

Ce qu'il faut savoir sur les variables

Concernant les variables

Les variables constituent l'élément de base de tous les langages de programmation.

Elles servent à stocker une information.

Si j'écris :

```
string myString = "toto";
```

Je dis que je souhaite enregistrer la chaîne de caractères (ou la phrase) "toto" dans une variable que j'ai appelé "myString".

Je dois également donner l'information du type de la variable c'est pour ça que j'ai préfixé le nom de ma variable avec le mot clef string. Cependant, il est possible de remplacer `string` par `var` ce qui nous donnera :

```
var myString = "toto" ;
```

Le compilateur est suffisamment intelligent pour déduire le type de ma variable et à la compilation il remplacera lui-même `var` par `string` (ou par tout autre type le cas échéant).

👉 "Mais si le compilateur déduit lui-même le type, pourquoi ai-je besoin de mettre un type ou même var ?"

Pour répondre à cette question, il faut une explication supplémentaire.

Quand j'écris `string myString = "toto";` j'effectue, en réalité, deux étapes en une seule.

J'aurai pu la décomposer de la manière suivante.

1. **Je déclare** ma variable en faisant `String myString;` Cette instruction est parfaitement juste. Elle dit "je vais utiliser une variable qui contiendra une chaîne de caractères que je vais appeler `myString`". Pour l'instant, **ma variable n'a aucune valeur*** mais ce n'est pas grave elle en aura une plus tard.
2. Si, plus tard, je souhaite que ma variable `myString` ait effectivement la valeur "toto". Alors je peux faire simplement `myString = "toto";` Autrement dit, **j'affecte** à ma variable `myString` la valeur que je souhaite. Là, je n'ai plus besoin d'ajouter le type `string` parce que j'ai déjà, au préalable, déclaré ma valeur comme étant de type string.

Donc quand je fais `string myString = "toto";` **Je déclare ma variable (1) et j'affecte une valeur (2) en une seule instruction.**

👉 Et le rapport avec `var` dans tout ça ?

Supposons que maintenant je souhaite **déclarer** une variable `myString` à laquelle j'affecterai une valeur plus tard, comme dans mon cas précédent.

Est-ce que je peux faire simplement `var myString;` ? La réponse est non.

Le compilateur ne peut pas deviner à cette étape là le type de la variable. On pourrait argumenter que `myString` porte le nom string mais le compilateur ne lit pas les noms de variables de cette manière. Si j'avais déclaré une variable que j'aurai appelé `result` **le compilateur n'a simplement aucun moyen de déterminer le type au moment où il lit l'instruction.**

Une fois qu'on a déclaré une variable comme étant un string par exemple, il n'est plus possible de réaffecter à cette même variable une information d'un autre type.

```
var myString = "toto"; //string  
myString = 1; //int
```

👉 **Les lignes de code ci-dessus vont provoquer une erreur !**

Il n'est juste pas possible d'avoir la même variable avec deux types différents.

Le c# est donc un langage que l'on appelle fortement typé à la différence d'autres langages comme le javascript par exemple.


- Notez bien que j'ai précisé, "**n'a aucune valeur**". Je n'ai pas dit que la valeur était une chaîne de caractères vides (c'est quand même une valeur) ou bien 0 ou bien autre chose. La variable n'a aucune valeur dans le sens où il y a véritablement aucune information. Pour représenter cela, on utilise le mot clef NULL (pas nul, ni nulle) mais bien NULL. Donc dans mon exemple myString vaut NULL.

Concernant les conventions de nommage.


Il est d'usage d'écrire le nom des variables en anglais ainsi qu'en **camelCase**. Ce qui signifie :

1. Qu'il s'écrit avec des majuscules à chaque début de mot à l'exception de la première.
2. Qu'on ne met pas de ponctuation ou de chiffres ou d'espaces dans le nom des variables.

Donc :

myString 

myGreatVariable 

my2ndVariable 

MYBIGVARIABLE 