

Variable et constante

Qu'elle est la différence?

- Une variable est un emplacement réservé dans la mémoire vive de l'ordinateur dans lequel on peut inscrire une valeur.
 - On utilise les variables pour stocker de l'information qui va évoluer pendant l'exécution du programme.
- Une constante est exactement la même chose, à un détail près. La valeur doit à être assigné à la constante, dès sa création, et ne peut être modifié par la suite.
 - On utilise les constantes pour donner un nom significatif à une valeur qui sera fixe pendant l'exécution du programme.

Par exemple

Calculateur de taxe

ENTRÉES

réelle montant

CONSTANTE

réelle TAXE = 1.15

SORTIE

réelle montantFinal

DÉBUT

 ECRIRE "Entrez le montant avant taxe : "

 LIRE montant

 montantFinal = montant * TAXE

 ECRIRE "Le montant final est de: " + montant

FIN

Syntaxe en C#

- Nous avons déjà vu la syntaxe pour définir variable en C#: `string name;`
- Pour définir une constante en C#, il suffit d'ajouter le mot-clé: `const` devant le type de donnée. Nous devons ensuite assigné immédiatement la valeur à la constante, car celle-ci ne peut être modifié par la suite.
- Par exemple:
 - `const string NAME = "Alan Turing"`
- Notez que pour les indentifiers facilement, nous utiliserons la syntaxe SNAKE_CASE pour les noms de constantes.

Type de données

- Il y en a beaucoup!! Pour l'instant nous allons nous concentrer sur les types de base les plus utilisés.
 - Nous en verrons d'autre plus tard dans la session (au fur et mesure que nous en aurons besoin.)
- Gardez en tête que les types de données sont les mêmes pour définir une *variable* ou une *constante*

Type de données (syntaxe en pseudo code)

- chaine : Pour stocker du texte.
- caractère: Pour stocker un seul caractère.
- entier: Pour stocker un nombre entier.
- réelle: Pour stocker un nombre à virgule.
- booléen: Pour stocker une valeur booléenne (vrai ou faux)

Syntaxe en c#

- string: `string name = "Alan Turing";` *Notez les guillemets doubles pour définir une chaîne*
- char: `char grade = 'A';` *Notez les guillemets simple pour définir un caractère*
- int: `int age = 18;` *Les entier ne sont pas entouré de guillemet*
- réelle: `double price = 15.99;` *Notez l'utilisation du . pour définir les décimales.*
- bool: `bool isVisible = true;` **Les seules valeurs acceptées sont `true` ou `false`*

Quiz

Quel est le type de donnée de chacune des valeurs suivantes?

- 'X'
- 4.0
- -114
- "B"
- false
- 50000
- ''
- "Claude Shannon"

Réponse

- 'X' -> char
- 4.0 -> double
- -114 -> int
- "B" -> string
- false -> booléen
- 50000 -> int
- ' ' -> char
- "Claude Shannon" -> string

Conversion de type

- La conversion de type, souvent appelé *Casting", sert à assigner une valeur d'un certain type à une variable d'un autre type.
 - Par exemple, lorsque nous utilisons `Console.ReadLine()` la valeur retourné par la fonction de *input* est automatiquement une chaîne. Mais supposons que nous voulons lire un chiffre et le stocker dans une variable de type entier.
 - Le code suivant nous retournerait un erreur: `error CS0029: Impossible de convertir implicitement le type 'string' en 'int'`
- ```
Console.WriteLine("Entrez votre âge:");
int age = Console.ReadLine();
Console.WriteLine("Votre âge est: " + age);
```
- Nous devons convertir la chaîne en entier avant de pouvoir la stocker dans la variable `age`

# Conversion implicite et explicite

- Nous avons déjà vu que le type de variable sert à réserver un espace mémoire.
- Certains types nécessitent plus d'espace que d'autre. `double > int` car on doit réserver de l'espace pour les décimales aussi.
- *Implicit Casting*: Ce fait automatiquement. Convertir une variable vers un type plus grand.

```
int myInt = 9;
double myDouble = myInt; // int vers double se fait automatiquement

Console.WriteLine(myInt); // Output 9
Console.WriteLine(myDouble); // Output 9
```

- *Explicit Casting*: Doit être fait manuellement. Convertir une variable vers un type plus petit.

# Explicit Casting avec la fonction *Convert*

- Il existe plusieurs possibilités de *Casting*, pour le moment nous allons nous limiter à la fonction `Convert`

```
int myInt = 10;
double myDouble = 5.25;
string myIntString = "4";
string myDoubleString = "3.5";

Console.WriteLine("===== Casting types =====");
Console.WriteLine(Convert.ToString(myInt)); // int vers string
Console.WriteLine(Convert.ToInt32(myIntString));
Console.WriteLine(Convert.ToDouble(myDoubleString)); // string ver double
Console.WriteLine(Convert.ToInt32(Convert.ToDouble(myDoubleString))); // string vers int
Console.WriteLine(Convert.ToInt32(myDouble)); // convert double to int
```

## "3.5" ou "3,5" ?

- Lorsqu'on convertit la chaîne d'un nombre à virgule vers un double, nous aurons parfois des problèmes avec la virgule.
  - Sur les systèmes français, nous devons utiliser une virgule `3,5`
  - Sur les systèmes anglais, nous devons utiliser le point `3.5`
- L'internationalisation des programmes est un très vaste sujet qui sera abordé beaucoup plus tard dans le programme. Pour l'instant, il ne vous sera pas demandé de gérer cela, limitez-vous à celui de la langue de votre compilateur.