

Apprenez à développer en C#

Quelques notions à travers des images

En général, une instruction en code C# s'écrit sur une ligne et se termine par un point-virgule.

Ainsi, voici une instruction :

```
1 Console.WriteLine("Hello World !!");
```

csharp

Les variables

Une variable est représentée par son nom, caractérisée par son type et contient une valeur. Le C# possède plein de types prédéfinis, comme les entiers (int), les chaînes de caractères (string), etc. Et on utilise l'opérateur = pour affecter une valeur à une variable.

```
1 int age;  
2 age = 30;
```

csharp

Nous pouvons à tout moment demander la valeur contenue dans la variable age, par exemple :

```
1 int age = 30;  
2 Console.WriteLine(age); // affiche 30
```

csharp

Les principaux types de base du framework .NET sont :

Type	Description
byte	Entier de 0 à 255
short	Entier de -32768 à 32767
int	Entier de -2147483648 à 2147483647
long	Entier de -9223372036854775808 à 9223372036854775807
float	Nombre simple précision de -3,402823e38 à 3,402823e38
decimal	Nombre décimal convenant particulièrement aux calculs financiers (en raison de ses nombres significatifs après la virgule)
char	Représente un caractère

Type	Description
string	Une chaîne de caractères

bool Une valeur booléenne (vrai ou faux)

Les opérateurs de comparaison

Une condition se construit grâce à des opérateurs de comparaison. On dénombre plusieurs opérateurs de comparaisons, les plus courants sont :

Opérateur	Description
==	Egalité
!=	Différence
>	Supérieur à
<	Inférieur à
>=	Supérieur ou égal
<=	Inférieur ou égal
&&	ET logique
	OU logique
!	Négation

L'instruction "if"

L'instruction if permet d'exécuter du code si une condition est vraie (if = si en anglais). Par exemple :

```

1 decimal compteEnBanque = 300;
2 if (compteEnBanque >= 0)
3     Console.WriteLine("Votre compte est créditeur");

```

csharp

L'instruction else, qui veut dire « sinon » en anglais.

« Si la valeur est vraie, alors on fait quelque chose, sinon, on fait autre chose », ce qui se traduit en C# par :

csharp

```

1 decimal compteEnBanque = 300;
2 if (compteEnBanque >= 0)
3     Console.WriteLine("Votre compte est créditeur");
4 else
5     Console.WriteLine("Votre compte est débiteur");

```

Créer une méthode

Le but de la méthode est de factoriser du code afin d'éviter d'avoir à répéter sans arrêt le même code :

csharp

```

1 static void AffichageBienvenue()
2 {
3     Console.WriteLine("Bonjour Nicolas");
4     Console.WriteLine("-----" + Environment.NewLine);
5     Console.WriteLine("\tBienvenue dans le monde merveilleux du C#");
6 }

```

Il est possible d'appeler la méthode avec :

csharp

```

1 static void AffichageBienvenue()

```

Les tableaux

En déclarant une variable de type tableau, nous allons en fait utiliser une variable qui contient une suite de variables du même type. Prenons cet exemple :

csharp

```

1 string[] jours = new string[] { "Lundi", "Mardi", "Mercredi", "Jeudi", "Vendredi", "Samedi",
    "Dimanche" };

```

Un tableau, c'est un peu comme une armoire dans laquelle on range des variables. Chaque variable est posée sur une étagère. Pour accéder à la variable qui est posée sur une étagère, on utilise le nom de l'armoire et on indique l'indice de l'étagère où est stockée la variable, en utilisant des crochets [] :

csharp

```

1 Console.WriteLine(jours[3]); // affiche Jeudi
2 Console.WriteLine(jours[0]); // affiche Lundi
3 Console.WriteLine(jours[10]); // provoque une erreur d'exécution car l'indice n'existe pas

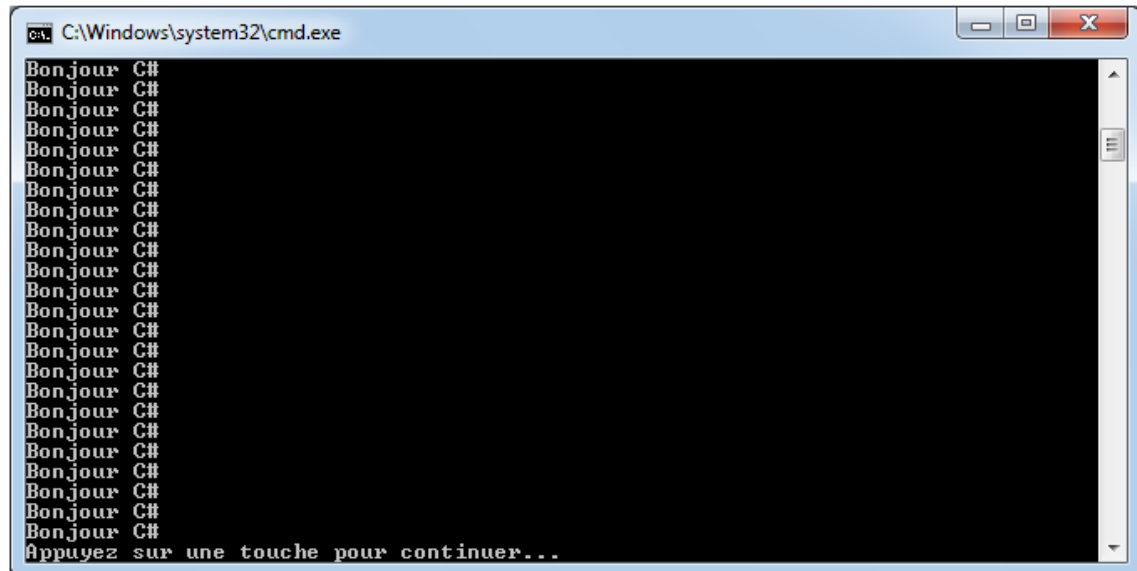
```

