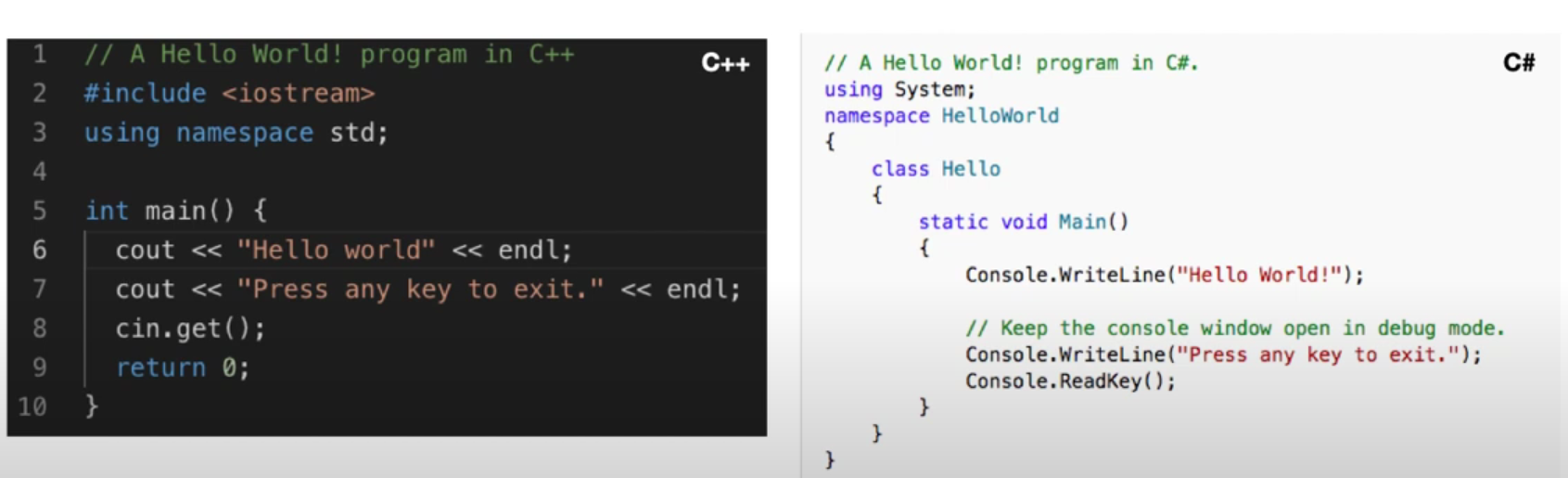
Introduction

Le C++ est complètement différent du c#, voici un exemple de code

Les 2 codes font la même chose :

* Afficher un message « Hello World »,
* Afficher un message « Press any key to exit. »
* Et ensuite on attend que l’utilisateur appuie sur une touche

Le namespace std est le namespace Standard.

# Référence

Une référence est un autre nom pour une variable qu’on a créé. Et pour se faire on va lui affecter la variable créée en question. Pour montrer que l’on affecte une référence de variable, on ajoute un ‘**&**’ après le type de la nouvelle référence/variable.

Type**& nomReference** = **Var** ;

‘**nomReference**’ et ‘**Var**’ pointent vers la même adresse mémoire

L’adresse mémoire de la ‘*VariableCrééJusteAvant’* est affecté à la variable ‘*nomReference*’.

**/!\** On ne peut pas affecter une nouvelle adresse mémoire à un type référence qui a déjà été affecté.

Type& Ref1 = var ;

Ref1 = var3 ; **-> impossible car Ref1 a déjà été affectée**

**/!\** La variable référence doit être déclarer et affecter directement, sinon ça ne compile pas

Type& Ref1 ;

Ref1 = var ; **-> Une erreur de compilation est retournée**

# Main

Main est une fonction (Pas une méthode car elle renvoi un entier)

Il y a 3 manières de l’écrire :

* **Int main()**

Lance le main

* **Int main(int argc, char\* argv[])**

Permet de lancer la classe main avec des arguments lors de la compilation (g++ …. ).

Int argc représente le nombre d’arguments

Char\* argv est un string (tableau de char) qui contient les arguments utilisés

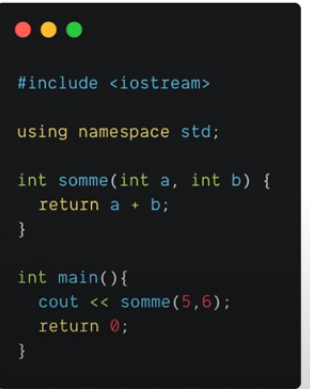
* **Int main(int argc, char\* argv[], char\* envp[])**

Le même qu’au-dessus

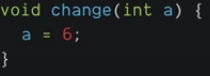
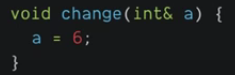
Char\* envp[] est un string qui contient les variables d’environnements utilisées.

# Les fonctions

On peut créer des fonctions sans classe.



## Passage de paramètre par valeur et par référence

(La méthode permet de changer la valeur de la référence (dans le passage par référence))