Programation Orientée objets C#

Contents

1 Introduction

Le C# est un language de programation, il peut être utilsé pour créer des applications type Windows (**Client Lourd**), de la progra de sites en lignes (**Client Léger**) et des aplications qui se rapprochent d'applications Windows mais fonctionnent dans un navigateur (**Client Riche**).

Le code C# est compilé dans un code de language intermédiaire (CIL) qui sera par la suite transformé en un CLR pour Common Language Runtime et enfin en exécutable binaire. Cette particularité du c# le rends plus facile à exporter dans d'autres systèmes d'exploitations (passer de windows à linux ou mac ou encore être capable de faire fonctionner un programme qui n'à pas été spécialement écrit pour un type particulier d'architecture processeur.

On peut ainsi écrire des .EXE qui sont des applications et des .dll qui sont des librairies partagées par plusieurs applications .EXE.

Le C# est un language qui est sensible à la case. Celà signifie qu'une majuscule dans le nom d'une variable ou d'une fonction indique une autre variable ou une autre fonction.

1.1 Installation

Ajout des outils nécéssaires à l'utilisation de C# dans Visual Studio 2017 : Une fois l'instalation de VS2017 effectuée, allez dans menu Outils > Obtenir les outils et fonctionnalités et ajouter :

- Développement de la plateforme windows universelle
- Développement .net Desktop
- Développement Desktop C++

1.2 Premier projet

menu > nouveau projet > visual C# > Bureau classique Windows > Application console

2 Introduction

2.1 Les bases

La manipulation du chapitre précédent va créer par défaut la classe program.cs suivante :

```
using System;

namespace Project{
    class Program{
        static void Main(string[] args){ // Main est la première methode du programme
        }
    }
}
```

2.2 Java

Le c# est plus permisif sur certains aspects de la programation que le java. En effet, il est possible d'écrire plusieurs classes au sein d'un même fichier c#. Il n'y a pas non plus de règles qui définissent les noms de fichiers et les noms de classes en C#. Cependant, il est toutefois conseillé de respecter les bonnes pratiques du java.

3 Coder en c

3.1 Les variables

Les variables sont assez semblables au java, à la distinction que String et string, int et int32,... ne sont pas des équivalents mais bien des synonymes dans le language. On peut définir une variable entière comme suit :

```
int nom_variable; // variable entière
string nom_variable; // variable texte
bool nom_variable; // variable boolenne true/false
decimal nom_variable; // variable qui retiens les décimaux (nécéssaire quand on traite avec

// mettre une valeur dans une variable
nom_variable = 10;
nom_variable++;//ajoute 1
nom_variable--;//retire 1
nom_variable+=2;//équivalent à var= var+2
nom_variable-=2;//idem suppra
nom_variable/=2; //idem suppra
```

>string.Empty est équivalent à "".

On peut concaténer des strings avec l'opérateur +, " est le caractère d'échapement et il permets nottament d'écrire :

```
string tabulation = "\t";
string retourLigne = "\n";
```

```
string slash = "\\";
string slash = @"\"; // le @ évite de devoir écrire le double slash
```

3.2 Conditions

3.2.1 IF

```
if (cond){}
else if(cond){}
else{}
```

3.2.2 Switch

```
switch (variable){
    case 1: // si la variable variable vaut 1 on exécute le code
        /* code */
        break;
    case 2:
        /* code */
        break;
    default: // si tous les case échouent
        /* code */
        break;
}
```

On peut aussi grouper plusieurs case qui auraient le même effet :

```
switch (variable){
    case 1: // si la variable variable vaut 1,2 ou 3 on exécute le code
    case 2:
    case 3:
        /* code */
        break;
    case 4:
        /* code */
        break;
    default: // si tous les case échouent
        /* code */
        break;
}
```

3.3 Les Méthodes

Une mathode s'écrit sous la forme suivante :

```
static void AffichageBienvenue(string nom,string prenom){/* code */}
```

- *static* : ...
- *void* : est la valeur de retour de la fonction (ici elle ne retourne rien)
- *AffichageBienvenue* : nom de la fonction
- *string nom,string prenom* : arguments de la fonction

  >une méthode statique ne peut appeler que des méthodes statiques

3.4 Les Tableaux

```
string[] joueurs = {"Bob","Marc","Tom"};
Console.WriteLine (joueurs[2]); //--> Tom
Array.Sort(joueurs); //classe les éléments du tableau
```

3.5 Les Listes

```
List<int> chiffres = new List<int>();
chiffres.Add(4); // la liste contient : 4
chiffres.Add(5); // la liste contient : 4,5
chiffres.RemoveAt(1); // la liste contient : 4

foreach(int chiffre in chiffres){...} //effectue une ittération sur tous les éléments de la

IndexOf(var);//recherche var dans la liste et retourne son indice
```

3.1 Les Classes

Le nom d'une classe commence par une **majuscule**, ne comprend que des caractères alphanumériques (lettres accentuées ou non) ou le caractère '_' à condition de ne pas commencer par un chiffre. Les conventions de code conseillent d'utiliser "l'upperCamelCase"

3.2 Les Namespaces

Un namespace ou espace de noms est une structure hierarchique groupant des classes ar utilité ou par contexte d'utilisation. Dans le milieu professionel on utilise le namespace pour y noter le nom de l'auteur, le projet et/ou celui qui édite le logiciel. (ex : Henallux.IG.Presences.UI pour désigner la partie UI du projet presences)

3.3 Using

Comparable au mot clé inlude en java, il permets de préciser que la classe utilise une librairie déjà existante. On importe les namespaces contenant ces librairies.

3.4 Les Objets

Un objet est identifié par son nom, son état et son comportement.

L'**état** est l'ensemble des valeurs données aux attributs de la classe dont il est l'instance, à un moment donné.

L'**encapsulation** est le fait de cacher aux yeux de l'extérieur une partie de l'objet déclarée privée, et de ne la rendre accessible, si c'est souhaitée, que par des méthodes dites setters((set) = modification) et getters ((get) = accéder à l'information).

3.5 Les types standards (entiers, booleens et réels)

Un attribut commence par une **minuscule** et fait maximum 511 char (alphaNum, accents, 'ç', Mu et '_'). par défaut, chaque attribut standard est initialisé à 0 du type.

Le Framework .NET comprends beaucoup de types prédéfinis, nottament des alias et un type décimal qui retiens les chiffres des nombres avec une décimale importante (vs approximation binaire des autres types)

On différencie aussi les classes et les structures dans l'écriture et la gestion de la mémoire.

Contrairement en java, en C#, les uint32 et les int ne sont pas deux entités similaires mais des synonymes.

Les strings ou Strings sont aussi synonymes. Ils peuvent être comparés entre eux via l'utilisation de '==' contrairement au java. On peut ainsi les comparer car le C# authorise la redéfinition des méthodes usuelles (ex: ToString) mais aussi des comparaisons '==', '+' ou '*'.

3.6 Attributs et constructeurs