La boucle "For"

La première instruction que nous avons aperçue est la boucle for. Elle permet de répéter un bout de code tant qu'une condition est vraie. Souvent cette condition est un compteur. Nous pouvons par exemple afficher un message 50 fois avec le code suivant :

```
1 int compteur;  
2 for (compteur = 0; compteur < 50; compteur++)  
3 {  
4     Console.WriteLine("Bonjour C#");  
5 }
```

Ce qui donne :



```
C:\Windows\system32\cmd.exe  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Bonjour C#  
Appuyez sur une touche pour continuer...
```

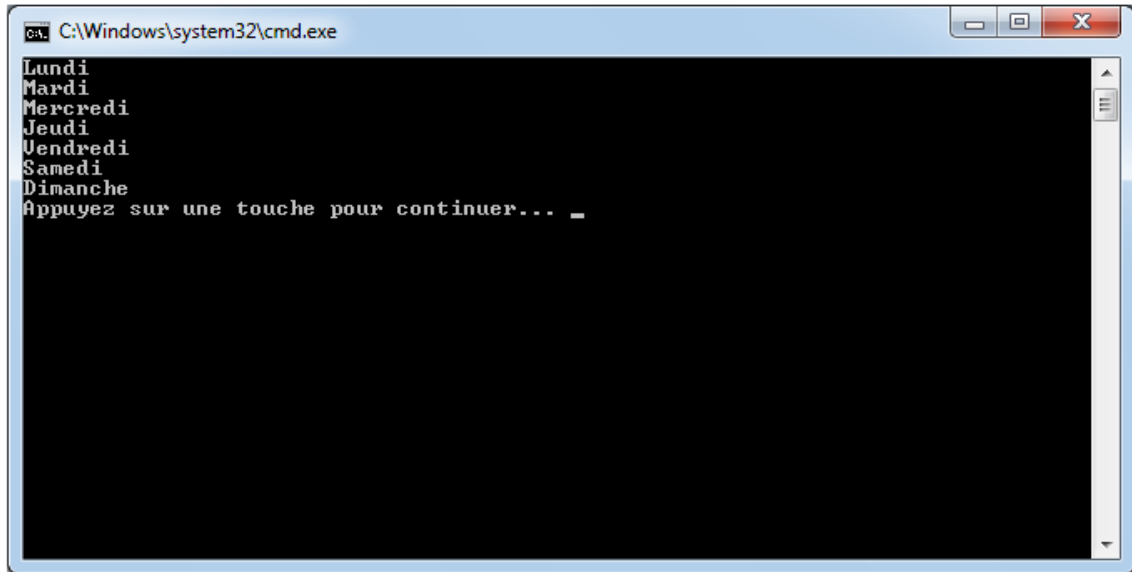
La boucle "Foreach"

Comme il est très courant d'utiliser les boucles pour parcourir un tableau ou une liste, le C# dispose d'un opérateur particulier : foreach que l'on peut traduire par « pour chaque » : pour chaque élément du tableau faire ...

```
1 string[] jours = new string[] { "Lundi", "Mardi", "Mercredi", "Jeudi", "Vendredi", "Samedi",  
2 "Dimanche" };  
3 foreach (string jour in jours)  
4 {  
5     Console.WriteLine(jour);  
6 }
```

Cette boucle va nous permettre de parcourir tous les éléments du tableau « jours ». À chaque itération, la boucle va créer une chaîne de caractères « jour » qui contiendra l'élément courant du tableau.

Ce qui produira :



```
C:\Windows\system32\cmd.exe
Lundi
Mardi
Mercredi
Jeudi
Vendredi
Samedi
Dimanche
Appuyez sur une touche pour continuer... _
```

La boucle "While"

C'est la boucle qui va nous permettre de faire quelque chose tant qu'une condition est vérifiée

```
1 int i = 0;
2 while (i < 50)
3 {
4     Console.WriteLine("Bonjour C#");
5     i++;
6 }
```

csharp