -> doit être dans le même module
 - Chaque module est compilé indépendamment

Les différents types d'erreur

- Compilation -> Deux grande étapes **successives**
 - i. Précompilation, Vérification, Génération du code (obj)
 - ii. Vérification et Edition des liens (entre les fichiers obj), Assemblage final du programme.exe
- L'étape 2 ne peut être accomplie sans le succès de la première
- Chaque étape entraîne des erreurs

Compilation et erreur

- Fenêtre de sortie de VS

```
1>----- Début de la génération : Projet : Travaux, Configuration : Debug x64 -----
1>main.cpp
1>sondage.cpp
1>statistic_01.cpp
1>statistic_02.cpp
1>statistic_03.cpp
1>e:\code\statistic_03.cpp(13): error C2065: 'reponses' : identificateur non déclaré
1>statistic_04.cpp
1>statistic_05.cpp
1>statistic_06.cpp
1>statistic_07.cpp
1>statistic_08.cpp
1>e:\code\statistic_08.cpp(9): error C2065: 'nbPersonne' : identificateur non déclaré
1>statistic_09.cpp
1>Génération de code en cours...
1>Génération du projet "Travaux.vcxproj" terminée -- ÉCHEC.
===== Génération : 0 a réussi, 1 a échoué, 0 mis à jour, 0 a été ignoré =====
```

Liste des erreurs

- Ligne 13 du fichier `statistic_03.cpp`
- Ligne 9 du fichier `statistic_08.cpp`

Une fois corrigé

```
1>----- Début de la génération : Projet : Travaux, Configuration : Debug x64 -----
1>statistic_03.cpp
1>statistic_08.cpp
1>Génération de code en cours...
1>Travaux.vcxproj -> C:\Users\Alain\source\repos\Travaux\x64\Debug\Travaux.exe
===== Génération : 1 a réussi, 0 a échoué, 0 mis à jour, 0 a été ignoré =====
```

- La compilation est incrémentale

Quelle est cette erreur ?

- Fenêtre de sortie de VS

```
1>----- Début de la régénération globale : Projet : Travaux, Configuration : ebug x64 -----
1>main.cpp
1>sondage.cpp
1>statistic_01.cpp
1>statistic_02.cpp
1>statistic_03.cpp
1>statistic_04.cpp
1>statistic_05.cpp
1>statistic_06.cpp
1>statistic_07.cpp
1>statistic_08.cpp
1>statistic_09.cpp
1>Génération de code en cours...
1>main.obj : error LNK2019: symbole externe non résolu "int __cdecl stat2(bool * const)" (?stat2@@YAHQEAN@Z) référencé dans la fonction main
1>statistic_05.obj : error LNK2001: symbole externe non résolu "int BIDON" (?BIDON@@3HA)
1>C:\Users\Alain\source\repos\Travaux\x64\Debug\Travaux.exe : fatal
error LNK1120: 2 externes non résolus
1>Génération du projet "Travaux.vcxproj" terminée -- ÉCHEC.
===== Régénération globale : 0 a réussi, 1 a échoué, 0 a été ignoré =====
```

Deux erreurs de liens -> Étape 2 de la compilation

- Toutes les erreurs liens commencent par le code LNKxxxx
 - Pour plus de détail sur le type d'erreur, cliquez sur le code
- 1. Dans main.obj -> Erreur LNK2019 -> Sur int stat2(bool * const) -> Dans la fonction main
 - La définition de la fonction stat2 n'existe pas dans le contexte
 - Soit la fonction n'est pas définie
 - Soit le prototype est différent
- 2. Dans statistic_05.obj -> Erreur LNK2001 -> Sur int BIDON -> Quelque part dans le module
 - Aucune variable BIDON de type int existe
 - Soit que la déclaration et la définition est différente
 - Soit que la variable est privée

Quelle est cette erreur ?

- Fenêtre de sortie de VS

```
1>----- Début de la régénération globale : Projet : Travaux, Configuration : ebug x64 -----
1>main.cpp
1>sondage.cpp
1>statistic_01.cpp
1>statistic_02.cpp
1>statistic_03.cpp
1>statistic_04.cpp
1>statistic_05.cpp
1>statistic_06.cpp
1>statistic_07.cpp
1>statistic_08.cpp
1>statistic_09.cpp
1>Génération de code en cours...
1>statistic_04.obj : error LNK2005: "int __cdecl stat2(bool * const)" (?stat2@@YAHQEAN@Z) déjà défini(e) dans statistic_02.obj
1>C:\Users\Alain\source\repos\Travaux\x64\Debug\Travaux.exe : fatal error
NK1169: un ou plusieurs symboles définis à différentes reprises ont été rencontrés
1>Génération du projet "Travaux.vcxproj" terminée -- ÉCHEC.
===== Régénération globale : 0 a réussi, 1 a échoué, 0 a été ignoré =====
```

Une erreur de lien -> Étape 2 de la compilation

1. Dans `statistic_04.obj` -> Erreur LNK2005 -> Sur `int stat2(bool * const)` ->
Défini dans `statistic_02.obj`
 - Une fonction ne peut jamais être définie deux fois
 - Laquelle faut-il appeler ?
 - Même problème pour les définition multiples des variables et des constantes